Analysis of Pattern Dynamics for a Nonlinear Model of the Human Cortex via Bifurcation Theories

A thesis

submitted in fulfilment

of the requirements for the degree

of

Doctor of Philosophy in Physics

at

The University of Waikato

by

Kaier Wang

THE UNIVERSITY OF
WAIKATO
Te Whare Wānanga o Waikato

2014

The University of Waikato

Abstract

This thesis examines the bifurcations, i.e., the emergent behaviours, for the Waikato cortical model under the influence of the gap-junction inhibitory diffusion D_2 (identified as the Turing bifurcation parameter) and the time-to-peak for hyperpolarising GABA response γ_i (i.e., inhibitory rate-constant, identified as the Hopf bifurcation parameter). The cortical model simplifies the entire cortex to a cylindrical macrocolumn (~ 1 mm^3) containing $\sim 10^5$ neurons (85% excitatory, 15% inhibitory) communicating via both chemical and electrical (gap-junction) synapses. The linear stability analysis of the model equations predict the emergence of a Turing instability (in which separated areas of the cortex become activated) when gap-junction diffusivity is increased above a critical level. In addition, a Hopf bifurcation (oscillation) occurs when the inhibitory rate-constant is sufficiently small. Nonlinear interaction between these instabilities leads to spontaneous cortical patterns of neuronal activities evolving in space and time. Such model dynamics of delicately balanced interplay between Turing and Hopf instabilities may be of direct relevance to clinically observed brain dynamics such as epileptic seizure EEG spikes, deep-sleep slow-wave oscillations and cognitive gamma-waves.

The relationship between the modelled brain patterns and model equations can normally be inferred from the eigenvalue dispersion curve, i.e., linear stability analysis. Sometimes we experienced mismatches between the linear stability analysis and the formed cortical patterns, which hampers us in identifying the type of instability corresponding to the emergent patterns. In this thesis, I investigate the pattern-forming mechanism of the Waikato cortical model to better understand the model nonlinearities. I first study the pattern dynamics via analysis of a simple pattern-forming system, the Brusselator model, which has a similar model structure and bifurcation phenomena as the cortical model. I apply both linear and nonlinear perturbation methods to analyse the near-bifurcation behaviour of the Brusselator in order to precisely capture the dominant mode that contributes the most to the final formed-patterns. My nonlinear analysis of the Brusselator model yields Ginzburg-Landau type amplitude equations that describe the dynamics of the most unstable mode, i.e., the dominant mode, in the vicinity of a bifurcation point. The amplitude equations at a Turing point unfold three characteristic spatial structures: honeycomb H_π, stripes, and reentrant honeycomb H_0. A codimension-2 Turing–Hopf point (CTHP) predicts three mixed instabilities: stable Turing–Hopf (TH), chaotic TH,

and bistable TH. The amplitude equations precisely determine the bifurcation conditions for these instabilities and explain the pattern-competition mechanism once the bifurcation parameters cross the thresholds, whilst driving the system into a nonlinear region where the linear stability analysis may not be applicable.

Then, I apply the bifurcation theories to the cortical model for its pattern predictions. Analogous to the Brusselator model, I find cortical Turing pattens in H_π, stripes and H_0 spatial structures. Moreover, I develop the amplitude equations for the cortical model, with which I derive the envelope frequency for the beating-waves of a stable TH mode; and propose ideas regarding emergence of the cortical chaotic mode. Apart from these pattern dynamics that the cortical model shares with the Brusselator system, the cortical model also exhibits "eye-blinking" TH patterns latticed in hexagons with localised oscillations. Although we have not found biological significance of these model pattens, the developed bifurcation theories and investigated pattern-forming mechanism may enrich our modelling strategies and help us to further improve model performance.

In the last chapter of this thesis, I introduce a Turing–Hopf mechanism for the anaesthetic slow-waves, and predict a coherence drop of such slow-waves with the induction of propofol anaesthesia. To test this hypothesis, I developed an EEG coherence analysing algorithm, `EEG_coherence`, to automatically examine the clinical EEG recordings across multiple subjects. The result shows significantly decreased coherence along the fronto-occipital axis, and increased coherence along the left- and right-temporal axis. As the Waikato cortical model is spatially homogenous, i.e., there are no explicit front-to-back or right-to-left directions, it is unable to produce different coherence changes for different regions. It appears that the Waikato cortical model best represents the cortical dynamics in the frontal region. The theory of pattern dynamics suggests that a mode transition from wave–Turing–wave to Turing–wave–Turing introduces pattern coherence changes in both positive and negative directions. Thus, a further modelling improvement may be the introduction of a cortical bistable mode where Turing and wave coexist.

Dedication

To my parents

To my grandparents

For their endless love and supports.

Acknowledgements

I would like to express my deep thanks to my supervisors, Professor Moira Steyn-Ross, Associate Professor Alistair Steyn-Ross, Dr. Marcus Wilson and Professor Jamie Sleigh, for helping me accomplish this thesis.

First and foremost, I own a great debt of gratitude to my chief supervisor, Moira, for her theoretical guidance in my mathematical analysis of the cortical model, which is the main theme of this thesis. My sincere thanks go to my second supervisor, Alistair, for his discussion at every difficult moment and graceful guidance in improving my programming skills. I would like to especially thank Marcus for his valuable comments on my papers and thesis. And also thanks for his tutorials on my teaching skills. In addition, I am grateful to Jamie for providing the EEG data and sharing his clinical expertise. Special thanks are due to my Japanese supervisor Associate Professor Yoichi Shiraishi for his hospitality during my research visiting to his lab in Gunma University, Japan.

I would also like to thank funding supports from the University of Waikato Strategic Investment in Biomedical Research and Fisher and Paykel Healthcare Study Award. Thank you to all my friends, classmates for much encouragements, and fun time I have enjoyed during my PhD studies.

Contents

Abstract i

Dedication iii

Acknowledgements v

List of Figures xii

List of Tables xv

Acronyms and Abbreviations xvi

Chapter 1 **Introduction** 1

Chapter 2 **Pattern dynamics of the Brusselator model** 9
2.1 Introduction to reaction–diffusion systems 9
2.2 The Brusselator model 12
2.2.1 Linear stability analysis vs simulations 14
2.3 Discussion 20

Chapter 3 **Pattern dynamics of the Waikato cortical model** 21
3.1 Modelling strategy 23
3.1.1 Cortico-cortical flux 24
3.1.2 Intra-cortical flux 25
3.1.3 Soma voltage 26
3.1.4 Summary 28
3.2 Cortical stability 30
3.2.1 Cortical stability of SR2007 model 30
3.2.2 Cortical stabilities of SR2009s and SR2009f models 32
3.3 Cortical simulations 33
3.3.1 Cortical simulations of SR2007 model 33
3.3.2 Cortical simulations of SR2009s and SR2009f models 34
3.4 Cortical pattens at a codimension-2 Turing-Hopf point 35

3.5 Discussion 38

Chapter 4 **Derivation of the amplitude equation for reaction–diffusion systems via computer-aided multiple-scale expansion** 41
4.1 Introduction 41
4.2 Derivation of Brusselator amplitude equation for the stripes mode 44
4.3 Concepts underpinning multiple-scale expansion 48
4.3.1 Structure of order equations 48
4.3.2 Structure ansatz 49
4.3.3 Solvability condition 50
4.3.4 Order-2 pattern matching and order-2 solution 52
4.3.5 General comments 54
4.4 `Amp_solving` algorithm 54
4.4.1 Constructing the order equation 55
4.4.2 Solving the order equation 57
4.4.3 Establishing the solvability condition 57
4.5 Summary 58

Chapter 5 **Mode stability analysis by amplitude equations** 59
5.1 Brusselator amplitude equations for the hexagonal mode 59
5.1.1 Steady-state solutions 60
5.1.2 Eigenvalue analysis 62
5.1.3 Simulation results 64
5.2 Brusselator amplitude equations for the TH mode 65
5.2.1 Steady-state solutions and stability analysis 65
5.2.2 Application of amplitude-equation theory to the Brusselator model 68
5.2.3 Analysis of the mixed mode via the complex TH mode amplitude equations 72
5.2.4 Discussion 74
5.3 Mode stability analysis for the Waikato cortical model 77
5.3.1 Hexagonal mode of the cortical model 78
5.3.2 Turing–Hopf mode of the cortical model 89
5.3.3 Discussion 95

Chapter 6 **A gap-junction modulated Turing–Hopf mechanism for anaesthetic slow-waves** 99
6.1 Anaesthetic unconsciousness 99
6.2 The effect of gap-junctions on cortical stability 100
6.3 Modelling anaesthesia 101
6.4 Cortical stability 102

6.5 The measure of EEG phase-coherence . . . 103
6.5.1 Hilbert transform . . . 104
6.6 Simulation results . . . 105
6.6.1 Effect of inhibitory diffusion on "Awake" cortex . . . 105
6.6.2 Effect of inhibitory diffusion on "Comatose" cortex . . . 108
6.7 Global coherence for the cortex at "Awake" and "Coma" . . . 109
6.8 Mode stability analysis for the anaesthetic cortical model . . . 113
6.9 Phase-coherence analysis of the clinical EEG with induced propofol anaesthesia . . . 114
6.9.1 Materials . . . 114
6.9.2 `EEG_coherence`: An automatic EEG processing algorithm for EEG coherence analysis . . . 114
6.9.3 Results . . . 117
6.10 Discussion . . . 120

Chapter 7 **Summary and future work** 125
7.1 Summary . . . 125
7.2 Major achievements . . . 129
7.3 Future work . . . 130

Appendix A **Source codes** 133
A.1 Brusselator simulation . . . 133

Appendix B **Derivation of the cortical model** 139
B.1 Total synaptic inputs . . . 139
B.2 Soma modelling - chemical synaptic inputs . . . 140

Appendix C **Comparision between slow- and fast-soma cortical models** 143

Appendix D `Simulink` **reconstruction of the Waikato cortical model** 145
D.1 Background . . . 145
D.2 Modelling differential equations in SIMULINK . . . 146
D.2.1 Van der Pol oscillator . . . 147
D.3 Brusselator model . . . 150
D.4 SIMULINK versions of Waikato cortical model equations . . . 154
D.5 Cortico-cortical flux . . . 156
D.6 Intra-cortical flux . . . 157
D.7 Soma voltage . . . 158
D.8 Comment on SIMULINK running efficiency . . . 159

Appendix E **Coding strategies used in `Amp_solving`** 161
E.1 Assigning values and defining equations . . . 161
E.2 Vectorising variables . . . 162
E.3 Manipulating expressions . . . 163
E.4 Coding strategies in multiple-scale expansion . . . 164
E.4.1 Example 1 . . . 164
E.4.2 Example 2 . . . 164
E.4.3 Example 3 . . . 165
E.4.4 Example 4 . . . 165
E.4.5 Example 5 . . . 166
E.4.6 Example 6 . . . 167

Appendix F **Derivation of Brusselator amplitude equations for the hexagonal mode using `Amp_solving`** 171
F.1 Nonlinear expansion . . . 171
F.1.1 Initialise model configuration . . . 171
F.1.2 Define model equations . . . 172
F.1.3 Expansions . . . 173
F.2 Linear stability analysis . . . 175
F.3 Order-2 solvability condition . . . 177
F.3.1 Structure ansatz . . . 177
F.3.2 Order-2 solvability condition . . . 178
F.4 Solution of order-2 equation . . . 179
F.4.1 Expand the RHS of the order-2 equation . . . 179
F.4.2 Determine the structure of the order-2 solution . . . 180
F.4.3 Substitute order-2 solution structure into the LHS of the order-2 equation . . . 181
F.4.4 Construct coefficient equations by matching exponential patterns . 181
F.4.5 Solve coefficient equations . . . 181
F.5 Order-3 solvability condition . . . 182
F.5.1 Substitute the structure ansatz and order-2 solution into the order-3 equation . . . 182
F.5.2 Apply the orthogonal condition . . . 182
F.5.3 Tidy up amplitude equation . . . 183
F.5.4 Return arguments to normal scales . . . 183

Appendix G **Derivation of Brusselator amplitude equations for the Turing–Hopf mode using `Amp_solving`** 185

Appendix H **Symbol notations in `Amp_solving`** 193

References 195

List of Figures

2.1 Hopf simulations for the Brusselator model 15
2.2 Brusselator stability curves for various B values 17
2.3 Turing simulations of the Brusselator model 18
2.4 TH simulations of the Brusselator model 19
2.5 2-hour Turing simulations of the Brusselator model 20

3.1 Schematic representation of the connective topology within a cortical macrocolumn 22
3.2 Illustration of the connection between two neurons 23
3.3 Schematic of the Waikato cortical model 24
3.4 IPSP response (inhibitory alpha-function) curve 26
3.5 Eigenvalue distribution curves for SR2007 model 31
3.6 Eigenvalue distribution curves for SR2009s and SR2009f models 33
3.7 Cortical patterns of SR2007 model 34
3.8 Patterns of SR2009s model 36
3.9 Patterns of SR2009f model 37
3.10 Beating-wave pattern of SR2007 model 38

4.1 Wave vectors of a hexagonal pattern 49
4.2 Demonstrated patterns of the Brusselator model 50
4.3 `Amp_solving` flowchart for deriving the order-3 amplitude equation. 56
4.4 A general flowchart for deriving the order-k amplitude equation from the order-k equation 56

5.1 Three basic modes of the Turing patterns. 61
5.2 Demonstrated phase evolution $\Phi(t)$ with respect to the sign of v 61
5.3 Turing mode stability of the Brusselator model 65
5.4 Bifurcation diagram with $C_{\mathrm{T}}/C_{\mathrm{H}}$ being the control parameter 67
5.5 Mode stability of the TH mode amplitude equations for the Brusselator model when $\Delta > 0$ 69
5.6 Simulations for the Brusselator model when $\Delta > 0$ 70

5.7 Mode stability of the TH mode amplitude equations for the Brusselator model when $\Delta < 0$ . . . 70
5.8 Simulations for the Brusselator model when $\Delta < 0$ . . . 71
5.9 Chaotic mode of the Brusselator generated from two sets of parameters . . 73
5.10 Demonstrated H_π and H_0 modes . . . 75
5.11 Cortical simulations of the SR2007 model starting from nine selected steady-states . . . 79
5.12 Cortical simulations with respect to various γ_i at fixed D_2 . . . 80
5.13 Cortical simulations with respect to various D_2 at fixed γ_i . . . 80
5.14 Mode stabilities of the Turing pattern for the SR2007 model when subcortical drive $s = 0.5$ . . . 81
5.15 H_0 patterns generated by the SR2009s model with PSP described by the alpha-function . . . 82
5.16 Cortical H_0 patterns with various D_2 and fixed γ_i . . . 83
5.17 Cortical H_0 patterns with various γ_i and fixed D_2 . . . 84
5.18 Gallery of LSA predicting curves for the SR2009s model . . . 85
5.19 Four frames of the animated cortical gallery . . . 86
5.20 Four frames of the animated cortical gallery (continued) . . . 87
5.21 Mode stability of the TH mode amplitude equations for the SR2007 model 90
5.22 Demonstrated cortical chaotic mode . . . 90
5.23 400-s simulations of the SR2007 model starting from eight different steady-states . . . 91
5.24 Parameter settings of s, $D_2^{\mathrm{SS_o}}$ and $\gamma_i^{\mathrm{SS_o}}$ for simulations at different steady-states . . . 92
5.25 Mode stabilities for the steady-states in the different branches . . . 92
5.26 Pattern simulations of the SR2007 model with increased D_2 . . . 93
5.27 Pattern simulations of the SR2007 model with increased D_2 and γ_i . . . 94
5.28 Cortical twinkling-eye pattern . . . 96

6.1 Manifold of equilibrium states Q_e^o for homogeneous model cortex . . . 102
6.2 The steady-state firing rates Q_e^o as a function of varying anesthetic inhibition λ_i . . . 103
6.3 Awake cortex simulations . . . 107
6.4 Anaesthetised cortex simulations . . . 110
6.5 Global phase-coherence trends with respect to inhibitory strength for the cortex at awake and comatose states . . . 111
6.6 Q_e time-series for anaesthetic slow-wave and delirium . . . 112
6.7 TH mode stability with respect to the inhibitory diffusion D_2 for awake and comatose cortex . . . 113

6.8 Sample wake and sedated unconsciousness EEG recording 115
6.9 Filtered EEG in sub-delta band 115
6.10 Flowchart for processing EEG of two brain states to determine electrode-pairs with significantly altered phase-coherence 118
6.11 Electrode-pairs with significantly reduced or increased phase-coherence of the sub-delta band EEG induced by propofol anaesthesia 119
6.12 Electrode-pairs comparison 120
6.13 Bistable modes of the Brusselator model 124

A.1 Comparision of the Brusselator simulation using `ode45` and Euler methods 137

D.1 Solution of the van der Pol equation, produced via MATLAB code sheet. . . 148
D.2 SIMULINK built-in example for the van der Pol model called by the MATLAB command `vdp`. 148
D.3 Comparision for the van der Pol equation solved from SIMULINK and MATLAB code sheet 150
D.4 SIMULINK modelling of the convolution with toroidal boundaries 152
D.5 SIMULINK construction of the Brusselator model. 154
D.6 Flux flows for the Waikato cortical model between its subsystems 156
D.7 SIMULINK-based cortico-cortical wave-equation 156
D.8 SIMULINK-based subcortical flux 157
D.9 SIMULINK-based $e \to e$ post-synaptic flux Φ_{ee} for the intra-cortex 157
D.10 SIMULINK-based $i \to e$ post-synaptic flux Φ_{ie} for the intra-cortex 158
D.11 SIMULINK-based excitatory soma equation 159
D.12 Overview of the SIMULINK implementation of the Waikato cortical equations 160
D.13 Detailed connection diagram for the SIMULINK-based Waikato cortical equations 160

List of Tables

2.1 Threshold of bifurcations (in theory) 12
2.2 Threshold of bifurcations (Brusselator model) 14
2.3 LSA predictions vs simulations of the Brusselator model at Hopf mode . . 16

3.1 Symbol definition for the cortico-cortical equations 24
3.2 Symbol definition for the intra-cortical equations 26
3.3 Symbol definition for the soma equations 28
3.4 Referenced Waikato cortical models 30
3.5 Turing instability predictions for selected steady-states 34

C.1 Slow- and fast-soma limits of the Waikato cortical model 143

H.1 Mathematical symbols and their expressions used in `Amp_solving` 193

Acronyms and Abbreviations

ACh	Acetylcholine
AMPA	α-amino-3-hydroxy-5-methyl-4 isoxazole propionic acid (a neurotransmitter associated with the fast-decaying "early" current in the excitatory postsynapse)
CMR	centre manifold reduction
CTHP	codimension-2 Turing–Hopf point
EEG	electroencephalogram (brain activity recorded via scalp electrodes)
EPSP	excitatory postsynaptic potential
FFT	fast Fourier transform
GABA	γ-aminobutyric acid (an inhibitory neurotransmitter)
IPSP	inhibitory post-synaptic potential
LSA	linear stability analysis
MSE	multiple-scale expansion
NREM	non-rapid eye movement
PSP	postsynaptic potential
REM	rapid eye-movement
SR2007	Steyn-Ross cortical model published in *Physics Review E* (2007)
SR2009s	Steyn-Ross cortical model in slow-soma limit published in *Neuroimage* (2009)
SR2009f	Steyn-Ross cortical model in fast-soma limit published in *Neuroimage* (2009)
SS	steady-states
SWS	slow-wave sleep
TH mode	interacted mode between Turing and Hopf instabilities

Chapter 1

Introduction

The human brain is known to exhibit features of pattern-forming dynamical systems, including self-organisation and multistability. These dynamical behaviours are observed traditionally by placing electrodes on the subject's scalp to record the electroencephalogram (EEG): electrical activity reflecting the summation of excitatory postsynaptic potentials in apical dendrites of pyramidal neurons in the more superficial layers of the cortex [109].

The spatiotemporally oscillating EEG forms wave patterns which reveal brain states, distinguished through frequency and amplitude. For example, the EEG is a powerful tool for tracking brain changes during different phases of sleep: When awake, the brain exhibits β (>13–30 Hz) and γ (13–100+ Hz) waves in which γ waves are the highest in frequency and lowest in amplitude. During the period of relaxation, the brain waves become slower, and increase in amplitude; these are called α waves (8–13 Hz). The first stage of sleep (or anaesthetised unconsciousness) is characterised by θ waves (4–8 Hz), which are even slower in frequency and greater in amplitude than α waves. Passing this stage, the sleeper enters into the deep-sleep stage where δ waves (< 4 Hz) are the slowest and highest amplitude brain waves.

Apart from categorising brain functioning via EEG, the detection of EEG patterns brings evidence of significant differences between normal and pathological states of the brain [136]. For example, EEG recording during epileptic seizure shows abrupt bursts in a single EEG channel and global synchronisation over the whole cortex region [68]. Such pathological synchronisation may be the main mechanism responsible for an epileptic seizure [80, 89]. Further previous studies also indicate other large-scale coherent phenomena in neural pathologies, including hand tremor in Parkinson disease and hallucinations [46].

The above findings regarding transitions between two brain states, either from consciousness to unconsciousness, or from normal to pathological, are in principle bifurcations. A bifurcation is a sudden change in a dynamical system's activity, which can be either static (Turing) or periodic (Hopf) in nature; the type of bifurcation is determined by the dominant eigenvalue of the Jacobian matrix of the system, namely zero or purely imaginary, respectively. Over the last few decades, there has been increasing interest in attempting to explain the transitions in the brain using bifurcation theories. This has

required development of nonlinear models of the brain to simulate brain dynamics. In the community of brain modelling, there are two major classes: spatially discrete neural network models and spatially continuous population models (see [20] for a review).

The ambitious Blue Brain Project is a neuron-by-neuron model representing the large-scale neural simulation of about one million neurons in cortical columns. The model includes considerable biological detail, neural spatial structures, connectivity statistics and other neural properties [72]. More recent work has extended the spatial scale to one billion neurons in the Cognitive Computation Project [3] and even up to 100 billion neurons reported in [50]. These types of models enable us to relate the incredibly complex behaviour of animals to the equally complex neural activity of their brains [27]; but they cannot provide clear delineation as to the scale of details that should be included, and the lack of adequate empirical detail leads to the failure of these types of models to produce EEG [65]. The second class of models, in which the cortical tissue is described as a spatial continuum, only considering the *mean* activities of densely interconnected neuronal aggregates in a *field*, is arguably more suited to the description of EEG as it relies on spatial averaging by implicitly defining a spatial scale while the EEG is also a spatial average depending on the geometry of recording [87]. Such spatially continuous theories are often referred as cortical mean-field theories [13, 42].

Brain modelling may give rise to the development of artificial intelligence mechanisms (e.g., Blue Brain Project); and is used to investigate the dynamics of neural activity in cortex. The PhD work presented here is focused on the second aim, based on a mean-field cortical model developed by Cortical Modelling Group of The University of Waikato (see [112, 117, 120, 121] for representative publications corresponding to the Waikato cortical model).

The principal configuration of the Waikato cortical model follows the continuum theories initially proposed by Wilson and Cowan [143] and shares features of Nunez's [84–86], Wright's [146–148], Robinson and Rennie's [100–102] and Liley's models [65]:

- Wilson and Cowan describe the interaction of excitatory and inhibitory neurons to model the cortical tissue as a two-dimensional sheet with these two kinds of neurons homogeneously and isotropically distributed. The model allows four types of interconnections: excitatory–excitatory, excitatory–inhibitory, inhibitory–excitatory and inhibitory–inhibitory. The long-range action potential delivery is assumed to be preserved without any decay in a constant velocity along the axon. A sigmoid function is used to represent the relationship between the membrane potential and the population-averaged neuronal firing-rate.
- The Nunez model distinguishes action potential propagation along the long-range cortico-cortical and short-range intracortical fibres by introducing a significant delay

in the distant neuronal connections. The model assumes that the long-range cortico-cortical fibres are exclusively excitatory and the short-range introcortical fibres are both excitatory and inhibitory. The idea of spatial divergence in neuronal fibres allows a more realistic modelling of the large mammalian brain, such as human brain.

- The Wright model visualises the cortex as a 2-dimensional cortical sheet consisting of lumped excitatory and inhibitory neural populations characterised of their mean membrane potentials, mean firing-rates, and mean interconnection densities. The Wright model embeds the mostly physiologically-based parameters, enabling comparable predictions to real measurements.
- The Robinson and Rennie model is similar to the Wright model while replacing the axonal propagation from the original Green function to a damped wave equation. This modification greatly improves the spatial resolution and permits analytic study of wave properties and stability.
- The Liley model assumes a homogeneous cortex consisting of excitatory and inhibitory neuronal populations, which communicate via long-range (cortico-cortical) and short-range (intra-cortical) connections. Here, the long-range connections are also exclusively excitatory. A sigmoid function is used to link the mean soma membrane potential and the mean firing-rate of the action potential. Analogous to the Wright model, the Liley model uses a Green function for the axonal propagation approximation. This model has successfully simulated the alpha rhythmic activity in the human brain.

The Waikato cortical model utilises coupled differential equations to model the dynamics of two neuronal populations: excitatory and inhibitory. This type of model is referred to as *activity-based model* [28], in which differential equations describe the spatiotemporal evolution of mean firing-rate for each of the neural sub-populations. To study the model output (of simulated EEG patterns) as a response of model internal properties, a stability analysis is performed. Dynamical systems depend on parameters, which, when varied in certain regions of the parameter space, can result in bifurcations from equilibrium [71]. In a cortical model, the bifurcation-controlling parameters carry physiological significance, thus providing a better understanding of how the bifurcation mechanism contributes to clinically-observed EEG.

Bifurcation analysis of a dynamical system usually follows a standard procedure. For simple systems such as the Hodgkin-Huxley model [64], the steady-states are first determined by setting the time derivatives to zero. Then the stability of these steady-states is examined by linear eigenvalue analysis. By examining the phase plot and its isoclines, one

is able to investigate the dynamics of a simple system. This method of bifurcation concentrates on the dynamics in the linear region, thus being named linear stability analysis (LSA).

Chapters 6, 8 and 9 of the textbook *Spikes, Decisions & Actions: Dynamical Foundations of Neuroscience*[1] by Wilson [144] offer details for bifurcation analysis on simple neural models. Even for more complicated models, LSA is capable of capturing the linear growth of the mode [100, 118, 120]. However, LSA cannot predict further details of the mode when the system has fully evolved into its nonlinear regime.

To be more specific, the nonlinear pattern dynamics raises numerical questions: Is the emerged pattern (Turing or Hopf) stable? Is there a possibility that a second bifurcation might occur? What kind of Turing pattern will be featured: striped, square or hexagonal? What will happen if Turing and Hopf coincide? The amplitude equation is a nonlinear analysis technique that can address some of these issues. Comprehensive reviews of amplitude equations can be found in [4, 22].

It is well known that the amplitude equation has the form of the Ginzburg-Landau equation (GLE) for superconductivity in the absence of a magnetic field. The GLE associated with *Turing* bifurcation is referred as the real GLE (RGLE), since the coefficients in the equation are real. Correspondingly, the amplitude equation associated with *Hopf* bifurcation gives the complex GLE (CGLE) in which the coefficients are complex numbers [4]. The basic idea of amplitude equations are that they describe the time evolution of the amplitudes corresponding to the critical wavenumbers that characterise the critical modes at the onset of bifurcations.

Nonlinear bifurcation analysis has previously been applied to the cases of the Brusselator model [75], Belousov-Zhabotinsky model [48], the Gray-Scott model [24] and the Lengyel-Epstein model [104]. These investigate pattern-forming systems near a codimension-two[2] Turing-Hopf point (CTHP) where the mode is characterised by a critical wave vector (Turing feature) and a critical oscillating frequency (Hopf feature). The CTHP is triggered by the coordinated tuning of both Turing and Hopf control parameters in the system. Coupled amplitude equations are derived at a CTHP for the analysis of the pattern dynamics [56]. The amplitude equation provides a reduced, universal description of weakly nonlinear spatiotemporal phenomena [4]; that is, any pattern-forming system can be transformed into an amplitude equation regardless of the details of the original system.

The most widely applied techniques to derive the amplitude equations are *centre manifold reduction* (CMR) and *multiple scale expansion* (MSE). The main idea of both techniques is to give insight into system dynamics by mapping the original system to a normal form showing reduced dynamics captured by the amplitude equations. The centre manifold is the surface separating the unstable and stable manifolds in the space spanned by

[1] ebook available at http://cvr.yorku.ca/webpages/wilson.htm

[2] The "two" refers to the number of control parameters responsible for the bifurcation

the eigenvectors of the bifurcating eigenvalues. These subspaces are linear approximations of the full nonlinear dynamics described by their corresponding manifolds. The centre manifold theorem proposed in [14] states that the full dynamics of a nonlinear model can be reduced to the centre manifold near a bifurcation point. The dynamics on the centre manifold is described by universal normal forms of the amplitude equations. As an alternative to CMR, MSE introduces scaled space and time coordinates that capture the slow modulation of the dominant mode. However, these new scaled variables may cause the perturbation method to fail because of a resonant driving of a higher order term by a lower-order term [70]. To prevent this, a solvability condition is applied. The lowest-order nontrivial solvability condition often produces an evolution equation that is called the amplitude equation [15].

We chose MSE for the analysis of pattern dynamics because it is conceptually more straightforward than CMR. Although the calculation of MSE is extremely cumbersome, it is an iteration-based algorithm, thus has the advantage when programming an automated process. A few attempts for MSE programming have been made in the last decades: Pismen *et al.* [97] outlined a computer software written in MATHEMATICA for computer-assisted derivation of the amplitude equation valid in the vicinity of a bifurcation point. Later, Pismen *et al.* showed a more sophisticated illustration of their MATHEMATICA software [96]. However, the proposed software is not available to the public. Yu has published a series of works on the analysis of a double-Hopf bifurcation using MSE in the MAPLE programming environment [156–158]. Although the source code of Yu's work is available [158], the lack of explanation and its specialised application (double-Hopf bifurcations only) restrict its extension to other bifurcation cases. Khanin *et al.* developed another MATHEMATICA package with the implementation of MSE on generalising amplitude equations for a wide range of bifurcations [55]. To our knowledge the software by Khanin *et al.* may be the most promising work regarding bifurcation analysis. Unfortunately, the MATHEMATICA codes are not given in their paper.

The difficulty for locating a suitable software for the amplitude equation derivation was the initial motivation for developing our own software. Furthermore, the published analysis and text books often omit many mathematical details, which are necessary for fully understanding the derivation and application of amplitude equations. There are limited resources describing the full implementation of MSE. Most publications reference Kidachi's 1980 paper [56], which offers a relatively detailed application of MSE to the Brusselator model at a CTHP. This work is arguably the most valuable contribution to the community of pattern dynamicists since it gives explicit expressions for all coefficients of the amplitude equations.

The amplitude equation, representing the reduced dynamics at a bifurcation point, is deduced from the original system. Parameters of the original system will feed into the coefficients of the derived amplitude equation. Therefore it is essential to understand how

the coefficients of amplitude equations are formalised. In this PhD work special attention will be paid to these details.

Thesis structure

In **Chapter 2**, I outline the standard LSA by examining the pattern dynamics of the Brusselator model. The reason I choose to use the Brusselator model as my calibration test is that it represents the simplest reaction–diffusion system capable of generating complex patterns. In the Brusselator model, the competition between two reactors and the introduction of diffusion satisfy the key requirements for pattern formation [127]. The structure of the Brusselator model is a miniature of more sophisticated pattern-forming systems, such as the Waikato cortical model. The techniques reviewed there can be applied to any pattern-forming system directly. The Brusselator simulation codes provided in **Appendix A** can also be applied to simulate other pattern-forming systems.

In **Chapter 3**, a brief introduction to the Waikato cortical model is presented. The cortical modelling logic will be explained in detail; the necessary mathematical deductions are supplied in **Appendices B and C**. I then review the cortical dynamics previously published in [118, 120]. In **Appendix D**, I present the simulation program of Waikato cortical model coding in SIMULINK supplementary to its original MATLAB code-sheet environment.

At the end of Chapters 2 and 3, I present patterns from simulations that cannot be explained by LSA. To have a better understanding of the pattern formation, I developed an MSE based semi-automated algorithm, `Amp_solving`, for the derivation of the amplitude equation at an arbitrary bifurcation point (Hopf, Turing or CTHP). The algorithm is encoded in the MAPLE programming environment. A comprehensive tutorial of MSE is given in **Chapter 4**. The practical derivations of amplitude equations in the vicinity of distinct bifurcation points via `Amp_solving` are detailed in the **Appendices E–H**.

In **Chapter 5**, I move back to the Brusselator and cortical models. I analyse the mode stability of the Brusselator model via its amplitude equations; then I apply the MSE program to derive the amplitude equation of the cortical model to explain those patterns that cannot be predicted by LSA.

In **Chapter 6**, an anaesthesia model for the cortex is established for understanding the mechanism of the slow waves that feature during the unconscious state of the brain. We argue that such slow waves may result from complicated bifurcations of the brain. The `Amp_solving` MAPLE program is applied to the anaesthesia model for the derivation of the amplitude equation at CTHP. Nonlinear analysis of the bifurcation is then performed. The model suggests an EEG coherence drop with the induction of propofol anaesthesia. This theoretical prediction is examined in my clinical EEG studies via a custom MATLAB algorithm `EEG_coherence`.

Chapter 7 concludes the whole thesis and outlines possible future work which extends current research in pattern selection and formation of the Waikato cortical model.

Original work

My original contributions to the work include:

- Chapter 4: detailed explanation of the multiple-scale expansion and the development of its automatic algorithm `Amp_solving`
- Chapter 5: nonlinear analysis of the pattern dynamics for both the Brusselator and the Waikato cortical models via amplitude equations
- In Chapter 6, the nonlinear analysis for the anaesthesia model; and the development of `EEG_coherence` algorithm to study the phase-coherence of clinical EEG recordings
- Appendix A: MATLAB codes for solving numerically the 2D Brusselator model by Runge-Kutta solver `ode45`
- Appendix D: SIMULINK modelling for the 2D Brusselator and Waikato cortical models
- Appendices E–H: all `Amp_solving` codes dedicated to realising the multiple-scale expansion and its application to the Brusselator model.

Publications

Refereed Journal Papers

- Kaier Wang, Moira L. Steyn-Ross, D. Alistair Steyn-Ross, and Marcus T. Wilson: Derivation of amplitude equations for reaction–diffusion systems using a computer-aided multiple-scale expansion. *International Journal of Bifurcation and Chaos* **24**(7), 1450101 (2014).
- Kaier Wang, Moira L. Steyn-Ross, D. Alistair Steyn-Ross, Marcus T. Wilson, Jamie W. Sleigh and Yoichi Shiraishi: Simulations of pattern dynamics for reaction–diffusion systems via SIMULINK. BMC Systems Biology **8**(1), 45 (2014)
- Kaier Wang, Moira L. Steyn-Ross, D. Alistair Steyn-Ross, and Marcus T. Wilson: EEG coherence changes in propofol-induced general anaesthesia: Experiment and theory. *submitted to Frontiers in Systems Neuroscience*, under review

Refereed Conference Proceedings:

- Kaier Wang, Moira L. Steyn-Ross, D. Alistair Steyn-Ross, Jamie W. Sleigh and Marcus T. Wilson. Modelling of non-cognitive cortical dynamics and its analysis through computerised derivation of amplitude equations, *20th Australian Institute of Physics Congress*, the University of New South Wales, Sydney, Australia, 9th-13th December, 2012

Conference Presentations:

- Kaier Wang, Moira L. Steyn-Ross, D. Alistair Steyn-Ross, Jamie W. Sleigh and Marcus T. Wilson. Underlying mechanism for the slow oscillation observed in non-REM sleep, general anesthesia and generalized seizure. *Abstracted in Conference on Computational Physics 2012*, Kobe, Japan, 14th-18th October, 2012
- Kaier Wang, Moira L. Steyn-Ross, D. Alistair Steyn-Ross, Jamie W. Sleigh and Marcus T. Wilson. Mean-field cortical modelling and its seizure event prediction through computerized normal form derivation. *Abstracted in NZIP Conference 2011*, Rutherford House, Victoria University, Wellington, New Zealand, 17th-19th October, 2011
- Kaier Wang, Moira L. Steyn-Ross, D. Alistair Steyn-Ross, Jamie W. Sleigh and Marcus T. Wilson. Nonlinear dynamic based continuum cortical modelling and its simulation of interacting Turing and Hopf instabilities. *Abstracted in 28th International Australasian Winter Conference on Brain Research*, Edgewater Resort, Wanaka, New Zealand, 28th August-1st September, 2010
- Kaier Wang, Moira L. Steyn-Ross, D. Alistair Steyn-Ross, Jamie W. Sleigh and Marcus T. Wilson. Interacting Turing and Hopf instabilities in a mean-field model of the cerebral cortex. *Abstracted in 2009 New Zealand Postgraduate Conference*, Victoria University, Wellington, New Zealand, 20-21st November, 2009

Overseas research visiting

The SIMULINK modelling work in Appendix D was conducted during my 3-month research visit to *Embedded Systems Lab* (led by Associate Professor Yoichi Shiraishi) in Gunma University, Japan; and has been presented at:

- Research seminar at Faculty of Information and Systems, 3rd April 2013, University of Tsukuba, Japan, hosted by Professor Hidetoshi Konno;
- Research seminar at Department of Production Science and Technology, 25th April 2013, Gunma University, Japan, hosted by Associate Professor Yoichi Shiraishi;
- Postgraduate seminar at School of Engineering, 11th July 2013, the University of Waikato, New Zealand.

This work is supported by *New Zealand–Japan Exchange Programme*[3].

[3] `http://ilep.ac.nz/pld-opportunities/pld-opportunities-abroad/japanese-programmes/the-nzjep-programme`

Chapter 2

Pattern dynamics of the Brusselator model

Reaction–diffusion systems describe dynamics under the influence of two processes: the reaction, which transforms one set of substances, the reactants, to another set of substances, the products; and the diffusion, which allows the random movement of substances in space. British mathematician Alan Turing proposed that diffusion can be a destabilising factor for the homogeneous stable state of a two-component reaction [127], that is, the introduction of diffusion can lead to the spontaneous formation of patterns. A prominent example of reaction–diffusion is the Brusselator — a system of model reactions developed in 1970 by Ilya Prigogine and his collaborators at the Free University of Brussels [22]. The name *Brusselator* is a portmanteau of the words *Brussels* and *oscillator*. The Brusselator model is one of the simplest chemical models exhibiting Turing instability and other pattern-forming instabilities [24,153]. In this chapter we first review the reaction–diffusion system in a general form and discuss the Hopf and Turing instabilities via linear stability analysis (LSA). We then apply these theories to the Brusselator model to investigate the performance of LSA in pattern prediction.

2.1 Introduction to reaction–diffusion systems

Consider a generalised reaction–diffusion system for two reacting and diffusive species U and V of the form:

$$\begin{aligned} \frac{\partial U}{\partial t} &= f_U(U,V) + D_U \nabla^2 U \\ \frac{\partial V}{\partial t} &= f_V(U,V) + D_V \nabla^2 V \end{aligned} \tag{2.1}$$

The diffusion constant $D_{U,V}$ [with units (length)^2/time] is an important parameter indicative of the diffusion mobility. For a multi-component system, the higher the diffusivity, the faster the species diffuse into each other. Here, $f_{U,V}(U,V)$ is typically a nonlinear function of concentrations U and V.

The stationary uniform state (U_0, V_0) is derived by setting $\partial/\partial t = \nabla^2 = 0$, leading to

$$\begin{aligned} f_U(U_0, V_0) &= 0 \\ f_V(U_0, V_0) &= 0. \end{aligned} \tag{2.2}$$

Linear stability analysis (LSA) is performed by introducing weak perturbations $(\delta U, \delta V)$ around the steady states: $\delta U = U - U_0$ and $\delta V = V - V_0$. The original system Eq. (2.1) is simplified by keeping only the linear terms of a Taylor expansion about the steady state:

$$\begin{aligned} \frac{\partial}{\partial t}(U_0 + \delta U) &= f_U(U_0 + \delta U, V_0 + \delta V) + D_U \nabla^2 (U_0 + \delta U) \\ &= f_U(U_0, V_0) \; + \; \left.\frac{\partial f_U}{\partial U}\right|_0 \delta U \; + \; \left.\frac{\partial f_U}{\partial V}\right|_0 \delta V \; + \; D_U \nabla^2 \delta U \; + \; \text{higher order terms} \end{aligned}$$

which approximates to

$$\frac{\partial}{\partial t}\delta U = \left.\frac{\partial f_U}{\partial U}\right|_0 \delta U \; + \; \left.\frac{\partial f_U}{\partial V}\right|_0 \delta V \; + D_U \nabla^2 \delta U. \tag{2.3}$$

where the "$|_0$" notation indicates evaluation of steady-state.

Similarly, the second equation of Eq. (2.1) has the linearised form:

$$\frac{\partial}{\partial t}\delta V = \left.\frac{\partial f_V}{\partial U}\right|_0 \delta U \; + \; \left.\frac{\partial f_V}{\partial V}\right|_0 \delta V \; + \; D_V \nabla^2 \delta V. \tag{2.4}$$

The linearised system Eqs. (2.3) and (2.4) can be converted to the vectorised form:

$$\frac{\partial}{\partial t}\begin{bmatrix} \delta U \\ \\ \delta V \end{bmatrix} = \left.\begin{bmatrix} \dfrac{\partial f_U}{\partial U} & \dfrac{\partial f_U}{\partial V} \\ \dfrac{\partial f_V}{\partial U} & \dfrac{\partial f_V}{\partial V} \end{bmatrix}\right|_0 \begin{bmatrix} \delta U \\ \\ \delta V \end{bmatrix} + \begin{bmatrix} D_U \nabla^2 & 0 \\ \\ 0 & D_V \nabla^2 \end{bmatrix} \begin{bmatrix} \delta U \\ \\ \delta V \end{bmatrix}. \tag{2.5}$$

The substitution of a trial solution set $(\delta U, \delta V) = (A_U e^{i\vec{q}\cdot\vec{r}}, A_V e^{i\vec{q}\cdot\vec{r}})$ yields

$$\frac{\partial}{\partial t}\begin{bmatrix} A_U \\ \\ A_V \end{bmatrix} = \left.\begin{bmatrix} \dfrac{\partial f_U}{\partial U} - q^2 D_U & \dfrac{\partial f_U}{\partial V} \\ \dfrac{\partial f_V}{\partial U} & \dfrac{\partial f_V}{\partial V} - q^2 D_V \end{bmatrix}\right|_0 \begin{bmatrix} A_U \\ \\ A_V \end{bmatrix}, \tag{2.6}$$

where

$$\left.\begin{bmatrix} \dfrac{\partial f_U}{\partial U} - q^2 D_U & \dfrac{\partial f_U}{\partial V} \\ \dfrac{\partial f_V}{\partial U} & \dfrac{\partial f_V}{\partial V} - q^2 D_V \end{bmatrix}\right|_0 \tag{2.7}$$

is called the Jacobian matrix. The trial solution set is actually a mode ansatz featuring the wavenumber q and amplitudes (A_U, A_V) for reactants (U, V). The linear combination of eigenvectors $\begin{bmatrix} V_1 \\ V_2 \end{bmatrix}$ corresponding to the dominant eigenvalue σ_q (that has the largest real part) of the Jacobian matrix yields the solution of Eq. (2.6):

$$\begin{bmatrix} A_U \\ A_V \end{bmatrix} = c_1 \begin{bmatrix} V_1 \\ V_2 \end{bmatrix} e^{\sigma_q t} + c_2 \begin{bmatrix} V_1^* \\ V_2^* \end{bmatrix} e^{-\sigma_q t} \tag{2.8}$$

where c_1 and c_2 are constants; the superscript * stands for complex conjugate. The dominant eigenvalue is a function of q. The real part α describes the mode's linear growth rate, and the imaginary part ω indicates the mode's oscillating frequency if $\omega \neq 0$. It is straightforward to understand that the amplitude of the perturbation will grow (decay) exponentially if the real part of the dominant eigenvalue is positive (negative). Since eigenvalues are derived from the Jacobian matrix at a steady state, we have a stability condition for the steady state:

Theorem (Linear stability condition)

$$\begin{aligned} \alpha = \mathrm{Re}(\sigma_q) > 0 &\Rightarrow \text{the steady state is unstable} \\ \alpha = \mathrm{Re}(\sigma_q) < 0 &\Rightarrow \text{the steady state is stable} \end{aligned} \tag{2.9}$$

There are four main classes of instability:

1. A **Hopf bifurcation** manifests with a global oscillation (at zero wavenumber) with the nonzero frequency that is the imaginary part of the dominant eigenvalue:
   - $q = 0, \omega > 0 \Rightarrow$ Hopf instability; A sign change in α signals the onset of a Hopf instability, with $\alpha = 0$ marking the bifurcation point.
2. A **Turing bifurcation** manifests with spatial inhomogeneity at a nonzero wavenumber:
   - $q \neq 0, \alpha > 0, \omega = 0 \Rightarrow$ Turing instability; with $\alpha = 0$ marking the Turing bifurcation point.
3. A **mixed Turing-Hopf (TH)** instability occurs near a codimension-2 TH point (CTHP) when critical Turing and Hopf conditions coincide:
   - $\alpha = 0, \omega > 0$ at $q = 0$, with $\sigma_q = 0$ at $q \neq 0$.
4. A **wave instability**[1] manifests with spatial inhomogeneity at a nonzero wavenumber with nonzero frequency:
   - $q \neq 0, \alpha \geq 0, \omega > 0 \Rightarrow$ wave instability.

We summarise the threshold for each type of bifurcation in Table 2.1.

It is straightforward to predict the mode stability by inspecting the eigenvalue dispersion curve showing the variation of the dominant eigenvalue as a function of wavenumber

[1]The analysis of the wave instability is detailed in [81]. In our work, we mainly focus on investigating Turing, Hopf and TH bifurcations.

Table 2.1: Threshold of bifurcations (in theory)

			Pattern stability	
Bifurcation	Critical wavenumber	Eigenvalue ($\sigma_q = \alpha + i\omega$)	Spatially	Temporally
Turing	$q \neq 0$	$\sigma_q = 0$	unstable	stable
Hopf	$q = 0$	$\alpha = 0$, $\omega \neq 0$	stable	unstable
TH	$q \geq 0$	$\alpha = 0$, $\omega \neq 0$ at $q = 0$	unstable	unstable
Wave	$q \neq 0$	$\alpha \geq 0$, $\omega > 0$	wave instability	

q. In the next section, we will apply LSA to the Brusselator model to examine the LSA performance in predicting the dynamics of the system.

2.2 The Brusselator model

The Brusselator model describes chemical reactions given as [82]

$$\begin{aligned} A &\xrightarrow{k_1} X \\ B + X &\xrightarrow{k_2} Y + D \\ 2X + Y &\xrightarrow{k_3} 3X \\ X &\xrightarrow{k_4} E \end{aligned} \tag{2.10}$$

where k_i is the reaction rate-constant[2] quantifying the rate of the i-th chemical reaction. A, B, X and Y are species. D, E, the final products, are removed from the reaction pool as soon as they are produced.

The law of mass action states:

"*The rate change of the concentration of each species in a reaction is the product of its stoichiometric coefficient with the rate of the reaction, adjusted for sign ("+" if product, "−" if reactant)*".

So the rate equations for X and Y are:

$$\begin{aligned} \frac{\partial [X]}{\partial T} &= k_1[A] - (k_2[B] + k_4)[X] + k_3[X]^2[Y] + d_1 \nabla^2 [X] \\ \frac{\partial [Y]}{\partial T} &= k_2[B][X] - k_3[X]^2[Y] + d_2 \nabla^2 [Y] \end{aligned} \tag{2.11}$$

[2] The units of the rate constant depend on the order of the reaction. If the concentration is measured in units of mol$\cdot$ L^{-1} (abbreviated as M), then:

- For order one, the rate constant k_1 and k_4 have units of s^{-1}
- For order two, the rate constant k_2 has units of (M$\cdot$ s)$^{-1}$
- For order three, the rate constant k_3 has units of (M$^2\cdot$ s)$^{-1}$
- For order $m + n$, the rate constant k_i has units of mol$^{1-(m+n)}\cdot$ L$^{(m+n)-1}\cdot$ s^{-1}

The concentration for each species is denoted by [ ] with units mol· L^{-1}. Here, d_1 and d_2 are diffusion constants for $[X]$ and $[Y]$ respectively.

For better insight into the qualitative behaviour, the following scaled variables are introduced:

$$\begin{aligned} t &= k_4 T, & X &= \sqrt{\frac{k_3}{k_4}}[X], & Y &= \sqrt{\frac{k_3}{k_4}}[Y] \\ A &= \sqrt{\frac{k_1^2 k_3}{k_4^3}}[A], & B &= \frac{k_2}{k_4}[B], & D_i &= \frac{d_i}{k_4} \end{aligned} \tag{2.12}$$

By substituting these relations into Eq. (2.11), one finds the commonly seen dimensionless Brusselator system (note we have replaced $D_{1,2}$ by $D_{X,Y}$ for better readibility):

$$\begin{aligned} \frac{\partial}{\partial t}X &= A - (B+1)X + X^2 Y + D_X \nabla^2 X \\ \frac{\partial}{\partial t}Y &= BX - X^2 Y + D_Y \nabla^2 Y \end{aligned} \tag{2.13}$$

Here, X is the activator enhancing its own rate of production; Y is the inhibitor suppressing X's and its own rate of production. Turing established that a necessary condition for the formation of spatial patterns in a chemical reaction is that the inhibiting substance must diffuse more rapidly than the activator [127]: $D_Y > D_X$.

The steady state of the Brusselator model $(X_0, Y_0) = (A, B/A)$ is given by setting $\partial X/\partial t = \partial Y/\partial t = 0, D_X = D_Y = 0$ in Eq. (2.13). We determine the stability of the steady states by examining the eigenvalue solved from the Jacobian matrix:

$$J^{\mathrm{Bru}} = \begin{bmatrix} B - 1 - q^2 D_X & A^2 \\ -B & -A^2 - q^2 D_Y \end{bmatrix} \tag{2.14}$$

A bifurcation occurs when there is a qualitative change in the dynamics of a system as a control parameter is varied, which can be predicted by the value of the dominant eigenvalue (see Eq. (2.8)). Following Nicolis and Prigogine's and other researchers' work [24, 82, 93, 94, 151], we choose B as the bifurcation control parameter. Eigenvalues are found from the characteristic equation:

$$\sigma^2 - \sigma\Sigma + \Delta = 0 \tag{2.15}$$

where Σ and Δ are the trace and determinant of the matrix J^{Bru}: $\Sigma = B - 1 - A^2 - q^2(D_X + D_Y)$, $\Delta = A^2 + q^2[A^2 D_X + (1-B)D_Y] + q^4(D_X D_Y)$. The two eigenvalues are:

$$\sigma_{\pm} = \frac{1}{2}[\Sigma \pm \sqrt{\Sigma^2 - 4\Delta}] \tag{2.16}$$

The onset of the Hopf bifurcation occurs when $\Sigma = 0$, which results in the critical Hopf condition $B_c^{\mathrm{H}} = 1 + A^2$. Two eigenvalues are $\sigma_{\pm}^{\mathrm{H}} = \pm iA$, where $i \equiv \sqrt{-1}$.

The threshold of the Turing instability is obtained by setting $\sigma_{\pm} = 0$, which results to the critical Turing condition $B_c^{\mathrm{T}} = (1 + A\eta)^2$ where $\eta = \sqrt{D_X/D_Y}$. The wave number corresponding to the most unstable Turing mode is given by $q_c^2 = A/\sqrt{D_X D_Y}$.

If $\eta = (\sqrt{A^2+1} - 1)/A$ then the two bifurcations are close together, leading to a combination of spatial patterns superimposed with temporal oscillations. Here $B_c^{\mathrm{T}} = B_c^{\mathrm{H}} = 1 + A^2 \equiv B_c^{\mathrm{TH}}$. Since the diffusion rate η and B both participate in modulating such mixed bifurcations, it is called a Turing-Hopf (TH) bifurcation at a codimension-2 TH point (CTHP). A summary of the critical settings for each type of bifurcation is shown in Table 2.2.

Table 2.2: Threshold of bifurcations (Brusselator model)

Bifurcation	B setting	Critical wavenumber	Critical frequency
Turing	$(1 + A\eta)^2$	$\sqrt{A}/(D_X D_Y)^{1/4}$	0
Hopf	$1+A^2$	$q = 0$	A
TH	$1+A^2$	$\sqrt{A}/(D_X D_Y)^{1/4}$	A

2.2.1 Linear stability analysis vs simulations

Assuming A and $D_{X,Y}$ are fixed, we vary B and investigate the LSA predictions. Following the ideas discussed in the previous section, the critical Turing condition B_c^{T} and the critical Hopf condition B_c^{H} can be numerically determined. B should be greater than the critical value for the emergence of the corresponding bifurcation. In the following, we will observe the simulated pattern dynamics and examine the performance of LSA.

Hopf simulation

A Hopf bifurcation is manifest by homogeneous temporal oscillations, observed in a diffusion-free (i.e., $D_X = D_Y = 0$) simulation. The simulation is coded in MATLAB using the built-in `ode45()` 4th order Runge-Kutta solver[3]. We set $A = 3$ for this experiment, thus $B_c^{\mathrm{H}} = 10$.

We investigated the time evolution of the Brusselator model for B below the bifurcation threshold B_c^{H}: $B_1 = 9.9$; above the threshold: $B_2 = 10.2$; and at $B_3 = 10.8$, a further distance from the threshold. All three simulations start from the equilibrium state plus a small perturbation, which plays an important role to kick the system away from the equilibrium state. The simulation runs for 100 s[4] with the time step 0.1 s. Simulation results are shown in Fig. 2.1

[3]Source code available in Appendix A; a SIMULINK model is presented in Appendix 2.2

[4] Strictly speaking, the simulation for the Brusselator model runs in a unit-time, because the model equation Eq. (2.13) is dimensionless. Here, we just assume that the simulation runs in sec.

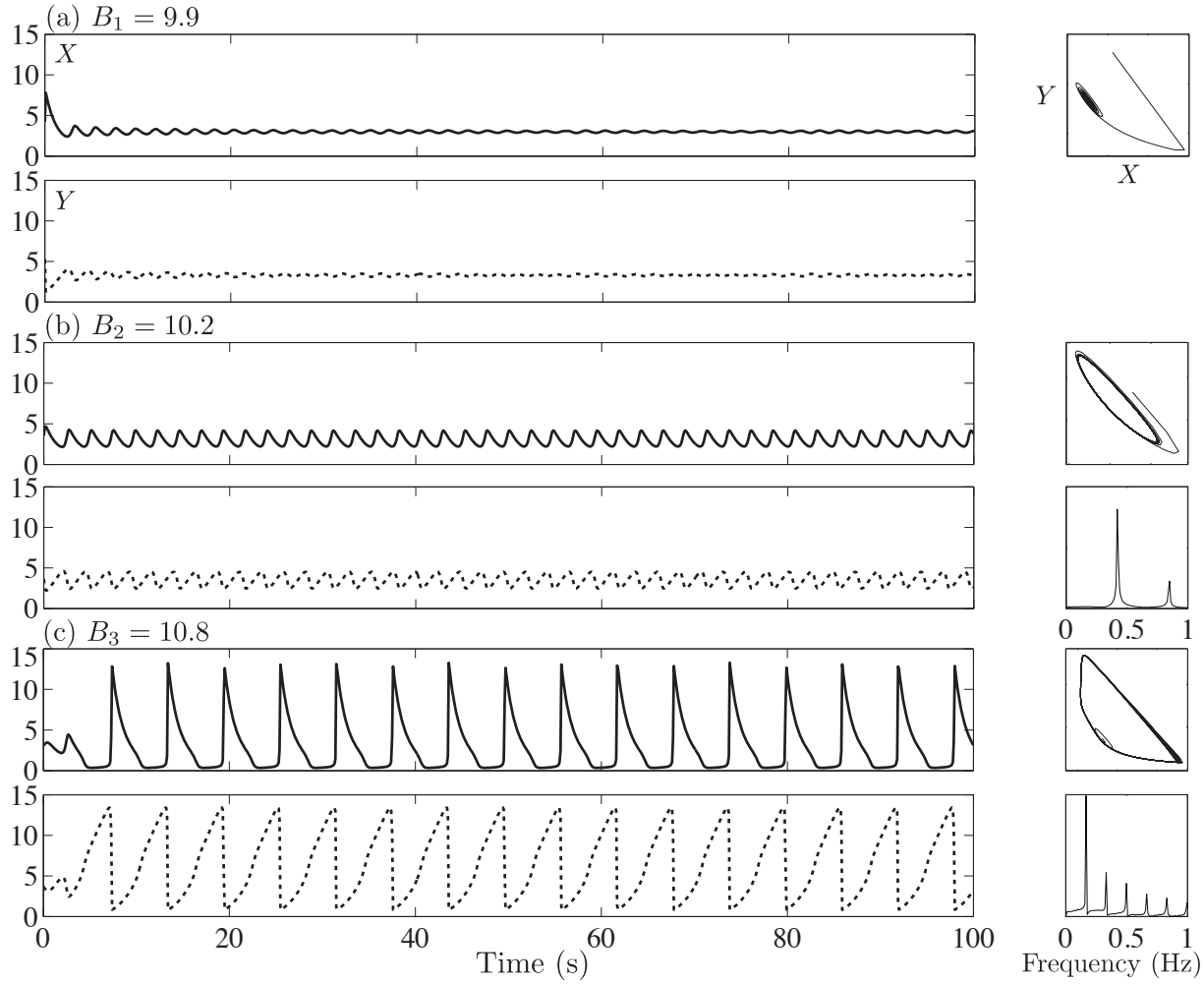


Figure 2.1: 1-D simulation of the Brusselator model when the bifurcation control parameter B is (a) $B_1 = 9.9$ (below Hopf threshold), (b) $B_2 = 10.2$ (above threshold), and (c) $B_3 = 10.8$ (further away from threshold). For each simulation, the time evolution for the activator X is shown at the upper panel and the inhibitor Y is shown at the lower panel. The plot of the phase plane of X and Y is located at top right for each simulation. Fourier power spectrum measuring the oscillating frequency is placed at lower right for each simulation run. (Note that the Hopf oscillation cannot be formed at $B_1 < B_c^{\mathrm{H}}$, so we did not show the Fourier power spectrum.)

The comparison of LSA predictions and simulations observed from Fig. 2.1 are summarised in Table 2.3. It is clear to see that LSA matches well with two cases: $B_1 = 9.9$ and $B_2 = 10.2$. But when B is relatively further away from the critical point, reaching $B_3 = 10.8$, there is a discrepancy between the oscillating frequency predicted by LSA and the numerical simulation.

When $B < B_c^{\mathrm{H}}$, the dominant eigenvalue derived from the Jacobian matrix has a negative real part, meaning that the added perturbation to the steady state will decay exponentially. This is because in Eq. (2.8), the amplitude of the perturbation is proportional to $e^{\sigma_q t}$ where $\mathrm{Re}(\sigma_q)$ is now negative. Therefore, the perturbed system will decay back to the initial steady state. From the first phase plane in Fig. 2.1, the limit cycle

Table 2.3: LSA predictions vs simulations of the Brusselator model at Hopf mode

B value	Dominant eigenvalue	Predicted frequency	Simulations
9.9	$-0.0500 + 2.9996i$	0.4774 Hz	weakly damped oscillations about the steady state
10.2	$0.1000 + 2.9983i$	0.4772 Hz	∼0.47 Hz stable oscillation
10.8	$0.4000 + 2.9732i$	0.4732 Hz	∼0.15 Hz stable oscillation

cannot form but converges to a dot, which means the system is gradually becoming quiescent. The actual observed fluctuations may be from the initially added noise. Moreover, it is natural that noise will exist in a living system.

When $B > B_c^{\mathrm{H}}$, the Hopf oscillation is expected to emerge. From Fig. 2.1, we can see that even when B_2 is just above threshold, stable oscillations over time are observed, giving rise to a limit cycle. The frequency of the X time-series is detected as ∼0.47 Hz, which is very close to the LSA prediction.

When the value of B is increased from 10.2 to 10.8, Fig. 2.1(c) shows the amplitude of the time-series becomes larger, which can be explained from Eq. (2.8): the increased real part of the dominant eigenvalue (see Table. 2.3) leads to the exponential growth of the perturbation. We also find that competition between the activator X and its counterpart inhibitor Y gives rise to anti-phased time-series. Unfortunately, the measured frequency from Fourier spectrum reads ∼0.15 Hz, which is different from the LSA predicted value of 0.4732 Hz. We argue that this discrepancy results from the fact that the nonlinear effect has become sufficiently strong that LSA not longer applied since it deals only with weakly nonlinear phenomena.

Turing simulation

The Turing bifurcation is commonly known by its beautiful patterns that are spatially structured with temporal stability. The parameter setting for this experiment reads: $A = 5, D_X = 5, D_Y = 40$. From Table 2.2, the critical value B_c^{T} to trigger the Turing bifurcation is $(1 + A\eta)^2$ that turns out to be 7.66. In the following experiments, the pattern dynamics are observed by increasing the value of B from the neighbourhood of the threshold to a large value. Through LSA, the stability of the Turing mode is examined according to the real part of the dominant eigenvalue at a specific wavenumber as shown in Fig. 2.2.

In Fig. 2.2, the onset of Turing instability is predicted to be observed when $B = 8.04$ ($> B_c^{\mathrm{T}} = 7.66$) as the real part of the dominant eigenvalue becomes positive in the vicinity of the wavenumber $q/2\pi = 0.09$ (waves per cm). Increasing the value of B, the eigenvalue dispersion curve moves upwards which implies a larger growth rate and larger amplitude of the mode. The parameter settings from Fig. 2.2 are fed into the MATLAB program[5] for

[5]Source code available in Appendix A

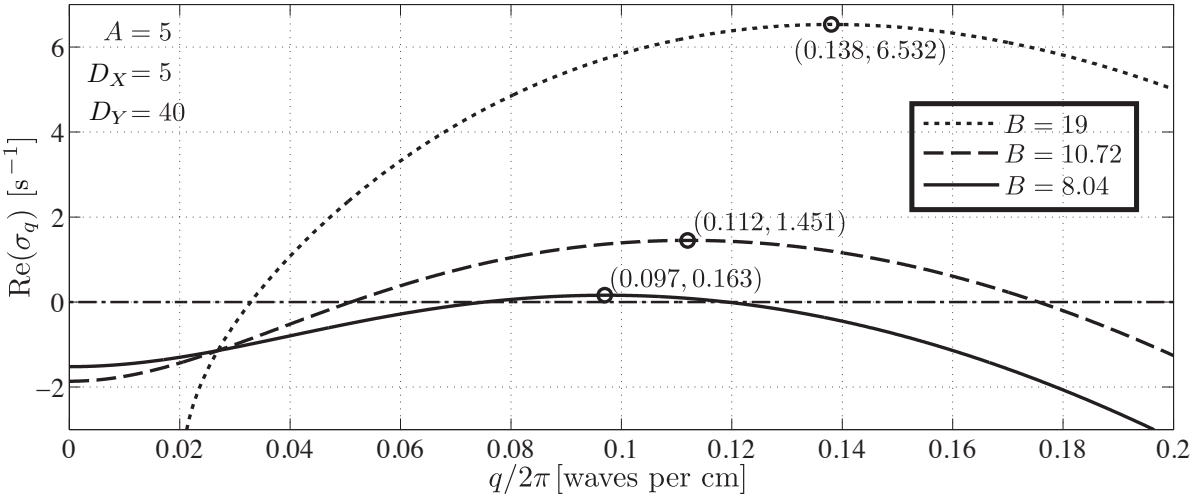


Figure 2.2: Brusselator stability curves for various B values. Real parts of dominant eigenvalue σ_q are plotted as a function of scaled wavenumber for three settings of bifurcation control parameter B: $B = 8.04$, 10.72 or 19. The peak of the curve (corresponds to the most unstable mode) is located at $q/2\pi = 0.097$ when $B = 8.04$; $q/2\pi = 0.112$ when $B = 10.72$; $q/2\pi = 0.138$ when $B = 19$.

the Brusselator model. For the 2D simulation of the Brusselator model, we initialise the model as a 2D sheet consisting of a 60×60 grid[6] (spatial resolution = 1 cm/grid-point) with each grid-point representing the local concentration of the reactant. Edges of the sheet are joined to give toroidal boundaries. The Laplacian operator ∇^2 is implemented as a circular convolution of the 3×3 second-difference operator L:

$$L = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \tag{2.17}$$

All simulations commence from the homogeneous steady-state with added small amplitude Gaussian-distributed white noise. The noise perturbation is turned off after one iteration step leaving the pattern-forming system to organise itself spontaneously. Simulation results with respect to different B values are seen in Fig. 2.3.

From panel (a) of Fig. 2.3, although $B = 8.04$ is only 5% above the threshold ($B_c^{\mathrm{T}} = 7.66$), a Turing pattern can form if the simulation runs for sufficient time. By increasing the B value, stable Turing patterns are found more promptly in panels (b) and (c) of Fig. 2.3. These simulations show that, provided the value of the bifurcation control parameter exceeds the Turing threshold condition (see Table 2.2), a Turing pattern will emerge.

It is useful to examine more carefully the texture of the Turing patterns and their corresponding two-dimensional Fourier amplitude spectrum. The pattern formed at $B = 19$ ("bullet") seems like an anti-phased (upside down) version of the one formed at $B = 8.04$

[6]Since the Brusselator model is dimensionless, we imply the space scale between two grid-point is $\Delta x = 1$cm. So the model 2D-reaction pool has the side length 60 cm

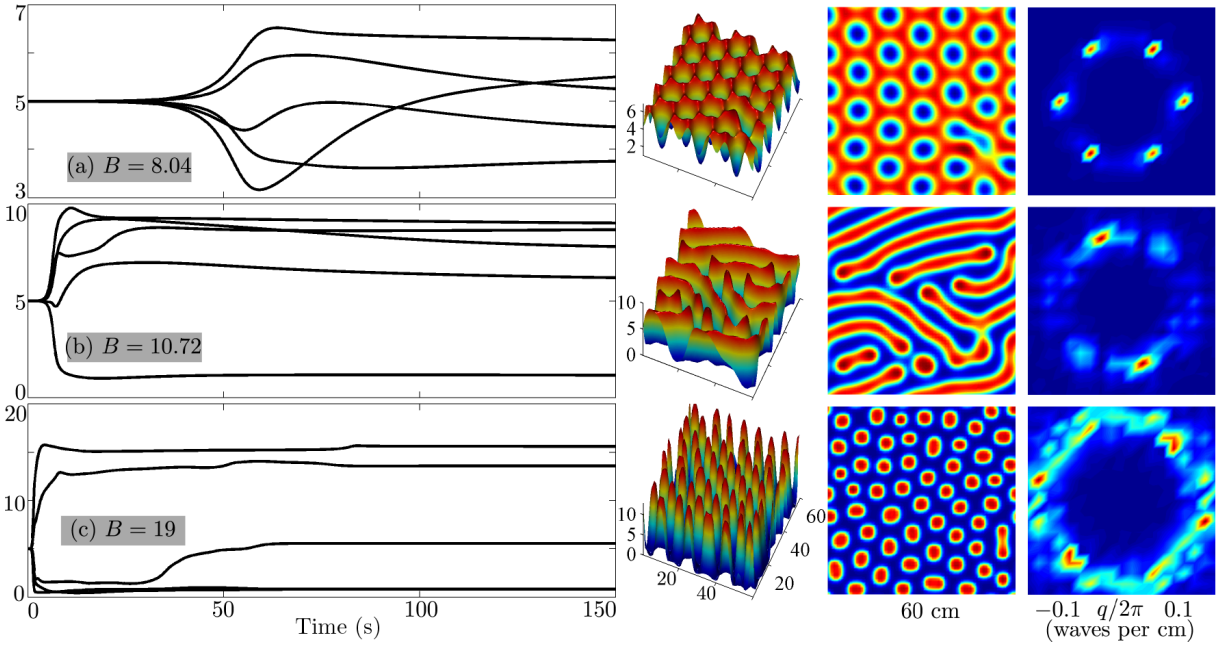


Figure 2.3: Grid simulations of the Brusselator model for the observation of Turing pattern at (a) $B = 8.04$, (b) $B = 10.72$ and (c) $B = 19$. Other parameter settings are: $A = 5, D_X = 5, D_Y = 40$. The pattern formed at $t = 150$ s of the activator X is shown in the second (3D-view) and third (2D-view) columns with its Fourier amplitude spectrum placed at the rightmost column. Colour of the pattern indicates the local concentration of the reactant: [red] high concentration, [blue] low concentration. At the left hand side, is the time-series extract of 5 equally spaced grid-points along the mid-vertical axis of the pattern.

("ice-cream cone"). We also argue that the stripes formed at $B = 10.72$ may be a middle-state between "ice-cream cone" and "bullet" patterns. On the other hand, the Fourier amplitude spectrum shows increased radius from the top (~ 0.1 cm^{-1}) to bottom (~ 0.13 cm^{-1}) plot, in good agreement with the predicted wavenumber for maximum instability indicated in Fig. 2.2. Furthermore, the top spectrum hints at a hexagonal spectral distribution, indicative of a fixed spatial frequency whose wave vector has preferred directions. Similarly, the spectrum of the stripes shows two opposed wave vectors, while the "bullet" gives rise to a hexagonal-ish shape in which red spots indicate the direction of the wave vector. However, we cannot infer this information from LSA. It is very likely that a nonlinear mechanism is responsible for the fine details of the pattern. This nonlinear mechanism will be investigated in Chapter 5.

Turing-Hopf simulation

From our investigation of the Hopf and Turing modes, we see that the LSA plays an important role in predicting patterns. From LSA the mode stability can be determined by examing the sign of the real part of the dominant eigenvalue. We again refer to Table 2.1 for basic rules in pattern prediction. In this section, we finely adjust the bifurcation

control parameter B and the ratio between diffusion terms D_X/D_Y to observe mixed Turing and Hopf modes.

In the simulations shown in Fig. 2.4, we fixed model parameters $A = 2.5, D_X = 5, D_Y = 10$. B is initially set to 7.4 (upper row of Fig. 2.4) to generate a homogeneous Hopf oscillation, then B is raised to 8 to promote the Turing mode. The stability curve at $B = 7.74$ clearly shows the co-existence of Hopf and Turing modes. In its corresponding strip-chart, the vertical and horizontal view indicates, respectively, the spatial and temporal evolution of the pattern. Such an oscillating Turing mode is the key feature of a TH bifurcation. The time-series of the TH mode shows two frequencies: the Hopf and its subharmonic. de Wit *et al.* [22] point out that such subharmonic phenomena are common near a CTHP. In the third experiment, we increased the value of B to 9. The stability curve of the lower panel predicts that both Turing and Hopf appear with larger amplitude and growth rate. However, the simulation shows a Turing bifurcation only. Such discrepancy may result from the pattern competition with strong nonlinearity.

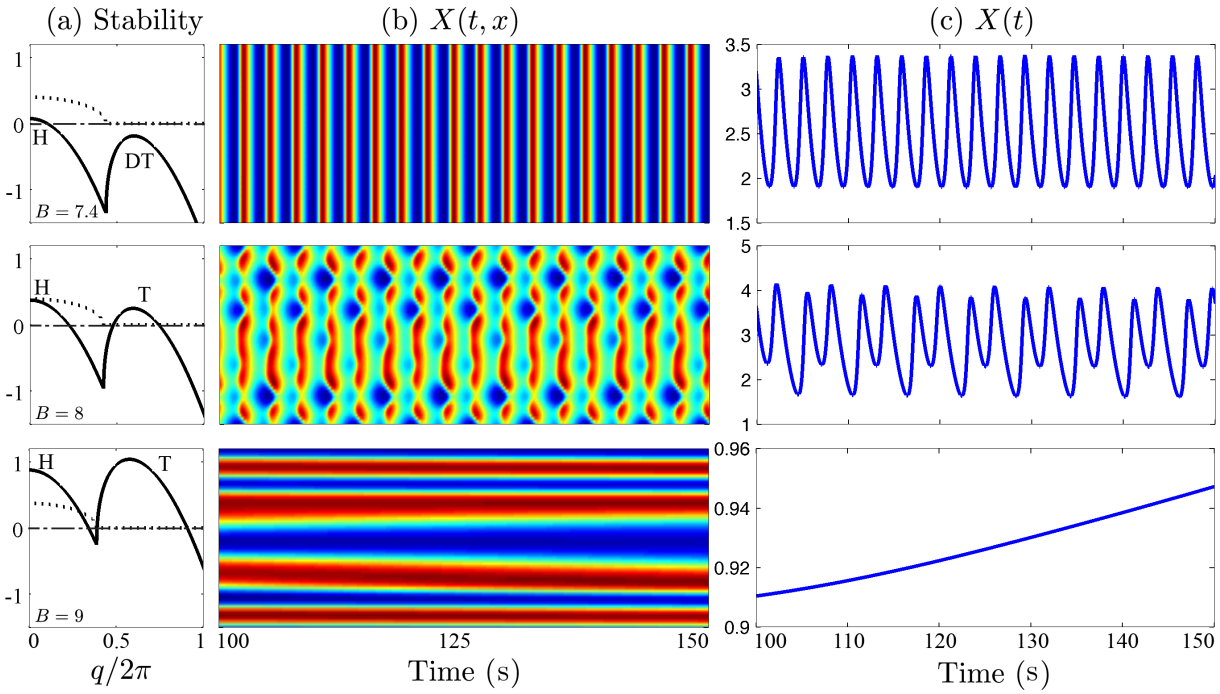


Figure 2.4: Grid simulations of the Brusselator model for the observation of TH pattern with the value of $B = 7.4$ (upper row), $B = 8$ (middle row) and $B = 9$ (bottom row). Other parameter settings are: $A = 2.5, D_X = 5, D_Y = 10$. (a) column is the dispersion curve of [solid] real and [dashed] imaginary parts of dominant eigenvalues predicting the mode stability. H: Hopf mode at wavenumber $q = 0$; T: Turing mode with $\mathrm{Re}(\sigma_q) > 0$ at $q \neq 0$; DT: damped Turing with $\mathrm{Re}(\sigma_q) < 0$ at $q \neq 0$. (b) column is the activator $X(t,x)$ time vs space strip-chart for the last 50 s of the 150-s simulation. The strip-chart is formed by stacking strips extracted from the vertical mid-line of the Brusselator grid at a specific time. Colour indicates the local concentration of the reactant: [red] high concentration, [blue] low concentration. (c) column is the time evolution of X located at the centre of the 2-D sheet.

The Turing time-series shown in the third row of Fig. 2.4 exhibits a slow increase in a small data range. Theoretically the Turing pattern is temporally stable, but the nonlinear effect will drive the pattern (even with completely closed Hopf instability) away from an absolutely frozen state. Fig. 2.5 shows that the Turing time-series is still not completely stable after 2 hours evolution. The bifurcation for a nonlinear system is complicated and practically difficult to predict. Several bifurcations may occur over a slow time-scale. Given sufficient time, we argue that the Turing pattern may evolve to other spatiotemporal dynamics.

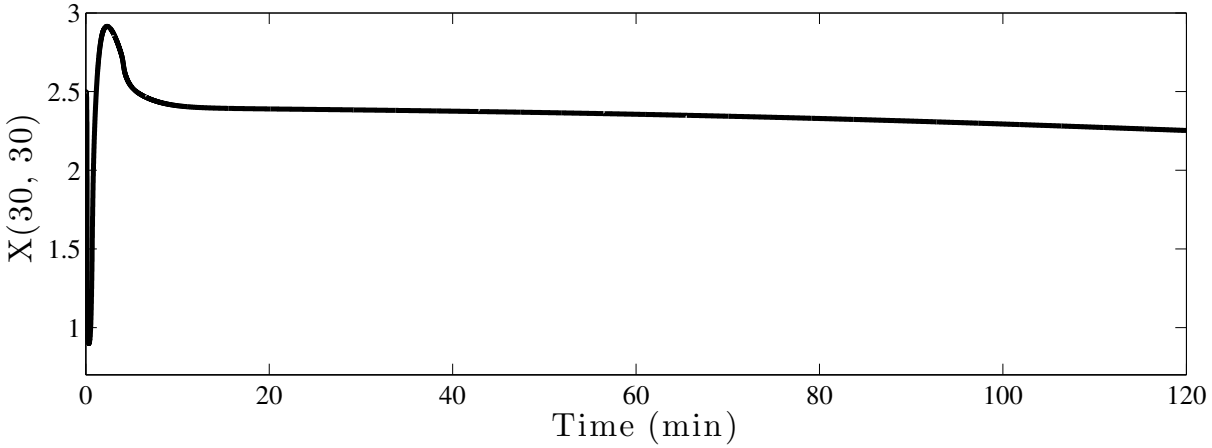


Figure 2.5: 2-hour time-series of the mid-point extracted from a Brusselator pattern (60×60) experencing a Turing bifurcation.

2.3 Discussion

In this chapter, we introduced the reaction–diffusion system and its basic pattern-forming theory. To gain some insight into the pattern formation, linear stability analysis (LSA) was applied. LSA introduces a small spatiotemporal perturbation into the system. By examining the growth rate of the perturbation, the mode stability can be predicted. We applied LSA to the well-known Brusselator model for a preliminary analysis of various pattern dynamics: Hopf, Turing and TH coupled modes. LSA was moderately successful in predicting patterns. However, we did find a mismatch between LSA prediction and pattern simulation: predicted TH mode vs actual Turing mode shown in Fig. 2.4 (bottom row). Moreover, LSA is not able to explain the intrinsic mechanism of the pattern structure, for example the different Turing patterns shown in Fig. 2.3. Later in Chapter 5, we will apply a nonlinear method called amplitude equations for a detailed analysis of pattern formation.

Chapter 3

Pattern dynamics of the Waikato cortical model

Cognition, as a high-level brain function, is not a single neural response, yet appears as a cooperative phenomenon of multiple-interacting neural populations [107]. This raises the question: How can widely separated neural populations that are anatomically unconnected be in very similar states of activity, thereby becoming functionally connected and giving rise to coherent percepts and actions? Aiming to answer this question, the Cortical Modelling Group at the University of Waikato developed a mean-field cortical model (referred as the Waikato cortical model) by treating the cortex as a continuum of excitable tissues, rather than following the exact details of neuron-by-neuron interactions.

The Waikato cortical model has a rich history of development: The model is based on early work by Liley *et al.* [66], with enhancements motivated by Rennie *et al.* [100] and Robinson *et al.* [101]; and has been recently extended to include electrical synapses (also referred as gap junctions) [118, 120] supplementing neural communications via standard chemical synapses.

The Waikato cortical modelling assumptions are:

1. Cortical element is the "macrocolumn" containing ∼100,000 neurons.
2. There are only two distinct kinds of neuron groups: 85% excitatory, and 15% inhibitory neurons. "Excitatory" and "inhibitory" are classified according to their effects on other neurons.
3. There are three isotropic neuronal interactions: cortico-cortical (long-range from other macrocolumns), intra-cortical (short-range within macrocolumn) and connections from subcortical structures such as the thalamus and brain-stem. A macrocolumn with diameter ∼1mm and depth ∼3mm is drawn in Fig. 3.1. We further assume that inhibitory connections via their short axons are locally restricted within a macrocolumn. In contrast, the axons from excitatory neurons are often longer and extensively branched, having length ranging from centimetres to several meters (in giraffe's neck), allowing exclusively excitatory cortico-cortical connections. Both excitatory and inhibitory connections are permitted in local neuron connections.

4. Neurons exchange information via both chemical and electrical (gap-junction) synapses. Gap-junctions are more abundant within inhibitory populations, while being rare within excitatory neuronal populations. It is known that gap-junctions form between only the same types of neurons, i.e., excitatory–excitatory and inhibitory–inhibitory.

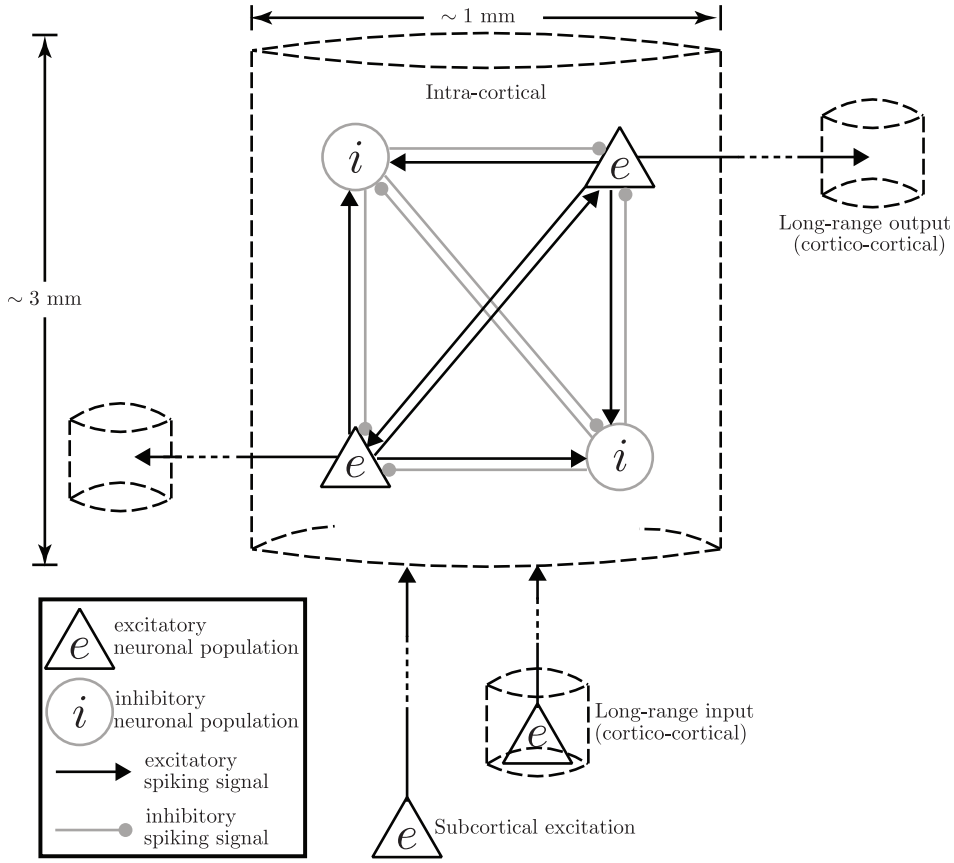


Figure 3.1: Schematic representation of the connective topology within a cortical macrocolumn. Only four of the ∼100,000 neurons are shown. The shapes of neurons are determined based on their observations under microscope: Triangles are excitatory (pyramidal) neurons; Circles are inhibitory neurons.

The Cortical Modelling Group first introduced gap-junctions into the cortical model in the paper *Gap-junction mediate large-scale Turing structures in a mean-field cortex driven by subcortical noise* [118]. Gap-junctions, acting in a similar fashion as the inhibitory diffusion in the Brusselator model, are found to be the critical factor for spontaneously emerging spatial (Turing) patterns of brain activities. Later, Steyn-Ross *et al.* [120] presented two limiting cases of the cortical model: the "slow-soma" limit with slow voltage feedback from soma to dendrite, and the "fast-soma" limit in which such feedback is prompt. Steyn-Ross *et al.* argued that the "slow-soma" limit might describe a "default" background state for the idling, non-cognitive brain; the "fast-soma" limit might relate to the cognition-driven cortical activation. Notice that the "slow-soma" exhibits only temporally stable Turing instability in [120], which does not match with ultra-slow BOLD oscillations ($\leq$0.1 Hz) characterised the default mode of the cortical activation. In a

recent work [117], Steyn-Ross *et al.* proposed that the slowly oscillating spatial mode of the BOLD images might arise from the nonlinear interaction between Turing and Hopf instabilities of the "slow-soma" at a codimension-2 Turing-Hopf point (CTHP).

In the following, we will review modelling strategies of the Waikato cortical model and its multiple dynamics.

3.1 Modelling strategy

The neuron is the basic element of the cortex. A simplified structure of the neuron is described in Fig. 3.2: The dendrite tree gathers information and passes this to the soma body. The axon is the bridge connecting soma and the terminal where signals will be delivered to the next neuron via the synapse.

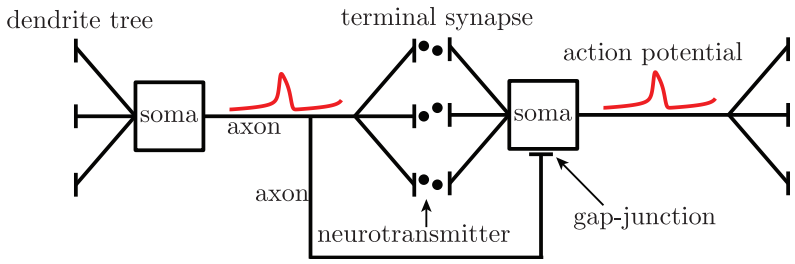


Figure 3.2: Illustration of the connection between two neurons. Soma body receives incoming signals gathered at the dendrite tree. Soma converts these signals to spikes and sends them to the terminal synapse through the axon, further provoking postsynaptic potentials at the target neuron.

Multiple signals are integrated at the soma to trigger action potentials (spikes). Spikes travel along the axon to its terminal. At the axon terminal, incoming spikes can provoke the chemical synapse to release neurotransmitter molecules, which will be captured by the postsynaptic neuron thus giving rise to the postsynaptic potential. In the meantime, electrical synapses (gap-junctions) let ions or charged particles pass through into the next neuron. Gap-junctions can create either a depolarisation (a more positive charge) or a hyper-polarisation (a more negative charge) in the adjacent neuron. If the cell becomes sufficiently depolarised, or sufficiently positive, it generates an action potential, in turn setting off a rapid wave of signals that excites connected neurons.

The Waikato mean-field cortical model is an extended version of the single neuron model. The basic frame for delivering spikes is the same: Firing flux transmits through the axon to the terminal synapse, these incoming spikes will induce postsynaptic potentials, and all spike activities invading the dendrite tree are integrated at the soma, imposing a time-dependent perturbation about the soma resting potential.

With the knowledge of the connective topology of neurons within a cortical macro-column shown in Fig. 3.1, we explain our modelling strategy following the flow of neural

firing flux, from the left to right in Fig. 3.3.

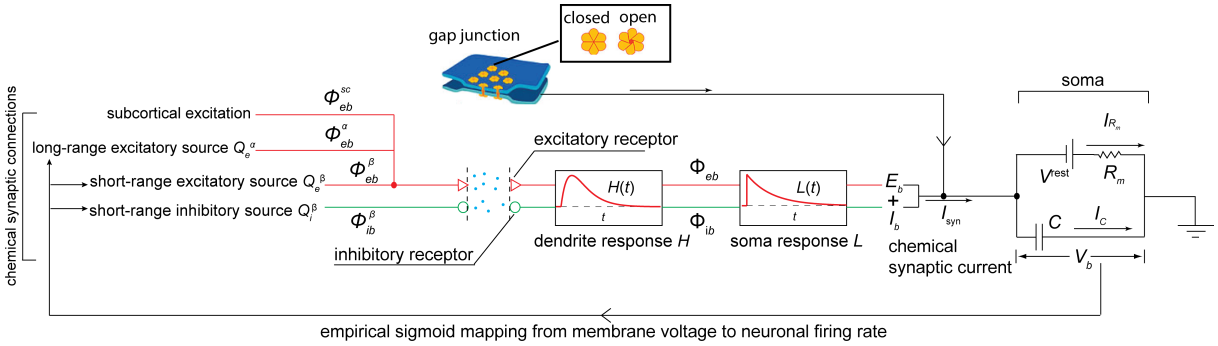


Figure 3.3: Schematic of the Waikato cortical model

3.1.1 Cortico-cortical flux

Flux ϕ_{eb} generated by excitatory source Q_e obeys a 2-D damped-wave equation [101] connecting to population-averaged neuron type b where b can be either e (excitatory) or i (inhibitory):

$$\left[\left(\frac{\partial}{\partial t}+v\Lambda_{eb}\right)^2-v^2\nabla^2\right]\phi_{eb}^{\alpha}=(v\Lambda_{eb})^2Q_e \tag{3.1}$$

Q_e is the spike-rate flux, which is a mapping from membrane voltage V_e to population-averaged firing rates:

$$Q_e=\frac{Q_e^{\max}}{1+e^{-C(V_e-\theta_e)/\sigma_e}} \tag{3.2}$$

According to our modelling assumptions, cortico-cortical fibres are exclusively excitatory since the inhibitory neurons typically have short axons. Thus the long-range wave source is excitatory only. ϕ_{eb}^{α} is the long-range flux from distant excitatory neural source Q_e. The subscript eb is read "$e\to b$", the connection from the presynaptic neuron type e to postsynaptic neuron type b.

Symbol definitions for the cortico-cortical terms are shown in Table 3.1.

Table 3.1: Symbol definition for the cortico-cortical equations

Symbols	Description	Value	Unit
$\Lambda_{e,b}$	inverse-length scale for $e\to b$ axonal connection	4	cm^{-1}
v	axonal conduction speed	140	$\mathrm{cm\ s}^{-1}$
$Q_{e,i}^{\max}$	maximum firing rate	30, 60	s^{-1}
$\theta_{e,i}$	sigmoid threshold voltage	-58.5, -58.5	mV
$\sigma_{e,i}$	standard deviation for threshold	3, 5	mV
C	constant	$\pi/\sqrt{3}$	

3.1.2 Intra-cortical flux

The total input flux arriving at the post-synapse is given as,

$$M_{ab} = \underbrace{N_{eb}^{\alpha}\,\phi_{eb}^{\alpha}}_{\text{long-range}} + \underbrace{N_{ab}^{\beta}\,\phi_{ab}^{\beta}}_{\text{local}} + \underbrace{\phi_{eb}^{\text{sc}}}_{\text{subcortical}} \tag{3.3}$$

where superscripts α and β distinguish long-range and local chemical synaptic inputs; N_{eb}^{α} and N_{ab}^{β} are the number of such input connections. Subcortical excitation ϕ_{eb}^{sc} is modelled as small-amplitude spatiotemporal Gaussian-distributed white noise superimposed on a background excitatory "tone" $\langle\phi_{eb}^{\text{sc}}\rangle$ whose constant level is set via a subcortical drive scale-factor s:

$$\phi_{eb}^{sc} = s\langle\phi_{eb}^{\text{sc}}\rangle + \sqrt{s\langle\phi_{eb}^{\text{sc}}\rangle}\xi_{eb} \tag{3.4}$$

where ξ_{eb} is the Gaussian-distributed white-noise sources are delta-correlated in time and space (generated by MATLAB command `randn()`). Inclusion of white-noise stimulus is motivated by physiological evidence that the cortex requires a continuous background "wash" of input noise to function normally.

The total input flux Φ_{ab} is the temporal convolution of the dendrite impulse response $H(t)$ with the input flux M_{ab}:

$$\begin{aligned}\Phi_{ab} &= (\text{dendrite response}) \otimes (\text{input flux}) \\ &= H_b(t) \otimes M_{ab}(t) \\ &= \int_0^t H_b(t-t')\,M_{ab}(t')dt'\end{aligned} \tag{3.5}$$

The input spike flux M_{ab} releases quanta of neurotransmitters into the synaptic cleft, altering the dendritic conductance and thereby causing a momentary change in the post-synaptic neuron via charge transfer through opened ion channels. Experimental measurements of the post-synaptic potentials (PSPs) shows a rapid-rise, slow-decay curve which is well approximated by the so-called alpha-function $H_b(t)$:

$$H_b(t) = \gamma_{ab}^2 t e^{-\gamma_{ab}t} \tag{3.6}$$

where the time-to-peak is given by $t = 1/\gamma_a$, and the total area under the curve is normalized to unit on the interval $0 \leq t < \infty$ for ease of calculation. Meanwhile, writing $\gamma_{eb} = \gamma_e$ and $\gamma_{ib} = \gamma_i$, Eq. (3.6) can be simplified to:

$$H_b(t) = \gamma_a^2 t e^{-\gamma_a t} \tag{3.7}$$

Reducing γ_i is equivalent to increasing the time-to-peak $(1/\gamma_i)$ for the hyperpolarising GABA response, as indicated in Fig. 3.4.

By taking derivatives[1], Eq. (3.5) becomes

[1]For detailed calculation see Appendix B.1

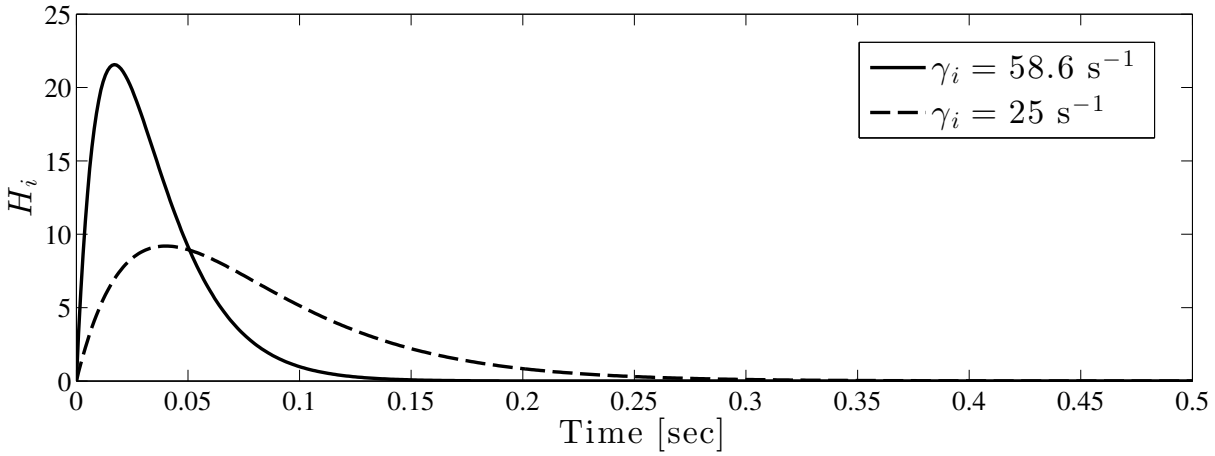


Figure 3.4: IPSP response (inhibitory alpha-function) curve. With delays in the inhibitory postsynaptic response (reduction of the inhibitory rate-constant γ_i), the occurance of the IPSP peak will be postponed. Here, the position of the peak shifts from 0.017 to 0.04 s by reducing γ_i from 58.6 to 25 s^{-1}.

$$\left(\frac{\partial}{\partial t} + \gamma_a\right)^2 \Phi_{ab} = \gamma_a^2 M_{ab}(t) \tag{3.8}$$

Therefore, the flux received by target neural populations are:

$$\left(\frac{\partial}{\partial t} + \gamma_e\right)^2 \Phi_{eb} = [N_{eb}^{\alpha}\phi_{eb}^{\alpha} + N_{eb}^{\beta}Q_e + \phi_{eb}^{sc}]\gamma_e^2 \tag{3.9a}$$

$$\left(\frac{\partial}{\partial t} + \gamma_i\right)^2 \Phi_{ib} = N_{ib}^{\beta}Q_i\gamma_i^2 \tag{3.9b}$$

Symbol definitions for the intra-cortical terms are shown in Table 3.2.

Table 3.2: Symbol definition for the intra-cortical equations

Symbols	Description	Value	Unit
N_{eb}^{α}	long-range $e \to b$ axonal connectivity	2000	-
$N_{eb,ib}^{\beta}$	local $e \to b$, $i \to b$ axonal connectivity	800, 600	-
$\gamma_{e,i}$	excitatory, inhibitory rate-constant	170, 50	s^{-1}
$\langle\phi_{eb}^{\mathrm{sc}}\rangle$	$e \to b$ tonic flux entering from subcortex	300	s^{-1}

3.1.3 Soma voltage

Soma input from chemical synapses

Voltage input to the soma is modelled as postsynaptic potentials (PSPs) entering the neurons somatic "capacitor" (considered to be a single RC compartment), which describes

the low-pass filter characteristics of the soma:

$$\begin{aligned} V_b(t) &= (\text{soma resting potential}) + (\text{input perturbation}) \\ &= (\text{soma resting potential}) + (\text{soma response} \otimes \text{PSPs}) \\ &= V_b^{\text{rest}} + (\text{soma response}) \otimes (\underline{\text{excitatory}} \text{ and } \underline{\text{inhibitory}} \text{ voltage inputs}) \\ &= V_b^{\text{rest}} + L_b \otimes [E_b(t) + I_b(t)] \\ &= V_b^{\text{rest}} + \int_0^t L_b(t-t')[E_b(t') + I_b(t')]dt' \end{aligned} \tag{3.10}$$

The excitatory and inhibitory voltage inputs read

$$E_b(t) = \rho_e \psi_{eb}(t) \Phi_{eb}(t), \quad (\rho_e > 0) \tag{3.11a}$$

$$I_b(t) = \rho_i \psi_{ib}(t) \Phi_{ib}(t), \quad (\rho_i < 0) \tag{3.11b}$$

where

$$\psi_{ab}(t) = \frac{V_a^{\text{rev}} - V_b(t)}{V_a^{\text{rev}} - V_b^{\text{rest}}} \tag{3.12}$$

is a dimensionless weighting factor to capture the dendritic response sensitivity to reversal potential for either excitatory (AMPA) receptors or inhibitory (GABA) receptors. In the present work, we take $V_e^{\text{rev}} = 0$ mV for AMPA (excitatory) receptors, and $V_i^{\text{rev}} = -70$ mV for GABA (inhibitory) receptors.

The soma response is represented as:

$$L_b = \frac{1}{\tau_b} e^{-t/\tau_b} \tag{3.13}$$

where τ_b is the time-constant that gives the decay time for the cell to relax back to its resting voltage. τ_b depends on the resistance R_m and capacitance C_m of the membrane:

$$\tau = R_m C_m \tag{3.14}$$

By taking the derivative[2] of Eq. (3.10), we obtain

$$\begin{aligned} \tau_b \frac{dV_b}{dt} &= -(V_b(t) - V_b^{\text{rest}}) + E_b(t) + I_b(t) \\ &= V_b^{\text{rest}} - V_b(t) + E_b(t) + I_b(t) \end{aligned} \tag{3.15}$$

Dividing both sides of Eq. (3.15) by the membrane resistance R_m:

$$\frac{\tau_b}{R_m} \frac{dV_b}{dt} = \frac{V_b^{\text{rest}} - V_b}{R_m} + \frac{E_b + I_b}{R_m} \tag{3.16}$$

Note that $(E_b + I_b)/R_m$ carries the unit of current, and thus is the chemical synaptic input current I_{syn} generated by the action potentials incident at the dendrite. Considering Eq. (3.14), we re-arrange Eq. (3.16) as:

$$C_m \frac{dV_b}{dt} = \frac{V_b^{\text{rest}} - V_b}{R_m} + I_{\text{syn}} \tag{3.17}$$

[2]For detailed calculation see Appendix B.2

Soma input from electrical synapses

In addition to chemical synapses, gap-junctions are clusters of protein channels that connect interiors of adjacent neurons, allowing direct exchange of ions and small molecules. The gap-junction diffusive current is modelled as (for details of derivation, see [118]):

$$I_{\text{gap}} = \frac{a}{R}\nabla^2 V_b \tag{3.18}$$

with a being the area of the Fukuda-cell and R being the gap-junction resistance [118].

Including I_{gap}, Eq. (3.17) becomes:

$$C_m \frac{\partial V_b}{\partial t} = \frac{V_b^{\text{rest}} - V_b}{R_m} + I_{\text{syn}} + I_{\text{gap}} \tag{3.19}$$

Multiplying both sides by R_m yields:

$$\begin{aligned} \tau_b \frac{dV}{dt} &= V_b^{\text{rest}} - V_b + I_{\text{syn}} R_m + D_{bb}\nabla^2 V \\ &= V_b^{\text{rest}} - V_b + (E_b + I_b) + D_{bb}\nabla^2 V \end{aligned} \tag{3.20}$$

with $D_{bb} = aR_m/R$ the effective diffusion constant [cm^2].

Consequently, the soma voltage changes due to chemical synaptic inputs and the gap-junction diffusion results in the following equation:

$$\tau_b \frac{\partial V_b}{\partial t} = V_b^{\text{rest}} - V_b + \underbrace{\rho_e \psi_{eb}\Phi_{eb} + \rho_i \psi_{ib}\Phi_{ib}}_{\text{chemical synapses}} + \underbrace{D_{1,2}\nabla^2 V_b}_{\text{electrical synapses}} \tag{3.21}$$

We write D_{bb} as the diffusive-coupling strength between electrically adjoined $e \to e$, $i \to i$ neuron pairs. To simplify the notation, we write $(D_1, D_2) \equiv (D_{ee}, D_{ii})$.

Symbol definitions for the soma voltage terms are shown in Table 3.3.

Table 3.3: Symbol definition for the soma equations

Symbols	Description	Value	Unit
$\tau_{e,i}$	neuron time constant	0.04, 0.04	s
$V_{e,i}^{\text{rev}}$	reversal potential at dendrite	0, -70	mV
$V_{e,i}^{\text{rest}}$	neuron resting potential	-64, -64	mV
$\rho_{e,i}$	excitatory, inhibitory synaptic gain	1.00×10^{-3}, -1.05×10^{-3}	mV s
D_2	$i \to i$ gap-junction diffusive coupling strength	0–1.0	cm^2
D_1	$e \to e$ gap-junction diffusive coupling strength	$D_2/100$	cm^2

3.1.4 Summary

To summarise, the cortical model consists of three parts:

Cortico-cortical equations

$$\left[\left(\frac{\partial}{\partial t} + v\Lambda_{eb}\right)^2 - v^2\nabla^2\right]\phi_{eb}^{\alpha} = (v\Lambda_{eb})^2 Q_e, \qquad b = e, i$$

Intra-cortical equations

$$\left(\frac{\partial}{\partial t}+\gamma_e\right)^2\Phi_{eb}=[N_{eb}^{\alpha}\phi_{eb}^{\alpha}+N_{eb}^{\beta}Q_e+\phi_{eb}^{sc}]\gamma_e^2$$
$$\left(\frac{\partial}{\partial t}+\gamma_i\right)^2\Phi_{ib}=N_{ib}^{\beta}Q_i\gamma_i^2$$

Soma equations

$$\tau_b\frac{\partial V_b}{\partial t}=V_b^{\text{rest}}-V_b+(\rho_e\psi_{eb}\Phi_{eb}+\rho_i\psi_{ib}\Phi_{ib})+D_{bb}\nabla^2V_b$$

This model was first proposed in Physics Review E **2007** by **Steyn-Ross** *et al.* (see [118]). Hereafter, we refer to it as the SR2007 model of the cortex.

Later, by modifying the SR2007 model, Steyn-Ross *et al.* developed "slow-"and "fast-soma" limits to model brain activation patterns for the default and cognitive states respectively [120]. The cortical model for the "slow-soma" limit is basically the same as the SR2007 model, except that the dendrite response Eq. (3.6) is replaced by a biexponential function:

$$H_{ab}(t)=\frac{\alpha_{ab}\beta_{ab}}{\beta_{ab}-\alpha_{ab}}(e^{-\alpha_{ab}t}-e^{-\beta_{ab}t}) \tag{3.22}$$

Thus the intra-cortical equation for the "slow-soma" limit changes to:

$$\left(\frac{\partial}{\partial t}+\alpha_{ab}\right)\left(\frac{\partial}{\partial t}+\beta_{ab}\right)\Phi_{ab}(t)=\alpha_{ab}\beta_{ab}M_{ab}(t) \tag{3.23}$$

The other equations for the "slow-soma" model remain the same. For the "fast-soma" limit, we assumed that the soma voltage changes on a time-scale similar to that of dendritic integration, which is the fact that the input flux M is integrated at the dendrite, then modulated by the reversal function Eq. (3.12):

$$\Phi_{ab}=H_{ab}(t)\otimes[\psi_{ab}(t)\cdot M_{ab}(t)], \tag{3.24}$$

leading to two fast-soma differential equations for the soma voltage,

$$\tau_b\frac{\partial V_b}{\partial t}=V_b^{\text{rest}}-V_b+(\rho_e\Phi_{eb}+\rho_i\Phi_{ib})+D_{bb}\nabla^2V_b \tag{3.25}$$

For convenient reference, we refer to the slow-soma model as SR2009s and the fast-soma model as SR2009f (both reported in *NeuroImage* **2009** by **Steyn-Ross** *et al.*). The SR2009s and the previously introduced SR2007 models both describe the slow-soma limit. The subtle difference between them are detailed in Table 3.4.

Table 3.4: Referenced Waikato cortical models

	Slow-soma limit		Fast-soma limit
	SR2007 model [118]	SR2009s model [120]	SR2009f model [120]
Dendritic response H	alpha-function	biexponential function	biexponential function
Dendritic flux	$H^{\alpha,\mathrm{bi}} \otimes$ (input flux)		$H^{\mathrm{bi}} \otimes (\psi \cdot$ input flux)

The major difference between slow- and fast-soma models is the order of the convolution:

- The slow-soma limit allows a *slow* soma voltage feedback from soma to dendrite, thus the weighting factor ψ_{ab} is applied after the input flux M_{ab} has been integrated at the dendrite to give the dendritic flux Φ_{ab}:

$$\begin{aligned}\Phi_{ab} &= (\text{dendrite response}) \otimes (\text{input flux}) \\ &= H_b(t) \otimes M_{ab}(t)\end{aligned}$$

$$\tau_b \frac{\partial V_b}{\partial t} = V_b^{\mathrm{rest}} - V_b + (\rho_e \psi_{eb} \Phi_{eb} + \rho_i \psi_{ib} \Phi_{ib}) + D_{bb} \nabla^2 V_b$$

- The fast-soma limit allows *instantaneous* feedback from soma voltage onto dendrite reversal potential (see Eqs. (3.24, 3.25)), thus the weighting factor ψ is applied directly to the input flux M, with the product being integrated at the dendrite.

The comparison of slow- and fast-soma limits are detailed in Table C.1.

In the next section, we will show that this simple alteration of the order of convolution gives rise to dramatic divergence of cortical dynamics.

3.2 Cortical stability

3.2.1 Cortical stability of SR2007 model

Before exploring the full two-dimensional dynamics of the model, it is instructive to calculate the homogeneous steady states of the model. To locate the steady states for the cortical model, we set all temporal and spatial derivatives to zero, and remove the noise source ϕ_{eb}^{sc}. Then we compute numerically the steady-state membrane voltage V_e^o and firing rates Q_e^o of the excitatory neural population as a function of subcortical drive s.

The upper left panel of Fig. 3.5 shows the distribution of V_e^o equilibrium values as a function of subcortical drive s. Positions M and N, respectively locate the upper and lower turning points of the S-shape distribution, identifies the region of multiple steady-states, $0.2 \leq s \leq 0.3$, where the noiseless homogeneous cortical model can either reside at low-firing quiescent state (below N) or at high-firing activated state (above M).

By examining the effect of gap-junction diffusion on the cortical stability at different subcortical drives (plots A to G in Fig. 3.5), we find:

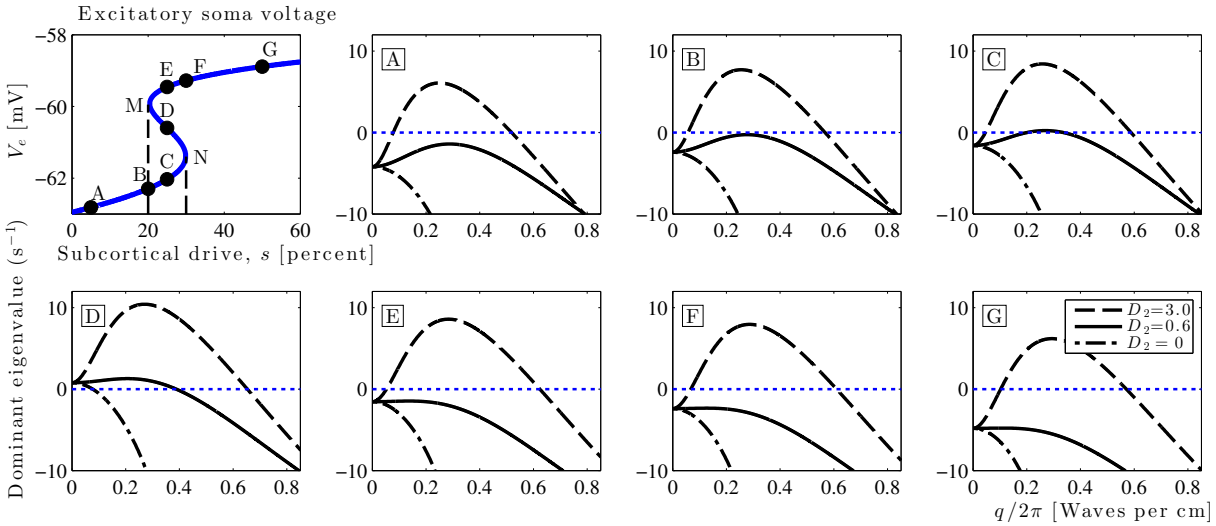


Figure 3.5: [Upper left] Distribution of equilibrium states for the homogeneous model cortex as a function of the subcortical drive. The distribution curve forms an S-shape where two turning points M and N demarcate a range ($0.2 \leq s \leq 0.3$) of multiple steady states belonging to three branches: the high-firing activated state (above M, top branch); the low-firing quiescent state (below N, bottom branch) and the unstable middle branch (between M and N). Plots A to G are eigenvalue dispersion (real parts only) curves for representative points from the steady-state distribution: A ($s = 0.05$), B ($s = 0.2$) and C ($s = 0.25$) from the lower branch; D ($s = 0.25$) from the midbranch; E ($s = 0.25$), F ($s = 0.3$) and G ($s = 0.5$) from the top branch. At each steady state, we examined the effect of gap-junction strength [cm^2] on its stability: $(D_1, D_2) = (0, 0)$ (no diffusion, dash-dot curve); $(D_1, D_2) = (0.006, 0.6)$ (weak diffusion, solid cruve); $(D_1, D_2) = (0.03, 3)$ (strong diffusion, dashed curve). (Figure modified from [118].)

- The strong-, weak- and zero-diffusion curves converge at $q = 0$, i.e., the whole-of-cortex with "infinite wavelength". Inspecting plots A to C, the equilibria of the homogenous cortex is becoming less stable (real part α of the dominant eigenvalue is approaching zero at $q = 0$). At D, the midbranch, the homogenous cortex becomes unstable ($\alpha > 0$). When the equilibria is moving away from point M along the top branch (plots E, F and G), the homogenous cortex is becoming more and more stable (α is turning to be more negative at $q = 0$).
- Strong gap-junction diffusion ($D_2 = 3$) leads to an instability at a nonzero wavenumber on all branches. In contrast, closure of the gap-junction ($D_2 = 0$) predicts that the cortex is likely to be stable, and this stability becomes greater with increasing wave numbers.
- For the weak gap-junction diffusion ($D_2 = 0.6$) case, the stability of the cortical equilibria follows the same trend as the homogeneous cortex: The instability emerges in favour of a band of non-zero wavenumbers when the steady state gets closer to the lower turning point N, and dies away when the steady state moves away from the region of multiple steady states towards the high-firing branch. The midbranch equilibria D is unstable within a range of wavenumbers.

- Plots C and D indicate that increases in the inhibitory diffusion increase the range of spatial frequencies that can destabilise the homogeneous equilibrium.

From the above observations, we can conclude a general trend that for the SR2007 model, stronger inhibitory diffusion leads to greater instability. We also find that in the multiroot region ($0.2 \leq s \leq 0.3$), a Turing instability can be precipitated by rather smaller values of inhibitory diffusion. For example in our predictions, weak inhibitory diffusion $D_2 = 0.6$ is sufficient to trigger a Turing instability in the multiroot zone, but it is too weak to provoke an instability when $s < 0.2$ or $s > 0.3$.

3.2.2 Cortical stabilities of SR2009s and SR2009f models

In Fig. 3.6, we compare responses of the SR2009s (slow-soma limit) and SR2009f (fast-soma limits) models to inhibitory diffusion and subcortical excitation. The arrows labelled with "D_2 increasing" and "s increasing" indicate the fact that increasing inhibitory diffusion and subcortical drive provoke contrary cortical instabilities:

- For the SR2009s model, at a fixed subcortical drive, Fig. 3.6(a) shows that stronger inhibitory diffusion D_2 leads to greater Turing instability, while at a fixed inhibitory diffusion, Fig. 3.6(b) shows that greater subcortical drive s tends to restore the Turing instability to the homogeneous steady state.
- For the SR2009f model, at a fixed subcortical drive, Fig. 3.6(c) shows stronger inhibitory diffusion D_2 leads to weaker wave instability, while at a fixed inhibitory diffusion, Fig. 3.6(d) shows greater subcortical drive s tends to boost the wave instability.

We also note that the SR2009f model is about two orders of magnitude more sensitive to variations of inhibitory diffusion D_2 than is the SR2009s model. Comparing Fig. 3.6(a) and (c), we see that the SR2009s model reaches the maximum instability (growth rate $\sim 8\ \mathrm{s}^{-1}$) when $D_2 = 4\ \mathrm{cm}^2$; in contrast, the SR2009f model experiences the maximum instability when there is no inhibitory diffusion $D_2 = 0$.

In addition, the SR2009s model, modified by replacing the alpha-function in SR2007 model with the biexponential form, is not strictly identical in cortical stability with respect to subcortical drive. In Fig. 3.5, the Turing instability is enhanced with increasing subcortical drive until it reaches the midbranch root D, then the Turing instability is suppressed with future increases in subcortical drive. For the SR2009s and SR2009f models, we used another setting of cortical parameters proposed by Rennie *et al* [100] in order to compare performance of different models (full parameter settings for SR2009s and SR2009f models are listed in [120]). The setting of model parameters is considered to be the major factor for altered stabilities of cortical homogeneous steady-states.

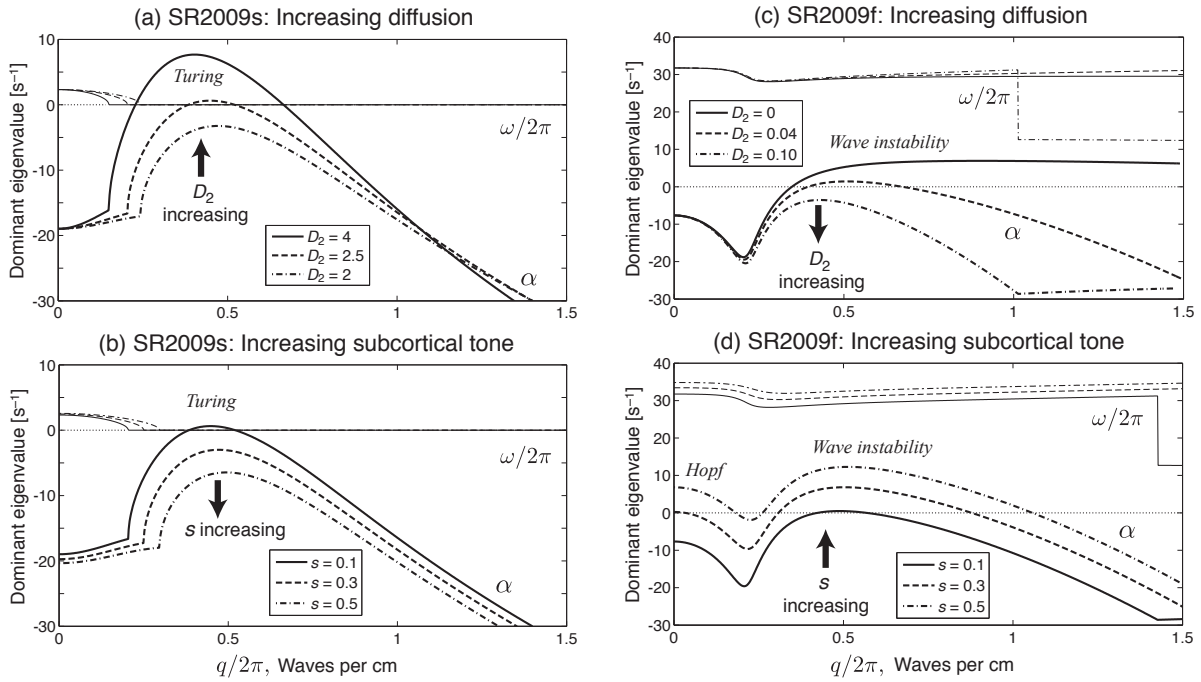


Figure 3.6: SR2009s (slow-soma limit) dispersion curve at the left column for (a) increasing inhibitory diffusion D_2 with fixed subcortical drive $s = 0.1$ and (b) increasing subcortical drive s with fixed excitatory and inhibitory diffusions $(D_1, D_2) = (0.025, 2.5)$ cm^2. SR2009f (fast-soma limit) dispersion curve at the right column for (c) increasing inhibitory diffusion D_2 with fixed subcortical drive $s = 0.1$ and (d) increasing subcortical tone s with fixed excitatory and inhibitory diffusions $(D_1, D_2) = (0.0005, 0.05)$ cm^2. The SR2009s model predicts a temporally stable Turing instability, and the SR2009f predicts a $\sim$30 Hz gamma wave instability. (Figure reproduced from [120].)

3.3 Cortical simulations

3.3.1 Cortical simulations of SR2007 model

In Sec. 3.2.1, we applied linear stability analysis (LSA) to the cortical steady-states (as a function of subcortical drive), and predicted that the Turing instability could be triggered with rather smaller inhibitory diffusion in the multiroot region ($0.2 \leq s \leq 0.3$, see Fig. 3.5) or a sufficiently large diffusion in the single-root region. In Table 3.5, we show LSA predictions at three selected points sampling different regions from the steady-state curve in Fig. 3.5.

In our cortical simulations, the substrate was a 100×100 square grid, of spatial resolution $\sim$0.25 cm/grid-point[3], joined at the edge to provide toroidal boundaries. We used a forward-time, centred-space Euler algorithm custom-written in MATLAB[4]. The basic coding principles are the same as for the Brusselator simulation in Sec. 2.2.1. We used

[3]The spatial resolution can be increased by increasing the grid density or decreasing the side length. Higher resolution provides more pattern details, but lowers the simulation speed. The resolution setting does not affect pattern dynamics.

[4]Simulation codes were written by A/Prof. Alistair Steyn-Ross. The complete codes, plus README files and movies of cortical dynamics, are available from the web site:

specific pairs of steady state and inhibitory diffusion D_2 (see Table 3.5) to test the LSA predictions. We find that simulated patterns in Fig. 3.7 are in good agreement with LSA predictions in Table 3.5. In Fig. 3.7, by scanning the first column, we can see that under the weak inhibitory diffusion, the midbranch homogeneous equilibrium D has evolved to a Turing pattern; in contrast, simulations commencing at homogeneous equilibria A or G show spatially unstructured patterns. Scanning from left to right on the first and third rows, the top-branch equilibrium G is found to be less sensitive to inhibitory diffusion with respect to emergence of Turing instability; for A, $D_2 = 1.2$ is sufficient to trigger Turing patterns; while for G, this value is too weak but should be increased to 1.3.

Table 3.5: Turing instability predictions for selected steady states (SS) in Fig. 3.5. $\times$: decayed Turing pattern; $\surd$: emerged Turing pattern

SS \ D_2	0.6	1.2	1.3
A	$\times$	$\surd$	$\surd$
D	$\surd$	$\surd$	$\surd$
G	$\times$	$\times$	$\surd$

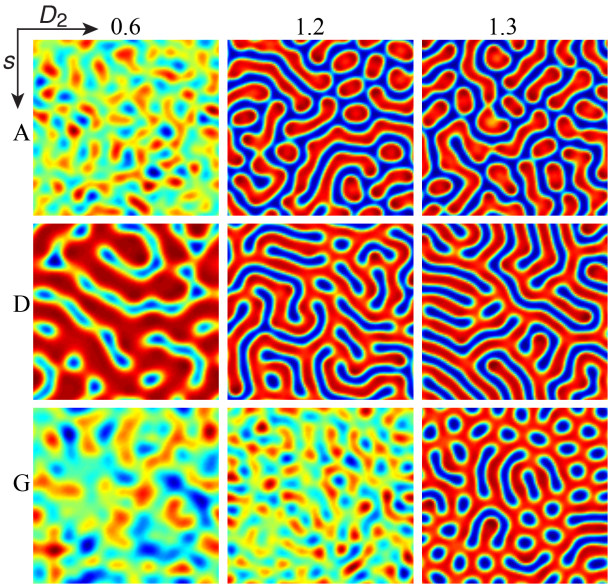


Figure 3.7: Cortical patterns of SR2007 model for three steady-states (vertical axis) and three values of inhibitory diffusion D_2 (horizontal axis). The subcortical drives for A, D and G are $s = 0.01, 0.25, 0.5$ respectively.

3.3.2 Cortical simulations of SR2009s and SR2009f models

Linear stability predictions for the SR2009s model in Figs. 3.6(a) and (b) reveal that increases in inhibitory diffusion and in subcortical drive act in contrary directions with

http://www2.phys.waikato.ac.nz/asr/
A SIMULINK representation of the cortical model is demonstrated in Appendix D.4

respect to maintaining or breaking the symmetry of the homogeneous cortical sheet. This counteracting tendency is illustrated in Fig. 3.8. A vertical scan from top to bottom shows that for a constant subcortical drive, strengthening the inhibitory diffusion leads to enhanced Turing patterns. In contrast, when the inhibitory diffusion is held constant (e.g., $D_2 = 3$), a horizontal scan from left to right shows that increasing the subcortical drive intends to wash out the Turing formation. However, when the inhibitory diffusion is sufficiently strong (e.g., $D_2 = 5$), the LSA predicted Turing instability more than compensates the pattern-suppression effect of large subcortical drive. Thus, in Fig. 3.8 we see that $D_2 > 3$ induced Turing formation has the resistance to washout from subcortical drive $s > 0.1$.

For the SR2009f model, Figs. 3.6(c) and (d) indicate reversed stability tendencies to that of the SR2009s model. Comparing the top-right corner of Figs. 3.8 and 3.9, the SR2009f model shows clear patterning at zero inhibitory diffusion and strong subcortical drive; but for the SR2009s case the strongest patterns are at the bottom-left corner (large inhibitory diffusion and weak subcortical drive). In Fig. 3.9, a top to bottom scan shows the pattern contrast will gradually blur and eventually disappear with moderate inhibitory diffusion. The reason can be found in Fig. 3.6(c): Increasing inhibitory diffusion D_2 will diminish the wave instability, thus there will be fewer unstable wavenumbers to support a spatial structure.

3.4 Cortical pattens at a codimension-2 Turing-Hopf point

Until now, we have reviewed cortical dynamics with respect to a single bifurcation: Turing bifurcation for the SR2007 and SR2009s models; wave instability for the SR2009f model. We have showed the crucial importance of the inhibitory diffusion D_2 on cortical stability. For SR2007 and SR2009s models, the value of D_2 is directly related to the occurrence of Turing patterns. Therefore, inhibitory diffusion D_2 is considered to be a bifurcation control parameter for the Turing instability.

The Turing spatial instability induced by strong inhibitory diffusion can be suppressed by closing the gap-junctions (i.e., by setting $D_2 = 0$). Alternatively, we can induce a low-frequency Hopf *temporal* instability with a reduction in γ_i (equivalent to rate-constants β_{ie} and β_{ii}), the rate-constant for the inhibitory post-synaptic potential. Hence, γ_i is determined to be a bifurcation control parameter for the Hopf instability.

Steyn-Ross *et al.* [117] had found that temporally stable Turing patterns could be made to oscillate in place at a low Hopf frequency, which is very similar to the Brusselator Turing-Hopf (TH) mode seen in Sec. 2.2.1. Such a cortical TH mode can be induced by

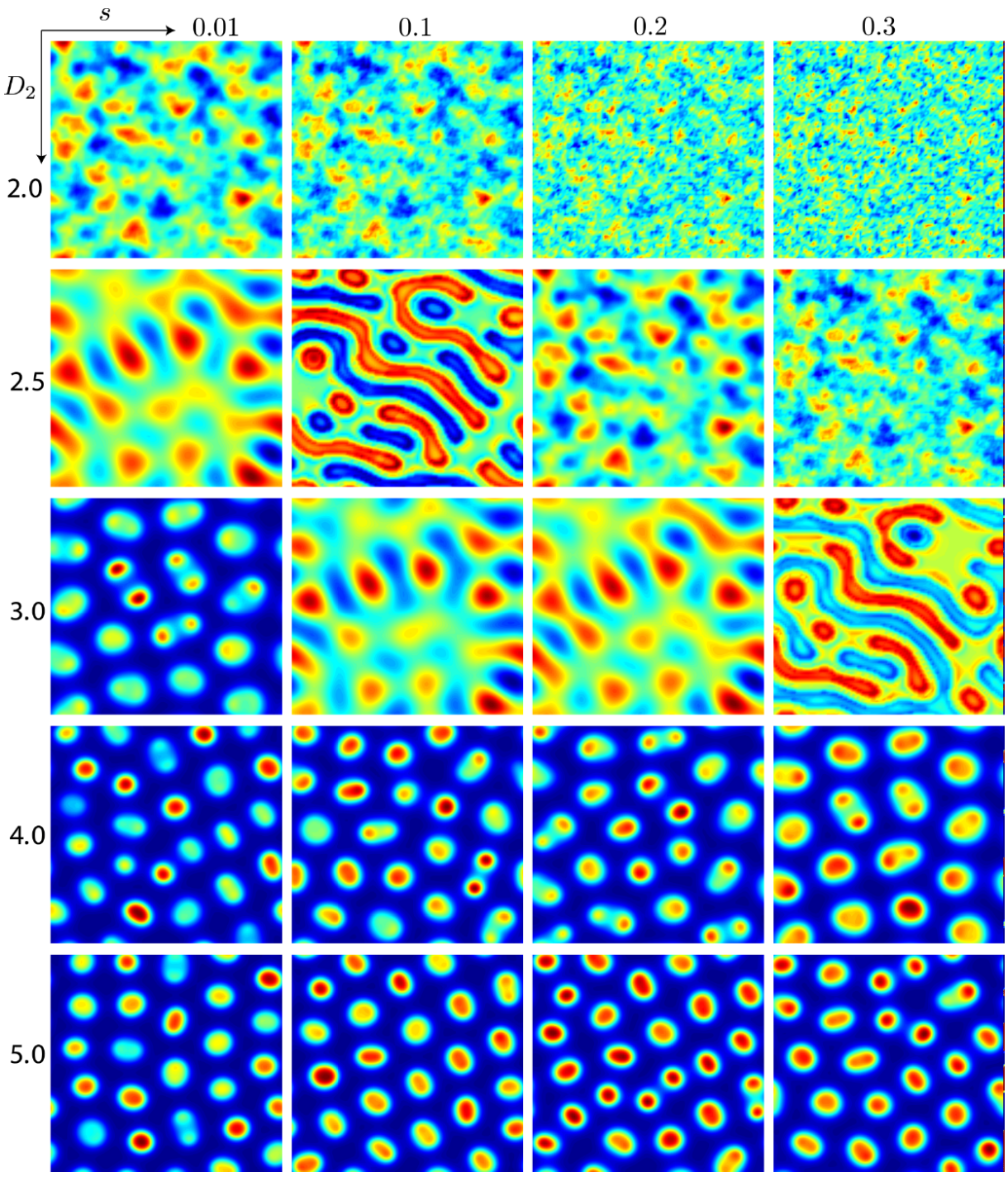


Figure 3.8: Patterns of **SR2009s** (**slow-soma** limit) model for five values of inhibitory diffusion D_2 (vertical axis) and four values of subcortical drive s (horizontal axis). Cortical sheet is initialised at the homogeneous steady state, stimulated by one-off spatiotemporal white noise, and iterated for 5 s. Colour indicates the activation of cortical tissue: [red] high-firing activated, [blue] low-firing suppressed. Cortical sheet has side length 10 cm containing 100×100 grid-points (spatial resolution = 1 mm/grid-point). (Figure reproduced from [111].)

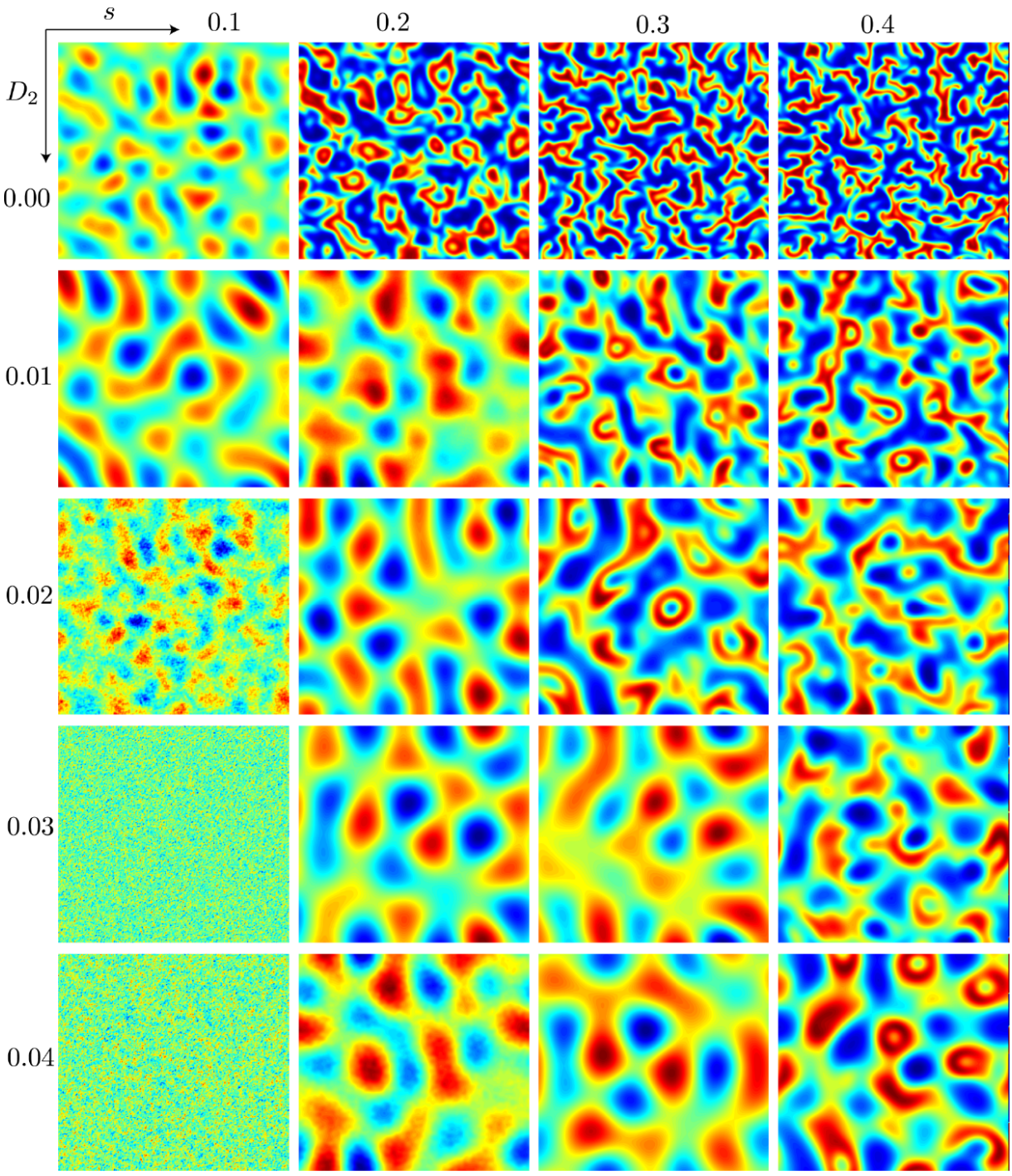


Figure 3.9: Patterns of **SR2009f** (**fast-soma** limit) for five values of inhibitory diffusion D_2 (vertical axis) and four values of subcortical drive s (horizontal axis). See Fig. 3.8 for simulation details. (Figure reproduced from [111].)

a coordinated tuning of both the Turing control parameter D_2 and the Hopf control parameter γ_i, to place the cortex at a CTHP.

Let us relate these ideas to Fig. 3.5, the S-bend homogeneous equilibriums of SR2007 model. We found for a subcortical drive $s = 0.2989$ that is located at the right edge of the multiroot region, a reduction of γ_i below a critical value $\sim$30.94 s^{-1} is sufficient to produce a complex dominant eigenvalue at zero wavenumber whose real part is positive;

thus suggesting a global Hopf oscillation. In Fig. 3.10(d), setting $\gamma_i = 29.45\ \mathrm{s}^{-1}$ predicts a ∼0.95 Hz Hopf oscillation. Independently, a Turing instability is boosted with moderately strong inhibitory diffusion $D_2 = 1\ \mathrm{cm}^2$ above its critical value 0.9066 cm^2. Similar to the Brusselator TH mode in Fig. 2.4, cortical TH interactions lead to unpredicted beating patterns revealed in the time series recorded in Fig. 3.10(b) for a single pixel on the cortical grid. The Fourier spectrum shows two frequency components whose difference matches with the ultra-slow envelope frequency, which is likely to be the weakly-damped resonance at ∼0.152 Hz. Clearly, the frequency-splitting and its associated ultra-slow oscillation arise from the the nonlinear nature of the TH interaction, since neither phenomenon is predicted by LSA.

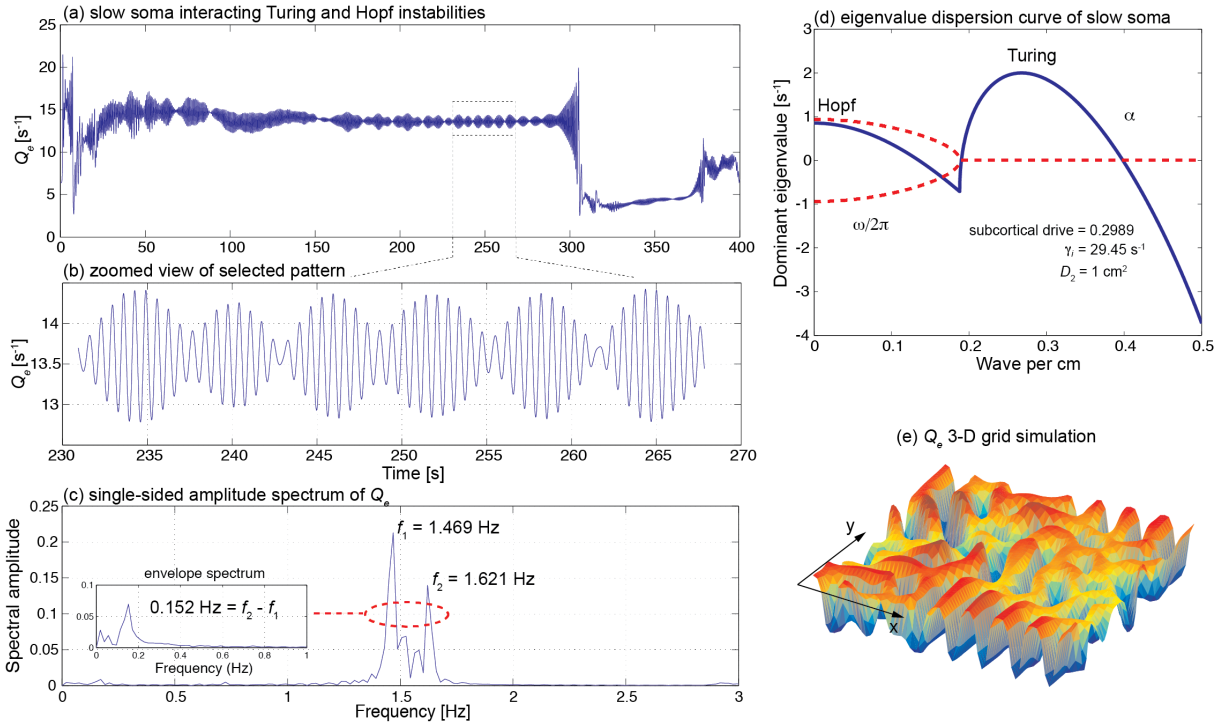


Figure 3.10: With strong gap-junction diffusion D_2 and carefully chosen inhibitory rate-constant γ_i, eigenvalue dispersion curve (d) predicts a mixed pattern of Turing and Hopf instabilities. α and ω are the real and imaginary part of the dominant eigenvalue respectively. Through a 400-s simulation, the Fourier spectrum (c) indicates a 0.15-Hz ultra-slow oscillation of the beating pattern (b) zoomed from (a) Q_e time evolution of the point at position (1, 30) shown in (e) 25-×25-cm grid 3-D plot. (Figure modified from [117].)

3.5 Discussion

In this chapter, we have discussed the gap-junction mediated mean-field model of the human cortex in a slow-soma limit (SR2007, SR2009s models). Subtle modifications to the temporal convolutions of the slow-soma model leads to a distinct fast-soma limit (SR2009f

model) of the cortex. We consider that the SR2007 and SR2009s models share the same feature of the slow-soma pattern dynamics: LSA and numerical simulations reveal that they exhibit spatial structured Turing and weak Hopf instabilities. In contrast, the fast-soma model exhibits a wave instability oscillating at a low gamma frequency. What are the biophysical implications for these patterns?

The slow-soma models suggest that the strong gap-junction diffusivity provides a natural mechanism for Turing bifurcation that leads to the spontaneous formation of Turing patterns of high and low neural activity that spread over the whole cortex, allowing multiple, spatially separated cortical regions to become activated simultaneously. It is possible that such spatial synchrony explains the cognition "binding" question proposed at the beginning of this chapter. However, cognition is thought to involve spatiotemporal cortical dynamics, thus the temporally stable Turing patterns may not match the cognitive cortical state.

At the vicinity of a Turing instability, a weak Hopf instability can be induced in parallel by prolonging the timing of delivery of inhibition at chemical synapses to destabilise the temporal stability of Turing patterns, permitting 1.5-Hz Hopf oscillations with the spatial structure maintained. Our numerical simulations revealed a beating pattern, unpredicted by LSA, with ultra-slow envelope frequency $\sim$0.152 Hz. We posit that the interacting low-frequency Hopf and Turing instabilities may form the substrate for the cognitive state, namely, the "default" background state for the non-cognitive brain. These slow patterned oscillations may relate to very slow ($\leq$0.1 Hz) fluctuations in BOLD (blood-oxygen-level dependent) signals detected using fMRI (functional magnetic resonance imaging) of relaxed, non-tasked human brains [30, 32]. This default state will be suppressed with elevated levels of subcortical drive during goal-directed tasks [16, 35, 38, 63, 99].

The work by Rodriguez *et al.* [103] and Varela *et al.* [130] demonstrated that a long-range synchrony associated with gamma-band activity was observed during cognition. In spite of a comprehensive scan in parameter space, we found no evidence that the slow-soma models can generate gamma-band activity. A reordering of the convolution in which the reversal potential weighting was applied on the dendritic integration before it comes into the soma body (instant feedback or "fast-soma" limit) allows the cortical model to support gamma oscillations, indicating that the frequency of cortically-generated rhythms is critically dependent on the nature and timeliness of the feedback from soma to dendrite. Our grid simulations show that these gamma oscillations are coherent over distance of several centimetres that provides a possible basis of "instantaneous" action-at-a-distance normally associated with cognition [106].

We see that elevated subcortical drive induces a transition from non-cognitive to cognitive brain state. The fast-soma model predicts that cortical cognitive patterns will be washed out with strong inhibitory diffusion D_2 by increasing gap-junction strength. So, what is the clinical relevance for the sensitivity of fast-soma patterns on the gap-junction

diffusion? Steyn-Ross *et al.* argue that unusual gap-junction strength in the brain might be relevant to the so-called "dopamine hypothesis" for schizophrenia disease. The neuromodulator dopamine, known as a potent gap-junction blocker [39], is found in excess in schizophrenia patients [128]. For the schizophrenic brain, coherent gamma activity is predicted to emerge by lowering inhibitory diffusion (equivalent to applying excess dopamine) in the fast-soma model. But these spindly fast-soma patterns (see upper-right panel of Fig. 3.9) are observed to be less spatially generalised than those observed in a normal brain with lower dopamine levels and therefore stronger inhibitory diffusion (e.g., bottom-right panel of Fig. 3.8). This prediction is consistent with diminished long-range synchrony seen in the EEG signals recorded from schizophrenics [129].

In the next chapter, we will introduce a multiple-scale expansion, a nonlinear perturbation method, to derive the pattern-forming system's amplitude equations for the analysis of its near-bifurcation behaviours, e.g., predicting the structures of Turing patterns and the envelope frequency of the cortical TH pattern.

Chapter 4

Derivation of the amplitude equation for reaction–diffusion systems via computer-aided multiple-scale expansion

The amplitude equation describes a reduced form of a reaction–diffusion system yet still retains its essential dynamical features. By approximating the analytic solution, the amplitude equation allows examination of mode instability when the system is near a bifurcation point. Multiple-scale expansion (MSE) offers a straightforward way to systematically derive the amplitude equations. The method expresses the single independent variable as an asymptotic power series consisting of newly introduced independent variables with differing time and space scales. The amplitude equations are then formulated under the solvability conditions which remove secular terms.

To our knowledge, there is little information in the research literature that explain how the exhaustive workflow of MSE is applied to a reaction–diffusion system. In this chapter, detailed mathematical operations underpinning the MSE are elucidated, and the practical ways of encoding these operations using MAPLE are discussed. A semi-automated MSE computer algorithm `Amp_solving` is presented for deriving the amplitude equations in this research. `Amp_solving` has been applied to the classical Brusselator model for the derivation of amplitude equations when the system is in the vicinity of a Turing codimension-1 and a Turing–Hopf codimension-2 bifurcation points. Full open-source `Amp_solving` codes for the derivation are comprehensively demonstrated and available to the public domain.

4.1 Introduction

The reaction–diffusion system is a mathematical representation of the interaction between two *morphogens* [127] competitively reacting in time and spreading in space, which could give rise to a symmetry-breaking transition bifurcating from a homogeneous to patterned state, either stationary in a spatial (Turing) structure or in a temporal (Hopf) oscillation

[22]. These spatiotemporal instabilities are believed to be relevant to patterns widely observed in nature: animal coats (e.g., zebra stripes, leopard spots and fish spirals), chemicals in a gel [44], laser light in a cavity [9], charges on the surface of a semiconductor [7], ecological balance between two species [36] and even neuronal activations in human cortex [121].

The remarkably similar pattern formation observed in such diverse systems can often be understood as a consequence of a universal form of the amplitude equation, a near-bifurcation description highlighting the dynamics of the most unstable mode that is relevant to the emergent pattern. The analysis of the amplitude equation allows us to determine the stability and selection of the spatial pattern arising beyond the bifurcation point, thus enabling precise control of the pattern in its spatial texture and temporal evolution [34, 92, 93].

The derivation of the amplitude equation is a procedure that separates pattern modes into their fast- and slowly-evolving components. Approaching an instability (before a bifurcation occurs), the real part of the dominant eigenvalue[1] for a system tends to vanish, resulting in *slowed* recovery to perturbations (*critical slowing down*) [12]. Such slow modulations of the system can be extracted via a perturbation expansion, over multiple temporal and spatial scales, in a small parameter ϵ characterising the distance to the threshold (bifurcation point). This perturbation method is referred to as *multiple-scale expansion* (MSE), by which the coefficients for the amplitude equation are obtained based on the solvability condition of the resulting linear differential equations at different orders of ϵ (see e.g. [137]).

Over the last decade, MSE has been extensively used to derive the amplitude equation for a wide range of systems: nonlinear wave propagation [159]; planar beam model [69]; double pendulum system [156]; van der Pol–Duffing oscillator [74]; Klein–Gordon model [91]; Swift–Hohenberg model [59]; and especially chemical reaction–diffusion systems [22]. However, to our knowledge few papers give precise details on the derivation of amplitude equations. For example, the pattern dynamics of the well-known Brusselator model are discussed in the bifurcation literatures [24, 92, 154] with the aid of the amplitude equation, but with few clues as to its derivation. Although the authors of these works referred other literatures (e.g., [37, 56, 62]) for the amplitude equation derivation, the sophisticated mathematical language used in these references limits MSE uptake by non-mathematicians.

Besides, despite that fact that the MSE strategy is well formulated, its manipulation involves tedious analytical calculations. That is, the amplitude equation derivation via a MSE is difficult, involving a long sequence of computations by hand. This provides the motivation to convert the hand work to a systematic procedure with machine assistance.

[1]The eigenvalue of the Jacobian matrix that has the largest real part.

A few attempts for MSE programming have been made in the last decades: Pismen *et al.* [97] outlined a computer software written in MATHEMATICA for computer-assisted derivation of the amplitude equation valid in the vicinity of a bifurcation point. Later, Pismen *et al.* showed a more sophisticated illustration of their MATHEMATICA software [96]. However, the proposed software is not available to the public. Yu has published a series of works on the analysis of a double Hopf bifurcation using MSE in the MAPLE programming environment [156–158]. Although the source code of Yu's work is available [158], the lack of explanation and its specialised application (double Hopf bifurcation) restrict its extension to other bifurcation cases. Khanin *et al.* developed another MATHEMATICA package with the implementation of MSE on generalising amplitude equations for a wide range of bifurcations [55]. To our knowledge, the software by Khanin *et al.* may be the most promising work regarding bifurcation analysis. Unfortunately the MATHEMATICA codes are not given in their paper.

The difficulty of locating suitable software for the amplitude equation derivation was the initial motivation for developing our own software. Furthermore, there are limited resources describing the full implementation of MSE and necessary mathematical details for fully understanding the derivation of the amplitude equation. Consequently, in this chapter we aim to detail our customised (semi-automated) MSE algorithm `Amp_solving` encoded using MAPLE. To our knowledge, it is the first time that the application of MSE on reaction–diffusion systems has been carefully explained, systematically programmed, and released into the public domain.

The chapter is organised as follows. We start, for pedagogical reasons, with a brief demonstration of the amplitude equation derivation for the well-known Brusselator model at a Turing bifurcation point (Sec. 4.2). The necessary mathematical operations of the MSE is outlined for deriving the amplitude equation for a stripe mode, the basic element of the hexagonal Turing patterns.

In Sec. 4.3, we extract the basic MSE ideas and expand them for the amplitude equation derivation for a general reaction–diffusion system at a specific bifurcation point (either Turing, Hopf or mixed Turing–Hopf).

In Sec. 4.4, we briefly introduce our customised MSE algorithm `Amp_solving`. We elucidate the algorithm logic and core mechanisms for automating the MSE manipulations. A summary of this chapter is made in Sec. 4.5.

We include tutorial guidance and example MAPLE codes in the three appendices. Appendix E provides a short MAPLE tutorial manifesting our coding philosophies in `Amp_solving`. Appendixes F and G deliver comprehensively explained `Amp_solving` codes for the derivation of the Brusselator amplitude equations, respectively, for a full hexagonal Turing and a Turing–Hopf mixed modes. These examples give step-by-step guidance toward realisation of efficient MSE calculation in the MAPLE environment. Appendix

H supplies a quickcheck table summarising mathematical symbols and their expressions used in `Amp_solving`.

4.2 Derivation of Brusselator amplitude equation for the stripes mode

The Brusselator model is represented by a set of coupled differential equations describing the spatially and temporally varying chemical concentrations X (the activator) and Y (the inhibitor) [92]:

$$\begin{aligned}\frac{\partial X}{\partial t} &= A - (B+1)X + X^2Y + D_X\nabla_r^2 X \\ \frac{\partial Y}{\partial t} &= BX - X^2Y + D_Y\nabla_r^2 Y\end{aligned} \tag{4.1}$$

where $D_{X,Y}$ are the diffusion constants setting the pace of diffusion for chemicals X and Y respectively. A and B are constants. t is time and r is space. Following Nicolis and Prigogine's and other researchers' work [24,82,93,94,151], we choose B as the bifurcation control parameter.

Close to the Turing bifurcation point $B \equiv B_{\mathrm{c}}^{\mathrm{T}} = (1 + A\eta)^2$ (where $\eta = \sqrt{D_X/D_Y}$), the model variables $\mathbf{u} = [X, Y]^{\mathrm{Transpose}}$ can be expressed in a nonlinear expansion $\boldsymbol{\delta}\mathbf{u}$ in the deviation from the homogeneous steady-state $\mathbf{u_0} = [X_0, Y_0]^{\mathrm{Transpose}}$:

$$\mathbf{u} = \mathbf{u_0} + \underbrace{\epsilon\mathbf{u_1} + \epsilon^2\mathbf{u_2} + \epsilon^3\mathbf{u_3} + \ldots}_{\boldsymbol{\delta}\mathbf{u}} = \sum_{m=0}^{k} \epsilon^m \mathbf{u}_m \tag{4.2}$$

which is equivalent to

$$\begin{bmatrix} X \\ Y \end{bmatrix} = \begin{bmatrix} X_0 \\ Y_0 \end{bmatrix} + \epsilon \begin{bmatrix} x_1 \\ y_1 \end{bmatrix} + \epsilon^2 \begin{bmatrix} x_2 \\ y_2 \end{bmatrix} + \epsilon^3 \begin{bmatrix} x_3 \\ y_3 \end{bmatrix} = \begin{bmatrix} X_0 \\ Y_0 \end{bmatrix} + \sum_{p=1}^{k} \epsilon^p \begin{bmatrix} x_p \\ y_p \end{bmatrix} \tag{4.3}$$

The original model variables X and Y depend on (t, r), while the expanded model variables depend on the multiply scaled temporal and spatial arguments, e.g.,

$$\begin{aligned}\mathbf{u_1} &\equiv \mathbf{u_1}(T_0, T_1, T_2, \ldots T_k; R_0, R_1, R_2, \ldots, R_k), \\ \mathbf{u_2} &\equiv \mathbf{u_2}(T_0, T_1, T_2, \ldots T_k; R_0, R_1, R_2, \ldots, R_k), \\ \mathbf{u_3} &\equiv \mathbf{u_3}(T_0, T_1, T_2, \ldots T_k; R_0, R_1, R_2, \ldots, R_k)\end{aligned}$$

in which $T_k = \epsilon^k t$ and $R_k = \epsilon^k r$.

The small parameter ϵ relates the distance to the instability via the expansion:

$$B = B_0 + \epsilon B_1 + \epsilon^2 B_2 + \epsilon^3 B_3 + \ldots = \sum_{m=0}^{k} \epsilon^m B_m, \quad B_0 = B_{\mathrm{c}}^{\mathrm{T}} \tag{4.4}$$

According to the chain rule of the total derivative, the temporal derivative on expanded variables has the expression:

$$\begin{aligned}
\frac{\partial}{\partial t}\boldsymbol{\delta}\mathbf{u} &= \epsilon\left[\frac{\partial \mathbf{u_1}}{\partial T_0}\frac{dT_0}{dt} + \frac{\partial \mathbf{u_1}}{\partial T_1}\frac{dT_1}{dt} + \frac{\partial \mathbf{u_1}}{\partial T_2}\frac{dT_2}{dt} + \ldots + \frac{\partial \mathbf{u_1}}{\partial T_k}\frac{dT_k}{dt}\right] + \\
&\quad \epsilon^2\left[\frac{\partial \mathbf{u_2}}{\partial T_0}\frac{dT_0}{dt} + \frac{\partial \mathbf{u_2}}{\partial T_1}\frac{dT_1}{dt} + \frac{\partial \mathbf{u_2}}{\partial T_2}\frac{dT_2}{dt} + \ldots + \frac{\partial \mathbf{u_2}}{\partial T_k}\frac{dT_k}{dt}\right] + \\
&\quad \epsilon^3\left[\frac{\partial \mathbf{u_3}}{\partial T_0}\frac{dT_0}{dt} + \frac{\partial \mathbf{u_3}}{\partial T_1}\frac{dT_1}{dt} + \frac{\partial \mathbf{u_3}}{\partial T_2}\frac{dT_2}{dt} + \ldots + \frac{\partial \mathbf{u_3}}{\partial T_k}\frac{dT_k}{dt}\right] + \ldots \\
&= \epsilon\left[\frac{\partial \mathbf{u_1}}{\partial T_0} + \epsilon\frac{\partial \mathbf{u_1}}{\partial T_1} + \epsilon^2\frac{\partial \mathbf{u_1}}{\partial T_2} + \ldots + \epsilon^k\frac{\partial \mathbf{u_1}}{\partial T_k}\right] + \epsilon^2\left[\frac{\partial \mathbf{u_2}}{\partial T_0} + \epsilon\frac{\partial \mathbf{u_2}}{\partial T_1} + \epsilon^2\frac{\partial \mathbf{u_2}}{\partial T_2} + \ldots + \epsilon^k\frac{\partial \mathbf{u_2}}{\partial T_k}\right] \\
&\quad + \epsilon^3\left[\frac{\partial \mathbf{u_3}}{\partial T_0} + \epsilon\frac{\partial \mathbf{u_3}}{\partial T_1} + \epsilon^2\frac{\partial \mathbf{u_3}}{\partial T_2} + \ldots + \epsilon^k\frac{\partial \mathbf{u_3}}{\partial T_k}\right] + \ldots \\
&= \epsilon\frac{\partial \mathbf{u_1}}{\partial T_0} + \epsilon^2\left(\frac{\partial \mathbf{u_1}}{\partial T_1} + \frac{\partial \mathbf{u_2}}{\partial T_0}\right) + \epsilon^3\left(\frac{\partial \mathbf{u_1}}{\partial T_2} + \frac{\partial \mathbf{u_2}}{\partial T_1} + \frac{\partial \mathbf{u_3}}{\partial T_0}\right) + O(\epsilon^4)
\end{aligned} \tag{4.5}$$

which can be simplified to the expression:

$$\frac{\partial}{\partial t}\left(\sum_{p=1}^{k}\epsilon^p u_p\right) = \sum_{p=1}^{k}\left(\epsilon^p \sum_{m=0}^{k}\frac{\partial u_p}{\partial T_m}\frac{dT_m}{dt}\right) \tag{4.6}$$

One may write the temporal derivative operator $\partial/\partial t$ as:

$$\frac{\partial}{\partial t} = \frac{\partial}{\partial T_0} + \epsilon\frac{\partial}{\partial T_1} + \epsilon^2\frac{\partial}{\partial T_2} + \ldots = \sum_{m=0}^{k}\epsilon^m\frac{\partial}{\partial T_m} \tag{4.7}$$

Similarly, the Laplacian operation may be expanded in the same manner:

$$\nabla = \sum_{m=0}^{k}\epsilon^m\nabla_{R_m} \tag{4.8}$$

By introducing the above expansions into the original model equations, one can collect terms with the same order of ϵ to construct a series of order equations.

The zeroth-order terms recover the uniform reference state

$$\mathbf{u_0} = \begin{bmatrix} X_0 \\ Y_0 \end{bmatrix} \tag{4.9}$$

The order-1 equation recovers the linear stability analysis:

$$\left(\frac{\partial}{\partial T_0} - L_\mathrm{c}\right)\mathbf{u_1} = 0 \tag{4.10}$$

where L_c is the Jacobian matrix of the Brusselator model:

$$L_\mathrm{c} = \begin{bmatrix} (B_0 - 1) - D_X q^2 & A^2 \\ -B & -A^2 - D_Y q^2 \end{bmatrix} \tag{4.11}$$

Eq. (4.10) has a first-guess solution of a standing-wave (structure ansatz of the stripes mode) related to the Turing pattern:

$$\mathbf{u_1} = A_{\mathrm{T}} e^{i\vec{q_c}\cdot\vec{r}}\mathbf{R_T} + c.c. \tag{4.12}$$

where A_{T} describes slow modulations of the Turing pattern with respect to slow space and time arguments, in short we call it Turing amplitude; q_{c} is the critical wavenumber associated with the zero eigenvalue (where a Turing bifurcation occurs). $\mathbf{R_T}$ is the critical right eigenvector corresponding to the zero dominant eigenvalue of L_{c}. *c.c.* stands for the complex conjugate pairs.

By collecting terms of order ϵ^2, one obtains the order-2 equation:

$$\left(\frac{\partial}{\partial T_0} - L_{\mathrm{c}}\right)\mathbf{u_2} = \mathbf{I_2} \tag{4.13}$$

where

$$\begin{aligned}\mathbf{I_2} = -\frac{\partial}{\partial T_1}\begin{bmatrix}x_1\\ y_1\end{bmatrix} + \begin{bmatrix}B_1 + 2D_X\nabla_{R_0}\nabla_{R_1} & 0\\ -B_1 & -2D_Y\nabla_{R_0}\nabla_{R_1}\end{bmatrix}\begin{bmatrix}x_1\\ y_1\end{bmatrix}\\ +\left(\frac{B_0}{A}x_1^2 + 2Ax_1y_1\right)\begin{bmatrix}1\\ -1\end{bmatrix}\end{aligned} \tag{4.14}$$

To guarantee a solution of Eq. (4.13), the solvability condition (Fredholm alternative [40]) states that a matrix equation $\mathcal{L}\mathbf{v} = \mathbf{b}$ has a nontrivial solution if and only if the inner product $\langle\mathbf{w}|\mathbf{b}\rangle = 0$, where $\mathbf{w}$ is the left nullspace (in row vectors) of $\mathcal{L}$, i.e., $\mathbf{w}\mathcal{L} = 0$. If $\mathbf{w}$ is complex, the inner product can be expressed as:

$$\langle\mathbf{w}|\mathbf{b}\rangle = \frac{1}{V}\int_V \mathbf{w}^*\mathbf{b}\; d\mu \tag{4.15}$$

where $\mathbf{w}^*$ is the complex conjugate of $\mathbf{w}$; V is the space of the function $\mathcal{L}$; and μ is the space variable.

Thus we have the orthogonal condition between the left nullspace of $(\partial/\partial T_0 - L_{\mathrm{c}})$ and $\mathbf{I_2}$:

$$\langle\mathbf{v_T}|\mathbf{I_2}\rangle = 0 \tag{4.16}$$

where $\mathbf{v_T} = \mathbf{L_T}e^{i\vec{q_c}\cdot\vec{r}}$. $\mathbf{L_T}$ is the critical left eigenvector corresponding to the zero dominant eigenvalue of L_{c}. Analogous to Eq. (4.15), the inner product Eq. (4.16) in the $2\pi/q_{\mathrm{c}}$-space is:

$$\langle\mathbf{v_T}|\mathbf{I_2}\rangle = \frac{q_{\mathrm{c}}}{2\pi}\int_0^{\frac{2\pi}{q_{\mathrm{c}}}} \mathbf{v_T}^* \;\cdot\; \mathbf{I_2}\; dr = 0 \tag{4.17}$$

which yields the following constraints:

$$B_1 = 0 \tag{4.18a}$$

$$\frac{\partial}{\partial T_1}A_{\mathrm{T}} = \frac{\partial}{\partial T_1}A_{\mathrm{T}}^* = 0 \tag{4.18b}$$

Eq. (4.18b) is the order-2 amplitude equation and indicates that the Turing amplitude does not depend on the time scale T_1. So we should carry on to third order to examine the dependence of the Turing amplitude on a slower time scale T_2. Before doing this, we will need to solve the order-2 equation.

By substituting constraints Eq. (4.18a) and (4.18b), and the structure ansatz Eq. (4.12) into the order-2 equation (4.13), one is able to determine the order-2 solution $\mathbf{u_2}$:

$$\mathbf{u_2} = \begin{bmatrix} x_2 \\ y_2 \end{bmatrix} = \begin{bmatrix} c_{11} \\ c_{21} \end{bmatrix} e^{i\vec{q_c}\cdot\vec{r}} + \begin{bmatrix} c_{12} \\ c_{22} \end{bmatrix} e^{2i\vec{q_c}\cdot\vec{r}} + \begin{bmatrix} c_{13} \\ c_{23} \end{bmatrix} + c.c. \tag{4.19}$$

with

$$\begin{aligned} c_{11} + \frac{A}{\eta(1+A\ \eta)} c_{21} &= -\frac{2iq_c\sqrt{D_X D_Y}}{A(1+A\ \eta)}\ \nabla_{R_1} A_T \\ c_{12} &= \frac{4(1-A^2\eta^2)}{9A^2\eta} A_T^2 \\ c_{22} &= -\frac{(1-A^2\eta^2)(1+4A\ \eta)}{A^3} A_T^2 \\ c_{13} &= 0 \\ c_{23} &= -\frac{2(1-A^2\eta^2)}{A^3} |A_T|^2 \end{aligned} \tag{4.20}$$

Analogous to the order-2 equation, the order-3 equation has the form:

$$\left(\frac{\partial}{\partial T_0} - L_c\right)\mathbf{u_3} = \mathbf{I_3} \tag{4.21}$$

where

$$\begin{aligned} \mathbf{I_3} = &- \frac{\partial}{\partial T_1}\begin{bmatrix} x_1 \\ y_1 \end{bmatrix} - \frac{\partial}{\partial T_2}\begin{bmatrix} x_2 \\ y_2 \end{bmatrix} + \begin{bmatrix} B_2 + D_X\nabla_{R_1}^2 & 0 \\ -B_2 & D_Y\nabla_{R_1}^2 \end{bmatrix}\begin{bmatrix} x_1 \\ y_1 \end{bmatrix} \\ &+ \begin{bmatrix} 2D_X\nabla_{R_0}\nabla_{R_1} & 0 \\ 0 & 2D_Y\nabla_{R_0}\nabla_{R_1} \end{bmatrix}\begin{bmatrix} x_2 \\ y_2 \end{bmatrix} \\ &+ \left(2\frac{B_0}{A}x_1x_2 + 2A(x_1y_2 + x_2y_1) + x_1^2y_1\right)\begin{bmatrix} 1 \\ -1 \end{bmatrix} \end{aligned} \tag{4.22}$$

Finally, the solvability condition of the order-3 equation

$$\langle \mathbf{v_T} | \mathbf{I_3} \rangle = 0 \tag{4.23}$$

yields the following amplitude equation for the stripes mode:

$$\frac{\partial}{\partial T_0} A_T = \mu A_T - g|A_T|^2 A_T + D_{Ts}\nabla_{R_0} A_T \tag{4.24}$$

with

$$\mu = \frac{1+A\eta}{1-\eta^2}(\frac{B-B_0}{B_0}),\ g = \frac{-8A^3\eta^3+5A^2\eta^2+38A\eta-8}{9A^3\eta(1-\eta^2)},\ D_{\mathrm{Ts}} = \frac{4D_X(1+A\eta)}{B_0(1-\eta^2)} \quad (4.25)$$

Eq. (4.24) describes the slow growth and nonlinear saturation of the amplitude of the most unstable mode (at the wavenumber q_c) near bifurcation threshold. The form of Eq. (4.24) is generic for all reaction–diffusion systems undergoing a spatial symmetrical breaking, leading to a stripes mode. This equation governs the spatiotemporal evolution of the stripes amplitude after a Turing instability.

In the next section, we will describe how the MSE procedures can be applied to a general case.

4.3 Concepts underpinning multiple-scale expansion

In the previous section, we demonstrated application of MSE to derive the Brusselator amplitude equation for the stripes mode, the simplest Turing pattern. The MSE approach for this simple case can be generalised to a multiple dimensional reaction–diffusion system at a more complicated bifurcation point as follows.

4.3.1 Structure of order equations

The order equations are basic elements of the MSE. After expanding variables and differential operators in a power series of ϵ, successive families of order equations are generated by collecting coefficients with the same power of ϵ.

- Order-1 equation
$$\left(\frac{\partial}{\partial T_0} - L_{\mathrm{c}}\right)\mathbf{u_1} = \mathbf{I_1} = 0 \quad (4.26)$$
where L_{c} is the Jacobian matrix of the original system at the critical point.
- Order-2 equation
$$\left(\frac{\partial}{\partial T_0} - L_{\mathrm{c}}\right)\mathbf{u_2} = \mathbf{I_2} = -\frac{\partial}{\partial T_1}\mathbf{u_1} + \mathbf{F_2} \quad (4.27)$$
$\mathbf{F_2}$ is the nonlinear combination of $\mathbf{u_0}$ and $\mathbf{u_1}$ (e.g., Eq. (4.14)).
- Order-3 equation
$$\left(\frac{\partial}{\partial T_0} - L_{\mathrm{c}}\right)\mathbf{u_3} = \mathbf{I_3} = -\frac{\partial}{\partial T_1}\mathbf{u_1} - \frac{\partial}{\partial T_2}\mathbf{u_2} + \mathbf{F_3} \quad (4.28)$$
$\mathbf{F_3}$ is the nonlinear combination of $\mathbf{u_1}$ and $\mathbf{u_2}$ (e.g., Eq. (4.22)).

The principle for constructing order equations is intuitive: In an order-k equation, order-k variables are at the LHS, lower orders are sorted at the RHS, e.g., $\mathbf{F_k}$ is the nonlinear combination of expanded variables up to order $(k-1)$:

$$\left(\frac{\partial}{\partial T_0} - L_{\mathrm{c}}\right)\mathbf{u_k} = \mathbf{I_k} = -\frac{\partial}{\partial T_1}\mathbf{u_1} - \frac{\partial}{\partial T_2}\mathbf{u_2} - \ldots - \frac{\partial}{\partial T_{(k-1)}}\mathbf{u_{(k-1)}} + \mathbf{F_k} \quad (4.29)$$

4.3.2 Structure ansatz

The structure ansatz describes the pattern mode of interest. In Eq. (4.12), we have shown an ansatz for the Turing stripes mode. Other commonly used structure ansatzes are:

- Hopf mode

$$\mathbf{u_1}^{\mathrm{H}} = A_{\mathrm{H}} e^{i\omega_c t}\mathbf{R_H} + c.c. \tag{4.30}$$

  in which A_{H} is the Hopf mode amplitude for a temporal oscillation at frequency ω_c; $\mathbf{R_H}$ is the critical right eigenvector corresponding to the critical dominant eigenvalue that is purely imaginary, namely, $\lambda_c = i\omega_c$.

- Hexagonal mode of the Turing pattern

$$\mathbf{u_1}^{\mathrm{Hex}} = A_{\mathrm{T}1} e^{i\vec{q_1}\cdot\vec{r}}\mathbf{R_T} + A_{\mathrm{T}2} e^{i\vec{q_2}\cdot\vec{r}}\mathbf{R_T} + A_{\mathrm{T}3} e^{i\vec{q_3}\cdot\vec{r}}\mathbf{R_T} + c.c. \tag{4.31}$$

  $\vec{q_1}$, $\vec{q_2}$ and $\vec{q_3}$ are three equal-length (magnitude is identical to q_c in Eq. (4.12)) wave vectors at 120° to each other (see Fig. 4.1):

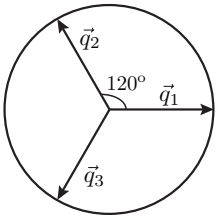


Figure 4.1: Superposition of three wave vectors at an angle of 120 degree with each other to form a hexagonal pattern.

$$\vec{q_1} + \vec{q_2} = -\vec{q_3}, \quad \vec{q_1} + \vec{q_3} = -\vec{q_2}, \quad \vec{q_2} + \vec{q_3} = -\vec{q_1} \tag{4.32a}$$

$$\vec{q_1}\cdot\vec{q_2} = \vec{q_1}\cdot\vec{q_3} = \vec{q_2}\cdot\vec{q_3} = q_c^2\cos(\theta), \qquad \theta = 120^\circ \tag{4.32b}$$

  $A_{\mathrm{T}1}$, $A_{\mathrm{T}2}$ and $A_{\mathrm{T}3}$ are complex mode amplitudes.

- Turing–Hopf mode

$$\mathbf{u_1}^{\mathrm{TH}} = A_{\mathrm{T}} e^{i\vec{q_c}\cdot\vec{r}}\mathbf{R_T} + A_{\mathrm{H}} e^{i\omega_c t}\mathbf{R_H} + c.c. \tag{4.33}$$

  Mixed Turing and Hopf modes appear when the system is above a codimension-2 Turing–Hopf point (CTHP) where the dominant eigenvalue becomes zero at a non-zero wavenumber and becomes purely imaginary at a zero wavenumber, thus allowing an oscillatory Turing mode at a low Hopf frequency. The frequency of the Turing–Hopf modulated pattern envelope is normally smaller than $\omega_c/2\pi$ Hz.

As we can see, the Turing–Hopf structure ansatz is a combination of Turing and Hopf modes. The hexagonal mode ansatz is a summation of three stripes modes oriented at 120° angular separations.

Examples of above mentioned modes are illustrated in Fig. 4.2

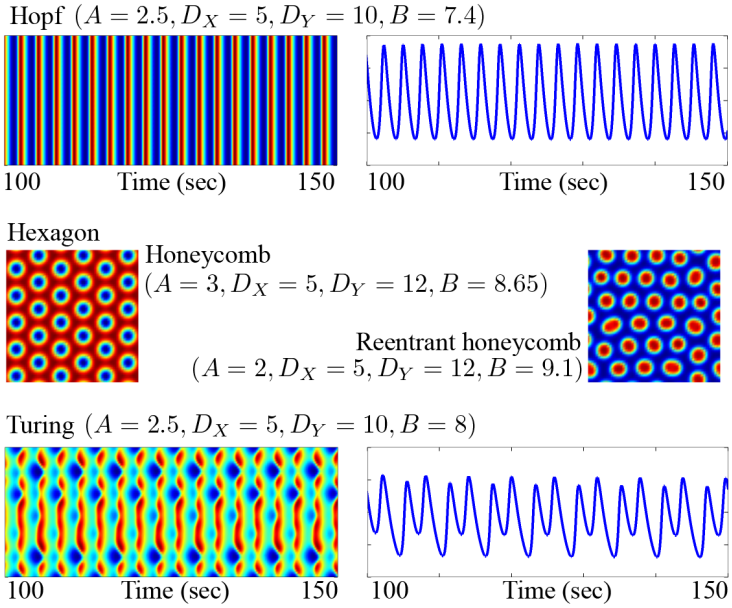


Figure 4.2: Simulations of the Brusselator model to demonstrate its pattern dynamics of Hopf (upper panel), hexagon (mid panel) and Turing–Hopf (lower panel). The parameter settings for generating these patterns are shown in the figure. A 2D grid-simulation consisting of a 60×60 grid (spatial resolution = 1 cm/grid-point) is utilised to exhibit the pattern dynamics of the concentration of X, the reactor in Brusselator. The Hopf simulation demonstrates a time-space strip-chart (left) and time-series of the centre grid-point (right) for the homogeneous oscillations; the hexagonal simulation shows the 2D Turing pattern in honeycomb (left) and reentrant honeycomb (right) structures; the Turing–Hopf simulation reveals a time-space strip-chart (left) and time-series of the centre grid-point (right) for an oscillatory Turing pattern. Colour of the pattern indicates the local concentration of the reactant: [red] high concentration, [blue] low concentration.

4.3.3 Solvability condition

In Sec. 4.2, we demonstrated the importance of the solvability condition: The order-2 solvability condition Eq. (4.16) yields the constraints allowing the order-2 amplitude equation (i.e., Eq. (4.18b)); and the order-3 solvability condition Eq. (4.23) yields the desired order-3 amplitude equation.

For an equation in the form:

$$\mathcal{L}\mathbf{u_k} \equiv \left(\frac{\partial}{\partial T_0} - L_c\right)\mathbf{u_k} = \mathbf{I_k} \tag{4.34}$$

its solvability condition with respect to each mode is:

- Stripes mode of the Turing pattern

  Stripes mode requires an inner product in the $2\pi/q_c$ spatial space between the left null space of $\mathcal{L}$ and $\mathbf{I_k}$:

  $$\langle \mathbf{v_T} | \mathbf{I_k} \rangle_{\frac{2\pi}{q_c}} = 0 \tag{4.35}$$

  where $\mathbf{v_T} = e^{i\vec{q_c}\cdot\vec{r}}\,\mathbf{L_T}$. $\mathbf{L_T}$ is the critical left eigenvector associated with the zero dominant eigenvalue at a Turing bifurcation.

- Hopf mode
  For the Hopf mode, we require an inner product in the $2\pi/\omega_c$ temporal space between the left null space of $\mathcal{L}$ and $\mathbf{I_k}$:

$$\langle \mathbf{v_H} | \mathbf{I_k} \rangle_{\frac{2\pi}{\omega_c}} = 0 \tag{4.36}$$

  where $\mathbf{v_H} = e^{i\omega_c t}\ \mathbf{L_H}$. $\mathbf{L_H}$ is the critical left eigenvector associative with the critical dominant eigenvalue being purely imaginary at a Hopf bifurcation.
- Hexagonal mode of the Turing pattern
  The solvability condition of the hexagonal mode is the repetitive application of the condition (4.35) on the three oriented stripes modes:

$$\langle \mathbf{v_{T1}} | \mathbf{I_k} \rangle_{\frac{2\pi}{|\vec{q_1}|}} = \langle \mathbf{v_{T2}} | \mathbf{I_k} \rangle_{\frac{2\pi}{|\vec{q_2}|}} = \langle \mathbf{v_{T3}} | \mathbf{I_k} \rangle_{\frac{2\pi}{|\vec{q_3}|}} = 0 \tag{4.37}$$

  where

$$\mathbf{v_{T1}} = e^{i\vec{q_1}\cdot\vec{r}}\ \mathbf{L_T}, \quad \mathbf{v_{T2}} = e^{i\vec{q_2}\cdot\vec{r}}\ \mathbf{L_T}, \quad \mathbf{v_{T3}} = e^{i\vec{q_3}\cdot\vec{r}}\ \mathbf{L_T} \tag{4.38}$$

  The three inner products in Eq. (4.37) are symmetric with respect to cyclic permutation of their indices.
- Turing–Hopf mode
  The solvability condition for the TH mode consists of a pair with the first being the Turing component (i.e., Eq. (4.35), leading to Turing constraints) and the second being the Hopf component (i.e., Eq. (4.36), leading to Hopf constraints).

$$\langle \mathbf{v_T} | \mathbf{I_k} \rangle_{\frac{2\pi}{q_c}} = 0 \tag{4.39a}$$

$$\langle \mathbf{v_H} | \mathbf{I_k} \rangle_{\frac{2\pi}{\omega_c}} = 0 \tag{4.39b}$$

Practically, the solvability condition for a Turing mode is

$$\begin{aligned} \langle \mathbf{v} | \mathbf{I_k} \rangle_{\frac{2\pi}{q_c}} &= \frac{q_c}{2\pi} \int_0^{\frac{2\pi}{q_c}} \mathbf{L_T}^* e^{-i\vec{q_c}\cdot\vec{r}}\ \cdot\ \mathbf{I_k}\ dr \\ &= \frac{q_c}{2\pi} \int_0^{\frac{2\pi}{q_c}} \Big[\underbrace{(\cdots)}_{\text{secular terms}} + (\cdots) e^{i\vec{q_c}\cdot\vec{r}} + (\cdots) e^{i2\vec{q_c}\cdot\vec{r}} + \ldots + c.c. \Big]\ dr = 0 \end{aligned} \tag{4.40}$$

Provided

$$\int_0^{\frac{2\pi}{q_c}} e^{in\vec{q_c}\cdot\vec{r}}\ dr = 0,\ \{n \in Z | n \neq 0\} \tag{4.41}$$

the condition (4.40) yields the constraint $(\cdots) = 0$, in which $(\cdots)$ are called secular terms, to be eliminated.

Similarly, the solvability condition for a Hopf mode

$$\begin{aligned}\langle \mathbf{v}|\mathbf{I_k}\rangle_{\frac{2\pi}{\omega_c}} &= \frac{\omega_c}{2\pi}\int_0^{\frac{2\pi}{\omega_c}} \mathbf{L_H}^* e^{-i\omega_c t} \ \cdot \ \mathbf{I_k}\ dt \\ &= \frac{\omega_c}{2\pi}\int_0^{\frac{2\pi}{\omega_c}} \Big[\underbrace{(\cdots)}_{\text{secular terms}} + (\cdots)e^{i\omega_c t} + (\cdots)e^{i2\omega_c t} + \ldots + c.c. \Big]\ dt = 0\end{aligned} \tag{4.42}$$

leads to constraints $(\cdots) = 0$, provided

$$\int_0^{\frac{2\pi}{\omega_c}} e^{in\omega_c r}\ dt = 0,\ \{n \in Z | n \neq 0\} \tag{4.43}$$

The algorithm to locate the secular terms is described in Sec. 4.4.3.

4.3.4 Order-2 pattern matching and order-2 solution

In Sec. 4.2, before operating the order-3 solvability condition Eq. (4.23), we must already know the explicit form of $\mathbf{u_2}$ in $\mathbf{I_3}$ (i.e., Eq. (4.22)). In the following, we will introduce a pattern forming strategy to solve $\mathbf{u_2}$.

After substituting the structure ansatz into the order-2 equation, its RHS will become a linear combination of exponential functions. e.g., by substituting the structure ansatz Eq. (4.12) into the order-2 equation (4.13), its RHS becomes:

$$\begin{aligned}\mathbf{I_2} = &\begin{bmatrix} 2D_X i q_c \mathbf{R_T}^{(1)} \nabla_{R_1} A_T \\ -2D_Y i q_c \mathbf{R_T}^{(2)} \nabla_{R_1} A_T \end{bmatrix} e^{i\vec{q_c}\cdot\vec{r}} + \left[B_0/A\ (A_T \mathbf{R_T}^{(1)})^2 + 2A A_T^2 \mathbf{R_T}^{(1)} \mathbf{R_T}^{(2)} \right] \begin{bmatrix} 1 \\ -1 \end{bmatrix} e^{i2\vec{q_c}\cdot\vec{r}} \\ &+ \left[2B_0/A\ |A_T|^2 |\mathbf{R_T}^{(1)}|^2 + 2A|A_T|^2 \mathbf{R_T}^{(1)} \mathbf{R_T}^{(2)*} \right] \begin{bmatrix} 1 \\ -1 \end{bmatrix} + c.c.\end{aligned} \tag{4.44}$$

which can be read as a pattern[2] of exponential functions

$$\mathbf{I_2} = (\cdots)e^{i\vec{q_c}\cdot\vec{r}} + (\cdots)e^{i2\vec{q_c}\cdot\vec{r}} + (\cdots) + c.c. \tag{4.45}$$

Thus, the LHS of the order-2 equation (4.13) must follow the same pattern, which determines the structure of $\mathbf{u_2}$ (see Eq. (4.19)).

The exponential pattern of $\mathbf{I_2}$ is affected by the choice of $\mathbf{u_1}$ (since $\mathbf{I_2}$ is a function of $\mathbf{u_1}$); the structure ansatz describes the mode of interest. The exponential patterns of $\mathbf{I_2}$ for the proposed structure ansazt in Sec. 4.3.2 and their corresponding $\mathbf{u_2}$ structure are summarised as follows:

- Stripes mode of the Turing pattern

$$\mathbf{I_2}^{\mathrm{T}} = (\cdots)e^{i\vec{q_c}\cdot\vec{r}} + (\cdots)e^{2i\vec{q_c}\cdot\vec{r}} + (\cdots) + c.c. \tag{4.46}$$

[2]We need to distinguish the word "pattern" used here for the description of a series of mathematical expressions (e.g., exponential functions) from the meaning of a spatial structure (e.g., spatial Turing structure)

$$\mathbf{u_2}^{\mathrm{T}} = \begin{pmatrix} c_{11} \\ c_{21} \\ \vdots \\ c_{n1} \end{pmatrix} e^{i\vec{q_c}\cdot\vec{r}} + \begin{pmatrix} c_{12} \\ c_{22} \\ \vdots \\ c_{n2} \end{pmatrix} e^{2i\vec{q_c}\cdot\vec{r}} + \begin{pmatrix} c_{13} \\ c_{23} \\ \vdots \\ c_{n3} \end{pmatrix} + c.c. \tag{4.47}$$

- Hopf mode

$$\mathbf{I_2}^{\mathrm{H}} = (\cdots)e^{i\omega_c t} + (\cdots)e^{2i\omega_c t} + (\cdots) + c.c. \tag{4.48}$$

$$\mathbf{u_2}^{\mathrm{H}} = \begin{pmatrix} c_{11} \\ c_{21} \\ \vdots \\ c_{n1} \end{pmatrix} e^{i\omega_c t} + \begin{pmatrix} c_{12} \\ c_{22} \\ \vdots \\ c_{n2} \end{pmatrix} e^{2i\omega_c t} + \begin{pmatrix} c_{13} \\ c_{23} \\ \vdots \\ c_{n3} \end{pmatrix} + c.c. \tag{4.49}$$

- Hexagonal mode of the Turing pattern

$$\begin{aligned} \mathbf{I_2}^{\mathrm{Hex}} =& (\cdots)e^{2i\vec{q_1}\cdot\vec{r}} + (\cdots)e^{2i\vec{q_2}\cdot\vec{r}} + (\cdots)e^{2i\vec{q_3}\cdot\vec{r}} + \\ & (\cdots)e^{i(\vec{q_1}-\vec{q_2})\cdot\vec{r}} + (\cdots)e^{i(\vec{q_1}-\vec{q_3})\cdot\vec{r}} + (\cdots)e^{i(\vec{q_2}-\vec{q_3})\cdot\vec{r}} + (\cdots) + c.c. \end{aligned} \tag{4.50}$$

$$\begin{aligned} \mathbf{u_2}^{\mathrm{Hex}} =& \begin{pmatrix} c_{11} \\ c_{21} \\ \vdots \\ c_{n1} \end{pmatrix} e^{2i\vec{q_1}\cdot\vec{r}} + \begin{pmatrix} c_{12} \\ c_{22} \\ \vdots \\ c_{n2} \end{pmatrix} e^{2i\vec{q_2}\cdot\vec{r}} + \begin{pmatrix} c_{13} \\ c_{23} \\ \vdots \\ c_{n3} \end{pmatrix} e^{2i\vec{q_3}\cdot\vec{r}} + \\ & \begin{pmatrix} c_{14} \\ c_{24} \\ \vdots \\ c_{n4} \end{pmatrix} e^{i(\vec{q_1}-\vec{q_2})\cdot\vec{r}} + \begin{pmatrix} c_{15} \\ c_{25} \\ \vdots \\ c_{n5} \end{pmatrix} e^{i(\vec{q_1}-\vec{q_3})\cdot\vec{r}} + \begin{pmatrix} c_{16} \\ c_{26} \\ \vdots \\ c_{n6} \end{pmatrix} e^{i(\vec{q_2}-\vec{q_3})\cdot\vec{r}} + \begin{pmatrix} c_{17} \\ c_{27} \\ \vdots \\ c_{n7} \end{pmatrix} + c.c. \end{aligned} \tag{4.51}$$

- Turing–Hopf mode

$$\begin{aligned} \mathbf{I_2}^{\mathrm{TH}} =& (\cdots)e^{i\omega_c t} + (\cdots)e^{i\vec{q_c}\cdot\vec{r}} + (\cdots)e^{i(\omega_c t+\vec{q_c}\cdot\vec{r})} + \\ & (\cdots)e^{i(\omega_c t-\vec{q_c}\cdot\vec{r})} + (\cdots)e^{2i\omega_c t} + (\cdots)e^{2i\vec{q_c}\cdot\vec{r}} + (\cdots) + c.c. \end{aligned} \tag{4.52}$$

$$\begin{aligned} \mathbf{u_2}^{\mathrm{TH}} =& \begin{pmatrix} c_{11} \\ c_{21} \\ \vdots \\ c_{n1} \end{pmatrix} e^{i\omega_c t} + \begin{pmatrix} c_{12} \\ c_{22} \\ \vdots \\ c_{n2} \end{pmatrix} e^{i\vec{q_c}\cdot\vec{r}} + \begin{pmatrix} c_{13} \\ c_{23} \\ \vdots \\ c_{n3} \end{pmatrix} e^{i(\omega_c t+\vec{q_c}\cdot\vec{r})} + \\ & \begin{pmatrix} c_{14} \\ c_{24} \\ \vdots \\ c_{n4} \end{pmatrix} e^{i(\omega_c t-\vec{q_c}\cdot\vec{r})} + \begin{pmatrix} c_{15} \\ c_{25} \\ \vdots \\ c_{n5} \end{pmatrix} e^{2i\omega_c t} + \begin{pmatrix} c_{16} \\ c_{26} \\ \vdots \\ c_{n6} \end{pmatrix} e^{2i\vec{q_c}\cdot\vec{r}} + \begin{pmatrix} c_{17} \\ c_{27} \\ \vdots \\ c_{n7} \end{pmatrix} + c.c. \end{aligned} \tag{4.53}$$

Here, c_{ij} is the unknown coefficient with the first index i being the dimension identifier (e.g. for the Brusselator model, $i = 1, 2$) and the second index j counts which exponential function c_{ij} is multiplied with.

Once the structure of $\mathbf{u_2}$ is determined, it will be substituted into the LHS of the order-2 equation. Then we apply the constraints derived from the order-2 solvability condition, and balance the exponential pattern for both sides of the order equation to construct a series of coefficient equations. The unknown coefficient c_{ij} can be solved from these coefficient equations to give an explicit description of the order-2 solution $\mathbf{u_2}$.

At last, the derived order-2 solution $\mathbf{u_2}$ and the structure ansatz $\mathbf{u_1}$ will be substituted into the order-3 equation. By applying the solvability condition (see Sec. 4.3.3), the constraints yield the order-3 amplitude equations.

4.3.5 General comments

The application of MSE involves many tedious calculations. For example, the term X^2Y in the Brusselator model Eq. (4.1) has the third-order multiple-scale expansion $(X_0 + \epsilon x_1 + \epsilon^2 x_2 + \epsilon^3 x_3)^2(Y_0 + \epsilon y_1 + \epsilon^2 y_2 + \epsilon^3 y_3)$, which, when fully expanded, gives 64 terms. Also, Eq. (4.53) shows that there are 14 unknown coefficients c_{ij} (and their 14 complex conjugates denoted as *c.c.*) in $\mathbf{u_2}$ for the Turing–Hopf mode, i.e., we need to solve for these coefficients from 28 coefficient equations. If a system has i model equations and j exponential components, there will be a total of $2 \times (i \times j + i)$ coefficient equations (including complex conjugates). For example, the Waikato cortical model has 8 first-order differential equations for the simplest case [118]. The pattern analysis of this cortical model requires us to solve 112 order-2 TH mode coefficient equations.

In the next section, we introduce `Amp_solving` — a semi-automated MSE algorithm in the MAPLE programming platform. `Amp_solving` allows for automatic MSE operation, thus eliminating the need for manually mathematical manipulations.

4.4 `Amp_solving` algorithm

`Amp_solving` algorithm, encoded in MAPLE, follows a standard MSE procedure to derive the order-3 amplitude equation of a n-dimensional reaction diffusion system

$$\frac{\partial}{\partial t}\mathbf{u} = f(\mathbf{u}, \boldsymbol{\phi}) + \nabla^2 \mathbf{u} \tag{4.54}$$

in which $\boldsymbol{\phi}$ is the bifurcation control parameter. The algorithm proceeds as follows:

1. Introduce a weakly linear perturbation $\boldsymbol{\delta}\mathbf{u}$ around the steady state $\mathbf{u_0}$: $\mathbf{u} = \mathbf{u_0} + \boldsymbol{\delta}\mathbf{u}$.
2. • Expand $\boldsymbol{\delta}\mathbf{u}$ as a nonlinear series:

$$\boldsymbol{\delta}\mathbf{u} = \sum_{p=1}^{k} \epsilon^p \mathbf{u}_p(T_0, T_1, \ldots T_k; R_0, R_1, \ldots R_k) \tag{4.55}$$

in which $T_m = \epsilon^m t$ and $R_m = \epsilon^m r$ $(m = 0, 1, 2 \ldots k)$ are scaled temporal and spatial arguments respectively.

- Expand the bifurcation control parameter $\boldsymbol{\phi}$:

$$\boldsymbol{\phi} = \boldsymbol{\phi}_0 + \sum_{p=1}^{k} \epsilon^p \boldsymbol{\phi}_p \tag{4.56}$$

Order equations can be obtained by grouping coefficients with the same order of ϵ. For example, the order-k equation is obtained by equating the coefficients of ϵ^k from both sides of the expanded model equations.

3. Determine the structure of the ansatz for $\mathbf{u_1}$.
   This structure ansatz is a linear combination of the specific modes of interest.
4. Substitute the ansatz $\mathbf{u_1}$ into the order-2 equation.
5. Derive the order-2 solvability condition by eliminating secular terms.
6. Substitute both the derived order-2 solvability condition and the structure ansatz into the order-2 equation, then solve the order-2 equation to give an explicit expression for $\mathbf{u_2}$.
7. Substitute the structure ansatz, derived order-2 solvability condition and the order-2 solution into the order-3 equation, which yields the order-3 amplitude equations.

We outline above steps in the flowchart of Fig. 4.3 for deriving the order-3 amplitude equations. A general flowchart for deriving the order-k amplitude equation is shown in Fig. 4.4. Although we can obtain any higher terms iteratively, it is sufficient to examine the pattern dynamics from the order-3 amplitude equation [56].

`Amp_solving` automates the tedious calculations of MSE mainly in the following parts: Constructing the order equation, solving the order equation and establishing the solvability condition. We now briefly explain the core commands used in `Amp_solving` to ease MSE calculations.

4.4.1 Constructing the order equation

The multiple-scale expansion starts by introducing a linear perturbation into the original reaction–diffusion system, leading to perturbation equations. The next step is to expand these perturbation nonlinearly in a scaling constant ϵ characterising the distance to the bifurcation threshold.

After substituting multiple-scale expansions (the multiple-scale expansion block in Fig. 4.3), `Amp_solving` uses

```
simplify(collect(..., e), {e^(e_order+1) = 0})
```

to collect all the coefficients with the same rational power of ϵ (ϵ is represented by `e` in `Amp_solving`) up to $\epsilon^{\texttt{e\_order}}$, leading to a series of *raw* order equations. In `collect`

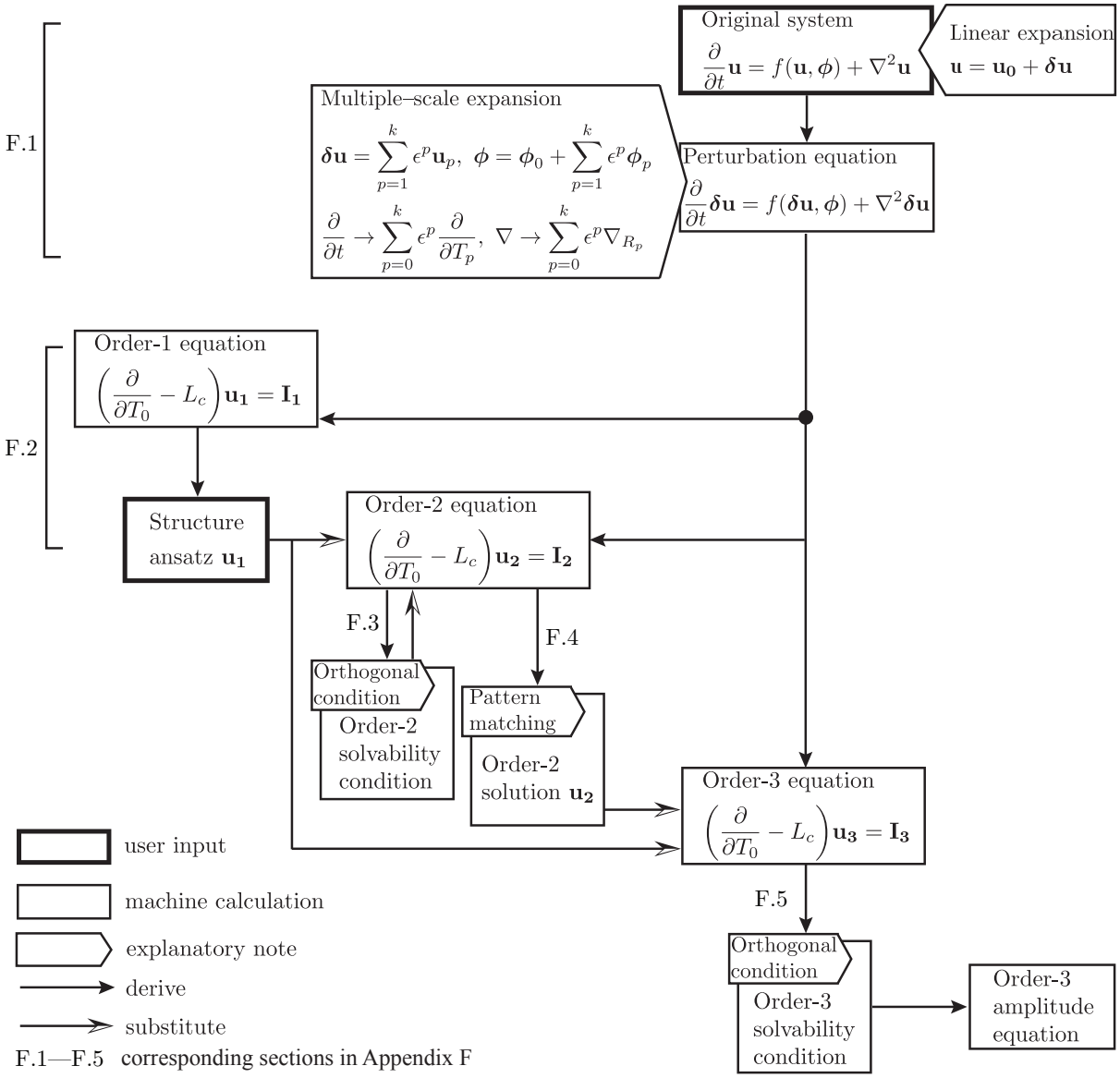


Figure 4.3: `Amp_solving` flowchart for deriving the order-3 amplitude equation.

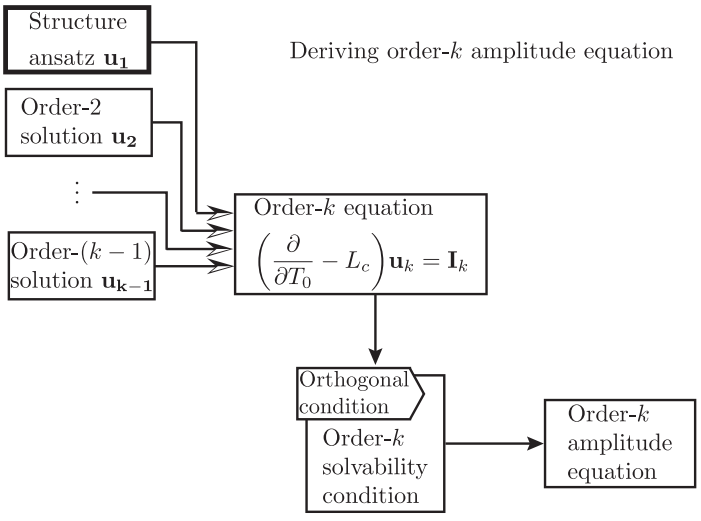
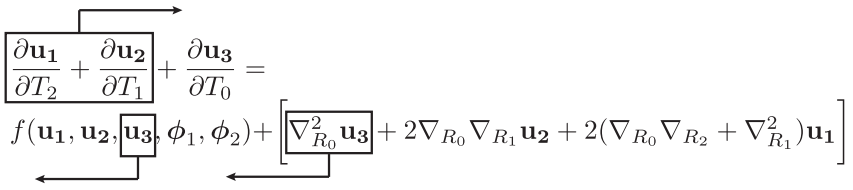


Figure 4.4: A general flowchart for deriving the order-k amplitude equation from the order-k equation. Figure properties see Fig. 4.3.

function, "..." is the set of multiple-scale expanded model equations. The raw order equation, for example, at order ϵ^3 has the expression:

$$\begin{aligned}&\frac{\partial \mathbf{u_1}}{\partial T_2}+\frac{\partial \mathbf{u_2}}{\partial T_1}+\frac{\partial \mathbf{u_3}}{\partial T_0}=\\&f(\mathbf{u_1},\mathbf{u_2},\mathbf{u_3},\boldsymbol{\phi}_1,\boldsymbol{\phi}_2)+\left[\nabla^2_{R_0}\mathbf{u_3}+2\nabla_{R_0}\nabla_{R_1}\mathbf{u_2}+2(\nabla_{R_0}\nabla_{R_2}+\nabla^2_{R_1})\mathbf{u_1}\right]\end{aligned} \tag{4.57}$$

`Amp_solving` uses the `selectmove` function to *select* the terms with order-3 variable $\mathbf{u_3}$ and group them at the LHS, meanwhile placing the remaining terms at the RHS. This process is shown below:



`Amp_solving` outputs the order equation in the from $\left(\frac{\partial}{\partial T_p}-L_{\mathrm{c}}\right)\mathbf{u}_p=\mathbf{I}_p$.

4.4.2 Solving the order equation

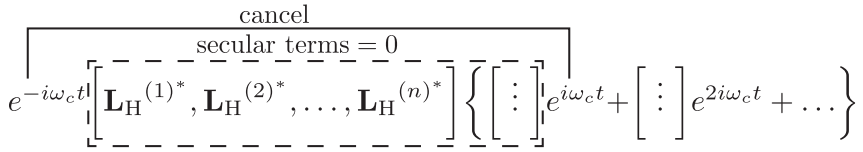
As mentioned in Sec. 4.3.4, to solve the order equation, we need to determine the structure of the solution, which `Amp_solving` constructs via the exponential pattern matching strategy: `Amp_solving` first substitutes the structure ansatz into the RHS of the order-2 equation, then calls `op` command to identify all exponential components. `Amp_solving` will count the number of extracted unique exponential components, and generate their corresponding coefficients c_{ij} (index definitions see Sec. 4.3.4). At last, `Amp_solving` combines coefficients and exponential components to form the structure of the order-2 solution.

After substituting the constructed order-2 solution (currently with unknown coefficients) into the order-2 equation, `Amp_solving` will produce coefficient equations and call `solve` command to solve all unknown coefficients.

4.4.3 Establishing the solvability condition

Applying the solvability condition requires two significant operations: dot product and identification of secular terms. `Amp_solving` uses `VectorCalclus[DotProduct]` to perform the dot product, then calls the `coeff` function to find secular terms that disobey the Fredholm alternative [40] (see Sec. 4.3.3 for details).

For example, the solvability condition for the Hopf mode reads $\langle \mathbf{v_H}|\mathbf{I_k}\rangle = 0$. `Amp_solving` finds the secular terms according to the following scheme:



Scheme for identifying secular terms: coefficient of e^0 after the dot product.

The order-2 solvability condition yields constraints that will be applied back to the order-2 equation for its solution. `Amp_solving` utilises both order-2 solvability condition and solution to derive the order-3 amplitude equation.

Appendix E provides a mini MAPLE tutorial focusing on coding strategies specific to `Amp_solving`:

- vectorising variables
- defining and displaying functions
- computing the total derivative via the differentiation chain rule
- defining constants

In Appendixes F and G, we provide an expanded commentary of the `Amp_solving` codes used to derive the Brusselator amplitude equation for the hexagonal and Turing–Hopf modes. These commentaries are intended to help readers to master the application of `Amp_solving`.

4.5 Summary

The derivation of the amplitude equation for a reaction–diffusion system via a multiple-scale expansion (MSE) in the vicinity of a bifurcation point is carefully explained in this chapter. A customised MAPLE algorithm `Amp_solving` is proposed to automate the MSE manipulation. By taking the advantage of the powerful MAPLE symbolic computation engine, `Amp_solving` can derive the amplitude equation in an analytical expression with high efficiency and accuracy. A programming tutorial is provided in the appendix to introduce the coding strategies used in `Amp_solving`. Following the tutorial, examples are given to demonstrate using `Amp_solving` to derive the Brusselator amplitude equation for the hexagonal (codimension-1 point) and Turing–Hopf (codimension-2 point) modes. The derived amplitude equations are consistent with the work by other researchers [56, 92]. Since `Amp_solving` is designed for a general reaction–diffusion system (for multiple bifurcation cases), we hope that this chapter will be valuable for readers who wish to study pattern dynamics but are not familiar with the MSE approach. Our intention is that readers will find our symbolic codes useful for expediting the extraction of the amplitude equation for other reaction–diffusion systems.

In the next chapter, we will investigate the mode stabilities of the Brusselator and Waikato cortical models via use of amplitude equations.

Chapter 5

Mode stability analysis by amplitude equations

In Chapter 2, linear stability analysis (LSA) suggests that a bifurcation occurs when the bifurcation control parameter crosses a certain threshold. This prediction matches moderately well with our Brusselator and cortical model simulations. However, we also found LSA-unpredicted Turing structures (Fig. 2.3) and TH competitions (Fig. 2.4) for the Brusselator model, which we argue may arise from the nature of the mode competition when the system has crossed the bifurcation threshold and evolved into a nonlinear region where LSA is no longer applicable. To address this issue, in Chapter 4 we utilised multiple-scale expansion (MSE) to introduce a nonlinear perturbation[1] to the pattern-forming system, which results to the so-called amplitude equations describing the near-bifurcation behaviours of the most unstable modes. In this chapter, we will further examine the stability of emergent patterns by investigating the amplitude equations. This will entail a comprehensive analysis of the pattern dynamics of the Brusselator model: in Sec. 5.1, we will discuss the mechanism of the Turing pattern selection; In Sec. 5.2, we will investigate the stability of the mixed Turing–Hopf mode. All developed theories of pattern dynamics will be adopted in analysing the mode stability of the Waikato cortical model in Sec. 5.3.

5.1 Brusselator amplitude equations for the hexagonal mode

Recall the Brusselator amplitude equations for the hexagonal mode, as derived in Appendix F:

$$\begin{aligned}
\frac{\partial}{\partial t}Z_1 &= \mu Z_1 + vZ_2^*Z_3^* - g|Z_1|^2Z_1 - h(|Z_2|^2 + |Z_3|^2)Z_1 \\
\frac{\partial}{\partial t}Z_2 &= \mu Z_2 + vZ_1^*Z_3^* - g|Z_2|^2Z_2 - h(|Z_1|^2 + |Z_3|^2)Z_2 \\
\frac{\partial}{\partial t}Z_3 &= \mu Z_3 + vZ_1^*Z_2^* - g|Z_3|^2Z_3 - h(|Z_1|^2 + |Z_2|^2)Z_3
\end{aligned} \tag{5.1}$$

[1] LSA introduces linear perturbation

As we can see, the equations for Z_2 and Z_3 can be obtained by simple permutation of the indices. In the previous chapter, we calculated the explicit expressions for equation coefficients:

$$\mu = (B - B_c)/B_c \qquad v = \frac{2}{A}\left(\frac{1 - A\eta}{1 + A\eta}\right) + \frac{2}{A}\mu$$
$$g = \frac{38A\eta + 5(A\eta)^2 - 8 - 8(A\eta)^3}{9A^3\eta(1 + A\eta)} \quad h = \frac{5A\eta + 7(A\eta)^2 - 3 - 3(A\eta)^3}{A^3\eta(1 + A\eta)}, \quad \eta = \sqrt{D_X/D_Y}$$

We consider μ as a bifurcation control parameter of the amplitude equations since it measures the relative distance of the bifurcation setting B to its Turing threshold B_c.

5.1.1 Steady-state solutions

To split the modulus and phase, we use an ansatz $Z_i = \rho_i e^{i\phi_i}$, thus obtaining modulus equations:

$$\begin{aligned}
\frac{\partial \rho_1}{\partial t} &= \mu\rho_1 + v\rho_2\rho_3 - g\rho_1^3 - h(\rho_2^2 + \rho_3^2)\rho_1 \\
\frac{\partial \rho_2}{\partial t} &= \mu\rho_2 + v\rho_1\rho_3 - g\rho_2^3 - h(\rho_1^2 + \rho_3^2)\rho_2 \\
\frac{\partial \rho_3}{\partial t} &= \mu\rho_3 + v\rho_1\rho_2 - g\rho_3^3 - h(\rho_1^2 + \rho_2^2)\rho_3
\end{aligned} \tag{5.2}$$

and phase equations:

$$\begin{aligned}
\Phi &= \phi_1 + \phi_2 + \phi_3 \\
\frac{\partial \Phi}{\partial t} &= -v\left(\frac{\rho_1^2\rho_2^2 + \rho_1^2\rho_3^2 + \rho_2^2\rho_3^2}{\rho_1\rho_2\rho_3}\right)\sin\Phi
\end{aligned} \tag{5.3}$$

The steady-state of the modulus takes the form

$$0 = \mu\rho_1 + v\rho_2\rho_3 - g\rho_1^3 - h(\rho_2^2 + \rho_3^2)\rho_1 \tag{5.4}$$

The system possesses three types of solutions:

- Single root – Stripes (one wavevector)

$$\rho_1^{\mathrm{s}} = \sqrt{\mu/g}, \quad \rho_2^{\mathrm{s}} = \rho_3^{\mathrm{s}} = 0 \tag{5.5}$$

- Triple identical roots – Hexagon (three wavevectors with identical modulus)

$$\begin{aligned}
\rho_1^{\mathrm{h}} &= \rho_2^{\mathrm{h}} = \rho_3^{\mathrm{h}} \equiv \rho^{\mathrm{h}} \\
\rho^{\mathrm{h}} &= \frac{v \pm \sqrt{v^2 + 4\mu(g + 2h)}}{2(g + 2h)} \equiv \rho_{\pm}^{\mathrm{h}}
\end{aligned} \tag{5.6}$$

- Squeezed hexagon

$$\rho_1^{\mathrm{m}} = \frac{v}{h-g}, \quad \rho_2^{\mathrm{m}} = \rho_3^{\mathrm{m}} = \sqrt{\frac{\mu - g(\rho_1^{\mathrm{m}})^2}{g+h}} \equiv \rho_{2,3}^{\mathrm{m}} \tag{5.7}$$

Fig. 5.1 illustrates the specific modes.

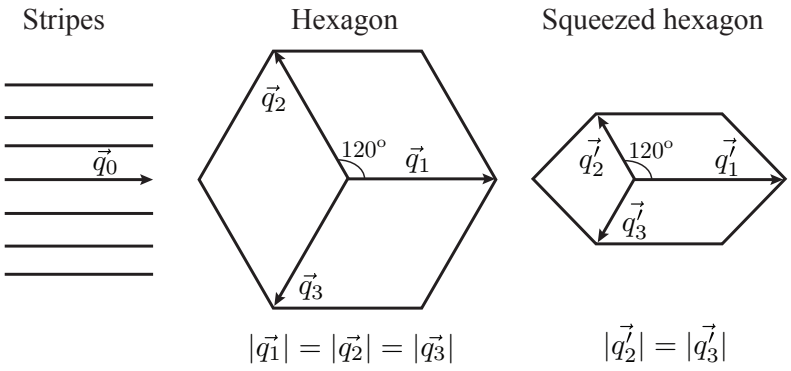


Figure 5.1: Three basic modes of the Turing patterns.

Examining Eq. (5.3), Φ relaxes monotonically to zero (π) when $v > 0$ ($v < 0$), as demonstrated in Fig. 5.2.

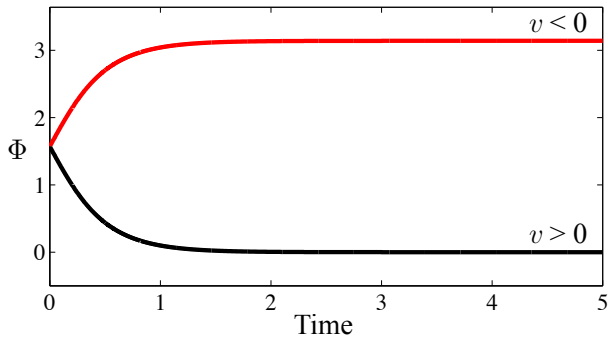


Figure 5.2: Demonstrated phase evolution $\Phi(t)$ with respect to the sign of v: [upper curve] Φ saturates to π when $v < 0$; [bottom curve] Φ decays to 0 when $v > 0$.

Fig. 5.2 shows that Φ has two stationary solutions: π and 0, which corresponds to two hexagonal structures (see examples in Fig. 5.10). When $\Phi = \pi$, we denote this hexagonal structure H_π. When $\Phi = 0$, the hexagonal structure will be reentrant with H_π; we denote this structure H_0.

The sign of v determines the nature of the hexagonal structure. Thus there is an exchange of stability between two structures at $v = 0$, where $\mu = (A\eta - 1)/(A\eta + 1) \equiv \mu_p$.

From

$$v = \frac{2}{A}\left(\frac{1 - A\eta}{1+\eta}\right) + \frac{2}{A}\mu \equiv \alpha\mu + \alpha\beta, \tag{5.8}$$

we have $v \propto \mu$. Increasing μ, H_π appears first, then it evolves to H_0.

5.1.2 Eigenvalue analysis

To examine the stability of various Turing modes for the Brusselator model, and to locate their corresponding parametric space where the mode is stable, we will apply linear stability analysis (LSA) on the amplitude equations Eq. (5.2):

$$\rho_i = \rho_i^0 + \delta\rho_i \tag{5.9}$$

where $i = 1, 2, 3$ and ρ_i^0 is the steady-state solution of ρ_i. To find the stability condition for each Turing mode, we first substitute Eq. (5.9) into Eq. (5.2), then derive the Jacobian matrix with respect to $\delta\rho_1$, $\delta\rho_2$ and $\delta\rho_3$. The mode stability is determined by the sign of the real part of the dominant eigenvalue.

- **Stripes**

  The stripes mode has one trivial solution, as shown in Eq. (5.5). The Jacobian matrix of the stripes mode reads:

$$\mathrm{Jac}^{\mathrm{s}} = \begin{bmatrix} a & 0 & 0 \\ 0 & b & c \\ 0 & c & b \end{bmatrix} \tag{5.10}$$

  where $a = -2\mu, b = \mu - h\dfrac{\mu}{g}, c = v\sqrt{\dfrac{\mu}{g}}$. Three eigenvalues are solved:

$$\begin{aligned} \lambda_1^{\mathrm{s}} &= -2\mu \\ \lambda_2^{\mathrm{s}} &= \mu\left(1 - \frac{h}{g}\right) + v\sqrt{\frac{h}{g}} \\ \lambda_3^{\mathrm{s}} &= \mu\left(1 - \frac{h}{g}\right) - v\sqrt{\frac{h}{g}} \end{aligned}$$

  To reach the point where the stripes mode loses stability, we set the dominant eigenvalue to zero:

$$\begin{aligned} \lambda_2^{\mathrm{s}} = 0 \text{ or } \lambda_3^{\mathrm{s}} = 0 \quad &\Rightarrow \quad \mu = \frac{g}{(g-h)^2} v^2 \\ &\Rightarrow \quad v^2 = \mu \frac{(g-h)^2}{g} \equiv \mu_{\mathrm{s}}\gamma_{\mathrm{s}} \end{aligned} \tag{5.11}$$

  By substituting Eq. (5.8) into (5.11), we have a quadratic equation in μ_{s}:

$$\begin{aligned} &\alpha^2\mu_{\mathrm{s}}^2 + (2\alpha^2\mu_{\mathrm{s}}\beta + \alpha^2\beta^2) = \mu_{\mathrm{s}}\gamma_{\mathrm{s}} \\ \Rightarrow \quad &\alpha^2\mu_{\mathrm{s}}^2 + (2\alpha^2\beta - \gamma_{\mathrm{s}})\mu_{\mathrm{s}} + \alpha^2\beta^2 = 0 \\ \Rightarrow \quad &\mu_{\mathrm{s}} = \frac{-(2\alpha^2\beta - \gamma_{\mathrm{s}}) \pm \sqrt{\gamma_{\mathrm{s}}^2 - 4\alpha^2\beta\gamma_{\mathrm{s}}}}{2\alpha^2} \equiv \mu_{\mathrm{s}}^{\pm} \end{aligned} \tag{5.12}$$

  $\mu_{\mathrm{s}}^{\pm}$ indicates where the stripes change mode stability.

- **Hexagon**

  The hexagonal mode possesses three identical steady-state solutions, as shown in Eq. (5.6). Jacobian matrix of the hexagonal structure reads:

$$\mathrm{Jac}^{\mathrm{h}} = \begin{bmatrix} a & b & b \\ b & a & b \\ b & b & a \end{bmatrix} \tag{5.13}$$

  where $a = \mu - 3g(\rho^{\mathrm{h}})^2 - 2h(\rho^{\mathrm{h}})^2$, $b = v\rho^{\mathrm{h}} - 2h(\rho^{\mathrm{h}})^2$. In this case there are three eigenvalues but two of these are identical:

$$\lambda_1^{\mathrm{h}} = \lambda_2^{\mathrm{h}} = a - b = \mu - 3g(\rho^{\mathrm{h}})^2 - v\rho^{\mathrm{h}}$$
$$\lambda_3^{\mathrm{h}} = a + 2b = \mu - 3g(\rho^{\mathrm{h}})^2 + 2v\rho^{\mathrm{h}} - 6h(\rho^{\mathrm{h}})^2$$

  Similar to the stripes case, we will locate where the hexagon loses its stability. By substituting the hexagonal solution Eq. (5.6) into λ_1^{h}, λ_2^{h} and λ_3^{h}, we hereby obtain:

$$\mu \equiv \mu_{\mathrm{h}} = \frac{v^2(2g+h)}{(g-h)^2},$$

  thus

$$v^2 = \frac{\mu_{\mathrm{h}}(g-h)^2}{2g+h} \equiv \mu_{\mathrm{h}}\gamma_{\mathrm{h}} \tag{5.14}$$

  Again, by substituting Eq. (5.8) into (5.14), we have a quadratic equation about μ_{h}:

$$\begin{aligned} &(\alpha\mu_{\mathrm{h}} + \alpha\beta)^2 = \mu_{\mathrm{h}}\gamma_{\mathrm{h}} \\ \Rightarrow \quad &\mu_{\mathrm{h}}^2 + (2\beta - \frac{\gamma_{\mathrm{h}}}{\alpha^2})\mu_{\mathrm{h}} + \beta^2 = 0 \\ &\mu_{\mathrm{h}} = \frac{\gamma_{\mathrm{h}} - 2\alpha^2\beta \pm \sqrt{\gamma_{\mathrm{h}}^2 - 4\alpha^2\beta\gamma_{\mathrm{h}}}}{2\alpha^2} \equiv \mu_{\mathrm{h}}^{\pm} \end{aligned} \tag{5.15}$$

  $\mu_{\mathrm{h}}^{\pm}$ determines where the hexagonal mode changes its stability.

- **Squeezed hexagon**

  In practice, in pattern simulations, it is difficult to obtain ideal stripes or hexagonal structures because of the interference from the squeezed hexagonal mode. This mode causes distorted stripes and hexagons, as shown in Fig. 5.3. The analysis of the mode distortion will not be considered hereafter since our interests are of the stability of the stripes and hexagonal modes. Here, we only present the explicit forms of the Jacobian matrix and its eigenvalues.

  The Jacobian matrix of the squeezed hexagon reads:

$$\mathrm{Jac}^{\mathrm{m}} = \begin{bmatrix} a & d & d \\ d & b & c \\ d & c & b \end{bmatrix} \tag{5.16}$$

where $a = \mu - 3g(\rho_1^{\mathrm{m}})^2 - 2h(\rho_{2,3}^{\mathrm{m}})^2, b = a = \mu - 3g(\rho_{2,3}^{\mathrm{m}})^2 - h(\rho_1^{\mathrm{m}})^2 - h(\rho_{2,3}^{\mathrm{m}})^2, c = v\rho_1^{\mathrm{m}} - 2h(\rho_{2,3}^{\mathrm{m}})^2$ and $d = v\rho_{2,3}^{\mathrm{m}} - 2h\rho_1^{\mathrm{m}}\rho_{2,3}^{\mathrm{m}}$. Three eigenvalues are:

$$\lambda_1^{\mathrm{m}} = 2\frac{\mu(g-h)^2 - v^2(2g+h)}{(g+h)(h-g)}$$
$$\lambda_2^{\mathrm{m}} = \frac{gv^2 - (3h+g)(h-g)\mu + (h^2-g^2)R}{2(g+h)(h-g)}$$
$$\lambda_3^{\mathrm{m}} = \frac{gv^2 - (3h+g)(h-g)\mu - (h^2-g^2)R}{2(g+h)(h-g)}$$

in which $R = \sqrt{a^2 - 2ab - 2ac + b^2 + 2bc + c^2 + 8d^2}$

5.1.3 Simulation results

In Sec. 5.1.1, we have noted the steady-state solutions for the stripes and hexagonal modes. Now let us investigate stabilities of these modes by varying the bifurcation control parameter μ.

Fig. 5.3 predicts the stabilities of stripes (red), H_0 (blue) and H_π (black) modes for the Turing instability of the Brusselator model. From the range of solid curves, we have the summary of parametric space where a specific mode is stable: The stripes mode is stable when $\mu_{\mathrm{s}}^- < \mu < \mu_{\mathrm{s}}^+$; the hexagonal mode is stable (H_π and H_0) when $\mu < \mu_{\mathrm{h}}^-$ or $\mu > \mu_{\mathrm{h}}^+$. H_π and H_0 interact at μ_{p} where they exchange mode stability, that is, H_π will transit to H_0 when μ crosses μ_{p} from its LHS to the RHS.

To verify the predictions from the bifurcation diagram, we select five different values of μ then examine the simulated patterns:

(a) $\mu = 0.0495$ (i.e., $B = 8.04$, see Fig. 2.3(a)) falling into a range where only H_π is stable;
(b) $\mu = 0.1100$ falling into a range where stripes and H_π modes coexist;
(c) $\mu = 0.3994$ (i.e., $B = 10.72$, see Fig. 2.3(b)) where only the stripes structure is stable;
(d) $\mu = 0.7000$ again falling into a bistable range where stripes and H_0 coexist;
(e) $\mu = 1.4802$ (i.e., $B = 19$, see Fig. 2.3(c)) where only H_0 is stable.

In Fig. 5.3, we see good agreement between simulated patterns and theoretical predictions. Clear H_π, stripes and H_0 structures are observed at (a), (c) and (e) cases respectively. (b) and (d) show mixed states between forward and backward stable structures. Consequently, we can conclude that by increasing μ, the distance to the critical point, positively, Brusselator forms sequentially from H_π, to stripes, to H_0. Here, amplitude equations precisely capture the mechanism of the pattern selection thus providing a straightforward guidance for determining the mode of Turing patterns.

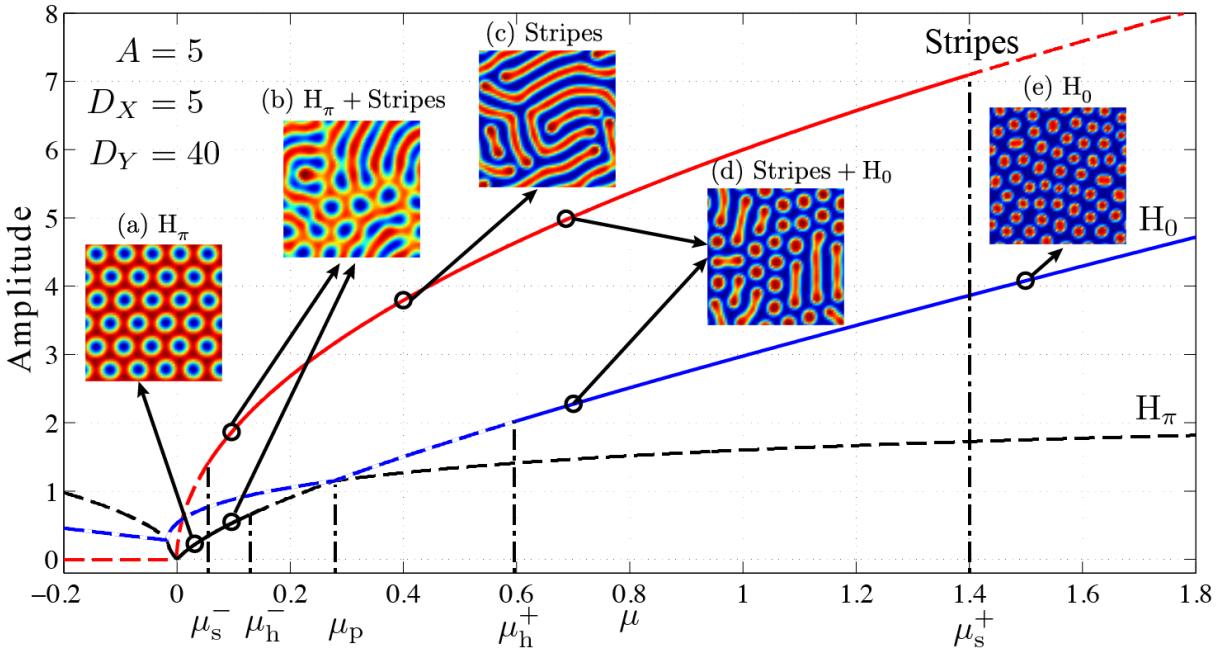


Figure 5.3: Turing mode stability of the Brusselator model. Red – Stripes, blue – H_0, black – H_π. Solid and dashed curves correspond to stable and unstable modes respectively, according to the eigenvalue analysis. Five representative μ values are selected for comparison of theoretical predictions for mode stability against practical simulations (shown as subplots). Colour of the pattern indicates the local concentration of the reactant: [red] high concentration, [blue] low concentration. Simulations follow the same settings as in Sec. 2.2.1. Model parameters: $A = 5, D_X = 5, D_Y = 40$.

5.2 Brusselator amplitude equations for the TH mode

In Appendix G, we derive the TH mode amplitude equations for the Brusselator model:

$$\frac{\partial T}{\partial t} = C_{\mathrm{T}} T - C_{\mathrm{TT}}|T|^2 T - C_{\mathrm{TH}}|H|^2 T + D_{\mathrm{T}}\nabla^2 T \tag{5.17a}$$

$$\frac{\partial H}{\partial t} = C_{\mathrm{H}} H - (C_{\mathrm{HHr}} + iC_{\mathrm{HHi}})|H|^2 H - (C_{\mathrm{HTr}} + iC_{\mathrm{HTi}})|T|^2 H + (D_{\mathrm{Hr}} + iD_{\mathrm{Hi}})\nabla^2 H \tag{5.17b}$$

where C_{T} and C_{H} are bifurcation control parameters. For the Brusselator model, C_{TT}, C_{TH}, D_{T}, C_{HHr}, C_{HTr} and D_{Hr} are positive to assure super-critical bifurcations.

5.2.1 Steady-state solutions and stability analysis

Ignoring diffusion terms, amplitude equations Eqs. (5.17a, 5.17b) possess three homogeneous steady-state solutions:

1. Turing solution

$$T_{\mathrm{Ts}} = \sqrt{\frac{C_{\mathrm{T}}}{C_{\mathrm{TT}}}},\ H_{\mathrm{Ts}} = 0 \tag{5.18}$$

2. Hopf solution

$$H_{\mathrm{Hs}} = \sqrt{\frac{C_{\mathrm{H}}}{C_{\mathrm{HHr}}}},\ T_{\mathrm{Hs}} = 0 \tag{5.19}$$

3. TH mixed mode solution

$$T_{\mathrm{Ms}} = \sqrt{(C_{\mathrm{HHr}}C_{\mathrm{T}} - C_{\mathrm{TH}}C_{\mathrm{H}})/\Delta} \tag{5.20a}$$

$$H_{\mathrm{Ms}} = \sqrt{(C_{\mathrm{TT}}C_{\mathrm{H}} - C_{\mathrm{HTr}}C_{\mathrm{T}})/\Delta} \tag{5.20b}$$

where $\Delta = C_{\mathrm{HHr}}C_{\mathrm{TT}} - C_{\mathrm{TH}}C_{\mathrm{HTr}}$. T_{Ms} and H_{Ms} coexist

$$\text{when } \frac{C_{\mathrm{TH}}}{C_{\mathrm{HHr}}} < \frac{C_{\mathrm{T}}}{C_{\mathrm{H}}} < \frac{C_{\mathrm{TT}}}{C_{\mathrm{HTr}}} \quad \text{if} \quad \Delta > 0, \text{or}$$

$$\text{when } \frac{C_{\mathrm{TT}}}{C_{\mathrm{HTr}}} < \frac{C_{\mathrm{T}}}{C_{\mathrm{H}}} < \frac{C_{\mathrm{TH}}}{C_{\mathrm{HHr}}} \quad \text{if} \quad \Delta < 0$$

We then analyse the stability of these steady states with regard to homogeneous perturbations.

- Stability of the Turing mode
  Assuming $T = T_{\mathrm{Ts}} + \delta T$; $H = \delta H$, Eqs. (5.17a, 5.17b) are linearised to:

$$\frac{\partial \delta T}{\partial t} = -2C_{\mathrm{T}}\delta T \tag{5.21a}$$

$$\frac{\partial \delta H}{\partial t} = (C_{\mathrm{H}} - C_{\mathrm{HTr}}T_{\mathrm{Ts}}^2)\delta H \tag{5.21b}$$

  We see that the Turing mode is super-critically stable since δT in Eq. (5.21a) will be linearly decayed if C_{T} is positive. However, the Turing mode will become unstable with respect to the enhanced homogeneous limit cycle (δH will be linearly increased) if

$$C_{\mathrm{H}} - C_{\mathrm{HTr}}T_{\mathrm{Ts}}^2 > 0,$$

  which leads to

$$\frac{C_{\mathrm{T}}}{C_{\mathrm{H}}} < \frac{C_{\mathrm{TT}}}{C_{\mathrm{HTr}}} \tag{5.22}$$

- Stability of the Hopf mode
  Assuming $T = \delta T$; $H = H_{\mathrm{Hs}} + \delta H$, Eqs. (5.17a, 5.17b) are linearised to:

$$\frac{\partial \delta T}{\partial t} = (C_{\mathrm{T}} - C_{\mathrm{TH}}H_{\mathrm{Hs}}^2)\delta T \tag{5.23a}$$

$$\frac{\partial \delta H}{\partial t} = -2C_{\mathrm{H}}\delta H \tag{5.23b}$$

  Hopf mode is super-critically stable. However, the Hopf mode will become unstable if

$$C_{\mathrm{T}} - C_{\mathrm{TH}}H_{\mathrm{Hs}}^2 > 0, \tag{5.24}$$

which leads to

$$\frac{C_{\mathrm{T}}}{C_{\mathrm{H}}} > \frac{C_{\mathrm{TH}}}{C_{\mathrm{HHr}}} \tag{5.25}$$

- Stability of the mixed TH mode
  Assuming $T = T_{\mathrm{Ms}} + \delta T$; $H = H_{\mathrm{Ms}} + \delta H$, Eqs. (5.17a, 5.17b) are linearised to:

$$\frac{\partial \delta T}{\partial t} = -2C_{\mathrm{TT}}T_{\mathrm{Ms}}^2\delta T - 2C_{\mathrm{TH}}H_{\mathrm{Ms}}T_{\mathrm{Ms}}\delta H \tag{5.26a}$$

$$\frac{\partial \delta H}{\partial t} = -2C_{\mathrm{HTr}}H_{\mathrm{Ms}}T_{\mathrm{Ms}}\delta T - 2C_{\mathrm{HHr}}H_{\mathrm{Ms}}^2\delta H \tag{5.26b}$$

  The product of the two eigenvalues is the determinant of the Jacobian matrix:

$$\lambda_1\lambda_2 \equiv I = 4H_{\mathrm{Ms}}^2H_{\mathrm{Ms}}^2(C_{\mathrm{TT}}C_{\mathrm{HHr}} - C_{\mathrm{TH}}C_{\mathrm{HTr}}) = 4H_{\mathrm{Ms}}^2H_{\mathrm{Ms}}^2\Delta$$

  The sum of the two eigenvalues is the trace of the Jacobian matrix:

$$\lambda_1 + \lambda_2 \equiv S = -2(C_{\mathrm{TT}}T_{\mathrm{Ms}}^2 + C_{\mathrm{HHr}}H_{\mathrm{Ms}}^2)$$

  Assuming C_{TT} and C_{HHr} are both positive, $S < 0$ always holds. Two eigenvalues can be both negative or have opposite signs, depending on the sign of Δ:

$$\text{Mixed mode is stable} \quad \text{if } \Delta > 0 \tag{5.27a}$$

$$\text{Mixed mode is unstable} \quad \text{if } \Delta < 0 \tag{5.27b}$$

From the above stability analysis, we can conclude stabilities of Turing and Hopf modes near a codimension-2 point (CTHP) in the bifurcation sketch of Fig. 5.4:

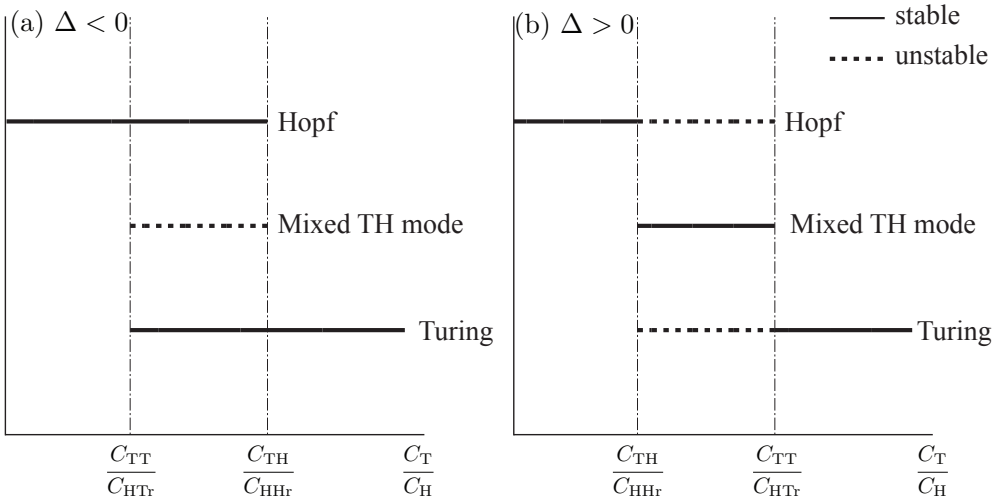


Figure 5.4: Bifurcation diagram with $C_{\mathrm{T}}/C_{\mathrm{H}}$ being the control parameter. Each mode has its corresponding range where it is stable. $C_{\mathrm{TT}}/C_{\mathrm{HTr}}$ and $C_{\mathrm{TH}}/C_{\mathrm{HHr}}$ are the boundaries for the mode stability. In the case (a) $\Delta < 0$, the range where mixed mode is unstable experiences bistability between Hopf and Turing modes. In the case of (b) $\Delta > 0$, mixed mode is always stable between $C_{\mathrm{TH}}/C_{\mathrm{HHr}}$ and $C_{\mathrm{TT}}/C_{\mathrm{HTr}}$.

If $\Delta < 0$, by successively increasing $C_{\mathrm{T}}/C_{\mathrm{H}}$, mode evolves from stable Hopf to stable Turing, with a transition where mixed TH is unstable. When $C_{\mathrm{T}}/C_{\mathrm{H}}$ falls into this unstable range, the system may exhibit either temporal or spatial instabilities depending on initial conditions.

If $\Delta > 0$, the mixed mode is always stable in the field where Hopf and Turing are both unstable. Increasing $C_{\mathrm{T}}/C_{\mathrm{H}}$, the following sequence of states are observed: Hopf $\rightarrow$ mixed TH $\rightarrow$ Turing.

Until now we have conducted the bifurcation analysis on Eqs. (5.17a, 5.17b) without considering the imaginary parts of Eq. (5.17b) (assuming H is real). The sign of Δ determines the stability of the mixed mode. When $\Delta > 0$, mixed mode is always stable. However, this conclusion is deduced based on real H. In Sec. 5.2.3, we will investigate the stability of the mixed mode by introducing a complex H, when $\Delta > 0$.

5.2.2 Application of amplitude-equation theory to the Brusselator model

In this section, we will apply the TH mode stability analysis we derived from the amplitude equations to predict the pattern dynamics of the Brusselator model.

Let us remark that the bifurcation control parameter $C_{\mathrm{T}}/C_{\mathrm{H}}$ in the amplitude equations (5.17a, 5.17b) is related to $(B - B_0)$ and $(\sigma - \sigma_0)$:

$$\frac{C_{\mathrm{T}}}{C_{\mathrm{H}}} = \frac{2}{1+\sigma_0} - \frac{4}{(-1+\sigma_0)^2}\frac{\sigma - \sigma_0}{B - B_0} \tag{5.28}$$

and

$$\frac{C_{\mathrm{T}}}{C_{\mathrm{H}}} \text{ is proportional to } B - B_0$$

In the following, we will investigate the mode stability of the Brusselator amplitude equations by varying B with σ fixed. Since Fig. 5.4 shows the sign of Δ plays an important role in mode dynamics, $\Delta < 0$ and $\Delta > 0$ cases will be discussed separately.

$\Delta > 0$

When $\Delta > 0$, Fig. 5.4(b) shows that the bifurcation follows stable Hopf $\rightarrow$ stable mixed mode $\rightarrow$ stable Turing by increasing the bifurcation control parameter. The Brusselator model setting $A = 2.5, D_X = 5, D_Y = 10$ ensures $\Delta > 0$. As a result, Fig. 5.5 predicts that the Brusselator model will experience a stable Hopf instability when $B < 7.7$, a stable TH instability when $7.735 < B < 7.771$ and a stable Turing instability when $B > 7.771$.

Then, we select three different values of B and forecast the corresponding model dynamics from Fig. 5.5: (1) $B = 7.4$, predicted stable Hopf; (2) $B = 7.76$, predicted stable mixed TH mode; (3) $B = 8.3$, predicted stable Turing. We started the simulation with specific B to verify these predictions.

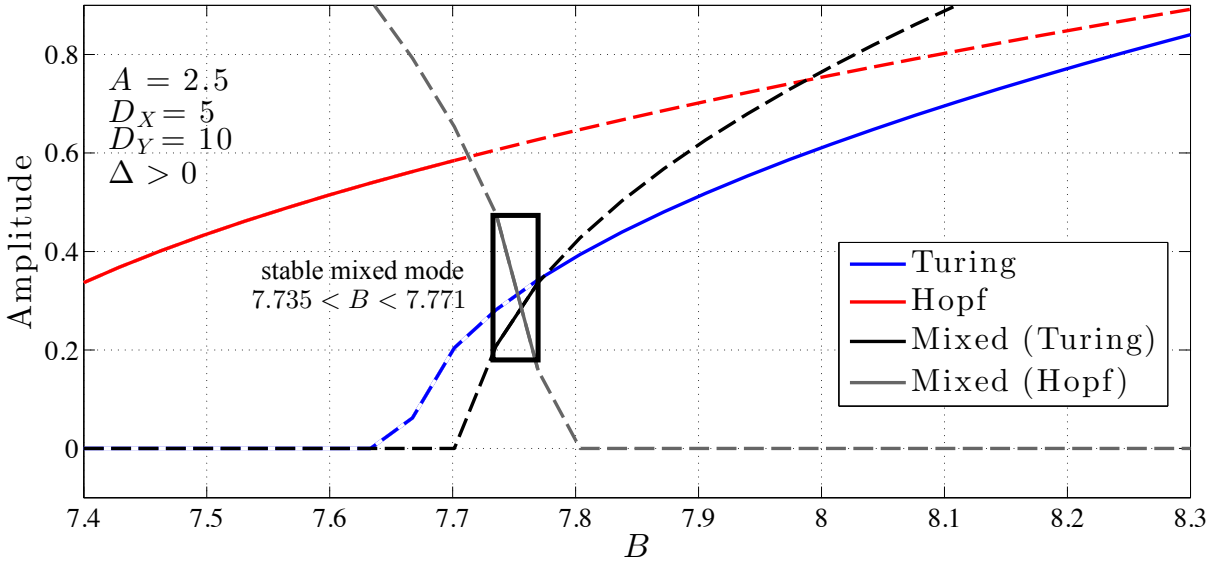


Figure 5.5: Mode stability of the TH mode amplitude equations for the Brusselator model when $\Delta > 0$. The mode solution at a specific B is solved from the real amplitude equations Eqs. (5.17a, 5.17b) by ignoring their imaginary parts: [blue] Pure Turing solution Eq. (5.18); [red] Pure Hopf solution Eq. (5.19); [black] Turing solution Eq. (5.20a) and [grey] Hopf solution Eq. (5.20b) of the mixed mode. The [solid curve] stable and [dashed curve] unstable modes are determined via the eigenvalue analysis. Mixed mode are stable when $7.735 < B < 7.771$.

In Fig. 5.6, we see that the LSA predicts a Hopf mode when $B = 7.4$ and a TH mode when $B = 7.76$. Both predictions agree with the amplitude equation method. However, the LSA loses its success when $B = 8.3$, where the LSA predicts a TH mode while the simulation exhibits a stable Turing mode. We argue that the LSA gradually loses its capability in describing pattern dynamics when the bifurcation distance $(B - B_0)$ is becoming larger with stronger nonlinearity (LSA is a weakly linear perturbation method).

$\Delta < 0$

Let us now examine the pattern dynamics for the Brusselator model when $\Delta < 0$. To ensure this condition, we introduced another parameter setting: $A = 2, D_X = 4, D_Y = 9.7$.

When $\Delta < 0$, Fig. 5.4(a) shows that the bifurcation follows stable Hopf $\rightarrow$ bistability $\rightarrow$ stable Turing by increasing the bifurcation control parameter.

In Fig. 5.7, we find the bistable range when $5.3 < B < 6.3$. At the LHS of this range, Hopf is the only stable mode; at the RHS of this range, Turing is the only stable mode. The mixed mode is unstable at all B values.

Again, we selected specific B values: (1) $B = 5.2$, predicted stable Hopf mode; (2) $B = 5.4$ and (3) $B = 6.0$, predicted either Hopf or Turing mode; (4) $B = 6.6$, predicted stable Turing mode. Then we compared their simulation results (see Fig. 5.8) with these mode predictions.

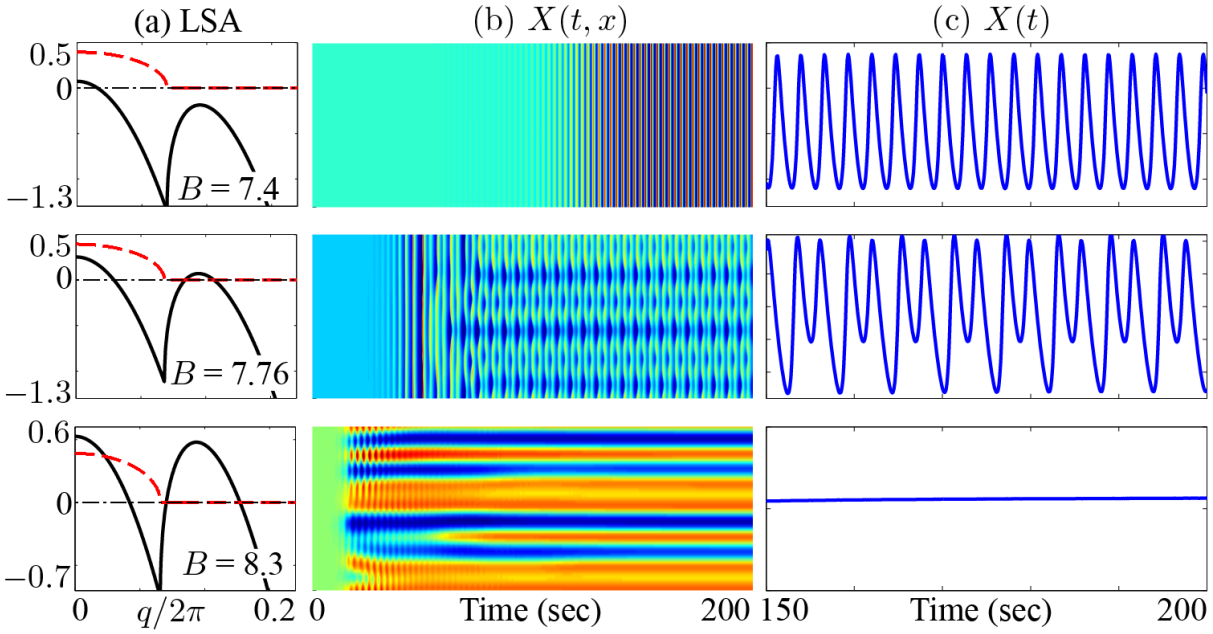


Figure 5.6: Simulations for the Brusselator model when $\Delta > 0$. Simulation starts separatively from three different B values: (1) $B = 7.4$; (2) $B = 7.76$; (3) $B = 8.3$. For each case, we compare the (a) linear stability analysis (LSA) of the Brusselator model (the solid and dashed curves are the real and imaginary parts respectively of dominant eigenvalues with respect to scaled wave numbers); (b) Activator $X(t, x)$ space-time strip charts for a full 200-s simulation (see the description of Fig. 2.4) and (c) Time-series of the final 50 s at the centre grid-point of the 60×60 grid.

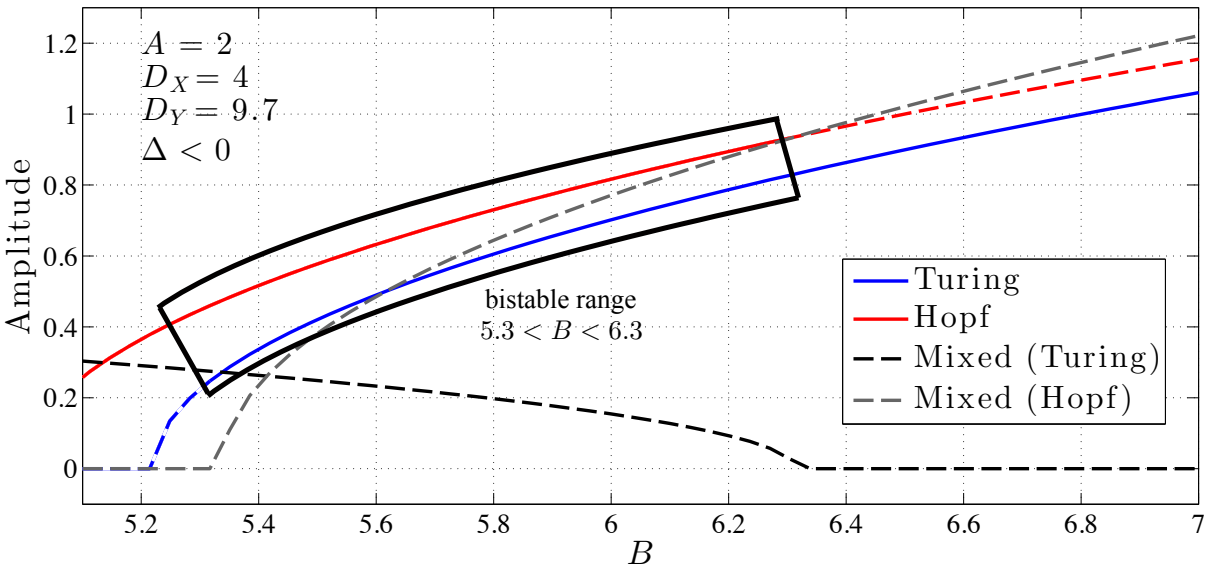


Figure 5.7: Mode stability of the TH mode amplitude equations for the Brusselator model when $\Delta < 0$. Bistable range for Turing and Hopf are found when $5.3 < B < 6.3$. See Fig. 5.5 for the figure properties.

When $\Delta < 0$, the mixed TH mode is always unstable. Comparing the second row of Figs. 5.6 and 5.8, although we see very similar eigenvalue dispersion curves both predicting a TH mode, the simulations show that this mode occurs only when $\Delta > 0$. In Fig. 5.7, when $B = 5.4$, 6.0 or 6.6, LSA predicts a mixed mode between Turing and Hopf. However, the simulations show a single mode only. This phenomenon suggests that one mode eventually becomes dominant over the other, thus being the winner of the mode competition. When $B = 6.6$, the strip-chart exhibits equally spaced bands, implying well-structured spots of the Turing pattern. Further increasing B induces a mode transition following the prediction by the amplitude equation for the hexagonal mode: $H_\pi \rightarrow$ stripes $\rightarrow H_0$.

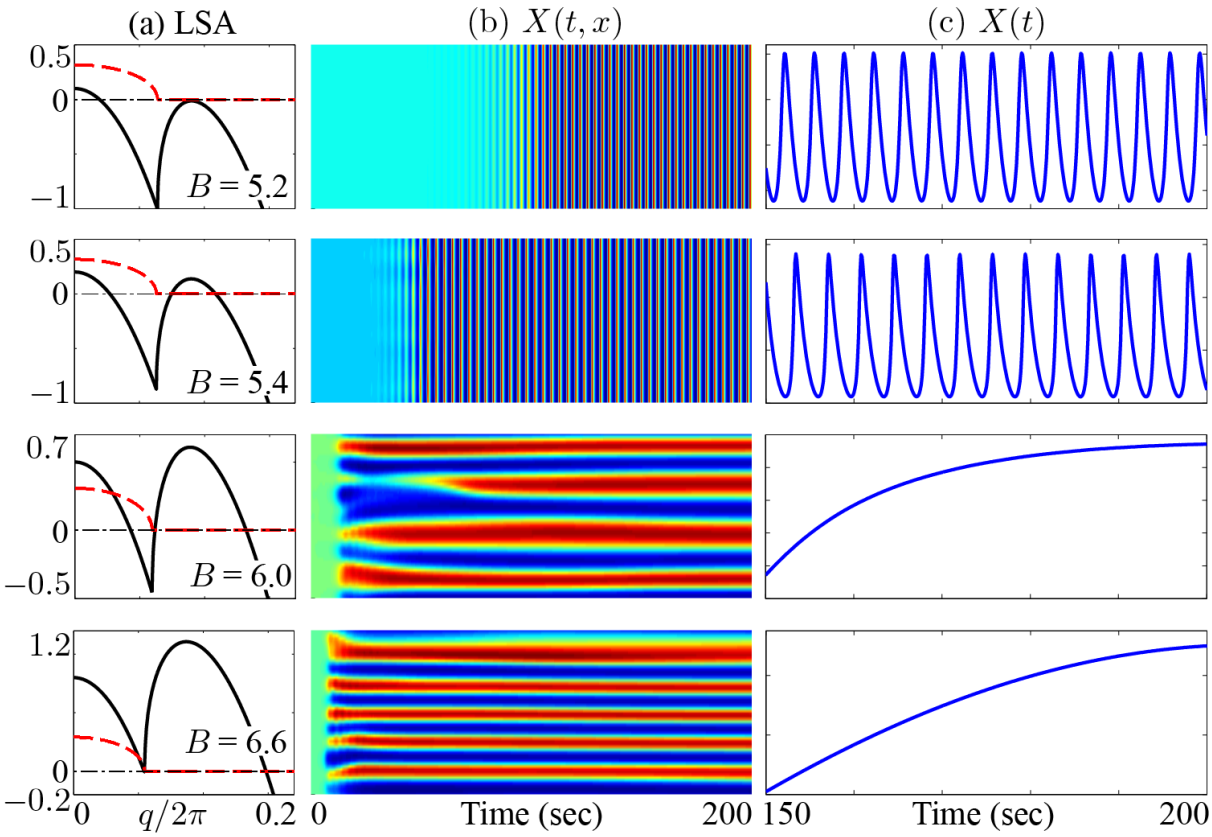


Figure 5.8: Simulations for the Brusselator model when $\Delta < 0$. Simulation starts separatively from four different B values: (1) $B = 5.2$; (2) $B = 5.4$; (3) $B = 6.0$; (4) $B = 6.6$. (a) linear stability analysis (LSA); (b) Activator $X(t, x)$ space-time strip charts for a full 200-s simulation and (c) Time-series of the final 50 s for the centre grid-point of the 60×60 grid.

Although the analysis of the amplitude equations cannot determine the dominant mode at the bistable region, mode predictions outside of this region by the amplitude equations have good agreements with simulations.

Note that above analysis is based on the real amplitude equations, ignoring their imaginary parts. The real and imaginary parts of the amplitude equations represent respectively the modes and their phase. Therefore, the analysis on the complex amplitude

equations predicts the stability of the mode phases. Unstable phases may lead to chaotic evolutions, which we will discuss in the next section.

5.2.3 Analysis of the mixed mode via the complex TH mode amplitude equations

Assuming T is still real but H is complex ($H = H_\mathrm{r} + iH_\mathrm{i}$), introducing the weak perturbations

$$\begin{aligned} T &= T_\mathrm{Ms} + \delta T \\ H_\mathrm{r} &= H_\mathrm{Ms} + \delta H_\mathrm{r} \\ H_\mathrm{i} &= \delta H_\mathrm{i} \end{aligned}$$

to Eqs. (5.17a, 5.17b) leads to the Jacobian matrix with respect to δT, δH_r and δH_i:

$$\begin{bmatrix} -2C_\mathrm{TT}T_\mathrm{Ms}^2 - D_\mathrm{T}k^2 & -2C_\mathrm{TH}T_\mathrm{Ms}H_\mathrm{Ms} & 0 \\ -2C_\mathrm{HTr}T_\mathrm{Ms}H_\mathrm{Ms} & -2C_\mathrm{HHr}H_\mathrm{Ms}^2 - D_\mathrm{Hr}k^2 & D_\mathrm{Hi}k^2 \\ -2C_\mathrm{HTi}T_\mathrm{Ms}H_\mathrm{Ms} & -2C_\mathrm{HHi}H_\mathrm{Ms}^2 - D_\mathrm{Hi}k^2 & -D_\mathrm{Hr}k^2 \end{bmatrix} \tag{5.29}$$

Notice that diffusion terms are included because we consider here T and H depend on both time and space. The amplitude equations have the wavenumber k, which is distinct from the wavenumber q of the original system. The determinant is calculated as:

$$I = -4H_\mathrm{Ms}^2 T_\mathrm{Ms}^2 D k^2 + O(k^3)$$

in which

$$D = D_\mathrm{Hi}(C_\mathrm{HHi}C_\mathrm{TT} - C_\mathrm{TH}C_\mathrm{HTi}) + D_\mathrm{Hr}(C_\mathrm{HHr}C_\mathrm{TT} - C_\mathrm{TH}C_\mathrm{HTr}) \tag{5.30}$$

Using MAPLE, we calculated the explicit forms for three eigenvalues, two of which are complex conjugate pairs with negative real parts. Hence, we have the following conclusion:

$$\text{Amplitude equation system is stable} \quad I < 0 \quad \text{when } D > 0 \tag{5.31a}$$

$$\text{Amplitude equation system is unstable} \quad I > 0 \quad \text{when } D < 0 \tag{5.31b}$$

By forcing parameters related to the coupling coefficients ($C_\mathrm{TH}, C_\mathrm{HTi}, C_\mathrm{HTr}$) to zero, the standard Benjamin-Feir instability criterion of a homogeneous limit cycle is recovered [23]:

$$D^\mathrm{BF} = D_\mathrm{Hi}C_\mathrm{HHi} + D_\mathrm{Hr}C_\mathrm{HHr} < 0 \tag{5.32}$$

However, it is important to note that the inequality (5.31b) can be satisfied even when the Benjamin-Feir condition (5.32) is not fulfilled, i.e., stable limit cycle coupled with spatial instability (Turing bifurcation).

Chaos at $\Delta > 0$

When $\Delta > 0$, we found certain parameter settings lead to chaotic patterns (continuous competition between Turing and Hopf). Here, we will investigate the pattern dynamics with two such sets of parameters: Setting (1) $A = 2.5, B = 7.9, D_X = 5, D_Y = 10$; Setting (2) $A = 2.5, B = 8, D_X = 4.49, D_Y = 8.91$ (following de Wit's work [24]).

Mode stability prediction Fig. 5.5 shows that setting (1) is just beyond the right edge of the mixed mode region, falling into the stable Turing region. Setting (2) has the same prediction (not shown here). However, the simulation results for both settings, demonstrated in Fig. 5.9, are spatiotemporal patterns whose time-series appear to be turbulent and chaotic.

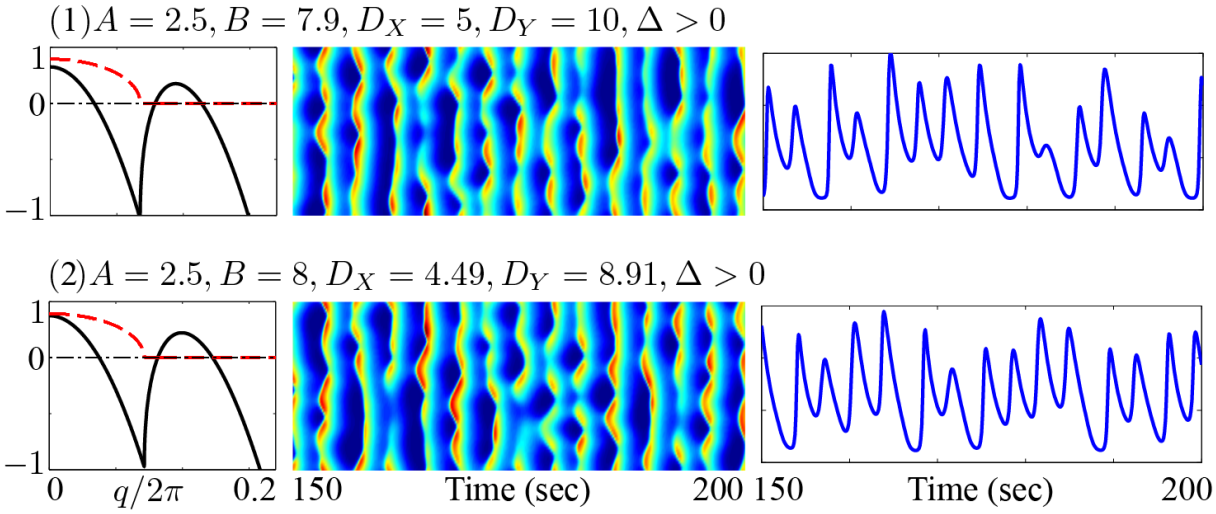


Figure 5.9: Chaotic mode of the Brusselator generated from two sets of parameters. Both settings satisfy $\Delta > 0$, which leads to a region of stable mixed TH mode (see Fig. 5.5). See Fig. 5.7 for figure properties.

Is there a way to predict such chaotic mode? de Wit responded to our enquiry as follows:

"*Chaotic dynamics arises from desynchronisation of the phases of the oscillators (i.e., Benjamin-Feir instability). The related criterion is given by looking at the stability of the Turing and Hopf modes with regard to perturbations of the phase of the Hopf mode (Eqs. (5.31a, 5.31b)).*"

Following de Wit's suggestion, we then examined TH mode stability conditions (5.31a), (5.31b) and (5.32) with both sets of parameters. Surprisingly, both settings (1) and (2) give negative I and positive D, which suggests a stable TH mode. This prediction conflicts with the observed chaotic patterns seen in Fig. 5.9.

How did de Wit find the parameter settings [24] for the chaotic patterns? She explained:

"*I did not use all these criteria to find parameters for simulations in the Brusselator. There are on one hand simulations using the Brusselator in which I just vary parameters*

and find various dynamics. Then ***in parallel and totally independently*** (bolded by the thesis author)*, I have integrated amplitude equations on their own. But I do not connect the two. I think there are anyway corrections at higher orders that are neglected in the derivation of the amplitude equations that make them not matching quantitatively the full nonlinear model like the Brusselator.*"

From de Wit's statement, it is important to note that the phase stability condition (5.31a, 5.31b) and (5.32) are only applicable to the system of amplitude equations[2], in order words, they may not be applicable to the original system.

Although we have not found theoretical criteria to directly predict the Brusselator chaotic patterns, there are clues to locate the parametric space where the chaos may occur: chaotic mode exists when $\Delta > 0$; chaotic mode is available within or near the stable TH region. In the vicinity of the TH region, the competition between the Turing and Hopf modes is most likely to transit to a chaotic evolution, resulting from the nonlinearity and possibly the unstable Hopf phases.

5.2.4 Discussion

To analyse a pattern-forming system, the linear stability analysis (LSA) is first applied to examine the stability of the steady-states with respect to weak perturbation, which predicts the emergent mode (first bifurcation). After sufficient time, the system may evolve into a nonlinear region where the LSA may not be applicable, meanwhile, the previous emergent mode may experience another instability bifurcating to a more stable state (second bifurcation). Such complicated mode stability may be predicted via the eigenvalue analysis of the amplitude equations.

In Chapter 4, we introduced a multiple-scale expansion to capture the subtle mode dynamics near a bifurcation point (i.e., at a point where new dynamical behaviour is about to emerge). The product of this nonlinear expansion is the so-called amplitude equations which we utilised to analyse the mode stability in this chapter.

Our analysis of the Brusselator amplitude equations for the hexagonal mode shows that the Turing pattern of the Brusselator model can be spatially structured to honeycomb (H_π), stripes or reentrant honeycomb (H_0) modes. Fig. 5.3 shows good agreements between the mode predictions by the amplitude equations and simulations. It is noticed that the H_0 structure in Fig. 5.3 is only roughly reentrant in shape with H_π, but is not its "upside down" version. It is always true that the H_0 cannot be the perfect reentrant of H_π by only tuning the bifurcation parameter B. There are two reasons for this phenomenon:

1. Increasing the bifurcation distance $(B - B_0)$ introduces strong nonlinearity, which coupled with the effect from squeezed hexagonal mode will distort the H_0 hexagons.

[2]See examples of predicting chaotic modes for the amplitude equations in [23].

2. Larger B will shift the peak of the eigenvalue dispersion curve to the right, which implies a larger wavenumber (i.e., smaller wavelength) for the most unstable mode. Thus the H_0 mode has a higher spot density than the H_π mode.

To obtain a well-structured H_0 mode, one needs to maintain a small $(B - B_0)$ distance as well as tuning the parameter A. For example, to produce patterns in Fig. 5.10, we only tuned the value of A. Reducing the value of A is equivalent to increasing B, as well as shortening the distance from the the Turing threshold to the H_0 region. Thus this tuning will lead the model to the H_0 region without changing significantly the wavenumber of the most unstable mode.

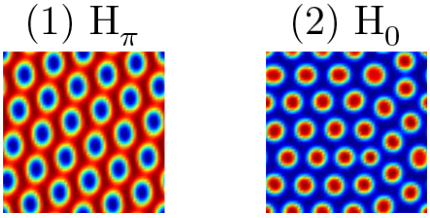


Figure 5.10: Demonstrated H_π and H_0 modes generated by (1) $A = 3, B = 9, D_X = 5, D_Y = 12$ and (2) $A = 2, B = 9, D_X = 5, D_Y = 12$. The Turing conditions for setting (1) is $B_0 = 8.623$, for setting (2) is $B_0 = 5.249$.

After investigating the Brusselator Turing patterns, we then moved our attention to the Brusselator interacting Turing–Hopf patterns. To assist this analysis, we derived the TH mode amplitude equations in Appendix G. The TH mode amplitude equations have an argument Δ (see Eqs. (5.20a, 5.20b)) whose sign determines if the TH mode is stable: stable when $\Delta > 0$; unstable when $\Delta < 0$. When $\Delta < 0$, we found that the TH mode amplitude equations will experience a bistability, where the Brusselator evolves to either a Turing or Hopf mode. It should be noticed that the bistable mechanism for the TH mode is distinct from the bistable Turing structures since Fig. 5.3 shows that multiple Turing modes can coexist at the bistable range. When $\Delta > 0$, a chaotic mode exists in the vicinity of the stable TH region: At the onset of a stable TH mode, finely tuning the bifurcation parameter will cause the Hopf phases to lose stability, leading to a chaotic mode. From our discussions with de Wit, the phase stability criteria (5.31a, 5.31b) and (5.32) are only applicable to the system of the amplitude equations. In order to find the chaotic mode, it is suggested to manually scan the parametric space close to a codimension-2 point. Our amplitude equations-based mode predictions have better agreement than the linear stability analysis with numerical simulations.

Generally speaking, predicting the Turing–Hopf dynamics is more challenging than the pure Turing dynamics since the Turing–Hopf interference induces a strong nonlinearity and interplay among multiple modes. Our TH mode analysis is based on the simplest structure ansatz with one wavelength q_c and one frequency ω_c:

$$\mathbf{u_1}^{\mathrm{TH}} = A_{\mathrm{T}} e^{i\vec{q}_c \cdot \vec{r}} \mathbf{R_T} + A_{\mathrm{H}} e^{i\omega_c t} \mathbf{R_H} + c.c.,$$

which is a preliminary investigation in the mechanisms of the pattern selection and competition. A comprehensive understanding of the TH dynamics may require the derivations of the amplitude equations based on other structure ansatzes describing the mode interactions, e.g., subharmonic modes [22, 24]:

- Subharmonic instability of a Turing mode
  Near a CTHP, a Turing mode can give rise to subharmonic latticed patterns oscillating in time and generated by subharmonic instabilities. Such subharmonic instability is the combination of a steady structure with wave number q_c and of a standing wave formed by the superposition of the left- and right-travelling waves (A_R and A_L) with wave number $q_c/2$ and frequency $\omega_q = \omega_{q_c/2}$. The structure ansatz has the expression:

$$\mathbf{u_1}^{\mathrm{subT}} = A_T e^{i\vec{q_c}\cdot\vec{r}}\mathbf{R_T} + A_L e^{i[\omega_{q_c/2}t+(\vec{q_c}/2)\cdot\vec{r}]}\mathbf{R_L} + A_R e^{i[\omega_{q_c/2}t+(\vec{q_c}/2)\cdot\vec{r}]}\mathbf{R_R} + c.c. \quad (5.33)$$

  where $\mathbf{R_L}$ and $\mathbf{R_R}$ are the critical eigenvectors corresponding to the left- and right-travelling waves of the wave number $q_c/2$ and frequency $\omega_{q_c/2}$. Such a structure ansatz yields three amplitude equations about the Turing mode A_T, left-travelling wave A_L and right-travelling wave A_R. By introducing linear perturbations $A_T = A_{T_0} + \delta A_T$, $A_R = A_L = (A_{RL_0} + \delta A_{RL})e^{i\Omega t}$, one can examine the stabilities of the Turing mode δA_T or travelling-wave mode δA_{RL} by a standard linear stability analysis of the amplitude equations.
- Subharmonic instability of a Hopf mode
  The resonance between a homogeneous temporal oscillation with frequency ω_c and of a standing wave with frequency $\omega_c/2$ and wave number $q_\omega = q_{\omega_c/2}$ lead to a subharmonic Hopf mode with one wave number and two frequencies. The resulting dynamics is then a pattern with one wave number oscillating with two frequencies. The structure ansatz has the expression:

$$\mathbf{u_1}^{\mathrm{subH}} = A_H e^{i\omega_c t}\mathbf{R_H} + A_L e^{i[(\omega_c/2)t+\vec{q}_{\omega_c/2}\cdot\vec{r}]}\mathbf{R_L} + A_R e^{i[(\omega_c/2)t-\vec{q}_{\omega_c/2}\cdot\vec{r}]}\mathbf{R_R} + c.c. \quad (5.34)$$

  which yields three amplitude equations about A_H, A_L and A_R. The linear perturbation $A_H = (A_{H_0}+\delta A_H)e^{i\varphi t}$ and $A_R = A_L = (A_{RL_0}+\delta A_{RL})e^{i\phi t}$ allow a linear stability analysis for examining the respective mode stability.
- Subharmonic instability of a Turing–Hopf mode
  Near a CTHP, a subharmonic Hopf mode characterised by the wavenumber $q_{\omega_c/2}$ could resonate with the Turing mode of wavenumber q_c if $q_c = 2q_{\omega_c/2}$. In this case, the structure ansatz reads:

$$\begin{aligned}\mathbf{u_1}^{\mathrm{subTH}} =& A_H e^{i\omega_c t}\mathbf{R_H} + A_L e^{i[(\omega_c/2)t+\vec{q}_{\omega_c/2}\cdot\vec{r}]}\mathbf{R_L} \\ &+ A_R e^{i[(\omega_c/2)t-\vec{q}_{\omega_c/2}\cdot\vec{r}]}\mathbf{R_R} + A_T e^{i\vec{q_c}\cdot\vec{r}}\mathbf{R_T} + c.c.\end{aligned} \quad (5.35)$$

where the amplitudes obey a set of four coupled amplitude equations describing a spatiotemporal dynamic with two wave numbers $q_{\omega_c/2}$ and $2q_{\omega_c/2}$ and two frequencies ω_c and $\omega_c/2$.

In the next section, we will investigate the mode stability of the Waikato cortical model via the theories of pattern dynamics we developed from the Brusselator model.

5.3 Mode stability analysis for the Waikato cortical model

The amplitude equations derivation for the Waikato cortical model follows the multiple-scale expansion (MSE) flowchart Fig. 4.3; however, the complexity of the model (high dimensions) allows only numerical implementation of MSE. The cortical model has more than twenty parameters (Brusselator model has only four), and the implicit combinations of them form the steady-state (SS), critical eigenvalues and eigenvectors, which can only be found numerically. These numerical terms will further construct the order solutions of the MSE and eventually merge into the coefficients of amplitude equations. To our knowledge, this is the first attempt to apply MSE numerically to a system as complicated as the cortical model. Such a procedure may raise two major difficulties:

1. The numerical errors in each MSE step may accumulate;
2. We are not able to obtain explicit expressions for the coefficients in the amplitude equations.

As the amplitude equation is a universal form only relating to specific modes but not the original model, the amplitude equations for the Brusselator and the cortical model will share the same expressions. Due to the simplicity of the Brusselator model (or other commonly seen reaction–diffusion systems like Gray [49], Leygyel-Epstein [104] or Belousov-Zhabotinsky [150] models), all the coefficients of the amplitude equations can be derived precisely in analytical expressions; this enables us to examine the dependence of mode dynamics on coefficients by tuning any model parameter (A, B, D_X or D_Y). In comparison, the numerical implementation of MSE leads to amplitude equations for the cortical model only explicit to bifurcation distance for the Turing ($D_2 - D_2^c$) and Hopf ($\gamma_i - \gamma_i^c$). In other words, ($D_2 - D_2^c$) and ($\gamma_i - \gamma_i^c$) are the only terms we can tune in the amplitude equations. Thus we are not able to investigate how other cortical parameters form the coefficients and affect the mode stabilities of the amplitude equations. For example, the subcortical drive s is a control parameter for the cortical SS distribution (see Fig. 3.5), where each s-dependent SS may have distinct stabilities. Since we cannot vary the subcortical drive s in the amplitude equations (s is fixed to derive the SS numerically, and cannot be identified with a particular coefficient), we are only allowed to derive

amplitude equations for the cortical model at a specific s of interest. Then we can examine the mode stability of this particular s with respect to bifurcation parameters.

The above-mentioned numerical operations of MSE on the cortical model require us to first scan the cortical parametric space in order to find a "special" coordinate where the mode stability experiences a bifurcation from the simulation. Then we apply the MSE on this coordinate to derive the amplitude equations for a better understanding of the mode dynamics.

5.3.1 Hexagonal mode of the cortical model

In this section, we will investigate various Turing patterns of the cortical model, then discuss their stabilities by the amplitude equations.

We first ran a series of simulations starting from selected SS from the Q_e^o SS-distribution (see Fig. 5.11) of SR2007 model of the cortex. We intended to find the effect of subcortical drive s on the spatial structure of cortical Turing patterns. We selected nine SS from the $Q_e^o(s)$ curve: A to C are from the bottom branch; D is from the middle branch; E to I are from the top branch. C, D and E are the multiroots of the same subcortical drive $s = 0.25$. All simulations share the same parameter settings except the starting SS points and bifurcation parameters. LSA shown in Fig. 3.5 implies that simulations starting from the SS single-root zone at the top branch requires a rather larger gap-junction strength D_2 to trigger the Turing instability. So H and I at the top branch have slightly larger $D_2 = 1.7$ cm^2 than A to G's setting $D_2 = 1.3$ cm^2. The other bifurcation parameter γ_i is set to 35 ensuring a sufficient distance to its Hopf threshold (ranging from 25 to 29 for all s), so that only Turing mode is expected to be dominant.

Fig. 5.11 shows G is a mode-transition point, before which ($s < 0.5$) Turing patterns have the stripes structure; after which ($s \geq 0.5$) Turing patterns have the H_π structure. We had tested other SS for $0 < s < 0.5$ with various D_2 values, which all gave rise to stripes. The SS-curve of Fig. 5.11 covered in grey indicates that simulations starting here will evolve to stripes of the Turing pattern.

When $s \geq 0.5$ (SS-curve in Fig. 5.11 covered in black), moderately strong D_2 and appropriate setting of γ_i in favour of a H_π structure of the Turing patterns. In the following we will investigate the stability of the H_π mode with different D_2 and γ_i at the mode transition point $s = 0.5$ where the critical setting $[D_2^c, \gamma_i^c] = [1.2159, 29.9999]$ leads to a CTHP. First, we examined the sensitivity of the H_π mode with respect to γ_i. With D_2 fixed at 1.3 cm^2, the LSA in Fig. 5.12 demonstrates that decreasing γ_i from 60 to 30 raises the Hopf instability dramatically (real part of the dominant eigenvalue at $q = 0$ is getting less negative) and the Turing instability slightly, which accordingly gives rise to larger growth rate of the Turing instability. When $\gamma_i = 30$, we see the best structured H_π

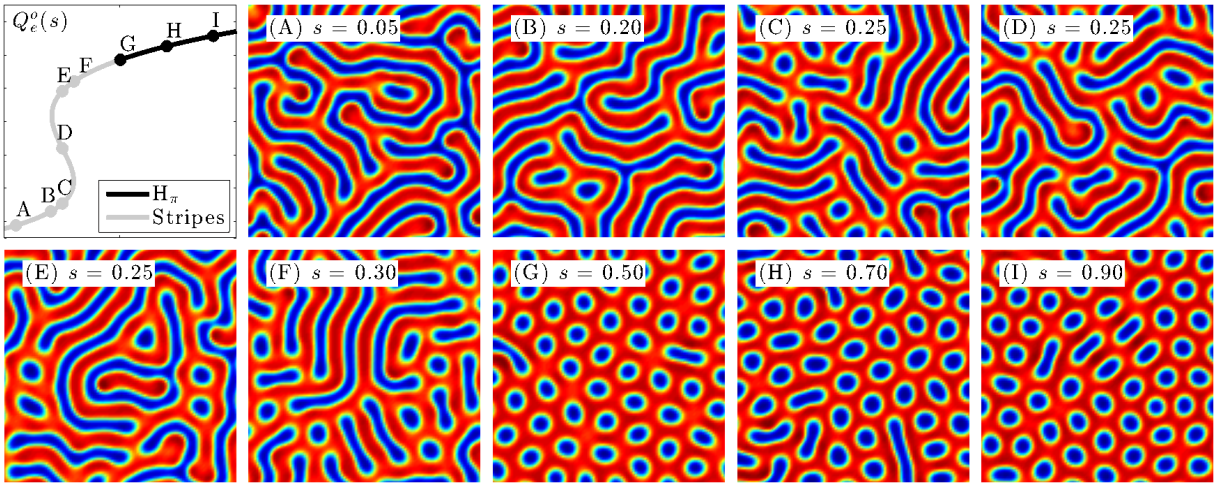


Figure 5.11: Cortical simulations of the SR2007 model starting from nine selected SS (A to I) of the Q_e^o SS-curve. Q_e^o is a function of the subcortial drive s ranging from 0 to 1. $Q_e^o(s)$ and $V_e^o(s)$ (Fig. 3.5) are linked via the sigmoidal mapping of function Eq. (3.2). Simulations starting from A to G are set with $[D_2, \gamma_i] = [1.3, 35]$, from H to I are set with $[D_2, \gamma_i] = [1.7, 35]$. D_2 is set to be sufficiently large to trigger the Turing instability; γ_i is set to maintains a sufficient distance to its critical value for supressing the Hopf instability. 30-s simulation shows that A to F evolve to stripes of the Turing pattern; G to I evolve to a H_π structure (see Fig. 5.3 for reference) of the Turing pattern. Simulated 2D cortical tissue has a side length 25 cm with 100×100 grid-points.

mode where both simulated patterns and the Fourier spectrum show spatially periodic hexagons.

In our next investigation, we fixed $\gamma_i = 30$, then gradually increased the bifurcation distance of D_2 from its critical value 1.2159 cm^2 to observe the corresponding pattern dynamics. Fig. 5.13 illustrates simulated patterns by increasing D_2 from 1.23 that is only 1.15% over its critical value to 3.20 that is 62% over its critical value. Clearly, the smallest D_2 experiences the best structured H_π mode. By increasing D_2, the H_π mode will become unstable then eventually be replaced by the more stable stripes structure.

Until now we have located the parameter setting $(D_2, \gamma_i) = (1.23, 30)$ leading to the best structured H_π mode of the Turing pattern for the SR2007 model. D_2 and γ_i are within 1.15% and 3.33×10^{-2}% of their critical values, respectively. Theoretically we could set D_2 even closer to D_2^c for a better H_π structure, however, the growth rate for the mode would be too small to obtain a stable mode in a reasonable time frame. On the other hand, once γ_i crosses its bifurcation threshold, a temporally unstable Hopf instability will be induced into the Turing patterns, thus breaking their temporal stabilities. As a result, we must keep $(\gamma_i - \gamma_i^c) > 0$ to suppress the Hopf instability for static Turing patterns.

From Figs. 5.12 and 5.13, we see that D_2 plays a major role in mode stability of the Turing patterns. To precisely examine how D_2 affects the Turing structures, we applied the multiple-scale expansion (MSE) on the cortical model at $s = 0.5$ to derive the amplitude equations for the hexagonal mode. In our mode stability analysis, we varied

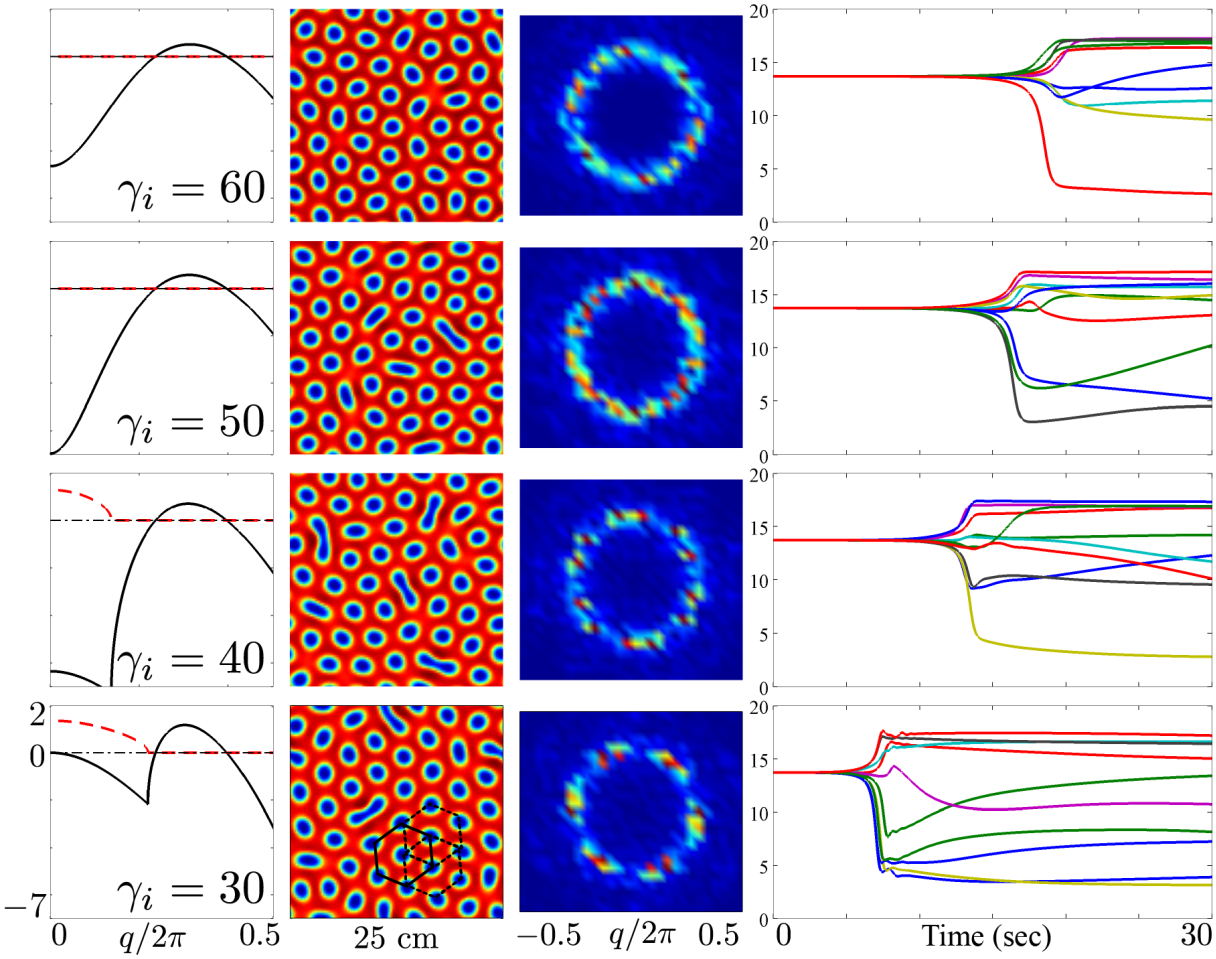


Figure 5.12: 30-s cortical simulations starting from G ($s = 0.5$) in the SS-curve of Fig. 5.11 with γ_i is 60, 50, 40 or 30 and fixed D_2 at 1.3 cm^2. At each panel, from left to right there are LSA predicting curves, patterns formed at 30 s, its 2D Fourier amplitude spectrum and time-series extract of 10 equally spaced grid-points along the vertical axis at the centre of the horizontal axis. Best H_π patterns are structured at $\gamma_i = 30$.

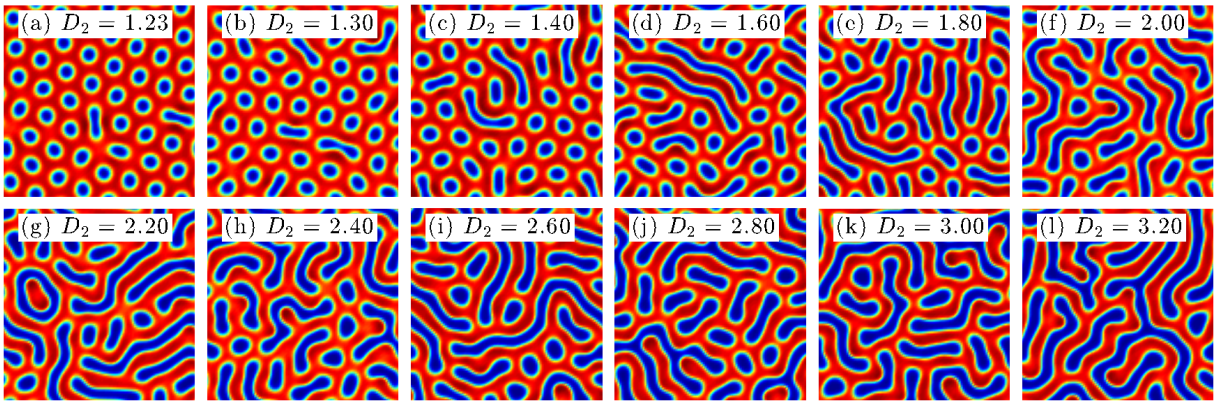


Figure 5.13: Patterns formed from 60-s cortical simulations starting from G ($s = 0.5$) in the SS-curve of Fig. 5.11 with increased D_2 and fixed $\gamma_i = 30$. Best H_π patterns are structured at $D_2 = 1.23$. Here, both D_2 and γ_i are extremely close to their bifurcation thresholds.

D_2 but kept γ_i fixed at 30. According to theories of mode stability for the Brusselator model in Sec. 5.1, we derived mode stability diagram 5.14 for the cortical model.

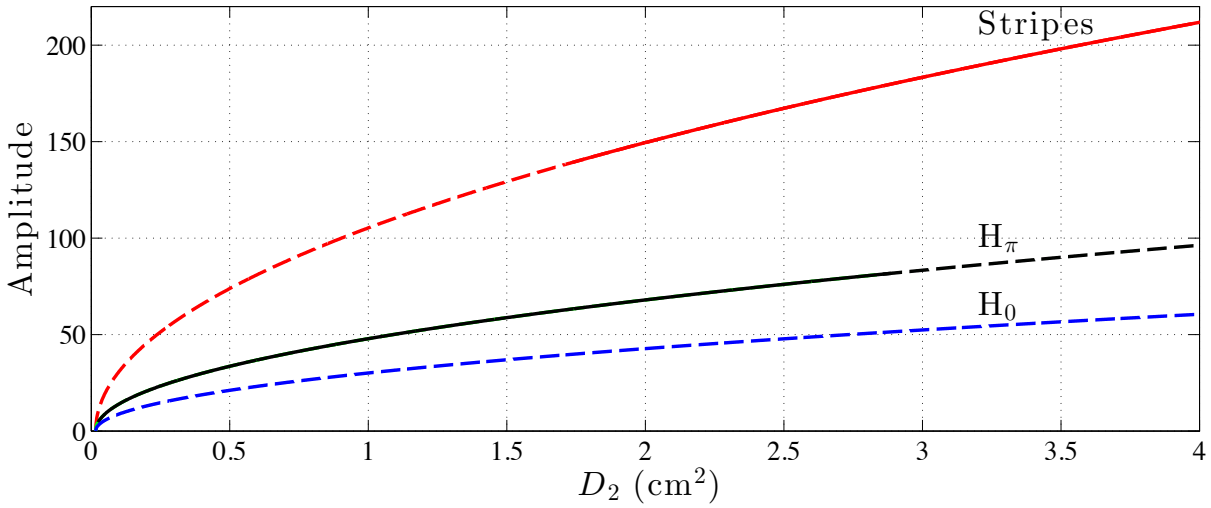


Figure 5.14: Mode stabilities of the Turing pattern for the SR2007 model when subcortical drive $s = 0.5$. γ_i is fixed at 30. The Turing pattern of the SR2007 model possesses three modes: stripes (red), H_π (black) and H_0 (blue). The stability of the mode at each D_2 is examined via eigenvalue analysis (see Sec. 5.1.2): solid curve — stable mode; dashed curve — unstable mode.

Clearly shown in Fig. 5.14, H_π is dominant when $D_2 < 1.7$. Stripes and H_π are mixed when $1.7 < D_2 < 2.8$. Stripes become dominant when $D_2 > 2.8$. H_0 is unstable across the whole range of D_2. Such theoretical predictions from the amplitude equation are in good agreement with the experimental observations in Fig. 5.13. Besides $s = 0.5$, we have also examined other settings when $0.5 < s < 1$: The trend of the mode stabilities with respect to D_2 keeps the same as shown in Fig. 5.14.

With reference to Fig. 5.3, increasing the Turing bifurcation distance introduces a mode transition from H_π to stripes first, then to H_0. However, we could not find the H_0 structure for the SR2007 model. When the subcortical drive $0.5 < s < 1$, we comprehensively scanned the (D_2, γ_i) space for the H_0 mode without any success. Once the cortical patterns have reached stripes, they become absolutely dominant no matter how large the setting for D_2.

Since we could not find the existence of the H_0 patterns in the SR2007 model, we switched our attention to the SR2009s model (another slow-soma limit, see Table 3.4 for its definition) of the cortex. The SR2009s model we introduced in Sec. 3.3.2 is different from the SR2007 model mainly in two aspects:

1. It uses two different sets of cortical parameters;
2. The PSP is described either by the alpha-function Eq. (3.7) in SR2007 model or the biexponential function Eq. (3.22) in SR2009s model.

In Sec. 3.2.2, we discussed the different dynamics of the two models of the cortex. Fig. 3.8 drew to our attention that strong inhibitory diffusion D_2 leads to H_0-like patterns:

red spots embedded in the blue background, which is very similar to the H_0 patterns (bullet structure) of the Brusselator model seen in Fig. 2.3. However, we could not find well-constructed H_0 patterns of the slow-soma model after comprehensively scanning the parametric space. Later, in an experiment testing the effect of the alpha-function on the SR2009s model (this experiment follows our earlier work in [139]), we successfully found the H_0 patterns, shown in Fig. 5.15:

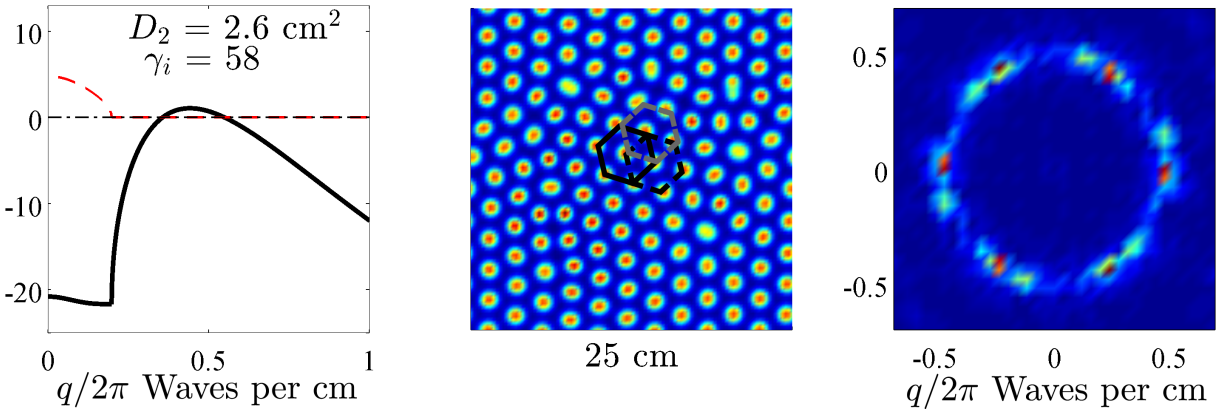


Figure 5.15: H_0 patterns generated by the SR2009s model with PSP described by the alpha-function. Turing bifurcation parameter D_2 is set to 2.6 cm^2 that leads to a weak Turing peak of the LSA predicting curve. Hopf bifurcation parameter γ_i is set to 58 where LSA predicts a highly damped Hopf instability. The simulated 25-cm cortical tissue (100×100 grid) exhibits well-structured H_0 hexagonal patterns. We sketched three adjoint hexgons in the patterns. Such consecutive hexagonal structure of the patterns is revealed in the 2D Fourier spectrum.

Then we investigated the stabilities of the H_0 mode with respect to varying D_2 and γ_i. As shown in Fig. 5.15, a stable H_0 mode is found when D_2 is 2.6 cm^2 that is very close to its threshold (Turing peak is very close to the dash-dotted zero line). Therefore we first examined the stability of the H_0 mode by increasing D_2 to 3.0, 3.8 and 4.0 correspondingly at 15.39%, 46.15%, 53.85% above $D_2 = 2.6$, respectively with γ_i fixed at 58. In Fig. 5.16, a top to bottom scan of the LSA predicting curves shows that the Turing instability is enhanced by increasing D_2, meanwhile, the Hopf instability at $q = 0$ is highly damped at all D_2 choices. Surprisingly, by examining seven snapshots captured within 1.2 s, we see localised oscillations (at the red spots) for the H_0 mode. When $D_2 = 3.0$, the pattern repeats its spatial structure every 0.78 s: Snapshots captured at 16.0994 and 16.8794 s are similar, so are snapshots captured at 15.8498 and 16.6298 s. However, the oscillatory patterns for $D_2 = 3.8$ and 4.0 are irregular: the red spots "light" and "dim" randomly in time and space.

Next, we fixed D_2 at 2.6 cm^2 then investigated the pattern dynamics by decreasing γ_i. As we have learned from Fig. 5.12, decreasing γ_i will enhance the Hopf instability significantly and boost the Turing instability slightly. Examining the LSA predicting curves in Fig. 5.17, $\gamma_i = 46$ experiences a single Turing peak; when $\gamma_i = 28$ or 38, it

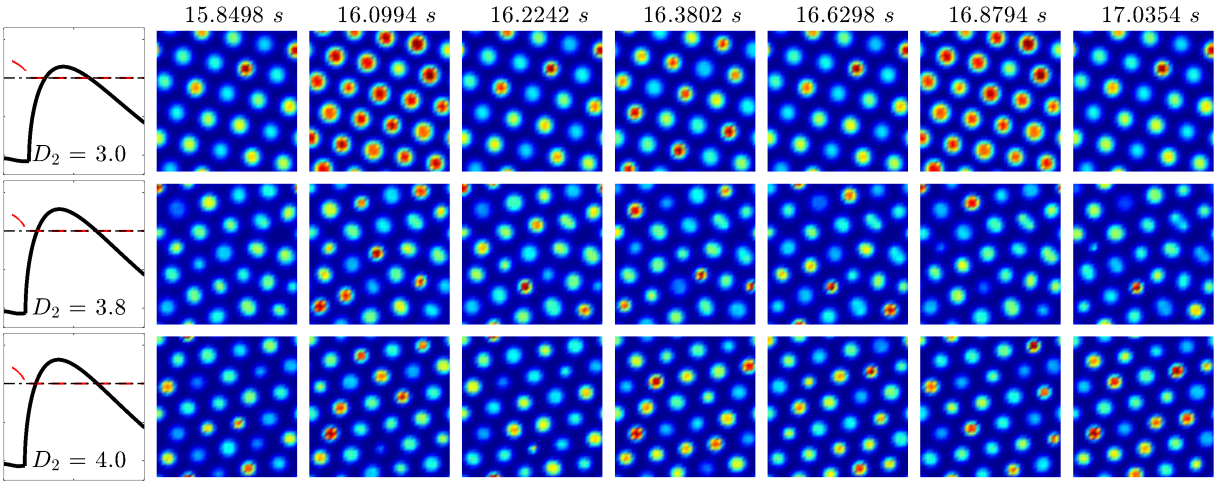


Figure 5.16: With fixed $\gamma_i = 58$, simulation snapshots for three distinct $D_2 = 3.0, 3.8$ and 4.0 cm^2 of the SR2009s model with PSP described by the alpha-function. LSA predicting curve for each D_2 setting is listed in the first column. Simulated cortical tissue has a side-length 10 cm and 40×40 grid.

has crossed its Hopf critical point, so a TH mode is predicted. When $\gamma_i = 46$, we found localised oscillating patterns without significant temporal periodicity, that is, repeated patterns with the same spatial structure are not observed within a certain time-period. When $\gamma_i = 38$ or 28, we observed not only the periodically emerged patterns, but also regular transitions between two modes:

- At $\gamma_i = 38$, the first snapshot at 15.9434 s shows equally spaced red spots (high firing-rate zone) decorated in the blue background (low firing-rate zone). After 0.0936 s, the second snapshot shows the spot mode has locally evolved to the "ring" mode. Notice that this mode will return to the spot structure at 16.1930 s, but the position of spots are shifted, which illustrates a fact that the red spots and blue background in the first snapshot ($t = 15.9434$ s) exchange contrast in the third snapshot ($t = 16.1930$ s). Then these spots will locally transit to rings, seen in the fourth snapshot ($t = 16.2866$ s), which take place in the blue area of the second snapshot. We may group the first four snapshots as a complete cycle of mode transitions, in which two patterns with exchanged contrast (either spot to spot or ring to ring modes) are in precisely 0.2496-s interval; the mode transition from spots to rings takes 0.0936 s, and from rings to spots takes 0.1560 s. The complete cycle of mode transitions repeats every 0.3432 s.
- At $\gamma_i = 28$, we also found the consecutive mode transition between the thin (first snapshot) and thick (second snapshot) stripes. Meanwhile, patterns with exchanged contrast can be observed, e.g., the second and fourth snapshots. Here, the complete cycle of mode transitions (thin to thick stripes, then return to thin and thick stripes at shifted positions) repeats precisely every 0.5926 s (15.9434 s to 16.536 s is a complete cycle).

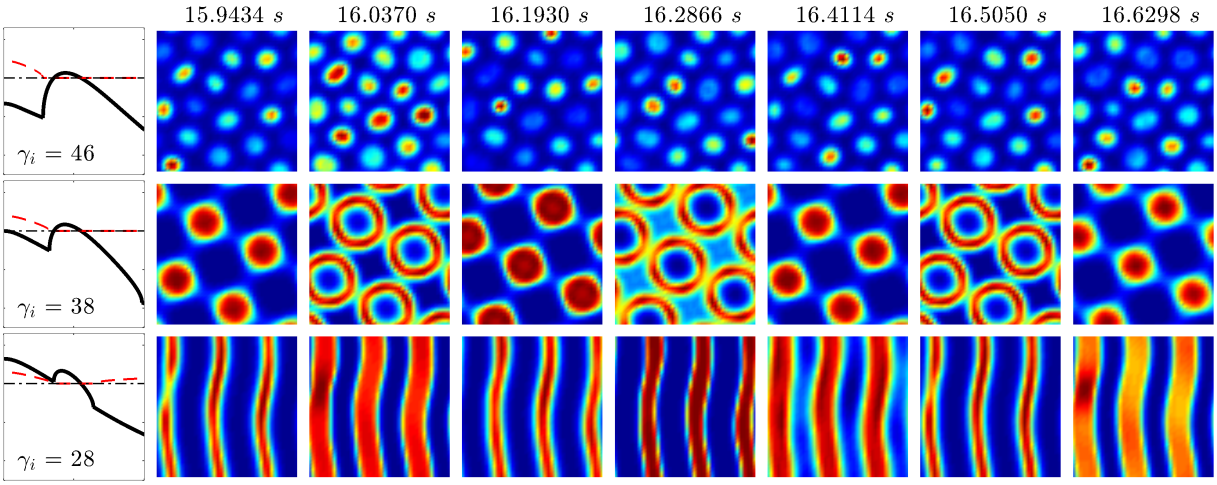


Figure 5.17: With fixed $D_2 = 2.6$ cm^2, simulation snapshots for three distinct $\gamma_i = 46$, 38 or 28 of the SR2009s model with PSP described by the alpha-function. LSA predicting curve for each γ_i setting is listed in the first column. Simulated cortical tissue has a side-length 10 cm of 40×40 grid.

In Figs. 5.16 and 5.17, by increasing D_2 or decreasing γ_i, we see that the enhanced Turing or Hopf instabilities will break the temporal stability of the H_0 mode of the Turing patterns, leading to localised oscillations and mode transitions. Fig. 5.18 clearly demonstrates that the Turing peak at a nonzero wavenumber is raised by increasing D_2, and the Hopf peak at the zero wavenumber is raised by decreasing γ_i. LSA also predicts a single Turing instability when $\gamma_i \geq 42$ and a TH instability when $\gamma_i \leq 38$.

To better examine the pattern dynamics relating to the choice of D_2 and γ_i, we made an animated gallery[3] with 64 windows showing individual simulations for a specific (D_2, γ_i) coordinate in Fig. 5.18. Fig. 5.19 shows four frames of the animation. The first column of the frame ($D_2 = 2.6$) shows the H_0 mode is temporally stable when $\gamma_i \geq 50$. When $38 \leq \gamma_i \leq 46$ meanwhile $D_2 \geq 3.0$, the mode dynamics are chaotic, the observed patterns of which do not have significant periodicities (the gif animation shows better visibility). When the Hopf instability coincides with the Turing instability (see the last two rows of Fig. 5.18), LSA predicted TH mode is manifest with dynamical stripes in simulations. Fig. 5.19 exhibits cortical simulations driven with one-off noise. We also tested the robustness of the cortical dynamics with continuous noise[4], the results of which show little difference with the one-off noise simulations.

The cortical H_0 patterns with localised oscillations is similar to the so-called "twinkling-eyes" patterns, first seen in a reaction–diffusion experiment involving the chlorite-iodide-malonic acid (CIMA) reaction in a thin layer gel reactor [37]. Yang *et al.* argued that

[3]The 10-s animation (last 10 s of the 20-s simulation) can be downloaded as a .gif file at `http://www2.phys.waikato.ac.nz/~kw89/spots_noiseless.gif`

[4]The 10-s simulation with contineous noise can be downloaded as a gif animation file at `http://www2.phys.waikato.ac.nz/~kw89/spots_noise.gif`

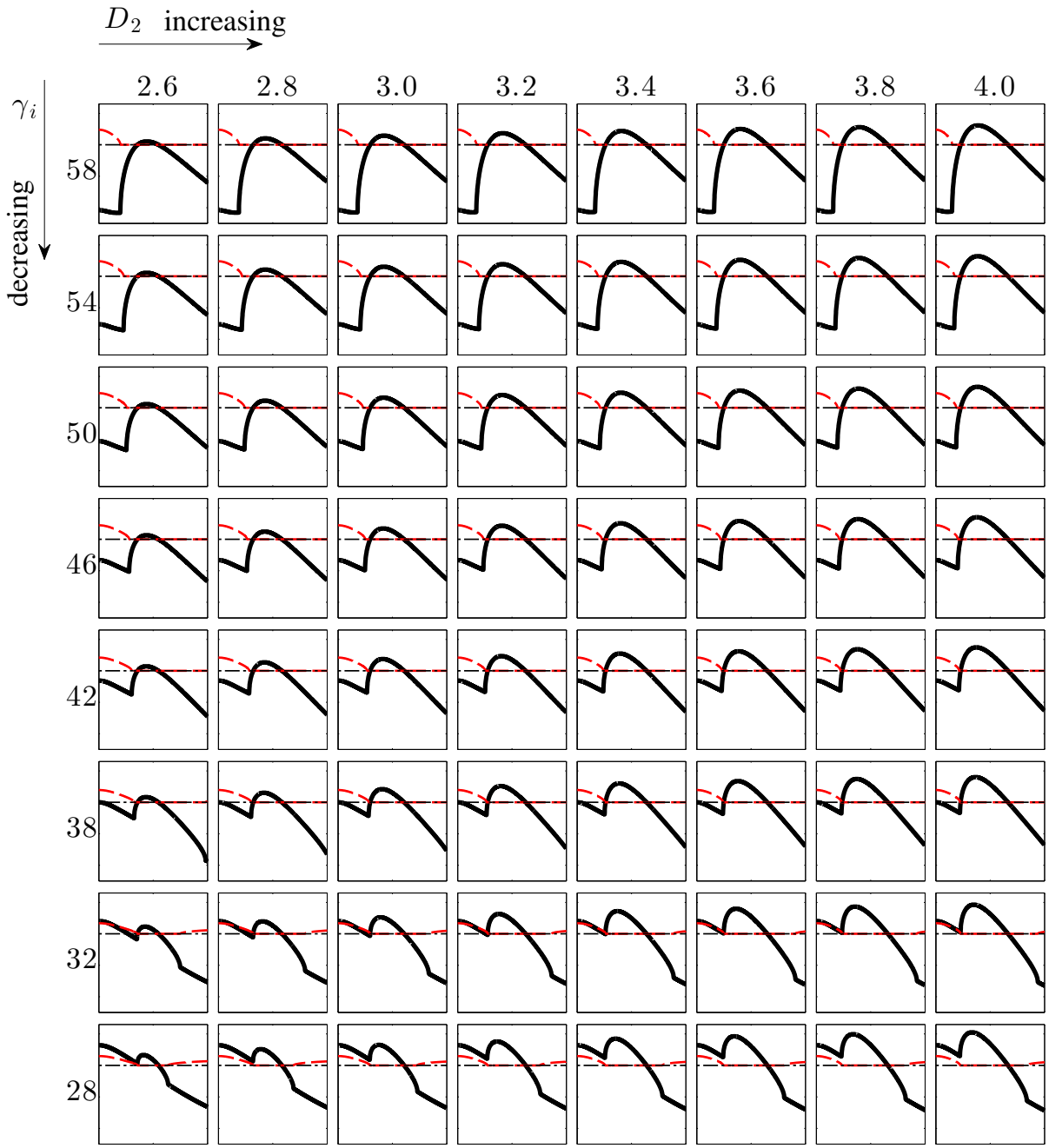


Figure 5.18: Gallery of LSA predicting curves for the SR2009s model with PSP described by the alpha-function for eight values of inhibitory diffusion D_2 (horizantal axis) and eight inhibitory rate-constant γ_i (vertical axis), the subcortical drive s is fixed 0.1. The solid-black and dashed-red curves are respective real and imaginary parts of the dominant eigenvalues over the scaled wavenumbers $(q/2\pi)$ ranging from 0 to 1 waves per cm. All subplots share the same y-axis scale from -25 to 13.

this oscillatory Turing pattern is a mixed-pattern, which originates from following possible mechanisms:

- Resonance between a Turing mode and its subharmonic [154]

  Yang *et al.* coupled two identical Brusselator systems to represent two thin layers of

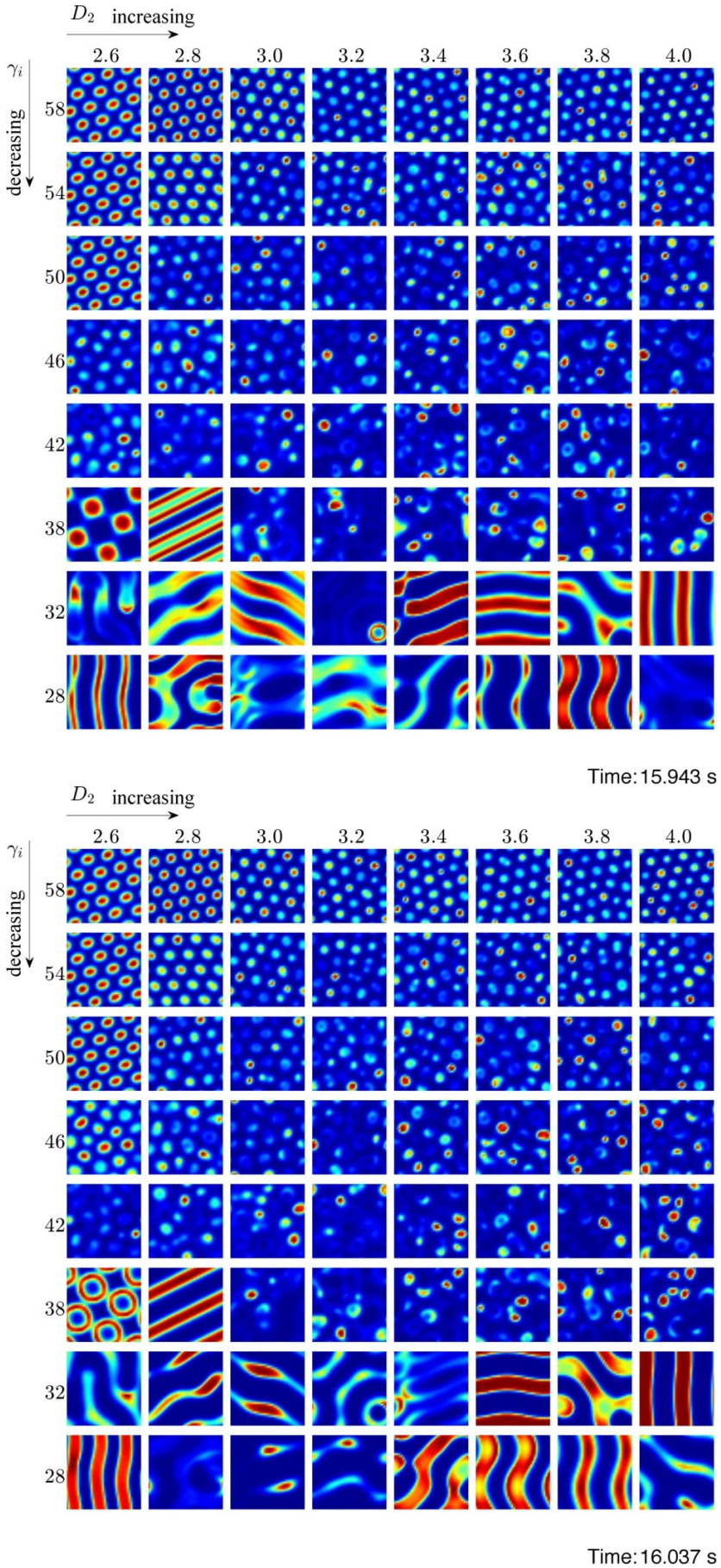


Figure 5.19: Four frames (continued in Fig. 5.20) of the animated cortical gallery containing 64 separated cortical simulations at various (D_2, γ_i) coordinates shown in Fig. 5.18.

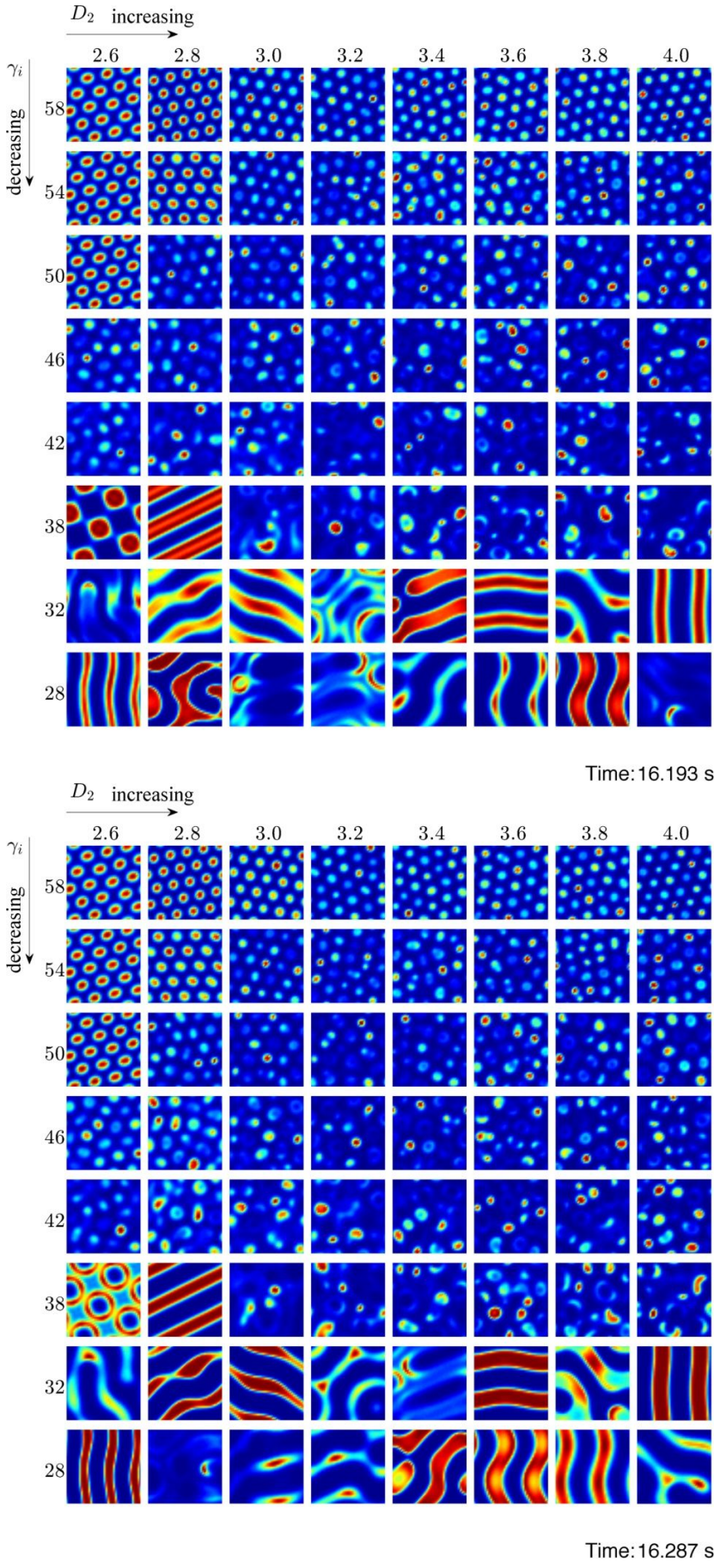


Figure 5.20: (Continued from Fig. 5.19) Each simulation runs on a substrate with 10-cm side-length of 40×40 grid. The cortical model is fed with one-off noise.

gel meeting at an interface which allows both horizontal and vertical diffusions. Assuming either Brusselator system possesses a single Turing mode, by adjusting the coupling rate for the vertical diffusion, the two-layer system may exhibit two overlapped Turing structure with their respective wavenumbers. Twinkling-eye patterns are found when the basic Turing mode (with larger wavenumber) lies just above the onset point and the sub-Turing (with smaller wavenumber) is also close to onset but within the Hopf domain where the imaginary parts of the dominant eigenvalues in the vicinity of the zero wavenumber are positive. When the basic mode is too far above the onset, the Turing pattern is striped, which agrees with our nonlinear pattern analysis for the Brusselator (see Fig. 5.3) and the cortical (see Figs. 5.13 and 5.14) models.

- Turing–wave interaction [152]
  Yang and Epstein linearly coupled two extended Oregonators [29] to represent a two-layer Belousov-Zhabotinsky (BZ) reaction. One layer possesses a Hopf instability, the other layer supports a Turing instability. The coupling produces a Turing–wave instability in which oscillating Turing spots are arranged as a hexagonal lattice.

Yang's theories imply that the oscillatory Turing patterns may have an oscillation source, from either the Hopf or the wave instabilities. In Sec. 5.2.4, we also introduced de Wit's theories for subharmonic instabilities which also suggest a Hopf oscillation source. However, neither of the oscillation sources is identified in our oscillatory H_0 patterns for the cortical model. Fig. 5.18 shows that the cortical model owns only a single Turing mode and a damped Hopf mode when $\gamma_i \geq 42$, whereas the simulations reveal another temporal instability (see supplied animation file). We argue that the Turing instability may destabilise the damped Hopf mode through nonlinearities.

On the other hand, we also noticed that the H_0 structure is well formed when γ_i is sufficiently large (see Fig. 5.19, row $\gamma_i = 58$). When D_2 is increased, LSA predicted Turing peak will slightly shift towards a larger wavenumber, thus giving rise to smaller wavelength, which visually denotes higher spot-density in the simulated patterns. From Fig. 5.19, we see that the H_0 structure cannot be maintained if Turing or Hopf instabilities are too strong.

Unfortunately, we cannot apply the current MSE scheme to the "twinkling-eye" analysis for the cortical model, due to lack of a determined oscillation source. The structure ansatz of the MSE is a linear combination of LSA predicted modes. For the cortical model, LSA predicted single Turing mode leads to only hexagonal amplitude equations. Moreover, we cannot combine Hopf and Turing modes as an ansatz for the cortical model since the homogeneous oscillation of the Hopf instability is conflict with observed localised oscillations. On the other hand, to our knowledge, the wave instability is only available to the fast-soma SR2009f model of the cortex (see Fig. 3.6(c) and (d)), but unavailable to

the SR2009s mode (slow-soma model). In principle, LSA predicts damped wave modes. We have discussed with de Wit possible reasons for the mismatches between the LSA and simulations, but she also had difficulty suggesting possible mechanisms. The analysis of the oscillatory H_0 patterns of the cortical model may inform our future research.

5.3.2 Turing–Hopf mode of the cortical model

In Sec. 3.4, we introduced a TH mode of the SR2007 model for the cortex. Shown in Fig. 3.10, by tuning D_2 and γ_i to cross their respective bifurcation thresholds, LSA predicts simultaneously raised Turing and Hopf instabilities, thus leading to oscillatory Turing patterns with ultra-slow envelope frequency. To derive the TH mode amplitude equations, we simply use the structure ansatz Eq. (4.33) with the cortical critical conditions of a CTHP for the `Amp_solving` algorithm in MAPLE. The amplitude equation for any pattern-forming system has the universal expression only depending on the mode it describes. So the TH mode amplitude equations for the cortical model are structured the same as Eqs. (5.17a, 5.17b).

The cortical ultra-slow wave is modulated by both Turing and Hopf instabilities. The envelope frequency W can be derived by applying a pair of ansatz $T = T_{\rm Ms}$ and $H = H_{\rm Ms}e^{iWt}$ to the amplitude equations:

$$W = H_{\rm Ms}^2 C_{\rm HHi} + T_{\rm Ms}^2 C_{\rm HTi} \tag{5.36}$$

in which expressions for $H_{\rm Ms}^2$ and $T_{\rm Ms}^2$ refer to Eqs. (5.20a, 5.20b). Given the (D_2, γ_i) settings in Fig. 3.10, numerical calculation yields $W = 0.9592$, which is equivalent to 0.1527 Hz. This theoretical prediction of the envelope frequency is in good agreement with observation.

From the mode stability diagram in Fig. 5.21, we see that the mixed mode is unstable when $\gamma_i > \gamma_i^{\rm c} = 30.94$, while the pure Turing is the only stable mode. The Hopf mode increases its stability dramatically when $\gamma_i < \gamma_i^c$. Although Fig. 5.21 predicts that the mixed mode is stable when $\gamma_i < \gamma_i^{\rm c}$, the strong Hopf mode may collide with the mixed mode, thus giving rise to Hopf dominated chaotic turbulence. The experiment shown in Fig. 5.22 illustrates such a chaotic mode of the cortical model when γ_i is a long way above the bifurcation threshold ($\gamma_i - \gamma_i^{\rm c} < 0$, cortical Hopf bifurcation is subcritical).

The TH mode of a pattern-forming system is normally introduced by the interaction between Turing and Hopf instabilities. The TH mode may also evolve to a chaotic state if the TH instability collides with a strong Hopf instability. We found that the emergence of the TH mode for the cortical model does not necessarily require both Turing and Hopf LSA peaks to be positive; sometimes the TH mode can be triggered by a single Turing instability. Fig. 5.23 shows a series of cortical simulations starting from eight cortical steady-states (SS), extracted from different branches of the subcortical-tone driven SS-curve. For each SS, we deliberately tuned the bifurcation parameters D_2 and γ_i to reach

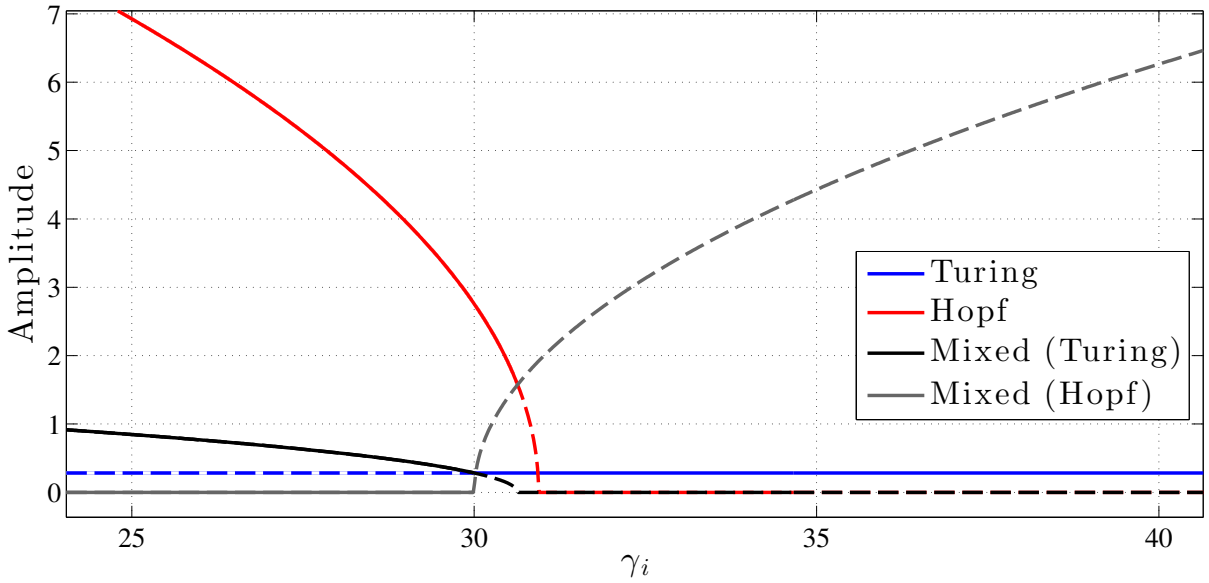


Figure 5.21: Mode stability of the TH mode amplitude equations for the SR2007 model when D_2 is fixed at 1 cm^2, subcortical drive $s = 0.2989$ (see Fig. 3.10). See the description of Fig. 5.5 for figure properies.

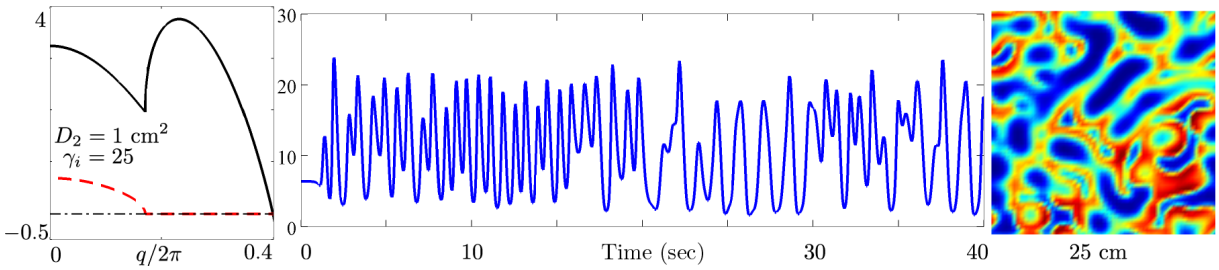


Figure 5.22: LSA predicts that sufficiently small inhibitory rate-constant γ_i raises both Turing and Hopf instabilities. Turing–Hopf mode interacting with a strong Hopf instability leads to a chaotic pattern: the time-series of the centre grid shows turbulent oscillations; a snapshot of the pattern formed at 40 s does not have identifiable periodic spatial structures, thus being spatiotemporally unstable.

a specific coordinate $(D_2^{\mathrm{SS_o}}, \gamma_i^{\mathrm{SS_o}})$ in which the LSA predicts a single Turing instability coupled with a damped Hopf instability, and both instabilities have the same absolute growth rate. According to LSA predictions, all SS will have the same instabilities, and may incur a Turing bifurcation. However, the simulations reveal that the SS at the top- and bottom-branches have completely different dynamics: Simulations starting from A to D at the bottom-branch exhibit chaotic dynamics; while from E to H at the top-branch evolve to Turing patterns.

To better understand the mode stability of the cortical model for the SS in the region from A to D, we derived their respective TH mode amplitude equations. By fixing γ_i at $\gamma_i^{\mathrm{SS_o}}$ (SS stands for a steady-state in Fig. 5.24), the TH mode stability with respect to D_2 is presented in Fig. 5.25. In common, the Turing mode experiences a "turning" at the critical point $D_2^{\mathrm{SS_c}}$. Scanning from A to D plots, the "turning" point moves towards the

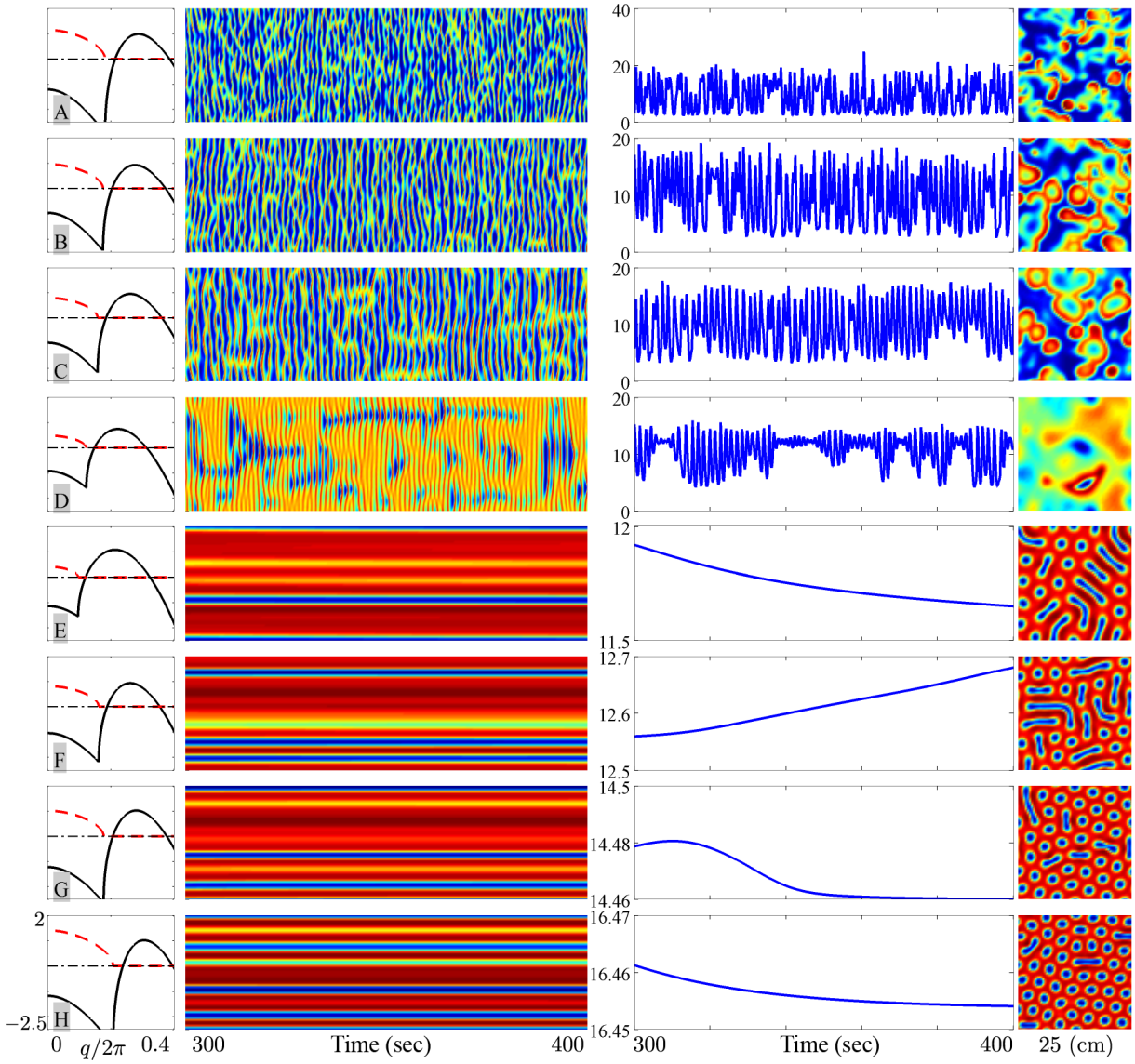
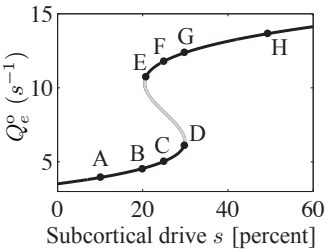


Figure 5.23: 400-s simulations of the SR2007 model starting from eight different steady-states (SS) indicated in Fig. 5.24. At each SS, the bifurcation parameters are tuned to specific $D_2^{\mathrm{SS_o}}$ and $\gamma_i^{\mathrm{SS_o}}$ so that the LSA-predicting curve shows Turing peak at 1 (i.e., $\mathrm{Re}(\lambda) = 1$ at $q \neq 0$) and Hopf peak at -1 (i.e., $\mathrm{Re}(\lambda) = -1$ at $q = 0$). In the LSA-predicting curves, solid-black and dashed-red curves are respective real and imaginary parts of the dominant eigenvalues. So eight steady-states are predicted to have the same linear instabilities for Turing and Hopf. The second column is the 100-s space-time strip-chart extracted from the centred-vertical axis of the simulated cortex with a side length 20cm and 60×60 grid-points. The third column is the time-series of a centred gird-point at (30, 30) of the cortical substrate. The fourth column is a snapshot of the pattern formed at 400 s.

left hand side, which implies that increasing the subcortical drive at the bottom branch lowers the Turing threshold. Amplitude equations show that the Turing mode becomes dominant when $D_2 > D_2^{\mathrm{SS_c}}$. In the vicinity of $D_2 = D_2^{\mathrm{SS_c}}$, stable TH mode (black and grey curves are both solid) collides with the Hopf mode (red), thus predicting a possible chaotic mode.



SS	A	B	C	D	E	F	G	H
s	0.1	0.2	0.25	0.29	0.21	0.25	0.3	0.5
$D_2^{\mathrm{SS_o}}$	0.8	0.68	0.62	0.54	0.70	0.85	0.97	1.30
$\gamma_i^{\mathrm{SS_o}}$	28	29	30	31	33.5	33	33	32

Figure 5.24: Parameter settings of the subcortical drive s, inhibitory diffusion $D_2^{\mathrm{SS_o}}$ and inhibitory rate-constant $\gamma_i^{\mathrm{SS_o}}$ for simulations at different steady-states (SS) in Fig. 5.23. The mid-branch of the SS curve is marked in grey, indicating the SS here are highly unstable (very small D_2 may cause a Turing instability).

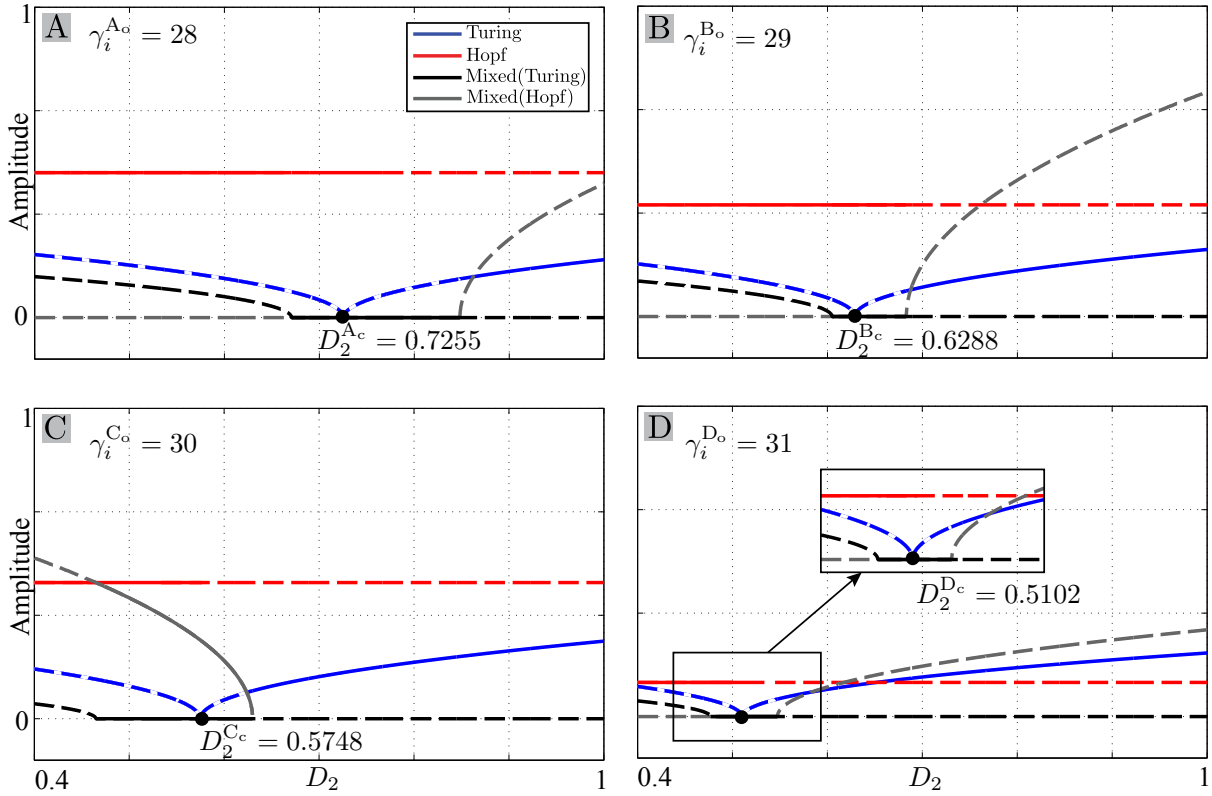


Figure 5.25: Mode stabilities for the steady-states in the region from A to D. The mode stability, derived from the TH mode amplitude equations, is examined by fixing γ_i at $\gamma_i^{\mathrm{SS_o}}$ (see Figs. 5.23 and 5.24) and varying D_2 from 0.4 to 1 cm^2. Mode stability is represented by solid (stable mode) or dashed (unstable mode) curves. $D_2^{\mathrm{SS_c}}$ (SS stands for A, B, C or D here) is the Turing threshold for each SS.

Moreover, increasing D_2, the mode stability analysis suggests a very narrow stable TH region after the interacting TH with Hopf modes. At large D_2, the TH mode will be replaced by the stable Turing mode. To test these hypotheses, we ran a series of simulations with D_2 increased up to 180% over $D_2^{\mathrm{SS_o}}$ and γ_i fixed at $\gamma_i^{\mathrm{SS_o}}$. Examining from Fig. 5.26(a) to Fig. 5.27 (c) reveals that increasing D_2 does induce a transition from TH mode to pure Turing mode.

On the other hand, we also investigated the TH mode stability with respect to the

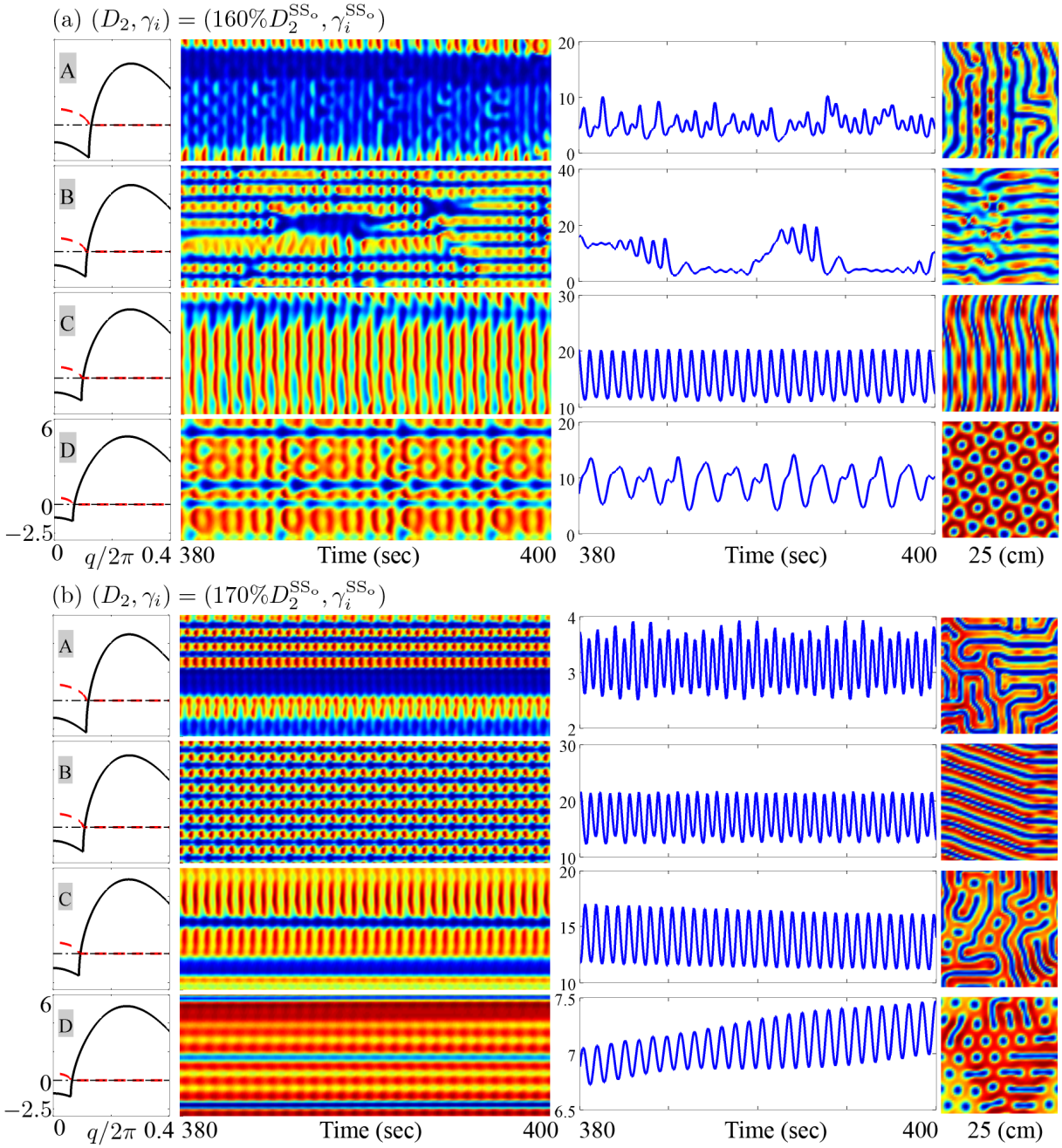


Figure 5.26: Pattern simulations of the excitatory firing-rate Q_e for the SR2007 model starting from the steady-states in the range from A to D shown in Fig. 5.24 with increased D_2 from $D_2^{\mathrm{SS_o}}$ by (a) 60%, (b) 70% and (c) 80% (see Fig. 5.27) meanwhile fixing γ_i at $\gamma_i^{\mathrm{SS_o}}$. (Continued in Fig. 5.27.)

Hopf parameter γ_i. Fig. 5.27(d) illustrates a case that D_2 is maintained the same level as in Fig. 5.26(a), but γ_i is increased by 5% over $\gamma_i^{\mathrm{SS_o}}$: A, B and C all evolve to Turing patterns as expected due to reduced Hopf instability. Surprisingly, cortical evolution from D returns to the steady-state, which is completely against the LSA prediction for a strong Turing instability. To our knowledge, there is little publications referring to this phenomenon. We argue that the current (D_2, γ_i) settings may lead the two instabilities to completely cancel with each other. This steady-state can be easily broken by increasing

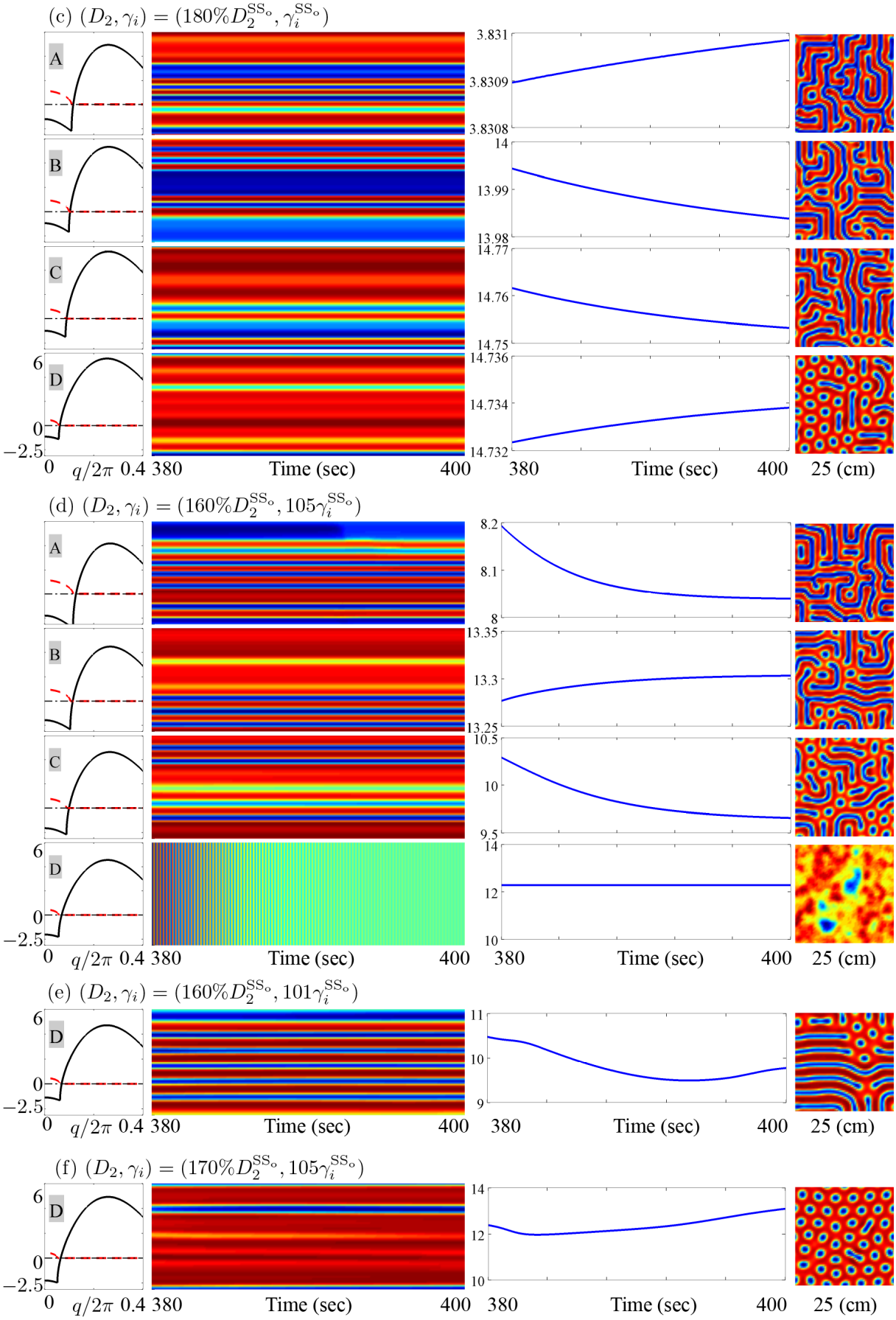


Figure 5.27: (continued from Fig. 5.26) In (d), we increased D_2 by 60% and γ_i by 5%. In (e), we maintained D_2 the same as in (d) and only increased γ_i by 1%. In (f), we maintained γ_i the same as in (d) and increased D_2 by 70%.

slightly either the Hopf instability (see Fig. 5.27(e)) or the Turing instability (see Fig. 5.27(f)), which will consequently prompt a dominant mode.

In addition, the hexagonal structure shown in the last row of Fig. 5.26(a) has also attracted our attention. We examined the pattern dynamics in the last few seconds of this 400-s simulation. The upper panel of Fig. 5.28 reveals "twinkling-eye" patterns, which are very similar to Yang's observations on chemical reaction–diffusion systems [152,154]. The 2D Fourier spectrum shows that the pattern periodically switches the spatial structure between solo- and dual-hexagon. The spatial Q_e pattern has three different localised structures: light blue (intermediate firing-rate) and dark blue (low firing-rate) spots, and red (high firing-rate) background. The 20-s time-series reveal that the red background oscillates homogeneously, which is also in phase with the blue spots at the intermediate firing-rate region but out of phase with the low firing-rate spots. Thus we can see spots "blink" alternately. The Fourier spectrum in panel (d) shows the high firing-rate zones (i.e., red background) oscillate two times faster than other pattern components, indicating that the pattern has two characteristic wavenumbers as well as two frequencies, which is very similar to the subharmonic instability of a Turing–Hopf mode (see Eq. (5.35)).

5.3.3 Discussion

In the second half of this chapter, we investigated the pattern dynamics of the Waikato cortical model, and found the appropriate model parameter settings to form specific spatiotemporal patterns.

The cortical model is expected to exhibit similar dynamics to a chemical reaction–diffusion system: The interaction between the excitatory and inhibitory neurons is analogous to the competition between activator and inhibitor of the chemical reaction–diffusion model, e.g. the Brusselator. The gap-junction strength between the inhibitory neurons also plays the same role as the diffusion terms in the Brusselator, which allows a spatial evolution of the patterns. Consequently, it is reasonable that the cortical model shares the same pattern dynamics with the chemical reaction–diffusion system. For the cortical model, the H_π (Figs. 5.12 and 5.13(a)), stripes (Fig. 5.11(A)) and H_0 (Fig. 5.15) modes of the Turing patterns plus TH (Fig. 3.10) and chaotic (Fig. 5.22) modes are all found to exist in our investigations.

Besides, the strong nonlinearity of the cortical model leads to richer pattern dynamics than the Brusselator model. The cortical model possesses a unique S-shaped steady-state distribution with respect to subcortical drives, the top-, mid- and bottom-branch of which have distinct dynamics in

- Mode dominance (see Fig. 5.11)
  The top-branch dominates the H_π mode, and the mid- and bottom-branches dominate the stripes mode of the Turing patterns;

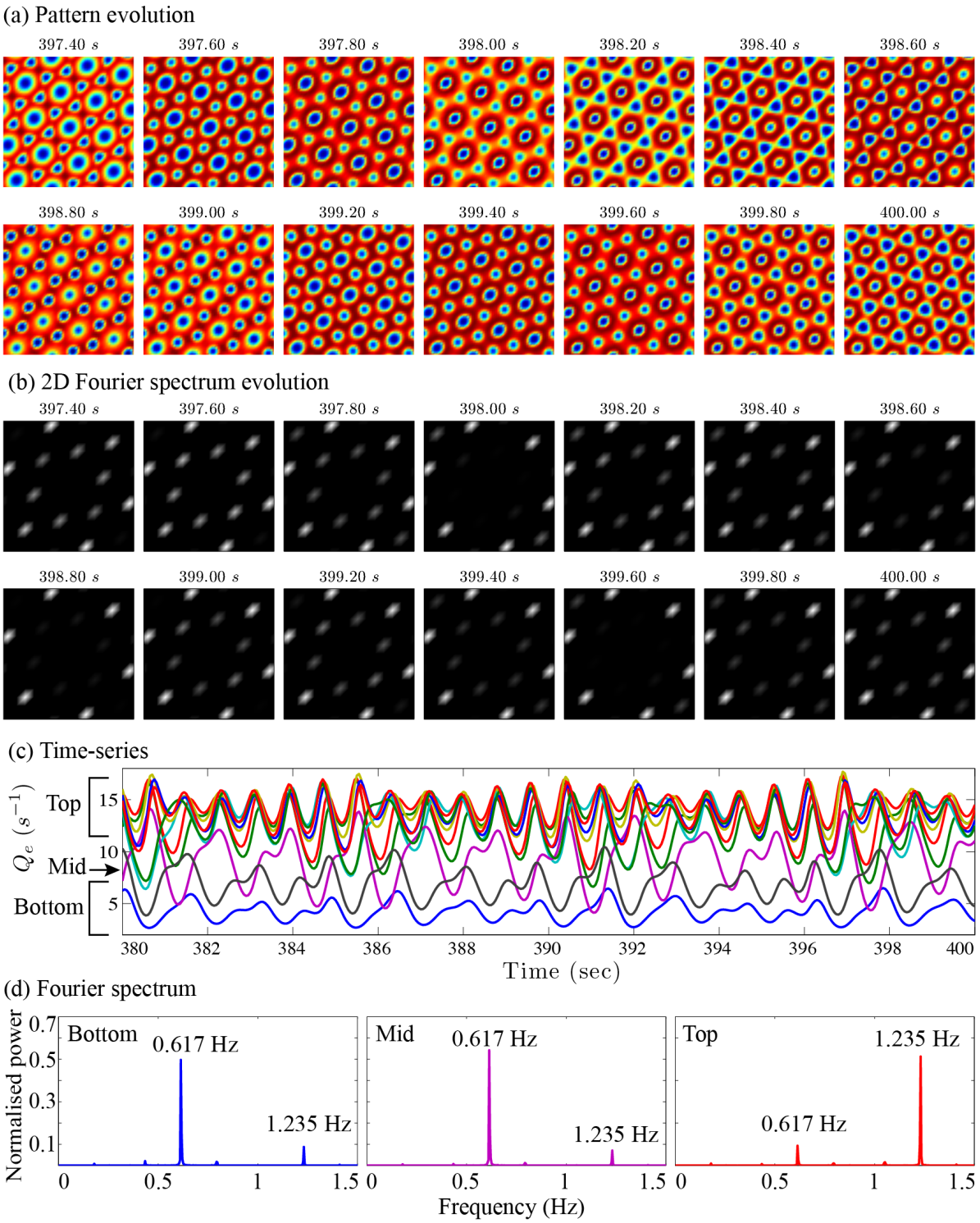


Figure 5.28: Pattern dynamics for the SR2007 model at steady-state D with $(D_2, \gamma_i) = (160\% D_2^{\mathrm{SS_o}}, \gamma_i^{\mathrm{SS_o}})$ (see the last row of Fig. 5.26(a)). Panel (a) shows the evolution of the cortical Q_e firing-rate pattern from 397.40 to 400 s, and the corresponding 2D Fourier spectra are shown in panel (b). Panel (c) shows 20-s Q_e time-series of 10 equally spaced grid-points down the vertical mid-line of the cortical grid. Q_e oscillates in three regions: Top, high-firing zone ranging from 10 to 18 s^{-1}; Mid, intermediate-firing zone ranging from 5 to 13 s^{-1}; Bottom, low-firing zone below 6 s^{-1}. Panel (d) measures the contribution of the frequency components to the total spectral power for three representative Q_e time-series from three regions.

- Hopf instability sensitivity (see Fig. 5.23)
  The bottom-branch is more sensitive to the Hopf instability than the top-branch, and the mid-branch is highly unstable to the Turing or wave instabilities.
- Also, presence of turning point in steady-states enhances instabilities.

From Figs. 5.12 and 5.13, we see that the stability of the H_π mode for the cortical mode has stronger dependency on the Turing parameter D_2 than the Hopf parameter γ_i. Once the model experiences a Turing bifurcation, the choice of γ_i has little effect on forming the H_π structure; in comparison, increasing D_2 from its Turing threshold to a large value, a well-structured H_π mode will gradually be taken over by stripes. It is also noticed that the H_π mode can exist in the bottom-branch close to the right turning point where both the subcortical drive and D_2 are sufficiently large (see Fig. 5.27(f)).

The hexagonal amplitude equations for the SR2007 model shows no evidence of the H_0 mode (see Fig. 5.14). The H_0 mode, also referred as reentrant H_π mode, is found in the SR2009s model (see Fig. 5.15). Increasing the Turing or Hopf instabilities will induce localised oscillations to the H_0 mode, thus leading to the oscillatory H_0 patterns. However, the source of the oscillation is unclear since the linear stability analysis (LSA) shows a highly damped Hopf temporal instability (see Fig. 5.16).

On the other hand, once the Turing and Hopf instabilities are both enhanced sufficiently, a stable TH mode will occur. The TH mode amplitude equations provide a more accurate temporal measurement of the envelope frequency for the pattern evolution than the LSA, and provide a better understanding of the TH mode stability relating to a bifurcation parameter. Our TH mode analysis are in agreement with simulations (Fig. 5.23) except for the case seen in the last row of Fig. 5.27(d) where the pattern evolution is quiescent and the system returns to the steady-state despite the fact that both amplitude equations and LSA predict a strong Turing instability. We argue that the system may not evolve if intrinsic instabilities are completely cancelled with each other.

Figure 5.16 signals that a moderately induced Hopf instability will temporally destabilise the H_0 mode. With the same mechanism, the H_π mode shown in Fig. 5.27(f) can also be introduced with temporal instability for localised oscillations, evident in Figs. 5.26(a) (last row) and 5.28. Interestingly, both temporal H_π and H_0 instabilities are only induced by a damped Hopf instability.

In Fig. 5.28, the 2D Fourier spectrum shows that the oscillatory H_π mode has two dominant hexagonal spatial periodicities in different sized wavenumbers. At 397.40 s, the double hexagonal structures are clearly unveiled. In the 2D Fourier spectrum, the internal hexagon may be induced from the Hopf instability since it waxes and wanes over time. Commonly a system possesses a single dominant wavenumber[5]. To obtain a multiple-wavenumber system, Yang *et al.* [152,154] linearly coupled two identical chemical systems

[5]A pattern can have multiple wavevectors with the identical magnitude (wavenumber)

with distinct wavenumbers and instabilities. The ratio between two wavenumbers and the total instability can be adjusted via the coupling rate. According to Yang's theories, the "twinkling-eye" pattern is a wave instability, which can be produced by either modulating the Turing instability of a subsystem with the Hopf instability of the other subsystem or maintaining a dominant Turing instability meanwhile mixing a sub-Turing (with a smaller wavenumber) with the Hopf instabilities. To our understanding, Yang's twinkling-eye pattern essentially results from the interaction between two coupled systems, and its spatial structure is modulated by the coupling rate. In Yang's simulations, the model shows stable multiple wavenumbers, which stay fixed over time.

Why do our our "twinkling-eye" patterns (Fig. 5.28) have a temporally unstable wavenumber? Does such wavenumber belong to a subsystem of the cortical model? Although the cortical model can be separated to three parts: cortico-cortical, intra-cortical and soma models, they are coupled nonlinearly with *dynamical coupling-parameters*. For example, the coupling-parameters between intra-cortical and soma models are $\rho_e\psi_{eb}$ and $\rho_i\psi_{ib}$, in which ψ_{eb} and ψ_{ib} are functions of V_e and V_i respectively. Q_e, a function of V_e, also plays as a coupling-parameter connecting intra-cortical and cortico-cortical models; meanwhile it receives feedbacks from excitatory membrane potentials from the soma model. We have tried to decompose the cortical model into subsystems. By examining the eigenvalue dispersion curves of these subsystems, we did not find the characteristic wavenumber. We propose a hypothesis that the second wavenumber is generated spontaneously from the nonlinearity in the cortical model.

Mathematically, the wavenumber comes from the spatial derivative – the Laplacian operator. The cortical model has two types of spatial derivatives: one in the soma model and the other in the cortico-cortical model. We found that mismatches between LSA prediction and simulations are mostly from the effect of the spatial derivative in the cortico-cortical model. To remove this Laplacian operator, we simply force v to zero, which will cut-off the long-range wave propagation. Then LSA shows a very strong Turing instability, which appears to be labyrinth pattern with very thin stripes arranged in high density. It is biologically unreasonable for a cortical system to have only local connections, so we need to retain the spatial derivative in the cortico-cortical model.

The Waikato cortical model is a biological system providing theoretical guidance for brain dynamics, it is also a complicated pattern-forming system exhibiting comprehensive pattern dynamics: H_π, stripes, H_0, TH, travelling wave, chaotic, H_π-Hopf and H_0-Hopf modes. Without adjusting the model structure, these pattern dynamics can be observed by tuning the balance between Turing and Hopf instabilities.

In the next chapter, we will tune the model dynamics for a representation of the anaesthetised cortex, in order to have a better understanding of the mechanism of slow-waves observed from the comatose cortex.

Chapter 6

A gap-junction modulated Turing–Hopf mechanism for anaesthetic slow-waves

Electroencephalogram (EEG) recorded from non-REM (NREM) sleep and general anaesthesia are characterised by large amplitude slow-waves. Tracking the properties of such slow-waves from consciousness to unconsciousness may help us to identify the effects of the anaesthetic and offer an insight into the mechanisms of general anaesthesia. EEG coherence is considered to be a qualitative measure of the functional cooperation between brain regions. By examining EEG recorded from volunteers undergoing propofol anaesthesia, we observed a drop in EEG coherence during the slow-wave oscillations, mainly at the frontal cortex. However, the mechanisms of such reduced coherence, and the origin of slow-wave oscillations in the unconscious state, are still unclear.

A recent theoretical study by Steyn-Ross *et al.* [116] forced the pre-developed cortical model to describe the anaesthesia-induced transition from consciousness to unconsciousness. At the modelled unconsciousness state, the intermediate gap-junction coupling originates a codimension-2 point (CTHP) where competing Turing and Hopf instabilities coexist (TH mode), signifying chaotic slow-wave oscillations arising spontaneously in the cortical simulation. At this brain state, Steyn-Ross *et al.* also found that neuronal synchronisation across the cortex drops dramatically compared to its high level during consciousness.

In this chapter, we will first review Steyn-Ross *et al.*'s anaesthesia model of the cortex and its pattern dynamics with respect to gap-junction coupling strength. Then, we will present our clinical study for analysing EEG data during deep anaesthesia to test the model prediction.

6.1 Anaesthetic unconsciousness

Propofol induced anaesthesia is a sleep-like stage that is signposted by the propagation of large amplitude, slow oscillation of EEG activity of the cortex. Although it remains unclear how these slow oscillations are produced, deep anaesthesia is commonly used to model slow-wave sleep.

Sleep is modulated by a complex system of brainstem modulators. For ease of modelling the sleep cycle, Steyn-Ross *et al.* [112] assumed that there are just two neuromodulators controlling cortical state in a transition from awake (REM sleep) to slow-wave sleep (SWS):

- **Adenosine**, one of the "fatigue agents", its accumulation intends to suppress cortical activity through increases in potassium leak-current, thus lowering the neuronal resting potential and making cortical neurons less likely to activate [41].
- **Acetylcholine (ACh)**, reported to have direct influences on depolarising cortical pyramidal cell (excitatory) membrane potentials [41]. ACh release is greatest during wake, and is reduced during slow-wave sleep (SWS) [131].

Both adenosine and ACh are the body's self-modulation to enter into the sleep stage. On the other hand, anaesthetic unconsciousness is a manually induced deep sleep. General anaesthetic agents potentiate neuronal inhibition by prolonging the opening of GABA (γ-ambinobutyric acid) channels, increasing the influx of chloride ions, causing the neuron to become hyperpolarised [19]. Although there is ongoing debate about the exact mechanisms of general anaesthetic agents at the molecular level, it is widely acknowledged that, for example, the anaesthetic propofol tends to increase the area under the inhibitory post-synaptic potential (IPSP) by prolonging the duration without increasing its peak amplitude, thus increasing the level of inhibition within the brain [108].

6.2 The effect of gap-junctions on cortical stability

Unconsciousness is characterised by low-frequency, large-amplitude traveling waves in scalp electroencephalogram (EEG), which is also observed in generalised seizure manifest with the synchronised spike-wave discharge [43]. Distinct from chemical synaptic connection, gap-junctions allow direct electrical communication between adjoining neurons, thus enhancing synchronous behaviour is likely to relate to the emergence of seizure (expression of gap-junctions is often found to be higher after clinically induced seizures than before [134]); conversely, reducing gap-junction conductance may suppress seizure [110, 121]. It has been acknowledged that gap-junction plays an crucial role in the cortical stability [117, 119, 121]. However, it remains an open question as to whether or not gap-junctions are necessary for neuronal synchronisation [25]. The debate arises from the contradictory experimental observations from different research groups: It has been observed that open gap-junctions aid synchronisation of sleep spindles [76, 126], while opening gap-junctions via the stimulant modafinil is also found to increase vigilance [5]. There is already some evidence that induction of anaesthetic drugs (such as propofol and thiopental) depress intercellular communication thus mediating the hypnotic effects by altering gap-junctional gating [53]. Furthermore, Jacobson *et al.* [51] suggests that

the hypnotic effects of anaesthetic drugs may be moderately enhanced by gap-junction blockade. In therapeutic seizure, some report that gap-junction blocking agents suppress seizure by limiting the spread of synchronised activities [33, 52, 83], while others provide evidence that increased gap-junctional communication plays an intrinsic role in the epileptogenic process [6, 95]. The role of Cx36 gap-junction on anticonvulsant effects is examined in [135]; comparison between pharmacological (mefloquine) and genetic (Cx36 knockout mice) manipulation of Cx36 gap-junctions shows that Cx36 gap-junctions may not directly promote seizure activity.

Overall, it seems intuitively reasonable that opening gap-junctions will reduce input resistance of the neuron and allow rapid information exchange, thus enhancing synchrony between neurons and promoting seizure activity. Experimental studies, as introduced, however, have reported conflicting results. These discrepancies highlight our inadequate understanding of the role of gap-junctions.

Does opening gap-junction promote or inhibit neuronal synchronisation? To help address this conflict, in Sec. 3.1 we have presented a physiologically-motivated cortical model that comprises neuron groups communicating via chemical (neurotransmitter controlled) and electrical (gap junctions) synapses simultaneously. In the following sections, we will show how coupling via gap junctions can be involved in the slow-wave dynamics featured in general seizure and anaesthetic coma.

6.3 Modelling anaesthesia

Consistent with the discussion in Sec. 6.1, Steyn-Ross *et al.* [116] modelled propofol effect by scaling the area of the IPSP response with induced anaesthetic drug concentration without changing the IPSP peak amplitude, thus the following mappings are introduced to the intra-cortical equation (3.7) and soma equation (3.21):

$$\begin{aligned} \gamma_i &\to \gamma_i^0/\lambda_i \\ \rho_i &\to \lambda_i \rho_i^0 \end{aligned} \tag{6.1}$$

An increase in propofol concentration corresponds to an increase of a dimensionless scale factor λ_i, which will decrease the inhibitory rate-constant γ_i but enhance the synaptic strength ρ_i. Such balance between γ_i and ρ_i allows the area of the IPSP response grow linearly with propofol concentration while ensuring that the IPSP peak remains unchanged.

In addition, Steyn-Ross *et al.* introduced a resting potential offset ΔV_e^{rest}:

$$V_e^{\text{rest}} \to V_e^{\text{rest}} + \Delta V_e^{\text{rest}} \tag{6.2}$$

acting as an excitatory drive from subcortical sources.

As a result, the cortical steady-state is determined in a domain of anaesthetic effect λ_i and resting potential offset ΔV_e^{rest}.

6.4 Cortical stability

In order to locate the steady-states for the cortical model, we set all time- and space-derivatives in the differential equations to zero, and set the noise sources $\phi_{eb}^{\text{sc}}(t) = 0$, then compute numerically to locate the steady-state firing rates (Q_e^o, Q_i^o) of the excitatory and inhibitory neuronal populations as a function of ΔV_e^{rest} and λ_i. The resulting 3-D manifold of steady-states is displayed in Fig. 6.1.

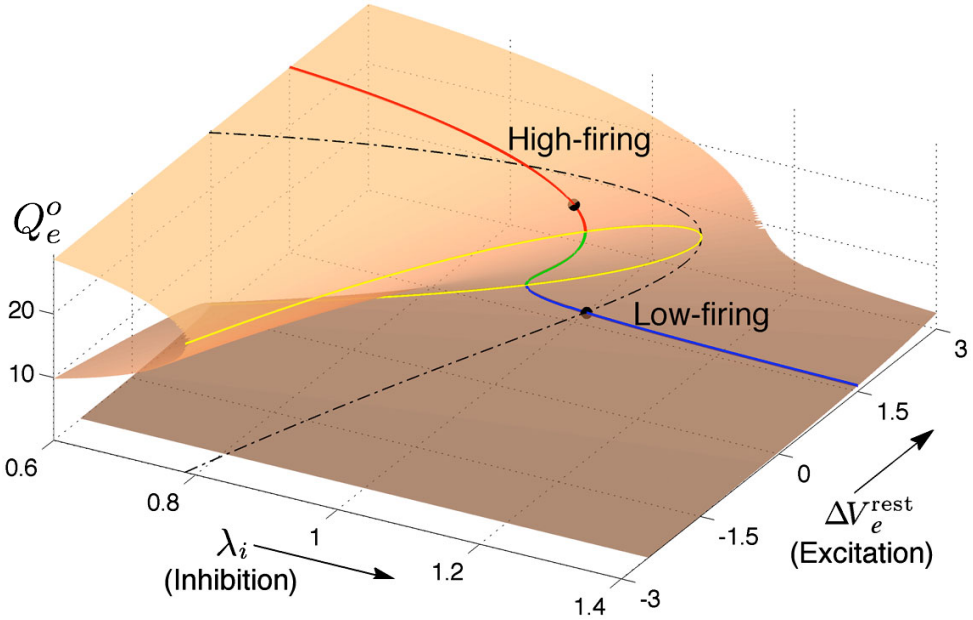


Figure 6.1: Manifold of equilibrium states Q_e^o for homogeneous model cortex. Anaesthetic effect controller λ_i and soma membrane potential offset ΔV_e^{rest}. Multiple stationary states exist with the *yellow* curve bounding the upper and lower surfaces. *Red-green-blue* curve is a reprentative distribution of multiple steady states for varying anaesthetic inhibition at a constant cortical excitation $\Delta V_e^{\text{rest}} = 1.5$ mV (vertical slice plot shown in Fig. 6.2). (Figure reproduced from [116])

Figure 6.1 shows that the excitatory neuronal steady-state firing rates Q_e^o of the model forms a reversed S-shape distribution with the upper branch corresponding to an activated cortical state identified as awake (or REM sleep), and the lower branch corresponding to a suppressed cortical state identified as propofol anaesthetic induced coma (or SWS) [112, 121]. In this study we investigate the cortical dynamics at selected "awake" and "coma" coordinates from a vertical slice Fig. 6.2 of the multiple steady-states domain in Fig. 6.1 at a constant cortical activation $\Delta V_e^{\text{rest}} = 1.5$ mV. By increasing the concentration of propofol anaesthesia λ_i, the model describes the anaesthesia-induced transition from consciousness to unconsciousness.

In Fig. 6.2, two contrasting states for the homogeneous model cortex are chosen: An awake state from the upper manifold branch at anaesthetic effect $\lambda_i = 1.0$; and a coma state just beyond the manifold branch at $\lambda_i = 1.018$.

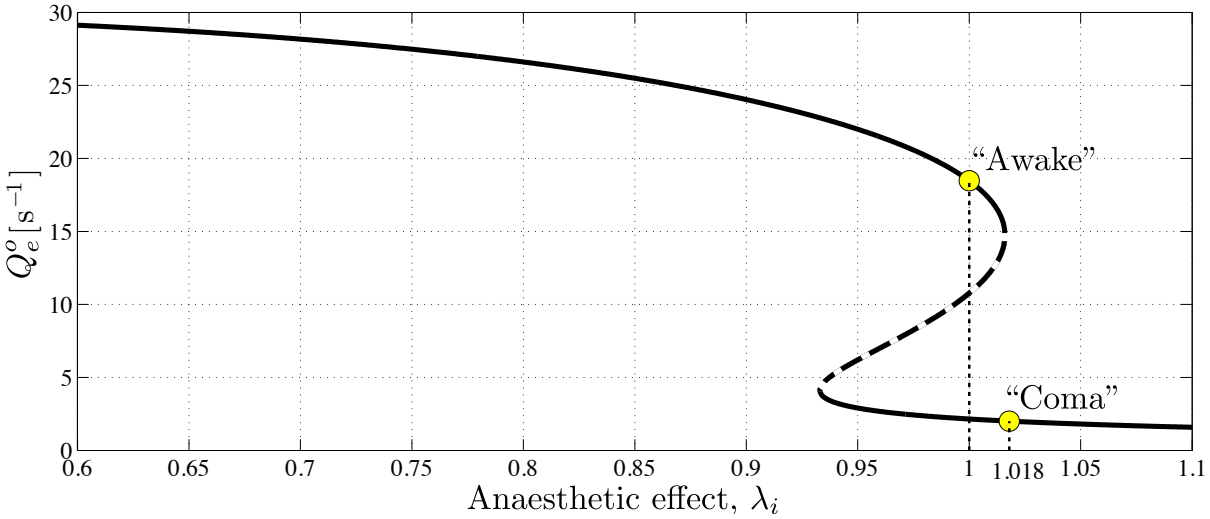


Figure 6.2: The steady-state firing rates Q_e^o as a function of varying anesthetic inhibition λ_i at constant cortical excitation $\Delta V_e^{\text{rest}} = 1.5$ mV. The upper, high-firing and lower, low-firing branches (solid curve) are considered to be "awake" and "coma" states respectively with "coma" state being associated with anaesthetic-induced unconsciousness. Dashed curve indicates an unstable branch where a steady-state has the potential to jump to the upper or lower stable branch. Upper and lower marked circles indicate references at $\lambda_i = 1.0$ and 1.018 on awake and coma branches respectively. (Figure reproduced from [116])

The model cortex is represented as an $N_x \times N_y = 120 \times 120$ square grid of side length 6 cm, joined at the edges to provide toroidal boundaries. The cortical sheet is initialised at the homogeneous steady-state corresponding to a specified value of cortical activation ΔV_e^{rest} and anaesthetic effect λ_i.

In the present work, we investigate cortical dynamics at "awake" and "coma" (initialised from circled points in Fig. 6.2) brought about by stepped increase in the gap-junction conductance D_2. This increase in conduction of the electrical synapse may also modulate neuronal synchrony that will be explored via the coherence measure.

6.5 The measure of EEG phase-coherence

In this study, we tracked coherence of modelled EEG across the cortical substrate before ($\lambda_i = 1$) and after ($\lambda_i = 1.018$) propofol anaesthesia to identify the theoretical effects of the anaesthetic

EEG coherence is considered to be a qualitative measure of the degree of association or coupling between two EEG channels, which can be extended to measure the functional co-operation between two brain regions [87], and reveal subtle changes in brain dynamics. For example, coherence has been utilised to investigate brain cognition [124], maturation [61], spatial tasks [105] and various clinical diagnoses of brain disease [17, 18, 98]. EEG coherence between two electrode sites is usually computed by one of two methods: the Fourier

transform (FT) cross spectrum [1], or the Hilbert transform (HT) instantaneous phase difference [80] between two EEG time-series. Since EEG represents the activities of non-linearly interacting neuronal populations, it is neither truly linear nor stationary. Thus it may be unreliable to use FT-based methods for EEG analysis since these assume that the time-series is stationary [67, 161]. The Hilbert transform [45, 123] circumvents the requirement for stationarity by generating an analytic signal to extract the instantaneous frequency and phase angle from the original nonstationary signal. The mean of the phase divergence between two time-series yields an index characterising the phase synchronisation between them. The advantages of the HT over the traditional FT-based approaches have been appreciated in many studies of cortical neuronal synchronisation under different circumstances such as Parkinson's disease [125], abrupt seizure [90], sleep [155] and anaesthetic coma [60].

6.5.1 Hilbert transform

A real time-series $X(t)$ can be transformed to a complex function known as the analytic signal:

$$\hat{X}(t) = X_r(t) + iX_i(t) \tag{6.3}$$

where $X_r(t)$ is the original series $X(t)$ and $X_i(t)$ is the Hilbert transform of $X(t)$ [60, 80]. The instantaneous phase of $X(t)$ is computed by:

$$\phi(t) = \tan^{-1}\left(\frac{X_i(t)}{X_r(t)}\right) \tag{6.4}$$

To quantify the phase synchronisation between two time-series $X_m(t)$ and $X_n(t)$, an index presented by Steyn-Ross *et al.* [116, 121] is used:

$$R_{(m,n)} = |\langle e^{i[\phi_m(t)-\phi_n(t)]}\rangle| \tag{6.5}$$

The mean phase coherence R measures the time-averaged phasor for the angular distribution of the phase difference between the two time-series. The index R lies between 0 to 1, where 1 represents perfect phase coupling.

A MATLAB implementation for computing the mean phase coherence between two signals reads as follows [121]:

```
% Compute analytic (complex) signals for Xm and Xn
  Xmc = hilbert(Xm);  Xnc = hilbert(Xn);
% Extract instantaneous phase angles
  phi_Xm = angle(Xmc); phi_Xn = angle(Xnc);
% Measure the average phase-coherence
  R = abs(mean(exp( 1i*(phi_Xm - phi_Xn))));
```

Here, we imagine that the vertical mid-column of the cortical grid is attached to a series of electrodes, located at positions 1, 2,..., 120. Let $X_m(t)$ and $X_n(t)$ be a pair of

EEG recordings respectively from the electrodes m and n down the middle of the cortical grid. A 120-channel EEG recording has a total of 120×120 pairs of R values, but half of these are redundant since $R_{(m,n)} = R_{(n,m)}$. The coherence matrix is represented as an $m \times n = 120 \times 120$ square grid with the unit diagonal ($R_{(m,n)} = 1$ when $m = n$), which separates the matrix into two symmetrical triangles ($R_{(m,n)} = R_{(n,m)}$). Practically, we need only examine the upper triangle (i.e., $R_{(m,n)}$) of the R matrix. See Figs. 6.3(e) and 6.10 for an illustration of the structure of the coherence matrix.

For coherence calculations, if a given time-series is longer than 5 s, we will use a 5-s moving window with 1-s overlap, and follow Mormann *et al.* [80] and Steyn-Ross *et al.* [116, 121] in applying a Hann window, retaining only the middle 80% of each segment to minimise edge distortions from the Hilbert transform. The final determined coherence is the average of those obtained from windowed signal segments.

6.6 Simulation results

The numerical simulation demonstrates 20-s evolution of the cortex starting from homogeneous "awake" and "coma" respectively. In this section, we will investigate the effect of variation in the strength of inhibitory diffusion D_2 on each of the cortical states.

6.6.1 Effect of inhibitory diffusion on "Awake" cortex

Pathological brain: An instability takes precedence

Strong inhibitory diffusion $D_2 = 0.8$ cm^2 (bottom row in Fig. 6.3) reveals an extremum in which the homogeneous cortex has evolved to a relatively-frozen (patterns have little temporal evolution) Turing pattern with maze-like high- and low-firing activities. Such Turing dominant cortical status is identified to be pathological, since the frozen pattern blocks information flow to distinct cortical areas. At the other extreme, weak inhibitory diffusion (see $D_2 = 0.1$ cm^2 at top row) leads to Hopf dominated global oscillations (phase-coherence map shows that evolution at distinct grid-points is highly synchronised). Such unrestrained Hopf instability manifest with globally synchronised oscillation over time is identified as seizure [121]. To conclude, a cortical state either dominated by Hopf oscillation with highly synchronised global oscillation, or frozen Turing structure with desynchronised standing waves are two pathological states of the brain.

Inhibitory diffusion modulates the balance between Turing and Hopf instabilities

It is noticed from Fig. 6.3(a) that weak $D_2 = 0.1$ shows a Hopf instability at up-state plus a damped-Hopf at down-state, while the strong $D_2 = 0.8$ shows not only Hopf at up-state, also a Turing instability (instability peak at $q \neq 0$) at down-state. Scanning column (a) from top to bottom, we can find the cortical stability at up-state has little change, but

at down-state stepped increases of D_2 favour the occurrence of a Turing instability. From the time-series in column (c), it is clear that the Hopf oscillation is gradually suppressed by Turing structure with increasing D_2. Thus the variation of D_2 leads to a competition between Turing and Hopf modes and emergent spatiotemporal pattern dynamics.

Healthy brain: Balanced Turing and Hopf instabilities

In the awake cortical simulations of Fig. 6.3, when the gap-junction strength is sufficiently large ($D_2 = 0.7$ cm^2), linear stability analysis of the up-branch steady-state at $\lambda_i = 1$ in Fig. 6.2 predicts whole-of-cortex Hopf oscillations; while the down-branch steady-state shows a damped-Hopf at wavenumber $q = 0$ plus a damped-Turing at $q \neq 0$. The time-series and strip-chart depict a stable Turing–Hopf mode evolution where the cortical Turing patterns oscillate in small amplitudes. Such Turing-interacted Hopf slow-oscillation has been characteristic of a resting state of the cortex [121] or non-cognitive idling state [117]. These slow patterned oscillations may relate to very slow ($\leq$0.1 Hz) fluctuations in BOLD (blood-oxygen-level dependent) signals detected using fMRI (functional magnetic resonance imaging) of relaxed, non-tasked human brains [30, 32]. In Sec. 3.4, we have discussed the idea that a balance between cortical Turing and Hopf instabilities may form a possible substrate of a healthy functioning brain.

Early warning sign of seizure: Turbulent, chaotic waves

Two extreme cortical states are associated with either too large or too small inhibitory diffusion, leading respectively to Turing pattern or Hopf oscillation. Now let us examine the transition between these two extrema with D_2 being altered from 0.7 to 0.3 cm^2. With stepped reduction of D_2 from 0.7 to 0.3 cm^2, the coherent, small oscillations about the up-and down-states are replaced by incoherent fluctuations at $D_2 = 0.4$ cm^2 (see column (e)), the fluctuations appearing turbulent and chaotic. The above inspection shows that the onset of seizure (identified as maximally coherent, whole-of-cortex oscillation at suppressed inhibitory diffusion D_2) might follow on turbulent, chaotic EEG signals with decreased coherence, serving as a potential early warning sign of seizure. This modelling prediction is supported by clinical observations [2, 11, 79].

Conclusion

The modelling of cortex at "awake" state enables two predictions:

1. Opening gap-junctions protects brain against seizure, while closure of gap-junctions promotes seizure;
2. The survey of coherence maps implies an underlying trend from healthy to seizing brain: The phase coherence between two separated cortical areas initially decreases, then increases dramatically.

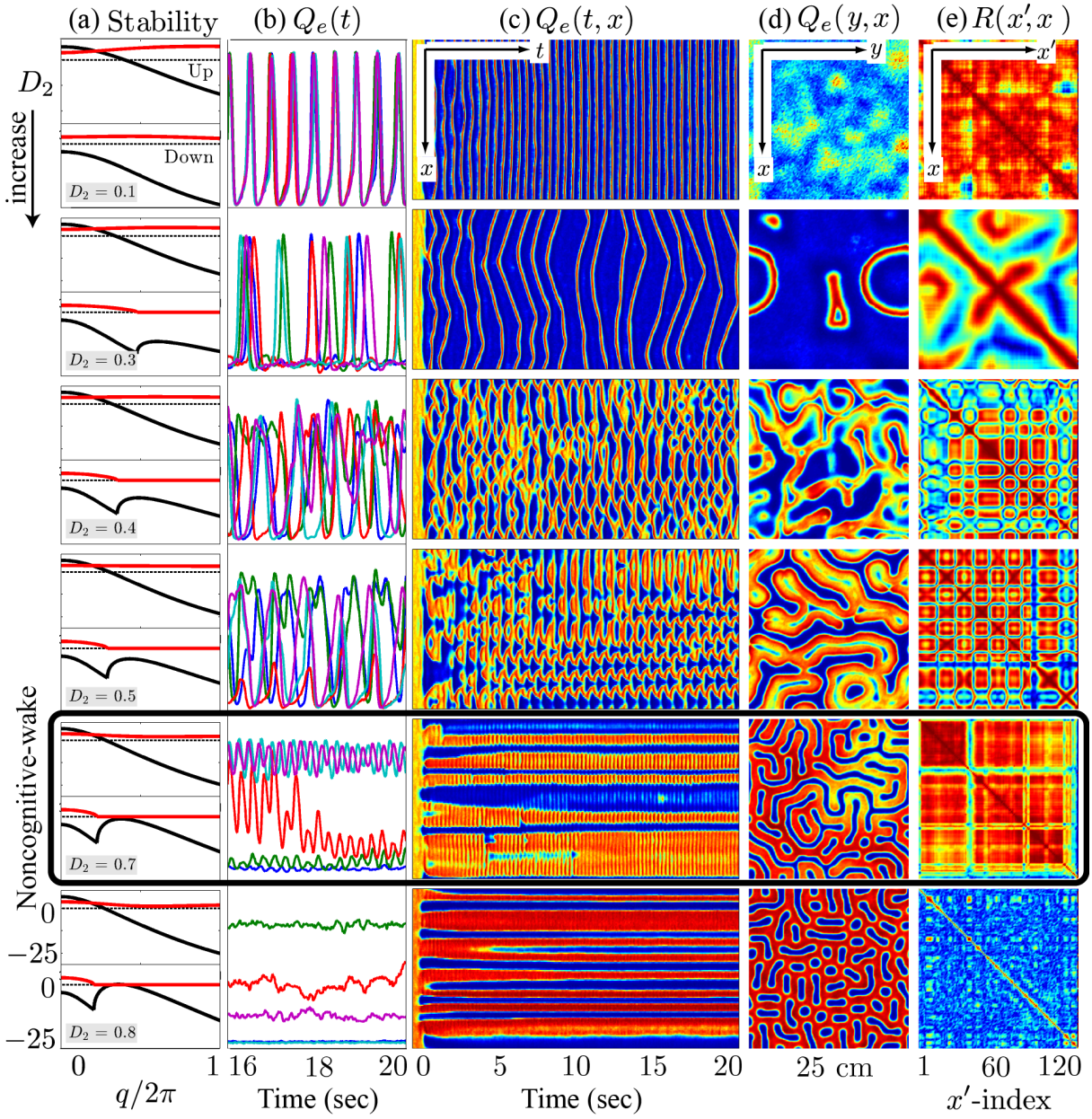


Figure 6.3: At the $\lambda_i = 1.0$ wake state, cortical stability analysis and spatiotemporal dynamics for varying gap-junction strength D_2 from 0.1 (top row) to 0.8 cm^2 (bottom). Model cortex is initialised from the top high-firing branch of steady-state manifold marked as "Awake" in Fig. 6.2. (a) Cortical stability analysis showing dominant eigenvalue dispersion curve of the real (black) and imaginary (red) parts as a function of scaled wavenumber for top- and bottom-branch equilibria at fixed anaesthetic effect $\lambda_i = 1.0$ in Fig. 6.2. Thus each panel has two parts in it—the upper part corresponds to the top-branch, the lower part to the bottom-branch. The dotted line is zero. (b) Last 4-sec time-series of excitatory firing-rate $Q_e(t)$ extracted from 5 equally-spaced grid-points in (c) $Q_e(t,x)$ space-time chart representing the full 20-sec time-evolution of cortical activity along the $y = 60$ midline strip. y-axis ranges from 0 to 30 s^{-1}. (d) Bird's-eye snapshot $Q_e(y,x)$ of the cortex when $t = 20$ sec. (e) Phase coherence map $R(x',x)$ showing synchronisation level of firing-rate between $Q_e(t,x)$ and $Q_e(t,x')$ for the final 5-sec time evolution. The coherence level is computed via Hilbert transform Eq. (6.5) with a transition from *red* to *blue* meaning high to low coherence. In (c)–(e), colour scale from blue to red indicates the numerical range from low to high. The size of the red-blocks is proportional to the coherence level. (Figure modified from [116])

6.6.2 Effect of inhibitory diffusion on "Comatose" cortex

Now let us examine the effect of variation of inhibitory diffusion on an anaesthetised cortex. The model cortex is initialised from a low-firing state at coordinate $(\lambda_i, \Delta V_e^{\text{rest}})$ = (1.018, 1.5 mV) in Fig. 6.2.

Too large or too small inhibitory diffusion leads to pathological brain state.

Analogue to the two extrema of the "awake" cortex in Fig. 6.3, the anaesthetised cortex with strong inhibitory diffusion $D_2 = 0.9$ cm^2 progresses to a Turing structure with low global coherence (bottom row in Fig. 6.4), while insufficient inhibitory diffusion ($D_2 \leq 0.1$ cm^2) suppresses cortical activity and maintains neuronal firing at the initial low-firing state, as seen in the top row; This contrasts with our previous observation on an "awake" cortex exhibiting highly coherent global oscillation with weak inhibitory diffusion.

Mechanism of anaesthetic slow-wave

For the anaesthetised cortex, anaesthetic effect $\lambda_i = 1.018$ is just beyond the multiple steady-states region where the awake cortex stays at the up-branch of $\lambda_i = 1.0$. This subtle change in coordinates causes the cortical stability to be guided only by the steady-state at the low-firing bottom branch. In Fig. 6.4, at the near-closure of the gap-junction $D_2 = 0.1$ cm^2, linear stability analysis (column (a)) predicts a heavily damped Hopf, which is consistent with the grid simulation. Most general anaesthetics will enhance the strength of the inhibitory postsynaptic potential (IPSP) [31, 58], as well inhibit gap-junction communication [142]. Consequently further increases in D_2 (for $D_2 < 0.7$ cm^2 of Fig. 6.4) lead the cortex into a chaotic phase, arising from the competitive interference between Hopf and Turing instabilities. Such mixed instabilities may provide a mechanism for the emergence of turbulent slow-waves of inductive anaesthesia, characterised by low phase-coherence. $D_2 = 0.7$ cm^2 is the border of the anaesthetic slow oscillations; larger values of D_2 (e.g., $D_2 = 0.8$ cm^2) rebalances the Turing and Hopf instabilities in favour of spatially structured Turing pattern oscillating at a low Hopf frequency (~ 3 Hz). Such mixed-mode interference is very similar to the noncognitive-wake cortex at $D_2 = 0.7$ cm^2 in Fig. 6.3. Nevertheless, because the cortex is still under anaesthetic coma, Steyn-Ross *et al.* label this coherent oscillation as "anaesthetic delirium", a clinical state common during emergence from general anaesthesia and associated with excitability and confusion [88].

Closure of gap-junctions in anaesthetic cortex does not promote seizure.

Comparing the weak inhibitory diffusion cases ($D_2 = 0.1$ cm^2) for Figs. 6.3 and 6.4, we see that the mode interplay between top-branch Hopf and bottom-branch damped-Turing instabilities is replaced by bottom-branch mode competition between damped-Hopf and damped-Turing. An attractor is formed to set the anaesthetic cortex to the homogeneous low-firing state. Scanning column (a) of Fig. 6.4 from bottom to top, we see that the

closure of gap-junctions does not bring seizure-like bursting oscillations, but completely suppresses cortical neuronal activity.

Conclusion

By forcing the "awake" cortex into "anaesthetic-coma" with a slight increase in anaesthetic effect λ_i, damped-Hopf and damped-Turing instabilities interact to destabilise the homogeneous state, leading to either quiescent silence at weak inhibitory diffusion $D_2 = 0.1$ cm^2, or a mixed mode of balanced Turing and Hopf at $D_2 = 0.8$ cm^2 that is similar to a "noncognitice" cortical state (see Fig. 3.10), while the intermediate values of D_2 give rise to incoherent chaotic oscillations. Further increasing D_2 to 0.9 cm^2 triggers a spatially structured Turing pattern. The modelling of the cortex under anaesthetic coma implies a prediction that seizure is suppressed by closure of gap-junctions.

6.7 Global coherence for the cortex at "Awake" and "Coma"

A comprehensive inspection of the global coherence relating to changes in inhibitory diffusion is presented in Fig. 6.5. Here, we simply computed the global coherence at a given value of D_2 by taking the mean of the upper-triangle of the coherence matrix $R(x', x)$ defined in Figs. 6.3(e) or 6.4(e).

Let us first examine the mean neuronal synchronisation across the whole cortex relating to different levels of inhibitory diffusion, when the cortex is at an awake state. We see a high global coherence in the non-cognitive state, where the inhibitory diffusion is moderately strong $D_2 \simeq 0.7$ cm^2. The global coherence is reduced when the inhibitory diffusion is in the intermediate range 0.2 cm^2 $\lesssim D_2 \lesssim$ 0.6 cm^2; and back to rather higher values in the seizure state with the closure of gap-junctions $D_2 < 0.2$ cm^2. In the intermediate range of D_2, the reduced global coherence manifests with spatiotemporally chaotic pattern dynamics, which we take to be an early warning sign of upcoming seizure with reduced gap-junction strength.

For the anaesthetised cortex, the anaesthetic drug promotes two major effects: complete elimination of seizure at low inhibitory diffusion values ("Suppression": $D_2 \lesssim$ 0.4 cm^2), leading to rather smaller globally coherent levels of cortical activations; and a shifting of the activated "Noncognitive-wake" coherence peak to the right, implying a possible hysteresis effect such that an anaesthetised cortex requires a stronger Turing instability to reinforce an activated state. To the left of the peak for the delirium state, there is a broad intermediate zone of D_2 experiencing reduced coherence, which results from large, low frequency chaotic oscillations.

The marked contrasts between the anaesthetic slow-wave and delirium (activated cortex) are visualised in Fig. 6.6 (bottom row): The slow-waves are almost 6 times larger

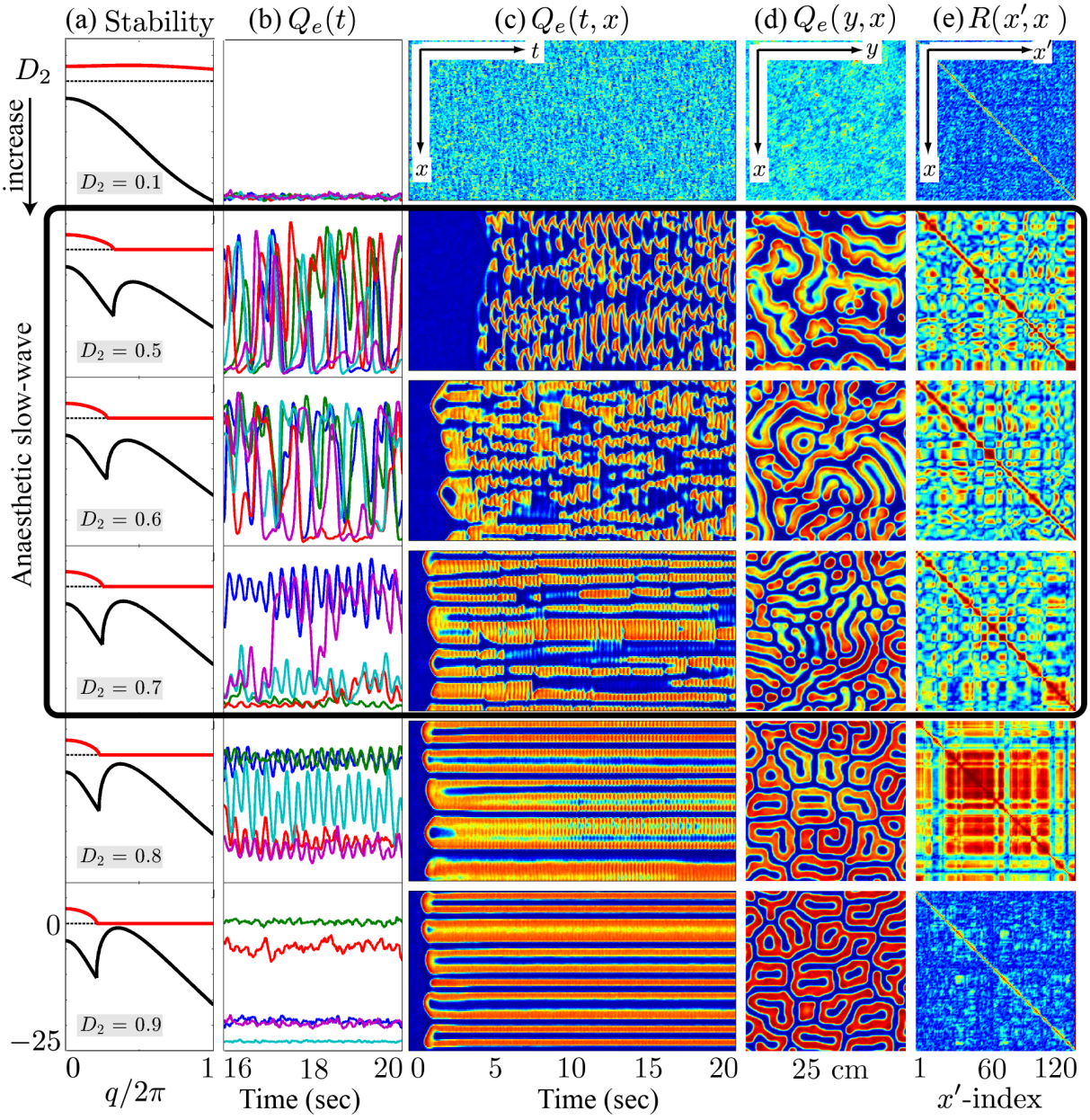


Figure 6.4: At the $\lambda_i = 1.018$ coma state, cortical stability analysis and spatiotemporal dynamics of varying gap-junction strength D_2 from 0.1 (top row) to 0.9 cm^2 (bottom). Model cortex is initialised from the bottom low-firing branch of steady-state manifold marked as "Coma" in Fig. 6.2. (a) Cortical stability analysis showing dominant eigenvalue dispersion curve of the real (*black*) and imaginary (*red*) parts as a function of scaled wavenumber at anaesthetic effect $\lambda_i = 1.018$ in Fig. 6.2. (b) Last 4-s time-series of excitatory firing-rate $Q_e(t)$ extracted from 5 equally-spaced grid-grids in (c) $Q_e(t,x)$ space-time chart representing the full 20-s time-evolution of cortical activity along the $y = 60$ midline strip. (d) Bird's-eye snapshot $Q_e(y,x)$ of the cortex when $t = 20$ s. (e) Phase coherence map $R(x',x)$ showing synchronisation level of firing-rate between $Q_e(t,x)$ and $Q_e(t,x')$ for the final 5-s time evolution. In (c)–(e), colour scale from blue to red indicates the numerical range from low to high. (Figure modified from [116])

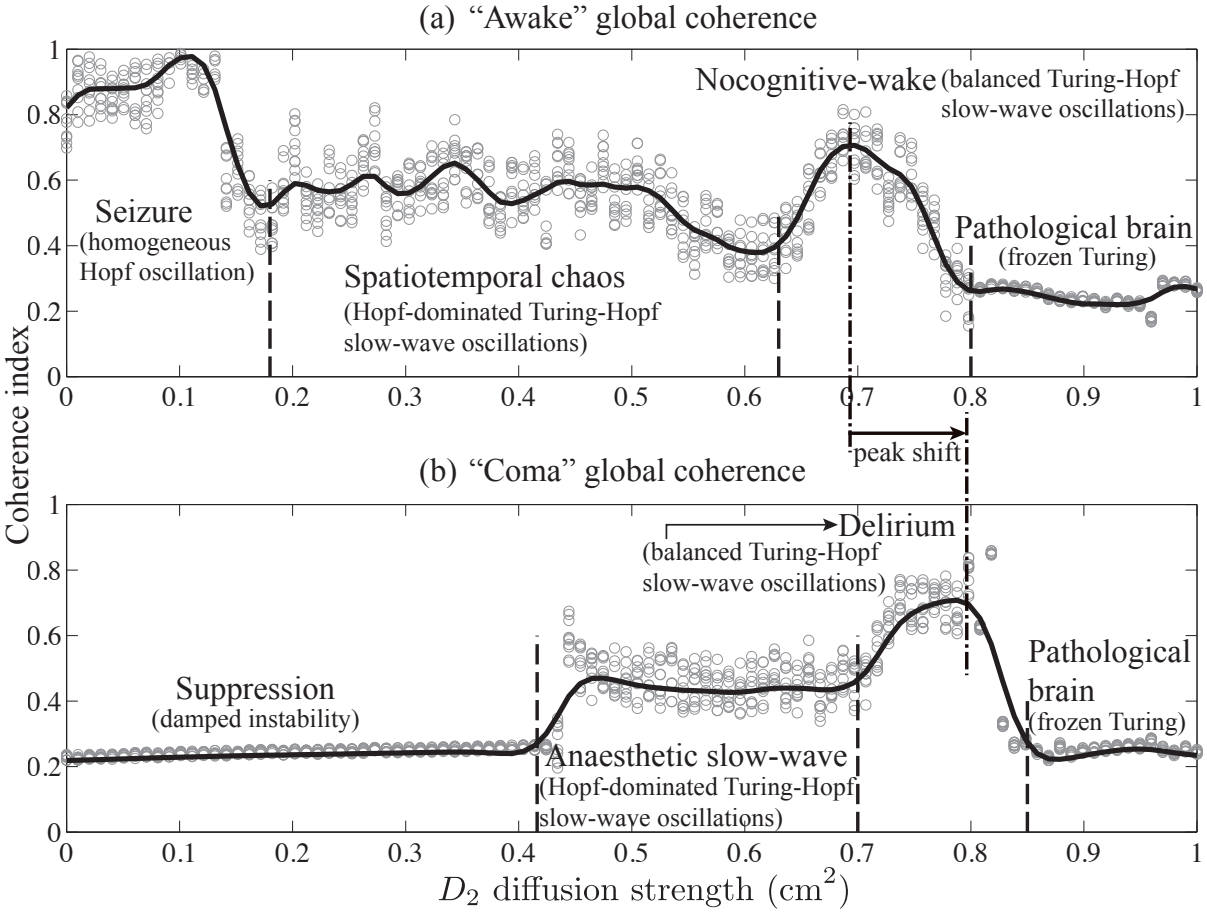


Figure 6.5: Global phase-coherence trends with respect to inhibitory strength for the cortex at (a) awake ($\lambda_i = 1$) and (b) comatose ($\lambda_i = 1.018$) states. Inhibitory strength D_2 is evenly spaced (0.01 cm^2 interval) in the range 0.0 to 1.0 cm^2. At a given D_2, simulations were repeated 10 times. For each simulation, we first computed the phase-coherence matrix $R(x', x)$ for the final 5-s time evolution (see Figs. 6.3(e) and 6.4(e)), then extracted its upper-triangular matrix mean as an estimate of global phase-coherence, which is represented as a gray circle in the figure. The trend curves were produced by `spline` function in MATLAB curve-fitting toolbox. (Figure modified from [116])

in power while 40 times smaller in the dominated frequency than the anaesthetic delirium. The slow-waves have a broad energy distribution over the low-frequency domain $0 \lesssim f \lesssim 2$ Hz, whereas the activated brain state (delirium) evinces a strong resonant frequency at $\sim$4.3 Hz.

These model results allow a promising prediction that the passage from wake to anaesthetic unconsciousness should manifest as a decrease in phase coherence between separated cortical electrodes.

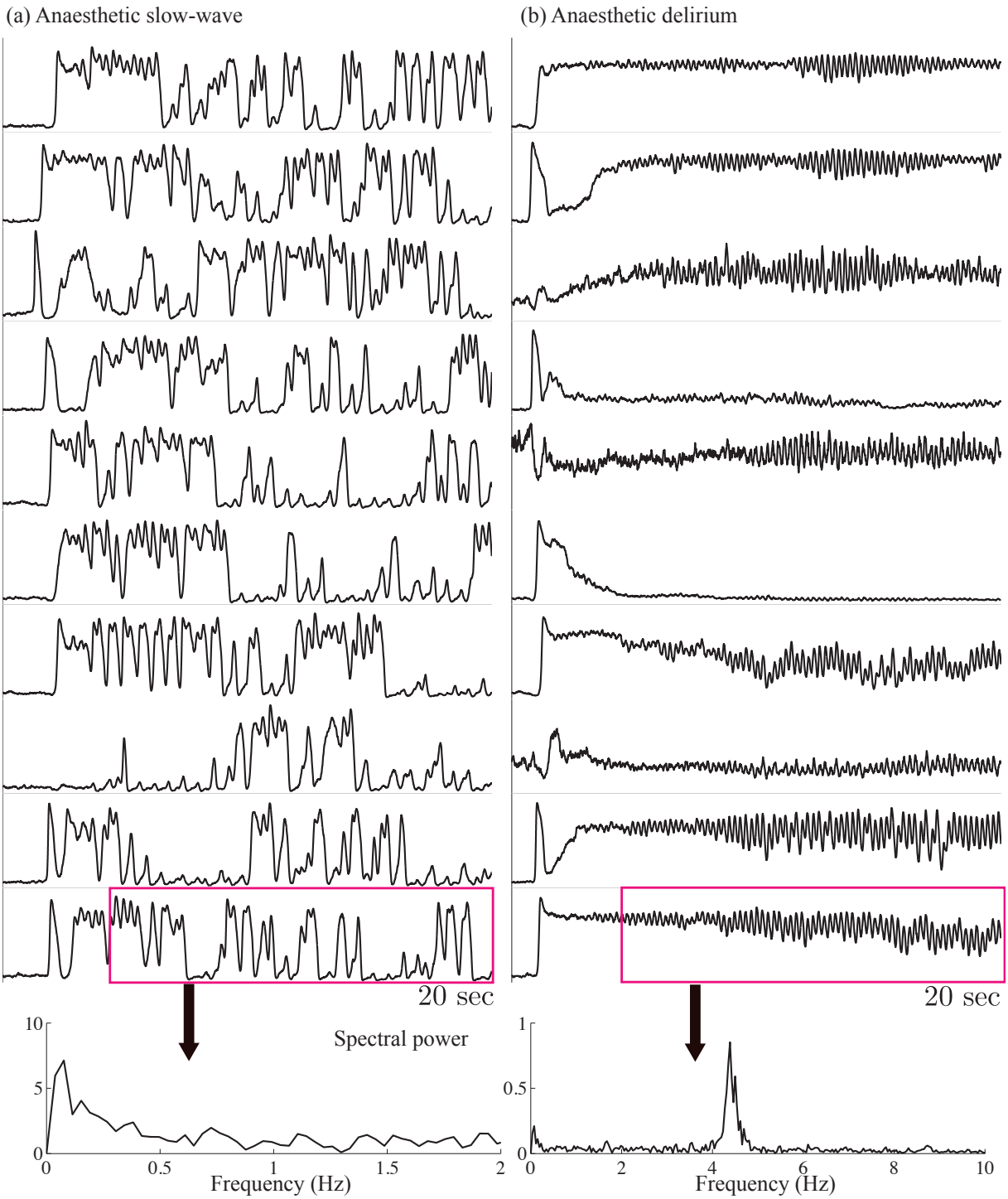


Figure 6.6: Q_e time-series of 10 equally-spaced grid-points along the vertical midline of the cortical grid for (a) Anaesthetic slow-wave ($D_2 = 0.6$ cm^2) and (b) Anaesthetic delirium ($D_2 = 0.8$ cm^2). The Fourier spectral power for two sampled time-series are displayed at the bottom. (Figure modified from [116])

6.8 Mode stability analysis for the anaesthetic cortical model

A pattern-forming system usually yields a pair of complex amplitude equations representing the dynamics of the Turing–Hopf (TH) mode. In Chapter 5, we discussed the mechanism of mode stability by analysing the amplitude equations. The existence of the TH mode depends on the sign of Δ (see Eqs. (5.20a, 5.20b)) — a parameter derived from the amplitude equations; the TH mode exists only when Δ is positive (see Fig. 5.4).

Using our multiple-scale expansion algorithm, `Amp_solving`, we derived the TH mode amplitude equations for the cortical model at "Awake" and "Coma" states, and their corresponding bifurcation diagrams, which are presented in Fig. 6.7. Although the "Awake" (Fig. 6.7(a)) and "Coma"(Fig. 6.7(b)) TH mode stability trends are broadly similar, the effect of anaesthetic mainly shifts the TH region (solid Turing–Hopf curves) to the right, indicating the dependence on rather larger inhibitory diffusion for the TH mode in comatose cortex.

In Fig. 6.7, we see that the TH mode exists in a particular D_2 region. Within this D_2 region, Fig. 6.5 shows that the chaotic evolution induced coherence drops. Meanwhile, we see a subcritically stable Hopf mode colliding with the TH mode, possibly implying a Hopf-dominated Turing–Hopf mechanism for the chaotic slow-waves. Since de Wit suggested that the unstable Hopf phase leads to a chaotic mode, we tested the phase stability conditions (5.31a, 5.31b) and (5.32) but all failed to produce correct predictions. To find a chaotic mode of the cortical model, one may need to manually scan the parameter space close to a CTHP.

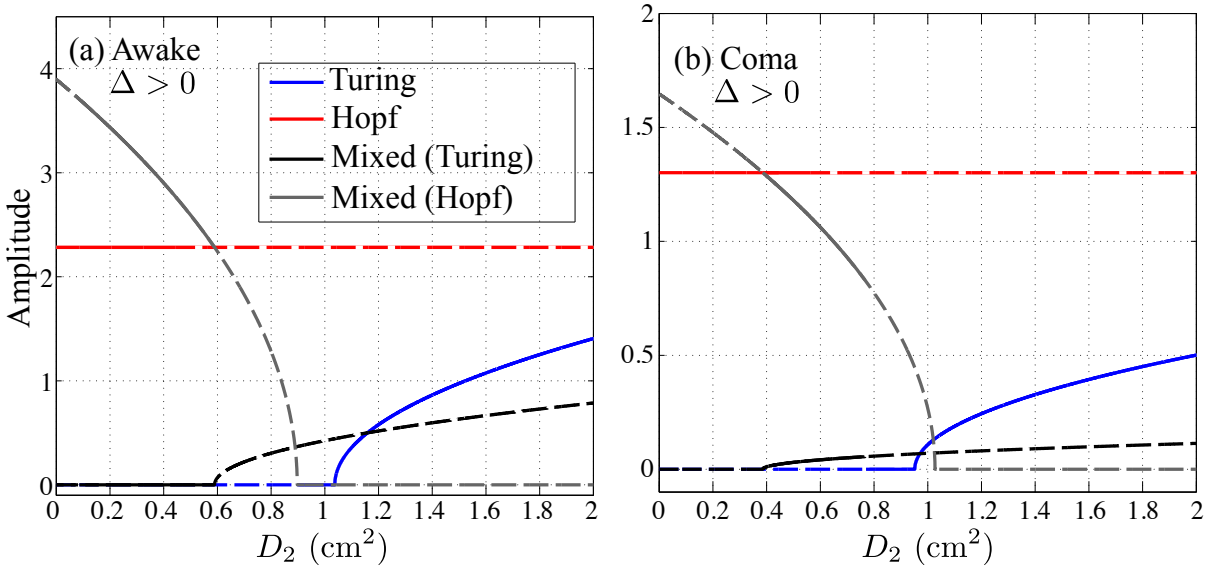


Figure 6.7: TH mode stability with respect to the inhibitory diffusion D_2 for (a) "Awake" and (b) "Coma" cortex. Mode stability is represented by solid (stable mode) or dashed (unstable mode) curves. For both brain states, Δ is positive, meaning the existance of a TH mode (see Fig. 5.4(b)).

6.9 Phase-coherence analysis of the clinical EEG with induced propofol anaesthesia

In this section, we will analyse clinically recorded EEG to test the model predictions for phase-coherence changes during the transition from consciousness to anaesthetic unconsciousness.

6.9.1 Materials

The EEG dataset used in this study are archived files from Waikato Clinical School, Hamilton, New Zealand, and were previously used in investigating the anaesthetic response of EEG in different frequency bands [54]. The dataset contains pairs of 60-s EEG (sampling frequency 250 Hz) recordings for two distinct brain states: wake and propofol anaesthetic coma, recorded from 5 healthy adult subjects via 129 electrodes[1] using an EGI^{TM} dense array with Cz (vertex) being the reference electrode. The archival EEG dataset are manually selected epochs that are relatively artifact-free.

An example of EEG recorded from electrode Fp1 is represented in Fig. 6.8. This demonstrates the transition from wakefulness (upper EEG trace) to sedated unconsciousness (lower trace) with the appearance of spindles (12–15 Hz) and slow rhythms including delta activity (1–4 Hz) and slow oscillations (0.2–1 Hz). By focusing on the EEG in sub-delta band ($\leq$ 1.5 Hz), Fig. 6.9 shows that the power of the slow-waves in sedated unconsciousness is nearly twice as large as that in the wake state.

6.9.2 `EEG_coherence`: An automatic EEG processing algorithm for EEG coherence analysis

The raw EEG data were visually inspected and the artifacts were manually marked using `EEGLAB`[3]. The one or two bad channels were replaced by substituting with the average of the 4 neighbouring channels. Eye-blink artifacts were removed by `AAR` (see Fig. 6.8 description) then examined (see Fig. 6.8 for an `AAR` correction example). Since the archival EEG data are relatively artifact-free, only minor corrections were needed. Besides, we verified that `AAR` operation does not affect phase-coherence between EEG channels by comparing the statistical difference for EEG phase-coherence between raw and `AAR`-corrected epileptic EEG data[4]. Indeed, phase-coherence is an important measure and must be preserved by the artifact removal process [160]. We then filtered EEG to the sub-delta band

[1]The electrodes map is available at `http://psychophysiology.cpmc.columbia.edu/software/CSDtoolbox/tutorial.html`

[2]Automatic Artifact Removal toolbox, an `EEGLAB` plug-in available at `http://www.germangh.com/eeglab_plugin_aar/index.html`

[3]An open source EEG processing MATLAB toolbox available at `http://sccn.ucsd.edu/eeglab/`

[4]Data available from `AAR` authors' website `http://kasku.org/projects/eeg/databases.htm#epilepsy`

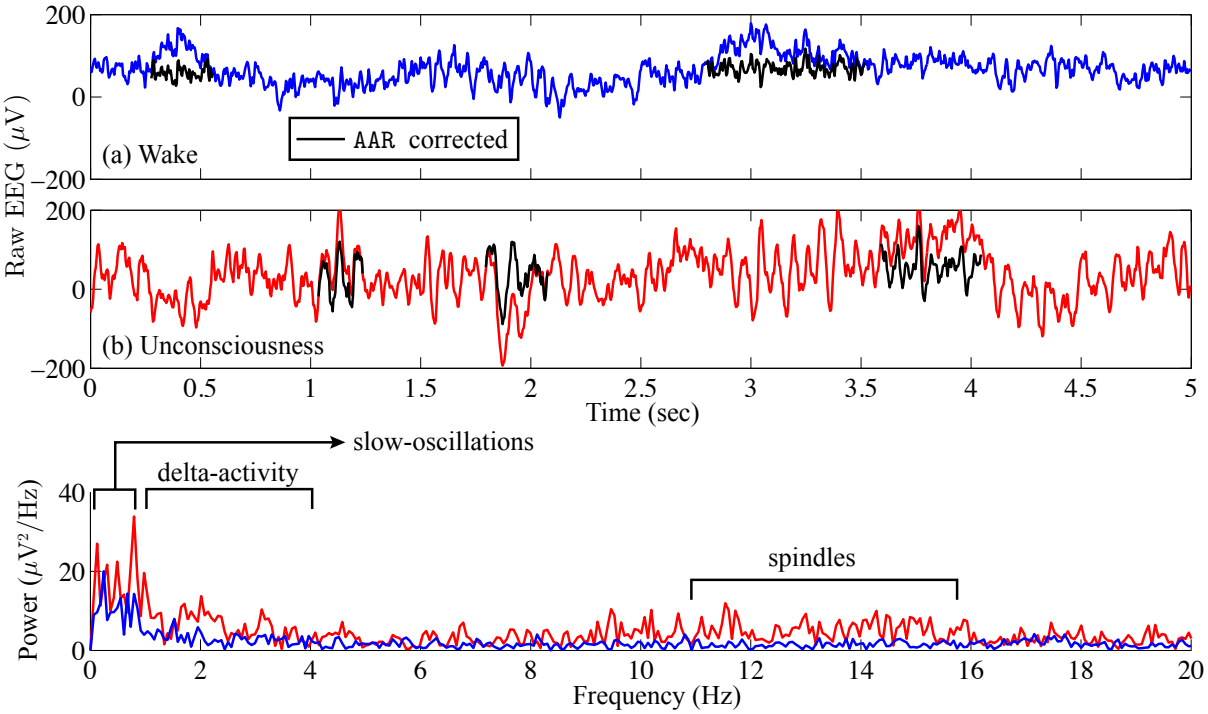


Figure 6.8: Sample wake (blue) and sedated unconsciousness (red) EEG from archival Fp1 recording. Raw EEG data are filtered via `AAR`[2] to remove the eye-blink artifacts. `AAR` corrected EEG are marked in black. The power spectra show that slow-wave oscillations are dominant in the sedated unconsciousness state.

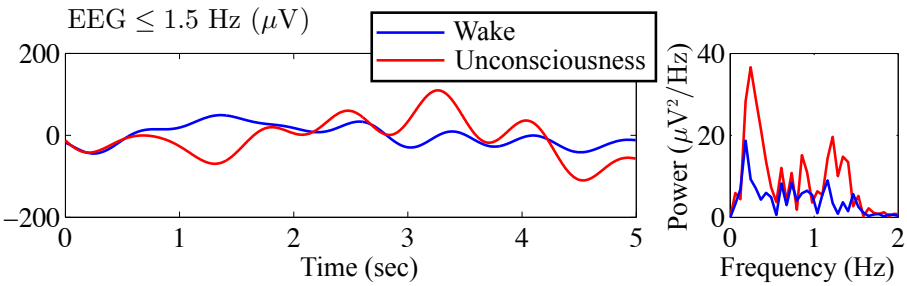


Figure 6.9: Filtered Fig. 6.8 EEG in sub-delta band ($\leq$ 1.5 Hz) and corresponding power spectra revealing a strong slow-wave ($\sim$ 0.3 Hz) in the sedated unconsciousness state.

($\leq$ 1.5 Hz) using `EEGLAB` built-in basic order-2 FIR (linear finite impulse response) filter with the pass-band between 0.05 and 1.5 Hz. During filtering, `EEGLAB` uses the MATLAB routine `filtfilt()` applying the filter forward and then backward, to ensure that phase delays introduced by the filter are nullified. The obtained sub-delta band EEG are characteristic of slow oscillatory activity, which is the most prominent feature of the EEG activity during the non-rapid eye movement (non-REM) sleep in humans (see Fig. 6.9 for an example) [73].

The `EEGLAB` pre-processed EEG data were then passed to `EEG_coherence`, a custom MATLAB algorithm that identifies electrode-pairs with significantly altered phase-coherence between the two brain states. The user specifies the folder location where the EEG data are stored and configures some basic parameters (e.g., window and overlap

length for the coherence measure). `EEG_coherence` automatically generates a summary table including identified electrode-pairs and their corresponding phase-coherence indices at two distinct brain states for all subjects. p-values that are used to identify those electrode-pairs whose phase-coherence has significantly altered are included in the table to permit further statistical analysis.

As shown in Fig. 6.10, `EEG_coherence` processes EEG data in three steps:

1. **Construction of coherence matrices**
   The measure of phase-coherence is based on the Hilbert transform, as described in Sec. 6.5.1. Each subject will have two coherence matrices `awake_pair` and `sleep_pair`, respectively for the wake and sedated unconsciousness (i.e., deep sleep or unconsciousness) states.

2. **Extraction of coherence summaries**
   At each brain state (wake or unconsciousness), `EEG_coherence` will define a summarised coherence matrix by integrating all subjects' coherence matrices for this state. This summary matrix has three dimensions: the first dimension (row-index k) identifies the subject; the second dimension (channel-index m) and the third dimension (base-channel index n) coordinate the coherence between EEG channel pairs m and n. Thus, coordinate (k, m, n) indicates the coherence $R_{(k,m,n)}$ between EEG channels m and n for subject k. Since we only consider the upper triangle of the coherence matrix, the redundant coherence[5] in the summarised matrix will be filled with `NaN` (not a number). The output from this step is a pair of coherence summary matrices for wake and unconscious states.

3. **Statistical comparison**
   A one-tail Mann-Whitney U-test is performed to scan the coherence for all electrode-pairs between two summarised coherence matrices over the base-channels. We tested the null hypothesis that the coherence in two coherence summary matrices are samples with equal medians, against the alternative that they are not. More specifically, the alternative hypothesis is that the median of coherence in one coherence summarising matrix is greater (right-tailed; or smaller, left-tailed) than the median of those in the other matrix. The statistical comparison for a base-channel n returns a p-strip containing p-values for channel-pairs: 1-n, 2-n,...,128-n. All p-strips forms a three dimensional matrix, which is obtained via the following MATLAB implementation:

```
awake_size = size(awake_pair);
prop_size = size(sleep_pair);

% Check if two coherence matrices have the same size
if ~isempty(find((awake_size == sleep_size)==0))
```

[5]The lower triangle of the coherence matrix $R_{(k,n,m)}$ and the diagonal unit coherence.

```
    error('unequal size');
end

% Create p-strip matrix
for base_ch = 1: size(awake_pair, 3)
    for ch_ind = 1: size(awake_pair, 2)
        if isnan(awake_pair(:,ch_ind, base_ch))
            p(:,ch_ind, base_ch) = NaN;
        else
            [p(:,ch_ind, base_ch), h(:,ch_ind, base_ch)]...
                = ranksum(awake_pair(:,ch_ind, base_ch),
                          sleep_pair(:,ch_ind, base_ch),...
                          'alpha', p_limit, 'tail', direction);
            % direction: left: awake < sleep; right: awake > sleep
        end
    end
end

% Squeeze the 3D p-strip matrix, leading to a 2D p-matrix
p_matrix = squeeze(p);  % e.g. E1-E2 is at row 2 (channel), col 1 (base-channel)
```

If, across all subjects, a given electrode-pair shows a statistically significant difference in coherence between wake and unconscious state (i.e., $p <$ `p_limit`), `EEG_coherence` will store this electrode-pair in the summary table.

6.9.3 Results

Figure 6.11 visualises those electrode-pairs identified by `EEG_coherence` as having significantly altered (i.e., decreased or increased) coherence between wake and unconscious states. The comparison between the upper and lower panels of Fig. 6.11 reveals two major features of the coherence changes with respect to propofol anaesthesia:

- **Decreased coherence for frontal and occipital electrode-pairs**
  The electrode-pairs with reduced coherence are densely clustered at the frontal and occipital areas of cortex. Similarly, electrode-pairs connecting the frontal and occipital regions show significantly reduced coherence, indicating that neuronal activities between them are less strongly coupled when the brain is switched to a unconscious state. Scanning the top panels of Fig. 6.11 from left to right, we see that the front electrodes manifest the highest significant level in decreased coherence, in other words, propofol anaesthesia leads to increased disorder in neuronal activity in the frontal cortex.
- **Increased coherence for left- and right-temporal electrode-pairs**
  Electrodes at the left- and right-temporal areas detect enhanced coherence. These maps of enhanced connectivities seem to be complementary to the preceding maps showing decreased frontal-occipital connectivity: coherence trends have been reversed with the significant front-back *uncoupling* (top panel) occurring simultaneously with a left-right *coupling*. Examining the lower panels of Fig. 6.11, we see

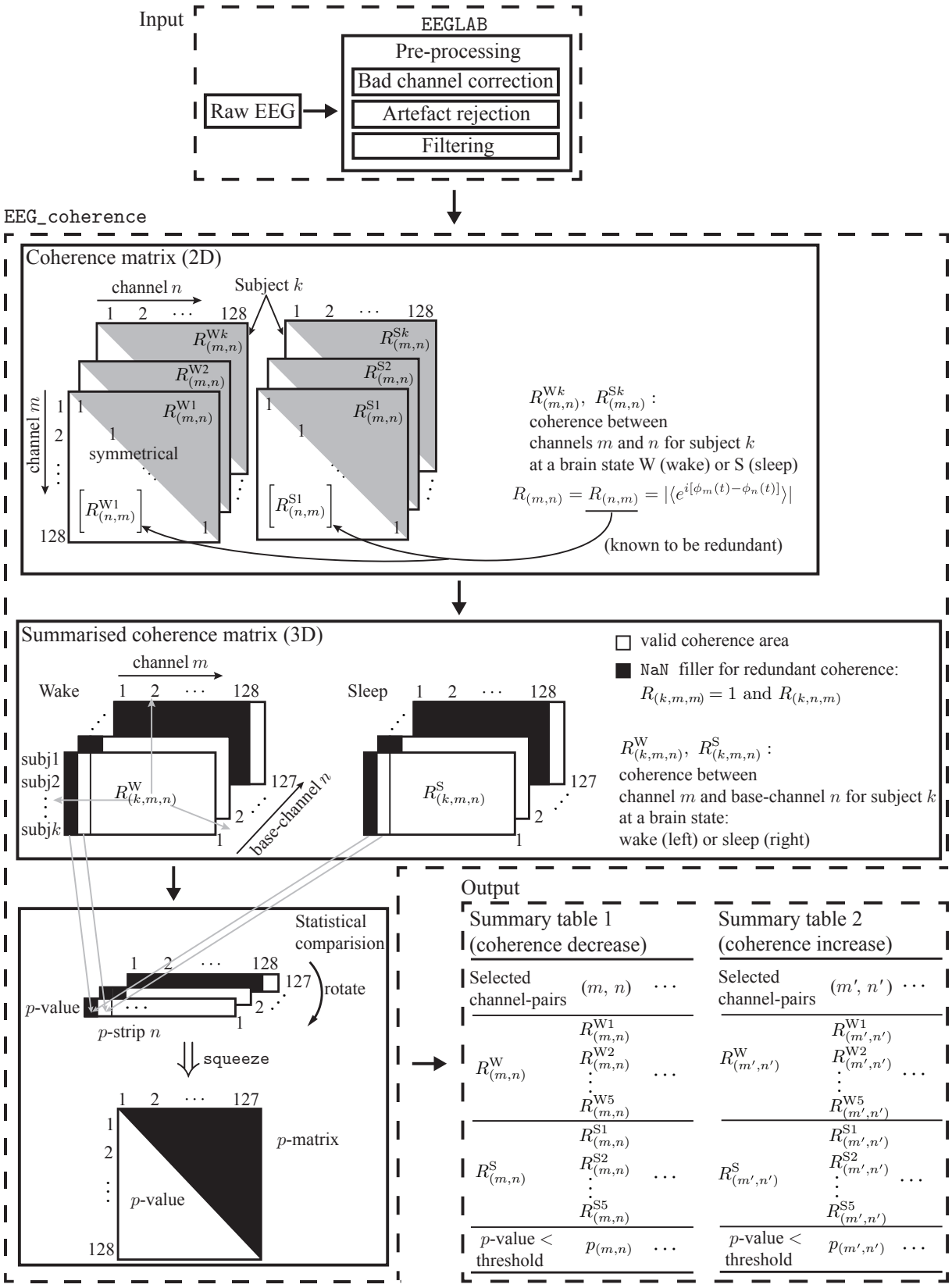


Figure 6.10: Flowchart for processing EEG of two brain states to determine electrode-pairs with significantly altered phase-coherence. EEG data undergo a pre-processing in EEGLAB before passing to EEG_coherence, a customised MATLAB algorithm automatically identifying electrode-pairs with significantly altered phase-coherence between two brain states across multiple subjects, and store these electrode-pairs in summary tables.

evidence of strengthened left-right electrode connectivitiy, showing increased EEG coherence with the induction of propofol anaesthesia.

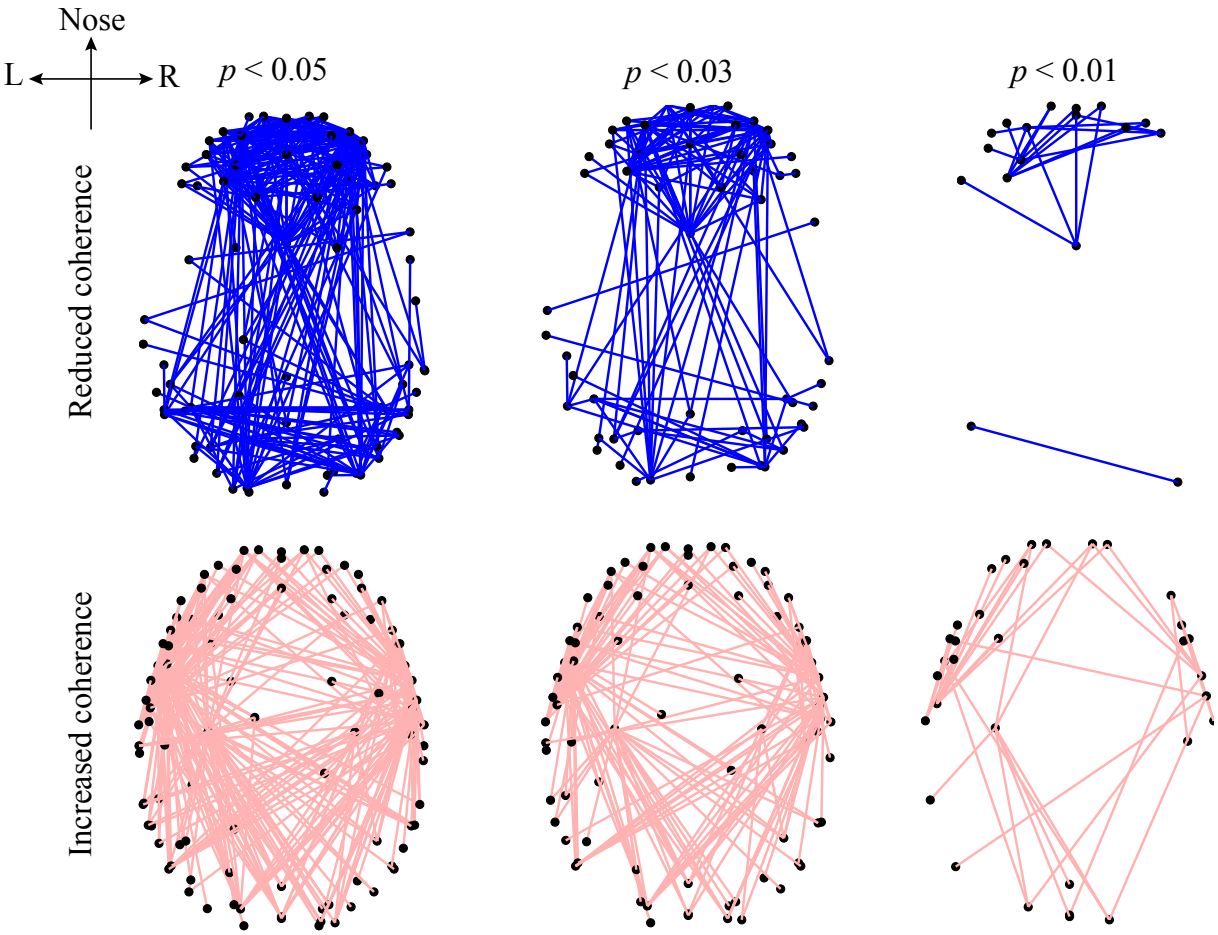


Figure 6.11: Graphical representations of the electrode-pairs with significantly *reduced* (upper panel) or *increased* (lower pannel) phase-coherence of the sub-delta band (0.05–1.5 Hz) EEG induced by propofol anaesthesia. EEG data (129-channel recording) were recorded from 5 subjects and processed by the `EEG_coherence` algorithm diagrammed in Fig. 6.10. The electrode-pairs with significant ($p < 0.05$) changes in phase coherence are connected with lines. The electrode-pair map is represented in a bird's-eye view of the 3D head model (created via the modified `EEGLAB` function `plotchans3d`). The black dots are `EEG_coherence` selected electrodes. Electrode-pairs for altered phase coherence are determined with different levels of significance (the significant level p was set at 0.05, 0.03 and 0.01 in the Mann-Whitney U-test). Smaller p thresholds result in a lower density of electrode-pair cluster due to the stricter selection criterion, however, the electrode-pair distributions are generally preserved in trend.

If we overlap the upper and lower panels of Fig. 6.11, we find some frontal electrodes have decreased coherence with the occipital electrodes, while having increased coherence with the left- and right-temporal electrodes. Similarly, some occipital electrodes have decreased coherence with the electrodes in the frontal area, while having increased coherence with those in the temporal areas. These observations suggest an underlying compensatory mechanism between a subsystem of fronto-occipital and other cortical regions at sub-delta

frequencies. Cantero *et al.* [8] reported a similar compensatory phenomenon in coherence between the temporal and other cortical regions for the alpha (8–12 Hz) and sleep spindle (12–15 Hz) frequency ranges.

Furthermore, we examined the EEG decreased coherence pattern by nine electrodes (see the description of Fig. 6.12) that Koskinen *et al.* utilised in their work [60], which identified electrode-pairs with sub-delta EEG coherence decrease due to propofol-induced anaesthesia. We set the significance level ($p < 0.05$) in `EEG_coherence` to be the same as that used by Koskinen *et al.* The comparison result shown in Fig. 6.12 illustrates that `EEG_coherence` produced a similar electrode-pair distribution pattern to the Koskinen result, which reinforces our finding about the sub-delta EEG coherence reduction in the frontal cortex.

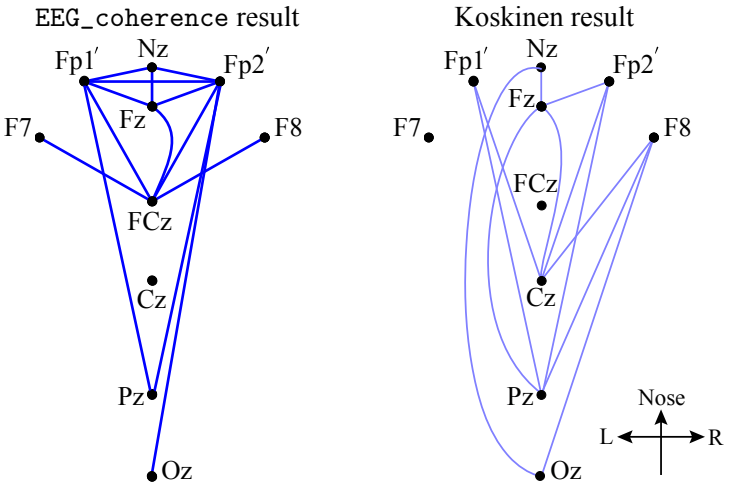


Figure 6.12: A subset of electrode-pairs (left) showing significant ($p < 0.05$) reduction in phase coherence extracted from the upper left corner plot of Fig. 6.11 (referenced to Cz, in dark blue lines) and Koskinen *et al.* reported pattern [60] (right, referenced to FCz, in light blue lines) for the coherence measured from 9 electrodes: Nz (nasion), Fp1′(about 1 cm down from Fp1, just above the eyebrow), Fp2′, Fz, F7, F8, Cz, Pz and Oz.

6.10 Discussion

In this chapter, we first reviewed the pattern dynamics of an anaesthesia model of the cortex. The anaesthetic effect was introduced into the cortical model (SR2007 model) through an inhibitory drive λ_i, an increase of which suppresses cortical activity, leading to an unconscious state. A transition from unconsciousness to consciousness requires sufficient depolarisation to the cortical excitatory membrane potentials, which was modelled by raising the soma membrane resting potential offset ΔV_e^{rest}, an excitatory drive. The inhibitory drive λ_i and excitatory drive ΔV_e^{rest} define a reversed-S manifold (Fig. 6.1) of the cortical steady-states. With a fixed $\Delta V_e^{\text{rest}} = 1.5$ mV (see Fig. 6.2), we identified an "Awake" state ($\lambda_i = 1$) at the high-firing branch and a "Coma" state ($\lambda_i = 1.018$)

at the low-firing branch for the homogeneous model cortex. For both cortical states, we investigated the effect of variation in the strength of inhibitory diffusion D_2 on the cortical dynamics.

At the modelled "Awake" state (Fig. 6.3), a strong inhibitory diffusion generates a Turing spatial cortical pattern while a weak inhibitory diffusion leads to a homogeneous global oscillation. We identified these two extrema as pathological brain states: The frozen Turing pattern lacks any temporal evolutions, so blocks information flow[6]; the highly synchronised whole-of-cortex oscillation (top row of Fig.6.3) favours abrupt seizure. Intermediate values of inhibitory diffusion give rise to spatiotemporal chaos which we postulated to be a competition between Turing and Hopf modes. The model prediction that the abrupt seizure is preceded by a chaotic regime with low coherence is consistent with a series of clinical reports [2, 11, 79].

When there is little anaesthetic effect, a sufficiently strong inhibitory diffusion (i.e., gap-junction strength) allows a bistability between the Turing pattern and Hopf oscillations, leading to slow Hopf oscillations of high global coherence with the spatial structure maintained (see $D_2 = 0.7$ simulation in Fig. 6.3). Such interacting low-frequency Hopf and Turing instabilities may form the substrate for the cognitive state, namely, the "default" background state for the non-cognitive brain during wake. Its slow beating dynamics ($\leq$0.1 Hz) is similar to what is observed in BOLD functional MRI recording of relaxed, non-tasked human brains [30, 32].

At the modelled "Coma" state (see Fig. 6.4), a strong inhibitory diffusion again encounters a pathological brain state dominated by the Turing mode while the weak inhibitory diffusion suppresses cortical activity completely, leading to a quiescent EEG trace (top row of Fig. 6.4). Compared to the global oscillations seen in the awake cortex with weak inhibitory diffusion, the induced propofol anaesthesia completely suppressed seizure.

An increase in anaesthetic effect λ_i suppresses cortical activity, leading to an anaesthetised coma state. Here, intermediate values of D_2 are expected since propofol blocks gap-junctions. The weak gap-junction connections allow spontaneous emergence of large-amplitude slow chaotic oscillations (see the highlighted simulations in Fig. 6.4). This mechanism for the slow oscillation is quite distinct from the conventional view of a cyclic alternation in extracellular ionic (Ca^{2+}) concentrations [73] or a recent view of an alternation of numerous pacemakers played by tiny clusters of neurons in the cerebral cortex [122].

These cortical dynamical patterns, from the view of mathematics, are governed by bifurcation theories. We applied linear stability analysis (the first column of Figs. 6.3 and 6.4) to determine the dominant modes for observed cortical instabilities. However,

[6]Steyn-Ross *et al.* [116] proposed that the cortex with a Turing structure may possibly be relevant to early brain development when the immature brain is richly endowed with gap junctions. Such abundent gap junctions favour the formation of permanent Turing patterns.

predictions of linear stability analysis do not always agree with cortical simulations. For example, we started an "awake" cortex simulation from a high-firing steady-state (Fig. 6.2), where the linear stability analysis predicts an emergent Hopf instability for all D_2 values in Figs. 6.3, whereas the cortical patterns experienced spatiotemporal turbulence for intermediate values of D_2, and maze-like labyrinthine Turing structure for larger values of D_2. Examining closely the last row of Fig. 6.3, the linear stability analysis at the low-firing branch predicts a weak Turing instability, which is likely to be responsible for the observed Turing patterns. However, at the onset of the multiple-instabilities, how to determine the dominant instability (i.e., the more stable mode): Hopf at the top-branch or Turing at the bottom-branch? We cannot infer this information from the linear stability analysis. We experienced the same issues in predicting the pattern dynamics at a "coma" cortex (Fig. 6.4): there seems no confident links between the linear stability analysis and the actual pattern dynamics. To address these problems, a weakly nonlinear analysis (see Sec. 6.8) was then carried out. Assisted by the multiple-scale expansion algorithm `Amp_solving`, TH mode amplitude equations in the vicinity of a CTHP were derived. A mode stability analysis (see Fig. 6.7) was then performed based on the amplitude equations, to determine the dominant modes at a bifurcation setting. In the D_2 region where the chaotic mode exists, we applied phase stability conditions, but they also could not offer correct predictions. By observing the bifurcation diagram Fig. 6.7, we argue that the chaotic mode may occur when a strong Hopf mode collides with a TH mode, and the interaction between them may lead to a strong nonlinearity disturbing the Hopf phase stability.

On the other hand, the cortical simulations revealed another feature characterising the transit from wake to anaesthetic unconsciousness (see Fig. 6.5): A movement from the "noncognitive-wake" high coherence peak to a "anaesthetic slow-wave" region with diminished coherence. We analysed the changes in coherence for archival human electroencephalogram (EEG) recorded from volunteers undergoing propofol anaesthesia, looking for confirmation of the model prediction of a drop in coherence during the slow-wave oscillations. In this study, we investigated systematic phase-synchronisation changes between pairs of EEG channels in the sub-delta band during propofol anaesthetic induction. An EEG phase-coherence processing algorithm, `EEG_coherence`, was developed in MATLAB and applied to the archival EEG data from a group of subjects. `EEG_coherence` uses the Hilbert transform to extract instantaneous phase-angles from nonstationary EEG signals, and yields a phase-coupling index appraising the phase-shift consistency between pairs of EEG channels. The trends of such EEG coherence change between two brain states are statistically tested via a Mann-Whitney U-test, which is a simple non-parametric test but convincing even for small data sets lacking assumptions of distribution (e.g., the data sets do not necessarily to be normal distribution). Our EEG examination discloses a coherence

decrease in the frontal and occipital regions, as well in the connection between them. Simultaneously, more strongly coupled neuronal activities are found in the temporal-frontal, temporal-occipital and left-right temporal regions. Such contrasts in coherence changes suggest an underlying compensatory mechanism of the sub-delta band activities between a subsystem of fronto-occipital and cortical temporal regions. Our findings of reduced-coherence between particular electrode-pairs is similar to clinical reports [60, 78] where the frontal cortical region exhibits a negative inter-correlation during anaesthetic coma. Moreover, the presented reduced phase-coherence along the fronto-occipital axis is consistent with an animal study by Imas *et al.* [47] that the anterio-posterior coherence in both 5–25 Hz and 26–50 Hz bands was significantly reduced by isoflurane in the rat. In contrast, Dumermuth and Lehmann [26] reported a high interhemispheric coherence between the left and right parietal areas with deepening slow wave sleep. They postulated that the high coherence may reflect the interhemispheric transfer of information. Later, research by Mölle *et al.* [77] reinforced Dumermuth and Lehmann's findings and verified their hypothesis by comparing coherence changes for subjects during the slow-wave sleep with or without pre-learning tasks. Mölle *et al.* observed significantly increased coherence during the occurrence of slow oscillations (< 1 Hz) for subjects after learning tasks (Figs. 1 and 2 in [77] show the delta-band EEG coherence have increased coherence between the left- and right-temporal regions).

Our clinical observation of reduced EEG coherence is consonant with cortical model predictions. However, our clinical EEG study indicates that the coherence alternation is a regional concept while the cortical model describes a uniform trend. Moreover, we did not find any theoretical prediction for the left- and right-temporal increased-coherence patterns seen in Fig. 6.11. As the cortical model by Steyn-Ross *et al.* is spatially homogenous, i.e., there are no explicit front-to-back or right-to-left directions, it is unable to produce different coherence changes for different regions. It appears that the Steyn-Ross cortical model best represents the cortical dynamics in the frontal region.

To allow the cortical model include two contrast coherence changes in one pattern, it may be suggested to induce a bistable mode with separable instabilities. For example, we tuned the parameters of the well-known chemical reaction–diffusion system Brusselator [24], so that it exhibits independent Turing and wave instabilities, as shown in Fig. 6.13. Unlike the cortical model experiencing a single instability (frozen Turing, homogeneous Hopf or chaotic Turing–Hopf modes), the Brusselator model manifests stable fronts between Turing domains and trains of plane waves existing in the bistable domain. The phase-coherence descriptions show that the Turing standing waves have high coherence while the travelling waves have low coherence. Examining panel (a) in Fig. 6.13, we see a Turing mode embedded in the travelling waves background, leading to low coherence in the top and bottom sides while high coherence in the centre range. Conversely, panel (b) shows a travelling wave mode embedded in the Turing background, leading to low

coherence in the centre range while high coherence in the top and bottom sides. A transition from panels (a) to (b) is very similar to our clinical observation of reduced and increased EEG coherence with induced propofol anaesthetic (Fig. 6.11): emergent Turing structures at both sides are related to the synchronised neuronal activities in the left- and right-temporal regions for information processing during the slow-wave sleep; the occurrence of the low coherent oscillations at the centre is analogous to the chaotic waves in the fronto-occipital areas.

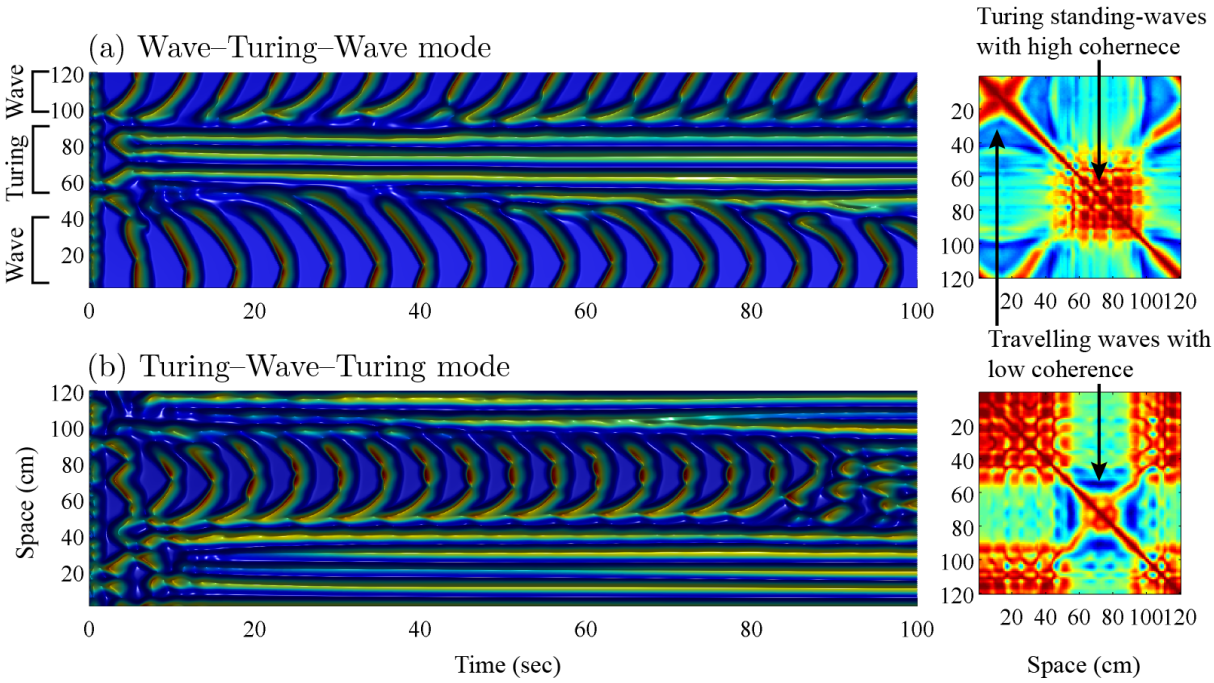


Figure 6.13: Bistable modes of the Brusselator model for (a) wave–Turing–wave and (b) Turing–wave–Turing. The first column is the space-time map of the 1D Brusselator (120 grid-points in a totaly of 120 cm) simulation. To the right is the phase-coherence representation of the 120 time-series in the space-time map.

Besides, Fig. 6.4 shows that the gap-junction cluster synchronises the neuronal activities positively in high density ($D_2 \gtrsim 0.8$ cm^2) but negatively in lower density ($D_2 \lesssim 0.7$ cm^2). Since our clinical coherence study suggests higher coherence in the temporal lobes than other cortical areas, we posit that the temporal lobes may possess a larger density of gap-junctions than other cortical areas during unconsciousness. Although there is little direct evidence to support our hypothesis (the measure of gap-junction density in the living cortex may be experimentally difficult), it is widely acknowledged that gap-junctions are vitally important to shape neuronal population rhythms [6, 25, 52, 118, 120, 132, 134].

In future work, we will further investigate the pattern dynamics of the cortical model for its possible bistabilities; meanwhile we may refine the cortical model into separated cortical regions to investigate their distinct responses to gap-junction modulation and how these sub-systems contribute to whole-of-cortex dynamics.

Chapter 7

Summary and future work

The Waikato cortical model describes the mean activities of densely interconnected neuronal aggregates, in which neuron groups communicate via chemical (neurotransmitter controlled) and electrical (gap-junction) synapses simultaneously. Although the cortical model is essentially a mathematical description comprising a series of differential equations, the neurophysiological significance of the model allows simulations of brain dynamics such as anaesthesia [113–115], sleep cycles [112], K-complexes [145], and Turing pattern formation [118]. These simulated spatiotemporal patterns describe, in principle, the emergent behaviours of the Waikato cortical model. What is the theoretical mechanism of these emergent patterns? We investigated the answer in this thesis via bifurcation theories.

7.1 Summary

Considering the high dimensionality and strong nonlinearity of the cortical model, we first examined the reliability and applicability of bifurcation theories in a simple pattern-forming system Brusselator for pattern predictions and analysis.

The reason we chose to analyse the Brusselator was explained in Sec. 5.3.3:

The cortical model is expected to exhibit similar dynamics to a chemical reaction–diffusion system: The interaction between the excitatory and inhibitory neurons is analogous to the competition between the activator and inhibitor of the chemical reaction–diffusion model, e.g. the Brusselator. The gap-junction strength between inhibitory neurons also plays the same role as the diffusion terms in the Brusselator, which allows a spatial evolution of the patterns. Consequently, it is reasonable that the cortical model shares the same pattern dynamics with the chemical reaction–diffusion system.

To analyse the pattern dynamics of the Brusselator model, we applied the linear stability analysis and nonlinear perturbation method:

Linear stability analysis of the Brusselator model

The linear stability analysis (LSA) is the most commonly used technique to examine the stability of a model's steady-state with respect to weak perturbations. It is conceptually clear and manipulatively simple. We applied LSA to the Brusselator model to derive its parameter settings to produce wave, Turing, Hopf and mixed Turing–Hopf (TH) instabilities. These bifurcation conditions were validated via simulations. However, we also found some LSA unpredicted phenomena: well-structured Turing patterns with lattices in particular arrangements and Turing-dominant TH mode. As we also experienced mismatches between the LSA and patterns from cortical models, we believe that a comprehensive understanding of the pattern dynamics of Brusselator may help us to better control and explain the cortical patterns. Noticed that the LSA is the first-order *linear* simplification of the original model, some important information may be hidden in the higher-order terms. To include these high-order terms, we introduced a *nonlinear* perturbation method — multiple-scale expansion.

Multiple-scale expaniosn of the Brusselator model

The multiple-scale expansion (MSE) splits the model variables into a perturbative power series in the vicinity of a bifurcation point. Each new variable carries arguments in multiple spatiotemporal scales. The idea of MSE is to separate the fast- and slow-evolutions of a patterned mode of interest, in which the slow-evolution corresponds to the pattern dynamics with the most unstable wave number, i.e., the critical wave number. The LSA shows that the mode with the critical wavenumber has the major contribution to the emergent pattern, so MSE highlights this mode and further gives rise to amplitude equations describing the mode envelope dynamics in a slow temporal scale close to a bifurcation point. Although MSE is conceptually simple, its manipulation is rather tedious and lacks of explanation in published materials. To facilitate an efficient MSE of the Brusselator model, we developed an open-source algorithm `Amp_solving` in MAPLE. `Amp_solving` follows a standard MSE scheme as well as allowing its automatic implementation. We supplied full `Amp_solving` tutorials to comprehensively demonstrate the manipulation of MSE and its automation. Although the tutorials interpret the application of `Amp_solving` only on the Brusselator model, it is a general purposed algorithm that can be utilised to any pattern-forming system, e.g., the cortical model.

`Amp_solving` simplifies the Brusselator model to the so-called amplitude equations, which provide supplementary pattern information to the LSA: Turing pattern structures and Turing–Hopf stabilities. Turing or Hopf bifurcation is the *linear instability* (first bifurcation) predicted by the linear stability analysis, they may become unstable (second bifurcation) due to the natural nonlinearity in the system. For example, the Turing instability is predicted via the positive real part of dominant eigenvalue of the Jacobian

matrix of the original system. Theoretically, such positive real part will cause the perturbation grow exponentially, so the Turing amplitude should have increased without bound. However, the simulation reveals that the Turing amplitude only grows *linearly* in a *fast* temporal scale, then saturates in a *slow* temporal scale since the nonlinearity becomes dominant. Consequently, it is necessary to consider the nonlinear effects of the system in order to predict its final state. To do this, we applied the stability analysis to the amplitude equations to investigate the stability conditions of the first bifurcation.

Analysis of the amplitude equations

The amplitude equations provide a universal expression relating to specific modes of interest, i.e., all pattern-forming systems have the same amplitude equations with respect to specific modes. Thus the stability analysis for one kind of amplitude equations can be shared between different systems. Here, we first analysed the amplitude equations for the Brusselator model. The same analysing methods were then applied to the cortical model.

In this thesis, we discussed the stability of the Turing and Turing–Hopf patterns via their respective amplitude equations. The Turing amplitude equations reveal three spatial structures: honeycomb like H_π, stripes and bullet like H_0 modes. Three modes follow a super-critical bifurcation: $H_\pi \rightarrow$ stripes $\rightarrow H_0$. The stripes can mix with either H_π or H_0 leading to a mixed Turing patterns.

The TH amplitude equations are more complicated than the Turing ones since they possess an additional temporal component, the Hopf mode. The inclusion of the Hopf mode allows a Turing–Hopf (TH) mode, pattern with a spatial structure oscillating in a homogeneous Hopf frequency. Such a TH mode may be unstable, eventually replaced by a stabler Turing or Hopf mode; or stable, possibly becoming a chaotic mode. We found that the suggested Hopf phase-stability conditions could not predict the chaotic mode. From discussions with Prof. Anne de Wit, an expert in pattern dynamics, we understood that these phase-stability conditions are restricted to the system of amplitude equations but not the original model. Thus, to find the chaotic mode of the original model, it is still suggested to manually scan the parametric space close to a codimension-2 point where the competitive Turing and Hopf are most likely to disturb the Hopf phase-stability, giving rise to chaotic turbulence.

After the pattern analysis of the Brusselator model, we applied developed theories to the cortical model for its pattern predictions and explanation.

Pattern analysis of the Waikato cortical model

The Waikato cortical model has three different versions (see Table 3.4): the SR2007 and SR2009s models belong to the slow-soma limit, and the SR2009f model belong to the fast-soma limit. The SR2009f model possesses a wave instability that is not the interest of this thesis. The LSA reveals that the coupling strength between gap-junctions

D_2 (i.e., inhibitory diffusion) and inhibitory rate-constant γ_i are two major bifurcation control parameters, responsible for Turing and Hopf instabilities respectively. A cortical Turing pattern occurs super-critically when the inhibitory diffusion D_2 is sufficiently large; seizure-like global oscillations are emergent sub-critically when the inhibitory rate-constant is sufficiently small. For both SR2007 and SR2009s models, a strong D_2 favours a Turing pattern of high- and low-neuronal activities; meanwhile suppressing seizure oscillations induced by the delays in the inhibitory postsynaptic response (i.e., reducing γ_i). Moderately strong D_2 and appropriate setting of γ_i introduce a cortical TH mode characteristic of a cortical Turing pattern oscillating homogeneously in ultra-slow envelope frequency, implying mechanisms of beating waves observed in BOLD fMRI recording of relaxed, non-tasked human brains; as well as the long-range neuronal synchronisation during the cognitive tasks.

The cortical model exhibits a series of cortical activations with biological significance, meanwhile it is a pattern-forming system following the bifurcation theories. As a result, the discipline of pattern dynamics in Brusselator may also be applicable to the cortical model. Aided by the LSA and Turing amplitude equations, we found the H_π and stripes modes in SR2007 model. Later, an investigation of the SR2009s model with the post-synaptic potential (PSP) described by a bi-exponential function unveiled the H_0 mode. These findings expand our previous experience on the cortical labyrinth (stripes) pattern in simulations. As in the Turing investigations, we applied bifurcation theories to the cortical TH mode. Similar to the Brusselator, we observed the bistable and chaotic TH modes of the cortical model. However, the complexity of the cortical model leads to a strong nonlinearity, sometimes the bifurcation theories may completely fail in pattern prediction.

Mixed mode of the Waikato cortical model

A mixed mode can be a combination of Hopf and Turing, Turing and Turing, Turing and wave, or Hopf and wave. Our bifurcation analysis shows that the Turing–Turing is the simplest mixed mode, the spatial structure of which can be predicted correctly from amplitude equations. The inclusion of the Hopf temporal instability dramatically increases the pattern uncertainty. In the cortical model, a sufficiently strong Turing instability is likely to activate a damped Hopf mode. This mechanism is very distinct to the conventional view that the Hopf is introduced by a first-quadrant complex eigenvalue. As a result, we discovered oscillatory H_0 patterns with a Turing instability and a damped Hopf instability. Similarly, by slightly increasing the Turing instability of a H_π mode, we activated its damped Hopf instability, leading to an oscillatory H_π mode. There are previous reported oscillatory hexagonal patterns generated by linearly coupling a temporal source (Hopf or wave instabilities) and a Turing mode, but we argue that such a mechanism may not be applicable to our cases since the temporal source is not clearly identified. We

are still looking for the theoretical support to our Turing–damped Hopf mode, and we doubt that the Laplacian terms in the long-range wave equation may be responsible for two reasons: it imbalances the symmetry of a reaction–diffusion system that has the same numbers of activatory and inhibitory diffusion terms; it acts on the excitatory membrane potentials but not the long-range flux, the derivative term of this equation.

Further, we discussed TH induced chaotic mode of the cortical model. Although we could not apply the Hopf phase-stability condition to predict a chaotic mode, it is widely acknowledged that an unstable Hopf phase leads a TH mode to chaotic turbulence. The cortical chaotic mode is very similar to the slow-wave oscillation observed during the deep sleep. So we postulate that these slow-waves may be formed spontaneously by the interacting spatial (Turing) and temporal (Hopf) neuronal activities. The balance between Turing and Hopf instabilities are modulated by the inhibitory diffusion D_2. A rather larger D_2 recovers the unconscious subject back to a wake state, implying two clinical predictions: moderately gap-junction coupling strength drives an awake cortex with balanced Turing and Hopf instabilities; fatigue agents (e.g., adenosine and propofol anaesthetic) will suppress the cortical activation further lowering the gap-junction coupling strength, and causing a Hopf-dominated chaotic slow-waves.

Propofol anaestheic induced EEG coherence drop

Another promising result from investigating the cortical TH mode was that a transit from wake to unconsciousness is accompanied by reduced EEG coherence in the sub-delta band. We analysed clinical EEG recordings from five subjects to examine the significant EEG coherence changes between wake and anaesthetic unconsciousness. To do this, we developed an automatic EEG processing algorithm, `EEG_coherence`, to separate the electrode-pairs with significantly reduced or increased coherence. We found that from wake to anaesthetic unconsciousness, sub-delta band EEG coherence was reduced mainly along the fronto-occipital axis, and increased along the left-right temporal axis. These EEG coherence alternations were also reported by other research teams [26, 47, 60, 77, 77, 78], but the cortical model predicts only a reduced coherence. Consequently, we posit that the Waikato cortical model may best represent the cortical dynamics in the frontal region.

7.2 Major achievements

To conclude, the work completed in this thesis lead to the following major achievements:

- We comprehensively explained the multiple-scale expansion, and developed a MAPLE algorithm `Amp_solving` allowing an automatic implementation of the multiple-scale expansion to derive amplitude equations.

- The analysis of the amplitude equations are well demonstrated. We showed how to examine the mode stability by performing the linear stability analysis of amplitude equations. Three Turing modes are determined, and the stability of the TH mode is examined.
- We investigated pattern dynamics for the Waikato cortical model at all steady-state branches. We located the cortical parametric space for Turing patterns in H_π, stripes and H_0 modes, for Turing–Hopf patterns in stable and chaotic TH modes. We also found interesting mixed modes such as oscillatory H_π and H_0 modes.
- We developed a MATLAB algorithm `EEG_cohernece` for automatically analysing EEG phase-coherence changes between two brain states. By applying `EEG_coherence` to the clinical EEG recordings we found particular cortical regions experiencing reduced or increased coherence. Our clinical EEG study supports the cortical model prediction that propofol anaesthesia will induce a EEG coherence drop at the frontal cortex.
- In addition, our introduction of the linear stability analysis and simulation strategies for the pattern-forming system may provide a basic tutorial for readers who wish to study pattern dynamics.

7.3 Future work

Prediction of the cortical Turing–Hopf mode

As discussed in the previous section, we still experienced difficulties when attempting to predict the cortical TH mode, especially the chaotic TH mode. Although the clue to locate chaotic mode is to scan the neighbourhoods of a codimension-2 point, we intend to find strong theoretical evidence to support the simulations. To do this, an analytical manipulation of the multiple-scale expansion is necessary. An analytical expression such as the amplitude equations has dramatic advantages over a numerical approach since we will be given more freedom to investigate all coefficients in the equations. The demonstrated amplitude equations are derived from the order-3 MSE. To predict pattern dynamics more precisely, we may consider to derive the higher order amplitude equations since de Wit indicated that high-order terms may contain corrections to the pattern dynamics. Besides, it is still worthwhile to develop new signal analysing algorithm to directly detect the early warning sign of the cortical chaotic evolutions or similar cortical emergent behaviours.

In addition, we consider an analytical amplitude equations may help us better predict the cortical bistable mode. The cortical model predicts an EEG coherence drop with induced propofol anaesthesia, but it lacks of information about other increased coherence trends at the temporal areas of the cortex. In Sec. 6.10, we utilised a Brusselator model to demonstrate a pattern transition with reduced coherence in the centre and enhanced

coherence at edges occurred simultaneously via a bistable mode. Considering the cortical model has a similar pattern-forming mechanism with the Brusselator model, we expect that the analytical amplitude equations may assist us to find the cortical bistable mode.

Development of the cortical model with mixed slow- and fast-soma limits

The slow-soma limit describes a resting state of the awake brain, i.e., a baseline level of activation which is suspended during goal-oriented tasks [99]; while the fast-soma limit characterises gamma oscillations associated with the cognitive function [106]. The major difference between two limits are the pattern oscillation frequency: the slow-soma limit introduces up to 4-Hz oscillations while the fast-soma limit can boost the oscillations up to 40-Hz. It is biological reasonable that a cortical model should switch between a slow- and fast-soma limits automatically or under certain circumstances. Currently the slow- and fast-soma limits are two different models, so how should we combine them together? A new model should possess two characteristic frequencies, a small one for the slow-soma and a large one for the fast-soma, and one frequency can become dominant by certain internal adjustments. From the view of pattern dynamics, such model will have a typical mixed mode with two frequencies. Following Yang *et al.* modelling strategies [152, 154], there may be a solution: couple two cortical models together, each of which has a critical frequency. Then set the coupling rate as a weighting factor to control the dominant mode.

Alternatively, we may need to find the appropriate parameter setting for the slow-soma (or fast-soma) limit to generate fast-soma (or slow-soma) patterns. A strategy is to use machine-learning based methods such as particle swarm optimisation [141] and differential evolution algorithm [140] to estimate the model parameters based on the practical model response. For example, we may construct a fitness function based on the slow-soma limit then feed the machine-learning algorithm with fast-soma data. So the algorithm will learn the features of the fast-soma limit meanwhile update the slow-soma model to fit the fast-soma data. However, one disadvantage for this method should be considered: the estimated model parameters may not be biological reasonable. So how to refine the searching range of the cortical parameters should be carefully investigated.

Appendix A

Source codes

This appendix includes selected MATLAB source codes that generate important results in the thesis.

A.1 Brusselator simulation

Runge-Kutta and Euler methods are commonly used to solve differential equations for pattern-forming systems, e.g., Brusselator and the cortical model. These pattern-forming systems are essentially mathematical differential equations. The evolved patterns we see in this thesis are time and space dependent solutions solved from the differential equations. The MATLAB built-in `ode45` solver provides a simple way to integrate the differential system in 1D space (e.g., van der Pol oscillator and Hodgkin-Huxley neuronal model), but there are few examples that demonstrate the application of `ode45` solver in 2D space. The major advantage of `ode45` is its automatic adjustments in time steps to maintain the accuracy of the solutions, and its faster calculations than the customised Euler algorithm. We illustrate the source codes for solving the Brusselator model numerically by both `ode45` solver and Euler algorithm in 2D space.

```
% Brusselator: reaction-diffusion model in 2D spatial grid
%              with periodic boundary conditions
% dX/dt = A - (B+1) * X + X^2 * Y + Dx * nabla^2 * X
% dY/dt = B*X - X^2 * Y + Dy * nabla^2 * Y
% in which B is the control parameter
%
% The differential system is solved by ode45 solver
% 12-Oct-2010: KWang

clear
clc; close all

% grid size
N= 60;

% dimensions for the full-resolution grid
[Nx Ny] = deal(N);         % no. of sampling points along each axis
[Lx Ly] = deal(60);         % square substrate (cm)
[dx dy] = deal(Lx/Nx, Ly/Ny);    % spatial resolution (cm)

L = [0 1 0; 1 -4 1; 0 1 0];  % Laplacian matrix
```

```
% define model parameters
a = 3;
Dx = 5/dx^2;  % diffusion multipliers (depend on spatial resolution)
Dy = 12/dx^2;
b = 9.1;

params = {N, Dx, Dy, a, b, L};

% initial conditions
% steady-states superimposed with one-off noise
xini = (a)*ones(N,N) + 0.001*randn(N,N);
yini = (b/a)*ones(N,N) + 0.001*randn(N,N);

% transform matrix to vector being ready to input ode45
ini_reshape = reshape([xini yini],1,2*N^2);

% time parameters
tstep= 0.1; % sec
tend = 50;

% run simulation
[t v] = ode45(@dxdt_wrap, 0:tstep:tend , ini_reshape, [], params);

% save output in a file
tv=[t v];

% split output to two variables
x = v(:,1:N*N);
y = v(:,N*N+1:end);
% structure of x and y
% matrix is arranged in vectors
%          1st grid-point   2nd grid-point   3rd grid-point
%  0
%  dt
%  dt*2
%  dt*3


% convert x and y to matrix form
for t_ind = 1: length(t)
     X(:,:, t_ind) = reshape(x(t_ind, :), N, N);
     Y(:,:, t_ind) = reshape(y(t_ind, :), N, N);
end


figure(1)
% time series of a given grid-point
plot(t, squeeze(X(30,30, :)));

figure(2)
% final pattern plot
pcolor(X(:,:,end))
set(gcf, 'InvertHardCopy', 'off')
shading interp
axis tight
colormap(jet)
```

The `ode45` solver calls `dxdt_wrap` function that contains following codes:

```
function dv = dxdt_wrap(t,v,params)

N = params{1,1}; Dx = params{1,2}; Dy = params{1,3}; a = params{1,4}; b = params{1,5};
L = params{1,6};

% input vector split into two variables
x = v(1:N*N);
y = v(N*N+1:end);

x = reshape(x,N,N);   % transform the vector into N*N matrix
y = reshape(y,N,N);   % transform the vector into N*N matrix

% differential equations
[dx, dy] = deal(zeros(N));

dim = 1: N;
dx(dim,dim) = a-(b+1)*x(dim,dim) + x(dim,dim).^2.*y(dim,dim);
dy(dim,dim) = b*x(dim,dim) - x(dim,dim).^2.*y(dim,dim);

conv1 = convolve2(x, L, 'wrap');  % periodic boundaries (toroid)
conv2 = convolve2(y, L, 'wrap');

dx = dx + Dx * conv1;
dy = dy + Dy * conv2;

% monitor spatial evolution over the time
imagesc(dx);
drawnow

dx = reshape(dx,N*N,1);
dy = reshape(dy,N*N,1);

dv = [dx; dy];
```

The Euler implementation of the Brusselator simulation is coded as following:

```
% Simulate 2D Brusselator model by Euler algorithm
%
% 21-June-2013: KWang

clear;clc;close all

a = 3;
b = 9.1;
Dx = 5;
Dy = 12;

time_end = 50;        % length of simulation, in sec

% dimensions for the full-resolution grid
[Nx Ny] = deal(60);            % no. of sampling points along each axis
[Lx Ly] = deal(60);            % square substrate (cm)
[dx dy] = deal(Lx/Nx, Ly/Ny);    % spatial resolution (cm)

% time resolution and time-base
dt = 1*1e-3;
Nsteps = round(time_end/dt)
```

```
% 3x3 Laplacian matrix (used in grid convolution calculations)
Laplacian = [0 1 0; 1 -4 1; 0 1 0];

% set up storage vectors and grids
[U_grid  V_grid] = deal(0.001*randn(Nx, Ny));
U0 = a; V0 = b/a;

% initialize the grids at steady-state values
[U_grid  V_grid] = deal(U0 + U_grid, V0 + V_grid);

% diffusion multipliers (depend on spatial resolution)
Dx = Dx/dx^2;
Dy = Dy/dx^2;

% Simulation
stride2 = 100;                % iterations per screen update
time = [0:Nsteps-1]'*dt;    % timebase
ii = 1;
for i = 1: Nsteps

    U_grid = U_grid + dt*(a-(b+1)*U_grid + U_grid.^2.*V_grid + ...
        Dx*convolve2(U_grid, Laplacian, 'wrap'));

    V_grid = V_grid + dt*(b*U_grid - U_grid.^2.*V_grid + ...
        Dy*convolve2(V_grid, Laplacian, 'wrap'));

    if (mod(i, stride2) == 1 || i == Nsteps)
        mesh(x, y, U_grid);
        drawnow;
        U_save(:,:, ii) = U_grid;
        ii = ii + 1;
    end

end
```

In Fig. A.1, we see that, given the identical initial condition (as well as the same `randn` seed), the `ode45` and Euler integration algorithms give rise to the nearly same result. The discrepancy between two algorithms experiences a jump between 20 and 30 s, when the pattern evolution encounters the second bifurcation, a saturation effect that suppresses the initial Turing instability. After this unstable period, the two algorithms converge, which is evident as the flat, zero-error curve.

Comparing `ode45` and Euler algorithms in the Brusselator simulation, we have the following comments:

- **Time efficiency**

  In this example, `ode45` uses less time to finish the simulation. Notice that the time-step settings in our codes: for `ode45` is 0.1 s, while for Euler is 10^{-3} s, 100 times smaller than `ode45`. Such distinct time-step settings result from two factors: `ode45` automatically adjusts the internal time-step to maintain the integration accuracy; in order words, `ode45` decreases the internal time-step only when necessary to achieve a balance between speed and accuracy. Euler is a first-order integration method, so

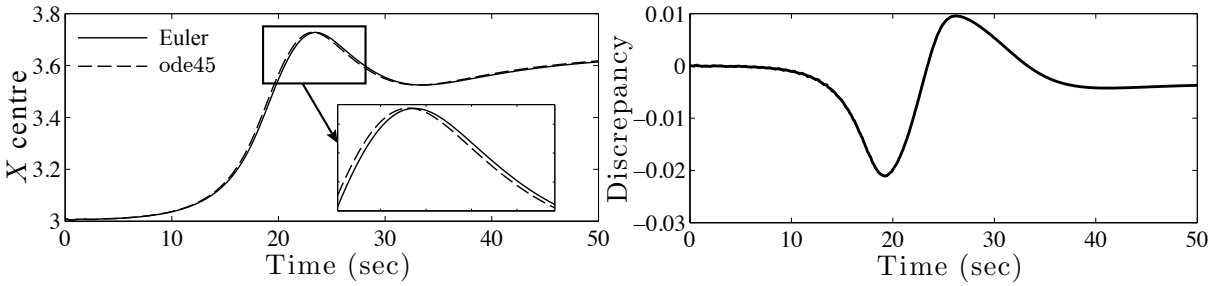


Figure A.1: Comparision of the Brusselator simulation using `ode45` and Euler codes presented in this section. Note that the time-step for `ode45` is 0.1 s while for Euler is 0.001 s. `ode45` and Euler spend respectively 6.3467 s and 22.1422 s to complete this 50-sec simulation. The left plot demonstrates the temporal evolutions of a centre grid-point in the Brusselator X grid using two integration methods, the discrepancy of which see the right plot.

we need to use a rather smaller time-step for the Euler method to maintain accuracy. For a given time-step setting, the Euler method runs generally faster than the `ode45` since Euler has fewer internal calculations, and does not need the matrix packing and unpacking operations that `ode45` does.

- **Accuracy**

  Provided we choose a sufficiently small time-step, the Euler method can achieve the same accuracy as the `ode45` method. `ode45` automatically adjusts the internal time-step to preserve integration accuracy. In a practical case, one may need to carefully tune the Euler time-step for a balance between the time-efficiency and accuracy.

Appendix B

Derivation of the cortical model

This appendix includes detailed calculations for deriving the cortical model.

B.1 Total synaptic inputs

The total input flux Φ is the temporal convolution of the dendrite impulse response $H(t)$ with the synapse spike-rate M (all subscripts are ignored for simplicity):

$$\begin{aligned} \Phi &= H(t) \otimes M(t) \\ &= \int_0^t H(t-t')\, M(t') dt' \end{aligned} \tag{B.1}$$

Theorem B.1.1

$$\begin{aligned} & F(t) = \int_{u(t)}^{v(t)} f(t',t) dt' \\ \Rightarrow \quad & \frac{dF}{dt} = \int_u^v \frac{\partial}{\partial t} f(t',t) dt' + \frac{dv}{dt} f(t',t)\Big|_{t'=v} - \frac{du}{dt} f(t',t)\Big|_{t'=u} \end{aligned}$$

By substituting alpha-function Eq. (3.7) and applying Theorem B.1.1, Eq. (B.1) yields:

$$\begin{aligned} \frac{d\Phi}{dt} &= \int_0^t \frac{\partial}{\partial t} H(t-t') M(t') dt' \\ &= \int_0^t \left[\gamma^2 e^{-(t-t')\gamma} - \gamma H(t-t') \right] M(t') dt' \\ &= \gamma^2 \int_0^t e^{-(t-t')\gamma} M(t') - \gamma \int_0^t H(t-t') M(t') dt' \\ &= \gamma^2 \int_0^t e^{-(t-t')\gamma} M(t') dt' - \gamma \Phi \end{aligned} \tag{B.2}$$

Take the second derivative, we have:

$$\begin{aligned}
\frac{d^2\phi}{dt^2} &= \gamma^2\left[\int_0^t \frac{\partial}{\partial t}e^{-(t-t')\gamma}M(t')dt' + M(t)\right] - \gamma\frac{d\Phi}{dt} \\
&= \gamma^2\left[-\gamma\int_0^t e^{-(t-t')\gamma}M(t')dt' + M(t)\right] - \gamma\frac{d\Phi}{dt} \\
&= \gamma^2\left\{-\gamma\left[\left(\frac{d\Phi}{dt} + \gamma\Phi\right)/\gamma^2\right] + M(t)\right\} - \gamma\frac{d\Phi}{dt} \\
&= \gamma^2\left[-\frac{1}{\gamma}\left(\frac{d\Phi}{dt} + \gamma\Phi\right) + M(t)\right] - \gamma\frac{d\Phi}{dt} \\
&= -\gamma\frac{d\Phi}{dt} - \gamma^2\Phi + \gamma^2 M(t) - \gamma\frac{d\Phi}{dt} \\
&= -2\gamma\frac{d\Phi}{dt} - \gamma^2\Phi + \gamma^2 M(t)
\end{aligned} \tag{B.3}$$

Arranging the above equation, we have

$$\left(\frac{d^2}{dt^2} + 2\gamma\frac{d}{dt} + \gamma\right)\Phi = \gamma^2 M(t) \tag{B.4}$$

which is equivalent to:

$$\left(\frac{d}{dt} + \gamma\right)^2\Phi = \gamma^2 M(t) \tag{B.5}$$

Considering differentiation on both time t and space $\vec{r}$, we replace d/dt by $\partial/\partial t$. Thus the rate of the total synaptic inputs into the dendritic tree is:

$$\left(\frac{\partial}{\partial t} + \gamma\right)^2\Phi(\vec{r},t) = \gamma^2 M(\vec{r},t) \tag{B.6}$$

B.2 Soma modelling - chemical synaptic inputs

The soma potential is its resting potential superimposed with inputs from excitatory E and inhibitory I chemical synapses:

$$\begin{aligned}
V &= V^{\text{rest}} + L \otimes [E(t) + I(t)] \\
&= V^{\text{rest}} + \int_0^t L(t-t')[E(t') + I(t')]dt'
\end{aligned} \tag{B.7}$$

in which

$$L(t-t') = \frac{1}{\tau}e^{-(t-t')/\tau} \tag{B.8}$$

with its differential form being:

$$\frac{\partial}{\partial t}L(t-t') = -\frac{1}{\tau^2}e^{-(t-t')/\tau} = -\frac{1}{\tau}L(t-t') \tag{B.9}$$

Using theorem B.1.1, the differentiation of Eq. (B.7) reads:

$$\begin{aligned}\frac{dV(t)}{dt} &= \int_0^t \frac{1}{\tau} L(t-t') \Big[E(t') + I(t') \Big] dt' + L(t-t') \Big[E(t) + I(t) \Big] \\ &= -\frac{1}{\tau} \int_0^t L(t-t') \Big[E(t') + I(t') \Big] dt' + \frac{1}{\tau} \Big[E(t) + I(t) \Big]\end{aligned} \tag{B.10}$$

Substituting Eq. (B.7) into the above equation, we obtain:

$$\begin{aligned}\tau \frac{dV}{dt} &= -(V(t) - V^{\text{rest}}) + E(t) + I(t) \\ &= V^{\text{rest}} - V(t) + E(t) + I(t)\end{aligned} \tag{B.11}$$

Considering both time and space , we have

$$\begin{aligned}\tau \frac{\partial}{\partial t} V(\vec{r}, t) &= -(V(\vec{r}, t) - V^{\text{rest}}) + E(\vec{r}, t) + I(\vec{r}, t) \\ &= V^{\text{rest}} - V(\vec{r}, t) + E(\vec{r}, t) + I(\vec{r}, t)\end{aligned} \tag{B.12}$$

Appendix C

Comparision between slow- and fast-soma cortical models

Table C.1: Slow- and fast-soma limits of the Waikato cortical model

Step 1. Flux ϕ_{ab} generated by source Q_a obeys 2-D damped wave equation through axon from population-averaged neuron type a to the population-averaged neuron type b

$$[(\partial/\partial t + v\Lambda_{ab})^2 - v^2\nabla^2]\phi_{ab} = v^2\Lambda_{ab}^2 Q_a$$

▲ Q_a is a mapping from membrane voltage $V_{a,b}$ to population-averaged firing rates
a, b are shorthand for either the population-average neuron being excitatory (e) or inhibitory (i)

Step 2.1. Total input flux arrive at axon terminal

$$M_{ab} = \underbrace{N_{ab}^{\alpha}\,\phi_{ab}^{\alpha}}_{\text{long-range}} + \underbrace{N_{ab}^{\beta}\,\phi_{ab}^{\beta}}_{\text{local}} + \underbrace{\phi_{ab}^{\text{sc}}}_{\text{subcortical}}$$

Step 2.2. Dendrite postsynaptic flux response

Slow-soma limit:
slow soma voltage feedback from soma to dendrite

$$\Phi_{ab} = \int_0^t \underbrace{H_{ab}(t-t')}_{\text{dendrite response}}\ \underbrace{M_{ab}(t')}_{\text{input flux}}\,dt'$$

Fast-soma limit:
prompt feedback from soma to dendrite

$$\Phi_{ab} = \int_0^t H_{ab}(t-t')\ \underbrace{\psi_{ab}(t')}_{\text{reversal weight}}\ \underbrace{M_{ab}(t')}_{\text{input flux}}\,dt'$$

▲ The dendrite dynamics are determined by the alpha-function impulse response $H_{ab}(t) = \gamma_{ab}^2 t e^{-\gamma_{ab}t}$ or biexponential function $H_{ab}(t) = \alpha_{ab}\beta_{ab}/(\beta_{ab}-\alpha_{ab})(e^{-\alpha_{ab}t} - e^{-\beta_{ab}t})$

Step 3. Voltage input to the soma

$$V_b^{\text{int}} = \int_{-\infty}^t \underbrace{L_b(t-t')}_{\text{soma response}}\ [\ \underbrace{E_b(t') + I_b(t')}_{\text{excitatory and inhibitory voltage inputs to the soma}}\ ]dt'$$

$$E_b^{\text{slow}}(t) = \rho_e\psi_{eb}(t)\Phi_{eb}(t) \qquad E_b^{\text{fast}}(t) = \rho_e\Phi_{eb}(t), \quad (\rho_e > 0)$$
$$I_b^{\text{slow}}(t) = \rho_i\psi_{ib}(t)\Phi_{ib}(t) \qquad I_b^{\text{fast}}(t) = \rho_i\Phi_{ib}(t), \quad (\rho_i < 0)$$

▲ $\psi_{ab}(t) = [V_a^{\text{rev}} - V_b(t)]/[V_a^{\text{rev}} - V_b^{\text{rest}}]$ is a dimensionless weighting factor to capture the dendritic response sensitivity to reversal potential for either excitatory (AMPA) or inhibitory (GABA) receptors.

Time-dependent perturbation about V_b^{rest}
$$V_b(t) = V_b^{\text{rest}} + V_b^{\text{int}}$$

Including electrical synapses, the differential equations for the soma voltage are

Slow-soma limit

$$\tau_b\frac{\partial V_b}{\partial t} = V_b^{\text{rest}} - V_b + \underbrace{\rho_e\psi_{eb}\Phi_{eb} + \rho_i\psi_{ib}\Phi_{ib}}_{\text{chemical synapses}} + \underbrace{D_{1,2}\nabla^2 V_b}_{\text{electrical synapses}}$$

Fast-soma limit

$$\tau_b\frac{\partial V_b}{\partial t} = V_b^{\text{rest}} - V_b + \underbrace{\rho_e\,\Phi_{eb} + \rho_i\,\Phi_{ib}}_{\text{chemical synapses}} + \underbrace{D_{1,2}\nabla^2 V_b}_{\text{electrical synapses}}$$

▲ $D_{1,2}$ = e-to-e, i-to-i gap-junction diffusive coupling (cm^2), $\rho_{e,i}$ = chemical synaptic strength (mV·s)

Appendix D

Simulink reconstruction of the Waikato cortical model

This chapter is extracted from our report to the New Zealand–Japan Exchange Programme (NZJEP[1]). The report describes my visiting scholar research in the Embedded Systems Lab at Gunma University in Japan for the MATLAB/SIMULINK construction of the Waikato cortical model from February to May 2013.

D.1 Background

The current Waikato cortical model simulation is executed via the standard MATLAB programming code script, which does not have a user-friendly interface and requires high-level programming skills to tune the model parameters. We are attempting to develop a graphic-based, user-friendly interface for the Waikato cortical model, thus providing a better access for users who are not familiar with programming. We are hoping our cortical model may offer a theoretical guidance for the clinical monitoring of brain states.

We found that SIMULINK might be an appropriate environment to reconstruct the Waikato cortical model. SIMULINK, an add-on product to MATLAB, provides an interactive, graphical environment for modelling, simulating, and analysing of dynamic systems. It enables modelling via a graphical user interface (GUI) for building models as block diagrams.

Embedded Systems Lab led by Associate Professor Yoichi Shiraishi at Gunma University (Japan) has a long history working with MATLAB/Simulink. The research at Embedded Systems Lab has a strong focus on transferring medical engineering theory into practical application. For example, they established a signal-processing algorithm for human pulse recorded from a fingertip. They implemented the algorithm by fabricating a mobile pulse-wave detector to monitor patient physical condition such as circulatory disease, mental disease, etc. Recently, they challenged the traditional method for measuring blood pressure, arguing that it was not reliable. They have begun to re-model

[1] http://ilep.ac.nz/pld-opportunities/pld-opportunities-abroad/japanese-programmes/the-nzjep-programme

the human circulatory system in SIMULINK incorporating their ideas into an embedded system.

Some preliminary ideas and research proposals of the collaboration between the Waikato Cortical Modelling Group and Embedded Systems Lab were discussed during my short visit to the Embedded Systems Lab following my presentation at the Conference on Computational Physics in Kobe, Japan in October of 2012.

The collaborative research for translating the cortical model from code sheet to SIMULINK blocks commenced in March 2013. After such translation, the cortical model is presented as graphical neuronal-group connections instead of MATLAB code, and simple drag-drop operation is supported to modulate neuronal properties; this allows a better control of the cortical behaviour and a clear demonstration of the rich diversity of brain states. The translated cortical model allows general access for medical researchers and neuroscientists who have an interest in cortical modelling.

As a PhD student working as a member of the Waikato Cortical Modelling Group, I was honoured to be the representative working at the Embedded Systems Lab to conduct our collaborative project. During my stay at Embedded Systems Lab, we completed a prototype of the SIMULINK-based cortical model that was able to show a range of brain dynamics. In a recent update, we have introduced spatial scales into the model, thus we can restrict the simulation running at a specific spatial scale, e.g., simulation about a piece of squared cortex with side length 6 cm, which brings more clinical significance.

This collaborative project has been presented at:

- Research seminar at Faculty of Information and Systems, 3rd April 2013, University of Tsukuba, Japan, hosted by Professor Hidetoshi Konno;
- Research seminar at Department of Production Science and Technology, 25th April 2013, Gunma University, Japan, hosted by Associate Professor Yoichi Shiraishi;
- Postgraduate seminar at School of Engineering, 11th July 2013, the University of Waikato, New Zealand.

D.2 Modelling differential equations in Simulink

The Waikato cortical model, as shown in Chapter 3, consists of a series of partial differential equations in first- and second-order. Solving these equations requires interpreting the differential operators for time ($\partial/\partial t, \partial^2/\partial t^2$) and space ($\nabla^2$). As a first example, we model the van der Pol oscillator in SIMULINK to explain how we interpret the differential operators. In the second example, the famous Brusselator reaction–diffusion system is modelled in SIMULINK. The interpretation of the Laplacian operator ∇^2 is addressed here. To our knowledge, it may be the first SIMULINK-based Brusselator model.

D.2.1 Van der Pol oscillator

The van der Pol oscillator was originally developed by the Dutch electrical engineer and physicist Balthasar van der Pol [10]. The van der Pol oscillator was the first mathematical model proposed for the heartbeat, and it has also been used to simulate brain waves [57, 149]:

$$\frac{d^2x}{dt^2} - \mu(1-x^2)\frac{dx}{dt} + x = 0 \tag{D.1}$$

We wish to solve this equation for the case $\mu = 1$ with initial conditions $x(0) = 2$ and $dx/dt = 0$ at $t = 0$. The traditional way to solve a second-order differential equation is to convert to a pair of coupled first-order differential equations:

$$\begin{aligned} \dot{x} &= y \\ \dot{y} &= \mu(1-x^2)y - x \end{aligned} \tag{D.2}$$

We would now integrate these equations with time using the MATLAB numerical integrator `ode45`. This helps to form the link with the integration in SIMULINK .

We code the first-order van der Pol equations into a MATLAB function[2] as follows:

```
function dydt = vanderpoldemo(t,y,Mu)
%VANDERPOLDEMO
%Defines the van der Pol equation for ODEDEMO.

% Copyright 1984-2002 The MathWorks, Inc.
% Revision: 1.2   Date: 2002/06/17 13:20:38

dydt = [y(2); Mu*(1-y(1)^2)*y(2)-y(1)];
```

To solve Eq. (D.2), we specify the coefficient μ, the initial conditions and the time-span over which the integration is to proceed; then pass these values, along with the name of the van der Pol function, to the Runge-Kutta solver `ode45`:

```
tspan = [0, 20];
y0 = [2; 0];
Mu = 1;
ode = @(t,y) vanderpoldemo(t,y,Mu);
[t,y] = ode45(ode, tspan, y0);

% Plot of the solution
plot(t,y(:,1), t, y(:,2))
xlabel('t')
ylabel('solution y')
title('van der Pol Equation, mu = 1')
```

The calculated results are plotted in Fig. D.1.

Alternatively, we may use the SIMULINK construction of Eq. (D.1), as shown in Fig. D.2.

[2] `vanderpoldemo` is a MATLAB pre-coded function

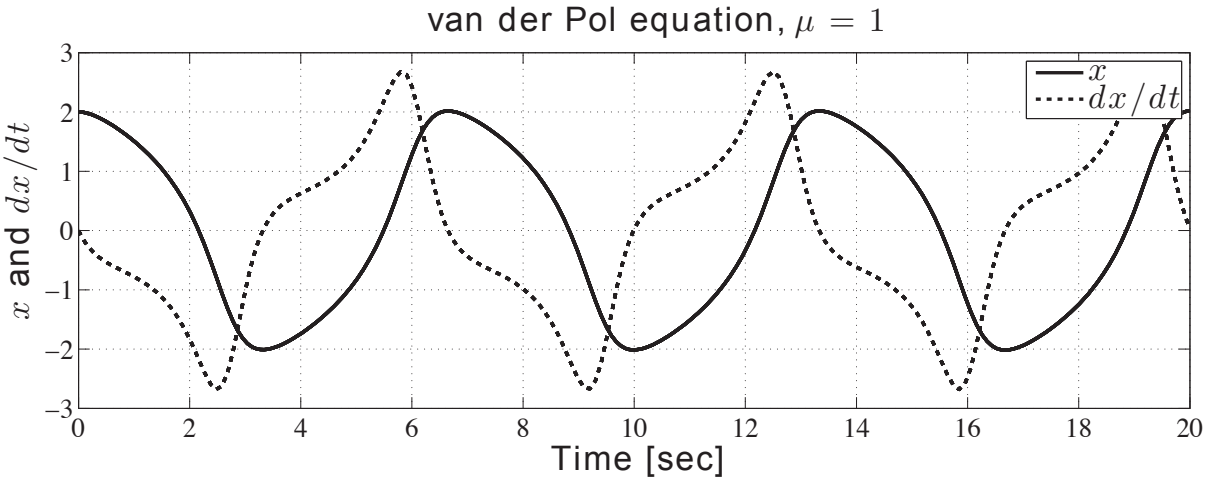


Figure D.1: Solution of the van der Pol equation, produced via MATLAB code sheet. Program running time: 0.384 s in variable time-step. Simulation platform (same for all simulations in this paper): MATLAB R2013a, MAC OS X 10.9.1, XCODE 5.0.2; CPU 2.4 GHz Intel Core i7, memory 8 GB 1600 MHz DDR3.

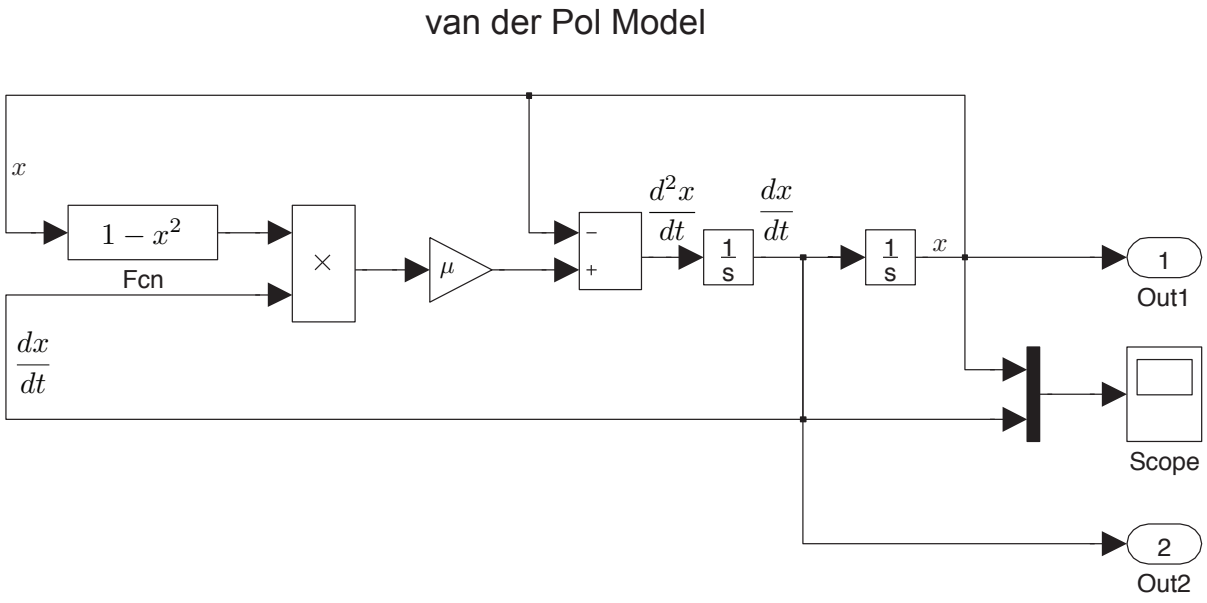


Figure D.2: SIMULINK built-in example for the van der Pol model called by the MATLAB command `vdp`.

At a first glance, the interface for SIMULINK is completely different from the code sheet. In SIMULINK, all calculating elements are displayed by blocks. We select blocks from the SIMULINK library, then connect them to build a model.

The basic principle to model a differential equation in SIMULINK is to find the input and output of an integrator. Since we have:

$$\int \left[\int \frac{d^2x}{dt^2} dt \right] dt = \int \frac{dx}{dt} dt = x \tag{D.3}$$

then it follows that for a second-order differential equation, we need at least two integrators. As seen in Fig. D.2, we first place an integrator block (the left block labelled with

$\frac{1}{s}$) to process the inner integration of Eq. (D.3): $\int \frac{d^2x}{dt^2}dt$. The output of this integrator reads dx/dt, which is sent into the second integrator (the right block labelled with $\frac{1}{s}$). We assume the integrated x is known, thus being used to construct the input of the left integrator block, which is equivalent to $\frac{d^2x}{dt^2}$ with the form:

$$\frac{d^2x}{dt^2} = \mu(1 - x^2)\frac{dx}{dt} - x \tag{D.4}$$

The product block (labelled with $\times$) in Fig. D.2 combines $(1 - x^2)$ and dx/dt. The result is amplified by a gain (triangle block, valued μ), then passed through a function block where x is subtracted. Here, the RHS of Eq. (D.4) is constructed.

Modelling a differential equation in SIMULINK requires forming a closed loop, where the integrated variables are fed back into the system. Evolution proceeds until reaching the desired final time. The `scope` block shows the real-time output of the two integrators; the `scope` can be placed anywhere to monitor the response of a sub-system. The `Out1` and `Out2` terminals send outputs of two integrators to the MATLAB workspace for further analysis. The results of this SIMULINK model are exactly the same as shown in Fig. D.1.

Both MATLAB and SIMULINK allow fixed or self-adaptive (i.e., auto) time-steps for the Runge-Kutta solver[3]. Fig. D.3 shows that the discrepancy between MATLAB and SIMULINK Runge-Kutta solvers in either fixed or auto time-step mode are sufficiently small ($< 10^{-10}$). Consequently, we can see that the accuracy of the model simulation does not depend on the modelling platform since MATLAB and SIMULINK share the same integration algorithm to solve differential equations. However, modelling in SIMULINK is more straightforward and intuitive, and requires less programming skill than the MATLAB code sheet. The original mathematical equations can be converted into SIMULINK by matching its pattern with SIMULINK blocks directly. Moreover, in SIMULINK, by simply adding more blocks, or replacing blocks, a new model is able to be built in a very short time. SIMULINK may be an ideal tool to efficiently perform the simulations of a mathematical model. In the next section, we will extend the SIMULINK modelling method to describe a Brusselator system considering both its temporal and spatial evolutions on 1-D and 2-D Cartesian grids.

Readers should be aware of the choice of an appropriate differential solver for a specific problem, depending on the stiffness of differential equations. Applying a wrong solver may lead to either unstable solution or exceptional computation time. However, it is practically difficult to identify the stiffness of a differential model, thus one should try at least two different solvers, and compare the results. If they concur, i.e. give the same solution, they are likely to be correct. As suggested by MATLAB help file, it is worthwhile to try `ode45`

[3]fixed time-step ODE solvers are not built into MATLAB, but they can be acquired from a release by the MathWorks Support Team:
`http://www.mathworks.com/matlabcentral/answers/uploaded_files/5693/ODE_Solvers.zip`

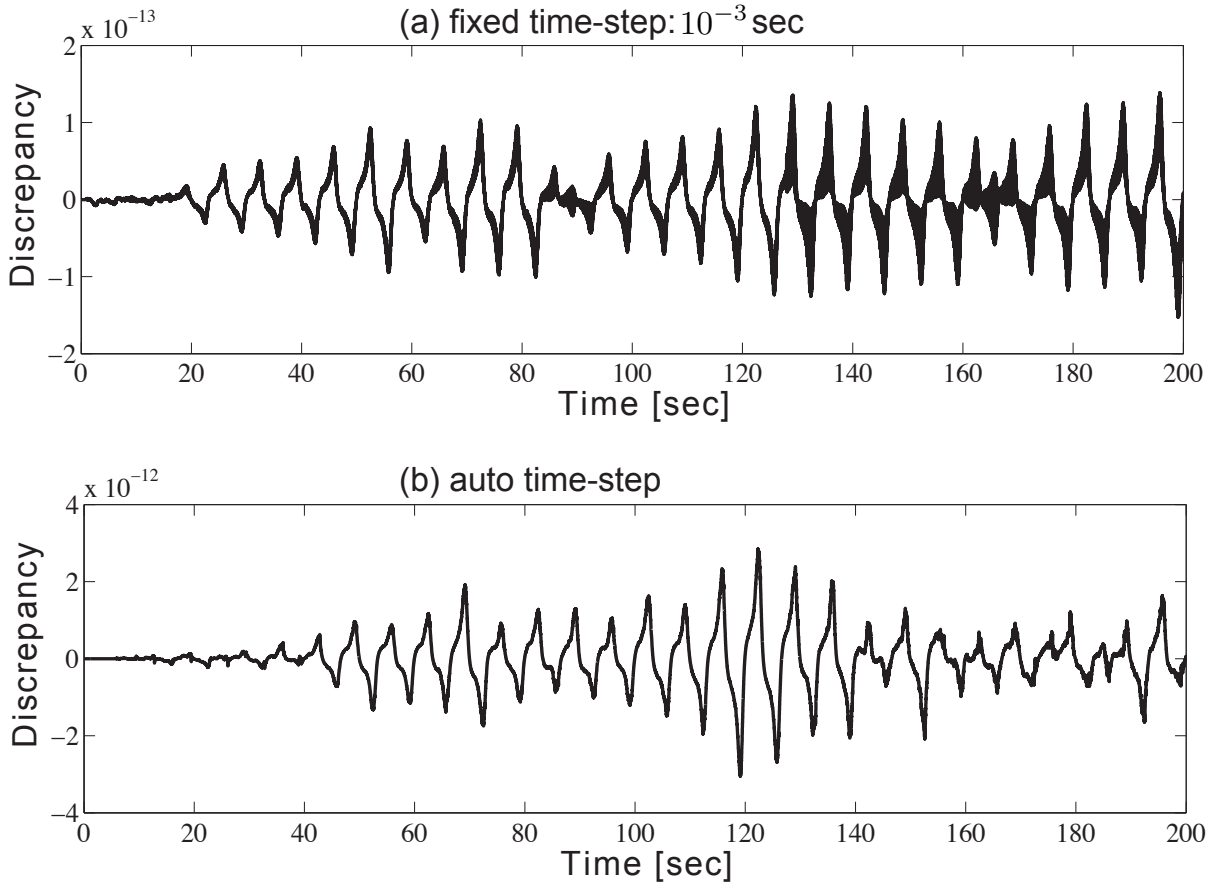


Figure D.3: Discrepancy over time for the solution of Eq. (D.1) is calculated from two modelling methods: Simulink and Matlab code sheet. Both methods use **(a)** fixed time-step 10^{-3} s; or **(b)** auto time-step. Program running time for Simulink in fixed and auto time-steps are respective 1.3410 and 0.7404 s; for Matlab the corresponding time are 17.3952 and 9.4569 s.

first since it is the most widely used method. For pattern-forming systems, we can also compare the numerical solution with the theoretical prediction to identify the applicability of the solver. For the demonstrated Brusselator and cortical models, `ode45` and `ode23` both work well and give rise to similar result; moreover, the numerical solutions match well with the theoretical predictions in emergent patterns (see [138] for full simulation results). So we choose `ode45` solver to integrate the differential-equation models in this chapter.

D.3 Brusselator model

The Brusselator model describes the competition of two chemical species in a chemical reaction, and is the simplest reaction-diffusion system capable of generating complex spatial patterns. The competition between two reactors and the introduction of diffusion satisfy the key requirements for pattern formation [127]. The pattern dynamics of the

Brusselator has been comprehensively examined in Chapters 2 and 5. Here, our purpose is to introduce SIMULINK modelling strategies.

The simplest form of the model reads [24],

$$\begin{aligned}\frac{\partial}{\partial t}X &= A - (B+1)X + X^2Y + D_X\nabla^2 X \\ \frac{\partial}{\partial t}Y &= BX - X^2Y + D_Y\nabla^2 Y\end{aligned} \tag{D.5}$$

where X and Y denote concentrations of activator and inhibitor respectively; D_X and D_Y are diffusion constants; A is a constant and B is a parameter that can be varied to result in a range of different patterns.

The LHS of Eq. (D.5) is a partial derivative on time since X and Y are functions of both time and space. At the RHS, the spatial derivative is represented by a Laplacian operator ∇^2. In the numerical simulation, the spatial dimension of the model is discretised into a grid by using the finite difference method. In the two-dimensional system the Laplacian with respect to the concentration field U in the node (i, j) is calculated along the x and y directions simultaneously:

$$\nabla^2 U_{i,j} \approx \frac{\Delta_x^2 U_{i,j}}{h_x^2} + \frac{\Delta_y^2 U_{i,j}}{h_y^2} \tag{D.6}$$

where

$$\Delta_x^2 U_{i,j} = U_{i+1,j} - 2U_{i,j} + U_{i-1,j}; \quad \Delta_y^2 U_{i,j} = U_{i,j+1} - 2U_{i,j} + U_{i,j-1} \tag{D.7}$$

The $h_{x,y}$ demonstrators in Eq. (D.6) are the respective x and y grid spacings; they define the spatial resolution. Assuming $h \equiv h_x = h_y$ (i.e., a square grid), the discrete Laplacian operation in a one-dimensional Cartesian coordinates along the y-axis has the form:

$$\nabla_{1\mathrm{D}}^2 U_{i,j} \approx \frac{U_{i,j+1} - 2U_{i,j} + U_{i,j-1}}{h^2}; \tag{D.8}$$

for the two-dimensional case, we have

$$\nabla_{2\mathrm{D}}^2 U_{i,j} \approx \frac{U_{i+1,j} + U_{i-1,j} - 4U_{i,j} + U_{i,j+1} + U_{i,j-1}}{h^2} \tag{D.9}$$

In SIMULINK, we initialise the Brusselator model as a column vector consisting of a 60×1 grid (spatial resolution = 1 cm/grid-point) for the one-dimensional case; or as a 60×60 grid for the two-dimensional case. Grid edges are joined to give toroidal boundaries.

The Laplacian operator $\nabla_{1\mathrm{D}}^2$ in Eq. (D.8) is implemented as a circular convolution of the 3×1 second-difference kernel $L_{1\mathrm{D}}^y$ acting along the y-axis:

$$\nabla_{1\mathrm{D}}^2 \approx L_{1\mathrm{D}}^y = \frac{1}{h_y^2}\begin{bmatrix} 1 \\ -2 \\ 1 \end{bmatrix} \tag{D.10}$$

The two-dimensional Laplacian operator $\nabla^2_{\rm 2D}$ in Eq. (D.9) is built up from the sum of two orthogonal $L_{\rm 1D}$ operators:

$$\nabla^2_{\rm 2D} \approx L_{\rm 2D} \;=\; \frac{1}{h_x^2}\begin{bmatrix}0 & 1 & 0\\ 0 & -2 & 0\\ 0 & 1 & 0\end{bmatrix} + \frac{1}{h_y^2}\begin{bmatrix}0 & 0 & 0\\ 1 & -2 & 1\\ 0 & 0 & 0\end{bmatrix} \;=\; \frac{1}{h^2}\begin{bmatrix}0 & 1 & 0\\ 1 & -4 & 1\\ 0 & 1 & 0\end{bmatrix} \tag{D.11}$$

where we have again assumed a square grid so that $h_x = h_y = h$.

In Simulink, the 1-D or 2-D Laplacian operator with toroidal boundaries is processed through two blocks: The "`wrap-around`" and "`2-D CONV`" (can process both 1-D and 2-D convolutions) . The "`wrap-around`" block wraps the input matrix on both axes to allow a valid convolution in the "`2-D CONV`" block against the Laplacian kernel L to return the cyclic convolution. We created a subsystem to compute the convolution, as shown in Fig. D.4.

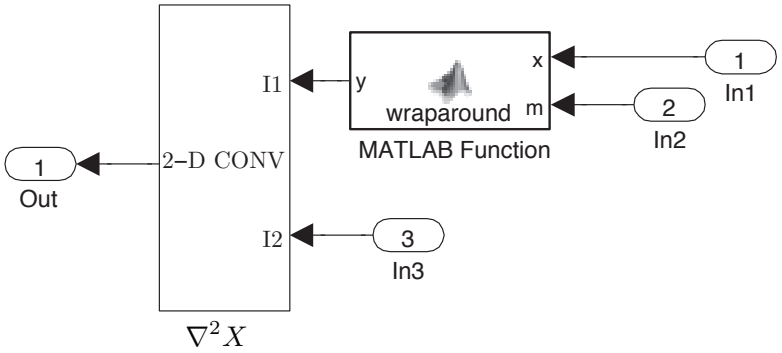


Figure D.4: Simulink modelling of the convolution with toroidal boundaries. The spatial derivative $\nabla^2 X$ approximates to the discrete convolution of X given by the kernal L. X will be fed into the port `In1`, the kernel L enters `In2` and `In3`.

In Fig. D.4, the custom block labelled "wraparound" contains codes extracted from the `convolve2()` function[4].

```
function y = wraparound(x, m)
% Extend x so as to wrap around on both axes, sufficient to allow a
% "valid" convolution with m to return the cyclical convolution.
% We assume mask origin near centre of mask for compatibility with
% "same" option.
[mx, nx] = size(x);
[mm, nm] = size(m);
if mm > mx | nm > nx
    error('Mask does not fit inside array')
end

mo = floor((1+mm)/2); no = floor((1+nm)/2);  % reflected mask origin
ml = mo-1;            nl = no-1;             % mask left/above origin
mr = mm-mo;           nr = nm-no;            % mask right/below origin
```

[4]The two-dimensional circular convolution algorithm was written by David Young, Department of Informatics, University of Sussex, UK. His convolve2() code can be downloaded from MathWorks File Exchange, `http://www.mathworks.com/matlabcentral/fileexchange/22619-fast-2-d-convolution`

```
me = mx-ml+1;          ne = nx-nl+1;          % reflected margin in input
mt = mx+ml;            nt = nx+nl;            % top of image in output
my = mx+mm-1;          ny = nx+nm-1;          % output size

y = zeros(my, ny);
y(mo:mt, no:nt) = x;      % central region
if ml > 0
    y(1:ml, no:nt) = x(me:mx, :);                    % top side
    if nl > 0
        y(1:ml, 1:nl) = x(me:mx, ne:nx);             % top left corner
    end
    if nr > 0
        y(1:ml, nt+1:ny) = x(me:mx, 1:nr);           % top right corner
    end
end
if mr > 0
    y(mt+1:my, no:nt) = x(1:mr, :);                  % bottom side
    if nl > 0
        y(mt+1:my, 1:nl) = x(1:mr, ne:nx);           % bottom left corner
    end
    if nr > 0
        y(mt+1:my, nt+1:ny) = x(1:mr, 1:nr);         % bottom right corner
    end
end
if nl > 0
    y(mo:mt, 1:nl) = x(:, ne:nx);                    % left side
end
if nr > 0
    y(mo:mt, nt+1:ny) = x(:, 1:nr);                  % right side
end
```

The reason we introduce the custom block is that the SIMULINK built-in `2-D CONV` block provides only the "valid" (non-flux) boundary condition, and cannot handle periodic boundaries.

Following the ideas of modelling the van der Pol oscillator, we can easily convert Eq. (D.5) to SIMULINK blocks, seen in Fig. D.5.

In summary, to construct the Brusselator model in SIMULINK, we first place an integrator block to represent time derivative. The temporal integrator's output will be fed back into the system to engage with the system's evolution, then form the input of this integrator block, closing the loop. The model parameters can be adjusted by tuning the settings of the blocks A and B as well as two gains (labelled D_X and D_Y respectively). The real-time spatiotemporal evolution of X and Y are monitored via the `Matrix viewer` block. The `simout` block delivers the solution of Eq. (D.5) to the MATLAB workspace for future analysis. The solution is a three-dimensional matrix with the third dimension the same length as the time span.

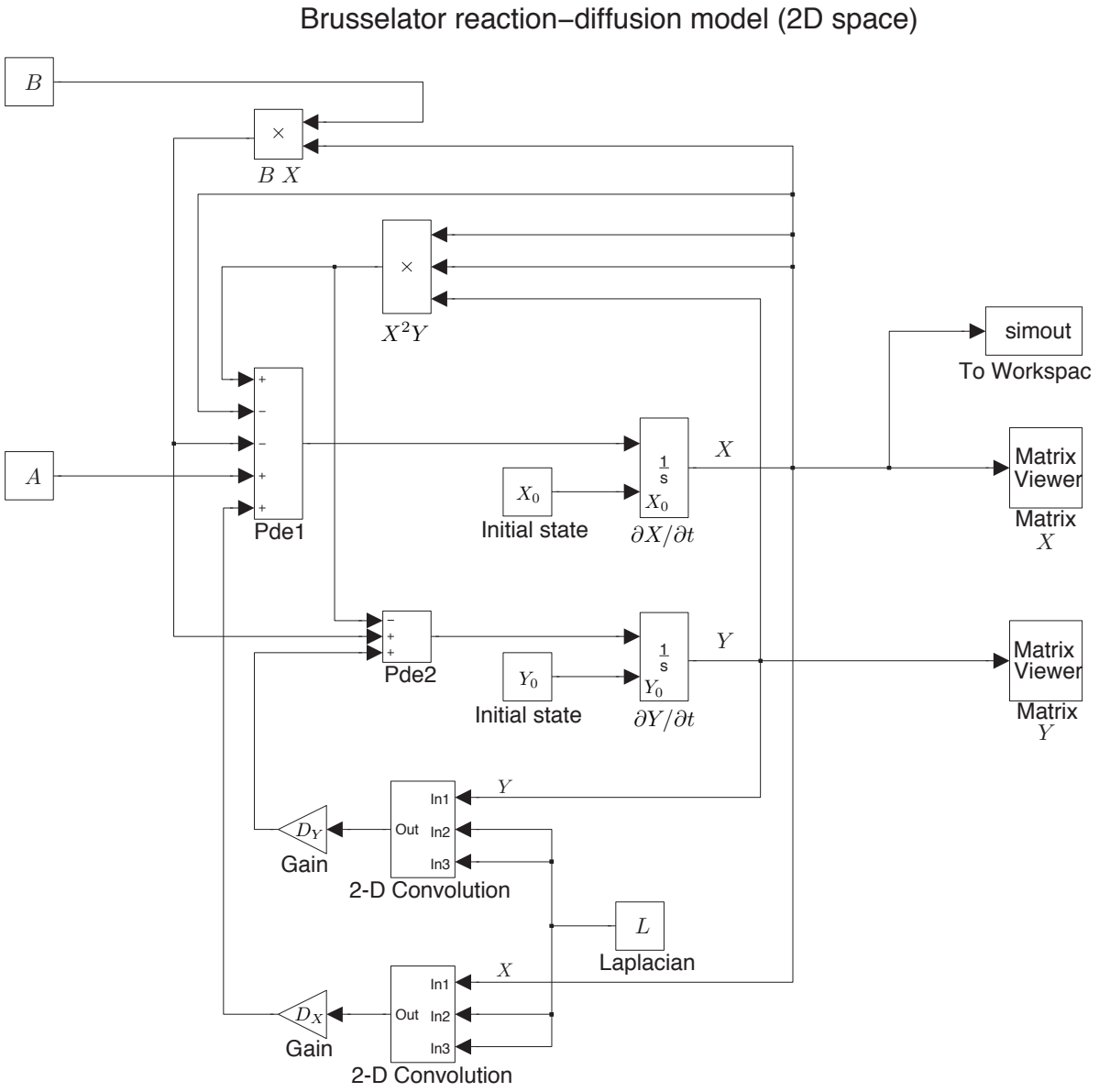


Figure D.5: Simulink construction of the Brusselator model.

D.4 Simulink versions of Waikato cortical model equations

Let us first list the mathematical equations for the Waikato cortical model and examine their characteristics.

- The cortico-cortical equation

$$\left[\left(\frac{\partial}{\partial t}+v\Lambda_{eb}\right)^2-v^2\nabla^2\right]\phi_{eb}^{\alpha}=(v\Lambda_{eb})^2Q_e$$

can be arranged by collecting temporal derivatives to the LHS:

$$\frac{\partial^2}{\partial t^2}\phi_{eb}^{\alpha}+2v\Lambda_{eb}\frac{\partial}{\partial t}\phi_{eb}^{\alpha}=v^2\nabla^2\phi_{eb}^{\alpha}-v^2\Lambda^2\phi_{eb}^{\alpha}+(v\Lambda_{eb})^2Q_e \tag{D.12}$$

- The intra-cortical equations

$$\left(\frac{\partial}{\partial t}+\gamma_e\right)^2\Phi_{eb}=[N_{eb}^{\alpha}\phi_{eb}^{\alpha}+N_{eb}^{\beta}Q_e+\phi_{eb}^{sc}]\gamma_e^2$$
$$\left(\frac{\partial}{\partial t}+\gamma_i\right)^2\Phi_{ib}=[N_{ib}^{\beta}Q_i]\gamma_i^2$$

have different RHS, but their LHS are in the same mathematical pattern:

$$\left(\frac{\partial}{\partial t}+\gamma\right)^2\Phi=\frac{\partial^2}{\partial t^2}\Phi+2\gamma\frac{\partial}{\partial t}\Phi+\gamma^2\Phi \tag{D.13}$$

We can move the term $\gamma^2\Phi$ to the RHS of the intra-cortical equations, then the LHS of the intra-cortical equations have the expression:

$$\frac{\partial^2}{\partial t^2}\Phi+2\gamma\frac{\partial}{\partial t}\Phi \tag{D.14}$$

which is similar to the LHS of the cortico-cortical equation.

- The soma equation

$$\tau_b\frac{\partial V_b}{\partial t}=V_b^{\text{rest}}-V_b+(\rho_e\psi_{eb}\Phi_{eb}+\rho_i\psi_{ib}\Phi_{ib})+D_{bb}\nabla^2V_b$$

can be re-arranged as

$$\frac{\partial V_b}{\partial t}=\frac{1}{\tau_b}\left[V_b^{\text{rest}}-V_b+(\rho_e\psi_{eb}\Phi_{eb}+\rho_i\psi_{ib}\Phi_{ib})+D_{bb}\nabla^2V_b\right] \tag{D.15}$$

Following the ideas of SIMULINK modelling in van der Pol oscillator, we need two integrator blocks for Eqs. (D.12) and (D.14), and two convolution processing for Eqs. (D.12) and (D.15).

The strategy for modelling a large system is to focus on its subsystems first, then connect them together. The Waikato cortical model has three major parts: cortico-cortical, intra-cortical and soma equations. Fig. D.6 shows how neuronal fluxes are transferred from one to another: cortico-cortical flux ϕ_{eb}^{α} is delivered to the long-range targets Φ_{ee} and Φ_{ei}; intra-cortical flux Φ_{ee} and Φ_{ei}, Φ_{ie} and Φ_{ii} merge into the soma equations. The output of the soma V_e is connected to source neurons to form the closed loop through the excitatory sigmoid function:

$$Q_e=\frac{Q_e^{\max}}{1+e^{-C(V_e-\theta_e)/\sigma_e}}$$

In following sections, we detail the SIMULINK implementation of the three subsystems (drawn as three blocks in Fig. D.6) of the cortical model.

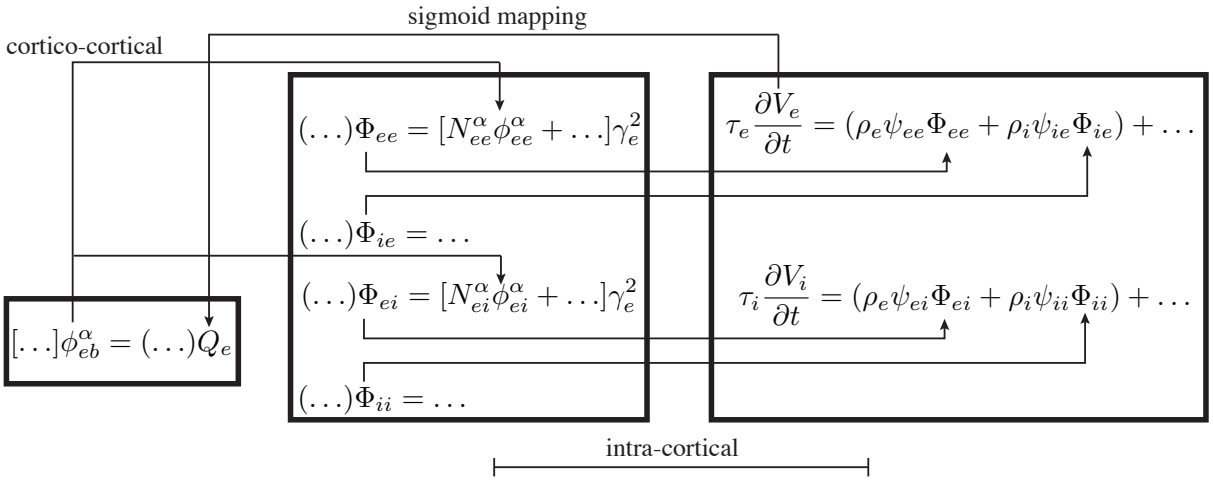


Figure D.6: Flux flows for the Waikato cortical model between its cortico-cortical, intra-cortical and soma equation subsystems.

D.5 Cortico-cortical flux

The Simulink based cortico-cortical block (see Fig. D.7) is converted from Eq. (D.12). The flux-source Q_e is a mapping from the excitatory soma voltage sent via the `Goto` block, to the firing-rate received via the `From` block. After two integrations, signals will be passed to the excitatory synapses via port-1 (upper right corner) and inhibitory synapses via port-2 in the intra-cortex.

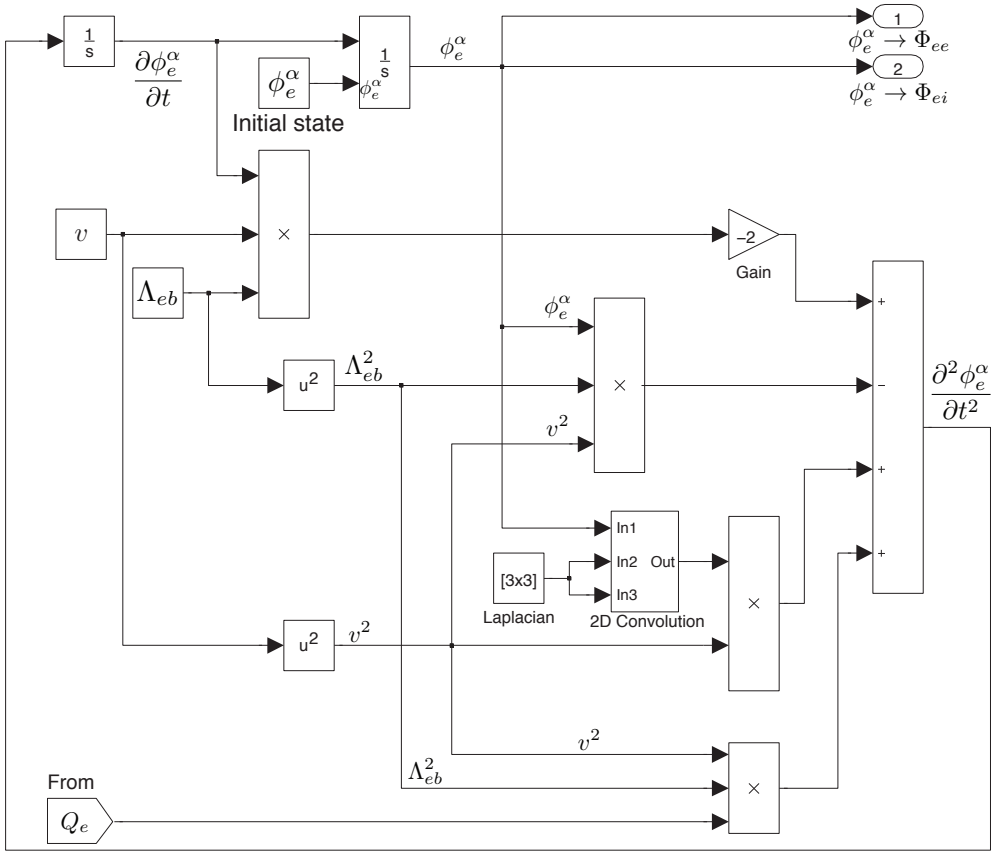


Figure D.7: Simulink-based cortico-cortical wave-equation

D.6 Intra-cortical flux

In SIMULINK modelling, we divide Eq. (3.4) into two parts: the constant (i.e. dc-level) excitatory background $s\langle\phi_{eb}^{\rm sc}\rangle$ and the one-off kick $\sqrt{s\langle\phi_{eb}^{\rm sc}\rangle}\xi_{eb}$. As illustrated in Fig. D.8, we use a `Clock` block to count the iteration step. Once the counter is above one, the "switch" will turn off the kick, allowing only the constant excitation to enter the intra-cortex (removing the `Clock` block would allow on-going noise stimulus from the subcortex). The 2-D spatial white noise are generated by the `Band-Limited White Noise` block.

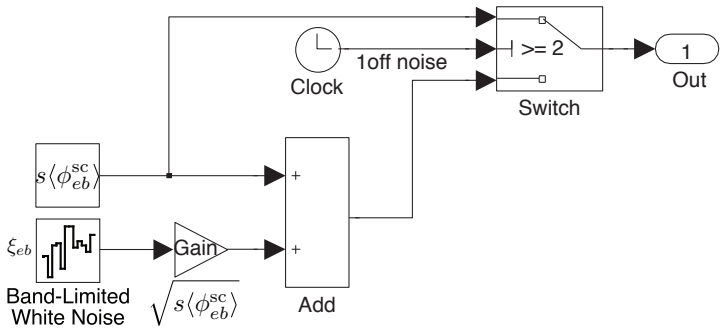


Figure D.8: SIMULINK-based subcortical flux

The intra-cortical model describes how post-synaptic fluxes evolve over time. In Fig. D.9, the local fluxes (input via the `From` block) along with the long-range fluxes (from input-1 at the left, labelled as ϕ_e^α) and subcortical drive are summed, then filtered at the post-synaptic dendrite, thus forming the post-synaptic fluxes Φ_{ee} at the output port-1 (upper right corner). The Φ_{ee} and Φ_{ei} flux models have symmetric structure.

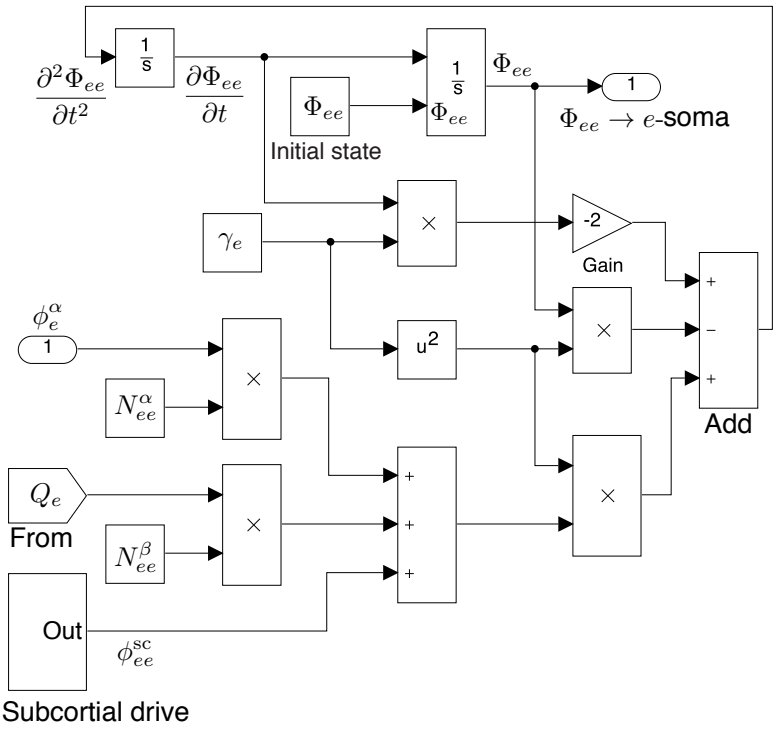


Figure D.9: SIMULINK-based $e \rightarrow e$ post-synaptic flux Φ_{ee} for the intra-cortex

We assume that the cortico-cortical fibres are exclusively excitatory, thus there are no long-range inhibitory fluxes entering into the soma. Fig. D.10 shows that the local inhibitory fluxes Φ_{ie} come from local source Q_i only. The Φ_{ie} and Φ_{ii} models also have symmetric structure.

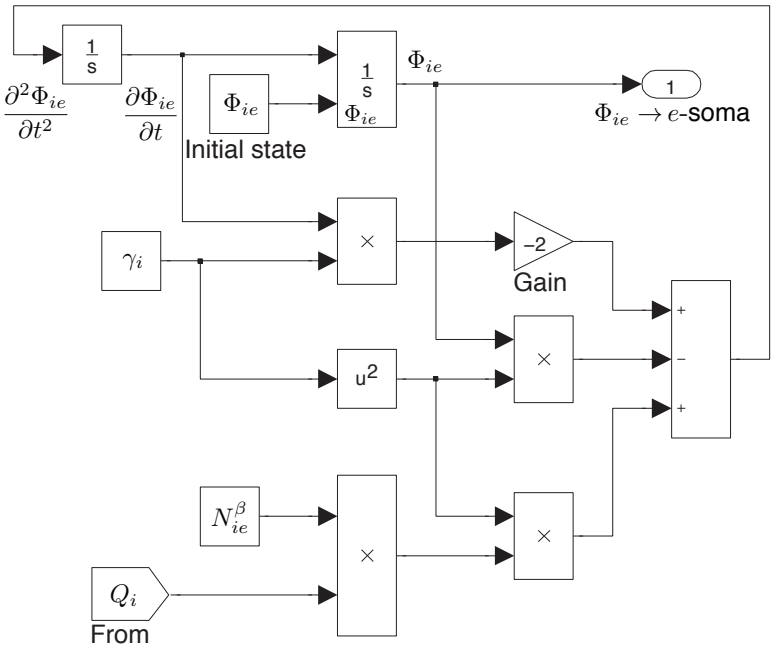


Figure D.10: Simulink-based $i \rightarrow e$ post-synaptic flux Φ_{ie} for the intra-cortex

D.7 Soma voltage

Fig. D.11 presents the soma model of the excitatory neuronal group. The short-range fluxes are accumulated at the soma, from ports 1 and 2. The soma voltage V_e is converted to firing-rate Q_e locally in this sub-model (block labelled with Q_e sigmoid), then fed back into the cortico-cortical and intra-cortical models.

Finally, we connect all subsystems to form the completed Waikato cortical model, as illustrated in Fig. D.12. It follows the flux flow-chart of Fig. D.6, with the detailed Simulink block connections shown in Fig. D.13. We argue that such model-based-design is an advantage for representing differential equations in Simulink . Although Simulink is useful for rapid prototyping, the Simulink implementation of the cortical model runs slower than our pre-coded Euler integration[5] since it is time-consuming to interpret the Matlab function `wraparound` (see Fig. D.4) in Simulink . A 60×60 grid (side length 20 cm) 5-s cortical simulation takes $\sim$10 s via Matlab Euler integration

[5]Matlab simulation codes were written by Alistair Steyn-Ross. The complete codes, plus README files and movies of cortical dynamics, are available from the web site:
`http://www2.phys.waikato.ac.nz/asr/`

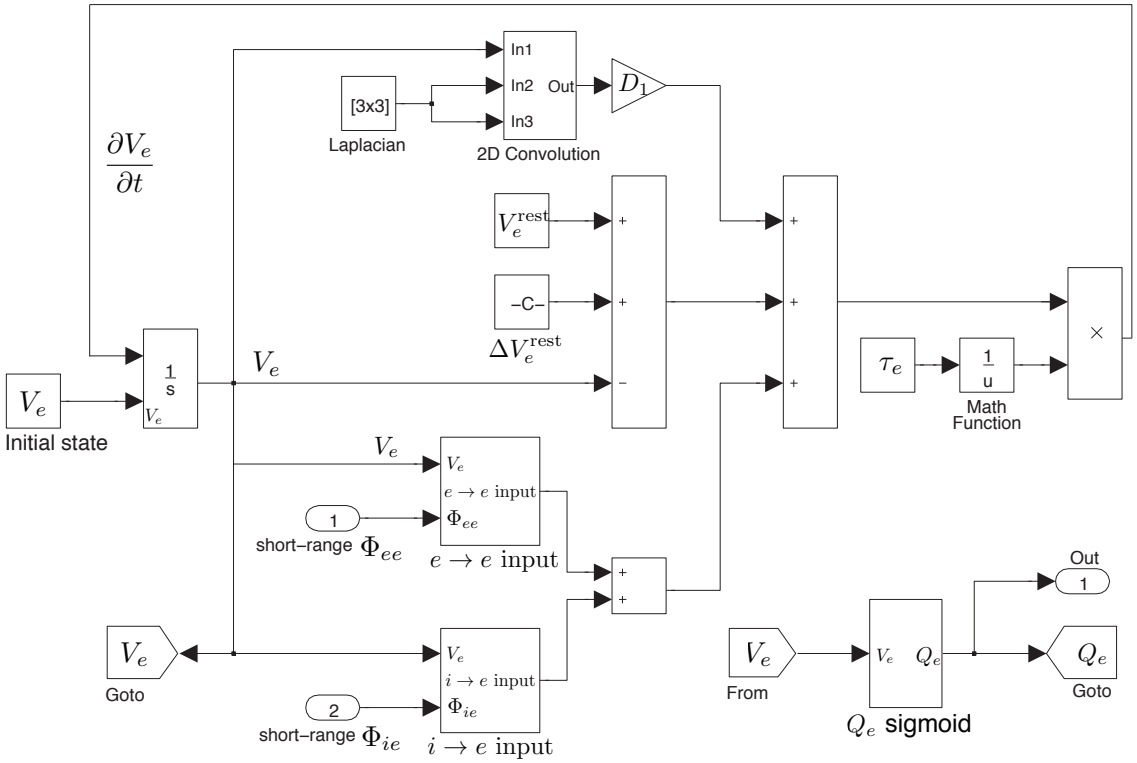


Figure D.11: SIMULINK-based excitatory soma equation

(fixed time-step 0.8×10^{-3} s), while ∼40 s via SIMULINK (auto time-step and in accelerator mode). Thus, it is recommended to avoid using MATLAB functions in SIMULINK unless necessary.

D.8 Comment on Simulink running efficiency

Although SIMULINK has an intuitive programming logic and comparable accuracy to MATLAB, it sometimes runs much slower than MATLAB, e.g., in the demonstrated Brusselator and cortical model simulations. The reason is that we embed MATLAB functions `wraparound` in the model to expand the SIMULINK capability. Once a MATLAB function block is present, the MATLAB interpreter is called at each time-step. This drastically reduces the simulation speed. So, one should use the built-in blocks whenever possible. Without using MATLAB function blocks, SIMULINK shows a higher performance than MATLAB, e.g., see the description of Fig. D.3. MathWorks Support Team also presented comprehensive guidance to speed up the SIMULINK simulation, which are available at `http://www.mathworks.com/matlabcentral/answers/94052`. In the further optimisation of our SIMULINK model, we will consider replacing MATLAB function with the MEX S-function, which may help to accelerate the simulation in the merit of its direct communication with the SIMULINK engine (avoid the time consuming compile-link-execute cycle).

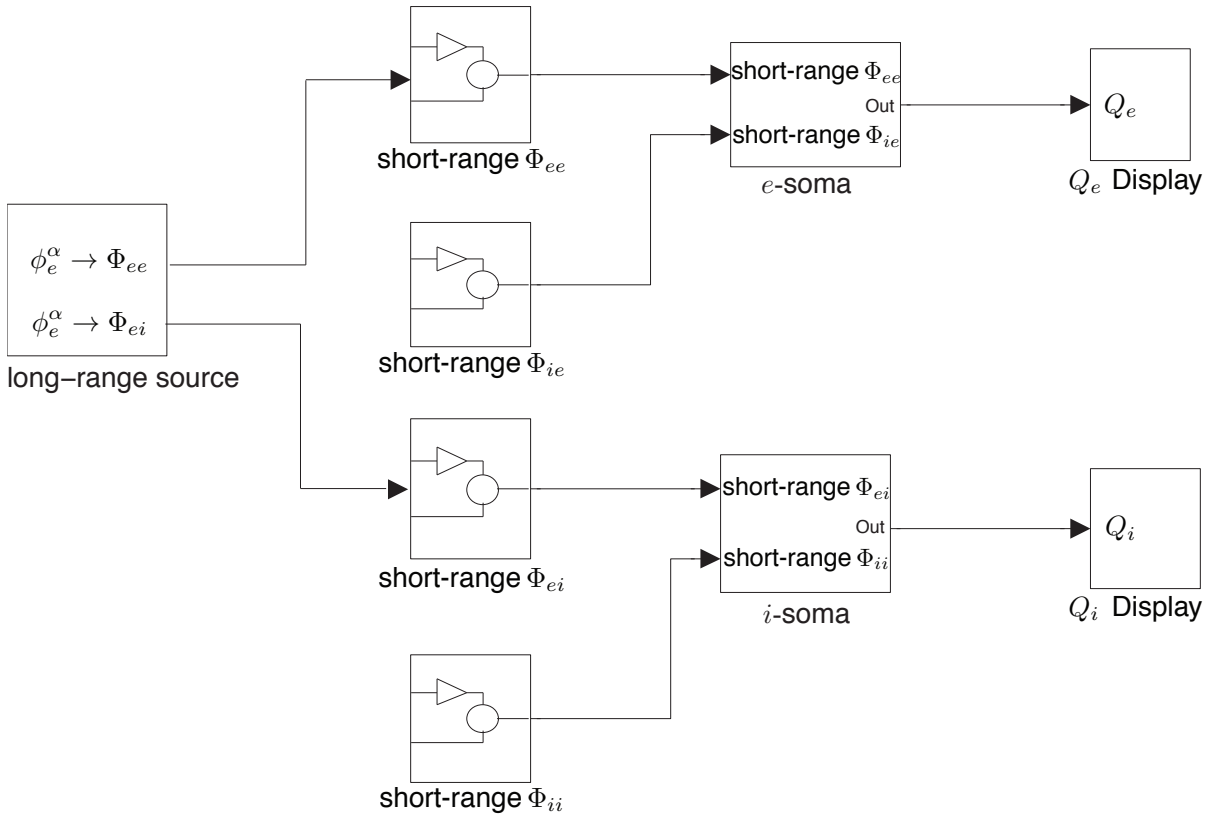


Figure D.12: Overview of the Simulink implementation of the Waikato cortical equations

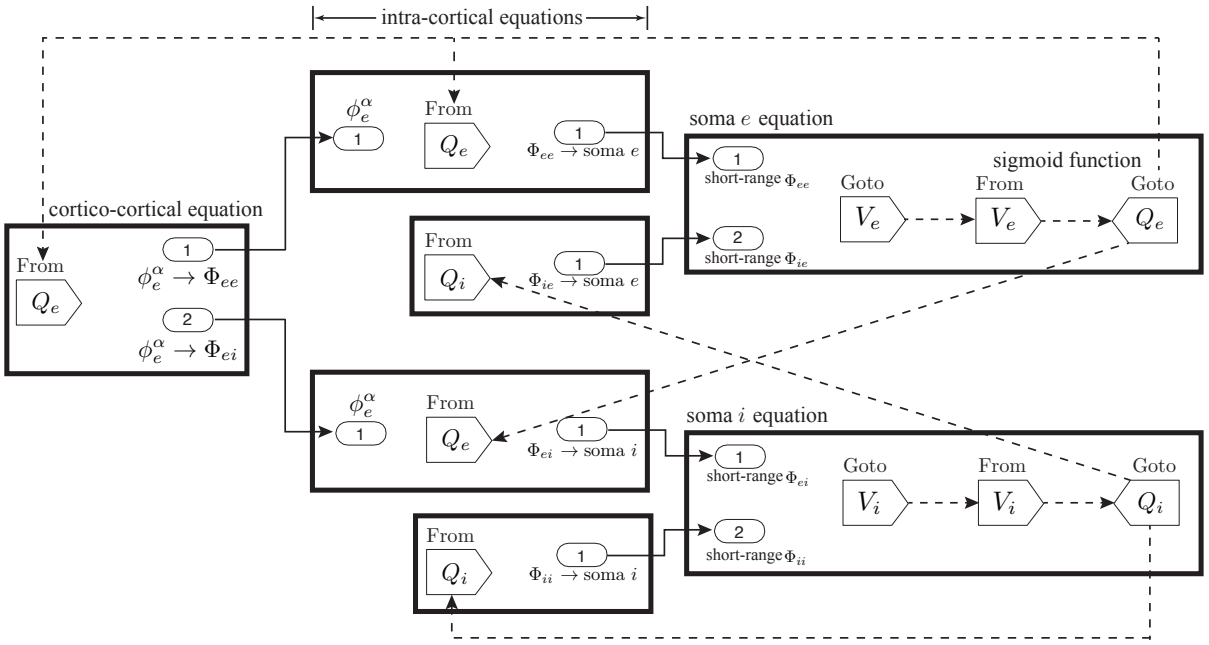


Figure D.13: Detailed connection diagram for the Simulink-based Waikato cortical equations. Solid arrow: direct connection; dashed arrow: `Goto`–`From` connection.

Appendix E

Coding strategies used in `Amp_solving`

This MAPLE tutorial covers basic programming ideas used in `Amp_solving`. We present a stepwise instructed implementation of annotated example code to help readers master our coding strategies in a time-efficient manner. We encourage readers to run these examples in MAPLE for a better understanding of our coding logic.

E.1 Assigning values and defining equations

In MATLAB, we simply use "=" to assign a value to a variable, which will be stored in the MATLAB workspace. But in MAPLE, the assignment is declared by ":=", while "=" stands for an equation. Let us see an example in MAPLE:

```
> a := 5;
```

$$a := 5$$

Here we assign 5 to the variable a, which will be stored in the MAPLE workspace. Hereafter a will automatically be substituted by its value 5 if we call a. Such syntax is the same as in MATLAB:

```
> a + 2;
```

$$7$$

The equation describes an identical relationship between its left and right hand sides. Equation will NOT be stored in the MAPLE workspace:

```
> a = 5:  # "colon" suppresses the output
> a + 2;  # "semicolon" terminates the statement with the output shown
```

$$a + 2$$

We can however assign an equation to a variable:

```
> f := a = 5;
```

$$f := a = 5$$

then substitute this relationship between a and 5 into `a + 2`:

```
> subs(f, %%);
                                        7
```

The `%` sign in MAPLE is a ditto operator allowing us to refer to a previously computed result. In our example, `%%` reevaluates the second last expression computed, i.e., `a + 2`.

E.2 Vectorising variables

There are two ways to vectorise variables in MAPLE. The first, the same as in MATLAB, creates an array in which each element is a variable:

```
> seq(a[i], i=1..5);
                                  a_1, a_2, a_3, a_4, a_5
```

But such manipulation has a drawback: a and its index i are actually separated:

```
> a := 5;
                                        a := 5
> a[1];
                                         5_1
```

Although we can assign 5 to `a[1]` directly:

```
> a[1] := 5;
                                      a_1 := 5
```

the variable a cannot be declared anymore. What is more, MAPLE has difficulty distinguishing a and a_1:

```
> f := a * a[1] + a[2] + a[3]:
> select(has, f, a);  # select terms with a in f
                                   a a_1 + a_2 + a_3
```

To overcome this problem and maintain program stability, we use `||` to concatenate a name and its index:

```
> seq(a||i, i=1..5);
                                  a1, a2, a3, a4, a5
> a := 5;
                                        a := 5
> a1;
                                         a1
> f2 := a * a1 + a2 + a3:
> select(has, f2, a);  # select terms with a in f2
                                      a a1
```

Note that variable a is distinct from "subscripted" variables $a1$, $a2$, $a3$, etc.

E.3 Manipulating expressions

In MAPLE there are many ways to tidy up and manipulate expressions, e.g., `simplify`, `factor` and `collect`. In this section, we will review a few commands used frequently in `Amp_solving`. Commands are explained using simple examples relevant to multiple-scale expansion to help readers master the syntax quickly.

- `collect(f, x)`: Collect all the coefficients with the same rational (fractional) power of x from f.

```
> f := (e*a1 + e^2*b1) * (e*a2 + e^2*b2);
```
$$f := (e\ a1 + e^2\ b1)(e\ a2 + e^2\ b2)$$
```
> f2 := expand(f);
```
$$f2 := e^2\ a1\ a2 + e^3\ a1\ b2 + e^3\ b1\ a2 + e^4\ b1\ b2$$
```
> f3 := collect(f2, e^3);
```
$$f3 := e^4\ b1\ b2 + (a1\ b2 + b1\ a2)\ e^3 + e^2\ a1\ a2$$

- `select(has, f, x)`: Select operands x in f:

```
> select(has, f3, e^3);
```
$$(a1\ b2 + b1\ a2)\ e^3$$

- `remove(has, f, x)`: Remove operands x from f.

```
> remove(has, f3, e^3);
```
$$e^4\ b1\ b2 + e^2\ a1\ a2$$

- `selectremove(has, f, x)`: Partitions into terms containing, and not containing, x in f.

```
> f4 := selectremove(has, f3, e^3);
```
$$f4 := (a1\ b2 + b1\ a2)\ e^3,\ e^4\ b1\ b2 + e^2\ a1\ a2$$
```
> f4[1]; # terms containing e^3
```
$$(a1\ b2 + b1\ a2)\ e^3$$
```
> f4[2]; # terms not containing e^3
```
$$e^4\ b1\ b2 + e^2\ a1\ a2$$

- `coeff(f, x)`: Extract coefficient of x in the polynomial f.

```
> coeff(f3, e^3);
```
$$a1\ b2 + b1\ a2$$

- `subs(x = a, f)`: Substitute x by a in f.

```
> subs(e = E, f);
```
$$(E\ a1 + E^2\ b1)\ (E\ a2 + E^2\ b2)$$

E.4 Coding strategies in multiple-scale expansion

In the first five examples of this section, we will demonstrate how to define a differential equation used in `Amp_solving`. All these ideas will be combined in the last example: Programming the multiple-scale expansion.

E.4.1 Example 1

In MAPLE, to define a partial differential operation in which the derivative acts on the first and second arguments of the function $y(x1, x2, x3)$

$$\frac{\partial^2}{\partial x1 \partial x2} y(x1, x2, x3) + c, \qquad c = \text{constant}$$

we can use "`D`" operator:

```
> f:= D[1,2](y)(x1,x2,x3) + c;
```

$$f := \mathrm{D}_{1,2}(y)(x1,\ x2,\ x3) + c$$

Then convert f to the commonly seen "`diff`" differential form:

```
> f_diff := convert(f, diff);
```

$$f\_diff := \frac{\partial^2}{\partial x2\, \partial x1} y(x1,\ x2,\ x3) + c$$

To have compact display, we replace $y(x1, x2, x3)$ by an alias Y:

```
> alias(Y = y(x1,x2,x3));
```

$$Y$$

```
> f_diff;
```

$$\frac{\partial^2}{\partial x2\, \partial x1}\ Y + c$$

```
> is(Y, function); # Demonstrate that Y is a function
```

$$true$$

```
> is(c, function); # Demonstrate that c is not a function (i.e., is a constant)
```

$$false$$

These codes demonstrate that the `convert(f, diff)` command converts functions only, from "`D`" to "`diff`" form. In MAPLE, a function is defined as a symbol attached with its arguments, the case here is $y(x1, x2, x3)$. Y, the alias of $y(x1, x2, x3)$, is also recognized as a function. But, can we define an alias before applying the `D` operator?

E.4.2 Example 2

```
> restart:
> alias(Y = y(x1,x2,x3));
```

$$Y$$

```
> is(Y, function);
```

$$true$$

```
> f:= D[1,2](Y) + c;
```

$$f := \mathtt{D}_{1,2}(Y) + c$$

```
> f_diff := convert(f, diff);
```

$$f\_diff := \mathtt{D}_{1,2}(Y) + c$$

We see that MAPLE fails to convert `D` to `diff` form if the function is defined in an alias. Thus we conclude that `convert` only works for explicit arguments. If we have a large numbers of functions to input, it will be tedious to write out their full expressions. A solution is given in the following example.

E.4.3 Example 3

```
> restart:
> f := D[1](y1) + D[2](y2) + D[1,2](y3);
```

$$f := \mathtt{D}_1(y1) + \mathtt{D}_2(y2) + \mathtt{D}_{1,2}(y3)$$

In the above partial differential expression f, there are three functions $y1$, $y2$ and $y3$. Assume that they have the same arguments $(x1, x2, x3)$. By converting f to the `diff` form, we simply attach $(x1, x2, x3)$ to f :

```
> f_diff := convert(f(x1,x2,x3), diff);
```

$$f\_diff := \frac{\partial}{\partial x1} y1(x1,\ x2,\ x3) + \frac{\partial}{\partial x2} y2(x1,\ x2,\ x3) + \frac{\partial^2}{\partial x2\,\partial x1}\ y3(x1,\ x2,\ x3)$$

Obviously, the above example shows a more efficient manipulation than what was achieved in **Example 1**. Thus we can use `alias` to exhibit compact display. Note that in this example there are no constant terms in f. Let us examine what happens if a constant term is added to f.

E.4.4 Example 4

```
> restart:
> f := D[1](y1) + D[2](y2) + D[1,2](y3) + c;
```

$$f := \mathtt{D}_1(y1) + \mathtt{D}_2(y2) + \mathtt{D}_{1,2}(y3) + c$$

```
> f_diff := convert(f(x1,x2,x3), diff);
```

$$f\_diff := \frac{\partial}{\partial x1} y1(x1,\ x2,\ x3) + \frac{\partial}{\partial x2} y2(x1,\ x2,\ x3) + \frac{\partial^2}{\partial x2\,\partial x1} y3(x1,\ x2,\ x3) + c(x1,\ x2,\ x3)$$

Unfortunately, by attaching arguments to f, the constant term c becomes a function, which is not what we want. Hence we need to inform MAPLE that c is to be treated as a constant.

E.4.5 Example 5

```
> restart:
> f := D[1](y1) + D[2](y2) + D[1,2](y3) + c;
```

$$f := \mathtt{D}_1(y1) + \mathtt{D}_2(y2) + \mathtt{D}_{1,\ 2}(y3) + c$$

```
> c := ()->c;
```

$$c := (\ ) \rightarrow c$$

```
> f_diff := convert(f(x1,x2,x3), diff);
```

$$f\_diff := \frac{\partial}{\partial x1} y1(x1,\ x2,\ x3) + \frac{\partial}{\partial x2} y2(x1,\ x2,\ x3) + \frac{\partial^2}{\partial x2\,\partial x1} y3(x1,\ x2,\ x3) + c$$

```
> is(c, procedure);
```

$$true$$

In this example, we propose a strategy to hold a constant while converting it from `D` to `diff` form: Define the constant as a *procedure* with itself being the output. As a result, with taking arbitrary inputs the output of c is also c:

```
> for i from 1 to 5 do
    c(i)
  od;
```

$$c$$
$$c$$
$$c$$
$$c$$
$$c$$

Since c has become a procedure, MAPLE preserves it from being attached to other arguments.

To define a sequence of constants, e.g., $c1$, $c2$, $c3$, the following codes show that the "`()->`" definition cannot be directly used in a `for` loop:

```
> for i from 0 to e_order do
      c||i := ()->c||i
 od:
> is(c1, procedure);
```

$$false$$

Alternatively, we can separate the input and the output of a procedure in the `for` loop:

```
> for i from 0 to e_order do
      c||i := subs([variables=(), body=c||i], (variable->body)):
 od:
> is(c1, procedure);
```

$$true$$

Th above idea is useful in defining the expanded bifurcation parameters, e.g., B_1, B_2, B_3 in Eq. (4.4).

E.4.6 Example 6

As a final example, we will demonstrate how to employ these coding strategies in the chain rule for the total derivative after introducing the multiple-scale expansion (MSE) for the operation $\partial u(t,r)/\partial t$.

The MSE of $\partial u(t,r)/\partial t$ starts with a linear perturbation

$$u(t,r) = u_0 + \delta u(t,r)$$

to obtain the perturbation derivative $\partial \delta u(t,r)/\partial t$. MSE introduces a further nonlinear expansion to $\delta u(t,r)$:

$$\delta u(t,r) = \sum_{p=1}^{k} \epsilon^p u_p(T_0, T_1, \ldots T_k; R_0, R_1, \ldots R_k)$$

in which ϵ is an arbitrary small expansion constant; $T_m = \epsilon^m t$ and $R_m = \epsilon^m r$ $(m = 0, 1, 2 \ldots k)$ are scaled temporal and spatial arguments respectively. The following codes show an expansion up to the order ϵ^3.

First, we define the expansion settings:

```
> restart:
> e_order := 3: # expansion to the order of epsilon
> vas := op(op~([seq([R[j],T[j]], j=0..e_order)])); # variable arguments
```

$$vas := R_0,\ T_0,\ R_1,\ T_1,\ R_2,\ T_2,\ R_3,\ T_3$$

`op(op~([]))` merges sets generated by `seq`:

```
> vas_sample := seq([R[j],T[j]], j = 0..e_order);
```

$$vas\_sample := [R_0,\ T_0],\ [R_1,\ T_1],\ [R_2,\ T_2],\ [R_3,\ T_3]$$

```
> op~([vas_sample]);
```

$$[R_0,\ T_0,\ R_1,\ T_1,\ R_2,\ T_2,\ R_3,\ T_3]$$

```
> op(op~([vas_sample]));
```

$$R_0,\ T_0,\ R_1,\ T_1,\ R_2,\ T_2,\ R_3,\ T_3$$

Then we define the original partial derivative

```
> f_original := Dt(u);
```

$$f\_original := Dt(u)$$

`Dt()` is a customised differential operator that is analogue to $\partial/\partial t$. Similarly, we have other customised differential operator conversions:

• Spatial derivative $\nabla \rightarrow$`Ds()`, $\nabla^2 \rightarrow$`Ds@@2()`.

Now let us expand u:

```
> u := u_0 + du;  # du stands for perturbation
> u_0 := ()->u_0; # define constant
```

The assignment of u is automatically substituted into *f_original*:

```
> f_original;
```

$$Dt(u\_0 \ + \ du)$$

Let us introduce the nonlinear expansions:

```
> du :=  sum('e^k*u||k', 'k'=1..e_order); # expand du
```

$$du \ := \ e \ u1 + e^2 \ u2 + e^3 \ u3$$

```
> Dt := sum('e^m*D[2*m+2]', 'm'=0..e_order);  # m: index of temporal expansion
```

$$Dt \ := \ \mathrm{D}_2 + e \ \mathrm{D}_4 \ + \ e^2 \ \mathrm{D}_6 \ + \ e^3 \ \mathrm{D}_8$$

in which the Dt expansion is equivalent to:

$$\frac{\partial}{\partial t} = \frac{\partial}{\partial T_0} + \epsilon \frac{\partial}{\partial T_1} + \epsilon^2 \frac{\partial}{\partial T_2} + \epsilon^3 \frac{\partial}{\partial T_3}$$

i.e.,

$$\mathrm{D}_2 \equiv \frac{\partial}{\partial T_0}, \quad \mathrm{D}_4 \equiv \frac{\partial}{\partial T_1}, \quad \mathrm{D}_6 \equiv \frac{\partial}{\partial T_2}, \quad \mathrm{D}_8 \equiv \frac{\partial}{\partial T_3}$$

in other words, $\mathtt{D}_{2m+2}$ operator in the Dt expansion is the temporal derivative acting on the $(2m+2)$-th temporal argument in *vas*.

We have used e instead of ϵ for simplicity. Since e is a constant, it must be declared:

```
> e := ()->e;
```

Let us examine the current expression of *f_original*:

```
> f_original;
```

$$D_2(u\_0) \ + \ \mathtt{D}_2(e) \ u1 \ + \ e \ \mathtt{D}_2(u1) \ + \ 2\mathtt{D}_2(e) \ e \ u2 \ + \ e^2 \ \mathtt{D}_2(u2) \ + \ 3 \ \mathtt{D}_2(e) \ e^2 \ u3 \ + \ e^3 \ \mathtt{D}_2(u3)$$
$$+ \ e \ (\mathtt{D}_4(u_0) \ + \ \mathtt{D}_4(e) \ u1 \ + \ e \ \mathtt{D}_4(u1) \ + \ 2 \ \mathtt{D}_4(e) \ e \ u2 \ + \ e^2 \ \mathtt{D}_4(u2) \ + \ 3 \ \mathtt{D}_4(e) \ e^2 \ u3$$
$$+ \ e^3 \ \mathtt{D}_4(u3)) \ + \ e^2 \ (\mathtt{D}_6(u_0) \ + \ \mathtt{D}_6(e) \ u1 \ + \ e \ \mathtt{D}_6(u1) \ + \ 2 \ \mathtt{D}_6(e) \ e \ u2$$
$$+ \ e^2 \ \mathtt{D}_6(u2) \ + \ 3 \ \mathtt{D}_6(e) \ e^2 \ u3 \ + \ e^3 \ \mathtt{D}_6(u3)) \ + \ e^3 \ (\mathtt{D}_8(u_0) \ + \ \mathtt{D}_8(e) \ u1 \ + \ e \ \mathtt{D}_8(u1)$$
$$+ \ 2 \ \mathtt{D}_8(e) \ e \ u2 \ + \ e^2 \ \mathtt{D}_8(u2) \ + \ 3 \ \mathtt{D}_8(e) \ e^2 \ u3 \ + \ e^3 \ \mathtt{D}_8(u3))$$

Conversion from `D` to `diff` will eliminate the temporal derivative on constant e:

```
f_diff := convert(f_original(vas), diff);
```

$$f\_diff \ := \ e \left(\frac{\partial}{\partial T_0} u1(R_0, T_0, R_1, T_1, R_2, T_2, R_3, T_3)\right) \ + \ e^2 \left(\frac{\partial}{\partial T_0} u2(R_0, T_0, R_1, T_1, R_2, T_2, R_3, T_3)\right)$$
$$+ \ e^3 \left(\frac{\partial}{\partial T_0} u3(R_0, T_0, R_1, T_1, R_2, T_2, R_3, T_3)\right)$$
$$+ \ e \left(e \left(\frac{\partial}{\partial T_1} u1(R_0, T_0, R_1, T_1, R_2, T_2, R_3, T_3)\right) \ + \ e^2 \left(\frac{\partial}{\partial T_1} u2(R_0, T_0, R_1, T_1, R_2, T_2, R_3, T_3)\right)\right.$$

$$+ e^3 \left(\frac{\partial}{\partial T_1} u3(R_0, T_0, R_1, T_1, R_2, T_2, R_3, T_3)\right)\Bigg)$$
$$+ e^2 \left(e \left(\frac{\partial}{\partial T_2} u1(R_0, T_0, R_1, T_1, R_2, T_2, R_3, T_3)\right) + e^2 \left(\frac{\partial}{\partial T_2} u2(R_0, T_0, R_1, T_1, R_2, T_2, R_3, T_3)\right)\right.$$
$$+ e^3 \left(\frac{\partial}{\partial T_2} u3(R_0, T_0, R_1, T_1, R_2, T_2, R_3, T_3)\right)\Bigg)$$
$$+ e^3 \left(e \left(\frac{\partial}{\partial T_3} u1(R_0, T_0, R_1, T_1, R_2, T_2, R_3, T_3)\right) + e^2 \left(\frac{\partial}{\partial T_3} u2(R_0, T_0, R_1, T_1, R_2, T_2, R_3, T_3)\right)\right.$$
$$+ e^3 \left(\frac{\partial}{\partial T_3} u3(R_0, T_0, R_1, T_1, R_2, T_2, R_3, T_3)\right)\Bigg)$$

Compact display is obtained using `alias`:

```
> alias(seq(u_||i = u||i(vas), i=1..e_order)):
> f_diff;
```

$$e \left(\frac{\partial}{\partial T_0} u\_1\right) + e^2 \left(\frac{\partial}{\partial T_0} u\_2\right) + e^3 \left(\frac{\partial}{\partial T_0} u\_3\right)$$
$$+ e \left(e \left(\frac{\partial}{\partial T_1} u\_1\right) + e^2 \left(\frac{\partial}{\partial T_1} u\_2\right) + e^3 \left(\frac{\partial}{\partial T_1} u\_3\right)\right)$$
$$+ e^2 \left(e\left(\frac{\partial}{\partial T_2} u\_1\right) + e^2 \left(\frac{\partial}{\partial T_2} u\_2\right) + e^3 \left(\frac{\partial}{\partial T_2} u\_3\right)\right)$$
$$+ e^3 \left(e\left(\frac{\partial}{\partial T_3} u\_1\right) + e^2 \left(\frac{\partial}{\partial T_3} u\_2\right) + e^3 \left(\frac{\partial}{\partial T_3} u\_3\right)\right)$$

At last, we rearrange `f_diff` by collecting terms with the same order of e:

```
> f_sorted := collect(f_diff, e);
```

$$f\_sorted := \left(\frac{\partial}{\partial T_3} u\_3\right) e^6 + \left(\frac{\partial}{\partial T_2} u\_3 + \frac{\partial}{\partial T_3} u\_2\right) e^5 + \left(\frac{\partial}{\partial T_1} u\_3 + \frac{\partial}{\partial T_2} u\_2 + \frac{\partial}{\partial T_3} u\_1\right) e^4$$
$$+ \left(\frac{\partial}{\partial T_0} u\_3 + \frac{\partial}{\partial T_1} u\_2 + \frac{\partial}{\partial T_2} u\_1\right) e^3 + \left(\frac{\partial}{\partial T_0} u\_2 + \frac{\partial}{\partial T_1} u\_1\right) e^2 + \left(\frac{\partial}{\partial T_0} u\_1\right) e$$

The coefficient of a specified ϵ (i.e., e in the code) order can be extracted by the command `coeff` for further analysis (e.g., construct order equations). For example, to extract the coefficients of ϵ^2:

```
> coeff(f_sorted, e^2);
```

$$\frac{\partial}{\partial T_0} u\_2 + \frac{\partial}{\partial T_1} u\_1$$

Appendix F

Derivation of Brusselator amplitude equations for the hexagonal mode using `Amp_solving`

In this section, we will demonstrate a practical application of `Amp_solving` to derive the order-3 Brusselator amplitude equations for the hexagonal mode. This amplitude equation allows us to better understand the Turing pattern-forming mechanism, enabling precise control of the patterned structure.

To derive the order-3 amplitude equation, `Amp_solving` is modularised into 5 parts:

1. Nonlinear expansion
2. Linear stability analysis
3. Order-2 solvability condition
4. Solving order-2 equation
5. Order-3 solvability condition

The Order-3 solvability condition yields the order-3 amplitude equations. Higher-order amplitude equations can be obtained by iterating the similar procedures of steps 3, 4 and 5: Order-$(k-1)$ solvability condition $\rightarrow$ Solve order-$(k-1)$ equation $\rightarrow$ Order-k solvability condition.

F.1 Nonlinear expansion

F.1.1 Initialise model configuration

Following the work by Peña et al. [93], we define $\eta = \sqrt{D_X/D_Y}$. The Brusselator model

$$\frac{\partial X}{\partial t} = A - (B+1)X + X^2Y + \eta^2\nabla^2 X$$
$$\frac{\partial Y}{\partial t} = BX - X^2Y + \nabla^2 Y$$

has the critical Turing condition $B_0 = (1 + A\eta)^2$. X and Y are functions of temporal and spatial arguments $(t; r)$ in original scales. We will expand the variable to third order in ϵ:

$$\mathbf{u}(t;r) = \mathbf{u_0} + \epsilon\mathbf{u_1}(T_0, T_1, T_2, T_3; R_0, R_1, R_2, R_3) + \epsilon^2\mathbf{u_2}(T_0, T_1, T_2, T_3; R_0, R_1, R_2, R_3) + \epsilon^3\mathbf{u_3}(T_0, T_1, T_2, T_3; R_0, R_1, R_2, R_3) \tag{F.1}$$

then solve the amplitude equation at order ϵ^3.

To start the computerised MSE, we first initialise `Amp_solving` for the expansion order, system dimension (number of model equations) and multiple-scaled arguments.

```
> # Initialisation
> e_order := 3: # Expansion order
> sys_dim := 2: # System dimensions
> vas := op(op~([seq([R[m],T[m]], m = 0..e_order)])); # Variable arguments
```

$$vas := R_0,\ T_0,\ R_1,\ T_1,\ R_2,\ T_2,\ R_3,\ T_3$$

- `e_order`: expansion order of the final amplitude equation
- `sys_dim`: number of model equations
- `vas`: variable arguments. Temporal and spatial arguments in multiple-scales

F.1.2 Define model equations

In `Amp_solving`, model equations are declared via our customised syntax:

- Differential operator conversion
  Temporal derivative $\partial/\partial t$ →`Dt()`; Spatial derivative ∇ →`Ds()`, ∇^2 →`Ds@@2()`.
- Vectorised variables
  Brusselator variables X and Y are defined as `uo1` and `uo2` respectively. The symbol `o` means that the variable has the original temporal and spatial scales. In general, we have the variable definition `uoi`, in which `i` = 1...`sys_dim`. In MAPLE programming, `uoi` is equivalent to `uo||i`, in which `||` is an MAPLE operator to concatenate expressions. The form `uo||i` is useful in a `for` loop over index `i`.
- Define constants
  Constants of the model are defined as MAPLE *procedures* to preserve their independence (see Appendix E.4 for a detailed explanation).

Following these rules, the Brusselator mode equations are defined as:

```
> # Define the Brusselator model
> ## Activator
> PDE[1] := Dt(uo1) = A - (Bo + 1) * uo1 + uo1^2 * uo2 + N^2 * (Ds@@2)(uo1);
> A := ()->A: # define the constant as a procedure
> N := ()->N:
>
```

```
> ## Inhibitor
> PDE[2] := Dt(uo2) = Bo * uo1 - uo1^2 * uo2 + (Ds@@2)(uo2);
```

- `Bo`: the original and unexpanded bifurcation control parameter B
- `PDE[i]`: i-th partial differential equation of the model
- `N`: $\eta = \sqrt{D_X/D_Y}$

F.1.3 Expansions

1. Weakly linear perturbation
   Expand the model variable $\mathbf{u}$ as a small perturbation $\boldsymbol{\delta}\mathbf{u}$ around the steady state $\mathbf{u_0}$:

   $$\mathbf{u} = \mathbf{u_0} + \boldsymbol{\delta}\mathbf{u}$$

   Following codes automatically solve $\mathbf{u_0}$ and substitute the linear expansion to the model equation `PDE[i]`.

```
> # Remove temporal and spatial derivatives
> for i from 1 to sys_dim do
>     PDE_ss[i] := 0 = expand(subs(Ds = 0, rhs(PDE[i])));
> od:
> # Automatically solve homogeneous steady-states and store them in ss
> ss := solve( seq(PDE_ss[i], i=1..sys_dim),
               seq(uo||i, i=1..sys_dim) );
```

$$ss := \left\{ uo1 = A,\ uo2 = \frac{Bo}{A} \right\}$$

```
> # Introduce linear perturbations
> for i from 1 to sys_dim do
      uo||i      := u||i||_||0 + du||i;  # Linear perturbation
      u||i||_||0 := rhs(ss[i]);          # Substitute steady-states
  od:
```

   - `dui`: $\boldsymbol{\delta}\mathbf{u}$, small perturbation around the steady state `ui_0`

2. Expand differential operators
   (see Appendix E.4 for more details)
   Introduce differential operator expansions Eqs. (4.7) and (4.8):

```
> Dt := sum('e^m*D[2*m+2]', 'm'=0..e_order): # Expand temporal derivative
> Ds := sum('e^m*D[2*m+1]', 'm'=0..e_order): # Expand spatial derivative
> e := ()->e:  # Define scaling constant
```

3. Nonlinear expansions
   Introduce the nonlinear variable expansion

   $$\boldsymbol{\delta}\mathbf{u} = \epsilon\mathbf{u_1} + \epsilon^2\mathbf{u_2} + \epsilon^3\mathbf{u_3}$$

and the bifurcation parameter expansion

$$B = B_0 + \epsilon B_1 + \epsilon^2 B_2 + \epsilon^3 B_3 \tag{F.2}$$

```
> for i from 1 to sys_dim do                    # Nonlinear variable expansion
      du||i :=  sum('e^j*u||i||j', 'j'=1..e_order);
 od:
> Bo := sum('e^i*B_||i', 'i'=0..e_order);      # Bifurcation parameter expansion

> # Define expanded bifurcation parameters as constants
> for i from 0 to e_order do
      B_||i := subs([variables=(), body=B_||i], (variables-> body)):
 od:
```

- `uij`: expanded variables. `i` stands for the variable index, `j` is the variable expansion index. i.e., for the Brusselator model

$$\mathbf{u_1} = \begin{bmatrix} \mathrm{u11} \\ \mathrm{u21} \end{bmatrix}, \quad \mathbf{u_2} = \begin{bmatrix} \mathrm{u12} \\ \mathrm{u22} \end{bmatrix}, \quad \mathbf{u_3} = \begin{bmatrix} \mathrm{u13} \\ \mathrm{u23} \end{bmatrix}$$

4. Obtain order equations

   The expanded terms (differential operators, variables and bifurcation parameters) will be substituted back into the model equations automatically. Then we convert the differential expression from `D` to `diff` mode.

```
> # Define alias for compact display
> for i from 1 to sys_dim do
      alias(seq(u||i||_||j = u||i||j(vas), j=1..e_order));
 od:

> # Convert D to diff mode for differential equations
> for i from 1 to sys_dim do
      PDE_sorted[i]:= simplify(collect(PDE[i] ,e), e^(e_order+1)=0):
      PDE_diff[i]  := convert(PDE_sorted[i](vas), diff):
 od:
```

- `PDE_sorted[i]`: fully expanded model equation up to the order $\epsilon^{\texttt{e\_order}}$
- `PDE_diff[i]`: `PDE_sorted[i]` in the `diff` form

Using `coeff()` to extract coefficients at a specific order of ϵ from both sides of `PDE_diff[i]`, we can obtain a series of raw order equations. This is done automatically in `Amp_solving` by scanning each model equation at all ϵ orders.

```
> # Collect terms with the same order of e
> for i from 1 to sys_dim do
      for j from 1 to e_order do # ith equation at order j
          eq||i||_order||j :=
             expand(coeff(lhs(PDE_diff[i]), e,j) = coeff(rhs(PDE_diff[i]), e,j));
      od:
 od:
```

- `eqi_orderj`: raw order equation, read as the i-th equation of ϵ^j. It is obtained by equating the coefficients of ϵ^j from both sides of the expanded equation.

We need to rearrange the raw order equation `eqi_orderj` to the form as Eqs. (4.10), (4.13) and (4.21), so that the LHS of the order equation has only the linear operation $(\partial/\partial T_0 - L_c)$ on the current order variables. We use `selectremove()` to *select* the current order variables then group them at the LHS while placing other terms at the RHS (see Sec. 4.4.1 for more details). The sorted order equation is read `eqi_orderj_diff`, the suffix of which `diff` indicates its explicit differentiation display.

```
> # Tidy up order equations
> for i from 1 to sys_dim do
    for j from 1 to e_order do

       PDE_temp||i||j := lhs(eq||i||_order||j) - rhs(eq||i||_order||j);
       PDE_components||i||j := selectremove(has,PDE_temp||i||j,
                                            seq(u||ii||j, ii=1..sys_dim) );
       eq||i||_order||j||_diff := PDE_components||i||j[1]
                                            = -PDE_components||i||j[2];
    od:
 od:
```

At last, we vectorise the RHS of order equations at each ϵ order to a column vector (e.g., $\mathbf{I_j}$ in Eqs. (4.13) and (4.21)). This is required for the scalar product of the solvability condition. At the expansion order j, RHS for all order-j equations are stored in `eq_vec_orderj` (representing $\mathbf{I_j}$).

```
> # Vectorise rhs of order equations
> for j from 1 to e_order do
      eq_vec_order||j :=
             Vector[column]([seq(rhs(eq||i||_order||j||_diff), i = 1..sys_dim)]);
 od:
```

F.2 Linear stability analysis

The order-1 equation recovers the linear stability analysis (LSA), which is coded in a separate MAPLE worksheet `LSA.mw`. To do this, `Amp_solving` will save the order-1 equation to a `.txt` file:

```
> eqlist_order1 := seq(eq||i||_order1, i = 1..sys_dim);
> save eqlist_order1, e_order, sys_dim, vas, `eqlist_order1.txt` :
```

LSA in `LSA.mw` is coded as following:

```
> # LSA.mw
>
```

```
> read "eqlist_order1.txt":
> # convert from diff to D form prepared for Jac calculation
> eqlist := convert(eqlist_order1, D):
> # Remove function properties required for deriving the Jac matrix
> eqlist_subs := subs(seq(D[2](u||i||1)(vas) = D[2](u||i||1), i=1..sys_dim),
                      seq(D[1,1](u||i||1)(vas) = D[1,1](u||i||1), i=1..sys_dim),
                      seq((u||i||1)(vas) = (u||i||1), i=1..sys_dim), eqlist);
> # Jacobian matrix
> Jac_list := [seq(rhs(eqlist_subs[i]), i=1..sys_dim)]: # eq list
> Jac_vars := [seq(u||i||1, i=1..sys_dim )]:
> Jac := VectorCalculus[Jacobian](Jac_list, Jac_vars);
> Jac_ss := subs( seq(diff(D[1, 1](u||i||1), u||i||1) = -q^2, i=1..sys_dim), Jac):
> eig_Jac := LinearAlgebra[Eigenvalues](Jac_ss):  # eigenvalues
>
> # Pure Turing condition: vanished eigenvalue
> eig_real := remove(hastype, expand(eig_Jac[1]), sqrt); # real part of the dom eig
> eig_sqrt := select(hastype, expand(eig_Jac[1]), sqrt); # imag part of the dom eig
> eig_zero := eig_real = eig_sqrt: # force to get zero eigenvalue
> B_T := expand(solve(eig_zero, B_0));  # B is a function of q
>
> # Critical wavenumber
> q_eq := convert(select(has, B_T, q), `list`):
> qc_T := solve(q_eq[1] = q_eq[2], q)[1]:  # only choose the positive solution
>
> # At the critical wavenumber, derive the critical B
> assume(A>0, q>0, N>0): interface(showassumed=0):
> Bc_T := 2*sqrt(q_eq[1]) * sqrt(q_eq[2]) + remove(has, B_T, q):
>
> # Derive the right and left critical eigenvectors
> unassign('q'):
> Jac_T := simplify(subs(B_0 = Bc_T, q = qc_T, Jac_ss)):
> RT, eigT := MTM[eig](Jac_T):
> R_T := RT(1..2, 2)/(RT(1,2)):                  # Turing right eigenvector
> Jac_T_Tr := MTM[transpose](Jac_T):
> LT, L2 := MTM[eig](Jac_T_Tr);
> L_T := MTM[transpose](LT(1..2, 1))/(LT(1,1));  # Turing left eigenvector
>
> save R_T, L_T, Bc_T, qc_T, `Brusselator_Tsets.txt`;
```

From `LSA.txt`, the critical Turing condition, the critical wavenumber, and critical right and left eigenvectors are calculated and saved in a standard `.txt` file that can be loaded in the main program by using the `read` command.

```
> read "Brusselator_Tsets.txt":   # read LSA results
> Tsets := {
             seq(RT||i = R_T(i), i=1..sys_dim),   # right eigenvectors
             seq(LT||i = L_T(i), i=1..sys_dim),   # left eigenvectors
             B_0 = Bc_T                           # Turing condition
           };
> Tsets_extra := q0 = qc_T;         # critical wavenumber
```

Note that we define the critical eigenvectors and wavenumber implicitly in Maple *sets* since we only need their explicit expressions when necessary. e.g., we want to extract the coefficient of e^y from the expression f:

```
> f := (y*exp(-y) + x*exp(2*y))^2:
> exp_coeff := coeff(expand(f), exp(y));
```

$$exp\_coeff := 2\ x\ y$$

Suppose that x and y are functions of a and b, we can define a comma-separated *set* to include their expressions:

```
> fun_set := y = 2*a + 5*b, x = (a + b)^2:
```

To obtain the explicit form of *exp_coeff*, we simply substitute `fun_set` into it:

```
> subs(fun_set, exp_coeff)
```

$$(2(2a+5b))(a+b)^2$$

If x and y are declared explicitly before f, we are not able to extract the coefficient of e^y since it will automatically become e^{2a+5b} in f. If x and y are declared explicitly after f, we can extract the coefficient of e^y, but f is not in its original form anymore. This idea is important to define the structure ansatz since it is a function of the wavenumber q_0, more specifically, it is an exponential function of q_0: e.g., $e^{i\vec{q_1}\vec{r}}$, $|\vec{q_1}| = q_0$. In the solvability condition, we need to extract the coefficient of exponential functions, thus we must keep q_0 as a symbol. Moreover, this technique enhances computing efficiency dramatically since we can substitute the explicit expressions when necessary.

F.3 Order-2 solvability condition

F.3.1 Structure ansatz

The structure ansatz for the hexagonal mode is written as:

$$\mathbf{u_1} = A_{\mathrm{T}1}\ e^{i\vec{q_1}\cdot\vec{r_0}}\ \mathbf{R_T} + A_{\mathrm{T}2}\ e^{i\vec{q_2}\cdot\vec{r_0}}\ \mathbf{R_T} + A_{\mathrm{T}3}\ e^{i\vec{q_3}\cdot\vec{r_0}}\ \mathbf{R_T} + c.c.$$

where *c.c.* stands for complex conjugate pairs. $\vec{q_1}$, $\vec{q_2}$ and $\vec{q_3}$ are three critical wavevectors with identical modulus (equal to the critical wavenumber q_0) and 120° angular separation. One wavevector coincidences with the resultant wavevector of other two. In `Amp_solving`, we state this condition as `q_loop`:

$$\vec{q_1} + \vec{q_2} = -\vec{q_3},\ \vec{q_1} + \vec{q_3} = -\vec{q_2},\ \vec{q_2} + \vec{q_3} = -\vec{q_1}, \tag{F.3}$$

In the meantime, the product of any two wavevectors has the `q_times` relationship:

$$\vec{q_1}\cdot\vec{q_2} = \vec{q_1}\cdot\vec{q_3} = \vec{q_2}\cdot\vec{q_3} = q_0^2\cos(\theta), \quad \theta = 120^\circ \tag{F.4}$$

$A_{\mathrm{T}1}$, $A_{\mathrm{T}2}$ and $A_{\mathrm{T}3}$ are mode amplitudes. The mode amplitude describes the mode dynamics in slow temporal and extended spatial scales, thus depending on scaled arguments $(T_1, T_2; R_1, R_2)$, that is, `aas`(amplitude arguments) in the program.

```
> aas := op(op~([seq([R[j],T[j]], j = 1..e_order)])); # Amplitude arguments
>
> # Right and left eigenvectors
> RT := Vector[column]([seq(RT||i, i=1..sys_dim)]):
> LT := Vector[column]([seq(LT||i, i=1..sys_dim)]):
>
> # Apply alias for compact amplitude display
> alias(AT1=A_T1(aas),   AT1c=A_T1c(aas)):
> alias(AT2=A_T2(aas),   AT2c=A_T2c(aas)):
> alias(AT3=A_T3(aas),   AT3c=A_T3c(aas)):
>
> # Wavevector conditions for the hexagonal mode
> q_subs  := subs(Tsets_extra, [q1=q0, q2=q0, q3=q0]):
> q_times := [q1*q2 = q0^2*(-1/2), q1*q3 = q0^2*(-1/2), q2*q3 = q0^2*(-1/2)]:
> q_loop  := q2+q3=-q1, q1+q3=-q2, q1+q2=-q3:
>
> # Order 1 ansatz
> ans1 := AT1*exp(I*q1*R[0])*RT + AT2*exp(I*q2*R[0])*RT +
          AT3*exp(I*q3*R[0])*RT + AT1c*exp(-I*q1*R[0])*RT +
          AT2c*exp(-I*q2*R[0])*RT + AT3c*exp(-I*q3*R[0])*RT;
```

- `AT1c, AT2c, AT3c`: $A^*_{\mathrm{T}1}, A^*_{\mathrm{T}2}, A^*_{\mathrm{T}3}$, complex conjugate pairs.
- `I`: Maple notation for the complex number $i \equiv \sqrt{-1}$.

F.3.2 Order-2 solvability condition

Recall the column vector `eq_vec_orderj` (where `j` represents the expansion order, i.e., $\mathbf{I_j}$ in Eqs. (4.13) and (4.21)). Here, we apply the `DotProduct` (from `VectorCalculus` package) between the critical left eigenvector `LT` and the RHS of the order-2 equations `eq_vec_order2` for the orthogonal condition (see. Eq. (4.16)). According to Eqs. (4.37), the solvability condition yields a series of constraints that require us to collect secular terms and force them to zero.

```
> # Orthogonal condition
> SC2 := VectorCalculus[DotProduct](LT, eq_vec_order2):
>
> # Substitute order 1 ansatz
> SC2_temp1 := PDETools[dsubs]([seq(u||i||1(vas)=ans1[i], i=1..sys_dim)], SC2):
> SC2_temp2 := simplify(expand(SC2_temp1)):
>
> # Substitute wavevector conditions for the hexagonal mode
> SC2_temp3 := subs(q_loop, SC2_temp2):
>
> # Collect secular terms and force them to zero
> for i from 1 to 3 do
    SC2_temp4_||i   := coeff(simplify(SC2_temp3), exp(I*q||i*R[0])):
    SC2_temp4cc_||i := coeff(simplify(SC2_temp3), exp(-I*q||i*R[0])):

    SC2_temp5_||i   := collect(simplify(SC2_temp4_||i), B_1, diff, 'distributed'):
    SC2_temp5cc_||i := collect(simplify(SC2_temp4cc_||i), B_1, diff, 'distributed'):

    SC2_temp6_||i   := collect(simplify(subs(Tsets, SC2_temp5_||i)), B_1, diff, AT);
    SC2_temp6cc_||i := collect(simplify(subs(Tsets, SC2_temp5cc_||i)), B_1, diff, AT);
```

```
    # Eliminate secular terms
    SC2_Case_||i    := denom(simplify(SC2_temp6_||i))*simplify(SC2_temp6_||i) = 0;
    SC2_Casecc_||i  := denom(simplify(SC2_temp6cc_||i))*simplify(SC2_temp6cc_||i)  = 0;

    # Solve B_1 from above equations
    B1_subsCase||i   := B_1 = solve(SC2_Case_||i, B_1);
od:
```

The `for` loop in the above code extracts all six solvability conditions (i.e., Eqs. (4.37)), yielding six constraints: `SC2_Case_1`, `SC2_Case_2` and `SC2_Case_3`; conjugate pairs `SC2_Casecc_1`, `SC2_Casecc_2` and `SC2_Casecc_3`. By examining the constraint `SC2_Case_1`:

```
> SC2_Case_1;
```

$$AT1\ A\ B\_1 - \left(\frac{\partial}{\partial T_1} AT1\right) A - \left(\frac{\partial}{\partial T_1} AT1\right) A\hat{}2\ N + \left(\frac{\partial}{\partial T_1} AT1\right) A\ N^2 + \left(\frac{\partial}{\partial T_1} AT1\right) A^2\ N^3 - 2\ AT2c\ AT3c\ N^2\ A^2 + 2\ AT2c\ AT3c = 0$$

we see that the mode amplitude $A_{\mathrm{T}1}$ does not depend on T_1 or R_1, thus we can re-define the amplitude arguments `aas`:

```
> aas := R[2], T[2], R[3], T[3];
```

Now `SC2_Case_1` outputs:

```
> SC2_Case_1;
```

$$AT1\ A\ B\_1 - 2\ AT2c\ AT3c\ N^2\ A^2 + 2\ AT2c\ AT3c = 0$$

`B_1` is solved from above equation and stored in `B1_subsCase1`.

F.4 Solution of order-2 equation

`Amp_solving` follows the exponential pattern matching method (see Secs. 4.3.4 and 4.4.2 for detailed explanations) to solve the order equation, which we will demonstrate in the following five sections.

F.4.1 Expand the RHS of the order-2 equation

To construct coefficient equations, we need to first determine the precise structure of the order-2 solution, which can be derived by examining the fully expanded order-2 equation. To do this, we will substitute the order-2 solution structure and the order-2 solvability conditions into the order-2 equation.

```
> for j from 1 to sys_dim do
    # Subs structure ansatz
    eq||j||_order2_withAnsatz_temp1 := PDETools[dsubs](
                                [seq(u||i||1(vas)=ans1[i], i=1..sys_dim)],
                                                    eq||j||_order2_diff):
    # Subs Turing conditons
    eq||j||_order2_withAnsatz_temp2 := subs(op(Tsets), eq||j||_order2_withAnsatz_temp1):
    # Subs q_loop condition
    eq||j||_order2_withAnsatz_temp3 := subs(q_loop,
                        simplify(expand(eq||j||_order2_withAnsatz_temp2)) ):
    # Collect exp functions
    eq||j||_order2_withAnsatz_temp4 :=
                               collect(eq||j||_order2_withAnsatz_temp3, exp):
    # Subs order-2 solvability conditions
    eq||j||_order2_withAnsatz := subs(seq(-SC2_Case_||i, i=1..3),
                                      seq(-SC2_Casecc_||i, i=1..3),
                                          eq||j||_order2_withAnsatz_temp4):
od:
```

F.4.2 Determine the structure of the order-2 solution

Amp_solving uses op() to extract unique exponential components from the fully expanded order-2 equation eqj_order2_withAnsatz, then produces a series of coefficients ci_j to determine the structure of the order-2 solution.

```
> # Extract exp components
> exp_fun := op(select(has,
             indets(simplify(expand(eq||1||_order2_withAnsatz))), exp));
>
> # Unknown coefficient cij, i-ith equations, j-jth coefficient
> cas := aas;
> for i from 1 to sys_dim do
      seq( alias(c||i||_||j=c||i||j(cas)),  j=1..nops([exp_fun])+1 );
  od:
>
> # Order 2 ansatz
> for i from 1 to sys_dim do
      ans2[i] := sum('c||i||_||j * exp_fun[j]', 'j'=1..nops([exp_fun]))
                                          + c||i||_||(nops([exp_fun])+1):
 od:
```

It should be mentioned that coefficient ci_j is not a constant but a function having the same dependence as the mode amplitude, so we name the coefficient dependence cas (coefficient arguments). It should also be noticed that the order-2 solution structure ans2 and structure ansatz ans1 share the same dependence vas.

F.4.3 Substitute order-2 solution structure into the LHS of the order-2 equation

```
> for i from 1 to sys_dim do
     # Subs order 2 ansatz into the LHS
     eq||i||_lhs_temp1 := collect(
        PDETools[dsubs]([seq(u||i||2(vas)=ans2[i], i=1..sys_dim)],
                                        lhs(eq||i||_order2_withAnsatz)), exp):
     # Subs q_times conditions into the LHS
     eq||i||_lhs_temp2 := collect(simplify(simplify(eq||i||_lhs_temp1, q_times)), exp);
     # Re-construct the order 2 equation
     eq||i||_order2_subs:= eq||i||_lhs_temp2  = rhs(eq||i||_order2_withAnsatz):
 od:
```

F.4.4 Construct coefficient equations by matching exponential patterns

Now, the order-2 equation has been fully expanded with both sides in the summation of exponential series (can be checked by calling `eq1_order2_subs` or `eq2_order2_subs`).

In Sec. 4.3.4, we see that the exponential series of the expanded order-2 equation starts from the zero power. `Amp_solving` treats separately these exponential patterns that have nonzero and zero exponential powers since `op()` cannot extract terms with zero exponential power (i.e., ϵ^0) directly. Assuming the model has i equations and j nonzero exponential components, there will be a total of $2(i \cdot j + i)$ coefficient equations.

```
> for i from 1 to sys_dim do
    # Coefficient equations for nonzero exp power
    for j from 1 to nops([exp_fun]) do
       eq||i||_coef||j :=
            coeff( lhs(eq||i||_order2_subs),  exp_fun[j]  ) =
            coeff( rhs(eq||i||_order2_subs),  exp_fun[j]  );
    od:

    # Coefficient equations for zero exp power
       eq||i||_coef||(nops([exp_fun])+1) :=
            remove(has, lhs(eq||i||_order2_subs),
                   seq(exp_fun[j], j=1..nops([exp_fun])) ) =
            remove(has, rhs(eq||i||_order2_subs),
                   seq(exp_fun[j], j=1..nops([exp_fun])) );
 od:
```

F.4.5 Solve coefficient equations

To solve unknown coefficients `ci_j` in the order-2 solution structure `ans2`, `Amp_solving` will group coefficient equations in `eq_list` and unknown coefficients in `coef_list`, then feed them to the MAPLE solver `solve()` to obtain the solution `coeff_order2`.

```
> # List of coefficient equations
> eq_list  := [seq(seq(eq||i||_coef||j, i=1..sys_dim), j=1..nops([exp_fun])+1 )]:
> # List of unknwon coefficients
> coef_list := [seq(seq(c||i||_||j, i=1..sys_dim ), j=1..nops([exp_fun])+1 )]:
> # Solve coefficients
> coeff_order2_temp := solve(eq_list, coef_list):
> coeff_order2 := subs(Tsets_extra, subs(q_subs, coeff_order2_temp) ):
```

F.5 Order-3 solvability condition

The order-3 solvability condition is derived in the same fashion as the order-2 solvability condition: Substitute the structure ansatz and order-2 solution into the RHS of the order-3 equation `eq_vec_order3`, then apply the orthogonal condition to find secular terms and force them to zero. The order-3 amplitude equation appears here.

F.5.1 Substitute the structure ansatz and order-2 solution into the order-3 equation

The explicit order-2 solution $\mathbf{u_2}$ is obtained by plugging solved coefficients `coeff_order2` into the corresponding ansatz `ans2`.

```
> # Explicit order-2 solution
> ans2_subs          := subs(op(op(coeff_order2)), seq(ans2[i], i=1..sys_dim)):
>
> # Subs the structure ansatz and order-2 solution into the RHS of
    the order-3 equation
> eq_vec_order3_subs := subs(
                [seq(u||i||1(vas)=ans1[i], i=1..sys_dim)],
            [seq(u||i||2(vas)=ans2_subs[i], i=1..sys_dim)], eq_vec_order3):
```

F.5.2 Apply the orthogonal condition

```
> # Orthogonal condition
> SC3       := VectorCalculus[DotProduct](LT, eq_vec_order3_subs):
> # Collect exp components
> SC3_temp1 := collect(simplify(SC3), exp):
> # Subs q_loop condition
> SC3_temp2 := subs(q_loop, SC3_temp1):
> # Extract secular terms
> SC3_temp3 := simplify( subs(Tsets, coeff(SC3_temp2, exp(I*q1*R[0]))) ) = 0:
> # Tidy up the amplitude equation for the q1 mode
> SC3_temp4 := collect(SC3_temp3, diff,factor):
```

The hexagonal mode has 6 amplitude equations relating to $A_{\mathrm{T}1}$, $A_{\mathrm{T}2}$ and $A_{\mathrm{T}3}$; $A^*_{\mathrm{T}1}$, $A^*_{\mathrm{T}2}$ and $A^*_{\mathrm{T}3}$. Here, we derive the amplitude equation only with respect to $A_{\mathrm{T}1}$ (i.e.,

SC3_temp4). Amplitude equations about $A_{\mathrm{T}2}$ and $A_{\mathrm{T}3}$ can be obtained by permutations of indices. We then tidy up SC3_temp4 and return its argument scales.

F.5.3 Tidy up amplitude equation

The following algorithm will automatically sort out the raw amplitude equation into the form of Eq. (4.24): Find the derivative term and place it at the LHS, keeping other terms at the RHS.

```
> # Extract the coefficient of the temporal derivative
> AT_coeff   := PDEtools[dcoeffs](lhs(SC3_temp4), AT1)[1]:
> # Normalize the coefficient of the temporal derivative Eq.
> SC3_temp5  := collect(expand(SC3_temp4/(AT_coeff)), diff,factor):
> # Select the temporal derivative
> time_diffT := selectfun(SC3_temp5, diff)[1]:
> # Place the temporal derivative at LHS, other terms at RHS
> SC3_temp6  := time_diffT = -selectremove(has, lhs(SC3_temp5), time_diffT)[2]:
> # AT1_abs: modulus of the amplitude AT1, i.e., |AT1|.
> SC3_temp7  := simplify(SC3_temp6,
              AT1*AT1c = AT1_abs^2, AT2*AT2c = AT2_abs^2, AT3*AT3c = AT3_abs^2):
> # Tidy up the equation
> SC3_temp8  := collect(SC3_temp7, AT1_abs, AT2_abs, AT3_abs, factor);
```

$$
\begin{aligned}
SC3\_temp8 \;:=\; \frac{\partial}{\partial T_2} AT1 \;=\; & -\frac{1}{9}\frac{AT1\ (A\ N+2)\ (8\ N^2\ A^2\ +\ 4\ -\ 21\ A\ N)\ AT1\_abs^2}{(N-1)\ (N+1)\ A^3\ N} \\
& -\frac{AT1\ (3\ A^3\ N^3\ -\ 7\ N^2\ A^2-5\ A\ N+3)\ AT2\_abs^2}{(N-1)\ (N+1)\ A^3\ N} \\
& -\frac{AT1\ (3\ A^3\ N^3\ -\ 7\ N^2\ A^2-5\ A\ N+3)\ AT3\_abs^2}{(N-1)\ (N+1)\ A^3\ N} \\
& -\frac{2\ B\_1\ AT2c\ AT3c\ +\ B\_2\ A\ AT1}{(N-1)(N+1)(1+A\ N)A}
\end{aligned}
$$

F.5.4 Return arguments to normal scales

Assume $Z_1 = \epsilon A_{\mathrm{T}1}$ (representing amplitude scaling), where Z_1 is the amplitude carrying unscaled arguments (r_0, T_0), namely nas: normal arguments. Meanwhile considering $T_2 = \epsilon^2 T_0$ (temporal argument scaling), the LHS of SC3_temp8 turns to be $1/\epsilon^3 \cdot \partial Z_1/\partial T_0$. Similarly, we have the following conversions:

$$
\begin{array}{lll}
A_{\mathrm{T}1} = Z_1/\epsilon & A_{\mathrm{T}2} = Z_2/\epsilon & A_{\mathrm{T}3} = Z_3/\epsilon \\
A^*_{\mathrm{T}1} = Z^*_1/\epsilon & A^*_{\mathrm{T}2} = Z^*_2/\epsilon & A^*_{\mathrm{T}3} = Z^*_3/\epsilon \\
|A_{\mathrm{T}1}|^2 = |Z_1|^2/\epsilon^2 & |A_{\mathrm{T}2}|^2 = |Z_2|^2/\epsilon^2 & |A_{\mathrm{T}3}|^2 = |Z_3|^2/\epsilon^2
\end{array}
\tag{F.5}
$$

Substituting these relationships into SC3_temp8 generates the Z_1 amplitude equation with normal scales.

```
> nas := R[0], T[0]; # normal arguments
> alias(Z1=Z_1(nas),   Z1c=Z1_c(nas)):
> alias(Z2=Z_2(nas),   Z2c=Z2_c(nas)):
> alias(Z3=Z_3(nas),   Z3c=Z3_c(nas)):
> SC3_temp9 := 1/e^3 * diff(Z1, T[0]) = subs(
                 AT1 = Z1/e,   AT2 = Z2/e,   AT3 = Z3/e,
                AT1c = Z1c/e, AT2c = Z2c/e, AT3c = Z3c/e,
                AT1_abs^2 = Z1_abs^2/e^2, AT2_abs^2 = Z2_abs^2/e^2,
                AT3_abs^2 = Z3_abs^2/e^2, rhs(SC3_temp8)):
> SC3_temp10 := simplify(e^3*SC3_temp9):
> SC3_temp11 := collect(SC3_temp10, Z1_abs, Z2_abs, Z3_abs, factor);
```

$$
\begin{aligned}
SC3\_temp11 \;:=\; \frac{\partial}{\partial T_0} Z1 \;=\; & -\frac{1}{9}\frac{Z1\ (A\ N+2)\ (8\ N^2\ A^2+4-\ 21\ A\ N)\ Z1\_abs^2}{(N-1)\ (N+1)\ A^3\ N} \\
& -\frac{Z1\ (3\ A^3\ N^3-7\ N^2\ A^2-5\ A\ N+3)\ Z2\_abs^2}{(N-1)\ (N+1)\ A^3\ N} \\
& -\frac{Z1\ (3\ A^3\ N^3-7\ N^2\ A^2-5\ A\ N+3)\ Z3\_abs^2}{(N-1)\ (N+1)\ A^3\ N} \\
& -\frac{(2\ B\_1\ T2c\ T3c\ +\ B\_2\ A\ Z1\ e)\ e}{(N-1)(N+1)(1+A\ N)A}
\end{aligned}
$$

`SC3_temp11` is the amplitude equation about the amplitude Z_1 with respect to the unscaled arguments (T_0, R_0). We find that the last term of `SC3_temp11` still has an existing ϵ (`e` in the code), which can be eliminated by substituting the order-2 solvability condition `SC2_Case_1` and the bifurcation parameter expansion Eq. (F.2). This process was done manually in MAPLE. Finally, we have the Brusselator amplitude equation for the hexagonal mode

$$
hexAMP := \frac{\partial}{\partial T_0} Z1 = \mu Z1 + v\ Z2c\ Z3c - g\ Z1\_abs^2\ Z1 - h\ (Z2\_abs^2 + Z3\_abs^2)\ Z1
$$

which reads equivalently as:

$$
\frac{\partial}{\partial t} Z_1 = \mu\ Z_1 + v\ Z_2^*\ Z_3^* - g\ |Z_1|^2\ Z_1 - h\ (|Z_2|^2 + |Z_3|^2)\ Z_1
$$

where

$$
\mu = (B - B_c)/B_c \qquad v = \frac{2}{A}\frac{1 - A\eta}{1 + A\eta} + \frac{2}{A}\mu
$$

$$
g = \frac{38A\eta + 5(A\eta)^2 - 8 - 8(A\eta)^3}{9A^3\eta(1 + A\eta)} \quad h = \frac{5A\eta + 7(A\eta)^2 - 3 - 3(A\eta)^3}{A^3\eta(1 + A\eta)}, \quad \eta = \sqrt{D_X/D_Y}
$$

This result is consistent with the work by Peña *et al.* [92, 93], and Verdasca *et al.* [133]. These literatures also provide further analysis of the amplitude equations.

Appendix G

Derivation of Brusselator amplitude equations for the Turing–Hopf mode using `Amp_solving`

The Turing–Hopf (TH) bifurcation for the Brusselator model can be triggered by the simultaneous tuning of the two bifurcation parameters B and $\sigma = D_X/D_Y$. Critical settings

$$B \equiv B_0 = 1 + A^2$$
$$\sigma \equiv \sigma_0 = \frac{(-1 + \sqrt{A^2+1})^2}{A^2}$$

lead the system to a codimension-2 point (CTHP), where the real part of the dominant eigenvalue touches zero twice, respectively at a zero wavenumber $q = 0$ at eigenvalue $\lambda_c^{\mathrm{H}} = 0 + i\omega_c$ and at a nonzero wavenumber $q_c = \sqrt{A}/\sigma_0^{1/4}$ with eigenvalue $\lambda_c^{\mathrm{T}} = 0 + i0$.

The critical right and left eigenvectors associated with the Hopf eigenvalue λ_c^{H} are

$$\mathbf{R_H} = \begin{bmatrix} 1 \\ \dfrac{i-A}{A} \end{bmatrix}, \ \mathbf{L_H} = \begin{bmatrix} 1 & \dfrac{A(A+i)}{1+A^2} \end{bmatrix}, \ \ i = \sqrt{-1}$$

The critical right and left eigenvectors associated with the Turing eigenvalue λ_c^{T} are

$$\mathbf{R_T} = \begin{bmatrix} 1 \\ \dfrac{-\sqrt{\sigma_0}}{A(1+A\sqrt{\sigma_0})} \end{bmatrix}, \ \mathbf{L_T} = \begin{bmatrix} 1 & \dfrac{A\sqrt{\sigma_0}}{(1+A\sqrt{\sigma_0})} \end{bmatrix}$$

For a codimension-2 bifurcation, we need to expand both bifurcation parameters simultaneously:

$$B = B_0 + \epsilon B_1 + \epsilon^2 B_2 + \dots$$
$$\sigma = \sigma_0 + \epsilon\sigma_1 + \epsilon^2\sigma_2 + \dots$$

Now, we will utilise `Amp_solving` to derive the order-3 amplitude equation at a CTHP. We first initialise the algorithm:

```
> # Initialisation
> e_order := 3: # Expansion order
> sys_dim := 2: # System dimensions
> vas := op(op~([seq([R[m],T[m]], m = 0..e_order)])); # Variable arguments
```

$$vas := R_0,\ T_0,\ R_1,\ T_1,\ R_2,\ T_2,\ R_3,\ T_3$$

then define the Brusselator model (σ_0 is represented by `No` in the code)

```
> # Define the Brusselator model
> ## Activator
> PDE[1] := Dt(uo1) = A - (Bo + 1) * uo1 + uo1^2 * uo2 + No * (Ds@@2)(uo1);
> A := ()->A: # define the constant as a procedure
>
> ## Inhibitor
> PDE[2] := Dt(uo2) = Bo * uo1 - uo1^2 * uo2 + (Ds@@2)(uo2);
```

`Amp_solving` starts the expansion by introducing a linear perturbation:

```
> # Remove temporal and spatial derivatives
> for i from 1 to sys_dim do
>     PDE_ss[i] := 0 = expand(subs(Ds = 0, rhs(PDE[i])));
> od:
> # Automatically solve homogeneous steady-states and store them in ss
> ss := solve( seq(PDE_ss[i], i=1..sys_dim),
               seq(uo||i, i=1..sys_dim) );
>
> # Introduce linear perturbations
> for i from 1 to sys_dim do
      uo||i      := u||i||_||0 + du||i;  # Linear perturbation
      u||i||_||0 := rhs(ss[i]);          # Substitute steady-states
  od:
```

followed by expanding differential operators

```
> Dt := sum('e^m*D[2*m+2]', 'm'=0..e_order): # Expand temporal derivative
> Ds := sum('e^m*D[2*m+1]', 'm'=0..e_order): # Expand spatial derivative
> e := ()->e:  # Define scaling constant
```

and model variables perturbations and bifurcation parameters,

```
> for i from 1 to sys_dim do                    # Nonlinear variable expansion
      du||i :=  sum('e^j*u||i||j', 'j'=1..e_order);
 od:
> Bo := sum('e^i*B_||i', 'i'=0..e_order);      # Bifurcation parameter expansion
> No := sum('e^i*N_||i', 'i'=0..e_order);
>
> # Define expanded bifurcation parameters as constants
> for i from 0 to e_order do
      B_||i := subs([variables=(), body=B_||i], (variables-> body)):
      N_||i := subs([variables=(), body=N_||i], (variables-> body)):
 od:
```

The order equations can be obtained by the following codes:

```
> for i from 1 to sys_dim do
>    alias(seq(u||i||_||j = u||i||j(vas), j=1..3));
> od:
>
> for i from 1 to sys_dim do
>    PDE_sorted[i] := simplify(collect(PDE[i] ,e), e^(e_order+1)=0):
>    PDE_diff[i]   := convert(PDE_sorted[i](vas), diff):
> od:
>
> # Collect terms with the same order to construct order equations
> for i from 1 to sys_dim do
    for j from 0 to e_order do # ith equation at order j
       eq||i||_order||j :=
          expand(coeff(lhs(PDE_diff[i]), e,j)=coeff(rhs(PDE_diff[i]), e,j));
    od:
> od:
>
> # Tidy up each order equation
> for i from 1 to sys_dim do
>    for j from 1 to e_order do

     PDE_temp||i||j := lhs(eq||i||_order||j) - rhs(eq||i||_order||j);
     PDE_components||i||j :=
            selectremove(has,PDE_temp||i||j, seq(u||ii||j, ii=1..sys_dim) );

     eq||i||_order||j||_diff := PDE_components||i||j[1] = -PDE_components||i||j[2];
    od:
> od:
>
> # Vectorize rhs of order equations
> for j from 1 to e_order do
     eq_vec_order||j :=
               Vector[column]([seq(rhs(eq||i||_order||j||_diff), i=1..sys_dim)]);
> od:
```

The critical bifurcation conditions are calculated in an external worksheet `LSA_TH.mw` then loaded to the main program (critical frequency ω_c is represented by `w` in the code):

```
> read "Brusselator_THsets.txt":         # read LSA results, created by LSA_TH.mw
> Tsets := {
      seq(RH||i    = R_H(i), i=1..sys_dim),   # Hopf right eigenvectors
      seq(RH||i||c = R_Hc(i), i=1..sys_dim),  # Hopf right eigenvectors (c.c.)
      seq(RT||i    = R_T(i), i=1..sys_dim),   # Turing right eigenvectors
      seq(LH||i    = L_H(i), i=1..sys_dim),   # Hopf left eigenvectors
      seq(LH||i||c = L_Hc(i), i=1..sys_dim),  # Hopf left eigenvectors (c.c.)
      seq(LT||i    = L_T(i), i=1..sys_dim),   # Turing left eigenvectors
      B_0 = Bc_TH,                            # critical bifurcation setting
      q0  = qc_T                              # critical wavenumber
          };
> THsets_extra := w = A, N_0 = sc_HT;
```

The structure ansatz is defined according to Eq. (4.33):

```
> aas := op(op~([seq([R[j],T[j]], j=1..e_order)]));
> RT  := Vector[column]([seq(RT||i, i=1..sys_dim)]):   # Turing right eigenvector
> LT  := Vector[column]([seq(LT||i, i=1..sys_dim)]):   # Turing left eigenvector
```

```
> RH  := Vector[column]([seq(RH||i, i=1..sys_dim)]):   # Hopf right eigenvector
> RHc := Vector[column]([seq(RH||i||c, i=1..sys_dim)]):# Hopf right eigenvector (c.c.)
> LH  := Vector[column]([seq(LH||i, i=1..sys_dim)]):   # Hopf left eigenvector
> LHc := Vector[column]([seq(LH||i||c, i=1..sys_dim)]):# Hopf left eigenvector (c.c)
> alias(AT=A_T(aas),   ATc=A_Tc(aas)):   # Turing amplitude
> alias(AH=A_H(aas),   AHc=A_Hc(aas)):   # Hopf amplitude
> ans1 := AT*exp(I*q0*R[0])*RT + ATc*exp(-I*q0*R[0])*RT +
          AH*exp(I*w*T[0])*RH  + AHc*exp(-I*w*T[0])*RHc;
```

Following Eqs. (4.39a) and (4.39b), the TH mode solvability condition has separated Turing and Hopf components:

```
> # Solvability condition - Turing
> SC2T := VectorCalculus[DotProduct](LT,eq_vec_order2):
> SC2T_temp1   := PDETools[dsubs]([seq(u||i||1(vas)=ans1[i], i=1..sys_dim)], SC2T):
> SC2T_temp2   := simplify(expand(SC2T_temp1)):
> SC2T_temp3   := coeff(SC2T_temp2, exp(I*q0*R[0])):
> SC2T_temp3cc := coeff(SC2T_temp2, exp(-I*q0*R[0])):
> SC2T_temp4   := collect(simplify(SC2T_temp3), B_1, diff, 'distributed'):
> SC2T_temp4cc := collect(simplify(SC2T_temp3cc), B_1, diff, 'distributed'):
> SC2T_temp5   := collect(simplify(subs(subs(THsets, SC2T_temp4))), B_1, diff, AT):
> SC2T_temp5cc := collect(simplify(subs(subs(THsets, SC2T_temp4cc))), B_1, diff, AT):
> SC2T_temp6   := denom(simplify(SC2T_temp5)) * simplify(SC2T_temp5) = 0:
> SC2T_temp6cc := denom(simplify(SC2T_temp5cc)) * simplify(SC2T_temp5cc) = 0:
> SC2T_Case    := collect(SC2T_temp6, B_1, N_1, diff, AT):
> SC2T_Casecc  := collect(SC2T_temp6cc, B_1, N_1, diff, ATc):
```

```
> # Solvability condition - Hopf
> SC2H := VectorCalculus[DotProduct](LHc,eq_vec_order2):
> SC2H_temp1   := PDETools[dsubs]([seq(u||i||1(vas)=ans1[i], i=1..sys_dim)], SC2H):
> SC2H_temp2   := simplify(expand(SC2H_temp1)):
> SC2H_temp3   := coeff(SC2H_temp2, exp(I*w*T[0])):
> SC2H_temp3cc := coeff(SC2H_temp2, exp(-I*w*T[0])):
> SC2H_temp4   := collect(simplify(SC2H_temp3), B_1, diff, 'distributed'):
> SC2H_temp4cc := collect(simplify(SC2H_temp3cc), B_1, diff, 'distributed'):
> SC2H_temp5   := collect( subs(THsets, SC2H_temp4), B_1, diff, AH):
> SC2H_temp5cc := collect(subs(THsets, SC2H_temp4cc), B_1, diff, AHc):
> SC2_temp6    := denom(simplify(SC2T_temp5)) * simplify(SC2T_temp5) = 0:
> SC2H_temp6cc := denom(simplify(SC2T_temp5cc)) * simplify(SC2T_temp5cc) = 0:
> SC2H_Case    := collect(SC2H_temp6, B_1, N_1, diff, AH):
> SC2H_Casecc  := collect(SC2H_temp6cc, B_1, N_1, diff, AHc):
```

`Amp_solving` yields the order-2 solvability condition:

```
> SC2T_Case;
```

$$AT\ B\_1\ \sqrt{N\_0} \;+\; (-A \;-\; A^2\sqrt{N\_0})\ AT\ N\_1 \;+\; (N\_0^{3/2} \;+\; N\_0^2\ A \;-\; \sqrt{N\_0} \;-\; N\_0\ A)\ \frac{\partial}{\partial T_1} AT \;=\; 0$$

```
> SC2H_Case;
```

$$AH\ B\_1 \;+\; I\ AH\ B\_1\ A \;-\; 2\frac{\partial}{\partial T_1} AH \;-\; 2I\ A\ \frac{\partial}{\partial T_1}\ AH \;=\; 0$$

The order-2 solvability condition gives rise to following constraints:

$$\frac{\partial}{\partial T_1} A_{\mathrm{T}} = \frac{\partial}{\partial T_1} A^*_{\mathrm{T}} = \frac{\partial}{\partial T_1} A_{\mathrm{H}} = \frac{\partial}{\partial T_1} A^*_{\mathrm{H}} = 0$$
$$B_1 = 0,\ \sigma_1 = 0$$

which are defined in `Amp_solving` as:

```
> aas := R[1], R[2], T[2], R[3], T[3]:
> B_1 := 0:
> N_1 := 0:
```

Using the order-2 solvability condition, we can solve the order-2 equation.

```
> for j from 1 to sys_dim do
    eq||j||_order2_withAnsatz_temp :=
    PDETools[dsubs]([seq(u||i||1(vas)=ans1[i], i=1..sys_dim)], eq||j||_order2_diff):

    eq||j||_order2_withAnsatz :=
             simplify(expand(eq||j||_order2_withAnsatz_temp), exp):
> od:
>
> exp_fun := op(select(has, indets(eq2_order2_withAnsatz), exp));
>
> # Determine the order-2 structure
> cas := aas;
> for i from 1 to sys_dim do
    seq( alias(c||i||_||j=c||i||j(cas)),  j=1..nops([exp_fun])+1 );
> od:
> for i from 1 to sys_dim do
    ans2[i] := sum('c||i||_||j * exp_fun[j]', 'j'=1..nops([exp_fun]))
                                          + c||i||_||(nops([exp_fun])+1):
> od:
>
> # substitution
> for i from 1 to sys_dim do
     eq||i||_lhs :=
       collect( PDETools[dsubs]([seq(u||i||2(vas)=ans2[i], i=1..sys_dim)],
                                    lhs(eq||i||_order2_withAnsatz)),exp):
     eq||i||_order2_subs:= eq||i||_lhs = rhs(eq||i||_order2_withAnsatz):
> od:
>
> # Construct coefficient equations - exponential pattern match
> for i from 1 to sys_dim do
    for j from 1 to nops([exp_fun]) do

       eq||i||_coef||j :=
            coeff( lhs(eq||i||_order2_subs),  exp_fun[j]  ) =
            coeff( rhs(eq||i||_order2_subs),  exp_fun[j]  );
    od:

       eq||i||_coef||(nops([exp_fun])+1) :=
            remove(has, lhs(eq||i||_order2_subs),
                   seq(exp_fun[j], j=1..nops([exp_fun])) ) =
            remove(has, rhs(eq||i||_order2_subs),
                   seq(exp_fun[j], j=1..nops([exp_fun])) );
> od:
```

```
>
> # symbolic solutions of coefficients
> unassign(N_0, A): assume(A>0, N_0>0): interface(showassumed=0):
> eq_list_temp  :=  [seq(seq(eq||i||_coef||j, i=1..sys_dim), j=1..nops([exp_fun])+1 )]:
> eq_list := simplify(subs(THsets, eq_list_temp)):
> coef_list := [seq(seq(c||i||_||j, i=1..sys_dim ), j=1..nops([exp_fun])+1 )]:
> coeff_order2_temp := expand(solve(eq_list, coef_list)):
> coeff_order2 := simplify(coeff_order2_temp, AH*AHc =AH_abs, AT*ATc = AT_abs):
```

Finally, apply the solvability condition Eqs. (4.39a) and (4.39b) to the order-3 equation to derive the order-3 amplitude equations for the Turing and Hopf components.

```
> # Solvability condition: Turing amplitude equation
> SC3T := VectorCalculus[DotProduct](LT, eq_vec_order3_subs):
> SC3T_temp1 := simplify(expand(SC3T), exp):
> SC3T_temp2 := simplify(subs(THsets, coeff(SC3T_temp1, exp(I*q0*R[0])))) = 0:
> SC3T_temp3 := collect(simplify(subs(THsets_extra, SC3T_temp2)), diff,factor):
> AT_coeff   := PDEtools[dcoeffs](lhs(SC3T_temp3), AT)[1]:
> SC3T_temp4 := collect(expand(SC3T_temp3/expand(AT_coeff)), diff,factor):
> time_diffT := selectfun(SC3T_temp4, diff)[2]: # time derivative
> SC3T_temp5 := time_diffT = -selectremove(has, lhs(SC3T_temp4),time_diffT)[2]:
> SC3T_temp6 := simplify(SC3T_temp5, AH*AHc =AH_abs, AT*ATc =AT_abs):
> SC3T_temp7 := collect(SC3T_temp6, AT_abs, AH_abs, diff, B_2, factor):
>
> # Return arguments to normal scales
> nas := R[0], T[0]; # normal arguments
> alias(To = T_o(nas)):
> SC3T_temp8 := diff(To, T[0])
               = subs( diff(AT, R[1], R[1]) = diff(To, R[0], R[0]), AT = To,
                       AT_abs = T_abs, AH_abs = H_abs,rhs(SC3T_temp7)):
> SC3T := collect(SC3T_temp8, T_abs, H_abs, diff, factor):
```

```
> # Solvability condition: Hopf amplitude equation
> SC3H := VectorCalculus[DotProduct](LHc, eq_vec_order3_subs):
> SC3H_temp1 := simplify(expand(SC3H), exp):
> SC3H_temp2 := simplify(subs(THsets, coeff(SC3H_temp1, exp(I*w*T[0])))) = 0:
> SC3H_temp3 := collect(simplify(subs(THsets_extra, SC3H_temp2)), diff,factor):
> AH_coeff   := PDEtools[dcoeffs](lhs(SC3H_temp3), AH)[1]:
> SC3H_temp4 := collect(expand(SC3H_temp3/expand(AH_coeff)), diff,factor):
> time_diffH := selectfun(SC3H_temp4, diff)[2]: # time derivative
> SC3H_temp5 := time_diffH = -selectremove(has, lhs(SC3H_temp4),time_diffH)[2]:
> SC3H_temp6 := simplify(SC3H_temp5, AH*AHc =AH_abs, AT*ATc =AT_abs):
> SC3H_temp7 := collect(SC3H_temp6, AT_abs, AH_abs, diff, B_2, factor):
>
> # Return arguments to normal scales
> nas := R[0], T[0]; # normal arguments
> alias(Ho = H_o(nas)):
> SC3H_temp8 := diff(Ho, T[0])
              = subs( diff(AH, R[1], R[1]) = diff(Ho, R[0], R[0]), AH = Ho,
                      AT_abs = T_abs, AH_abs = H_abs, rhs(SC3H_temp7)):
> SC3H := collect(SC3H_temp8, T_abs, H_abs, diff, factor):
```

`Amp_solving` derives the Turing mode amplitude equation `SC3T`

$$\frac{\partial}{\partial t}T = C_{\mathrm{T}}T - C_{\mathrm{TT}}|T|^2T - C_{\mathrm{TH}}|H|^2T + D_{\mathrm{T}}\nabla^2T \tag{G.1}$$

where

$$C_{\mathrm{T}} = \frac{1}{1+\sigma_0}\,(B-B_0) - \frac{2}{(-1+\sigma_0)^2}\,(\sigma-\sigma_0), \quad C_{\mathrm{TT}} = \frac{1}{18}\,\frac{39\;\sigma_0^2 - 25\;\sigma_0 + 2}{\sigma_0^2\;(-1+\sigma_0)},$$
$$C_{\mathrm{TH}} = \frac{-2\;\sigma_0 - 1 + 2\;\sigma_0^2 + 13\;\sigma_0^3 + 3\;\sigma_0^4 + \sigma_0^5}{\sigma_0\;(3\;\sigma_0 + 1 + \sigma_0^2)\;(-1+\sigma_0^2)}, \quad D_{\mathrm{T}} = \frac{4\sigma_0}{1+\sigma_0} \tag{G.2}$$

and the Hopf mode amplitude equation `SC3H`:

$$\frac{\partial}{\partial t}H = C_{\mathrm{H}}H - (C_{\mathrm{HHr}} + iC_{\mathrm{HHi}})|H|^2H - (C_{\mathrm{HTr}} + iC_{\mathrm{HTi}})|T|^2H + (D_{\mathrm{Hr}} + iD_{\mathrm{Hi}})\nabla^2H \tag{G.3}$$

where

$$C_{\mathrm{H}} = \frac{1}{2}\,(B-B_0), \qquad C_{\mathrm{HHr}} = \frac{1}{2}\,\frac{2+A^2}{A^2}, \qquad C_{\mathrm{HHi}} = \frac{1}{6}\,\frac{4\;A^4 - 7\;A^2 + 4}{A^3},$$
$$C_{\mathrm{HTr}} = \frac{1}{2}\,\frac{(-7\;\sigma_0 - 17\;\sigma_0^2 + 1 + 25\;\sigma_0^3 + 14\;\sigma_0^4)}{(3\;\sigma_0 + 1 + \sigma_0^2)\;\sigma_0\;(-1+\sigma_0)}, \qquad D_{\mathrm{Hr}} = \frac{1}{2} + \frac{1}{2}\,\sigma_0, \tag{G.4}$$
$$C_{\mathrm{HTi}} = -\frac{1}{2}\,\frac{1 - 5\;\sigma_0 - 9\;\sigma_0^2 + 38\;\sigma_0^3 + 22\;\sigma_0^4 + \sigma_0^5}{\sigma_0^{3/2}\;(3\;\sigma_0 + 1 + \sigma_0^2)\;(-1+\sigma_0)}, \quad D_{\mathrm{Hi}} = \sqrt{\sigma_0}$$

C_{TH} in Eq. (G.1) and $C_{\mathrm{HTr}} + iC_{\mathrm{HTi}}$ in Eq. (G.3) are cross terms indicating the interacting Turing and Hopf instabilities. Removing cross terms, Eq. (G.1) becomes the amplitude equation for the stripes mode (i.e., Eq. (4.24)), and Eq. (G.3) becomes the amplitude equation for the pure Hopf mode. `Amp_solving` derived TH mode amplitude equations are consistent with those derived by de Wit [21].

Appendix H

Symbol notations in `Amp_solving`

Table H.1: Mathematical symbols and their expressions used in `Amp_solving`

Mathematical symbol	`Amp_solving` expression	Comment
$\partial/\partial t$	`Dt()`	partial derivative in time
∇	`Ds()`	partial derivative in space
∇^2	`Ds@@2()`	second spatial derivative (Laplacian)
X, Y	`uo1, uo2`	Brusselator variables
X_0, Y_0	`u1_0, u2_0`	Brusselator steady-states
η	`N`	$\eta = \sqrt{D_X/D_Y}$
σ	`No`	$\sigma = D_X/D_Y$
B	`Bo`	bifurcation parameter
$\delta x, \delta y$	`du1, du2`	linear perturbations
x_1, x_2, x_3	`u11, u12, u13`	expanded variables for δx
y_1, y_2, y_3	`u21, u22, u23`	expanded variables for δy
B_0, B_1, B_2	`B_0, B_1, B_2`	expanded B
$\sigma_0, \sigma_1, \sigma_2$	`N_0, N_1, N_2`	expanded σ
ϵ	`e`	scaling constant
$\mathbf{I_j}$	`eq_vec_orderj`	`j` $= 1, 2, 3$
$\mathbf{R_T}$	`RT`	Turing critical right eigenvectors
$\mathbf{L_T}$	`LT`	Turing critical left eigenvectors
$\mathbf{R_H}$	`RH`	Hopf critical right eigenvectors
$\mathbf{R_H}^*$	`RHc`	Hopf critical right eigenvectors (c.c.)
$\mathbf{L_H}$	`LH`	Hopf critical left eigenvectors
$\mathbf{L_H}^*$	`LHc`	Hopf critical left eigenvectors (c.c.)
q_c	`q0`	Turing critical wavenumber
ω	`w`	Hopf critical frequency
A_{T1}, A_{T2}, A_{T3}	`AT1`, `AT2`, `AT3`	hexagonal amplitudes
A^*_{T1}, A^*_{T2}, A^*_{T3}	`AT1c`, `AT2c`, `AT3c`	hexagonal amplitudes (c.c.)
$Z_1,\ Z_2,\ Z_3$	`Z1`, `Z2`, `Z3`	$Z_1 = \epsilon A_{T1}$ $Z_2 = \epsilon A_{T2}$, $Z_3 = \epsilon A_{T3}$
$\vec{q_1}$, $\vec{q_2}$, $\vec{q_3}$	`q1`, `q2`, `q3`	hexagonal wavevectors
A_T, A_H	`AT`, `AH`	Turing, Hopf amplitudes for the TH mode
A^*_T, A^*_H	`ATc`, `AHc`	Turing, Hopf amplitudes for the TH mode (c.c.)
c_{ij}	`ci_j`	coefficients in the order-2 solution
$\vert\ \vert$	`_abs`	modulus of a complex number
i	`I`	$\sqrt{-1}$

References

[1] Achermann, P., Borbely, A.: Coherence analysis of the human sleep electroencephalogram. Neuroscience **85**(4), 1195–1208 (1998)

[2] Amor, F., Baillet, S., Navarro, V., Adam, C., Martinerie, J., Quyen, M.L.V.: Cortical local and long-range synchronization interplay in human absence seizure initiation. NeuroImage **45**(3), 950–962 (2009)

[3] Ananthanarayanan, R., Modha, D.S.: Anatomy of a cortical simulator. In: Proceedings of the 2007 ACM/IEEE Conference on Supercomputing-SC '07 (2007)

[4] Aranson, I.S., Kramer, L.: The world of the complex Ginzburg-Landau equation. Reviews of Modern Physics **74**(1), 99–143 (2002)

[5] Beck, P., Odle, A., Wallace-Huitt, T., Skinner, R.D., Garcia-Rill, E.: Modafinil increases arousal determined by p13 potential amplitude: an effect blocked by gap junction antagonists. Sleep **31**(12), 1647 (2008)

[6] Benardo, L.S.: Gap junctions in epileptogenesis: chicken or egg? Epilepsy Currents **4**(2), 80–81 (2004)

[7] Bose, S., Rodin, P., Scholl, E.: Competing spatial and temporal instabilities in a globally coupled bistable semiconductor system near a codimension-two bifurcation. Physical Review E **62**(2), 1778–1789 (2000)

[8] Cantero, J.L., Atienza, M., Salas, R.M., Dominguez-Marin, E.: Effects of prolonged waking-auditory stimulation on electroencephalogram synchronization and cortical coherence during subsequent slow-wave sleep. Journal of Neuroscience **22**(11), 4702–4708 (2002)

[9] Carmon, T.T., Buljan, H.H., Segev, M.M.: Spontaneous pattern formation in a cavity with incoherent light. Optics Express **12**(15), 3481–3487 (2004)

[10] Cartwright, M.L.: Balthazar van der Pol. Journal of the London Mathematical Society. Second Series **35**, 367–376 (1960)

[11] Chávez, M., Le Van Quyen, M., Navarro, V., Baulac, M., Martinerie, J.: Spatio-temporal dynamics prior to neocortical seizures: amplitude versus phase couplings. IEEE Transactions on Bio-medical Engineering **50**(5), 571–583 (2003)

[12] Chisholm, R.A., Filotas, E.: Critical slowing down as an indicator of transitions in two-species models. Journal of Theoretical Biology **257**(1), 8–8 (2009)

[13] Cooper, L.N., Scofield, C.: Mean-field theory of a neural network. Proceedings of the National Academy of Sciences of the United States of America **85**(6), 1973 (1988)

[14] Crawford, J.: Introduction to bifurcation theory. Reviews of Modern Physics **63**(4), 991–1037 (1991)

[15] Cross, M., Greenside, H.: Pattern formation and dynamics in nonequilibrium systems. Cambridge University Press, London (2009)

[16] Damoiseaux, J.S., Rombouts, S.A.R.B., Barkhof, F., Scheltens, P., Stam, C.J., Smith, S.M., Beckmann, C.F.: Consistent resting-state networks across healthy subjects. PNAS **103**(37), 13848–13853 (2006)

[17] Dauwels, J., Vialatte, F., Cichocki, A.: Diagnosis of Alzheimers disease from EEG signals: Where are we standing? Current Alzheimer Research **7**(6), 487–505 (2010)

[18] Davey, M.P., Victor, J.D., Schiff, N.D.: Power spectra and coherence in the EEG of a vegetative patient with severe asymmetric brain damage. Clinical Neurophysiology **111**(11), 1949–1954 (2000)

[19] Davies, P.A., Hanna, M.C., Hales, T.G., Kirkness, E.F.: Insensitivity to anaesthetic agents conferred by a class of $GABA_A$ receptor subunit. Nature **385**(6619), 820–823 (1997)

[20] de Garis, H., Shuo, C., Goertzel, B., Ruiting, L.: A world survey of artificial brain projects, Part I: Large-scale brain simulations. Neurocomputing **74**(1-3), 27–27 (2010)

[21] de Wit, A.: Brisure de symétrie spatiale et dynamique spatio-temporelle dans les systèmes réaction-diffusion. Ph.D. thesis, Université Libre de Bruxelles (1993)

[22] de Wit, A.: Spatial patterns and spatiotemporal dynamics in chemical systems. Advances in Chemical Physics **109**, 435–514 (1999)

[23] de Wit, A., Dewel, G., Borckmans, P.: Chaotic Turing-Hopf mixed mode. Physics Review E **48**(6), 4191 (1993)

[24] de Wit, A., Lima, D., Dewel, G., Borckmans, P.: Spatiotemporal dynamics near a codimension-two point. Physical Review E **54**, 261 (1996)

[25] Dudek, F.E.: Gap junctions and fast oscillations: a role in seizures and epileptogenesis? Epilepsy Currents **2**(4), 133 (2002)

[26] Dumermuth, G., Lehmann, D.: EEG power and coherence during non-REM and REM phases in humans in all-night sleep analyses. European Neurology **20**(6), 429–434 (1981)

[27] Eliasmith, C., Stewart, T.C., Choo, X., Bekolay, T., DeWolf, T., Tang, Y., Rasmussen, D.: A large-scale model of the functioning brain. Science **338**(6111), 1202–1205 (2012)

[28] Ermentrout, B.: Neural networks as spatio-temporal pattern-forming systems. Reports on Progress in Physics **61**(4), 353–430 (1998)

[29] Field, R.: Oregonator (2007), URL `http://www.scholarpedia.org/article/Oregonator`

[30] Fox, M.D.M., Snyder, A.Z.A., Vincent, J.L.J., Corbetta, M.M., Van Essen, D.C.D., Raichle, M.E.M.: The human brain is intrinsically organized into dynamic, anticorrelated functional networks. PNAS **102**(27), 9673–9678 (2005)

[31] Franks, N., Lieb, W.: Molecular and cellular mechanisms of general-anesthesia. Nature **367**(6464), 607–614 (1994)

[32] Fransson, P.: Spontaneous low-frequency BOLD signal fluctuations: An fMRI investigation of the resting-state default mode of brain function hypothesis. Human Brain Mapping **26**(1), 15–29 (2005)

[33] Gigout, S., Louvel, J., Pumain, R.: Effects in vitro and in vivo of a gap junction blocker on epileptiform activities in a genetic model of absence epilepsy. Epilepsy Research **69**(1), 15–29 (2006)

[34] Gollub, J., Langer, J.: Pattern formation in nonequilibrium physics. Reviews of Modern Physics **71**(2), S396–S403 (1999)

[35] Greicius, M.D.M., Krasnow, B.B., Reiss, A.L.A., Menon, V.V.: Functional connectivity in the resting brain: a network analysis of the default mode hypothesis. PNAS **100**(1), 253–258 (2003)

[36] Gui-Quan, S., Zhen, J., Quan-Xing, L., Li, L.: Pattern formation induced by cross-diffusion in a predator–prey system. Chinese Physics B **17**, 3936–3941 (2008)

[37] Gunaratne, G., Ouyang, Q., Swinney, H.: Pattern formation in the presence of symmetries. Physical Review E **50**(4), 2802–2820 (1994)

[38] Gusnard, D.A., Raichle, M.E.: Searching for a baseline: Functional imaging and the resting human brain. Nature Reviews Neuroscience **2**(10), 685–694 (2001)

[39] Hampson, E.C.E., Vaney, D.I.D., Weiler, R.R.: Dopaminergic modulation of gap junction permeability between amacrine cells in mammalian retina. Journal of Neuroscience **12**(12), 4911–4922 (1992)

[40] Haragus, M., Iooss, G.: Local bifurcations, center manifolds, and normal forms in infinite-dimensional dynamical systems. Springer (2010)

[41] Hasselmo, M.E.: Neuromodulation and cortical function: modeling the physiological basis of behavior. Behavioural Brain Research **67**(1), 1–27 (1995)

[42] Hertz, J., Lerchner, A., Ahmadi, M.: Mean field methods for cortical network dynamics, vol. 3146. Computational Neuroscience: Cortical Dynamics, Berlin, Heidelberg (2004)

[43] Holmes, M.D., Brown, M., Tucker, D.M.: Are "generalized" seizures truly generalized? Evidence of localized mesial frontal and frontopolar discharges in absence. Epilepsia **45**(12), 1568–1579 (2004)

[44] Horváth, J., Szalai, I., De Kepper, P.: An experimental design method leading to chemical Turing patterns. Science **324**(5928), 772–775 (2009)

[45] Huang, N.E., Shen, Z., Long, S.R., Wu, M.C., Shih, H.H., Zheng, Q., Yen, N.C., Tung, C.C., Liu, H.H.: The empirical mode decomposition and the Hilbert spectrum for nonlinear and non-stationary time series analysis. Proceedings of the Royal Society A **454**(1971), 903–995 (1998)

[46] Hutt, A., Frank, T.: Critical fluctuations and 1/f(alpha)-activity of neural fields involving transmission delays. Acta Physica Polonica A **108**(6), 1021–1040 (2005)

[47] Imas, O.A., Ropella, K.M., Wood, J.D., Hudetz, A.G.: Isoflurane disrupts anterio-posterior phase synchronization of flash-induced field potentials in the rat. Neuroscience letters **402**(3), 216–221 (2006)

[48] Ipsen, M., Hynne, F., Sørensen, P.G.: Amplitude equations for reaction-diffusion systems with a Hopf bifurcation and slow real modes. Physica D: Nonlinear Phenomena **136**, 66 (2000)

[49] Ipsen, M., Kramer, L., Sorensen, P.: Amplitude equations for description of chemical reaction–diffusion systems. Physics Reports **337**(1-2), 193–235 (2000)

[50] Izhikevich, E.M., Edelman, G.M.: Large-scale model of mammalian thalamocortical systems. PNAS **105**(9), 3593–3598 (2008)

[51] Jacobson, G.M., Voss, L.J., Melin, S.M., Cursons, R.T.M., Sleigh, J.W.: The role of connexin36 gap junctions in modulating the hypnotic effects of isoflurane and propofol in mice. Anaesthesia **66**(5), 361–367 (2011)

[52] Jahromi, S.S., Wentlandt, K., Piran, S., Carlen, P.L.: Anticonvulsant actions of gap junctional blockers in an in vitro seizure model. Journal of Neurophysiology **88**(4), 1893–1902 (2002)

[53] Johansson, J.S.: Central nervous system electrical synapses as likely targets for intravenous general anesthetics. Anesthesia and Analgesia **102**(6), 1689–1691 (2006)

[54] Johnson, B.W., Sleigh, J.W., Kirk, I.J., Williams, M.L.: High-density EEG mapping during general anaesthesia with xenon and propofol: a pilot study. Anaesthesia and Intensive Care **31**(2), 155–163 (2003)

[55] Khanin, R., Cartmell, M., Gilbert, A.: A computerised implementation of the multiple scales perturbation method using Mathematica. Computers & Structures **76**(5), 565–575 (2000)

[56] Kidachi, H.: On mode interactions in reaction diffusion equation with nearly degenerate bifurcations. Progress of Theoretical Physics **63**(4), 1152–1169 (1980)

[57] Kirschfeld, K.: The physical basis of alpha waves in the electroencephalogram and the origin of the "Berger effect". Biological Cybernetics **92**(3), 177–185 (2005)

[58] Kitamura, A., Marszalec, W., Yeh, J.Z., Narahashi, T.: Effects of halothane and propofol on excitatory and inhibitory synaptic transmission in rat cortical neurons. Journal of Pharmacology and Experimental Therapeutics **304**(1), 162–171 (2002)

[59] Klepel, K., Blömker, D., Mohammed, W.W.: Amplitude equation for the generalized Swift-Hohenberg equation with noise. Zeitschrift für angewandte Mathematik und Physik p. 98 (2013)

[60] Koskinen, M., Seppänen, T., Tuukkanen, J., Yli-Hankala, A., Jäntti, V.: Propofol anesthesia induces phase synchronization changes in EEG. Clinical Neurophysiology **112**(2), 386–392 (2001)

[61] Kurth, S., Achermann, P., Rusterholz, T., LeBourgeois, M.: Development of brain EEG connectivity across early childhood: does sleep play a role? Brain Sciences **3**(4), 1445–1460 (2013)

[62] Kuznetsov, E.A., Nepomnyashchy, A.A., Pismen, L.M.: New amplitude equation for Boussinesq convection and nonequilateral hexagonal patterns. Physics Letters A **205**, 261–265 (1995)

[63] Laufs, H.H., Kleinschmidt, A.A., Beyerle, A.A., Eger, E.E., Salek-Haddadi, A.A., Preibisch, C.C., Krakow, K.K.: EEG-correlated fMRI of human alpha activity. NeuroImage **19**(4), 14–14 (2003)

[64] Lee, S.G., Neiman, A., Kim, S.: Coherence resonance in a Hodgkin-Huxley neuron. Physical Review E **57**(3), 3292–3297 (1998)

[65] Liley, D.T.J., Cadusch, P.J., Dafilis, M.P.: A spatially continuous mean field theory of electrocortical activity. Network-Computation in Neural Systems **14**(2), 369–369 (2002)

[66] Liley, D.T., Cadusch, P.J., Wright, J.J.: A continuum theory of electro-cortical activity. Neurocomputing **26**, 795–800 (1999)

[67] Lo, M.T., Tsai, P.H., Lin, P.F., Lin, C., Hsin, Y.L.: The nonlinear and nonstationary properties in EEG signals: probing the complex fluctuations by Hilbert–Huang transform. Advances in Adaptive Data Analysis **1**(03), 461–482 (2009)

[68] Lopes da Silva, F., Blanes, W., Kalitzin, S.N., Parra, J., Suffczynski, P., Velis, D.N.: Epilepsies as dynamical diseases of brain systems: basic models of the transition between normal and epileptic activity. Epilepsia **44 Suppl 12**, 72–83 (2003)

[69] Luongo, A., Di Egidio, A.: Bifurcation equations through multiple-scales analysis for a continuous model of a planar beam. Nonlinear Dynamics **41**(1-3), 171–190 (2005)

[70] Luongo, A.: On the use of the multiple scales method in solving difficult bifurcation problems. In: 4th Canadian Conference on Nonlinear Solid Mechanics, Montreal, QC, Canada

[71] Luongo, A., D'Annibale, F.: Linear stability analysis of multiparameter dynamical systems via a numerical-perturbation approach. AIAA Journal **49**, 2047–2056 (2011)

[72] Markram, H.: The blue brain project. Nature Reviews Neuroscience **7**(2), 153–160 (2006)

[73] Marshall, L., Mölle, M., Born, J.: Spindle and slow wave rhythms at slow wave sleep transitions are linked to strong shifts in the cortical direct current potential. Neuroscience **121**(4), 1047–1053 (2003)

[74] Mayol, C., Toral, R., Mirasso, C.: Derivation of amplitude equations for nonlinear oscillators subject to arbitrary forcing. Physical Review E **69**(6), 066141 (2004)

[75] Meixner, M., de Wit, A., Bose, S., Scholl, E.: Generic spatiotemporal dynamics near codimension-two Turing-Hopf bifurcations. Physical Review E **55**(6), 6690–6697 (1997)

[76] Minlebaev, M., Ben-Ari, Y., Khazipov, R.: Network mechanisms of spindle-burst oscillations in the neonatal rat barrel cortex in vivo. Journal of Neurophysiology **97**(1), 692–700 (2007)

[77] Mölle, M., Marshall, L., Gais, S., Born, J.: Learning increases human electroencephalographic coherence during subsequent slow sleep oscillations. PNAS **101**(38), 13963–13968 (2004)

[78] Morikawa, T.T., Hayashi, M.M., Hori, T.T.: Auto power and coherence analysis of delta-theta band EEG during the waking-sleeping transition period. Electroencephalography and Clinical Neurophysiology **103**(6), 633–641 (1997)

[79] Mormann, F., Kreuz, T., Andrzejak, R.G., David, P., Lehnertz, K., Elger, C.E.: Epileptic seizures are preceded by a decrease in synchronization. Epilepsy Research **53**(3), 173–185 (2003)

[80] Mormann, F., Lehnertz, K., David, P., E Elger, C.: Mean phase coherence as a measure for phase synchronization and its application to the EEG of epilepsy patients. Physica D: Nonlinear Phenomena **144**(3-4), 358–369 (2000)

[81] Nicola, E.M.: Interfaces between competing patterns in reaction-diffusion systems with nonlocal coupling. Ph.D. thesis, Technische Universität Dresden (2001)

[82] Nicolis, G., Prigogine, I.: Self-organization in nonequilibrium systems. from dissipative structures to order through fluctuations, Wiley, New York (1977)

[83] Nilsen, K.E., Kelso, A.R.C., Cock, H.R.: Antiepileptic effect of gap-junction blockers in a rat model of refractory focal cortical epilepsy. Epilepsia **47**(7), 1169–1175 (2006)

[84] Nunez, P.L.: The brain wave equation: a model for the EEG. Mathematical Biosciences **21**(3-4), 279–297 (1974)

[85] Nunez, P.L.: Wavelike properties of the alpha rhythm. IEEE Transactions on Bio-medical Engineering **BME-21**(6), 473–482 (1974)

[86] Nunez, P.L.: Neocortical dynamics and human EEG rhythms. Oxford University Press, USA (1995)

[87] Nunez, P.L., Srinivasan, R.: Electric fields of the brain. The Neurophysics of EEG, Oxford University Press, USA (2006)

[88] Olympio, M.A.: Postanesthetic delirium: historical perspectives. Journal of Clinical Anesthesia **3**(1), 60–63 (1991)

[89] Ouyang, G., Li, X., Dang, C., Richards, D.A.: Using recurrence plot for determinism analysis of EEG recordings in genetic absence epilepsy rats. Clincal Neurophysiology **119**(8), 1747–1755 (2008)

[90] Oweis, R.J., Abdulhay, E.W.: Seizure classification in EEG signals utilizing Hilbert-Huang transform. BioMedical Engineering OnLine **10**(1), 38 (2011)

[91] Ozer, M.N., Eser, D.: Derivation of separable amplitude equations by multiple scales method. Turkish Journal of Mathematics **22**(3), 309–317 (1998)

[92] Pena, B., Perez-Garcia, C.: Stability of Turing patterns in the Brusselator model. Physical Review E **64**, 056213–056213 (2001)

[93] Pena, B., Perez-Garcia, C.: Selection and competition of Turing patterns. Europhysics Letters **51**(3), 300–306 (2007)

[94] Peng, R., Wang, M.X.: Pattern formation in the Brusselator system. Journal of Mathematical Analysis and Applications **309**(1), 16–16 (2005)

[95] Perez-Velazquez, J.L., Valiante, T.A., Carlen, P.L.: Modulation of gap junctional mechanisms during calcium-free induced field burst activity: a possible role for electrotonic coupling in epileptogenesis. Journal of Neuroscience **14**(7), 4308–4317 (1994)

[96] Pismen, L.M., Rubinstein, B.Y.: Computer tools for bifurcation analysis: general approach with application to dynamical and distributed systems. International Journal of Bifurcation and Chaos **9**(6), 983–1008 (1999)

[97] Pismen, L., Rubinstein, B., Velarde, M.: On automated derivation of amplitude equations in nonlinear bifurcation problems. International Journal of Bifurcation and Chaos **6**(11), 2163–2167 (1996)

[98] Pogarell, O.: EEG coherence reflects regional corpus callosum area in Alzheimer's disease. Journal of Neurology, Neurosurgery & Psychiatry **76**(1), 109–111 (2005)

[99] Raichle, M.E., MacLeod, A.M., Snyder, A.Z., Powers, W.J., Gusnard, D.A., Shulman, G.L.: A default mode of brain function. PNAS **98**(2), 676–682 (2001)

[100] Rennie, C.J., Wright, J.J., Robinson, P.A.: Mechanisms of cortical electrical activity and emergence of gamma rhythm. Journal of Theoretical Biology **205**(1), 17–35 (2000)

[101] Robinson, P.A., Rennie, C.J., Wright, J.J.: Propagation and stability of waves of electrical activity in the cerebral cortex. Physical Review E **56**(1), 826–840 (1997)

[102] Robinson, P.A., Rennie, C.J., Wright, J.J., Bourke, P.D.: Steady states and global dynamics of electrical activity in the cerebral cortex. Physical Review E **58**(3), 3557–3571 (1998)

[103] Rodriguez, E.E., George, N.N., Lachaux, J.P.J., Martinerie, J.J., Renault, B.B., Varela, F.J.F.: Perception's shadow: long-distance synchronization of human brain activity. Nature **397**(6718), 430–433 (1999)

[104] Rovinsky, A., Menzinger, M.: Interaction of Turing and Hopf bifurcations in chemical-systems. Physical Review A **46**(10), 6315–6322 (1992)

[105] Sarnthein, J., Rappelsberger, P., Petsche, H., Rauscher, F.H., Shaw, G.L.: Persistent patterns of brain activity: an EEG coherence study of the positive effect of music on spatial-temporal reasoning. Neurological Research **19**(2), 107–116 (1997)

[106] Singer, W., Gray, C.M.: Visual feature integration and the temporal correlation hypothesis. Annual Review of Neuroscience (1995)

[107] Singer, W.: Binding by synchrony. Scholarpedia **2**(12), 1657 (2007)

[108] Sleigh, J.W., Voss, L., Steyn-Ross, M.L., Steyn-Ross, D.A., Wilson, M.T.: Sleep and Anesthesia. Springer New York, New York, NY (2011)

[109] Smith, S.J.M.: EEG in the diagnosis, classification, and management of patients with epilepsy. Journal of Neurology, Neurosurgery & Psychiatry **76**(suppl 2), ii2–ii7 (2005)

[110] Stafstrom, C.E.: Closing the gap: electrotonic junctions in seizure control. Epilepsy currents / American Epilepsy Society **4**(5), 174–175 (2004)

[111] Steyn-Ross, D.A., Steyn-Ross, M.L.: Modeling Phase Transitions in the Brain. Springer Verlag (2010)

[112] Steyn-Ross, D.A., Steyn-Ross, M.L., Sleigh, J.W., Wilson, M.T., Gillies, I.P., Wright, J.J.: The sleep cycle modelled as a cortical phase transition. Journal of Biological Physics **31**, 547–569 (2005)

[113] Steyn-Ross, D.A., Steyn-Ross, M.L., Wilcocks, L.C., Sleigh, J.W.: Toward a theory of the general-anesthetic-induced phase transition of the cerebral cortex. II. Numerical simulations, spectral entropy, and correlation times. Physical Review E **64**, 011918–011918 (2001)

[114] Steyn-Ross, M.L., Steyn-Ross, D.A., Sleigh, J.W., Liley, D.T.: Theoretical electroencephalogram stationary spectrum for a white-noise-driven cortex: evidence for a general anesthetic-induced phase transition. Physical Review E **60**, 7299–7311 (1999)

[115] Steyn-Ross, M.L., Steyn-Ross, D.A., Sleigh, J.W., Wilcocks, L.C.: Toward a theory of the general-anesthetic-induced phase transition of the cerebral cortex. I. A thermodynamics analogy. Physical Review E **64**(1), 011917 (2001)

[116] Steyn-Ross, M.L., Steyn-Ross, D.A., Sleigh, J.W.: Interacting Turing-Hopf instabilities drive symmetry-breaking transitions in a mean-field model of the cortex: a mechanism for the slow oscillation. Physical Review X **3**(2), 021005 (2013)

[117] Steyn-Ross, M.L., Steyn-Ross, D.A., Sleigh, J.W., Wilson, M.T.: A mechanism for ultra-slow oscillations in the cortical default network. Bulletin of Mathematical Biology **73**(2), 398–416 (2011)

[118] Steyn-Ross, M.L., Steyn-Ross, D.A., Wilson, M.T., Sleigh, J.W.: Gap junctions mediate large-scale Turing structures in a mean-field cortex driven by subcortical noise. Physics Review E **76**, 011916–011916 (2007)

[119] Steyn-Ross, M.L., Steyn-Ross, D.A., Wilson, M.T., Sleigh, J.W.: Interacting Turing and Hopf instabilities drive pattern formation in a noise-driven model cortex. Advances in Cognitive Neurodynamics ICCN 2007 pp. 225–230 (2007)

[120] Steyn-Ross, M.L., Steyn-Ross, D.A., Wilson, M.T., Sleigh, J.W.: Modeling brain activation patterns for the default and cognitive states. NeuroImage **45**(2), 298–311 (2009)

[121] Steyn-Ross, M.L., Steyn-Ross, D.A., Sleigh, J.W.: Gap junctions modulate seizures in a mean-field model of general anesthesia for the cortex. Cognitive Neurodynamics **6**(3), 215–225 (2012)

[122] Stroh, A., Adelsberger, H., Groh, A., Rühlmann, C., Fischer, S., Schierloh, A., Deisseroth, K., Konnerth, A.: Making waves: initiation and propagation of corticothalamic Ca^{2+} waves in vivo. Neuron **77**(6), 1136–1150 (2013)

[123] Sweeney-Reed, C.M., Nasuto, S.J.: A novel approach to the detection of synchronisation in EEG based on empirical mode decomposition. Journal of Computational Neuroscience **23**(1), 79–111 (2007)

[124] Tao, H.Y., Tian, X.: Coherence characteristics of gamma-band EEG during rest and cognitive task in MCI and AD. In: 2005 IEEE Engineering in Medicine and Biology 27th Annual Conference, pp. 2747–2750, IEEE (2005)

[125] Tass, P., Rosenblum, M.G., Weule, J., Kurths, J., Pikovsky, A., Volkmann, J., Schnitzler, A., Freund, H.J.: Detection of n:m phase locking from noisy data: application to magnetoencephalography. Physical Review Letters **81**(1), 3291–3294 (1998)

[126] Timofeev, I., Bazhenov, M., Seigneur, J., Sejnowski, T.: Jasper's basic mechanisms of the epilepsies. Oxford University Press

[127] Turing, A.M.: The chemical basis of morphogenesis. Philosophical Transactions of the Royal Society of London. Series B **237**(6), 37–72 (1952)

[128] Uhlhaas, P., Singer, W.: Neural synchrony in brain disorders: Relevance for cognitive dysfunctions and pathophysiology. Neuron **52**(1), 155–168 (2006)

[129] Uhlhaas, P.J.P., Linden, D.E.J.D., Singer, W.W., Haenschel, C.C., Lindner, M.M., Maurer, K.K., Rodriguez, E.E.: Dysfunctional long-range coordination of neural activity during Gestalt perception in schizophrenia. Journal of Neuroscience **26**(31), 8168–8175 (2006)

[130] Varela, F., Lachaux, J.P., Rodriguez, E., Martinerie, J.: The brainweb: phase synchronization and large-scale integration. Nature Reviews Neuroscience **2**(4), 229–239 (2001)

[131] Vazquez, J., Baghdoyan, H.A.: Basal forebrain acetylcholine release during REM sleep is significantly greater than during waking. American Journal of Physiology, Regulatory, Integrative and Comparative Physiology **280**(2), R598–601 (2001)

[132] Venance, L.L., Prémont, J.J., Glowinski, J.J., Giaume, C.C.: Gap junctional communication and pharmacological heterogeneity in astrocytes cultured from the rat striatum. Journal of Physiology **510**(Pt 2), 429–440 (1998)

[133] Verdasca, J., de Wit, A., Dewel, G., Borckmans, P.: Reentrant hexagonal Turing structures. Physics Letters A **168**(3), 194–198 (1992)

[134] Volman, V., Perc, M., Bazhenov, M., Sporns, O.: Gap junctions and epileptic seizures – two sides of the same coin? PLoS ONE **6**(5), 0572– (2011)

[135] Voss, L.J., Mutsaerts, N., Sleigh, J.W.: Connexin36 gap junction blockade is ineffective at reducing seizure-like event activity in neocortical mouse slices. Epilepsy Research and Treatment **2010**, 1–6 (2010)

[136] Wackermann, J.J.: Towards a quantitative characterisation of functional states of the brain: from the non-linear methodology to the global linear description. International Journal of Psychophysiology **34**(1), 65–80 (1999)

[137] Walgraef, D.: Spatio-temporal pattern formation: with examples from physics, chemistry, and materials science. Springer Verlag (2012)

[138] Wang, K., Steyn-Ross, M.L., Steyn-Ross, D.A., Wilson, M.T., Sleigh, J.W., Shiraishi, Y.: Simulations of pattern dynamics for reaction-diffusion systems via SIMULINK. BMC Systems Biology **8**(1), 45 (2014)

[139] Wang, K., Steyn-Ross, M.L., Steyn-Ross, D.A., Sleigh, J.W., Wilson, M.: Interacting Turing and Hopf instabilities in a mean-field model of the cerebral cortex. In: 2009 New Zealand Postgraduate Conference, pp. 1–1, Victoria University, Wellington, New Zealand (2009)

[140] Wang, K., Ye, M.: Parameter determination of Schottky-barrier diode model using differential evolution . Solid State Electronics **53**(2), 234–240 (2009)

[141] Wang, K., Ye, M.: Parameter estimation of Schottky-barrier diode model by particle swarm optimization. International Journal of Modern Physics C **20**(5), 687–699 (2009)

[142] Wentlandt, K., Samoilova, M., Carlen, P.L., El Beheiry, H.: General anesthetics inhibit gap junction communication in cultured organotypic hippocampal slices. Anesthesia and Analgesia **102**(6), 1692–1698 (2006)

[143] Wilson, H., Cowan, J.: Excitatory and inhibitory interactions in localized populations of model neurons. Biophysical Journal **12**, 1–24 (1972)

[144] Wilson, H.R.: Spikes, Decisions, and Actions. The Dynamical Foundations of Neuroscience, Oxford University Press, USA (1999)

[145] Wilson, M.T., Steyn-Ross, D.A., Sleigh, J.W., Steyn-Ross, M.L., Wilcocks, L.C., Gillies, I.P.: The K-complex and slow oscillation in terms of a mean-field cortical model. Journal of Computational Neuroscience **21**(3), 243–257 (2006)

[146] Wright, J.J.: EEG simulation: variation of spectral envelope, pulse synchrony and approximately 40 Hz oscillation. Biological Cybernetics **76**(3), 181–194 (1997)

[147] Wright, J.J.: Simulation of EEG: dynamic changes in synaptic efficacy, cerebral rhythms, and dissipative and generative activity in cortex. Biological Cybernetics **81**(2), 131–147 (1999)

[148] Wright, J.J., Liley, D.: Dynamics of the brain at global and microscopic scales: Neural networks and the EEG. Behavioral and Brain Sciences **19**, 285–320 (1996)

[149] Yamapi, R., Filatrella, G., Aziz-Alaoui, M.A.: Global stability analysis of birhythmicity in a self-sustained oscillator. Chaos **20**(1), 013114–013114 (2010)

[150] Yang, L., Dolnik, M., Zhabotinsky, A., Epstein, I.: Oscillatory clusters in a model of the photosensitive Belousov-Zhabotinsky reaction system with global feedback. Physical Review E **62**(5), 6414–6420 (2000)

[151] Yang, L., Dolnik, M., Zhabotinsky, A., Epstein, I.: Pattern formation arising from interactions between Turing and wave instabilities. Journal of Chemical Physics **117**(15), 7259–7265 (2002)

[152] Yang, L., Epstein, I.: Oscillatory turing patterns in reaction-diffusion systems with two coupled layers. Physical Review Letters **90**(17), 178303 (2003)

[153] Yang, L., Zhabotinsky, A., Epstein, I.: Stable squares and other oscillatory Turing patterns in a reaction-diffusion model. Physical Review Letters **92**(19), 198303 (2004)

[154] Yang, L., Dolnik, M., Zhabotinsky, A., Epstein, I.: Spatial resonances and superposition patterns in a reaction-diffusion model with interacting Turing modes. Physical Review Letters **88**(20), 208303 (2002)

[155] Yi, L., Fan, Y.L., Li, G., Tong, Q.Y.: Sleep stage classification based on EEG Hilbert-Huang transform. In: 2009 4th IEEE Conference on Industrial Electronics and Applications (ICIEA), pp. 3676–3681, IEEE (2009)

[156] Yu, P.: Computation of normal forms via a perturbation technique. Journal of Sound and Vibration **211**(1), 19–38 (1998)

[157] Yu, P.: Symbolic computation of normal forms for resonant double Hopf bifurcations using a perturbation technique. Journal of Sound and Vibration **247**(4), 615–632 (2001)

[158] Yu, P.: Analysis on double Hopf bifurcation using computer algebra with the aid of multiple scales. Nonlinear Dynamics **27**(1), 19–53 (2002)

[159] Zamani Aghaie, K., Shahabadi, M.: Multiple-scale analysis of plane wave refraction at a dielectric slab with Kerr-type nonlinearity. Progress in Electromagnetics Research **56**, 81–92 (2006)

[160] Zeman, P.: Phase preservation: ABVS EEG artifact removal algorithm vs. ICA standard. URL `http://artifactremoval.abvsciences.com/preserved-phase-new-eeg-artifact-removal-algorithm-vs-ica-standard/`

[161] Zhang, X.Z., Yin, L., Wang, W.X.: Wavelet time-frequency analysis of electroencephalogram (EEG) processing. International Journal of Advanced Computer Science and Application **1**, 1–5 (2010)