Modeling complex cell regulation in the zebrafish circadian clock

Raphaela Heussen

Thesis for PhD

Centre for Mathematics and Physics in the Life Sciences and EXperimental Biology (CoMPLEX)

University College London

I, Raphaela Heussen, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the thesis.

Raphaela Heußen

Abstract

The interdisciplinary "systems biology" approach of combining traditional biological investigations with tools from the mathematical and computer sciences has enabled novel insights into many highly complex and dynamic biological systems. The use of models has, for instance, revealed much about the intricate feedback mechanisms and acute importance of gene regulatory networks, and one such network of special note is our internal time keeper, or circadian clock. The circadian clock plays a pivotal role in modulating critical physiological processes, and has also been implicated, either directly or indirectly, in a whole range of pathological states.

This research project investigates how the underlying dynamics of the circadian clock in the zebrafish model organism may be captured by a mathematical model, considering in particular the entrainment effect due to external cues such as light. Simulated data is contrasted with experimental results from different light regime experiments to validate the model and guide its refinement. Furthermore, various statistical methods are implemented to process the raw data and support its analysis. Extending the initial deterministic approach to take into account stochastic effects and additive population level effects emerges as a powerful means of representing the circadian signal decay in prolonged darkness, as well as light initiated re-synchronization as a strong component of entrainment. Consequently, it emerges that stochastic effects may be considered an essential feature of the circadian clock in zebrafish. A further cornerstone of the project is the implementation of an integrated simulation environment, including a Sequential Monte Carlo parameter estimation function, which succeeds in predicting a range of previously determined and also novel suitable parameter values. However, considerable difficulties in obtaining parameter values that satisfy the entire range of important target values simultaneously highlights the inherent complexity of accurately simulating the circadian clock.

Table of Contents

Abstract 3

Table of Contents 4

List of Tables 6

List of Figures 6

Introduction 7

Chapter 1: Background to Modelling the Circadian Clock 12

1.1 Biological Basis of the Circadian Clock 12

1.1.1 The Role of GRNs in Regulating Biological Processes 12

1.1.2 Genes and their Transcription 18

1.1.3 The Significance of the Circadian Clock 22

1.2 Ordinary Differential Equations Based Models........ 30

1.2.1 Describing a System by Fluctuating Concentrations 30

1.2.2 Major Dynamics in Representing GRNs as ODEs 37

1.3 Accounting for Stochastic Variation 50

1.3.1 Biological systems are noisy 50

1.3.2 Stochastic Modeling Approaches 54

1.4 Modeling the Circadian Clock 62

1.4.1 The Circadian Clock as a Major Regulator 62

1.4.2 Concepts Relevant to Circadian Modeling 64

1.4.3 Circadian Models across Species 71

1.4.4 The Circadian Clock in Zebrafish 82

Aims and Objectives 92

Chapter 2: Creating the Core Model and Analytical Tools 94

2.1 Building an Initial Model of the Zebrafish Circadian Clock 94

2.1.1 The Underlying System of ODEs 94

2.1.2 The Significance of Fine-Tuning the Model 99

2.2 Laboratory Experiments 102

2.2.1 The Use of Bioluminescence Reporter Genes 102

2.2.2 Overview of Different Experiments Conducted 105

2.2.3 Denoising and Detrending 107

2.2.4 Results and Discussion 111

2.3 Analyzing the Prepared Data 115
2.3.1 Time-frequency Analysis by Hilbert Transform 115
2.3.2 Period Calculations........ 121
2.3.3 Wavelet Transform 123
2.3.4 Oscillator Quantification 129
2.4 Analyzing and Capturing Underlying Dynamics 131
2.4.1 Biological Findings to be Integrated into the Model 131
2.4.2 Analyzing Underlying Model Dynamics 134
2.4.3 Further Extensions to the Model 141
3. Materials and Methods........ 142
3.1 Experimental Conditions........ 142
3.2 Data Processing and Model Simulation 143
4. Summary of Findings........ 144
4.1 Parameter Estimation via Sequential Monte Carlo 144
4.2 Sensitivity to Parameter Changes Across Different Models 146
4.3 Deterministic and Stochastic Behaviour 151
5. Discussion........ 155
5.1 An Integrating Workflow Can Provide Novel Insights........ 155
5.2 Basic Model Structure Replicates A Limited Set Responses 157
5.3 Stochastic Behaviour as an Essential Circadian Feature 157
5.4 Feeding Stochastic Predictions Into Parameter Estimation........ 159
5.5 Future Outlook - Embracing Stochasticity 160
Bibliography 163
Appendix 176
Code for the Zebrafish Model in Mathematica 176
Code for the Hilbert Transform 177
Code for the Wavelet Transform 181
Code for the Integrated Simulation Environment 192
Code for the Sequential Monte Carlo Algorithm 210

List of Tables

Table 1 A comparison of modelling and analysis techniques for high-throughput data.......31

List of Figures

Figure 1 Examples of Feed-Forward Loops 14
Figure 2 Interlocked feedback loop model of the *Neurospora* circadian clock. 74
Figure 3 Drosophila Model Network diagram 76
Figure 4 Plant Model Network diagram..78
Figure 5 Mammalian Model Network diagram..80
Figure 6 Zebrafish Model Network diagram 85
Figure 7 Proposed GRN for zebrafish circadian clock. 89
Figure 8 Zebrafish Model Network diagram corresponding to the model 95
Figure 7 Parameters..98
Figure 10 Initial Model runs with and without light input..98
Figure 11 Detrending 111
Figure 12 15-min light pulse 113
Figure 13 1hr light pulse 114
Figure 14 Light pulse data analysis 115
Figure 15 Dampened Sine Wave..119
Figure 16 Sample Sensitivity Analysis Output..138
Figure 17 Sample Bifurcation Analysis output..138
Figure 18 Sample PRC Output..140
Figure 19 Mean and STD of smc results for base model compared to expected values.....145
Figure 20 Distribution of smc results for base model..146
Figure 21 Sensitivity analysis for base model 148
Figure 22 Sensitivity analysis for Extended model 148
Figure 23 Bifurcation for Different parameters...149
Figure 24 Bifurcation for the same parameter, but measured using different variables....150
Figure 25 Phase Response Curve for Base model...151
Figure 26 Phase Response Curve for extended model 151
Figure 27 Deterministic versus Stochastic curves...153
Figure 28 Stochastic simulation exposed to light pulses 154

Introduction

The Rise of Systems Biology

Systems biology is an interdisciplinary approach that draws heavily on mathematical and computational modelling techniques in order to further understanding of obscure mechanisms and complex interactions in biology. The main aim is to identify, analyze, and predict trends and underlying dynamics in biological processes, with a special focus on viewing systems in a pluralistic and holistic way, as well as on discovering instances of emergence. Emergent properties are those qualities of a system that cannot be reduced to the individual components of this system when considered separately, but rather they depend on the complex interrelations of the various constituents. In this way, systems biology has previously been able to shed light on convoluted and highly dynamic settings, such as enzyme kinetics or neurophysiological signalling cascades, which tend to defy elucidation through classic reductionist approaches. Of course, the use of bioinformatics also plays a prominent role in the field, and with the recent rise to prominence of big data analysis, data mining, and related approaches, it appears likely that this trend is bound to open up fascinating new research insights.

As one major avenue of investigation that already sees widespread use, models can be constructed to capture the dynamics of interactions in complex systems, and subsequently the simulated data thusly generated may be compared and contrasted with the available experimental data. In this way modeling does not only improve the theoretical grasp of a system, revealing for example if any postulated relationships are suitable inputs for generating plausible solutions, but it can also provide practical cues to guide experimental studies and make them faster, cheaper and more effective. Namely, gaps in the existing knowledge can be more readily pointed out, sensible starting points for lab research determined, and unexpected predictions pin pointed as the target for future investigation. Once a model has been found to adequately depict the essential behaviours under scrutiny, it can of course also provide significant advantages by bringing to bear the immense speed advantage of present day numerical solvers and hardware. In this way it may be possible, for example, to run in the space of mere minutes or hours, dozens of simulated trials over varying substrate concentrations or environmental conditions, which may otherwise require weeks or months of laboratory work. While simulations may admittedly never become a full substitute for traditional bench work, we are nevertheless likely only beginning to see their potential as powerful compliments.

Gene Regulatory Networks

A central target for systems biology investigations are gene regulatory networks (GRNs). These collections of DNA segments are present in all cells, where they express certain key RNA and, indirectly, proteins, and interact via these products with one another or other substances in the cell. The specific protein products may range from structural cell membrane constituents over enzymes, i.e. molecular catalysts that drive important biochemical reactions, to transcription factors, which play a major role in modulating regulatory cascades by binding to the promoter regions of other genes. Throughout all these and other modes of action, GRNs are highly involved in controlling and coordinating most critical functions at the cellular level - housekeeping intracellular processes, reactions to external stimuli, as well as developmental and adaptational switches. The precise mechanisms are often multifaceted and highly complicated, as GRNs can describe various processes on multiple levels, such as transcriptional control, RNA processing, transport and degradation, translation, and protein activity and degradation. In consequence, it is not rare to discover additional, hereto unknown functions of even well studied GRNs, and our understanding is frequently updated and challenged by novel insights into the basis of regulation at the cellular level. For example, the important role played by chromatin modification on re-folding DNA and subsequently allowing or negating transcription, a mechanism know as epigenesis, has been repeatedly confirmed, but remains poorly understood in the context of GRNs. Not surprisingly, this level of intricacy poses a considerable challenge to make sense of it all. Thankfully, mathematical techniques can aid immensely with the quantification and qualitative description of data and elucidation of underlying relationships. For example, a schematic view could represent transcription factors as network inputs, genes as nodes of the network, and gene expression levels as the resulting output. Disentangling these varying interconnections plays an indispensable part in building functional models of cellular behaviour.

The Circadian Clock

A GRN concerned with keeping track of earth's rotation around its axis and the accompanying changes in light, temperature and humidity etc. is commonly referred to as a circadian clock. The ability to anticipate recurring environmental fluctuations such as the aforementioned has emerged as an immense selective advantage, allowing organisms to tailor their behaviour and biological processes to expected future opportunities and challenges, and before this background it may not be surprising that most living entities, from human beings to cyanobacteria, make use of such daily time-keeping mechanisms. Moreover, these systems are generally much more multifaceted than simple hourglass timer, featuring for example the ability to adjust to different day light spans, and in doing so can even double as a useful seasonal timer. The importance of time keeping is easily evident on the behavioral level, for example looking to flowers synching their blooming periods to daylight hours, nocturnal rodents sensing when to return to the safety of their burrows in time before dawn, or migratory birds punctually embarking on their yearly journeys across the globe. Many of us are also very familiar with the experience of waking up just a few minutes before the alarm clock, or having a good sense of when our usual meal times come around, and understanding the behavioural influences of the circadian clock is also acutely relevant from a therapeutic angle, for example in the context of chronic sleeping disorders, or the jet lag evoked by modern means of travel.

As fascinating as the interactions of our subconscious routines with our inner time keeper may be, it is also well worth remembering that there exists another, far deeper level to circadian rhythms. In fact, in many organisms the functioning, regulation, and adaptation of circadian time is exclusively effected at the cellular level, including of course all circadian clocks in unicellular organisms; but there are also more complex organisms and even vertebrates, such as zebrafish, where no central pacemaker has been identified and instead many cells and tissues contain autonomous circadian clocks. Even in other species where a central circadian pacemaker has been identified, usually in discrete regions in or close to the brain, such as the optic lobes of Drosophila or the suprachiasmatic nuclei (SCN) in the hypothalamus of mammals, the circadian signal nevertheless may be bolstered by peripheral oscillators and remains closely integrated with cellular processes. It is known, for example, that the critical process of mitosis, or cell division, is strictly timed around certain key checkpoints, such as the transition from G2 to M, G1 to S, and the so-called metaphase checkpoint. Interestingly, even in the absence of environmental stimulation the circadian clock follows an oscillating rhythm termed the free running period. However, in nature the clock is habitually entrained by external inputs to the daily rhythm of exactly 24 hours, and making sense of the particulars of this entrainment remains an active research area. While even species living in constant darkness are known to possess circadian clocks, and an entire variety of potential environmental cues has been identified including feeding and tidal rhythms, the widely most prominent and important regulator remains exposure to sun light.

Zebrafish as a Model Organism

Zebrafish are not only a widely used vertebrate model species in general, but also appear as an especially interesting candidate in the context of studying the circadian clock and light entrainment in particular. One of the main reasons lies in the fact that they exhibit several similarities to mammals in the circadian clock makeup, but no central circadian pacemaker has been found in zebrafish, with timekeeping seemingly effected at a cellular level. This makes it possible to work with populations of zebrafish cell lines while limiting distortions due to interference due to centralized coupling, etc. as would be the case in many other vertebrates. Furthermore, individual cells are known to be very light sensitive and possess a direct light entrainment pathway including photopigments, allowing for a strong entrainment effect due to light exposure, and recent studies also point to many other aspects of cell biology being influenced by light-induced gene expression in zebrafish.

The aim of the project is to construct a model of the GRN underlying the zebrafish circadian clock, and subsequently to adapt and extend this model to elucidate the mechanisms behind entrainment. Firstly, various existing mathematical techniques have to be selected, adapted and combined in innovative ways to support the quantitative and qualitative description of the available data. Utilizing this processed data as a guide alongside well established mathematical modelling techniques, an initial mathematical model of zebrafish circadian clock key molecular components and their respective interactions is constructed. Key behaviours from this stimulatory framework are then compared and contrasted with results obtained from laboratory experiments, in which the effect of various light regimes on populations of zebra fish cells is measured. Bolstered by additional experimental insights and data generated specifically to aid the modeling efforts, the initial deterministic model is supplemented with stochastic approaches to take into account noisy cellular processes and to better capture the dynamics of natural de-synchronization over time and re-synchronisation under the influence of light. A final, and at times surprising, challenge is the identification and implementation of how and where light entrainment should be incorporated into the model. It is hoped that the efforts presented here will help to clarify the functioning and underlying dynamics of the zebrafish circadian clock, while also revealing implications for circadian clocks, light entrainment, and especially the importance of stochastic behaviour in basic cellular processes in general. Moreover, the employed mathematical techniques, and the computationally implemented models and experimental simulations should also posses intrinsic value, as they could be easily adapted as a starting point for further investigations on the zebrafish circadian clock and related GRNs, especially those where noise and entrainment effects play a major role.

Chapter 1: Background to Modelling the Circadian Clock

1.1 Biological Basis of the Circadian Clock

1.1.1 The Role of GRNs in Regulating Biological Processes

It has been shown that the incredible complexity of the human physiology is contained in over 20,000 protein-coding genes, while even the much simpler unicellular yeast organism already relies on over 6,000 genes. However, the majority of these genes are not expressed universally, but rather differentially across a range of specific tissues and cells. In order to ensure a smooth physiological operation, this process of selective expression is tightly controlled at various levels, including transcription, mRNA stability, mRNA translation, and protein stability, and further modulated in response to intracellular or extracellular cues. A cornerstone is the control of mRNA synthesis, which is largely effected by specialized molecules known as transcription factors (TFs), which interact with *cis*-regulatory DNA sequences located within or close to the target genes. Next to changes in the concentrations of TFs, expression levels of genetic downstream targets are also affected by transcriptional cofactors binding to TFs, specialized proteins or microRNAs binding to mRNAs to influence their translation or stability, and there has also been much recent attention paid to the effect of chromatin modification and the mechanism of epigenesis.

These different regulatory functions, such as TFs, are in turn themselves extensively regulated and embedded in complex interactions, such as between multiple other TFs and genes. In order to make sense of the critical functioning of these intricate systems for regulating growth, developmental patterning, and proper stress responses, the specific transcriptional programs and DNA sequences involved are often thought of and visualized as a Gene Regulatory Network (GRN). As an example of a GRN, the expression pattern of Hox genes may provide positional information to the developing *Drosophila* embryo, and alterations to usual expression levels may result in legs in the place of antennae (Casares et al. 1996). Interestingly, these genetic building blocks are often conserved throughout development and across species, but may have markedly different purposes in each case. Ras signalling, for example, is involved in eye development in the fly, but controls vulval development in *C. elegans* and mutations have been linked to cancer in humans (Han 1992). It has been speculated that core networks conserved across species may be adapted to such variable functions under the presence of additional network nodes or variations in the expression levels of existing nodes (Macneil & Walhout 2011). Additionally, many GRNs are also capable of receiving and integrating multiple inputs, in the form of activating or repressing regulatory proteins that recognize specific sequences within them.

Important Network Motifs of GRNs

The complexity of GRNs does not stop there, however, with modularity also frequently playing a role. The target overlap between different TFs has previously been identified via topological overlap coefficient analysis and used to differentiate GRN modules (Vermeirssen et al. 2007), and it has been suggested that this architecture may act to increase TF redundancy of the network, while also insulating compartments against local surges or drops in expression levels. Finally, modularity may also promote differentiated responses to external stimuli and is frequently encountered in networks that react to environmental changes, as has been demonstrated in the case of a *C. elegans* metabolic GRN (Arda et al. 2010). Moreover, studies of yeast transcriptional networks have also found that regulatory mechanisms may be subject to a hierarchical organization (Jothi et al. 2009). TFs could be categorized into distinct groups, with shared properties and similar abundance and noise levels across each group, and it has been suggested that this organization may support adaptive responses, by enabling the same network to show varied responses for different cells within a population.

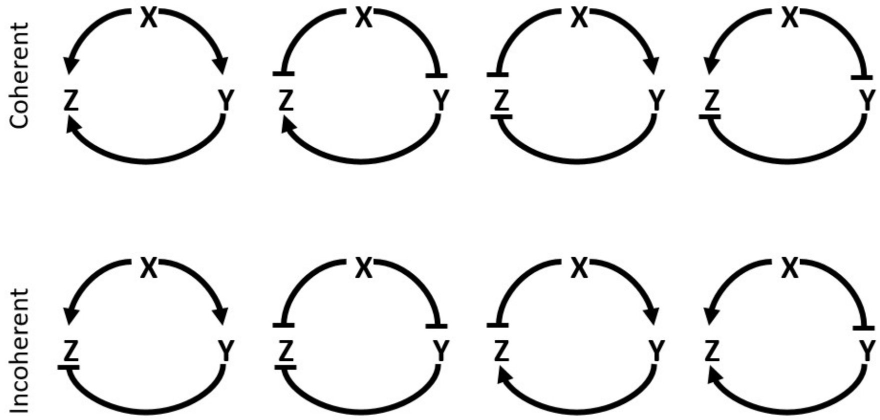


FIGURE 1 Examples of Feed-Forward Loops
The eight different types of Feed-Forward loops (FFL) can be categorized into coherent (the sign of the effect on Z is the same if coming from X directly and indirectly from X over Y) and incoherent (the sign of effect is different – e.g. the direct effect is activating while the indirect one, over Y, is inhibitory and vice versa). The arrow sign denotes activation, whereas the ⊥ sign denotes repression.

Another network motif that is highly relevant and common in the case of GRNs is autoregulation, i.e. a TF increasing or decreasing its own expression via interactions with its promoter. It is noteworthy that positive autoregulation has been found to increase stochasticity (Alon 2007), which can increase the variability of a population, whereas negative autoregulatory loops are more commonly found in settings, where the aim seemingly is to establish or maintain steady-state levels. Another ubiquitous GNR motif are Feed-Forward Loops (FFLs), sets of two regulators affecting a downstream target, while one regulator also regulates the other. Regulatory interaction can be either positive or negative, and either coherent or incoherent, resulting in eight possible FFL permutations. Examples of all these possible FFLs have indeed been described in vivo, but interestingly it was also noted that the most abundant FFLs of the coherent type display very little noise, while the most abundant FFLs of the incoherent type exhibit the highest levels of noise (Ghosh et al. 2005).

Here, the term "coherent" describes a loop, in which the first regulator's indirect, via the interaction between the regulators, and direct effects on the downstream target are of the same quality, i.e. both positive or both negative. In an "incoherent" loop, on the other hand, the regulator may up-regulate the downstream target directly, but its effect on the other regulator would down-regulate the downstream target. However, it is important to note that these opposing effects would not simply cancel each other out, but might often be part of a mechanism to delay or otherwise modulate the timing of a translational response following a stimulating cue. Such a delay could be useful, for instance, to differentiate between sporadic and sustained stimuli, and to only trigger a response for the latter, a mechanisms that may also serve to filter out noise. It was also demonstrated that an incoherent FFL may generate a transient spike in expression levels (Kuttykrishnan et al. 2010). In the case of the yeast *S. cerevisiae*, Rgt1 and Mig2 are part of an incoherent FFL that stimulates expression of the glucose transporter Hxt4 following exposure to glucose. Rgt1 subsequently suppresses both *MIG2* and *HXT4*, and the magnitude of this negative response determines whether a pulse in gene expression can be observed or not. The example also serves to point to an important qualifier, namely that the ability of a network motif to generate a specific response may be modulated by input from elsewhere in the network.

Understanding the Network Topology of GRNs

Technological advances over the last decade or so in high-throughput approaches have enabled the description of a multitude of genes and TFs. Following on from this, modeling and computational analysis have been employed to generate valuable insights into the network topology of GRNs, and to elucidate how system architecture contributes to differential gene expression. Furthermore, understanding these conserved paradigms is also considered useful for tracing the flow of information in biological systems, for understanding fluctuations in expression levels under varying conditions, and for indentifying novel mechanisms in biological systems. In order to graphically represent these interconnections in GNRs, genes and/or their regulators are commonly designated as the nodes of a regulatory system, while their interactions comprise the edges joining these nodes together. The precise nature of those physical interactions is frequently investigated by scanning the genome for regions that may interact with a TF of interest. Alternatively, or rather complementarily, one would seek to identify TFs that may be able to interact with a target DNA sequence of interest. The latter method, also known as "gene-centered", is still less frequently implemented, but as one example the yeast one-hybrid (Y1H) system has been used to study GRNs in *Arabidopsis* (Brady et al. 2011).

However, apart from physical interactions the edges in a GRN may also represent regulatory relationships, for example those inferred by correlating gene expression profiles. This approach offers a powerful tool for deciphering complex GRNs, but may struggle to reveal redundancies or to differentiate between direct and more indirect interactions between TFs and their gene targets. Yet another limitation would be the inability to identify interactions based on post-translational modifications or cofactors, and consequently it appears desirable to integrate data on regulatory and physical relationships in order to construct GRN models. Another quality of GRNs that should be mentioned, is the concept of in-bound and out-bound directionality, which is based on the circumstance that TFs regulate target genes, but are usually not likewise affected. Following on from this, the "in-degree" describes the number of TFs that a given gene can interact with, while the "out-degree" refers to the number of genes that a specific TF can interact with. Many GRN nodes exhibit relatively low in- or out-degrees, but there also occur some very highly connected "hubs" (Yu & Gerstein 2006), whereby the information paths linking through such a note are referred to by the "flux capacity" and calculated as the product of in-degree and out-degree. It has also been noted that the distribution of out-degree follows a power-law distribution, while the distribution of in-degree follows an exponential distribution (Luscombe et al. 2004). Finally, the concept of "between-ness" describes the number of shortest paths that are routed through a specific node, and consequently centrally located and well connected nodes would exhibit a high between-ness.

Robustness as a Key Feature

Interestingly, there appears to exist a considerable level of phenotypic robustness, that is a retention of seemingly normal traits despite changing expression levels, and it has been reported for example that >80% of yeast genes can be individually deleted without lethal effect (Dixon et al. 2009). This observation is very interesting from system-level view and points to a high level of redundancy, so that gene loss may only become apparent in double mutants or under challenging environmental conditions. It has also been noted, regarding the level of gene expression, that essential genes appear to exhibit lower levels of variability, thusly linking gene expression to phenotypic robustness. While not all details of gene expression outputs and redundancy at the GRN are clearly understood, recent research has been able to shed a lot of light on several underlying mechanisms. Firstly, it has been reported for *Drosophila* that next to the established primary enhancers, there also exist shadow enhancers, which are located relatively distant to the transcriptional start site, but appear to demonstrate largely similar expression patterns. In the case of the *Drosophila snail* gene it was further shown that loss of one distal or proximal enhancer lead to no discernible differences for embryos reared at 22°C, but that embryos reared at 30°C showed marked deficits for the knock-out, but not the wild type control group (Perry et al. 2010). It appears that redundancy may be more critical under adverse than under optimal conditions. As a second avenue for bringing about redundancy, multiple TFs from the same family may be able to bin the same *cis*-regulatory DNA element, as has been shown in *C. elegans*, where both FLH-1 and FLH-2 TFs bind the same microRNA promoters. Loss of either TF had little effect on microRNA expression, but double mutants showed a significant increase in larval lethality (Ow et al. 2008). Finally, redundancy can also be furthered, when TFs from different families bind to different *cis*-regulatory elements on a single enhancer, as appears to be the case in *C. elegans* in a GRN centered on the TF SKN-1. A study found *skn-1* mutants to show highly variable expression of the TF END-1, which has in turn be linked to lethal phenotypes only occurring in a proportion of genetically identical organisms (Raj et al. 2010).

The critical network aspect of robustness discussed above may serve as a reminder just how multifaceted GRNs can present themselves. In turn, the inability, due to strong redundancy, of single TF knock-outs to produce observable differences in gene expression may clearly defy reductionist attempts to elucidate their function, making instead a case for a more holistic understanding of the entire embedding system.

1.1.2 Genes and their Transcription

The Central Dogma of Molecular Biology

It was in 1909, not long after Gregor Mendel's groundbreaking work on inheritance patterns in pea plants, that the Danish botanist Wilhelm Johannsen proposed the word "gene" to describe the basic hereditary units of form and function. Several decades later, genetic information was found to be contained in large molecules of Deoxyribonucleic acid (DNA), consisting of sequences of four nitrogen-containing nucleobases - namely adenine, cytosine, guanine, and thymine - as well as a backbone of a phosphate group and the monosaccharide sugar deoxyribose linking individuals nucleotides into long chains. Double strands of DNA with paired nucleotide bases - A to T and C to G - were soon found to be coiled into the famous double helix shape, and one of the originator of this discovery, Francis Crick, further coined the "central dogma of molecular biology", describing the transfer of sequence information between information-carrying biopolymers (Crick Mc 1970). Specifically, individual DNA stands can firstly replicate to form more DNA, as observed in the preparation of cell division; secondly, DNA can be transcribed into shorter molecules of RNA, in eukaryotes mainly pre-mRNA subsequently processed with a 5' cap and a poly-A tail, acting primarily as cellular messengers of genetic information; finally, RNA is translated in cellular factories called ribosome into protein, with each codon - a triplet of nucleobases - signifying a specific amino acid to be integrated into the newly formed protein's polypeptide chain.

The "central dogma" remains the cornerstone of our understanding of how genetic information is expressed and converted into physical and physiological structures, processes, and behaviours. At the same time, however, important exceptions and special cases have been described since its inception to the rule that information cannot be transferred back up the sequence, including for example the ability of retroviruses to convert RNA into new DNA segments through a reverse-transcription process, or the direct replication of RNA, a mechanism found not only in viruses, but also eukaryote RNA silencing. Prions posses the ability to pass information to other proteins by inducing conformational changes, while parasitic intein sequence can even integrate their protein-contained information back into the DNA strand (Wu et al. 1998). More commonly and importantly, it has been revealed that a number of ubiquitous processes not covered by central dogma also affect genetic impression in crucial ways, such as methylation of DNA affecting transcription rates, or the posttranslational modification of amino acid chains, which is a major contributor to protein shapes, functions, and stability.

Molecular Topology of a Gene

Similarly to how our understanding of the central dogma has been challenged and expanded over the years, so has been our grasp of what constitutes a gene. Initially, it was assumed that each gene is represented by a continuous segment on the DNA, an assumption that in fact mostly holds true in single cell prokaryote organisms. In more complex eukaryotes however, a single strand of mRNA is derived from non-contiguous regions on the chromosome through the interactions of numerous regulatory and modifying elements. Most fundamentally one can distinguish between exon and intron sequences, with the former coding for a protein, while the latter have no bearing on the associated amino acid chain. In this context it is also important to point to the concept of the reading frame; since nucleotides are clustered into triplets when guiding specific amino acids into the polypeptide-chain, the reading frame can be thought of as sorting the metaphorical nucleotide "letters" into codon "syllables". The open reading frame (ORF), in turn, refers to a continuous stretch of DNA codons with the potential to encode a protein, often spanning both active exons and intervening introns, and contained between a transcription initiation site and a stop codon.

Upstream of the actual coding sequence exists a promoter region, where the enzyme RNA polymerase attached to the DNA in order start the transcription process. Promoters typically contain a TATA box sequence acting as a docking point, that RNA polymerase can bind to in conjunction with additional nuclear proteins known as basal transcription factors, whereas group of transcription factors called TBP-associated factors, or TAFs, can stabilize the TATA-binding protein complex to prevent it from falling off the promoter.

Usually close by downstream of the promoter lies the transcription initiation site, or cap, which signifies the start of the transcribed sequence and will contribute the 5' end of the RNA strand. Downstream of the cap, by a distance varying widely across genes, the translation initiation site can be found, also known as the start codon. It consists of the ATG sequence, and the distance between transcription and translation initiation sites is referred to as the 5' untranslated region, or also as the leader sequence, and can determine the rate at which the resulting mRNA is translated (Gilbert 2000). The end of the ensuing coding sequence is marked by a translation termination codon - appearing as either TAG, TAA, or TGA - where the ribosome will halt translation and dissociate. Not translated, but still transcribed follows the 3' untranslated region, which contains the AATAAA sequence as a facilitator for polyadenylation. This addition of 200 to 300 adenylate residues on the RNA transcript, known collectively as the poly(A) tail, has important functions in effecting mRNA stability, transport, and initiation of translation.

Lastly, it should be noted that most eukaryote cells contain multiple copies of the same chromosome, and that here genes may be either expressed from multiple alleles, most commonly biallelic expression, or only from one allele, which is termed monoallelic expression. Significantly, the latter is often driven by a stochastic, somatic mechanism known as random monoallelic expression (RME), with different cells throughout the organism expressing different alleles, and recent research has differentiated between both long term and short term periods of monoallelic expression patterns (Reinius & Sandberg 2015). This finding appears especially noteworthy in the case of the Zebrafish, which is known to contain multiple copies of core circadian clock components as explained in further detail below, but specific implications are not clear at this time.

Enhancers, Silencers, and Transcription Factors

In addition to promoters, there are often also other regulatory sequences located on either end or even within a gene, and these so called enhancers can activate the utilization of a promoter. Enhancers are *cis*-linked, meaning they only affect promoters on the same chromosome, but can act over a distance of more than 50 thousand bases, and be located either side of the gene, in an intron, or even as part of the complementary strand of DNA. They are essential for activating transcription in most genes, can differentially regulate specific temporal and local expression patterns, and notably, several enhancer elements can act on the same gene, either individually or in combination, to determine its precise transcription. An important sub-type of enhancers are known as silencers, or more rarely negative enhancers, and can in fact act to inhibit transcription in their target genes, and it is in turn through the stimulation by a so called transcription factor that enhancers, silencers, or sometimes both group in parallel are activated.

Transcription factors (TFs) are specific proteins that regulate the transcription of a particular gene by binding to its associated promoter or enhancer regions, thus activating or repressing its expression. In fact, while evolution is often intuitively associated with changes in a coding transcriptome, it has been pointed out that mutation can also readily act, often much more comprehensively, via changes in regulatory elements such as TFs to drive differentiation in species and tissues (Necsulea & Kaessmann 2014). Most TFs belong to families with a shared structure, particularly with regards to the framework of their DNA-binding sites, although a small in amino acids may be sufficient to alter the specific DNA sequence the factor will bind to. Apart from the DNA-binding domain, TFs also contain a *trans*-activating domain that effects the regulation of a gene's associated promoters and enhancers, often by interacting with proteins that play a role in binding RNA polymerase. Finally, there may also be a protein to protein interaction domain present, via which the TF may itself be modulated by other transcription factors or TAFs.

Chromatin and the Role of Non-Coding RNAs

Apart from the much more complex structure of gene architecture in eukaryotes, a second important difference to prokaryotes consists in the fact that eukaryote DNA is mostly found folded into a complex called chromatin. Apart from the DNA itself, chromatin contains roughly equal amounts of protein, and its basic structure is the nucleosome, where two loops of the

DNA strand, corresponding to circa 140 base pairs, are wrapped around an octamer of histone proteins H2A, H2B, H3, and H4. Nucleosomes are in turn most of the time packed tightly into "solenoid" structures and stabilized by histone H1. This closely spaced default configuration is thought to be naturally repressed, inhibiting the access of transcription factors and RNA polymerase. It has long been argued that interruption of this tight coiling plays a role in tissue-specific gene activation (Weintraub 1984), but research has revealed over the last few decades, how the three dimensional shape of our DNA can also play a pivotal role in genetic regulation, and in fact also epigenetic heredity. Recent studies further report how non-coding segments of RNA act in turn to modify chromatin structure and to affect genome stability, specifically by attracting Argonaute-containing complexes to nascent RNA scaffolds, before effecting the recruitment of DNA methyltransferases and histone. Moreover, apart from acting small RNAs have also been shown to repress translation, to silence transcription via RNA interference, or also to activate transcription (Holoch & Moazed 2015).

1.1.3 The Significance of the Circadian Clock

In evolution's continuous battle for the survival of the fittest, the ability to anticipate earth's rotational movements and the corresponding, recurring environmental fluctuations has emerged as an immense selective advantage, allowing organisms to tailor their behaviour and biological processes to expected future opportunities and challenges. Consequently, it may not be surprising that most living entities, from human beings to cyanobacteria, make use of daily time-keeping mechanisms, also known as circadian clocks. These intricate and highly connected GRNs, however, are much more multifaceted than simple hourglass timers, featuring for example the ability to adjust to different day light spans, or photoperiods, and in doing so can even double as a useful seasonal timer. Traditionally, the importance of time keeping has been most apparent on a behavioural level, for example looking to the way different higher organisms would structure their daily or seasonal behaviour in line with an inner "sense of time"; flowers starting up their photosynthetic machinery just before dawn, migratory birds punctually embarking on their yearly journeys across the globe, or nocturnal rodents sensing when to return to the safety of their burrows in time before dawn. Indeed, SCN-lesioned chipmunks in the wild have been shown to fall to predators more frequently (DeCoursey & Krulas 1998), while in cyanobacteria, strains well adapted to the photoperiod outcompete those strains less well aligned (Ouyang et al. 1998). It has further been shown that Drosophila can anticipate light-dark cycles even in laboratory conditions (Helfrich-Förster 2001), and that a functional circadian system increases their reproductive fitness (Beaver et al. 2002).

However, leaving aside predator and prey relationships, the circadian clock also remains acutely relevant in our highly cultured and streamlined world. Many of us are very familiar with the experience of waking up just a few minutes before the alarm clock, or having a good sense of when our usual meal times come around. It is also on this behavioural level that researches started looking at the circadian clock intently from a therapeutic angle, for example in the context of chronic sleeping disorders, as it is not only known that the circadian clock determines the timing of sleep and sleep-dependent events, such as the nocturnal secretion of prolactin and growth hormone (Czeisler & Klerman 1999), but also that disturbed sleep patterns can significantly impact this endocrine programme and lead to metabolic and mental health imbalances (Knutson et al. 2007). Clock mutant mice were also shown to frequently develop obesity and metabolic syndrome (Turek 2005). Another prominent example would be the jet-lag evoked by modern means of travel exposing us to radically opposed time zones in a matter of hours. While most frequent travellers tend to think of this delayed adaptation of the circadian rhythm to such a radical perturbations as a pesky inconvenience, experiments with mice exposed to chronic jet-lag have also pointed to increased mortality rates (Davidson et al. 2006).

Circadian Clocks at the Molecular Level

As fascinating as the interactions of our inner time keeper with our subconscious and rational routines may be, it is also well worth remembering that there exists another, far deeper level to circadian rhythms. In fact, in many organisms the functioning, regulation, and adaptation of circadian time is effected at the cellular level, including of course all circadian clocks in unicellular organisms; but there are also more complex organisms and even vertebrates, such as zebrafish, where no central pacemaker has been identified and instead many cells and tissues contain autonomous circadian clocks (Whitmore et al. 1998; Tamai et al. 2005). It was also reported that in plants even neighbouring cells exhibit circadian rhythms independent from one another (Thain et al. 2000). In many other species a central circadian pacemaker has been identified in discrete regions in or close to the brain, such as the optic lobes of Drosophila or the suprachiasmatic nuclei (SCN) in the hypothalamus of mammals; but recent research has also made it clear that, in mammals for instance, there exists not only one centrally controlled circadian clock, but additionally self-sustained oscillations in most major organ systems throughout the body (Yamazaki 2000). These are referred to as peripheral oscillators and appear to control local rhythmic events. In fact, the occurrence of two oscillators in parallel has even been reported for unicellular and syncitial organisms, such as the fungus *N. crassa* or the dinoflagellate *Gonyaulax polyedra*.

In any case, it holds true that the circadian signal, whether originated by autonomous oscillator systems or orchestrated from central and peripheral circadian clocks, is very closely integrated with cellular processes. Endocrine activity, cardiovascular activation, renal filtration function, xenobiotic and nutrient metabolism, and cell growth are all subject to circadian regulation (Hastings et al. 2003). For example, the critical process of mitosis, or cell division, is strictly timed around certain key checkpoints, such as the transition from G2 to M, G1 to S, and the so-called metaphase checkpoint, and one rational for this observed behaviour may be to minimize exposure of vulnerable DNA states to the destructive effects of UV radiation in the form of sun light, and to coordinate the effects of damage-repair functions. Overall, about 5-10% of local transcriptomes may be directly controlled by the circadian signal, with as much as 10-20% of the proteome being similarly rhythmic (Reddy et al. 2006). Many of the processes thusly modulated are vital functions, such as control of oxidative pathways, carbohydrate metabolism, or also corticosteroid biosynthesis, which is driven by a local clock in the adrenal cortex (Oster et al. 2006). The SNC in mammals can also communicate circadian cues through parasympathetic and/or sympathetic pathways to drive for instance glucocorticoid synthesis (Ishida et al. 2005). Within the hypothalamus, SCN projections were found to regulate levels of adrenocorticotrophins (ACTH), gonadotrophins, and metabolic hormones (Kalsbeek et al. 2006). Nevertheless, the role of peripheral circadian clocks appears to be prominent, too, and a knock-out study in mice revealed that the local hepatic signal was indispensable for nearly 90% of the circadian transcriptome in the liver (Kornmann et al. 2007).

Health Implications of Understanding the Circadian Clock

Considering the central and far-reaching position of the circadian clock in our physiological regulatory cascades, the study of underlying mechanisms is of course also acutely relevant for developing novel diagnostic and therapeutic approaches, and recent research reveals implications of the circadian clock in a whole range of pathological states. This breadth includes cases, where a compromised state of the molecular basis of the circadian clock directly constitutes or contributes to the health problem, others in which the circadian clock may be indirectly affected, or those cases in which its cyclical effects on physiological processes may simply be leveraged to modulate treatment options. After all, the human body has evolved to promote smooth and maximized functioning by operating within constrained daily cycles, that synchronize constituents with each other, as well as with social and wider environmental conditions. It is helpful to remember, however, that these rhythms may also interact with pre-existing pathologies, as manifested in the observation that cardiovascular and cerebrovascular events are most likely following the routine increase of cardiovascular activity after waking up (Hastings et al. 2003). On the other hand, there exist also well documented defects originating directly in the circadian system, such as a Per2 mutations that speeds up the circadian period and consequently causes the familial advanced sleep phase syndrome(Toh et al. 2001). A related field that is seeing a lot of interest lately, are disturbances of the circadian system brought about not by congenital defects, but rather adverse working or living conditions. Since animal models show very clearly that circadian cascades influence very directly blood pressure, heart rate (Curtis et al. 2007) and blood glucose levels (Rudic et al. 2004), it is not entirely unexpected that greater risks of hypertension, metabolic syndrome, mental health problems, and gastrointestinal disturbances have all been identified for shift workers (Sookoian et al. 2007).

The Link between Chronobiology and Cancer

Even more likely to grab headlines might be the diverse link of the circadian clock to malignant neoplasms. Once again, shift workers have been found to possess an elevated risk of being diagnosed with cancer (Schernhammer et al. 2006), and rotating lighting schedules have also been found to promote tumour progression in mice (Filipski et al. 2004). Looking at molecular mechanisms, it was found that altered expression of Per1 can affect the apoptosis behaviour of human cancer cells (Gery et al. 2006), and of course a disrupted circadian system could also affect tumour progression in indirect ways, such as altering immunocompetence, growth factor expression, or the endocrine environment (Hastings et al. 2007). Understanding better the exact role of the circadian clock in timing and modulating critical cellular processes offers not only insights into the disrupted states of cancer cells, however, but this knowledge may also readily and significantly boost the outcome of therapeutic procedures. For example, it was shown in several models how outcomes were significantly improved by simply varying the administration of cytotoxic chemotherapy drugs, so as to coincide with windows of vulnerability within tumour cells (Bernard et al. 2010; Altinok et al. 2009). Even so, it has been pointed out that the approach of such timed regimes should ideally also take into account the base circadian effect on drug metabolism and inactivation (Levi & Schibler 2007). Despite all the fascinating findings relayed above and the already surging interest in novel chronotherapeutic avenues, it is likely still fair to claim that we are presently only seeing the proverbial tip of the iceberg in the field of chronotherapy, and that years to come are likely to fundamentally alter our thinking on the timing aspect of health, disease, and therapeutic intervention.

Following on from the diverse regulatory effects exerted by the circadian clock, there is an understandably strong desire to elucidate its core functioning and underlying mechanisms. Indeed, since *Period* was identified in *Drosophila melanogaster* as the first clock gene (Konopka & Benzer 1971), understanding has moved forward considerably. Classical experiments by Colin Pittendrigh shaped the hypothesis of not just a single, but two distinct oscillators with different properties, termed E(vening) and M(orning) oscillators. There is now a wealth of evidence, both from behavioral and physiological studies, for multioscillator organization in circadian systems, and numerous gene outputs purely controlled by the circadian clock appear to fluctuate with coordinated periods (Storch et al. 2002). It has further become clear that rhythm generation is based on molecular transcription–translation feedback loops consisting of various clock genes and their protein products, and featuring both positive and negative reinforcement (Young & Kay 2001). Looking at biological oscillators in general, negative feedback is found to be essential to ensure a network is carried back to its starting point, while a sufficient delay ensures that reactions do not settle on a stable steady state. Oscillations in particular appear impossible in a two-component negative-feedback loop. It was also reported that circadian phases in *Neurospora* were aligned to clock protein levels, but not the respective RNA expression (Tan et al. 2004), suggesting that feedback loops may be partially uncoupled from clock function. Aside from the transcription-translation loops featured in most circadian mathematical models, there is now affirmation of post-transcriptional and post-translational mechanisms, such as protein phosphorylation, which have been demonstrated in cyanobacteria, and recently also in human red blood cells and the green algae *Ostereococcus tauri*. This phosphorylation action is not only understood to influence circadian rhythms by modulating clock-protein half-life (Görl et al. 2001; Young & Kay 2001), but might also be involved in shuttling between cellular compartments or triggering TFs.

Key Protein Interactions in the Circadian Clock

The molecular basis of the circadian system has been found to be remarkably conserved among species, and key components including *Per*, *Cry*, *Clock*, and *Bmal* have been identified repeatedly; the zebrafish model species, for instance, even contains extra copies of most key clock genes. As a general theme in the auto-regulatory feedback loop of core clock components, CLOCK and BMAL1 proteins hetero-dimerize and bind to E-box regulatory sequences on the *Period* and *Cytochrome* families of genes, triggering their transcription at the start of a new circadian cycle (Vallone et al. 2004). In this context, it has also been shown that Clock and Bmal1 both contain basic helix–loop–helix (bHLH) motifs at their N-terminus, which are known to play a key role in DNA binding. The resulting Per protein products in turn inhibit CLOCK/BMAL1, even if the nature of this interaction is not entirely clear. Proposed mechanisms include induction of phosphorylated states (Dardente et al. 2007), or physical interposition in CLOCK and BMAL1 binding, thereby compromising their dimeric action. Even possible effects on histone structure have been investigated (Ripperger & Schibler 2006), but in any case the C-terminus of Bmal1 plays a key role in activating transcription and has also been suggested to form part of a switch routing repression by Cry (Kiyohara et al. 2006). In fact, it was shown in zebrafish that Cry1a is up-regulated by light and may directly interact with specific regions of CLOCK - namely PAS B - and BMAL1 - including bHLH, PAS B and C-terminal domains - blocking their ability to form an active dimer and initiate downstream transcriptional activation (Tamai et al. 2007). This core loop is further augmented by stabilizing accessory loops, such as the one involving the two orphan nuclear receptor proteins Rev-Erb α and Rora. Both are upregulated alongside Per and Cry, and are understood to direct rhythmic expression of the Clock and Bmal genes, with Rora exhibiting positive feedback and Rev-Erb α acting as a suppressor of Bmal1 (Sato et al. 2004). As a consequence, Bmal1 and Per mRNAs usually rise and fall in anti-phase, and a surge of Bmal1 expression is seen to promote Per and Cry levels so as to just coincide with the waning of negative feedback. It has also been noted that the existence of various circadian loops may markedly increase robustness, and single knock-outs of Per1, Per2, Per3, Cry1, Cry2, Rev-Erbα and Clock have all proven insufficient to completely silence clock function (Ko & Takahashi 2006).

The Role of Environmental Cues

Despite recent advances in elucidating the molecular basis of the circadian clock, neurophysiological and system-level properties, for instance, still demand more investigation. After all, the endogenous capacity to generate circadian oscillations does not immediately explain the pacemaker's ability to effect rhythmicity to other structures. Another basic research area of particular note is the way in which external stimuli may interact with and adjust our inner time keeper. Such environmental cues, named Zeitgeber - meaning time giver in German - on the suggestion by Aschoff, play an important role in setting the phase of many circadian systems and can affect the underlying GRN directly at the cellular level, or indirectly through interactions or coupling among cells and networks at many levels of organization. An entire variety of potential environmental cues has been identified including feeding and tidal rhythms, and it appears well established, for example, that entrainment is effected by temperature cycles in the case of frequency (frq) -less *Neurospora* strains (Merrow et al. 1999); but even if species living in constant darkness are known to possess circadian clocks, the generally most prominent and important entrainment mechanism remains exposure to sun light. Interestingly, it was reported that retinal rod and cone photoreceptors are not required for circadian entrainment, and that the photopigment melanopsin, which is found in certain retinal ganglion cells (Qiu et al. 2005), alone is sufficient to achieve entrainment in the central circadian pacemaker. Even in the absence of light input, the circadian clock follows an oscillating rhythm with a period usually close to 24 hours, which is termed the free running period and specific to each species. In nature, however, the clock is habitually entrained by light input to the daily rhythm of exactly 24 hours. In this context, it is very interesting to note that in many species the overall signal strength of the circadian clock may also readily degrade in the absence of normal light/dark cycles. However, it has been demonstrated that individual oscillators continue functioning, but will increasingly desynchronize over time (Carr & Whitmore 2005). This suggests, in turn, that the major way in which light entrains the circadian clock may only become apparent when considering synchronization effects at the cell population level.

1.2 Ordinary Differential Equations Based Models

1.2.1 Describing a System by Fluctuating Concentrations

While some basic modeling methods are relatively simple to implement and yet able to point out important trends, they often fall short of capturing the finer points of dynamical systems. These dynamical networks are usually relatively small and their components can change their properties and hence the system state in a non-linear fashion over time, mirroring both interactions among these components and influences from outside the system. A reliable prediction of such a network's response to various environmental changes and stimuli calls for a more mathematically robust modus operandi, which is readily provided in the form of ODEs. ODEs modeling the concentration changes of molecules over time have already been widely used to analyze biological systems and GRNs, and by expressing the rate of production of a component of the system as a function of the concentrations of other components, this procedure is well suited to emulating gene regulation (Polynikis et al. 2009). Several reactants and reactions can be accounted for by systems of differential equations, but in practice the number of equations is oftentimes reduced by making a quasi-steady-state assumption on the mRNA concentrations. Of course, there exists a wide range of possible modification to the basic ODE methodology, each with their own specific advantages and drawbacks, such as the approach of discrete-time maps, in which the system progression is resolved in discrete time steps, rather than on a continuous scale (Coutinho et al. 2006), or the combination of pathway-specific ODE and mixture models in order to tackle kinetic snapshot data on the dynamics of distinct subpopulations (Hasenauer et al. 2014).

TABLE 1 A comparison of modeling and analysis techniques for high-throughput data. Taken from (Bordbar et al. 2014).

Method	**Model systems**	**Parameterization**	**Typical prediction type**	**Advantages**	**Disadvantages**
Stochastic kinetic modeling	Small-scale biological processes	Detailed kinetic parameters	Reaction fluxes, component concentrations and regulatory states	• Mechanistic • Dynamic • Captures biological stochasticity and biophysics	• Computationally intensive • Difficult to parameterize • Challenging to model multiple timescales
Deterministic kinetic modeling	Small-scale biological processes	Detailed kinetic parameters	Reaction fluxes, component concentrations and regulatory states	• Mechanistic • Dynamic	• Computationally intensive • Difficult to parameterize
Constraint-based modeling	Genome-scale metabolism	Network topology, and uptake and secretion rates	Metabolic flux states and gene essentiality	• Mechanistic • Large scale • No kinetic information is required	• No inherent dynamic or regulatory predictions • No explicit representation of metabolic concentrations
Logical, Boolean or rule-based formalisms	Signalling networks and transcriptional regulatory networks	Rule-based interaction network	Global activity states and on–off states of genes	• Can model dynamics and regulation	• Biological systems are rarely discrete
Bayesian approaches	Gene regulatory networks and signalling networks	High-throughput data sets	Probability distribution score	• Non-biased • Can include disparate and even non-biological data • Takes previous associations into account	• Statistical • Issues of over-fitting • Requires comprehensive training data
Graph and interaction networks	Protein–protein and genetic interaction networks	Interaction network that is based on biological data	Enriched clusters of genes and proteins	• Incorporates prior biological data • Encompasses most cellular processes	• Dynamics are not explicitly represented
Pathway enrichment analysis	Metabolic and signalling networks	Pathway databases (for example, KEGG, Gene Ontology and BioCyc)	Enriched pathways	• Simple and quick • Takes prior knowledge into account	• Biased to human-defined pathways • Non-modeling approach

Notably, where exact solutions cannot be found, it is possible to solve ODEs systems numerically, a functionality offered by and indeed at the centre of a great number of software implementations. In this context an important concept is the idea of convergence, i.e. the quality of the numerical solution to approach the exact solution as the size of integration steps tends to 0. Since all numerical algorithms are iterative processes, it is easy for errors to propagate, and step sizes have to be determined carefully to account for this tendency. The most common approach to evaluating lies looking at residuals, but at other times it is also possible to critically evaluate overall system dynamics, or to track a variable's value stabilizing at a monitor point. There a different methods available for arriving at numerical solutions, which can be broadly categorized as either linear multistep methods or Runge-Kutta methods, and many of which interestingly date back as far as the 19th century (Butcher 2000); for example, the Adams-Bashforth method that is widely used in software was first presented in a paper by Boshforth and Adams in 1883, while a manuscript by Runge in 1895 is considered the origin of all modern one-step methods. On the other hand, there is of course nevertheless constant adaptation and refinement occurring in the field of numerical solvers, such as applications of non-standard finite difference methods (Mickens 2003), or further developments to e.g. specifically preserve fixed points (Vigo-Aguiar & Ramos 2011) .

Limitations of ODEs

A key limitation of systems of ODEs, however, is the need for complete and quantitative data on the parameters used, such as concentrations, reaction rates and degradation rates, which may be difficult to measure experimentally. As a result, these values are often estimated in practice from the limited data available, or even established by trial and error from an intuitively plausible range. In addition to this, ODEs have difficulties handling non-homogenous local concentrations, and if certain processes such as compartmentalization can be expected to factor in partial differential equations (PDEs) should be preferred, as these can process spatial as well as temporal dependencies. There have also been numerous recent extensions to basic ODE methods, e.g. explicitly defined compartment exchange rates, to symbolize biological processes more realistically. Considering ODE sets on a practical level, the state of a dynamical system is, at any instant of time, specified by the values of the concentrations of all species in the network. However, the differential equations do not tell us "where we are" but "where we are going", that is how much each concentration will change in the next small increment of time. For the mathematical description of the GRN, each change in variable, that is the component concentration of interest, is described by the other components via parameters - quantities that are given a value - and constants - quantities with a fixed value. In order to predict the temporal progression of each component, the differential equations must be integrated, while a time course can graphically show the time evolution of one of the studied components over time, e.g. concentration of a protein or mRNA. It is also possible to visualize the dynamical system as a vector field or phase plane plot.

Vector Fields

In short, a vector fields maps vectors to individual points in a given subspace. The constitutive components include the state space - the set of numbers that quantify each component - the parameter space with the numbers that influence the rates of change of state variables, and fixed points, where the rates of change of all variables in a system are exactly zero. In the conservative sense, a fixed point simply signifies that a point of the function's domain is effectively mapped to itself, but attracting fixed points also see close-by values converging to them. Each point in state space is a vector that specifies the magnitude and direction in which the state variables are changing, and looking at a two-dimensional vector field, for example, the vectors could be represented as arrows with varying magnitudes and directions, each attached to a point on a plane. Of course, vector fields are not generally limited to a bi-dimensional form, and in an n-dimensional Euclidean space the vector valued-function linking to the points in the domain would likewise generate n-dimensional values. Consequently, vector fields are more generally defined on differentiable manifold, which appear similar to Euclidean space on small, but not larger scales. An orbit or trajectory is a path through state space traversed over time as a dynamical system follows the underlying vector field from an initial state to a final state. Since most real systems are nonlinear, they show interesting behaviours under such analysis.

Concepts of Stability

An example of noteworthy behavior can be that the vector field will point to certain attractors, the stable solutions of the differential equations, and such attractors can be seen as points of no movement, representing observable physiological states. With equilibrium points or periodic orbits, small perturbations to a system at a fixed point will often be accompanied by a return to this steady state, in which case the steady state is considered not only stable, but also attracts nearby orbits. Put a different way, a nearby orbit remaining close to a given orbit on an indefinite time scale would be considered stable, while a convergence of the first orbit to the given orbit would show asymptotical stability. When specifically considering dynamical systems, Lyapunov stable refers to a forward orbit remaining contained in a small neighborhood. Generally, stability of an equilibrium can be tested through the use of Lyapunov functions, scalar functions named after the Russian mathematician Aleksandr Mikhailovich Lyapunov. There exists no universal method for constructing Lyapunov functions, but the qualitative perturbation response of an orbit may also be analyzed based on the linearization of the local system. Specifically, for a smooth dynamical system with *n*-dimensional space, according to the Hartman–Grobman theorem an *n* x *n* matrix can be determined with eigenvalues describing the behaviour of nearby points, where the proportion of imaginary and real valued eigenvalues can provide insights into the underlying stability.

However, there are also unstable steady states, where perturbations grow larger with time so that the control system ultimately leaves the vicinity of the steady state. These repellers have surrounding vectors pointing away from them, while saddle points attract some nearby orbits and repel others. A popular metaphor for the different steady states is that of a damped pendulum. It would normally return to rest at a stable steady state with the bob hanging directly below, but an unstable steady state might be possible with the bob precariously balanced directly above the pivot point.

Oscillations and Limit Cycles

There are also oscillatory solutions, where the attractor is a closed orbit in state space. This behaviour, known as a limit cycle when one or more other trajectories spiral into or away from this closed orbit over time, was first described by Poincaré, and has traditionally been an important tool for modeling nonlinear systems. This type of system could for example denote protein concentrations changing in time, but repeating themselves after a characteristic period, as would for instance be observed for oscillating clock genes. Generally speaking, self-sustained oscillations and perfectly periodic system behaviour would be implied by attracting limit cycles, and these in turn act mostly analogous to "simpler" single point attractors, in that the system returns to their orbit following any small perturbations. Following the Jordan curve theorem, the plane would be divided into an exterior and interior of the curve by the closed trajectory, and identifying a trajectory approaching the limit cycle over time would point to a vicinity where all trajectories, or periodic solutions, are winding towards the attracting cycle. This constellation is known as a stable limit cycle, or ω-limit cycle, but there also exists the opposite constellation of an unstable, or α-limit cycle, where the neighbouring trajectories are approaching the limit cycle as time approaches negative infinity. In other words, considering this case from the normally assumed positive directionality of time, the trajectories would in fact be moving away from the limit cycle. Finally, a semi-stable limit cycle would exhibit some trajectories spiraling towards it, and others away from it.

Bifurcation Diagrams

Bifurcation theory, another term coined by Henri Poincaré in 1885, describes the study of qualitative or topological changes within e.g. a family of vector fields or solutions of differential equations representing a dynamical system. A bifurcation is observed when small changes in the bifurcation parameter under investigation lead to a sudden change of the systems behaviour, and this concept can be applied to continuous as well as to discrete systems. In a 2D or even 3D vector field, the different steady states can habitually be identified by simply looking at the diagram, and they all, including even the unstable ones that cannot be directly observed in vivo, carry far reaching physiological consequences. Attractors could, for example, represent different differentiation states of a cell, while unstable fixed points may denote physiological thresholds. In the case of a dynamical system with numerous components, these deductions principally hold true; n components can be represented by a point in an n-dimensional state space, where the set of all p parameters entailed can be represented by a point in a p-dimensional parameter space. Bifurcation theory can be used to analyze the bifurcations in parameter space by comparing simulation results to those obtained for the normal form, comparing their similarity as for example expressed by topological equivalence. This can often serve the objective of confirming the structural stability of the normal form. However, it is evident that for larger values of n a convenient visualization is impossible, and also that the exact nature of the steady states will depend on the precise values of the numerous parameters. The way out of this predicament lies in considering that there are only certain possible transitions between the different steady states, e.g. two steady states annihilating each other, or a stable steady state giving way to a stable limit cycle. In this way, bifurcation theory allows the analysis of the variation of steady states with certain combinations of parameters, as bifurcation points occur at specific values of the parameters. There are also several corresponding software packages that rely on different algorithms to solve defining equations for these bifurcation types and to determine coefficients of the normal form.

Once again, this is easily viewed in a one-parameter bifurcation diagram, where possible long-term values - including equilibria, fixed points, or periodic orbits - of a variable representative of the whole dynamic system are plotted against one parameter of choice. It is usual to represent stable solutions with a solid line and unstable solutions with a dotted line. In the setting of a biological system, these would be concentrations of a master factor, e.g. in the zebrafish circadian clock, Per1 concentration stands out as possible candidate. Common patterns on bifurcation diagrams include bi-stability of two stable fixed points. There is usually also a third steady, but unstable state, which can act as a separator between the orbital attractions of the two stable states. When the steady state makes a sudden jump at a particular value of the parameter, hysteresis occurs. There is also multi-stability, but in such a system, the random fluctuations in molecule numbers would tend to preferentially mask the existence of steady states with low average copy numbers. It should be noted, of course, that multi-stability, unstable steady states, and hysteresis can all be coexisting and interrelated. Other examples are a Hopf bifurcation, where a limit cycle evolves out of a stable point, and a saddle-node bifurcation, where two fixed points of a dynamical system collide and disappear. Bifurcation analysis will also give insight into which parameters are important for a transition from a stationary point to oscillations or vice versa, and at what parameter range oscillations occur. In local bifurcations, the parameter changes leading to an altered equilibrium stability can be confined to arbitrarily small vicinities by approximating the critical threshold bifurcation point with the bifurcation parameter. Global bifurcations, on the other hand, are often a result of larger invariant sets colliding with an equilibrium or each other. It is not possible to detect this category of bifurcation purely by stability analysis of the fixed points, and the resulting changed in the trajectory topology in phase space cannot be pinpointed to a limited neighbourhood, but rather reach out to an arbitrarily large distance. Finally, the term "codimension" describes the number of parameters that have to be changed simultaneously in order to manifest the bifurcation.

1.2.2 Major Dynamics in Representing GRNs as ODEs

GRNs exert their regulatory function on multiple levels and through various processes, such as transcriptional control, RNA processing, transport and degradation, translation and protein activity and degradation. In order to mathematically express many of these mechanisms, it is possible to utilize enzyme kinetic models, and given the analogy between enzyme kinetics and other biological processes, for instance transcription reactions, these kinetic models now find many uses. For example, many network models, including translation-transcription, signalling, and metabolic networks, are described by equations containing Michaelis Menten (MM) kinetics to model enzyme-mediated degradation of gene products, Hill functions to represent transcriptional activation, and linear functions for translation and protein transport rates in and out of the nucleus. Furthermore, if multiple types of posttranscriptional regulation and trafficking play a pivotal role in a GRN, these may be included in a model by treating them as different intermediary or states.

Law of Mass Action and Michaelis Menten Kinetics

The basis of practically all kinetic models is the Law of Mass Action, originally a mathematical construct that describes the behaviour of solution in dynamic equilibrium. It was first devised by Cato M. Guldberg and Peter Waage in 1864 following their work on reaction mixtures and equilibrium constants on the one hand, and rate equations for elementary reactions on the other. It was recognized early on that in order to maintain equilibria in a dynamic process, forward and backward reaction must exhibit equal rates of reaction, and this novel take on rate equations proved to be a breakthrough in predicting molecular kinetics. Specifically, the Law of Mass Action states that the reaction rate of any chemical reaction is proportional to the probability of a collision of the reactants, which in turn is proportional to the concentration of reactants to the power of the molecularity, i.e. the number in which they enter a specific reaction. Consequently, if two copies of a molecule are jointly participating in one part of the reaction, the concentration term for this component is squared. The rate constants - e.g. k_1, etc. - are constants of proportionality in the application of the Law of Mass Action, which represent the relative affinity for different reaction steps to occur. It should be noted, however, that an important assumption, and in some cases possible limitation, is that all reactants are in well-mixed conditions throughout the reaction process (Murray 2002; Klipp et al. 2009). Outside of describing classical chemical or kinetic processes, the law has also been applied as the basis of, for example, the Lotka–Volterra equations, which describe the dynamics of predator and prey populations in mathematical ecology. As another example of its universal usefulness, the Law of Mass Action has also inspired the compartmental model of disease spread in mathematical epidemiology, which simulates disease dynamics as the interactions of susceptible, infected, and recovered individuals (Adivar & Selen 2011).

Michaelis Menten (MM) kinetics are a simple, well known and widely used model of enzyme kinetics. After Victor Henri had first established that enzyme catalysis occurred through a binding interaction with the substrate, his work was continued by Leonor Michaelis and Maud Menten, who investigated the kinetics of the invertase enzymatic reaction mechanism hydrolyzing sucrose into fructose and glucose (Cornish-Bowden 2013). Having successfully accounted for factors such as pH, they were able in 1913 to develop their observations into the now famous mathematical model of conversion from substrate to product based on enzyme concentration. In detail, the MM equation is derived from the kinetics of a one-substrate reaction without backward reaction. The substrate *S* and the enzyme catalyst *E* reversibly form an enzyme-substrate complex *ES* and irreversible release product *P*, while k_1, k_{-1} and k_2 are constants of proportionality associated with the reaction rate. Double arrows indicate a reversible reaction and a single arrow a one-way one.

$$E + S \underset{k_{-1}}{\overset{k_1}{\longleftrightarrow}} ES \overset{k_2}{\rightarrow} E + P \tag{1.1}$$

Using the Law of Mass Action, this simple system can be mathematically described by the following system of ODEs, expressing the rate of formation of the different constituents dependant on their respective substrate concentrations, as modified by the different constants of proportionality:

$$\frac{dS}{dt} = -k_1 \cdot E \cdot S + k_{-1}ES \tag{1.2}$$

$$\frac{dES}{dt} = k_1 \cdot E \cdot S - (k_{-1} + k_2)ES \tag{1.3}$$

$$\frac{dE}{dt} = -k_1 \cdot E \cdot S + (k_{-1} + k_2)ES \tag{1.4}$$

$$\frac{dP}{dt} = k_2 \cdot ES \tag{1.5}$$

The reaction rate is equal to the negative decay of the substrate or the rate of product formation:

$$v = -\frac{dS}{dt} = \frac{dP}{dt} \tag{1.6}$$

This system cannot be solved analytically as it is, so several assumptions have been made for simplification. Firstly, it is assumed that there is a quasi-equilibrium between the free enzyme and the enzyme-substrate complex, meaning that the reversible conversion from E and S to ES is much faster than the decomposition of *ES* to *P* and *E*, i.e. $k_1, k_{-1} \gg k_2$. Secondly, it is also assumed that a quasi-steady state is achieved, where the concentration of *ES* remains virtually constant:

$$\frac{dES}{dt} = 0 \tag{1.7}$$

This statement is only justified if the initial substrate concentration is much higher than the enzyme concentration, otherwise such a state could not realistically be established.

Looking at the ODEs for *ES* and *E*, it can be seen that

$$\frac{dES}{dt} + \frac{dE}{dt} = 0 \; or \; E_{total} = E + ES = constant \tag{1.8}$$

showing that enzyme is neither produced nor consumed in this reaction, so the total enzyme concentration remains the same throughout and is either found unbound or as part of the complex.

Adding equation 1.3 in at this point under steady state assumption Eq. 1.7 gives:

$$\begin{aligned} \frac{dES}{dt} = k_1 \cdot (E_{total} - ES) \cdot S - (k_{-1} + k_2)ES = 0 \\ \text{thus} \\ k_1 \cdot (E_{total} - ES) \cdot S = (k_{-1} + k_2)ES \\ \text{thus} \\ k_1 \cdot E_{total} \cdot S - k_1 \cdot ES \cdot S = (k_{-1} + k_2)ES \end{aligned} \tag{1.9}$$

and thus

$$ES = \frac{k_1 \cdot E_{total} \cdot S}{k_1 \cdot S + k_{-1} + k_2} = \frac{E_{total} \cdot S}{S + (k_{-1} + k_2)/k_1} \quad (1.10)$$

From Eq. 1.5 and 1.6 it then follows that:

$$v = \frac{k_2 \cdot E_{total} \cdot S}{S + (k_{-1} + k_2)/k_1} \quad (1.11)$$

To represent this in a more convenient form, the following parameters are introduced: maximum velocity

$$V_{max} = k_2 \cdot E_{total} \quad (1.12)$$

which is the maximal rate that can be achieved when the enzyme is completely saturated with substrate;

and the Michaelis constant

$$K_m = \frac{k_{-1} + k_2}{k_1} \quad (1.13)$$

which is equal to the substrate concentration that yields a half-maximal reaction rate. Using this simpler form the classical expression of the MM kinetics can be derived:

$$v = \frac{V_{max} \cdot S}{S + K_m} \quad (1.14)$$

In other words, it can be seen that the reaction rate increases with substrate concentration *S*, approaching steadily but asymptotically, that is never quite reaching, the theoretical maximum reaction rate V_{max} when all enzyme would be bound to substrate at the same instant in time. As has been described above, the Michaelis constant K_m points to the substrate concentration resulting in half the maximum reaction rate, and this way acts as an inverse indicator of the affinity for the substrate binding the enzyme. Thus, a small Michaelis constant would signify a high affinity and speedy approach to the maximum reaction rate, but it should be noted that the exact value of K_m is not only dependent on both the enzyme and the substrate, but also such environmental conditions as temperature and pH. Oftentimes, in order to determine the different constants used in a specific application of MM, first initial reaction rates are measured through enzyme assay measurements. Subsequently, suitable parameters for the equation can be obtained through graphical methods involving linearization or, especially since the advent of corresponding computational capabilities, more commonly through non-linear regression.

Interestingly, it is widely accepted that many underlying assumptions of MM kinetics, including a rapid equilibrium and steady state, withstand a close scrutiny only for a small subset of conditions. For instance, the Law of Mass action used in the derivation of MM relies on free diffusion, but it has been noted that the cytoplasmic environment in a cell may more resemble a gel than a liquid and consequently affect molecular movement and reaction rates (Zhou et al. 2008). Especially in regulatory processes relying on relatively few copies of key molecules, the condition that substrate concentrations far exceed the enzyme concentration may further be compromised, and only sustain valid results for large values of K_m. Finally, the assumed irreversibility of the product formation step is also rarely confirmed. Practically speaking, the simplification may remain a good approximation when substrate concentrations far exceed product concentrations, such us when the product is continually removed by follow-on reaction steps, or when the reaction's energy release is very large, but in other conditions the simple MM kinetics may break down. Despite these caveats, however, MM has been found to describe many reactions adequately well, and has been employed for related settings, including antibodies binding antigen or DNA-DNA hybridization, but also outside areas such as the photosynthesis-irradiance relationship, richness of the species pool, or clearance of blood alcohol (Jones 2010). Numerous extension have also been proposed over the years, including the application of the Michaelis-Menten equation to single molecules (Kou et al. 2005).

Hill function

A Hill function is yet another, if more complex, enzyme kinetic law, or more generally one that describes the fraction of a macromolecule bound by a ligand as a function of ligand concentration. In order to make sense of its biological context, one first needs to visit the concept of cooperative binding. While enzyme mediated binding events already add a layer of complexity to reactions of "pure" mono-valent substrates, cooperative binding considers the case of one molecular species, possessing multiple binding sites and being commonly classified as a macromolecule, binding to multiple copies of another molecular species, the latter one being commonly referred to as a ligand. However, rather than being mutually independent events, each successive binding of a ligand may alter the shape of the macromolecule, a mechanism known as cooperativity and affecting the subsequent affinity for binding further ligands. It should be noted, however, that cooperativity can be negative as well as positive, with Hill coefficients <1 denoting the former and coefficients >1 the latter, and since it is observed in a wide range of proteins and other biopolymers, cooperative binding is understood as affecting a large range of physiological processes. One important consequence can be the occurrence of ultra-sensitivity, that is a response that is much more sensitive to ligand concentration as compared to standard responses such as the MM equation, and can lead to sudden fluctuations in steady state in biological systems. In a signalling pathway, such as activation, ultrasensitivity would alter the quality of the response to the extracellular stimulation from the normal hyperbolic response to a more sigmoidal curve.

One important example of cooperativity in a multi-site protein is the binding of oxygen to haemoglobin, which was extensively studied by Christian Bohr as early as 1904. Building upon this work, Archibald Vivian Hill described in 1910 equilibrium relationships between oxygen tension and the resulting saturation of haemoglobin, and in particular he was looking to explain the sigmoidal curve obtained for the binding of oxygen to haemoglobin with the interaction between the binding sites located on haemoglobin's subunits. Hill suggested a phenomenological equation that assumed complete cooperativity and a predicted Hill coefficient for haemoglobin's binding to oxygen of 2.8 from experimental data (Hill 1910). It is known today that haemoglobin has four binding sites and that cooperativity is not complete, but nevertheless the Hill coefficient remains a key element in modeling the various instances of physiological multi-site binding, including the regulation of multimeric enzymes, the opening of ion channels composed of several pseudo-identical domains, or cooperative binding in proteins, either found in complexes of homologous subunits, but also some proteins with several binding sites for a specific ligand. Moreover, the Hill equation has also been employed extensively in pharmacology to model quantitative drug–receptor relationships and other pharmacokinetic–pharmacodynamic dynamics (Goutelle et al. 2008). In the mathematical representation of the Hill function, a high Hill coefficient suggests extensive cooperativity and ultrasensitivity. Significantly, while in MM kinetics a fairly large change in substrate concentration range is required to effect a significantly different outcome, Hill kinetics work on a much smaller range and are thus more economical for the cell. Similar to MM, the Hill function is derived from enzyme kinetics (Polynikis et al. 2009).

A molecule that binds to a protein is called a ligand, e.g. a ligand *S* binds to a protein *E*:

$$E + S \leftrightarrow ES \tag{1.15}$$

the binding constant K_B is given by

$$K_B = \frac{ES}{E \cdot S} \tag{1.16}$$

The fractional saturation *Y* of the protein is the number of subunits that have ligands bound, divided by the total number of subunits available. For one subunit this would be:

$$Y = \frac{ES}{E_{total}} = \frac{ES}{ES + E} = \frac{K_B \cdot S}{K_B \cdot S + 1} \tag{1.17}$$

It should be noted that here the plot of *Y* versus *S* looks like the plot of *V* versus *S* in MM kinetics. In a process, where binding of *S* to *E* is followed by the release of a product and the initial concentration of *S* is much higher than *E*, the reaction rate is proportional to the concentration of *ES*:

$$\frac{v}{V_{max}} = \frac{ES}{E_{total}} = Y \tag{1.18}$$

If considering a dimeric protein with two identical binding sites, where the first ligand binding facilitates the second ligand binding, and *E* is the monomer and *E₂* the dimer, the equation can be written as:

$$\begin{gathered} E_2 + S \overset{slow}{\rightarrow} E_2S \\ E_2S + S \overset{fast}{\rightarrow} E_2S_2 \end{gathered} \tag{1.19}$$

The fractional saturation *Y* is now given by

$$Y = \frac{E_2S + 2E_2S_2}{2E_{2,total}} = \frac{E_2S + 2E_2S_2}{2E_2 + 2E_2S + 2E_2S_2}. \tag{1.20}$$

Assuming the affinity of the second ligand is strongly increased by the binding of the first ligand, *E₂S* will react with another *S* as soon as it is formed, and the intermediate *E₂S* can be ignored. This provides a case of complete cooperatively, i.e., the protein is either unbound or fully bound, which would reduce the above system to:

$$E_2 + 2S \rightarrow E_2S_2 \tag{1.21}$$

with a binding constant of

$$K_B = \frac{E_2S_2}{E_2 \cdot S^2} \tag{1.22}$$

and fraction saturation

$$Y = \frac{2E_2S_2}{2E_{2,total}} = \frac{E_2S_2}{E_2 + E_2S_2} = \frac{K_B \cdot S^2}{1 + K_B \cdot S^2}. \tag{1.23}$$

Generally, for a protein with *n* subunits, the rate is expressed as

$$v = V_{max} \cdot Y = \frac{V_{max} \cdot K_B \cdot S^n}{1 + K_B \cdot S^n} \tag{1.24}$$

which is known as the general form of the Hill equation.

Notably, in the derivation complete homotropic cooperativity, with all ligands binding at the same time, was assumed. A plot of fractional saturation *Y* versus substrate concentration *S* shows a sigmoidal curve with an inflection point at $1/K_B$, and an indication of the slope of the Hill plot is given by the Hill coefficient *n*. Furthermore, plotting the Hill function on a log scale will appear as a logistic function, but in the inverse a pure logistic function may model those behaviors better, in which concentrations leading to saturation do not extend over several orders of magnitude.

Today the Hill coefficient is often still regarded as a quantitative measure of cooperativity, particularly as an estimate of the minimal number of interacting binding sites in positively cooperating systems (Abeliovich 2005), although it is not directly related to how ligand binding at one site is affected by that at another and is usually less than the number of binding sites. Moreover, even where the Hill coefficient may give information on the number of interacting sites, it fails to distinguish between various underlying mechanisms (Prinz 2010). It has further been shown that for simple sequential or independent binding schemes the Hill coefficient is always less than 2 for up to 10 binding sites, and thus best thought off as an interaction coefficient (Weiss 1997). In summary, the Hill coefficient is only an accurate estimation of the number of binding sites if there is a high degree of cooperativity and when intermediate states are short-lived. Indeed, attempts have been made to replace the Hill functions commonly utilized in models, for instance using piecewise-linear approximations (Casey et al. 2006), but nevertheless the Hill function remains widely used and useful model, especially when the coefficient is considered as a more general interaction coefficient - a notion of a general change in affinity upon binding.

Gene regulation function

The various kinetic laws presented above can be utilized as building blocks in constructing more involved kinetic models of gene expression pathways, and in doing so it is often simplistically assumed that transcription rates only depend on regulator activity, while other influences are ignored. In simpler prokaryotes, a transcription process description may indeed accurately be derived from theoretical models, but in eukaryotes transcription is an extraordinarily complex process that depends on the precise orchestration of myriad interactions (Maston et al. 2006). Genetic promoters can be influenced by a large number of inputs and enhancers, DNA sequences that can bind a variety of transcription factors at multiple sites and activate transcription irrespective of their location and orientation relative to target promoters, in some cases even initiating transcription on different chromosomes (Lomvardas et al. 2006), and the interactions of enhancers and promoters can in turn be further modulated by elements including cohesin and non-coding RNAs (Ong & Corces 2011). As a tool for untangling this intricate mechanisms, a gene regulation function describes the binding of regulators to their recognition nucleotide sequence, and for a quantitative model two basic assumptions are routinely made here:

Firstly, on the time scale of interest thermodynamic equilibrium between different states is achieved and the probability of each state is dependent on the binding energy and concentration of any regulator present. Secondly, in each instance transcription initiation occurs randomly at a certain state. In order to translate a network structure into a dynamical model, signal strength *s* and response magnitude *r* have to be taken into account:

$$\frac{dr}{dt} = f(s,r) \tag{1.25}$$

Solving the steady state equation *0=f(s,r)* for *r* yields a steady-state response curve, i.e. the input-output relation for this arrow in the network.

Looking to the response curves resulting from different underlying dynamics, it can be observed that a linear network modelled with MM kinetics yields a hyperbolic response. Similarly, a loop structure modelled with linear kinetics gives an overall hyperbolic response, while a sigmoid response can be achieved by a loop structure with MM kinetics or a double loop structure with linear kinetics. Notably, it is thus emphasized that different motifs with different kinetics implementations can result in a similar overall response curve, a circumstance that can be utilized to reduce the number of equations and thus computation effort in a model system, while still capturing the most important dynamics of more complicated underlying mechanisms. A Hill function, for example, is often used in GRN models to represent in one term, what would instead have to be written out in several successive steps, such as transcription, modulation and trafficking.

Parameter estimation

One of the limitations of constructing ODE-based mathematical models, as mentioned before, consists of the requirement to first define various parameters. Especially in the case of many biological systems, this necessity may pose a problem as exact rate constant are often unknown and experimental data available tends to be incomplete and noisy. Previously, parameters were chosen manually in a time consuming trial and error manner, frequently not producing the optimal set and only generating a rudimentary view of the viable parameter space. Moreover, in complex models relying on a large number of parameters, the sheer number of possible combinations can quickly defy such manual attempts, except when sets perform well over a particularly large range of values, for instance using high Hill coefficients. Nowadays, once again largely driven by advances in computational techniques and prowess, several new approaches have thankfully been applied to searching the available parameter space and detecting optimal parameter sets. While top-down modeling statistical estimation procedures, for instance, are used to fit parameters from experimental data, Bayesian methods use Bayes' theorem to provide a fully probabilistic framework for describing experimental data, as summarized by a likelihood function, and prior knowledge about parameters to form a posterior distribution. This method is often applied to microarray or other high throughput data. However, the data analyzed may also be low-throughput with only a few different biochemical species, but a high time resolution, and data could also be collected either at the population or at single cell level.

Even with noisy and varied time course data, where absolute and relative levels are unknown, parameters can still be fitted for a deterministic model to reproduce experimental results.

It should be noted, however, that all approaches of parameter estimation are showing some interplay between value optimization and, since most ODEs show nonlinear behaviour, simulating the equation trajectory. In this context it has also be noted that most estimation approaches display either a relatively small convergence region or considerable computational costs (Peifer & Timmer 2007). Individual optimization techniques can differ markedly and are often labelled as either local or global optimization procedures, with global routines often being based on clustering approaches, simulated annealing, adaptive stochastic methods, or evolutionary computation (Banga et al. 2004). The equation's trajectory, on the other hand, is usually simulated using available ODE solvers, and a simple approach to improve the "goodness of fit" of a given model lies in calculating the distance between the simulated and the experimental data; parameters can be tuned with an optimization procedure to minimize this distance. In order to provide a quantitative value for the goodness of fit, a cost function can also be constructed by summation of several terms, each quantifying on a different level the agreement between model and experimental data, such as correct period, phase, and entrainment. Most optimization algorithms are constructed around checking different parameter sets, finding the best ones, and then using these as a starting point to find alternative or even better sets, as the loop starts over again. Of course, there are numerous specific implementations of this general theme, each with their own strengths and drawbacks, and numerous extensions of the methodology include, for example, utilizing a framework of measurement error in connection with a local smoothing approach and a pseudo-least squares (PsLS) principle (Liang & Wu 2008). Once suitable parameter sets have been identified, sensitivity and robustness analysis can also be readily performed to check how variation of parameters affects the output of a model.

1.3 Accounting for Stochastic Variation

1.3.1 Biological systems are noisy

Much bio-molecular research may aim to neatly identify pathways, classify interaction partners, and measure concentrations, but even so, at a microscopic level many biological processes are fundamentally characterized by noisy and random events that, among other things, lead to fluctuating amounts of substance molecules. As genes are generally only present in very few copies, i.e. one or two, and transcription factor molecules in the order of tens or hundreds, deterministic modeling approaches, where a given initial state always leads to the same observed state at a specific time later, may not be unconditionally valid. Rather, such fluctuating dynamics may be more accurately represented by several types of mathematical random or stochastic processes, based either on individual reaction events such as in the Chemical Master Equation and direct simulation, with frequencies in a given time interval as utilized in τ-leaping, or randomly drifting substance concentration, for instance implemented in Chemical Langevin Equations.

Noise as a Selective Advantage

It has been frequently observed that even genetically identical or highly similar cells, which are furthermore at least apparently exposed to entirely the same environmental conditions and stimuli, can show marked divergence in gene expression, protein levels, and more general phenotype. Such variability is attributed to inherent stochasticity, or the presence of random behaviours, which can not only frustrate research studies not prepared for its impact, but which has also been linked to detrimental clinical outcomes. For instance, increased transcriptional noise in older cardiomyocytes has generated the suggestion that DNA damage in the aging heart may be at least in part attributed to this increased stochasticity (Bahar et al. 2006). Overall though, stochastic gene expression and the resulting phenotypic diversity across population of cells or organisms is mostly described as highly evolutionary advantageous, and for instance genes highly involved in stress response and energy production have been shown to display greater translational fluctuations than other genes (Bar-Even et al. 2006). Indeed, in cells exposed to extreme stress, noisy gene expression provides a demonstrable fitness advantage (Blake et al. 2006). In particular, noisy expression has frequently linked to flipping cellular switches, such as the entry and exit from "persistence" in bacteria, which relies on the stochastic expression of the hipA gene and transforms a fraction of a given populations into a slow-growing state that can protect from antibiotic treatment or other environmental stress (Rotem et al. 2010). Related mechanisms also exist in complex multicellular organisms, and so it was shown that a population of haematopoietic progenitor cells exhibited frequently arising outliers as determined by SCA1 levels, with either very high or low SCA1 expression. Subsequent populations propagated from these outlier cells started off with similarly unusual SCA1 levels, but slowly self-corrected to the more broadly distributed SCA1 levels of the original cell population (Chang et al. 2008). There are also various examples, in which a switch appears originally fuelled by stochastic behaviour, but is subsequently stabilized in a more robust state. For example, somatic cells can be reprogrammed into induced pluripotent stem cells (iPS) by the TFs OCT4, SOX2, KLF4, and MYC (Wernig et al. 2007) in a stochastic manner, but subsequently display a stable pluripotent state and robust expression program (Boyer et al. 2005). Moreover, noisy gene expression can also be an ingrained part of guiding permanent cellular variation, such as the stochastic expression of olfactory receptors in mammals, whereby each sensory neuron expresses only one of hundreds of potential olfactory receptors encoded by the genome (Mombaerts 1999). The resulting distribution of different receptors across a population of cells is critical for establishing a response to odors with immense granularity. Finally, it should be noted that processes may also quickly switch from stochastic to more robust modes. For example, in Drosophila eye development only a single cell in each ommatidium develops into an R8 photoreceptor, whereupon other cells are immediately repressed from developing into R8 cells and instead guided to become other photoreceptors (Roignant & Treisman 2009).

Types of Noise

Noise can be classified into intrinsic and extrinsic noise. Intrinsic noise cannot be controlled for and stems from the inherently probabilistic nature of such cellular mechanisms as promoter/DNA binding events, mRNA transcription and degradation, translation, as well as protein-protein interactions. Notably, it can be observed even for identical genes in the same intracellular environment, and these chance events are able to have such a prominent effect due to the small number of molecules within a single cell. Extrinsic noise, on the other hand, can theoretically be controlled for and is related to different cellular environments, e.g. cell-to-cell differences in cell size and number of ribosomes, or to inputs from elsewhere in the network, such as in the concentrations of the specific trans-acting gene regulators (Swain et al. 2002).

While noise is undoubtedly an inherent feature of practically all biological system, it can be reduced or otherwise regulated in a gene specific mode. In fact, individual genes have been found to range considerably in their propensity for plasticity, and inspecting elements of their genetic architecture, the promoters of high-plasticity genes display high nucleosome occupancy upstream of transcriptional start sites and low occupancy more distally, whereas low-plasticity genes exhibit with greater frequency nucleosome free regions around their promoters (Tirosh & Barkai 2008). Nucleosomes are known to adversely affect the binding of TFs to target DNA segments, and so interactions or competitions between nucleosomes and TFs may contribute to stochasticity (Choi & Kim 2009). Further to this proposed link between gene architecture and expression noise, gene promoters featuring a TATA box were also shown to display more noise in their expression pattern (Tirosh & Barkai 2008). On the side of TFs, it is evident that their expression levels can also have an impact on noise in their target gene expression levels. After all, the interactions of TFs and genes are inherently probabilistic and depend not only on TF diffusion rates, affinity for different DNA sequences, the DNA's orientation, etc., but critically also the TFs concentration. TFs with low expression levels may thus exhibit a lower probability of binding a particular DNA sequence, especially if it competes for the same downstream target with a more abundant TF, and the response to the lowly expressed TF may thus show greater variability. Genes with low expression levels have also been shown to fluctuate more in their expression (Bar-Even et al. 2006), and noise can further be transmitted through a network from a TF to a downstream target (Pedraza & van Oudenaarden 2005), including other activator or repressor TFs, which may further propagate this noisiness. On a tightly related note, mutations in the binding sites of TFs may also change the strength and residence time of its interaction with regulatory DNA sites, thus altering the level of stochasticity in the expression of these genes. It could be speculated that a change to a lower binding affinity would increase noise in this way, while an approximation of the theoretically optimal TF motif would strengthen protein–DNA interactions and result in more robust downstream target expression.

Stochastic Timing of Expression

It should also be noted that stochastic or robust expression patterns are not limited to the variation in acute mRNA levels, but may also manifest in the timing of expression. For instance, it has been shown that in the embryogenesis of *Drosophila* many promoters are preloaded with RNA polymerase II, a mechanism that can accelerate the induction of gene expression (Hendrix et al. 2008), and furthermore that this preloading can reduce variability in not only transcriptional induction, but also overall phenotype; conversely, genes lacking stalled RNA polymerase II displayed not only significant stochasticity in their activation times across different cells, but also much greater variability in the expression profiles in the Drosophila presumptive mesoderm (Boettiger & Levine 2009). Furthermore, there is evidence that transcription occurs in bursts, with short periods of rapid production of multiple transcripts, interspersed with relatively long periods of no production, and this pattern, in turn, would have considerable implications for our understanding of general system dynamics and the origin of noise. However, it may be useful to remember at this point that stochastic behaviour is neither strictly negative nor positive, but rather that it depends on the precise nature of target genes whether changes in noise level may be beneficial or not; e.g. stress genes could possibly benefit from increased plasticity, whereas the disruption of robust essential processes might be more problematic. As such, it is not surprising that some systems have evolved to suppress noisy gene expression or to exploit it, e.g. the bi-stable systems in which cells can select from two phenotypes even in uniform genetic and surrounding conditions to facilitate adaptation to fluctuating environments. Even at the GRN level, different notes have been found to display different levels of noise. For example, highly connected nodes in protein-protein interaction networks, which are more likely to be essential and involved in multiple regulatory processes, also exhibit significantly lower levels of noise in the expression pattern when compared to nodes with fewer edges (Lehner 2008).

1.3.2 Stochastic Modeling Approaches

Chemical Master Equation

Before delving into the description of different modeling approaches for simulating the behaviour and effects of biological stochasticity, it should also be appropriate to point out that a process may, and in fact usually will, exhibit and be classified by different properties at the same time. Of particular interest in the modeling of noise is, of course, the random or stochastic process, which describes a system that can move randomly between different states in state space. On the other hand, A Markov process, named after the Russian mathematician Andrey Andreyevich Markov, is one, in which the behaviour of the system is determined by only the present state and not by past ones, unlike e.g. delay equations. Indeed, the Markov property is not only important for deterministic ODEs, but plays an even more prominent role in many stochastic processes, as long as a transition probability between the current and future states, without dependence on the past, can be assumed. When time and space are considered on a discrete scale, the process is habitually termed a Markov chain, and when the scale is continuous, the term continuous Markov process is used. Falling somewhere in between this major distinction, in a Markov jump process states are still discrete, but state transitions occur in continuous time. For instance, a discrete random walk would be an example of a Markov chain, but for many biological processes discrete time steps may not be suitable, and thus continuous time is often preferred for constructing models in this context.

A very exact representation of a system is provided by the Chemical Master Equation (CME), an equation that determines at an elementary level the probability that each species will have a specified molecular population at a given future time (Gillespie 1992). Here, the assumption is that for a very small interval $\Delta t \to 0$ the probability for a transition within this interval is proportional to the size of Δt, and the transition probability can be approximated by the proportionality constant, i.e. the transition rate. The transitions will lead to temporal changes of the state probabilities depending on a system of differential equations, namely the CME. Even though the cellular environment is densely populated and may exhibit localization, it is further assumed that the system is well-stirred and thus the positions and velocities of individual molecules are ignored.

- N different molecules or chemical species $S_1, \ldots ,S_N$, which interact in one or more of
- M chemical reactions $R_1, \ldots, R_M$ in volume Ω
- $\boldsymbol{X}_i(t)$ = number of molecules of species S_i in the system at time t
- making up the state vector $\boldsymbol{X}(t) \equiv (X_1(t),\ldots,X_N(t))$

The state vector $X(t)$ can change whenever one of the M type reactions takes place according to the state-change vector.

- Each reaction channel R_j is characterized mathematically by

1. stoichiometric state-change vector $v_j \equiv (v_{1j}, \ldots, v_{Nj})$
   where v_{ij} is the change in the S_i molecular population caused by one R_j reaction, i.e. if from state $\boldsymbol{x}$ one R_j reaction occurs, the system immediately jumps to state $\boldsymbol{x} + v_j$
2. propensity function a_j (see below)
   the function whose product with dt gives the probability that a particular reaction will occur in the next infinitesimal time interval dt

The aim is to estimate the state vector *X(t)*, given that the system was in state $X(t_0)=x_0$ at some initial time t_0.

Propensity

The propensity function provides the stochastic rates of the participating reactions:

$a_j(\boldsymbol{x})dt$ = the probability, given $\boldsymbol{X(t)} = x$, that one R_j reaction will occur somewhere inside Ω in the next infinitesimal time interval *[t,t+dt)*

Furthermore,

$$a_j(\boldsymbol{x}) = c_j\, h_j(x)$$

where c_j is the *specific probability rate constant* for R_j, defined so that $c_j\, dt$ gives the probability that a randomly chosen pair of R_j reactant molecules will react accordingly in the next infinitesimal time interval *dt*. This probability is related to the average relative speed, collision cross section, and inversely to volume Ω, and the probability that the collision energy exceeds a threshold level. $h_j(\boldsymbol{x})$ is defined to be the number of distinct combinations of R_j reactant molecules available in state $\boldsymbol{x}$.

If R_j is the unimolecular reaction $S_1 \rightarrow$ product(s), the underlying physics, which is usually quantum mechanical, dictates the existence of some constant c_j, s.t. $c_j\, dt$ = probability that any particular S_1 molecule will react in the next infinitesimal time *dt*.

Thus, if there are currently x_1 S_1 molecules in the system the probability that any one of them will undergo the R_j reaction in the next *dt* is $x_1 * c_j\, dt$. Following on from this observation, the propensity function can be written $a_j\, (\boldsymbol{x}) = c_j\, x_1.$

For a bimolecular reaction of the form $S_1 + S_2 \rightarrow$ product(s), the propensity function becomes $a_j\, (\boldsymbol{x}) = c_j\, x_1\, x_2$ whereas for $S_1 + S_1 \rightarrow$ product(s): the propensity function becomes $a_j\, (\boldsymbol{x}) = c_j\, ½\, x_1\, (x_1\text{-}1).$

It is noteworthy that for a unimolecular reaction, c_j is numerically equal to the reaction-rate constant k_j of conventional deterministic chemical kinetics, whereas for a bimolecular reaction, c_j is equal to $\frac{k_j}{\Omega}$ if the reactants are different species, or 2 $\frac{k_j}{\Omega}$ if they are the same species.

Derivation of the CME

The propensity function is probabilistic in nature, and so exact predictions on the state vector ***X****(t)* cannot be made, but the probability that ***X****(t)=x* given ***X****(0)* can still be attempted to be inferred as follows:

$$P(x,t\,|\,x_0,t_0) = Prob\{X(t) = x, given\,X(t_0) = x_0\} \quad (1.26)$$

In other words, the probability to obtain a specific state *x* at time *t* given a set of initial condition equals the probability evolution of the state vector over this time interval. In order to derive the appropriate time evolution, it can be considered what happens in the time increment *dt*, which is so small that the probability of several reactions occurring is negligible compared to the probability of only a single one; $1-\sum_{j=1}^{M} a_j(x)dt$ is the probability of no reaction taking place:

$$P(x,t+dt\,|\,x_0,t_0) = P(x,t\,|\,x_0,t_0)\left(1-\sum_{j=1}^{M} a_j(x)dt\right) + \sum_{j=1}^{M}[P(x-\nu_j,t\,|\,x_0,t_0)a_j(x-\nu_j)dt]. \quad (1.27)$$

Here, the probable reaction over the next time increment *dt* is considered in terms of the mutually exclusive and collectively exhaustive paths leading to a specific state over a time interval. As can be seen, the multitude of potential paths to reach this state depends on the set of potential reactions - or no reaction - occurring, with corresponding initial states and individual associated probabilities that, in light of their mutual exclusiveness, can be simply added up. This leads to the CME:

$$\frac{\partial P(x,t\,|\,x_0,t_0)}{\partial t} = \sum_{j=1}^{M}[a_j(x-\nu_j)P(x-\nu_j,t\,|\,x_0,t_0) - a_j(x)P(x,t\,|\,x_0,t_0)] \quad (1.28)$$

Principally, the CME completely determines the function $P(x,t\,|\,x_0,t_0)$ over time evolution based on fixed initial conditions; on a practical level the CME is actually a set of coupled ODEs, with one equation for every possible combination of reactant molecules. However, considering each molecule individually in this fashion, the sheer amount of possible states of the system can quickly reach truly exorbitant values, and as a practical consequence, a CME generally has such high dimensions that it cannot he handled analytically or computationally at the present time on any feasible scale for simulation purposes.

Stochastic Simulation Algorithm

It has been pointed out above that the CME can rarely be solved for the probability density function of ***X****(t)*, but knowledge of the system can still be gained by generating numerical realizations of ***X****(t)*, i.e. sample trajectories of ***X****(t)* over time. The stochastic simulation algorithm (SSA), also called Gillespies algorithm, uses a Monte Carlo procedure and, rather than solving the full set of CME ODEs to generate a probability distribution over all possible states for each time interval, random samples of ***X****(t)* are produced; i.e. realizations of the state vector *{t,* ***X****(t)}* are computed in such a way that the chance of a particular realization being selected reflects the corresponding probability given by the CME. After the algorithm was presented by Dan Gillespie in 1976 (Gillespie 1977), drawing heavily on earlier work by Joseph Doob, it soon rose to widespread use in systems biology and for modeling biochemical dynamics in general. Its properties are especially appealing for system with only few reactants, which are only poorly represented by "bulk" biochemical rate equations, whereas the SSA handles well discrete and stochastic simulation of low numbers of molecules, with each reaction being explicitly simulated. On the other hand, while the SSA is clearly based on the CME, shares most of its base assumptions, and presents a trajectory that is an exact sample from the CME, its computational requirements are on a much more realistic scale. Indeed, with the constant advance of computational power, increasingly complex systems have been implemented in the SSA form.

For the SSA, the next-reaction probability density function - i.e. given state **X**, the probability that the next reaction will occur within time τ - is computed:

$p(\tau, j \mid x,t)d\tau$ = Probability that, given ***X(t)***=*x,* the next reaction in Ω will occur in the infinitesimal time interval *[t+τ, t+ τ +d τ),* and will be an R_j reaction.

In order to derive an exact formula for $p(\tau, j \mid x,t)$, laws of probability can be applied to the propensity function:

$$p(\tau, j \mid x,t)d\tau = a_j(x)exp(-a_0(x)\tau), \tag{1.29}$$

where

$$a_0(x) = \sum_{j'=1}^{M} a_{j'}(x)$$

This is the mathematical basis for the SSA, implying that τ is an exponential random variable. $\sum_{j=1}^{M} a_j(x)dt$ is the probability that some reaction will occur in the next *dt*. The exponential distributions are a class of continuous probability distribution, describing the time between events in a Poisson process, i.e. a process in which events occur continuously and independently at a constant average rate.

Two random numbers r_1 and r_2 are drawn from the uniform distribution in the unit interval, and used to compute

$$\tau = \frac{1}{a_0(x)} ln\left(\frac{1}{r_1}\right), \tag{1.30}$$

$$j = \text{ the smallest integer satisfying } \sum_{j'=1}^{j} a_{j'(x)} > r_2 a_0(x) \tag{1.31}$$

The algorithm goes as follows:

1. Initialize the time $t = t_0$ and the initial system's state $x = x_0$
2. With the system in state *x* at time *t*, evaluate propensities $a_j x$ for all possible reactions and their sum $a_0 x$
3. Generate values for the time to next reaction event τ and reaction *j* using Eq. 28 and 29
4. Effect the next reaction by replacing $t \leftarrow t + \tau$ and $x = x + \nu_j$
5. Record (x,t). Return to step 2 or end simulation.

One ***X(t)*** trajectory produced by this algorithm can be thought of as a stochastic version of one that would be obtained solving ODEs. If every SSA-generated trajectory is indistinguishable from the ODE counterpart, it can be concluded that micro-scale randomness and noise are ignorable, but in the opposite case, where there is a significant deviation, the micro-scale randomness and noise are an important part of the system's true dynamics. As the number of reactions increases, the factor $\frac{1}{a_0(x)}$ in Eq.1.30 causes the SSA to become very slow, and so despite its widespread use, SSAs may be infeasible for larger, complex networks and different timescales.

Approximate Simulation Strategies

The direct SSA can be prohibitively slow if the network under scrutiny is complex or if reactions otherwise occur frequently, but one way to speed up calculations is to lump several reactions together for an interval of length tau and to only update state vectors once these reactions have all fired. This so called tau-leaping method introduces an error, but one that will be relatively small if the state vector changes are small, and in return the efficiency of simulating the chemically reacting system is improved considerably by updating the propensity function less often. The number of reaction firings are approximated by Poisson random numbers, and since the tau-leaping methodology's original inception various modifications have been proposed, e.g. to increase overall accuracy, or to prevent the population of some reactants being driven negative due to a randomly selected extremely large Poisson value (Cao et al. 2005). Of course, estimating a good value for τ to advance time by is also important. After all, the assumption that no time step will alter the value of any propensity function significantly will critically depend on preselecting τ from within an appropriate range, and once again different procedures have been presented to automate or aid with this selection (Cao et al. 2006). It is interesting to note that, while the tau-leaping approach is removed by several notable efficiency and thus computational speed boosting simplifying assumptions from the CME, it still remains rooted in it and is frequently utilized when simulations are concerned with very low copy numbers of molecules.

Moreover, when the number of firings of each reaction channel during tau-leaping is much greater than one, a further simplification to the approximation can be made to yield the Chemical Langevin Equation (CLE). A diffusion approximation of the process can be implemented by closely matching a Markov jump process with a Stochastic Differential Equation (SDE), which acts much like an ODE and also features continuous trajectories, but by contrast contains an additional, stochastic term. Over a wide range of conditions and especially when computing with large numbers of molecules, any differences between the approximate and exact model are usually understood to be dwarfed relative to the error introduced by the underlying assumptions to this point. Now Poisson random numbers can also be well approximated by normal random numbers, and the CLE resembles a set of *N* SDEs. The state vector is now a continuous time, real valued process, and in moving from the CME to the CLE, the dimensions of the system are reduced from integers to real values when describing the number of molecules, and changed from a probability distribution over a large, discrete set to a continuous probability distribution for *N* chemical species. The Fokker-Planck equation can be used for a Langevin equation with continuous states, rather than in discrete state space and continuous time as the CME, by computing the probability density for the corresponding SDEs. More precisely, it is a partial differential equation (PDE) specified by a drift vector and a diffusion tensor, that describe the time evolution of the probability distribution functions of the SDEs.

It has been pointed out above that SDEs can be instrumental in implemented Approximate Simulation Strategies, and they are indeed widely used to model diverse phenomena ranging from thermal dynamics, over stock price fluctuations, to biological regulatory systems. Mathematically, they are differential equations, in which at least one term is a stochastic process, such as Gaussian white noise derived from the Wiener process, or also randomly fluctuating jump processes. It should be noted that the Wiener process is very complex mathematically and practically impossible to differentiate, and consequently specific rules had to be devised to handle this kind of stochastic calculus, the two most widely used versions being Ito and Stratonovich stochastic calculus. There are significant difference between the two approaches, with advantages and disadvantages for each, so care should be taken when selecting one over the other. Moreover, it has been found that algorithms used for solving ODEs will generally show very poor numerical convergence when applied to SDEs and thus deliver unsatisfactory results, except in simpler cases of "additive noise" type stochastic terms. Fortunately, a selection of methods has been developed over the last decades to find numerical solutions specifically for SDEs, the most common being the Euler-Maruyama method, Milstein method, and a generalized form of the Runge-Kutta method.

1.4 Modeling the Circadian Clock

1.4.1 The Circadian Clock as a Major Regulator

This following section will aim to provide a short overview and description of modeling approaches for circadian clocks in general, as well as specifically in different important model species. The immense advantage imbued by possessing an internal time keeper is highlighted by that fact, that circadian clocks have probably evolved several times independently, an occurrence termed parallel evolution, and cyanobacteria circadian proteins for instance appear unrelated to any other. On the other hand, however, it is also speculated that there may have been a basic clock mechanism in an ancient common ancestor, e.g. even before separation of insects and mammals. In any case, while there are some orthologs between mammals, flies and maybe even fungi, the basic circadian clock layout has also been extensively modified and extended over the course of evolution and across species, e.g. through the inclusion of central pacemakers and sensitivity to new environmental cues.

Variation within and across Species

It has also been observed in other contexts that TF binding sites can differ greatly between species, even in genes and pathways that are highly conserved, and CEBPA and HNF4A binding in livers of human, mouse, and dog, for example, only appears to overlap as little as 10%–22% between any two of these species (Schmidt et al. 2010). Moreover, looking at the variation of TF binding within the same species, it has been shown in human lymphoblastoid cell lines that 25% of NFKB and 7.5% of Pol II binding sites were different even between specific individuals (Kasowski et al. 2010). An explanation for this somewhat surprising finding may be the realization that non-coding regions of the genome evolve much faster, a circumstance that can also affect TF binding sites and in this way drive variations in gene expression and, ultimately, speciation. Indeed, a study scrutinizing wild versus laboratory strains of the yeast S. cerevisiae found significant variations in the expression of almost a quarter of all genes, namely 1528, and 62 genes even varied in their respective expression more than eightfold (Brem et al. 2002). Related gene expression differences investigated through the use of S. cerevisiae and Saccharomyces paradoxus hybrids were mostly attributed to cis-effects (Tirosh et al. 2009) , pointing to variations in regulatory DNA sequences. However, it is also known that environmental stress tends to strengthen variability through trans-effects, such as TF activity, and these findings may point to the coexistence of several mechanisms to drive evolutionary mutations depending on various circumstances.

In the case of the circadian clock, several circadian proteins with certain highly conserved domains have been identified in a range of species, such as the DNA binding basic Helix-Loop-Helix (bHLH) domain and the protein-protein interacting PAS domain. Often there exists also a transcriptional activator, i.e. a positive network element, that is active as a heterodimer and can activate one or more negative elements after transport from the cytoplasm to the nucleus. The negative element typically accumulates in the cytoplasm for several hours after its synthesis and, in order to exert repressive effects, it must translocate to the nucleus, where it clears the positive element. Adding significantly to overall complexity, there may also be elements with a dual activator and inhibitor role in different loops, and light entrainment can occur through components acting as a light input pathway to the GRN.

1.4.2 Concepts Relevant to Circadian Modeling

Advanced Concepts

Various species-specific, but also more general circadian clock models have been implemented to date, but before delving into a discussion of their exact implementation, it is warranted to first mention certain areas and behaviours that beg particular attention in the simulation of the clock. While most broader points for the modeling of GNRs raised elsewhere are of course still valid in this context, there are also some less common elements, which are nevertheless of special importance for our inner time keeper. Firstly, the ability to synchronize internal to external rhythms through a mechanism called entrainment is of clearly recognized importance, and a critical aspect of this feature in turn lies in the fact that an external stimulus can have varying or even opposing effects, depending on the timing of when the system is exposed to it over the course of its underlying trajectory. This basic drift component is of course manifested as an oscillation pattern, which should on one hand persist robustly in isolation, as well as under the influence of significant perturbing and persisting cues, while still being sufficiently flexible to adjust to these outside signals. Before this background it is interesting to remember that "adaptation", i.e. the challenge for a system to adjust its state in response to continuous exposure to physiological, environmental, or pathological cues, can be a common occurrence in biology, as well exemplified by the adaptation of neuronal responses. In some cases, new states are reached, but under perfect adaptation the system would return to exactly its original state. Interestingly, contemplating how biochemical adaptation can be affected at the level of three-node network structures, researchers found that only two core topologies support adaptation, namely a form of FBL with a buffering node, or secondly an incoherent FFL including a node proportionally activated by upstream elements (Ma et al. 2009). A fascinating question lies in linking this finding of highly robust small-scale design principles in with more complex, larger-scale networks and GRNs.

Entrainment

Across species where central pacemaker are found, the exact structure of these central rhythm generators can vary widely, but all are capable of generating a uniform and reliable timing signal with remarkable precision, often only deviating by a few minutes a day. Looking to the mode in which this system is handling information on external conditions, it was found that several stimuli are capable of entraining the oscillator to their rhythm. In order to describe the standardized 24-hour notation of the phase in the entrained clock, habitually the notation of Zeitgeber time (ZT) is used, in which ZT 0 indicates the beginning of the photoperiod, e.g. the lights being switched on or the sun rising. While photic entrainment, that is caused by the alternation between day and night and to which most species are sensitive, is often most prominent, other stimuli might be relevant, too, such as food availability, social contacts and even tides, temperature, and moonlight. Strictly speaking, entrainment is not the same as synchronization, which would imply that the waveform of the driving rhythm coincides with the waveform of the driven rhythm. However, the molecular clock is not necessarily synchronized to the environmental cycle when it is entrained; rather, the consequences of entrainment are that the period of the biological rhythm becomes equal on average to that of the entraining stimuli and that a stable phase relationship is established between the entraining and entrained oscillations. Thus, in order to demonstrate that a zeitgeber cycle has indeed entrained the molecular rhythm, firstly the period of the rhythm should equal the period of the zeitgeber cycle with a stable, unique phase angle, and secondly, after removing the inputs from a zeitgeber cycle, the FRP should resume with a phase determined by the zeitgeber cycle. Conversely, if the FRP starts up from a different phase, the stimuli may have forced expression of the overt behaviour without actually entraining the underlying dynamics, a phenomenon that is referred to as masking, and that can be a serious artefact of investigations relating to entrainment. It is also interesting to note that a previous entrainment cycle may influence the FRP. This circumstance means that light-dark cycles of different photoperiods or different periods can have profound after-effects on the subsequent FRP, but these after-effects usually decay in time after transfer to constant conditions.

For many species the main entrainment factor is light, as has been pointed out repeatedly, but notably light does not always have the same effect on the circadian clock. Further complicating matters, tests in the laboratory using square-type photoperiods may not necessarily mimic real-life entrainment and, for instance, behavioural responses to simulated twilights have been shown to differ in the range and quality of entrainment. It also appears that multiple photoreceptor pathways are involved in circadian entrainment and masking. While short light pulses induce discrete or nonparametric entrainment of rhythms, longer or even constant light durations tonically affect the frequency of the clock through continuous or parametric entrainment. These disruptions of circadian rhythmicity depend on the intensity of light and exhibit a continuum of responses, ranging from subtle behavioural changes to complete arrhythmicity. Interestingly, although constant light (LL) is able to desynchronize SCN cells in mammals, the molecular clock is not stopped and circadian changes in the expression of clock genes persist. This finding may signify that brief light pulses cause phase advance or delay in the clock through changes in the expression of clock genes, while chronic light affects SCN neurons coupling via a mechanism as of yet unidentified. Following a transfer to constant darkness (DD), circadian rhythms in behaviour and gene expression are also observed to quickly resume from a specific phase, suggesting that the clock output may have been masked. The parametric entrainment model has been based on the observation by Jürgen Aschoff, considered one of the founders of chronobiology, that the FRP is dependent upon light intensity. This view further postulates that light has a continuous action on the clock in entraining it to the light-dark (LD) cycle, and a proposed mechanism is the acceleration and deceleration of the FRP, via its angular velocity, by varying light conditions, which would allow the continuous adjustment of clock cycle length to the duration of the environmental rhythm. In a mathematical model, these velocity changes necessarily affect at least one parameter for the duration of light exposure, hence the term parametric, and in turn this continuous manipulation of photic parameters can modify circadian period and phase. The nonparametric model was put forward by Pittendrigh and is suggested to act through instantaneous phase shifts in response to light transitions. The basic premise of the model is that an entrained clock is in equilibrium with a LD cycle consisting of repetitive light pulses (the zeitgeber), and that equilibrium would be achieved when each light pulse falls at a phase so as to elicit a phase shift that is equal to the difference between the FRP and the period of the entraining cycle. In nature the zeitgeber would be the dawn and dusk transitions, which can be mimicked in the laboratory by brief light pulses. Since the effect of light is due to discrete time cues, as opposed to parameter changes in the underlying molecular oscillator, this mechanism has been called discrete or nonparametric entrainment. However, the view has emerged that in nature entrainment is likely shaped by both parametric and nonparametric effects.

Phase Response Curve

One of the most useful approaches for investigating entrainment is the construction of phase response curves (PRCs), which describe the effect of the same stimuli at different times of the reference period. The concept of Circadian time (CT) (cf ZT) describes the state of the clock and timing of activity in constant conditions, where CT0 is usually the time when lights would have been turned on as part of a LD cycle, i.e. the start of subjective day. The PRC indicates the time points at which an entrainment signal can induce phase delays, phase advances, or no change at all, thus transforming the intrinsic circadian period τ to T, the period of the environmental zeitgeber. Graphically, the PRC plots the phase shifts of a circadian rhythm, as for example determined by a peak of gene expression or locomotor activity, as a function of the circadian phase of a zeitgeber, and by convention, phase advances are plotted as positive and phase delays as negative values. Photic PRCs are typically biphasic, such that light pulses presented in the subjective day have little or no effect, a behaviour sometimes described as a dead zone, but phase delays occur at the beginning and phase advances at the end of the subjective night. In contrast, non-photic PRCs may exhibit clear phase advances during the subjective day and little, if any, responses during the subjective night. However, it should be noted that the precise waveform and amplitude of the PRCs are an intrinsic property of the circadian oscillator and thus species specific. Moreover, these qualities also depend on the type, strength and duration of the stimulus and may even be affected by previous photoperiodic history, such as the length of time an organism has been subjected to constant DD or LL conditions.

PRCs are also often classified depending on their strength in terms of phase shifting, where weak type 1 PRCs display maximal phase shifts in the order of a few hours and gradual transitions between phase advances and delays, while type 0 PRCs have larger maximal phase shifts, of up to 12 h, and the transition between delays and advances is quite abrupt and discontinuous. It has been pointed out, however, that the breakpoint discontinuity of type 0 PRCs appears to be in some cases merely a plotting convention of arbitrarily assigning phase shifts as delays vs. advances. Examples of the different PRC types include *Drosophila pseudoobscura*, which exhibits a type 0 strong light resetting PCR, whereas nocturnal rodents display type 1 weak resetting. It is further noteworthy that in many multicellular, but rarely in unicellular organisms, there often exists transient behaviour in the first few cycles after a perturbation by a stimulus. Following from this, a PRC approach may make important assumptions that cannot be directly measured, since the steady-state consequences of an intervention can only be observed one or even more days after it has been presented. It is not known then, whether the steady state was due to a transient change in velocity, i.e. parametric in nature, or due to an acute change in phase, that is nonparametric. Considering the more recent understanding of how complex circadian systems usually are at all levels, and with a view to the poor temporal resolution of circadian experiments, it is conceivable that the system could in fact need much of its cycle to manifest a response (Roenneberg et al. 2010).

Modeling Oscillators and Entrainment

Oscillators are commonly observed in physiological regulatory systems, and accordingly also frequently modelled in biological and synthetic control systems. Generally, all biochemical oscillator GRNs are characterized by negative feedback with time delay (Novák & Tyson 2008), and his time delay can be achieved by several approaches, including a few highly non-linear reactions or chains of intermediates between cause and effect. An additional way to generate oscillations in genetic circuits consist of modifying bistable systems by adding a destabilizing negative feedback loop, and in analogy to magnetic spin systems this has been called frustration (Krishna et al. 2009). Bistable systems are quite common in biological systems, where they are used as switches and memory elements, but in contrast to "true" oscillators, bistable systems must necessarily contain a positive feedback loop. The shape of the resulting oscillations can be readily tuned to produce spiky and asymmetric oscillations, and the time period and amplitude can also be manipulated. Furthermore, it has been found that these oscillators are easy to reset or to switch on and off using a tuneable external input.

In order to describe and quantify a particular rhythm, common qualities to note include the period *T*, cyclic frequency, i.e. oscillation cycles per unit time, $f = \frac{1}{T}$ and angular frequency $\Omega = 2\pi f = \frac{2\pi}{T}$($2\pi$ *radians =360°*). The phase determines the state of a periodic oscillator; it increases by 2π within one oscillatory cycle, the period, and thus phases that differ by 2π are in the same state. In a dynamical system, the circadian clock could be presented as a limit cycle, where the state variables move around a stable trajectory in phase space, corresponding to rhythmicity.

In contrast, arrhythmicity would simply correspond to a fixed point in phase space. The periodic output of an oscillator can be denoted by the process *x(t)*, but in order to describe the state of an oscillator the value of *x* is not sufficient, and in many cases two variables *x* and *y* are used, e.g. angle of a pendulum and angular velocity. A complete description of the system is achieved by the time evolution of the pair *(x,y)*, and the coordinates *(x,y)* are called the coordinates of phase space, or state space, and can be plotted as *y(t)* vs *x(t)*. After period *T* the oscillation is repeated, thus corresponding to a closed curve in the phase plane, making up the limit cycle. Should the oscillations be close to a sine wave, then the oscillator is quasi-linear, or quasi-harmonic, and the limit cycle is represented as a circle. If the oscillator is perturbed, i.e. the state variables are pushed off the limit cycle, they will return to it, in phase with a specific point on the limit cycle (Pikovsky et al. 2001). Here, isochrons are a set of points in phase space, which specify values for the state variables that will return to the limit cycle in the same phase. I.e. an isochron appears as a line in phase space originating in the centre of the limit cycle, that leads to the same asymptotic phase (iso=same; chronos=time). For example, a sinusoidal oscillator has equally distributed phase points and isochrons, whereas a spike-like oscillator would display an asymmetric distribution of isochrons, compressing them around the point where the majority of time is spent.

Mathematically, external stimuli may reset a limit-cycle oscillator by changing the parameters, or additional terms in the differential equations may allow direct changes of the state variables, and in these ways state variables could change rapidly in response to excitation by zeitgebers. If a change moves the state variables from one isochron to a different one, a phase shift would be observed, since a different phase is reached when the state variables move back to the limit cycle. Stimuli presented at phases in the dead zone may not modify the state variables, since no phase shift results; however, while this can be true for some specific models, it is not a necessity of a limit cycle model in general. Alternatively, the stimuli presented during the dead zone may induce changes of the state variables, but these altered values would move the variables approximately along the original isochron. Consequently, state variables of the oscillator are not necessarily insensitive to the stimulus during the dead zone, as a stimulus could potentially induce large changes of the state variables, which however do not move the oscillator to a different isochron. Moreover, since limit cycle oscillators are nonlinear, incremental increases in stimulus strength may not necessarily be linear. The two different types of PRCs mentioned could be explained in turn by the distinction, whether the state variable is moved beyond the singularity, i.e. the central point from which the isochrons radiate. Type 1 resetting would be expected if there is a small move, not reaching beyond this central point, whereas Type 0 resetting occurs if the stimulus is strong enough to move the variables beyond the singularity. Thus a transition from Type 1 to Type 0 resetting could simply follow from an increase in the magnitude of the stimulus, then pushing beyond the singularity.

The transient time required to reach a stable phase relation would depend on the initial conditions, the entrainment signal, and the properties of the oscillator, but theoretical studies have also found that this transient time is governed chiefly by two basic properties inherent in oscillators: the radial relaxation time and the phase velocity distribution around the limit cycle (Granada & Herzel 2009). Furthermore, the radial relaxation timescale determines the rate of convergence back to the unperturbed amplitude, and the phase velocity distribution determines the waveform of the oscillation. Finally it can be noted, that the longest entrainment time is observed in a sinusoidal limit cycle temporal pattern and when the radial relaxation time is long, a pattern sometimes described as a sloppy, not rigid oscillator. As a final point of note regarding the quality of oscillating systems, it should be pointed out that flexibility and robustness are not necessarily opposing concepts. After all, flexibility is a measure of the degree, to which the rhythmic profiles of the various interacting system components can be varied by modulating either external inputs or biochemical parameters. Robustness, on the other hand, describes the maintaining of a system function, such as the phase of a particular component of the circadian clock, under varying conditions. The mechanisms of flexibility and robustness are thus interlocked in a complex relationship, the precise nature of which will depend on the particular properties of the individual system; in some instances, flexibility may reduce overall robustness by boosting sensitivity to perturbations, but in other cases robustness could even be improved, as the network gains a greater scope to tackle key environmental responses.

1.4.3 Circadian Models across Species

Minimal models

The earliest attempts to model circadian rhythms were not based on the then unknown molecular mechanisms, but generic properties of limit cycle solutions to nonlinear dynamical systems. As it became clear that repression of gene transcription was important for circadian timekeeping, a model was constructed based on Goodwin's negative-feedback paradigm (Goodwin 1965), equations that were first used to model periodic enzyme synthesis in bacteria using a negative feedback loop without autocatalytic terms. Nowadays, the Goodwin model is considered a minimal model for the circadian clock, i.e. it contains only its most important elements.

The equations for the original Goodwin model are:

$$\frac{dX}{dt} = v_1 \frac{K_1^n}{K_1^n + Z^n} - v_2 X \tag{1.32}$$

$$\frac{dY}{dt} = k_2 X - v_3 Y \tag{1.33}$$

$$\frac{dZ}{dt} = k_3 Y - v_4 Z \tag{1.34}$$

Here, clock gene mRNA (*X*) produces a clock protein, (*Y*) which, in turn, activates a transcriptional repressor (*Z*). The latter inhibits the transcription of the clock gene, closing a negative feedback loop. Sustained oscillations can be obtained only by choosing a steep feedback function with a high Hill coefficient, and this constraint exists mainly due to the linear terms used for the degradation steps. Specifically, Gonze and collaborators found that a Hill coefficient of n=8 was effective in obtaining limit cycle oscillations, representing a very high value for how bound ligands induce cooperative affinity. Subsequently, the original Goodwin model was often adapted to include Michaelian kinetics for the degradation steps, and this modification is generally deemed reasonable in circadian clocks, as protein degradation is controlled by phosphorylation, ubiquitination, and proteasomal degradation (Gonze et al. 2005).

$$\frac{dX}{dt} = v_1 \frac{K_1^n}{K_1^n + Z^n} - v_2 \frac{X}{K_2 + X} \tag{1.35}$$

$$\frac{dY}{dt} = k_3 X - v_3 \frac{Y}{K_4 + Y} \tag{1.36}$$

$$\frac{dZ}{dt} = k_5 Y - v_4 \frac{Z}{K_6 + Z} \tag{1.37}$$

In this updated version of the model, limit cycle oscillations can be obtained for a much lower Hill coefficient of *n=4*, much more in line with experimental observations on cooperative binding, and this iteration has further been utilized as the basis for several other clock models. The variable *X* represents mRNA concentration of a *clock* gene, *per* or *cry*; *Y* is the resulting protein, PER or CRY; and *Z* would stand for the active protein or the nuclear form of the protein acting as an inhibitor.

A characteristic feature in the Goodwin and related models is that degradation of clock-mRNA and clock protein species plays an important role in the control of the oscillator's period. Indeed, as predicted by this assumption, experimental results from *Neurospora crassa* indicate that the clock (FRQ) protein of the4 long period mutant frq 7 is degraded only approximately half as fast as the corresponding wild-type protein (Ruoff et al. 1999). The Goodwin model has also been used for stochastic simulations, to simulate and assess the effect of molecular noise (Gonze et al. 2002), and here the deterministic model is decomposed into elementary reaction steps. The oscillations predicted by the stochastic simulations agree with those obtained with the deterministic version of the model, showing that robust circadian oscillations can occur already with a limited number of mRNA and protein molecules, in the range of tens and hundreds, respectively. Furthermore, entrainment by light and cooperativity in repression enhance the robustness of circadian oscillations with respect to molecular noise.

Fungi

The fungus *Neurospora crassa* possesses a comprehensively studied and well understood circadian system. The output is measured in constant darkness as a 22-hour rhythm in asexual spore formation, as well as other circadian rhythms in, for example, metabolism and stress response. While the components of the *Neurospora* clock are not homologous to the other species discussed, the principle of the feedback loops is nevertheless conserved. The central components include the rhythmic gene frequency (*frq*) and the constitutively expressed genes white collar-1 (*wc-1*) and white collar-2 (*wc-2*), which form a heterodimeric white collar complex (WCC) via PAS domains, comprising the positive elements by activating transcription of *frq*. When FRQ accumulates, it inhibits WCC's activation of *frq* transcription, thus closing the negative feedback loop. In addition, FRQ positively regulates expression of WC-1, resulting in a positive feedback loop interlocking with the primary loop, and photoentrainment has been found to occur through the blue-light photoreceptor WC-1. Light-activated WC-1 enhances transcription of *frq* by leading to a slower migration of the WCC complex.

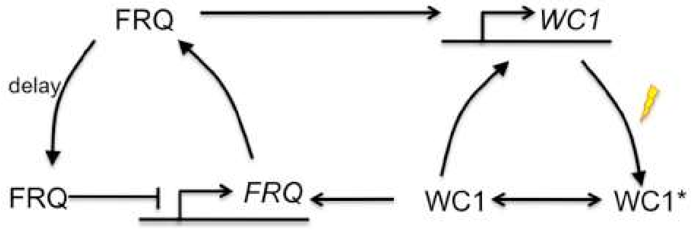


FIGURE 2 Interlocked feedback loop model of the *Neurospora* circadian clock. (Akman et al. 2010) WC-1 is the positive element, while FRQ is the negative element. FRQ also upregulates the level of WC-1, yielding a positive interlocked feedback loop. WC1* represents light-activated WC-1. There is a delay between the translation of FRQ and conversion into its active form.

A stochastic version of a simple Neurospora model further showed that robust circadian oscillations can already occur with a limited number of mRNA and protein molecules, once again in the range of a few tens and hundreds, respectively (Gonze & Goldbeter 2006). Robustness is enhanced by a range of factors, including an increase in the number of molecules, entrainment through light-dark cycles, cooperativity in repression, and also intercellular coupling, whereas the proximity of a bifurcation point leads to less robust oscillations. The binding/unbinding rate of the inhibitory protein to the promoter of the clock gene appears to be crucial for the coherence of circadian rhythms, and it was also shown that multiple interlocked feedback loops increase the flexibility and in turn promote robustness of the clock's rhythmic behaviour (Akman et al. 2010). A loss of free-running rhythmicity observed in a mutant strain was found to arise as a consequence of a supercritical Hopf bifurcation, as coupling strength is altered, and decreasing the bifurcation parameter past a certain critical value collapses the DD limit cycle onto an equilibrium point, with the amplitude of oscillations decreasing continuously to zero as this happens. Alternative mechanisms could be the destruction of the limit cycle through its collision with an unstable limit cycle generated by a subcritical Hopf bifurcation or a saddle node on an invariant circle bifurcation, in which stable and unstable equilibrium points are created simultaneously on the limit cycle. In contrast to the supercritical Hopf bifurcation, the latter two are characterized by a sudden loss of rhythmicity, but without significant amplitude changes at the bifurcation point. In addition, the saddle node bifurcation has a distinct experimental signature in which the rhythm freezes at a well-defined phase as the bifurcation is approached.

Fruitfly

The fly *Drosophila* circadian clock proteins show some homology to mammalian ones, and a group of 20-30 lateral neurons in the adult fly brain have been found to act as a pacemaker. However, rhythmic clock gene expression has also been observed outside pacemaker structures, e.g. in other cells of the nervous system, gut, thorax and abdomen. These oscillations can continue for several cycles in absence of the brain and environmental cues, indicating that peripheral oscillators may function at least as local circadian pacemakers. Period (PER) and timeless (TIM) proteins accumulate and dimerize in the cytoplasm during the day, peaking in the early evening, and subsequently translocate to the nucleus, where they may dissociate. Once in the nucleus, they also interact with the DNA-binding heterodimer CLOCK/CYCLE (CLK/CYC), and since one of the targets of CLK/CYC are the *per* and *tim* genes, PER and TIM thus negatively regulate their own expression. PER and TIM are degraded before dawn, so that CLK/CYC can once again activate PER and TIM, but posttranslational regulation also appears to cause a temporal delay between CLK/CYC transcriptional activation and PER/TIM repression. Several other factors, such as doubletime (dbt), shaggy (sgg) and vrille (vri) refine this system with additional interlocked feedback loops. In order to synchronize the internal clocks to the 24-h cycle of sunlight, *Drosophila* utilize the cell-autonomous blue-light photoreceptor Crytochrome (CRY), which interacts with TIM, promoting its degradation. In constant light, wildtype flies become arrhythmic, while mutant flies lacking cryptochrome as one of the light input factors remain rhythmic.

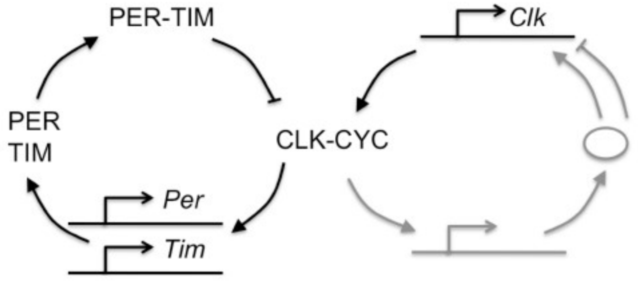


FIGURE 3 Drosophila Model Network diagram

Simplified interlocked feedback loop model of the Drosophila circadian clock as modelled by (Fathallah-Shaykh et al. 2009). PER and TIM proteins accumulate, dimerize, and inhibit CLK/CYC. Several other factors that make up interlocked feedback loops are summarized in grey

The effect of molecular noise was considered in a stochastic version of a ten-variable PER and TIM deterministic *Drosophila* model (Gonze et al. 2003). Namely, the previous model was decomposed into elementary steps and numerical simulations were performed with Gillespie's algorithm. As with the *Neurospora* stochastic model, the predictions of the *Drosophila* stochastic approach agree with those of the deterministic model with respect to both to sustained oscillations of the limit cycle type, as well as to the influence of the proximity from a bifurcation point beyond which the system evolves to a stable steady state. It is further confirmed once more that robust circadian oscillations can emerge at the cellular level even when the maximum numbers of mRNA and protein molecules involved in the oscillations are of the order of only a few tens or hundreds. Interestingly, chaotic behaviour seen in the deterministic approaches is also observed in the presence of molecular noise.

Arabidopsis models

A current *A. thaliana* model has been suggested on the basis of experimental investigation and extensive mathematical modeling, after several plant circadian clock models had previously been constructed in succession, adding on details to keep up to date with experimental data. The first, minimal description of the *Arabidopsis* clock network contained seven coupled differential equations with 29 parameters: LHY and TOC1 mRNA, protein in the cytoplasm or nucleus, and light (Locke, Millar, et al. 2005). MM kinetics were used to describe enzyme-mediated degradation of proteins, and Hill functions to describe the transcriptional activation term of the mRNA for LHY, with LHY and CCA1 being modelled as one gene, since they were considered indistinguishable for the model's purpose. Light input was modelled using a simple mechanism involving an interaction of a light sensitive protein P with the LHY gene promoter: 1 when light was present, 0 otherwise. The essential features are that P is produced only when light is absent and is degraded strongly when light is present. Notably, this model was one of the first clock models to use an empirical cost function to provide a quantitative value for the goodness of fit to essential qualitative features present in experiments. Parameter stability analysis revealed that a small reduction in transcription rate of TOC1 mRNA causes the oscillations to dampen to experimentally undetectable levels after 300 h in darkness. However, as the LHY/CCA1-TOC1 network alone did not account for some aspects of circadian behaviour, such as the long delay between TOC1 transcription in the evening and LHY/CCA1 activation the following morning, the model had to be extended (Locke, Southern, et al. 2005). First, an extra gene, called X, was added to the pathway, a constant light activation term was added to TOC1, and in order to ensure a better experimental fit, the interlocked feedback loop model was extended by an extra loop. Here, a hypothetical gene Y activates TOC1, and TOC1 then feeds back to repress Y. Furthermore, the light input into this loop is moved from TOC1 to Y.

Additional experimental data next lead to a model of three interlocked transcriptional-translational feedback loops (Locke et al. 2006). In the first negative feedback loop, CCA1 and LHY directly inhibit TOC1, while TOC1 in turn upregulates the expression of CCA1 and LHY via a still unknown factor X. In the second, or morning loop, the expression of CCA1 and LHY is inhibited by morning-phased clock components, such as PRR9, PRR7 and PRR5. While single mutant phenotypes are subtle, the prr5 prr7 prr9 triple mutant was found to be essentially arrhythmic, providing the rational for this loop. A further evening negative feedback is formed by an unknown component Y, which positively regulates TOC1 expression and is itself negatively regulated by TOC1, CCA1, and LHY; the evening-expressed GIGANTEA (GI) has been suggested to play a role here. The precise mechanism of how light entrainment is achieved is still unclear, but it may occur via modulation of multiple clock genes at different regulatory levels. Expression of CCA1, LHY, PRR9, and GI, for instance, is induced by light and these are target genes for light resetting, but light also promotes degradation of CCA1 mRNA and increases the translation rate of LHY mRNA, as well as regulating the stability of many clock proteins.

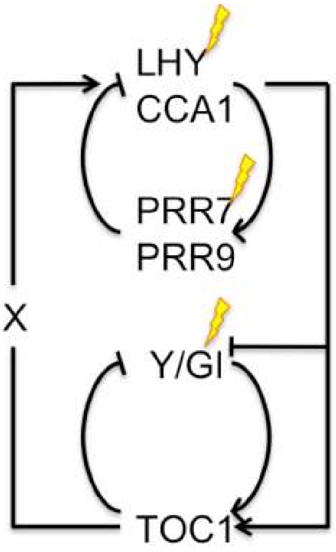


FIGURE 4 Plant Model Network diagram

Three interlocked feedback loop model of the plant circadian clock as modeled by (Locke et al. 2006). CCA1/LHY inhibit TOC1 which in turn activates CCA1/LHY via X. A morning and evening loop are interlocked

A more recent model now further includes PSEUDO-RESPONSE REGULATOR 7 (PRR7) and ZEITLUPE (Pokhilko et al. 2010), and this revised model matches data in varying environments and mutants, and also gains robustness to parameter variation. The results suggest that the activation of important morning-expressed genes follows their release from a night inhibitor, and experiments support the predicted night inhibitor function and implicate a PRR5 gene contribution.

Mammals

The mammalian circadian clock has been intensively studied, mainly with rodents, such as mice, rats, and hamsters as model organisms. In mammals, the circadian clock once again consists of several integrated feedback and feed-forward loops, and many mammalian clock genes have been identified, including bHLH-PAS transcription factors, *Clock* and *Bmal1*, *period* genes, *cryptochrome* genes, and two orphan nuclear hormone receptors Rev-Erbα and Rora. While the mammalian circadian master clock is primarily located in the suprachiasmatic nucleus (SCN) in the hypothalamus and entrained by light through the retina, different peripheral oscillators in other organs and tissues possess endogenous clocks, but are still synchronized by the SCN. The SCN contains about 8000 neurons on each side and, while the circadian rhythmicity is cell-autonomous, rhythmicity in some cells may be driven by rhythmic neighboring cells, i.e. in SCN slices, where tissue architecture is better preserved than in cell culture, a higher percentage of cells are found to be rhythmic. Individual, dissociated SCN neurons, on the other hand, display large variability in period length, and cells are independently phased. Interestingly, each SCN nucleus contains two anatomically and functionally different regions, the ventrolateral and dorsomedial SCN, which are coupled and show different properties during re-entrainment to a shifted environmental Light/dark cycle. It will be of great interest to investigate which functions the distinct subregions within the pacemaker serve. While some peripheral rhythms decline in amplitude within several cycles in absence of the SCN, possibly caused by desynchronization across rhythmic cells, some tissues are able to express persistent circadian rhythms at the tissue level even in absence of the SCN. It might also be the case that peripheral oscillators are synchronized not only by systemic cues, but also by local oscillators. Additionally, it appears that interactions also exist between the SCN and the periphery, with information from outside the SCN having direct effects on the SCN neuronal activity, either phase shifting the pacemaker or attenuating phase resetting by light.

At the molecular level, CLOCK and BMAL1 dimerize and activate, both directly and indirectly, transcription of the *Per* and *Cry* genes through E-box elements. The PER and CRY proteins accumulate in the cytosol and are then translocated, following phosphorylation, into the nucleus, where they form regulatory complexes and inhibit the activity of CLOCK and BMAL1, by binding to the CLOCK-BMAL1 complex. *Bmal1* expression is also subjected to negative autoregulation by BMAL1, through the product of the Rev-Erbα gene. The complex between PER2 and CRY1 or CRY2 enhances *Bmal1* expression in an indirect manner by binding to CLOCK-BMAL1, thereby also reducing the transcription of the *Rev-Erbα* gene. Light can entrain circadian rhythms by inducing the expression of a *Per* gene, though this mechanism needs further investigation, and constant light has been found to desynchronize mammalian clock neurons, while individual neuronal oscillators still have the ability to generate circadian rhythms (Ohta et al. 2005). Therefore, constant light appears to disrupt cellular organization of the SCN clock, thus causing desynchronization among rhythmic pacemaker cells, but does not stop the individual clocks themselves. These findings emphasize that, for proper functioning of the circadian timing system, synchronization and coupling mechanisms within the SCN are indispensable. Interactions also exist between the SCN and the periphery, for example and as in green algae, it was recently shown that rhythmic cycles in activity of peroxiredoxin enzymes can occur without transcription (O'Neill & Reddy 2011). An interesting insight on this link is provided by mature human red blood cells, which lack a nucleus and several other organelles, and are thus unable to perform transcription. However, within them peroxiredoxins can dimerize and these redox transitions were found to occur with a self-sustained approximate 24-hour period that could be entrained by temperature cycles.

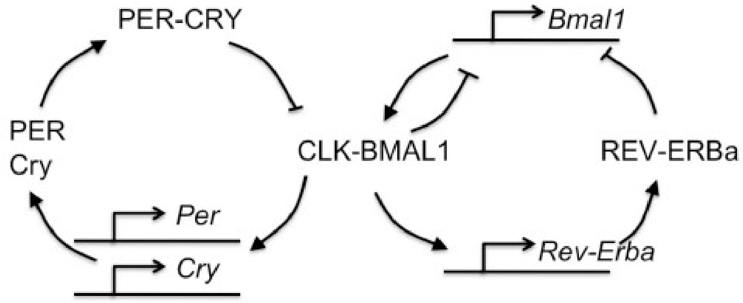


FIGURE 5 Mammalian Model Network diagram
Simplified interlocked feedback loop of a more complex model of the mammalian circadian clock with 19 kinetic equations (Leloup et al. 2003). PER and CRY proteins are phosphorylated and transported to the nucleus (not shown for simplicity), dimerize and inhibit CLK/Bmal1. REV-ERBa compromises a negative feedback loop.

A detailed mammalian model has been proposed (Leloup et al. 2003) and includes the regulatory interactions between the products of *Per*, in several phosphorylation states, *Cry*, *Bmal1*, *Clock*, and *Rev-Erbα* genes. The model is governed by a set of 16 ODEs, or 19 if *Rev-Erbα* is included, and since most parameter values remain to be determined experimentally, semiarbitrary choice of parameter values was used to obtain oscillations. Furthermore, light was included to have an effect on the maximum rate of Per expression in the form of a square wave, and interestingly, when trying to entrain to an LD cycle, the model often gave quasi-periodic oscillations. Entrainment, if and when it occurred, was observed only over a reduced range of the maximum rate of light-induced Per expression, and the reason for this lack of robust entrainment could be traced to the need for a sufficiently high level of CRY protein. Indeed, during the light phase, Per mRNA increases, and as a result, the level of PER protein also rises. If CRY is not present in adequate amounts, free PER will accumulate, because there is not enough CRY present to form a complex with it, and in such conditions, entrainment by the LD cycle fails to take place. In a different, dynamical model for the coupling of a population of circadian oscillators in the SCN, cellular oscillators based on the three-variable Goodwin model are coupled through the global level of neurotransmitter concentration. It was found that global coupling is efficient to synchronize a population of 10,000 cells and that entrainment by a 24-h light-dark cycle can be observed. Synchronization is achieved by the oscillatory component of the mean field, although phases of individual cells are governed by their intrinsic periods and efficient synchronization by average neurotransmitter concentration would dampen individual oscillators. Moreover, simulations of the two regions of the SCN also demonstrate that the driven population can be phase-leading (Gonze et al. 2005).

Circadian Pacemakers

In conclusion, the circadian GRNs of several species have been well defined, even though the importance of post translational modifications for these is only just emerging. Complexity is added to many such systems by the discovery of peripheral oscillators, as well as by the observed responses of circadian pacemakers to stimuli other than light, such as sleep and behaviour. The original view of the central circadian pacemaker has to be consequently updated not just in the mammalian clock to take into account various new insights, such as not only a dominant master pacemaker that is mainly responsive to light, but also the capacity to integrate signals from within and outside the central nervous system. The conventional model of the circadian system, consisting of a linear signalling pathway with a light input pathway, rhythm generating component, and output pathway thus should likely be reconsidered as overly simplistic. Unlike in mammals, where the SCN is required to maintain synchrony among different tissues, the circadian system in other species is also organized in a less hierarchical way. While the isolated peripheral tissues of several species exhibit circadian rhythms in the expression of clock genes, the zebrafish clock shows a high degree of autonomy as peripheral rhythms can be directly entrained by light. This circumstance singles it out as a great model organism to investigate the clock and entrainment pathways.

1.4.4 The Circadian Clock in Zebrafish

Zebrafish as Model Organism

Zebrafish (*Danio rerio*), its name derived from the five horizontal blue stripes on both sides of the body, is a tropical freshwater fish, member of the minnow family, and not least an important vertebrate model organism. The species is believed to have evolved in the Ganges region, and is found not only natively in the waters and streams of the southeastern Himalayan region, but also cultured in home aquariums around the world. Here, especially budding fish keepers value the sturdy nature of zebrafish, allowing to raise it at high density and very low cost, but those are just some of many extraordinary qualities that have destined *D. rerio* as a common and useful biological model system in scientific research, in particular studies into gene function and vertebrate development. In this context, researches profit not only from the aforementioned easy upkeep, but also the relatively short lifecycles, large clutch sizes, and rapid embryonic development of zebrafish.

The species has consequently been used as the basis for several transgenic strains, including a much noted transparent variety (White et al. 2008), was the first vertebrate to be cloned, and is even among the few species so far to have been sent into space. Zebrafish have also been noted for their unusual regenerative ability, allowing them to re-grow fins, skin, hair cells, or heart tissue. In fact, zebrafish have even been found to regenerate photoreceptor cells and retinal neurons, and, moreover, they also display similarity to mammals in toxicity testing and in their diurnal sleep cycle. As a result of this multitude of fascinating qualities, research with *D. rerio* has facilitated new discoveries in cardiovascular research (Major & Poss 2007), developmental biology, regenerative medicine, toxicology (Hill et al. 2005), or also environmental sciences. In oncology, zebrafish have been the basis of several transgenic models of cancer (Liu & Leach 2011), such as melanoma, leukemia, or pancreatic cancer, and zebrafish research investigating the mechanisms of genetic defects is even shedding new light on human musculoskeletal diseases or neurodegenerative diseases.

Qualities Particularly Relevant to the Circadian Clock

Zebrafish being such an important model organism, it may not be surprising that its genome has been fully sequenced, and the zebrafish reference genome sequence was recently published (Howe et al. 2013). There also exists a dedicated online database of genetic, genomic, and developmental information for zebrafish (Sprague et al. 2003), and seeing this wealth of information, studies of gene expression in the species have also lead to the elucidation of several important signalling pathways, for example including the role of Wnt in hair cell repair (Steiner et al. 2014) . Importantly, zebrafish further exhibit several similarities to mammals in the circadian clock makeup, and zebrafish cultured fish organs and embryonic cell lines do not only posses a functional clock, but also a direct light entrainment pathway including photopigments (Whitmore et al. 2000). Notably, no central pacemaker has been found yet and therefore, unlike mammalian cell cultures, which may be affected by lack of SCN input, zebrafish cell lines should give a better representation of clock functioning and especially entrainment in the organism as a whole. In this sense, they practically represent a complete vertebrate clock system contained within a single cell. Furthermore, core clock component transcription output can be readily investigated using transgenic cell lines, in which luciferase reporter gene activity is driven by a clock-regulated promoter, so that addition of luciferin will generate a bioluminescent signal that can be measured in a scintillation counter. Turning to the apparent dynamics of the zebrafish circadian rhythm, the free running period is slightly longer than 24 hours, namely ca. 25 hours in constant darkness and 24.4 hours in dim light (Cahill 2002). Cell lines kept in DD show a dampened rhythm over time, which was found to originate largely due to desynchronization effects across single rhythmic cells that continue to express functional oscillations, but doing so over a wide range of periods. A single 15-minute light pulse is sufficient to reset the cells to a common phase and reduces the range of periods, which would implicate a high amplitude, Type 0 PRC, usually characterized by mainly phase delays and some phase advance around dawn. Sustained light has been found to stop oscillations when the light period begins to exceed 12 hours, but if light is removed the oscillator starts back again from a preserved dusk state. Here, the sustained light induction of *Cry1a* is believed to play a critical role in this light stopping response.

Zebrafish Circadian Clock GRN

Various zebrafish clock genes have been identified to date, including *Clock*, *Bmal*, *period* and *cryptochrome* genes. The GRN resembles that of mammals in many ways, but differences include multiple copies of several clock genes, and the fact that *Per2* and *Cry1a* are regulated by light. In brief, the core clock components constitute an auto-regulatory feedback loop: CLOCK and BMAL1 hetero-dimerize and activate transcription of *Period* (*Per*) (Vallone et al. 2004) and *Cryptochrome* (*Cry*) genes, which in turn inhibit CLOCK/BMAL1. In addition, it was shown that Cry1a is up-regulated by light and may directly interact with specific regions of CLOCK, namely the PAS B domain, and BMAL1, here at the bHLH, PAS B and C-terminal domains. These interactions have been shown to block the ability of CLOCK and BMAL1 to form an active dimer and to initiate downstream transcriptional activation (Tamai et al. 2007). There exists also a stabilizing feedback loop, where Rev-Erbα and Rora direct rhythmic expression of the Clock and Bmal genes. Both, light intensity and the current phase of the clock have been shown to have an effect on the magnitude of Cry1a induction and the resulting Per1 phase shift, and depending on the specific timing of light pulses, light can advance, delay or have no effect on the circadian rhythm, effectively resetting the clocks in asynchronous zebrafish cell cultures to a common phase. Cry1a is a strong clock repressor, meaning that it persists at high levels under constant light and can consequently stop the oscillation system dynamic under LL constant light conditions.

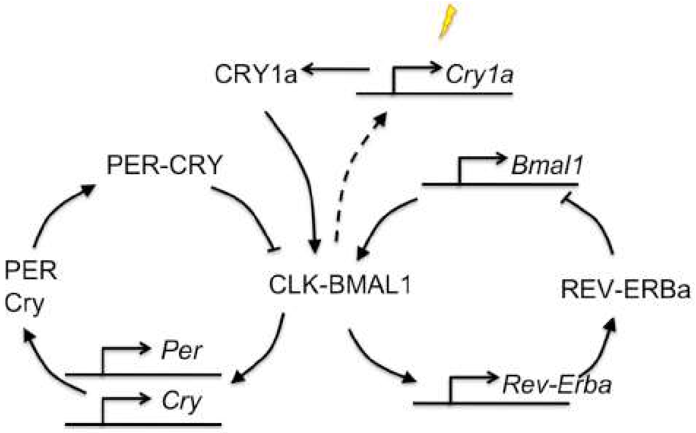


FIGURE 6 Zebrafish Model Network diagram
Proposed network for zebrafish circadian clock. CLOCK and BMAL1 hetero-dimerize and activate transcription of Per and Cry genes, which in turn inhibit CLOCK/BMAL1. Cry1a is the light input to the clock. REV-ERBa might have an effect on fine tuning the clock trough an interlinked feedback loop.

One of the characteristic features of the zebrafish clock, as was touched on above, is the presence of extra copies of the key clock genes. While the number of important clock genes is already nearly doubled from *Drosophila* to mammals, in zebrafish this situation is even more complex due to a genome duplication event that occurred during the evolution of the teleost lineage. Duplicated gene copies may have subsequently been lost during evolution or, in many cases, the extra copies persist. These additional genes may subsequently show redundancy or diverge in function from the original gene, giving rise to slightly different and potentially more specialized copies.

Clock/Bmal genes

Three *Clock* genes, namely *Clock 1a, 1b* and *2*, formerly known as *Clock1, 2* and *3*, respectively, and three *Bmal* genes, namely *Bmal1a, 1b* and *2*, formerly known as *Bmal 1, 3* and 2, respectively, have been found. They display subtle differences in timing of rhythms, e.g. the expression of clock1a is rhythmic in tissues tested, with the exception of testis, with a peak just after the light-dark transition. This is in contrast to mammals, where *Bmal*, but not *Clock*, shows rhythmic mRNA expression. CLOCK and BMAL have been shown to interact pair-wise in various heterodimeric combinations, and these heterodimers display different transactivation properties and susceptibilities to be inhibited.

Per genes

Three *Period* genes have been identified, namely *Per1*, *Per2*, and *Per3*. *Per1* has two homologs, *per1a* and *1b*, the latter also sometimes termed *per4*, and along with per3 its mRNA rhythms peak near dawn, whereas *Per2* is stimulated by light; its rhythmic expression dampens immediately following transfer to DD and thus appears to be an important element of the light input pathway. *Per1* is also known to contain several E-box elements (CACGTG) in its promoter region (Vallone et al. 2004), and these elements seem to play a key role in the circadian clock by forming a binding site for several bHLH transcription factors. Only a subset of E-boxes, termed circadian, which exhibit additional flanking sequences and occur in a group of multiple, randomly spaced E-boxes, seem to act as specific binding sites for Clock-BMAL heterodimers (Link 1 in Fig. 6). *Per2*, on the other hand, was found to possess a Light Responsive Module (LRM) within its promoter (Vatine et al. 2009), which is both necessary and sufficient for light-driven gene expression and also for a light-dependent circadian clock regulation. This LRM sequence is strongly conserved in other vertebrate *per2* genes and contains closely spaced E- and D-box elements. The E-box allows circadian clock regulation through Clock-BMAL activity, whereas the D-box confers light-driven expression through the zebrafish homolog of the thyrotroph embryonic factor (TEF) (Link 3 in Fig. 6). TEF is induced by light, and knocking it down attenuates light-driven transcription from the *per2* promoter in vivo. While predominantly regulated by light, it seems that *tef* expression is also modulated by the circadian oscillator, as in DD *tef* mRNA levels exhibit low amplitude cycling. Furthermore, a study with *period3*-luciferase transgenic zebrafish showed remarkable diversity in oscillator properties, such as period, phase and response to light, in various peripheral organs and tissues (Kaneko et al. 2006). Interestingly, per3 rhythms have been found here to free run in both DD and LL with similar amplitudes, phases, and periods, whereas this behaviour is not the case with mRNA of per2 and per1, which has by contrast been shown to not oscillate in constant light. It is yet unclear if there would be differences to this observation in cell lines, and whether per2 and per1 mRNA would cycle in LL in the organs studied. Three E-boxes have also been found in the upstream sequences of the per3 gene, but not tested for functionality.

Cry genes

Zebrafish posses six rhythmic cryptochrome genes. *Cry1a* and *Cry1b*, which peak during daytime, as well as *Cry2a* and *Cry2b*, which peak later in evening, are all similar to mammalian *Cry1* in sequence and function and can inhibit CLOCK-BMAL dimers. *Cry3*, which peaks in the morning, is the most divergent from other vertebrates and, like *Cry4*, which peaks during day, cannot inhibit CLOCK-BMAL transactivation (Cahill 2002). However, due to sequence similarity to the *Drosophila Cry*, Cry4 function has been implicated as a photoreceptor. While *cry* genes are predominantly clock regulated, *Cry1a* shows a strong light-driven pattern of expression and appears to represent a key element of the mechanism underlying entrainment by light and also the maintenance of high amplitude cycling. A *Cry1a*-luciferase reporter cell line was generated and used to investigate the light induction of *Cry1a*, and how this induction correlates with the magnitude of Per1 phase shift and light intensity at different times during the day (Tamai et al. 2007). The PRC shows the largest shift at CT20, that is late subjective night, causing a 15-hour shift, while at CT4, the early subjective day, there is almost no phase shift observed. In LL, there is an increase of *Cry1a*, and

Per1 is highly repressed. If light is sustained for longer than 12 hours, new monomers of CLOCK and BMAL1, which are ordinarily formed in the late afternoon, cannot dimerize due to the presence of CRY1a and thus the oscillator is stopped. Similarly, *Cry1a* overexpression abolishes rhythmic expression of *Per1* and significantly reduces basal levels in a dose-dependent manner, thus mimicking the effect of light.

A yeast two-hybrid system was used to further test the interactions of CLOCK and BMAL and to identify specific domains involved in their protein binding. It was reported that CLOCK1 and BMAL1 interact strongly at the bHLH and PAS B domains, with little or no binding between the two PAS A domains. *Cry1a*, on the other hand, binds strongly to the PAS B domain of CLOCK1 and to multiple regions of BMAL1, including the bHLH, PAS B, and C-terminal transactivation domains. It has also been shown to interact with CLOCK3 and BMAL3, but not CLOCK2 or BMAL2. Cry1a interferes with CLOCK:BMAL by inhibiting transactivation directly, binding to the C-terminal domain of BMAL, and furthermore by preventing formation of active dimers, competing for the bHLH and PAS B domains. However, if the dimer has already formed, Cry1a has little effect. In summary, Cry1a can interact with key regions of the Clock and Bmal activators, thus preventing their heterodimerization and hindering their ability to transactivate from E-box elements (Link 2 in Fig. 6). It seems that the induction of *Cry1a* is additionally mediated by a light activated MAPK pathway, which was also linked to DNA repair via the gene z64Phr, (Hirayama et al. 2009), and moreover, there is some evidence that non-canonical clock genes could contribute to the circadian expression of *Cry1a* gene in a cell autonomous manner (Miyamura et al. 2009).

<u>Rev-erbα</u>

Rev-erbα is a ubiquitously expressed orphan nuclear receptor, which functions as a constitutive transcriptional repressor and is expressed in vertebrates following a robust circadian rhythm. Two Rev-erbα mRNA isoforms, Rev-erbα and Rev-erbα 2, are generated through alternative promoter usage and both display a circadian expression pattern

(Triqueneaux et al. 2004). The promoter regions contain several E-box elements, and CLOCK-BMAL1 has been demonstrated to regulate Rev-erbα. This regulation is conserved in vertebrates, and has also been confirmed for the zebrafish Rev-erbα (Link 4 in Fig. 6).

In summary, the zebrafish circadian clock GRN resembles the mammalian one in several important ways, although some duplicated zebrafish clock components have evolved to carry out specialized functions. A diagrammatic representation is shown in figure 6.

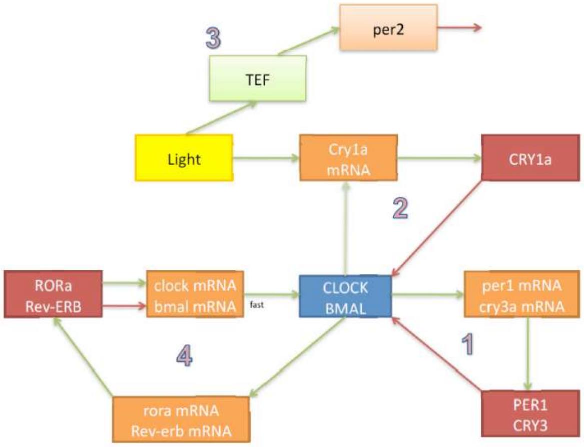


FIGURE 7 Proposed GRN for zebrafish circadian clock.
CLOCK and BMAL1 hetero-dimerize and activate transcription of Per and Cry genes, which in turn inhibit CLOCK/BMAL1. Cry1a is the light input to the clock. REV-ERBα might have an effect on fine tuning the clock trough an interlinked feedback loop.
1: Per1 contains several E-box elements in its promoter region (Vallone et al. 2004) that allow regulation of Clock-BMAL heterodimers.
2: Cry1a is up-regulated by light and may directly interact with CLOCK-BMAL dimer formation. (Tamai et al. 2007).
3: Per2 is light responsive trough the action of TEF (Vatine et al. 2009).
4: CLOCK-BMAL1 can regulate Rev-erbα (Triqueneaux et al. 2004).

Light Entrainment in the Zebrafish Circadian Clock

In the zebrafish circadian clock, entrainment occurs primarily in response to light, with exposure triggering photoreceptors, their coupled signalling pathways, and finally a set of clock genes, namely *per2* and *cry1a*. As mentioned, the clock shows varied sensitivity to resetting cues, i.e. depending on the time of day, light causes phase advance, delay or has no effect. Moreover, the resetting efficiency also correlates with the level of *Cry1a* upregulation. In a recent study, zebrafish larvae, heart organ cultures and cell cultures were light pulsed or kept in DD to examine light induced changes in gene expression (Weger et al. 2011). It was found that of the 117 light regulated genes, the majority were induced and some repressed by light, and that these genes are involved in circadian clock function, stress response and DNA repair, retinal light reception and metabolism. Promoters of upregulated genes revealed an enrichment for E- and D-box binding sites, indicating that light induction of these genes was similar as in the *per2* gene. It is also important to note, that the exact entrainment response of the oscillator varies according to different light regimes. A single light pulse can, as previously mentioned, shift an asynchronous population of clock cells to a common phase of the circadian cycle, equivalent to the early day or ZT 4, and consequently, if cells were close to ZT 4, only a small phase shift and modest increase in Cry1a would be necessary for entrainment. In the late night, however, a much larger phase shift and higher level of Cry1a induction would be required. In LL *cry1a* is induced, while *per1* is highly repressed until the light is removed, at which point the oscillator restarts again from dusk, at around ZT12, and indeed the circadian oscillator appears to be held motionless at about CT12, when the day length begins to exceed 12 hours. Curiously, light pulses also lead to the acute induction of Per1, which occurs before the increase in Cry1a levels and subsequent repression in Per1 expression. It has been described that it takes approximately 3 hours for Cry1a to reach peak transcript levels following light exposure, during which time a transient increase in Per1 is observed, but the mechanism and potential role of this transient Per1 increase are not yet understood.

A single light pulse every 24 hours was shown to mimic LD in per1 peak, timing and waveform, although the timing of the trough does not match, and the rising phase is advanced slightly on each entraining cycle; moreover, Cry1a also phase advances each day, and when the medium is transferred to DD, an aftereffect is observed with the FRP being about 3-4 hours shorter than expected. In a two-pulse light regime, also known as a skeleton light cycle, *per1* rising phase appears more accurate, despite an acute transient induction of *per1* at the second pulse, making the traces less clear. The second, or "dusk", light pulse also appears to generate a phase delay in the rhythm, as predicted by the shape of the PRC, and overall, phase and period are more similar to LD compared to a single pulse. The amplitude, however, is significantly smaller, probably due to the lack of extended repression of *per1* during the day, which would be key to high amplitude oscillations. Here, the FRP in DD is comparable to cells entrained in LD, which could point to the presence of a morning and evening oscillator. Cry1a furthermore displays two smaller peaks as it is up-regulated at each light pulse (Tamai et al. 2007). Turning to a study with single-cell imaging, it was revealed that cells in LD show a robust rhythm with a high level of synchrony. In DD, there are still oscillations, but now the peak levels are distributed throughout the day with widely varying phases and marked stochastic fluctuations in FRP. This effect is greater when the cells are kept in DD for several months, providing evidence that for populations left in DD, single cells may still exhibit a functional oscillator, even if the global signal averages to a non-oscillating flat level due to divergence and desynchronization. However, a light pulse succeeds in shifting the phase of individual cells to become synchronized again and hence stabilizes the subsequent FRP (Carr & Whitmore 2005). In this regard it appears that even clonal cells may exhibit significant deviation, but which is ordinarily compensated for by the mechanisms of exceptional light responsiveness, yielding an overall accurate timing mechanism.

As a final note, it is intriguing to note that light regulation of gene expression in zebrafish may not be reserved for the circadian clock, but rather appears to also play a role in the repair of DNA damaged by radiation, such as UV light. Specifically, the gene encoding the DNA repair enzyme 6D4 DNA photolyase is not only closely related to the Cry family, but was also shown to be light-inducible, with a proposed mechanism similar to the one of cry1a directing its mRNA expression. Moreover, light appears to regulate not only the transcription, but also the activity of this repair enzyme, and both zebrafish larvae and cell lines cope better with UV radiation, when also being exposed to light. With reports of yet more light-activated gene expression in the zebrafish transcriptome, it appears plausible that significantly more aspects of cellular function in zebrafish may be regulated by light exposure than currently documented (Vatine et al. 2011).

Aims and Objectives

This project was envisioned with a broad view towards elucidating on one hand the underlying dynamics of the circadian clock in zebrafish, but also more generally the mechanisms of how GNRs may be regulated or entrained through signals originating from outside their network. Integrating this guiding motif with new insights gleaned from an extensive review of the existing literature, and especially the emerging view of the importance of noise in shaping the behaviour of cellular functions, the following aims and objectives are set out for the investigation at hand.

Aims

- to construct a functional oscillator model of the GRN underlying the zebrafish circadian clock, adapted and extended in order to facilitate simulations and improved understanding of the entrainment effects observed experimentally under different light regimes
- to furthermore evaluate the validity of stochastic approaches, comparing and contrasting their output with deterministic simulations, in explaining signal dampening and entrainment effects as functions that are critically shaped by the noisy nature of cellular processes

Objectives

- A first deterministic model will be constructed as a system of linked ODEs on the basis of the known circadian core system in Zebrafish, namely describing the dynamic interactions of CLOCK, BMAL1, CRY1A, and PER1 using appropriate kinetic functions, and simulations will seek to establish the occurrence of stable oscillations within this core network representation.
- An input pathway for the effect of light will be defined, primarily acting through CRY1A, and integrated with the deterministic model. Computation runs will subsequently be carried out over a range of simulated light conditions in order to fine-tune the dynamics and strength of this input, and to compare results to those suggested by the scientific literature.
- Select laboratory experiments will be planned and implemented to generate additional reference data used in guiding refinement of

above model, in particular pertaining to the effect of light pulses of varying intensities and durations in effecting phase shifts and entrainment in populations of zebrafish cell lines.

- Automated computational functions will be implemented to transform and analyse, e.g. using Hill Transforms and related techniques, the signals generated in both laboratory and simulated experiments. Moreover, summary statistics, such as period, relative amplitude, or phase, are going to be defined and utilized for a quantitative description of the data generated.
- Considering the difficulty and possible bias inherent to selecting appropriate parameters manually, a probabilistic and self-improving tool will be selected from the range of existing approaches, such as Approximate Bayesian Computation (ABC) or Markov-Chain Monte-Carlo (MCMC), and utilized to improve the effectiveness of this critical step.
- Different stochastic implementations will be contemplated, including for example stochastically shifted deterministic curves, Gillespie algorithms, or SDEs, and potentially utilized as a basis for re-implementing the deterministic ODE model. In particular, it will be evaluated whether noise terms succeed in demonstrating the mechanism of natural de-synchronization over time and re-synchronization under the influence of light, as suggested by findings from single cell observations as a likely avenue for entrainment effects.
- Finally, with the models and tools described above in place, it will be possible to readily adjust the simulation system by adding in e.g. the stabilizing Rora loop, or by varying the target or mechanism of light signal input. These adaptations should help elucidate the possible significance of adding complexity to the model, while also demonstrating the usefulness of the model environment as a basis or supporting tool for further investigations.

Chapter 2: Creating the Core Model and Analytical Tools

2.1 Building an Initial Model of the Zebrafish Circadian Clock

2.1.1 The Underlying System of ODEs

One of the first steps towards investigating the dynamics of entrainment and noise in the circadian clock consists of constructing an initial model of the clock in zebrafish as a set of linked ODEs, representing the concentration changes of the different core molecular species of this GRN.

Zebrafish Circadian Clock as a System of ODEs

It has been described above, how representing the various molecular components of a GRN, or rather the changes in their concentrations, as a system of ODEs is a well-established and highly useful modeling approach, which combines feasible resolution speeds with the mostly accurate handling of even very dynamic systems, while also readily pointing to stationarities and other points of interest. Looking to "translate" the zebrafish circadian clock into such an ODE system, it has been noted that the core clock components constitute an auto-regulatory feedback loop, with CLOCK and BMAL1 hetero-dimerizing and activating transcription of Per and Cry genes, which in turn inhibit CLOCK/BMAL1. Moreover, negative feedback has been identified as essential in biological oscillators, carrying the network back to its starting point, while a sufficient delay ensures that reactions do not settle on a stable steady state. Consequently, a first sketch of the zebrafish clock is designed around two interlocked negative feedback loops; the first one consisting of the Clock-Bmal heterodimer and *Per1*, while the second one features Clock-BMAL and *Cry1a*. The equations for the five ODEs are:

$$c1am' = v_1 \frac{ClkBmal}{k_a + ClkBmal} - k_{dm} \frac{c1am}{k_{dg} + c1am} + light \tag{2.1}$$

$$cry1a' = k_1 * c1am - k_d \frac{cry1a}{k_{dg} + cry1a} \tag{2.2}$$

$$ClkBmal' = k_2 \frac{k_{c1}^n}{k_{c1}^n + cry1a^n} + k_3 \frac{k_{c2}^n}{k_{c2}^n + per1^n} - k_d \frac{ClkBmal}{k_{dg} + ClkBmal} \tag{2.3}$$

$$p1m' = v_2 \frac{ClkBmal}{k_p + ClkBmal} - k_{dm} \frac{p1m}{k_{dg} + p1m} \tag{2.4}$$

$$per1' = k_4 \cdot p1m - k_d \frac{per1}{k_{dg} + per1} \tag{2.5}$$

The Clock-Bmal heterodimer is denoted by *ClkBmal*, Cry1a mRNA and protein by *c1am* and *cry1a*, respectively, and Per1 mRNA and protein by *p1m* and *per1*, respectively. Moreover, a simple light input mechanism is already included in the form of the term "light", which acts as a simple constant addition to Cry1a mRNA levels, but was initially set to 0.

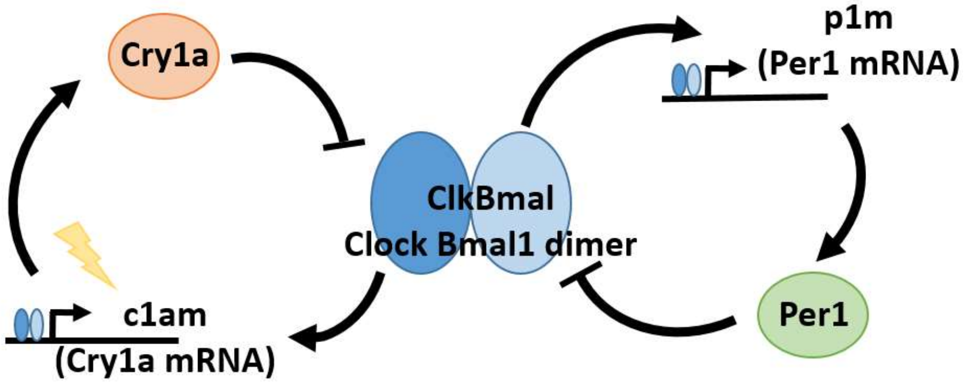


FIGURE 8 Zebrafish Model Network diagram corresponding to the model
Graphic representation of the zebrafish circadian clock model. CLOCK and BMAL1 hetero-dimers activate transcription of Per and Cry genes, which in turn inhibit CLOCK/BMAL1. Cry1a is also upregulated by light.

Practical Implementation and Integration Steps

In order to solve these linked ODEs numerically, they were transcribed into the Mathematica computational software program, developed by Wolfram Research, using Mathematica's own specific programming language. The exact code can be found in the appendix, but principally a standard configuration of the function "NDSolve" was utilized. This function typically aims to find solutions to differential equations by dynamically determining, depending on the nature of the problem to solved, a set and order of methods to employ, which would generally centre on a time integration process for a system of differential equations, and the setting of boundary values. Other steps may include the simplification of equation form, the processing for handling of discontinuous differential equations, or also discretization and symbolic index reduction steps for PDE and differential algebraic equations, respectively. The core numerical solver, specifically, is reported to be based on a multi-step Adams method (Baumgartner et al. 2006), an expansion of the more familiar single-step methods, such as Euler or Runge-Kutta. Conceptually, all these numerical methods utilize initial points, starting from which they try to undertake a small step forward in time to pin down the next solution point, thereby following an iteration of small time steps to map out the complete solution. However, single steps processes, such Euler's method, are based on only one previous point and its derivative in order to localize the current point, whereas other methods, such as Runge-Kutta for instance, further include some half- or other intermediate steps to obtain a higher order method for calculating the next solution value. Multistep methods, on the other hand, are set apart by retaining and utilizing information from several previous points and derivative values, which they may combine in linear or non-linear fashion to gain efficiency in their computations. Furthermore, multi-step methods are usually credited with producing less error than single-step methods due to their multiple initial points, but the precise selection of a solver method remains depending on a variety of factors, importantly including the equation's stiffness; this quality may be roughly summarized as a measure of the solution's numerical stability, particularly when operating outside of extremely small step sizes. An example of a multi-step method would be:

$$y_{n+1} = y_n + h(b_0 f(t_n, y_n) + b_p f\left(t_{n-p}, y_{n-p}\right)$$

where h refers to the step size, while the precise method is determined by the coefficient $b_0, \dots, b_s$, the values of which, even if they are regularly set to zero, are critically important for balancing ease of use versus a faithful approximation to the true solution. The three most commonly used families of linear multistep methods are Adams-Bashforth, Adams-Moulton, and backwards differentiations formulas, with the former two both going back to the work of 19th century British mathematician and astronomer John Couch Adams, who is further famous for the purely theoretical prediction of

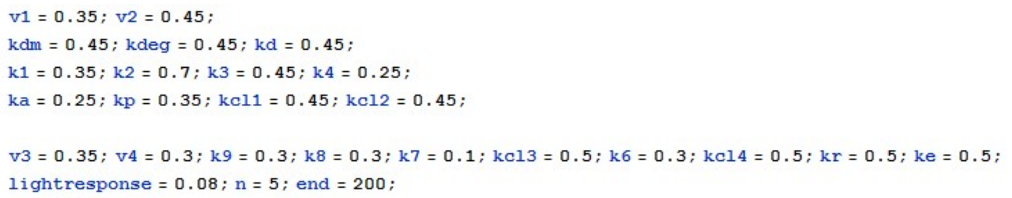
Neptune's position and, in fact, mere existence. The Adams-Bashforth and Adams-Moulton methods both share a characteristic coefficient pattern and order maximization, but the former is an explicit method, predicting the system's future state purely based on the current one, while the latter method is implicit in nature, utilizing both current and future state in determining a solution. In either case however, as has been clearly stated, the Adams methods do not evaluate derivative functions by simply working with points close to the solution value, as Euler, Taylor, or Runge-Kutta methods would, but instead also includes interpolation based on old solution values and derivatives. It follows, that continuity of the function and stability of the numerical solution with respect to perturbations of initial points can be considered critical here, and only those linear multistep methods exhibiting "zero-stability" for a certain differential equation, that is the ability to contain the growth of a perturbation, would reliably converge to the exact solution.

In order to conduct an initial survey of the potential effects of the light term on not only the levels of Cry1a mRNA, but also the behaviour of the system as a whole, the corresponding input was applied at a level expressed by a separately determined parameter of light intensity via a "Piecewise" function, triggering for Sin(t/4) > 0 and set to 0 otherwise, thus mimicking a regular, if simple, light/dark cycle. Not surprisingly, the system would exhibit stagnating concentrations for the vast majority of randomly selected parameter values, and consequently parameters were manually selected and adjusted. Within the limitations of this approach, it is found that stable oscillations can be achieved if the Hill coefficient is 4 or higher for the activation of Clock-Bmal by Cry1a and Per1, and further if degradation rates are Michaelian rather than linear decay rates.

Determining Initial Parameters

However, oscillations do not automatically display experimental phase relations, but these can be replicated through careful further adjustment of parameters. Similarly, the required initial values were first randomly selected and subsequently manually adjusted to appear in line with the relative concentrations observed for the different system constituents over several test runs. Specifically, the parameters and initial values used are summarized below.

FIGURE 9 PARAMETERS

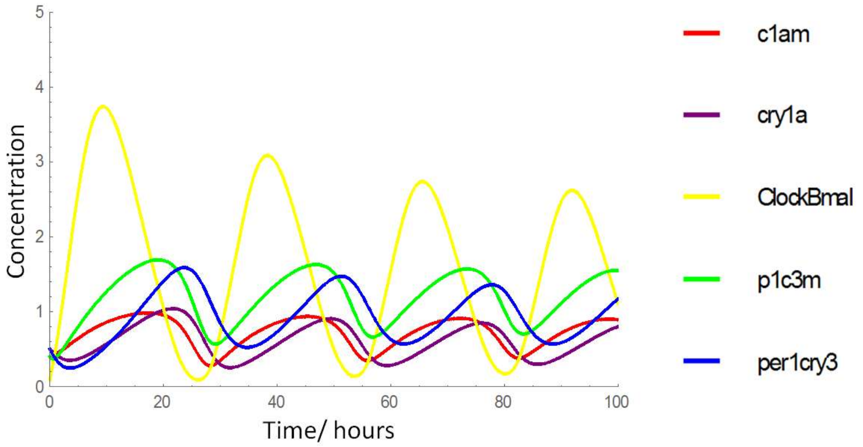


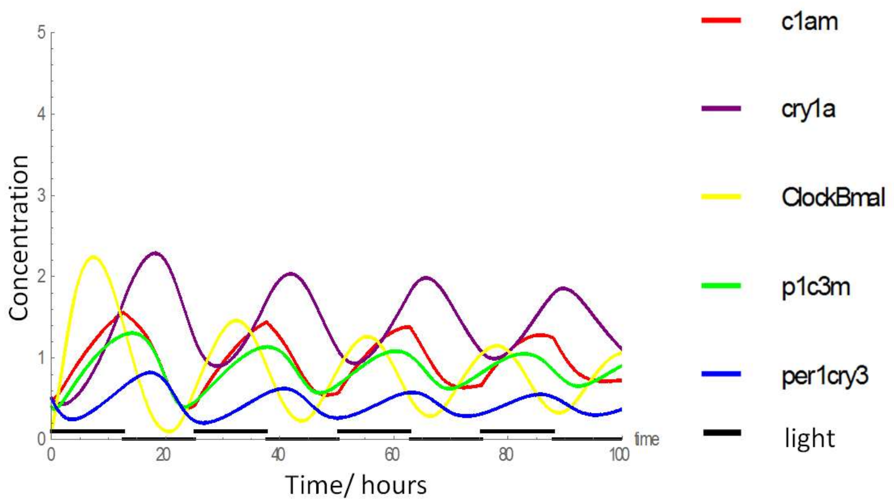
```
v1 = 0.35; v2 = 0.45;
kdm = 0.45; kdeg = 0.45; kd = 0.45;
k1 = 0.35; k2 = 0.7; k3 = 0.45; k4 = 0.25;
ka = 0.25; kp = 0.35; kcl1 = 0.45; kcl2 = 0.45;

v3 = 0.35; v4 = 0.3; k9 = 0.3; k8 = 0.3; k7 = 0.1; kcl3 = 0.5; k6 = 0.3; kcl4 = 0.5; kr = 0.5; ke = 0.5;
lightresponse = 0.08; n = 5; end = 200;
```

A graphical representation of sample simulations with parameters and initial values selected as outlined above can be seen in Figure 10 and clearly demonstrates the capacity of the system to exhibit oscillating behaviour. The integration interval was set to 100, to be in line with laboratory experiments commonly carried out on this one-week scale. It should also be noted, that the oscillations appear to be clearly dampening. While decaying oscillations under constant darkness are a feature of circadian oscillators to be possibly achieved through extensions of the model, it should be noted that in this early iteration the observed decay is unintentional, and can almost certainly be attributed to poorly calibrated parameters and initial values. Subsequent implementations consequently feature specific automated initial value and parameter correction algorithms, with early runs showing stable oscillations over ten thousands of cycles and under signal analysis via Hilbert transform.

FIGURE 10 INITIAL MODEL RUNS WITH AND WITHOUT LIGHT INPUT





Notably, a distinct decay in oscillation strength can be observed, which can be likely attributed to insufficiently calibrated initial values, thus starting the oscillations off with an extra momentum that is gradually lost over subsequent periods. Moreover, it can be gleaned that light appears to slightly alter the shape of the output curve, while also expectedly increasing the amplitude of oscillations. These simple observations appear to encourage the use of the presented network as a starting point for more detailed investigations, even if various additions and refinements are necessary to verify and enhance the predictive power and theoretical insights presented by the model.

2.1.2 The Significance of Fine-Tuning the Model

It has been described for a wide variety of areas, ranging from theoretical physics to economics and biochemical systems, that models may have to be fine-tuned, that means their parameters adjusted very precisely, in order to bring their output in line with observations. While there is some ongoing discussion about the underlying origin and justification of this necessity, with different sides contributing arguments drawing on various naturalistic and anthropic principles, at least in the case of biological systems an explanation is more readily apparent. After all, it has been described how individual processes may, via dynamics represented by Michaelis Menten or Hill kinetics, be very sensitive to concentration changes around a very specific range; and evidently the need to predict these threshold values becomes all the more acute, when representing the emergent behaviour of a system relying on several of such hypersensitive elements. For instance, it was hypothesized in the discussion of the preliminary results, how even a modest misestimation of initial values could have given rise to an uncharacteristically strong first circadian cycle, with subsequent ones experiencing a significant signal decay. This possibility is especially noteworthy when considering that the predefined initial values are only directly utilized by the numerical solver in the very first one, or few depending on the algorithm, of many hundred seemingly affected time step calculations. Of course, the significance of the equation parameters referenced at every single calculation step would arguably be much greater still, highlighting that slight initial dislocations may not only persist, but actually be propagated and inflated over time. Consequently, already small variations in parameter values can give rise to significant behavioural variability, pointing to the need for a thorough and methodical parameter estimation procedure when evaluating the dynamics and merit of a particular model. Moreover, considering next the obviously beckoning questions of precisely what quality to optimize parameters for, the task of creating a fit with experimental data can be surprisingly multifaceted. After all, a "simple" readout could be broken down into dozens of primary observable features, which may be further refined by statistical analysis into a plethora of system properties, such as maximum or minimum values interpreted as either absolutes or as standard deviations, average values understood as either means or medians, signal intensities measured as peak amplitudes or as integrals of signal strength over time, etc.

Summary Statistics for the Circadian Clock

At least in the case of the circadian clock, thankfully several main descriptors are relatively well established, and these include the periods, phase relationships and peak amplitudes of, and between, the oscillating concentrations of different core clock proteins and mRNA. Arrays of data exist for various experimental setups and measurement routines recording these values, and while not necessarily perfectly consistent, this trove proves very valuable for training model systems. Nevertheless, it can still be worthwhile to additionally implement experimental runs especially attuned to particular aspects of the simulation objectives. Not only does this provide an opportunity to generate very specific conditions and readouts, which may uniquely boost the development and hence accuracy of the desired model properties, but moreover, experimental data thusly generated may also help to bring into better perspective the experimental frameworks and readout conventions employed elsewhere. For instance, it may be found that lengthy exposure to constant darkness may affect the subsequent observable behaviour under different light regimes, and insights such as this one may consequently help to select and sort existing data sets so as to better compare "like with like". Yet another, critically important aspect for testing simulation outputs with results from laboratory experiments refers to the quantification of data. While there are instances where an intuitive description may be potent in sorting for important qualities and/or may be difficult to supersede quantitatively, e.g. when sorting curves by a complex shape or pattern, it is generally considered more rigorous, reliable, and reproducible to express results along a numerical spectrum. Even in instances where this quality is poorly provided by raw experimental readouts, it is often possible to utilize mathematical techniques to extract from even relatively complex data certain key summary statistics, which may then be readily compared and contrasted across different cohorts, conditions, or investigative settings. In the context of analyzing oscillation, and indeed a wide variety of signals, various signal transforms are frequently employed, including for instance Fourier, Hilbert, or Wavelet transforms, to decompose the signal and readily extract different underlying qualities, most importantly periods, phase, or amplitude. It can be further noted, that the techniques can in principle be applied to simulated data as easily as to laboratory results, opening up the perspective of largely automated programmed functions that reference key dynamics of the model system to established observations. In addition, techniques such as sensitivity and bifurcation analysis can be employed on this basis *in silico*, monitoring the effect on key statistical values of cycling one or several parameters across a possibly wide parameter space. Once again, these approaches can provide a fresh and important outlook on the viable range of parameter values, but work best when being centered on an evidently functional system as a "gold standard".

Capturing Underlying Dynamics

Finally, having glanced over different challenges and approaches for fitting the model behaviour, it should of course be pointed out that this goal is not intended as an end in itself, but rather serves to evaluate and establish the validity of the suggested underlying dynamics. Once these interaction steps are accepted as sufficiently faithful representations of the system behaviour under scrutiny, the advantages of the modeling approach truly begin to shine, allowing easy manipulation of the system over a wide range of simulated condition, on vastly accelerated or decelerated time scales, and all while providing life readouts of not only phenotypic characteristics, but also the dynamics themselves. Once again, it is hoped that such a finely attuned system would permit novel insights into the nature of entrainment in the Zebrafish Circadian Clock and GRNs more generally. The following sections will provide a description of the laboratory experiments carried out as part of this project, as well as of the program tools that are being implemented to analyse and compare corresponding results.

2.2 Laboratory Experiments

2.2.1 The Use of Bioluminescence Reporter Genes

Following on from an initial foray into running a simulated Zebrafish Circadian Clock network, a set of laboratory experiment is envisioned and implemented to investigate the entrainment effect of exposing zebrafish cell line populations to single light pulses. In order to detect any resulting changes in gene expression levels, a common investigative approach in molecular biology centers on the use of embedded reporter genes. Here, a gene that is not natively expressed by the cell line or organism of interest is inserted in the form of specifically designed DNA constructs, e.g. for cultured prokaryotic cells or bacteria oftentimes a plasmid circular DNA segment, usually directly into the intracellular space.

Different Approaches to Using Reporter Genes

Different specialized variations and applications exist for this general principle, including the use of reporter genes in transfection or transformation trials, where they can be utilized alongside a target gene that is to be inserted into an organism's genetic code. The procedure is only effective in a modest percentage of the trial population, and expressing the reporter gene constitutively or inducibly, so that it is always active or can be switched on by a trigger, can consequently serve as a crucial indicator to help identify those targets, where the gene of interest was successfully transferred. Here, the use of an independent promoter can allow for detection of the reporter irrespective of target activity, which may be desirable if the activation conditions for the latter are uncertain or laborious to bring about. A second important use of reporter genes consists of attaching them directly to a target gene in a gene fusion approach, signifying that the reporter will be transcribed, under the action of the same promoter, into a single mRNA molecule alongside the target gene. Provided that both resultantly linked elements of the translated amino acid chain are able to fold into their active protein conformations, which is often facilitated by linking the two active parts with a flexible polypeptide linker region, the procedure effectively gives rise to a "double-headed" protein. Such a molecular chimera can be highly useful for precisely tracking the movement and activity levels of target proteins, the role of which is as of yet poorly understood. Yet another, also highly important use of reporter genes lies in assaying the activity of a particular promoter. By simply placing them under the control of this promoter of interest, their expression levels can be quantitatively detected, often put into perspective relative to the strong gene expression levels encountered in some consensus promoter, and taken as a good approximation for the expression timing and levels of the genes that are normally induced by the same promoter. Throughout the various approaches of gene reporting, it is evidently critical that the output of the chosen reporter gene can be readily identified and measured. Routinely genes are selected that equip its target organism with clearly distinguishable features, such as bacteria with chloramphenicol acetyltransferase genes thriving on media with the otherwise antibiotic chloramphenicol, or cultures changing the colour of their substrate due to the effect of inserted beta-galactosidase, but another important group of reporter genes selected for visually identifiable characteristics encode fluorescent and luminescent proteins. Examples include the red fluorescent protein dsRed, green fluorescent protein, or also the group of luciferase enzymes.

Luciferase as a Key Reporter

These oxidative enzymes occur in diverse organisms ranging from mushrooms, to marine creatures and, most famously, fireflies, and play a central role in the process of bioluminescence by catalyzing variety of light-emitting reactions. Looking to fireflies in particular, there are over two thousand known species, many with their own versions of luciferase so distinct, they are considered a useful criterion in molecular phylogeny, but one particularly well-studied example is the Photinini firefly *Photinus pyralis*. Photinini luciferase is widely used as a laboratory reagent, characterized by an optimum pH of 7.8, and its catalytic function converts luciferin into luciferyl adenylate, and further into oxyluciferin in an electronically excited state, and light is emitted as oxyluciferin returns to the ground state by releasing a photon. The light emitted by luciferases via this mechanism can vary between yellow-green to red, with wavelengths ranging from 550nm to 620 nm, but constitutes a high signal, which predestines them for use in high throughput screening applications, and has helped to turn luminescent reporter gene assay (LRGA) into one of the most prominent types of reporter gene assay, valued particularly in the fields of pharmaceutical development and molecular biology for its sensitivity and reliability (J. Miraglia et al. 2011). For instance, it was recently described how a CRE-luciferase reporter gene can be employed to easily detect the activities of G protein-coupled 5-HT receptors, which in turn are valued as potential targets in antipsychotic drug discovery (Chen et al. 2015). There have also been various technical advancements to improve signal stability, and newly developed forms of luciferase include red shifted variants, to reduce the absorption effect of short-wavelength photons by biological tissues, leading to diminished sensitivity at non-superficial locations (Loening et al. 2010).

2.2.2 Overview of Different Experiments Conducted

For the light pulse experiments presented here, a period1-luciferase zebrafish cell line was used to monitor gene expression of per1, and hence progression of the circadian oscillator, and the details of its creation can be reviewed in (Vallone et al. 2004). As a short overview of the reporter cell line's mechanism, it can be summarized that the enzyme luciferase is synthesized when transcription is activated by the promoter of per1, and this enzyme subsequently interacts with the substrate luciferin, which can be added to the medium, to release light by the process of bioluminescence. This bioluminescence can then be systematically detected and measured as counts per seconds (CPS). While it is also theoretically possible to take measurements at a single cell level, the experimental setup and bioluminescence detection is much more challenging, as a single cell produces relatively few photons (Welsh et al. 2010). Accordingly, most bioluminescence experiments utilize populations of cell lines, which can furthermore be useful for readily comparing different implementations with each one holding a specific clock reporter gene construct, thus allowing a look at various transcriptional activities with high time resolution.

Experimental Parameters

Here, approximately 25x10^3 *per1-luciferase* cells per well were plated in quadruplicate wells of a 96-well plate in media containing 0.5 mM beetle luciferin. Test were carried out over a range of light intensities, and for each light intensity one separate plate was used, with all plates being kept in a dark incubator for 5 days before data recording. Light pulses were performed either after 24 hours of continuous readings for a duration of 15 mins, or after 48 hours of continuous readings for a duration of 60 mins, and each timing format was repeated with intensities of 0.1, 1 , 10, and 1000 μW cm^{-2}, where a short duration and low intensities were chosen to determine what amount of light may be sufficient to cause an effect. The wavelength spectrum was 400-700nm, and assuming a mean wavelength of 520nm, this range of irradiance can be calculated to correspond to a photon flux ranging from 0.0043 to 43μmol m^{-2} s^{-1}.

Estimation of Photon Flux

Photon flux refers to the number of photons impacting a reference area, usually expressed as 1 m^2, every second, and this concept is increasingly considered more accurate and meaningful than simple irradiance readings when investigating light stimulation, also and especially in biological and biochemical settings. Here, the light intensity per cm^{-2}, which can be expressed as Watt or equivalently as Joule per second, is linked via the energy per photon to the number of photons hitting the same area every second. However, as the amount of energy of a photon is directly related to its wavelength, the photon flux is most easily calculated for light of a specific wavelength. For light being emitted over a spectrum, as is usually the case however, the exact quantities of all spectral components are required to arrive at a precise value for the number of photons carrying the observed energy, and moreover, this wavelength composition is rarely constant, or even linear, across a natural light spectrum. A rough wavelength distribution can often be approximated by referencing the light temperature, but here a value was simple selected as a representative mean, keeping in mind that any inaccuracies should be relatively small compared to the large variability of the irradiance employed.

Light Regime

Bioluminescence was monitored on a Packard TopCount NXT scintillation counter, and for the administration of light pulses, the plates were taken out of the Packard scintillation counter and kept in a dark chamber until light pulsed at the desired intensity. Following the light pulse, the plates were returned to the scintillation counter and left in constant darkness for an extended period, namely 11 to 12 days, before being exposed to two consecutive regular light/dark photoperiods. A control sample was also exposed to the same constant darkness and ultimate light/dark cycles, but not subjected to any light pulses.

2.2.3 Denoising and Detrending

Theoretically, it would of course be possible to carry out comprehensive analysis directly on the raw data sampled from the scintillation counter. However, it is generally considered prudent to first filter out misleading perturbation, mainly classified as noise and trends, through specialized procedures known as denoising and detrending, respectively.

The Origin and Treatment of Noise

When considering the origin of these effects, it is useful remember how fundamentally stochastic our world is, and apart from fluctuations in expression rates and protein concentrations, which have even been pointed to as potentially essential features of the circadian clock, there could also be random variation in the reporter reaction with luciferin, or the levels of emitted light. Not least, there are also inherent inaccuracies in detecting photons and recording the corresponding data, a phenomenon that is well characterized in the context of photography and digital imaging, and all this underlying "noisy" variability, although in its sum often oriented along a normal Gaussian distribution, can add up to create phantom peaks and troughs, or to distort phase and period relationships. Depending on its precise cause, the noise can further be classified as either correlated or uncorrelated, but generally a good approximation can be achieved in denoising approaches by treating most deviations as independent and identically distributed. One possible avenue for handling noise consists of employing more robust data analysis tools, such as complex signal transforms, that are inherently better at filtering out small, random distortion than simple maxima/minima detection algorithms. Secondly, however, it is also possible to filter out noisy patterns in a separate step, even if oftentimes requiring a trade-off between data fidelity, noise reduction, and computational cost, for example by utilizing averaging filters or detail filters.

The Use of Discrete Wavelet Transforms

Specifically, discrete wavelet transforms can be utilized to decompose a dataset into discrete subbands with a corresponding set of wavelet coefficients. Here, the high frequency subbands describe the finer details of the signal, which usually contain the noise component, and provided that this high frequency component is small relative to the overall signal, simply cancelling it with a coefficient of zero can be an effective avenue for "killing the noise". The basic procedure is often further refined into thresholding, a technique that relies on a framework of cut-off values to cancel all subbands deemed insignificant, before reassembling the complex, but now denoised dataset through an inverse wavelet transformation. Other possible adaptations include the use of hybrid schemes of wavelet transforms and optimization algorithms, which can for instance be used to effectively remove non-stationary noise from electocardiogram (ECG) signals. Here, the critical selection of wavelet denoising parameter is guided by a genetic algorithm, resulting in maximized filtration performance with significantly improved quality and signal to noise ratio, when compared to wavelet thresholding algorithms in the same setting (El-Dahshan 2010).

In the context of this project, a discrete wavelet transform was carried out on the experimental data using the inbuilt *wpdencmp* function of the MATLAB® computational software suite, as well as using the dedicated application WAVOS. However, it was observed that analysis using different signal transforms could detect not notable differences between the original and denoised data, pointing to both the overall reliability of the data analysis tools, as well as to the relative clarity and smoothness of the experimental readouts. Furthermore, in the context of the simulated data, the very nature of the data generation should not produce appreciable levels of noise, except where explicitly desired through the use of stochastic simulation approaches, and so denoising is not deemed appropriate in this context.

Recognizing Underlying Trends

The second type of signal distortion to be considered, before quantifying the entrainment effect of light on an asynchronous cell population by such measures as amplitude decay or the amplitude just after the pulse, is the existence of underlying trends in the data set, which would hinder the quantitative analysis and may occur in bioluminescence circadian rhythms in cultured cells for several reasons: Firstly, the response of cell cultures to different treatments is not only inherently variable, but may also be influenced by unaccounted factors. Secondly, the rhythms of the cell cultures exhibit damping, or in other words variance non-stationarities. Thirdly, these rhythms often show unstable baseline shifting, i.e. mean non-stationarities, the exact extend of which may change from experiment to experiment, or even from sample to sample. The factors that could give rise to these various non-stationarities across the time series, and to the variability between individual cells, sample populations, or test runs are likewise multifaceted. Next to more general stochastic effects pointed to above, although in the case of "fabricated" trends likely of a different, slower quality, there are also countless potentially relevant surrounding conditions, such as the physiological state and age of the cell populations, the existence of background temperature fluctuations, or artefacts related to the handling of the sample and collection of the data. It can be downright impossible on a practical level to control all these effects, and while some trends may point to valuable insights, it is oftentimes preferable to reduce any corrupting influence before the further data analysis process, and accordingly various approaches exist for removing these trends. One relatively simple procedure for removing baseline drift involves calculating and subtracting a moving average from the raw data, while MATLAB® also provides for an automated detrend function that subtracts either the mean or, depending on the data set, a least-squares best-fit line from the signal. The statistical self-affinity can also be evaluated with the use of detrended fluctuation analysis (DFA), which is related to spectral techniques such as autocorrelation, and is frequently employed for long-memory processes, even where mean or variance are found to be non-stationary. Although DFA has gained much popularity since its introduction in 1994 by Peng et al (Peng et al. 1995), various update techniques for the detection of long-range correlations have also been suggested, including a Modified Detrended Fluctuation Analysis and Centered Moving Average (CMA). In particular, a recent comparison found that at least for weak trends, CMA shows a comparable performance as DFA in long data, but better results in short data(Bashan et al. 2008). Finally, data-driven techniques for decomposing multi-component signals include Empirical Mode Decomposition, which can be employed for both detrending and denoising by making use of partial reconstructions ("Detrending and denoising with empirical mode decompositions"). In this context it is interesting to note that it is difficult to distil a precise definition of a trend, but it has been demonstrated for climate data how EMD can be utilized to determine intrinsic trends and natural variability, namely by sorting for intrinsically determined monotonic function, or alternatively a function with at most one extremum, within a certain temporal span (Wu et al. 2007).

Detrending According to Moving Averages

In the case of removing possible masking effects from the circadian rhythms under investigation here, it is found that satisfactory results can be readily achieved by detrending traces on the basis of a 24-hour moving average; for an example, please see Figure 11. After all, the oscillating signal appears to constitute a relatively strong pattern around a naturally apparent anchor point, so that removal of the underlying distortion is well suited to an appropriately calibrated moving average approach. In the context of the simulated results, however, it should once again be noted, that all inputs are perfectly controlled by the computational environment, signifying that detrending might be, if anything, counterproductive for these sets of data. In fact, the ability to model even very long time spans in a tightly regulated environment is a good example of the many advantages of a well-established simulation system.

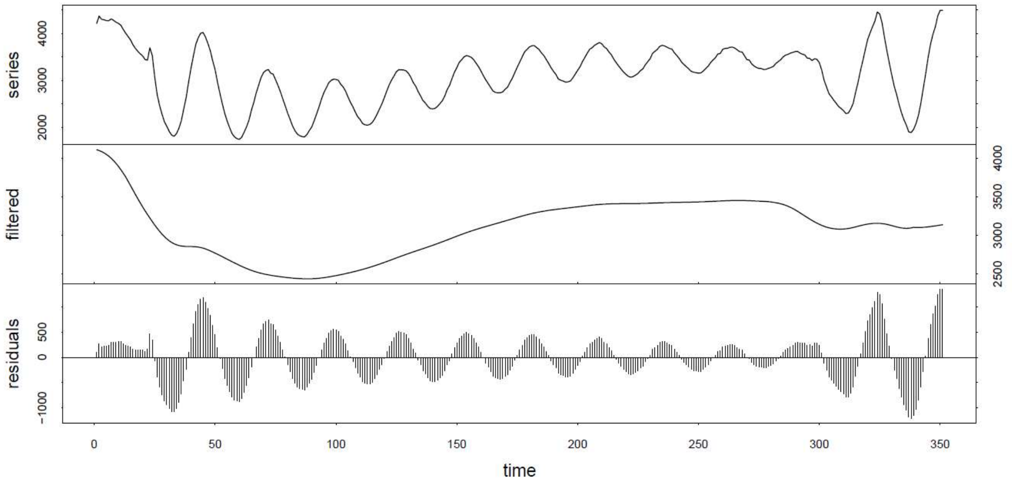


FIGURE 11 Detrending
Example of 24-hour moving average detrending on a light pulse trace. The top shows the raw data, the middle the trend that is removed and the bottom the residual detrended data

2.2.4 Results and Discussion

The bioluminescence raw and dentrended traces for the 15 minutes and 1 hour light pulse experiments can be seen in Figure 12 and Figure 13, respectively. Furthermore, amplitude after the light pulse and the decay rate were determined using a Hilbert Transform, described in more detail in the next section, and the results of the decay rate and amplitude analysis can be seen in Figure 14. The decay rate seems to increase slightly with higher light intensity and longer light pulse, while the amplitude, as expected due to a higher level of entraining stimulation, also increases with higher light intensity and longer light pulse. However, considering the fact that the number of photons stimulating the cell is proportional to light intensity times duration, it is interesting to note that 10 μW cm^{-2} for 15 minutes has a lower initial amplitude than the 1 μW cm^{-2} for 1 hour, corresponding roughly to a photon exposure of 378 μmol m^{-2} versus 155 μmol m^{-2}, respectively, implying that the cell does not simply take account of the total number of photons it is stimulated by. In order to check if the light pulses also had an immediate effect, the data points for the first full cycle were removed and the analysis performed again, showing that interestingly the decay rate between the original and "cut" data is very similar; only the amplitude is smaller, as would be expected.

Possible Limits to the Stimulating Effect of Light Exposure

One other result that emerges is the fact that no hard lower or higher limit to the stimulating effect on the oscillations was detected at the range of light intensities employed. Rather it appears, that there is a relatively constant relationship between the intensity of the light stimulation and resulting overall amplitude. As so far as this can be attributed to the re-synchronization of asynchronous individual oscillators, it could be argued that stronger pulses succeed in harmonizing the phases of individual oscillators more removed from the average. At the lower end it is confirmed, however, that light pulses of only 15 minutes are generally sufficient to evoke a clear response from completely asynchronous cell populations, and further that these boosted oscillation decay gradually, but overall persists for the entire duration of the experiment, that is at least 11 days. Finally, it is very interesting to note that there appears to exist an inversely proportional relationship between the intensity of the initial light pulse and the amplitude of oscillations upon the initiation of regular light/dark cycles. It can be speculated that this unexpected effect may either be attributed to statistical calibration error, or otherwise result from for example more of the luciferase medium being used up by the more intensely stimulated cells by that point. However, it may also be worth to follow up to what extent, if any whatsoever, asynchronous and synchronous cells may differ in their capacity to react to new light stimuli. As a more general point, readings for each separate light pulse run were based on 4 individual wells and appear to correspond well over the range of different traces, but the experimental design may still have been subject to additional distortions, not adequately removed by the detrending process.

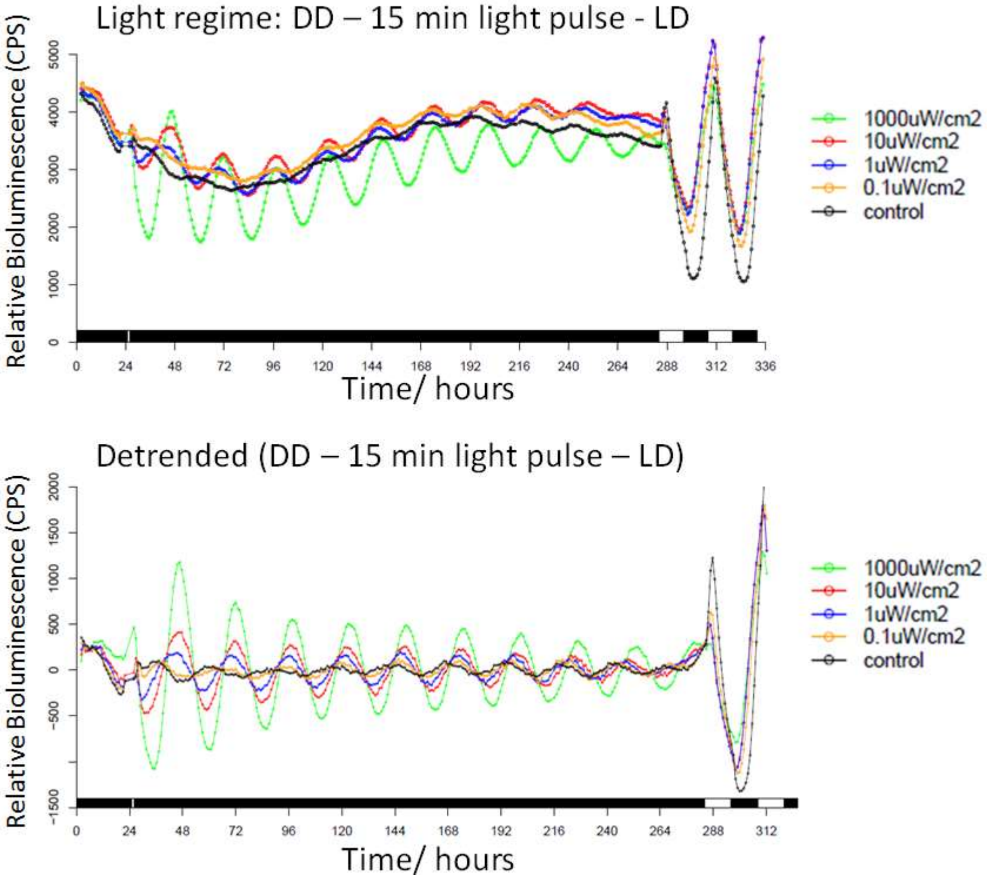


FIGURE 12 15-min light pulse

Bioluminescence trace of *Per1* reporter cell line. Cells were kept in the dark for 5 days before data recording. A 15 minute light pulse of varying strength as indicated was administered at 24 hours (control - no light pulse). At the end of the experiment cells were kept in LD for two days. Top - raw data, Bottom - detrended data

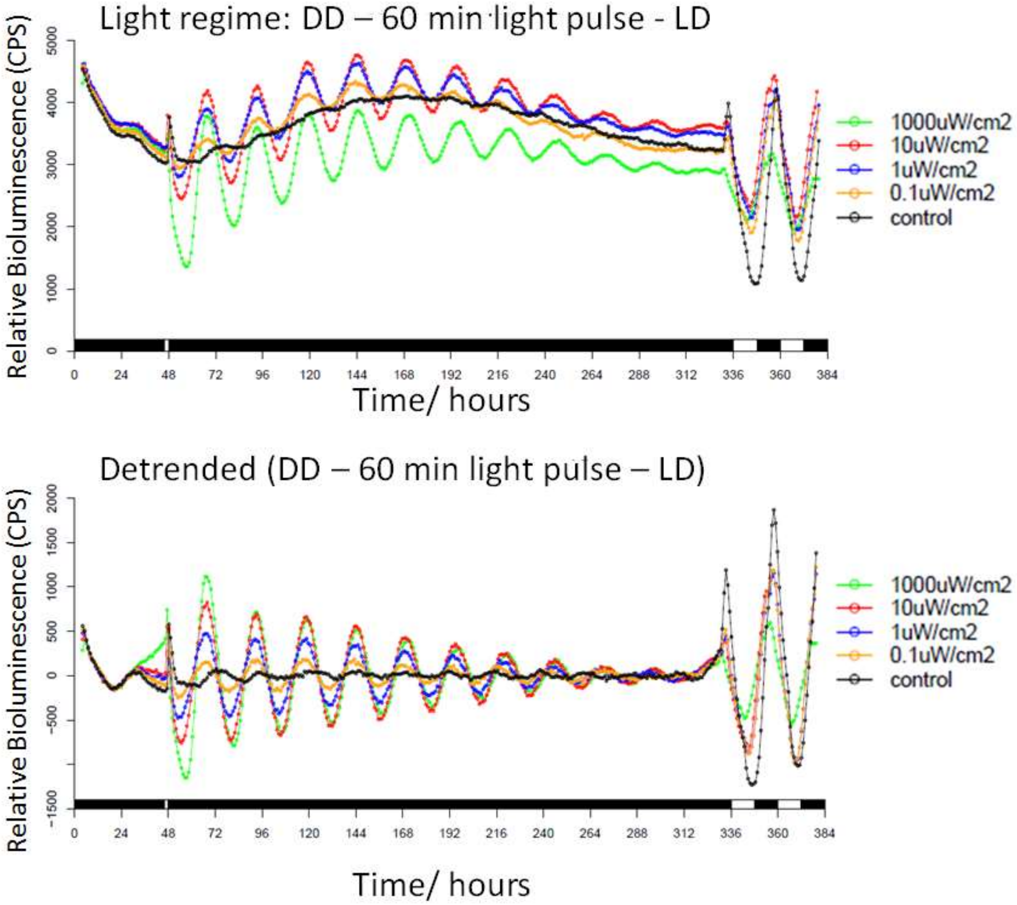


FIGURE 13 1hr light pulse

Bioluminescence trace of *Per1* reporter cell line. Cells were kept in the dark for 5 days before data recording. An 1 hour light pulse of varying strength as indicated was administered at about 48 hours (control - no light pulse). At the end of the experiment cells were kept in LD for two days. Top - raw data, Bottom - detrended data

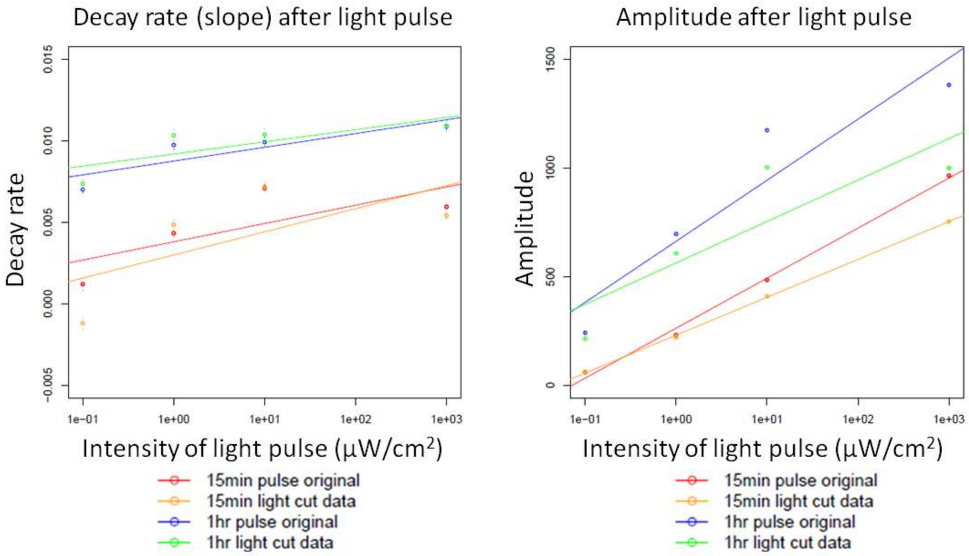


FIGURE 14 Light pulse data analysis
Decay rate and amplitude were calculated for the different length of light pulse and intensities of light. Additionally, the first complete cycle of the original detrended data was ignored for the cut data set.

2.3 Analyzing the Prepared Data

2.3.1 Time-frequency Analysis by Hilbert Transform

Time–frequency analysis methods see widespread use across a wide variety of areas, ranging from audio signal processing, over fault detection in industrial production (Peng et al. 2005), to medical research. In nearly all cases, computation efficiency and a good resolution of the time and frequency domains are considered advantageous, but the precise selection of a method, although frequently based on Hilbert transform, Wavelet transform, or their derivatives, will depend on the specific data set and research question under scrutiny.

Basis of the Hilbert Transform

The Hilbert transform (HT) is an analytical technique for transforming a time series into corresponding values of instantaneous amplitudes, frequencies and phases, which can then be employed, for example, to determine the dampening of amplitudes by using linear regression. The HT is named and traced back to the German mathematician David Hilbert, who, apart from his famous work on the invariant theory and the axiomatization of geometry, also proposed the theory of Hilbert spaces, which became an important cornerstone of functional analysis. Following on from this pioneering work on integral equations, the British Mathematician G. H. Hardy described a rigorous implementation of a transform in 1932, which he named in honour of Hilbert. Over the following decades the underlying definitions were vastly extended and improved by a series of other mathematicians, which lead up to such complex concepts as trilinear Hilbert transforms, and helped to apply the technique to areas ranging from telecommunication to biomolecular studies. In short, the HT is a linear operator, transforming functions while keeping their domain unchanged, and its capacity to extend a real signal into the complex plane has proven to be a very powerful tool in the field of signal processing. Specifically, the HT matches a real function *x(t)* with a companion function *y(t)*, so that *z(t) = x(t) + i*y(t)* can be analytically extended from the real line $t \in R$ to the upper half of the complex plane (Liu 2011). On a practical level, it is common to first apply a Fourier transform to the signal of interest, before rejecting any negative frequencies and applying the inverse Fourier transform, a procedure that will give rise to a complex valued signal with a real part and an imaginary part, also known as a Hilbert-transform pair. Notably, provided that the original signal is narrow-banded, the modulus of the transformed function will appear as its slow-varying envelope, while the phase derivative will be an instantaneous frequency, effectively signifying that the signal will be restated by the HT in terms of amplitude and frequency modulation. This capacity has seen the HT being used in varied functions, such as latency analysis in neuro-physiological signals (Recio-Spinoso et al. 2011), even if it was initially narrowly defined for period or circle functions. It can further be noted that the Hilbert transform of a function, or in other words the companion function generated, will not strictly be unique, but is an example of a singular integral operator and will constitute a harmonic conjugate in Fourier analysis. The HT also has the effect of shifting the phase of all negative frequency components of the signal it is applied to by *π/2* radians, while positive frequency components will shift by - *π/2* radians, but factoring in the imaginary component "*i*" will restore the positive frequencies while negating the negative ones. Discarding the negative frequency components in this way, which through the transform is possible without loss of information, designates the complex-valued function as an analytic signal. Here, the real and imaginary parts linked by the HT are both real-valued functions, and as a corollary the analytic representation of a real-valued function comprises the original function and its HT. As a clear advantage, certain attributes of the function can be more readily manipulated, and modulation and demodulation steps are facilitated in this configuration. However, as long as it still contains no negative frequency component, simply dropping the imaginary part of a manipulated complex function will revert it to a real state.

Metrics Revealed by the Analytical Signal

The analytical signal approach will thus, as follows readily from the aforesaid, enable the instantaneous phase and amplitude for a signal, the original real function *s(t),* to be found via construction of the analytical signal *ζ(t)*, created by the combination of the real function *s(t)* and its HT $is_H(t)$:

$$\zeta(t) = s(t) + is_H(t) = A(t)e^{i\phi(t)}$$

The instantaneous amplitude *A(t)* and the instantaneous phase *ϕ(t)* are uniquely defined by the above equation. Furthermore, the derivative of the phase, i.e. its rate of change, can be identified as the instantaneous frequency. It can be noted that for a pure sine wave, the instantaneous amplitude and frequency are constant, while the instantaneous phase, however, is a sawtooth, reflecting the way in which the local phase angle varies linearly over a single cycle.

$s_H(t)$: is the Hilbert Transform of *s(t)*:

$$s_H(t) = \frac{1}{\pi} P.V. \int_{-\infty}^{\infty} \frac{s(\tau)}{t-\tau} d\tau$$

provided this integral exists as a principal value, meaning that the integral is taken in the sense of the Cauchy principal value. This is precisely the convolution of s with the tempered distribution $p.v. \frac{1}{\pi t}$, and therefore the Fast Fourier Transform (FFT) based on the convolution theorem can be used to calculate the HT. Thus the envelope, which can be thought of as the amplitude variation, of a time signal can be computed, and in order to determine the dampening equivalent to the decay rate, a linear regression is performed on the logarithm of the envelope. Moreover, if the phase portrait of an oscillator is not a circle, the amplitude is not constant, but oscillates with a frequency $2\ \omega = 2 \cdot 2\ \pi\ /\ T$, where T is the period of oscillation. Here, the instantaneous phase is also not linear. Finally, it is noteworthy that HT, like many data analysis methods, can also suffer from end effects, which are related to extending the data beyond the available range, e.g. by predicting the missing data based on the available points. Fortunately, these end effects, stemming from HT's roots in the Fourier Transform, are considered relatively easy to deal with.

Computational Implementation

Turning to practical, computational implementation, in the R programming environment the package *seewave* can be used to compute the analytical signal. *Seewave* was designed for sound analysis and synthesis and can determine the analytic signal of a time wave as a complex matrix through the HT, which constitutes the imaginary part of this matrix (Sueur et al. 2008). The function *ifreq* can be utilized to obtain the instantaneous frequency and phase through HT, while the function *env* returns the absolute or Hilbert amplitude envelope of a time wave. An example can be seen in Figure 15, where a dampened perfect sine wave was computed, and in accordance to experiments readings are once per hour.

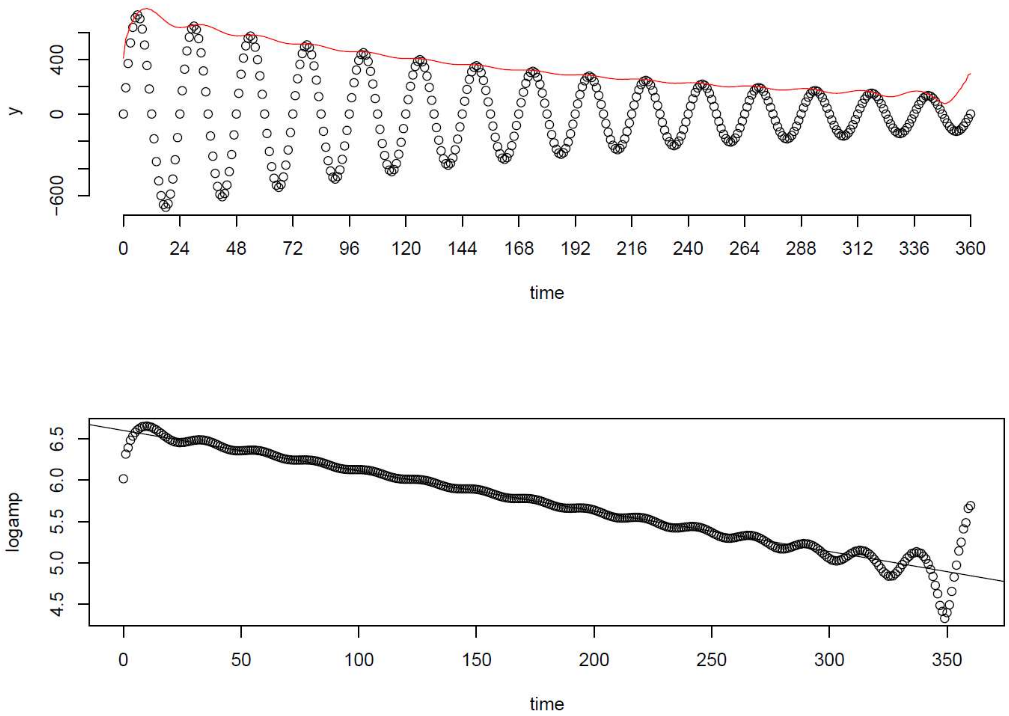


FIGURE 15 Dampened Sine Wave

A dampened sine wave was computed using the relationship $A*e^{-k*time}*sin(2*\pi*f*time)$, where *A* is the amplitude, *k* the decay constant and f the frequency. The time runs between 0 and 360 hours, with one data point every hour; Parameters used were A=750, decay=0.005. Results of HT: A=734.9663, decay=0.004873855 , Std.Error=5.828887e-05.

Top: Computed sine wave and envelope, **Bottom:** Linear regression of the logarithm of the envelope.

As a next step, it is envisioned to code a function in Matlab, that can be used to collect a set of summary statistics by leveraging the HT. This endeavour is much facilitated by that fact, that the program possesses an inbuilt function *hilbert* that "computes the so-called discrete-time analytic signal X = Xr + i*Xi such that Xi is the Hilbert transform of Xr", by utilizing the fast Fourier transform (FFT). Specifically, when the FFT is applied to an original signal, all elements with frequency corresponding to −π < ω < 0 are set to zero, and finally the inverse FFT is calculated, but the function also has the capacity to add zero-padding or to truncate as appropriate. In this way, a complex helical sequence, namely the analytic signal, is returned from the real data with an imaginary part exhibiting a 90° phase shift. Of course, the amplitude and frequency content is identical to the original sequence, and the included phase information depends on the original signal's phase, permitting an easy reading of the instantaneous attributes of the time series. Here, it holds true as described above that the instantaneous amplitude corresponds simply to the amplitude of the Hilbert transform, while the instantaneous frequency is the gradient of the change in instantaneous phase angle.

In the practical implementation of the code, the entirety of which can be found in the appendix, the function *hilbert* is applied to the data set of interest, before the results are sorted into their respective real and imaginary components. Maxima are located by determining the points, and their corresponding time values, just before and after the imaginary component moves from a negative to a positive value. The zero-crossing itself is then approximated by finding a point between the last negative and first positive point in such a way, as to lie between them proportionally to their respective modulus, that is the magnitude of their deviation from zero. Once the first and last value of each set are discarded in order to minimize edge effects, periods are determined by simply computing the time difference between consecutive maxima points. In order to determine trough-to-peak values over a period, real values are called back for the points just before and after the imaginary zero-crossing, and the real value exactly at the zero-crossing is approximated by first determining a real value between the preceding and following one as proportional to their time difference. It is then assumed, somewhat simplistically of course, that the real maximum value would exceed the higher of the two bordering values by as much, as the calculated midpoint lies below them. It is considered that even if this procedure would yield unintuitive results in some specific scenarios, such as for a midpoint exactly equidistant between two identically valued bordering points, the approximation should prove useful overall, with any errors being minimized by opting for a high time resolution. Of course the same procedure as described above can also easily be applied to calculating minima and their respective real values, one of the main differences being that the imaginary data component is scanned for positive values crossing over to negative values. The amplitude, in the sense of a trough-to-peak difference, can then easily be found by considering the difference between maxima and minima real values. It can be noted, that the procedure purposefully does not specify whether the reference period would start with a maximum or a minimum; it is proposed that such a prescription would not only be arbitrary, but also better controlled by selecting appropriate recording and experimental stimulation and simulation timings. Moreover, a segment is included to determine relative phases of different graphs, by first determining the maxima timing relative to the underlying period, and then the time differences in between the maxima of different traces. This values is further adjusted to safeguard it not exceeding half a period. Finally, a set of optional, automated graphical outputs is defined, including a plot of the maxima and minima on the complex HT, the instantaneous phase values over the last period, or a histogram of the period distribution.

2.3.2 Period Calculations

Initial runs using simulated data point to accurate summary statistic aggregation using the procedure described above. However, given the importance of estimating period timings from time-course data for inferring the underlying properties of the cyclical biological functions they are collected from, such as the circadian signal, and further factoring in the asymmetric and noisy scenarios frequently encountered in organisms, it is not surprising that an entire range of very sophisticated techniques has been developed in this field. For instance, periodic gene expression profiles in circadian clock related studies are frequently approximated using additive sine and cosine functions within a Fourier approximation context (Levine et al. 2002), such as the widely used Fast Fourier Transform Non-linear Least Squares (FFT-NLLS) method put forward by Plautz in 1997 (Plautz et al. 1997). Upon noting that circadian time series robustly produce a clear dominant spectral peak over a wide set of common conditions, various methodologies based on spectrum respampling (SR) techniques have also been developed, which have been shown as more robust to non-sinusoidal and noisy cycles than commonly used Fourier approximations, while also allowing the treatment of period estimates with different variances (Costa et al. 2013). Here, the use of bootstrap can play a pivotal role in shaping the spectral analysis, generating information on the distribution of an estimator through repeated resampling steps drawing from the original sample of values. As the desired estimate is calculated each time, the procedure gives rise to a set of estimates, which can be utilized as the basis of a point estimate and of corresponding confidence intervals. Specifically, it has been described how a consistent estimator can be obtained based on a smoothing parameter built into a kernel spectrum estimate. Further processing including a regression step on a set of residuals, and the generation of periodogram values through the use of yet another kernel estimate and specific smoothing parameter. The bootstrap periodogram can ultimately employed to arrive at the final bootstrap estimate, with all smoothing parameters acting in concert to control the bias and variance at this step. Accordingly, there values are generally chosen to minimize an appropriately defined mean square error estimator. This particular SR methodology would require a minimum of two completely recorded cycles to work on, but the improved estimator for the period has been shown to outperform the popular FFT-NLLS routine, while also allowing for a simple oscillation fit through the use of linear least squares.

Hilbert-Huang Transform

Yet another spectral analysis method that has seen much interest was first proposed by Huang and colleagues in 1998, and designated by NASA as the Hilbert–Huang transform (HHT). Here, a signal is decomposed by empirical mode decomposition (EMD) into so-called intrinsic mode functions (IMF) with a trend, and importantly HHT works not only well for data exhibiting nonstationary and nonlinear properties, but moreover preserves many important characteristics of the original signal, and specifically its varying frequency. This latter property stems from a decomposition in the time domain, with the length of the original signal and IMF being identical, and constitutes a marked advantage when dealing with signals containing causes in different time intervals. In many ways, the core of HHT consists of the EMD, a highly efficient and adaptive approach to break the signal down according to local characteristic time scales of the data. While the concept of decomposition is shared with more theoretical approaches, such as Fourier transform, the EMD is, unsurprisingly, more empirical in nature, and complicated data sets can be readily separated into a complete and nearly orthogonal basis of a finite, and often even small, number of components. These are the IMFs, which are characterized by the local maxima and minima defined envelopes being symmetrical, and the mean of those envelopes consequently reducing to zero. Furthermore, the number of zero-crossings and extrema must be the same, or at the very least not differ by more than one, and the IMF thus represents a simple oscillatory mode as a counterpart to a simple harmonic function. Having obtained the IMFs, Hilbert spectral analysis can be applied to each of them, revealing its instantaneous frequency as a function of time, ultimately resulting in a Hilbert spectrum, that is frequency-time distribution of signal amplitude (Cong et al. 2009). In addition to defining frequency as a function of time by differentiation rather than convolution analysis, HHT also stands out by not imposing a priori assumption on the data. Notably, subjecting the IMFs to the Hilbert transform also enables sharp identification of imbedded structures, and so the HHT has been applied in disciplines ranging from correction of satellite data, over speech analysis and speaker identification, to machine health monitoring.

2.3.3 Wavelet Transform

One other family of analysis techniques that acts by decomposition, but is distinct from the group of Hilbert transforms, relies on wavelets. A wavelet is usually defined as a brief oscillation originating and ending at zero, and based on the ability to purposefully craft various wavelets, they have found widespread adoption in signal processing. Notably, wavelets can be combined through the technique of convolution with portions of a signal in order to extract information. A well-known subtype is the Morlet wavelet, traced back to the Hungarian-British electrical engineer and physicist, Dennis Gabor, further famous for inventing holography, who pioneered in 1946 the use of Gaussian-windowed sinusoids for time-frequency decomposition as part of the Gabor transform, essentially a type of short-time Fourier transform. Specifically, the wavelet consists of a complex exponential as the carrier, which is multiplied by a Gaussian window as the envelope. The concept rose to renewed prominence when, in 1984, the French geophysicist Jean Morlet introduced Gabor's concepts to the field of seismology, developing with others a format of maintaining the same wavelet shape over equal octave intervals, now better known as the continuous wavelet transform. It has since been found that the Morlet wave holds special relevance for the field of human perception, but also for music transcription surpassing the Fourier transform, or for electrocardiogram (ECG) analysis to discriminate abnormal heartbeats. However, it should be noted that the wavelet exists not only as purely real-valued, but also as a complex version, sometimes referred to as "Morlet wavelet" and "Gabor wavelet", respectively, but more commonly known simply as real and complex Morlet. Following on from this distinction, the complex wavelet transform (CWT) was derived as a complex-valued extension to the initial discrete wavelet transform (DWT). The popular use of complex wavelet transforms dates back to 1995, when they were introduced to image processing by J.M. Lina and L. Gagnon, and the two-dimensional CWT is noted for its sparse representation, multi-resolution, and structural image characterization. While a drawback includes significantly increased, dimension dependant redundancy when compared to the DWT, its many advantages have earned it a prominent role in, for instance, computer vision, where the CWT is noted for the ability to elucidate precise features of candidate regions, facilitating accurate recognition of smaller target objects. However, the underlying techniques are of course useful in a multitude of other settings, including prominently data sets displaying suggestively periodic behaviour, especially when combined with a period, amplitude, or mean that are varying over time. This sort of variation can introduce significant inaccuracies to standard Fourier analysis techniques, considering that signal stationary and an unbounded basis function are generally assumed here, but are handled much more confidently by wavelets, which are themselves localized in time and frequency. This property permits, in turn, to localize the analysis and thus follow changing signal properties over time, and wavelets have consequently employed for not only studying sunspot cycles (Krivova & Solanki 2002), ecological time series (Cazelles et al. 2008), blood-flow dynamics and ECG readings (Addison 2005), but also in studying circadian systems (Meeker et al. 2011). Notably, circadian contexts are prone to a variety of attributes, such as variable period length, sharp transients, and phase shifts (Herzog et al. 2004), in addition to experimental artefacts including loss of amplitude, shifting means, and noise introduced by bioluminescence, all of which can complicate traditional signal analysis. It has been described before, how extensive pre-processing and detrending are widely employed to combat these hurdles (Levine et al. 2002), but the inherent properties of wavelet analysis, and namely their treatment of nonstationary oscillators that can bypass many of the problems inherent to techniques assuming stationarity, would also appear to hold great potential for the analysis of circadian data.

Continuous vs Discrete Transforms

Indeed, both CWT and DWT present themselves for interesting, albeit differing depending on their respective qualities, uses in this setting. While the DWT is sometimes criticized for its coarse time-frequency decomposition, it has been proven competent of reconstructing a signal and of permitting statistical testing, earning it a role in a range of signal-processing tasks, including denoising, detrending, decomposing the signal into different bands of wavelengths, or also discarding or shrinking the coefficients associated with those bands. Specifically, the discrete version, also known as Daubechies, of the wavelet transform allows analysing a signal on a multi-scale level, with a sequence of compactly supported filters used to decompose the signal into a defined set of component frequency bands (Daubechies 1992). The frequency estimates are less precise than with the CWT method, but different frequency bands can be readily subjected to statistical significance testing, and strictly time-limited transients can be efficiently removed due to better time localization properties. As signal reconstruction in DWT can also preserve the mean of the signal, it can also be utilized easily as pre-processing step for further, non-wavelet analyses. The CWT, on the other hand, would struggle with many of these processing functions, but its roots as a modification of the windowed Fourier transform, which adjusts the boundaries of the window to a constant number of wavelengths for any analysis frequency, afford it a notable frequency detection across a wide spectrum of periods, and let it shine at examining local features (Harang et al. 2012). Specifically, it has been noted that its precise decomposition along the time and frequency domain permits very close tracking of key statistics, even if the signal cannot be efficiently reassembled from the individual components. Specifically, the CWT is able to nonparametrically denoise, detrend, and analyze local frequency content of a signal within a single operation (Baggs et al. 2009), and further permits an array of applications, including the estimation of signal period and phase evolution across time, localization of peaks and troughs even under conditions of high levels of noise, recording the evolution of amplitude over time, and even the possible identification of multiple simultaneous oscillators.

Existing Computational Implementations

Considering practical software applications of the wavelet transform, while disregarding proprietary iterations specialized for commercial audio or picture processing such as WaveLab, a well-known example includes the WAVECLOCK implementation of the Morlet CWT in the R statistical programming environment. While it has been a component of several investigations (Etchegaray et al. 2010), its use naturally depends intimately on the R language and command-line interface. Other solutions natively implemented in the MATLAB environment include a dedicated Wavelet Toolkit, or also a program under the name of WAVOS, which was specifically optimized for use in circadian rhythm analysis. Ultimately, it is deemed worthwhile to code a wavelet analysis tool specifically for this project, thus allowing a seamless and potentially automated integration with other software components. However, there are several useful elements found in open license projects that can be integrated here, and while most have been extensively modified, optimized, or expanded, credit is given specifically to the WAVOS package (Harang et al. 2012), the "crazy climber" method for ridge detection (Carmona et al. 1999), and several smaller code fragments by Didier Gonze. The full code of the wavelet transform tool can be found in the appendix, but the most important functions are summarized as follows.

Adaptation of the Algorithm

Following some general definitions and other housekeeping functions, as well as the optional addition of zeros to "pad" the data and reduce edge effect, starting and stopping scales are manually or automatically defined. Following on from this, the Fast Fourier Transform (FFT) is applied to each trace before, considering each scale separately, the FFT data is multiplied element-by-element to a function defining the Morlet wavelet, depending mainly on the current scale and adjusted index for FFT frequencies, and finally the Inverse FFT is applied to the resulting product. After removing possible zero-padding at this point and re-aligning the newly transformed data to norm, result arrays are populated and passed on to second sub-function for ridge detection. After determining whether or not to exclude edges, a trade-off that reduces possible distortions at the edges by applying a mask obscuring part of the data, before passing it on to one of two methods for ridge detection, with one being simulated annealing based and the other utilizing the crazy climber methodology. Both are capable of handling multiple ridge selection and local maxima, and are essentially derived from Markov-Chain Monte-Carlo (MCMC). This methodology combines two important ideas in computational simulations, the first one being Monte Carlo methods, named after a city famous for its casinos and gambling, which draw on random sampling in order to obtain numerical results particularly for systems with high uncertainty or several degrees of freedom. A Markov-Chain, on the other hand, describes a movement through state space that is characterized by the "memory-less" Markov property, namely that the probability distribution of the next state depends purely on the current state, while disregarding the sequence of events and previous states leading up to it. MCMC consequently refers to a class of algorithms that effect sampling a probability distribution by constructing a Markov chain with a specific equilibrium distribution, i.e. a stationary distribution the chain will converge on over time irrespective of its starting point. MCMC applications are very useful for numerical approximation and detection of rare events, particularly in complex systems, and a very common implementation are random walks. Here, different points are consecutively selected by a walker and added to the integral, depending on which tentative next step holds a reasonably high contribution for moving forward. The sample is consequently shaped by the evolution of the chain's state, and not surprisingly, the quality tends to improve with the number of steps.

Obtaining Maxima and Other Properties

Of the two ridge detection methods, the simulated annealing aims to minimize a penalty function, in this case with smoothness parameters constructed to remain proportional, while the crazy climber instead generates weighted occupation densities to draw the ridges. Specifically, a random walk would be initiated on the time-frequency plane, but in such a way that the walker is attracted to the ridges of the hills, a quality that also inspired the tongue-in-cheek name "crazy climber". Moreover, while the movement of the walker is never prematurely stopped or restricted, the affinity for approaching the ridges is controlled by a temperature, which is in turn modulated by a cooling schedule and thus permits to gradually map out a more complete topology and lesser ridges. It is generally observed that the crazy climber can be more sensitive towards detecting ridges in more complicated scenarios of high noise levels and multiple ridges, but in turn also requires significantly more computation time. In any case, as the next step under both methods the difference between neighbouring values is calculated, and the point where the sign of the differences changes is identified as a turning point. It is also at this step that maximum values can be specified as cut-offs, an option that allows for discriminating local maxima. More background information and a general descriptions of the different ridge detection algorithms is also available in book form (Carmona et al. 1998). In any case, once the desired maxima have been identified in the CWT, it is a trivial matter to read out a number of associated properties, such as a table of wavelengths, periods, phases, and amplitudes. Finally, an additional step is also implemented to read out and record various points of interest, such as zero crossings, from the list of phases observed. While the summarized characteristics may be readily inspected in a table format at this point, there is also an additional sub function included to support improved accessibility through visualization. Here, a range of properties, specifically period, peak-to-trough, and phase, may be plotted relative to the data, while a second group of characteristics, including the CWT, ridges, and phases, can also be represented as a heat map.

Results of simulated runs attest high plausibility and reliability to the CWT function implemented here, when considered from an intuitive angle or *a priori* information, as well as when compared to the simpler HT function described earlier. Nevertheless, it is conceded that there is always room for improvement, especially in the rapidly changing field of signal analysis. For instance, a study on the interpretation of QRS complex characteristics in the context of processing ECG signals looked at R wave identification using Hilbert methods, wavelet transforms, and adaptive thresholding. Rather than seeing any one of these different approaches emerge as clearly superior, it is actually found that a combination of employing these techniques significantly outperforms other techniques quantitatively and qualitatively (Rabbani et al. 2011).

2.3.4 Oscillator Quantification

Having described different approaches and implementations of analysing biological and simulated signals, it should be warranted to explain briefly some of the key characteristics thusly extracted.

Amplitude

In physiological and molecular studies, phase and period have traditionally been considered to be the most reliable indicators of pacemaker action, but in the modeling of circadian oscillations the importance of amplitude has been equally appreciated, also due its role in photoperiodic induction. Specifically, the amplitude of a rhythm can give an indication of oscillatory strength, although it is also a reflection of the output pathway characteristics, including transients, masking, and other amplitude effects. The amplitude could, for example, be the difference between the peak and trough bioluminescence CPS readings. However, it should be noted that differences in oscillatory amplitude in cell cultures can be due to a number of reasons, including cell lines, cell density and biological variation. Consequently, it may be more accurate to consider relative amplitude, for example compared to an LD cycle. Also, in the case of population studies the amplitude of individual oscillators can still not be accurately discriminated, and even only estimated when reliable numbers on population counts are available. In summary, quantifying the amplitude of the bioluminescence rhythms as a gauge of the amplitude of the underlying oscillator may contribute important information for modeling, and serve as a useful benchmark for simulation runs, especially when recreating the effect of light stimulation.

Dampening

There has been evidence in several species that rhythms of cells in culture in constant conditions decay not due to a dampening of amplitudes in individual oscillators, but due to increasing desynchronization among these individual oscillators. While asynchronous individual cells still have high amplitude rhythms, their peaks are at different times, to the point that the average dampens to a flat signal. Therefore, differences in oscillatory amplitude elicited by different treatments would be a reflection of the phase-shifting and coordinating efficacy of these various treatments, and capturing this dynamic should be an important feature of at least any circadian model investigating synchronization effects. Beyond the basic occurrence of dampening, relative levels might also serve as valuable indicative pointers, even if absolute levels may be difficult to pinpoint accurately, also in light of their possible entanglement with experimental artefacts and other trends.

Period and Frequency

Period and frequency are intimately related by the equation $T=1/f$, and it has been noted how the instantaneous frequency, and thus period, can be determined using the HT, CWT, or a host of related techniques. Not surprisingly, the period is often considered the most integral part of an oscillating or otherwise periodic system, and the one most frequently cited as a reference point. Indeed, in the simulation environment checking for a regular period within plausible parameters can often be a decisive first step in establishing the occurrence of stable oscillation for a given equation or set of parameters. In the case of comparing simulated with biological periods, it should be noted that, while well entrained rhythms should have the same period as the entrainment signal, free-running rhythms will occur at their own distinct period, often subject to significantly higher levels of variability.

Phase and Form of Oscillations

The instantaneous phase can once again be determined using an array of signal transforms. Importantly, the evolution of the phase can allow significant and quantifiable insights into the shape of a trace, and in the case of a sine wave for instance, the phase changes at a constant rate. However, if the shape of the oscillation is not exactly sinusoidal, as is often the norm in practical settings and especially physiological contexts, this rate of change of phase will not be constant, and thus a lot more complicated to describe. Here, the specific shape or form of oscillations can be visualized either in a plot of time evolution or the phase trajectory in a phase plot. Considering the most important broad categories commonly used, clock components may oscillate with sinusoidal, spiky, or square-wave form and the rise and decline may furthermore not be symmetric. Moreover, the plateaus of peak and trough may also differ in broadness, additionally complicating matters. Nevertheless, these factors may hold important clues to the underlying regulation, and as often as subtle differences may be the result of stochasticity or experimental inaccuracy, they may also hold the key to understanding critical mechanisms.

2.4 Analyzing and Capturing Underlying Dynamics

2.4.1 Biological Findings to be Integrated into the Model

As it comes to summarizing the analysis of the experimental data, an entire array of observations can be noted. Firstly, the molecular basis of the circadian clock has been described before in some detail, regarding both elements generally conserved over a range of species and those pathways that are specific to the zebrafish model organism. Without a doubt, the interactions, negative feedback loops, and other network motifs constituted by this genetic basis should serve as the foundation and cornerstone of any modeling implementation developed here, as they have for the initial model presented earlier in this text. However, there also falls critical importance to filling this framework with the animating details of parameterization. Specifically, the experimental data reported here and elsewhere can provide insights on minute behaviour to either integrate directly into the model architecture, or alternatively to benchmark simulated results against. The first number habitually considered in such an array of summary statistics would be the period. The value extracted via the different signal transform functions from the data measured in the light pulse experiments is 24.6h, close to the 25h commonly cited in literature for oscillations in DD, especially when considering that the strong dampening observed due to the experimental framework may also lead to a growing instability in period timings. Regarding the amplitude, it has been noted that is remains difficult due to a host of factors to work with absolute numbers here, and so it is merely considered useful at this time to potentially include minimum cut-off numbers with the purpose of discriminating strong and viable oscillating patterns. More interesting appear the phase relationships between the different molecular species, i.e. the exact time points and relative temporal differences observed for their maximum and minimum concentrations. The values measured here are consistently close to ZT0, with any small differences likely attributed to the practical setup, involving cells in constant condition suddenly exposed to light pulses, which could inherently introduce deviations. Alternatively, possible light contaminations during experimental setup might already be sufficient to cause small shifts in relative phase. In any case, it is likely advisable to consider different characteristics in a somewhat hierarchical fashion. Of course a perfect model might be expected to perfectly mirror all aspects observed *in vivo*, but both the simplification necessarily inherent to most workable models, as well as the challenge of shaping model evolution and parameter selection step by step may signify easily, that accounting of a narrowed set of key statistics may be preferable at this point.

How Light Stimulation Modulates the Circadian Clock's Oscillations

One of the points that stands out strongly with regard to basic model features, however, is the strong stimulation exerted by even the shortest and least intense light pulses triggered in the experiments. While this observation may straightforwardly point to the relatively strong effect of light stimulation, it is more confusing to note how biological response appeared only proportional to light intensity across either one of the durations, but not total light exposure across different light pulse durations. The effect is too pronounced, with the response being nearly three times stronger than expected by comparison, to be simply attributed to differences in relative amplitude, and so a first impulse may be to suggest that the cell populations may be limited in the amount of light they can process per second. As such, they would be able to make better use of the same total amount of photon exposure spread out over a longer duration. However, this explanation falls short of explaining, why no plateau is observed in the response to even much higher intensities, by orders of magnitude, of light stimulation. A second possible explanation may come in the form of the phase response curve, meaning that the responsiveness to light changes over the course of the circadian cycle. Following on from this, the stronger oscillations resulting from longer, but less intense light cues may signify, that the cell populations become increasingly more sensitive to light as the period proceeds. Of course, implementing experiments with an even greater variation in light pulse duration may provide valuable insights in this regard.

There is a host of other information conveyed in other experimental reports, the circadian clock being of course an area of intense interest, however two observations stand out in particular as regards the basic behaviour of the model. The first refers to the characteristic shape of the oscillations under a LD cycle, and namely the fact that particular reactant concentrations appear to change already before the onset of light. A likely explanation would be that these trends are owed to the momentum arising from the previous cycle, which would also see oscillations continue, although dampening, for considerable time in constant darkness. However, once the light input is activated, the gradient of the molecular oscillation increases markedly, resulting in a very characteristic transition at the onset of the cycle. It is further estimated from data, that oscillations in the presence of an appropriate light source reach more than twice the amplitude observed in the first few cycles of constant darkness. The second fundamental observation, which was also previously mentioned, lies in the insight from single cell data that individual oscillations continue under constant conditions without the amplitude being significantly decaying, but with much increased variability in the periods recorded. Clearly, this finding points strongly to both signal dampening and entrainment as potentially driven by de- and resynchronization effects of an inherently stochastic cell population. It is considered a central objective of the modeling efforts going forward to replicate these possible dynamics.

2.4.2 Analyzing Underlying Model Dynamics

Sensitivity Analysis

The two different signal transforms discussed above, namely the Hilbert Transform and Morlet Continuous Wavelet Transform, are not only suitable for extracting key metrics from the experimental data, but can also be readily employed to automatically analyze simulation outputs, thus also safeguarding that both kinds of data can be subjected to the same methodology for comparing their summary statistics. In order to smoothly integrate this capability, the clock model previously implemented in the Mathematica software environment was transferred to MATLAB. Here, the same set of differential equations described earlier is defined as a local function, and passed to the command *ode45*. Although MATLAB boasts an array of different numerical solvers, the medium order method *ode45* is one of the most widely used iterations in the context of nonstiff differential equations, and was constructed as a variable-step solver based on an explicit Runge-Kutta formula referred to as the Dormand-Prince pair (Dormand & Prince 1980). Its general syntax, *ode45(odefun, tspan, y0)*, involves primarily the function handle *odefun* of the system to be integrated, the integration time span *tspan* usually defined by its starting and end time, and finally the set of initial condition *y0*. However, here an alternative syntax is employed where the differential system is passed indirectly as an anonymous function, which not only permits routing through an intermediary nested function that defines the time sensitive light input, but moreover also allows for the passing of a multitude of additional function parameters.

Moreover, additional analysis functions are implemented to easily, and relatively quickly, visualize and explore further phenotypic aspects and behavioural dynamics of the model iterations run in the simulation environment. It should be noted, however, that unlike the previous tools, this set is restricted to pure *in silico* application and unsuitable for analyzing biological data, as the functions work by directly manipulating inputs. In that sense, they are not analyzing existing data, but rather generating novel, complementary output through a series of pre-scripted virtual experimental runs. Firstly, a simple tool for sensitivity analysis is coded, which is a technique linking uncertainty in the output of a mathematical model to uncertainty contained in its inputs, and which can be utilized to test the system's robustness, to weigh relationships between input and output variables, or to generally scan the model for aspects featuring uncertainty, needless complexity, or downright errors. Using an example from the economic sphere, where sensitivity analysis is widely employed, it could be investigated which one of a list of potentially changing factors, such as production costs, taxes, or pricing of logistic partners, would have the biggest impact on overall profitability. In simple cases these deductions may appear rather intuitive, but more complex model systems, featuring different layers and multiple variables, can often act like a "black box", that is they generate output that is only opaquely related to the inputs. In the specific context of this project, the method of sensitivity analysis is considered valuable for delineating which parameters have an especially prominent effect on the model system, as observed from summary statistics.

Moving on to the practical implementation, the basic differential equation system was first enriched with the capacity to accept an override for any of its parameter values from outside the function. As the appropriate command for the sensitivity analysis is triggered, a first simulation is carried out with the parameters as given, and summary statistics are retrieved through an externally linked HT function. In the next step, a regime for varying every parameter individually by a certain percentage increase or decrease is defined, either manually or automatically, and applied to one parameter at a time. Importantly, each time a parameter is updated in this fashion, a new test simulation is initiated and summary statistics are once again extracted by the HT and stored in a central array, before the parameter under investigation is reset, and the procedure moves on to the next one. Once the function has calculated results for all parameters and all entries in the variation table, results may either be inspected in table format, but are also output to a bar graph, plotting relative change in amplitude and period relative to the percentage change in each parameter. A sample output can be seen in Figure 16. One of the strong points of this procedure lies in the capacity to easily compare the robustness of the different parameters, picking out in particular the ones, the changes of which have either very drastic or almost no effects on the selected summary statistics. Finally, it should be mentioned, that for ease of use and visualization all relative changes are calculated for a single molecular species, by default the first one in the list, over a standard time window of 240 units, but both these values can be easily adjusted over the function parameters.

Bifurcation Analysis

A second tool closely related to the sensitivity analysis, and one moreover relatively easy to implement once the necessary modification for the former are introduced to the code, is a bifurcation analysis function. It has been described in a previous chapter in some detail how topological changes in a group of vector fields, integral curves, or solutions to differential equation can be studied by bifurcation theory, allowing insights specifically into the often seemingly erratic behaviour of dynamical system. The underlying principle of its practical implementation bears much resemblance to the sensitivity function discussed above, but a pivotal difference lies in the fact that here only one parameter is investigated at a time, but in return at a much greater resolution. After all, the objective is not so much to determine and compare the respective system relevance of the various parameters, but rather to scan for small changes in the bifurcation parameter leading to sudden leaps in the system's behaviour. In order to facilitate this objective practically, a range of values is either manually defined or by default set to run from 0 to 2 in increments of 0.01, to be cycled for the parameter to be investigated, which can itself be selected when calling the function. Subsequently, each value in the test range is in turn passed to the system of differential equations to fill in for the bifurcation parameter, and each time a test simulation is carried out at the length and time resolution specified. Once the simulation data is generated and stored in a temporary array for processing, one of the variable traces, which can once again be freely selected, is sorted for simple, that is including those of the local variety, maxima and minima. The entire range of values specified is cycled through, and subsequently both minimum values and maximum values are plotted against the tested range of the bifurcation parameter, as illustrated in Figure 17. Specifically the relationship of the maxima to the minima lines is noteworthy here, namely if the lines coincide it can be assumed that there is only stationary value for the reactant concentration, which would indicate an inactive system. On the other hand, if distinct maximum and minimum concentration are recorded within the same simulation run, these can usually be taken as strong indicators of oscillating behaviour, especially as the function includes a sub function for initial value normalization. In this manner, the bifurcation analysis can provide an accessible representation of the minimum parameter value at which oscillating behaviour is initiated, the maximum value at which it cedes, and the point where maximum amplitude is realized, all embedded into a visualization of the overall, for practical purposes nearly continuous dynamic behaviour. Therefore, the information gleaned here can be a valuable contribution to the effort of parameter selection, even if it should be remembered that considering only one parameter at a time will inherently constitute a great simplification of the underlying inter-parameter dynamics.

FIGURE 16 SAMPLE SENSITIVITY ANALYSIS OUTPUT

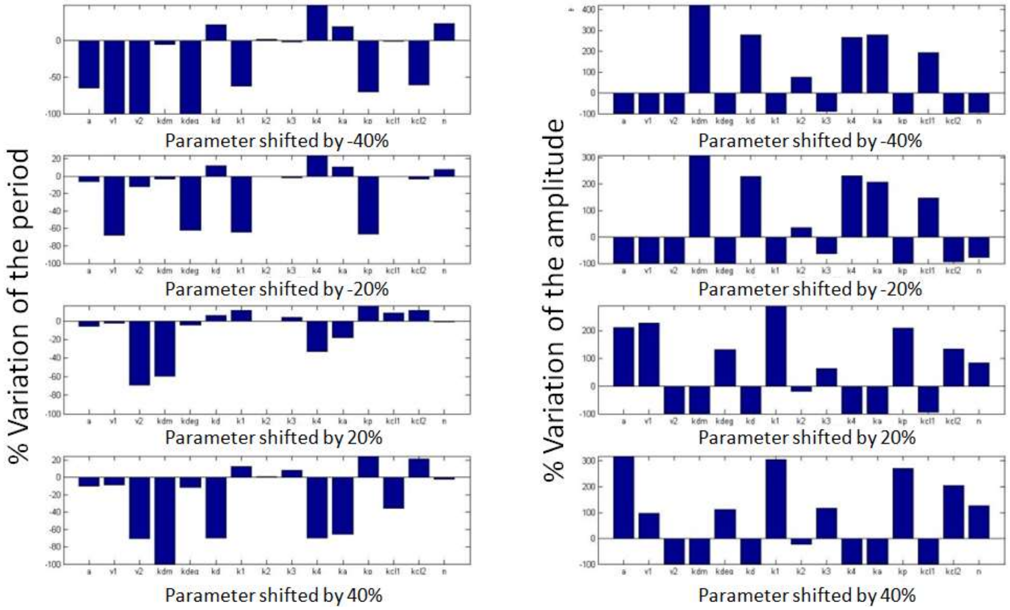
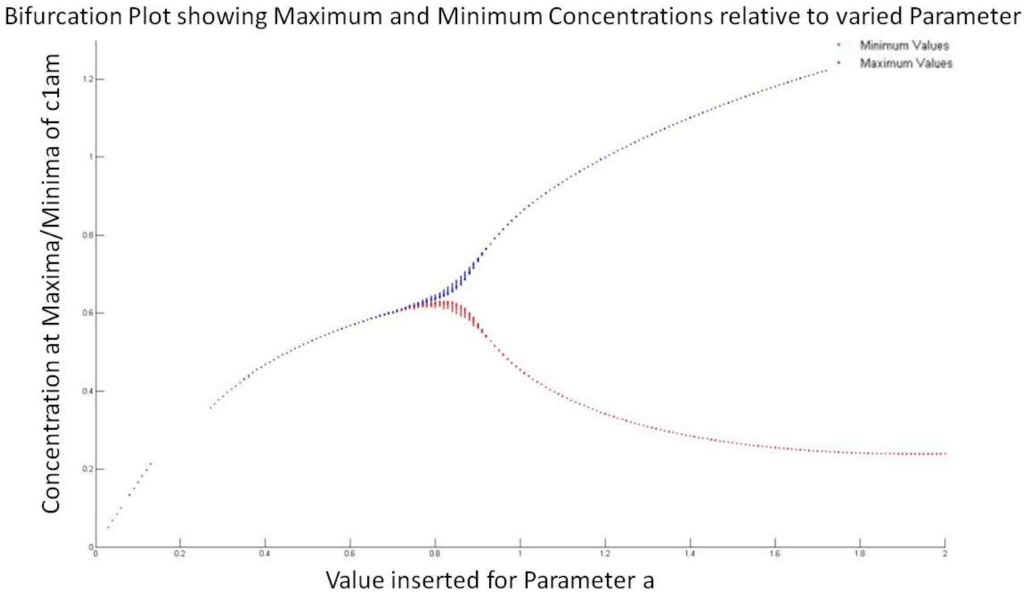


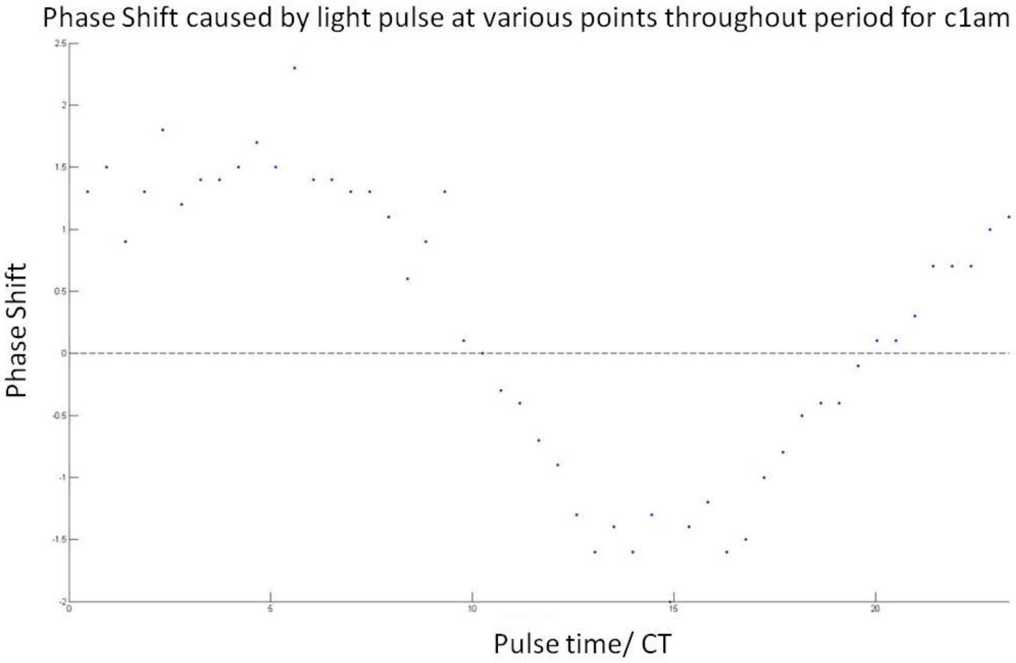
FIGURE 17 SAMPLE BIFURCATION ANALYSIS OUTPUT



Phase Response Curve

The next function to be introduced to the simulation environment is a phase response curve generator, which can induce perturbations at different phases of an oscillation period and illustrate the transient variation in subsequent cycles resulting from this. It has been described before how PRCs are popularly constructed for biological settings, including heartbeats, neuronal firing, but also and especially circadian rhythms, and they usually take the form of a plot of the subject's endogenous day along one axis versus the phase shift evoked by a stimulus on the other. In this manner they may for example be used to display how exposing subjects to light therapy or doses of the hormone melatonin may shift sleeping patterns, notably pointing to different effects depending on the precise time of administration. Staying with the example introduced above, exposing a test subject to light close to the personal regular bedtime would lead to a phase delay, with the delaying effect increasing in strength relative to light intensity, but also over the progression of the evening. However, as the body temperature nadir is observed after around five hours of sleep, the effect of light stimulation suddenly switches from phase delay to a strong phase advance. In the context of this project, an automated PRC function can be used to easily examine the response of the model system to light stimuli over the course of a circadian cycle, thus allowing insights into its inherent light response behaviour as relating in particular to a possibly synchronizing mode of action. Turning to the practical coding steps, once again a simulation under standard conditions if first run as a benchmark, recording in particular the last two maxima and the time difference between them. Now, a series of additional simulations is performed featuring freely adjustable light pulse exposure at different points, specifically by default fifty individual points distributed equally over the course of one period, as established in the initial benchmarking run. In each case the maximum is detected and the time difference, or phase shift, to the reference maximum is calculated, and once all specified points have been run these shift values are plotted over the course of a period. In addition, it is also possible to plot any selection of the individual curves following light stimulation, so as to compare and contrast the different effects from a different, and more complete angle. It should also be noted here that in order to faithfully pick up the differences in the system's reaction between the finely scaled individual points, the time intervals should not be set too coarsely, and without manual override they default to 0.1 of a time unit. A sample plot produced by the PRC function can be seen in Figure 18, where it can be noted how the phase shifts fluctuate, and indeed change direction, over the course of the period. This result demonstrates that an important quality of the circadian clock is already contained within the model, but of course even relatively small shifts to this behaviour may still have the potential to decidedly vary the system's behaviour in relation to light stimulation.

FIGURE 18 SAMPLE PRC OUTPUT



Visual Feedback Functions

Finally, a range of less conspicuous visualization options was added to the modeling environment, each of which, while not producing dramatic new insights in their own right, can help clarify and bring to attention the output already generated. For instance, a bar is added on the x-axis to display the presence of light and dark periods over the simulation time, marking the former with a light and the latter with a black box, which serves the obvious benefit of providing a reference frame to view the model output against, especially as it replicated various light regimes. Subsequently, a second, separate function was added to plot light input next to the other equation variables, adding additional value in a number of ways. Firstly, the inherent redundancy of displaying the light input in two only marginally related ways helped to uncover several discrepancies with the light input function during early test runs, secondly the light plot has the capacity to represent different relative light intensity levels, while the light/dark bar is easier to read on a larger scale, and finally both methods of representing light cues could also be appropriated differently, for example showing single light pulses affecting a population entrained, and marked, to a 24 hour light/dark background rhythm. Another visualization aid that was carefully implemented is a "zoom" function, allowing to close in on a specific section of the graphical output, the precise start and size of which can be freely specified. This functionality can help to combine a macro overview of a long simulation run with a more detailed window, showing for instance with much greater resolution phase relationships at a specific point in time, or permitting to compare shapes of the oscillator curve near the beginning and end of the run. Several more functions are included, helping for example to display individual and aggregated stochastic simulation runs, and all of these can be found in the annotated code.

2.4.3 Further Extensions to the Model

A range of further noteworthy additions is implemented for the model and the wider simulation environment, including an initial value correction regime, and more notably, a parameter generation algorithm based on the sequential Monte Carlo concept described in recent literature (Toni et al. 2009). Another important addition consists of the integration of an SDE solver, allowing to transform the deterministic system into a stochastic one. Subsequently, multiple simulation runs are aggregated to replicate a population level view, with the results found to introduce a remarkable instance of emergent behaviour, or in other words with the combined result showing unexpected dynamics when compared to the individual traces.

3. Materials and Methods

As part of the light pulse experiments carried out as part of this study, a period1-luciferase zebrafish cell line was used to monitor gene expression of per1, and hence progression of the circadian oscillator. In short, cells from the zebrafish embryonic cell line PAC-2 were transfected with a construct consisting of part of the zebrafish Per1 (*zfper4*) regulatory region cloned into a luciferase reporter construct. Now the enzyme luciferase is synthesized when transcription is activated by the promoter of per1, and this enzyme subsequently interacts with the substrate luciferin, which can be added to the medium, to release light by the process of bioluminescence. This bioluminescence can then be systematically detected and measured as counts per seconds (CPS).

3.1 Experimental Conditions

25x10^3 *per1-luciferase* cells per well were plated in quadruplicate wells of a 96-well plate in media containing 0.5 mM beetle luciferin. For each light intensity duration (either 15 minutes or 1 hour), test were carried out over a range of light intensities, and for each light intensity one separate plate with quadruplicate wells was used, with all plates being kept in a dark incubator for 5 days before data recording to create a desynchronized population. Light pulses were performed either after 24 hours of continuous readings for a duration of 15 mins, or after 48 hours of continuous readings for a duration of 60 mins, and each timing format was repeated with intensities of 0.1, 1, 10, and 1000 μW cm^{-2}, where a short duration and low intensities were chosen to determine what amount of light may be sufficient to cause an effect. The wavelength spectrum was 400-700nm, and assuming a mean wavelength of 520nm, this range of irradiance can be calculated to correspond to a photon flux ranging from 0.0043 to 43μmol m^{-2} s^{-1}.

Bioluminescence was monitored on a Packard TopCount NXT scintillation counter, and for the administration of light pulses, the plates were taken out of the Packard scintillation counter and kept in a dark chamber until light pulsed at the desired intensity. Following the light pulse, the plates were returned to the scintillation counter and left in constant darkness for an extended period, namely 11 to 12 days, before being exposed to two consecutive regular light/dark photoperiods. A control sample was also exposed to the same constant darkness, handling (taking out and returning to the Packard TopCount NXT scintillation counter for light pulsing) and ultimate light/dark cycles, but not subjected to any light pulses.

3.2 Data Processing and Model Simulation

Experimental data were imported into Microsoft Excel 2007 and were further analyzed on a MacBook Pro "Core 2 Duo" purchased 2009, featuring a 2.4 GHz Intel processor with two independent processor cores on a single silicon chip, a 3 MB shared on chip level 2 cache, a 1066 MHz frontside bus, 2 GB of 1066 MHz DDR3 SDRAM, and a 250 GB Serial ATA (5400 RPM) hard drive.

Wolfram Mathematica 7, consecutively upgraded to Version 10, was utilized to create early Oscillator models and Sequential Monte Carlo Algorithms, and the precise code can be found in the Appendix. Later versions of the Oscillator Model and the SMC Algorithm were implemented in MathWorks MatLab R2011b, consecutively upgraded to Version R2014a, and once again the code can be found in the Appendix. The same is true for the Hilbert and Wavelet Transform functions utilized, with the latter being based on the "WAVOS" toolkit for MatLab, quoted in the bibliography. SDEs were solved using the "SDETools" plug-in by Andrew D. Horchler. In addition, early iterations of the Hilbert transform and other analysis tools were run in C and R, but were mostly superseded in the course of the project by the corresponding MatLab implementations.

4. Summary of Findings

4.1 Parameter Estimation via Sequential Monte Carlo

Having developed the simulation environment described in detail over the last two chapters, the following section will present and summarize the results obtained by using these tools to analyze the model and to contrast various potential extensions to it. As a first step, the SMC parameter estimation procedure is used to find a range of values capable of producing oscillations of the appropriate periods, namely 24 hours under regular and symmetric L/D cycles of that same duration, around 25 hours under constant darkness, and a suppression of rhythmic behaviour under conditions of constant light. The mean results obtained from the procedure are listed in Table 2. Interestingly, when contrasting a plot of the SMC results in Figure 19, it is apparent how some parameters, such as k2, will facilitate viable results over a very significant range, at least in combination with the variation of other parameters, while others, including v1 or k4, are much more tightly constrained. This pattern provides a valuable insight into the hierarchical importance of the various parameters, even if a careful analysis cannot stop with simply pointing to simple standard deviations, seeing how intrinsic system relevance is mingled with effects due to variable connectedness and degrees of freedom among parameters. In other words, less critical parameters would generally show a viability over a greater range of values, but even parameters of comparable relevance could differ significantly in this regard if, for instance, one of them is much more directly affected by changes in other parameters. It should also be noted in this context that values are not separately generated for the corresponding stochastic model. While the SMC algorithm was specifically implemented in a way to also handle stochastic validation simulations, scheduling for instance multiple runs and weighing the differing results obtained in this way, it is noted that initial results for stochastic runs, consisting after all of random shifts applied to the same underlying model, show a somewhat wider distribution, but comparable mean values. As such, a comprehensive parameter estimation for this model extension is not explicitly carried out at this point, also with a view to the time constraints of concluding this project, but could be easily implemented in the future on the basis of the existing simulation functions. It is deemed more pressing, however, to obtain a parameter value distribution for potential extensions to the system of differential equations, and so the procedure is repeated for an expanded model including the Rora stabilizing loop. As a final, but nevertheless very significant point, SMC runs checking the fit of intermediate populations not only against periods, but also the phases, represented by the timing of the maxima, of individual molecular species proved unable to approximate these reference values, even when repeated over a wide range of sample sizes, number of intermediate populations, and other specifications.

TABLE 2 PARAMETER VALUES GENERATED BY THE SMC PROCEDURE FOR THE BASE MODEL

V1	V2	kdm	kdeg	kd	K1	K2	K3	K4	ka	kp	Kcl1	Kcl2	Ll	n
0.46	0.46	0.60	0.61	0.35	0.52	0.53	0.40	0.49	0.74	0.69	0.45	0.60	0.70	0.18

FIGURE 19 MEAN AND STD OF SMC RESULTS FOR BASE MODEL COMPARED TO EXPECTED VALUES

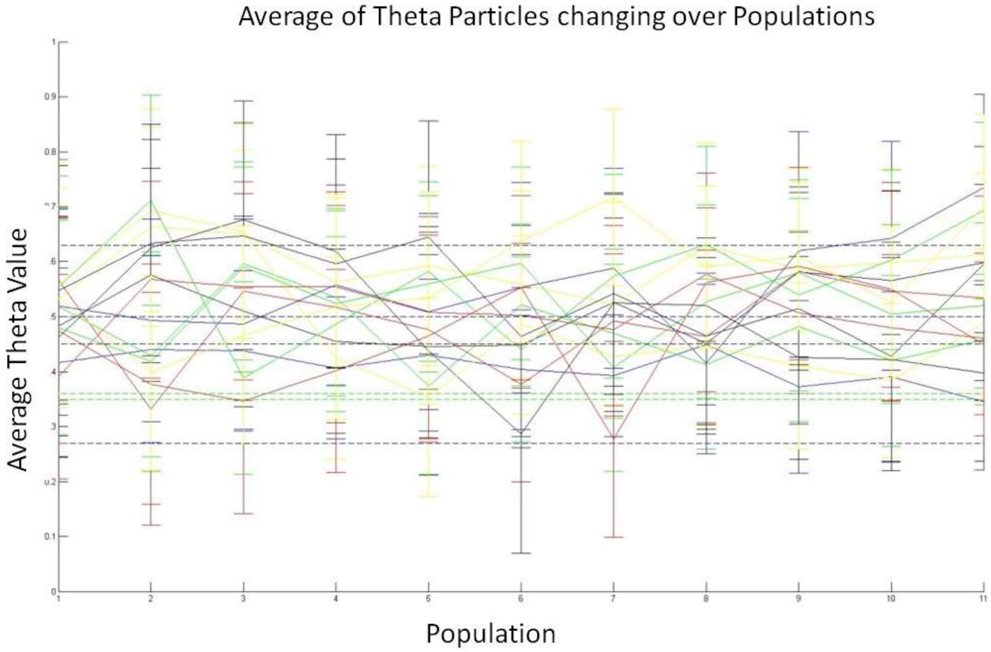
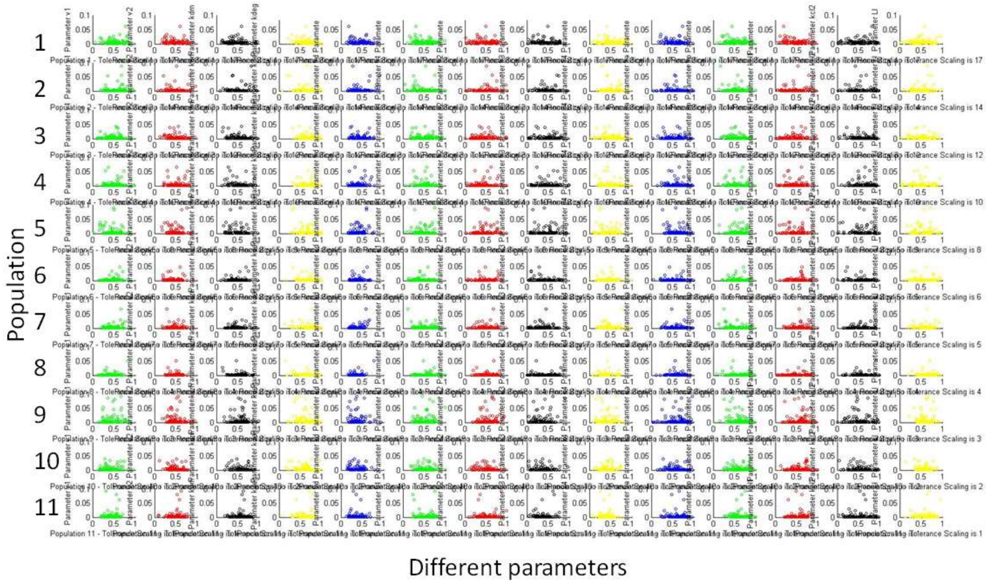


FIGURE 20 DISTRIBUTION OF SMC RESULTS FOR BASE MODEL



4.2 Sensitivity to Parameter Changes Across Different Models

In any case, the next step consists of entering the mean parameter values presented above into the two model system, i.e. the equations with and without the Rora stabilizing loop, and of forwarding to the additional analysis functions. Firstly, the results of the sensitivity analysis for the base model are presented in Figure 21, showing significant variation between the system's reaction to increases and decreases in certain parameters. Specifically, decreasing v1 by 10% appear sufficient to cause a drastic change in period, while the same is true for increasing v2 by 10%. Other parameters, on the other hand, appear to exert far less influence on the period duration, at least when being varied by only 10%. It is also interesting to note the relative effects on period versus amplitude, showing only very partial overlap and generally much great variation in amplitude. A comparison may be drawn to Figure 22, showing the results obtained by the same analysis on the basis of the extended model. Here, it can be seen that results differ significantly, with the extended model showing a generally more balance reaction across parameters for period, but very strong spikes in amplitude in a few cases. It is furthermore interesting to note the relationship to the results of the SMC procedure, seeing that the results for the former appeared much more in line with each other. Moving to a closer inspection of individual parameters, bifurcation analysis results for the base model can be found in Figure 23 and Figure 24, the latter comparing a bifurcation plot for the same parameter, but different molecular species, demonstrating that maximum and minimum concentration may vary depending on the selected variable, but that the typical shape in the bifurcation diagram is conserved, showing an onset of oscillation beyond 0.8 in each case. Notably, the bifurcation perspective provides a unique view into more precise dynamics contained with the range of generally viable parameter values, for instance when considering parameter kd, where oscillations appear to occur over a significant range, whereas v1 appears much more restrained. Following on from the contrast provided by the extended model in the preceding steps, it can be seen from the phase response curves generated for each iteration in Figure 25 and Figure 26, that there exist also difference in the reaction to light input. Once again no separate runs are carried out for stochastic varieties, seeing how not only random shifts would have necessitated averaging across a number of repeated runs, but also more significantly that these stochastic simulations are still based on the same underlying equations and parameters. However, when considering the overlap between the base and extended iterations of the model, it is certainly interesting to note what changes in behaviour can be linked to the inclusion of the stabilizing loop.

FIGURE 21 SENSITIVITY ANALYSIS FOR BASE MODEL

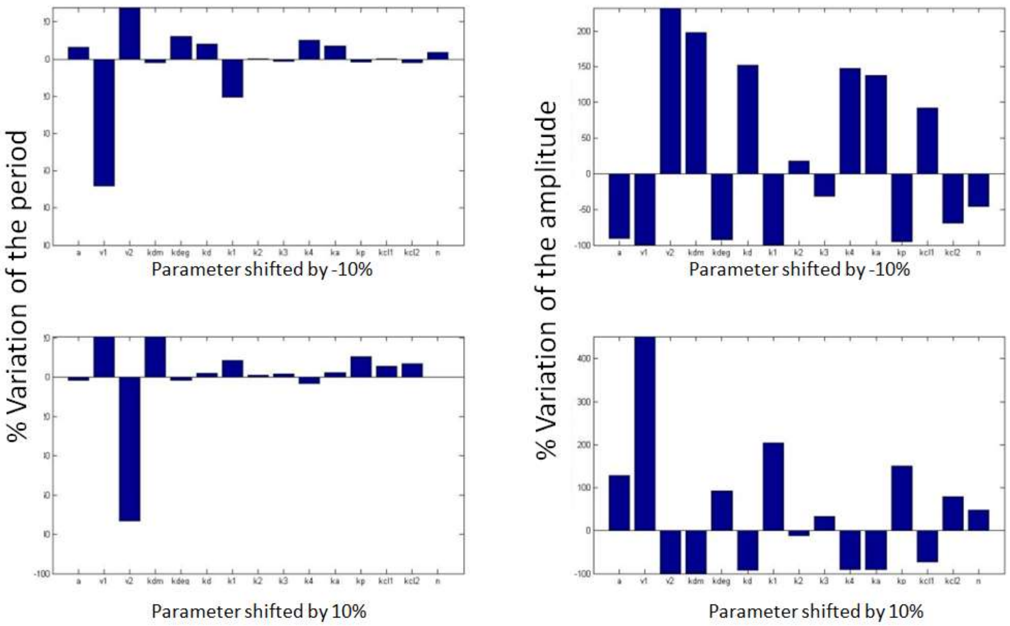


FIGURE 22 SENSITIVITY ANALYSIS FOR EXTENDED MODEL

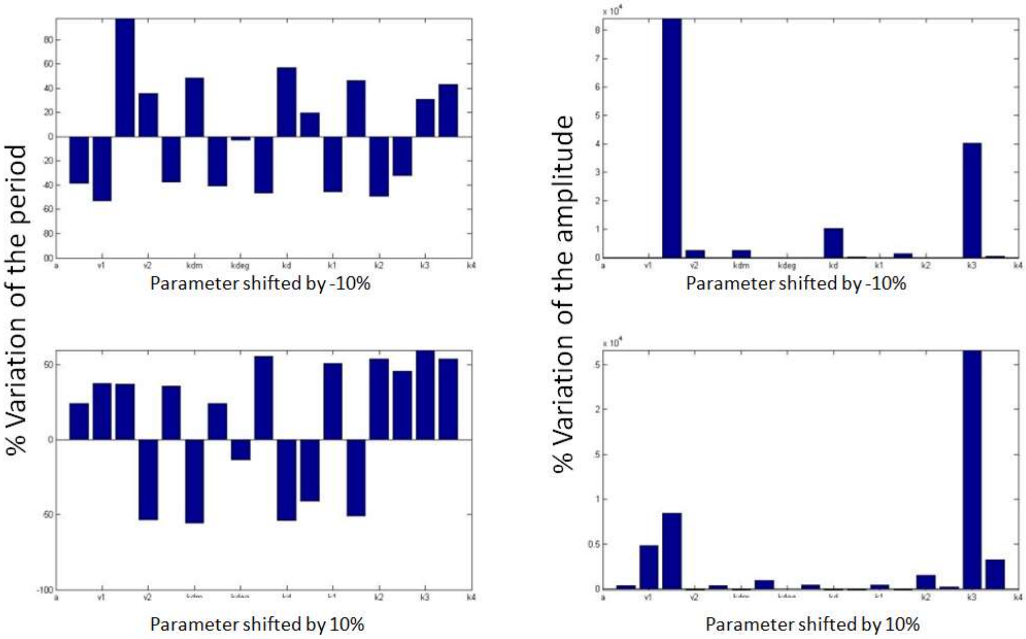
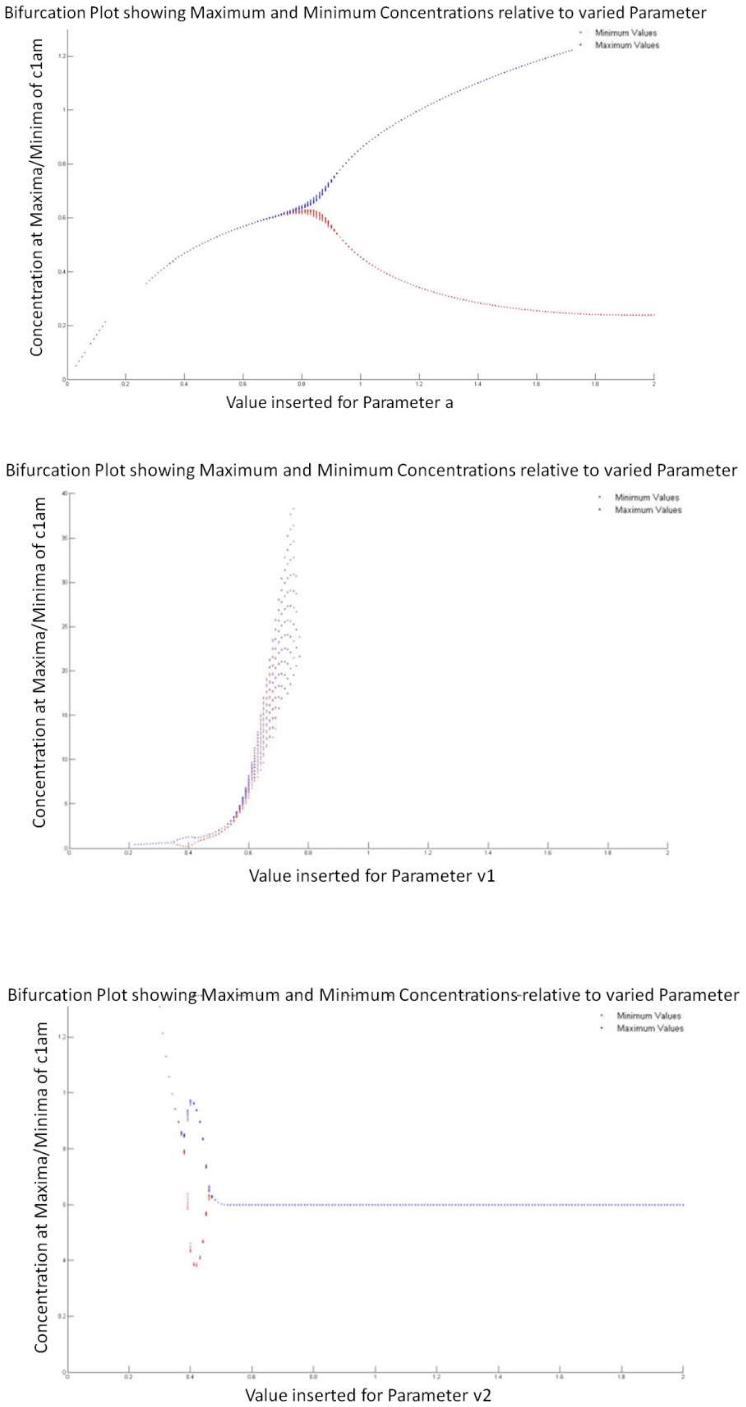
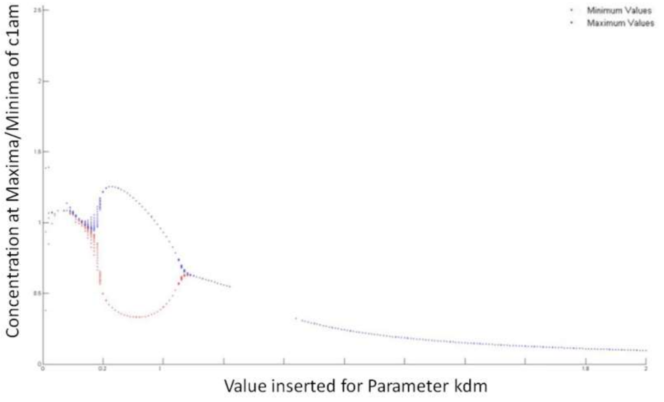


FIGURE 23 BIFURCATION FOR DIFFERENT PARAMETERS



Bifurcation Plot showing Maximum and Minimum Concentrations relative to varied Parameter



Bifurcation Plot showing Maximum and Minimum Concentrations relative to varied Parameter

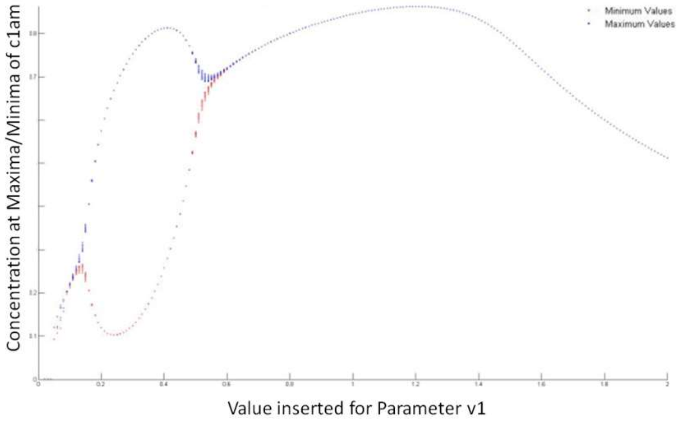


FIGURE 24 BIFURCATION FOR THE SAME PARAMETER, BUT MEASURED USING DIFFERENT VARIABLES

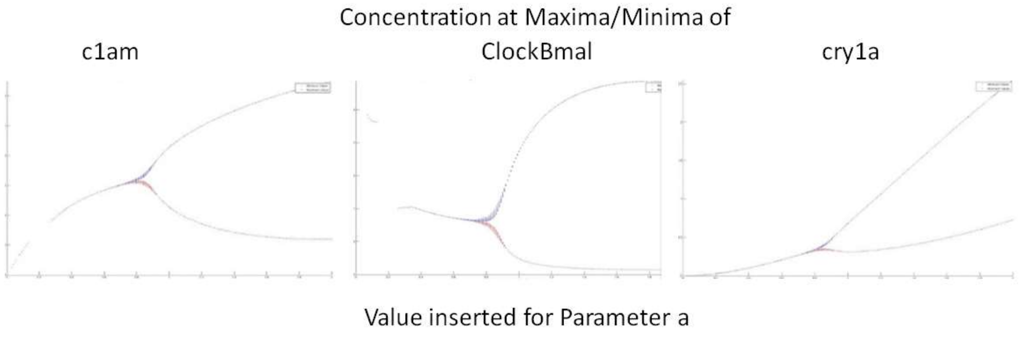


FIGURE 25 PHASE RESPONSE CURVE FOR BASE MODEL

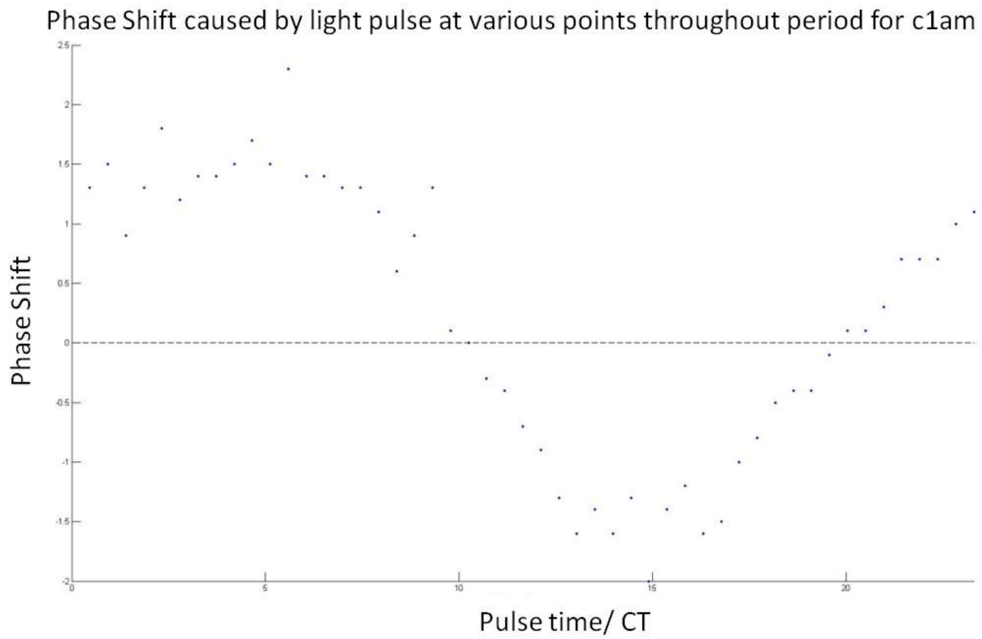
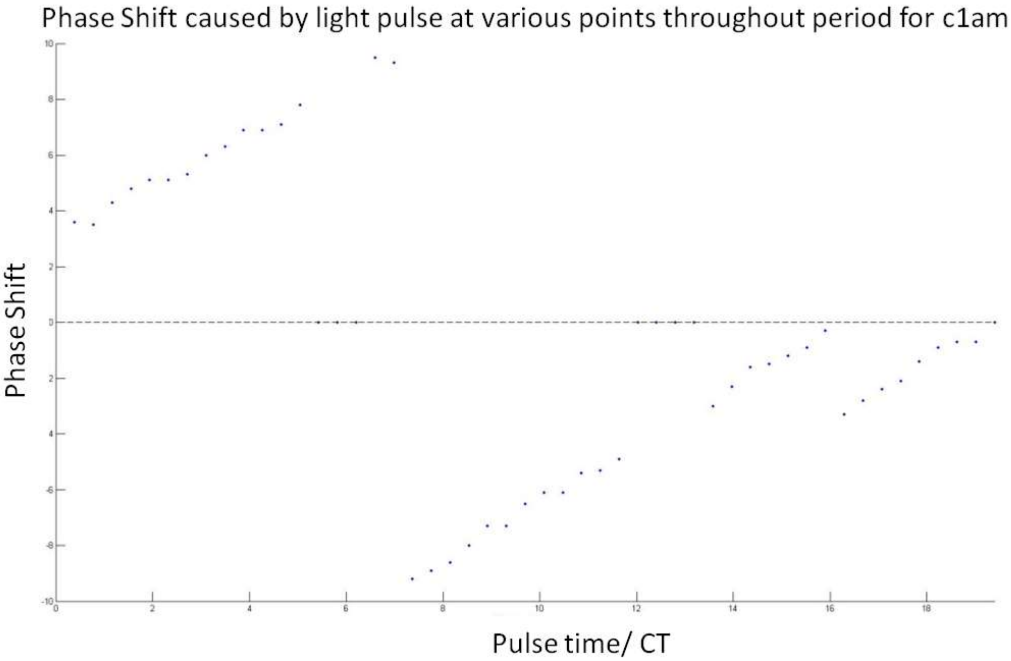


FIGURE 26 PHASE RESPONSE CURVE FOR EXTENDED MODEL



4.3 Deterministic and Stochastic Behaviour

Having considered the dynamics of the underlying parametric basis of the model, attention now shifts to the phenotypic behaviour displayed by base and extended models under different light regimes. It is also in this context that stochastic effects can be well appreciated, the noise element shifts results not only away from the deterministic bases, but also leads to considerable variation between individual stochastic runs. Consequently, in order to obtain a representative curve as the basis of an analysis the stochastic simulations are repeated a number of times and the mean of individual outputs is calculated, whereby a greater number of simulations will naturally lead to a smoother. The next visualization, Figure 27, puts the base model next to both the stochastic model. The most interesting revelations is reached, however, when comparing and contrasting the results under DD, that is conditions of constant darkness, as also shown in this figure. Here, a very significant deviation is observed between the stochastic and deterministic plots, as only the simulations containing a noise term exhibits a clear signal decay under these conditions. Plotting not only the mean curve, but also five random, individual stochastic, it is apparent that the individual amplitudes do not decrease, but rather the decay in overall amplitude appears to originate from the gradual shifting of the individual phases. In this way the stochastic model mirrors observations made in singe cells oscillation experiments, and thus demonstrates that deterministic and stochastic iterations may produce similar or even identical results under some experimental or, more generally, environmental conditions, but very disparate results under a different set of conditions. Moving on from this essential observation, the next objectives is to replicate the light pulse scenario investigated in the course of laboratory experiments as part of this project, thereby providing a direct comparison of biological and simulated data. Figure 28 display 15-min light pulse runs at different light intensities acting on oscillators previously, and subsequently, subjected to DD, and it can be seen that the stochastic model largely succeeds in replicating experimental results. Following on from this, it would appear that the addition of noise constitutes a critical element in this context of faithfully replicating the experimental setting *in silico*. In conclusion it has been demonstrated that this circadian clock model, constructed on the basis of the core molecular clock components reported for the zebrafish model organism, succeeds in replicating a range of important behaviours under various light conditions, including stable oscillations under LD cycles and arrest under LL conditions.

However, it is interesting to note that many finer points, as well as the predicted behaviour of signal decay under DD and following light pulse stimulation is only captured in a stochastic extension of the model. The inclusion of the stabilizing loop centered on Rora, on the other hand, was shown to significantly affect underlying dynamics, but with a more diffuse effect on behaviour under light regimes.

FIGURE 27 DETERMINISTIC VERSUS STOCHASTIC CURVES

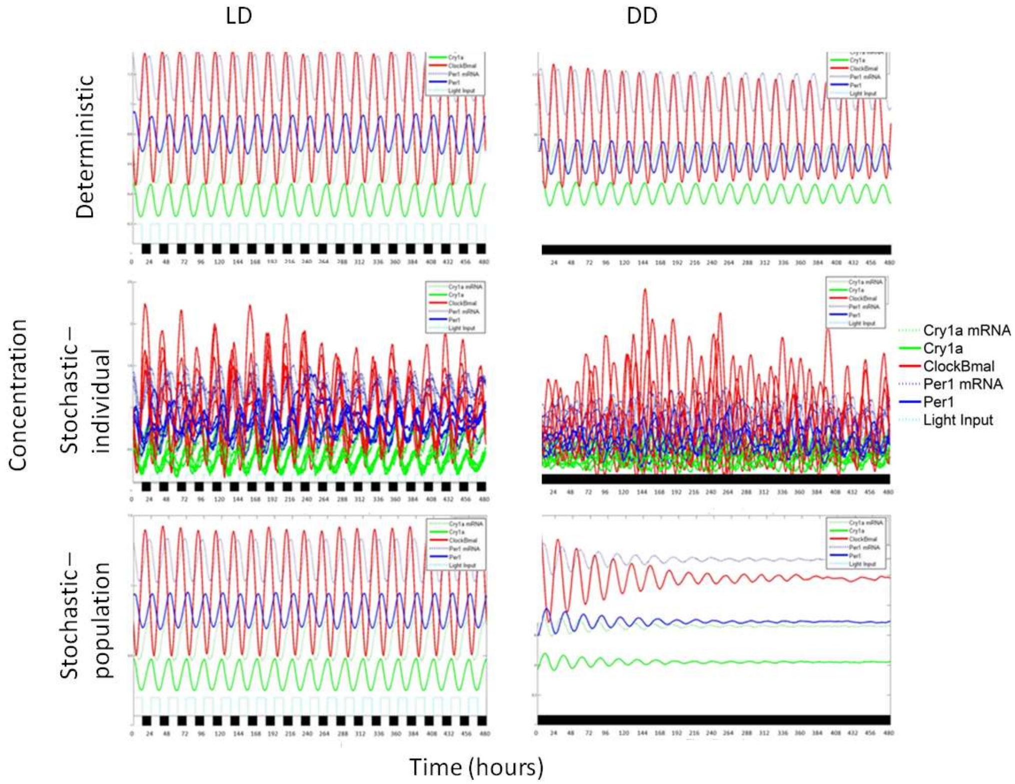
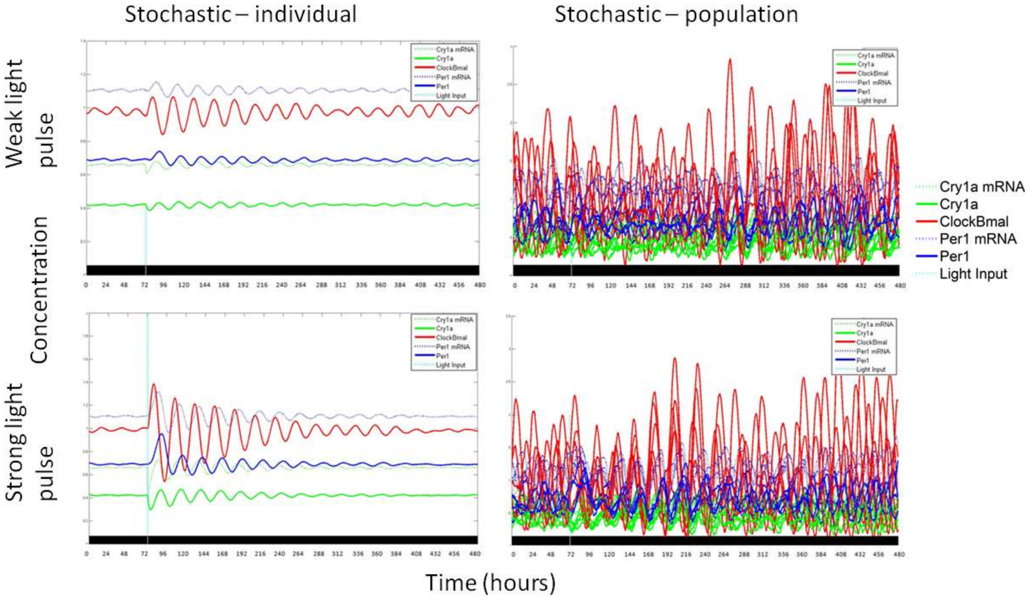


FIGURE 28 STOCHASTIC SIMULATION EXPOSED TO LIGHT PULSES



5. Discussion

Considering the overview of results provided, it first appears worth noting that the integrated model environment evidently succeeds in computing, with minimal manual input, simulation runs encompassing a range of light condition, passing outputs automatically on to a range of functions for further analysis, and finally generating plausible results. Moreover, the integration of a SMC parameter generator assists decidedly in adjusting the underlying parameters to changes in model specification, a task that has traditionally been characterized by significant trial and error, and this functionality therefore is not only capable of saving time and effort, but arguably also of introducing more rigor to this step. In fact, it has been noted in the literature that the highly nonlinear behaviour of complex models with numerous state variables and parameters makes it difficult to discern the relationship between these factors, and to distinguish essential from irrelevant or accidental features, and that there consequently exists a great need for optimization functions and analytical tools to facilitate fitting (Domijan & Rand 2015).

5.1 An Integrating Workflow Can Provide Novel Insights

Integrating parameter estimation and analysis tools into a single automated workflow to scan the entire parameter space for covariance with model outputs allows fresh insights into the dynamics of the model, firstly by identifying numeric values that much more precisely match the targets set for behavior under a range of light regimes, but secondly and just as importantly by pointing with near certainty to underlying limitations, such as the inability for the entire parameter space to replicate *in vivo* phase relationships.

Viewed as a whole, it can be confidently said that, following changes to the underlying equations in the form of e.g. additionally included reaction steps, altered interaction dynamics, or shifted external input terms, the simulation environment can be utilized to quasi-automatically prepare a comprehensive analysis; essentially a single mouse click can instruct the system to independently detect appropriate parameter values, insert these values into the model solver, run preliminary tests including sensitivity and bifurcation analysis, subject the new model to a range of simulated light regimes, and screen the results through several signal transform functions, generating finally not only a multitude of data arrays, but also a selection of visual outputs. This utility should significantly speed up the otherwise arduous process of evaluating changes to the underlying model, although it should be noted that on top of the minimal requirement for entering specifications, actual computation will require additional time ranging from mere seconds for short simulations, and minutes for analysis functions, to 48 hours or more for a thorough SMC parameter generation. On the other hand, especially this last step is generally only required following major model changes and can be skipped otherwise, and simulations were furthermore carried out mostly on a personal computer, having by now reached an age of over six years, so that actual time requirements on a modern, dedicated workstation would likely only measure a fraction of the values reported here. Moreover, considering that the author, coming from a background in the biological sciences, was new to programming as well as to many underlying mathematical techniques at the outset of this project, the successful implementation in code should go to show how much more accessible dedicated mathematical software has become over the years, requiring little prior knowledge to take on even relatively complex programming projects. As such, it would appear that this lowering of barriers to entry may contribute to a bright future of interdisciplinary research generally, and modeling specifically. Just as no architect would commence building on any sizeable project, or a professional investor would not commit capital, without both having carried out extensive modeling, it is imaginable that it might become equally uncommon in the future to conduct biological experiments, without first studying a corresponding *in silico* replication. In fact, certain key areas, such as pharmacological research, can already be observed very actively embracing the concept of integrating laboratory with computational work.

5.2 Basic Model Structure Replicates A Limited Set Responses

The basic set of circadian clock pathways incorporated in the model is found to replicate biological observations under a range of light regimes, such as DD, LL, and LD, but exhibits unexpected properties in other regards, such as a dilemma between a lack of stability in constant darkness on one hand, and the assumption of an overly strong reaction to short light pulses on the other.

More specifically to this project, it has been documented, as in numerous comparable research undertakings, that knowledge of molecular components and their broad interactions can be sufficient for constructing a functional mathematical model. However, it was noted that exposing this model to a range of conditions, especially including those likely more challenging to the underlying biological system, can help to distinguish the relative level of faithfulness and functionality. In particular, it was shown that the replication of basic oscillation behaviour was relatively easy to achieve, assuming perfectly regular light/dark cycles, whereas the introduction of external stimuli, in the form of light input, required significant modifications in order to match model output to the expected biological behaviour. In fact, it has been noted that even in its current state the model has proven unable to replicate certain key characteristics, such as the timing of peak phases, pointing to a need of further refinement, but maybe also the inherent limitations of representing highly complex interactions as a simplified model. In fact, one very noteworthy result lies in the fact that a critical behavioural adaption of the model system was achieved not through changes in parameter values, or the addition of extra equations, even if it has been found that the stabilizing Rora loop can significantly affect system dynamics, but rather by introducing a noise term and thereby changing the mathematical system from ODEs to SDEs.

5.3 Stochastic Behaviour as an Essential Circadian Feature

After evaluating different extensions to the model, including different entry points for light stimulation and the stabilizing Rora loop described above, a marked difference was achieved by considering populations of thousands of individual stochastic oscillators, thereby combining stability under base conditions with the ability to quickly adapt to even low levels of light stimulation on the basis of different points on the phase response curve.

Stochastic behaviour, which may otherwise be seen as a mere nuisance as indicated by the term "noise" and frequently assumed negligible in, for example, many likelihood function for parameter estimation (Domijan & Rand 2015), is in this way revealed as a vital feature of the circadian clock, and specifically as regards the interaction dynamics with different modes of light input. Indeed, these findings clearly support the notion of entrainment to an external stimulus, such as light in this case, acting through desynchronization and resynchronization effects between individual constituents, a mechanism that can only be made sense of in the presence of stochastic fluctuations. Indeed, recent studies of single cell levels of messenger RNA and proteins have revealed the presence of considerable heterogeneity, pointing to highly dynamic fluctuations over time. Specifically, the measurement of prolactin transcription via a reporter gene construct revealed clear cycles of transcriptional activity with an average period of approximately 11 h, which were furthermore shown to originate not from environmental effects, but intrinsic expression processes (Harper et al. 2011). In this case, as in others, heterogeneous expression patterns have been linked to the capacity for a flexible and differentiated cellular response, showing that stochastic behaviour may actually be a cornerstone of many key biological processes. While it may be argued at this point that the largely predictable behaviour of sufficiently sizeable stochastic populations could surely also be represented as a deterministic system, it is important to differentiate between the single oscillator and population level. Misleadingly equating one with the other and incorporating, if at all possible, population level effects, such as amplitude dampening in DD, into a framework based on singe cell molecular components through parametric distortions would, after all, lead to inherently flawed assumptions about the underlying reaction rates, likely to lead to clearly abnormal behaviour in at least some circumstances. Rather, the way forward may consist in more readily accepting, and embracing, the basic stochastic quality of many GRNs, as this aspect may also be relevant for putting other components, such as the stabilizing loop specifically evaluated as part of this project, into their proper perspective. After all, with random shifts emerging as such a pivotal driver of adaptation processes, it would only be consequential to suspect that physiological processes must be adapted to this circumstance on multiple levels.

5.4 Feeding Stochastic Predictions Into Parameter Estimation

One additional level of complexity that was programmatically implemented, but could not be successfully run at a statistically significant scale due to immense computational requirements, focuses on filtering the parameter space on the basis of groups of stochastic oscillators, rather than using the binary cutoff points of deterministic runs to supply parameters for the stochastic simulation.

Surely this area will constitute a field of much future research activity, and even as an extension of this project it may well prove worthwhile to re-evaluate the role of non-critical clock components in the context of stochastic fluctuations. Furthermore, while it was judged reasonable under the existing practical constraints to exclude noisy behaviour from the SMC parameter generation up to this point, future work may seek to integrate stochastic behaviour into the selection of the underlying parametric basis. While it is speculated that the result would mostly consist of a slight scattering of the deterministic distribution, it has to be accepted that the model is sufficiently complex to hinder exact predictions, especially when considering emergent behaviour only apparent at the population level. In this context it can be noted firstly that ongoing computational advancements may well have brought time requirements for a thorough SMC procedure using a stochastic validation function, which based on the deterministic runs may be extrapolated to last weeks, down into much more manageable magnitudes. Secondly, there have also been a number of analytical tools developed that may permit to probe the relationship, possibly using a constraint based approach, between passing stochastic versus deterministic equations to the SMC. A distinct shortcoming of the model likewise calling for further investigation consists of the unexpected peak phases currently obtained under the different light regimes. In the absence of light input as a factor and reference point, peak timings could evidently be assigned arbitrarily, and so it appears to follow that the source of the aberrant phases should be suspected in the light input pathway. Indeed, the light term is currently included in a very simple way, and so the key to resolving the phases in question may lie in formulating a more differentiated mechanism for altering the reactant concentrations in relation to the light input. Thankfully, in this case the simulation environment should really begin to shine, given enough computation time, allowing to easily evaluate the merit of a range of alternative light input terms. In detail, it might be especially interesting to investigate the use of intermediate steps or of self-limiting qualities. Finally, it should be noted that the existing functions could be used to simulate a wide range of light regimes, the results of which, whether they appear plausible or not, may inform the basis of future laboratory investigations, either probing an interesting biological prediction or trying to elucidate the source of unexpected model behaviour, thereby helping to further cement our theoretical understanding of circadian clock dynamics.

5.5 Future Outlook - Embracing Stochasticity

The scientific method, for all the immense advances in knowledge and understanding it has allowed us to claim, may have tempted some researches through an overzealous focus on clear statistical cutoff points and binary outcomes into a type of linear thinking, that does not pay sufficient respect to the inherent variability and stochasticity of our world.

Taking a step back, it can of course be safely predicted that the future will hold exciting new progress not only regarding our insights into the functioning of biological rhythms or modes of external stimulus propagation, but that also the computational tools utilized to investigate these settings are set to see a continuing fast-paced evolution. After all, this text already includes in the relevant sections several references to alternative, potentially even more sophisticated and powerful techniques than the ones employed here, even if significant hurdles persist to seeing these tools applied in a truly widespread way "on the ground", that is across biological research institutions. It is also before this background that this project, having started out with the use of numerous disparate functions across various programming environments before slowly moving towards the integrated setting achieved towards its culmination, may hope to contribute to the understanding that just as the algorithmic "engine block" of a function deserves focused attention, so should the implementation of streamlined data-handling and intuitive interface design, factors that could ultimately decide the practical use provided to those biological researchers without extensive programming knowledge. Another burning point to be addressed by the scientific community, as is felt by the author in the wake of this project, might lie in the widespread unease confronting the concept of uncertainty. This phenomenon in biomedical research has previously been termed, in a tongue-in-cheek manner of course, the "Human Linearity Virus", describing the tendency of scientists to try and press complex nonlinear systems into linear moulds (Cong et al. 2009). Having intensely ingrained the aim for precision, quantification, and reproducibility, it may indeed initially seem counterintuitive to pay heed to underlying noise and randomness, but as more and more findings of the pivotal role of these dimensions in many regulatory processes are reported, rather than assuming these factors negligible or simplifying them away, science may be better served by ultimately embracing the inherent stochasticity and non-linearity of biological existence. In this context the study of the circadian clock may play an important role in promoting the importance of stochastic variation, all the more as the circadian clock is not only an abstract research concept, but also holds special relevance for a host of medical applications. It has been reported, for instance, that circadian timing systems can directly affect tumour development, and very recently robust coupling between the circadian clock and cell cycle oscillators has been described (Feillet et al. 2015). It has been suggested on this basis that the circadian clock may directly synchronize or otherwise modulate the progression of the cell cycle, in turn having far reaching effects for not only tumour growth, but also a host of other pathological states. Consequently, it would appear that unlocking the secrets of the circadian clock might ultimately represent a key piece in making sense of our most essential and vital physiological processes at large.

Bibliography

Abeliovich, H., 2005. An empirical extremum principle for the hill coefficient in ligand-protein interactions showing negative cooperativity. *Biophysical journal*, 89(1), pp.76–9.

Addison, P.S., 2005. Wavelet transforms and the ECG: a review. *Physiological measurement*, 26(5), pp.R155–R199.

Adivar, B. & Selen, E.S., 2011. Compartmental disease transmission models for smallpox. *Discrete and Continuous Dynamical Systems- Series A*, (SUPPL.), pp.13–21.

Akman, O.E. et al., 2010. Robustness from flexibility in the fungal circadian clock. *BMC systems biology*, 4(1), p.88.

Alon, U., 2007. Network motifs: theory and experimental approaches. *Nature reviews. Genetics*, 8(6), pp.450–61.

Altinok, A. et al., 2009. Identifying mechanisms of chronotolerance and chronoefficacy for the anticancer drugs 5-fluorouracil and oxaliplatin by computational modeling. *Eur J Pharm Sci*, 36(1), pp.20–38.

Arda, H.E. et al., 2010. Functional modularity of nuclear hormone receptors in a Caenorhabditis elegans metabolic gene regulatory network. *Molecular systems biology*, 6, p.367.

Baggs, J.E. et al., 2009. Network features of the mammalian circadian clock. *PLoS biology*, 7(3), p.e52.

Bahar, R. et al., 2006. Increased cell-to-cell variation in gene expression in ageing mouse heart. *Nature*, 441(7096), pp.1011–4.

Banga, J.R., Moles, C.G. & Alonso, A.A., 2004. *Frontiers in Global Optimization: Global Optimization of Bioprocesses using Stochastic and Hybrid Methods* C. A. Floudas & P. Pardalos, eds., Boston, MA: Springer US.

Bar-Even, A. et al., 2006. Noise in protein expression scales with natural protein abundance. *Nature genetics*, 38(6), pp.636–43.

Bashan, A. et al., 2008. Comparison of detrending methods for fluctuation analysis. *Physica A: Statistical Mechanics and its Applications*, 387(21), pp.5080–5090.

Baumgartner, M. et al., 2006. Solving ODEs with Mathematica - ControlsWiki. *controls.engin.umich.edu/wiki/index.php/Solving_ODEs_with_Mathematica*. Available at: https://controls.engin.umich.edu/wiki/index.php/Solving_ODEs_with_Mathematica [Accessed August 24, 2015].

Beaver, L.M. et al., 2002. Loss of circadian clock function decreases reproductive fitness in males of Drosophila melanogaster. *Proceedings of the National Academy of Sciences of the United States of America*, 99(4), pp.2134–9.

Bernard, S. et al., 2010. Tumor growth rate determines the timing of optimal chronomodulated treatment schedules. *PLoS Comput Biol*, 6(3), p.e1000712.

Blake, W.J. et al., 2006. Phenotypic consequences of promoter-mediated transcriptional noise. *Molecular cell*, 24(6), pp.853–65.

Boettiger, A.N. & Levine, M., 2009. Synchronous and stochastic patterns of gene activation in the Drosophila embryo. *Science (New York, N.Y.)*, 325(5939), pp.471–3.

Bordbar, A. et al., 2014. Constraint-based models predict metabolic and associated cellular functions. *Nature reviews. Genetics*, 15(2), pp.107–20.

Boyer, L.A. et al., 2005. Core transcriptional regulatory circuitry in human embryonic stem cells. *Cell*, 122(6), pp.947–56.

Brady, S.M. et al., 2011. A stele-enriched gene regulatory network in the Arabidopsis root. *Molecular systems biology*, 7, p.459.

Brem, R.B. et al., 2002. Genetic dissection of transcriptional regulation in budding yeast. *Science (New York, N.Y.)*, 296(5568), pp.752–5.

Butcher, J.C., 2000. Numerical methods for ordinary differential equations in the 20th century. *Journal of Computational and Applied Mathematics*, 125(1-2), pp.1–29.

Cahill, G.M., 2002. Clock mechanisms in zebrafish. *Cell Tissue Res*, 309(1), pp.27–34.

Cao, Y., Gillespie, D.T. & Petzold, L.R., 2005. Avoiding negative populations in explicit Poisson tau-leaping. *Journal of Chemical Physics*, 123(5).

Cao, Y., Gillespie, D.T. & Petzold, L.R., 2006. Efficient step size selection for the tau-leaping simulation method. *The Journal of chemical physics*, 124(4), p.044109.

Carmona, R., Hwang, W.-L. & Torresani, B., 1998. *Practical Time-Frequency Analysis: Gabor and Wavelet Transforms, with an Implementation in S*, Academic Press.

Carmona, R.A., Hwang, W.L. & Torresani, B., 1999. Multiridge detection and time-frequency reconstruction. *IEEE Transactions on Signal Processing*, 47(2), pp.480–492.

Carr, A.-J.F. & Whitmore, D., 2005. Imaging of single light-responsive clock cells reveals fluctuating free-running periods. *Nat Cell Biol*, 7(3), pp.319–321.

Casares, F., Calleja, M. & Sánchez-Herrero, E., 1996. Functional similarity in appendage specification by the Ultrabithorax and abdominal-A Drosophila HOX genes. *The EMBO journal*, 15(15), pp.3934–42.

Casey, R., de Jong, H. & Gouzé, J.-L., 2006. Piecewise-linear models of genetic regulatory networks: equilibria and their stability. *Journal of mathematical biology*, 52(1), pp.27–56.

Cazelles, B. et al., 2008. Wavelet analysis of ecological time series. *Oecologia*, 156(2), pp.287–304.

Chang, H.H. et al., 2008. Transcriptome-wide noise controls lineage choice in mammalian progenitor cells. *Nature*, 453(7194), pp.544–547.

Chen, Y. et al., 2015. Luciferase reporter gene assay on human 5-HT receptor: which response element should be chosen? *Scientific reports*, 5, p.8060.

Choi, J.K. & Kim, Y.-J., 2009. Intrinsic variability of gene expression encoded in nucleosome positioning sequences. *Nature genetics*, 41(4), pp.498–503.

Cong, F. et al., 2009. Hilbert-Huang versus Morlet wavelet transformation on mismatch negativity of children in uninterrupted sound paradigm. *Nonlinear biomedical physics*, 3(1), p.1.

Cornish-Bowden, A., 2013. The origins of enzyme kinetics. *FEBS letters*, 587(17), pp.2725–30.

Costa, M.J. et al., 2013. Inference on periodicity of circadian time series. *Biostatistics (Oxford, England)*, 14(4), pp.792–806.

Coutinho, R. et al., 2006. Discrete time piecewise affine models of genetic regulatory networks. *Journal of mathematical biology*, 52(4), pp.524–70.

Crick Mc, F., 1970. Central Dogma of Molecular Biology. *NATURE*, 227(8).

Curtis, A.M. et al., 2007. Circadian variation of blood pressure and the vascular response to asynchronous stress. *Proceedings of the National Academy of Sciences of the United States of America*, 104(9), pp.3450–5.

Czeisler, C.A. & Klerman, E.B., 1999. Circadian and sleep-dependent regulation of hormone release in humans. *Recent progress in hormone research*, 54, pp.97–130; discussion 130–2.

Dardente, H. et al., 2007. Cryptochromes impair phosphorylation of transcriptional activators in the clock: a general mechanism for circadian repression. *The Biochemical journal*, 402(3), pp.525–36.

Daubechies, I., 1992. *Ten Lectures on Wavelets (Society for Industrial and Applied Mathematics)*, Society for Industrial and Applied Mathematics.

Davidson, A.J. et al., 2006. Chronic jet-lag increases mortality in aged mice. *Current biology : CB*, 16(21), pp.R914–6.

DeCoursey, P.J. & Krulas, J.R., 1998. Behavior of SCN-lesioned chipmunks in natural habitat: a pilot study. *Journal of biological rhythms*, 13(3), pp.229–44.

Dixon, S.J. et al., 2009. Systematic mapping of genetic interaction networks. *Annual review of genetics*, 43, pp.601–25.

Domijan, M. & Rand, D.A., 2015. Using constraints and their value for optimization of large ODE systems. *Journal of the Royal Society, Interface / the Royal Society*, 12(104), p.20141303.

Dormand, J.R. & Prince, P.J., 1980. A family of embedded Runge-Kutta formulae. *Journal of Computational and Applied Mathematics*, 6(1), pp.19–26.

El-Dahshan, E.-S.A., 2010. Genetic algorithm and wavelet hybrid scheme for ECG signal denoising. *Telecommunication Systems*, 46(3), pp.209–215.

Etchegaray, J.-P. et al., 2010. Casein kinase 1 delta (CK1delta) regulates period length of the mouse suprachiasmatic circadian clock in vitro. *PloS one*, 5(4), p.e10303.

Fathallah-Shaykh, H.M., Bona, J.L. & Kadener, S., 2009. Mathematical model of the Drosophila circadian clock: loop regulation and transcriptional integration. *Biophys J*, 97(9), pp.2399–2408.

Feillet, C. et al., 2015. Coupling between the Circadian Clock and Cell Cycle Oscillators: Implication for Healthy Cells and Malignant Growth. *Frontiers in Neurology*, 6.

Filipski, E. et al., 2004. Effects of chronic jet lag on tumor progression in mice. *Cancer research*, 64(21), pp.7879–85.

Gery, S. et al., 2006. The circadian gene per1 plays an important role in cell growth and DNA damage control in human cancer cells. *Molecular cell*, 22(3), pp.375–82.

Ghosh, B., Karmakar, R. & Bose, I., 2005. Noise characteristics of feed forward loops. *Physical biology*, 2(1), pp.36–45.

Gilbert, S.F., 2000. Developmental Biology.

Gillespie, D., 1977. Exact stochastic simulation of coupled chemical reactions. *The journal of physical chemistry*, 81(25), pp.2340–2361.

Gillespie, D.T., 1992. A rigorous derivation of the chemical master equation. *Physica A: Statistical Mechanics and its Applications*, 188(1-3), pp.404–425.

Gonze, D. et al., 2005. Spontaneous synchronization of coupled circadian oscillators. *Biophys J*, 89(1), pp.120–129.

Gonze, D. et al., 2003. Stochastic models for circadian rhythms: effect of molecular noise on periodic and chaotic behaviour. *C R Biol*, 326(2), pp.189–203.

Gonze, D. & Goldbeter, A., 2006. Circadian rhythms and molecular noise. *Chaos*, 16(2), p.26110.

Gonze, D., Halloy, J. & Goldbeter, A., 2002. Robustness of circadian rhythms with respect to molecular noise. *Proceedings of the National Academy of Sciences of the United States of America*, 99(2), pp.673–8.

Goodwin, B.C., 1965. Oscillatory behavior in enzymatic control processes. *Advances in Enzyme Regulation*, 3, pp.425 – 428, IN1–IN2, 429–430, IN3–IN6, 431–437.

Görl, M. et al., 2001. A PEST-like element in FREQUENCY determines the length of the circadian period in Neurospora crassa. *The EMBO journal*, 20(24), pp.7074–84.

Goutelle, S. et al., 2008. The Hill equation: a review of its capabilities in pharmacological modelling. *Fundamental & clinical pharmacology*, 22(6), pp.633–48.

Granada, A.E. & Herzel, H., 2009. How to achieve fast entrainment? The timescale to synchronization. *PLoS One*, 4(9), p.e7057.

Han, M., 1992. Ras proteins in developmental pattern formation in Caenorhabditis elegans and Drosophila. *Seminars in cancer biology*, 3(4), pp.219–28.

Harang, R., Bonnet, G. & Petzold, L.R., 2012. WAVOS: a MATLAB toolkit for wavelet analysis and visualization of oscillatory systems. *BMC research notes*, 5, p.163.

Harper, C. V et al., 2011. Dynamic analysis of stochastic transcription cycles. *PLoS biology*, 9(4), p.e1000607.

Hasenauer, J. et al., 2014. ODE constrained mixture modelling: a method for unraveling subpopulation structures and dynamics. *PLoS computational biology*, 10(7), p.e1003686.

Hastings, M., O'Neill, J.S. & Maywood, E.S., 2007. Circadian clocks: regulators of endocrine and metabolic rhythms. *The Journal of endocrinology*, 195(2), pp.187–98.

Hastings, M.H., Reddy, A.B. & Maywood, E.S., 2003. A clockwork web: circadian timing in brain and periphery, in health and disease. *Nature reviews. Neuroscience*, 4(8), pp.649–61.

Helfrich-Förster, C., 2001. The locomotor activity rhythm of Drosophila melanogaster is controlled by a dual oscillator system. *Journal of Insect Physiology*, 47(8), pp.877–887.

Hendrix, D.A. et al., 2008. Promoter elements associated with RNA Pol II stalling in the Drosophila embryo. *Proceedings of the National Academy of Sciences of the United States of America*, 105(22), pp.7762–7.

Herzog, E.D. et al., 2004. Temporal precision in the mammalian circadian system: a reliable clock from less reliable neurons. *Journal of biological rhythms*, 19(1), pp.35–46.

Hill, A. V, 1910. The possible effects of the aggregation of the molecules of haemoglobin on its dissociation curves. *The Journal of Physiology*, 40(Suppl), pp.i–vii.

Hill, A.J. et al., 2005. Zebrafish as a model vertebrate for investigating chemical toxicity. *Toxicological sciences : an official journal of the Society of Toxicology*, 86(1), pp.6–19.

Hirayama, J. et al., 2009. Common light signaling pathways controlling DNA repair and circadian clock entrainment in zebrafish. *Cell Cycle*, 8(17), pp.2794–2801.

Holoch, D. & Moazed, D., 2015. RNA-mediated epigenetic regulation of gene expression. *Nature Reviews Genetics*, 16(2), pp.71–84.

Howe, K. et al., 2013. The zebrafish reference genome sequence and its relationship to the human genome. *Nature*, 496(7446), pp.498–503.

Ishida, A. et al., 2005. Light activates the adrenal gland: timing of gene expression and glucocorticoid release. *Cell metabolism*, 2(5), pp.297–307.

J. Miraglia, L., J. King, F. & Damoiseaux, R., 2011. Seeing the Light: Luminescent Reporter Gene Assays. *Combinatorial Chemistry & High Throughput Screening*, 14(8), pp.648–657.

Jones, A.W., 2010. Evidence-based survey of the elimination rates of ethanol from blood with applications in forensic casework. *Forensic science international*, 200(1-3), pp.1–20.

Jothi, R. et al., 2009. Genomic analysis reveals a tight link between transcription factor dynamics and regulatory network architecture. *Molecular systems biology*, 5, p.294.

Kalsbeek, A. et al., 2006. SCN outputs and the hypothalamic balance of life. *Journal of biological rhythms*, 21(6), pp.458–69.

Kaneko, M., Hernandez-Borsetti, N. & Cahill, G.M., 2006. Diversity of zebrafish peripheral oscillators revealed by luciferase reporting. *Proc Natl Acad Sci U S A*, 103(39), pp.14614–14619.

Kasowski, M. et al., 2010. Variation in transcription factor binding among humans. *Science (New York, N.Y.)*, 328(5975), pp.232–5.

Kiyohara, Y.B. et al., 2006. The BMAL1 C terminus regulates the circadian transcription feedback loop. *Proceedings of the National Academy of Sciences of the United States of America*, 103(26), pp.10074–9.

Klipp, E. et al., 2009. *Systems biology: a textbook*, Wiley-VCH.

Knutson, K.L. et al., 2007. The metabolic consequences of sleep deprivation. *Sleep medicine reviews*, 11(3), pp.163–78.

Ko, C.H. & Takahashi, J.S., 2006. Molecular components of the mammalian circadian clock. *Human molecular genetics*, 15 Spec No(suppl_2), pp.R271–7.

Konopka, R.J. & Benzer, S., 1971. Clock mutants of Drosophila melanogaster. *Proceedings of the National Academy of Sciences of the United States of America*, 68(9), pp.2112–6.

Kornmann, B. et al., 2007. System-driven and oscillator-dependent circadian transcription in mice with a conditionally active liver clock. *PLoS biology*, 5(2), p.e34.

Kou, S.C. et al., 2005. Single-molecule Michaelis-Menten equations. *The journal of physical chemistry. B*, 109(41), pp.19068–81.

Krishna, S., Semsey, S. & Jensen, M.H., 2009. Frustrated bistability as a means to engineer oscillations in biological systems. *Physical biology*, 6(3), p.036009.

Krivova, N. a. & Solanki, S.K., 2002. The 1.3-year and 156-day periodicities in sunspot data: Wavelet analysis suggests a common origin. *Astronomy and Astrophysics*, 394(2), pp.701–706.

Kuttykrishnan, S. et al., 2010. A quantitative model of glucose signaling in yeast reveals an incoherent feed forward loop leading to a specific, transient pulse of transcription. *Proceedings of the National Academy of Sciences of the United States of America*, 107(38), pp.16743–8.

Lehner, B., 2008. Selection to minimise noise in living systems and its implications for the evolution of gene expression. *Molecular systems biology*, 4, p.170.

Leloup, J. et al., 2003. Toward a detailed computational model for the mammalian circadian clock. *Proc Natl Acad Sci U S A*, 100(12), pp.7051–7056.

Levi, F. & Schibler, U., 2007. Circadian rhythms: mechanisms and therapeutic implications. *Annual review of pharmacology and toxicology*, 47, pp.593–628.

Levine, J.D. et al., 2002. Signal analysis of behavioral and molecular cycles. *BMC neuroscience*, 3, p.1.

Liang, H. & Wu, H., 2008. Parameter Estimation for Differential Equation Models Using a Framework of Measurement Error in Regression Models. *Journal of the American Statistical Association*, 103(484), pp.1570–1583.

Liu, S. & Leach, S.D., 2011. Zebrafish models for cancer. *Annual review of pathology*, 6, pp.71–93.

Liu, Y., 2011. *Hilbert Transform and Applications*,

Locke, J., Southern, M.M., et al., 2005. Extension of a genetic network model by iterative experimentation and mathematical analysis. *Molecular systems biology*, 1(Ll), p.2005.0013.

Locke, J., Millar, A. & Turner, M., 2005. Modelling genetic networks with noisy and varied experimental data: the circadian clock in Arabidopsis thaliana. *Journal of Theoretical Biology*, 234(3), pp.383–393.

Locke, J.C.W. et al., 2006. Experimental validation of a predicted feedback loop in the multi-oscillator clock of Arabidopsis thaliana. *Mol Syst Biol*, 2, p.59.

Loening, A.M., Dragulescu-Andrasi, A. & Gambhir, S.S., 2010. A red-shifted Renilla luciferase for transient reporter-gene expression. *Nature methods*, 7(1), pp.5–6.

Lomvardas, S. et al., 2006. Interchromosomal interactions and olfactory receptor choice. *Cell*, 126(2), pp.403–13.

Luscombe, N.M. et al., 2004. Genomic analysis of regulatory network dynamics reveals large topological changes. *Nature*, 431(7006), pp.308–312.

Ma, W. et al., 2009. Defining network topologies that can achieve biochemical adaptation. *Cell*, 138(4), pp.760–73.

Macneil, L.T. & Walhout, A.J.M., 2011. Gene regulatory networks and the role of robustness and stochasticity in the control of gene expression. *Genome research*, 21(5), pp.645–57.

Major, R.J. & Poss, K.D., 2007. Zebrafish Heart Regeneration as a Model for Cardiac Tissue Repair. *Drug discovery today. Disease models*, 4(4), pp.219–225.

Maston, G.A., Evans, S.K. & Green, M.R., 2006. Transcriptional Regulatory Elements in the Human Genome.

Meeker, K. et al., 2011. Wavelet measurement suggests cause of period instability in mammalian circadian neurons. *Journal of biological rhythms*, 26(4), pp.353–62.

Merrow, M., Brunner, M. & Roenneberg, T., 1999. Assignment of circadian function for the Neurospora clock gene frequency. *Nature*, 399(6736), pp.584–6.

Mickens, R.E., 2003. A nonstandard finite-difference scheme for the Lotka–Volterra system. *Applied Numerical Mathematics*, 45(2-3), pp.309–314.

Miyamura, N. et al., 2009. CLOCK:BMAL-independent circadian oscillation of zebrafish Cryptochrome1a gene. *Biol Pharm Bull*, 32(7), pp.1183–1187.

Mombaerts, P., 1999. Molecular biology of odorant receptors in vertebrates. *Annual review of neuroscience*, 22, pp.487–509.

Murray, J.D., 2002. *Mathematical Biology : I . An Introduction , Third Edition*,

Necsulea, A. & Kaessmann, H., 2014. Evolutionary dynamics of coding and non-coding transcriptomes. *Nature reviews. Genetics*, 15(11), pp.734–48.

Novák, B. & Tyson, J.J., 2008. Design principles of biochemical oscillators. *Nat Rev Mol Cell Biol*, 9(12), pp.981–991.

O'Neill, J.S. & Reddy, A.B., 2011. Circadian clocks in human red blood cells. *Nature*, 469(7331), pp.498–503.

Ohta, H., Yamazaki, S. & McMahon, D.G., 2005. Constant light desynchronizes mammalian clock neurons. *Nat Neurosci*, 8(3), pp.267–269.

Ong, C.-T. & Corces, V.G., 2011. Enhancer function: new insights into the regulation of tissue-specific gene expression. *Nature reviews. Genetics*, 12(4), pp.283–93.

Oster, H. et al., 2006. The circadian rhythm of glucocorticoids is regulated by a gating mechanism residing in the adrenal cortical clock. *Cell metabolism*, 4(2), pp.163–73.

Ouyang, Y. et al., 1998. Resonating circadian clocks enhance fitness in cyanobacteria. *Proceedings of the National Academy of Sciences of the United States of America*, 95(15), pp.8660–4.

Ow, M.C. et al., 2008. The FLYWCH transcription factors FLH-1, FLH-2, and FLH-3 repress embryonic expression of microRNA genes in C. elegans. *Genes & development*, 22(18), pp.2520–34.

Pedraza, J.M. & van Oudenaarden, A., 2005. Noise Propagation in Gene Networks. *Science*, 307(5717), pp.1965–1969.

Peifer, M. & Timmer, J., 2007. Parameter estimation in ordinary differential equations for biochemical processes using the method of multiple shooting. *IET systems biology*, 1(2), pp.78–88.

Peng, C.K. et al., 1995. Quantification of scaling exponents and crossover phenomena in nonstationary heartbeat time series. *Chaos (Woodbury, N.Y.)*, 5(1), pp.82–7.

Peng, Z.K., Tse, P.W. & Chu, F.L., 2005. A comparison study of improved Hilbert-Huang transform and wavelet transform: Application to fault diagnosis for rolling bearing. *Mechanical Systems and Signal Processing*, 19(5), pp.974–988.

Perry, M.W. et al., 2010. Shadow enhancers foster robustness of Drosophila gastrulation. *Current biology : CB*, 20(17), pp.1562–7.

Pikovsky, A., Rosenblum, M. & Kurths, J., 2001. *Synchronization*, Cambridge University Press.

Plautz, J.D. et al., 1997. Quantitative analysis of Drosophila period gene transcription in living animals. *Journal of biological rhythms*, 12(3), pp.204–17.

Pokhilko, A. et al., 2010. Data assimilation constrains new connections and components in a complex, eukaryotic circadian clock model. *Mol Syst Biol*, 6(416), p.416.

Polynikis, A., Hogan, S.J. & di Bernardo, M., 2009. Comparing different ODE modelling approaches for gene regulatory networks. *Journal of theoretical biology*, 261(4), pp.511–30.

Prinz, H., 2010. Hill coefficients, dose-response curves and allosteric mechanisms. *Journal of chemical biology*, 3(1), pp.37–44.

Qiu, X. et al., 2005. Induction of photosensitivity by heterologous expression of melanopsin. *Nature*, 433(7027), pp.745–9.

Rabbani, H. et al., 2011. R peak detection in electrocardiogram signal based on an optimal combination of wavelet transform, hilbert transform, and adaptive thresholding. *Journal of medical signals and sensors*, 1(2), pp.91–8.

Raj, A. et al., 2010. Variability in gene expression underlies incomplete penetrance. *Nature*, 463(7283), pp.913–8.

Recio-Spinoso, A., Fan, Y.-H. & Ruggero, M.A., 2011. Basilar-membrane responses to broadband noise modeled using linear filters with rational transfer functions. *IEEE transactions on bio-medical engineering*, 58(5), pp.1456–65.

Reddy, A.B. et al., 2006. Circadian orchestration of the hepatic proteome. *Current biology : CB*, 16(11), pp.1107–15.

Reinius, B. & Sandberg, R., 2015. Random monoallelic expression of autosomal genes: stochastic transcription and allele-level regulation. *Nature reviews. Genetics*, 16(11), pp.653–64.

Ripperger, J.A. & Schibler, U., 2006. Rhythmic CLOCK-BMAL1 binding to multiple E-box motifs drives circadian Dbp transcription and chromatin transitions. *Nature genetics*, 38(3), pp.369–74.

Roenneberg, T. et al., 2010. Entrainment concepts revisited. *J Biol Rhythms*, 25(5), pp.329–339.

Roignant, J.-Y. & Treisman, J.E., 2009. Pattern formation in the Drosophila eye disc. *The International journal of developmental biology*, 53(5-6), pp.795–804.

Rotem, E. et al., 2010. Regulation of phenotypic variability by a threshold-based mechanism underlies bacterial persistence. *Proceedings of the National Academy of Sciences of the United States of America*, 107(28), pp.12541–6.

Rudic, R.D. et al., 2004. BMAL1 and CLOCK, two essential components of the circadian clock, are involved in glucose homeostasis. *PLoS biology*, 2(11), p.e377.

Ruoff, P. et al., 1999. The Goodwin Oscillator: On the Importance of Degradation Reactions in the Circadian Clock. *Journal of Biological Rhythms*, 14(6), pp.469–479.

Sato, T.K. et al., 2004. A functional genomics strategy reveals Rora as a component of the mammalian circadian clock. *Neuron*, 43(4), pp.527–37.

Schernhammer, E.S. et al., 2006. Night work and risk of breast cancer. *Epidemiology (Cambridge, Mass.)*, 17(1), pp.108–11.

Schmidt, D. et al., 2010. Five-vertebrate ChIP-seq reveals the evolutionary dynamics of transcription factor binding. *Science (New York, N.Y.)*, 328(5981), pp.1036–40.

Sookoian, S. et al., 2007. Effects of rotating shift work on biomarkers of metabolic syndrome and inflammation. *Journal of internal medicine*, 261(3), pp.285–92.

Sprague, J. et al., 2003. The Zebrafish Information Network (ZFIN): the zebrafish model organism database. *Nucleic acids research*, 31(1), pp.241–3.

Steiner, A.B. et al., 2014. Dynamic gene expression by putative hair-cell progenitors during regeneration in the zebrafish lateral line. *Proceedings of the National Academy of Sciences of the United States of America*, 111(14), pp.E1393–401.

Storch, K.-F. et al., 2002. Extensive and divergent circadian gene expression in liver and heart. *Nature*, 417(6884), pp.78–83.

Sueur, J.J., Aubin, T. & Simonis, C., 2008. Equipment Review: Seewave, a free modular tool for sound analysis and synthesis. *Bioacoustics*, 18(2), p.226.

Swain, P.S., Elowitz, M.B. & Siggia, E.D., 2002. Intrinsic and extrinsic contributions to stochasticity in gene expression. *Proc Natl Acad Sci U S A*, 99(20), pp.12795–12800.

Tamai, T.K., Carr, A.J. & Whitmore, D., 2005. Zebrafish circadian clocks: cells that see light. *Biochem Soc Trans*, 33(Pt 5), pp.962–966.

Tamai, T.K., Young, L.C. & Whitmore, D., 2007. Light signaling to the zebrafish circadian clock by Cryptochrome 1a. *Proc Natl Acad Sci U S A*, 104(37), pp.14712–14717.

Tan, Y. et al., 2004. Entrainment dissociates transcription and translation of a circadian clock gene in neurospora. *Current biology : CB*, 14(5), pp.433–8.

Thain, S.C., Hall, A. & Millar, A.J., 2000. Functional independence of circadian clocks that regulate plant gene expression. *Current biology : CB*, 10(16), pp.951–6.

Tirosh, I. et al., 2009. A yeast hybrid provides insight into the evolution of gene expression regulation. *Science (New York, N.Y.)*, 324(5927), pp.659–62.

Tirosh, I. & Barkai, N., 2008. Two strategies for gene regulation by promoter nucleosomes. *Genome research*, 18(7), pp.1084–91.

Toh, K.L. et al., 2001. An hPer2 phosphorylation site mutation in familiar advanced sleep phase syndrome. *Science*, 291(5506), pp.1040–1043.

Toni, T. et al., 2009. Approximate Bayesian computation scheme for parameter inference and model selection in dynamical systems. *Journal of The Royal Society Interface*, 6(31), pp.187–202.

Triqueneaux, G. et al., 2004. The orphan receptor Rev-erbα gene is a target of the circadian clock pacemaker. *Journal of Molecular Endocrinology*, 33(3), pp.585–608.

Turek, F.W., 2005. Obesity and Metabolic Syndrome in Circadian Clock Mutant Mice. *Science*, 308(5724), pp.1043–1045.

Vallone, D. et al., 2004. E-box function in a period gene repressed by light. *Proc Natl Acad Sci U S A*, 101(12), pp.4106–4111.

Vatine, G. et al., 2011. It's time to swim! Zebrafish and the circadian clock. *FEBS letters*, 585(10), pp.1485–1494.

Vatine, G. et al., 2009. Light directs zebrafish period2 expression via conserved D and E boxes. *PLoS Biol*, 7(10), p.e1000223.

Vermeirssen, V. et al., 2007. Transcription factor modularity in a gene-centered C. elegans core neuronal protein-DNA interaction network. *Genome research*, 17(7), pp.1061–71.

Vigo-Aguiar, J. & Ramos, H., 2011. A numerical ODE solver that preserves the fixed points and their stability. *Journal of Computational and Applied Mathematics*, 235(7), pp.1856–1867.

Weger, B.D. et al., 2011. The Light Responsive Transcriptome of the Zebrafish: Function and Regulation. *PLoS ONE*, 6(2), p.e17080.

Weintraub, H., 1984. Histone-H1-dependent chromatin superstructures and the suppression of gene activity. *Cell*, 38(1), pp.17–27.

Weiss, J.N., 1997. The Hill equation revisited: uses and misuses. *The FASEB Journal*, 11(11), pp.835–841.

Welsh, D.K., Takahashi, J.S. & Kay, S.A., 2010. Suprachiasmatic nucleus: cell autonomy and network properties. *Annu Rev Physiol*, 72, pp.551–577.

Wernig, M. et al., 2007. In vitro reprogramming of fibroblasts into a pluripotent ES-cell-like state. *Nature*, 448(7151), pp.318–24.

White, R.M. et al., 2008. Transparent adult zebrafish as a tool for in vivo transplantation analysis. *Cell stem cell*, 2(2), pp.183–9.

Whitmore, D. et al., 1998. Zebrafish Clock rhythmic expression reveals independent peripheral circadian oscillators. *Nature neuroscience*, 1(8), pp.701–7.

Whitmore, D., Foulkes, N.S. & Sassone-Corsi, P., 2000. Light acts directly on organs and cells in culture to set the vertebrate circadian clock. *Nature*, 404(6773), pp.87–91.

Wu, H., Hu, Z. & Liu, X.Q., 1998. Protein trans-splicing by a split intein encoded in a split DnaE gene of Synechocystis sp. PCC6803. *Proceedings of the National Academy of Sciences of the United States of America*, 95(16), pp.9226–31.

Wu, Z. et al., 2007. On the trend, detrending, and variability of nonlinear and nonstationary time series. *Proceedings of the National Academy of Sciences of the United States of America*, 104(38), pp.14889–94.

Yamazaki, S., 2000. Resetting Central and Peripheral Circadian Oscillators in Transgenic Rats. *Science*, 288(5466), pp.682–685.

Young, M.W. & Kay, S.A., 2001. Time zones: a comparative genetics of circadian clocks. *Nature reviews. Genetics*, 2(9), pp.702–15.

Yu, H. & Gerstein, M., 2006. Genomic analysis of the hierarchical structure of regulatory networks. *Proceedings of the National Academy of Sciences of the United States of America*, 103(40), pp.14724–31.

Zhou, H.-X., Rivas, G. & Minton, A.P., 2008. Macromolecular crowding and confinement: biochemical, biophysical, and potential physiological consequences. *Annual review of biophysics*, 37, pp.375–97.

Appendix

Code for the Zebrafish Model in Mathematica

```
ClearAll[v1,v2,v3,v4,kdm,kdeg,kd,k1,k2,k3,k4,k6,k7,k8,k9,ka,kp,kcl1,kcl2,kcl3,kcl4,kr,ke,lightresponse,n,end];
Needs["PlotLegends`"]
(*--------------------------------parameters--------------------------------*)
v1=0.35;v2=0.45;
kdm=0.45;kdeg=0.45;kd=0.45;
k1=0.35;k2=0.7;k3=0.45;k4=0.25;
ka=0.25;kp=0.35;kcl1=0.45;kcl2=0.45;

v3=0.35;v4=0.3;k9=0.3;k8=0.3;k7=0.1;kcl3=0.5;k6=0.3;kcl4=0.5;kr=0.5;ke=0.5;
lightresponse=0.08;n=5;end=200;
(*equations*)
(*--------------------------------without light--------------------------------*)
sol=NDSolve[{
   c1am'[t]==v1(ClockBmal[t]/(ka+ClockBmal[t]))-kdm*(c1am[t]/(kdeg+c1am[t])),
   cry1a'[t]==k1*c1am[t]-kd*(cry1a[t]/(kdeg+cry1a[t])),

ClockBmal'[t]==k2(kcl1^n/(kcl1^n+cry1a[t]^n))+k3(kcl2^n/(kcl2^n+per1cry3[t]^n))+k7(kcl3^n/(kcl3^n+RevErb[t]^n))+k6(Ro
ra[t]^n/(kcl4^n+Rora[t]^n))-kd*(ClockBmal[t]/(kdeg+ClockBmal[t])),
   p1c3m'[t]==v2(ClockBmal[t]/(kp+ClockBmal[t]))-kdm*(p1c3m[t]/(kdeg+p1c3m[t])),
   per1cry3'[t]==k4*p1c3m[t]-kd*(per1cry3[t]/(kdeg+per1cry3[t])),
   reverbm'[t]==v3(ClockBmal[t]/(ke+ClockBmal[t]))-kdm*(reverbm[t]/(kdeg+reverbm[t])),
   RevErb'[t]==k9*reverbm[t]-kd*(RevErb[t]/(kdeg+RevErb[t])),
   roram'[t]==v4(ClockBmal[t]/(kr+ClockBmal[t]))-kdm*(roram[t]/(kdeg+roram[t])),
   Rora'[t]==k8*roram[t]-kd*(Rora[t]/(kdeg+Rora[t])),

c1am[0]==1,cry1a[0]==1.1,ClockBmal[0]==0.5,p1c3m[0]==1,per1cry3[0]==1.2,reverbm[0]==0.3,RevErb[0]==
0.3,roram[0]==0.2,Rora[0]==0.2},
  {c1am,cry1a,ClockBmal,p1c3m,per1cry3,reverbm,RevErb,roram,Rora},{t,0,end}];
(*graphs*)
Plot[{c1am[t]/.sol,cry1a[t]/.sol,ClockBmal[t]/.sol,p1c3m[t]/.sol,per1cry3[t]/.sol,reverbm[t]/.sol,RevErb[t]/.sol,ror
am[t]/.sol,Rora[t]/.sol},{t,0,end},PlotRange->{{0,end},{0,25}},PlotStyle-
>{{Red,Thick},{Purple,Thick},{Yellow,Thick},{Green,Thick},{Blue,Thick},{Cyan,Thick},{Brown,Thick},{M
agenta,Thick},{Gray,Thick}},AxesLabel->{"time", "conc" },
  PlotLabel->"Model 1 extension without light", PlotLegend->{"c1am",
"cry1a","ClockBmal","p1c3m","per1cry3","reverbm","RevErb","rora","Rora"},LegendPosition->{1.1,-
0.4},ImageSize->Medium];

Plot[{c1am[t]/.sol,cry1a[t]/.sol,ClockBmal[t]/.sol,p1c3m[t]/.sol,per1cry3[t]/.sol,reverbm[t]/.sol,RevErb[t]/.sol,ror
am[t]/.sol,Rora[t]/.sol},{t,0,end},PlotRange->{{0,end},{0,2.5}},PlotStyle-
>{{Red,Thick},{Purple,Thick},{Yellow,Thick},{Green,Thick},{Blue,Thick},{Cyan,Thick},{Brown,Thick},{M
agenta,Thick},{Gray,Thick}},AxesLabel->{"time", "conc" },
 PlotLabel->"Model 1 extension without light zoom", PlotLegend->{"c1am",
"cry1a","ClockBmal","p1c3m","per1cry3","reverbm","RevErb","rora","Rora"},LegendPosition->{1.1,-
0.4},ImageSize->Large]



(*--------------------------------with light--------------------------------*)
light[t_]:=Piecewise[{{lightresponse,Sin[t/4]>0}},0];
Plot[light[t],{t,-2,100},ImageSize->Tiny];
sol=NDSolve[{
   {
    {c1am'[t]==v1(ClockBmal[t]/(ka+ClockBmal[t]))-kdm*(c1am[t]/(kdeg+c1am[t]))+light[t],
     cry1a'[t]==k1*c1am[t]-kd*(cry1a[t]/(kdeg+cry1a[t])),

ClockBmal'[t]==k2(kcl1^n/(kcl1^n+cry1a[t]^n))+k3(kcl2^n/(kcl2^n+per1cry3[t]^n))+k7(kcl3^n/(kcl3^n+RevErb[t]^n))+k6(Ro
ra[t]^n/(kcl4^n+Rora[t]^n))-kd*(ClockBmal[t]/(kdeg+ClockBmal[t])),
     p1c3m'[t]==v2(ClockBmal[t]/(kp+ClockBmal[t]))-kdm*(p1c3m[t]/(kdeg+p1c3m[t])),
     per1cry3'[t]==k4*p1c3m[t]-kd*(per1cry3[t]/(kdeg+per1cry3[t])),
     reverbm'[t]==v3(ClockBmal[t]/(ke+ClockBmal[t]))-kdm*(reverbm[t]/(kdeg+reverbm[t])),
     RevErb'[t]==k9*reverbm[t]-kd*(RevErb[t]/(kdeg+RevErb[t])),
   roram'[t]==v4(ClockBmal[t]/(kr+ClockBmal[t]))-kdm*(roram[t]/(kdeg+roram[t])),
   Rora'[t]==k8*roram[t]-kd*(Rora[t]/(kdeg+Rora[t])),

c1am[0]==1,cry1a[0]==1.1,ClockBmal[0]==0.5,p1c3m[0]==1,per1cry3[0]==1.2,reverbm[0]==0.3,RevErb[0]==
0.3,roram[0]==0.2,Rora[0]==0.2},
   {□}
  }},
  {c1am,cry1a,ClockBmal,p1c3m,per1cry3,reverbm,RevErb,roram,Rora},{t,0,end}];
(*graphs*)
Plot[{c1am[t]/.sol,cry1a[t]/.sol,ClockBmal[t]/.sol,p1c3m[t]/.sol,per1cry3[t]/.sol,reverbm[t]/.sol,RevErb[t]/.sol,ror
am[t]/.sol,Rora[t]/.sol},{t,0,end},PlotRange->{{0,end},{0,25}},PlotStyle-
>{{Red,Thick},{Purple,Thick},{Yellow,Thick},{Green,Thick},{Blue,Thick},{Cyan,Thick},{Brown,Thick},{M
agenta,Thick},{Gray,Thick}},AxesLabel->{"time", "conc" },
 PlotLabel->"Model 1 extension with light (LD)", PlotLegend->{"c1am",
"cry1a","ClockBmal","p1c3m","per1cry3","reverbm","RevErb","rora","Rora"},LegendPosition->{1.1,-
0.4},ImageSize->Medium];

Plot[{c1am[t]/.sol,cry1a[t]/.sol,ClockBmal[t]/.sol,p1c3m[t]/.sol,per1cry3[t]/.sol,reverbm[t]/.sol,RevErb[t]/.sol,ror
am[t]/.sol,Rora[t]/.sol,light[t]},{t,0,end},PlotRange->{{0,end},{0,2.5}},PlotStyle-
>{{Red,Thick},{Purple,Thick},{Yellow,Thick},{Green,Thick},{Blue,Thick},{Cyan,Thick},{Brown,Thick},{M
agenta,Thick},{Gray,Thick},{Black,Thick}},AxesLabel->{"time", "conc" },
 PlotLabel->"Model 1 extension with light (LD) zoom", PlotLegend->{"c1am",
"cry1a","ClockBmal","p1c3m","per1cry3","reverbm","RevErb","rora","Rora","light"},LegendPosition->{1.1,-
0.4},ImageSize->Large]
```

Code for the Hilbert Transform

```
function [HilbertResults] = PHD_OOAA_HilbertAnalysis(Data, Time, Traces,
Plotless)

% PHD_OOAA_HilbertAnalysis(Data, Time, Traces)
% specify input data and seperate timeline, columns to analyse, and format
of output

% setting plotless to 1 surpresses graph output and reports abridged output
data

VarNames = {'c1am','cry1a','ClockBmal','p1c3m','per1cry3'};

TrMar = 0;

if Plotless == 1

    HilbertResults = zeros(length(Traces),4);

else

    HilbertResults = zeros(length(Traces),7);

end

for j = Traces

    TrMar = TrMar+1;       %marker for cycling through traces
    DataCol = Data(:,j);

    % Hilbert Transform and Instantaneous Phase
    % -------------------------------------------------------------

    HilData = hilbert(DataCol);
    ImagData = imag(HilData);       %imaginary component
    RealData = real(HilData);       %real component

    InstPhase = angle(HilData);     %instantaneous phase
    % Period by taking time between Maxima
    % ------------------------------------------------------------

    %finding the points just before and after the imaginary component
increases
    %beyond 0, then calculating the estimated time at maxima and resulting
periods

    PtsAfMax = [0; (ImagData(2:end)>0).*(ImagData(1:end-1)<0) ];
    PtsBfMax = [PtsAfMax(2:end); 0];    %logic array of points before and
after 0'up'

    TmAfMax = PtsAfMax.*Time;              %corresponding time at points
    TmBfMax = PtsBfMax.*Time;

    ImagAfMax = PtsAfMax.*ImagData;     %imaginary value at points
    ImagBfMax = PtsBfMax.*ImagData;

    % calculate time at the maxima, weighing time at points before and
after using
    % relative imaginary values, then taking time between maxima for period

    TimesMaxRaw = (TmAfMax(2:end).*ImagAfMax(2:end)-TmBfMax(1:end-
1).*ImagBfMax(1:end-1))./(ImagAfMax(2:end)-ImagBfMax(1:end-
1)+(0==(ImagAfMax(2:end)-ImagBfMax(1:end-1))));
    TMax = TimesMaxRaw(TimesMaxRaw ~= 0);      %only keeping non-zero
values
    TMax = TMax(2:(end-1));           %discarding the first and last value

    PeriodList = [];

    for k = 1:(length(TMax)-1)             %taking time difference between
maxima

        PeriodList(k) = TMax(k+1)-TMax(k);
        k=k+1;

    end
    clear k;

    if (isempty(PeriodList) == 1)          %prevents empty array from
causing reference errors

        PeriodList = 0;

    end

    PeriodMean = mean(PeriodList);
    PeriodStd = std(PeriodList);

    % Trough to Peak Difference (Amplitude)
    % ------------------------------------------------------------

    %calculating trough to peak difference based on real values at maxima
and minima

    RealAfMax = PtsAfMax.*RealData;       %real values before and after
maxima
    RealBfMax = PtsBfMax.*RealData;

    % calculate real value at maxima, weighing points before and after by
time

    RealMax = (((TmAfMax(2:end)-TimesMaxRaw).*RealBfMax(1:end-
1))+((TimesMaxRaw-TmBfMax(1:end-1)).*RealAfMax(2:end)))./(TmAfMax(2:end)-
TmBfMax(1:end-1)+(0==(TmAfMax(2:end)-TmBfMax(1:end-1))));
    RealMax = RealMax(RealMax ~= 0);     %only keeping non-zero values
    RealMax = RealMax(2:(end-1));      %discarding the first and last value
    %finding the points just before and after the imaginary component
decreases
    %beneath 0, then calculating the estimated time at minima

    PtsAfMin = [0; (ImagData(2:end)<0).*(ImagData(1:end-1)>0) ];
    PtsBfMin = [PtsAfMin(2:end); 0];    %logic array of points before and
after 0'down'

    TmAfMin = PtsAfMin.*Time;              %corresponding time at points
    TmBfMin = PtsBfMin.*Time;

    ImagAfMin = PtsAfMin.*ImagData;     %imaginary value at points
    ImagBfMin = PtsBfMin.*ImagData;

    % calculate time at the minima, weighing time at points before and
after using
    % relative imaginary values

    TimesMinRaw = (TmAfMin(2:end).*ImagAfMin(2:end)-TmBfMin(1:end-
1).*ImagBfMin(1:end-1))./(ImagAfMin(2:end)-ImagBfMin(1:end-
1)+(0==(ImagAfMin(2:end)-ImagBfMin(1:end-1))));
    TMin = TimesMinRaw(TimesMinRaw ~= 0);      %only keeping non-zero
values
    TMin = TMin(2:(end-1));           %discarding the first and last value

    RealAfMin = PtsAfMin.*RealData;     %real values before and after
minima
    RealBfMin = PtsBfMin.*RealData;

    % calculate real value at minima, weighing points before and after by
time

    RealMin = (((TmAfMin(2:end)-TimesMinRaw).*RealBfMin(1:end-
1))+((TimesMinRaw-TmBfMin(1:end-1)).*RealAfMin(2:end)))./(TmAfMin(2:end)-
TmBfMin(1:end-1)+(0==(TmAfMin(2:end)-TmBfMin(1:end-1))));
    RealMin = RealMin(RealMin ~= 0);   %only keeping non-zero values
    RealMin = RealMin(2:(end-1));      %discarding the first and last value

    % check number of cycles TtP, then their value

    FullTtP = min(length(RealMax), length(RealMin));
    PtTList = [];

    for l = 1:FullTtP       % PtT could also be used, depending on section
of data

        PtTList(l) = (RealMax(l)-RealMin(l));

    end                     %difference between successive trougs and peaks
    clear l;

    if (isempty(PtTList) == 1)      %prevents empty array from causing
reference errors

        PtTList = 0;

    end

    AmplitudeMean = mean(PtTList);
    AmplitudeStd = std(PtTList);

    % Relative Phases for Last Period
    % ------------------------------------------------------------

    if (isempty(TMax) == 1) || (isempty(TMin) == 1)
                                    %prevents empty array from causing
reference errors
```

```
        TMax = 0;
        TMin = 0;

    end

    if Plotless == 1

        HilbertResults(j,:) = [PeriodMean, AmplitudeMean, TMax(end),
TMin(end)];

    else

        LastPeak = mod(TMax(end),PeriodMean);
        LastRelPeak = mod(TMax(end),PeriodMean)*100/PeriodMean;
        LastRelTrough = mod(TMin(end),PeriodMean)*100/PeriodMean;

        if TrMar == 1

            RefPhase = LastPeak;

        end

        VarShift = LastPeak-RefPhase;

        if VarShift > PeriodMean/2                   %adjust relative to
period

            VarShift = VarShift-PeriodMean;

        elseif VarShift < -PeriodMean/2

            VarShift = VarShift+PeriodMean;

        end

        RelVarShift = VarShift*100/PeriodMean;

        HilbertResults(TrMar,:) = [PeriodMean, PeriodStd, AmplitudeMean,
AmplitudeStd, LastRelTrough, LastRelPeak, RelVarShift];

        % Figures
        % -------------------------------------------------------------

        figure('Name',['Hilbert Transform Summary for ', VarNames{j}])

        subplot(2,3,1)
        plot(HilData, 'g')
        hold on
        plot(RealData.*PtsAfMax, ImagData.*PtsAfMax, '.k','markersize',15)
        plot(RealData.*PtsAfMin, ImagData.*PtsAfMin, '.r','markersize',15)
        hold off
        title('Hilbert Transform: Maxima, Minima','fontsize',14)
         xlabel('Real Component')
         ylabel('Imaginary Component')
         legend('Complex Transform', 'Maxima', 'Minima')

        subplot(2,3,2)
        plot(Time, InstPhase,'r','LineWidth',2.5)
        xlim([(TMax(end-1)) TMax(end)])
        title('Instantaneous Phase Last Period','fontsize',14)
        set(gca,'XTick',[linspace((TMax(end-1)), TMax(end),5)])

set(gca,'XTickLabel',[0;round(PeriodList(end))/4;round(PeriodList(end))/2;r
ound(PeriodList(end))*3/4;round(PeriodList(end))])
         xlabel('Time')
         ylabel('Radians')
         legend('Phase Angle')
        subplot(2,3,3)
        plot(Time,[RealData, ImagData, PtsAfMax, -PtsAfMin])
        title('Real and Imaginary Components','fontsize',14)
         xlabel('Time')
         ylabel('Real or Imaginary Values')
         legend('Real Component', 'Imaginary Component', 'At Maxima', 'At 
Minima')

        subplot(2,3,4)
        plot(TMax,PtTList,'Marker','*','color','m','LineWidth',2)
        title('Amplitude over Time','fontsize',14)
         xlabel('Time')
         ylabel('Amplitude of Oscillation')
         legend('Trough to Peak at Maxima')

        subplot(2,3,5)
        bins = [0:2:40];
        hist(PeriodList,bins)
        xlim([0 40])
        title('Histogram of Period Distribution','fontsize',14)
         xlabel('Period Length in Hours')
         ylabel('Number observed')
         legend('Periods')

        subplot(2,3,6)
        plot(TMax(2:end),PeriodList,'LineWidth',2)
        title('Periods over Time','fontsize',14)
         xlabel('Time')
         ylabel('Period Lenght in Hours')
         legend('Periods at Maxima')

    end

end
clear j

if Plotless ~= 1

    disp('   Period     +/-       Amplitude +/-     Min%Phase Max%Phase 
Max%Shift')
    disp(HilbertResults)

end

end
```

Code for the Wavelet Transform

```
function [RidgeData] = PHD_OOAA_CWTStarter(Data,SampleRate,Traces,varargin)

%PHD_OOAA_CWTStarter(Data,SampleRate,Traces,[options])
% specify input data, the sampling rate (e.g. 1), and the specific columns 
to analyze
%
%Optional arguments:
%       ScalesPerOctave - scales within each doubling of period (defaults 
to 32)
%       Eta - CWT tuning parameter (defaults to 2*pi)
%       MinPeriod - The minimum period to analyze (defaults to 
2/SampleRate)
%       MaxPeriod - The maximum period to analyze (defaults to smaller of 
N/(2*sqrt(2)*SampleRate) or 48)
%           where N is the common length of the data
%       ZeroPadding - zeros prepended/appended to reduce edge effects 
(defaults to N)
```

```
%       Method - selects method by which the CWT ridge is located (defaults
to 'Proportional')
%       Cutoff - proportion of maximum amplitude a local maximum must have
to be considered
%           a ridge under the 'Proportional' method (defaults to 1)
%       ForceSingleRidge - when set to 1, permits only a single ridge per
timepoint. If multiple
%           ridges are present, the highest-magnitude one is selected
(defaults to 0)
%       ExcludeEdges - removes any points potentially subject to edge
effects (defaults to 0)


%Required and Optional Inputs
% ------------------------------------------------------------

p = inputParser;

p.addRequired('Data', @isnumeric);
p.addRequired('SampleRate', @(x)isnumeric(x)&(x>0));
p.addRequired('Traces', @isnumeric);
p.addOptional('ScalesPerOctave', 32, @(x)isnumeric(x)&(x>0));
p.addOptional('Eta', 2*pi, @(x)isnumeric(x)&(x>0));
p.addOptional('MinPeriod', -1, @(x)isnumeric(x)&(x>0));
p.addOptional('MaxPeriod', -1, @(x)isnumeric(x)&(x>0));
p.addOptional('ZeroPadding', -1, @(x)isnumeric(x)&(x>=0));
p.addOptional('Method', 'Proportional', @(x)max(strcmpi(x,
{'Proportional','Crazy_climber'})) );
p.addOptional('Cutoff', 1, @(x)(isnumeric(x) & (x>0) & (x<=1)));
p.addOptional('ForceSingleRidge', 0, @(x)(x==0)||(x==1));
p.addOptional('ExcludeEdges', 0, @(x)(x==0)||(x==1));

p.parse(Data, SampleRate, Traces, varargin{:});
WavTran = p.Results;
WavTran.SampleRate = (1/SampleRate);
WavTran.Data = WavTran.Data(:,Traces);

WT1 = WavTran.ScalesPerOctave;
WT2 = WavTran.Eta;
WT3 = WavTran.MinPeriod;
WT4 = WavTran.MaxPeriod;
WT5 = WavTran.ZeroPadding;

RF1 = WavTran.Method;
RF2 = WavTran.Cutoff;
RF3 = WavTran.ForceSingleRidge;
RF4 = WavTran.ExcludeEdges;

% ------------------------------------------------------------
%I. PHD_CWTransform(Data, SampleRate, [options])
%   specify columns of data and the sampling rate (e.g. 1)
% ------------------------------------------------------------
%   Optional arguments:       (N = common length of data)
%   'ScalesPerOctave' - number >0, defaults to 32
%   'Eta'(CWT tuning parameter) - number >0, defaults to 2*pi
%   'MinPeriod' - number >= 2/SampleRate, defaults to 2/SampleRate
%   'MaxPeriod' - number >0 and <= N/(2*sqrt(2)*SampleRate), defaults to
smaller of N/(2*sqrt(2)*SampleRate) or 48
%   'ZeroPadding' - number >=0, defaults to N

TransformedData =
PHD_OOAA_CWTransform(WavTran.Data,WavTran.SampleRate,'ScalesPerOctave',WT1,
'Eta',WT2,'MinPeriod',WT3,'MaxPeriod',WT4,'ZeroPadding',WT5);


% ------------------------------------------------------------
%II. PHD_CWTRidges(CWT_obj,[options])
%    enter a CWTransform to find ridge points using the selected method
% -------------------------------------------------------------
%   Optional arguments:
%   'Method' - 'Proportional' or 'Crazy_climber', defaults to
'Proportional'
%   'Cutoff' - number >0 and <=1, defaults to 1
%   'ForceSingleRidge' - 0 or 1, defaults to 0
%   'ExcludeEdges' - 0 or 1, defaults to 0

RidgeData =
PHD_OOAA_CWTRidges(TransformedData,'Method',RF1,'Cutoff',RF2,'ForceSingleRi
dge',RF3,'ExcludeEdges',RF4);


% -------------------------------------------------------------
%III. PHD_CWTPlots(Data, [options])
%     forward transformed or ridge data to construct a heatmap or other
plot
% -------------------------------------------------------------
%   Optional arguments:
%   'Dataset' - number >0, defaults to 1
%   'PlotType' -
'CWT','Phase','Ridge','CWT+Ridge','Data+Period','Data+Peak/Trough','Data+Ph
ase', defaults to CWT
%   'ExcludeEdges' - 0 or 1, defaults to 0 (or value in input object)

VarNames = {'c1am','cry1a','ClockBmal','p1c3m','per1cry3'};
DataCol = 0;

for Tr = Traces

    DataCol = DataCol+1;    %counter to cycle through different traces

    figure('Name',['Wavelet Transform Summary for ', VarNames{Tr}])
    %generate subplot combining different graphs
    subplot(2,2,1)                                  %only exclude edges on
heatmaps (subplots 1 and 2)

PHD_OOAA_CWTPlots(RidgeData,'PlotType','CWT+Ridge','ExcludeEdges',0,'Datase
t',DataCol);
    title('Ridges on Heatmap of Wavelets','fontsize',14)

    subplot(2,2,2)

PHD_OOAA_CWTPlots(RidgeData,'PlotType','Phase','ExcludeEdges',0,'Dataset',D
ataCol);
    title('Heatmap of Wavelet Phases','fontsize',14)

    subplot(2,2,3)

PHD_OOAA_CWTPlots(RidgeData,'PlotType','Data+Peak/Trough','Dataset',DataCol
);
    title('Peaks and Troughs relative to Data','fontsize',14)

    subplot(2,2,4)
    PHD_OOAA_CWTPlots(RidgeData,'PlotType','Data+Phase','Dataset',DataCol);
    title('Phases relative to Data','fontsize',14)

end
clear Tr

end



function [CWT_obj] = PHD_OOAA_CWTransform(Data, SampleRate, varargin)

%PHD_OOAA_CWTransform(Data, SampleRate, [options])
%specify columns of data and the sampling rate (e.g. 1)
```

```
%
%Optional arguments:
%       ScalesPerOctave - scales within each doubling of period (defaults
to 32)
%       Eta - CWT tuning parameter (defaults to 2*pi)
%       MinPeriod - The minimum period to analyze (defaults to
2/SampleRate)
%       MaxPeriod - The maximum period to analyze (defaults to smaller of
N/(2*sqrt(2)*SampleRate) or 48)
%           where N is the common length of the data
%       ZeroPadding - zeros prepended/appended to reduce edge effects
(defaults to N)


%Required and Optional Inputs
% ------------------------------------------------------------

p = inputParser;

p.addRequired('Data', @isnumeric);
p.addRequired('SampleRate', @(x)isnumeric(x)&(x>0));
p.addOptional('ScalesPerOctave', 32, @(x)isnumeric(x)&(x>0));
p.addOptional('Eta', 2*pi, @(x)isnumeric(x)&(x>0));
p.addOptional('MinPeriod', -1, @isnumeric);
p.addOptional('MaxPeriod', -1, @isnumeric);
p.addOptional('ZeroPadding', -1, @isnumeric);

p.parse(Data, SampleRate, varargin{:});
CWT_obj = p.Results;

% Housekeeping
% ------------------------------------------------------------

CWT_obj.Header = 'CWTransform Results';

CWT_obj.RescalingFactor = (2*pi)/(CWT_obj.Eta);     %defining rescaling
factor

[NumDataPts, Traces] = size(CWT_obj.Data);  %obtaining number of traces and
data points

% Zeropadding
% ------------------------------------------------------------

if (CWT_obj.ZeroPadding == -1)

    CWT_obj.ZeroPadding = NumDataPts;    % setting zeropadding default
length

end

if (CWT_obj.ZeroPadding > 0)             %Zero-pad the data

    CWT_obj.Data = [zeros(CWT_obj.ZeroPadding, Traces); CWT_obj.Data;
zeros(CWT_obj.ZeroPadding, Traces)];

end

N = NumDataPts + 2*CWT_obj.ZeroPadding;  %take zeropadding into account for
data length

% Maximum and Minimum Periods
% ------------------------------------------------------------

MinPdForRate = 2/CWT_obj.SampleRate;        %default minimum period
MaxPdForRate = NumDataPts/(2*sqrt(2)*CWT_obj.SampleRate); %default maximum
period
```

```
if (CWT_obj.MinPeriod == -1) || (CWT_obj.MinPeriod < MinPdForRate)

    CWT_obj.MinPeriod = MinPdForRate;       %apply default also where
entered value implausible

end

if (CWT_obj.MaxPeriod == -1) || (CWT_obj.MaxPeriod > MaxPdForRate)

    CWT_obj.MaxPeriod = min(MaxPdForRate, 48);       %apply default (or 48)
also where entered value implausible

end

% Starting and Stopping Scales
% -------------------------------------------------------------

MinScale = CWT_obj.MinPeriod * CWT_obj.SampleRate /
CWT_obj.RescalingFactor;    %convert periods to scales
MaxScale = CWT_obj.MaxPeriod * CWT_obj.SampleRate /
CWT_obj.RescalingFactor;

CWT_start = floor(log2(MinScale)*CWT_obj.ScalesPerOctave);   %log base 2 of
our starting and stopping scales
CWT_stop = floor(log2(MaxScale)*CWT_obj.ScalesPerOctave);

CWT_obj.ScaleList = 2.^( (CWT_start:1:CWT_stop) ./
CWT_obj.ScalesPerOctave);     %list of scales and corresponding periods
CWT_obj.Periods =
CWT_obj.ScaleList*CWT_obj.RescalingFactor/CWT_obj.SampleRate;

% Fourier Transform of Morlet Wavelet
% -------------------------------------------------------------

morlwaveft = @(s,m,w) sqrt(s/(2*pi)).* pi^(-1/4) .* exp(-0.5.*(s.*w -
m).^2)- sqrt(s/(2*pi)).* pi^(-1/4) .* exp(-0.5.*((s.*w).^2+m.^2));

wk = ((2*pi)/(N)).*[(0:1:floor(N/2)), -((ceil(N/2)-1):-1:1)];    %adjusted
index for FFT frequencies

% Continuous Wavelet Transform
% -------------------------------------------------------------

CWT_obj.Data = CWT_obj.Data';

for lc = 1:Traces

    fx = fft(CWT_obj.Data(lc,:));         % Take the FFT of the current
trace

    CWT = zeros(CWT_stop-CWT_start+1, N);

    %get the FT for current scale, multiply element-by-element by the FFT
of data, and take the IFFT

    Current_ScaleID=1;

    for j = CWT_start:1:CWT_stop

        aj = 2^(j/CWT_obj.ScalesPerOctave);          %current scale
        CWT(Current_ScaleID, :) = ifft(fx.*morlwaveft(aj, CWT_obj.Eta,
wk));
        Current_ScaleID = Current_ScaleID+1;

    end
    clear j
    if CWT_obj.ZeroPadding > 0          %remove zero padding if any was
applied

        CWT = CWT(:, (CWT_obj.ZeroPadding+1):(CWT_obj.ZeroPadding +
NumDataPts));

    end

    ScaleNorm = repmat(CWT_obj.ScaleList', 1, NumDataPts);  %norm to
compare magnitudes across scales
    cwtNormed = (pi^0.25).*CWT./sqrt(ScaleNorm);

    %rescale amplitude terms associated with the CWT to peak-to-trough
measure

    amps = 2*(CWT.*conj(CWT)).*(pi /
((4*pi)^.25)).*((2*CWT_obj.Eta)./ScaleNorm);

    %store results

    CWT_obj.CWT{lc} = CWT;
    CWT_obj.CWTnormed{lc} = cwtNormed;
    CWT_obj.phases{lc} = atan2(imag(CWT_obj.CWT{lc}),
real(CWT_obj.CWT{lc}));
    CWT_obj.amplitudes{lc} = amps;

end
clear lc

CWT_obj.Data = CWT_obj.Data';
CWT_obj.Data =
CWT_obj.Data((CWT_obj.ZeroPadding+1):(CWT_obj.ZeroPadding+NumDataPts),:);
%remove zeropadding from original data
CWT_obj.Times = (1:NumDataPts)./CWT_obj.SampleRate;

end




function [ridge_object] = PHD_OOAA_CWTRidges(CWT_obj, varargin)

%PHD_OOAA_CWTRidges(CWT_obj,[options])
%enter a CWTransform to find ridge points using the selected method
%
%Optional arguments:
%       Method - selects method by which the CWT ridge is located (defaults
to 'Proportional')
%       Cutoff - proportion of maximum amplitude a local maximum must have
to be considered
%           a ridge under the 'Proportional' method (defaults to 1)
%       ForceSingleRidge - when set to 1, permits only a single ridge per
timepoint. If multiple
%           ridges are present, the highest-magnitude one is selected
(defaults to 0)
%       ExcludeEdges - removes any points potentially subject to edge
effects (defaults to 0)


%Required and Optional Inputs
% ------------------------------------------------------------

p = inputParser;

p.addRequired('CWT_obj', @(x)strcmpi(x.Header, 'CWTransform Results'));
p.addOptional('Method', 'Proportional', @(x)max(strcmpi(x,
{'Proportional','Crazy_climber'})) );
p.addOptional('Cutoff', 1, @(x)(isnumeric(x) & (x>0) & (x<=1)));
```

```
p.addOptional('ForceSingleRidge', 0, @(x)(x==0)||(x==1));
p.addOptional('ExcludeEdges', 0, @(x)(x==0)||(x==1));

p.parse(CWT_obj, varargin{:});
opts = p.Results;

%Housekeeping
% ------------------------------------------------------------

NumberOfTraces = length(CWT_obj.CWT);

CutOff = opts.Cutoff;                  %use desired cutoff

CWT_obj.Method = opts.Method;
CWT_obj.Cutoff = opts.Cutoff;
CWT_obj.ForceSingleRidge = opts.ForceSingleRidge;
CWT_obj.ExcludeEdges = opts.ExcludeEdges;

%Excluding Edges
% ------------------------------------------------------------

if opts.ExcludeEdges         %remove edge-influenced points

    Drops = floor(sqrt(2) .* CWT_obj.Periods);
    [r, c] = size(CWT_obj.CWTnormed{1});    %determine size of data/for
mask
    EdgeMask = zeros(r,c);

    for mc = 1:r        %creats mask of NaNs and zeros relative to periods

        EdgeMask(mc,:) = [NaN(1, Drops(mc)), zeros(1, c - 2*Drops(mc)),
NaN(1, Drops(mc))];

    end
    clear mc

end

%CWT Ridge Finder
% ------------------------------------------------------------

for lc = 1:NumberOfTraces

    CWTABS = abs(CWT_obj.CWTnormed{lc});       %absolute value of complex
data
    [rows, cols] = size(CWTABS);
    RidgeTable = zeros(rows, cols);

    %Crazy Climber
    % ------------------------------------------------------------

    %performs a simulated-annealing ridge location to generate mean
residence
    %times of all particles at each scale/translation pair

    if strcmpi(opts.Method, 'Crazy_climber')

        Nparticles = cols*4;           %number of particles for crazy
climber
        its_per_stage = rows*4;
                                       %cooling schedule
        CoolSched = [ones(1, its_per_stage)*1, ones(1, its_per_stage)*.1,
ones(1, its_per_stage)*.01, ones(1, its_per_stage)*.001];

        CrClW = abs(CWTABS);
        [CrClRow, CrClCol] = size(CrClW);
        MC = zeros(CrClRow,CrClCol);                    %empty storage
array
```

```
        P.sc = floor(rand(1, Nparticles)*CrClRow)+1;    %random particles
between 0 and 1 times number of rows etc.
        P.tr = floor(rand(1, Nparticles)*CrClCol)+1;

        Niter = length(CoolSched);                %number of iterations

        for crc = 1:Niter

            dx = randi([0,1],1, Nparticles)*2-1;  %random list of 1 and -1
            dy = randi([0,1],1, Nparticles)*2-1;

            dy(P.sc == 1) = 1;          %sets dy to 1 at all positions
where P.Sc equals 1
            dy(P.sc == CrClRow) = -1;   %sets dy to -1 at all positions
where P.Sc equals no of rows
            proposedA = P.sc+dy;

            P.tr = mod(P.tr + dx, CrClCol) + 1;

            Wstart = CrClW( ( (P.tr-1).*CrClRow) + (P.sc));     %select
values from CrClW/CWTABS
            Wend = CrClW( ( (P.tr-1).*CrClRow) + (proposedA));
            DM = Wend-Wstart;

            switchers = (rand(1,
Nparticles)<exp(DM/CoolSched(crc)))|(DM>0);
            stayers = 1-switchers;

            P.sc = sum([switchers; stayers].*[proposedA; P.sc]);

            MC( ((P.tr-1).*CrClRow) + (P.sc) ) = MC( ((P.tr-1).*CrClRow) +
(P.sc)) + CrClW( ((P.tr-1).*CrClRow) + (P.sc));

        end
        clear crc

        CWTABS = MC./Niter;

        if opts.ForceSingleRidge == 0

            RidgeTable = 1-CWTABS;

        end

    end

    %Detecting Maxima
    % ------------------------------------------------------------

    if strcmpi(opts.Method, 'Proportional') || (strcmpi(opts.Method,
'Crazy_climber') && opts.ForceSingleRidge == 1)

        for k = 1:cols          %determine if local maximum is above cutoff
-> it is a ridge

            xx = abs(CWTABS(:,k))';
            SignChgIdx  = [0, diff(sign(diff(xx))), 0]; %check the
difference between neighbouring values
                                                        %then check where
the sign of the difference changes
            col_CutOff = max(xx.*(SignChgIdx<0)).*CutOff;      %calculate
cutoff for column using greatest maximum

            RidgeTable(:,k) = (SignChgIdx<0) & (xx >col_CutOff); %find
maxima above cutoff

            if (CutOff==1)
```

```
                RidgeTable(find(xx == max(xx), 1, 'Last'),k ) = 1;
                                    %select only the last iteration of the
maximum value
            end

        end
        clear k

    end

    %Properties of Ridges
    % ------------------------------------------------------------

    if opts.ExcludeEdges

        RidgeTable = RidgeTable+EdgeMask;
                %add mask with NaNs to RidgeTable to block edge-influenced
points
    end

    TempIndex = repmat(1:cols, rows, 1);    %generate a table of indices to
the data

    WaveLengthIndex = repmat(CWT_obj.Periods', 1, cols);
    WlTable = (RidgeTable>0).*WaveLengthIndex;          %gives period
wherever there is a ridge

    CWT_obj.RidgeTable{lc} = RidgeTable;
    CWT_obj.RidgePeriods{lc} = CWT_obj.Periods * (RidgeTable>0);    %gives
periods at ridges
    CWT_obj.RidgePeriods{lc}(CWT_obj.RidgePeriods{lc}==0) = NaN;    %add
back NaN for zero periods

    CWT_obj.RidgePhases{lc} = NaN(1,cols);      %phases at the ridge
points, and NaN elsewhere
    CWT_obj.RidgePhases{lc}(TempIndex(RidgeTable==1)') =
CWT_obj.phases{lc}(RidgeTable==1)';

    CWT_obj.RidgeAmplitudes{lc} = NaN(1,cols);  %amplitudes at the ridge
points, and NaN elsewhere
    CWT_obj.RidgeAmplitudes{lc}(TempIndex(RidgeTable==1)') =
CWT_obj.amplitudes{lc}(RidgeTable==1)';

    %Zero-crossings (for a List of Phases)
    % ------------------------------------------------------------

    %all pairwise points where either the later point is >0 and the earlier
point is <0 or vice versa;
    %the sign of the later point indicates whether it's an "upcrossing" or
a "downcrossing"

    Timelist = CWT_obj.Times;
    px = CWT_obj.RidgePhases{lc};
    PhaCrossings = [(((px(2:end)>0) & (px(1:(end-1))<0)) | ((px(2:end)<0) &
(px(1:(end-1))>0)) ) .* sign(px(2:end)), 0];
    CWT_obj.PhaseCrossingIndex{lc} = PhaCrossings;            % This holds
a +1 for upcrossing, -1 for downcrossing.

    if ~isempty(PhaCrossings)

        CWT_obj.ZPtimes{lc} = Timelist(PhaCrossings==1);
        CWT_obj.MPtimes{lc} = Timelist(PhaCrossings==-1);
        CWT_obj.COtimes{lc} = Timelist((PhaCrossings~=0) &
~isnan(PhaCrossings));
        CWT_obj.PhaCrossings{lc} = PhaCrossings;

    else
```

```
        CWT_obj.ZPtimes{lc} = [];
        CWT_obj.MPtimes{lc} = [];
        CWT_obj.COtimes{lc} = [];

    end

end
clear lc

CWT_obj.PeriodTable = WlTable;

CWT_obj.SynchIndex = abs(sum(exp(1i*cell2mat(CWT_obj.RidgePhases')),
1))./NumberOfTraces;

ridge_object = CWT_obj;              %defining output of function

end




function [h] = PHD_OOAA_CWTPlots(Data, varargin)

%PHD_OOAA_CWTPlots(Data, [options])
%forward e.g. a ridge object to construct a heatmap or other plot
%
%Optional arguments:
%       Dataset - indicating which dataset should be plotted (defaults to
1)
%       PlotType - either 'CWT', 'Phase', 'Ridge', 'CWT+Ridge',
'Data+Period',
%           'Data+Peak/Trough',or 'Data+Phase' (defaults to CWT)
%       ExcludeEdges - either 1 or 0 (defaults to value of input object)


%Required and Optional Inputs
% ------------------------------------------------------------

p = inputParser;
p.addRequired('Data', @(x)strcmpi(x.Header, 'CWTransform Results'));
p.addOptional('Dataset', 1, @(x)isnumeric(x)&(x>0));
p.addOptional('PlotType', 'CWT', @(x)max(strcmpi(x,
{'CWT','Phase','Ridge','CWT+Ridge','Data+Period','Data+Peak/Trough','Data+P
hase'})));
p.addOptional('ExcludeEdges', 0, @(x)(x==0)||(x==1));

p.parse(Data, varargin{:});
opts = p.Results;

%Housekeeping
% ------------------------------------------------------------

[rows, cols] = size(Data.CWTnormed{1});
Mask = zeros(rows,cols);                  %updated for edge exclusion or
left empty
Hotmap = hot;                             %colour option for heatmap

%Excluding Edges
% ------------------------------------------------------------

if (Data.ExcludeEdges == 1) || (opts.ExcludeEdges ==1)

    Drops = floor(sqrt(2) .* Data.Periods);

    for mc = 1:rows
```

```
        Mask(mc,:) = [NaN(1, Drops(mc)), zeros(1, cols - 2*Drops(mc)),
NaN(1, Drops(mc))];

    end
    clear mc

end

%Selecting Data by PlotType
% ---------------------------------------------------------------

if strcmpi(opts.PlotType,'Ridge') || strcmpi(opts.PlotType,'CWT') ||
strcmpi(opts.PlotType,'Phase') || strcmpi(opts.PlotType,'CWT+Ridge')

    if strcmpi(opts.PlotType, 'Ridge')          %set to plot graph of
periods at ridges

        ColourmapTable = Data.RidgeTable{opts.Dataset}.*-1 +1;   %invert
zeros and ones

    elseif strcmpi(opts.PlotType, 'CWT')        %set to plot heatmap of
wavelets by periods

        ColourmapTable = Data.CWTnormed{opts.Dataset};

    elseif strcmpi(opts.PlotType, 'Phase')      %set to plot heatmap of
phases by periods

        ColourmapTable = Data.phases{opts.Dataset};

    elseif strcmpi(opts.PlotType, 'CWT+Ridge')  %set to combine heatmap of
wavelets and graph of ridges

        RidgePart = Data.RidgeTable{opts.Dataset}.*1.1;   %ridge as series
of points of value 1.1

        RidgeMask = Data.RidgeTable{opts.Dataset}.*-1 +1;   %divide CWT
table by single max absolute value to normalize
        CWTPart =
(Data.CWTnormed{opts.Dataset})/max(max(abs(Data.CWTnormed{opts.Dataset})));

        ColourmapTable = CWTPart .* RidgeMask + RidgePart;  %combine ridge
and CWT parts
        Hotmap = [Hotmap(1:62,:); [0 1 1]];                 %colour of
ridge plot

    end

    %Plotting HeatMaps
    % ---------------------------------------------------------------

    h = imagesc(flipud(abs(ColourmapTable) + Mask));    %plotting data
    colormap(Hotmap);                                   %setting heatmap
colour

    xlabel('Time')
    ylabel('Period')
    [TabRow, TabCol] = size(ColourmapTable);
    g = gca;

    yindex = floor(linspace(1, TabRow-1, 7));           %y-axis ticks and
numbering
    set(g, 'YTick', yindex);
    ylabs = floor(Data.Periods(yindex).*10^2)./(10^2);
    ylabs = fliplr(ylabs);
    set(g, 'YTickLabel', ylabs);
    xindex = floor(linspace(1, TabCol-1, 7));           %x-axis ticks and
numbering
    set(g, 'Xtick', xindex)
    Timelist = (1:TabCol)./Data.SampleRate;
    xlabs = floor(Timelist(xindex).*10^2)./(10^2);
    set(g, 'Xticklabel', xlabs)

else

    PlotData = Data.Data(:,opts.Dataset)';

    if strcmpi(opts.PlotType, 'Data+Period')

        Boost = 20;        %oscillations boosted for visualization
        plot(Data.Times, PlotData.*Boost, Data.Times,
Data.RidgePeriods{opts.Dataset});
        xlabel('Time')
        xlim([Data.Times(1),Data.Times(end)]);
        ylabel('Hours(/arbitrary for Data)')
        legend('Oscillation Data','Period at Ridges')

    elseif strcmpi(opts.PlotType, 'Data+Peak/Trough')
                           %oscillations centred on zero
        plot(Data.Times, PlotData-mean(PlotData), Data.Times,
Data.PhaCrossings{opts.Dataset}*(max(PlotData)/2));
        xlabel('Time')
        xlim([Data.Times(1),Data.Times(end)]);
        ylabel('Arbitrary')
        legend('Oscillation Data','Peaks and Troughs')

    elseif strcmpi(opts.PlotType, 'Data+Phase')
                            %oscillations centred on zero
        plot(Data.Times, PlotData-mean(PlotData), Data.Times,
Data.RidgePhases{opts.Dataset}*(max(PlotData)/6));
        xlabel('Time')
        xlim([Data.Times(1),Data.Times(end)]);
        ylabel('Arbitrary')
        legend('Oscillation Data','Phase at Ridges')

    end

end
```

Code for the Integrated Simulation Environment

```
function [ClockResults] = PHD_OOAA_ClockModel(EndTime,TimeStep,varargin)

%PHD_OOAA_ClockModel(EndTime,TimeStep,[options])
% solve the system of differential equation in local function at the bottom
in the time interval
% and resolution specified, using the parameters and visualization options
below)
%
%Optional arguments Initial Values and Plotting:
%   'IVCalibration', defaults to 0, 0 = none, 1 = end values of calibration
run in DD,
%       2 = sorted for minimum of SynchVariable in DD, 3 = end values of
run with LightID,
%       4 = sorted for minimum of SynchVariable
%   'SynchVariable', defaults to 1, used to sort for minimum above
%   'CalibrationTime', defaults to 120, lenght of calibration run
%   'PlotTimeRange', defaults to 1, outputs a graph of the entire time
range
```

```
%   'PlotTimeZoom', defaults to 0, outputs a graph of a specified time
frame
%   'TimeZoom', defaults to 48, specifies length of time frame ploted
%   'StartZoom', defaults to 0, specifies starting point of time frame
ploted
%
%Optional arguments Multiple and Stochastic Oscillations:
%   'NoOfOscillators', defaults to 1, defines number of simulations to run
and average
%   'StochasticMode', defaults to 0, 0 = purely deterministic, 1 = randomly
shifted deterministic curves,
%       2 = stochastic equations featuring a noise term
%   'ShiftSTDPercent', defaults to 5, STD of random period shift of curves
(for StochasticMode = 1)
%   'TypeOfNoise', defaults to 0, 0 = no noise, 1 = constant white noise,
%       2 = noise term relative to reactant concentration
%   'NoiseScale', defaults to 0.01, intensity of noise
%   'NoiseTermPlot', defaults to 0, plots a sample noise term curve
%   'StochasticPlot', defaults to 0, plots specified number of individual
stochastic curves
%   'PlotNumberStochPaths', defaults to 5, specifies number of individual
curves to plot above
%
%Optional arguments Phase Response Curve:
%   'PhaseResponse', defaults to 0, 1 = analyze phase response curve over
one period
%   'PRCVariable', defaults to 1, which variable to use in analysis
%   'PRCTime', defaults to 120, length of PRC simulation run
%   'PRCTimeStep', defaults to 0.1, time resolution of PRC simulation run
%   'PRCPoints', defaults to 50, number of points used over one period for
phase shifts
%   'PlotPRCPoint', defaults to 2, which points to plot: 2 = average,
%       3:PRCPoints+2 = individual points, possible to plot any number
%
%Optional arguments Bifurcation Plot:
%   'BifurcationPlot', defaults to 0, initiates bifurcation plotting
function
%   'BifurParameter', defaults to 1, selects parameter to investigate
%   'BifurRefVariable', defaults to 1, selects which variable to use as a
reference
%   'BifurMinimum', defaults to 0, minimum parameter value for bifurcation
run
%   'BifurMaximum', defaults to 2, maximum parameter value for bifurcation
run
%   'BifurSteps', defaults to 0.01, step size from minimum to maximum value
%   'BifurTime', defaults to 240, length of sample simulation for
bifurcation test
%   'BifurTimeStep', defaults to 0.1, time resolution of sample simulation
%
%Optional arguments Sensitivity Plot:
%   'SensitivityPlot', defaults to 0, runs a sensitivity test by varying
each parameter in turn as specified
%   'SensiRefVariable', defaults to 1, selects which variable to use as a
reference
%   'SensiTime', defaults to 240, length of sample simulation for
sensitivity test
%   'SensiShift', defaults to [-10,10], regime of shifts to apply to each
parameter
%
%Optional arguments Light Regime:
%   'LightID', defaults to 0, 0 = constant darkness, 1 = constant light, 2
= single light pulse,
%           3 = regular light/dark cycles, 4 = customized light regime
%   'LightResponse', defaults to 0.05, strength of light stimulus
%   'PulseStart', defaults to 24, starting time of light pulse (for LightID
= 2)
%   'PulseEnd', defaults to 36, end time of light pulse (for LightID = 2)
%   'FotoPeriod', defaults to 24,length of light/dark cycle (for LightID =
3)
%   'LightToDark', defaults to 0.5, proportion of cycle that has light (for
LightID = 3)


%Required and Optional Inputs
% ----------------------------------------------------------

cl = inputParser;
        %passed for initial values and plotting
cl.addRequired('EndTime', @(x)isnumeric(x)&(x>0));
cl.addRequired('TimeStep', @(x)isnumeric(x)&(x>0));
cl.addOptional('IVCalibration', 0,
@(x)(x==0)||(x==1)||(x==2)||(x==3)||(x==4));
cl.addOptional('SynchVariable', 1, @(x)isnumeric(x)&(x>0));
cl.addOptional('CalibrationTime', 120, @(x)isnumeric(x)&(x>0));
cl.addOptional('PlotTimeRange', 1, @(x)(x==0)||(x==1));
cl.addOptional('PlotTimeZoom', 0, @(x)(x==0)||(x==1));
cl.addOptional('TimeZoom', 48, @(x)isnumeric(x)&(x>0));
cl.addOptional('StartZoom', 0, @isnumeric);
        %passed for multiple and stochastic oscillations
cl.addOptional('NoOfOscillators', 1, @(x)isnumeric(x)&(x>0));
cl.addOptional('StochasticMode', 0, @(x)(x==0)||(x==1)||(x==2));
cl.addOptional('ShiftSTDPercent', 5, @(x)isnumeric(x)&(x>0));
cl.addOptional('TypeOfNoise', 0, @(x)(x==0)||(x==1)||(x==2));
cl.addOptional('NoiseScale', 0.01, @(x)isnumeric(x)&(x>0));
cl.addOptional('NoiseTermPlot', 0, @(x)(x==0)||(x==1));
cl.addOptional('StochasticPlot', 0, @(x)(x==0)||(x==1));
cl.addOptional('PlotNumberStochPaths', 5, @(x)isnumeric(x)&(x>0));
        %passed for PRC function
cl.addOptional('PhaseResponse', 0, @(x)(x==0)||(x==1));
cl.addOptional('PRCVariable', 1, @(x)isnumeric(x)&(x>0));
cl.addOptional('PRCTime', 120, @(x)isnumeric(x)&(x>0));
cl.addOptional('PRCTimeStep', 0.1, @(x)isnumeric(x)&(x>0));
cl.addOptional('PRCPoints', 50, @(x)isnumeric(x)&(x>0));
cl.addOptional('PlotPRCPoint', 2, @isnumeric);
        %passed for bifurcation plot function
cl.addOptional('BifurcationPlot', 0, @(x)(x==0)||(x==1));
cl.addOptional('BifurParameter', 1, @(x)isnumeric(x)&(x>0));
cl.addOptional('BifurRefVariable', 1, @(x)isnumeric(x)&(x>0));
cl.addOptional('BifurMinimum', 0, @(x)isnumeric(x)&(x>0));
cl.addOptional('BifurMaximum', 2, @(x)isnumeric(x)&(x>0));
cl.addOptional('BifurSteps', 0.01, @(x)isnumeric(x)&(x>0));
cl.addOptional('BifurTime', 240, @(x)isnumeric(x)&(x>0));
cl.addOptional('BifurTimeStep', 0.1, @(x)isnumeric(x)&(x>0));
        %passed for sensitivity plot function
cl.addOptional('SensitivityPlot', 0, @(x)(x==0)||(x==1));
cl.addOptional('SensiRefVariable', 1, @(x)isnumeric(x)&(x>0));
cl.addOptional('SensiTime', 240, @(x)isnumeric(x)&(x>0));
cl.addOptional('SensiShift', [-10,10], @isnumeric);
        %passed to light regime local function
cl.addOptional('LightID', 0, @(x)(x==0)||(x==1)||(x==2)||(x==3)||(x==4));
cl.addOptional('LightResponse', 0.05, @isnumeric);
cl.addOptional('PulseStart', 24, @(x)isnumeric(x)&(x>0));
cl.addOptional('PulseEnd', 36, @(x)isnumeric(x)&(x>0));
cl.addOptional('FotoPeriod', 24, @(x)isnumeric(x)&(x>=0));
cl.addOptional('LightToDark', 0.5, @(x)isnumeric(x)&(x>=0));
        %parameter passed to system of differential equations
cl.addOptional('Par_a', 0.9, @(x)(isnumeric(x) & (x>0) & (x<=1)));
cl.addOptional('Par_v1', 0.35, @(x)(isnumeric(x) & (x>0) & (x<=1)));
cl.addOptional('Par_v2', 0.45, @(x)(isnumeric(x) & (x>0) & (x<=1)));
cl.addOptional('Par_kdm', 0.45, @(x)(isnumeric(x) & (x>0) & (x<=1)));
cl.addOptional('Par_kdeg', 0.5, @(x)(isnumeric(x) & (x>0) & (x<=1)));
cl.addOptional('Par_kd', 0.5, @(x)(isnumeric(x) & (x>0) & (x<=1)));
cl.addOptional('Par_k1', 0.36, @(x)(isnumeric(x) & (x>0) & (x<=1)));
cl.addOptional('Par_k2', 0.45, @(x)(isnumeric(x) & (x>0) & (x<=1)));
cl.addOptional('Par_k3', 0.63, @(x)(isnumeric(x) & (x>0) & (x<=1)));
cl.addOptional('Par_k4', 0.27, @(x)(isnumeric(x) & (x>0) & (x<=1)));
cl.addOptional('Par_ka', 0.27, @(x)(isnumeric(x) & (x>0) & (x<=1)));
cl.addOptional('Par_kp', 0.36, @(x)(isnumeric(x) & (x>0) & (x<=1)));
cl.addOptional('Par_kcl1', 0.45, @(x)(isnumeric(x) & (x>0) & (x<=1)));
cl.addOptional('Par_kcl2', 0.45, @(x)(isnumeric(x) & (x>0) & (x<=1)));
cl.addOptional('Par_n', 0.4, @(x)(isnumeric(x) & (x>0) & (x<=1)));

cl.parse(EndTime, TimeStep, varargin{:});
ClockMod = cl.Results;

% Passing Inputs to Various Local Functions
% --------------------------------------------------------------

IVCal = ClockMod.IVCalibration;
SynchVar = ClockMod.SynchVariable;
CalTend = ClockMod.CalibrationTime;
PlotAll = ClockMod.PlotTimeRange;
PlotZoom = ClockMod.PlotTimeZoom;
ZoomTime = ClockMod.TimeZoom;
ZoomSt = ClockMod.StartZoom;
ZoomMarker = 0;

Osci = ClockMod.NoOfOscillators;
StochID = ClockMod.StochasticMode;
STDPercShift = ClockMod.ShiftSTDPercent;
NoiseID = ClockMod.TypeOfNoise;
NoiseScale = ClockMod.NoiseScale;
StochPlot = ClockMod.StochasticPlot;
StochPlotsNo = ClockMod.PlotNumberStochPaths;
NoisePlot = ClockMod.NoiseTermPlot;

PRC = ClockMod.PhaseResponse;
PRCVar = ClockMod.PRCVariable;
PRCTend = ClockMod.PRCTime;
PRCRes = ClockMod.PRCTimeStep;
PhasePoints = ClockMod.PRCPoints;
PlotPRC = ClockMod.PlotPRCPoint;
PRCID = 0;

BifPlot = ClockMod.BifurcationPlot;
BifParNo = ClockMod.BifurParameter;
BifRefVar = ClockMod.BifurRefVariable;
BifParMin = ClockMod.BifurMinimum;
BifParStep = ClockMod.BifurSteps;
BifParMax = ClockMod.BifurMaximum;
BifRes = ClockMod.BifurTimeStep;
BifTend = ClockMod.BifurTime;

SensPlot = ClockMod.SensitivityPlot;
SensVar = ClockMod.SensiRefVariable;
SensTend = ClockMod.SensiTime;
ParaShift = ClockMod.SensiShift;

LR1 = ClockMod.LightID;
LR2 = ClockMod.LightResponse;
LR3 = ClockMod.PulseStart;
LR4 = ClockMod.PulseEnd;
LR5 = ClockMod.FotoPeriod;
LR6 = ClockMod.LightToDark;

P(1) = ClockMod.Par_a;
P(2) = ClockMod.Par_v1;
P(3) = ClockMod.Par_v2;
P(4) = ClockMod.Par_kdm;
P(5) = ClockMod.Par_kdeg;
P(6) = ClockMod.Par_kd;
P(7) = ClockMod.Par_k1;
P(8) = ClockMod.Par_k2;
```

```
P(9) = ClockMod.Par_k3;
P(10) = ClockMod.Par_k4;
P(11) = ClockMod.Par_ka;
P(12) = ClockMod.Par_kp;
P(13) = ClockMod.Par_kcl1;
P(14) = ClockMod.Par_kcl2;
P(15) = ClockMod.Par_n *10;              % adjust hill coeficient to
facilitate beta sampling

ParaNames =
{'a','v1','v2','kdm','kdeg','kd','k1','k2','k3','k4','ka','kp','kcl1','kcl2
','n'};
VarNames = {'c1am','cry1a','ClockBmal','p1c3m','per1cry3'};

% Initial Values
% ------------------------------------------------------------------

c1am0 = 0.7;            % define initial conditions for variables
cry1a0 = 0.4;
ClockBmal0 = 1.5;
p1c3m0 = 1;
per1cry30 = 0.6;

xIn = [c1am0, cry1a0, ClockBmal0, p1c3m0, per1cry30];

% Initial Value Correction
% ------------------------------------------------------------------

x0 = ivcorrection(IVCal,xIn,P,1);       % calls local IV correction
function

if StochID == 2              % if SDEs selected, call alternative local IV
correction function

    x0 = stochivcorrection(IVCal,xIn,P,0);

end

disp('    Adjusted Initial Conditions');             %print out the initial
conditions forwarded for main simulation
disp(x0);

% Integration Proper
% ------------------------------------------------------------------

% apply ode/sde solver to underlying equation, specifying time, initial
% conditions, and a range of optional parameters

ShiftTable = (1+(STDPercShift/100*randn(Osci,1))); %random table for
shifting deterministic curves if selected
IntegrationResults = zeros((EndTime/TimeStep+1),length(x0),Osci);

for os = 1:Osci

    fprintf('Oscillator Number %d...\n', os);

    if StochID == 0     % purely deterministic simulation

        % ------------------------------------------------------------------
        [Time,Data] = ode45(@(Time,InitialConds)
lightinput(Time,InitialConds,LR1,LR2,LR3,LR4,LR5,LR6,P,1),[0:TimeStep:EndTi
me],x0);
        % ------------------------------------------------------------------

        IntegrationResults(:,:,os) = Data;

    elseif StochID == 1 % deterministic simulations shifted according to
random table generated above
```

```
        if IVCal == 1 || IVCal == 3

            xShift = ivcorrection(IVCal,x0,P,ShiftTable(os));

        else

            xShift = x0;    % only modify IV in free-running scenarios (to
de-synchronize), as already very similar otherwise

        end

        % ------------------------------------------------------------
        [Time,Data] = ode45(@(Time,InitialConds)
lightinput(Time,InitialConds,LR1,LR2,LR3,LR4,LR5,LR6,P,ShiftTable(os)),[0:T
imeStep:EndTime],xShift);
        % ------------------------------------------------------------

        IntegrationResults(:,:,os) = Data;

    elseif StochID == 2 % stochastic simulation with variable noise term
        %!!!calls SDE Solver in external function!!!%

        if IVCal == 1 || IVCal == 3

            xStoch = stochivcorrection(IVCal,x0,P,NoiseID);

        else

            xStoch = x0;    % only modify IV in free-running scenarios (to
de-synchronize), as still very similar otherwise

        end

        % ------------------------------------------------------------
        [Data,Noise] = sde_euler(@(Time,InitialConds)
lightinput(Time,InitialConds,LR1,LR2,LR3,LR4,LR5,LR6,P,1),@(Time,InitialCon
ds) noiseterm(Time,InitialConds,NoiseID),[0:TimeStep:EndTime],xStoch);
        % ------------------------------------------------------------

        IntegrationResults(:,:,os) = Data;
        Time = [0:TimeStep:EndTime];

    end

end
clear os

% Save and plot Results
% ------------------------------------------------------------

Data = mean(IntegrationResults,3);      % mean of all simulation runs

ClockResults = ClockMod;
ClockResults.TimeData = Time;            % save model datapoints into
struct
ClockResults.ModelData = Data;

if PlotAll == 1

    figure;
    clockplot(ClockResults);        % visualize model results

end

% Plotting Sample Stochastic Paths
% ------------------------------------------------------------
```

```
if StochPlot == 1

    StochResults = ClockResults;

    figure;
    hold on;        % plot specified number of individual stochastic curves

    for pp = 1:min(StochPlotsNo,Osci)

        StochResults.ModelData = IntegrationResults(:,:,pp);
        clockplot(StochResults);
        title([num2str(min(StochPlotsNo,Osci)),' Individual Stochastic
Paths for Circadian Simulation'],'fontsize',16);

    end
    clear pp

    hold off;

end

% Plotting Sample Noiseterm
% ------------------------------------------------------------

if NoisePlot == 1               % plot a single sample noise term

    NoiseResults = ClockResults;

    if NoiseID == 0     % no noise

        AdjustedNoise = Noise*0;
        AdjustedNoise(end,:) = 1;           %at least one non-zero value
allows smooth plotting

    elseif NoiseID == 1 % constant white noise

        AdjustedNoise = Noise*NoiseScale;

    elseif NoiseID == 2 % noise directly relative to reactant concentration
        size(Noise)
        size(IntegrationResults(:,:,Osci))
        AdjustedNoise = Noise.*IntegrationResults(:,:,Osci)*NoiseScale;

    end

    NoiseResults.ModelData = AdjustedNoise;

    figure;
    clockplot(NoiseResults);
    title(['Sample Noise Term of Type ', num2str(NoiseID), ' and Intensity
', num2str(NoiseScale)],'fontsize',16);

end

% Zooming in on Specific Time Window
% ------------------------------------------------------------

if PlotZoom == 1

    ZoomResults = ClockResults;
    ZoomResults.TimeData =
ZoomResults.TimeData((ZoomSt/TimeStep+1):(ZoomSt+ZoomTime)/TimeStep+1);
    ZoomResults.ModelData =
ZoomResults.ModelData((ZoomSt/TimeStep+1):(ZoomSt+ZoomTime)/TimeStep+1,:);
    ZoomMarker = 1;             %marker to differentiate zoom from main
plot

    figure;
    hold on;

    clockplot(ZoomResults);
    title('Local Magnification for Oscillation System','fontsize',16);

    hold off;
    ZoomMarker = 0;

end

disp(ClockResults);

% Phase Response Curve
% ------------------------------------------------------------

if PRC == 1

    phaseresponse(PRCVar);      %start nested function and plot
    disp(PRCurve);
    ColourString = 'bgrky';
    ColourMarker = 1;
    PRCID = 1;                                    %marker to differentiate
main simulation from PRC run

    figure;
    hold on;

    for ppr = [PlotPRC]         %pick data to plot relative to reference
data, where 1 = reference data, 2 = average phase shift, 3:(PhasePoints+2)
= each individual pulse point

        PlotPoint = ppr;
        clockplot(PRCurve);
        ColourMarker = ColourMarker+1;

    end
    clear ppr

    hold off;
    PRCID = 0;

end

% Bifurcation Plot by Parameter
% ------------------------------------------------------------

if BifPlot == 1

    BifParRange = [BifParMin:BifParStep:BifParMax];

    bifurcation(BifParRange,BifParNo,BifRefVar);    %start nested function
and plot

end

% Sensitivity Analysis by Parameter
% ------------------------------------------------------------
%!!!calls external function for determining period and amplitude!!!%

if SensPlot == 1

    sensiplot;      %start nested function and plot

end

% ------------------------------------------------------------
% Nested Functions
% ------------------------------------------------------------
```

```
    % Defining Light Regime For Solver
    % ------------------------------------------------------------

    function dL =
lightinput(Time,InitialConds,LightID,LightResponse,PulseStart,PulseEnd,Foto
Period,LightToDark,ModelParams,Shifting)

        %define light intensity and regime to be forwarded for solving the
system of differential equations

        if LightID == 0         %constant darkness

            Light = 0;

        elseif LightID == 1     %constant light

            Light = LightResponse;

        elseif LightID == 2     %single light pulse

            Light = LightResponse*(Time >= PulseStart)*(Time < PulseEnd);

        elseif LightID == 3     %regular light cycles

        LightPeriod = FotoPeriod*LightToDark;
        ReferencePeriod = mod(Time,FotoPeriod);
        Light = LightResponse*(ReferencePeriod < LightPeriod);

        elseif LightID == 4     %custom profile

            if sin(Time*pi/12) >= 0 && Time < 96         %light/dark cycle
for 96 hours

                Light = LightResponse;

            elseif (96 <= Time) && (Time < 196)          %constant light
for next 100 hours

                Light = LightResponse;

            else

                Light = 0;                              %no light
afterwards

            end

        end

        dL = clockmodel(InitialConds,Light,ModelParams,Shifting);
%forwad conditions to differential equations

    end

    % Deterministic Initial Value Correction
    % ------------------------------------------------------------

    function AdjustedIV = ivcorrection(IVCalFun,x0,P,Shift)

        if IVCalFun == 1 || IVCalFun == 2       %callibration simulation to
find suitable initial conditions

            [~,CalVal] = ode45(@(Time,InitialConds)
lightinput(Time,InitialConds,0,0,0,0,0,0,P,Shift),[0:CalTend],x0);
            AdjustedIV = CalVal(end,:);                 %end values of
calibraton simulation
```

```
            if IVCalFun == 2                     %set initial conditions
for minimum levels of selected variable

                [~,CalVal] = ode45(@(Time,InitialConds)
lightinput(Time,InitialConds,0,0,0,0,0,0,P,Shift),[0:CalTend],AdjustedIV);
                CalVal = CalVal(round(CalTend/2):end,:);
                SynVal = find(CalVal(:,SynchVar) ==
min(CalVal(:,SynchVar)),1,'last');     %find minimum of selected variable
and set all values accordingly
                AdjustedIV = CalVal(SynVal,:);
%this way, the selected variable always starts from a trough

            end

        elseif IVCalFun == 3|| IVCalFun == 4

            [~,CalVal] = ode45(@(Time,InitialConds)
lightinput(Time,InitialConds,LR1,LR2,LR3,LR4,LR5,LR6,P,Shift),[0:CalTend],x
0);
            AdjustedIV = CalVal(end,:);                 %end values of
calibraton simulation

            if IVCalFun == 4

                [~,CalVal] = ode45(@(Time,InitialConds)
lightinput(Time,InitialConds,LR1,LR2,LR3,LR4,LR5,LR6,P,Shift),[0:CalTend],A
djustedIV);
                CalVal = CalVal(round(CalTend/2):end,:);
                SynVal = find(CalVal(:,SynchVar) ==
min(CalVal(:,SynchVar)),1,'last');     %find minimum of selected variable
and set all values accordingly
                AdjustedIV = CalVal(SynVal,:);
%this way, the selected variable always starts from a trough

            end

        else

            AdjustedIV = x0;

        end

    end

    % Stochastic Initial Value Correction
    % ------------------------------------------------------------
    %!!!calls SDE Solver in external function!!!%

    function AdjustedStochIV = stochivcorrection(IVCalFun,x0,P,CalNoiseID)

        if IVCalFun == 1 || IVCalFun == 2       %callibration simulation to
find suitable initial conditions

            StochCalVal = sde_euler(@(Time,InitialConds)
lightinput(Time,InitialConds,0,0,0,0,0,0,P,1),@(Time,InitialConds)
noiseterm(Time,InitialConds,CalNoiseID),[0:TimeStep:CalTend],x0);
            AdjustedStochIV = StochCalVal(end,:);                %end
values of calibraton simulation

            if IVCalFun == 2                     %set initial conditions
for minimum levels of selected variable

                StochCalVal = sde_euler(@(Time,InitialConds)
lightinput(Time,InitialConds,0,0,0,0,0,0,P,1),@(Time,InitialConds)
noiseterm(Time,InitialConds,CalNoiseID),[0:TimeStep:CalTend],AdjustedStochI
V);
                StochCalVal = StochCalVal(round(CalTend/2):end,:);
                StochSynVal = find(StochCalVal(:,SynchVar) ==
min(StochCalVal(:,SynchVar)),1,'last');      %find minimum of selected
variable and set all values accordingly
                AdjustedStochIV = StochCalVal(StochSynVal,:);
%this way, the selected variable always starts from a trough

            end

        elseif IVCalFun == 3|| IVCalFun == 4

            StochCalVal = sde_euler(@(Time,InitialConds)
lightinput(Time,InitialConds,LR1,LR2,LR3,LR4,LR5,LR6,P,1),@(Time,InitialCon
ds) noiseterm(Time,InitialConds,CalNoiseID),[0:TimeStep:CalTend],x0);
            AdjustedStochIV = StochCalVal(end,:);                 %end
values of calibraton simulation

            if IVCalFun == 4

                StochCalVal = sde_euler(@(Time,InitialConds)
lightinput(Time,InitialConds,LR1,LR2,LR3,LR4,LR5,LR6,P,1),@(Time,InitialCon
ds)
noiseterm(Time,InitialConds,CalNoiseID),[0:TimeStep:CalTend],AdjustedStochI
V);
                StochCalVal = StochCalVal(round(CalTend/2):end,:);
                StochSynVal = find(StochCalVal(:,SynchVar) ==
min(StochCalVal(:,SynchVar)),1,'last');      %find minimum of selected
variable and set all values accordingly
                AdjustedStochIV = StochCalVal(StochSynVal,:);
%this way, the selected variable always starts from a trough

            end

        else

            AdjustedStochIV = x0;

        end

    end

    % Lightcurve as a Function of Time
    % ------------------------------------------------------------

    function [LightResult] = lightcurve(OptStr,VarTend,VarRes)

        %copied and modified from 'lightinput' to generate seperate
lightcurve data e.g. for plotting
        %(due to how the solver works, reading data out from the simulation
does not work well)

        LightResult = zeros(VarTend/VarRes+1,1);

        for lo = 1:VarTend/VarRes+1

            if OptStr.LightID == 0          %constant darkness

                LightResult(lo) = 0;

            elseif OptStr.LightID == 1      %constant light

                LightResult(lo) = OptStr.LightResponse;

            elseif OptStr.LightID == 2      %single light pulse

                LightResult(lo) = OptStr.LightResponse*(((lo-1)*VarRes) >=
OptStr.PulseStart & ((lo-1)*VarRes) <= OptStr.PulseEnd);

            elseif OptStr.LightID == 3      %regular light cycles
```

```
                LightPeriod = OptStr.FotoPeriod*OptStr.LightToDark;
                ReferencePeriod = mod(((lo-1)*VarRes),OptStr.FotoPeriod);
                LightResult(lo) = OptStr.LightResponse*(ReferencePeriod <=
LightPeriod);

            elseif OptStr.LightID == 4     %custom profile

                if (sin(((lo-1)*VarRes)*pi/12) >= 0) && (((lo-1)*VarRes) <
96)         %light/dark cycle for 96 hours

                    LightResult(lo) = OptStr.LightResponse;

                elseif (96 <= ((lo-1)*VarRes)) && (((lo-1)*VarRes) <= 196)
%constant light for next 100 hours

                    LightResult(lo) = OptStr.LightResponse;

                else

                    LightResult(lo) = 0;    %no light afterwards

                end

            end

        end
        clear lo

    end

    % Plotting Variables vs Time and Light Input
    % ------------------------------------------------------------

    function clockplot(PlotStruct)

        T = PlotStruct.TimeData;
        D = PlotStruct.ModelData;
        Ytop = max(max(D));             % maximum value used to put
markings in relation to scale of plot

        if PRCID == 1                   % special phase response curve plot

            PlotStruct.PulseStart = PlotStruct.PulseStartList(PlotPoint-1);
            PlotStruct.PulseEnd = PlotStruct.PulseEndList(PlotPoint-1);
            LiCu = lightcurve(PlotStruct,T(end),T(2))*4;          %light
strength boosted by factor of 4 for visibility

            plot(T,LiCu,':c', T,D(:,1),'b','Linewidth',3);

plot(T,D(:,PlotPoint),'Color',ColourString(mod(ColourMarker,5)+1),'Linewidt
h',3, 'LineStyle', '--');
            title(['Reference and Light-pulsed Simulations for ',
VarNames{PRCVar},', plotting No ' num2str(PlotPRC)],'fontsize',16);
            xlabel('Time','fontsize',16);
            ylabel('Concentration','fontsize',16);
            legend('Light Input', 'Reference Simulation', 'Phaseshifted
Simulation');
            ylim([0, 1.5]);

        else

            if ZoomMarker == 1              % adjusted light calculation
for zoom plot

                LiCu =
lightcurve(PlotStruct,ClockResults.EndTime,ClockResults.TimeStep)*2;
%light strength boosted by factor of 2 for visibility
```

```
                LiCu =
LiCu((ZoomSt/TimeStep+1):(ZoomSt+ZoomTime)/TimeStep+1,:);

            else

                LiCu = lightcurve(PlotStruct,T(end),T(2))*4;
%light strength boosted by factor of 4 for visibility

            end

            if length(D(end,:)) == 5

                plot(T,D(:,1),'g:', T,D(:,2),'g-', T, D(:,3),'r-',
T,D(:,4),'b:', T,D(:,5),'b-', T,LiCu,':c', 'Linewidth',3);
                title(['Mean of ', num2str(Osci), ' Oscillator Systems with
5 Variables + Light'],'fontsize',16);
                xlabel('Time','fontsize',16);
                ylabel('Concentration','fontsize',16);
                legend('Cry1a mRNA', 'Cry1a', 'ClockBmal','Per1
mRNA','Per1','Light Input');

            else                    % automated plotting for unexpected
number of variables

                plot(T,D, T,LiCu,':c', 'Linewidth',3);
                title('System with unexpected Number of Variables! - edit
clockplot','fontsize',16);
                xlabel('Time','fontsize',16);
                ylabel('Concentration','fontsize',16);

            end

        end

        if ZoomMarker == 1          % add markings for dusk and dawn

            DuskLine =
round(ClockResults.FotoPeriod*ClockResults.LightToDark*10);

            for zl = ZoomSt:0.1:(ZoomSt+ZoomTime)

                if round(mod(zl,ClockResults.FotoPeriod)*10) == DuskLine

                    plot([zl, zl], [0,Ytop],'b-.','linewidth',2);

                elseif mod(zl,ClockResults.FotoPeriod) == 0

                    plot([zl, zl], [0,Ytop],'m-.','linewidth',2);

                end

            end
            clear zl

        end

        FoPe = PlotStruct.FotoPeriod;
        LtD = PlotStruct.LightToDark;

        if PlotStruct.LightID == 0                  %constant darkness


rectangle('Position',[T(1),0,T(end),Ytop/20],'Curvature',[0,0],'FaceColor',
'k');

        elseif PlotStruct.LightID == 1              %constant light
```

```
rectangle('Position',[T(1),0,T(end),Ytop/20],'Curvature',[0,0],'FaceColor',
'w');

        elseif PlotStruct.LightID == 2              %single light pulse


rectangle('Position',[T(1),0,T(end),Ytop/20],'Curvature',[0,0],'FaceColor',
'k');

rectangle('Position',[PlotStruct.PulseStart,0,(PlotStruct.PulseEnd-
PlotStruct.PulseStart),Ytop/20],'Curvature',[0,0],'FaceColor','w');

        elseif PlotStruct.LightID == 3              %regular light cycles


rectangle('Position',[T(1),0,T(end),Ytop/20],'Curvature',[0,0],'FaceColor',
'w');
           for rc=1:(ceil(T(end)/FoPe))

               if FoPe*rc < T(end)

                   rectangle('Position',[FoPe*(rc-(1-LtD)),0,FoPe*(1-
LtD),Ytop/20],'Curvature',[0,0],'FaceColor','k');

               elseif T(end)-(FoPe*(rc-(1-LtD))) > 0

                   rectangle('Position',[FoPe*(rc-(1-LtD)),0,T(end)-
(FoPe*(rc-(1-LtD))),Ytop/20],'Curvature',[0,0],'FaceColor','k');

               end

           end
           clear rc

        elseif PlotStruct.LightID == 4              %custom profile


rectangle('Position',[T(1),0,T(end),Ytop/20],'Curvature',[0,0],'FaceColor',
'w');
           for rc=1:(ceil(96/FoPe)+1)

               if FoPe*rc < 96

                   rectangle('Position',[FoPe*(rc-(1-LtD)),0,FoPe*(1-
LtD),Ytop/20],'Curvature',[0,0],'FaceColor','k');

               elseif 96-(FoPe*(rc-(1-LtD))) > 0

                   rectangle('Position',[FoPe*(rc-(1-LtD)),0,96-(FoPe*(rc-
(1-LtD))),Ytop/20],'Curvature',[0,0],'FaceColor','k');

               else

                   rectangle('Position',[196,0,T(end)-
196,Ytop/20],'Curvature',[0,0],'FaceColor','k');

               end

           end
           clear rc

        end

        set(gca,'xtick',[0:FoPe:T(end)]);
        xlim([T(1), T(end)]);

    end
    % Phase Response Curve
    % -----------------------------------------------------------

    function phaseresponse(Var)

        xPRC = ivcorrection(2,xIn,P,1);

        [RefTime,RefVal] = ode45(@(Time,InitialConds)
lightinput(Time,InitialConds,0,0,0,0,0,0,P,1),[0:PRCRes:PRCTend],xPRC);
%create constant darkness reference simulation

        RefMax = 1+find(RefVal(2:end-1,Var)>RefVal(1:end-2,Var) &
RefVal(2:end-1,Var)>RefVal(3:end,Var));
        LastPer = RefTime(RefMax(end))-RefTime(RefMax(end-1));   %find
maxima and time difference between last two maxima

        PRCResults = zeros(PhasePoints,2);                       %empty
array for storage
        PRCInstances = zeros((PRCTend/PRCRes+1),PhasePoints);

        for pr =1:PhasePoints

            fprintf('Phase Point = %d...\n',pr);

            PointTime = pr*LastPer/PhasePoints;         %define points for
phase response test over the reference period
            [PRCTime, PRCData] = ode45(@(Time,InitialConds)
lightinput(Time,InitialConds,2,LR2,PointTime,PointTime+2,0,0,P,1),[0:PRCRes
:PRCTend],xPRC);  %simulation with light regime
                                                        %run simulations
with a light pulse at defined points
            PRCMax = 1+find(PRCData(2:end-1,Var)>PRCData(1:end-2,Var) &
PRCData(2:end-1,Var)>PRCData(3:end,Var));
            PhaseShift = PRCTime(PRCMax(end))-PRCTime(RefMax(end)); %find
maxima of pulsed simulation, and time difference between last pulsed and
reference maxima

            if PhaseShift > LastPer/2                   %adjust relative to
period

                PhaseShift = PhaseShift-LastPer;

            elseif PhaseShift < -LastPer/2

                PhaseShift = PhaseShift+LastPer;

            end

            PRCResults(pr,:) = [PointTime, PhaseShift]; %save pulse times
and resulting phase shifts
            PRCInstances(:,pr) = PRCData(:,Var);        %save data for
variable of interest for all pulse points

        end
        clear pr

        AveragedShift = mean(PRCInstances,2);           %calculate average
phase shift across all pulse points

        PRCurve = ClockResults;                        %store all results
in struct
        PRCurve.LightID = 2;
        PRCurve.TimeData = PRCTime;
        PRCurve.PulseStartList = [mean(PRCResults(:,1));PRCResults(:,1)];
        PRCurve.PulseEndList =
[mean(PRCResults(:,1)+2);(PRCResults(:,1)+2)];
        PRCurve.ModelData = [RefVal(:,Var),AveragedShift,PRCInstances];
        PRCurve.Results = PRCResults;

        figure;                                 %plot phase shifts
        hold on;
        plot(PRCResults(:,1),PRCResults(:,2),'b.');
        plot([0,LastPer],[0,0],'k--');
        hold off;
        title(['Phase Shifts caused by a Light Pulse at Various Points
throughout Period for ',VarNames{Var}],'fontsize',16);
        xlabel('Pulse time','fontsize',16);
        xlim([0, LastPer]);
        ylabel('Phase shift','fontsize',16);

    end

    % Bifurcation Plot
    % ------------------------------------------------------------

    function bifurcation(Range,Para,RefVar)

        BifMax = [];  % data (bifurcation diagram)
        BifMin = [];

        for bf = Range

                fprintf('Parameter Value = %4.2f\n',bf);
                P(Para) = bf;

                xBif = ivcorrection(2,xIn,P,1);     % IV correction and
proper simulation for adjusted value of parameter under investigation

                [~,x] = ode45(@(Time,InitialConds)
lightinput(Time,InitialConds,0,0,0,0,0,0,P,1),[0:BifRes:BifTend],xBif);

                for bm = 2:length(x(:,RefVar))-1    % finding maxima and
minima

                    if ((x(bm,RefVar) > x(bm-1,RefVar)) && (x(bm,RefVar) >
x(bm+1,RefVar)))

                        BifMax = [BifMax; bf, x(bm,RefVar)];

                    elseif ((x(bm,RefVar)< x(bm-1,RefVar)) && (x(bm,RefVar)
< x(bm+1,RefVar)))

                        BifMin = [BifMin; bf, x(bm,RefVar)];

                    end

                end
                clear bm

        end
        clear bf
                    % plotting maxima versus minima to reveal oscillating
behaviour
        figure;
        hold on;

plot(BifMin(:,1),BifMin(:,2),'ro','MarkerEdgeColor','r','MarkerFaceColor','
r','MarkerSize',1.5)

plot(BifMax(:,1),BifMax(:,2),'ro','MarkerEdgeColor','b','MarkerFaceColor','
b','MarkerSize',1.5)
        hold off;
        axis([Range(1), Range(end), 0, inf]);
        title('Bifurcation Plot showing Maximum and Minimum Concentrations
relative to varied Parameter','fontsize',16);
```

```
        xlabel(['Value inserted for Parameter
',ParaNames{Para}],'fontsize',16);
        ylabel(['Concentration at Maxima/Minima of
',VarNames{RefVar}],'fontsize',16);
        legend('Minimum Values', 'Maximum Values');

    end

    % Sensitivity Plot
    % --------------------------------------------------------------

    function sensiplot

        figure;

        xSen = ivcorrection(2,xIn,P,1); % IV correction and standard
simulation to obtain reference values

        [SensRefTime,SensRefVal] = ode45(@(Time,InitialConds)
lightinput(Time,InitialConds,0,0,0,0,0,0,P,1),[0:SensTend],xSen);
        SensReference = PHD_OOAA_HilbertAnalysis(SensRefVal, SensRefTime,
SensVar, 1);

        for sp = 1:length(ParaShift)

            fprintf('Sensitivity to a Shift of %d %%\n',ParaShift(sp));

            SensResults = zeros(length(P),4);

            for st = 1:length(P)
                                    % run IV correction and simulations for
each adjusted parameter and each shift
                PSen = P;
                PSen(st)=P(st)*(1+ParaShift(sp)/100);

                xSenShift = ivcorrection(2,xIn,PSen,1);

                [SensPerTime,SensPerVal] = ode45(@(Time,InitialConds)
lightinput(Time,InitialConds,0,0,0,0,0,0,PSen,1),[0:SensTend],xSenShift);
                SensPerturbed = PHD_OOAA_HilbertAnalysis(SensPerVal,
SensPerTime, SensVar, 1);

                SensResults(st,:) =
[SensPerturbed(1:2),(SensPerturbed(1:2)-
SensReference(1:2))./SensReference(1:2)];

            end
            clear st
                        % plot the difference in period and amplitude
resulting from shifting each parameter
            disp(SensResults);

            subplot(length(ParaShift),2,(2*sp-1))
            bar([1:length(P)],100*(SensResults(:,3)));
            ylabel('% Variation of the period')
            ylim([-100,inf])
            xlabel(['Parameter shifted by ',num2str(ParaShift(sp)),'%'])
            xlim([0, length(P)+1])
            set(gca,'XTickLabel',ParaNames)

            subplot(length(ParaShift),2,(2*sp))
            bar([1:length(P)],100*(SensResults(:,4)));
            ylabel('% Variation of the amplitude')
            ylim([-100,inf])
            xlabel(['Parameter shifted by ',num2str(ParaShift(sp)),'%'])
            xlim([0, length(P)+1])
            set(gca,'XTickLabel',ParaNames)
        end
        clear sp

    end

    % Noise Component for Stochastic Differential Equation
    % ------------------------------------------------------------

    function NT = noiseterm(~,InitialConds,NoiseIDFun)

        if NoiseIDFun == 0

            NT = 0;         % no noise

        elseif NoiseIDFun == 1

            NT = NoiseScale;    % constant white noise

        elseif NoiseIDFun == 2

            NT = InitialConds.*NoiseScale;  % noise relative to
concentration of reactants

        end

    end

end

function dC = clockmodel(x,Light,Par,Shift)

%specify underlying system of differential equations, including parameters,
%variables, and the equations themselves

% Parameters
% ------------------------------------------------------------

a = Par(1);
v1 = Par(2)*a;
v2 = Par(3)*a;
kdm = Par(4)*a;
kdeg = Par(5)*a;
kd = Par(6)*a;
k1 = Par(7)*a;
k2 = Par(8)*a;
k3 = Par(9)*a;
k4 = Par(10)*a;
ka = Par(11)*a;
kp = Par(12)*a;
kcl1 = Par(13)*a;
kcl2 = Par(14)*a;
n = Par(15);

% Variables
% ------------------------------------------------------------

c1am = x(1);
cry1a = x(2);
ClockBmal = x(3);
p1c3m = x(4);
per1cry3 = x(5);

% Equations
% ------------------------------------------------------------

dC = [
    Shift.*(v1*(ClockBmal/(ka + ClockBmal)) - (kdm + Light)*(c1am/(kdeg +
c1am))),       % d(c1am)/dt
```

```
    Shift.*(k1*c1am - kd*(cry1a/(kdeg + cry1a))),
% d(cry1a)/dt
    Shift.*(k2*(kcl1^n/(kcl1^n + cry1a^n)) + k3*(kcl2^n/(kcl2^n +
per1cry3^n)) - kd*(ClockBmal/(kdeg + ClockBmal))), % d(ClockBmal)/dt
    Shift.*(v2*(ClockBmal/(kp + ClockBmal)) - kdm*(p1c3m/(kdeg + p1c3m))),
% d(p1c3m)/dt
    Shift.*(k4*p1c3m - kd*(per1cry3/(kdeg + per1cry3)))
% d(per1cry3)/dt
    ];

end
```

Code for the Sequential Monte Carlo Algorithm

```
function [FinalValues] = PHD_OOAA_SMCParamGen(ReferenceData, LastTolerance,
ToleranceScaling, varargin)

%PHD_OOAA_SMCParamGen(ReferenceData, LastTolerance,
ToleranceScaling,[options])
% generates a set of plausible values for the parameter requested by
% randomly sampling from a multivariate distribution and forwarding the
result
% along with reference data to a validation function; the returned
difference to
% reference values is accpeted if below the respective population's
tolerance
% this is in turn determined by scaling the last Tolerance according to
input schedule

%Required and Optional Inputs
% -------------------------------------------------------------

smc = inputParser;

smc.addRequired('ReferenceData', @isnumeric);           % compare simulated
data to this
smc.addRequired('LastTolerance', @isnumeric);           % tolerance at
scale 1
smc.addRequired('ToleranceScaling', @isnumeric);        % scales tolerance
for different populations
smc.addOptional('AllParameters', 1, @(x)(x==0)||(x==1));    % sets (nearly)
all parameters to 0
smc.addOptional('ReRunPopulation', 0, @(x)isnumeric(x)&(x>=0)); % starts
algorith from specified intermediate population
smc.addOptional('Particles', 100, @(x)isnumeric(x)&(x>0)); % number of
particles obtained for each population
smc.addOptional('StochasticSims', 20, @(x)isnumeric(x)&(x>0));  % number of
stochastic simulations with each set of parameters
smc.addOptional('EstimatedValues', 0.5, @isnumeric); % estimated values for
parameters to compare to in graphical output
        %parameters used to run test simulations
smc.addOptional('SimulationTime', 240, @(x)isnumeric(x)&(x>0)); % duration
of validation simulation
smc.addOptional('SimulationTimeStep', 0.1, @(x)isnumeric(x)&(x>0)); %
resolution of validation simulation
smc.addOptional('SwitchOnSDE', 0, @(x)(x==0)||(x==1)); % switches from
deterministic to stochastic validation
smc.addOptional('NoiseLevel', 0.03, @(x)isnumeric(x)&(x>=0)); % magnitude
of noise term for SDEs
        %parameters passed to system of differential equations
smc.addOptional('Par_a', 0.9, @(x)(isnumeric(x) & (x>=0) & (x<=1)));
smc.addOptional('Par_v1', 0.35, @(x)(isnumeric(x) & (x>=0) & (x<=1)));
%passes specified value if not overriden by input
smc.addOptional('Par_v2', 0.45, @(x)(isnumeric(x) & (x>=0) & (x<=1)));
smc.addOptional('Par_kdm', 0.45, @(x)(isnumeric(x) & (x>=0) & (x<=1)));
smc.addOptional('Par_kdeg', 0.5, @(x)(isnumeric(x) & (x>=0) & (x<=1)));
smc.addOptional('Par_kd', 0.5, @(x)(isnumeric(x) & (x>=0) & (x<=1)));
smc.addOptional('Par_k1', 0.36, @(x)(isnumeric(x) & (x>=0) & (x<=1)));
smc.addOptional('Par_k2', 0.45, @(x)(isnumeric(x) & (x>=0) & (x<=1)));
smc.addOptional('Par_k3', 0.63, @(x)(isnumeric(x) & (x>=0) & (x<=1)));
smc.addOptional('Par_k4', 0.27, @(x)(isnumeric(x) & (x>=0) & (x<=1)));
smc.addOptional('Par_ka', 0.27, @(x)(isnumeric(x) & (x>=0) & (x<=1)));
smc.addOptional('Par_kp', 0.36, @(x)(isnumeric(x) & (x>=0) & (x<=1)));
smc.addOptional('Par_kcl1', 0.45, @(x)(isnumeric(x) & (x>=0) & (x<=1)));
smc.addOptional('Par_kcl2', 0.45, @(x)(isnumeric(x) & (x>=0) & (x<=1)));
smc.addOptional('Par_LI', 0.5, @(x)(isnumeric(x) & (x>=0) & (x<=1)));
smc.addOptional('Par_n', 0.4, @(x)(isnumeric(x) & (x>=0) & (x<=1)));

smc.parse(ReferenceData,LastTolerance,ToleranceScaling,varargin{:});
SMCParam = smc.Results;
                        % names used for labeling
ParaNames =
{'a','v1','v2','kdm','kdeg','kd','k1','k2','k3','k4','ka','kp','kcl1','kcl2
','LI','n'};

% Passing Inputs to Various Local Functions
% ------------------------------------------------------------

FindAll = SMCParam.AllParameters;
ReRun = SMCParam.ReRunPopulation;

SimTi = SMCParam.SimulationTime;
SimTiStp = SMCParam.SimulationTimeStep;
SDEId = SMCParam.SwitchOnSDE;
NoiseLvl = SMCParam.NoiseLevel;

EstimatedValues = SMCParam.EstimatedValues;
Particles = SMCParam.Particles;
StochasticSims = SMCParam.StochasticSims;

if SDEId == 0       % in deterministic mode only do one simulation per
parameter set

    StochasticSims = 1;

end

PriorTolerance = ToleranceScaling(1)*LastTolerance; % used to reject/accept
from prior to first intermediate
InterTolScale = [ToleranceScaling(2:end),0];        % drop 1st for prior,
add 0 for last step lookahead calculation
InterTolSchedule = InterTolScale'*LastTolerance;    % determine each
population's tolerance value
Populations = length(InterTolScale)-1;              % from first
intermediate to final population

P(1) = SMCParam.Par_a;
P(2) = SMCParam.Par_v1;
P(3) = SMCParam.Par_v2;
P(4) = SMCParam.Par_kdm;
P(5) = SMCParam.Par_kdeg;
P(6) = SMCParam.Par_kd;
P(7) = SMCParam.Par_k1;
P(8) = SMCParam.Par_k2;
P(9) = SMCParam.Par_k3;
P(10) = SMCParam.Par_k4;
P(11) = SMCParam.Par_ka;
P(12) = SMCParam.Par_kp;
P(13) = SMCParam.Par_kcl1;
P(14) = SMCParam.Par_kcl2;
P(15) = SMCParam.Par_LI;
P(16) = SMCParam.Par_n;
```

```
if FindAll == 0         % sets all parameters to 0 except 'a' and 'n'

    P(2:15) = 0;

end

MissingParams = find(P == 0);               % find parameters set to 0 to
be evaluated by this function
NumberVars = length(MissingParams);         % number of missing parameters

%% Moving from the prior to the first intermediate population
%-------------------------------------------------------------

% m particles sampled from a Beta, n expected values and corresponding
weight
% of values accepted within tolerance calculated

if ReRun == 0

    InterStart = 1;         % normally start with this intermediary
population
    fprintf('Prior Sampling...\n');
    SamplingResults = zeros(Particles,(NumberVars+2));
    ViableLookahead = 0;

    while ViableLookahead < 2   % repeat until there are at least 2
positive lookahead probabilities in the population
                                % otherwise re-normalizing the weight with
one particle in turn excluded will produce NAN
        for m = 1:Particles

            if mod(m,10) == 0

                fprintf('%d',m);

            else

                fprintf('*');

            end

            ProportionWithin = 0;

            while ProportionWithin == 0   % satisfy that at least one
simulation is within tolerance

                for v = 1:NumberVars

                PriorSample(v) = betarnd(2,2);  % random sample from
Betadistribution(2,2)
                P(MissingParams(v)) = PriorSample(v);

                end
                clear v

                AcceptedArray = zeros(StochasticSims,1);
                LookaheadArray = zeros(StochasticSims,1);

                for n = 1:StochasticSims              % expected value
simulation per particle
                                                              % performs
simulation and outputs difference to reference data
                    PriorSimdata =
PHD_OOAA_ModelSMCValidation(ReferenceData,SimTi,SimTiStp,SDEId,NoiseLvl,P);
                    WithinTolerance = min(PriorTolerance-PriorSimdata) >=
0;
```

```
                    AcceptedArray(n) = WithinTolerance;    % logic array of
values within tolerance
                    WithinNextTolerance = min(InterTolSchedule(1,:)-
PriorSimdata) >= 0;
                    LookaheadArray(n) = WithinNextTolerance;     % and
within next tolerance

                end
                clear n

                ProportionWithin = sum(AcceptedArray)/StochasticSims; %
proportion accepted
                ProportionWithinNext = sum(LookaheadArray)/StochasticSims;

            end

            SamplingResults(m,:)= [PriorSample, ProportionWithin,
ProportionWithinNext];

        end
        clear m

        ViableLookahead = sum(SamplingResults(:,end) ~= 0); % number of
positve lookahead values
        fprintf('\n');

    end
    % normalize the weight and lookahead weight

    WeightedPrior = SamplingResults;
    WeightedPrior(:,(end-1)) = SamplingResults(:,(end-
1))/sum(SamplingResults(:,(end-1)));
    WeightedPrior(:,end) =
SamplingResults(:,end)/sum(SamplingResults(:,end));

    % generate a covariance matrix for each particle, required for
perturbation
    % step when sampling the next intermediate population

    ListCovTotal = zeros(NumberVars,NumberVars,Particles);

    for b = 1:Particles

        CovParticle = WeightedPrior(b,1:NumberVars);  % select theta values
of particle b
        CovRest = WeightedPrior;
        CovRest(b,:) = [];          % drop particle b from list and re-
normalize
        CovRest(:,end) = CovRest(:,end)/sum(CovRest(:,end));

        ListCovInstance = zeros(NumberVars,NumberVars,(Particles-1));

        for c = 1:(Particles-1)   % calculate in turn covariance for
particles b and all c

            RestParticle = CovRest(c,1:NumberVars);    % select theta
values of particle c
            ParticleDiff = RestParticle-CovParticle;   % difference in
theta b and c
            CovInstance = CovRest(c,end)*(ParticleDiff'*ParticleDiff);
            ListCovInstance(:,:,c) = CovInstance;   % weighted covariance
matrixes

        end
        clear c
                       % sum covariance for all c into one matrix and
repeat for all b
        ListCovTotal(:,:,b) = sum(ListCovInstance,3);
    end
    clear b

    InterWeighted = WeightedPrior;  % rename lists for intermediate loop
below
    InterCovTotal = ListCovTotal;

    PriorFileName = ['Population Prior'];
    save(PriorFileName,'InterWeighted','InterCovTotal','SMCParam');

else

    InterStart = ReRun;

    if ReRun == 1

        ReRunSave = ['Population Prior'];

    else

        ReRunSave = ['Population ', num2str(ReRun-1)];

    end

    load(ReRunSave);

end

%% Starting the loop through all intermediate to posterior population
%--------------------------------------------------------------

%sampling particles from the previous intermediate, calculating their
expected
%values, proportion within tolerance, and adjusted weights

AverageParticles = zeros(Populations,NumberVars);
AverageSTD = zeros(Populations,NumberVars);

for t = InterStart:Populations

    fprintf('Population %d...\n', t);
    InterSResults = zeros(Particles,(NumberVars+2));
    InterViableLookahead = 0;

    while InterViableLookahead < 2        % repeat until there are at least 2
positive lookahead probabilities in the population
                                          % otherwise re-normalizing the
weight with one particle in turn excluded will produce NAN

        for o = 1:Particles

            if mod(o,10) == 0

                fprintf('%d',o);

            else

                fprintf('*');

            end

            InterProportionWithin = 0;

            while InterProportionWithin == 0     % satisfy that at least
one simulation is within tolerance

                PerturbedSample = 0;
                while any(PerturbedSample <= 0) || any(PerturbedSample >=
1)  % satisfy that 0<perturbed theta<1

                    [InterSample,CovIdx] =
datasample(InterWeighted,1,'Weights',InterWeighted(:,(end-1)));
                    InterSample = InterSample(:,1:NumberVars); % random
sample with weights from previous population
                    PerturbedSample =
mvnrnd(InterSample,InterCovTotal(:,:,CovIdx));
                        % perturb by random sampling from multivariate
distribution around selected particle
                end

                for v = 1:NumberVars

                P(MissingParams(v)) = PerturbedSample(v);

                end
                clear v

                AcceptedInterArray = zeros(StochasticSims,1);
                LookaheadInterArray = zeros(StochasticSims,1);

                for p = 1:StochasticSims              % expected value
simulation per particle
                                                             % performs
simulation and outputs difference to reference data
                    InterSimdata =
PHD_OOAA_ModelSMCValidation(ReferenceData,SimTi,SimTiStp,SDEId,NoiseLvl,P);
                    InterWthTolerance = min(InterTolSchedule(t,:)-
InterSimdata) >= 0;
                    AcceptedInterArray(p) = InterWthTolerance;      % logic
array of values within tolerance
                    InterWthNTolerance =  min(InterTolSchedule(t+1,:)-
InterSimdata) >= 0;
                    LookaheadInterArray(p) = InterWthNTolerance;     % and
within next tolerance

                end
                clear p

                InterProportionWithin =
sum(AcceptedInterArray)/StochasticSims; % proportion accepted
                InterProportionWithinNext =
sum(LookaheadInterArray)/StochasticSims;

            end

            PertPDFArray = zeros(Particles,1);

            for q = 1:Particles
                       % PDF of perturbed particle relative to all
particles of previous population
                PertRelativePDF = mvnpdf(PerturbedSample,
InterWeighted(q,1:NumberVars), InterCovTotal(:,:,q));
                PertPDFArray(q) = PertRelativePDF*InterWeighted(q,(end-1));
                       % adjusted by previous particle respective weights
            end
            clear q

            PertPDFTotal = sum(PertPDFArray);
            PriorPDF = prod(betapdf(PerturbedSample,2,2)); % PDF pf
perturbed particle relative to prior
            AdjustedWeight = InterProportionWithin*PriorPDF/PertPDFTotal;

            InterSResults(o,:) = [PerturbedSample, AdjustedWeight,
InterProportionWithinNext];
```

```
        end
        clear o

        InterViableLookahead = sum(InterSResults(:,end) ~= 0);

        if t == Populations

            InterViableLookahead = 2;

        end

        fprintf('\n');

    end

    % normalize the adjusted weight and lookahead weight

    InterWeighted = InterSResults;
    InterWeighted(:,(end-1)) = InterSResults(:,(end-
1))/sum(InterSResults(:,(end-1)));
    InterWeighted(:,end) = InterSResults(:,end)/sum(InterSResults(:,end));

    % generate a covariance matrix for each particle, required for
perturbation
    % step when sampling the next intermediate population

    InterCovTotal = zeros(NumberVars,NumberVars,Particles);

    for d = 1:Particles

        InterCovParticle = InterWeighted(d,1:NumberVars);  % select theta
values of particle d
        InterCovRest = InterWeighted;
        InterCovRest(d,:) = [];          % drop particle d from list and
re-normalize
        InterCovRest(:,end) = InterCovRest(:,end)/sum(InterCovRest(:,end));

        InterCovInstance = zeros(NumberVars,NumberVars,(Particles-1));

        for e = 1:(Particles-1)   % calculate in turn covariance for
particles d and all e

            RestInterParticle = InterCovRest(e,1:NumberVars);    % select
theta values of particle e
            ParticleIntDiff = RestInterParticle-InterCovParticle;   %
difference in theta d and e
            CovInstance =
InterCovRest(e,end)*(ParticleIntDiff'*ParticleIntDiff);
            InterCovInstance(:,:,e) = CovInstance;   % weighted covariance
matrixes

        end
        clear e
                  % sum covariance for all e into one matrix and repeat for
all d
        InterCovTotal(:,:,d) = sum(InterCovInstance,3);

    end
    clear d
                  % weigh and sum particles to generate a mean particle
    WeightedParticles = zeros(Particles,NumberVars);

    for w = 1:Particles

        WeightedParticles(w,:) =
InterWeighted(w,1:NumberVars)*InterWeighted(w,(end-1));
```

```
    end
    clear w

    AverageParticles(t,:) = sum(WeightedParticles);

    ColourString = 'bgrky';
                % plot accepted parameter values for each population
    for v = 1:NumberVars

        AverageSTD(t,v) = std(InterWeighted(:,v));  % determine std of
particles for next plot below

        subplot(Populations,NumberVars,((NumberVars*(t-1))+v));
        scatter(InterWeighted(:,v),InterWeighted(:,(end-
1)),10,ColourString(mod(v,5)+1));
        axis([0,1,0,0.1]);
        ylabel(['Parameter ', ParaNames{MissingParams(v)}]);
        xlabel(['Population ', num2str(t), ' - Tolerance Scaling is ',
num2str(InterTolScale(t))]);

    end
    clear v
                % save accepted parameter values and corresponding
covariance matrixes for each population
    if t == Populations

        InterFileName = ['Population Final'];

    else

        InterFileName = ['Population ', num2str(t)];

    end

    save(InterFileName,'InterWeighted','InterCovTotal','SMCParam');

end
clear t

figure
hold on
                % plot average values and stds for each population versus
expected values
for v = 1:NumberVars


errorbar([1:Populations],AverageParticles(:,v),AverageSTD(:,v),'Color',Colo
urString(mod(v,5)+1));
    plot([1, Populations], [EstimatedValues(v), EstimatedValues(v)],'--
','Color',ColourString(mod(v,5)+1));
    title('Average of Theta Particles changing over Populations');
    axis([1,Populations,0,1]);
    xlabel('Population');
    ylabel('Average Theta Value');

end
clear v

hold off

FinalValues = AverageParticles(end,:);
disp('    Average Theta values of each population');
disp(AverageParticles);

end
```