

Minerva Access is the Institutional Repository of The University of Melbourne

Author/s:

Forrest, Joshua Esli David

Title:

Towards whole-cell modelling with bond graphs

Date:

2024-07

Persistent Link:

<https://hdl.handle.net/11343/355593>

Terms and Conditions:

Terms and Conditions: Copyright in works deposited in Minerva Access is retained by the copyright owner. The work may not be altered without permission from the copyright owner. Readers may only download, print and save electronic copies of whole works for their own personal non-commercial use. Any use that exceeds these limits requires permission from the copyright owner. Attribution is essential when quoting or paraphrasing from these works.

Towards whole-cell modelling with bond graphs

Joshua Forrest

ORCID: 0000-0003-0767-5679 

Submitted in total fulfillment
of the requirements of the degree of
Doctor of Philosophy

Faculty of Science
School of Mathematics and Statistics
THE UNIVERSITY OF MELBOURNE

July 2024

Abstract

New mathematical and computational techniques are required to construct physically consistent and biologically plausible whole-cell models. Such models map genotype to phenotype and will be used to predict novel cellular behaviour and new laws of biology. The present challenge is how to feasibly build accurate and comprehensive whole-cell models.

Bond graphs are a proposed solution to this challenge. These are a physics-based modelling framework that implicitly enforce energy conservation laws, which enables modular model building across physical domains and scales. Bond graphs have already proven useful in systems biology applications, but more work is required before they can be used in whole-cell modelling. My research is on bridging the gaps in bond graphs towards this goal.

I first apply bond graphs to study Two Component Systems (TCS) in bacteria under different metabolic conditions. Bacteria sense and respond to the environment around them with TCS signalling cascades. They are a fundamental cellular subsystem in many well-studied bacteria and are thus ubiquitous in whole-cell models. Many different TCS variants exist with similar reaction structures, so they are naturally defined using a modular network. Using bond graphs, I generate physically realistic (energy conserving) mass-action equations and study how metabolic conditions affect signalling performance. Energy availability (via ATP concentration) was found to influence signal response strength, sensitivity, noise susceptibility, and signal bandwidth.

I then demonstrate new tools for symbolically generating bond graphs. To accelerate the modelling process, I designed and developed `BondGraphs.jl`, a new bond graph package for Julia. `BondGraphs.jl` is a high-level interface for a researcher to create, edit, plot, and simulate bond graphs. These bond graphs then symbolically derive differential equations which are solved within the SciML modelling ecosystem. The package is open-source and fully documented, and available through the Julia package registry.

Lastly, I expand the theory to incorporate stochasticity arising from thermal fluctuations and explore how bond graphs can be used to model probabilistic systems. I do this by extending the bond graph framework to work with a range of stochastic systems, including Brownian motion and chemical Langevin equations. Many systems in biology are sensitive to noisy processes, especially at the subcellular level. By modifying bond graphs so that they are compatible with stochastic processes, I broaden the scope of biological systems that can be modelled with physically realistic behaviour. I also address questions relating to stochastic energy conservation, entropy production, and probability in thermodynamics.

I conclude the thesis with a reflection on what gaps remain for whole-cell bond graphs and speculate on promising research directions and projects.

Declaration of Authorship

This is to certify that:

- this thesis comprises only my original work towards the PhD except where indicated in the preface;
- due acknowledgement has been made in the text to all other material used; and
- the thesis is fewer than 100,000 words in length, exclusive of tables, maps, bibliographies and appendices.

Joshua Forrest
July 2024

Preface

The following chapters contain material from published works that have resulted from the research presented in this thesis:

- Chapter 3 contains a slightly edited version of the preprint *Energy Dependence of Signalling Dynamics and Robustness in Bacterial Two Component Systems* [1] published on Feb. 12, 2023.
- Chapter 4 contains work published as an application note in *Bioinformatics*: “BondGraphs.Jl” [2] published on Sept. 19, 2023.
- Appendix C is original documentation for BondGraphs.jl [3].

This thesis was supported by the Melbourne Research Scholarship from the University of Melbourne. Additional financial support was provided by Michael Stumpf for the final writing stage.

Acknowledgements

I wish to express gratitude to all the people who have helped me on my PhD journey.

In particular, I wish to thank my supervisors, Prof. Michael Stumpf, Prof. Edmund Crampin, A/Prof. Vijay Rajagopal, and Dr. Michael Pan. In addition, I would like to thank the chair of my supervising committee Prof. Kim-Anh Lê Cao.

Thank you to Prof. Michael Stumpf, for the guidance, enthusiasm, insight, and stimulating discussions in my research.

Thank you to A/Prof. Vijay Rajagopal, for the encouragement, support, and insights, and for welcoming me into his research group amid difficult circumstances.

Thank you to Dr. Michael Pan, for the ongoing support and technical feedback and expertise, and generosity with time and advice.

I would also like to honour Prof. Edmund Crampin who is no longer with us, but whose vision and enthusiasm for this thesis inspired me from the start.

A special thanks to everyone in the research groups I have been a part of over these years: Theoretical Systems Biology, Melbourne Mathematical Biology, and Cell Systems and Mechanobiology Lab. There are more people to name than I can fit here.

Finally, I would like to thank my wife Jackie, for her enduring faith, patience, and support.

Contents

Abstract	i
Declaration of Authorship	ii
Preface	iii
Acknowledgements	iv
List of Figures	vii
List of Tables	ix
1 Introduction	1
1.1 The goal: whole-cell models	1
1.2 A proposed solution: Bond Graphs	3
1.3 Thesis structure	10
2 Bond Graph Theory	13
2.1 Bond Graphs	13
2.2 Definitions	16
2.3 Constructing bond graphs and deriving equations	19
2.4 Thermodynamics and bond graphs	21
3 Two Component Systems	30
3.1 Introduction	33
3.2 Results	37
3.3 Discussion	49
3.4 Methods	53
3.5 Supplementary Information	61
4 BondGraphs.jl	62
4.1 Introduction	64
4.2 Features	65
4.3 Results	69
4.4 Conclusion	70
4.5 Supplementary Information	71

5	Stochastic Bond Graphs	72
5.1	Background	72
5.2	Stochastic Theory	75
5.3	Motivating example: Brownian particle	77
5.4	New component: Stochastic resistor	81
5.5	Stochastic resistors with thermal ports	87
5.6	Applications of the Brownian bond graph	91
5.7	Stochastic chemical reactions	96
5.8	Concluding remarks	104
6	Conclusion	111
6.1	Summary of research	111
6.2	Future work	114
6.3	Bond graphs and whole-cell modelling	118
A	Two Component Systems Supplementary Information	120
B	BondGraphs.jl Supplementary Information	127
B.1	Bond graph theory	127
B.2	Model descriptions	130
C	BondGraphs.jl v0.4.0 Documentation	140
D	Extra Derivations	173
D.1	Langevin equation analytical solution	173
D.2	Non-equilibrium entropy production	175
D.3	Why the mean and variance of the Chemical Langevin equation are equal	176
	Bibliography	178

List of Figures

1.1	<i>M. genitalium</i> whole-cell model	2
1.2	Cardiac myocyte bond graph	7
2.1	Mass-spring-damper system and RLC circuit similarities	14
2.2	Mass-spring-damper bond graph	15
2.3	Simple word bond graph example	17
2.4	Bond graph construction steps	19
2.5	Simple biochemical bond graph	20
2.6	Thermodynamic energy domains	23
2.7	Free energy of bond graph components	24
2.8	Entropy of a mechanical spring	26
2.9	Entropy resistor	27
2.10	Electric circuit with entropy resistors	29
3.1	Two Component Systems diagram	34
3.2	HK binding sites for the unorthodox TCS	37
3.3	RRP fraction over time	38
3.4	Distribution of Hill coefficients	39
3.5	Temporal response curves	41
3.6	Performance metrics of TCS with noisy inputs	42
3.7	Reaction fluxes of TCS internal reactions	43
3.8	TCS signal response curves	44
3.9	Representative examples of the SR curve shape classes for the unorthodox TCS simulations	45
3.10	Representative examples of the SR peak-shift classes for the unorthodox TCS simulations	47
3.11	Sampled reaction rates used for the unorthodox TCSs numerical simulations	48
3.12	Bond graph representations of all TCS reactions and models	57
4.1	The BondGraphs.jl modelling process	68
5.1	Diagram of a Brownian particle in a thermal fluid	77
5.2	Brownian particle bond graph	79
5.3	Stochastic resistor bond graph	82
5.4	Brownian particle bond graph submodels	83
5.5	Stochastic resistor with alternative parameterisation	86
5.6	Brownian particle bond graph with a 2-port thermal stochastic resistor	88
5.7	Proposed extensions to the Brownian particle bond graph model	92
5.8	Stochastic resistor as a bridge between scales	95

5.9	Bond graph microstates and macrostates	96
5.10	Chemical Langevin Equation component	101
5.11	Chemical potential component with thermal port	103
5.12	Bond graph of $A \rightleftharpoons B$ reaction with thermal ports	104
A.1	Sampled thermodynamic constants K used for the unorthodox TCS numerical simulations	125
B.1	Bond graph model of a yeast metabolic network	139

List of Tables

1.1	Principles of whole-cell modelling	8
2.1	Mass-spring-damper system and RLC circuit equation similarities	15
2.2	Energy domains in physics	17
2.3	Component definitions	18
2.4	Thermodynamic bond graph domains	22
3.1	Proportions of SR Curve Classifications for Unorthodox TCS	45
3.2	Orthodox and Unorthodox TCS Chemical Reaction Systems	55
5.1	Comparing the chemical bond graph domain with a probability domain	108
A.1	Chemical reaction system of the orthodox TCS model	120
A.2	ODE model parameters for the orthodox TCS	121
A.3	Chemical reaction system of the unorthodox TCS model	122
A.4	ODE model parameters for the unorthodox TCS	122
A.5	Two Component Systems found in <i>E. coli</i>	126
B.1	Bond graph variables and analogous components for common physical do- mains	128
B.2	Internal and external concentrations and charges for Na^+ , K^+ , and Cl^-	130
B.3	Parameter values and initial conditions for the states used in the SERCA bond graph model	134

Chapter 1

Introduction

Whole-cell models are the pinnacle of systems biology. These are comprehensive, biologically complete mathematical models of an entire cell that map genotype to phenotype. Recent years have seen a rise in whole-cell modelling research and papers. The challenge now is finding a way to efficiently build whole-cell models that are comprehensive, accurate, reusable, and modular. Bond graphs fulfil many of the requirements of whole-cell modelling, but work needs to be done in other areas. Here I lay out where bond graphs are good for whole-cell models, and areas or challenges to overcome in this area. I then lay out the research plan for the thesis.

1.1 The goal: whole-cell models

Whole-cell models are the epitome of systems biology research. These are comprehensive, biologically complete mathematical models of an entire cell that map genotype to phenotype [4–8]. A complete whole-cell model will be able to predict the behaviour and function of a single-cell organism through computer simulations. It would give researchers the tools to: predict the effect of novel mutations; identify gaps in our biological knowledge; test antibacterial drug design; develop new biological hypotheses for experiments; and create synthetic biology circuits and designs [9].

Our understanding of biological purpose and gene function comes largely through experiments and bioinformatics. Understanding comes through correlating data and experiments and reverse-engineering the mechanisms that must exist in a cell that match expected cell behaviour. In contrast, a whole-cell model is a mechanistic model, built from first principles

and fundamental biological concepts. This provides an avenue for predicting new biology, for example, new genetic mutations in the wild or introduced through synthetic biology.

1.1.1 The development of whole-cell models

The past decade has seen a rise in whole-cell modelling research [10–15]. The earliest example is the *Mycoplasma genitalium* model developed by Karr *et al.* [4]. This was a major milestone in that it showed how such a model could exist in principle.

This whole-cell model (Figure 1.1) consisted of 28 submodels that described all known biological processes of the 525 genes in *M. genitalium*. In total, the model included over 1,900 parameters drawn from over 900 publications, synthesised together into one large composite model, with the goal of predicting cell division [4].

The diversity of models included is also its weakness. The mathematical models and parameters chosen were made under different assumptions and lab conditions, and are not automatically compatible with each other. Significant amounts of time were invested into tuning the parameters to achieve realistic outputs.

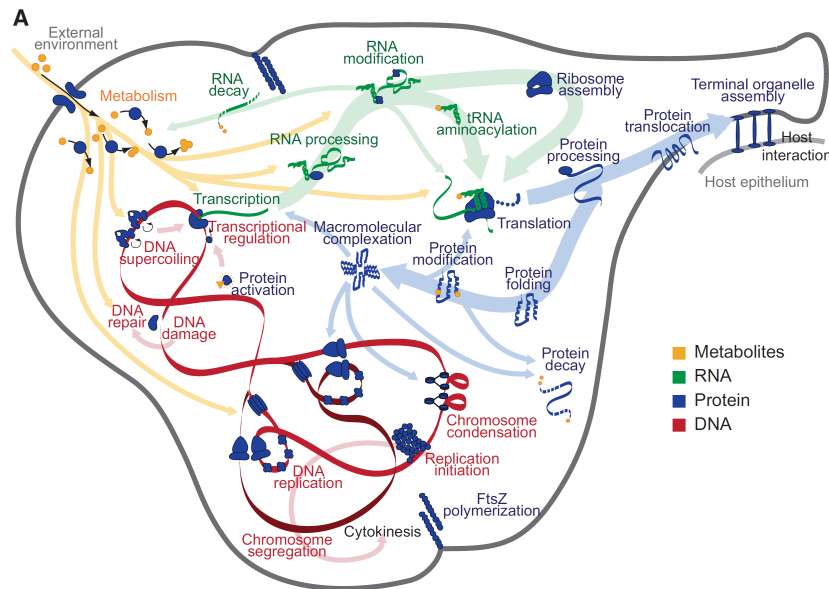


FIGURE 1.1: Schematic of *M. genitalium* whole-cell model. Reproduced from “A Whole-Cell Computational Model Predicts Phenotype from Genotype” [4] with permission.

Since 2020, several new whole-cell models that have made significant improvements. A model of *Escherichia coli* was released by Macklin *et al.* [5] in 2020 with 19,119 parameters and over 10,000 equations. The *E. coli* model is publicly accessible and executable in Python, and is continually being developed [7]. Recent work has even expanded to simulating colonies of *E. coli* [16]. Other recent whole-cell models include *Saccharomyces cerevisiae* [6], a 3D structural model of *M. genitalium* [8], and Syn3A, a virtual ‘minimal’ living cell [17].

1.1.2 The challenges of whole-cell modelling

Perhaps the biggest obstacle to overcome in whole-cell modelling is time. It can take a long time to develop a single whole-cell model, in terms of both years of development and research-hours. What is needed for the accelerated development of whole-cell models is a unifying mathematical and computational framework on which to build future projects.

Any such tool must align with the objectives of whole-cell modelling. Recent papers have outlined the objectives and challenges of the whole-cell modelling project [5, 11–14]. Karr *et al.* [11] published a list of minimum requirements for a whole-cell model. These principles are a manifesto for any future whole-cell model project. A realistic *in silico* whole-cell model must satisfy these principles.

The principles of whole-cell modelling

Summarised from Karr *et al.* [11].

Single-cell Individual, independent model of a cell

Functional closure Represent every known cellular and gene function within a cell

Molecular closure Every state variable and molecular pathway accounted for

Temporal closure The full life cycle of the cell, from birth to reproduction

Biophysical Laws of biophysics, biochemistry, and thermodynamics satisfied

Dynamics Differential equations describing cell-scale dynamics

Stochastic Stochastic and discrete equations, accounting for cell variation

Species-specific Parameterised by species-specific experimental data

Parsimonious Minimal complexity and as few parameters as possible

Modular Composable and hierarchical models built from reusable submodels

Reproducible Transparent, well-annotated, and accessible by other researchers

1.2 A proposed solution: Bond Graphs

If we are to create a successful whole-cell model, we will need an innovative mathematical modelling framework to accurately describe, predict, and design the behaviour of a cell. The

challenge now is finding a way to efficiently build whole-cell models that are comprehensive, accurate, and usable. There are many classes of models used in systems biology across scales and domains of physics. An ideal mathematical framework would be able to accurately incorporate all these model classes in any context.

Bond graphs are a powerful tool to construct multiscale, multiphysics, modular systems that automatically satisfy the laws of thermodynamics on a large scale [18–25]. They are a graph-based representation of a physical system that describes the energy flow between system compartments. Bond graphs implicitly enforce energy conservation laws, which enables modular model building across physical domains and scales. Recently, bond graphs have been proposed as a candidate modelling framework to achieve whole-cell modelling [23, 24].

A mathematical introduction of bond graphs will be given in [chapter 2](#). Here I will briefly go over why bond graphs may be useful for whole-cell modelling.

1.2.1 Comparison to other frameworks

The range of mathematical techniques used in whole-cell modelling are as diverse as the cells which they model. Machado *et al.* [26] provide detailed comparison of modelling formalisms used in systems biology. I highlight here a few pertinent examples:

- *Boolean networks* [27] represent gene reaction networks as binary matrices with boolean logic determining the cell state and state transitions. Although limited in complexity and predictive power, the simplicity of boolean networks means they can be trivially scaled to tens of thousands of state variables.
- *Constraint-based models* [28, 29] model the metabolic, signalling, or gene regulatory network of a cell as a steady state optimisation problem. The constraints are the stoichiometry of the network, and may include additional thermodynamic or enzyme capacity constraints.
- *Differential equations* [30, 31] model the continuous change of cell state over time. These may be extended to model spatial dynamics (partial differential equations) or stochastic effects (stochastic differential equations). They are best for capturing the full range of dynamics and states possible in a system, though this comes at a computational cost.
- *Agent-based models* [26] are a type of cell automata that simulate interacting ‘agents’ that follow simple, defined rules. These can be used to study emergent behaviours from populations of molecules, subcellular structures, or even whole-cells themselves. They can model arbitrary complexity with ‘black-box’ models, however the computational cost of simulating a granular system can be prohibitively expensive.

Bond graphs are a network representation of the (bio)physical equations of a system. At the point of simulation, bond graphs are effectively ordinary differential equations (ODEs) with implicit constraints on the parameters.

Bond graphs tend to have more variables and equations than similar models due to the network structure of the graph; each edge and vertex has at minimum two state variables (effort and flow). These superfluous variables disappear when the equations are symbolically derived, however the symbolic computation cost to derive the equations is a potential bottleneck here for scaling up to larger, more complex systems. With a correctly implemented modular approach, however, components can be efficiently constructed [23]. The additional variables are a trade-off for the flexibility, consistency, and interoperability that bond graphs provide. The extra junction nodes and equations, for example, allow for easier connection and modularity with other bond graph components. Likewise, a modular bond graph can represent a repeated component or motif in a hierarchical model with less repetition compared to a naïve ODE implementation.

Because bond graphs are ODEs, existing standard techniques for parameter fitting (e.g. optimisation, Bayesian statistics) can be used [32, 33]. Bond graphs may also have additional parameters compared to equivalent ODEs. This is because bond graphs tend to have fewer lumped kinetic parameters, preferring instead to split kinetic constants into multiple material or physical parameters (see for example the system of equations in [subsection 2.3.2](#)).

Using physical parameters (instead of lumped parameters) makes it easier to source data from other databases or experiments. In the context of metabolic networks, the kinetic catalytic reaction rate k_{cat} and Michaelis-Menten constant K_M are two commonly required constants for accurate models. While these are available on public databases (for example, BRENDA [34] and SABIO-RK [35]), thermodynamic inconsistencies may exist between reported parameters. Thermodynamic parameters are based on standard Gibbs free energies, which can in some cases be estimated from the chemical structure [36]. Additionally, if the number of reactions is less than the number of metabolites, then the parameter search space is less [37].

While bond graphs may have more variables and equations than other modelling frameworks, it can often be a simpler framework compared to other modelling frameworks with the same physical constraints imposed. Take for example Flux Balance Analysis (FBA), which models an entire metabolic chemical reaction network through a linear program optimisation problem at steady state [28]. One major limitation of FBA is that reactions do not account for thermodynamic constraints and are not reversible by default. Thermodynamic Flux Balance Analysis (TFBA) was developed to address this problem [29]. TFBA adds extra constraints to the model to ensure thermodynamic feasibility, however, this requires an additional 3 variables and 3 constraints per reaction. A model with hundreds of variables and constraints can easily jump

into the thousands. The problem space is now much bigger than the problem space of bond graphs, which already account for thermodynamic constraints.

1.2.2 From Bond Graphs to whole-cell Modelling

Bond graphs are well positioned to be the modelling framework of choice for whole-cell modelling.

Several standards, tools, and frameworks have arisen in recent decades to unify systems biology. A selection of these include: systems biology modelling standards, SBML [38], CellML [39], and SED-ML [40]; semantic model composition, SemGen [41] and ModelingToolkit.jl [42]; optimisation toolboxes [43]; computational frameworks, OpenCOR [44], COPASI [45], Cell Physioiome [46] and, the whole-cell modelling specific Vivarium [47].

The existing tools and frameworks listed above are comprehensive, and are primarily focussed on the computational elements of whole-cell modelling, i.e. database storage, annotations of variables with biological metadata, automated model classification and composition, and numerical simulation and analysis. What is generally missing is a focus on thermodynamics and adherence to the laws of physics, regardless of model depth and complexity. This is where bond graphs come in.

They have already proven useful in systems biology applications [25]. For example, bond graphs have been used to model reaction networks with electrochemical potentials [33], MAPK signalling cascades with modular components [23], gene regulatory networks [48], and have even been scaled up to model a full-body circulation system [49].

Figure 1.2 shows a good example of a large bond graph single-cell model. This is a physiological model of a cardiac myocyte cell [50], which was later converted into a bond graph [44]. Notice how the pictorial diagram and the bond graph model in Figure 1.2 are visually similar. This visual aspect of bond graphs reinforces model interpretability and modularity. Our aim is to expand this approach up to the scale of a whole-cell model.

If Bond Graphs are to become the core mathematical framework for large scale systems biology challenges such as whole-cell modelling, there are several modelling obstacles or hurdles they will need to overcome. Bond graphs can satisfy some principles of whole-cell modelling already (subsection 1.1.1), but not all principles are satisfied (see Table 1.1). More work is required before they can be used in whole-cell modelling.

Bond graphs satisfy some of the above criteria already. I outline below how bond graphs stand up to the principles of whole-cell modelling (subsection 1.1.1).

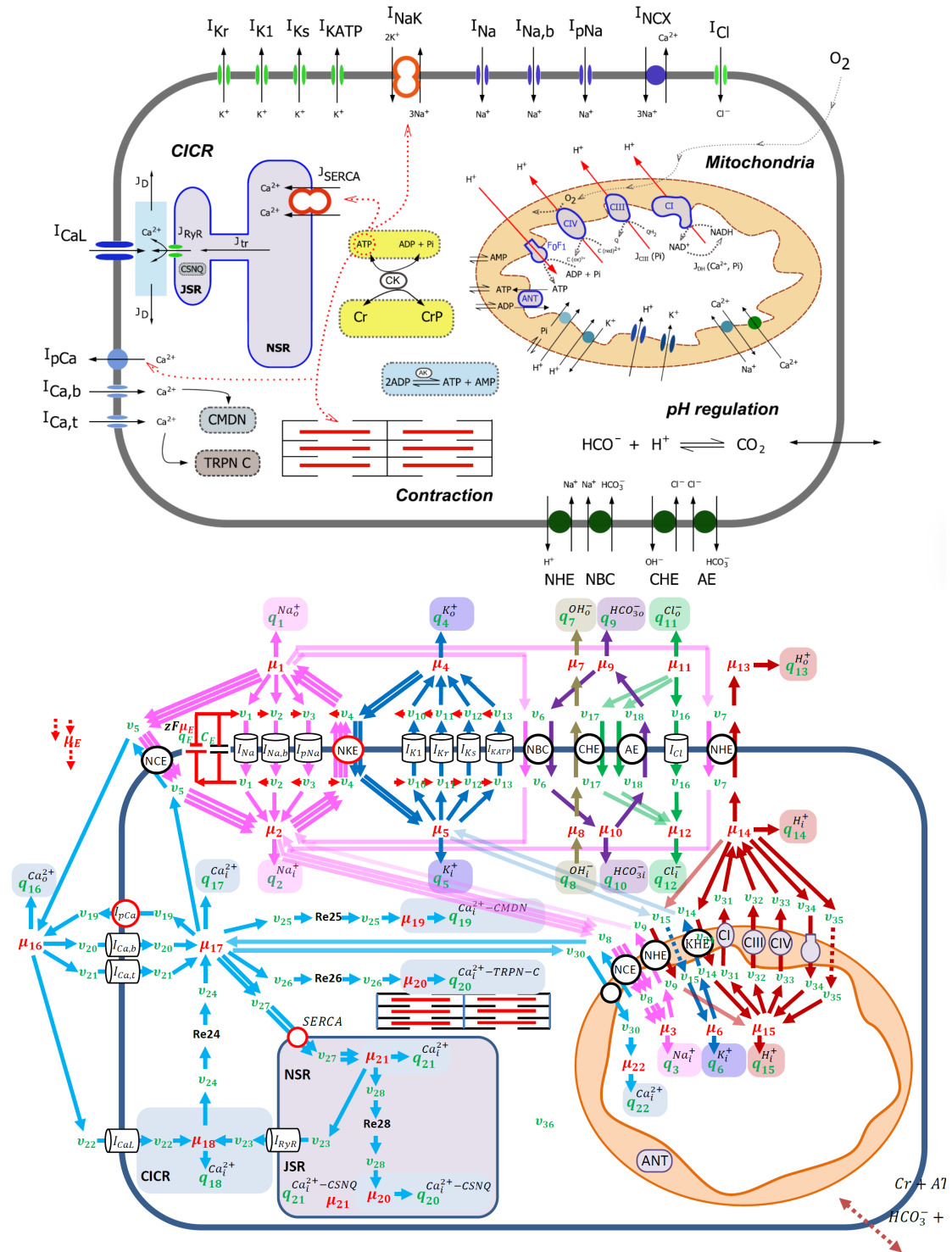


FIGURE 1.2: (Top) Cardiac myocyte cell model Tran et al. [50]; (Bottom) Bond graph schematic of the same cardiac myocyte cell Hunter & Safaei [44]. Each colour on the bond graph is a different subsystem (e.g. reaction pathway, or physical domain). Figures are openly licensed via <https://creativecommons.org/licenses/by/4.0/>.

	BN	CB	AB	DE	BG
Single-cell	✓	✓✓	✓✓	-	✓
Functional closure	-	-	-	-	-
Molecular closure	-	-	-	✓	✓✓
Temporal closure	-	-	✓✓	-	-
Biophysical	-	-	-	✓	✓✓
Dynamics	-	-	✓	✓✓	✓✓
Stochastic	-	-	✓	✓✓	-
Species-specific	-	-	✓	-	✓
Parsimonious	✓✓	✓	-	-	-
Modular	-	-	✓	-	✓✓
Reproducible	✓	✓	-	✓	✓

TABLE 1.1: Comparison of modelling frameworks in systems biology against the principles of whole-cell modelling [11]. Bond graphs satisfy many of the requirements already, but more work is needed in other areas. (BN) Boolean networks; (CB) Constraint Based models; (AB) Agent Based models; (DE) Differential Equations; (BG) Bond Graphs; (✓✓) does well; (✓) = possible in theory.

Single-cell Bond graphs can be built up to the scale of whole-cell models built from reusable modules of subcellular systems (cf. Figure 1.2). Biochemical bond graph models can be parameterised and adjusted for a particular biological species, and account for variation within a population. However, a bond graph on the scale of a whole-cell model yet to be attempted.

Functional closure A whole-cell model will need to include every known cellular and gene function within a cell. While bond graphs are capable of modelling a wide range of physical phenomena and cellular subsystems, there are certain categories of biological and biophysical phenomena that are yet to be modelled. Namely, stochastic reactions, spatial dynamics, and complex genetic logic networks [27].

Molecular closure Bond graphs by design enforce physical conservation laws. All energy exchanges within a cellular system must be explicitly represented, including mass exchanges with the environment, heat loss, electrochemical potentials, or any other physical law. Sources and sinks must be explicitly defined, and can be connected to other models through external ports.

Temporal closure A whole-cell model should account for the full lifecycle of the cell, from early development to cell reproduction (or death). More complex cells will need to somehow account for cell fates, the specialisation into a particular function in a multicellular organism.

Bond graphs at this stage do not adapt to a changing model structure. While they can model the dynamics of a *state* within a system, they are relatively poor at modelling a change to

the system *structure* over time. A structural change could appear as new molecules in the model (new state variables), or new reaction pathways forming (new equations). In bond graph terms, this would result in changes to the graph structure itself, e.g. new vertices and edges appearing over time.

Biophysical By construction, bond graphs enforce physical laws such as conservation of energy or mass. Bond graphs excel at modelling biochemical and biophysical models. Their physics abstraction allows them to model cross-physics systems, such as electro-physiological or mechano-chemical systems in biology, while still satisfying the laws of thermodynamics.

Dynamics Bond graphs define an energy function based on Hamiltonian (or Lagrangian) mechanics. A Hamiltonian function $\mathcal{H}(\mathbf{p}, \mathbf{q}, t)$ defines the movement of energy within a system between kinetic energy variables $\mathbf{p}$ and potential energy variables $\mathbf{q}$ over time t . The bonds and components of bond graphs are defined by efforts $\mathbf{e} := \dot{\mathbf{p}}$ and flows $\mathbf{f} := \dot{\mathbf{q}}$, which are derived from these variables.

Stochastic Stochastic dynamics play an important rule in cell biology [51], yet stochastic reaction networks have not been represented by bond graphs. Existing approaches to stochastic modelling in bond graphs (e.g. [52]) have done so with stochastic controls or parameters, e.g. a white noise input. There is currently no theoretical and physically consistent basis for stochastic differential equations within bond graphs.

Species-specific Parameterising large models is a significant challenge in systems biology [14], and bond graphs are no exception. Even with the ideal model, there is still the difficulty of fitting experimental data to whole-cell models.

Bond graphs provide some advantages here. Firstly, parameters are based on fundamental physical laws and principles, which can be understood independently of one particular model (as opposed to, say, conventional kinetic models [33]). For chemical reactions, bond graphs use a thermodynamic parameterisation. This separates the molecular species specific parameters (e.g. standard chemical potential) from the system specific parameters (e.g. reaction rates). Thermodynamic parameters also automatically satisfy detailed balance constraints in biochemical systems [21]. In theory, species specific parameters can be used in any model (assuming similar conditions).

Parsimonious One of the downsides of bond graphs is that they can have a larger number of internal state variables and parameters. Each bond has an associated effort and flow,

so each additional bond increases the state space by 2. In practice, a bond graphs are symbolically simplified to produce differential equations that have the same number of states as conventional models.

On the other hand, there are algorithms and techniques that can simplify bond graphs automatically. For example, by reducing or removing subgraphs based on the observed energy activity [19].

Modular Bond graphs are modular and hierarchical by design. They are built using a simple graph network of components and junctions (vertices) and bonds (edges). Combining models is as simple as a series of graph operations. This is a powerful feature for building large scale models that remain physically consistent.

Hierarchical models can be constructed by nesting bond graphs within each other. A complete bond graph with external ports can be encapsulated within a single component (vertex), which is itself repeated and connected within a higher-level bond graph.

Reproducible The Physiome model repository [53] holds a large collection of physiological model, including bond graphs. A more recent initiative that also features bond graphs is the Cell Physiome Project [46].

Some work has been done in making bond graphs more accessible and reproducible. An example is BondGraphTools [54], an open-source bond graph package in Python. Bond graph modelling standards have been proposed, such as BGML [55], but none have become universal.

1.3 Thesis structure

The overarching goal is to see if an energy-based biophysical approach from ‘first principles’ is a viable approach to whole-cell modelling.

The goal of my research is to work towards bridging the gap between energy modelling with bond graphs and whole-cell modelling. Specifically, how to develop bond graphs which are (1) *functionally closed*, (2) *reproducible*, and (3) *stochastic* (cf. Table 1.1). To this end, I have focused on the following objectives.

- (1) Model the effects of noise and energy dependence on a bacterial signalling system
- (2) Develop better computational tools for bond graphs for systems biologists

(3) Extending bond graphs to model stochastic biological systems

My thesis chapters and content are summarised below.

Bond Graph Theory

A generalised energy modelling framework

Bond graphs are the core modelling framework for this research. This chapter introduces the necessary theory to bond graphs, as well as other topics from mathematics, physics, biochemistry, and thermodynamics required to understand the rest of the thesis.

Two Component Systems

Simulation study of a bacterial signalling mechanism

The first project I undertook was to apply bond graphs to a common bacterial system: Two Component Systems (TCS). TCS are the signalling system found in many bacteria, including the model organism *E. coli*. I model a fundamental cellular subsystem of bacteria using the bond graph framework. Two Component Systems (TCS) are ubiquitous in many common bacteria, with different varieties working together in the same cell. A modular network naturally describes these signalling networks.

Using bond graphs, I generate physically realistic (i.e. energy conserving) mass-action equations and study how energy availability, by way of ATP concentrations, affect signalling performance. By modelling TCS with bond graphs, I introduce a novel energy based approach to TCS models.

This work is available as a preprint on bioRxiv: *Energy Dependence of Signalling Dynamics and Robustness in Bacterial Two Component Systems* [1]. It is currently under review at PLOS Computational Biology.

BondGraphs.jl

A Julia package for bond graphs

We also need to consider the practical or pragmatic side of building a computational cell. In particular, we need to build better computational tools for constructing, modifying, saving, reusing, and extending bond graphs.

For larger and more complex problems, we will need software tools that work at scale, and are easy to use by modellers. There should also be an easy way to export or interface the model with other languages or APIs. Furthermore, the tool should be open source and easily accessible, to support collaboration in this project.

To accelerate the modelling process for bond graphs, I developed a new package in Julia that is integrated in the SciML modelling ecosystem. `BondGraphs.jl` [2] is a high-level interface for a modeller to create, edit, and plot bond graphs using in native Julia. It is built upon `ModelingToolkit.jl` which gives it access to popular modelling packages including `Catalyst.jl`, `Plots.jl`, `DifferentialEquations.jl`, and `SBMLToolkit.jl`.

Stochastic Bond Graphs

Developing the theory to model stochasticity with bond graphs

Finally, I extend the bond graph framework to work with stochasticity.

Many systems in biology are sensitive to noisy processes, especially at the subcellular level. It is necessary to model noise within systems biology for the construction of a whole-cell model.

By modifying bond graphs so that they are compatible with stochastic processes, I broaden the scope of biological systems that can be modelled and construct models with more physically realistic behaviour.

This chapter begins with why noise is significant to model, and the variety of ways that is done in systems biology. The key innovation here is the creation of stochastic bond graphs. These are a way of capturing stochastic effects (i.e. representing SDEs) in the bond graph formalism. I start with the simplest example of a stochastic Brownian particle motion within a fluid. From here, I extend the new stochastic component to (continuous) reaction systems. This section also addresses some potential issues associated with introducing noise to bond graphs. Lastly, I show how one may extend to other probabilistic models, including the Chemical Langevin Equation.

Conclusion

I summarise the research in each chapter, including a retrospective analysis of the Two Component System and `BondGraphs.jl` papers.

I discuss some remaining challenges regarding bond graphs and whole-cell modelling, and suggest some future directions of research to pursue.

Chapter 2

Bond Graph Theory

In this chapter, I summarise the theory necessary to understand bond graphs and their applications in systems biology. I first provide an intuition for energy-based modelling, which motivates bond graph modelling. I then define key bond graph terminology, give examples of simple bond graphs, and how to derive systems of equation from a bond graph. Lastly, I explain thermodynamic theory which underpins bond graphs.

2.1 Bond Graphs

Bond graphs are fundamental to my research, so in this chapter I will summarise the necessary theory to understand bond graphs and how they can be used in systems biology. For more complete introductions on the topic, see [19–21, 24, 56].

They were originally introduced by Henry Paynter to model hydroelectric systems [57], and are still applied in control engineering [20, 22, 58]. Oster *et al.* [18] gives the first example of a chemical and thermodynamic bond graph, where they were named *network thermodynamics* [18, 59, 60]. but in recent years have been applied to systems biology in a variety of contexts [21–25, 54].

2.1.1 Intuition: energy-based modelling

I will first define energy-based modelling which underpins bond graphs. This will build an intuitive picture of where bond graphs are useful and how they are a natural extension of other energy-based frameworks.

It has been observed since the 19th century the physical analogies between different domains of physics [20]. By identifying these analogies, we can abstract the underlying mathematical principles and transfer ideas, definitions, and theorems from one domain of physics to another. This then allows us to describe multiphysics systems with the same unified mathematical tools.

Consider the two systems in Figure 2.1. The first system (left) is a mechanical spring-damper-mass system attached to a fixed wall. A force F is applied, pulling the mass m downwards with displacement x . The forces of the spring, damper, and mass all balance. The second system (right) is a resistor-inductor-capacitor electric circuit (also called an RLC circuit). The battery supplies a voltage V which is evenly distributed across the three electric components. The voltage induces a current, which for our purposes is defined as a change in charge over time $\dot{q}$.

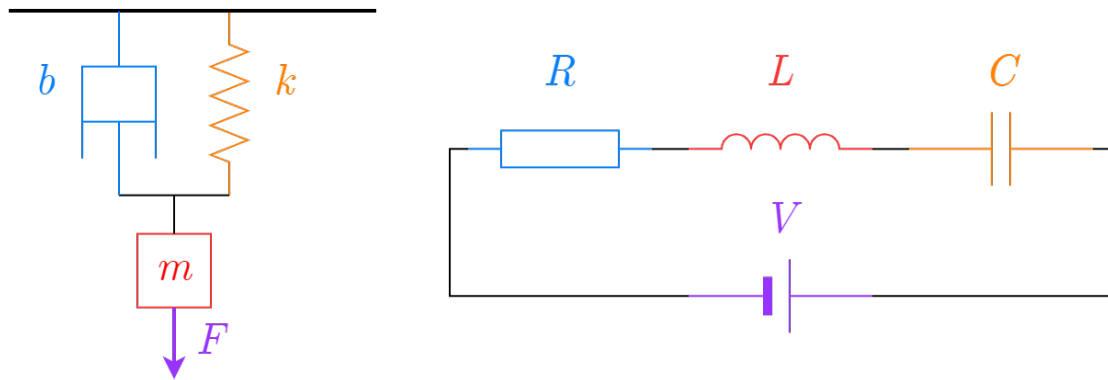


FIGURE 2.1: Two physical systems described by the same differential equation. **Left** Mass-spring-damper mechanical system. **Right** RLC electric circuit. Similar components are colour-coded to highlight the similarities.

The interesting observation is that these two different systems can be described by the same differential equations. In fact, there is a one-to-one correspondence between the components of the two systems, (shown in colour, Figure 2.1). Table 2.1 highlights this parallel in equation structure. Each component corresponds to a term in a differential equation.

$$\begin{array}{ll}
 \text{Mechanical} & m\ddot{x} + b\dot{x} + kx = F(t) \\
 \text{Electrical} & L\ddot{q} + R\dot{q} + \frac{1}{C}q = V(t)
 \end{array} \tag{2.1}$$

Notice also that the conservation laws in Equation 2.1 are analogous: Force balance $\sum F = 0$ and Kirchhoff's Voltage Law $\sum V = 0$. This is another property can be abstracted.

Since the two systems have the same mathematical structure, we can abstract the components and conservation laws to create a general modelling framework.

Force equation		Voltage equation	
Mass	$m\ddot{x}$	Inductor	$L\ddot{q}$
Spring	kx	Capacitor	$\frac{1}{C}q$
Damper	$b\dot{x}$	Resistor	$R\dot{q}$
External	$F(t)$	Battery	$V(t)$

TABLE 2.1: Similarities between the mass-spring-damper system and the RLC circuit. Note both the quantitative (equation) and qualitative (component) parallels. The RLC equations are written in terms of charge q instead of current $i = \dot{q}$.

Both physical systems are described by the same **bond graph**. Figure 2.2 shows the bond graph for the mass-spring-damper system, which is equivalent to the RLC circuit bond graph. The RLC bond graph is found by making the substitutions from Table 2.1. Crucially, only the variable labels change (e.g. $x \mapsto q$); the bond graph structure and component types do not change.

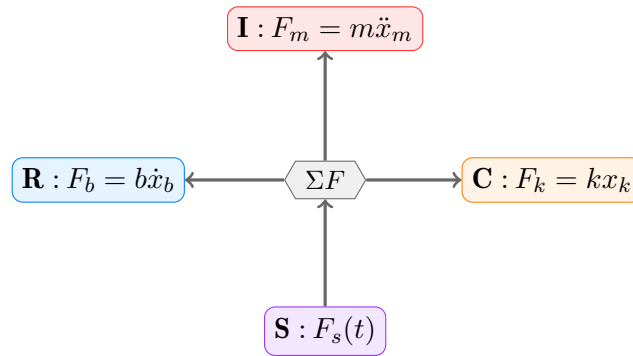


FIGURE 2.2: Bond graph of both a mass-spring-damper mechanical system. The equivalent RLC circuit bond graph is found by making the substitutions from Table 2.1.

Bond graphs abstract three essential elements of physical systems modelling: constitutive relations, conservation laws, and network topology.

Constitutive relations

The fundamental physical laws that govern the dynamics of the state variables. Constitutive relations are mathematical laws specific to the physical or material properties of the object. These are the physical objects and phenomena (spring, resistor, battery, etc.) in Figure 2.1, and represented in bond graphs as *components* in Figure 2.2.

Conservation laws

The balance equations that ensure compliance with the laws of physics and thermodynamics. These are algebraic relations which hold for all time and represent a conserved quantity, such as mass or momentum. In Figure 2.1 this is the force balance for the mechanical

example, and Kirchhoff's Voltage Law for the electrical example. In bond graphs, these are *junctions* which connect to components in Figure 2.2.

Network topology

The particular layout of all the components in relation to each other, much like the circuit diagram in Figure 2.1. The bond graph topology is the set of graph edges or *bonds* between component ports.

In section 2.2 I give precise definitions and terminology for these concepts.

2.2 Definitions

Common bond graph definitions and terminology used in this thesis. For a systematic introduction and an expanded list of definitions, see Borutzky [19], Tabatabaian [56], and Karnopp *et al.* [58].

2.2.1 Effort, flow, momentum, and state

The branches of physics and engineering are often categorised by a primary **energy domain** or discipline, e.g. electrical, (linear) mechanical, chemical, hydraulic etc. Each domain will have its own set of variables, definitions, components, conventions, and mathematical models.

A domain is usually defined by a single unit $[x]$ that represents a particular state of the system. Then we can define two energy variables: **effort** $e(t)$ with units 'energy per x' [Joules/ x], and **flow** $f(t)$ with units 'x per second' [x /second].

Efforts are *equilibrium determining* – they determine the total energy available in the system per x . Flows are *equilibrium establishing* – they determine how quickly the state changes.

Effort and flow together define energy transfer or **power** P [J/s].

$$P(t) \equiv \langle e, f \rangle = \int e(t) \cdot f(t) dt \quad (2.2)$$

Note that if we have time series data e_i and f_i , such as from a simulation, then we have simply $P_i = e_i \cdot f_i$.

We then integrate effort and flow to define two state variables: (generalised) **momentum** $p(t) \equiv \int e(t)dt$ with units [Js/ x], and (generalised) **state** or displacement $q(t) \equiv \int f(t)dt$ with units [x].

The definitions for p and q come from Hamiltonian mechanics, where the total energy of the system is given by the **Hamiltonian** energy function. Let $H(p, q, t) = T(p) + V(q)$ be the total energy, with kinetic energy $T(p)$ and potential energy $V(q)$. Then we have the following definitions

$$e = \frac{dp}{dt} = -\frac{\partial H}{\partial q}, \quad f = \frac{dq}{dt} = \frac{\partial H}{\partial p} \quad (2.3)$$

where e, f, p, q are defined as above. Note that this also defines power $\dot{H} = P$, as above.

Effort, flow, momentum, and state generalise across the domains of physics. Table 2.2 shows a selection of examples.

	Mechanical		Electrical		Chemical	
$[x]$	metres	$[m]$	Coulombs	$[C]$	moles	$[mol]$
e	Force	$[J/m]$	Voltage	$[J/C]$	Chemical potential	$[J/mol]$
f	Velocity	$[m/s]$	Current	$[C/s]$	Molar flow rate	$[mol/s]$
p	Momentum	$[Jm/s]$	Magnetic flux	$[Jmol/s]$	—	—
q	Displacement	$[m]$	Charge	$[C]$	Concentration	$[mol]$

TABLE 2.2: Effort, flow, momentum, and state generalise across the domains of physics. Units are given in terms of energy (Joules) and time (seconds). The chemical domain does not have a canonical definition for generalised momentum.

2.2.2 Elements of bond graphs

A **bond graph** is a directed graph of *components* and *junctions* connected with *bonds*. Components and junctions are vertices in the graph, and bonds are the edges connecting component ports.

Components are a single vertex in a bond graph with one or more of constitutive relations between an effort and flow variables. Junctions represent conservation laws in the system. Table 2.3 list a summary of common component and junctions types and some use cases. See [19] for a more comprehensive list.

Bonds transfer power between vertices in a bond graph. Figure 2.3 shows a simple ‘word bond graph’ connecting a mass to a spring. Note the use of a full arrow instead of the usual half-arrow used for the bond.

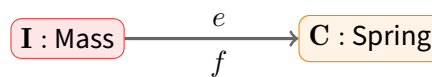


FIGURE 2.3: Bond graphs can be written as a ‘word bond graph’. Note the use of a full arrow instead of the usual half-arrow used for the bond.

Components are the basic model building blocks of a mechanistic model. Traditionally these had a one-to-one correspondence with a physical object, e.g. a resistor, capacitor, spring,

Symbol	Description	Equation	Examples
C : C	potential energy storage	$e = \phi_C(q)$	spring, capacitor, chemical species
I : I	kinetic (or dynamic) energy storage	$f = \phi_I(p)$	mass, inductor
R : R	resistive or energy dissipation	$e = \phi_R(f)$	damper, resistor, reaction
S : Se	effort sources	$e = E(t)$	battery, chemostat
S : Sf	flow sources	$f = F(t)$	current supply
TF	transformers	$ne_1 = e_2$ $f_1 = nf_2$	lever, transformer
GY	gyrators	$re_1 = f_2$ $f_1 = re_2$	DC motor
0	Flux balance	$\sum_i f_i = 0$	Kirchhoff's Current Law, mass balance
1	Energy balance	$\sum_i e_i = 0$	Kirchhoff's Voltage Law, force balance

TABLE 2.3: Basic components and junctions in bond graphs. Components contain the constitutive relations of a system, whereas junctions represent the conservation laws. Each of these components may also be modulated by with a control parameter, which is shown by a prefix, e.g. **MR**, **MSe** [19].

transformer, or power source. However, they can represent intangible effects (e.g. gravitational fields), or abstractions of a physical process (e.g. chemical reactions).

Each component type has a different form of *constitutive relation* between an effort and flow, or one of their derivatives or integrals.

The analogies between domains go deeper than the mathematical expressions. There is both a *quantitative* similarity in the mathematical expressions, and a *qualitative* similarity in the function of each component: masses and inductors have inertia and store ‘kinetic’ energy; springs and capacitors store potential energy; dampers and resistors dissipate energy through heat loss; external forces and batteries are power sources (or sinks) that set the energy equilibrium.

A component will have one or more *ports* that expose an internal effort/flow pair to another component. Ports are connection terminals for a bond to connect to a component. In bond graphs, ports are always bidirectional, meaning they can be both inputs and outputs.

A component has ports equal to the number of conjugate pairs of energy variables (equivalently, the number of constitutive relations), each port corresponding to a different effort/flow pair (e_i, f_i) . A component with n ports will have therefore $2n$ internal energy variables.

Ports can optionally have extra restrictions to guide model building. For example, a port may have defined energy domains (e.g. a port for thermal energy), or defined unit magnitudes (e.g. 10×10^{-6} mol), so that bonds only connect components of the same ‘type’.

2.3 Constructing bond graphs and deriving equations

2.3.1 Example: Mass-spring-damper system

I will continue with the example in Figure 2.1 to show how to build a bond graph systematically. Note that these steps are primarily for illustrative purposes. There are formalised standard techniques and algorithms to construct bond graphs (See e.g. [56]).

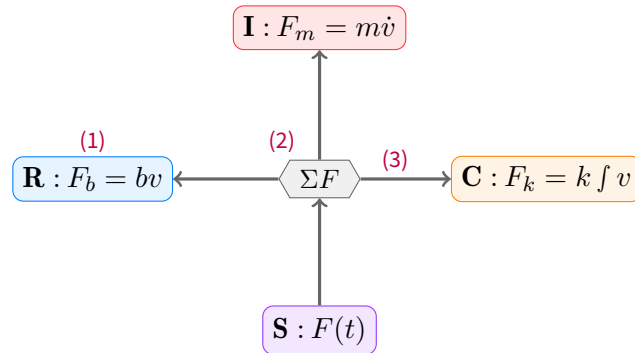


FIGURE 2.4: Steps to build a bond graph for the systems in Figure 2.1: **(1)** Determine the components and constitutive relations; **(2)** Determine the conservation laws and junctions; **(3)** Connect bonds between junctions and available component ports.

Using Figure 2.1 as an example, one can construct the bond graph in Figure 2.4 in the following order:

- (1) Determine the components and constitutive relations. These will relate a force F to a differential/integral term with respect to velocity v . In this case, there is one of each bond graph component with linear constitutive relations (Table 2.1)

$$\mathbf{I} : F_m = \phi_I(\dot{v}) = m\dot{v}$$

$$\mathbf{C} : F_k = \phi_C(\int v) = -k \int v$$

$$\mathbf{R} : F_b = \phi_R(v) = -bv$$

$$\mathbf{S} : F_s = F(t)$$

- (2) Determine the conservation laws and junctions. Here this is simply force balance, so a **1**-junction is used.
- (3) Connect the vertices with bonds according to the model topology. Bonds can only connect to a vertex if there is a free port. Each of these components has one port, so all simply connect to the **1**-junction.

We can generate a second order ODE from Figure 2.4. Starting with the **1**-junction, we write the force balance equation $\sum_{\text{out}} F = \sum_{\text{in}} F$

$$\begin{aligned} F_m + F_k + F_b &= F_s \\ m\dot{v} + k \int v dt + bv &= F(t) \\ m\ddot{x} + b\dot{x} + kx &= F(t) \end{aligned} \quad (2.4)$$

2.3.2 Example: Simple biochemical reaction

A key feature of bond graphs in systems biology is in modelling chemical reaction networks [18, 21]. Much of the later chapters will revolve around biochemical reaction bond graphs. For more examples of biochemical bond graphs in greater detail, see [21–24].

One of the added difficulties is that the constitutive relations are nonlinear in general. The chemical bond graph components are:

- Ce** Each chemical species i stores chemical energy defined as $\mu_i = RT \ln(K_i x_i)$ with concentration x_i , thermodynamic constant K_i , universal gas constant R , and temperature T [61]. The change in concentration is determined by the flow rate $\dot{x}_i = \nu$
- Re** Chemical reaction with molar flow rate $\nu_1 = \kappa \left[\exp\left(\frac{\mu_1}{RT}\right) - \exp\left(\frac{\mu_2}{RT}\right) \right] = -\nu_2$ [62]. Note that this is a two-port with variables $(\mu_1, \nu_1), (\mu_2, \nu_2)$.
- TF : 2** Stoichiometric coefficients are represented as transformers. A coefficient of 2 C means double the chemical potential on the products side ($\mu_C \rightarrow 2\mu_C$), but twice as much molar flow is needed ($\nu \rightarrow \frac{1}{2}\nu$).

Figure 2.5 shows a bond graph for the chemical reaction $A + B \rightleftharpoons 2C$. We will walk through this bond graph as an illustrative example.

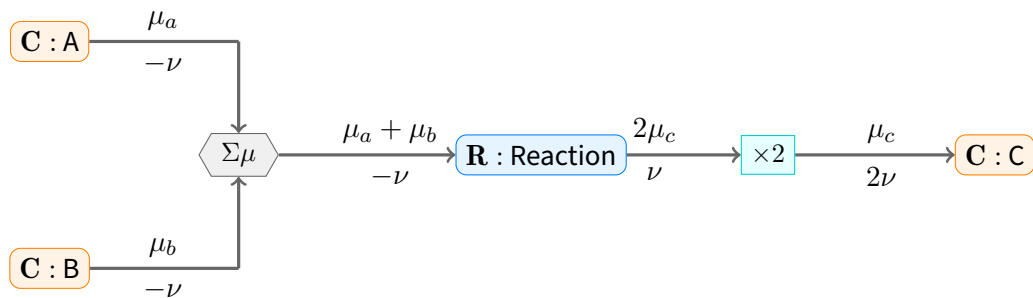


FIGURE 2.5: Biochemical bond graph for the reaction $A + B \rightleftharpoons 2C$.

Figure 2.5 is constructed using the same procedure:

- (1) Determine the components and constitutive relations.

$$\mathbf{Ce:A} \quad \mu_a = RT \ln(K_a x_a), \nu_a = \dot{x}_a$$

$$\mathbf{Ce:B} \quad \mu_b = RT \ln(K_b x_b), \nu_b = \dot{x}_b$$

$$\mathbf{Ce:C} \quad \mu_c = RT \ln(K_c x_c), \nu_c = \dot{x}_c$$

$$\mathbf{Re} \quad \nu_1 = \kappa \left[\exp\left(\frac{\mu_1}{RT}\right) - \exp\left(\frac{\mu_2}{RT}\right) \right] = -\nu_2.$$

$$\mathbf{TF:2} \quad \mu_2 = 2\mu_c, 2\nu_2 = \nu_c.$$

- (2) Determine the conservation laws and junctions. The chemical potential of the reactants is the sum of the chemical potentials, i.e. $\mu_1 = \mu_a + \mu_b$.
- (3) Connect the vertices with bonds according to the model topology: $A + B \rightleftharpoons 2C$.

We can then generate mass-action equations for $\dot{x}_a$, $\dot{x}_b$, and $\dot{x}_c$. Starting from the left-hand side of Figure 2.5, the reactants have a combined chemical potential of $\mu_1 = \mu_a + \mu_b = RT \ln(K_a K_b x_a x_b)$. Likewise, the chemical potential of the products side is $\mu_2 = 2\mu_c = RT \ln(K_c^2 x_c^2)$. The reaction equation is then

$$\begin{aligned} \nu_1 &= \kappa \left[\exp\left(\frac{\mu_1}{RT}\right) - \exp\left(\frac{\mu_2}{RT}\right) \right] \\ \nu_1 &= \kappa [K_a K_b x_a x_b - K_c x_c^2] \end{aligned} \tag{2.5}$$

Note that we have $\nu = -\nu_1 = \nu_2$. From this, we can write the full mass-action kinetic equations:

$$\begin{aligned} \dot{x}_a &= -\nu = -\kappa K_a K_b x_a x_b + \kappa K_c x_c^2 \\ \dot{x}_b &= -\nu = -\kappa K_a K_b x_a x_b + \kappa K_c x_c^2 \\ \dot{x}_c &= 2\nu = 2\kappa K_a K_b x_a x_b - 2\kappa K_c x_c^2 \end{aligned} \tag{2.6}$$

2.4 Thermodynamics and bond graphs

In their work on chemical reaction bond graphs, Oster *et al.* [59] originally called the field *network thermodynamics*. Bond graphs theory is closely linked with thermodynamics.

Much of the literature on bond graphs in recent decades has been on modelling thermodynamic forces explicitly. For example, Couenne *et al.* [63, 64] represent entropic contributions with multiport bond graphs. van der Schaft *et al.* [65] achieves a similar result by modelling chemical thermodynamics with port-Hamiltonians (recall that port-Hamiltonians are bond graphs without the full graph structure). See also for further examples of this approach Borutzky [19] and Breedveld [66].

In this section, I will highlight the key ideas of thermodynamic modelling with bond graphs. This theory will be particularly relevant when introducing stochasticity and thermal energy to components in [chapter 5](#).

2.4.1 The First Law: energy conservation

The First Law of Thermodynamics (*the total energy of an isolated system is conserved*), can be written as the fundamental energy equation [67]:

$$dU = T dS - p dV + \sum_i \mu_i dN_i \quad (2.7)$$

where we have defined state functions for internal energy U , temperature T , entropy S , pressure p , volume V , chemical potential μ_i and number of molecules N_i for species i . Other energy sources (such as magnetic fields) may be included as additional terms [68].

[Equation 2.7](#) uses differential notation to indicate how one state functions changes relative to another, regardless of how that change occurred. (This is the definition of a state function — change is independent of the path taken). If we interpret [Equation 2.7](#) as changing over time, then we can write

$$\dot{U} = \underbrace{T\dot{S}}_{\dot{Q}} - \underbrace{p\dot{V}}_{\dot{W}} + \sum_i \underbrace{\mu_i\dot{N}_i}_{\dot{X}_i} \quad (2.8)$$

where we write $\dot{U} \equiv dU/dt$ for short.

Notice that each term in [Equation 2.8](#) is the product of a conjugate pair of variables, the first defining an energy equilibrium, and the second a rate of change of state. In other words, each term defines a bond graph energy domain ([Table 2.4](#)). Thermodynamic state functions work naturally with the bond graph framework.

	Power	effort	flow
Thermal	$\dot{Q}$	T	$\dot{S}$
Work	$\dot{W}$	p	$\dot{V}$
Chemical	$\dot{X}$	μ	$\dot{N}$

TABLE 2.4: Thermodynamic bond graph domains

As an interesting observation, each of the domains in [Table 2.4](#) correspond to a class of bond graph component: thermal power is dissipative (**R**); work power is mechanical and holds ‘kinetic’ energy (**I**); chemical power is stored as a potential energy (**C**). [Figure 2.6](#) demonstrates this relationship, with the internal energy taken to be an energy ‘source’.

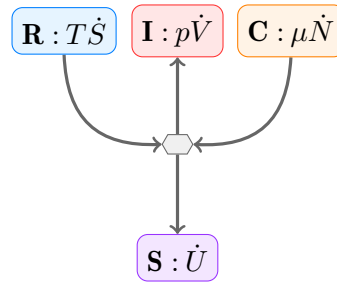


FIGURE 2.6: Thermodynamic energy domains correspond to the different bond graph component types.

2.4.2 The Second Law: entropy and free energy

The Second Law of Thermodynamics (*entropy in an isolated system does not decrease*) requires that entropy (on average) increases until thermal equilibrium is reached. It is difficult to measure entropy directly, so the **free energy** is often used instead.

The free energy is the effective thermodynamic driving force for a particular thermodynamic ensemble. It is also the remaining useful energy after accounting for the energy lost through irreversible processes. Mathematically, this is equivalent to a Legendre transform of a system's internal energy $U = F + TS$ [66]:

$$F(\mathbf{p}, \mathbf{q}, T) = U(\mathbf{p}, \mathbf{q}, S) - TS \quad (2.9)$$

where $\mathbf{p}$ and $\mathbf{q}$ are state variable vectors of the generalised momenta and displacement, respectively. The transformation to the free energy changes the dependent state function from entropy to temperature. If the temperature is then assumed constant, the free energy becomes a measure of entropy that is easier to use. Minimising free energy is equivalent to maximising entropy.

In general then, if the $\Delta F < 0$ then the system moves towards an equilibrium where $\Delta F = 0$. The free energy is therefore the driving force of the system and indicates whether the system will change to another equilibrium position. This only indicates the direction of change, and not the rate at which that equilibrium is reached.

The driving force in biochemical systems is **Gibbs free energy** $G = U + PV - TS$. We assume an NPT ensemble (i.e. constant N, P, T), which means that Gibbs free energy becomes the total chemical potential of the all the species: $\Delta G = -\sum_i \mu_i N_i$. If $\Delta G < 0$, a reaction occurs spontaneously.

Free energy of components

The free energy $F = E - TS$ can be defined in terms of bond graph components and variables (Figure 2.7). Note that the thermal port is defined by *macroscopic* variables T, S , while the free energy port is defined by *microscopic* variables e, f .

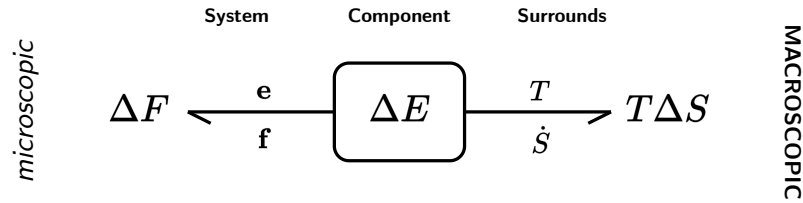


FIGURE 2.7: Bond graph component or system (centre) with internal energy change ΔE , free energy port(s) ΔF , and thermal port $T\Delta S$. The total free energy change is $\Delta F = \Delta E - T\Delta S$.

The free energy of a single component with n (non-thermal) ports is

$$\frac{d}{dt}F = \mathbf{e} \cdot \mathbf{f} = \sum_k^n e_k(t) \cdot f_k(t) \quad (2.10)$$

where $\mathbf{e}$ and $\mathbf{f}$ are vectors of the effort and flows of the component across all the (standard) ports. ‘Outward’ flows have a flow $f_k < 0$. Note that the direction of flow can change over time, so the bond graph remains acausal.

A component may have an internal energy store $E(\mathbf{p}, \mathbf{q})$ which depends on $\mathbf{p}$ and $\mathbf{q}$. Then the irreversible heat flow is

$$T\dot{S} = \dot{E} - \sum_k^n e_k(t) \cdot f_k(t) \quad (2.11)$$

If T is constant, the total change in entropy from a component

$$\Delta S = E - \frac{1}{T} \sum_k^n \int e_k(t) \cdot f_k(t) dt \quad (2.12)$$

Thermal ports can theoretically be added to any bond graph component (Figure 2.7). $T\dot{S}$ is the heat energy released from the thermal port, and F is the (work) energy from the standard port(s). While all components have a free energy, not all components have both an energy store and thermal port. The energy store E is present in **C**-type components, and the thermal port is only present in **R**-type components. Basic resistive **R**-components do not store energy ($E = 0$), so the free energy is $\Delta F = -T\Delta S$. In the biochemical reaction component **Re**, the reaction energy is

$$\Delta G_{\text{rxn}} = -T\Delta S_{\text{rxn}} \quad (2.13)$$

Strictly speaking, there are different entropies to consider in Figure 2.7: the entropy of the system S_{sys} , component S_{comp} , and surroundings S_{surr} . The entropy change that the component contributes to the system is the net change in internal and irreversible entropy.

$$\Delta S_{\text{sys}} = \Delta S_{\text{comp}} + \Delta S_{\text{surr}} \quad (2.14)$$

$\Delta S_{\text{surr}} = -Q/T$ is the heat energy Q lost and is by convention negative. As was noted with the entropy resistor definition, the entropy in the thermal port is irreversible.

S_{comp} can be understood as the contribution to the total entropy of the system. Even if a component does not dissipate energy, we can define its own *internal* entropy (see [69]). A good example of this is the chemical store component Ce , where the entropy of the species is defined in terms of the chemical concentration x

$$S_{\text{comp}} \propto R \ln(x) \quad (2.15)$$

with the change in entropy defined by the relative concentration

$$\Delta S_{\text{comp}} = R \ln\left(\frac{x}{x_0}\right) \quad (2.16)$$

where R is the gas constant and x_0 is the baseline concentration.

Entropy in other physical domains

Internal entropy is applicable in other physical domains. As a non-obvious example, take a linear mechanical spring with potential energy function $E(x) = \frac{1}{2}kx^2$ where $x(t)$ is the displacement (compression or elongation), and k is a spring constant. For simplicity, assume $x \in \mathbb{R}^+$. If there is some uncertainty of x at a given time such that x follows a probability distribution, then we can consider $\{x|x \in \mathbb{R}^+\}$ to be the set of all *microstates* of the spring. The *macrostate* is the possible energy levels E that the spring can store.

At equilibrium, no power is leaving through the free energy port, so $\Delta F = 0$. From Equation 2.11 the entropy production of the spring $\dot{S}_k$ is

$$\dot{S}_k = -\frac{F\dot{F}}{kT} \quad (2.17)$$

and the total change in entropy is

$$\Delta S = -\frac{k}{2T} (x(t_1)^2 - x(t_0)^2) \quad (2.18)$$

The notable thing here is that $\Delta S_k = 0$ if the spring returns to the same configuration state. The most direct way is when $x(t_0) = x(t_1)$, but there is also a symmetric position $-x(t_1)$ with the same entropy.

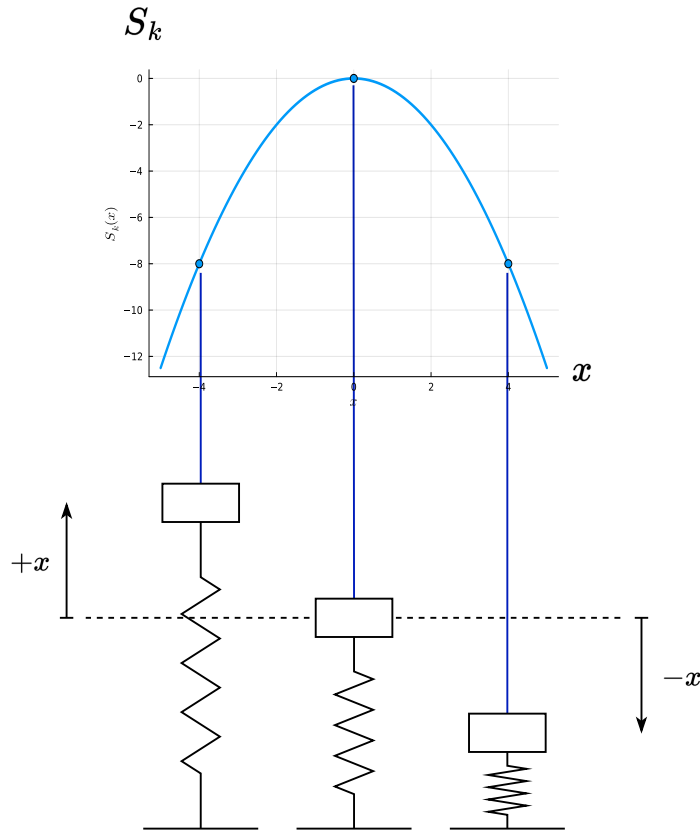


FIGURE 2.8: Entropy of a spring S_k and how it relates to the spring's displacement x . The equation for entropy is $S_k(x) = -kx^2/2T$ with $k = T = 1$. Entropy is maximised when the spring is in the default position (middle, $x = 0$). For each position $+x$, there is a corresponding opposite position $-x$ that has the same entropy.

Figure 2.8 shows the entropy of a spring S_k and how it relates to the spring's displacement x . Entropy is maximised when the spring is in the default position (middle, $x = 0$). A state-space trajectory which returns the spring to its starting position will have a net entropy production of zero.

For each position $+x$, there is a corresponding opposite position $-x$ that has the same entropy $S_k(+x) = S_k(-x)$. Therefore, the trajectory $+x \rightarrow -x$ will also produce zero entropy (ignoring the work that may be required to move the spring to that position).

2.4.3 Entropy resistors and thermal ports

Sometimes the heat dissipated by a resistive component is significant enough that it is worth measuring, such as in a volatile chemical reaction. At other times, the heat generated is by

design, such as with an electric heater or hydraulic brakes. At other times, the resistance term is inconvenient for mathematical derivations. Some powerful theorems and results in physics are valid assuming a conservative system.

It is possible to represent the thermal energy and entropy change in a bond graph by defining **thermal ports**. In this section, I take this concept of thermal ports from an entropy resistor ([19, 66]), and apply it to other bond graph components.

Thermal ports track the energy loss of a component through the thermal energy domain [70]. The effort variable *temperature* T [K] and the flow variable is *entropy flow* $\dot{S}$ [J/Ks] together define thermal energy flow $\dot{Q} = T\dot{S}$ [J/s]. See [58] for thermal bond graphs can be used, and also [71] for an alternative formulation based around heat convection.

Informally, an entropy resistor is a resistor with an extra port that tracks power loss through temperature and entropy flow variables. The total power is captured in these variables, and the component becomes power-conserving.

The constitutive relation of $\mathbf{R}$ is a function of only the efforts and flows of its ports, given in implicit form in Equation 2.19.

$$\phi_R(e_1, \dots, e_n, f_1, \dots, f_n) = 0 \quad (2.19)$$

n is the number of ports of the $\mathbf{R}$ component. The most common case seen in bond graph modelling is the linear resistor $e = Rf$. (some authors define the linear equation as the $\mathbf{R}$ -component and define components with multiple ports as an $\mathbf{R}$ -field instead[66]).

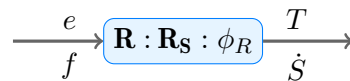


FIGURE 2.9: Entropy Resistor

A modification of the design leads to a power-preserving *entropy resistor* [19, 66] with symbol $\mathbf{R}_s$ (Figure 2.9). The thermal port on the right-hand side has effort as *temperature* T [K] and flow as *entropy flow* $\dot{S}$ [J/Ks]. Entropy flow (also called *entropy production*) is the change in entropy over time that *this* component contributes. It is alternatively written f_S or dS/dt .

Borutzky [19] and Breedveld [66] give the following constitutive relations for a resistor with a thermal port:

$$\begin{aligned} e &= Rf \\ \dot{S} &= \frac{e^2}{RT} \end{aligned} \quad (2.20)$$

The effort e and flow f here are generic bond graph variables, and R is the resistance parameter. The last relation in Equation 2.20 is from the power preserving requirement $ef = P_1 = P_2 = T\dot{S}$.

Note that ‘heat’ refers specifically to heat energy $Q = TS$ which has units of Joules, and not necessarily an increase in temperature. An increase in entropy production at constant temperature will still release heat energy.

One can show that the Second Law of Thermodynamics is satisfied by the entropy resistor.

$$\dot{S} = \frac{1}{T}ef = \frac{e^2}{RT} \geq 0 \quad (2.21)$$

Ignoring the trivial case when $e = 0$, we see that $\dot{S} > 0$, and so the second law of thermodynamics is satisfied¹.

Entropy resistors have three benefits for our purposes:

1. One can track thermal energy Q as an observable and explicit ‘output’ of the model;
2. $\mathbf{R}$ components become power-conserving and energy-conservative, and therefore applicable to a wider range of mathematical results;
3. Entropy as a state variable can help bridge microscopic and macroscopic models and introduce probability to energy modelling.

There are a few points to be aware of regarding [Equation 2.20](#).

First, the entropy from the thermal port is *irreversible* entropy. This is used to represent the irreversible thermal energy loss through dissipative effects such as friction.

Second, while T is assumed to be constant, it can be treated as a control parameter as long as the change is slow enough not to affect the dynamics of the system. Lowering the ambient temperature will increase the entropy flow, all other factors being constant. There is not an obvious way to control $\dot{S}$ in the same way.

Third, temperature and entropy are *global* quantities defined on the entire system. The variables T and S as they are used here can be thought of as *local* to the particular bond graph component. The individual components variables contribute to the global variables. If the system being modelled is assumed to be in thermal equilibrium, then all temperatures are the global temperature.

¹This assumes of course that $e, f \in \mathbb{R}$ which may not be the case in, for example, complex-valued voltages in AC circuits. In these cases, the complex-valued impedance Z is used instead of resistance R , so the constitutive relation is different.

Example: electric circuit with entropy resistors

Figure 2.10 shows an example of how entropy ports might be used in electric circuits. This example will also show why the bonds in the thermal domain should be connected with a 0-junction.

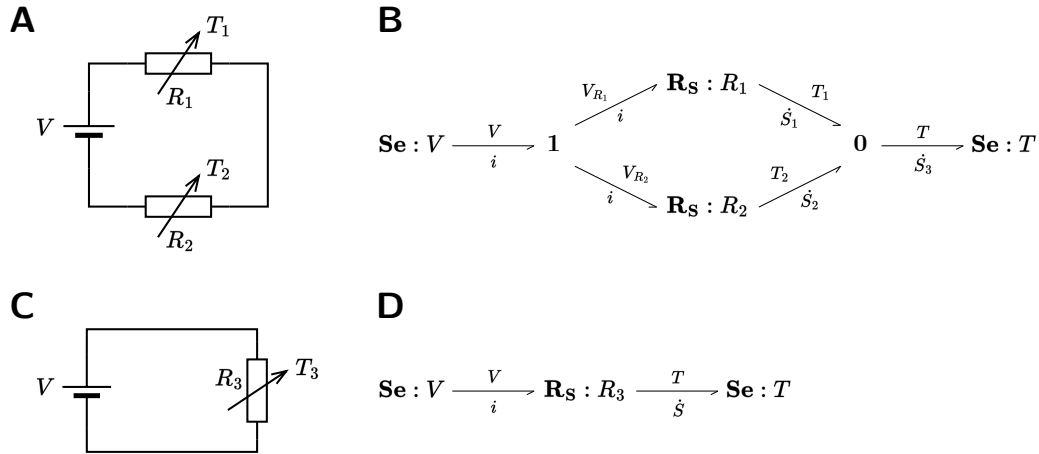


FIGURE 2.10: Entropy resistors in electric circuits in thermal equilibrium. (A-B) Electric circuit with two entropy resistors and the corresponding bond graph. Because the temperatures are equal, the entropy change is additive. (C-D) Circuit and bond graph with equivalent resistance. The circuits are equivalent, so the new resistance is $R_1 + R_2 = R_3$ and the power output $i^2 R_3 = T_3 \dot{S}_3$. If $T_1 = T_2 = T_3 = T$ then $i^2 R_1 + i^2 R_2 \equiv T \dot{S}_1 + T \dot{S}_2 = T \dot{S}_3$.

Circuits (A) and (C) are electric circuits with resistors that produce thermal energies $Q = T(\dot{S}_1 + \dot{S}_2) = T \dot{S}$. Circuit (C) is equivalent to circuit (A) with new resistance $R_3 = R_1 + R_2$ (resistors added in series). Figures (B) and (D) are bond graphs of (A) and (C) respectively.

The thermal ports of the resistors in circuit (A) will need to be merged somehow. Since the circuits are equivalent (they represent the same electric system) the total power produced must be the same. Intuitively, the local temperatures T_1 and T_2 should also be equal. If this wasn't the case, i.e. if $T_1 \neq T_2$ then there would be a new temperature T_3 would be some combination of the old temperatures, which would imply that two identical systems (circuits (A) and (C)) have different observed local temperatures. The more straightforward approach is, therefore, to assume a global temperature $T_1 = T_2 = T_3$. This is essentially the same as assuming thermal equilibrium, but note that this assumption is not absolutely required.

Chapter 3

Two Component Systems

Two-component systems are the main way many bacteria sense their environment. They have been extensively studied in bacteria, where they have been shown to detect a range of stimuli including nutrients, temperature, acidity, and pressure.

We understand only in outline how the network topology of a two component system affect its function, and how evolution has shaped the repertoire of existing two-component systems. In particular, there are important unanswered questions for the existence of different architectures of such fundamental and fundamentally important biological signalling and regulatory systems.

Here we apply a physics-based modelling framework to modelling two component systems. We find that tracking energy flow through the cell explicitly reveals new energy-dependent behaviour in signalling sensitivity, noise filtering, and maximal cell response. We conclude by mapping out the evolutionary scenarios that favour the two extant two component system architectures.

The first step in seeing if bond graphs can be used in whole cell modelling is to apply the framework to a simple bacterial system. Our objective was to explore fundamental questions in bacterial function while also developing new abstract methods for constructing modular bond graphs.

We decided to model Two Component System (TCS) signalling cascade as our representative bacterial system. TCS are the signalling mechanism that allow bacteria to sense and respond to their environment. TCS were chosen for a number of reasons:

1. They have a relatively simple structure with repeated motifs that could easily be built with modular bond graphs.

2. Several variations of the TCS exist with increasing model size and complexity. This means we could scale up the model complexity later while reusing the bond graph sub-models. In this study, we looked at TCS variants with one binding site (orthodox) and three binding sites (unorthodox).
3. Many different instances of TCS exist within the same cell. *E. coli*, for example, contains 26 orthodox and 4 unorthodox TCS variants [72]. Although the TCS variants may differ slightly in network topologies, they are comprised from the same base reactions (activation, phosphotransfer, degradation, etc.). Bond graphs can be used to build TCS variants from these base reactions as required.
4. TCS play a crucial role in the cell and are a core component of whole cell models. The *E. coli* whole cell model project explicitly lists TCS as a core function of the cell [13].
5. Adenosine triphosphate (ATP) are the essential energy currency molecules for TCS and other bacterial systems. It has been shown experimentally that bacteria can survive under starvation conditions [73, 74]. As an energy-based framework, bond graphs are a good tool to study this ATP dependence.

The main body of this chapter has been published as a preprint on bioRxiv: *Energy Dependence of Signalling Dynamics and Robustness in Bacterial Two Component Systems*[1]. Supplementary Information accompanying the paper is included in [Appendix A](#).

Energy dependence of signalling dynamics and robustness in bacterial two component systems

Abstract

One of the best known ways bacteria cells understand and respond to the environment are through Two-Component Systems (TCS). These signalling systems are highly diverse in function and can detect a range of physical stimuli including molecular concentrations and temperature, with a range of responses including chemotaxis and anaerobic energy production.

TCS exhibit a range of different molecular structures and energy costs, and multiple types co-exist in the same cell. TCSs that incur relatively high energy cost are abundant in biology, despite strong evolutionary pressure to efficiently spend energy. We are motivated to discern what benefits, if any, the more energetically expensive variants had for a cell.

We seek to answer this question by modelling energy flow through two variants of TCS. This was accomplished using bond graphs, a physics-based modelling framework that accurately models energy transfer through different physical domains. Our analysis demonstrates that energy availability can affect a cell's signal sensitivity, noise filtering effectiveness, and the stimulus level where cell response is maximal. We also found that these properties are determined not by the molecular parameters themselves, but the reaction rate parameters that govern the reaction systems as a whole.

This suggests possible connections between the molecular structure and evolutionary purpose of any two-component system. This opens the door to new synthetic circuit design in systems biology, and we propose new hypotheses about this link between structure and purpose that could be experimentally verified.

3.1 Introduction

Two-Component Systems (TCS) are the eyes and ears of bacteria. They are biochemical signalling elements that transmit information from the cellular exterior to the inside of the cell to initiate an appropriate response to environmental cues. TCSs provide the mechanisms by which bacteria sense nutrients, toxins, pH, temperature, and signals from other bacteria [75–85].

The two components of a TCS are a receptor histidine kinase (HK), which typically sits across the cell membrane and which also acts as a phosphatase; and the response regulator (RR), which triggers the appropriate cellular response. TCSs also exist in other domains of life, especially in plants.

In *Escherichia coli* there are 30 different TCSs that sense and respond to different environmental signals (see [Two Component Systems Supplementary Information](#) for a summary table). The TCSs can be divided into two groups characterised by different architectures of the sensor histidine kinase (see [Figure 3.1](#)). First, the majority of TCSs belong to the orthodox class, where the HK has a single phosphorylation domain; 26 of *E. coli*'s TCSs belong to this class. Second, the unorthodox class of TCSs has an HK with a more complicated phosphorelay architecture, where a signal travels along three successive HK domains through phosphotransfer.

The functional and evolutionary reasons for the existence of two distinct forms of a sensing mechanism have been the subject of considerable interest and debate. Why do the more complex signalling systems exist alongside the simpler variants? Do unorthodox TCSs have evolutionary benefits compared to orthodox TCSs? If so, what are these advantages? And under what conditions do they manifest?

Answers to such general evolutionary questions can be hard to come by: a multitude of factors shape evolutionary dynamics, and Simpson's paradox can be ubiquitous in any attempt to attribute causality to evolutionary questions [86].

Differences in the reliability of information transmission have been put forward as one reason why a seemingly 'over-engineered' reaction systems may actually be crucial for survival [78, 87, 88]. But this is only one potential factor affecting evolution of TCSs. The next obvious environmental and physiological factor that can affect evolution of molecular processes are imposed by the thermodynamic constraints on molecular and cellular systems.

For example, the amino acid length, and thus energy costs for the different TCSs vary considerably [72]: the EvgSA TCS (involved in drug resistance) requires twice as many amino acids as the QseCB pathway (used for cell motility) and therefore requires more energy to synthesise. At a physiological level, all TCS reactions rely on a phosphoryl group supplied by ATP,

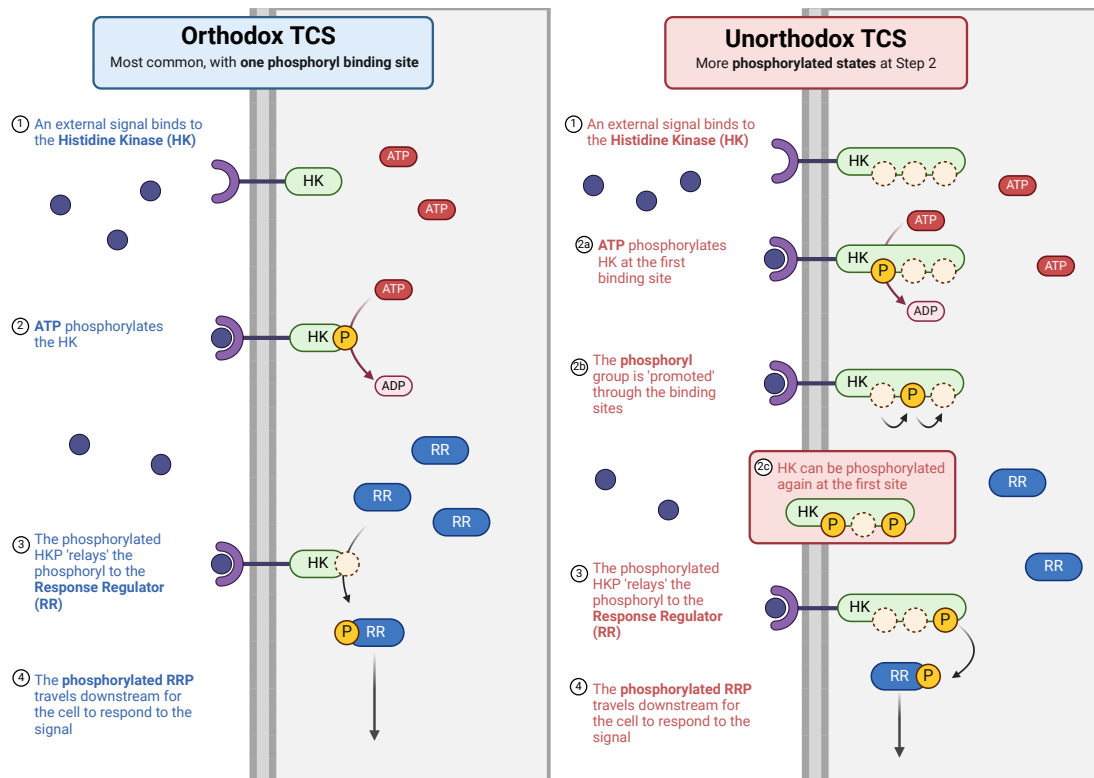


FIGURE 3.1: The Two Component Systems (TCS) signalling system found in many bacteria. The two main variants of TCSs shown here are the Orthodox and Unorthodox TCSs. The process in both systems are fundamentally the same: (1) An external stimulus binds or reacts with a Histidine Kinase; (2) ATP phosphorylates the HK; (3) The phosphoryl group is transferred to a Response Regulator; (4) The phosphorylated Response Regulator travels further into the cell to begin a downstream process. Created with BioRender.com.

the primary energy currency of a cell. A TCS with more reactions in its signalling cascade will be energetically more expensive to operate. It is beneficial for a cell to spend ATP efficiently, and yet these energetically costlier TCSs are abundant in biology.

This study aims to address the thermodynamics and energetics and their impact on qualitative differences between primary two TCS architectures. What different behaviour or functions can TCSs perform? How do TCSs adapt to a changing environment? How well do these systems perform when energy itself is limited? Can they function well in low energy (starvation) scenarios?

A suite of mathematical modelling studies have already attempted to understand TCS dynamics, their physiological roles, and their evolutionary properties [75, 78, 81, 84, 85, 87–93]. These models represent the chemical reaction system of TCSs as a set of ordinary differential equations (ODEs) and solve for the response variable numerically, although for some models analytical solutions can be derived [84, 85, 88, 89].

These models have proven useful in investigating the properties of the temporal response to a stimulus and how the response changes as a function of the signal strength. What existing

studies lack, however, is an analysis that accounts for the bioenergetics and thermodynamics of signal processing. The dependence on ATP concentration (the energy currency of the cell) has been routinely omitted from the models. But TCSs require ATP hydrolysis to drive the signalling cascade. While [87, 88] include ATP in their reaction schemes, they assume sufficient ATP for operation and drop it from the model, even though ATP is not fixed, and can become limited in extreme circumstances. Accounting for the dependence on ATP concentration is essential for understanding how these signalling systems operate [94]. We hypothesise that energy abundance in a cell, particularly as expressed in concentration of ATP, will have a measurable impact on the effectiveness and behaviour of cellular subsystems.

TCSs can contain 'futile cycles', designs whose function is to dissipate energy. This dissipation can serve important regulatory roles in biochemical pathways, where ATP (or other energy sources) has to be shared across different cellular functions [23, 95].

We capture the energetics of molecular signalling in a new set of mathematical and thermodynamically consistent models of TCS dynamics. Modelling the energetics of biochemical systems has been shown to be important for predicting the characteristics of biological systems [21, 96]. While conservation of mass (with e.g. mass action kinetics) is generally considered in modelling, energy conservation or realistic dissipation, are rarely enforced. Bond graphs are a generalised energy-based modelling framework. Bond graphs (i) capture cellular biophysics and thermodynamics; and (ii) provide a modular model structure that is the hallmark of many biological systems; see [Methods](#) for more details.

From the energy-based perspective, we create new insights about structure, function, and purpose of TCSs in different energy environments. In particular, we find that (1) Energy availability affects the performance and sensitivity of signalling systems. (2) Energy availability dictates the signal-to-noise ratio of TCS signalling. (3) Some systems can filter out noise and eliminate noise propagation in high energy systems. (4) TCS systems can dynamically select particular stimulus levels, in an energy dependent manner..

3.1.1 Orthodox and Unorthodox Two Component Systems

The most common TCS structures are the orthodox and unorthodox variants [76, 78, 79, 83, 97, 98]. All variants share the same core signalling structure: a membrane bound histidine kinase (HK) and a response regulator (RR). In the presence of an external stimulus, the HK is phosphorylated on its binding domain. In the orthodox TCS, the HK consists of a single histidine phosphotransfer domain and an ATP binding domain; the unorthodox HK has three binding domains. This phosphoryl group is passed on to the RR either immediately (orthodox) or after a series of phosphorelay reactions (unorthodox) to produce the response species

RRP. This RRP is the signalling output which travels into the cell interior to initiate a response to the signal (see [Figure 3.1](#)).

Our study considers how energy availability affects TCS signalling. We pay particular attention to differences between the two predominant TCS architectures, the orthodox and unorthodox systems. There has been considerable interest in why these different architectures exist alongside each other. For example, the unorthodox variant has a more complicated reaction system, and the proteins involved are on average longer than their orthodox counterparts ([Table A.5](#)). In *E. coli*, 4 of 30 TCSs are of the unorthodox type. It has been suggested that evolution may favour different architectures for different signalling scenarios. Modelling studies have shed some light on differences between orthodox and unorthodox TCSs: for example, the unorthodox systems appear to be more robust to noise [88]. However, from [Table A.5](#), unorthodox TCSs are also more costly to synthesise, which may point towards trade-offs between cost of production and performance.

Orthodox TCS

The orthodox TCS model is the simplest and most common form of a TCS. The histidine kinase can be in one of two states: unphosphorylated (HK) or phosphorylated (HKP). We model this as a two-step process; HK is activated by the stimulus (S) to produce HK_S. This stimulated species is then autophosphorylated by ATP to produce HKP. HKP passes the phosphoryl group on to its cognate response regulator (RR), to produce a phosphorylated response regulator (RRP). RRP then initiates a downstream response. Eventually, RRP becomes deactivated. The chemical reactions for this process are summarised in [Table A.1](#).

Unorthodox TCS

The unorthodox TCS is a more complex variant of the TCS in which the HK goes through a three-step process to phosphorylate the RR. In this variant, there are three distinct binding sites on an HK available for a phosphoryl group. This means that an HK can be in one of $2^3 = 8$ phosphorylated states. We have labelled these HK1 through HK8 — see [Figure 3.2](#) for the names of all HK states.

There are now multiple parallel autophosphorylation reactions for the unorthodox system. Four of these states (HK1, 3, 4 and 7), which have the first binding site empty, respond to the stimulus and act as starting points for the signalling process. The phosphorylated HK promote their phosphoryl group to a higher binding site through forward phosphorelay. The four states with the final site bound (HK4, 6, 7 and 8) can donate their phosphoryl group to the RR

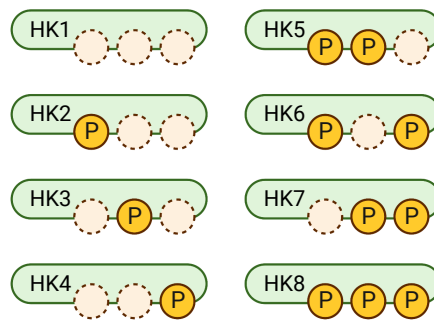


FIGURE 3.2: All HK phosphorylation states in our model of the unorthodox TCS. The first binding site is the start of the signalling cascade, and the last binding site is received by the response regulator. HK can move between different state non-sequentially (see [Table 3.2](#)). Naming convention is the same as in [\[88\]](#). Created with BioRender.com.

molecule. Each of the above reactions can also spontaneously unbind. Eventually, RRP decays back into its original state. The chemical reaction network for this process is summarised in [Table A.3](#). As we will discuss later, the multiple pathways that lead to the phosphorylation of RR allow for extra complexity and adaptive behaviour by the TCS.

3.2 Results

We investigate the role of energetics on TCS signalling through mathematical modelling with bond graphs. As discussed earlier, there are limited kinetic data available for TCSs. Accordingly, we have simulated many instances of this model with different parameters to cover the broad range of physiological possibilities and explore all the dynamics possible in TCSs. Unlike previous studies, we treat ATP concentration as a tuneable parameter. Details on model construction, parameter sampling and simulation protocols are given in [Methods](#). See [Two Component Systems Supplementary Information](#) for the full mathematical description.

3.2.1 ATP increases maximum signal response

To investigate how ATP affects signalling strength, we ran the orthodox and unorthodox models over 10,000 simulations and plotted the response against time ([Figure 3.3](#)). The RRP Fraction is defined as the amount of phosphorylated RR (RRP) normalised by the theoretical maximum RRP expression attained with unlimited ATP. We represent the RRP Fraction as a temporal response curve, which is also called the RRP curve. The response curves for the orthodox and unorthodox TCS models are shown in [Figure 3.3](#).

On average, the temporal response curves are consistent with earlier studies [\[88\]](#); the averaged orthodox response curve appears hyperbolic, while the averaged unorthodox response

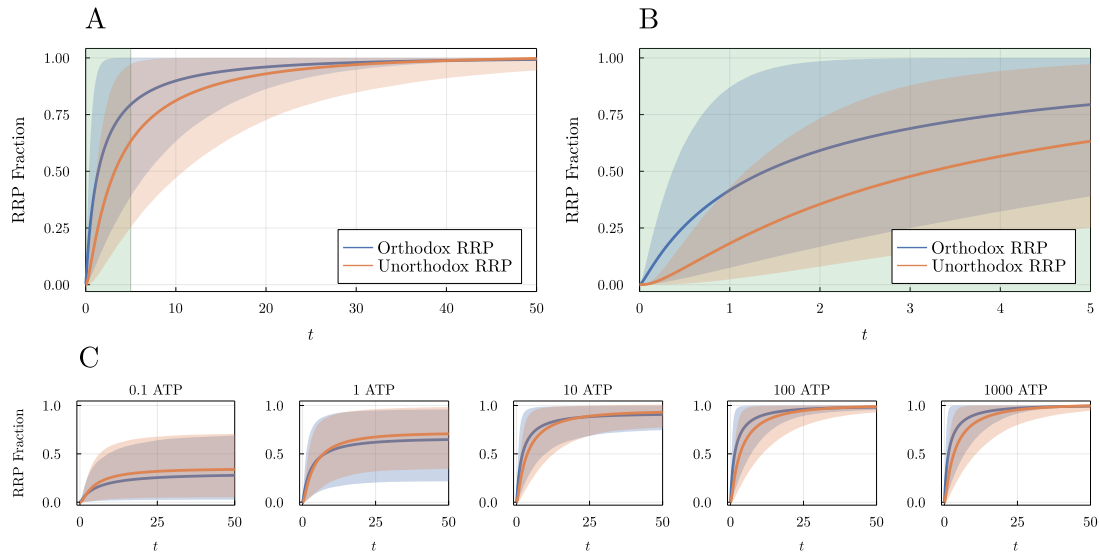


FIGURE 3.3: RRP fraction over time after the presence of a signal at $t = 0$ for the orthodox (blue) and unorthodox (orange) TCS models. **(A)** The RRP curves are each normalised by their maximum expression value and then averaged over 10,000 simulations with parameters sampled from a log-uniform distribution from 0.1 to 10 (9 parameters for orthodox, 35 for unorthodox). The shaded regions capture the RRP curves between the 10th and 90th percentile (ranked by expression level). **(B)** Zoomed in plot of figure (A) for $t \leq 5$, as indicated by the green band. (B) highlights that the orthodox curve is more hyperbolic, and the unorthodox curve is more sigmoidal. **(C)** RRP curves for each TCS model at increasing available ATP levels. Less ATP availability results in a lower signal strength and a slower response, but the maximum signal strength and speed saturates after sufficient energy level. ATP levels are unitless.

curve appears more sigmoidal (Figure 3.3A, solid lines). In Figure 3.3B, we focus on the first 5 seconds of Figure 3.3A where this difference is more distinct; the unorthodox response curve takes slightly longer to reach maximum signal and resembles a sigmoidal Hill function.

We found that the maximum expression level of this temporal response curve depends on the amount of ATP available. With higher ATP, a higher maximum response signal is possible (Figure 3.3C). When ATP concentration is low (leftmost subplot), the maximum response is (on average) less than 50% of value at the highest energy level. The highest response is reached at about 100 ATP before the effect saturates. This is true for both the orthodox and unorthodox models. A TCS in a low energy environment may not be able to respond to external stimuli as effectively as in an energy rich environment.

We find here that the average behaviour of the TCS variants aligns with what may expect from previous studies. What is apparent from Figure 3.3 however is that there is a wide band of possible response curve behaviours for a given instance of a TCS. As we will next see, the distinction between an orthodox and unorthodox TCSs response curve may be hard to identify.

3.2.2 ATP affects sensitivity

We next investigated how ATP affects the signalling sensitivity of a TCS. We can quantify the sensitivity of a response by calculating the Hill coefficient, which can be estimated using

$$n = \frac{\log(81)}{\log(I_{90}/I_{10})} \quad (3.1)$$

where I_{90} , I_{10} are the concentrations of the stimulus when the response is at 90% and 10% of the maximum response steady state value respectively [88, 99]. A coefficient of $n \approx 1$ corresponds to a subsensitive signal, whereas a coefficient of $n > 1$ corresponds to a super-sensitive signal.

The temporal sensitivity of a signal determines how quickly or slowly the signal responds to a stimulus: a subsensitive curve (Figure 3.4C, top) resembles a hyperbolic curve with a rapid initial response expression; a supersensitive curve (Figure 3.4C, bottom) resembles the characteristic sigmoidal Hill function, with a slower uptake followed by a rapid increase once the stimulus duration has crossed a threshold.

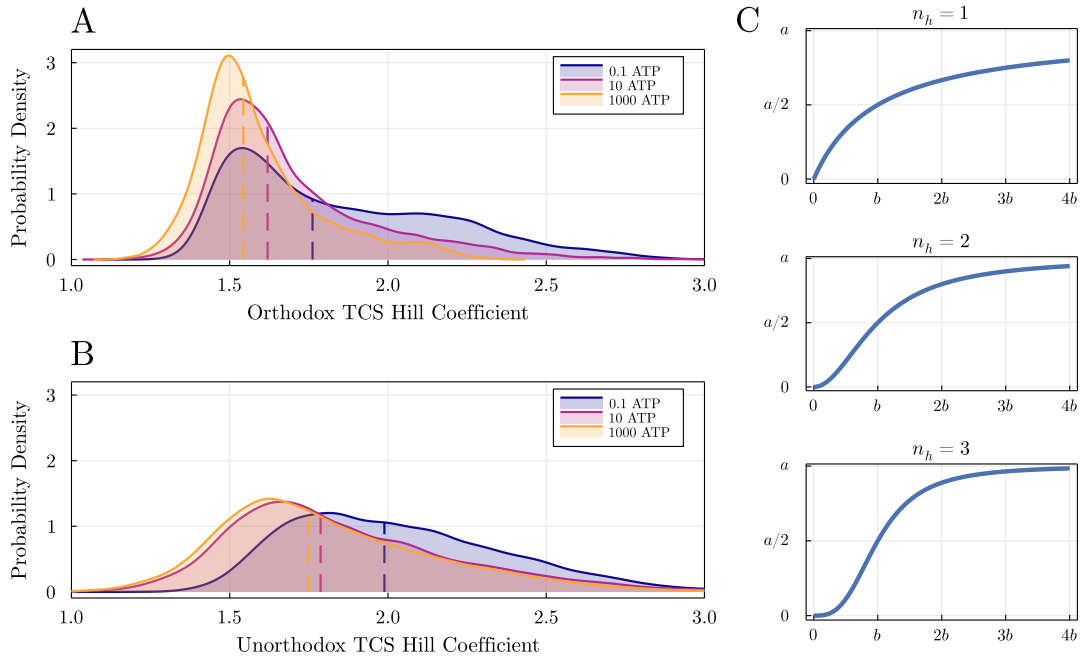


FIGURE 3.4: Hill coefficient distributions for **(A)** Orthodox TCS and **(B)** Unorthodox TCS. Dotted lines indicate median values for each distribution. The Hill coefficient is a measure of sensitivity and is defined by the Hill function $\frac{ax^n}{b^n + x^n}$, where n is the Hill coefficient. It can be estimated using $n = \frac{\log(81)}{\log(I_{90}/I_{10})}$, where I_{90} , I_{10} are the concentrations of the stimulus when the response is at 90% and 10% of the maximum RRP steady state value respectively [88, 99]. **(C)** Hill functions for $n_h = 1, 2, 3$. As the Hill coefficient increases, the response curves become increasingly sigmoidal.

In Figure 3.3B, we see that the unorthodox TCS is more sensitive than the orthodox TCS, which aligns with previous modelling studies [78, 88]. However, this simple dichotomy does not

capture the true range of dynamics displayed by the systems. [Figure 3.4A](#) [Figure 3.4B](#) plot the distribution of estimated Hill coefficients for the 10,000 simulations of both TCS models, at different levels of ATP concentration. There are orthodox systems with $n > 2$ and unorthodox systems with $n < 1.5$.

We also see a shift in the distributions to the right as ATP levels decrease ([Figure 3.3A, B](#)). This implies that when ATP availability is reduced, a signalling system will have a higher Hill coefficient and becomes more sensitive to stimuli. This is most striking for the orthodox TCS model ([Figure 3.4A](#)). For higher ATP levels in the orthodox model, the median Hill coefficient is around 1.54. As ATP decreases, the distribution flattens out towards the right, where we see a higher proportion of Hill coefficients above 2 and even higher than 2.5. The system moves from behaving as semi-hyperbolic to behaving like a sigmoidal switch. Under these low energy conditions, an orthodox TCS will appear to have the same sensitive behaviour as an unorthodox TCS.

This shift towards sigmoidal behaviour is less pronounced in the unorthodox TCS model, as this model tends to already exhibit supersensitive behaviour. This distribution remains stable even for low energy levels (10 ATP), but does still shift for extremely low ATP concentrations (0.1 ATP).

3.2.3 ATP determines noise filtering effectiveness

So far we have only considered signalling behaviour for an ideal stimulus: constant, stable, and unambiguous. We next look at signalling performance in the presence of a noisy stimulus; that is, a stimulus detected with an unbiased error (see [Methods](#) for mathematical details). A noisy stimulus may represent uncertainty in detecting the stimulus (especially at low concentrations), but it may also be an erroneous signal that the cell should not respond to. It is important therefore that critical signalling systems are able to filter out noise, or at least reduce its impact.

Orthodox and unorthodox variants respond to noise differently. Specifically, the unorthodox TCS structure is able to filter erroneous noise more effectively than the orthodox TCS [[78](#), [87](#), [88](#)]. We investigate this phenomenon using our bond graph model and suggest why this occurs. We repeat earlier simulations using the same generated parameter values but with a noisy stimulus input. We then compare the same TCS instance response with both a constant and a noisy signal.

We find that unorthodox TCS instances filter noise more effectively than orthodox TCS instances ([Figure 3.5A, B](#)). For the same noisy stimulus input, the averaged response curves

for the orthodox models (Figure 3.5A) exhibit more frequent spikes in the signal, when compared to the equivalent response for the unorthodox models (Figure 3.5B). The unorthodox TCS model appears to be better at handling noisy inputs on average, in line with previous studies. Moreover, the variability in the output is less prominent at high energy levels, compared to the orthodox system.

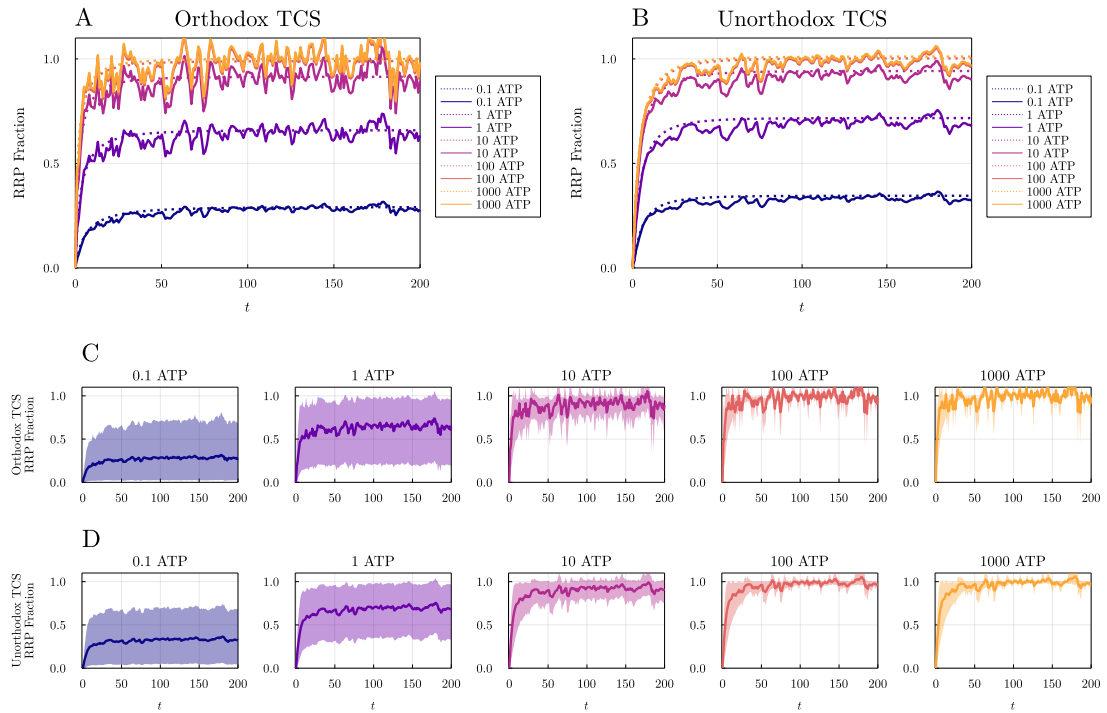


FIGURE 3.5: Temporal response curves (RRP Fraction) for TCS models when exposed to a noisy stimulus. The response curves for each ATP level averaged over all parameter set simulations for the (A) orthodox and (B) unorthodox TCS models. The response curves for a noisy stimulus (solid line) are compared to the response from a constant stimulus (dotted line). The averaged noisy response curves for (C) orthodox and (D) unorthodox TCSs over the same simulations are shown with 10-90th percentile ribbons.

When the aggregated response curves for a noisy signal (Figure 3.5A, B, solid lines) are plotted against the aggregated response curves for a constant signal (Figure 3.5A, B, dashed lines), we see that the noisy signals tend to give a slightly lower response level compared to their constant counterparts. This means that the mean signal magnitude is reduced by the extra noise; a noisy stimulus produces a slightly weaker signal strength. This appears to be the case on average across all models and energy levels.

Additionally, we find an interesting effect of energy availability on noise filtering. As ATP concentration is increased, the orthodox TCS amplifies the magnitude of the noise. With higher energy availability, the magnitude and variance of the noisy response orthodox curve increases (Figure 3.5C). On the other hand, the unorthodox TCS models are much better at noise filtering, even at the highest energy levels (Figure 3.5D).

We compared noise filtering performance with three simple measures: the signal mean μ , standard deviation σ , and coefficient of variation σ/μ (Figure 3.6). The coefficient of variation was used to compare the variability (standard deviation) of different distributions. These measures were calculated for the RRP concentration for each simulation run, each model structure, and each level of ATP.

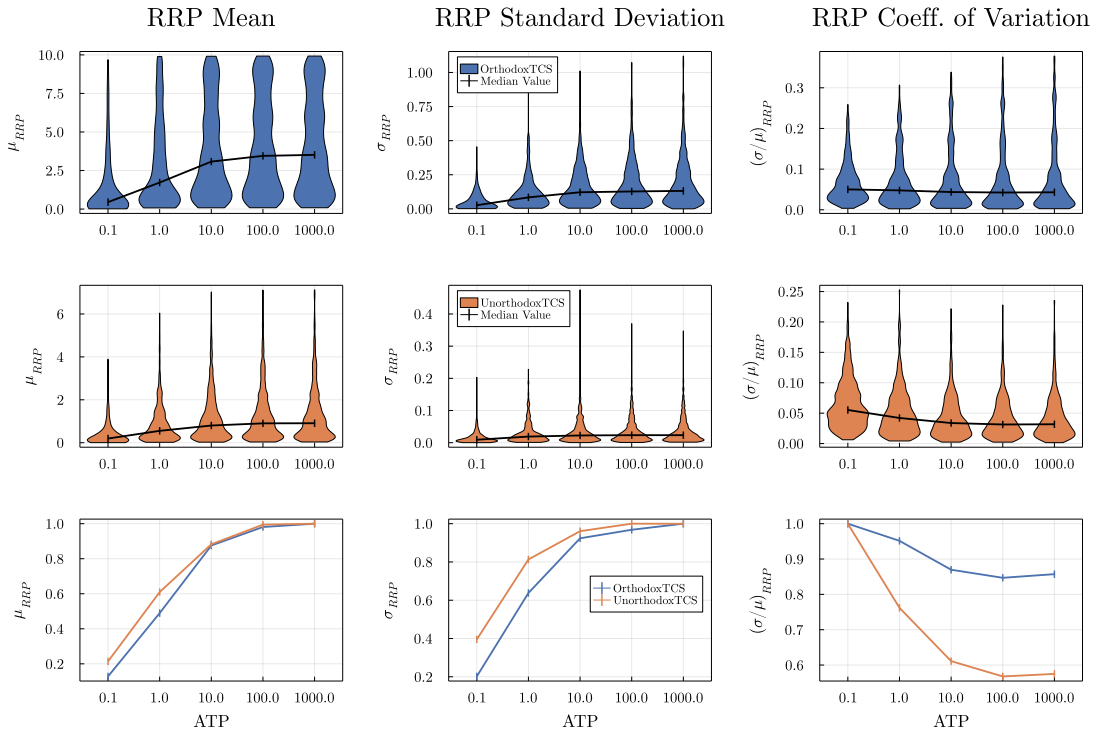


FIGURE 3.6: Performance metrics for the noisy response curves as ATP concentration increases for orthodox (blue) and unorthodox (orange) TCSs. The mean μ , standard deviation σ , and coefficient of variation σ/μ of the RRP concentration were calculated after a warm-up period of 100s. The last row of plots compares the normalised median values of both TCS models for each metric.

For both TCS models, the median signal response drops as ATP levels decrease (Figure 3.6, left column). This is in agreement with what we see in Figure 3.3C Figure 3.3D. The standard deviation of the signal for orthodox TCSs increases with more energy, while it remains stable for the unorthodox system (Figure 3.6, middle column). Again, this follows from what we have seen in Figure 3.5. However, the median coefficient of variation for the orthodox systems does seem to decrease with more ATP, although this decrease in variation is less than for the unorthodox systems (Figure 3.6, right column). The unorthodox TCSs are therefore on average more effective at reducing noise in the signal response.

Why does the unorthodox TCS filter noise better than an orthodox TCS? We begin to see why this may be by examining the reaction fluxes for each model. The bond graph framework allows us to extract the fluxes over time for each reaction in our model. Figure 3.7 plots the

reaction fluxes for two exemplar instances of an orthodox TCS (top) and unorthodox TCS (bottom), along with their response curves. The reaction fluxes (black) are overlaid on top of each other to highlight the cumulative effect of the whole system.

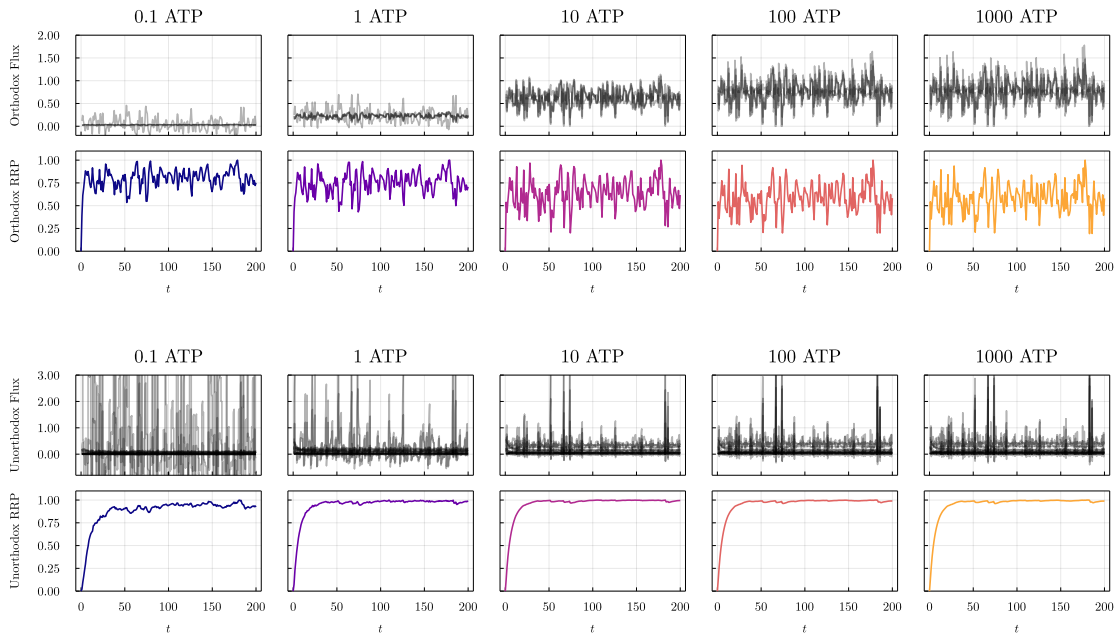


FIGURE 3.7: Comparing reaction fluxes of all reactions in the TCSs, for example orthodox TCSs (top two rows) and unorthodox TCSs (bottom two rows) simulations. ATP levels increase across the figures from left to right. The upper plots are the flux for each reaction in the model as outlined in Table 3.2, and are overlaid over each other in black. The lower plots are the corresponding response curve outputs. For the orthodox model, increasing ATP not only increases baseline flux and flux variability, the apparent baseline signal of the response curve also decreases; more ATP leads to a noisier, less reliable signal despite the higher reaction throughput. The unorthodox model however is robust to additional ATP; the noise is dampened, the signal is maintained, and the reaction flux does not vary considerably.

Figure 3.7 gives us three interesting observations. For the orthodox TCS, increasing ATP has three effects: (1) increases the baseline level of reaction flux, meaning each reaction is happening more often; (2) it increases the variability (noise) of the flux; and (3) it the baseline level for the response curve (second row) decreases as the signal becomes noisier. More ATP leads to a noisier, less reliable signal, despite the higher reaction throughput. By contrast, the unorthodox TCS here does not have the same trends. In fact, the flux variability decreases with additional ATP (third row). The response curve (fourth row) is robust to additional ATP; the noise is dampened, the signal is maintained, and the reaction flux does not vary considerably.

3.2.4 Unorthodox TCS selects for specific stimulus ranges

We finally examine the Signal-Response (SR) curve for the TCS variants. SR curves show the relationship between the stimulus and the response at steady state. We repeat simulations

for a wide range of stimulus levels and determined the final steady state response expression for each run.

The SR curve for all the orthodox TCS we investigate resemble monotonic logistic curves (Figure 3.8, top row). This aligns with what we expect from existing models of TCSs [87–89]. However, for the unorthodox TCSs, we find that only 22% of the SR curves are monotonic at every ATP concentration (Table 3.1a). In the non-monotonic cases, the maximum response expression level is not found at maximum stimulus expression (Figure 3.8, bottom row). This means that a system which optimises for a maximum signal response expression will select for an intermediate stimulus value, not the maximum possible value. Note that only unorthodox systems have this property; no orthodox system studied had a non-monotonic SR curve.

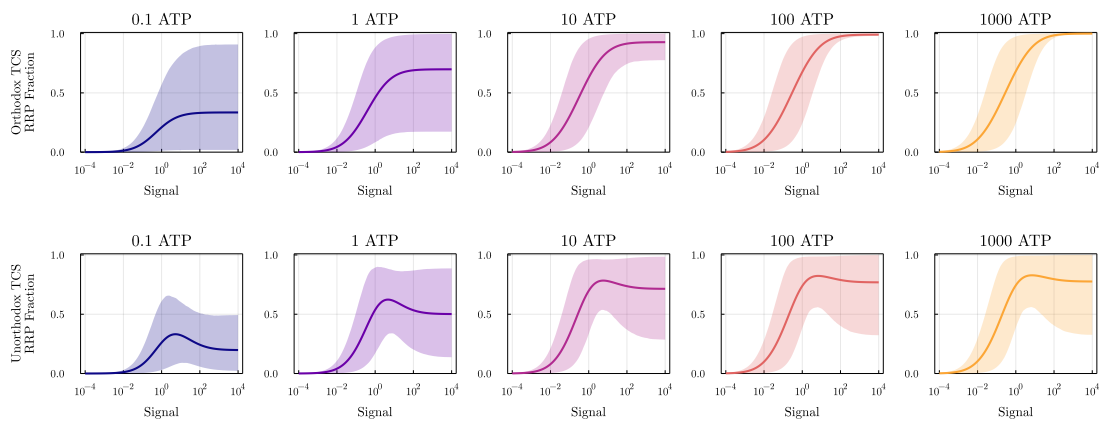


FIGURE 3.8: Signal Response (SR) curve plots for the orthodox and unorthodox TCS models for increasing ATP concentrations. SR plots show the final RRP steady state value (as a fraction of the maximum RRP value) for a range of stimulus sizes (plotted on a log scale). The solid line is the mean SR curve over 10,000 parameter sets, and the ribbon is the 10-90th percentile range. The orthodox SR curves are all sigmoidal, whereas for the unorthodox SR curves can exhibit non-monotonic behaviour.

Remarkably, the existence of a non-monotonic peak in an SR curve is itself energy dependent. For some unorthodox TCSs, the SR curve is only monotonic at low energy levels, and non-monotonic when energy levels are increased. For other unorthodox TCSs, the opposite is true: the curve is non-monotonic at low energy, and monotonic at high energy. We have categorised the SR curve shapes into four classes (Figure 3.9):

- (1) **All Monotonic** Monotonic SR curves at all energy levels
- (2) **Peak Gained** Non-monotonic at higher energy levels only
- (3) **Peak Lost** Non-monotonic at low energy levels only
- (4) **All Non-Monotonic** Non-monotonic SR curves at all energy levels.

These classes can be grouped together in several ways. In a low-energy environment (<0.1 ATP), the All Monotonic and Peak Gained classes are similar, and the All Non-Monotonic and

Peak Lost are similar. In a high energy environment (>100 ATP), the All Monotonic and Peak Lost classes are similar, and the All Non-Monotonic and Peak Gained are similar. Roughly 30% of the unorthodox TCSs instances switched their behaviour across the ATP gradient. Surprisingly, of the 10,000 simulations we ran, we found that only 22% of unorthodox TCSs had monotonic SR curves at all energy levels.

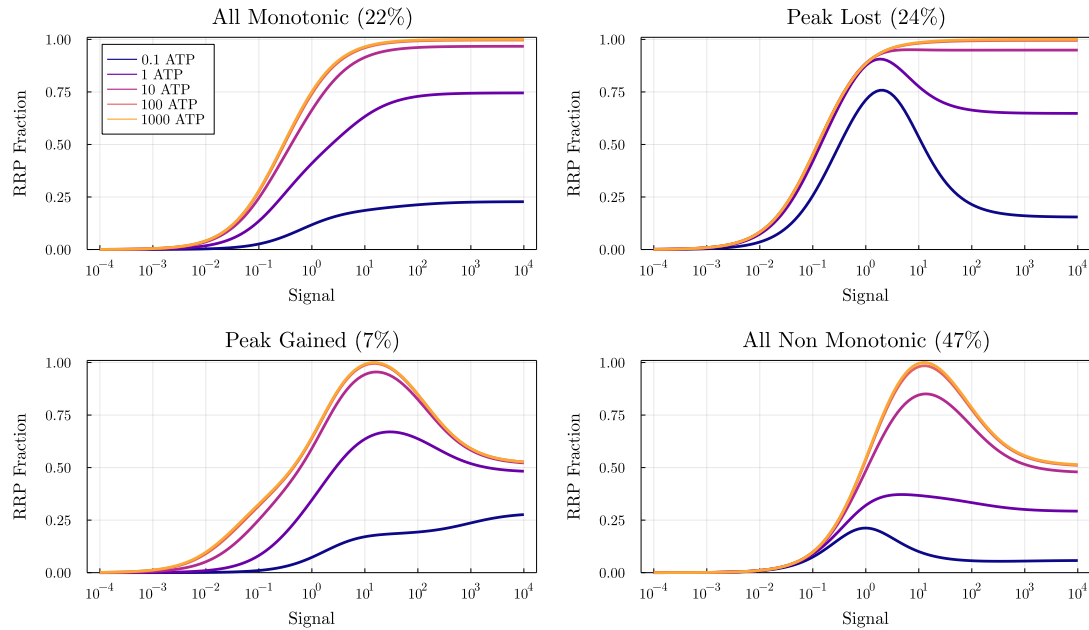


FIGURE 3.9: Representative examples of the SR curve shape classes for the unorthodox TCS simulations. The proportions of each class are included in parentheses. These classes partition each simulation instance based on how the SR curve shape changes as ATP is increased or decreased, all else being equal. **All Monotonic** Monotonic SR curves at all energy levels; **Peak Gained** Non-monotonic at higher energy levels only; **Peak Lost** Non-monotonic at low energy levels only; **All Non Monotonic** Non-monotonic SR curves at all energy levels. The Peak Gained and Peak Lost classes are so called as they are characterised by an appearing or disappearing local maxima (peak) as energy levels increase. Classes grouped by rows have similar high-energy behaviour (100 ATP); classes grouped by columns have similar low-energy behaviour (0.1 ATP).

TABLE 3.1: Proportions of SR curve classes by shape and peak shift for the unorthodox TCS simulations. Examples from each class are shown in Figure 3.9 and Figure 3.10. The Peak Shift classes are only defined for the SR curves, which are All Non Monotonic.

Class	Count
All Monotonic	2206
Peak Lost	2409
Peak Gained	669
All Non-Monotonic	4716
Total	10000

(A) SR Curve Shape

SR Curve Peak Shift	Count
Right Shift	381
No Shift	1414
Left Shift	2921
Total	4716

(B) SR Curve Peak Shift

Similarly, we also find that the position of the SR curve peak can also move across the ATP gradient. The stimulus required for the optimal signal response may change for different energy levels. The peak of the signalling curve can shift to the left or right as energy levels increase (e.g. the All Non-Monotonic example in [Figure 3.9](#)). We have likewise divided the non-monotonic unorthodox TCSs into peak-shift classes ([Figure 3.10](#)): A **Right Shift** means that as ATP increases, the signal peak moves to the right, thereby maximising the response for a higher signal strength (about 8% of cases). A **Left Shift** means that as energy increases, the response is maximised at a lower signal level (about 62% of cases). In 30% of simulations, there is no significant movement in one direction across energy levels (**No Shift**), though the peak may move slightly between levels. Note that these classes are only defined where a peak exists at all energy levels (i.e. only All Non-Monotonic SR curves).

In some rare cases (<0.5% of simulations) the solution exhibited ‘multi-peak’ behaviour, where there are two or more local maxima in the SR curve. It is currently unclear what leads to this phenomenon.

At this stage, it is difficult to predict which class a solution will fit in from the parameter values alone. To draw some insight, we began exploratory analysis of the distribution of parameter values and which SR curves they tended to create.

We first looked at how the parameter values were distributed for the monotonic behaviour of the SR curves. We plot the distribution of the 21 unorthodox reaction rate parameters κ in [Figure 3.11](#), split by our SR classes (see [Two Component Systems Supplementary Information](#) for a similar plot of the 14 thermodynamic constants K). From these plots we can already make several observations.

First, the difference in behaviour is determined entirely by the reaction rates κ . The distributions for the reaction rates all take different shapes for each class; in contrast, the thermodynamic constants K follow the same distribution shape regardless of class ([Two Component Systems Supplementary Information](#)). The reaction rates are what determine these SR curve behaviours. Since the thermodynamic constants are inherent properties of the species, we see that this dynamic behaviour is not a result of species but of the reaction network itself. Exotic behaviour cannot be identified by analysing the chemical species alone; it is necessary to consider the network structure in its entirety.

Second, certain reaction rates stand out as having greater explanatory power. For example, when κ_{ap1} (row 1, column 2) is low, we are more likely to get a Non-Monotonic (blue) or Peak Gained (green) solution. This parameter is responsible for the initial rate of phosphorylation of the HK, so we can infer that slow binding of ATP to the HK domain can cause peaks in the SR curve. A faster binding rate for ATP results in a monotonic switch for the SR. To get the signal selecting behaviour, this early stage reaction should be slow relative to the other TCS

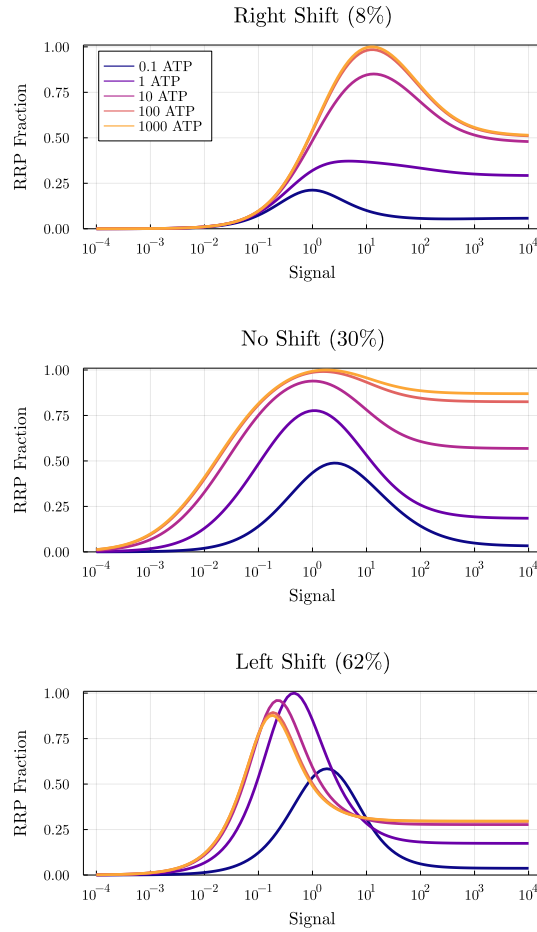


FIGURE 3.10: Representative examples of the SR peak-shift classes for the unorthodox TCS simulations. The proportions of each class are included in parentheses. Classes are only defined for the All Non-Monotonic SR curve shapes, as these classes have a local maxima (peak) at all energy levels. These classes partition the SR curves based on how the horizontal position of the peak shifts as energy levels increase. This means that the signal that produces the maximum response expression level changes as a function of the energy level. **Right Shift** peak moves to the right; **Left Shift** peak moves to the left; **No Shift** no consistent or discernible movement in the peak (some movement is still permitted). Note that vertical shift also occurs alongside horizontal shift, and this may not be monotonic itself, such as in the Left Shift example.

reactions. Incidentally, these classes resemble each other at high energy levels (bottom row of Figure 3.9).

Third, parameters which are nominally similar in function can have different, sometimes opposite effects on the system. For example, consider the forward phosphorylation reaction rates κ_{f2} (row 2, column 3) and κ_{f3} (row 2, column 4). These reaction rates both determine how quickly the second phosphoryl group is 'promoted' from binding site 2 to 3 in the HK, but for different initial HK states (HK3 and HK5 respectively). According to Figure 3.11a, these rates influence whether the SR curve is monotonic at low energy levels. Paradoxically, they exhibit opposite trends: If κ_{f2} is low, non-monotonic (blue) and peak-lost (red) solutions

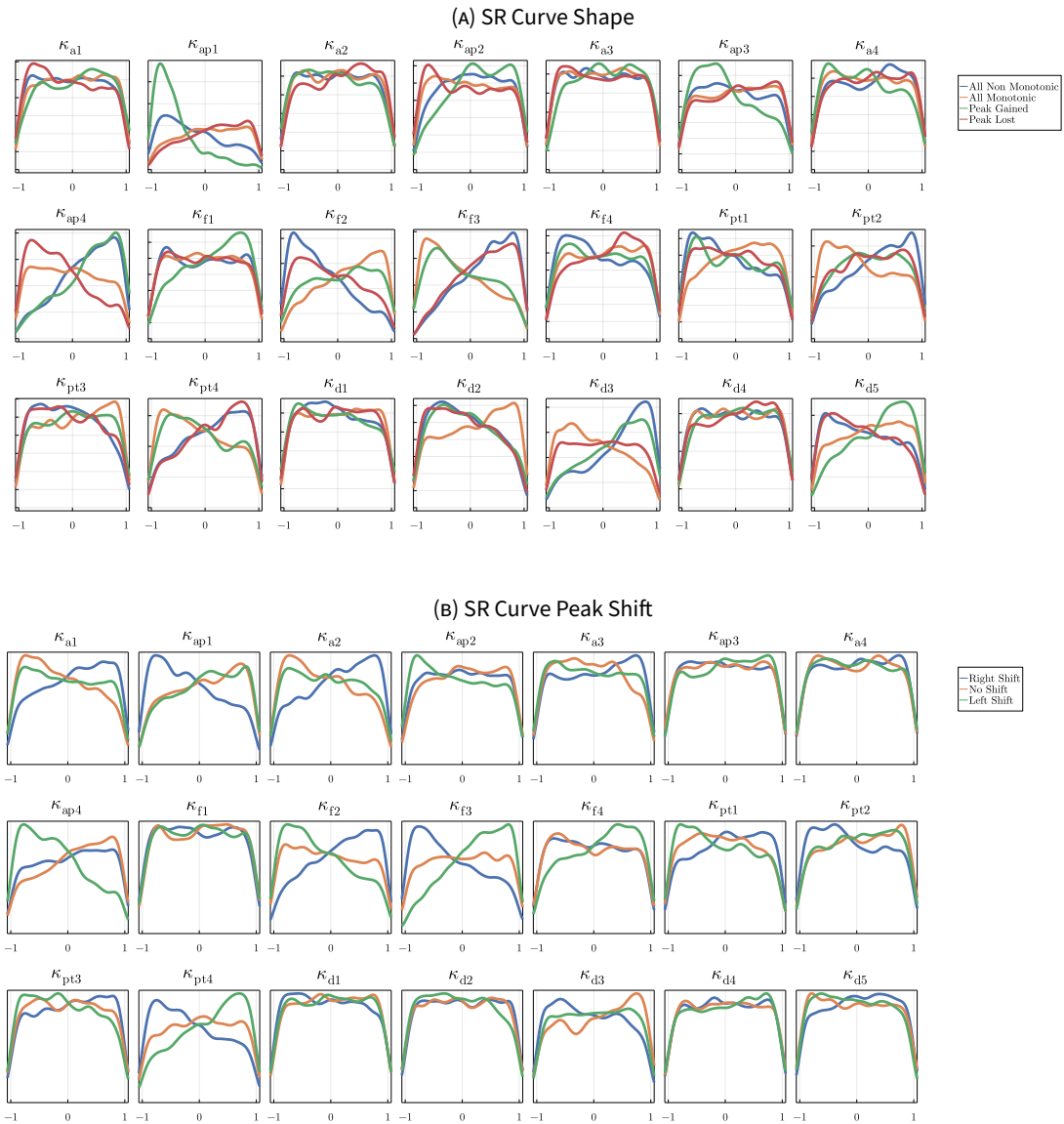


FIGURE 3.11: Log distribution of the sampled reaction rates κ used for the unorthodox TCSs numerical simulations, split by **(a)** SR curve shape and **(b)** SR peak shift (see Figure 3.9 and Figure 3.10). The reaction rate variables are the same as in Table 3.2 and defined in Two Component Systems Supplementary Information. Parameters are sampled from a log-uniform distribution (Equation 3.8). Without any group effects, we would expect all classes to follow a uniform distribution on these log plots; however, particular reaction rate values bias towards particular SR curve shapes. For example, a low value for κ_{ap1} (row 1, column 2) biases heavily towards the Peak Gained and All Non-Monotonic shape classes, and the Right Shift peak-shift class. By contrast, the distributions for the thermodynamic constants K (Two Component Systems Supplementary Information) all form the same distribution, and so do not determine SR curve (energy dependent) behaviour.

dominate. Both of these classes are non-monotonic at low energy levels; In contrast, if κ_{f3} is low, all-monotonic (green) and peak-gained (orange) solutions dominate, which demonstrate monotonic behaviour at low energy.

We can perform a similar analysis for the peak-shift classes (Figure 3.11b). This time, when the initial HK phosphorylation rate κ_{ap1} is low, the SR curve peak tends to move to the right as energy increases; that is, more stimulus is required to maximise the response when energy is abundant. Strangely, the converse is true for κ_{a1} (row 1, column 1), the rate of signal activation for the same pathway. κ_{f2} and κ_{f3} again exhibit similarly mirrored behaviour as before.

More investigation is required into how these parameters influence the SR curve, and which parameter combinations determine the unorthodox TCS signalling behaviour.

3.3 Discussion

3.3.1 Key insights from the TCS model

Here we have taken an energy-based approach to modelling TCSs, and we have shown how this thermodynamic perspective leads to new and unexpected insights:

- (1) Energy limits the maximum signal expression by the response curve.
- (2) Energy affects the TCS temporal response curve sensitivity.
- (3) Noisy signals are sensitive to energy, but some systems are still able to filter out noise effectively
- (4) Unorthodox TCSs can detect for specific stimulus levels; this behaviour is energy dependent.

We find that energy availability affects the signal strength of a TCS (Figure 3.3). This has implications for how well a cell can function in challenging, low energy environments. A lower signal response from an equivalent stimulus level means that the cell will not respond as "confidently" or rapidly as it should. A signal that is too weak to be detected may result in a false negative stimulus detection. This may not be detrimental in some cases, as the activities of the TCSs in a cell could be triaged according to need (for example, TCSs related to cell structure or metabolism may be prioritised [73]).

We find that the temporal signal sensitivity of a TCS increases as available energy decreases, particularly for orthodox TCSs (Figure 3.4). This implies that for some TCSs, the signalling

mechanism will become more selective and behave more like a biological switch as ATP levels are reduced. A more sensitive system takes longer to initially accept a signal, but will produce a stronger, more pronounced response once the threshold has been reached. Such a system will be prone to less false positives, and will more likely respond to a true stimulus. This response behaviour could be useful for a system which triggers energy-intensive operations within the cell. Bacteria have been shown to adjust protein abundance in starvation or other hostile conditions [73]. In low energy environments, where food is scarce or energy must be rationed carefully, a cell may not want to waste energy following a false positive signal for food.

Our models demonstrate that unorthodox TCSs tend to filter noisy stimuli better than orthodox TCSs (Figure 3.5-Figure 3.7). The signal variance and the mean signal reduction are lower for the unorthodox variants than for the orthodox variants. We also see that, while higher energy levels amplify noise in the orthodox system, the unorthodox system is more robust to this noise amplification. Scenarios where accurate signal detection is important may justify the extra operational cost of unorthodox TCSs. In *E. coli*, for example, the unorthodox TCS responsible for anaerobic metabolism TorS/TorR activates in the absence of oxygen (Table A.5, see also [72]). An accurate signal is important for the cell so as to not trigger aerobic energy pathways without oxygen, and vice versa.

This filtering feature is due to the structural differences between the variants. For the orthodox system, there are only three reactions that separate the input signal from the output response (Table A.1). The noise amplitude and variance do not change noticeably; so all the noise is passed on to the output signal (Figure 3.7, top rows). For the unorthodox system, there is a sequence of steps, so rapid changes in input do not cause a major increase in flux across the system (Table A.4). Our hypothesis is that the multiple steps delay the signal propagation enough so that random fluctuations do not have time to pass through the system. As a result, flux in the intermediate reactions is relatively smaller, and the output signal is less noisy (Figure 3.7, bottom rows). This noise-reducing property of the unorthodox TCS aligns with previous modelling studies [87, 88].

How well a system can filter noise appears to be energy dependent. More energy in the system means that there is more activity (i.e. higher reaction flux) in intermediate reactions. With additional ATP, noisy variations in the signal are processed and propagated through much more quickly, ultimately leading to noisier outputs (Figure 3.6). The unorthodox system, with its multiple signalling pathways, is able to mitigate this noise even in high-energy contexts (Figure 3.7, bottom rows). To our knowledge, no other study has investigated this effect.

An interesting observation in the unorthodox TCS models is the appearance of peaks in the signal response curves (Figure 3.8); this effectively means that such systems can act as band-pass filters [100]. The maximum response expression level is not found when the signal is maximised; rather, it is found at an intermediate stimulus value. An optimal response by the cell will involve moving towards this intermediate stimulus level (or alternatively, a strong response to move away from this level). We therefore suggest that these signalling systems are designed to select for a particular signal expression level. This behaviour is useful for a cell if, for example, the TCS is detecting a quantity such as pH via the concentration of H^+ ions [77]. In some cases, it may not make sense to search for "minimum" or "maximum" pH, rather a particular pH band would be desirable. This non-monotonic signal exists in similar biological contexts, such as TCS models extended with redox regulation [91], or drug dose-response curves in pharmacology [101]. It is of interest to investigate how different signalling structures give rise to similar non-monotonic behaviours in the future.

Furthermore, some unorthodox TCS exhibit the unusual property of a disappearing signalling peak as energy levels changed (Figure 3.9). This means that in high (or low) energy environments, the solution behaves monotonically, where a 'switch-like' decision is required; but in low (or high) energy environments, the solution is fine-tuned to a desired stimulus level, as above. In other words, at one energy level the cell selects for a particular stimulus, but at a different energy level a clear decision is made once a stimulus threshold is crossed.

We found that in some cases, peaks in the SR curve can shift horizontally with a change in energy (Figure 3.10). Much like a radio tuner honing in on a particular bandwidth, so too could such a TCS adjust the signal it is selecting for based on energy availability. This leads to greater adaptability available to a cell than previously thought. In the pH example, it may be the case that the cell reacts to different pH under different energy levels. Cells can adapt to hostile environments [73]; perhaps in low nutrient environments the cell seeks out slightly more acidic or basic conditions compared to an environment abundant in nutrient.

These exotic behaviours are not rare occurrences in our analyses (Table 3.1). We can thus be reasonably confident that in some bacteria unorthodox TCSs exist whose signalling behaviour is non-monotonic. ATP-modulating SR curves are also not uncommon. This implies that for some cells the signalling behaviour could be adapted to new environments depending on the availability of ATP. These features highlight the wide array of adaptable signalling measures at the disposal of a cell, all within unorthodox TCS variants.

While a full study of a causal link between parameter selection and SR curve behaviour is beyond the scope of this investigation, our analysis provides some interesting insights (Figure 3.11a): all the thermodynamics constants of the species (the K 's) for each class follow the same distribution. The differences in parameter distributions only appear for the reaction rates (the κ 's). We see then that the SR behaviour of the TCSs is determined not by

species-specific parameters of the molecules involved, but by the reaction rate parameters of the reaction system. We also detect a biological connection between the model and parameter and the behaviour. The autophosphorylation reaction rate parameter κ_{ap1} , which determines binding rate of ATP to an empty HK, greatly influences whether an unorthodox TCS will exhibit signal selecting behaviour. If this reaction occurs too quickly relative to the other reactions, signal selection is unlikely to occur.

From this parameter distribution analysis, we can conclude that it is not an individual molecular species that determines unusual behaviour; rather, it is the system-wide behaviour that matters. This supports the idea that a systems biology approach to modelling is required to understand the mechanisms; studying the structure of individual species alone will not be enough.

3.3.2 ATP and Energy Modelling

Our study focuses on the energy requirements for TCS in bacteria. ATP is the ubiquitous source of chemical energy in biochemical reactions, so we have used the concentration of ATP as our proxy measure of energy availability for TCS. We run our simulations across an ATP gradient from 'starvation' to 'abundance'. That is, when ATP levels are low, we interpret this as the bacteria being starved of a food source. Even in starvation or hostile conditions, cells can still continue to operate as normal [73].

The exact concentration of ATP can fluctuate greatly within a single population, even if the average concentration is stable. Yaginuma *et al.* [74] show that the distribution of ATP in *E. coli* follows a heavy-tailed right-skewed distribution. While the average concentration was 1.54mM, the actual values ranged from 14mM to 0mM (or negligible amounts). They theorise that "the diversity in the level of ATP may benefit the population by enabling individual cells to adopt different strategies under severe conditions" [74].

An alternative interpretation of "starvation conditions" is to imagine the amount of ATP allocated to the particular TCS in triage; certain signalling systems may not be as important in these scenarios, such as TCSs used to determine whether the cell should divide. Under this interpretation, we are really modelling the concentration of local ATP allocated or available for a particular TCS system in a particular scenario. As some of our results show (e.g. [Figure 3.9-Figure 3.10](#)), there may be additionally benefits to the cell when less ATP is allocated to some TCS.

3.3.3 Synthetic biology

We would ultimately like to link TCS structural properties (e.g. size and complexity) to their function or purpose (e.g. temperature sensing or cell motility). The range of signalling behaviour and the adaptability of the TCS provides us with much scope for synthetic biology design [90, 93]. There are many parameters available in the design of a TCS to control the behaviour of the system (9 for orthodox, 35 for unorthodox). It may be possible that by adjusting the parameter value for a single component we can select which kind of behaviour we want in our cell: adaptable sensitivity, energy modulation, noise filtering, fine-tuned signal selection, signal peak formation, and signal peak tuning. Further investigation is needed to determine precisely how parameter choices cause these TCS properties. It is likely that underlying meta-parameters (e.g. the ratio of reaction rates) determine TCS behaviour and that we can use qualitative approaches to help in delineating different regimes of TCS behaviour [102].

Reaction rate parameters are notoriously difficult to determine from experimental observations [103] and depend on environmental conditions such as temperature and pressure. Thermodynamic constants, by contrast, are connected with the molecular species and can be transferred across to other mathematical models. It is possible to derive reaction rates from kinetic rates found in literature, provided some parameter values are known already (e.g. [33, 104]). Future work will focus on measuring reaction rates to determine whether unorthodox TCSs have observable signalling peaks or shifts in peaks.

Our study of energy-dependent effects of TCS in bacteria reveal novel insights into the capabilities of cellular signalling systems. These findings generate new functional hypotheses that are experimentally testable [105]. We suggest that future experiments in TCS should investigate how signalling systems behave in low-energy environments; how signalling systems filter out noise; and how unorthodox TCS adapt their signalling behaviour.

3.4 Methods

We have modelled TCS dynamics with a system of ODEs, which is the modelling framework of choice for TCS [75, 78, 81, 84, 85, 87–89]. The chemical reactions used for our models are shown in Table 3.2. The full models for the orthodox and unorthodox TCS, the chemical reaction networks, ODE models, and parameter definitions, can be found in [Two Component Systems Supplementary Information](#).

Our chemical reaction system notation is based on the model described by Kim & Cho [88]. Unlike their study, we explicitly retain ATP in the autophosphorylation chemical reactions.

Note that the stimulus S is a generic input that may not be chemical (e.g. light, heat, envelope stress). We have treated S as such a stimulant which is ‘consumed’ by the signalling pathway. If the stimulant were a chemical species, the model would need to be modified so that the stimulant is regenerated during the cycle to ensure mass balance; we do not consider this case here.

Since heat is a possible stimulus for TCS, it is plausible for the heat generated by the chemical reactions to indirectly influence the signal. We assume that the thermal energy released is negligible compared to the cell environment heat bath, and so we do not model this here.

ATP enters the system through the autophosphorylation of HK. Though included in the reaction system, this is usually dropped from the model, since it is not considered rate limiting [87, 88]. Instead of assuming an unlimited concentration of ATP, we assume a finite (yet constant) supply of ATP. In order to maintain mass and energy conservation laws, the reaction system includes ADP and Pi (inorganic phosphate group) as well. For thermodynamic feasibility, we also require all chemical reactions to be reversible.

For the degradation reactions in the unorthodox TCS model, we make the biological assumption that phosphoryl unbinding only occurs in the second binding site. This assumption is also made by Kim & Cho [88]. This means we have reactions where HK8 degrades to HK6+Pi, for example, but not to HK7 or HK5. (Recall the labelling of phosphorylated states in [Figure 3.2.](#))

TABLE 3.2: Reaction systems for the TCS models used in this study, based on those used in Kim and Cho [88]. Parallels exist between the reaction types of the orthodox and unorthodox variants, except for the forward phosphorelay, which only applies to the latter. The species S , ATP , ADP , and Pi are modelled as chemostats (constant concentration) and so function as parameters for the reaction rate constants.

Reaction Name	Orthodox TCS Reactions	Unorthodox TCS Reactions
Signal Activation	$HK + S \xrightleftharpoons{K_a} HKS$	$HK1 + S \xrightleftharpoons{K_{a1}} HK1S$ $HK3 + S \xrightleftharpoons{K_{a2}} HK3S$ $HK4 + S \xrightleftharpoons{K_{a3}} HK4S$ $HK7 + S \xrightleftharpoons{K_{a4}} HK7S$
Autophosphorylation	$HKS + ATP \xrightleftharpoons{K_{ap}} HKP + ADP$	$HK1S + ATP \xrightleftharpoons{K_{ap1}} HK2 + ADP$ $HK3S + ATP \xrightleftharpoons{K_{ap2}} HK5 + ADP$ $HK4S + ATP \xrightleftharpoons{K_{ap3}} HK6 + ADP$ $HK7S + ATP \xrightleftharpoons{K_{ap4}} HK8 + ADP$
Forward Phosphorelay	n/a	$HK2 \xrightleftharpoons{K_{f1}} HK3$ $HK3 \xrightleftharpoons{K_{f2}} HK4$ $HK5 \xrightleftharpoons{K_{f3}} HK6$ $HK6 \xrightleftharpoons{K_{f4}} HK7$
Phosphotransfer	$HKP + RR \xrightleftharpoons{K_{pt}} HK + RRP$	$HK4 + RR \xrightleftharpoons{K_{pt1}} HK1 + RRP$ $HK6 + RR \xrightleftharpoons{K_{pt2}} HK2 + RRP$ $HK7 + RR \xrightleftharpoons{K_{pt3}} HK3 + RRP$ $HK8 + RR \xrightleftharpoons{K_{pt4}} HK5 + RRP$
Degradation	$RRP \xrightleftharpoons{K_d} RR + Pi$	$HK8 \xrightleftharpoons{K_{d1}} HK6 + Pi$ $HK5 \xrightleftharpoons{K_{d2}} HK2 + Pi$ $HK3 \xrightleftharpoons{K_{d3}} HK1 + Pi$ $HK7 \xrightleftharpoons{K_{d4}} HK4 + Pi$ $RRP \xrightleftharpoons{K_{d5}} RR + Pi$

3.4.1 Bond Graphs

We represent our TCS chemical reaction networks using biochemical bond graphs. For an introduction to bond graphs, see [19, 20]; for applications in biochemistry, see [21, 23, 33, 104, 106].

Bond graphs represent physical systems as mathematical graphs, with nodes representing either system elements (*components*) or conservation laws, (*junctions*), and edges representing flows of energy, (*bonds*).

We can represent any chemical reaction network as a bond graph network. Chemical species are represented by one-port **C** components that track concentrations; reactions are represented by two-port **Re** components, where the two ports correspond to the reactants and products of the reaction (Note: since all reactions are reversible, the distinction between reactants and products may not always apply). Species which appear in multiple reactions connect to a **0**-junction; these junctions conserve the mass of a species. Species on the same side of a reaction (the reactants or products) are combined with a **1**-junction; these junctions conserve chemical potential μ and represent the combined potential energy for the reaction.

As an illustrative example, take the combined signal activation and autophosphorylation reactions for the orthodox TCS (Table 3.2).



This reaction subsystem is represented as a bond graph in Figure 3.12A. The species HK, HKS, HKP, ATP, and ADP each have a **C** component that stores the chemical potential energy of that species. The stimulus S uses a **Se** ('Source of effort') component that represents a constant supply of chemical effort.

As mentioned earlier, the stimulus can be from a variety of physical sources. For simplicity and consistency, we treat the stimulus as a chemical signal with a molar concentration and so use a *chemical* effort source component. Other types of signals (light, heat, envelope stress, etc.) can always be transformed into an equivalent chemical signal with a transformer **TF** or gyrator **GY** component.

Re:Activate and **Re:Autophos** are the reaction events in the system. These spend chemical potential energy and describe flow by a constitutive relation (see Model Derivation). HK and S combine to form HKS, so they are connected with a **1**-junction. HKS appears in both reactions and so is split with a **0**-junction. Each chemical species appears only once, in order to ensure they are conserved across all reactions in the system.

This structure can be iteratively generated for our TCS models from Table 3.2. Figure 3.12A-D show the bond graphs for each reaction subsystem. These graphs become **BG** components used to build our hierarchical TCS models Figure 3.12E-F. Species in the lower level bond graphs which appear in other bond graphs are exposed to the higher level. The chemostat species ATP, ADP, Pi, and S are all held at constant concentrations. Note that the unorthodox TCS model uses multiple instances of the same reaction submodel. For this study, we generated our bond graph models using the Python package BondGraphTools [54].

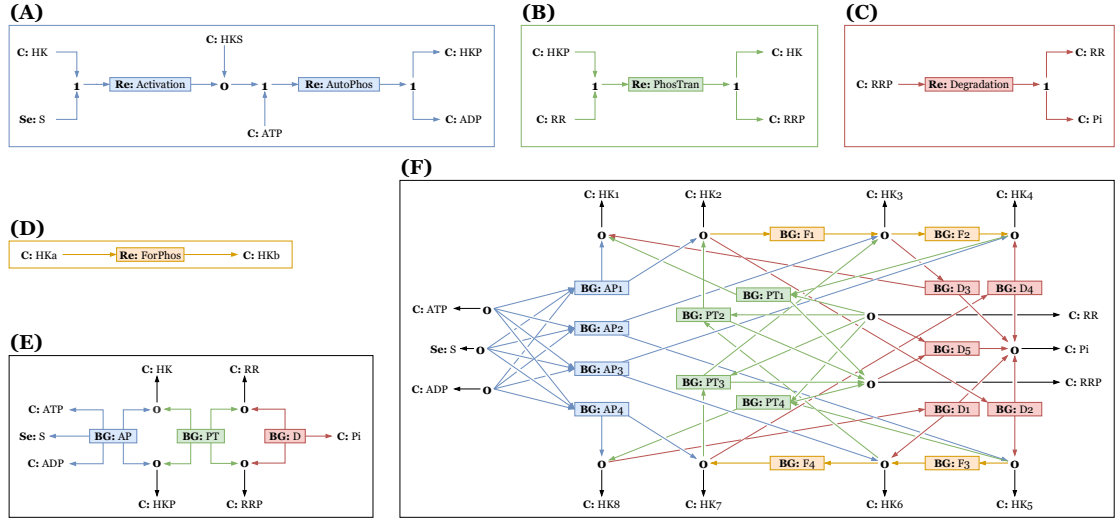


FIGURE 3.12: Bond graph representations of all TCS reactions and models: **(A)** Signal Activation and Autophosphorylation **(B)** Phosphotransfer **(C)** Degradation **(D)** Forward Phosphorelay **(E)** Orthodox TCS **(F)** Unorthodox TCS. Bond graphs **(A)**-**(D)** form the components used in the hierarchical TCS models **(E)**-**(F)**.

Model Derivation

Energy flow is described using the generalised units of *effort* [Joules/ x] and *flow* [x /second], where x is a tracked quantity for a particular physical system. For example, if x is the charge q in Coulombs, then the effort is [Joules/ x] $\equiv$ [Volts] and the flow is [q /second] $\equiv$ [Amperes]. The relationship between the effort and flow in a component (e.g. an electrical resistor) is described by a constitutive relation (e.g. $V = IR$). In addition to constitutive relations we have conservation laws that conserve mass, momentum, charge, or a particular species (e.g. Kirchhoff's Current Law).

For biochemical bond graphs, the effort variable is the chemical potential μ [J/mol]. Each reaction species i has its own chemical potential, defined by

$$\mu_i = RT \ln(K_i x_i) \quad (3.3)$$

where R [J/mol/K] is the universal gas constant, T [K] is the absolute temperature, K_i [mol⁻¹] is the species specific thermodynamic constant, and x_i [mol] is the concentration of species i .

The flow variable is the reaction flux ν [mol/s], which is the reaction rate for each reaction in the model. Assuming mass action kinetics, the flux through each reaction j is defined as

$$\nu_j = \kappa_j \left[e^{A_j^f/RT} - e^{A_j^r/RT} \right] \quad (3.4)$$

where κ_j [mol/s] is the molar reaction rate for reaction j , and A_j^f, A_j^r [J/mol] are the sums of the chemical potentials of the reactants (forward affinity) or products (reverse affinity), respectively.

$$A^f = \sum_f \mu_f \quad A^r = \sum_r \mu_r \quad (3.5)$$

where f and r are the forward and reverse reactants, respectively.

To derive mass action kinetics, we can substitute [Equation 3.3](#) and [Equation 3.5](#) into [Equation 3.4](#):

$$\nu_j = \kappa_j \left[e^{(\sum_f \mu_f)/RT} - e^{(\sum_r \mu_r)/RT} \right] \quad (3.6a)$$

$$= \kappa_j \left[e^{\sum_f \ln(K_f x_f)} - e^{\sum_r \ln(K_r x_r)} \right] \quad (3.6b)$$

$$= \kappa_j \left[\prod_f K_f x_f - \prod_r K_r x_r \right] \quad (3.6c)$$

When [Equation 3.6c](#) is expanded, the reaction rates κ_j and species' thermodynamic constants K_i can be combined to form the familiar kinetic rate constants of mass action kinetics.

Note that κ determines only the rate of a reaction, not the direction. The reaction flux direction is determined by whichever is greater of the product of $K_f x_f$ or $K_r x_r$. The species constants and concentrations themselves determine the reaction direction. This is in contrast to standard mass action kinetic models, where the forward and backward reaction rates can be selected independently of one another.

The concentration of each species over time is

$$\frac{dx_i}{dt} = \sum_{r \in \mathcal{R}_i} \nu_r \quad (3.7)$$

where $\mathcal{R}_i$ is the set of reactions that involve species x_i . With these equations and the bond graph structure described in [Figure 3.12](#) we can algorithmically generate a system of nonlinear ODEs. This method is used to generate our TCS ODE models ([Two Component Systems Supplementary Information](#)).

3.4.2 Parameter Selection

In order to explore the family of behaviours observed in TCS, we have performed a parameter sensitivity study on the model parameters. We pre-generated 10,000 parameter sets for each model in order to capture likely (normalised) biological parameter ranges. These parameters

sets are reused for each model when comparing similar simulations across ATP concentrations or other control variable.

For convenience, all initial conditions, parameters, and solution values used in our simulations are normalised and/or chosen as abstract quantities rather than experimentally recorded values. The universal gas constant R and temperature T are both normalised so that $RT = 1$ in Equation 3.3 and Equation 3.4. For our purposes, the use of abstract parameters is sufficient to compare the overall structure and behaviour of the two models under different conditions.

The sampling ranges for the parameters in the models can be found in Table A.2 and Table A.4. The thermodynamic constants K for the core component species (HK, RR, and their variants) are sampled on a $\log_{10}$ scale from 0.1 to 10. The reaction rates κ are sampled in the same way:

$$K = 10^p \quad \kappa = 10^p \quad p \sim \text{Uniform}[-1, 1] \quad (3.8)$$

Although exact parameter ranges for most TCS parameters are unknown, we can still use some physical and biological intuition to constrain our models. This limits the parameter space to exclude physiologically unrealistic parameter combinations. This helps refine our parameter search by filtering out the unrealistic or trivial TCS parameter sets, such as those which would flow backwards or not function. We do this with two changes.

First, due to the function of ATP as a reaction activator, we know that it will have a larger chemical potential than any of the other species involved. Likewise, ADP and Pi do not themselves drive reactions forward, and so will have relatively lower chemical potentials. If this were not the case, then we would find our reactions primarily travelling in the opposite direction. Since we are dealing in primarily abstract quantities, we only focus on the relative magnitudes of parameters. Therefore, in our models we have used $K_{\text{ATP}} = 100$, $K_{\text{ADP}} = 0.1$, and $K_{\text{Pi}} = 0.001$.

Second, we know *a priori* the dominant direction of all reactions. These are the reactions presented as irreversible in earlier TCS studies. We do not want to set all the reactions to be irreversible, as that would break the thermodynamic requirements and limit reaction possibilities. We can instead sample chemical potentials so that forward reactions are *favoured* rather than eliminating reverse reactions.

We have achieved this by ordering the sampled thermodynamic constants K before assigning them to specific species. In other words, for a model with N species, we sample N parameters, sort in descending order, and assign to the species in order of appearance in the

reaction system (Equation 3.9).

$$K_{HK} > K_{HKS} > K_{HKP} > K_{RR} > K_{RRP} \text{ (Orthodox)} \quad (3.9)$$

$$K_{HK1} > \dots > K_{HK8} > K_{RR} > K_{RRP} \text{ (Unorthodox)} \quad (3.10)$$

This method does not mean all the reactions are now irreversible; rather, it biases the reactions in a meaningful direction, while still allowing reverse reactions to occur. Note that there are multiple ways to order the unorthodox HK states (HK1 ...HK8), since their order of appearance in the system is not strictly linear. Nevertheless, this is a fairly natural ordering that achieves the desired effect.

3.4.3 Initial Conditions

The initial conditions for the core TCS species in all simulations are $[RR]_0 = 10$, $[HK]_0 = 1$ (or $[HK1]_0 = 1$ for the unorthodox model), and $[HKx] = 0$ for all other HK states x . Likewise, we set $[ADP]_0 = [Pi]_0 = 0$.

We used five levels of available ATP concentration in our model:

$$[ATP]_0 \in \{10^{-1}, 10^0, 10^1, 10^2, 10^3\} \quad (3.11)$$

This gave us an even spread of energy levels across a $\log_{10}$ scale.

For the first set of simulations (Figure 3.3, Figure 3.4) the stimulus input function was set to a constant value $S(t) = S_0 = 1$. For the SR curve simulations (Figure 3.8, Figure 3.9, Figure 3.10) we used 121,000 $\log_{10}$ -spaced values of $S_0 \in [10^{-4}, 10^4]$.

When modelling a noisy stimulus (Figure 3.5, Figure 3.6, Figure 3.7), we generate the noise with a piece-wise function $s_k(t)$ defined by:

$$s_k(t) = \begin{cases} S_0 + \alpha \varepsilon_k & \text{if } (k-1)\Delta t \leq t < k\Delta t \\ 0 & \text{otherwise} \end{cases} \quad (3.12)$$

where the noise $\varepsilon_k \sim \mathcal{N}(0, 1)$ is distributed normally, $\alpha = 0.5$ is the magnitude of the noise, $S_0 = 1$ is the constant baseline, and $\Delta t = 1$ is the length of interval k . We then define the stimulus input function as:

$$S(t) = \sum_{k=1}^N \max(s_k(t), 0) \quad (3.13)$$

where $N = 200$ is the total number of intervals. We include the max function in the summation so that the input is always positive. The noise terms ε_k are all pre-generated so that they can be reused, and direct comparisons can be made across all simulations.

3.4.4 Simulation

For each TCS model, we ran 10,000 simulations with pre-generated randomly sampled parameters at the five determined ATP levels and a constant stimulus input (100,000 in total). For the noisy stimulus scenario, we ran 500 simulations per model per ATP level (5,000 in total). We solved our ODE systems numerically using the Differential Equations library in Julia [107]. The simulations were all run for $t \in [0 \ 200]$, which was found to be sufficient to capture all steady state behaviour for the parameter ranges studied.

Each simulation output included the RRP concentration over time (response curve). To aggregate the response curves we first normalise by the maximum response value (to get an RRP fraction between 0 and 1), and then take the average across all parameter sets for each curve group (grouped by model, ATP, and stimulant). This is done for example in the results in [Figure 3.3](#).

The noise filtering performance measures ([Figure 3.6](#)) were calculated for $t \in [100 \ 200]$. We used a generous warm-up period (around half the simulation time) to allow for the wide range of possible simulations to reach steady state.

3.5 Supplementary Information

Supplementary Information, including model definitions and additional figures, can be found in [Appendix A](#).

Chapter 4

BondGraphs.jl

BondGraphs.jl [2] is a Julia implementation of the bond graph framework, built upon ModelingToolkit.jl and Graphs.jl. The generated models are composable, hierarchical, and compatible with the Julia modelling ecosystem. This chapter introduces the package and demonstrates the modelling capabilities across several scientific domains.

In order to accelerate the development of whole cell models, it is necessary to improve on existing computational tools. Existing bond graph tools tend to be part of proprietary modelling suites, e.g. MATLAB/Simulink [108] or Mathematica [109]. One notable exception is the Python package BondGraphTools [54]. This is a useful open-source package that includes a biochemical reaction module, but is constrained by Python’s limited scalability and abstraction (see [110]).

This has led to the creation of BondGraphs.jl, a new bond graphs modelling package written in pure Julia. Julia is a scientific programming language designed to solve the “two-language problem”; it is both easy to read and write code for, without sacrificing speed and scalability. Julia has been highlighted as particularly useful in systems biology [110].

Bondgraphs.jl is designed to interface with ModelingToolkit.jl [42], which connects it with the wider SciML modelling ecosystem automatically. The SciML ecosystem provides access to, for example, differential equation solvers [107], symbolic model construction [42, 111], parameter estimation [112], and APIs for systems biology model databases like CellML and SBML (e.g., with SBMLToolkit.jl [113]).

Since its creation, Bondgraphs.jl has begun to see use in bond graph research [48]. New package features and bug fix patches are planned for future releases.

This main content of this chapter was published in *Bioinformatics* journal as an application note: “BondGraphs.Jl: Composable Energy-Based Modelling in Systems Biology”[2]. Supplementary Information (parameters, models, code) for the application note are included in [Appendix B](#). Extended examples for ion transport [54], SERCA pump [114], and *Saccharomyces cerevisiae* metabolic network [115], are also included in [Appendix B](#).

A copy of the documentation for version `v0.4.0` of the package is included in [Appendix C](#). These pages are compiled from the same source files used for the online version. The package documentation [3] contains a brief *Introduction*, a *Getting Started* tutorial on how to use the package, *Examples* models showcasing features, including biochemical bond graphs, and an *API Reference*. Code repository is available on GitHub¹, which includes online documentation². BondGraphs.jl can be installed directly from the official Julia registry using the package interface:

```
using Pkg; Pkg.add("BondGraphs")
using BondGraphs
```

¹<https://github.com/jedforrest/BondGraphs.jl>

²<https://jedforrest.github.io/BondGraphs.jl/stable>

BondGraphs.jl: Composable energy-based modelling in systems biology

BondGraphs.jl is a Julia implementation of bond graphs. Bond graphs provide a modelling framework that describes energy flow through a physical system and by construction enforce thermodynamic constraints. The framework is widely used in engineering and has recently been shown to be a powerful approach for modelling biology. Models are mutable, hierarchical, multiscale, multiphysics, and BondGraphs.jl is compatible with the Julia modelling ecosystem.

BondGraphs.jl is freely available under the MIT license. Source code and documentation can be found at <https://github.com/jedforrest/BondGraphs.jl>. Supplementary data are available at *Bioinformatics* online.

4.1 Introduction

The last decade has seen an explosion of experimental methods that simultaneously measure many aspects of biological processes at multiple temporal and spatial scales. Extracting biological mechanisms from this big-data explosion requires modular, interoperable and computationally efficient software in systems biology [46].

Bond graph modelling is a promising approach to creating computational models for large-scale biological data analysis. This mathematical framework has been used to model many biological systems, as well as mechanical, electrical, and chemical engineering systems [20, 24].

Bond graph models are particularly advantageous for systems biologists and mathematical modellers. Bond graph models of biological networks are biophysics-based and are scalable

to large networks of interactions. They satisfy the laws of thermodynamics implicitly, and therefore they enable efficient construction of models that couple interactions between different sub-systems within a biological process. Recent biological bond graph models include mitochondrial respiration [106], two component bacterial systems [1], and vascular blood flow [49].

Bond graphs are most useful when the system being modelled satisfies one or more of the following: (i) different types of physics are involved (such as an electrochemical system); (ii) physical consistency is required on a large scale; (iii) the system exhibits a modular or hierarchical structure; (iv) model compartments are usable in other models. Thus, bond graphs are a promising approach to improving model reusability and model composition in systems biology.

Here we present *BondGraphs.jl*, a Julia implementation of the bond graph modelling framework. The package enables a wide range of analyses to be conducted on physical systems, including the derivation of differential equations and simulation. Graphs and equations are built with a high-level symbolic interface that integrates closely with the Julia modelling ecosystem.

4.2 Features

The most pertinent features are listed below. We refer to the documentation (<https://jedforest.github.io/BondGraphs.jl/stable>) for a full list of features and examples, and to the supplementary material for the basic principles of bond graph modelling.

4.2.1 Graphical construction and modification of models

One of the strengths of bond graphs is that they are semantic representations of models, much like an electrical circuit, free body diagram, or reaction network. Physical systems are represented as graphs and can be manipulated in many of the same ways.

BondGraphs.jl constructs a *BondGraph* object using *Components* and *Junctions* (graph vertices) and *Bonds* (graph edges).

Each *Component* encodes a set of acausal constitutive equations that represent a physical element, event, or law; for example Ohm's law, or mass action kinetics. *Junctions* are vertices that describe how components relate to each other according to conservation laws; for example, Kirchhoff's Voltage Law or mass balance in biochemistry. *Bonds* are the graph

edges that connect the vertices and represent energy flow or transfer between parts of the system.

Bond graphs can either be constructed one component at a time and connected together with bonds (using `add_node!` and `connect!`), or generated automatically using specialised algorithms (see [subsection 4.2.3](#)).

Extra functions are included for modifying and composing multiple bond graph objects. Vertices (components or junctions) can be swapped in for another (`swap!`). This is useful for rapidly creating bond graphs of systems that are structurally similar but with modified equations, such as a different constitutive relation for a spring or resistor. For example, in a biochemical context, one could replace standard mass-action rate laws with Michaelis-Menten rate laws. In fact, any arbitrary rate law can be used, so long as it is thermodynamically consistent [23].

Vertices can also be inserted between two already connected components (`insert_node!`), which is useful for adding a conservation law (with a junction) or a transformation to the variables.

When combining two separate bond graphs, there are two main approaches. The first is to take the two disconnected graphs and merge nodes representing the same object (`merge_nodes!`). This way the bond graph retains a 'flat' structure and system elements are not duplicated. Alternatively, a bond graph can be nested inside a `BondGraphNode` component, which functions like an ordinary component but contains another bond graph. This creates a hierarchical bond graph that is a powerful tool for model organisation [23].

These operations enable high-level semantic construction and modification of the model. Since all models are built and stored as a graph object, they can be traversed and manipulated using existing graph algorithms (see [subsection 4.2.4](#)). The bond graph plots in this paper, for example, use a graph layout plotting algorithm.

`BondGraphs.jl` includes a library of common bond graph component types. The default library includes a biochemical library for modelling chemical species and reaction networks. Additional custom components can be defined by the user and stored in the local library. See supplementary material for further details.

4.2.2 Symbolic generation of equations

Systems of differential equations are automatically generated from the bond graph. Stored equations are systematically composed using a computer algebra system (CAS) to automatically substitute, simplify, and differentiate equations. This produces a system of ordinary

differential equations (ODEs) or differential algebraic equations (DAEs). These equations can be solved numerically or analysed analytically using other packages in the Julia ecosystem ([subsection 4.2.4](#)). A modeller is not limited to differential equations, as they can derive other useful representations such as Hamiltonians, reaction fluxes, or power equations.

A user is free to specify parameter values at multiple stages during the modelling process. Parameter values can be set during initialisation using the constructor syntax, or modified later using Julia's indexing syntax.

```
Component(:C, "membrane"; C=1)
membrane.C = 1
```

The benefit of this approach is that the model building and equation solving steps are separated. Abstractions of models and repeated motifs can be stored and reused for different modelling contexts [20, 23]. The parameter values can be decided at the simulation stage, and the CAS simplification routines reduce numerical instability. With Julia, arbitrary functions can be used as control inputs, as long as they return a single real-valued output.

4.2.3 Generating models from chemical reaction networks

An important feature for system biologists is the chemical reaction network $\longrightarrow$ bond graph $\longrightarrow$ differential equation pipeline. BondGraphs.jl provides an interface for Catalyst.jl [111] which converts a reaction network into a bond graph. This enables rapid construction and analysis of models without needing to manipulate the graph structure or type equations directly (though this is still possible if desired). See [subsection 4.3.2](#) for an example. Existing models that were originally represented as chemical reaction networks can therefore be easily converted into bond graphs.

4.2.4 Integration with the Julia modelling ecosystem

Previous bond graph packages have largely been implemented through proprietary software languages such as MATLAB or Maple. More recently, the open source package BondGraphTools implemented the framework in Python [54]. However, since bond graphs are implemented as specialised Python objects, it is not possible to use or study the graph-theoretic properties of the model, or to easily integrate with other modelling or analysis libraries.

By contrast, BondGraphs.jl is built purely in Julia. Julia is computationally fast and scalable while retaining easy-to-read syntax. It is therefore, we would argue, better positioned

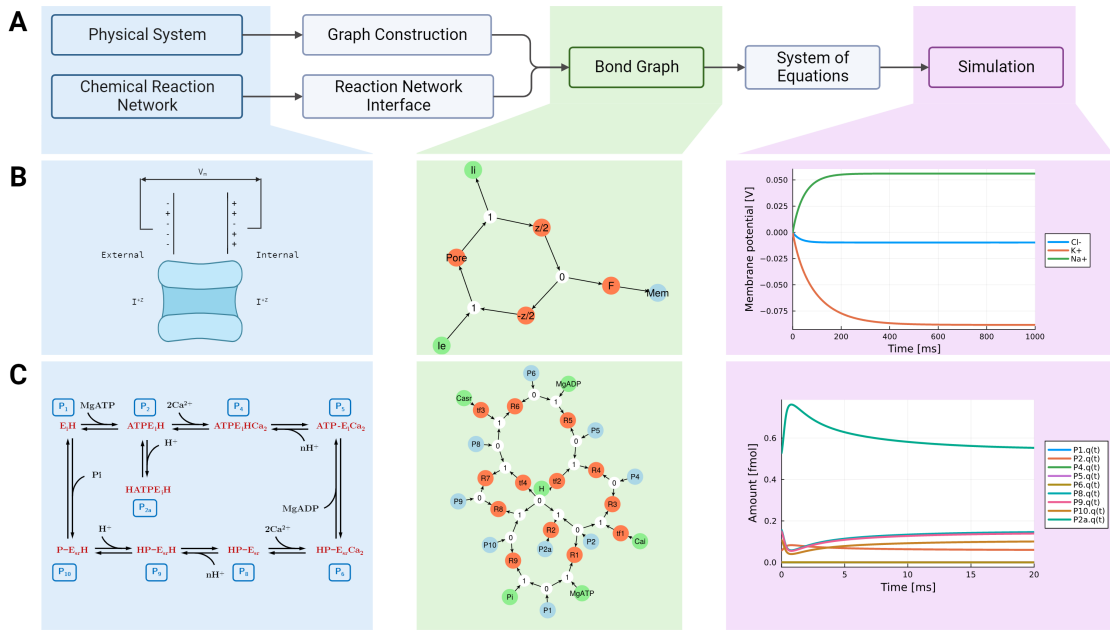


FIGURE 4.1: The BondGraphs.jl modelling process. **(A)** A physical system can be represented as a bond graph using either component-wise construction or, if it is a chemical reaction network, using the reaction network interface. BondGraphs.jl includes a plotting recipe for bond graphs: chemical species (blue), chemostats (green), reactions **Re** and transformers **TF** (orange), and junctions (white). The most common use case is to use the bond graphs to generate a symbolic system of equations that can be solved numerically. **(B)** The ion pore channel [54] demonstrates the modelling of a multiphysics (electrochemical) system. **(C)** The SERCA pump [33] demonstrates the use of the reaction network interface and scalability of Bond Graphs. Code used to generate the bond graphs and numerical solutions is available in the supplementary material. Diagrams created with BioRender.com.

to address the grand challenges of systems biology [110]. The data types and functionality of BondGraphs.jl are built upon widely used packages, which give users access to all their features. This also enables close integration with the Julia modelling ecosystem.

BondGraphs are AbstractGraphs as defined in Graphs.jl (<https://github.com/JuliaGraphs/Graphs.jl>). With this, we can use a library of graph operations, traversal algorithms, graph colouring, database storage, and plotting recipes (as seen in Figure 4.1). ODEs are generated and stored using ModelingToolkit.jl [42] and are solved with the parallelizable numerical solver library DifferentialEquations.jl [107]. Bridging BondGraphs and ModelingToolkit connects our bond graphs to many other useful libraries, including parameter estimation and optimisation, plotting libraries, reaction network interfaces (subsection 4.2.3), and standard model formatting and parsing (e.g. SBMLToolkit, part of ModelingToolkit, see supplementary material). This also makes the package code easier to maintain, and updates to other packages will automatically be available for use in BondGraphs.jl.

4.3 Results

Here we demonstrate the features of BondGraphs.jl through two biological examples from literature. These two examples highlight several important features of both bond graphs and BondGraphs.jl. A further large-scale model (281 species, 566 reactions) of a yeast metabolic network is included in the supplementary material. All code used to generate these plots and results can be found in the supplementary material.

4.3.1 Ion transport

Many biological systems in nature involve multiple processes described by different physics. Here we model an ion pore transport process, which couples electrical and chemical processes that drive the movement of ions across cell membranes [54]. The diagram, bond graph, and numerical solution are shown in Figure 4.1A.

Each vertex in the bond graph (middle column) represents a different physical element or law of the system. The membrane potential is an electrical capacitor (blue). The external and internal ion concentrations are constant sources of chemical energy (green). Orange vertices are two-port components that either 'transform' or 'dissipate' energy: the energy-dissipating ion transport reaction process (*pore*); the conversion of electrical to chemical potential (*F*); and the splitting of voltage and charge dependence across the two sides of the membrane ($z/2$ and $-z/2$). The junctions (**0** and **1**) are conservation laws: **0**-junctions represent equal effort (typically 0D) and **1**-junctions represent equal flow (1D or greater). In these examples, they are the conservation of chemical potential (**0**) and molar flow rate (**1**). Bonds (edges) are the flow of energy between components, with the arrowhead indicating sign convention.

Components are constructed with parameter values and initial conditions, however these can be determined later. All nodes are added to the bond graph and connected with bonds in a graph-like manner. Once the bond graph is made, it can easily be reused with different values for different ions. Solutions for three ion transporters (Na^+ , K^+ , Cl^-) are included in Figure 4.1A, right column. The steady-state solutions follow the well-known Nernst equations seen in electrophysiology [54].

4.3.2 SERCA pump

Figure 4.1 shows a bond graph construction of the Sarco/Endoplasmic Reticulum Ca^{2+} -ATPase (SERCA) pump, first presented in Tran *et al.* [114] and subsequently represented as a bond graph in Pan *et al.* [33]. Reaction network and parameter values are taken from these papers.

Figure 4.1B shows the reaction network schematic, bond graph, and numerical solution for species' concentrations over time.

The colour coding follows the same convention as the previous example: blue vertices store chemical energy for one species; green vertices are sources of chemical energy (chemostats); orange vertices are reactions; junctions are physical conservation laws.

The SERCA bond graph model is generated using the reaction network interface ([subsection 4.2.3](#)) with the following compact syntax:

```
using BondGraphs, Catalyst, Plots
rn_serca = @reaction_network SERCA begin
    (1, 1), P1 + MgATP <--> P2
    (1, 1), P2 + H <--> P2a
    (1, 1), P2 + 2Cai <--> P4
    (1, 1), P4 <--> P5 + 2H
    (1, 1), P5 <--> P6 + MgADP
    (1, 1), P6 <--> P8 + 2Casr
    (1, 1), P8 + 2H <--> P9
    (1, 1), P9 <--> P10 + H
    (1, 1), P10 <--> P1 + Pi
end
chemostats = ["MgATP", "MgADP", "Pi", "H", "Cai", "Casr"]
bg_serca = BondGraph(rn_serca; chemostats)
```

Note that the (1, 1) before each equation are placeholder reaction rates required for the catalyst interface.

This example highlights several features: (1) Model construction scales efficiently even for large complex reaction networks; (2) Bond graphs can be constructed from only the reactions network and interfaced at a high level without direct interaction with equations; (3) Parameter values and initial conditions are determined separately and can include control variable equations; the input concentration of Ca^{2+} is set as a nonlinear 'spike' function over time (see supplementary material for details).

4.4 Conclusion

BondGraphs.jl is a new powerful tool for large, multiphysics, biophysically constrained systems biology models. Bond graphs are a general framework beyond biology, however they

have specific benefits to the systems biologist: (i) Models are constructed abstractly without worrying about equation implementation; (ii) Governing equations are derived automatically; (iii) The energy-based framework ensures that generated models are physically and thermodynamically compliant; (iv) BondGraphs.jl naturally integrates with other state-of-the-art tools in the Julia modelling ecosystem.

4.5 Supplementary Information

The Supplementary Information (SI) submitted with the application note can be found in [Appendix B](#).

The published package includes full documentation accessible from the GitHub repository (<https://jedforrest.github.io/BondGraphs.jl/stable>).

The remaining documentation examples and API are included in [Appendix C](#).

Chapter 5

Stochastic Bond Graphs

This chapter extends the bond graph and energy-based modelling framework to incorporate stochasticity. This will enable bond graphs to model a wider range of phenomena, which is a significant and necessary step towards whole cell modelling. The groundwork is laid for producing stochastic differential equations with the bond graph approach, combining theory from statistical physics, thermodynamics, and probability distributions. The Brownian particle model and simple chemical reactions are used as illustrative examples.

5.1 Background

A complete understanding of cellular biology requires serious consideration of the stochastic forces acting on a cell. While noise is typically omitted from models for computational and mathematical simplicity, it has been shown that biological noise leads to non-trivial effects [51].

Many cellular processes are inherently stochastic. Examples in cell biology include: Brownian motion and random walks — the random motion of large particles in a thermal fluid [31]; stochastic gene expression which leads to ‘bursty’ production of proteins, modelled by the Telegraph model and its variations [116, 117]; chemical reaction networks [27]; molecular motors — ‘walking’ molecular machine that transport payloads across the cell [118]; population variability across single-cell genomics data [119]; cell fate decision making [120]; and cardiac cell ion channel gating mechanisms [121].

Thermodynamics and energy conservation principles can be the basis of physically plausible large-scale models [60]. However, it is not immediately clear how an energy-based approach can best incorporate stochasticity. Should such models conserve energy at every point in

time, or just in the mean-field approximation? How does noise propagate across different physical domains and scales? How are the laws of thermodynamics satisfied in the typically discrete models of gene expression?

Here I begin to investigate these questions, using simple stochastic bond graphs models as demonstrations. My goal is to see if bond graphs can faithfully represent the underlying stochastic processes of nature while still adhering to the laws of thermodynamics.

I begin by motivating why we want to study noise in systems biology. I then introduce the Brownian particle and Langevin equation as motivating examples for stochastic modelling in bond graphs. This leads to a definition of a stochastic resistive component for bond graphs. I then create the bond graph using this new stochastic component of the Brownian particle and a few slight variations.

Next, I address the issue of energy conservation and stochasticity. I achieve this by using thermal ports to define a thermal stochastic component, which connects the microscopic and macroscopic definitions of energy together.

In doing so, I find several benefits for bond graph modelling, including: (i) energy conservation of stochastic differential equations; (ii) a bridge between the microscopic and macroscopic thermodynamic scales; (iii) a non-dissipative, reversible **R**-type component; (iv) modelling probability distributions with the Fokker-Planck equation.

I then show how the stochastic component theory can be applied to chemical reaction networks with the Chemical Langevin Equation. Finally, I make progress towards defining probability distribution bond graphs with a proposed ‘probability’ energy domain. With further development, this may lead to a complete probability bond graph domain.

5.1.1 Stochasticity in cellular biology

At the molecular scale of biochemical reactions, the effects of noise on reactions with small molecule counts are much more significant.

Stochastic systems can exhibit behaviour not found in purely deterministic models: the mean-field average of a stochastic model may be different to its deterministic version; at low molecule counts, the variation in reaction fluxes can create qualitatively different outcomes and terminating states, e.g. the complete removal of a species from a system [122, 123].

Stochastic modelling is required when creating whole cell models and other cell-wide models. A deterministic whole cell model is unable to predict the behaviour and activity of a living cell. Stochastic models are, therefore, an integral part of existing whole cell models,

and have been identified as an essential component of necessity for future whole cell model projects [11, 13, 14].

I should also clarify what is meant by “noise” in this chapter. For all our models and simulations in this chapter where we include stochastic behaviour, we will talk of randomness at the thermodynamic level. This means we are not considering heterogeneity in a population of cells. By contrast, the probability arising from quantum physics is an intrinsic property of quantum mechanics [124].

Another type of ‘randomness’ in systems biology arises from parameter uncertainty [14]. This reflects uncertainty in the parameters, model definition, or data from experiments. These are uncertainties from a lack of information about the system, which can in theory be overcome with more better experiments and statistical tools. This randomness is not considered in this chapter.

In statistical mechanics, we consider both the trajectories of particular individual molecules over time, and the distribution of all the particles in the systems over time. The theory behind statistical mechanics allows us to switch between the ‘trajectory’ perspective and the ‘distribution’ perspective [125]. Both perspectives are useful. It is often easier to work with distributions of particle positions and velocities, but on other occasions the random variation between different instances can greatly affect the dynamics. Rare events happen at the tail-end of distributions, and it is helpful to quantify the probability of these events.

5.1.2 Stochasticity in bond graphs

Bond graphs are capable of modelling thermodynamic systems, chemical reaction networks, and molecular dynamics, all under the same framework [18, 20, 21, 24]. The energy abstraction means it is plausible for stochastic components to be included in the framework.

Existing approaches to stochastic modelling in bond graphs have done so with control functions or parameters (for example, in Simulink [52]). We lack however a theoretical basis for stochastic differential equations within bond graphs. In particular, it is not immediately clear how the energy-based approach can best incorporate stochasticity. Bond graphs work across scales, and physical domains, while conserving energy, mass, charge, and other similar state variables. A stochastic model implementation would need to still satisfy these conditions.

Much of bond graph theory can be understood from thermodynamic principles. The First Law (total energy is conserved) is already essential for bond graphs; it is often what is meant by “thermodynamic compliance”. The Second Law (entropy of an isolated system does not decrease) is less often enforced. However, it is especially useful for stochastic models in bond graphs as it introduces the idea of probability to energy modelling. Entropy can be defined as

the ensemble average over energy microstates, with each microstate having some probability. Since bond graphs are based in thermodynamics, a natural step is to emphasise entropy, and so bring in probability and stochasticity, into the bond graph framework.

I also address the issue of energy conservation in stochastic models in this chapter. In physics and biology, stochastic models are often driven by a thermal noise represented as a random process. Since the fluctuation is a random variable, care needs to be taken to ensure that conservation laws still hold.

From the perspective of energy modelling, this endless random fluctuation is an infinite pool of energy being supplied to the system. For example, consider the position x and of a particle initially at rest ($dx(0) = 0$) described by the stochastic equation $dx = \sigma dW$, where σ is the noise magnitude. At each time step, the particle will move randomly in either direction. On average the total displacement will be zero, but on a single run it will never truly be zero. The particle is effectively moving for “free” at each time step, which violates the first law of thermodynamics.

This is potentially a problem for the physical plausibility of stochastic bond graphs, unless additional conditions are met. I show in [section 5.5](#) that this is resolved by introducing the heat bath as an explicit part of the bond graph model, and to combine the dissipating forces and the fluctuating forces into a single component. This result come from the Fluctuation-Dissipation Theorem, which is discussed in [subsection 5.4.3](#).

5.2 Stochastic Theory

I will provide some preliminary definitions for stochastic modelling. For more information on stochastic modelling, see Jacobs [\[122\]](#) and Gardiner [\[123\]](#).

A dynamic system may be said to be ‘driven’ by a stochastic process. A commonly used stochastic process in physics is the *Wiener process* W_t . Mathematically, the Wiener process is a sequence of normally-distributed random numbers sampled at each infinitesimal time step.

The Wiener process (discrete) step is defined as

$$\Delta W_t \sim \mathcal{N}(0, \Delta t) \quad (5.1)$$

for a timestep Δt . Then taking the limit as $\Delta t \rightarrow 0$ we define the Wiener increment as

$$\lim_{\Delta t \rightarrow 0} \Delta W_t = dW_t \sim \mathcal{N}(0, dt) \quad (5.2)$$

In short, dW_t is a random variable sampled from a normal distribution with mean 0 and variance dt . We will often write this term as dW for brevity.

The derivative of W_t is undefined, but we can approximate this with a *white noise function*

$$\xi(t) \approx \frac{dW_t}{dt} \quad (5.3)$$

which has the auto-correlation function

$$\langle \xi(t) \xi(t + \tau) \rangle = \delta(\tau) \quad (5.4)$$

where $\delta(t)$ is the Dirac delta function [122]. The Dirac delta function (technically a distribution), is an infinitely narrow instantaneous impulse.

Stochastic differential equations

A useful property of Equation 5.2 is what's known as *Itô's Lemma*. It can be shown that if we square dW , we get

$$(dW)^2 = dt \quad (5.5)$$

In other words, dW is on the order of $(dt)^{1/2}$. We also have $\langle dW \rangle = 0$.

A consequence of Equation 5.5 is *Itô's formula*.

$$df(x) = \frac{\partial f}{\partial x} dx + \frac{1}{2} \frac{\partial^2 f}{\partial x^2} dx^2 \quad (5.6)$$

where $f(x)$ is some function or transformation of x . This formula is essentially the 'stochastic chain rule'. Unlike in standard calculus, we must expand derivatives to the second order, which may introduce an additional term.

Stochastic Differential Equations (SDE) are differential equations that include a noise term at each time step. An SDE for a random state variable x is defined as

$$dx = \lambda(x, t) dt + \sigma(x, t) dW \quad (5.7)$$

where λdt is the deterministic term and σdW is the noise term. If the noise term is constant ($\sigma(x, t) = \sigma_0$) then this is called *intrinsic noise*, otherwise it is called *extrinsic noise*. If $\sigma = 0$, then we return to having a deterministic ODE.

Equation 5.7 defines the trajectory of a single instance of x over time. It is often more useful to model an *ensemble* of all trajectories as a probability distribution. If the ensemble follows a

well-defined distribution, then these can be characterised by the moments of x . For example, the expected value and variance of x are $E[x] = \langle x \rangle$ and $\text{Var}[x] = \langle x^2 \rangle - \langle x \rangle^2$.

5.3 Motivating example: Brownian particle

The Brownian particle model is the canonical example of a physically-defined stochastic system. In this section I will first define the Brownian particle model, then I will model the system with a bond graph. To do so, I define a new stochastic bond graph component.

Consider a large particle with non-negligible mass as it moves about in a thermal fluid comprised of many smaller molecules (Figure 5.1). We will refer to the larger particle as the *Brownian particle*, and the many smaller particles as *fluid molecules*. In biology, the Brownian particle may be a complex protein or other large molecule, such as an RNA or DNA strand [126, 127].

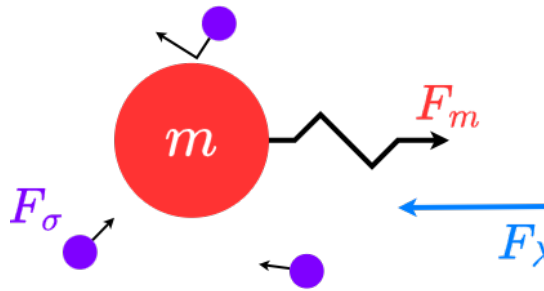


FIGURE 5.1: Diagram of a Brownian particle (red) in a thermal fluid (heat bath). A particle of mass m moves about in a heat bath and has a kinetic force F_m . Each collision of a fluid molecule (orange) with the particle produces some small change in momentum $\dot{p} \equiv F_\sigma$. The cumulative effect of the collisions with fluid molecule produces a net friction force F_λ . The heat bath is defined by the macroscopic state variables temperature T and entropy S . Since this is an NPT system, $dT = 0$, but entropy can still increase.

For this model, we assume all fluid particles are approximately spherical, including the massive particle. We also assume that the particle has a non-negligible mass, whereas the smaller particles are treated as part of the surrounding homogeneous fluid. The Brownian particle is larger than the rest of the fluid molecules, but small enough to still be affected by fluctuating forces. Strictly speaking, there may be other larger molecules in a biological context, but the smaller molecules often outnumber larger molecules by several magnitudes. For example, there are an estimated 2^{10} water molecules in *E. coli* [128].

We can assume that collision event (frequency and collision energy) with one small particle is independent of collision with a different small particle. Because of the differences in size and mass, it is also assumed that the Brownian particle will move on a slower timescale than

the rapidly colliding molecules in the fluid. It can be shown that the best way to represent these collisions is with a Wiener noise process (See e.g. [122]).

In cell biology, a common source of thermal fluctuations is the heat bath of the cell. This is the thermal energy stored in the surrounding fluid where the reaction takes place within the cell. This energy is stored in the movement of smaller molecules that comprise the fluid. These collide with larger molecules, causing them to move about randomly.

Each fluctuation of the forces acting on a particle is the result of some energy exchange with the environment. This might be through collisions with other particles occurring with some probability.

The Brownian particle collides with the many smaller particles that comprise the fluid. From these collisions, the particle will appear to move about randomly. The noise in the motion of the Brownian particle is the result of collisions with smaller molecules, which causes an exchange in momentum.

The collisions are momentum conserving, so some energy from the particle will be exchanged with the environment. Conversely, the particle will experience a drag force due to friction with the fluid, which dissipates heat back into the environment. (We will show later in [subsection 5.4.3](#) that these two energy exchanges are in fact equivalent.)

5.3.1 The Brownian particle bond graph

We wish to model the position x and velocity v of the Brownian particle. Three forces act on the particle: an inertial force F_m , a drag force F_λ , and a noise force F_σ ([Figure 5.1](#)). These forces correspond to standard bond graph components.

$$\mathbf{I} : F_m = m \frac{dv}{dt}$$

$$\mathbf{R} : F_\lambda = \lambda v$$

$$\mathbf{Se} : F_\sigma = \sigma \xi(t)$$

where $v(t)$ is the particle velocity, $x(t)$ is the (1D) particle position, m is the particle mass, λ is the drag coefficient, σ is the noise magnitude, and $\xi(t)$ is the white noise function.

To see why noise is a source of effort, recall that a change in momentum is the definition of a force ($F \equiv dp/dt$), so we can rewrite the fluctuating force experienced by the Brownian particle F_σ as

$$F_\sigma \equiv \sigma \xi(t) \tag{5.9}$$

where σ is the noise magnitude (determined later), and $\xi(t)$ is the white noise function (Equation 5.3).

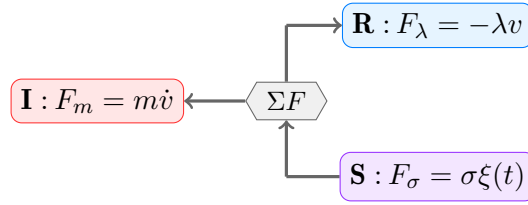


FIGURE 5.2: Brownian particle bond graph representing the Langevin equation $m \frac{dv}{dt} = -\lambda v + \sigma \xi(t)$. See section 2.2 for notational differences from typical bond graph diagrams.

Figure 5.2 shows the bond graph model of the Brownian particle. The velocity is shared across each component and the forces must all balance:

$$F_m + F_\lambda + F_\sigma = 0 \quad (5.10)$$

A 1-junction is therefore used to connect the components.

From this bond graph we obtain the Langevin equation (see Equation D.1a).

$$\frac{dx}{dt} = v \quad (5.11a)$$

$$m \frac{dv}{dt} = -\lambda v + \sigma \xi(t) \quad (5.11b)$$

In one sense, Figure 5.2 is a sufficient bond graph description of a stochastic system. The standard Se component which holds the noise function $\xi(t)$ can be interpreted as a time-varying effort source and sink. (It is also a sink because ΔW_t can be negative.) $\xi(t)$ is treated as a control function, similar to say [52].

Imagine we had sampled a sequence of *fixed* Dirac functions $\hat{\xi}(s) = \delta_0, \dots, \delta_s$. $\hat{\xi}$ is then interpreted as a time-varying forcing function. This is now a deterministic single trajectory of the stochastic system. This suggests a quick shortcut to turning *any* bond graph into a stochastic bond graph, since all bond graphs can add an arbitrary forcing term (Se component). The downside is that this approach only simulates one trajectory at a time. We lack the mathematical benefits (e.g. Ito calculus, arbitrary precision) and computational benefits (e.g. batch simulation runs) of stochastic analysis.

We can convert Equation 5.11a into an SDE by making the substitution $\frac{dx}{dt} = f(x) \implies dx = f(x) dt$:

$$dx = v dt \quad (5.12a)$$

$$m dv = -\lambda v dt + \sigma dW \quad (5.12b)$$

We have made the conversion $\xi(t) \approx \frac{dW}{dt} \rightarrow dW$ where $dW \sim \mathcal{N}(0, dt)$. (Technically, $\frac{dW}{dt}$ is undefined, but we are handwaving this with the approximation from [Equation 5.3](#).) We can also write these equations more succinctly in matrix form

$$d\mathbf{x} = \begin{bmatrix} 0 & 1 \\ 0 & \frac{\lambda}{m} \end{bmatrix} \mathbf{x} dt + \sigma \begin{bmatrix} 0 \\ \sigma \end{bmatrix} dW \quad (5.13)$$

where $\mathbf{x} = [x \ v]^\top$.

5.3.2 Analytical solutions

Derivations of the analytical solutions adapted from Jacobs [\[122\]](#) can be found in [Appendix D](#).

The Fluctuation-Dissipation Theorem states that the noise and drag coefficients are not independent. σ is in fact determined by λ . For the Brownian particle, the noise magnitude is

$$\sigma = \sqrt{2kT\lambda} \quad (5.14)$$

where k is the Boltzmann constant and T is the temperature.

[Equation 5.14](#) comes from the fact that the average kinetic energy $\langle E_k \rangle = \frac{1}{2}m\langle v^2 \rangle$ can be shown to be $k_B T/2$ per energy ‘mode’ or degree of freedom (e.g., 3 spatial degrees of freedom would have $\langle E_k \rangle = 3k_B T/2$) [\[67, 122\]](#).

Taking $\langle v^2 \rangle = \sigma^2/2m\lambda$ from [Equation D.9](#), we can solve for σ .

$$\frac{k_B T}{2} \equiv \frac{1}{2}m\langle v^2 \rangle = \frac{\sigma^2}{4\lambda} \quad (5.15a)$$

$$\implies \sigma = \sqrt{2k_B T \lambda} \quad (5.15b)$$

The full SDE equations are now

$$\begin{aligned} dx &= v dt \\ m dv &= -\lambda v dt + \sqrt{2k_B T \lambda} dW \end{aligned} \quad (5.16)$$

The general analytical solution for $v(t)$ is

$$v(t) = v_0 e^{-\lambda t/m} + \frac{1}{m} \sqrt{2k_B T \lambda} \int_0^t e^{-\lambda(t-s)/m} dW_s \quad (5.17)$$

and for $x(t)$ is

$$x(t) = \frac{1}{\lambda} \left(1 - e^{-\lambda t/m}\right) v_0 + \sqrt{2kT/\lambda} \int_0^t \left(1 - e^{-\lambda(t-s)/m}\right) dW_s \quad (5.18)$$

where $v_0 = v(0)$ is the initial velocity of the particle and s is an integration variable. We will usually assume that $x(0) = v(0) = 0$.

As $t \rightarrow \infty$ the system approaches equilibrium, and the means of x and v tend to zero.

$$\langle x \rangle \rightarrow 0 \qquad \langle v \rangle \rightarrow 0$$

The variances however are nonzero.

$$\langle x^2 \rangle \rightarrow \frac{2kTt}{\lambda} \qquad \langle v^2 \rangle \rightarrow \frac{kT}{m}$$

At equilibrium, the probabilities of the state variables, $p(x, t)$ and $p(v, t)$, are normal distributions (Equation 5.19).

$$p(x, t) \sim \mathcal{N}\left(0, \frac{2kT}{\lambda}t\right) \quad p(v, t) \sim \mathcal{N}\left(0, \frac{kT}{m}\right) \quad (5.19)$$

5.4 New component: Stochastic resistor

Equation 5.14 shows a direct relationship between the noise σ and drag coefficient λ parameters. This means we can parameterise the drag and noise components with a single parameter (λ). Put another way, it is necessary for $\sigma = \sqrt{2kT\lambda}$ to hold for our bond graph model to remain physically compliant.

This motivates us to ‘merge’ the drag and noise forces into a single bond graph component.

5.4.1 Stochastic resistor definition

We will define a new **stochastic resistive component** that combines the constitutive relations of the **R** resistive and **Se** components. Figure 5.3 shows how the components are combined in relation to the Brownian particle bond graph.

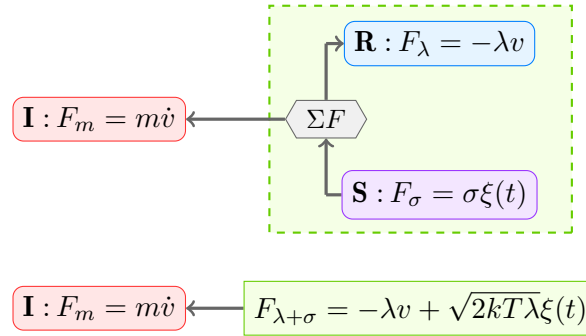


FIGURE 5.3: Stochastic resistor bond graph

Stochastic Resistive Component	
Define $\mathbf{RW} : F = \phi(v)$.	
	$F = -\lambda v + \sqrt{2kT\lambda}\xi(t) \quad (5.20)$
Define in differential form $\mathbf{RW} : F dt = \hat{\phi}(v dt)$.	
	$F dt = -\lambda v dt + \sqrt{2kT\lambda} dW \quad (5.21)$

5.4.2 General applicability of the stochastic resistive component

A single stochastic component ensures thermodynamic compliance is maintained. This is better than requiring two components to be added together in new bond graphs. To see why, we will consider two cases of a Brownian particle model, and show how these fail to conserve energy.

1. A noise term without a resistive force
2. A resistive force without a noise term

These models are shown as bond graph submodels in [Figure 5.4](#).

Case 1: Zero-drag model

The zero-drag model ([Figure 5.4B](#)) is when the friction force on the Brownian particle is ignored. This is equivalent to setting $\lambda = 0$ in [Equation 5.11a](#), or removing the $\mathbf{R}$ -component from the bond graph in [Figure 5.2](#).

The Langevin equation with $\lambda = 0$ is now

$$dv = \hat{\sigma} dW \quad (5.22)$$

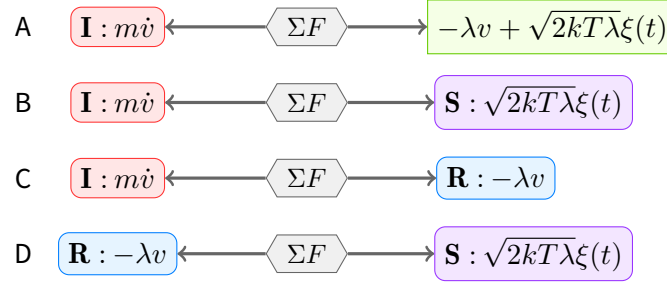


FIGURE 5.4: Brownian particle bond graph submodels. (A) Stochastic resistor, mathematically equivalent to the bond graph in Figure 5.2; (B) Zero-drag model; (C) No noise term (deterministic); (D) No inertial force ($m \approx 0$).

where $\hat{\sigma}$ is a new noise parameter to be determines. Starting with Equation 5.17, we set $\lambda = 0$ to get

$$v(t) = \hat{\sigma} \int_0^t dW_s = \hat{\sigma} W_t \quad (5.23)$$

where s is an integration variable and the initial velocity $v(0)$ is zero. The means of the position and velocity are still zero ($\langle x \rangle = \langle v \rangle = 0$). The velocity variance however is now

$$\text{Var}[v(t)] = \text{Var}[\hat{\sigma} W_t] = \hat{\sigma}^2 \text{Var}[W_t] = \hat{\sigma}^2 t \quad (5.24)$$

The variance of the velocity now increases over time, which implies the same for the average kinetic energy.

$$\langle E_k \rangle = \frac{1}{2} m \langle v^2 \rangle = \frac{1}{2} m \hat{\sigma}^2 t \quad (5.25)$$

This means $\langle E \rangle \rightarrow \infty$ as $t \rightarrow \infty$, so no steady state for the average energy can exist. In thermodynamic terms, this system is not isolated, and is gaining heat energy from its surroundings in a way unaccounted for by the model.

The zero-drag model is therefore not energy conserving and violates the first law of thermodynamics. From Noether's Theorem [129], energy is conserved if and only if the energy function is not time dependent.

From a bond graph perspective this is not so surprising, as the noise is modelled with an Se component, implying a supply of energy (as an effort supply) into the system.

Case 2: Deterministic model

The deterministic model (Figure 5.4C) has no noise term. The ODE for $v(t)$ is simply

$$\frac{dv}{dt} = \frac{\lambda}{m} v \quad (5.26)$$

which has the solution

$$v(t) = v_0 e^{-\lambda t/m} \quad (5.27)$$

where v_0 is the initial velocity. It is clear that for any v_0 the particle will eventually come to a stop $v \rightarrow 0$. Hence, energy is gradually lost over time.

The drag term is an energy sink, so this is to be expected. The deterministic model is a perfectly fine physical model, but this does not accurately reflect the observed trajectory of the system being studied.

What these two cases suggest is that they are incomplete. The kinetic energy of the particle in this system grows infinitely unless it is dampened by a resistive force.

For every source of random noise, there needs to be a resistive component to act as an energy sink. The inverse is also true; for every resistive component, there needs to be an energy source to be thermodynamically compliant.

5.4.3 Fluctuation-Dissipation Theorem

Equation 5.14 shows that σ is dependent on λ . This suggests that the physical phenomena they represent, the thermal fluctuations and viscous drag, are also dependent. This has been proved in a landmark result known as the Fluctuation-Dissipation Theorem [130, 131].

The Fluctuation-Dissipation Theorem relates the thermal fluctuating forces (noise) to the heat dissipation (drag force). It connects the *microscopic* molecular-level dynamics of mechanical movement to the *macroscopic* continuum-mechanics level of dissipation and diffusion.

The Brownian particle moves about randomly when it collides with the small molecules. From a macroscopic scale, the particle appears to experience a resistive drag force from the fluid acting on the particle as it moves. This force produces friction, which dissipates energy in the form of heat back into the fluid.

Fluctuation-Dissipation Theorem[131]

The diffusion coefficient D of a system in equilibrium is

$$D = \mu kT \quad (5.28)$$

where k is the Boltzmann Constant, T is the temperature, and μ is the ‘mobility’. For the Brownian particle, this is the ratio of the terminal velocity and the drag force $\mu = v/F$.

Cellular biology operates in a low Reynolds number (highly viscous) regime [127]. In this regime, the motion of the particle is dominated by the viscous forces over the inertial forces (Figure 5.4D). For highly viscous fluids, the mobility is the inverse of the drag $\mu = 1/\lambda$. The equation for the diffusion coefficient is

$$D = \frac{kT}{\lambda} = \frac{kT}{6\pi\eta r} \quad (5.29)$$

The last relation is the constitutive relation for drag $\lambda = 6\pi\eta r$, where η is the fluid viscosity and r is the particle's (spherical) radius.

One notable feature of this theorem is how it can apply in many different physical domains. The mobility in Equation 5.28 was defined as $\mu = v/F$, which is the ratio of flow and effort variables. This means we can generalise the stochastic component to other bond graph domains.

$$\mu = \frac{v}{F} \equiv \frac{\text{flow}}{\text{effort}} \quad (5.30)$$

In general, we can define the function for an arbitrary effort e and flow f with an alternate parameterisation.

$$e = \frac{kT}{D}f \quad \text{or} \quad f = \frac{D}{kT}e \quad (5.31)$$

This leads us to an alternative definition of the stochastic resistor in terms of the diffusion coefficient D . This is especially useful if modelling the Brownian particle as a random walk (e.g. [31]).

$\lambda = kT/D$ so Equation 5.21 becomes

$$F dt = -\frac{kT}{D}v dt + kT\sqrt{\frac{2}{D}} dW$$

Rearranging in terms of $v dt$ gives us an alternative constitutive relation.

Stochastic Resistive Component — Diffusion Form

$$v dt = -\frac{D}{kT}F dt + \sqrt{2D} dW \quad (5.32)$$

As a quick example, consider the Brownian particle in a highly viscous regime where the inertial forces are negligible (Figure 5.5). No external forces ($F = 0$) with Equation 5.32 means we have

$$dx \equiv v dt = \sqrt{2D} dW \quad (5.33)$$

and so we find that the distribution for $x(t)$ is

$$x \sim \mathcal{N}(0, 2Dt) \quad (5.34)$$

Comparing this with Equation 5.19 we again see that $D = kT/\lambda$. This may be a more familiar representation of x as the solution to a random walk.

$$\boxed{-\frac{D}{kT}F dt + \sqrt{2D} dW} \longrightarrow \boxed{\mathbf{S} : F = 0}$$

FIGURE 5.5: Stochastic resistor with alternative parameterisation Equation 5.32. A ‘null force’ component is included with $F = 0$ to complete the bond graph.

5.4.4 Stochastic resistor as the Fokker-Planck equation

Instead of modelling the trajectories of $x(t)$, we can model the probability distribution $p(x, t)$, where $p(x, t)$ is the probability of being in position x at time t . $p(x, t)$ can be found by solving the Fokker-Planck Equation [122, 123].

For an SDE of the form $dx = \lambda(x, t) dt + \sigma(x, t) dW$, the probability distribution $p(x, t)$ satisfies the Fokker-Planck Equation (FPE).

$$\frac{\partial p}{\partial t} = -\frac{\partial}{\partial x}(\lambda p) + \frac{1}{2} \frac{\partial^2}{\partial x^2}(\sigma^2 p) \quad (5.35)$$

This equation is closely related to Itô’s formula (Equation 5.6).

Our earlier SDEs can be written in terms of FPEs. (Equation 5.32) may be written as

$$dx = -\frac{D}{kT}F(x, t) dt + \sqrt{2D} dW \quad (5.36)$$

recalling that $v dt \equiv dx$. The external force $F(x, t)$ is a port variable which may be a function of x .

Equation 5.36 may now be written as an FPE.

$$\frac{\partial p}{\partial t} = \frac{D}{kT} \frac{\partial}{\partial x}(Fp) + D \frac{\partial^2 p}{\partial x^2} \quad (5.37)$$

When $F = 0$ (as in Figure 5.5), we have the FPE for Equation 5.33.

$$\frac{\partial p}{\partial t} = D \frac{\partial^2 p}{\partial x^2} \quad (5.38)$$

where $\sigma = \sqrt{2D}$. When solved with initial condition $p(x, 0) = \delta(x)$ this has the solution

$$p(x, t) = \frac{1}{\sqrt{4\pi Dt}} \exp\left(-\frac{x^2}{4Dt}\right) \quad (5.39)$$

This is equivalent to $x \sim \mathcal{N}(0, 2Dt)$, the same solution as in Equation 5.34.

The Fokker-Planck Equation has a physical interpretation. Equation 5.38 is equivalent to the *diffusion equation* (also called the heat equation).

$$\frac{\partial c}{\partial t} = D \frac{\partial^2 c}{\partial x^2} \quad (5.40)$$

where $c(x, t)$ by the concentration of some substance at position x and time t . A similar substitution of $p(x, t) \rightarrow c(x, t)$ for Equation 5.37 yields the *advection-diffusion equation*. The additional advection term is the force-dependent drag term in Equation 5.37. The nature of this force will depend on the port connections. For example, in the Langevin model (Figure 5.3), F is the inertial force $F_m = m\dot{v}$.

It is difficult to solve FPEs in general, and usually requires numerical simulations. It is easier instead to find solutions at steady state ($\partial p / \partial t = 0$). In statistical physics, the energy at steady-state follows a Boltzmann distribution [122].

$$p(E) = \frac{1}{Z} \exp\left(-\frac{E}{kT}\right) \quad (5.41)$$

where Z is a normalising constant (also called the partition function).

This suggests a potentially powerful tool for building distributions with bond graphs. If we can determine the energy function from the bond graph in terms of state variables $E(x, v)$, then we can determine the distributions of the state variables $p(x), p(v)$.

For example, the kinetic energy of the Brownian particle is $E(v) = \frac{1}{2}mv^2$. Then steady-state distribution $p(v)$ can be found by

$$p(v) = \frac{1}{Z} \exp\left(-\frac{E}{kT}\right) = \frac{1}{Z'} \exp\left(-\frac{mv^2}{2kT}\right) \equiv \mathcal{N}\left(0, \frac{kT}{m}\right) \quad (5.42)$$

where Z' is the new normalisation constant, following the appropriate change of variables [132]. This is the same distribution as in Equation 5.19.

5.5 Stochastic resistors with thermal ports

A remaining question is how energy is conserved in the stochastic resistor. It is necessary to answer this question if we want to use the stochastic resistor within the bond graph framework, since thermodynamic compliance is a specific requirement.

We address this question by defining the **thermal stochastic component** (Figure 5.6). This is done by adding a thermal port as defined in subsection 2.4.3. We will then give a physical and mathematical justification for its definition.

5.5.1 Thermal stochastic resistor

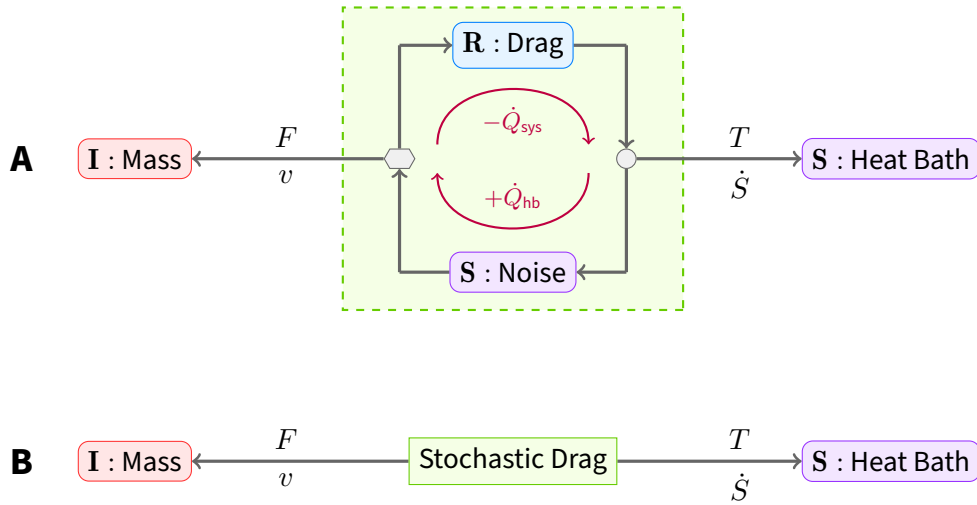


FIGURE 5.6: Brownian particle bond graph with a 2-port thermal stochastic resistor. **(A)** Thermal energy is lost via the resistor $-\dot{Q}_{sys}$, and gained via the modulated effort source $+\dot{Q}_{hb}$. At thermal equilibrium, these terms cancel out. **(B)** The stochastic resistor compacts to a two-port component connecting the mechanical domain to the thermal domain.

Figure 5.6A shows a thermal energy cycle for the stochastic resistor:

1. Brownian Particle (BP) loses energy to the heat bath through friction
2. Thermal energy gained translates to a higher average fluid molecule energy
3. Higher energy means increased collision energy and/or frequency with BP
4. Collisions transfer momentum to or from BP
5. BP movement updates stochastically, causing a change in friction force, which completes the cycle

Note that this is a *logical* rather than a *chronological* description of events. In reality, friction and molecular collision are the same physical phenomena at different scales. When a collision happens, there is an equal chance that the momentum increasing or decreasing. Note also that the thermal cycle is still valid when temperature is assumed constant. In that case, thermal energy change implies a change in entropy. This can loosely be understood as a change in ‘disorder’ of the fluid, and therefore a change in frequency of collisions.

We can determine the thermal energy lost by the brown particle through friction by adding a thermal port to our stochastic resistor (Figure 5.6B). See subsection 2.4.3 for a definition of thermal ports.

F , v , E_k are microstate variables since they correspond to a single state or trajectory of one particle, and are therefore random variables. Thermodynamic variables T , S , are macroscopic variables that define the thermodynamic ensemble, and are therefore deterministic.

From Equation 2.20, the *stochastic* thermal energy flow $\dot{q}_{\text{sys}}$ lost through friction is

$$\dot{q} = Fv \equiv \lambda v^2 \equiv \frac{1}{\lambda} F^2 \quad (5.43)$$

The macroscopic thermal energy is separately defined as $Q_{\text{sys}} = -TS_{\text{irr}}$ where S_{irr} is the irreversible entropy. $\dot{Q}_{\text{sys}}$ is also the ensemble average over all possible trajectories $\langle \dot{q}_{\text{sys}} \rangle$. Then we have

$$\dot{Q}_{\text{sys}} = \langle \dot{q}_{\text{sys}} \rangle = \lambda \langle v^2 \rangle \equiv \frac{1}{\lambda} \langle F^2 \rangle \quad (5.44)$$

We see from Equation 5.44 that the macroscopic heat is proportional to the variance of Brownian particle's velocity (or drag force). Likewise, the irreversible entropy flow is proportional to the variance of the velocity and force (assuming T is constant). At equilibrium, we know that $\langle v^2 \rangle = kT/m$ (Equation 5.19), which implies

$$\dot{Q}_{\text{sys}} = \frac{kT\lambda}{m} \implies \dot{S}_{\text{irr}} = -\frac{k\lambda}{m} \quad (5.45)$$

Note that the ensemble averages and variances are calculated for the distribution of *one* Brownian particle, rather than over all particles in the heat bath. In statistical physics, these are perspectives are equivalent. A distribution can be understood as both representing all existing copies of a particle (e.g. all velocities of fluid particles), or as all possible trajectories a single particle can take [125].

We can now define the thermal stochastic component equations.

Thermal Stochastic Component

Define **RS** : $(F, \dot{S}_{\text{irr}}) = \phi(v, T)$.

$$F dt = -\lambda v dt + \sqrt{2kT\lambda} dW \quad (5.46a)$$

$$\dot{S}_{\text{irr}} = -\frac{1}{\lambda T} \langle F^2 \rangle \quad (5.46b)$$

5.5.2 Stochastic energy conservation

The heat loss at equilibrium is exactly compensated by the gain in collision energy from the heat bath (Figure 5.6A). It turns out that this arises as a natural consequence of Itô calculus [133].

The microstate energy of the particle is its kinetic energy $E_k = \frac{1}{2}mv^2$. We take the second derivative of E_k according to Itô calculus rules, and substitute Equation 5.16 for dv .

$$\begin{aligned} dE_k &= \frac{dE_k}{dv} dv + \frac{1}{2} \frac{d^2 E_k}{dv^2} dv^2 \\ &= mv \left(-\frac{\lambda}{m} v dt + \frac{1}{m} \sqrt{2kT\lambda} dW \right) + \frac{1}{2} m \left(-\frac{\lambda}{m} v dt + \frac{1}{m} \sqrt{2kT\lambda} dW \right)^2 \\ &= -\lambda v^2 dt + \sqrt{2kT\lambda} v dW + \frac{kT\lambda}{m} dt + \mathcal{O}(dt^{1.5}) \end{aligned}$$

We neglect terms where the order of dt is greater than 1 ($dt dW$ has an order of 1.5).

The microstate power of the particle is therefore

$$\frac{d}{dt} E_k = \underbrace{-\lambda v^2}_{(i)} + \underbrace{\sqrt{2kT\lambda} v \xi(t)}_{(ii)} + \underbrace{\frac{kT\lambda}{m}}_{(iii)} \quad (5.47)$$

where $(dW)^2 \equiv dt$ and $\xi(t) \approx \frac{dW}{dt}$. The three terms in Equation 5.47 are the different energy exchanges in the model. These are:

- (i) heat energy loss through friction (always negative);
- (ii) instantaneous kinetic energy exchange from elastic collisions with fluid particles (symmetric around 0);
- (iii) thermal energy gain from the heat bath (always positive).

Notably, because of Itô's rule (iii) has appeared as an extra term in Equation 5.47. When out of thermal equilibrium, there is a net-positive energy from the heat bath passed on to the Brownian particle, even though (ii) is a net-zero energy exchange (momentum can be lost as much as it is gained). When the system is in thermal equilibrium with the heat bath, the energy gain is compensated by the heat loss from friction. This conservation of stochastic energy principle applies more broadly than what can be shown here; for a study into a wider class of Brownian motion systems, see [133].

Taking the ensemble average of Equation 5.47 yields

$$\frac{d}{dt} \langle E_k \rangle = -\lambda \langle v^2 \rangle + \frac{kT\lambda}{m} \equiv -\dot{Q}_{\text{sys}} + \dot{Q}_{\text{hb}} \quad (5.48)$$

where $\dot{Q}_{\text{hb}}$ is the supplied energy from the heat bath. We also have $\langle v\xi(t) \rangle = 0$ because of white noise autocorrelation.

At equilibrium, we had $\langle E_k \rangle = kT/2$ which means $d/dt \langle E_k \rangle = 0$, i.e. net power is zero. This implies $\dot{Q}_{\text{sys}} = \dot{Q}_{\text{hb}}$, which happens when $\langle v^2 \rangle = kT/m$, confirming again Equation 5.19.

Two interesting observations

First, we now have a definition for an acausal entropy resistor. The standard definition describes the irreversible entropy flow from a resistive component [19, 66]. We can also define Equation 5.48 in terms of total entropy flow, i.e. $\dot{S}_{\text{tot}} = -\dot{S}_{\text{irr}} + \dot{S}_{\text{hb}}$. $\dot{S}_{\text{irr}}$ is the entropy that *exits* the system, while $\dot{S}_{\text{hb}}$ is the entropy that *enters* the system. The component equations can be rewritten as

$$F dt = -\lambda v dt + \sqrt{2kT\lambda} dW \quad (5.49a)$$

$$\dot{S}_{\text{tot}} = -\frac{\lambda}{T} \langle v^2 \rangle + \frac{k\lambda}{m} \quad (5.49b)$$

$\dot{S}_{\text{tot}}$ includes the effect of the heat bath's entropy onto the particle. This does however require m as an extra parameter, which may limit which models this is applicable to.

Second, notice that because the thermal power from the heat bath is constant ($T\dot{S}_{\text{hb}} = kT\lambda/m$), we could write the noise magnitude in the Langevin equation as $\sigma^2 = 2mT\dot{S}_{\text{hb}}$, or in full as

$$F dt = -\lambda v dt + \sqrt{2mT\dot{S}_{\text{hb}}} dW \quad (5.50)$$

or, using Equation 5.49b as

$$F dt = -\lambda v dt + \sqrt{2mT \left(\dot{S}_{\text{tot}} + \lambda \langle v^2 \rangle \right)} dW \quad (5.51)$$

This means we can rewrite the 2-port thermal stochastic component as one equation. Equation 5.51 is, granted, a bit more unwieldy, but what is nice is there is now only one equation connecting across the two ports of the stochastic resistor. Interestingly, Equation 5.51 also neatly partitions the stochastic microstate variables v and F , from the deterministic macrostate variables T , $\dot{S}$, and $\langle v^2 \rangle$.

5.6 Applications of the Brownian bond graph

Here we briefly look at how the Brownian particle model can be extended now that we have a stochastic resistor (Figure 5.7).

5.6.1 Extensions to the bond graph

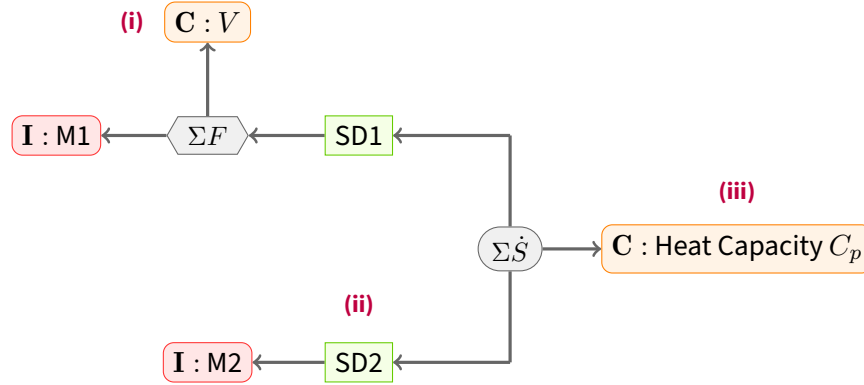


FIGURE 5.7: Proposed extensions to the Brownian particle bond graph model. **(i)** Particle (M1) subject to a potential field using a C component; **(ii)** Multiple instances of a Brownian particle sharing the same thermal reservoir; **(iii)** Available thermal energy determined by the heat capacity of the thermal fluid.

Potential field

A straightforward addition to the Brownian particle is to include a potential field term. An arbitrary potential field $V(x)$ can be added to the bond graph as a C component (Figure 5.7(i)).

A potential field applies a force $V'(x)$ on the particle proportional to the displacement of the particle against some gradient x . In cell biology, this could be an electrochemical gradient, such as was seen in an ion channel gate [54]. A more common example in classical mechanics is gravitational potential $F_g = -mgh$, where h is the height above ground and g is the gravitational constant. On the scale of cell biology the effect of gravity is negligible, but it is perhaps easier to conceptualise how a potential gradient affects the particle (i.e. will tend to arc ‘downwards’).

The updated Langevin equation SDE is

$$m dv = (-V'(x) dt - \lambda v) dt + \sqrt{2kT\lambda} dW \quad (5.52)$$

We can then easily find the steady state probability distributions for this system (see subsection 5.4.4). The total energy is now the sum of the potential and kinetic energy

$$E(x, v) = V(x) + \frac{1}{2}mv^2 \quad (5.53)$$

The steady-state distribution for x and v turn out to be

$$p(x) = \frac{1}{Z_1} \exp\left(-\frac{V(x)}{kT}\right) \quad (5.54a)$$

$$p(v) = \frac{1}{Z_2} \exp\left(-\frac{mv^2}{2kT}\right) \quad (5.54b)$$

where Z_1, Z_2 are normalisation constants (i.e., partition functions).

Multiple interacting particles

Another ‘simple’ extension to the bond graph is adding multiple copies of the Brownian particle module to a shared thermal reservoir (Figure 5.7(ii)). The thermal ports are connected with a thermal energy source at a 1-junction (shared temperature). The total entropy of the system is now the sum of the submodules of the bond graph: $S = S_1 + S_2$.

The analysis is more complicated because we now have to consider multiple noise processes dW_1, dW_2 , one for each particle. We can make the assumption that they are independent, but this may not always be the case.

Consider the case where M1 is a charged particle receiving energy from an electric field (Figure 5.7). If the M1 fluctuates more because of the additional energy, it will dissipate more thermal energy through friction. The thermal energy dissipated will influence other components in the model, including the noise process of particle M2. In this scenario, the action of one particle influences the noise magnitude of another particle.

Heat capacitance

The thermal source of effort $Se : T$ can be joined with (or replaced by) a *heat capacity* component $C : C_p$ (Figure 5.7(iii)). A heat capacitor is the thermal analogue of the electric capacitor or mechanical spring, which store potential energy that can be released later.

The $C : C_p$ constitutive relation is by definition between the temperature and entropy. The entropy change as a function of temperature with constant heat capacity is

$$\Delta S = C_p \ln\left(\frac{T}{T_0}\right) \quad (5.55)$$

where T_0 is the baseline temperature and C_p J/K is the heat capacitance at constant pressure [61].

The heat capacitance C_p is how much thermal energy is stored in the surrounding fluid (e.g. the water in a cell). Heat capacitance is defined as the change in energy per temperature change, given other variables (pressure, volume) are constant.

We can use a heat capacitor if the thermal properties of the fluid are subject to change. This might occur with an endothermic or exothermic chemical reaction occurring upstream of our particular system. Then a sudden release of thermal energy would cause the noise of all constituent components to increase. One could model how the amplitude of a noisy process changes in response to changes in the macroscopic environment — changes in temperature, pressure, or volume.

We have assumed an NPT ensemble with a thermal reservoir and a fixed temperature. This is a biologically sensible assumption, but we are not forced to make this assumption. We could, for example, model a thermodynamically isolated (NPE) system. An isolated system permits no mass or energy transfer with its surroundings. Then we would have a model of energy conservative, thermodynamically-driven Brownian motion with no sources or sinks.

5.6.2 Non-equilibrium thermal energy

The derivations up until [section 5.5](#) have assumed thermal equilibrium for the Brownian bond graph. This has made calculations and derivations easier, but this does not need to be assumed in general.

We had as our initial conditions $v(0) = x(0) = 0$, but since $\langle v \rangle = 0$ is an equilibrium point, there is no identifiable change in thermal energy or entropy. Specifically, $\Delta S = \Delta S_{\text{sys}} + \Delta S_{\text{hb}} = 0$.

For a non-equilibrium initial condition $v_0 \neq 0$ the net change in thermal energy of the system is

$$T\Delta S_{\text{sys}} = \left(-\frac{1}{2}mv_0^2 + \frac{k_B T}{2} \right) e^{-2\lambda t/m} \quad (5.56)$$

See [section D.2](#) for the derivation. At $t = 0$ energy is split between the initial kinetic energy $\frac{1}{2}mv_0^2$ and the average energy per ‘mode’ $k_B T/2$ (cf. [section D.1](#)).

The key point is that the entropy change within the system decays exponentially until the equilibrium point is reached.

$$\Delta S_{\text{sys}} \propto e^{-2\lambda t/m} \quad (5.57)$$

It would be interesting and beneficial to explore how non-equilibrium thermodynamic systems can be modelled with bond graphs. It is not clear at this stage if the newly defined bond

graph components work naturally with non-equilibrium thermodynamics, and what changes (if any) are required.

5.6.3 Bridging microstates and macrostates

An interesting implication of the stochastic resistor is how it connects microscopic and macroscopic modelling scales in thermodynamics and statistical physics. This creates a thermal stochastic ‘bridge’ that propagates noise across physical scales. On the microscopic scale, discrete quantities of particles with distinct positions, velocities, and energies interact with each other through (classical) collisions and reactions. On the macroscopic scale, intensive quantities such as temperature, concentration, and density describe the system as a whole. Fluctuation-Dissipation Theorem provides a physical and mathematical approach for how to reconcile the two.

A consequence of connecting a stochastic system to a heat bath is that we are converting energy from a *microscopic, stochastic* system to a *macroscopic, deterministic* thermodynamic ensemble (Figure 5.8, cf. Figure 2.7. See also subsection 2.4.2 for more detail on microscopic and macroscopic energy definitions.) This means we will need to ‘convert’ from stochastic variables to deterministic thermodynamic variables by taking the ensemble averages over the random variables.

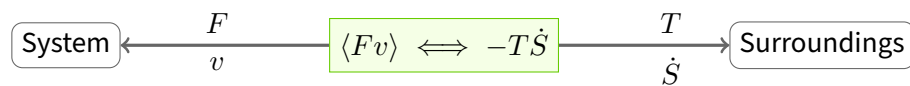


FIGURE 5.8: The thermal stochastic resistor (green) conceptually bridges the microscopic (stochastic) *System* with the macroscopic (deterministic) *Surroundings*. Compare this with the free energy compartment Figure 2.7 from section 2.4.2.

The stochastic resistor also enables us to define microstates and macrostates with bond graphs (Figure 5.9). In a hierarchical bond graph, the top level is the macrostate, and the sub graphs define discrete microstates. We can go further and define *mesostates*, which are intermediate collections of the microstates [134]. These micro/meso/macro states are the levels of scale that are bridged by a stochastic resistor.

The thermal stochastic bridge is a novel way to implement a bottom-up stochastic model. The stochasticity of a lower-level model (e.g. particle movement) defines the behaviour at higher levels (e.g. temperature). This process could be repeatedly applied up and down the layers of complexity in a cell. This has potentially interesting implications in whole-cell modelling. Each timescale and layer of complexity in a whole-cell model could be ‘bridged’ with thermal stochastic equations and the Fluctuation Dissipation Theorem. For example, bridging gene expression (discrete, stochastic) to molecular dynamics (continuous, random walk)

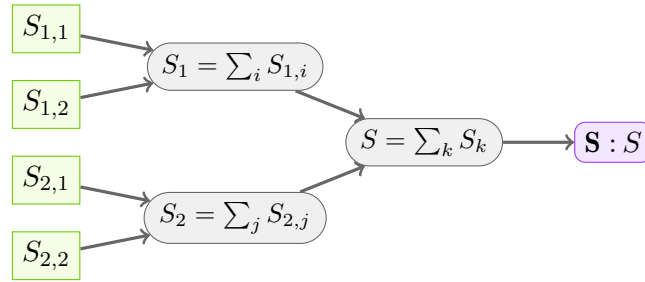


FIGURE 5.9: Bond graphs can represent thermodynamic microstates (left) and macrostates (right), with optional intermediate mesostates (centre).

to fluid mechanics (continuous, deterministic). The challenge is how to define micro and macro states precisely and consistently for whole-cell modelling.

5.7 Stochastic chemical reactions

In the following examples, we represent chemical reaction networks using the stochastic resistor component. The systems of equations produced are specific examples of the Chemical Langevin Equation.

Chemical reactions are ‘events’ that occur with some probability in a finite time interval [135, 136]. Two or more substrates collide (with some probability) and ‘react’ (with some other probability). A reaction can also be a transition between states, such as the removal from a system, or transport from one compartment to another.

This is where chemical reactions differ from resistive components in bond graphs. Resistive components and reaction both dissipate energy and release heat into their surroundings, yet a reaction is fundamentally a probabilistic event that at high population counts is approximated by a continuous concentration.

Unsurprisingly, the Chemical Langevin Equation has a similar structure to the Langevin Equation (Equation 5.11a). The key difference, and source of extra difficulty, is that the noise is now *multiplicative*. The magnitude of the noise is now dependent on the state variable

$$dx = \lambda(x) dt + \sigma(x) dW$$

One of the consequences of multiplicative noise is that we now have to consider which stochastic calculus convention to use. The two most common are the Itô and Stratonovich conventions. This difference in convention is essentially: “*do the dynamics happen after noise is applied, or before?*”. In physics and engineering, Stratonovich convention is typically used because it maintains the chain rule, and is consistent with existing calculus rules (as opposed to

Itô convention which requires Itô calculus). The downside is the more complex SDE formulation with the Stratonovich convention. See [51] for a discussion on the different conventions, and [137] for a good explanation of the two. For ease of derivation and continuity with the previous sections, I continue to use the Itô convention.

For the following examples we assume that the physical context is similar to that in the Brownian particle example: the molecules exist in a heat bath from which they can draw or dissipate energy; the fluid environment is highly viscous, and; the thermodynamics ensemble is NPT (constant particle number N , pressure P , and temperature T).

We also assume that all chemical reactions are reversible and that the noise process for the forward and reverse reactions are the same. Note in particular that we sample a different stochastic process dW_j for each reaction j . This is because we initially assume that the thermal noise that drives each reaction is independent of each other. This includes the forwards and reverse reaction ($dW_f \neq dW_r$). An implication of this independence is that $dW_i dW_j = 0$ if $i \neq j$ and $dW_i dW_i = dt$. For easier computation and equation manipulation we can make the simplifying assumption that all stochastic processes are identical, i.e. $dW_i \equiv dW_j$ for all i, j .

5.7.1 The Chemical Langevin Equation

Stochastic chemical reactions are fundamentally defined by the Chemical Master Equation (Equation 5.58). This is a discrete state process that describes the change in probability distribution of each chemical species [136, 138].

Let $p_i(n; t)$ be the probability that at time t there are n copies of species i . The Chemical Master Equation is defined as

$$\frac{\partial}{\partial t} P_i(n; t) = \sum_{j=1}^M [a_j(n - c_{ij}) P_i(n - c_{ij}; t) - a_j(n) P_i(n; t)] \quad (5.58)$$

where c_{ij} is the stoichiometric coefficient of species i in reaction j , M is the total reaction number, and $a_j(n)$ is the propensity function (defined later).

Equation 5.58 is an equation for the distributions of species, and is valid for all chemical reaction systems, including systems with low chemical copy counts ($n < 10$) and absorbing states ($n = 0$). It is difficult in general to find solutions to Equation 5.58, and the combinatorial explosion for large n render it impractical ($1 \text{ mol} = 6.022 \times 10^{23}$).

Instead, we derive the Chemical Langevin Equation (CLE) from Equation 5.58 by assuming a continuous limit (see [136] for a derivation).

The Chemical Master Equation (Equation 5.58) is the most accurate representation and best used for small chemical populations. For large sufficiently large concentrations or when $at \gg 1$, then the Chemical Langevin Equation (Equation 5.59) is a good approximation [136].

Let $\mathbf{x} \in \mathbb{R}^N$ be the concentrations of species in a chemical reaction network. For a chemical species with concentration $x_i \in \mathbf{x}$, the CLE is

$$dx_i = \sum_j c_{ij} a_j(\mathbf{x}) dt + \sum_j c_{ij} \sqrt{a_j(\mathbf{x})} dW \quad (5.59)$$

where c_{ij} is the stoichiometric coefficient of species i in reaction j , and $a_j(\mathbf{x})$ is the propensity function of the state vector $\mathbf{x}$.

dx_i follows a normal distribution

$$dx_i \sim \sum_{j=1}^M \mathcal{N}(c_{ij} a_j(\mathbf{x}), c_{ij}^2 a_j(\mathbf{x})) \quad (5.60)$$

Note that the since $c_{ij}^2 \geq 0$ we can add the variances.

It will be useful for bond graphs to define the CLE for the forward and reverse reaction directions. For convenience we will define $\nu = \nu_f + \nu_r$ with $\nu_f > 0$ and $\nu_r < 0$. These molar flow rates have the following CLE

$$\nu_f dt = a_f dt + \sqrt{a_f} dW_1 \quad \nu_r dt = -a_r dt + \sqrt{a_r} dW_2$$

where $a \equiv a(\mathbf{x})$ and dW_1, dW_2 are independent noise processes ($dW_1 dW_2 = 0$). The term $\sqrt{a_r} dW_2$ has a positive coefficient since it easier to add noise terms and technically the noise is equally likely to be positive or negative.

Then the total reaction flow rate CLE is

$$\nu dt = (a_f - a_r) dt + \sqrt{a_f} dW_1 + \sqrt{a_r} dW_2 \quad (5.61)$$

which again has follows a normal distribution

$$\nu \sim \mathcal{N}(a_f - a_r, a_f + a_r) \quad (5.62)$$

Propensity function

The *propensity function* a_j determines how likely a reaction is to occur within the next time interval ($t + dt$ in the continuous case). This can be defined as:

$$a_j(\mathbf{x}) = \kappa_j P(\mathbf{x}_j) \quad (5.63)$$

where κ_j is the reaction rate constant, and $P(\mathbf{x}_j)$ is the probability of finding the particular reaction configuration $\mathbf{x}_j \subseteq \mathbf{x}$.

The chosen reaction mechanism will determine the exact form of a_j . Mass action kinetics assumes the species are well mixed. For the reaction, $A + B \xrightarrow{k_1} C$ for example, mass action kinetics would have $P \propto x_A x_B$ and $a_1 = k_1 x_A x_B$.

In a bond graph, the propensity for a reaction is

$$a_j(\mathbf{x}) = \kappa \exp\left(\frac{-A_j}{RT}\right) \quad (5.64)$$

where j is the reaction direction (forward or reverse), κ is the thermodynamic reaction rate. Note that for thermodynamic consistency, we must have $\kappa_f = \kappa_r = \kappa$. The affinity A_j is the sum of potential functions for reaction j .

$$A_j = \sum_i u_i(x_i) \quad (5.65)$$

We have used the symbol u to reflect that potential forces in a reaction can come from other physical sources. The ion pore channel model from [subsection 4.3.1](#) is a good example of this, where electrochemical potentials drive the transport of ions across a membrane [54]. In a fully chemical reaction system, $A_j = -\sum_i \mu_i$.

Note the extra negative sign in the exponent in [Equation 5.64](#), which differs from the usual definition. By defining it this way, it is clearer to see that the propensity function follows a Boltzmann distribution [Equation 5.66](#)

$$P(\mathbf{x}_j) = \exp\left(\frac{-A_j}{RT}\right) \simeq \exp\left(\frac{-E}{kT}\right) \quad (5.66)$$

5.7.2 CLE reaction component definition

A nice property is that the noise term is proportional to the square root of the deterministic term. This is the result of approximating a Poisson distribution with a Normal distribution,

which have the same mean and variance [136]. The derivation of this property is instructive and has been included in [Appendix D](#).

This particular structural property of the CLE is convenient for bond graphs. Since we have a method to symbolically generate reaction equations with bond graphs, it is relatively straightforward to extend this method to include a stochastic term that is dependent on the deterministic term. Essentially, the same algorithms used to symbolically generate the reaction equations (such as in [chapter 4](#)) can be updated to include an additional term.

This parallel reaction perspective will help us define the CLE bond graph component. This is conceptually like having two parallel reactions for the mean and standard deviation, $\nu = \bar{\nu} + \omega$ on the orders of dt and dW , respectively. From [Equation 5.61](#) we can write

$$\bar{\nu} dt = (a_f - a_r) dt \quad (5.67a)$$

$$\omega dt = \sqrt{a_f} dW_1 + \sqrt{a_r} dW_2 \quad (5.67b)$$

Then the total reaction flow rate is $\nu = \bar{\nu} + \omega$ is the same as [Equation 5.61](#). If we assume the noise processes each direction are equivalent ($dW_1 = dW_2 = dW$) then we can write

$$\omega dt = (\sqrt{a_f} + \sqrt{a_r}) dW \quad (5.68)$$

Using [Equation 5.64](#), we can write [Equation 5.67a](#) out explicitly in bond graph convention as

$$\nu dt = \kappa \left[\exp\left(\frac{-A_f}{RT}\right) - \exp\left(\frac{-A_r}{RT}\right) \right] dt \quad (5.69a)$$

$$\omega dt = \sqrt{\kappa} \left[\exp\left(\frac{-A_f}{2RT}\right) + \exp\left(\frac{-A_r}{2RT}\right) \right] dW \quad (5.69b)$$

With these equations we now define a new **CLE reaction component** in a similar way to the stochastic resistor component from [section 5.4](#).

Chemical Langevin reaction component

Define **CLE** : $(\nu_f, \nu_r) = \phi(\mu_f, \mu_r)$.

$$\nu_f = -\nu_r \quad (5.70a)$$

$$\nu_f dt = \kappa \left[\exp\left(\frac{-A_f}{RT}\right) - \exp\left(\frac{-A_r}{RT}\right) \right] dt \quad (5.70b)$$

$$+ \sqrt{\kappa} \left[\exp\left(\frac{-A_f}{2RT}\right) + \exp\left(\frac{-A_r}{2RT}\right) \right] dW \quad (5.70c)$$

where κ is the reaction rate, R is the universal gas constant, T is the temperature, and $A_f = -\mu_f$, $A_r = -\mu_r$ are the reaction infinities.

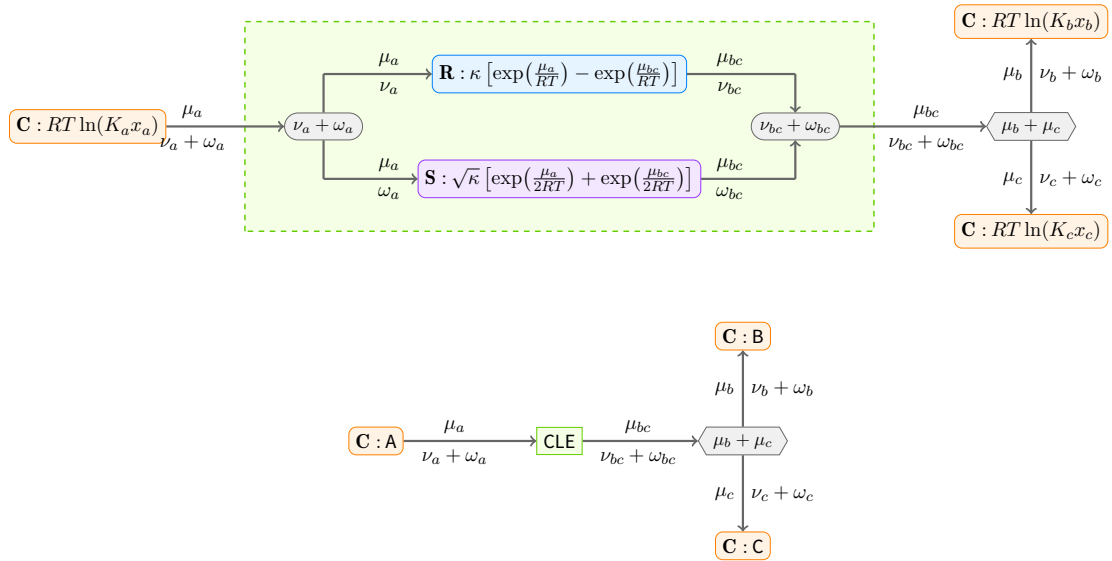


FIGURE 5.10: The Chemical Langevin Equation component **CLE**, shown here for the reaction $A \rightleftharpoons B + C$. (Top) One way to understand the CLE is as parallel ‘reactions’ of the mean (blue) and variance (purple) of the reaction rate. This view is mainly for visualising, other interpretations are possible. (Bottom) Condensed view of the CLE component.

Example

Figure 5.10 shows a simple example of a bond graph of the reaction $A \rightleftharpoons B + C$. The affinities are the sums of the chemical potentials

$$\begin{aligned}\mu_a &= RT \ln(K_a x_a) \\ \mu_{bc} &= RT \ln(K_b K_c x_b x_c)\end{aligned}$$

The subscript bc indicates the combined potential and flow rate of the products.

The mean reaction flow rates are

$$\begin{aligned}\nu_a dt &= \kappa \left[\exp\left(\frac{\mu_a}{RT}\right) - \exp\left(\frac{\mu_{bc}}{RT}\right) \right] dt \\ &= \kappa [K_a x_a - K_b K_c x_b x_c] dt \\ \nu_{bc} &= -\nu_a\end{aligned}$$

These are the same as in a deterministic reaction component. The ‘variance’ reaction flows are

$$\begin{aligned}\omega_a dt &= \sqrt{\kappa} \left[\exp\left(\frac{\mu_a}{2RT}\right) + \exp\left(\frac{\mu_{bc}}{2RT}\right) \right] dW \\ &= \sqrt{\kappa} \left[\sqrt{K_a x_a} + \sqrt{K_b K_c x_b x_c} \right] dW \\ \omega_{bc} &= \omega_a\end{aligned}$$

Since $\nu_a = -\nu_{bc}$ and $\omega_a = \omega_{bc}$, it is sufficient to define the CLE for dx_a only

$$\begin{aligned} dx_a &= (-\nu_a + \omega_a) dt \\ &= \kappa [K_a x_a - K_b K_c x_b x_c] dt \\ &\quad + \sqrt{\kappa} \left[\sqrt{K_a x_a} + \sqrt{K_b K_c x_b x_c} \right] dW \end{aligned}$$

with $dx_a = -dx_b = -dx_c$.

5.7.3 Thermal ports in chemical reactions

Connecting the CLE component to the heat bath (like in [section 5.5](#)) is more complicated and will not be fully explored here. There are a few points to consider for chemical systems.

For one thing, the variables involved are already thermodynamic quantities. These variables are already ‘macroscopic’ and do not need to be defined separately with thermal ports. They come from the fundamental energy equation [\[67\]](#). In a biochemical context (NPT ensemble) we have Gibbs free energy $\Delta G = -T\Delta S = \sum_i \mu_i \Delta N_i$. Taking the derivative with respect to time and recalling that $\nu \equiv dN/dt$ we get

$$\dot{G} = -T\dot{S} = \sum_i \mu_i \nu_i \quad (5.71)$$

In [section 5.5](#) we saw that the noise is driven by the thermal energy of the heat bath, determined by $T\dot{S}$. In that example, the Brownian particle receives energy from the heat bath. Here however, the chemical reaction network *is* the heat bath (at least in part). Unlike in other bond graph domains, the chemical storage compartment **Ce** does not represent a single physical object in one location. Instead, it holds the total chemical potential of species A dispersed throughout the environment it occupies. We can therefore describe a chemical species with thermodynamic variables, in a similar way to describing the heat bath in the Brownian particle example.

If thermal ports are added, it therefore may make more sense on the chemical storage **Ce** component ([Figure 5.11](#)). One reason is in the definition of chemical potential $\mu = RT \ln(Kx)$, which has an explicit dependency on T . (what is less obvious is dependency on $\dot{S}$, but it can be shown that $S \propto R \ln(x)$ [\[61\]](#).)

One might wish to model an exothermic reaction that releases a large amount of heat energy, possible raising the heat bath ambient temperature. The increase in heat will also increase the chemical potentials of the species in the system.

The definitions so far are straightforward if T is constant. One consideration for future models is if T is *not* constant and is relevant to the dynamics of the system (in other words, changes on a fast enough timescale). (Note that this does not really occur in systems biology, so is not required for our purposes. In modelling chemical reactions in general, however, this may need to be considered.)

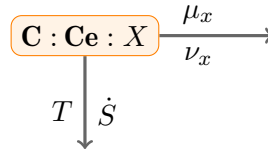


FIGURE 5.11: Ce chemical potential component with thermal port.

To determine $\dot{S}$ we would use

$$\dot{S} = \frac{\mu\nu}{T} = \dot{x}R\ln(Kx)$$

This equation is confirmed by the equation for entropy change with respect to concentrations $\Delta S = R\ln\left(\frac{x_1}{x_0}\right)$ [61].

Another factor to consider is that the reaction rates have a temperature dependence, defined by the Arrhenius equation [61].

$$k = Ae^{\frac{-E_a}{RT}} \quad (5.72)$$

where A is a pre-exponential factor, and E_a is the activation energy which follows a Boltzmann distribution. Note that k here is the kinetic rate and not the thermodynamic molar flow rate κ .

The activation energy E_a might include the kinetic energy of the two molecules when they collide. It is not always sufficient for molecules to make contact, they must also carry enough kinetic energy (they may also need to have the correct orientation [61, 135]). Cells reduce this energy barrier with enzymes.

Equation 5.72 is an empirical law with multiple supporting theories. One example is collision theory, which predicts the reaction rate by the frequency of collisions and the likelihood of a reaction occurring [61, 135]. Collision theory gives the kinetic rate for the reaction $A + B \xrightarrow{k} C$ as

$$A = \sigma_{ab}\sqrt{\frac{8RT}{\pi M}} \quad (5.73)$$

where σ_{ab} is the collision cross-section area of A and B and M is the molecular mass [61]. Equation 5.73 applies to bimolecular reactions, and modifications can be made to apply to unimolecular or trimolecular reactions [135].

The key idea here is that Equation 5.72 and Equation 5.73 both explicitly depend on T . If we factor in changes in temperature for chemical reactions, we will need to also connect the reaction component to the heat bath.

Figure 5.12 shows how a chemical bond graph with thermal ports may look. As well as connecting the heat bath to the reaction component, bonds are connected to each Ce component.

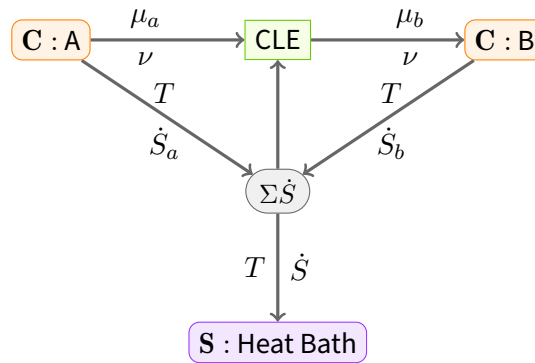


FIGURE 5.12: Bond graph of $A \rightleftharpoons B$ reaction with thermal ports on the chemical energy storage components.

5.8 Concluding remarks

Beyond Gaussian distributions

The probability distributions considered in this chapter have all been based on Gaussian (Normal) probability distributions. While this is the most natural distribution to use for the physics presented and the simplest to demonstrate the theory, they are also the most well understood behaviour in the limit and so the specific benefit of a bond graph approach is not apparent. Future work will demonstrate how stochastic bond graphs can model other classes of distributions.

A straightforward extension which generates a new class of distributions is the Brownian particle in 3D space. The position, velocity, and forces on the particles are now all 3-dimensional spatial vectors: $\mathbf{x}$, $\mathbf{v}$, $\mathbf{F}$. The distribution of the energy of the particle E now follows a *Maxwell-Boltzmann distribution* [132]:

$$p(E) = 2\sqrt{\frac{E}{\pi}} \left[\frac{1}{kT} \right]^{\frac{3}{2}} \exp\left(-\frac{E}{kT}\right) \quad (5.74)$$

This is in fact equivalent to a Chi-squared distribution with 3 degrees of freedom.

More exotic distributions appear in materials science. The Fermi-Dirac and Bose-Einstein distributions are two such examples that have applications in particle physics [139].

Simulating stochastic systems

Bond graphs are useful for generating differential equations in a form that is easier to solve (e.g. Equation 5.13). The equations will still need to be solved in some way, however, using either analytical methods (as seen in this chapter), or more commonly via numerical simulation.

Solutions to stochastic equations may be found analytically using generating functions [122, 123], although only a few classes of stochastic model have nice closed-form solutions. It is often better to simulate trajectories using numerical methods with, e.g., the Gillespie algorithm [27].

There are many software tools available for efficiently solving SDEs. In Julia, for example, one can solve SDEs over an ensemble using DifferentialEquations.jl [107] or Catalyst.jl [111]. Although BondGraphs.jl (the subject of chapter 4) is compatible with these modelling packages, it does not yet handle stochastic equations. This feature is planned for a future release.

Satisfying the second law of thermodynamics

Typically, when it is stated that bond graphs comply with the laws of thermodynamics, it usually refers to energy conservation only. In general, the second law of thermodynamics is not satisfied in bond graphs (reaction components being an exception). This is because entropy is not usually calculated or dealt with directly, unless dealing with entropy flows in fluid, (e.g. [66, 71]).

In general, however, entropy is not treated as fundamental in bond graph modelling as energy and power. When extending bond graphs to include thermal ports and probability distributions, we will need to also confirm that the second law is not broken if we are to claim that “the laws of thermodynamics are automatically satisfied”.

Bond graphs should satisfy all laws of thermodynamics, but the second law is often overlooked or not discussed. With thermal ports, the entropy of a system can be determined, and the second law validated explicitly, leading to more accurate models.

Further applications of the Fluctuation-Dissipation Theorem

The Fluctuation-Dissipation Theorem applies in many other domains of physics. A good example is with the Johnson-Nyquist noise found in electric resistors. Johnson [140] demonstrated that the thermal noise in electric resistors is given by

$$\langle V^2 \rangle \approx 4kTR\Delta f \quad (5.75)$$

where V is the circuit voltage, R is the (electrical) resistance, and $\Delta f = f_1 - f_2$ is the frequency bandwidth.

The takeaway point is that we can use this theory in many other bond graph models. If flow and effort are defined, we can create a stochastic component that applies to this domain of physics.

One bond graph produces multiple models

From one bond graph description we can generate multiple models. We can create an ODE and SDE from the same bond graph structure and constitutive relations. This has been shown already with the Brownian particle bond graph. In [subsection 5.3.1](#) we derived the Langevin equation for the Brownian particle bond graph ([Equation 5.11a](#)). In our component definitions we defined $\dot{x}$ as “change in x over time”. This notation is usually defined as an ODE, but we could alternatively interpret this notation with differential notation for an SDE.

Likewise with the forcing term $\text{Se} : \xi(t)$, we could interpret this as a:

- (i) deterministic time-varying parameter;
- (ii) samples from a stochastic process (e.g. dW_t);
- (iii) external port for the particle subsystem;
- (iv) thermal port for thermodynamically driven noise.

The key idea here is that from the same bond graph definition we can generate multiple different differential equation models.

The equation construction algorithm for generating ODEs will generally work for SDEs. This makes it easy to generate complicated SDEs easily, especially for much larger systems (e.g. the model of the *Saccharomyces cerevisiae* metabolic network in [subsection B.2.3](#)).

The modular nature of bond graphs means we can bridge the gap between deterministic and stochastic systems by graph operations. A deterministic system can be made ‘noisy’ by swapping out deterministic components for stochastic ones.

Modularity works across scales too. Small random effects accumulate to cause large, seemingly deterministic phenomena. For example, small water molecules randomly colliding with a large metabolite cause it to move along an apparent random path. In bond graph terms, we may think of a nested hierarchical bond graph where a noise distribution at a lower level influences dynamics at a larger space scale or slower timescale.

Adding noise to existing bond graphs

It is not necessary to update all the different components with stochastic versions of each component. In [subsection 5.3.1](#) the independent noise component is useful because it is much easier to add to an existing bond graph model. We simply connect a noise source component to a junction (if it exists), and all other connected components become stochastic. Likewise, if a stochastic model is merged with a deterministic model, the deterministic model “inherits” the property of stochasticity.

This means that the creator of the bond graph does not necessarily have to specify any additional stochastic notation. Instead, this can be determined *post-hoc*. It means also that an existing deterministic bond graph, when coupled with a stochastic bond graph, will become fully stochastic automatically.

Probability distribution reactions

Finally, I will speculate on how one could implement a *probability energy domain* in bond graphs using chemical reaction network definitions. These are preliminary definitions and serve more as motivation for future research in this area.

Consider again the first order chemical reaction $A \xrightleftharpoons[k_2]{k_1} B$ with kinetic rates k_1, k_2 . For this example, x_A and x_B are the concentrations of a molecule in one of two states, A and B.

We can write this system as a single mass-action ODE by assuming the conservation law $x_A + x_B = X_0$ where X_0 is some constant.

$$\frac{dx_A}{dt} = -(k_1 + k_2)x_A + k_2X_0 \quad (5.76)$$

The solution for the concentration x_A over time is

$$x_A(t) = \frac{k_2 X_0}{k_1 + k_2} - \left(\frac{k_2 X_0}{k_1 + k_2} - A_0 \right) e^{-(k_1 + k_2)t} \quad (5.77)$$

where $x_A(0) = A_0$ is the initial concentration of A. (See [67, 123] for derivations).

Now consider the analogous system in terms of the probability distributions $p_A \xrightleftharpoons[\lambda_2]{\lambda_1} p_B$, where $p_A(t), p_B(t)$ are the probabilities of being in state A and B, and λ_1, λ_2 are the transition rates. (Here we will be assuming continuous distributions, but the theory works just as well for discrete distributions.) It turns out that we can model the probability distribution $p_A(t)$ using the same equations and solutions as Equation 5.76 and Equation 5.77.

The probability distribution can be solved in general using the Fokker-Planck Equations (Equation 5.35). For this simple example, the equation for p_A can be found using the method of generating functions [67]. The ODE is

$$\frac{dp_A}{dt} = -(\lambda_1 + \lambda_2)p_A + \lambda_2 \quad (5.78)$$

with solution

$$p_A(t) = \frac{\lambda_2}{\lambda_1 + \lambda_2} - \left(\frac{\lambda_2}{\lambda_1 + \lambda_2} \right) e^{-(\lambda_1 + \lambda_2)t} \quad (5.79)$$

We have also used the probability ‘conservation law’ $p_A(t) + p_B(t) = 1$.

Notice how the Equation 5.76 and Equation 5.77, and Equation 5.78 and Equation 5.79 have the same basic structure. The similarities here motivate a definition for a new probability bond graph domain. Using the chemical reaction domain as a basis, we can draw natural definitions for a state variable (probability), flow, effort, rate parameters, and even a conservation law (Table 5.1).

	Chemical domain	Probability domain
state q	x_A [mol]	p_A [1]
flow f	ν [mol/s]	J [1/s]
effort e	μ [J/mol]	ε [J]
rate parameters	k_i	λ_i
conservation law	$\sum_i x_i = X_0$	$\sum_i p_i = 1$

TABLE 5.1: Comparing the chemical bond graph domain with the probability domain.

We will say that the probability of being in state A or B is proportional to the probability of being in a particular energy state. Let p_i be the probability distribution for being in energy state $i \in \{A, B\}$. Then we define the effort as the *probability potential* ε_i as

$$\varepsilon_i \equiv -kT \ln(Z_i p_i) \quad (5.80)$$

where Z_i is the partition function (akin to the thermodynamics constant K_i for chemical potential). The Boltzmann constant k and universal gas constant R differ by a factor (Avogadro's Number $N_a = R/k$). Defining the potential with a negative sign means we can rearrange Equation 5.80 to get the Boltzmann distribution: $p_i = \frac{1}{Z_i} \exp\left(\frac{-\varepsilon_i}{kT}\right)$.

For the flow variable, it is possible to define a *probability current* $-J \equiv \dot{p}$ [122]. This can also be defined using a variation of the Marcelin-de Donder equations [62].

$$J = \kappa \left[\exp\left(\frac{-\varepsilon_A}{kT}\right) - \exp\left(\frac{-\varepsilon_B}{kT}\right) \right] \quad (5.81)$$

Now we can derive Equation 5.78 in the same way as deriving mass-action equations for a chemical reaction network.

$$\begin{aligned} \varepsilon_A &= -kT \ln(Z_A p_B) \\ \varepsilon_B &= -kT \ln(Z_B p_A) \\ J &= \kappa \left[\exp\left(\frac{-\varepsilon_A}{kT}\right) - \exp\left(\frac{-\varepsilon_B}{kT}\right) \right] \\ J &= \kappa (Z_A p_A - Z_B p_B) \end{aligned}$$

Then with $\dot{p}_A = -J$ and $p_A + p_B = 1$ we have the equation

$$\frac{dp_A}{dt} = -J = -\underbrace{\kappa Z_A}_{\lambda_1} p_A + \underbrace{\kappa Z_B}_{\lambda_2} (1 - p_A) \quad (5.82)$$

which is the same as Equation 5.78.

The same components and equations for chemical reaction bond graphs can be used for probabilistic reaction networks. More work is required to test the limits of this idea, as not all definitions will have a corresponding interpretable meaning.

This analogy really shows that chemical reaction networks can be understood from a probabilistic perspective. A direct comparison can be made when considering a concentration to be the ‘probability’ of finding a molecule in a defined volume.

Describing probability using physical units seems odd, but it is not without precedent. Attard [69], for example, defines ‘weights’ as a physical unit with the same properties as a probability unit, with the exception that weights are ‘unnormalised’ and therefore have natural units.

Finally, the probability bond graph domain opens the possibility for *discrete* chemical reaction networks in bond graphs. p_i is not limited to continuous distributions, and with a some

modifications to the above equation (e.g. replacing the Boltzmann distribution with a Poisson distribution), one could model discrete reactions. This idea will need to be investigated in greater detail.

Chapter 6

Conclusion

I summarise the chapters of the thesis and accomplished results. I then discuss ideas for future projects and promising directions of research. I conclude the thesis with a retrospective on bond graphs and whole-cell modelling.

6.1 Summary of research

Introduction

The overarching theme of my research has been on how practical whole cell modelling can be achieved with bond graphs. In [chapter 1](#) I summarised the progress and challenges of whole cell models and recounted the *principles of whole cell modelling* as a target for candidate modelling frameworks. I discussed how bond graphs satisfy many of these requirements already, namely in modelling biophysics, molecular closure, dynamic simulations, and modularity. These *principles* also highlighted where bond graphs needed further research and development, which motivated the rest of the chapters.

Bond Graph Theory

In [chapter 2](#) I gave a short introduction to the bond graph framework, defining key terminology and working through simple examples. I sought to present the intuition of energy-based modelling in general and the reasoning behind the bond graph methodology. I then looked back at the thermodynamic theory that lay behind the earlier definitions of bond graphs, where they were known as *network thermodynamics*. I showed how thermodynamic definitions, namely free energy, internal energy, and entropy, can be naturally extended from existing bond graph components using thermal ports.

Two Component Systems

In [chapter 3](#) I began modelling bacteria with bond graphs. I applied bond graph modelling (and energy modelling in general) to TCS, a common signalling cascade found in well-studied bacteria such as *E. coli*. The repeated modular structure of the TCS cascade made them an ideal system to apply bond graphs. I modelled the orthodox and unorthodox variants of TCS using the same bond graph submodels, where each submodel was a single reaction or step in the network.

A key point of difference with these models was the explicit inclusion of ATP as an external control parameter. It was necessary to include all energy carrying metabolites (AMP, ADP, ATP) to satisfy mass balance for bond graphs. Simulations were run for 10^4 randomly sampled parameter sets for a series of fixed ATP concentrations. This study revealed interesting energy-based effects that had not been identified before. I found that energy availability (as represented by ATP concentration) influenced: (i) the maximum strength of response of the signal; (ii) the temporal signalling curve sensitivity (hyperbolic $\rightarrow$ sigmoidal); (iii) how effectively TCS filters out noise or false signals; (iv) the signal stimulus range for the unorthodox TCS.

The last insight was especially interesting and surprising. The structure of the signal response curve for the unorthodox TCS was non-monotonic for certain parameter values. This meant that the maximum response was found at an intermediate signal strength. Moreover, ATP was found to control the position and existence of a signal peak, acting as an effective tuning control for the signal bandwidth.

BondGraphs.jl

In [chapter 4](#) I introduced Bondgraphs.jl, a new package for modelling bond graphs in Julia. I presented the core capabilities of the package, namely: 1. graphical construction and modification of models; 2. symbolic generation of equations; 3. automatic generation of models from chemical reaction networks; 4. integration with SciML and the Julia modelling ecosystem. I then demonstrated the package on some systems biology models from literature: the Ion transport bond graph demonstrates multiphysics modelling ([subsection 4.3.1](#)); the SERCA pump bond graph demonstrates the reaction network interface to automatically construct a bond graph from chemical reactions ([subsection 4.3.2](#)). A larger example model of *S. cerevisiae* is provided in [Appendix B](#).

BondGraphs.jl is fully documented with continuous integration, and is included in the official Julia registry, meaning it can be installed directly from the Julia terminal. Online documentation was reproduced in [Appendix C](#), which includes a tutorial, additional examples, and API description.

Stochastic Bond Graphs

In [chapter 5](#) I tackled the challenge of bringing stochastic dynamics to bond graphs. I define new components for stochastic bond graphs and present multiple approaches to stochastic modelling with bond graphs. Doing so required incorporating theory from statistical mechanics, stochastic modelling, probability, and thermodynamics.

I start by constructing a bond graph of a particle under Brownian motion. The dynamics are described by the Langevin equation, which was used to motivate the stochastic resistor component. The stochastic component is a method of adding physics-based noise to a system in a way that preserves thermodynamic compliance. This component turns out to explicitly encode the Fluctuation-Dissipation theorem, which has found wide-reaching use across statistical physics. The theorem allows us to generalise the stochastic component to other bond graph energy domains.

I then go further to explicitly show the thermal energy exchange between the particle and heat bath. I do this by adding thermal ports to the stochastic component connected to the heat bath component. I then show that the thermal energy output appears a natural consequence of Itô calculus. The thermal port creates a bridge between microscopic and macroscopic states of the system.

This led to the incidental benefit of a ‘two-way’ entropy resistor. Thermal energy (and entropy) which leaves the particle through friction, is gained again through stochasticity. Energy sinks like resistors can be connected to thermal reservoirs, or even to an ensemble with fixed total energy, thus creating a thermodynamically isolated system in bond graphs. I ended this section by giving some brief examples of possible extensions to the Brownian particle bond graph: Potential fields, duplicate Brownian particles, and heat capacitance components.

Next, I began work on modelling the Chemical Langevin Equation in bond graphs by defining the CLE component. Chemical bond graphs differ in some ways from the Brownian particle, therefore slightly different justification is needed for the CLE component. Finally, I introduce the idea of probability reactions and the probability domain in bond graphs.

This chapter laid the groundwork for stochastic modelling with bond graphs, but more work is required to refine the theory and apply it to a wider range of stochastic models in the literature.

6.2 Future work

6.2.1 Further extensions to bond graphs

In [subsection 1.1.1](#) the principles of whole cell modelling were presented as a standard for whole cell models [11]. My research has taken steps towards satisfying these requirements with bond graphs. However, there remains work to do in the remaining areas ([Table 1.1](#)).

One particular area for improvement is in the functional closure of bond graphs. Bond graphs are not yet suited for several categories of biological models in whole cell models. There is an opportunity here to further extend bond graphs to handle new classes of model, like what was done with stochastic models in [chapter 5](#). Some possible extensions include:

Spatial dynamics e.g. fluid dynamics or compartment models. This could be done by re-defining effort and flow as 3D spatial vectors and updating the constitutive relations accordingly. An entry point for spatial modelling in bond graphs may be through port-Hamiltonians, which are directly compatible with the bond graph formalism [141, 142]. This will result in partial differential equation models being generated. It would again be necessary to check if these spatial bond graphs are still be compatible with existing bond graph models.

Logical networks e.g. boolean networks or gene regulatory networks. Bond graphs do allow for ‘signals’ which transfer information across components when certain state values are reached [19, 20]. Bond graphs signals may fill this niche, but they have not yet been systematically applied in systems biology.

Jump processes e.g. chemical master equations and the Telegraph model [117]. Jump processes are stochastic differential equations with discontinuous jumps [122, 123]. Much of the stochastic theory in [section 5.7](#) is derived from discrete state equations. For example, the continuous chemical Langevin equation is derived from the discrete chemical master equation. At the end of [section 5.8](#) it was noted that probabilistic ‘reaction’ networks could be modified to permit discrete probability distributions.

Discrete state functions An interesting question to investigate is how bond graphs would handle equations with discrete state functions. Strictly speaking, we require that time and energy are continuous functions (cf. Noether’s theorem [129]). It is plausible that discrete effort, flow, generalised state, and generalised momenta are permitted, so long as they obey

the same energy conservation laws. Discontinuous functions with well-defined analytical properties (e.g. Dirac delta function, Heaviside function) may be useful here.

6.2.2 Investigating signal selection in unorthodox TCS

One open question from the TCS project was how and why the unorthodox TCS signalling cascade had different signal response (SR) curves ([subsection 3.2.4](#)). While some SR curves were monotonic, others were non-monotonic with one peak, and others still had two or more peaks. I found that this property was energy dependent, that an SR curve could change shape based on ATP availability.

I found that certain parameter values correlated with particular SR curve shapes ([Figure 3.11](#)). In particular, I found that it was the reaction rates, and not the base chemical properties of the species, that determined this observed phenomenon. I speculated on possible reasons why these correlations appeared, but still lack a mechanistic understanding the relationship.

In a future study, I intend to explore this systematically. One or more of the following techniques may be taken in a new study:

- In the original study, parameters were randomly sampled across a log-uniform distribution. This phenomenon was essentially found by chance. Many of the possible SR curves would have been overlooked, and the true distribution of SR curves across the parameter space is not fully understood. A systematic study could make deliberate parameter sampling choices to try to isolate this behaviour down to a few key parameters. It may be that only a few parameters, or some lumped combination of parameters, may be accurate predictors of the observed SR curves.
- To guide the investigation, I could use machine learning algorithms or prediction tools (e.g. [\[112, 143\]](#)) to find reliable predictions of the SR classes observed. I now have well-defined classes to predict and 10,000 parameter sets with simulated trajectories, with more that can be generated. The focus will be on interpretable algorithms that provide hints for underlying relationships between the parameters and signal outputs.
- Bond graphs have been used to apply control theoretic ideas to systems biology [\[22, 144\]](#). If the signal selection behaviour really is ‘tuning’ by ATP, then it may be possible to quantify this precisely using the tools of control theory.
- A different and promising approach is to apply techniques from algebraic geometry. Algebraic geometry has been applied to systems biology and chemical reaction network theory with powerful results [\[145, 146\]](#). In one study, Kothamachu *et al.* [\[147\]](#) showed under what conditions TCS exhibit multi-stationarity, i.e. more than one steady state.

This study was applied to orthodox, unorthodox, and an additional hybrid TCS variant with two binding sites. It did not however include energy effects or reversible reactions. Future work could involve combining an energy-based analysis with these techniques.

6.2.3 Bondgraphs.jl v1.0

At time of writing, BondGraphs.jl is in beta at version v0.5.0. This update includes minor bugs fixes and quality-of-life improvements but does not change substantially change the core functionality of the package. New package features are planned for future releases, with the goal of reaching the v1.0 release milestone.

Bondgraphs.jl has started to see use in other research papers [48]. Now that the package has been ‘road tested’, I plan to work with current users of the package to help improvement the interface and overall create a better user experience.

There have been major developments for two of the underlying packages, ModelingToolkit.jl and Catalyst.jl. A new macro-style interface for ModelingToolkit models makes it much easier to construct component-based models (in a similar manner to Catalyst reaction systems now). It will be beneficial to incorporate these latest changes into a future version of BondGraphs, but this will likely require a change of the internal structure and therefore a major semantic version update (i.e. v1.0).

Desired features for the package, in no particular order, include:

- Modelling more types of differential equation model, especially stochastic differential equations
- Expanded macro interface to work with other bond graph models, not just chemical reaction models
- Exporting and importing of bond graphs into common model storage formats
- Integration with CellMLToolkit.jl¹ for direct uploading/downloading of bond graph models into Julia.
- Adding unit metadata to variables for automatic compatibility checking
- Improving the component library to make it easier to create custom components
- Implement bond graph algorithms, such as causality assignment and graph simplification

¹<https://github.com/SciML/CellMLToolkit.jl>

6.2.4 Applications of stochastic bond graphs

There are several systems biology models which are worth modelling with stochastic bond graphs. Here are some examples I am particularly interested in investigating.

In section [subsection 5.6.1](#) I provided some possible extensions to the Brownian particle bond graph: adding a potential field, multiple interacting Brownian particles, and adding heat capacity. Other examples of Brownian motion worth exploring include studies on entropy production [[126](#), [148](#)], Brownian particles in a harmonic oscillator [[133](#)], and various random walks in biology [[31](#)].

The Telegraph model [[116](#), [117](#)] is a relatively simple model for bursty gene expression. The complication is that the burstiness comes from the discrete ‘on-off’ gene regulation, which is modelled with discrete distributions. This would require further extensions to the stochastic bond graph theory to allow for distributions.

Molecular motors [[118](#)] are ‘walking’ molecular machine that transport cargo across the cell interior. This is a mechano-chemical system with observed noisy behaviour, which makes it an interesting system to study with stochastic bond graphs. Another interesting cell mechanics example to study is RNA folding and unfolding [[149](#)].

6.2.5 Additional applications in biology

Two Component Systems are just one such biochemical system we could have chosen to model. There are many other candidate systems that could be studied with bond graphs in future research projects. For example, proteasomes are a compartmentalised molecular structure for deconstructing proteins and the end of their use. Liepe *et al.* [[150](#)] give ten variations of a proteasome model, which would be a good demonstration of iterative modelling with bond graphs.

Mechano-chemistry is an area of promising research for bond graphs. Gawthrop *et al.* [[151](#)] for example, model single actin filament polymerisation using bond graphs. As already mentioned, molecular motors [[118](#)] are a mechano-chemical system with interesting noisy behaviour.

Beyond cell biology, bond graphs have had many applications in biology and physiology [[25](#), [152](#)]. The Physiome Project [[53](#)] is an initiative to model the full cell physiome. Many of these physiological systems are modelled with bond graphs, since they are well-suited for modelling multiphysics, multicompartment systems. See also the Cell Physiome project [[46](#)] for recent developments in this area. Some work has been done to apply biochemical bond graphs to gene regulatory networks and synthetic biology [[48](#)].

A natural question one might have is whether bond graphs can be applied to non-physical or empirical models in biology. While there is no immediate application for bond graphs in these fields, it is an interesting question whether this would be possible. Differential equation models similar to chemical reaction networks have been used to model systems in population biology, epidemiology, and ecology [30, 31]. These mathematical similarities motivate bond graph representations, however the transformation is not straightforward.

The primary challenge is that models in population biology, epidemiology, and ecology, lack a physical basis for the equations and units. A new bond graph domain requires a definition for *effort*, *flow*, *conserved state*, and *energy*. Any new abstract definition for these quantities would have to satisfy the Hamiltonian equations, and would need to somehow be interoperable with ‘classic’ energy domains in order to get the full benefit of bond graphs. We would also require models to be ‘thermodynamically’ realistic. This would exclude or require fundamental changes to models that have a birth-death process (in ecology), or an infection/-transmission rate (in epidemiology).

6.3 Bond graphs and whole-cell modelling

In [chapter 1](#), I argued that bond graphs are well-suited for whole-cell modelling. They excel especially in biophysical modelling, dynamic simulations, molecular (and energy) closure, and modularity ([Table 1.1](#)). Bond graphs have had diverse applications in biology and physiology, and the frequency of published bond graph research continues to grow [25].

This thesis has sought to show how bond graphs can be used for the grand challenge of whole-cell modelling. In my research I have shown how current limitations may be overcome through continued model development, improved computational tools, and extended theory and definitions. In [chapter 3](#), I demonstrated how bond graphs model bacterial subsystems with a modular structure, and the biological insights of explicitly including energy dependence in models. In [chapter 4](#), I presented newly developed computational tools to accelerate model development in systems biology. In [chapter 5](#), I explored new applications of bond graphs in modelling stochastic processes, and by way of illustration, showed how new component definitions extend the theory to overcome gaps in whole-cell modelling with bond graphs.

There are still several obstacles to overcome if bond graphs are to be the desired sole mathematical framework for whole-cell modelling.

- The majority of mathematical models in existing whole-cell models will need to be converted into bond graphs. In some cases this can be automated, but it is not always clear

from the outset whether a model is physically compliant (see for example [114] and [33]).

- Model composition is also not straightforward, and common points of similarity between differently models will need to be identified [153].
- To model the lifecycle of a cell, models must somehow account for both *state* change (i.e., change in variables) and *structural* change (i.e., change in equations). Cellular automata and agent-based models are examples of formalisms that account for structural change [26]. It is not clear how bond graphs can achieve this.
- There are more mathematical frameworks that require further extensions to bond graph theory: Spatial dynamics, logical networks, jump processes, and discrete state models, to name a few. See subsection 6.2.1 for more detail.
- Bond graphs are often not the most parsimonious model for small systems. Each bond and component port defines an effort/flow pair. While thermodynamic parameterisation ensures detailed balance is satisfied, this comes at the cost of introducing additional parameters (e.g. $k = \kappa K_A K_B$). Although these variables and parameters are simplified and reduced at simulation runtime, they can add extra bloat when loading models, storing metadata, or symbolically generating equations. There are, however, algorithms and techniques for reducing model complexity with bond graphs [19, 48].
- The nomenclature is confusing for those new to bond graphs. It is not obvious why a 0-junction means “flows sum to zero” and 1-junction means “efforts sum to zero”, for example. Unclear or confusing notation creates pedagogical challenges. This can be a barrier for the adoption of bond graphs in the scientific community.

There is still much work to do if we are to produce a functional whole-cell model bond graph. In the end, it may be that to achieve this requires fundamental changes to the theory. A new mathematical formalism that builds upon bond graphs may be necessary. Nevertheless, bond graphs have the potential to be the mathematical framework for unifying systems biology models, up to whole-cell modelling and beyond.

Appendix A

Two Component Systems Supplementary Information

Supplementary Information for *Energy Dependence of Signalling Dynamics and Robustness in Bacterial Two Component Systems*[\[1\]](#).

Orthodox TCS

TABLE A.1: Chemical reaction system for our orthodox TCS model. This system is an extension on the model used by Kim and Cho [\[88\]](#). Here we explicitly include *ATP* in the reaction system with a two-step phosphorylation of HK, and allow reactions to be reversible. Note that the input stimulant *S* may be an intangible quantity such as heat or mechanical stress, and so is not conserved in the model.

Reaction Name	Chemical Reactions
Signal Activation	$HK + S \xrightleftharpoons{K_a} HKS$
Autophosphorylation	$HKS + ATP \xrightleftharpoons{K_{ap}} HKP + ADP$
Phosphotransfer	$HKP + RR \xrightleftharpoons{K_{pt}} HK + RRP$
Degradation	$RRP \xrightleftharpoons{K_d} RR + Pi$

TABLE A.2: ODE model parameters for the orthodox TCS.

Symbol	Description	Value
K_{ATP}	Thermodynamic constant	100
K_{ADP}	Thermodynamic constant	0.1
K_{Pi}	Thermodynamic constant	0.001
K_{S}	Thermodynamic constant (stimulus equivalent)	1
K_i	Thermodynamic constant for all other species i	$\{10^p : p \sim U[-1, 1]\}$
κ_j	Reaction rate for reaction j	$\{10^p : p \sim U[-1, 1]\}$
$[\text{ATP}]_0$	Chemostat concentration	$\{10^{-1}, 10^0, 10^1, 10^2, 10^3\}$
$[\text{ADP}]_0$	Chemostat concentration	0.001
$[\text{Pi}]_0$	Chemostat concentration	0.001

$$\nu_{\text{a}} = \kappa_{\text{a}} (K_{\text{HK}} K_{\text{S}} [\text{HK}] S(t) - K_{\text{HKS}} [\text{HKS}])$$

$$\nu_{\text{ap}} = \kappa_{\text{ap}} (K_{\text{HKS}} K_{\text{ATP}} [\text{HKS}] [\text{ATP}]_0 - K_{\text{HKP}} K_{\text{ADP}} [\text{HKP}] [\text{ADP}]_0)$$

$$\nu_{\text{pt}} = \kappa_{\text{pt}} (K_{\text{HKP}} K_{\text{RR}} [\text{HKP}] [\text{RR}] - K_{\text{HK}} K_{\text{RRP}} [\text{HK}] [\text{RRP}])$$

$$\nu_{\text{d}} = \kappa_{\text{d}} (K_{\text{RRP}} [\text{RRP}] - K_{\text{RR}} K_{\text{Pi}} [\text{RR}] [\text{Pi}]_0)$$

$$\frac{d[\text{HK}]}{dt} = -\nu_{\text{a}} + \nu_{\text{pt}}$$

$$\frac{d[\text{HKS}]}{dt} = \nu_{\text{a}} - \nu_{\text{ap}}$$

$$\frac{d[\text{HKP}]}{dt} = \nu_{\text{ap}} - \nu_{\text{pt}}$$

$$\frac{d[\text{RR}]}{dt} = -\nu_{\text{pt}} + \nu_{\text{d}}$$

$$\frac{d[\text{RRP}]}{dt} = \nu_{\text{pt}} - \nu_{\text{d}}$$

TABLE A.3: Chemical reaction system for our unorthodox TCS model. This system is an extension on the model used by Kim and Cho [88]. Here we explicitly include *ATP* in the reaction system with a two-step phosphorylation of HK, and allow reactions to be reversible. Note that the input stimulant *S* may be an intangible quantity such as heat or mechanical stress, and so is not conserved in the model.

Reaction Name	Chemical Reactions
Signal Activation	$\text{HK1} + \text{S} \xrightleftharpoons{K_{a1}} \text{HK1S}$ $\text{HK3} + \text{S} \xrightleftharpoons{K_{a2}} \text{HK3S}$ $\text{HK4} + \text{S} \xrightleftharpoons{K_{a3}} \text{HK4S}$ $\text{HK7} + \text{S} \xrightleftharpoons{K_{a4}} \text{HK7S}$
Autophosphorylation	$\text{HK1S} + \text{ATP} \xrightleftharpoons{K_{ap1}} \text{HK2} + \text{ADP}$ $\text{HK3S} + \text{ATP} \xrightleftharpoons{K_{ap2}} \text{HK5} + \text{ADP}$ $\text{HK4S} + \text{ATP} \xrightleftharpoons{K_{ap3}} \text{HK6} + \text{ADP}$ $\text{HK7S} + \text{ATP} \xrightleftharpoons{K_{ap4}} \text{HK8} + \text{ADP}$
Forward Phosphorelay	$\text{HK2} \xrightleftharpoons{K_{f1}} \text{HK3}$ $\text{HK3} \xrightleftharpoons{K_{f2}} \text{HK4}$ $\text{HK5} \xrightleftharpoons{K_{f3}} \text{HK6}$ $\text{HK6} \xrightleftharpoons{K_{f4}} \text{HK7}$
Phosphotransfer	$\text{HK4} + \text{RR} \xrightleftharpoons{K_{pt1}} \text{HK1} + \text{RRP}$ $\text{HK6} + \text{RR} \xrightleftharpoons{K_{pt2}} \text{HK2} + \text{RRP}$ $\text{HK7} + \text{RR} \xrightleftharpoons{K_{pt3}} \text{HK3} + \text{RRP}$ $\text{HK8} + \text{RR} \xrightleftharpoons{K_{pt4}} \text{HK5} + \text{RRP}$
Degradation	$\text{HK8} \xrightleftharpoons{K_{d1}} \text{HK6} + \text{Pi}$ $\text{HK5} \xrightleftharpoons{K_{d2}} \text{HK2} + \text{Pi}$ $\text{HK3} \xrightleftharpoons{K_{d3}} \text{HK1} + \text{Pi}$ $\text{HK7} \xrightleftharpoons{K_{d4}} \text{HK4} + \text{Pi}$ $\text{RRP} \xrightleftharpoons{K_{d5}} \text{RR} + \text{Pi}$

TABLE A.4: ODE model parameters for the unorthodox TCS.

Symbol	Description	Value(s)
K_{ATP}	Thermodynamic constant	100
K_{ADP}	Thermodynamic constant	0.1
K_{Pi}	Thermodynamic constant	0.001
K_{S}	Thermodynamic constant (stimulus equivalent)	1
K_i	Thermodynamic constant for all other species i	$\{10^p : p \sim U[-1, 1]\}$
κ_j	Reaction rate for reaction j	$\{10^p : p \sim U[-1, 1]\}$
$[\text{ATP}]_0$	Chemostat concentration	$\{10^{-1}, 10^0, 10^1, 10^2, 10^3\}$
$[\text{ADP}]_0$	Chemostat concentration	0.001
$[\text{Pi}]_0$	Chemostat concentration	0.001

Unorthodox TCS

$$\begin{aligned}
\nu_{a1} &= \kappa_{a1} (K_{HK1}K_S[HK1]S(t) - K_{HK1S}[HK1S]) \\
\nu_{a2} &= \kappa_{a2} (K_{HK3}K_S[HK3]S(t) - K_{HK3S}[HK3S]) \\
\nu_{a3} &= \kappa_{a3} (K_{HK4}K_S[HK4]S(t) - K_{HK4S}[HK4S]) \\
\nu_{a4} &= \kappa_{a4} (K_{HK7}K_S[HK7]S(t) - K_{HK7S}[HK7S]) \\
\nu_{ap1} &= \kappa_{ap1} (K_{HK1S}K_{ATP}[HK1S][ATP]_0 - K_{HK2}K_{ADP}[HK2][ADP]_0) \\
\nu_{ap2} &= \kappa_{ap2} (K_{HK3S}K_{ATP}[HK3S][ATP]_0 - K_{HK5}K_{ADP}[HK5][ADP]_0) \\
\nu_{ap3} &= \kappa_{ap3} (K_{HK4S}K_{ATP}[HK4S][ATP]_0 - K_{HK6}K_{ADP}[HK6][ADP]_0) \\
\nu_{ap4} &= \kappa_{ap4} (K_{HK7S}K_{ATP}[HK7S][ATP]_0 - K_{HK8}K_{ADP}[HK8][ADP]_0) \\
\nu_{f1} &= \kappa_{f1} (K_{HK2}[HK2] - K_{HK3}[HK3]) \\
\nu_{f2} &= \kappa_{f2} (K_{HK3}[HK3] - K_{HK4}[HK4]) \\
\nu_{f3} &= \kappa_{f3} (K_{HK5}[HK5] - K_{HK6}[HK6]) \\
\nu_{f4} &= \kappa_{f4} (K_{HK6}[HK6] - K_{HK7}[HK7]) \\
\nu_{pt1} &= \kappa_{pt1} (K_{RR}K_{HK4}[RR][HK4] - K_{RRP}K_{HK1}[RRP][HK1]) \\
\nu_{pt2} &= \kappa_{pt2} (K_{RR}K_{HK6}[RR][HK6] - K_{RRP}K_{HK2}[RRP][HK2]) \\
\nu_{pt3} &= \kappa_{pt3} (K_{RR}K_{HK7}[RR][HK7] - K_{RRP}K_{HK3}[RRP][HK3]) \\
\nu_{pt4} &= \kappa_{pt4} (K_{RR}K_{HK8}[RR][HK8] - K_{RRP}K_{HK5}[RRP][HK5]) \\
\nu_{d1} &= \kappa_{d1} (K_{HK8}[HK8] - K_{HK6}K_{Pi}[HK6][Pi]_0) \\
\nu_{d2} &= \kappa_{d2} (K_{HK5}[HK5] - K_{HK2}K_{Pi}[HK2][Pi]_0) \\
\nu_{d3} &= \kappa_{d3} (K_{HK3}[HK3] - K_{HK1}K_{Pi}[HK1][Pi]_0) \\
\nu_{d4} &= \kappa_{d4} (K_{HK7}[HK7] - K_{HK4}K_{Pi}[HK4][Pi]_0) \\
\nu_{d5} &= \kappa_{d5} (K_{RRP}[RRP] - K_{RR}[RR][Pi]_0)
\end{aligned}$$

$$\begin{aligned}
\frac{d[\text{HK1}]}{dt} &= -\nu_{a1} + \nu_{pt1} + \nu_{d3} \\
\frac{d[\text{HK1S}]}{dt} &= \nu_{a1} - \nu_{ap1} \\
\frac{d[\text{HK2}]}{dt} &= \nu_{ap1} - \nu_{f1} + \nu_{pt2} + \nu_{d2} \\
\frac{d[\text{HK3}]}{dt} &= -\nu_{a2} + \nu_{f1} - \nu_{f2} + \nu_{pt3} - \nu_{d3} \\
\frac{d[\text{HK3S}]}{dt} &= \nu_{a2} - \nu_{ap2} \\
\frac{d[\text{HK4}]}{dt} &= -\nu_{a3} + \nu_{f2} - \nu_{pt1} + \nu_{d4} \\
\frac{d[\text{HK4S}]}{dt} &= \nu_{a3} - \nu_{ap3} \\
\frac{d[\text{HK5}]}{dt} &= \nu_{ap2} - \nu_{f3} + \nu_{pt4} - \nu_{d2} \\
\frac{d[\text{HK6}]}{dt} &= \nu_{ap3} + \nu_{f3} - \nu_{f4} - \nu_{pt2} + \nu_{d1} \\
\frac{d[\text{HK7}]}{dt} &= -\nu_{a4} + \nu_{f4} - \nu_{pt3} - \nu_{d4} \\
\frac{d[\text{HK7S}]}{dt} &= \nu_{a4} - \nu_{ap4} \\
\frac{d[\text{HK8}]}{dt} &= \nu_{ap4} - \nu_{pt4} - \nu_{d1} \\
\frac{d[\text{RR}]}{dt} &= -\nu_{pt1} - \nu_{pt2} - \nu_{pt3} - \nu_{pt4} + \nu_{d5} \\
\frac{d[\text{RRP}]}{dt} &= \nu_{pt1} + \nu_{pt4} + \nu_{pt2} + \nu_{pt3} - \nu_{d5}
\end{aligned}$$

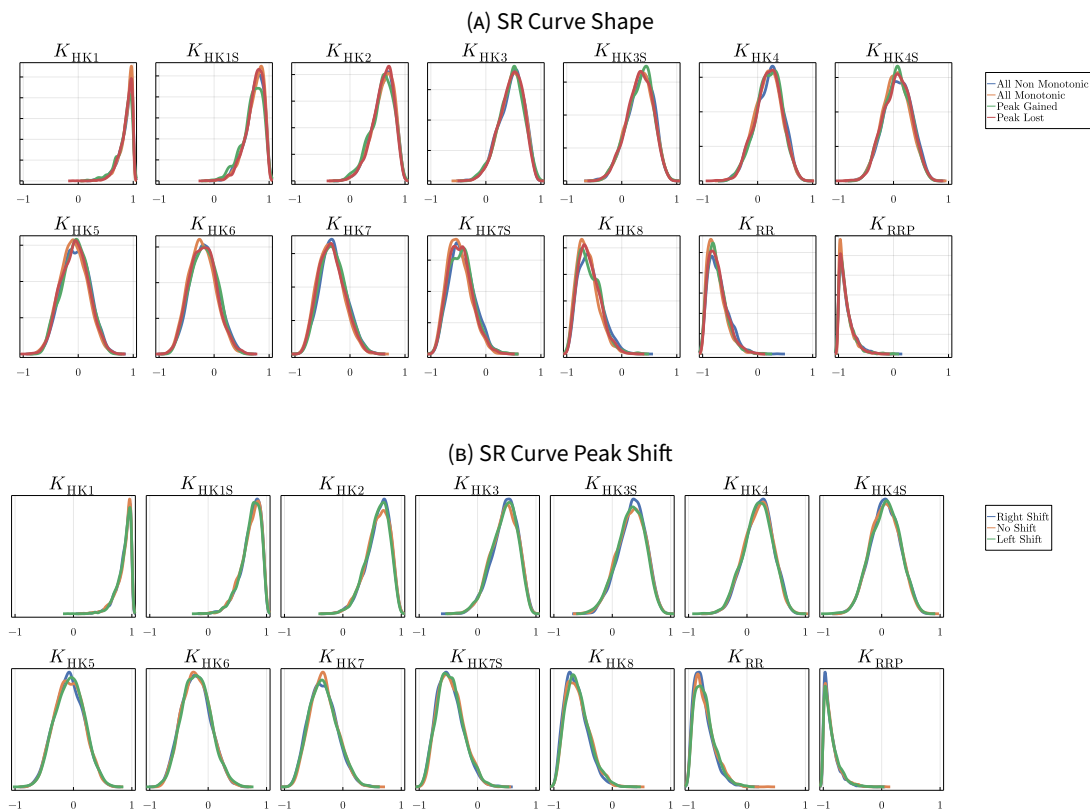


FIGURE A.1: Log distribution of the sampled thermodynamic constants K used for the unorthodox TCS numerical simulations, split by **(a)** SR curve shape and **(b)** SR peak shift. The thermodynamic constants defined in Table A.4. Parameters are sampled from a log-uniform distribution (Equation 3.8). The distributions for these parameters all follow roughly the same distribution, so we observe that these parameters do not affect SR shape or peak-shift classification (contrast with Figure 3.11 in the main text). Note that these parameters follow a skewed normal distribution trend due to how the parameters are sampled; see Methods section in main text for more details.

TABLE A.5: Two Component Systems (TCS) found in *E. coli*, with their recorded cellular functions. First table lists orthodox TCS; second table lists unorthodox TCS. The average amino acid length for the orthodox proteins are less than the corresponding unorthodox proteins. Source: EcoCyc Database [72]

HK/RR	Amino acid lengths	Function
AtoS/AtoC	608/461	Metabolism
BaeS/BaeR	467/240	Envelope homeostasis
BasS/BasR	363/222	Polymyxin resistance
BtsS/BtsR	561/239	Pyruvate sensing
CheA/CheB	654/349	Chemotaxis
CheA/CheY	654/129	Chemotaxis
CpxA/CpxR	457/232	Envelope homeostasis
CreC/CreB	474/229	Regulates aerobic growth
CusS/CusR	480/227	Copper response
DcuS/DcuR	543/239	Dicarboxylates response
DpiB/DpiA	552/226	Citrate fermentation
EnvZ/OmpR	450/239	Osmolarity sensing
GlrK/GlrR	475/444	Envelope homeostasis
HprS/HprR	452/223	Hydrogen peroxide sensitivity
KdpD/KdpE	894/225	Potassium transport
NarQ/NarP	566/215	Response to nitrite
NarX/NarL	598/216	Response to nitrite
NtrB/NtrC	349/470	Nitrogen response
PhoQ/PhoP	486/223	Response to magnesium
PhoR/PhoB	431/229	Phosphate regulation
PyrS/PyrR	565/244	Pyruvate sensing
QseC/QseB	449/219	Cell motility
RcsD/RcsB	890/216	Envelope homeostasis
RstB/RstA	433/239	Acid tolerance
UhpB/UhpA	500/196	Metabolism
ZraS/ZraR	465/441	Envelope homeostasis
Mean	531/263	
ArcB/ArcA	778/238	Respiratory control
BarA/UvrY	918/218	Hydrogen peroxide sensitivity
EvgS/EvgA	1197/204	Drug resistance
TorS/TorR	915/230	Anaerobic energy production
Mean	952/223	

Appendix B

BondGraphs.jl Supplementary Information

B.1 Bond graph theory

This section covers the basic principles of bond graph modelling. [Table B.1](#) summarises the variable and component analogies discussed in this section.

B.1.1 Construction

Bond graphs represent all systems and reactions in terms of energy flow between elements of a system. Physical variables are described in terms of abstracted *efforts* (forces, voltages, chemical potentials) and *flows* (velocities, currents, molar flow rate). Efforts are always in terms of Joules per some quantity x , and flows are always x per second, so that $J/x \cdot x/s = J/s$, or energy change over time. By abstracting these units, any of the above efforts and flows can be combined in a single model that conserves energy across all domains.

Bond graphs are built with collections of components. Components are generalised forms of physical laws and typically represent a physical element of the system, such as a resistor, point mass, or chemical species. Each component stores a constitutive relation that describes the relationship between the energetic *effort* [Joules/ x] and energy *flow* [x /second] in the component, where x is a measured quantity (displacement, charge, moles). BondGraphs.jl includes a library of common bond graph component types. These include the **C** type components (capacitors, chemical species), **R** type components (resistors, reactions), **I** type components (inductors, point masses), and energy sources (batteries, chemostats). For

more examples and details of bond graph components, refer to Gawthrop & Bevan [20] and Borutzky [19].

The biochemical reaction component **Re** differs from the standard **R** component in that it is a non-linear function of two efforts (chemical potentials). In other words, a distinction is made between the potential of the *reactants* and the potential of the *products*. In bond graph terms, the **Re** component has two ports instead of one. In other respects, the analogy with the resistive **R** component still holds, as they are all energy 'sinks' through the release of thermal energy.

The standard constitutive equations for a biochemical bond graph produce mass-action rate laws. Other rate laws may be specified. In the BondGraphs.jl documentation, there is an example of implementing a Michaelis-Menten rate law in place of mass-action kinetics (<https://jedforrest.github.io/BondGraphs.jl/stable/examples/>). Any rate law can be used in principle, so long as it is thermodynamically consistent.

Components are joined with a bond, representing the energy transfer between components. Each bond has an associated effort and flow variable and a direction that determines sign convention. Bonds may also connect to junctions, which describe particular conservation laws. These are either *Equal Effort* (effort is equal in connected components) or *Equal Flow* (flow is identical). In bond graph notation, these are equivalent to the **0** and **1** junctions, respectively. Intuitively, these can be thought of as junctions that conserve the (typically) 0D effort and 1D flow.

TABLE B.1: Bond graph variables and analogous components for common physical domains.

	Bond Graph	Mechanical	Electrical	Chemical
Quantity [x]	x	[metres]	[coulombs]	[moles]
Effort [Joules/x]	e	Force	Voltage	Chemical potential
Flow [x/second]	f	Velocity	Current	Molar flow rate
Integrated Effort	$p = \int e dt$	Momentum	Magnetic flux	n/a
Integrated Flow	$q = \int f dt$	Displacement	Charge	Concentration
Compliance	C	Spring	Capacitor	Species
Resistance	R	Damper	Resistor	Reaction
Impedance	I	Inertia	Inductor	n/a
Transform	TF	Lever	Transformer	Stoichiometry

B.1.2 Thermodynamic consistency

Since bonds are two-way energy connectors, the energy of the entire system is always conserved. (Assuming a closed system. In an open system, energy can be dissipated or added,

but it can always be tracked.) A similar rule applies to mass conservation or charge conservation. Bond graphs therefore by design enforce physical and thermodynamic constraints. More precisely, bond graphs satisfy energy conservation as all bonds and junctions are power conserving. While entropy is not always explicitly shown, the components can be modified to account for temperature and entropy flow [for example in 19].

However, it is up to the modeller to ensure that the components are ‘well-formed’ and physically plausible. In other words, if an arbitrary component’s constitutive law does not conserve energy (for example, through the use of irreversible rate laws), then the system as a whole will not conserve energy. Nonetheless, the bonds, ports, and junctions in the bond graph framework are thermodynamically safe. Thus, model composition across scales and domains will not break physical constraints. This makes it possible to construct realistic large-scale biophysical models.

Therefore, there is some onus on the modeller to ensure that newly designed components are physically plausible. However, existing standard components (such as the Reaction component **Re**) are physically plausible. Pre-made components in the BondGraphs.jl library all satisfy the laws of thermodynamics.

Specific to biochemistry, bond graphs are able to automatically satisfy detailed balance constraints in chemical reaction network models. This is in contrast to typical mass action models with kinetic parameters, where additional detailed balance constraints need to be identified and enforced [154]. Bond graphs assign a parameter for each species (thermodynamic constant $K_i[\text{mol}^{-1}]$) and reaction (reaction rate $\kappa_j[\text{mol} \cdot \text{s}^{-1}]$) in the system. In contrast, kinetic parameters in conventional models are combinations of thermodynamic parameters, and changing the value of one kinetic parameter without accounting for changes in the other kinetic rates may result in a thermodynamically infeasible system. Previous bond graph papers have addressed this issue [33, 155] and C. Mason & W. Covert [156] benchmark the fitting of thermodynamic parameters to data compared to conventional kinetic parameters.

Bond graphs have the additional benefit of reducing the parameter space to thermodynamically feasible regions. In this sense, they have a similar benefit to convenience kinetics and similar frameworks. A discussion about convenience kinetics and bond graphs is given in [23].

For further reading on how bond graphs satisfy detailed balance, see [33]. For a discussion on dealing with uncertainty in thermodynamic parameters, we refer to [104].

B.2 Model descriptions

Model specifications and code for the two models used in Figure 1 in the Section 3 of the main text: the ion pore transport model Cudmore *et al.* [54], and; the Sarco/Endoplasmic Reticulum Ca_2^+ -ATPase (SERCA) pump Pan *et al.* [33].

We have included an additional example on a large (281 species, 566 reactions) *Saccharomyces cerevisiae* (yeast) metabolic network Stanford *et al.* [115]. This example features automatically generating a bond graph from a published systems biology model stored in BioModels Malik-Sheriff *et al.* [157].

Each section includes the generated ODE system of equations (or a subset of the equations), model parameters and initial conditions (where applicable), and the Julia code used to create the bond graph models and plots in Figure 1.

B.2.1 Ion Transport

ODE

$$\frac{dV_m}{dt} = \frac{zFr}{C_m} \left[K_e c_e \exp\left(\frac{-zFV_m}{2RT}\right) - K_i c_i \exp\left(\frac{zFV_m}{2RT}\right) \right]$$

V_m is the membrane voltage, z is the ion charge (here ± 1), $C_m = 1 \mu\text{F}/\text{cm}^2$ is the membrane capacitance, $F = 96485 \text{ C/mol}$ is the Faraday constant, $R = 8.314 \text{ J/mol/K}$ is the molar gas constant, T is the absolute temperature, K_i , K_e are the internal/external thermodynamic constants, and c_i , c_e are the internal/external ion concentrations.

Parameters and Initial Conditions

TABLE B.2: Internal and external concentrations and charges for the ions Na^+ , K^+ , and Cl^- . Taken from Cudmore *et al.* [54].

Ion	charge z	c_{in} [M]	c_{ex} [M]
Na^+	+1	0.019	0.155
K^+	+1	0.136	0.005
Cl^-	-1	0.078	0.112

Julia Code

```

using BondGraphs, Plots, GraphMakie
import CairoMakie as CM

# Define a function to create a bond graph for each ion
function ion_pore(name=""; z=1, c_ex=1e-3, c_in=1e-3)
    bg = BondGraph(name * " Ion Transport")

    # Create components
    membrane = Component(:C, "Mem"; C=1)
    ion_ex = Component(:SCe, "Ie"; K=1, xs=t->c_ex)
    ion_in = Component(:SCe, "Ii"; K=1, xs=t->c_in)
    potential_mem = EqualEffort()
    flow_f, flow_r = EqualFlow(), EqualFlow()
    TF_F = Component(:TF, "F"; n=96485)
    TF_zf = Component(:TF, "-z/2"; n=-z/2)
    TF_zr = Component(:TF, "z/2"; n=z/2)
    re_pore = Component(:Re, "Pore"; r=1e-7)

    # Add and connect components within bond graph
    allcomps = [
        membrane, ion_ex, ion_in,
        potential_mem, flow_f, flow_r,
        TF_F, TF_zr, TF_zf, re_pore
    ]
    add_node!(bg, allcomps)
    connect!(bg, ion_ex, flow_f)
    connect!(bg, flow_f, re_pore)
    connect!(bg, re_pore, flow_r)
    connect!(bg, flow_r, ion_in)
    connect!(bg, flow_r, (TF_zr,2))
    connect!(bg, (TF_zr,1), potential_mem)
    connect!(bg, potential_mem, (TF_zf,1))
    connect!(bg, (TF_zf,2), flow_f)
    connect!(bg, potential_mem, (TF_F,2))
    connect!(bg, (TF_F,1), membrane)
    return bg
end

# Plot an ion pore bond graph
ion_pore_bg = ion_pore()

```

```

fig = graphplot(ion_pore_bg)
CM.save("ion_transport_bg.svg", fig)

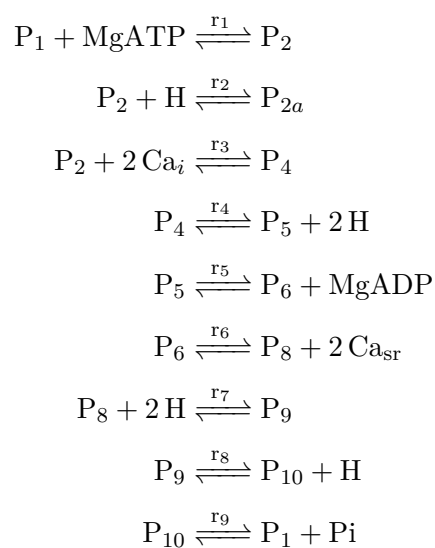
# Ion specific parameters
ions = Dict(
    "Na+" => (z=1, c_ex=155/1000, c_in=19/1000),
    "Cl-" => (z=-1, c_ex=112/1000, c_in=78/1000),
    "K+"  => (z=1, c_ex=5/1000, c_in=136/1000),
)

# Solve and plot each solution for the membrane potential
plot()
for (ionname, params) in ions
    model = ion_pore(ionname; params...)
    sol = simulate(model, (0., 1000.))
    plot!(sol, label=ionname, lw=3, xlabel="Time [ms]",
        ↪ ylabel="Membrane potential [V]")
end
plot!(; legend=:topright)

```

B.2.2 SERCA

Chemical Equations



ODE

$$\begin{aligned}
\frac{d[P_1]}{dt} &= r_9 (K_{10}[P_{10}] - K_1 K_{Pi}[P_1][Pi]_0) - r_1 (K_{MgATP} K_1 [MgATP]_0 [P_1] - K_2 [P_2]) \\
\frac{d[P_2]}{dt} &= r_1 (K_{MgATP} K_1 [MgATP]_0 [P_1] - K_2 [P_2]) - r_2 (-K_{P2a}[P_{2a}] + K_H K_2 [H]_0 [P_2]) \\
&\quad - r_3 (-K_4 [P_4] + K_{Ca_i}^2 K_2 [Ca_i]_0^2 [P_2]) \\
\frac{d[P_{2a}]}{dt} &= r_2 (-K_{P2a}[P_{2a}] + K_H K_2 [H]_0 [P_2]) \\
\frac{d[P_4]}{dt} &= r_3 (-K_4 [P_4] + K_{Ca_i}^2 K_2 [Ca_i]_0^2 [P_2]) + r_4 (-K_4 [P_4] + K_H^2 K_5 [H]_0^2 [P_5]) \\
\frac{d[P_5]}{dt} &= r_5 (-K_5 [P_5] + K_{MgADP} K_6 [MgADP]_0 [P_6]) + r_4 (K_4 [P_4] - K_H^2 K_5 [H]_0^2 [P_5]) \\
\frac{d[P_6]}{dt} &= r_5 (K_5 [P_5] - K_{MgADP} K_6 [MgADP]_0 [P_6]) - r_6 (K_6 [P_6] - K_{Ca_{sr}}^2 K_8 [Ca_{sr}]_0^2 [P_8]) \\
\frac{d[P_8]}{dt} &= r_6 (K_6 [P_6] - K_{Ca_{sr}}^2 K_8 [Ca_{sr}]_0^2 [P_8]) - r_7 (-K_9 [P_9] + K_H^2 K_8 [H]_0^2 [P_8]) \\
\frac{d[P_9]}{dt} &= r_7 (-K_9 [P_9] + K_H^2 K_8 [H]_0^2 [P_8]) - r_8 (K_9 [P_9] - K_H K_{10} [H]_0 [P_{10}]) \\
\frac{d[P_{10}]}{dt} &= r_8 (K_9 [P_9] - K_H K_{10} [H]_0 [P_{10}]) - r_9 (K_{10}[P_{10}] - K_1 K_{Pi}[P_1][Pi]_0)
\end{aligned}$$

Concentrations P_i and reaction rates r_j correspond to the chemical reaction network above. K_i is the thermodynamic constant for species i . Chemostats X are chemical species where the concentration is assumed constant ($\dot{X} = 0$). $[X]_0$ are the chemostat concentrations.

Parameters and Initial Conditions

TABLE B.3: Parameter values and initial conditions for the states used in the SERCA bond graph model. Taken from Pan *et al.* [33]. The input function for Casr was chosen to resemble a physiological calcium spike, characterised by an initial spike in the concentration of Ca^{2+} followed by an exponential decay.

Reaction	Rate [$\text{fmol}\cdot\text{s}^{-1}$]	Species	Affinity [fmol^{-1}]
r_1	0.00053004	P1	5263.6085
r_2	8326784.0537	P2	3803.6518
r_3	1567.7476	P2a	3110.4445
r_4	1567.7476	P4	16520516.1239
r_5	3063.4006	P5	0.82914
r_6	130852.3839	P6	993148.433
r_7	11612934.8748	P8	37.7379
r_8	11612934.8748	P9	2230.2717
r_9	0.049926	P10	410.6048
		Cai	1.9058
		Casr	31.764
		MgATP	244.3021
		MgADP	5.8126e-7
		Pi	0.014921
		H	1862.5406

Species	$x(t=0)$ [fmol]
P1	0.000483061870385487
P2	0.0574915174273067
P2a	0.527445119834607
P4	1.51818391164022e-09
P5	0.000521923287622898
P6	7.80721128535043e-05
P8	0.156693953834181
P9	0.149232225342376
P10	0.108044124948978

Chemostat	Concentration [fmol]
Cai	0.0057
H	0.004028
MgADP	1.3794
MgATP	3.8
Pi	570
Casr	$\frac{2.5 \sin(0.2t)}{(t^2+0.5)} + 0.2 \exp\left(\frac{-t}{100}\right)$

Julia Code

```
using BondGraphs, Catalyst, Plots, GraphMakie
import CairoMakie as CM
import DifferentialEquations: Rosenbrock23
```

```

# Generate the bond graph from a chemical reaction network
rn_serca = @reaction_network SERCA begin
    (1, 1), P1 + MgATP <--> P2
    (1, 1), P2 + H <--> P2a
    (1, 1), P2 + 2Cai <--> P4
    (1, 1), P4 <--> P5 + 2H
    (1, 1), P5 <--> P6 + MgADP
    (1, 1), P6 <--> P8 + 2Casr
    (1, 1), P8 + 2H <--> P9
    (1, 1), P9 <--> P10 + H
    (1, 1), P10 <--> P1 + Pi
end
chemostats = ["MgATP", "MgADP", "Pi", "H", "Cai", "Casr"]
bg_serca = BondGraph(rn_serca; chemostats)
fig = graphplot(bg_serca)
CM.save("serca_bg.svg", fig)

# Define reaction rates, affinities, chemostats, and initial
↪ conditions
ca_spike(t) = 2.5*sin(t/5)/(t^2 + 0.5) + 0.2*exp(-t/100)
reaction_rates = [
    :R1 => 0.00053004,
    ... # defined as in above table
]
species_affinities = [
    :P1 => 5263.6085,
    ... # defined as in above table
]
chemostat_amounts = [
    :Casr => t -> ca_spike(t),
    ... # defined as in above table
]
initial_conditions = [
    :P1 => 0.000483061870385487,
    ... # defined as in above table
]

for (reaction, rate) in reaction_rates
    getproperty(bg_serca, reaction).r = rate
end
for (species, affinity) in species_affinities
    getproperty(bg_serca, species).K = affinity

```

```
end
for (chemostat, amount) in chemostat_amounts
    getproperty(bg_serca, chemostat).xs = amount
end
for (species, ic) in initial_conditions
    getproperty(bg_serca, species).q = ic
end

# Solve the differential equations generated by the bond graph
sol = simulate(bg_serca, (0., 20.); solver=Rosenbrock23())
plot(sol, lw=3, xlabel="Time [ms]", ylabel="Amount [fmol]",
     ↪ legend=:right)
savefig("serca_numerical_solution.svg")
```

B.2.3 *Saccharomyces cerevisiae* Metabolic Network

This example demonstrates how one could generate a bond graph model of a large biological network stored in a computer-readable format. For this example, we have used the *Saccharomyces cerevisiae* metabolic network from Stanford *et al.* [115]. This model contains 281 species and 566 reactions. The SBML model file contains information on relevant chemical species, reaction stoichiometry, default parameter values, and various other metadata. The file is downloadable from <https://doi.org/10.1371/journal.pone.0079195.s002>.

The bond graph is constructed using the chemical reaction network interface in BondGraphs.jl. The bond graph in total contains 1019 components, 1127 junctions, and 3813 bonds. We assume mass-action rate laws for simplicity and demonstration purposes; in practice, one may update the **Re** reaction components with more sophisticated rate laws.

ODE

The system of ODEs generated by BondGraphs.jl has 293 equations. For the sake of conciseness, only one representative equation is shown here.

$$\begin{aligned}
 & \vdots \\
 \frac{dS_{0616}}{dt} &= r_{69} (K_{0615} K_{0706} S_{0615} S_{0706} - K_{0616} K_{0710} S_{0616} S_{0710}) \\
 & \quad + r_{38} (K_{0615} K_{0763b} S_{0615} S_{0763b} - K_{0616} K_{0763b} K_{1011} S_{0616} S_{0763b} S_{1011}) \\
 & \quad - r_{37} (-K_{0615} S_{0615} + K_{0616} K_{1011} S_{0616} S_{1011}) \\
 & \vdots
 \end{aligned}$$

S_i are the species amounts, K_i are the thermodynamic constants, r_j are the reaction rates. Species and reactions are indexed according to Stanford *et al.* [115].

Julia Code

```

using BondGraphs, SBMLToolkit, ModelingToolkit, Catalyst, Graphs,
    Plots

# Source: 'Systematic Construction of Kinetic Models from
    Genome-Scale Metabolic Networks' (Stanford et al. 2013)
# https://doi.org/10.1371/journal.pone.0079195
sbmlfile = "yeast_model.xml"
sbmlurl = "https://doi.org/10.1371/journal.pone.0079195.s002"

# Download and process the SBML file
!isfile(sbmlfile) && download(sbmlurl, sbmlfile)
SBMLToolkit.checksupport_file(sbmlfile)
mdl = readSBML(sbmlfile, doc -> begin
    set_level_and_version(3, 2)(doc)
    convert_promotelocals_expandfuns(doc)
end)

# Remove reactions from reaction network that have a 'null' species
# i.e. remove  $\emptyset \rightarrow X$  and  $Y \rightarrow \emptyset$ 
isnullreaction(reaction) = isempty(reaction.substrates) ||
    isempty(reaction.products)
function reducednetwork(network)

```

```
newnetwork = make_empty_network(name=network.name)
[addparam!(newnetwork, p) for p in parameters(network)]
for rx in reactions(network)
    if !isnullreaction(rx)
        [addspecies!(newnetwork, sub) for sub in rx.substrates]
        [addspecies!(newnetwork, prod) for prod in rx.products]
        addreaction!(newnetwork, rx)
    end
end
newnetwork
end

# Create the bond graph
rs = reducednetwork(ReactionSystem mdl)
bg = BondGraph(rs)

# OUTPUTS
# Size of system
length(components(bg)) # 1019 components
length(junctions(bg))  # 1127 junctions
length(bonds(bg))      # 3813 bonds

# Adjacency matrix plot
A = adjacency_matrix(bg) # 2146×2146 SparseMatrix
plot(spy(A), size=(1000,800))
savefig("yeast_model_adjmat.png")

# Graph plot
plot(bg, size=(1000,800), names=[], title="")
savefig("yeast_model_bg.png")

# ODE equations
eqns = equations(bg)
length(eqns) # 293
eqns[3] # only displaying equation 3 for this example
```

Figures

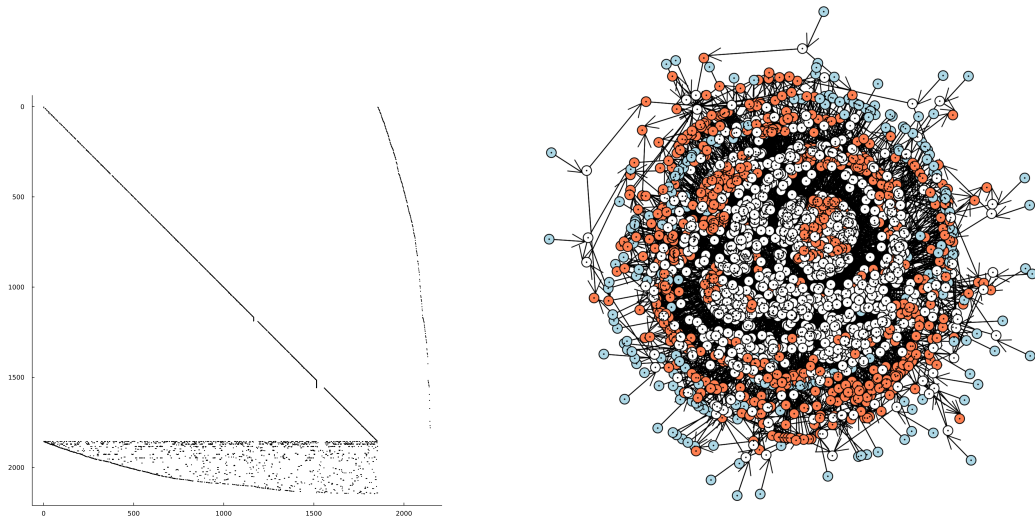


FIGURE B.1: Bond graph model of a yeast metabolic network [taken from 115]. **Left:** 2146×2146 adjacency matrix for the yeast network bond graph. Axes correspond to the indices $a_{i,j}$ of the adjacency matrix, where $a_{i,j} = 1$ (black dot) means node i (component or junction) is connected to node j . **Right:** Graph plot of the yeast network bond graph. The graph contains 1019 components (blue for species, orange for reactions and transformers), 1127 junctions (white), and 3813 bonds.

Appendix C

BondGraphs.jl v0.4.0 Documentation

Contents

Contents	i
1 Introduction	1
1.1 Overview	1
1.2 Installation	1
1.3 Tutorials	1
2 Getting Started	2
2.1 Bond graph construction	2
2.2 Simulating our model	4
2.3 Adding control variables	6
3 Examples	9
3.1 Simple Electric Circuit	9
3.2 Biochemical Reaction Networks	10
3.2.1 A simple biochemical example	10
3.2.2 Stoichiometry	12
3.2.3 The reversible Michaelis-Menten model	14
3.3 SERCA Pump	15
3.4 Electrochemical System: Ion Pore Transport	19
3.5 Enzyme Catalysed Reactions	21
4 API Reference	25
4.1 Reference	25
4.1.1 Types	26
4.1.2 Methods	27

Chapter 1

Introduction

1.1 Overview

[BondGraphs.jl](#) is a Julia implementation of the bond graph framework. Bond graphs are an energy-based modelling framework that describe energy flow through a physical system, and are especially useful for modelling multi-scale or multi-physical systems.¹

This package constructs a symbolic graph model of a physical system, which can then be converted into a system of differential equations. BondGraphs.jl includes specific methods that interact with the wider Julia modelling ecosystem, including ModelingToolkit.jl, Catalyst.jl, and Plots.jl.

- [Getting Started](#)
- [Examples](#)
- [Reference](#)

1.2 Installation

```
using Pkg; Pkg.add("BondGraphs")  
using BondGraphs
```

1.3 Tutorials

For tutorials and examples, refer to the [Examples](#) page. For interactive Jupyter Notebook tutorials, refer to [BondGraphsTutorials](#) on GitHub.

¹Gawthrop and Bevan, *Bond-graph modeling* (2007)

Chapter 2

Getting Started

This example runs through the basic steps of building and simulating a bond graph model. For a full list of functions refer to the API [Reference](#).

Our first example will be a simple electric circuit of a capacitor, resistor, and current supply in parallel. We will first model this circuit without the current supply.

2.1 Bond graph construction

We first create a BondGraph object which will hold all our components. This is an empty object that will soon be populated with components and bonds.

```
model = BondGraph("RC Circuit")
```

```
BondGraph RC Circuit (0 Nodes, 0 Bonds)
```

Next we will create a capacitor as a bond graph 'C'-component. The component type's description can be printed for extra information.

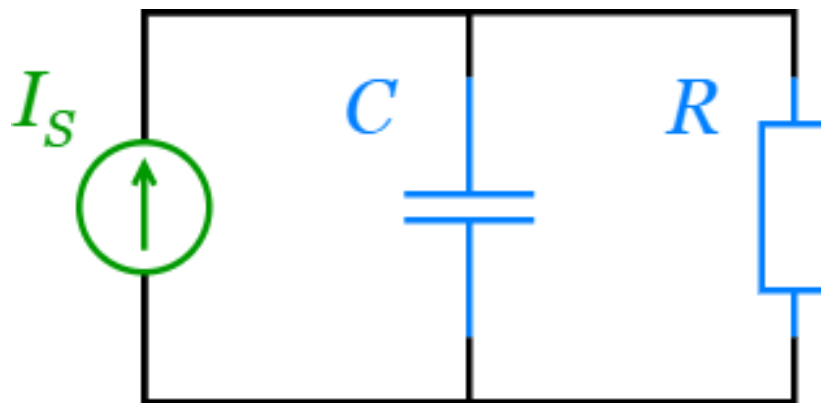


Figure 2.1:

```
C = Component(:C)
description(:C)
```

```
Generalised Linear Capacitor
e = (1/C)*q
dq/dt = f
C: Capacitance [1.0]
q: Generalised position [0.0]
```

Available component types are defined in `BondGraphs.DEFAULT_LIBRARY`. The default component types are:

- Standard components (:C, :R, :I, :TF)
- Biochemical components (:Ce, :Re, :ce, :re)
- Sources of effort or flow (:Se, :Sf, :SCe)

We next create a resistor 'R'-component and an `EqualEffort` node which represents Kirchoff's Voltage Law. In bond graph terminology, 0-junctions are `EqualEffort` nodes, and 1-junctions are `EqualFlow` nodes.

```
R = Component(:R)
kvl = EqualEffort()
```

```
0
```

Components and nodes are added to the model, and connected together as a graph network. Note that components must first be added to the model before they can be connected.

```
add_node!(model, [C, R, kvl])
connect!(model, R, kvl)
connect!(model, C, kvl)
model
```

```
BondGraph RC Circuit (3 Nodes, 2 Bonds)
```

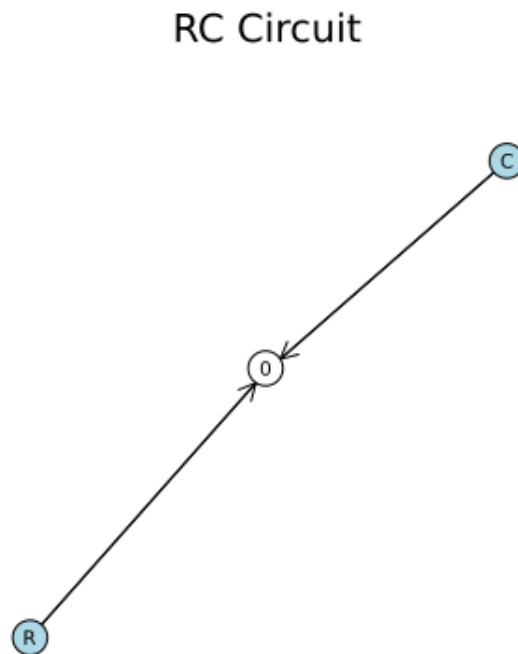
Because our bond graph is fundamentally a graph, we can use existing graph methods on our model.

```
using Graphs
incidence_matrix(model)
```

```
3×2 SparseArrays.SparseMatrixCSC{Int64, Int64} with 4 stored entries:
  ⋅  -1
 -1  ⋅
  1   1
```

We can also visualise our model structure by plotting it as a graph network using Plots.jl.

```
using Plots
plot(model)
```



2.2 Simulating our model

With a bond graph we can automatically generate a series of differential equations which combine all the constitutive relations from the components, with efforts and flows shared according to the graph structure.

```
constitutive_relations(model)
```

$$\frac{dC_+q(t)}{dt} = \frac{-C_+q(t)}{C_+CR_+R} \quad (2.1)$$

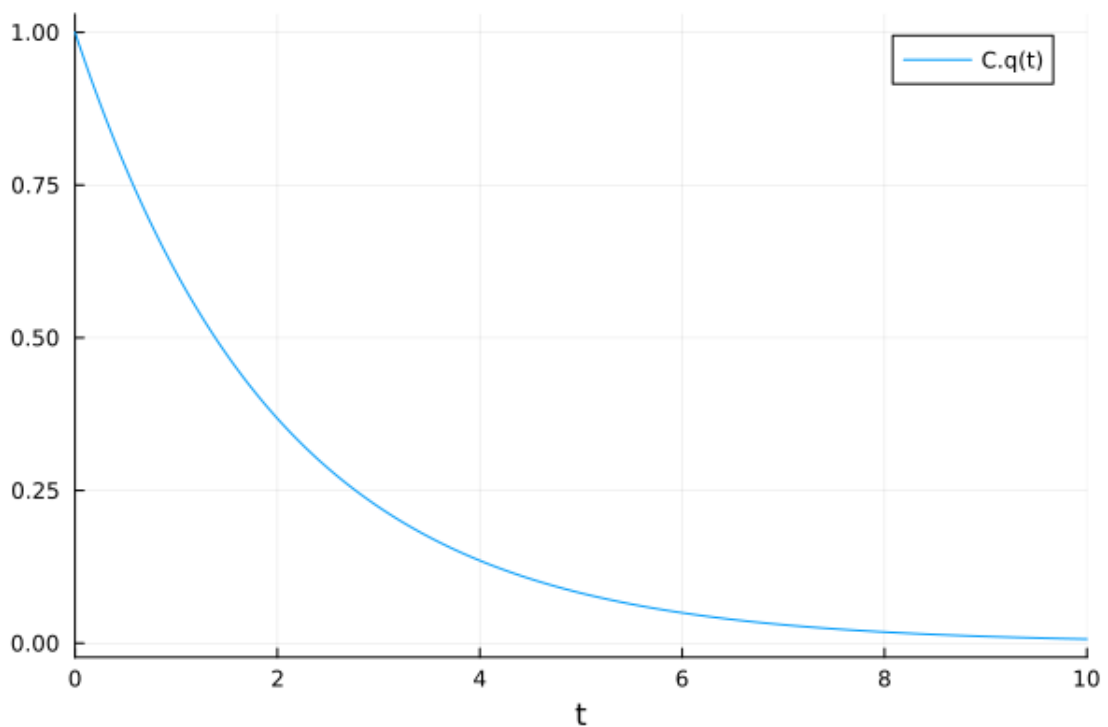
We will set values for the component parameters in the model. Each component comes with default values. When substituted into our equations, we get the following relation for the capacitor charge $C.q(t)$.

```
C.C = 1  
R.R = 2  
constitutive_relations(model; sub_defaults=true)
```

$$\frac{dC.q(t)}{dt} = \frac{-C.q(t)}{2} \quad (2.2)$$

We can solve this bond graph directly using the in-built `simulate` function.

```
tspan = (0., 10.)  
u0 = [1] # initial value for C.q(t)  
sol = simulate(model, tspan; u0)  
plot(sol)
```



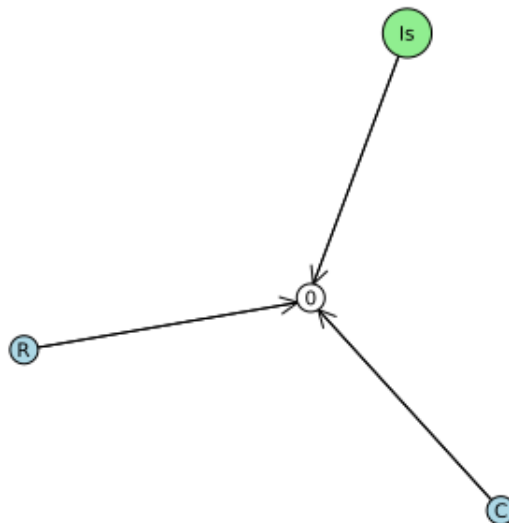
Under the hood, our `simulate` function is converting our bond graph into an `ODESystem`. We can choose instead to create an `ODESystem` directly and handle it with whatever functions we like.

2.3 Adding control variables

We will expand our model by adding an external current (flow) supply in parallel, represented by the component Sf (Source of Flow)

```
Is = Component(:Sf, "Is")
add_node!(model, Is)
connect!(model, Is, kvl)
plot(model)
```

RC Circuit

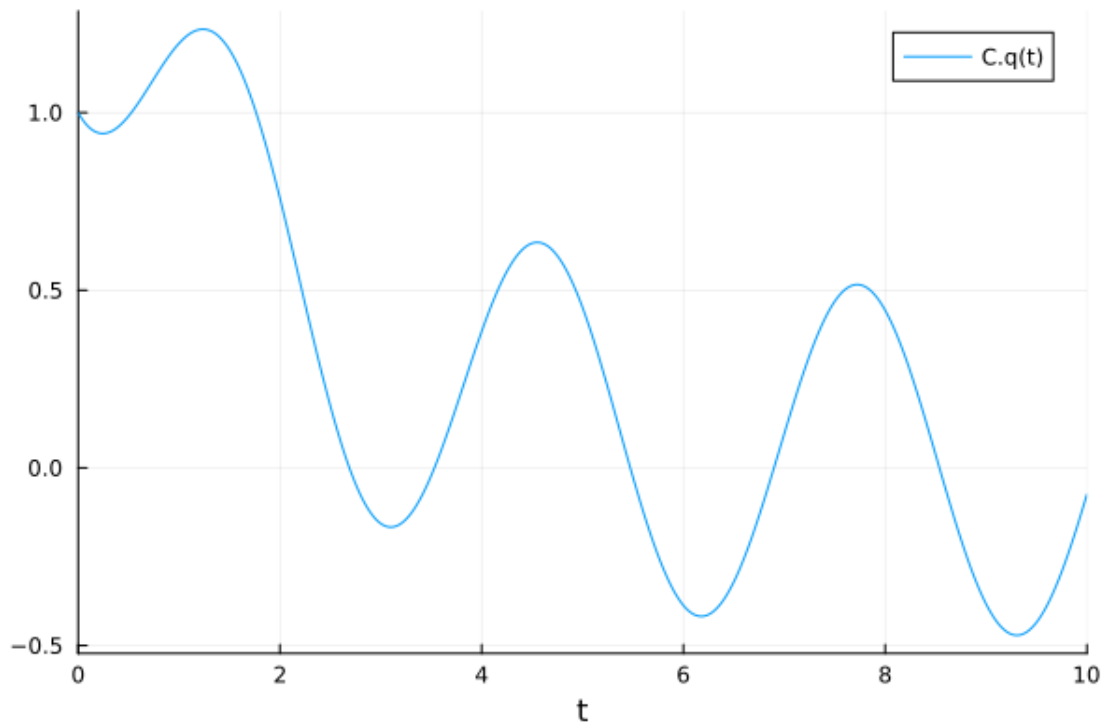


We will add a the forcing function $fs(t) = \sin(2t)$ as an external current input.

```
Is.fs = t -> sin(2t)
constitutive_relations(model; sub_defaults=true)
```

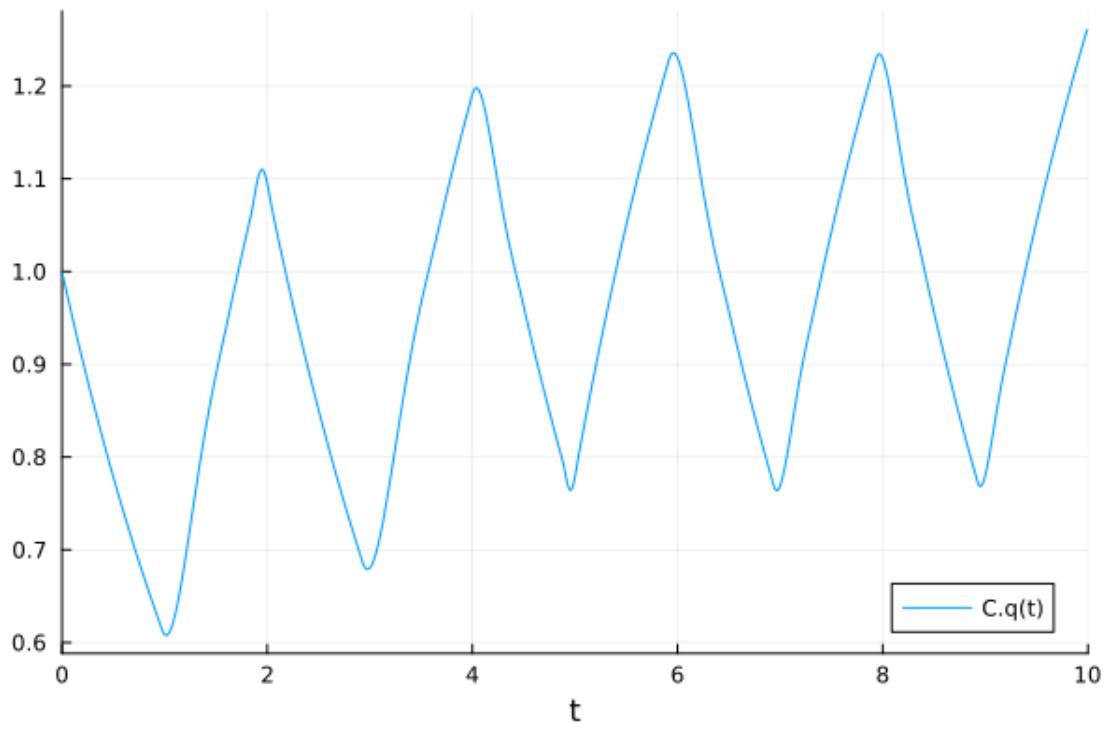
$$\frac{dC_{+q}(t)}{dt} = \frac{-C_{+q}(t) + 2 \sin(2t)}{2} \quad (2.3)$$

```
sol = simulate(model, tspan; u0)
plot(sol)
```



The input can be any arbitrary julia function of t , so long as it returns a sensible output. Note that for this to work you must register the custom function with `@register_symbolic`, so that the library knows not to simplify this function further.

```
using ModelingToolkit
@register_symbolic f(t)
Is.fs = t -> f(t)
f(t) = t % 2 <= 1 ? 0 : 1 # repeating square wave
sol = simulate(model, tspan; u0)
plot(sol)
```



Chapter 3

Examples

Interactive Jupyter Notebook versions of these tutorials can be found on [GitHub](#).

```
using BondGraphs
using Catalyst
using Plots
```

3.1 Simple Electric Circuit

This is a reduced copy-and-paste version of the electric circuit tutorial in the [Getting Started](#) section.

```
model = BondGraph("RC Circuit")
C = Component(:C)
R = Component(:R)
Is = Component(:Sf, "Is")
kvl = EqualEffort()
add_node!(model, [C, R, Is, kvl])
connect!(model, R, kvl)
connect!(model, C, kvl)
```

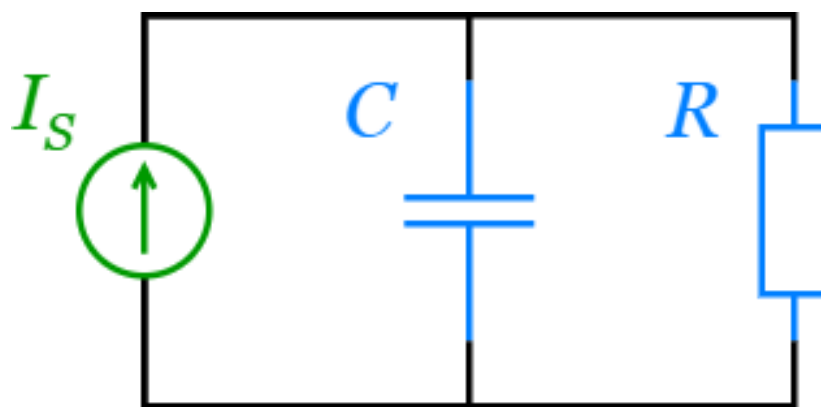
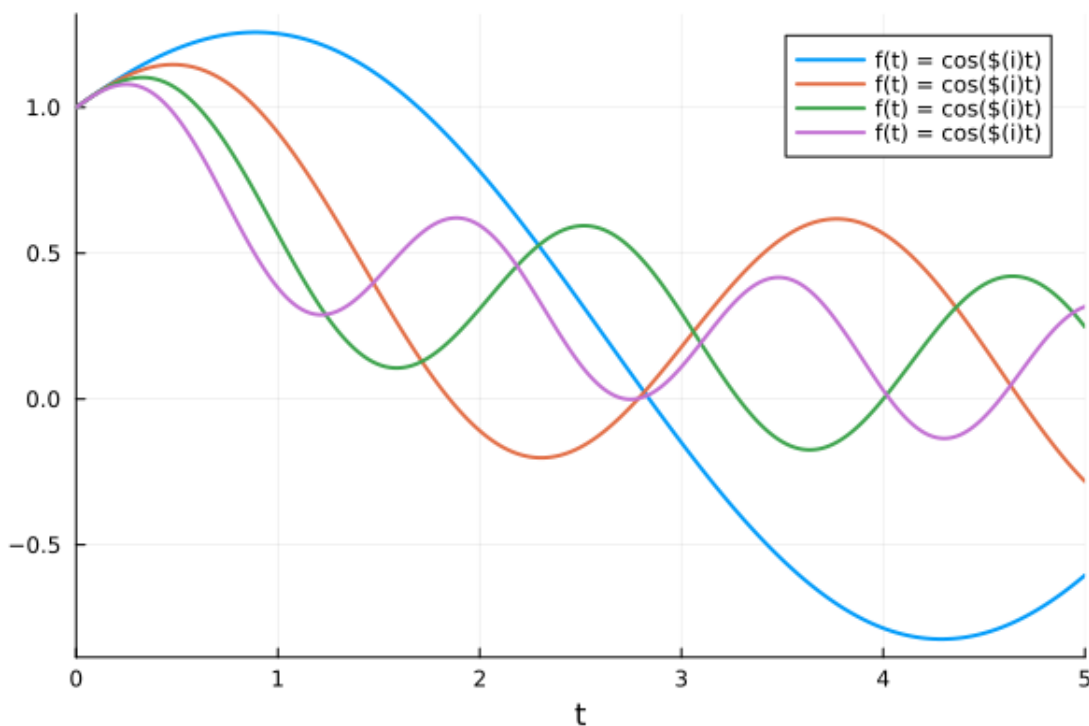


Figure 3.1:

```

connect!(model, Is, kv1)
C.C = 1
R.R = 2
u0 = [1]
p = plot()
for i in 1:4
    Is.fs = t -> cos(i * t)
    sol = simulate(model, (0., 5.); u0)
    plot!(p, sol, label = "f(t) = cos(\$(i)t)", lw=2)
end
plot(p)

```

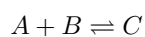


3.2 Biochemical Reaction Networks

BondGraphs.jl includes a special type conversion from a `Catalyst.ReactionSystem` to a `BondGraph`. This means we can easily create chemical reaction networks with the `@reaction_network` macro and automatically generate a biochemical bond graph.

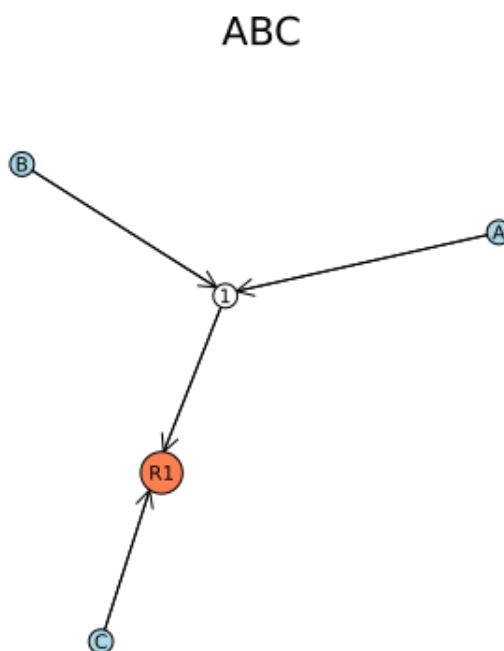
3.2.1 A simple biochemical example

We will start with a very simple chemical reaction.



Due to the design of bond graphs, all reactions must be reversible.

```
abc = @reaction_network ABC begin
    1, A + B --> C
end
bg_abc = BondGraph(abc)
plot(bg_abc)
```



Note

The 1 before the reaction definition is the reaction rate. This is currently not used in the conversion, so the value is only a placeholder. The reaction rate can be set later in the bond graph.

This model derives equations for mass-action kinetics. (Other reaction equations can be used with a custom reaction component.)

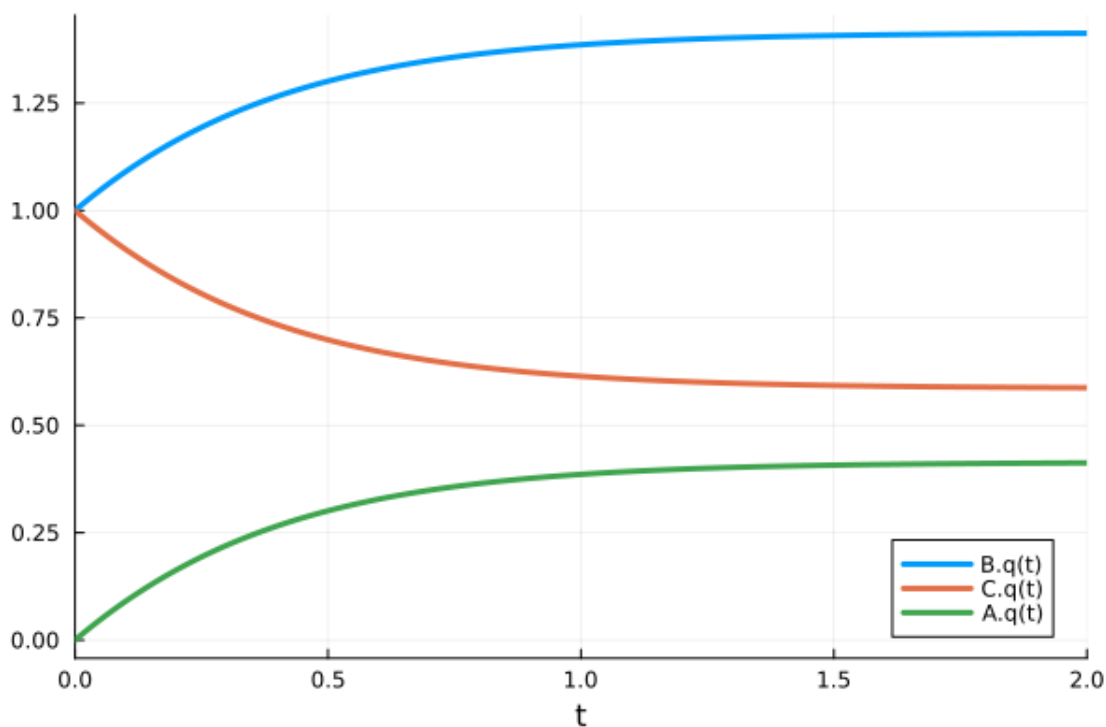
```
constitutive_relations(bg_abc; sub_defaults=true)
```

$$\frac{dB_{+q}(t)}{dt} = -A_{+q}(t) B_{+q}(t) + C_{+q}(t) \quad (3.1)$$

$$\frac{dC_{+q}(t)}{dt} = -(-A_{+q}(t) B_{+q}(t) + C_{+q}(t)) \quad (3.2)$$

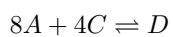
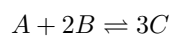
$$\frac{dA_{+q}(t)}{dt} = -A_{+q}(t) B_{+q}(t) + C_{+q}(t) \quad (3.3)$$

```
tspan = (0., 2.)
u0 = [1, 1, 0]
sol = simulate(bg_abc, tspans; u0)
plot(sol, lw=3)
```



3.2.2 Stoichiometry

This process works with multiple reactions with different stoichiometries.

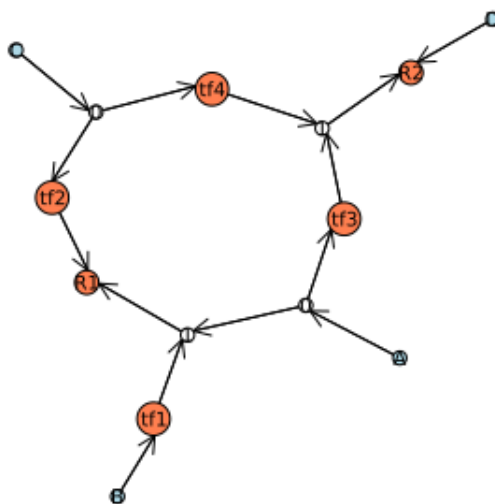


```

abcd = @reaction_network ABCD begin
    1, A + 2B --> 3C
    1, 8A + 4C --> D
end
bg_abcd = BondGraph(abcd)
plot(bg_abcd)

```

ABCD



```
constitutive_relations(bg_abcd; sub_defaults=true)
```

$$\frac{dB_{+q}(t)}{dt} = 2 \left(- (B_{+q}(t))^2 A_{+q}(t) + e^{3 \log(C_{+q}(t))} \right) \quad (3.4)$$

$$\frac{dA_{+q}(t)}{dt} = 8.0 D_{+q}(t) - 8 (C_{+q}(t))^4 (A_{+q}(t))^8 - (B_{+q}(t))^2 A_{+q}(t) + e^{3 \log(C_{+q}(t))} \quad (3.5)$$

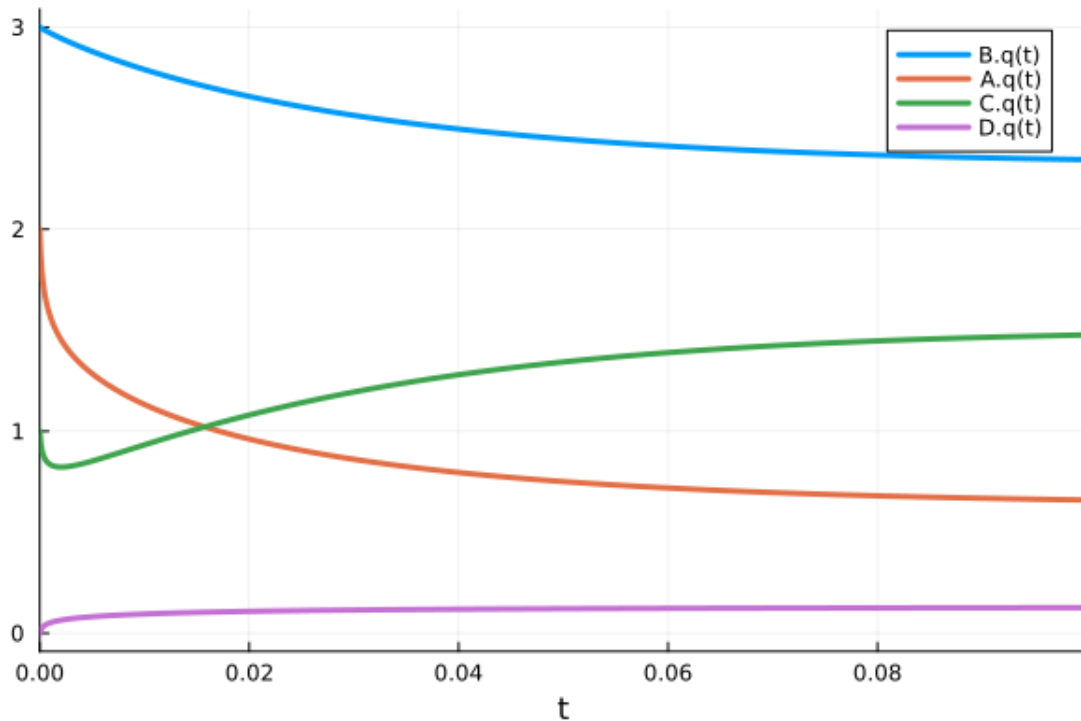
$$\frac{dC_{+q}(t)}{dt} = - 3.0 (C_{+q}(t))^3 + 4.0 D_{+q}(t) - 4 (C_{+q}(t))^4 (A_{+q}(t))^8 + 3.0 (B_{+q}(t))^2 A_{+q}(t) \quad (3.6)$$

$$\frac{dD_{+q}(t)}{dt} = - \left(- (C_{+q}(t))^4 (A_{+q}(t))^8 + D_{+q}(t) \right) \quad (3.7)$$

```

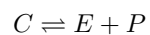
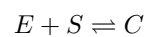
tspan = (0.0, 0.1)
u0 = [3, 2, 1, 0]
sol = simulate(bg_abcd, tspan; u0)
plot(sol, lw=3)

```



3.2.3 The reversible Michaelis-Menten model

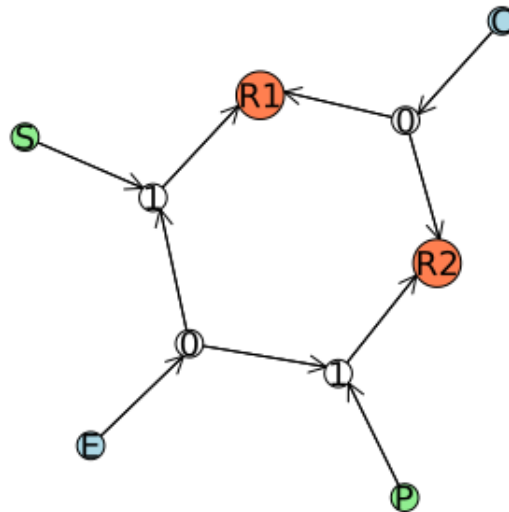
A realistic example with chemostats. Chemostats are chemical species that are held constant through the reaction by adding or removing the species to maintain the set concentration.



Here, the substrate S and P are chemostats. In bond graph terms, this means replacing the Ce components with a S_{Ce} component.

```
rn_mm = @reaction_network MM_reversible begin
    (1, 1), E + S <--> C
    (1, 1), C <--> E + P
end
bg_mm = BondGraph(rn_mm; chemostats=["S", "P"])
plot(bg_mm, fontsize=12)
```

MM_reversible



```
# bg_mm.S.xs = t -> 1 + t # substrate increases over time
tspan = (0., 10.)
# u0 = [1,2]
# sol = simulate(bg_mm, tspan; u0)
# plot(sol, lw=3)
```

```
(0.0, 10.0)
```

3.3 SERCA Pump

In this example we will demonstrate biochemical bond graph construction on a larger system.

We will model the SERCA reaction network as described in Tran et al.¹ and Pan et al.²

```
rn_serca = @reaction_network SERCA begin
  (1, 1), P1 + MgATP <--> P2
```

¹Tran et al., *A Thermodynamic Model of the Cardiac Sarcoplasmic/Endoplasmic Ca²⁺ (SERCA) Pump* (2009)

²Pan et al., *Bond graph modelling of the cardiac action potential: implications for drift and non-unique steady states* (2018)

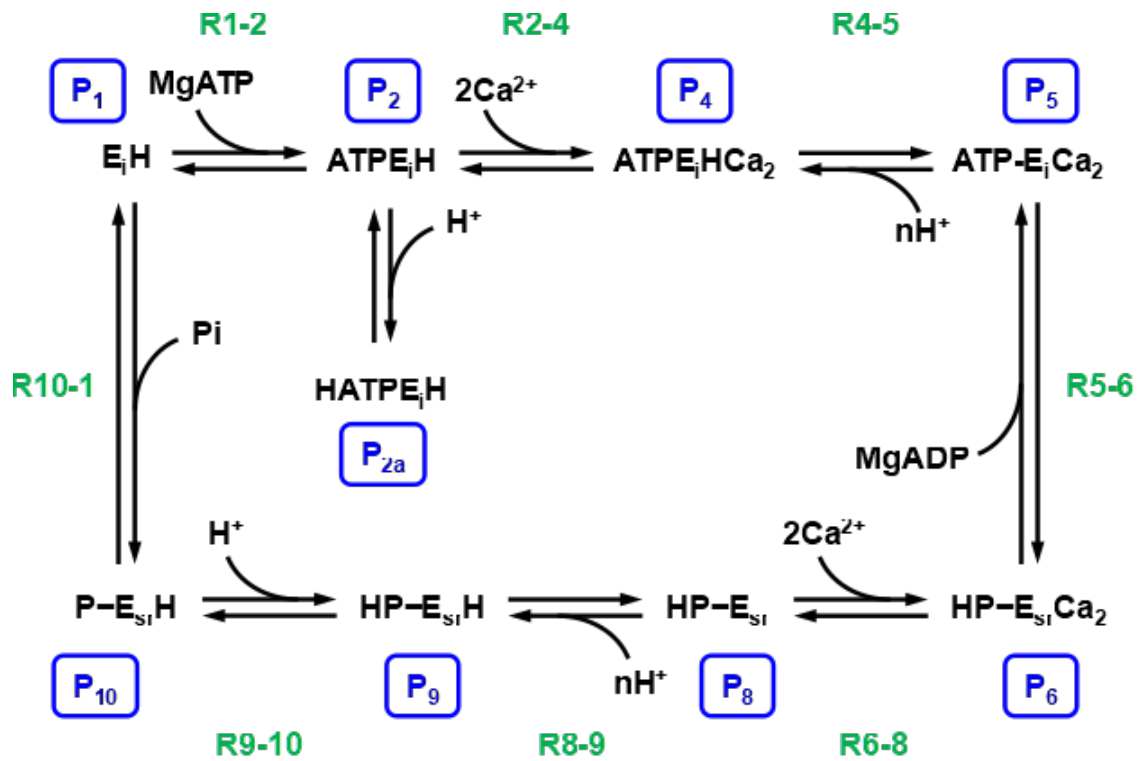


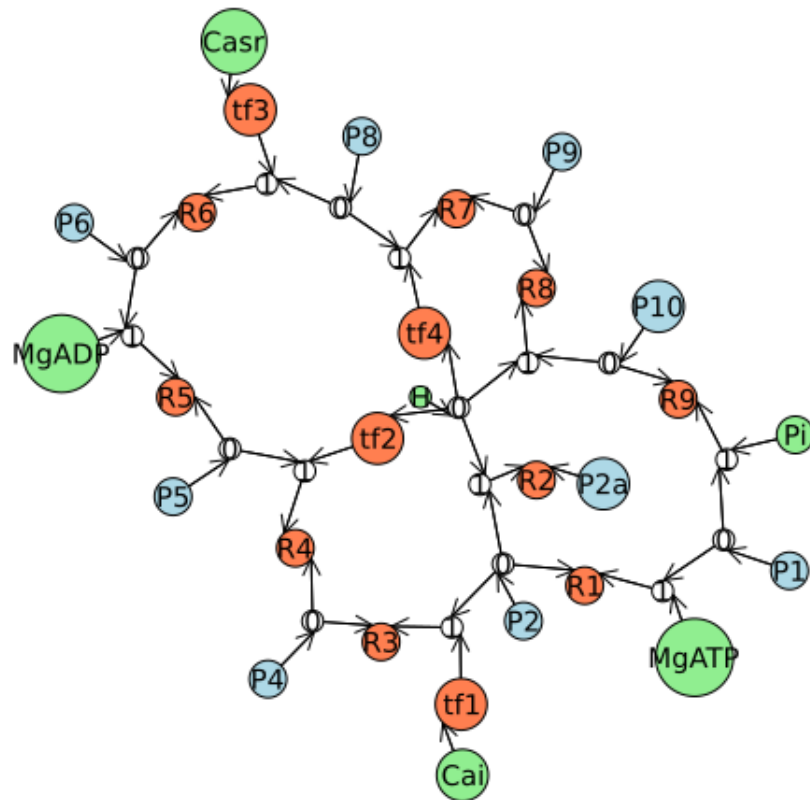
Figure 3.2:

```

(1, 1), P2 + H <--> P2a
(1, 1), P2 + 2Cai <--> P4
(1, 1), P4 <--> P5 + 2H
(1, 1), P5 <--> P6 + MgADP
(1, 1), P6 <--> P8 + 2Casr
(1, 1), P8 + 2H <--> P9
(1, 1), P9 <--> P10 + H
(1, 1), P10 <--> P1 + Pi
end
chemostats = ["MgATP", "MgADP", "Pi", "H", "Cai", "Casr"]
bg_serca = BondGraph(rn_serca; chemostats)
plot(bg_serca, size=(600,600), fontsize=10)

```

SERCA



For this example we need to set the parameter values for the reaction rates r , the species affinities K , the chemostat concentrations x_s , and the initial concentrations for all P_i .

We also set let the calcium concentration increase over time with $[\text{Ca}^{2+}] = 0.05 + 0.01t$

```
reaction_rates = [
  :R1 => 0.00053004,
  :R2 => 8326784.0537,
  :R3 => 1567.7476,
  :R4 => 1567.7476,
  :R5 => 3063.4006,
  :R6 => 130852.3839,
  :R7 => 11612934.8748,
  :R8 => 11612934.8748,
```

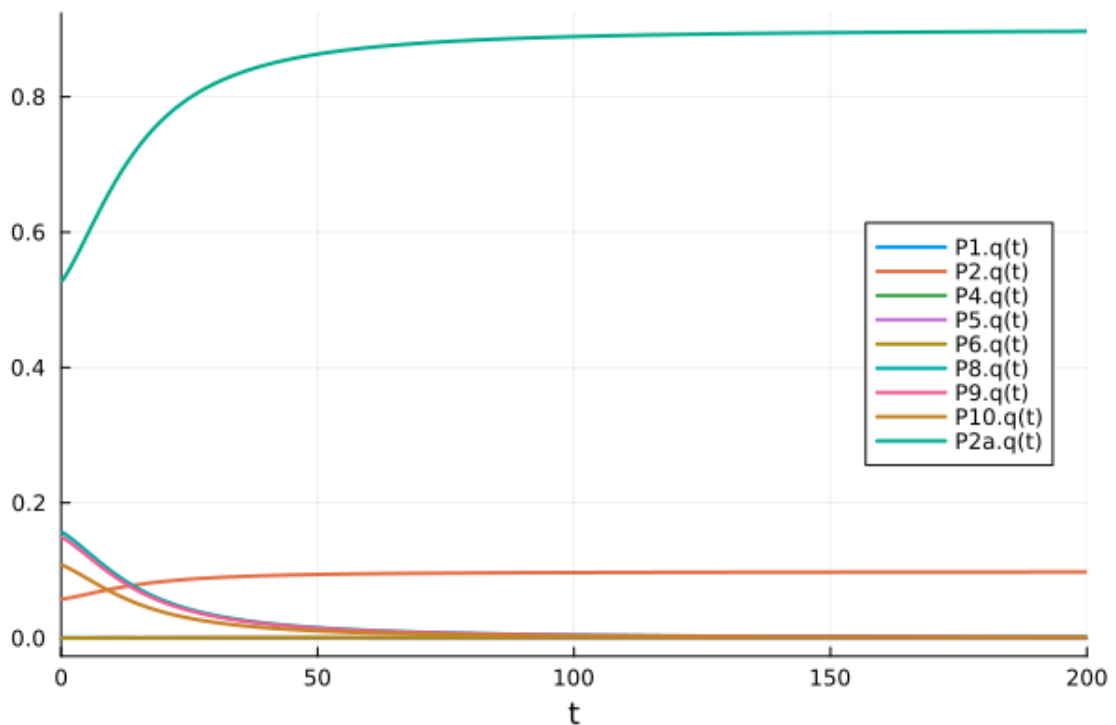
```

    :R9 => 0.049926
  ]
  species_affinities = [
    :P1 => 5263.6085,
    :P2 => 3803.6518,
    :P2a => 3110.4445,
    :P4 => 16520516.1239,
    :P5 => 0.82914,
    :P6 => 993148.433,
    :P8 => 37.7379,
    :P9 => 2230.2717,
    :P10 => 410.6048,
    :Cai => 1.9058,
    :Casr => 31.764,
    :MgATP => 244.3021,
    :MgADP => 5.8126e-7,
    :Pi => 0.014921,
    :H => 1862.5406
  ]
  vol_sr = 2.28
  chemostat_amounts = [
    :Cai => t -> 0.0057,
    :Casr => t -> vol_sr*(0.05 + 0.01t), # Ca2+ increases over time
    :H => t -> 0.004028,
    :MgADP => t -> 1.3794,
    :MgATP => t -> 3.8,
    :Pi => t -> 570
  ]
  initial_conditions = [
    :P1 => 0.000483061870385487,
    :P2 => 0.0574915174273067,
    :P2a => 0.527445119834607,
    :P4 => 1.51818391164022e-09,
    :P5 => 0.000521923287622898,
    :P6 => 7.80721128535043e-05,
    :P8 => 0.156693953834181,
    :P9 => 0.149232225342376,
    :P10 => 0.108044124948978
  ]
  for (reaction, rate) in reaction_rates
    getproperty(bg_serca, reaction).r = rate
  end
  for (species, affinity) in species_affinities
    getproperty(bg_serca, species).K = affinity
  end
  for (chemostat, amount) in chemostat_amounts
    getproperty(bg_serca, chemostat).xs = amount
  end
  for (species, ic) in initial_conditions
    getproperty(bg_serca, species).q = ic
  end
end

import DifferentialEquations: Rosenbrock23 # stiff equation solver
tspan = (0., 200.)

```

```
sol = simulate(bg_serca, tspan; solver=Rosenbrock23());
plot(sol, lw=2, legend=:right)
```



3.4 Electrochemical System: Ion Pore Transport

A multiphysics bond graph example that combines biochemical reactions with electrical (ion) forces. This example is taken from Cudmore et al.³

For this example we are modelling three ion pore channels for Na^+ , Cl^- and K^+ . It is a good idea to define a function that returns an ion pore base model.

```
function ion_pore(name=""; z=1, conc_ex=1//1000, conc_in=1//1000)
    bg = BondGraph(name * " Ion Pore Transport")
    membrane = Component(:C, "mem"; C=1)
    ion_ex = Component(:SCe, "Ie"; K=1, xs=t->conc_ex)
    ion_in = Component(:SCe, "Ii"; K=1, xs=t->conc_in)
    potential_mem = EqualEffort()
    flow_f = EqualFlow()
    flow_r = EqualFlow()
    TF_F = Component(:TF, "F"; n=96485)
    TF_zf = Component(:TF, "-z/2"; n=-z//2) # -ve voltage
    TF_zr = Component(:TF, "z/2"; n=z//2) # +ve voltage
    re_pore = Component(:Re, "pore"; r=1//10_000_000)
```

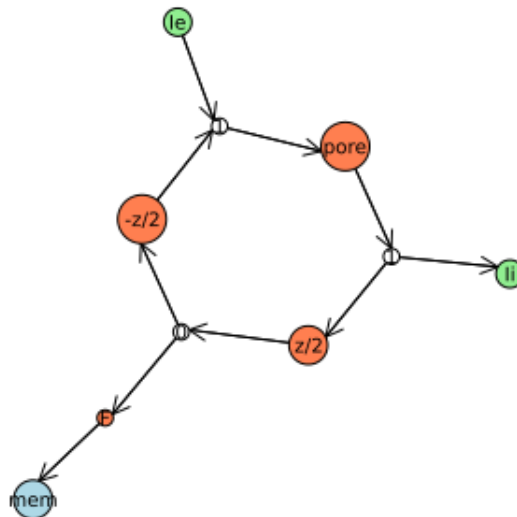
³Cudmore et al., *Analysing and simulating energy-based models in biology using BondGraphTools* (2021)

```

allcomps = [
    membrane, ion_ex, ion_in, potential_mem, flow_f, flow_r,
    TF_F, TF_zr, TF_zf, re_pore
]
add_node!(bg, allcomps)
connect!(bg, ion_ex, flow_f)
connect!(bg, flow_f, re_pore)
connect!(bg, re_pore, flow_r)
connect!(bg, flow_r, ion_in)
connect!(bg, flow_r, (TF_zr,2))
connect!(bg, (TF_zr,1), potential_mem)
connect!(bg, potential_mem, (TF_zf,1))
connect!(bg, (TF_zf,2), flow_f)
connect!(bg, potential_mem, (TF_F,2))
connect!(bg, (TF_F,1), membrane)
return bg
end
ion_pore_bg = ion_pore()
plot(ion_pore_bg)

```

Ion Pore Transport



```

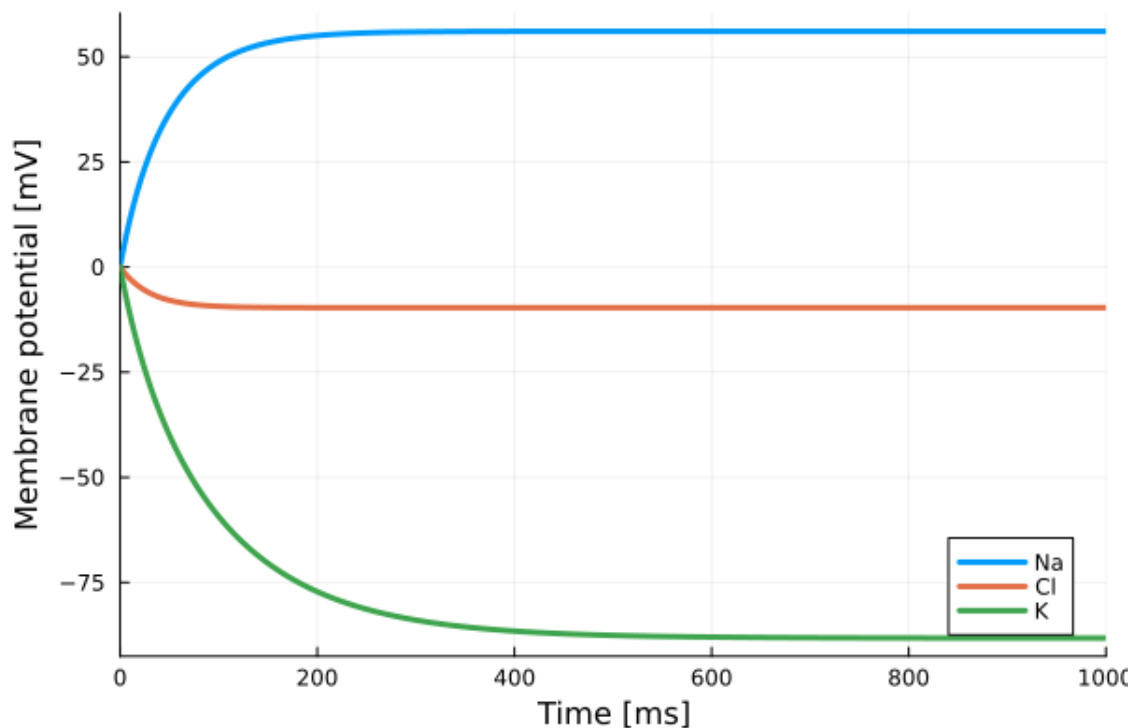
ions = [
    "Na" => (z=1, conc_ex=155//1000, conc_in=19//1000),
    "Cl" => (z=-1, conc_ex=112//1000, conc_in=78//1000),
    "K"  => (z=1, conc_ex=5//1000, conc_in=136//1000),
]
plot(; legend=:bottomright)

```

```

for (ionname, params) in ions
    model = ion_pore(ionname; params...)
    sol = simulate(model, (0., 1000.))
    plot!(sol, label=ionname, lw=3)
end
plot!(); xlabel="Time [ms]", ylabel="Membrane potential [mV]",
    ↪ yformatter=y->y*1000)

```



3.5 Enzyme Catalysed Reactions

We can create custom reaction components which describe enzyme-catalyzed reaction mechanics, such as the Michaelis-Menten rate law.

Warning

This method of adding custom components will be deprecated in the near future and replaced with a new method.

New components are constructed using the same dictionary structure as in the default component library. See [addlibrary!](#) for more details.

```

using ModelingToolkit
@parameters t R T r1 r2 k_c e_T
@variables E(t)[1:2] F(t)[1:2] # effort and flow variables

```

```

R, T = GlobalScope(R), GlobalScope(T) # no namespace
D = Differential(t)
# Custom Michaelis-Menten reaction component
ReMM = Dict(
  :description => """
  Michaelis-Menten reaction
  r1: Rate of reaction 1
  r2: Rate of reaction 2
  k_c: Affinity of complex relative to free enzyme
  e_T: Total amount of enzyme
  R: Universal Gas Constant
  T: Temperature
  """,
  :numports => 2,
  :variables => Dict(
    :parameters => Dict(
      r1 => 1,
      r2 => 1,
      k_c => 1,
      e_T => 1
    ),
    :globals => Dict(
      R => 8.314,
      T => 310.0
    )
  ),
  :equations => [
    0 ~ F[1] + F[2],
    0 ~ F[1] - e_T*r1*r2*k_c*(exp(E[1]/R/T) - exp(E[2]/R/T)) /
      (r1*exp(E[1]/R/T) + r2*exp(E[2]/R/T) + k_c*(r1+r2))
  ]
)
# add to component library
addlibrary!(Dict(:ReMM => ReMM))
ReMM[:equations]

```

$$0 = F(t)_1 + F(t)_2 \quad (3.8)$$

$$0 = \frac{-e_T k_c r_1 r_2 \left(-e^{\frac{E(t)_2}{RT}} + e^{\frac{E(t)_1}{RT}} \right)}{k_c (r_1 + r_2) + r_1 e^{\frac{E(t)_1}{RT}} + r_2 e^{\frac{E(t)_2}{RT}}} + F(t)_1 \quad (3.9)$$

We will create a bond graph with these new equations. \$ A \rightarrow B \rightarrow C \$

```

rn_enzyme = @reaction_network EnzymeNetwork begin
  (1, 1), A <--> B
  (1, 1), B <--> C
end
bg = BondGraph(rn_enzyme)
bg.A.K, bg.B.K, bg.C.K = 1, 1, 1
# Swap mass-action components for new michaelis-menten components

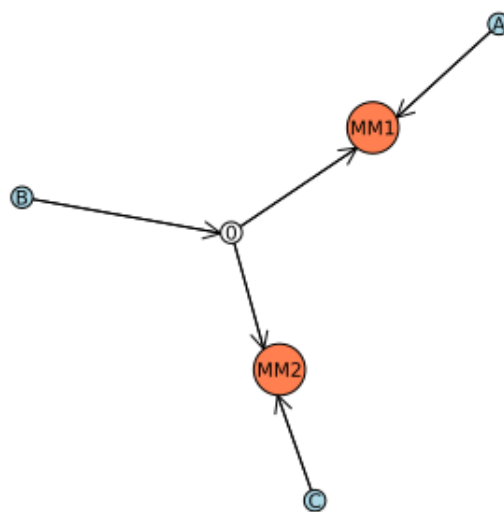
```

```

MM1 = Component(:ReMM, "MM1"; r1=100, r2=100)
MM2 = Component(:ReMM, "MM2"; r1=100, r2=100)
swap!(bg, bg.R1, MM1)
swap!(bg, bg.R2, MM2)
plot(bg)

```

EnzymeNetwork



```

constitutive_relations(bg; sub_defaults=true)

```

$$\frac{dB_{+q}(t)}{dt} = \frac{-2000000(B_{+q}(t))^2 + 2000000A_{+q}(t) - 4000000B_{+q}(t) + 2000000C_{+q}(t) + 2000000A_{+q}(t)C_{+q}(t)}{(200 + 100A_{+q}(t) + 100B_{+q}(t))(200 + 100B_{+q}(t) + 100C_{+q}(t))} \quad (3.10)$$

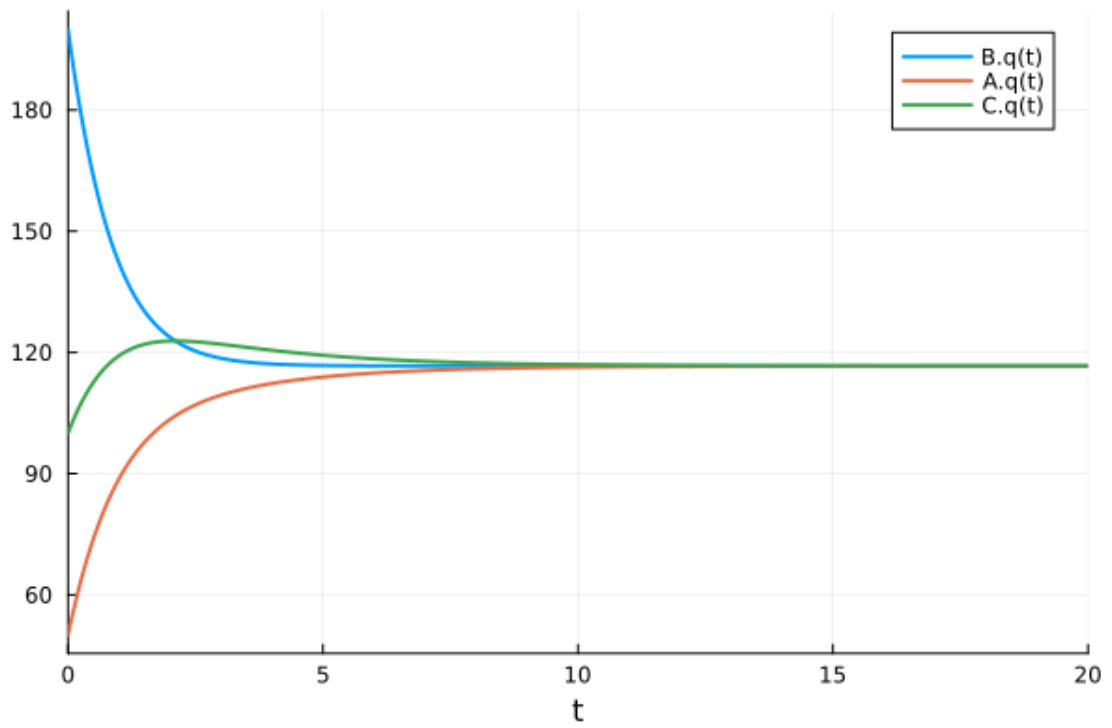
$$\frac{dA_{+q}(t)}{dt} = \frac{-100A_{+q}(t) + 100B_{+q}(t)}{2 + A_{+q}(t) + B_{+q}(t)} \quad (3.11)$$

$$\frac{dC_{+q}(t)}{dt} = \frac{100B_{+q}(t) - 100C_{+q}(t)}{2 + B_{+q}(t) + C_{+q}(t)} \quad (3.12)$$

```

sol = simulate(bg, (0., 20.); u0=[200,50,100])
plot(sol, lw=2)

```



Chapter 4

API Reference

4.1 Reference

- [BondGraphs.Bond](#)
- [BondGraphs.BondGraph](#)
- [BondGraphs.BondGraph](#)
- [BondGraphs.BondGraphNode](#)
- [BondGraphs.Component](#)
- [BondGraphs.EqualEffort](#)
- [BondGraphs.EqualFlow](#)
- [BondGraphs.SourceSensor](#)
- [BondGraphs.add_node!](#)
- [BondGraphs.addlibrary!](#)
- [BondGraphs.connect!](#)
- [BondGraphs.constitutive_relations](#)
- [BondGraphs.constitutive_relations](#)
- [BondGraphs.description](#)
- [BondGraphs.disconnect!](#)
- [BondGraphs.getbonds](#)
- [BondGraphs.getnodes](#)
- [BondGraphs.insert_node!](#)
- [BondGraphs.merge_nodes!](#)
- [BondGraphs.remove_node!](#)
- [BondGraphs.simplify_junctions!](#)
- [BondGraphs.simulate](#)
- [BondGraphs.swap!](#)

4.1.1.1 Types

BondGraphs.Bond – Type.

```
Bond(source::AbstractNode, destination::AbstractNode)
Bond((source_node, port_label), (destination_node, port_label))
```

Connect two bond graph components (or two ports of two components) with a bond. The bond direction is from source to destination. If the ports are not specified, the bond will be created between the next available ports in each component.

In most cases it is better to use `connect!` instead.

[source](#)

BondGraphs.BondGraph – Type.

```
BondGraph(name="BG") <: Graphs.AbstractGraph{Int64}
BondGraph(name, nodes::Vector{AbstractNode}, bonds::Vector{Bond})
```

The bond graph object which contains a vector of nodes and bonds. All operations on components or bonds must happen within the same bond graph. This inherits the methods of the AbstractGraph type and so will work with

See also [BondGraphNode](#).

[source](#)

BondGraphs.BondGraph – Method.

```
BondGraph(rs::ReactionSystem; chemostats=[])
```

Convert a Catalyst.ReactionSystem into a BondGraph.

chemostats are chemical species with fixed concentrations. In bond graph terms, these are "SCe" types (chemical energy sources) instead of "Ce" types (chemical energy store).

[source](#)

BondGraphs.BondGraphNode – Type.

```
BondGraphNode(bg::BondGraph, name=name(bg); deep_copy=false)
```

Convert a BondGraph into a component that can be added in another level bond graph. Components can be exposed to the outer bond graph by replacing them with a [SourceSensor](#) type using the `swap!` function.

See also [BondGraph](#).

[source](#)

BondGraphs.Component – Type.

```
Component{N} <: AbstractNode
Component(type, name=type)
Component(type, name=type; library=BondGraphs.DEFAULT_LIBRARY, <keyword
↪ arguments>)
```

Construct a Component of a defined (bondgraph) type $\in \{R, C, I, Se, Sf, TF, Ce, Re, SCe\}$.

Components have a N fixed ports when generated. This is usually determined by the bond graph type. Other properties and equations of available components are defined in BondGraphs.DEFAULT_LIBRARY (see [description](#)).

[source](#)

BondGraphs.EqualEffort – Type.

```
EqualEffort <: Junction
```

Efforts are all equal, flows sum to zero (0-junction). Has an unlimited number of ports.

[source](#)

BondGraphs.EqualFlow – Type.

```
EqualFlow <: Junction
```

Flows are all equal, efforts sum to zero (1-junction). Has an unlimited number of ports.

[source](#)

BondGraphs.SourceSensor – Type.

```
SourceSensor <: AbstractNode
```

Special component type that acts as a source of both effort and flow. SourceSensors are used as external ports for [BondGraphNode](#)s.

[source](#)

4.1.2 Methods

BondGraphs.add_node! – Method.

```
add_node!(bg: BondGraph, nodes)
```

Add a node to a bond graph `bg`. Can add a single node or list of nodes.

[source](#)

`BondGraphs.addlibrary!` – Method.

```
addlibrary!(newlib)
```

Combine the library `newlib` with the default library used within `BondGraphs`. `newlib` will need to be in the form of a dictionary, and new components should follow the below schema.

NOTE: This library is likely to change in the future. Do not rely on this schema too much.

Library Schema

- `description` -> written description of the component and definitions
- `numports` -> the number of ports in the component
- `variables` ->
 - `parameters` -> constant parameters, unique for each component instance
 - `globals` -> global parameters (i.e. no namespace)
 - `states` -> time-dependent state variables
 - `controls` -> time-dependent parameters that can accept julia functions
- `equations` -> symbolic description of the constitutive equations

[source](#)

`BondGraphs.connect!` – Method.

```
connect!(bg::BondGraph, source_node, destination_node)
connect!(bg::BondGraph, (source_node, port_label), (destination_node,
↪ port_label))
```

Connect two components together in the same bond graph. The bond direction is always from `source_node` to `destination_node`. The port index of `source_node` and `destination_node` can be optionally set.

[source](#)

`BondGraphs.constitutive_relations` – Method.

```
constitutive_relations(n::AbstractNode)
```

Return the constitutive relations (equations) for node `n`.

If `n` is a Junction, the flow and effort constraints are generated from its connections.

[source](#)

`BondGraphs.constitutive_relations` – Method.

```
constitutive_relations(bg::BondGraph; sub_defaults=false)
```

Generate the constitutive relations (equations) for bond graph bg.

The equations are symbolically derived from the equations of all the nodes and bonds in the bond graph. If sub_defaults is true, the default parameter values for each component are subbed into the equations.

NOTE: This creates a ModelingToolkit.ODESystem of the bond graph and returns only the equations. If you want to numerically solve the bond graph equations, either use `simulate` or create an ODESystem directly.

[source](#)

BondGraphs.description – Method.

```
description(comp::Symbol)
description(lib, comp::Symbol)
```

Print the description of the component with symbol comp in library lib.

[source](#)

BondGraphs.disconnect! – Method.

```
disconnect!(bg::BondGraph, node1, node2)
```

Remove the bond connecting node1 and node2. The order of nodes does not matter.

[source](#)

BondGraphs.getbonds – Method.

```
getbonds(bg::BondGraph, n1::AbstractNode, n2::AbstractNode)
getbonds(bg::BondGraph, (n1, n2))
```

Return the bond in bg connecting nodes n1 and n2, if it exists.

[source](#)

BondGraphs.getnodes – Method.

```
getnodes(bg::BondGraph, type)
```

Return all nodes a particular bond graph type in the bond graph bg.

type can be a DataType (e.g. Component{1}), a string (e.g. "C"), or a vector of strings.

[source](#)

BondGraphs.insert_node! – Method.

```
insert_node!(bg::BondGraph, bond, newnode)
insert_node!(bg::BondGraph, (node1, node2), newnode)
```

Inserts newnode between two existing connected nodes. The direction of the original bond is preserved.

Supply either the two nodes as a tuple, or the bond that connects them in bg.

[source](#)

BondGraphs.merge_nodes! – Method.

```
merge_nodes!(bg::BondGraph, node1, node2; junction=EqualEffort())
```

Combine two copies of the same component in bg by adding a junction and connecting the neighbours of node1 and node2 to the new junction.

Merging nodes this way means there is only one component representing a system element, and all other nodes connect to the component via the new junction.

[source](#)

BondGraphs.remove_node! – Method.

```
remove_node!(bg::BondGraph, nodes)
```

Remove a node to a bond graph bg. Can remove a single node or list of nodes.

[source](#)

BondGraphs.simplify_junctions! – Method.

```
simplify_junctions!(bg::BondGraph; remove_redundant=true,
↪ squash_identical=true)
```

Remove unnecessary or redundant junctions from bond graph bg.

If remove_redundant is true, junctions that have zero or one neighbours are removed, and junctions with two neighbours are squashed (connected components remain connected).

If squash_identical is true, connected junctions of the same type are squashed into a single junction.

[source](#)

BondGraphs.simulate – Method.

```
simulate(bg::BondGraph, tspan; u0=[], pmap=[], solver=Tsit5(),  
↪ flag_ODE=true, kwargs...)
```

Simulate the bond graph model.

The keyword arguments are the same as for ODEProblem and solve in DifferentialEquations.

[source](#)

BondGraphs.swap! – Method.

```
swap!(bg::BondGraph, oldnode, newnode)
```

Remove oldnode from bond graph bg and replace it with newnode. The new node will have the same connections (bonds) as the original model.

newnode must have a greater or equal number of ports as oldnode.

[source](#)

Appendix D

Extra Derivations

D.1 Langevin equation analytical solution

This derivation has been adapted from Jacobs [122]. We will solve the stochastic Langevin equation from [subsection 5.3.1](#), which has the following definition:

$$dx = v dt \quad (\text{D.1a})$$

$$m dv = -\lambda v dt + \sigma dW \quad (\text{D.1b})$$

where x, v are the position and velocity of the Brownian particle with mass m , λ is the drag coefficient, and σ is the noise magnitude.

The general solution for $v(t)$ is

$$v(t) = v_0 e^{-\lambda t/m} + \frac{\sigma}{m} \int_0^t e^{-\lambda(t-s)/m} dW_s \quad (\text{D.2})$$

and for $x(t)$ is

$$x(t) = \frac{1}{\lambda} \left(1 - e^{-\lambda t/m}\right) v_0 + \frac{\sigma}{\lambda} \int_0^t \left(1 - e^{-\lambda(t-s)/m}\right) dW_s \quad (\text{D.3})$$

where v_0 is the initial velocity of the particle and s is an integration variable.

Recall that $\langle dW \rangle = 0$ and $(dW)^2 = dt$ (Itô's lemma). From [Equation D.2](#), the mean and variance of $v(t)$ are

$$\mathbb{E}[v(t)] = \langle v(t) \rangle = v_0 e^{-\lambda t/m} \quad (\text{D.4a})$$

$$\text{Var}[v(t)] = \langle v(t)^2 \rangle - \langle v(t) \rangle^2 = \frac{\sigma^2}{2m\lambda} (1 - e^{-2\lambda t/m}) \quad (\text{D.4b})$$

The mean and variance of $x(t)$ can be found in with a similar approach.

$$\mathbb{E}[x(t)] = \langle x(t) \rangle = \frac{1}{\lambda}(1 - e^{-\lambda t/m})v_0 \quad (\text{D.5a})$$

$$\text{Var}[x(t)] = \langle x(t)^2 \rangle - \langle x(t) \rangle^2 \approx \frac{\sigma^2 t}{\lambda^2} \quad (\text{D.5b})$$

We have assumed here that $x(0) = 0$. We have also used the assumption that $\lambda t \gg 1$, which is valid for the biological context we are studying (see [subsection 5.3.1](#)).

We are usually interested in the long-term steady state behaviour, i.e. $t \rightarrow \infty$. This is the equilibrium state of the system (if it exists). Let v_{ss} be the steady state velocity. Then

$$\langle v_{ss} \rangle = v_0 e^{-\lambda t/m} \Big|_{t \rightarrow \infty} = 0 \quad (\text{D.6a})$$

$$\text{Var}[v_{ss}] = \langle v_{ss}^2 \rangle - \underbrace{\langle v_{ss} \rangle^2}_0 = \frac{\sigma^2}{2m\lambda} \quad (\text{D.6b})$$

For the steady state $x_{ss} := x(t)_{t \rightarrow \infty}$

$$\langle x_{ss} \rangle = \frac{1}{\lambda} v_0 \quad (\text{D.7})$$

$\text{Var}[x_{ss}]$ is not defined because $\text{Var}[x(t)]$ depends on t and so does not reach a steady state.

We can only derive exact solutions for $v(t)$ and $x(t)$ for a particular realisation or run of the model. If we wanted a general understanding of all possible outcomes, a good option is to determine the means and variances which are capable of generating the state variable distributions exactly in this case.

To simplify the calculations from here we will assume the particle is initially at rest so that $v_0 = 0$ and $x(0) = 0$. We have the following moment equations at equilibrium:

$$\langle x \rangle = 0 \quad \text{Var}[x] \equiv \langle x^2 \rangle = \frac{\sigma^2 t}{\lambda^2} \quad (\text{D.8a})$$

$$\langle v \rangle = 0 \quad \text{Var}[v] \equiv \langle v^2 \rangle = \frac{\sigma^2}{2m\lambda} \quad (\text{D.8b})$$

Since $\langle x \rangle = 0$ and $\langle v \rangle = 0$ the second moments are equivalent to the variances. x and v are random variables, so we can write them in terms of (in this case) a normal distribution.

$$x \sim \mathcal{N}\left(0, \frac{\sigma^2 t}{\lambda^2}\right) \quad v \sim \mathcal{N}\left(0, \frac{\sigma^2}{2m\lambda}\right) \quad (\text{D.9})$$

σ can be determined precisely. The average kinetic energy of the particle $\langle E_k \rangle = \frac{1}{2}m\langle v^2 \rangle$ can be shown from statistical mechanics to be $nk_B T/2$ where n is the degrees of freedom. (3 spatial degrees of freedom would be $3k_B T/2$, for example.)[[67](#), [122](#)].

$\langle v \rangle = 0$, so we have $\text{Var}[v] = \langle v^2 \rangle$. Equating this to the kinetic energy of our particle, we get

$$\frac{k_B T}{2} = \frac{1}{2} m \langle v^2 \rangle = \frac{1}{2} m \frac{\sigma^2}{2m\lambda} = \frac{\sigma^2}{4\lambda} \quad (\text{D.10a})$$

$$\implies \sigma = \sqrt{2k_B T \lambda} \quad (\text{D.10b})$$

For completeness, the means and variances are now:

$$x \sim \mathcal{N}\left(0, \frac{2k_B T t}{\lambda}\right) \quad v \sim \mathcal{N}\left(0, \frac{k_B T}{m}\right) \quad (\text{D.11a})$$

D.2 Non-equilibrium entropy production

We wish to determine entropy flow from the system $\dot{S}_{\text{sys}}$ starting from a non-equilibrium initial condition. We know already that $\langle v \rangle = 0$ is an equilibrium point, so for a non-equilibrium solution, let $v_0 \neq 0$ be the new initial condition. From the model we expect that as $t \rightarrow \infty$ we will reach the same equilibrium solution.

From [Equation D.2](#) the full time-varying ensemble solution for $v(t)$ is

$$v(t) = v_0 e^{-\lambda t/m} + \frac{1}{m} \sqrt{2k_B T \lambda} \int_0^t e^{-\lambda(t-s)/m} dW_s$$

where v_0 is the initial velocity. The solution for $v(t)$ is

$$v(t) \sim \mathcal{N}\left(v_0 e^{-\lambda t/m}, \frac{k_B T}{m} (1 - e^{-2\lambda t/m})\right) \quad (\text{D.12})$$

We get the expected solutions at $t = 0$ and $t \rightarrow \infty$

$$v_0 \sim \mathcal{N}(v_0, 0) \quad v(t \rightarrow \infty) \sim \mathcal{N}\left(0, \frac{k_B T}{m}\right)$$

The non-central second moment is found by the equation $\text{Var}[v] = \langle v^2 \rangle - \langle v \rangle^2$.

$$\langle v^2 \rangle = \text{Var}[v] + \langle v \rangle^2 = \frac{k_B T}{m} (1 - e^{-2\lambda t/m}) + v_0^2 e^{-2\lambda t/m}$$

The entropy production rate is therefore

$$\begin{aligned}
 \dot{S} &= \frac{\lambda}{T} \langle v^2 \rangle \\
 &= \frac{\lambda}{T} \left[\frac{k_B T}{m} (1 - e^{-2\lambda t/m}) + v_0^2 e^{-2\lambda t/m} \right] \\
 &= \frac{k_B \lambda}{m} (1 - e^{-2\lambda t/m}) + \frac{\lambda}{T} v_0^2 e^{-2\lambda t/m} \\
 &= \frac{k_B \lambda}{m} + \left(\frac{\lambda}{T} v_0^2 - \frac{k_B \lambda}{m} \right) e^{-2\lambda t/m}
 \end{aligned}$$

At $t = 0$ we have $\dot{S}(0) = \frac{\lambda}{T} v_0^2$ and $\dot{S}(t \rightarrow \infty) = \frac{k_B \lambda}{m}$, as expected. The change in entropy $\Delta S = S - S(0)$ is

$$\Delta S = \frac{k_B \lambda}{m} t + \left(-\frac{m}{2T} v_0^2 + \frac{k_B}{2} \right) e^{-2\lambda t/m} \quad (\text{D.13})$$

The entropy change from the particle to the heat bath at equilibrium is matched by the same entropy change from the heat bath onto the particle via the noise component, i.e. $S_{\text{eq}} = k_B \lambda t / m$. When this is accounted for, the entropy change out of equilibrium $\Delta S_{\text{neq}} = \Delta S - S_{\text{eq}}$ is

$$\Delta S_{\text{neq}} = \left(-\frac{m}{2T} v_0^2 + \frac{k_B}{2} \right) e^{-2\lambda t/m} \quad (\text{D.14})$$

Also notice that the thermal energy change $T \Delta S$ is

$$T \Delta S = \frac{k_B T \lambda}{m} t + \left(-\frac{1}{2} m v_0^2 + \frac{k_B T}{2} \right) e^{-2\lambda t/m} \quad (\text{D.15})$$

which includes a term for the initial kinetic energy $\frac{1}{2} m v_0^2$ and the average energy per ‘mode’ $k_B T / 2$ (cf. [section D.1](#)).

D.3 Why the mean and variance of the Chemical Langevin equation are equal

An interesting and useful property of the Chemical Langevin Equation is that the noise term is the square root of the deterministic term, i.e. $\lambda = \sigma^2$ where $dx = \lambda(x, t) dt + \sigma(x, t) dW$. This is because the reaction Poisson distribution is approximated by a normal distribution, and so the mean and variance are equivalent.

Reactions are events that occur with some probability. A natural way to model event probability is with a Poisson distribution $\mathcal{P}(\lambda, t)$ with probability mass function

$$f(k; \lambda, t) = \frac{e^{-\lambda t} (\lambda t)^k}{k!} \quad (\text{D.16})$$

where k is the number of events up to time t , and $\lambda(x) dt$ is a given rate function over an infinitesimal time span dt [136]. A useful property of the Poisson distribution is that the mean and variance are equivalent:

$$\mathbb{E}[\mathcal{P}(\lambda, t)] = \text{Var}[\mathcal{P}(\lambda, t)] = \lambda t$$

If $\lambda t \gg 1$ then $k \sim \mathcal{P}(\lambda t)$ can be approximated by a normal distribution.

$$k \sim \mathcal{P}(\lambda, t) \approx \mathcal{N}(\lambda t, \lambda t) \quad \text{if } \lambda t \gg 1 \quad (\text{D.17})$$

Therefore, in the continuous limit the reaction rate ν and change in concentration $dx \equiv \nu dt$ can be approximated by normal distributions

$$\begin{aligned} \nu &\sim \mathcal{N}(\lambda, \lambda) \\ dx &\sim \mathcal{N}(\lambda dt, \lambda dt) \end{aligned}$$

Finally, we see that

$$\sigma(x, t) = \sqrt{\text{Var}[dx]} = \sqrt{\mathbb{E}[dx]} = \sqrt{\lambda(x)} \quad (\text{D.18})$$

which means we can write

$$dx = \lambda(x) dt + \sqrt{\lambda(x)} dW \quad (\text{D.19})$$

Bibliography

1. Forrest, J., Pan, M., Crampin, E. J., Rajagopal, V. & Stumpf, M. P. *Energy Dependence of Signalling Dynamics and Robustness in Bacterial Two Component Systems* <https://www.biorxiv.org/content/10.1101/2023.02.12.528212v1> (2023). Pre-published.
2. Forrest, J., Rajagopal, V., Stumpf, M. P. H. & Pan, M. BondGraphs.Jl: Composable Energy-Based Modelling in Systems Biology. *Bioinformatics* (Sept. 19, 2023).
3. Forrest, J. *Jedforrest/BondGraphs.Jl* Jan. 16, 2024.
4. Karr, J. R. *et al.* A Whole-Cell Computational Model Predicts Phenotype from Genotype. *Cell* **150**, 389–401 (July 2012).
5. Macklin, D. N. *et al.* Simultaneous Cross-Evaluation of Heterogeneous E. Coli Datasets via Mechanistic Simulation. *Science* **369** (July 24, 2020).
6. Ye, C. *et al.* Comprehensive Understanding of *Saccharomyces Cerevisiae* Phenotypes with Whole-Cell Model WM_S288C. *Biotechnology and Bioengineering* **117**, 1562–1574 (2020).
7. Ahn-Horst, T. A., Mille, L. S., Sun, G., Morrison, J. H. & Covert, M. W. An Expanded Whole-Cell Model of E. Coli Links Cellular Physiology with Mechanisms of Growth Rate Control. *npj Systems Biology and Applications* **8**, 1–21 (Aug. 19, 2022).
8. Maritan, M. *et al.* Building Structural Models of a Whole Mycoplasma Cell. *Journal of Molecular Biology* **434**, 167351 (Jan. 30, 2022).
9. Carrera, J. & Covert, M. W. Why Build Whole-Cell Models? *Trends in Cell Biology* **25**, 719–722 (Dec. 2015).
10. Macklin, D. N., Ruggero, N. A. & Covert, M. W. The Future of Whole-Cell Modeling. *Current Opinion in Biotechnology* **28**, 111–115 (Aug. 2014).
11. Karr, J. R., Takahashi, K. & Funahashi, A. The Principles of Whole-Cell Modeling. *Current Opinion in Microbiology. Antimicrobials • Microbial Systems Biology* **27**, 18–24 (Oct. 1, 2015).

12. Szigeti, B. *et al.* A Blueprint for Human Whole-Cell Modeling. *Current Opinion in Systems Biology*. • *Future of Systems Biology*• *Genomics and Epigenomics* **7**, 8–15 (Feb. 1, 2018).
13. Sun, G., Ahn-Horst, T. A. & Covert, M. W. The E. Coli Whole-Cell Modeling Project. *EcoSal Plus* **9**, eESP-0001–2020 (July 9, 2021).
14. Stumpf, M. P. H. Statistical and Computational Challenges for Whole Cell Modelling. *Current Opinion in Systems Biology* **26**, 58–63 (June 1, 2021).
15. Georgouli, K., Yeom, J.-S., Blake, R. C. & Navid, A. Multi-Scale Models of Whole Cells: Progress and Challenges. *Frontiers in Cell and Developmental Biology* **11**, 1260507 (Nov. 7, 2023).
16. Skalnik, C. J. *et al.* Whole-Cell Modeling of E. Coli Colonies Enables Quantification of Single-Cell Heterogeneity in Antibiotic Responses. *PLOS Computational Biology* **19**, e1011232 (June 16, 2023).
17. Thornburg, Z. R. *et al.* Fundamental Behaviors Emerge from Simulations of a Living Minimal Cell. *Cell* **185**, 345–360.e28 (Jan. 20, 2022).
18. Oster, G., Perelson, A. & Katchalsky, A. Network Thermodynamics. *Nature* **234**, 393–399 (1971).
19. *Bond Graph Modelling of Engineering Systems* (ed Borutzky, W.) (Springer New York, New York, NY, 2011).
20. Gawthrop, P. J. & Bevan, G. P. Bond-Graph Modeling. *IEEE Control Systems Magazine* **27**, 24–45 (Apr. 2007).
21. Gawthrop, P. J. & Crampin, E. J. Energy-Based Analysis of Biochemical Cycles Using Bond Graphs. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* **470**, 20140459 (Nov. 8, 2014).
22. Gawthrop, P. J. & Crampin, E. J. Modular Bond-graph Modelling and Analysis of Biomolecular Systems. *IET Systems Biology* **10**, 187–201 (Oct. 1, 2016).
23. Pan, M., Gawthrop, P. J., Cursons, J. & Crampin, E. J. Modular Assembly of Dynamic Models in Systems Biology. *PLOS Computational Biology* **17**, e1009513 (Oct. 13, 2021).
24. Gawthrop, P. J. & Pan, M. Network Thermodynamics of Biological Systems: A Bond Graph Approach. *Mathematical Biosciences* **352**, 108899 (Oct. 1, 2022).
25. Akbarpour Ghazani, M., Pan, M., Tran, K., Rampadarath, A. & Nickerson, D. P. A Review of the Diverse Applications of Bond Graphs in Biology and Physiology. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* **480**, 20230807 (July 17, 2024).
26. Machado, D. *et al.* Modeling Formalisms in Systems Biology. *AMB Express* **1**, 45 (Dec. 5, 2011).

27. Covert, M. W. *Fundamentals of Systems Biology: From Synthetic Circuits to Whole-cell Models* 356 pp. (CRC Press, Oct. 19, 2017).
28. Price, N. D., Papin, J. A., Schilling, C. H. & Palsson, B. O. Genome-Scale Microbial in Silico Models: The Constraints-Based Approach. *Trends in Biotechnology* **21**, 162–169 (Apr. 1, 2003).
29. Henry, C. S., Broadbelt, L. J. & Hatzimanikatis, V. Thermodynamics-Based Metabolic Flux Analysis. *Biophysical Journal* **92**, 1792–1805 (Mar. 1, 2007).
30. Edelstein-Keshet, L. *Mathematical Models in Biology* 615 pp. (Society for Industrial and Applied Mathematics, Jan. 1, 2005).
31. Codling, E. A., Plank, M. J. & Benhamou, S. Random Walk Models in Biology. *Journal of The Royal Society Interface* **5**, 813–834 (Aug. 6, 2008).
32. Pan, M., Gawthrop, P. J., Tran, K., Cursons, J. & Crampin, E. J. Bond Graph Modelling of the Cardiac Action Potential: Implications for Drift and Non-Unique Steady States. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* **474**, 20180106 (June 30, 2018).
33. Pan, M., Gawthrop, P. J., Tran, K., Cursons, J. & Crampin, E. J. A Thermodynamic Framework for Modelling Membrane Transporters. *Journal of Theoretical Biology. Celebrating the 60th Birthday of Professor Philip Maini* **481**, 10–23 (Nov. 21, 2019).
34. Schomburg, I., Chang, A. & Schomburg, D. BRENDA, Enzyme Data and Metabolic Information. *Nucleic Acids Research* **30**, 47–49 (Jan. 1, 2002).
35. Wittig, U. *et al.* SABIO-RK—Database for Biochemical Reaction Kinetics. *Nucleic Acids Research* **40**, D790–D796 (Jan. 1, 2012).
36. Beber, M. E. *et al.* eQuilibrator 3.0: A Database Solution for Thermodynamic Constant Estimation. *Nucleic Acids Research* **50**, D603–D609 (Jan. 7, 2022).
37. Ederer, M. & Gilles, E. D. Thermodynamically Feasible Kinetic Models of Reaction Networks. *Biophysical Journal* **92**, 1846–1857 (Mar. 15, 2007).
38. Hucka, M. *et al.* The Systems Biology Markup Language (SBML): A Medium for Representation and Exchange of Biochemical Network Models. *Bioinformatics* **19**, 524–531 (Mar. 1, 2003).
39. Lloyd, C. M., Halstead, M. D. B. & Nielsen, P. F. CellML: Its Future, Present and Past. *Progress in Biophysics and Molecular Biology. Modelling Cellular and Tissue Function* **85**, 433–450 (June 1, 2004).
40. Waltemath, D. *et al.* Reproducible Computational Biology Experiments with SED-ML - The Simulation Experiment Description Markup Language. *BMC Systems Biology* **5**, 198 (Dec. 15, 2011).

41. Neal, M. L. *et al.* SemGen: A Tool for Semantics-Based Annotation and Composition of Biosimulation Models. *Bioinformatics* **35**, 1600–1602 (May 1, 2019).
42. Ma, Y. *et al.* *ModelingToolkit: A Composable Graph Transformation System For Equation-Based Modeling* Feb. 9, 2022.
43. Heirendt, L. *et al.* Creation and Analysis of Biochemical Constraint-Based Models Using the COBRA Toolbox v.3.0. *Nature Protocols* **14**, 639–702 (Mar. 2019).
44. Hunter, P. & Safaei, S. Bond Graphs, CellML, ApiNATOMY & OpenCOR (Sept. 21, 2017).
45. Hoops, S. *et al.* COPASI—a COMplex PATHway Simulator. *Bioinformatics* **22**, 3067–3074 (Dec. 15, 2006).
46. Rajagopal, V. *et al.* The Cell Physiome: What Do We Need in a Computational Physiology Framework for Predicting Single-Cell Biology? *Annual Review of Biomedical Data Science* **5**, 341–366 (2022).
47. Agmon, E. *et al.* Vivarium: An Interface and Engine for Integrative Multiscale Modeling in Computational Biology. *Bioinformatics* **38**, 1972–1979 (Mar. 28, 2022).
48. Pan, M., Gawthrop, P. J., Faria, M. & Johnston, S. T. *Thermodynamically-Consistent, Reduced Models of Gene Regulatory Networks* <https://www.biorxiv.org/content/10.1101/2023.11.13.566770v1> (2023). Pre-published.
49. Safaei, S., Blanco, P. J., Müller, L. O., Hellevik, L. R. & Hunter, P. J. Bond Graph Model of Cerebral Circulation: Toward Clinically Feasible Systemic Blood Flow Simulations. *Frontiers in Physiology* **9** (2018).
50. Tran, K., Loiselle, D. S. & Crampin, E. J. Regulation of Cardiac Cellular Bioenergetics: Mechanisms and Consequences. *Physiological Reports* **3**, e12464 (2015).
51. Coomer, M., Ham, L. & Stumpf, M. P. H. Shaping the Epigenetic Landscape: Complexities and Consequences. *bioRxiv*, 2020.12.21.423724 (Dec. 22, 2020).
52. Antić, D., Danković, B., Vidojković, B. & Vidojković, B. M. Bond Graph Modelling and Simulation of Stochastic Systems Using Bondsim–Simulink Tools. *Control Systems and Robotics*, 425 (2004).
53. Yu, T. *et al.* The Physiome Model Repository 2. *Bioinformatics* **27**, 743–744 (Mar. 1, 2011).
54. Cudmore, P., Pan, M., Gawthrop, P. J. & Crampin, E. J. Analysing and Simulating Energy-Based Models in Biology Using BondGraphTools. *The European Physical Journal E* **44**, 148 (Dec. 13, 2021).
55. Borutzky, W. BGML – a Novel XML Format for the Exchange and the Reuse of Bond Graph Models of Engineering Systems. *Simulation Modelling Practice and Theory* **14**, 787–808 (Oct. 1, 2006).

56. Tabatabaian, M. *Engineering Systems Dynamics Modelling, Simulation, and Design* (British Columbia Institute of Technology, Oct. 27, 2021).
57. Filippo, J. M.-D., Delgado, M., Brie, C. & Paynter, H. M. A Survey of Bond Graphs : Theory, Applications and Programs. *Journal of the Franklin Institute* **328**, 565–606 (Jan. 1, 1991).
58. Karnopp, D. C., Margolis, D. L. & Rosenberg, R. C. *System Dynamics: Modeling, Simulation, and Control of Mechatronic Systems* (John Wiley & Sons, Incorporated, Newark, UNITED STATES, 2012).
59. Oster, G. F., Perelson, A. S. & Katchalsky, A. Network Thermodynamics: Dynamic Modelling of Biophysical Systems. *Quarterly Reviews of Biophysics* **6**, 1–134 (Feb. 1973).
60. Soh, K. C. & Hatzimanikatis, V. Network Thermodynamics in the Post-Genomic Era. *Current Opinion in Microbiology. Ecology and Industrial Microbiology • Special Section: Systems Biology* **13**, 350–357 (June 1, 2010).
61. Atkins, P. *Physical Chemistry for the Life Sciences* ().
62. Povarov, V. G., Sokolova, O. B. & Karpova, I. L. Modified Marcelin-de Donder Equations for the Kinetics of One-and Two-Step Reversible Chemical Reactions in Nonideal Solutions. *Russian Journal of Physical Chemistry* **80**, 1874–1879 (Nov. 1, 2006).
63. Couenne, F., Jallut, C., Lefèvre, L., Le Gorrec, Y. & Maschke, B. in *Computer Aided Chemical Engineering* (eds Pleşu, V. & Agachi, P. Ş.) 69–74 (Elsevier, Jan. 1, 2007).
64. Couenne, F., Jallut, C., Maschke, B., Tayakout, M. & Breedveld, P. Structured Modeling for Processes: A Thermodynamical Network Theory. *Computers & Chemical Engineering* **32**, 1120–1134 (June 1, 2008).
65. Van der Schaft, A., Rao, S. & Jayawardhana, B. On the Mathematical Structure of Balanced Chemical Reaction Networks Governed by Mass Action Kinetics. *SIAM Journal on Applied Mathematics* **73**, 953–973 (Jan. 2013).
66. Breedveld, P. C. in *Bond Graphs for Modelling, Control and Fault Diagnosis of Engineering Systems* (ed Borutzky, W.) 5–25 (Springer International Publishing, Cham, 2017).
67. Beard, D. A. & Qian, H. *Chemical Biophysics: Quantitative Analysis of Cellular Systems* (Cambridge University Press, Cambridge, 2008).
68. Gao, X., Gallicchio, E. & Roitberg, A. E. The Generalized Boltzmann Distribution Is the Only Distribution in Which the Gibbs-Shannon Entropy Equals the Thermodynamic Entropy. *The Journal of Chemical Physics* **151**, 034113 (July 19, 2019).
69. Attard, P. *Non-Equilibrium Thermodynamics and Statistical Mechanics: Foundations and Applications* 479 pp. (OUP Oxford, Oct. 4, 2012).
70. Thoma, J. U. Bond Graphs for Thermal Energy Transport and Entropy Flow. *Journal of the Franklin Institute* **292**, 109–120 (Aug. 1, 1971).

71. Brown, F. T. Convection Bonds and Bond Graphs. *Journal of the Franklin Institute* **328**, 871–886 (Jan. 1, 1991).
72. Keseler, I. M. *et al.* The EcoCyc Database in 2021. *Frontiers in Microbiology* **12**, 711077 (July 28, 2021).
73. Orruño, M., Parada, C., Kaberdin, V. R. & Arana, I. *Survival of Escherichia Coli under Nutrient-Deprived Conditions: Effect on Cell Envelope Subproteome* (IntechOpen, July 12, 2017).
74. Yaginuma, H. *et al.* Diversity in ATP Concentrations in a Single Bacterial Cell Population Revealed by Quantitative Single-Cell Imaging. *Scientific Reports* **4**, 6522 (1 Oct. 6, 2014).
75. Amin, M., Porter, S. L. & Soyer, O. S. Split Histidine Kinases Enable Ultrasensitivity and Bistability in Two-Component Signaling Networks. *PLoS Computational Biology* **9** (ed Haugh, J. M.) e1002949 (Mar. 7, 2013).
76. Groisman, E. A. Feedback Control of Two-Component Regulatory Systems. *Annual Review of Microbiology* **70**, 103–124 (Sept. 8, 2016).
77. Jacob-Dubuisson, F., Mechaly, A., Betton, J.-M. & Antoine, R. Structural Insights into the Signalling Mechanisms of Two-Component Systems. *Nature Reviews Microbiology* **16**, 585–593 (Oct. 2018).
78. Jovanovic, G. *et al.* Phosphorelay of Non-Orthodox Two Component Systems Functions through a Bi-Molecular Mechanism in Vivo: The Case of ArcB. *Molecular Biosystems* **11**, 1348–1359 (2015).
79. Papon, N. & Stock, A. M. Two-Component Systems. *Current Biology* **29**, R724–R725 (Aug. 2019).
80. Stock, A. M., Robinson, V. L. & Goudreau, P. N. Two-Component Signal Transduction. *Annual Review of Biochemistry* **69**, 183–215 (June 1, 2000).
81. Groban, E. S., Clarke, E. J., Salis, H. M., Miller, S. M. & Voigt, C. A. Kinetic Buffering of Cross Talk between Bacterial Two-Component Sensors. *Journal of Molecular Biology* **390**, 380–393 (July 17, 2009).
82. Clarke, E. J. & Voigt, C. A. Characterization of Combinatorial Patterns Generated by Multiple Two-Component Sensors in E. Coli That Respond to Many Stimuli. *Biotechnology and Bioengineering* **108**, 666–675 (Mar. 2011).
83. Liu, C., Sun, D., Zhu, J. & Liu, W. Two-Component Signal Transduction Systems: A Major Strategy for Connecting Input Stimuli to Biofilm Formation. *Frontiers in Microbiology* **9**, 3279 (2019).
84. Bitbol, A.-F. & Wingreen, N. S. Fundamental Constraints on the Abundances of Chemotaxis Proteins. *Biophysical Journal* **108**, 1293–1305 (Mar. 10, 2015).

85. Rao, S. D. & Igoshin, O. A. Overlaid Positive and Negative Feedback Loops Shape Dynamical Properties of PhoPQ Two-Component System. *PLOS Computational Biology* **17**, e1008130 (Jan. 4, 2021).
86. Nee, S., Colegrave, N., West, S. A. & Grafen, A. The Illusion of Invariant Quantities in Life Histories. *Science* **309**, 1236–1239 (Aug. 19, 2005).
87. Csikász-Nagy, A., Cardelli, L. & Soyer, O. S. Response Dynamics of Phosphorelays Suggest Their Potential Utility in Cell Signalling. *Journal of The Royal Society Interface* **8**, 480–488 (Aug. 11, 2010).
88. Kim, J.-R. & Cho, K.-H. The Multi-Step Phosphorelay Mechanism of Unorthodox Two-Component Systems in *E. Coli* Realizes Ultrasensitivity to Stimuli While Maintaining Robustness to Noises. *Computational Biology and Chemistry* **30**, 438–444 (Dec. 2006).
89. Rowland, M. A. & Deeds, E. J. Crosstalk and the Evolution of Specificity in Two-Component Signaling. *Proceedings of the National Academy of Sciences* **111**, 5550–5555 (Apr. 15, 2014).
90. Rutter, J. W. *et al.* Engineered Acetoacetate-Inducible Whole-Cell Biosensors Based on the AtoSC Two-Component System. *Biotechnology and Bioengineering* **118**, 4278–4289 (Nov. 2021).
91. Pandey, R., Flockerzi, D., Hauser, M. J. B. & Straube, R. An Extended Model for the Repression of Photosynthesis Genes by the AppA/PpsR System in *Rhodobacter Sphaeroides*. *The FEBS Journal* **279**, 3449–3461 (2012).
92. Righetti, E. & Kahramanoğulları, O. The Inverse Correlation between Robustness and Sensitivity to Autoregulation in Two-Component Systems. *Mathematical Biosciences* **341**, 108706 (Nov. 1, 2021).
93. Straube, R. Analysis of Network Motifs in Cellular Regulation: Structural Similarities, Input-Output Relations and Signal Integration. *Bio Systems* **162**, 215–232 (Dec. 2017).
94. Qian, H. Phosphorylation Energy Hypothesis: Open Chemical Systems and Their Biological Functions. *Annual Review of Physical Chemistry* **58**, 113–142 (2007).
95. Qian, H. & Beard, D. A. Metabolic Futile Cycles and Their Functions: A Systems Analysis of Energy and Control. *IEE Proceedings - Systems Biology* **153**, 192–200 (July 1, 2006).
96. Gawthrop, P. J. & Crampin, E. J. Energy-Based Analysis of Biomolecular Pathways. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* **473**, 20160825 (June 30, 2017).
97. Wuichet, K., Cantwell, B. J. & Zhulin, I. B. Evolution and Phyletic Distribution of Two-Component Signal Transduction Systems. *Current Opinion in Microbiology* **13**, 219–225 (Apr. 2010).

98. Sheng, X., Huvet, M., Pinney, J. W. & Stumpf, M. P. H. in *Evolutionary Systems Biology* (ed Soyer, O. S.) 121–137 (Springer, New York, NY, 2012).
99. Goldbeter, A. & Koshland, D. E. An Amplified Sensitivity Arising from Covalent Modification in Biological Systems. *Proceedings of the National Academy of Sciences* **78**, 6840–6844 (Nov. 1981).
100. Haykin, S. S. & Moher, M. *Communication Systems* 422 pp. (Wiley, 2010).
101. Van Wijk, R., Tans, S. J., Wolde, P. R. ten & Mashaghi, A. Non-Monotonic Dynamics and Crosstalk in Signaling Pathways and Their Implications for Pharmacology. *Scientific Reports* **5**, 11376 (1 June 18, 2015).
102. Toni, T., Jovanovic, G., Huvet, M., Buck, M. & Stumpf, M. P. From Qualitative Data to Quantitative Models: Analysis of the Phage Shock Protein Stress Response in Escherichia Coli. *BMC Systems Biology* **5**, 69 (May 12, 2011).
103. Erguler, K. & H. Stumpf, M. P. Practical Limits for Reverse Engineering of Dynamical Systems: A Statistical Analysis of Sensitivity and Parameter Inferability in Systems Biology Models. *Molecular BioSystems* **7**, 1593–1602 (2011).
104. Gawthrop, P. J., Cursons, J. & Crampin, E. J. Hierarchical Bond Graph Modelling of Biochemical Networks. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* **471**, 20150642 (Dec. 8, 2015).
105. Silk, D., Kirk, P. D. W., Barnes, C. P., Toni, T. & Stumpf, M. P. H. Model Selection in Systems Biology Depends on Experimental Design. *PLOS Computational Biology* **10**, e1003650 (June 12, 2014).
106. Gawthrop, P. J., Cudmore, P. & Crampin, E. J. Physically-Plausible Modelling of Biomolecular Systems: A Simplified, Energy-Based Model of the Mitochondrial Electron Transport Chain. *Journal of Theoretical Biology* **493**, 110223 (May 2020).
107. Rackauckas, C. & Nie, Q. DifferentialEquations.jl – A Performant and Feature-Rich Ecosystem for Solving Differential Equations in Julia. *Journal of Open Research Software* **5**, 15 (1 May 25, 2017).
108. Geitner, G.-H. Bond Graph Add-on Block Library BG V.2.1 MATLAB Central File Exchange. <https://www.mathworks.com/matlabcentral/fileexchange/11092-bond-graph-add-on-block-library-bg-v-2-1> (2024).
109. Venuti, N. Bond Graph Empowered by Mathematica. *Simulation series* **33**, 171–178 (2001).
110. Roesch, E. et al. Julia for Biologists. *Nature Methods*, 1–10 (Apr. 6, 2023).
111. Loman, T. E. et al. Catalyst: Fast and Flexible Modeling of Reaction Networks. *PLOS Computational Biology* **19**, e1011530 (Oct. 18, 2023).
112. Rackauckas, C. et al. *Universal Differential Equations for Scientific Machine Learning* <http://arxiv.org/abs/2001.04385> (2023). Pre-published.

113. Lang, P. F., Jain, A. & Rackauckas, C. SBMLToolkit.JL: A Julia Package for Importing SBML into the SciML Ecosystem. *Journal of Integrative Bioinformatics* (May 28, 2024).
114. Tran, K., Smith, N. P., Loisel, D. S. & Crampin, E. J. A Thermodynamic Model of the Cardiac Sarcoplasmic/Endoplasmic Ca^{2+} (SERCA) Pump. *Biophysical Journal* **96**, 2029–2042 (Mar. 4, 2009).
115. Stanford, N. J. *et al.* Systematic Construction of Kinetic Models from Genome-Scale Metabolic Networks. *PLOS ONE* **8**, e79195 (Nov. 14, 2013).
116. Ham, L., Schnoerr, D., Brackston, R. D. & Stumpf, M. P. H. Exactly Solvable Models of Stochastic Gene Expression. *The Journal of Chemical Physics* **152**, 144106 (Apr. 14, 2020).
117. Ham, L., Jackson, M. & Stumpf, M. Pathway Dynamics Can Delineate the Sources of Transcriptional Noise in Gene Expression. *eLife* **10** (ed Grima, R.) e69324 (Oct. 12, 2021).
118. Mugnai, M. L., Hyeon, C., Hinczewski, M. & Thirumalai, D. Theoretical Perspectives on Biological Machines. *Reviews of Modern Physics* **92**, 025001 (Apr. 7, 2020).
119. Gorin, G., Vastola, J. J. & Pachter, L. *Studying Stochastic Systems Biology of the Cell with Single-Cell Genomics Data* <https://www.biorxiv.org/content/10.1101/2023.05.17.541250v1> (2023). Pre-published.
120. Ruess, J., Ballif, G. & Aditya, C. Stochastic Chemical Kinetics of Cell Fate Decision Systems: From Single Cells to Populations and Back. *The Journal of Chemical Physics* **159**, 184103 (Nov. 14, 2023).
121. Fill, M. *et al.* Ryanodine Receptor Adaptation. *Journal of General Physiology* **116**, 873–882 (Dec. 1, 2000).
122. Jacobs, K. *Stochastic Processes for Physicists: Understanding Noisy Systems* 203 pp. (Cambridge University Press, Feb. 18, 2010).
123. Gardiner, C. *Handbook of Stochastic Methods for Physics, Chemistry, and the Natural Sciences* (Springer, 2004).
124. Cohen, L. The History of Noise [on the 100th Anniversary of Its Birth]. *IEEE Signal Processing Magazine* **22**, 20–45 (Nov. 2005).
125. Lavenda, B. H. *Nonequilibrium Statistical Thermodynamics* 225 pp. (Courier Dover Publications, Apr. 17, 2019).
126. Chaudhuri, D. Active Brownian Particles: Entropy Production and Fluctuation Response. *Physical Review E* **90**, 022131 (Aug. 25, 2014).
127. Needleman, D. & Shelley, M. The Stormy Fluid Dynamics of the Living Cell. *Physics Today* **72**, 32–38 (Sept. 1, 2019).
128. Phillips, R., Kondev, J., Theriot, J. & Garcia, H. *Physical Biology of the Cell* 1089 pp. (Garland Science, Oct. 29, 2012).

129. Kosmann-Schwarzbach, Y. in *The Noether Theorems: Invariance and Conservation Laws in the Twentieth Century* (eds Kosmann-Schwarzbach, Y. & Schwarzbach, B. E.) 55–64 (Springer, New York, NY, 2011).
130. Einstein, A. Über Die von Der Molekularkinetischen Theorie Der Wärme Geforderte Bewegung von in Ruhenden Flüssigkeiten Suspendierten Teilchen. *Annalen der Physik* **322**, 549–560 (1905).
131. Callen, H. B. & Welton, T. A. Irreversibility and Generalized Noise. *Physical Review* **83**, 34–40 (July 1, 1951).
132. Laurendeau, N. M. *Statistical Thermodynamics: Fundamentals and Applications* 472 pp. (Cambridge University Press, Nov. 21, 2005).
133. Yaghoubi, M., Foulaadvand, M. E., Bérut, A. & Łuczka, J. Energetics of a Driven Brownian Harmonic Oscillator. *Journal of Statistical Mechanics: Theory and Experiment* **2017**, 113206 (Nov. 2017).
134. Jhavar, J., Morris, R. G. & Guttal, V. in *Handbook of Statistics* (eds Srinivasa Rao, A. S. R. & Rao, C. R.) 551–594 (Elsevier, Jan. 1, 2019).
135. Gillespie, D. T. A Rigorous Derivation of the Chemical Master Equation. *Physica A: Statistical Mechanics and its Applications* **188**, 404–425 (Sept. 1, 1992).
136. Gillespie, D. T. The Chemical Langevin Equation. *The Journal of Chemical Physics* **113**, 297–306 (July 2000).
137. Smith, L. *Itô and Stratonovich; a Guide for the Perplexed* - OATML <https://oatml.cs.ox.ac.uk/blog/2022/03/22/ito-strat.html> (2024).
138. Ham, L., Coomer, M. A. & Stumpf, M. P. H. The Chemical Langevin Equation for Biochemical Systems in Dynamic Environments. *The Journal of Chemical Physics* **157**, 094105 (Sept. 1, 2022).
139. Alexander, G. Bose–Einstein and Fermi–Dirac Interferometry in Particle Physics. *Reports on Progress in Physics* **66**, 481 (Mar. 2003).
140. Johnson, J. B. Thermal Agitation of Electricity in Conductors. *Physical Review* **32**, 97–109 (July 1, 1928).
141. Argus, F. J., Bradley, C. P. & Hunter, P. J. Theory and Implementation of Coupled Port-Hamiltonian Continuum and Lumped Parameter Models. *Journal of Elasticity* **145**, 339–382 (Aug. 1, 2021).
142. Golo, G., Schaft, A. van der, Breedveld, P. & Maschke, B. in *Nonlinear and Hybrid Systems in Automotive Control* (University of Groningen, Johann Bernoulli Institute for Mathematics and Computer Science, 2003).
143. Innes, M. et al. *Fashionable Modelling with Flux* <http://arxiv.org/abs/1811.01457> (2024). Pre-published.

144. Gawthrop, P. J. Energy-Based Modeling of the Feedback Control of Biomolecular Systems With Cyclic Flow Modulation. *IEEE Transactions on NanoBioscience* **20**, 183–192 (Apr. 2021).
145. Petean, J. *Mathematical Congress of the Americas* 214 pp. (American Mathematical Soc., Jan. 25, 2016).
146. Dickenstein, A. Algebraic Geometry Tools in Systems Biology. *Not. Am. Math. Soc* **67**, 1706–1715 (2020).
147. Kothamachu, V. B., Feliu, E., Cardelli, L. & Soyer, O. S. Unlimited Multistability and Boolean Logic in Microbial Signalling. *Journal of The Royal Society Interface* **12**, 20150234 (July 6, 2015).
148. Kim, K. H. & Qian, H. Entropy Production of Brownian Macromolecules with Inertia. *Physical Review Letters* **93**, 120602 (Sept. 16, 2004).
149. Collin, D. *et al.* Verification of the Crooks Fluctuation Theorem and Recovery of RNA Folding Free Energies. *Nature* **437**, 231–234 (Sept. 8, 2005).
150. Liepe, J. *et al.* Quantitative Time-Resolved Analysis Reveals Intricate, Differential Regulation of Standard- and Immuno-Proteasomes. *eLife* **4**, e07545 (Sept. 22, 2015).
151. Gawthrop, P. J., Pan, M. & Rajagopal, V. *Energy-Based Modelling of Single Actin Filament Polymerisation Using Bond Graphs* <https://www.biorxiv.org/content/10.1101/2024.06.14.598965v1> (2024). Pre-published.
152. *Mathematical Physiology* (eds Keener, J. & Sneyd, J.) red. by Antman, S., Marsden, J. & Sirovich, L. (Springer New York, New York, NY, 2009).
153. Shahidi, N., Pan, M., Tran, K., Crampin, E. J. & Nickerson, D. P. A Semantics, Energy-Based Approach to Automate Biomodel Composition. *PLOS ONE* **17**, e0269497 (June 3, 2022).
154. Liebermeister, W. & Klipp, E. Bringing Metabolic Networks to Life: Convenience Rate Law and Thermodynamic Constraints. *Theoretical Biology and Medical Modelling* **3**, 41 (Dec. 15, 2006).
155. Gawthrop, P. J., Pan, M. & Crampin, E. J. Modular Dynamic Biomolecular Modelling with Bond Graphs: The Unification of Stoichiometry, Thermodynamics, Kinetics and Data. *Journal of The Royal Society Interface* **18**, 20210478 (Aug. 25, 2021).
156. C. Mason, J. & W. Covert, M. An Energetic Reformulation of Kinetic Rate Laws Enables Scalable Parameter Estimation for Biochemical Networks. *Journal of Theoretical Biology* **461**, 145–156 (Jan. 14, 2019).
157. Malik-Sheriff, R. S. *et al.* BioModels—15 Years of Sharing Computational Models in Life Science. *Nucleic Acids Research* **48**, D407–D415 (Jan. 8, 2020).