ADVANCES IN RULE-BASED MODELING: COMPARTMENTS, ENERGY, AND HYBRID SIMULATION, WITH APPLICATION TO SEPSIS AND CELL SIGNALING

by

Justin S. Hogg

B. S. Mathematics, University of Pittsburgh, 2004

Submitted to the Graduate Faculty of

the School of Medicine in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

University of Pittsburgh

2013

UNIVERSITY OF PITTSBURGH

SCHOOL OF MEDICINE

This dissertation was presented

by

Justin S. Hogg

It was defended on

June 27th 2013

and approved by

Dr. James R. Faeder, Computational and Systems Biology

Dr. Gilles Clermont, Critical Care Medicine

Dr. Zoltan Oltvai, Pathology

Dr. Robert S. Parker, Chemical and Petroleum Engineering

Dr. Christopher J. Langmead, School of Computer Science, Carnegie Mellon University

Dissertation Director: Dr. James R. Faeder, Computational and Systems Biology

ADVANCES IN RULE-BASED MODELING: COMPARTMENTS, ENERGY, AND HYBRID SIMULATION, WITH APPLICATION TO SEPSIS AND CELL SIGNALING

Justin S. Hogg, PhD

University of Pittsburgh, 2013

Biological systems are commonly modeled as reaction networks, which describe the system at the resolution of biochemical species. Cellular systems, however, are governed by events at a finer scale: local interactions among macromolecular domains. The multi-domain structure of macromolecules, combined with the local nature of interactions, can lead to a combinatorial explosion that pushes reaction network methods to their limits. As an alternative, rule-based models (RBMs) describe the domain-based structure and local interactions found in biological systems. Molecular complexes are represented by graphs: functional domains as vertices, macromolecules as groupings of vertices, and molecular bonding as edges. Reaction rules, which describe classes of reactions, govern local modifications to molecular graphs, such as binding, post-translational modification, and degradation. RBMs can be transformed to equivalent reaction networks and simulated by differential or stochastic methods, or simulated directly with a network-free approach that avoids the problem of combinatorial complexity.

Although RBMs and network-free methods resolve many problems in systems modeling, challenges remain. I address three challenges here: (i) managing model complexity due to cooperative interactions, (ii) representing biochemical systems in the compartmental setting of cells and organisms, and (iii) reducing the memory burden of large-scale network-free simulations. First, I present a general theory of energy-based modeling within the BioNetGen framework. Free energy is computed under a pattern-based formalism, and contextual variations within reaction classes are enumerated automatically. Next, I extend the BioNetGen language to permit description of compartmentalized biochemical systems, with treatment of volumes, surfaces and transport. Finally, a hybrid particle/population method is developed to reduce memory requirements of network-free simulations. All methods are implemented and available as part of BioNetGen.

The remainder of this work presents an application to sepsis and inflammation. A multi-organ model of peritoneal infection and systemic inflammation is constructed and calibrated to experiment. Extra-corporeal blood purification, a potential treatment for sepsis, is explored *in silico*. Model simulations demonstrate that removal of blood cytokines and chemokines is a sufficient mechanism for improved survival in sepsis. However, differences between model predictions and the latest experimental data suggest directions for further exploration.

Keywords: rule-based modeling, biochemical kinetics, cell signaling, sepsis, inflammation, hemoadsorption.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1.0 RULE-BASED MODELING OF BIOCHEMICAL SYSTEMS

This chapter introduces reaction network and rule-based formalisms for modeling the kinetics of biochemical systems. First, I review reaction network formalism for kinetic modeling, the workhouse of systems modeling. The standard methods for generating trajectories from network models are presented, including continuous and discrete stochastic methods. Particular attention will be paid to the limitations of network-based methods in the context of cell signaling applications. Next, I will present Rule-based modeling as a solution to some of the limitations of network methods. The connection between rule-based models and reaction network, via network generation, will be presented. Network-free simulation methods, which extend simulation capability to massive and infinite networks, will be described. Finally, limitations of rule-based modeling will be discussed, which provides the starting point for methodological developments in subsequent chapters.

1.1 SYSTEMS BIOLOGY

If the task of the "omics", e.g. genomics and proteomics, is cataloging the parts of the cell and establishing the basic relationships among the parts; then the task of systems biology is understanding how parts come together to form systems that sense the environment, make decisions, and regulate the cellular activity [1]. There are two contrasting but complementary approaches to the task. The first is a data-driven approach, a natural extension of the "omics". The data driven paradigm emphasizes high-throughput experiments that collect data on a whole cell scale, such as microarray expression studies. Statistical analysis and machine learning methods are then applied to the data set with the intent of finding patterns and relationships in the data.

The big-data approach stands in contrast to the mechanistic modeling approach. Mechanistic modeling, a bottom-up approach, has roots in the principles of physics and chemistry. Mechanistic models begin with only a few parts and the interactions among them and construct formal models based on a biochemical and biophysical understanding of the interactions. The models are analyzed with both qualitative and quantitative methods to determine the properties of the system. As the small system is understood in isolation, the scope of the model is expanded to include more of the surrounding context. As the scope of mechanistic modeling increases, the necessity of reconciling mechanistic models with big-data becomes apparent. Thus, the two contrasting approaches are complementary in the end. Indeed, a number of studies that combine high-throughput data with mechanistic models have been published [2–4]

The *holy grail* of mechanistic modeling is a complete model of the cell based on biophysical and biochemistry principles. Much progress has been made towards a complete model of the cell. A complete genomic and biochemical model of the simple bacterium *Mycoplasma genitalium*, published in 2012 [5], demonstrates progress towards this goal. But much work remains, especially in regards to eukaryotic cells, which are an order of magnitude more complex than bacteria.

Understanding the behavior of biochemical systems is a difficult task that often confounds our natural intuition. Simple biochemical circuits are capable of producing a wide variety of behaviors, such as linear response, step response, oscillations, on-off switching, and so forth [6]. Predicting the behavior of a system requires knowledge of the mathematical relationships between the components. Furthermore, two systems with the same mathematical form but different parameters can behave very differently. For example, a negative feedback amplifier can become an oscillator with only a change in parameter values. Therefore, it is important to analyze dynamical systems with quantitative methods rather than relying on intuition.

Nonetheless, there do exist rigorous qualitative analysis methods for dynamical systems. Uri Alon et al. [7–9] identified a number of regulatory motifs that are capable of specific kinds of behavior. Although the presence of a motif is not always a sufficient condition for a certain behavior, it is often necessary condition. Thus, qualitative methods are a useful tool when anchored to theoretical underpinnings.

Biochemical systems are non-linear and, as there is no grand theory of non-linear dynamics, it is often necessary to rely on numerical simulations of biochemical systems. Computational methods are especially important for large systems with multiple feedback loops and a variety of network motifs that confound intuition.

1.2 REACTION NETWORK MODELING

The reaction network (RN) is a common formalism for kinetic modeling of biochemical systems [10]. A reaction network is composed of a set of biochemical species, and a set of reactions that describe transformations of species from one kind to another, and a propensity (i.e. rate) function for each reaction. RNs can be interpreted as continuous or discrete stochastic process. The former can be represented as systems of ODEs and the later as continuous-time, discrete-state Markov chains. RNs are a workhorse of systems modeling, with successful applications in cell cycle regulation [11–14], extrinsic apoptosis [15, 16], EGF receptor signaling [17, 18], the MAP kinase cascade [19], inflammation [20–23] and many more.

RN is best illustrated by example. Consider a simple model of ligand-induced receptor phosphorylation. Let $\{\mathtt{L}, \mathtt{R}, \mathtt{LR}, \mathtt{LRp}\}$ be a set of species representing free ligand, free receptor, ligand-receptor complex, and phosphorylated ligand-receptor complex. The set of reactions describes the biochemical interactions among the species: ligand-receptor binding, and receptor phosphorylation.

$$\begin{aligned} \mathtt{L} + \mathtt{R} &\overset{k_{1+},k_{1-}}{\rightleftharpoons} \mathtt{LR} \\ \mathtt{LR} &\overset{k_2}{\rightarrow} \mathtt{LRp} \\ \mathtt{LRp} &\overset{k_3}{\rightarrow} \mathtt{LR} \end{aligned}$$

The species on the left-hand side of a reaction are called *reactants* and those on the right-hand side are *products*. The arrow, $\rightarrow$, represents the direction of the reaction (usually left to right). A bi-directional arrow, $\leftrightharpoons$, indicates that the reaction can be reversed, i.e. the products will transform into the reactants. The parameters k_{1+}, k_{2-}, etc. are mass-action rate constants (more on this below).

In the present example there are four reactions (one reversible, and two irreversible). The first reaction transforms a free ligand `L` and a free receptor `R` into a ligand-receptor complex `LR`. The reaction is *bimolecular* since their are two reactants. The reverse reaction does the opposite: a ligand-receptor complex dissociates into free ligand and free receptor. The third reaction transforms a ligand-receptor complex into a phosphorylated receptor complex `LRp`. This reaction is *unimolecular* since it has just one reactant. The final reaction is the reverse of the third. If the mechanism of the forward and reverse reaction is distinct, by convention the reactions are written separately rather than as single reversible reaction. In this case, phosphorylation and dephosphorylation are usually accomplished by different enzymes, so the reactions are written separately.

Note that the names of the species are entirely arbitrary in a RN, so long as distinct species have distinct names. In the example, the names were chosen to imply the biochemical role of the species, e.g. the ligand is represented by `L`. However, the species could be labeled in a nondescript fashion, such as `X1`, `X2`, and so forth. An important implication of this fact is that the names do not necessarily contain any information about the composition of the biochemical species. For example, we may infer that `LR` is a ligand-receptor dimer, but a species like `X2` is a black box. Consistent naming conventions are important for human-readability. But even the best conventions, unless formalized, are prone to misinterpretation.

1.2.1 Mass-action kinetics

Mass-action kinetics, sometimes mistakenly called the Law of Mass Action, states that the rate of a reaction is proportional to the product of reactant concentrations (see, e.g. [24]). Mass-action kinetics can be derived from collision theory under the assumptions of well-mixed conditions and elementary reaction mechanism, i.e. the reaction occurs in one step. For this reason, mass-action kinetics are also called *elementary* kinetics. In the example above, mass-action kinetics assigns the following reaction rates:

$$\text{rate1} = k_{1+}[\mathtt{L}][\mathtt{R}]$$

$$\text{rate2} = k_{1-}[\mathtt{LR}]$$

$$\text{rate3} = k_{2}[\mathtt{LR}]$$

$$\text{rate4} = k_{3}[\mathtt{LRp}].$$

Mass-action kinetics are often applied in the continuum limit of large particle concentrations, where rates can be interpreted as a continuous flux of mass through the reaction. But mass-action principles also apply to discrete systems, where the rate is interpreted as the probability the reaction will "fire" per unit time. In a seminal 1976 paper, D.T. Gillespie [25] derived mass-action kinetics for discrete stochastic systems. Gillespie showed that, under certain assumptions, the probability for a specific set of particles to react via reaction r in the infinitesimal time interval δt is given by $k\delta t$, where k is the *per instance* rate constant. The total probability for any set of particles to react is given by the rate constant multiplied by a combinatorial factor corresponding to the number of distinct reactant sets.

Consider the simple reaction `A + B -> C` with rate constant k. Let the population vector $\vec{x} = (x_{\mathtt{A}}, x_{\mathtt{B}})$ have number units. Each pair of particles `A` and `B` has a probability $k\delta t$ of reacting in time δt. Assuming the population vector has number units, there are $x_{\mathtt{A}}x_{\mathtt{B}}$ ways to select a pair of reactants. Thus the probability of any pair reacting in the interval δt is $kx_{\mathtt{A}}x_{\mathtt{B}}\delta t$, corresponding to a total rate $kx_{\mathtt{A}}x_{\mathtt{B}}$.

The combinatorial factor is somewhat more complicated when two or more reactants belong to identical species. Nonetheless, it is still a simple matter of combinatorics. Consider the reaction `A + A -> B`. The number of ways to choose a pair of `A` particles is:

$$\binom{x_{\mathtt{A}}}{2} = \frac{x_{\mathtt{A}}(x_{\mathtt{A}}-1)}{2}.$$

Thus the total reaction rate is:

$$k\frac{x_{\mathtt{A}}(x_{\mathtt{A}}-1)}{2}. \tag{1.1}$$

In limit of large concentrations, the total rate approaches $kx_{\mathtt{A}}^2/2$, which is the continuum mass action rate with constant $k/2$. There should be no confusion about the origin of the factor of 2; this is a simple consequence of combinatorics.

1.2.2 Non-elementary kinetics

Non-elementary reaction rates are common in biological systems models. Non-elementary rates are appropriate where the reaction models a multi-step mechanism that is either unknown or unnecessary to model in detail. Michaelis-Menten kinetics, perhaps the best known non-elementary kinetic rate, describes a two-step enzymatic reaction:

$$\mathtt{S} + \mathtt{E} \overset{k_+,k_-}{\rightleftharpoons} \mathtt{ES} \overset{k_{cat}}{\rightarrow} \mathtt{P} + \mathtt{E}. \tag{1.2}$$

The first step is reversible binding of a substrate, `S`, to enzyme, `E`. The second step is the irreversible conversion of substrate to product, `P`. The Michaelis-Menten mechanism assumes that the intermediate complex, `ES`, is in quasi-equilibrium with `S` and `E`. Eliminating `ES`, the coarse-grain reaction is:

$$\mathtt{S} + \mathtt{E} \overset{k_{cat},k_M}{\rightarrow} \mathtt{P} + \mathtt{E}. \tag{1.3}$$

with reaction rate

$$\frac{k_{\text{cat}}[E]_{\text{tot}}[S]}{k_M + [S]}, \tag{1.4}$$

where $[E]_{\text{tot}}$ is the sum of free and bound enzyme, and

$$k_M = \frac{k_{\text{cat}} + k_-}{k_+} \tag{1.5}$$

is the Michaelis constant.

1.2.3 Formal description of reaction networks

Formally, a reaction network is defined by the tuple:

$$RN \quad \models \quad \left(N_S \in \mathbb{N}, \quad N_R \in \mathbb{N}, \quad \vec{x} \in \mathbb{N}^{N_s}, \quad \{\vec{r_j},\ \vec{p_j},\ f_j\colon \mathbb{N}^{N_s} \rightarrow \mathbb{R}_{\geq 0}\}_{j=1}^{N_r}, \right),$$

where N_S is the number of species, N_R is the number of reactions, $\vec{x}$ is the species population vector, and $\vec{r_j}$, $\vec{p_j}$, an f_j are the *reactant* vector, the *product* vector, and the propensity function for reaction j. The reactant vector encodes the reactants, with $r_{i,j}$ being the number of species i consumed per reaction j. The product vector is similarly defined. The propensity function maps the population vector to the *propensity* of reaction.

A reaction event transforms the species population vector by consuming the reactants and yielding the products. To make this precise, define the stoichiometry vector for reaction j to be $\vec{s_j} = \vec{p_j} - \vec{r_j}$. Element $s_{i,j}$ of the stoichiometry vector is the net change in the population of species i due to reaction j:

$$\vec{x} \overset{r_j}{\mapsto} \vec{x} + \vec{s}_j.$$

The propensity function $f_j(\vec{x})$ is the rate of reaction j. The function is usually based on mass-action kinetics, but in general is any function of the population species vector. The general form of the mass-action propensity function is given by

$$f_j(\vec{x}) = k \prod_{i=1}^{N_S} \prod_{n=1}^{r_{j,i}} \frac{x_i - (n-1)}{n} \quad \xrightarrow{\|\vec{x}\| \to \infty} \quad k \prod_{i=1}^{N_S} \frac{x_i^{r_j}}{r_j!}, \tag{1.6}$$

with units per time.

Reaction networks are equivalent to stochastic Petri nets, a formal language for modeling distributed computing systems [26]. Species correspond to places, reactions to transitions, stoichiometry matrix to arcs, and species populations to tokens. Thus, formal analysis techniques for Petri nets are equally applicable to reaction networks.

1.2.4 Simulation as a discrete stochastic system

1.2.4.1 Gillespie's Stochastic Simulation Algorithm (SSA) The Gillespie direct method is an algorithm for sampling kinetic trajectories from a discrete chemical reaction system [25, 27, 28]. In the limit of a well-mixed system, it is an exact sampling method. Since the method is simple and also important to the field, I will outline the algorithm here. At each step, a reaction is selected to "fire". The time until the next reaction is sampled from the exponential distribution with rate given by the sum of reaction propensities, and the specific reaction is selected by sampling from the propensity-weighted distribution over all reactions. The reaction is fired by adding the stoichiometry vector to the population vector and updating the time. Finally, reaction propensities are updated to reflect the new population vector. Pseudocode for Gillespie's direct method is listed in Algorithm 1.

Algorithm 1: Gillespie's direct method (SSA)

```
input..: $(\vec{s}_j, f_j)_{j=1}^{N_R}$,  list of reactions (stoichiometry vector, propensity fcn)
       : $\vec{x} = (x_i)_{i=1}^{N_S}$, initial species population;  $t$, start time;  $t_{\text{end}}$, end time
output: $\vec{x}$, final species population

foreach $j \leftarrow 1 \ldots N_R$ do
    $a[j] \leftarrow f_j(\vec{x})$                                  compute reaction propensity
$a_{\text{tot}} \leftarrow \sum_{j=1}^{N_R} a[j]$                   compute total propensity
while $t < t_{end}$ do                                              fire next reaction
    $u_1, u_2 \leftarrow \text{Uniform}(0,1)$
    $\Delta t \leftarrow -\log(u_1)/a_{\text{tot}}$                 time until next reaction
    if $t + \Delta t \geq t_{end}$ then last
    $j \leftarrow \min\{k : \sum_{l=1}^{k} a[l] > a_{\text{tot}} u_2\}$   select reaction
    $t \leftarrow t + \Delta t$                                     update time
    $\vec{x} \leftarrow \vec{x} + \vec{s}_j$                        update species vector
    foreach $j \leftarrow 1 \ldots N_R$ do
        $a[j] \leftarrow f_j(\vec{x})$                              update propensity
    $a_{\text{tot}} \leftarrow \sum_{j=1}^{N_R} a[j]$
```

The computational cost of each reaction event with respect to the number of reactions, N_R, depends on the cost of selecting the next event and updating propensities. The original direct method selected the next reaction by direct search, an $O(N_R)$ cost. The cost of updating propensities depends on the density of the reaction dependencies. At worst, every reaction must be updated, leading to $O(N_R)$ cost. But if the number of dependencies per reaction can be bound below some number k, then $O(1)$ cost can be achieved by implementing a dependency graph. In either case, the overall cost is $O(N_R)$ per reaction event.

Several variants of the direct method have been developed in an effort to reduce the computational cost. Gibson and Bruck [29] developed the "next reaction method", which relies on a heap structure for selecting the next reaction and a dependency graph for propensity updates. The Gibson-Bruck method reduces the asymptotic cost to $O(\log N_R)$ per event, assuming that the number of dependencies per reaction is bounded. Slepoy et al. developed a constant time algorithm by organizing reactions in logarithmic classes of reaction propensities [30]. The method relies on the assumptions that propensities are bounded above and below by positive values, and the number of dependencies per reaction is bounded.

In practice, Cao et al. found that the direct method, with reactions presorted by propensity, performed better than the the next reaction method or logarithmic classes on typical reaction networks [31]. Despite the higher asymptotic cost of the direct method, the constant factor was dominant for "typical" reaction networks considered.

1.2.4.2 Accelerated simulation methods The time cost of an exact stochastic simulation depends on various factors in addition to the number of reactions. Perhaps most important is the propensity of the reactions. As the average reaction propensity increases, the number of events that fire during a fixed time interval also increases. Accelerated methods have been developed in an attempt to reduce the cost of such simulations. The τ-leaping method assumes the number of times a reaction fires in a fixed interval τ is Poisson distributed [32–34]. This approximation holds so long the expected relative change in propensity during the fixed interval is small. The simulation parameter $0 < \epsilon \ll 1$ sets the tolerance of the approximation. As ϵ goes to zero, τ-leaping converges to the exact distribution.

τ-leaping is problematic when a rare reaction event leads to a large change (ϵ) in the propensity of other reactions. If this is the case, the τ-leaping assumption breaks down. Several methods have been developed to attack this situation. Slow-scale SSA divides the reactions into fast and slow groups [35]. The fast reactions are sampled with τ-leaping and the slow reactions by exact sampling. The "Partitioned Leaping Algorithm" (PLA) separates reactions into four regimes: exact stochastic, Poisson distributed, Langevin (SDE), and continuous (ODE) [36,37]. The partition is computed automatically and adjusts on the fly if the propensity of a reaction changes during the course of the simulation.

1.2.5 Simulation as a continuous system

In the limit of large concentrations, a reaction network can be interpreted as a system of ordinary differential equations (ODEs). Consider species i with population x_i. The rate of change of x_i is given by the summing over the contributions of all reactions

$$\frac{dx_i}{dt} = \sum_{j=1}^{N_r} s_{i,j} f_j(\vec{x}).$$

Defining the stoichiometry matrix, $S = (s_{i,j})$, and the vector of reaction propensities, $\vec{f} = (f_j)_{j=1}^{N_R}$, this can be written more compactly as a system of ODEs

$$\frac{d\vec{x}}{dt} = S\vec{f}(\vec{x}). \tag{1.7}$$

This system of (typically) non-linear ODEs can be numerically integrated to generate trajectories. A number of libraries exist for fast, accurate, and robust integration of ODEs, including SUNDIALS CVODE [38]. ODEs are also amenable to dynamical analysis and other qualitative, but rigorous, analytic techniques. The XPP software package is a tool for numerical analysis of ODEs, including facilities for plotting trajectories, nullclines, and vector fields, as well as bifurcation analysis [39].

1.3 COMBINATORIAL COMPLEXITY

Proteins involved in signaling, especially in eukaryotic cells, have a modular domain structure [40–42]. A domain is a component of a protein, often corresponding to a tertiary substructure, that has a specific function or property. A domain might serve as a site for protein-protein interaction, covalent modification (e.g. phosphorylation or ubiquitination), or both. Alternatively, a domain might have catalytic activity, such as a kinase or phosphotase activity, or anchor a protein in the cell membrane. Signaling proteins are often composed of multiple domains with different functions. For example, a transmembrane receptor might consist of an extracellular ligand-binding domain, a transmembrane domain, a cytosolic kinase domain, and several sites of covalent modification. Domains are modular in the sense that the function of each domain is retained in the absence of the other domains. For example, a kinase domain severed from its parent protein will typically retain its catalytic activity. Nonetheless, allostery, i.e. cooperative interactions among protein domains, often plays an important role in the protein function.

The domain structure of proteins and the local nature of interactions leads to a combinatorial explosion in the number of configurations that a signaling complexes can achieve [43, 44]. To illustrate combinatorial complexity, consider a protein with N phosphorylation sites. Assuming the sites are independently phosphorylated, the protein has 2^N different configurations. Each configuration corresponds to a species in a reaction network model. Each species can participate in N different reactions, each corresponding to switching one of the N sites between the phosphorylated and unphosphorylated state. In total there are $N \cdot 2^N$ reactions in the network. So a relatively simple protein model induces a large number of species and reactions due to combinatorial complexity.

The above example is contrived, but nonetheless represents a common feature of signaling networks. Combinatorial complexity arises in a variety of signaling pathways, including FcϵRI signaling [45,46], epidermal growth factor receptor activation [47], p53 [48], G-coupled protein receptors [49], JAK-family kinases [50], MAP kinase scaffolds [51] and more.

Consider the epidermal growth factor (EGF) receptor. Blinov et al. [52] constructed a model of EGF receptor activation including 5 proteins, representing a total of 10 domains, 6 protein-protein interactions, and 3 sites of phosphorylation. Despite the simple parts list, the reaction network contained 356 distinct species participating in more than 3000 reactions. Where do all these species come from? It is a simple consequence of combinatorics.

The 5 proteins are ligand (EGF), receptor (EGFR), and three cytosolic proteins (Shc, Grb2, and Sos). EGF has a single domain for binding EGFR. The receptor has four domains: an extracellular EGF-binding domain, a receptor dimerization domain, and two sites of tyrosine phosphorylation. Tyrosine phosphorylation activates binding interfaces for Grb2 and Shc, respectively. Grb2 has two binding domains, one for phosphorylated EGFR and other for Sos. Shc has an EGFR binding domain and a site of tyrosine phosphorylation and Grb2 binding. Sos has a single domain for Grb2 binding.

Consider the configurations of individual receptor sites, excluding the dimerization domain. The ligand binding site can unbound or bound to EGF (2 states). The Grb2 binding site can be unmodified, phosphorylated, bound to Grb2, or bound to a Grb2-Sos complex (4 states). The Shc site can be unmodified, phosphorylated, bound to Shc, phosphorylated Shc, a Shc-Grb2 complex, or a Shc-Grb2-Sos complex (6 states). The total number of a configurations for a single receptor is $2 \cdot 4 \cdot 6 = 48$. If we consider the dimerized receptor pairs, after accounting for symmetry, the number of configurations is $48 \cdot (48 + 1)/2 = 1176$. There are also 8 species that do not involve the receptor. In total there are 1232 possible species! But the model only has 356 due to simplifying assumptions, e.g. only ligand-bound receptors will dimerize.

In this relatively simple model, a few interactions among multidomain proteins leads to an explosion in the number of species and reactions in the network. While it is still theoretically possible to write a network model with any finite number of reactions, it is difficult in practice to implement a model with more than a few hundred reactions without relying on automated methods. Furthermore, large network models increase computational burdens. In the best case, the space complexity of a reaction network is linear in the number of reactions. But since the number of reactions in a combinatorial system tend to grow exponentially in the number of protein domains[1], the space complexity is exponential in the number of domains in the model. Thus, a small increase in the biological scope can leads to

[1]For an example, see the comparison of reaction network size for variants of the FcϵRI signaling model [53].

exponential growth in the network. The time cost of combinatorial systems also tends grows with the number of reactions, since the assumptions required by constant- and logarithmic-time algorithms do not typically hold for combinatorial systems. Specifically, the number of reaction dependencies per reaction grows with the number of reactions.

The combinatorial problem is further exacerbated in systems with branching polymerization. In such systems, the number of possible species is only limited by the number of particles in the system. But the number of species grows exponentially in the number of particles. Thus, even a few hundred particles can lead to a network size that exceeds Avogadro's number. Ligand-induced FcϵRI aggregation models prominently feature branching polymerization that confounds network-based approaches [45, 53].

Combinatorial complexity pushes reaction network methods to the brink and then over the edge. For such problems, alternative methods are required. Fortunately, the solution already exists: rule-based languages and network-free simulation.

1.4 RULE-BASED MODELING

Rule-based modeling (RBM) is motivated by the modular domain structure of macromolecules and the problem of combinatorial complexity [54]. In RBM, macromolecules are represented by structured objects that reflect the domain structure of their biological counterparts. Each domain is represented by a node in a graph, called a *site* or *component*. Each site has a *type* property and an optional *state* property. Type is a label that describes the domain represented by the site, e.g. SH2. State is a label that represents a discrete modification to the site, such as post-translational modification, conformation, etc. Domains that belong to the same macromolecule are grouped together to form a supernode, called a *molecule* or an *agent*. Each *molecule* has a type property corresponding to the type of macromolecule, e.g. EGFR. Noncovalent bonds between macromolecules are represented by edges between the sites that mediate the protein-protein interaction.

Transformation of macromolecular complexes is governed by *reaction rules*. A reaction rule consists of a set of reactant patterns, a set of product patterns, and a rate law. The reactant patterns describe molecular motifs that are potential sites of reaction. When a reaction occurs, the reactant sites are transformed to the configuration described by the product patterns. The rate law assigns the propensity for each set of reaction sites to undergo reaction.

Reaction rules are analogous to organic reactions. Consider the esterification of an alcohol and a carboxylic acid:

$$\texttt{R-OH} + \texttt{R}'\texttt{-COOH} \rightarrow \texttt{R}'\texttt{-COO-R} + \texttt{H}_2\texttt{O}. \tag{1.8}$$

This reaction scheme produces an ester and a water molecule by reacting an alcohol group with a carboxylic acid group. This scheme describes a vast number of reactions. `R` and `R'` represent the unknown portions of the reacting molecules, which are not relevant to the reaction mechanism. It is sufficient that the first reactant species has an alcohol group and the second has a carboxylic group. In RBM, reactant patterns play the role of functional groups. Like an organic reaction scheme, any molecular complex that contains the reactant pattern motif can participate in the reaction. Thus, a reaction rule efficiently describes a large class of reactions.

By representing molecular complexes by graphs, and classes of reactions by rules, RBM goes a long way towards solving the combinatorial problem. Since the graph object retains information about the composition and structure, the reactions that a species undergoes can be computed by matching reactant motifs in the molecular structure. Furthermore, the products of each reaction can be computed by operating on the reactant graphs, e.g. adding an edge, changing a state, deleting nodes. Therefore, starting from an initial set of species and reaction rules, it is possible to construct any reachable species by repeated application of the reaction rules.

1.4.1 Network generation

Network generation is the process transforming a rule-based model into an equivalent reaction network. Network generation is simple at its core. The reaction rules are applied to the initial set of species in every possible combination. Products are constructed by transforming the reactant graph. Unique products are added to the species set and the process is iterated until no new species are generated. Network generation usually proceeds under the assumption that unlimited copies of each initial species are available. Thus, the network is not limited by the availability of reagents.

It is not always possible to generate a network, since a RBM may encode an infinite number of species and reactions. For example, a polymerization rule will generate a polymer species for each possible length, resulting in a countably infinite species set. Conversely, every finite reaction network can be obtained by network generation applied to some RBM [2]. Therefore, RBM are a superset of finite reaction network models.

Network generation is outlined in Algorithm 2. If more depth is desired, see Appendix B for further details.

1.4.2 Network-free Simulation

Rule-based models permit compact representation of reaction network models; and, when the reaction network is not too large, network generation provides access to all the simulation methods available for reaction networks. But for larger reaction networks ($> 10^3$ species, $> 10^4$ reactions), the computational cost of simulating networks becomes prohibitive [53]. In such cases, an alternative approach is required.

Network-free (NF) simulation is an approach that avoids network generation [55]. NF methods are similar to agent-based approaches in that each molecule in the system is represented as an individual particle. This stands in contrast to network-based methods, where molecular entities belonging to the same species are lumped together and represented as a population variable. NF methods are fundamentally stochastic. Under the RBM formalism, molecular complexes are represented as graphs; thus, it is possible to identify reactant sites "on-the-fly" during a simulation. Upon selection of reactant sites, graph transformations are applied *in situ* to construct the products.

[2]A finite reaction network is a RBM where each rule describes exactly one reaction and each molecule contains zero components. Thus, every reaction network corresponds to a trivial RBM. But it is usually possible to find a corresponding RBM with fewer rules than reactions.

Algorithm 2: Network generation algorithm

```
input..: $(\vec{R}_r, \mathcal{T}_r, k_r)_{r=1}^{N_R}$, a list of rules (reactant patterns, transformation, rate)
       : seed_species, set of initial species
output: species, set of all species; reactions, set of all reactions

queue ← seed_species    //a queue of species to process
species ← seed_species; reactions ← {}
foreach $r \leftarrow 1 \ldots N_R$ do    //initialize rules
    foreach $p \leftarrow 1 \ldots |\vec{R}_r|$ do
        sites$[r][p]$ ← {}    //sites$[r][p]$ tracks reactant $p$ sites for rule $r$

while queue is not empty do    //generate network
    $s \leftarrow$ next(queue)
    foreach $r \leftarrow 1 \ldots N_R$ do    //loop over rules
        foreach $p \leftarrow 1 \ldots |\vec{R}_r|$ do    //loop over reactant patterns
            $Z_p \leftarrow \{$all reactant $p$ sites in species $s\}$
            foreach $(z_1, \ldots, z_{|\vec{R}_r|}) \in \Big($sites$[r][1] \times \ldots \times Z_p \times \ldots \times$ sites$[r][|\vec{R}_r|]\Big)$ do
                transform sites $z_1 \ldots z_{|\vec{R}_r|}$ under $\mathcal{T}_r$, obtaining products $y_1 \ldots y_{|\vec{P}_r|}$
                foreach $y' \leftarrow y_1 \ldots y_{|\vec{P}_r|}$ do
                    if $y' \notin$ species then add $y'$ to species and queue
                map each reactant site $z_1 \ldots z_{|\vec{R}_r|}$ to its parent species $x_1 \ldots x_{|\vec{R}_r|}$
                construct reaction: $rxn = [x_1 + \ldots + x_{|\vec{R}_r|} \xrightarrow{k} y_1 + \ldots + y_{|\vec{P}_r|}]$
                if $rxn \notin$ reactions then add $rxn$ to reactions
            add sites in $Z_p$ to sites$[r][p]$
```

Pseudocode for network-free simulation is listed in Algorithm 3. Network-free simulation gains efficiency by grouping reactions by rules. At each step of a simulation, the next reaction rule is selected. Then a reactant set is chosen from among the possible sets for the selected rule. This strategy is efficient since each potential reactant set has equal probability of reacting. Therefore, the total propensity of a rule is computed by multiplying the number of sites for each reactant pattern, and specific reactants are drawn from the uniform distribution over reactant sites for each reactant pattern. Thus, computing rule propensities and selecting reactants have constant cost in the number of sites. Therefore, the cost of selecting the next reaction is linear in the number of reaction rules (worst case) and constant in the number of reactant sites.

Upon selecting the reactants, the molecules are transformed in place. Transformation can create or destroy reactant sites, so the list of sites must be updated after each event. If site lists were calculated from scratch at each step, the algorithm would be quite inefficient. Fortunately, it is only necessary to update molecules that belong to the same complex as one of the reactant sites [3]. Thus, the time cost of updating site lists is a function of the number of reactant patterns in the rule, R; the number of molecules on the complexes matching the reactant patterns; and the total number of reactant patterns in the system, N_{patt}. If we assume the number of molecules per complex is bounded by C, then the number of updates is no more than RCN_{patt}. With suitable data structures, the cost of adding or removing items from the site list is constant in the size of the list [56]. Thus, the cost of an update is no greater than $CR\sum_{n=1}^{N_{\text{patt}}} \Phi(n)$, where $\Phi(n)$ is the cost of checking if a molecule is a site for reactant pattern n. Since reactant motifs are usually simple, we will assume $\Phi(n)$ is bounded. Likewise, since the number of reactant patterns is usually one or two, we will assume R is bounded. Under theses assumptions, the total update cost is $O(CN_{\text{patt}})$, which is constant in both the number of particles and number of reactions in the network. If we can also assume C is bounded (a reasonable assumption in non-polymerizing models), then the cost is $O(N_{\text{rules}})$. Since the number of rules is typically much less than the number of reactions, the NF update step scales efficiently in the class of models where the assumptions

[3]For a large class of models it is sufficient to update molecules within a fixed graph distance of the reactant site [56].

Algorithm 3: Network-free simulation algorithm

```
input..: $(\vec{R}_r, \vec{P}_r, \mathcal{T}_r, k_r)_{r=1}^{N_R}$, list of rules (reactant/product patterns, transform, rate);
       : $\mathsf{Mols}$, initial set of molecules;  $t$, start time;  $t_{\text{end}}$, end time
output: $\mathsf{Mols}$, final set of molecules

foreach $r \leftarrow 1 \ldots N_R$ do    //initialize rules
    foreach $p \leftarrow 1 \ldots |\vec{R}_j|$ do    //initialize reactant site lists
        $\mathsf{sites}[r][p] \leftarrow$ (all reactant $p$ sites in $\mathsf{Mols}$)
    $a[r] \leftarrow k_r \prod_{p=1}^{|\vec{R}_r|} |\mathsf{sites}[r][p]|$    //rule propensity
$a_{\text{tot}} \leftarrow \sum_{r=1}^{N_R} a[r]$    //total propensity
while $t < t_{end}$ do    //fire next event
    $u_1, u_2 \leftarrow \text{Uniform}(0,1)$
    $\Delta t \leftarrow -\log(u_1)/a_{\text{tot}}$
    if $(t + \Delta t \geq t_{end})$ then last
    $r \leftarrow \min\{j : \sum_{k=1}^{j} a[k] > a_{\text{tot}} u_2\}$    //select rule
    foreach $p \leftarrow 1 \ldots |\vec{R}_r|$ do    //select reactant sites
        $s_p \leftarrow \text{Uniform}(\mathsf{sites}[r][p])$
    if any pair of sites in $s_1, \ldots, s_{|\vec{R}_r|}$ belong to the same complex then
        $t \leftarrow t + \Delta t$; next
    $\mathcal{U}_{\text{mols}} \leftarrow$ {all molecules connected to sites $s_1, \ldots, s_p$}    //molecules to update
    remove all sites in molecules of $\mathcal{U}_{\text{mols}}$ from $\mathsf{sites}$
    apply transformation $\mathcal{T}_r$ to sites $s_1, \ldots, s_{n_R}$
    remove molecules deleted by $\mathcal{T}_r$ from $\mathcal{U}_{\text{mols}}$
    add molecules synthesized by $\mathcal{T}_r$ to $\mathcal{U}_{\text{mols}}$
    foreach $r \leftarrow 1 \ldots N_R$ do
        foreach $p \leftarrow 1 \ldots |\vec{R}_r|$ do    //update site lists
            add all reactant $p$ sites found in $\mathcal{U}_{\text{mols}}$ to $\mathsf{sites}[r][p]$
        $a[r] \leftarrow k_r \cdot \prod_{p=1}^{|\vec{R}_r|} |\mathsf{sites}[r][p]|$
    $a_{\text{tot}} \leftarrow \sum_{r=1}^{N_R} a[r]$
    $t \leftarrow t + \Delta t$
```

hold. However, C may grow very large for models with a gel-phase (e.g. trivalent-ligand bivalent-receptor aggregation [45,53]).

Several implementations of network-free simulators are available, including KaSim [57], NFsim [53], and DYNSTOCH [58]. NFsim is notable in its support for functional rate laws that depend on system properties ("global functions") and local properties of reactants ("local functions").

1.4.3 BioNetGen: a rule-based modeling platform

BioNetGen (BNG) is a collection of open-source software tools for specifying, simulating, and analyzing rule-based models of complex biochemical systems [59–62]. BNG software reads models written in the BNG language (BNGL), a formal, text-based grammar for specification of rule-based models. BNG software parses BNGL models and represents them internally as graph objects. BNG includes tools for network generation and network-based simulation, including ODE and SSA methods. Alternatively, BNGL models can be exported for simulation by NFsim, a network-free simulator compatible with BNGL, or VCell, a biochemical modeling platform with support for detailed spatial representation and PDE simulation [63–65]

Since material in subsequent chapters extends BNGL, it is important that the reader develop a familiarity with the syntax before proceeding. This chapter will present the basics of BNGL syntax and its interpretation as graph objects. This material should provide the necessary intuition, but further reading may be required for a deeper understanding. To that end, BNGL syntax is presented in extended Backus-Naur form in Appendix A and the graph-based formalism underlying BNG is included in Appendix B. For further reading, see the BioNetGen book chapters [60,61] or journal publications [59,66].

Although BNGL was chosen as a foundation for this work, it should be straightforward to generalize methods to other RBM languages. The κ-calculus [67,68] and the Stochastic Simulation Compiler (SSC) [69] are closely related to BNGL. The partial network expansion method (Chapter 4) and energy-modeling approach (Chapter 2) could be readily implemented for these RBM languages.

1.4.4 Macromolecules as structured objects

The fundamental object in BNGL is the *molecule*. Molecules are structured objects that contain *components*. Components have state values (possibly null) and serve as sites for bond formation. Typically, molecules correspond to biological macromolecules such as proteins, while components represent protein functional domains. Component states may describe post-translational modifications, such as phosphorylation, or other properties of the macromolecule, such as conformational state.

Molecules are defined through *molecule types* templates that specify the set of components, and the possible state values of those components, that comprise a molecule. For example, in Blinov et al. [52] the molecule type definition for the epidermal growth factor receptor (EGFR) is

```
Egfr(l,r,Y1068~Y~pY,Y1148~Y~pY) .
```

This specifies that `Egfr` molecules have four components: `l`, `r`, `Y1068`, and `Y1148`. The first two represent a ligand binding domain and a receptor dimerization domain, respectively. The other two represent tyrosine residues 1068 and 1148, which are sites of phosphorylation [70, 71]. Both `Y1068` and `Y1148` can exist in two possible states, `Y` (tyrosine) and `pY` (phosphorylated tyrosine). Names of molecule types, components, and states may be assigned any arbitrary string of alphanumeric characters [60,61], but conventionally are chosen to correspond to biological entities.

Molecules can be thought of as *instances* of molecule types, the difference being that in molecules a state for each molecular component must be explicitly specified. For example,

```
Egfr(l,r,Y1068~pY,Y1148~Y)
```

specifies an unligated `Egfr` monomer with a phosphorylated tyrosine at residue 1068 and an unphosphorylated tyrosine at residue 1148.

1.4.5 Molecular complexes as graphs

BNGL represents molecular complexes within a hierarchical graph formalism: components are represented by nodes, molecules are groupings of nodes, and bonds are edges between components [66, 72]. Molecules that are connected by bonds between their components are said to belong to the same *complex*. Bound components are designated by a shared `!LABEL` token following each component in the bond. For example,

```
Egf(r!0).Egfr(l!0,r!1,Y1068~pY,Y1148~Y).Egfr(l,r!1,Y1068~pY,Y1148~Y)
```

specifies a complex comprised of an `Egfr` dimer and a bound `Egf` ligand. The `Egf` ligand is bound to the first `Egfr` molecule through a bond, labeled '`0`', between its `r` component and the `l` component of the `Egfr` molecule. The two `Egfr` molecules are bound to each other through a second bond, labeled '`1`', between their `r` components. The '`.`' symbol specifies that two molecules are part of the same complex. Molecules that are bound to each other are necessarily in the same complex but molecules in the same complex need not be bound directly to each other (e.g., the `Egf` ligand and the second `Egfr` molecule).

Two complexes that have the same molecular composition, with the same configuration of bonds and states, are said to be instances of the same *species*. More formally, two complexes are of the same species if their graphs are identical up to a labeled isomorphism [66].

1.4.6 Molecular motifs as subgraphs

Patterns describe structural motifs that may be found within complexes. In BNGL, patterns are written just like complexes except that components, states, bonds, and molecules that are not part of the motif are omitted. A pattern is said to *match* a complex (or species) if the motif is found within the complex. Note that it is possible for a pattern to match at multiple places within the same complex. In terms of graphs, a pattern is a *subgraph* and a match is a *subgraph isomorphism* from the pattern into the complex graph [59, 66]. As an example, the pattern `Egfr(l,r)` matches each instance of `Egfr` with unbound `l` and `r` components, including

$\underline{\texttt{Egfr(l,r}}\texttt{,Y1068}\sim\texttt{Y,Y1148}\sim\texttt{pY}\underline{\texttt{)}}$,

$\underline{\texttt{Egfr(l,r}}\texttt{,Y1068}\sim\texttt{pY,Y1148}\sim\texttt{pY}\underline{\texttt{)}}$, and

$\underline{\texttt{Egfr(l,r}}\texttt{,Y1068}\sim\texttt{pY!0,Y1148}\sim\texttt{pY}\underline{\texttt{)}}\texttt{.Grb2(SH2!0,SH3)}$, where the matches have been underlined. Since the components `Y1068` and `Y1148` are omitted from the pattern they can be in any of their allowed states and either bound or unbound.

A pattern may also require that a component be bound without specifying its binding partner by substituting the bond label with the '`+`' wildcard (e.g., `Egfr(l!+,r)`). Similarly, the bonding state can be specified as irrelevant (either bound or unbound) by using the '`?`' wildcard (e.g., `Egfr(l!?,r)`). Finally, if a component is listed but no state is indicated then the pattern will match the component regardless of state.

Patterns have two primary purposes in BNGL [60,61]: (i) they specify parts of a complex that are involved in a reaction; (ii) they define the observable outputs (or simply *observables*) of a model. The second function is critical for facilitating comparisons between simulation predictions and experimental observations.

1.4.7 Biochemical reaction rules as graph transformations

Reaction rules (or simply *rules*) describe classes of reactions that transform the states of complexes in a system. A rule consists of a set of reactant patterns, a set of product patterns, a directionality arrow (unidirectional '`->`' or bidirectional '`<->`'), and a rate law. The reactant patterns match reaction sites (motifs) within complexes in the system. The product patterns indicate how the reactant patterns are modified when the rule fires. The rate law determines the firing rate per set of reactant matches. In terms of graphs, a rule is a graph transformation with a dynamic rate that determines how frequently the transformation is applied to each matching subgraph [66].

There are six basic transformations that can be encoded in a BNGL rule: (i) bond addition, (ii) bond deletion, (iii) state change, (iv) molecule synthesis, (v) molecule deletion, and (vi) complex deletion. [4] The transformations are implied by the differences between the reactant and product patterns in a rule. For example, the rule

```
Egf(r) + Egfr(l,r) -> Egf(r!0).Egfr(l!0,r) k
```

[4]Complex *synthesis* is not considered to be an elemental transformation but rather a composite action comprised of molecule syntheses and bond additions.

matches `Egf` molecules with unbound `r` components and `Egfr` molecules with unbound `l` and `r` components. The rate law is of the mass-action type (by default [60,61]) with a rate constant `k`. When the rule fires a bond is formed between the `r` component of `Egf` and the `l` component of `Egfr`. We refer the reader to Refs. [60,61] for further details regarding the specifics of the BNGL syntax.

1.4.8 Anatomy of a BNGL model file

BioNetGen models are specified within text files having the `.bngl` extension. Model elements are specified in blocks delimited by `begin` and `end` tags. The entire model specification is enclosed within a `begin model` / `end model` block and the model specification itself is contained within six additional blocks: `parameters`, `molecule types`, `seed species`, `observables`, `functions`, and `reaction rules`. Though not strictly required, it is suggested that these blocks appear in this order. Molecule types, reaction rules, and observables have been discussed above. Values of rate parameters, initial populations of species, etc., are listed in the `parameters` block and can be referenced in any subsequent block. All initially-populated species and their population levels are specified in the `seed species` block. These act as the starting point for network generation in BioNetGen and define the initial system state for any subsequent simulations. Arbitrary functions of observables can be defined in the `functions` block and used in place of rate constants in the `reaction rules` block. Two types of functions can be defined: global and local [53,73]. Local functions take as an argument a reference to a complex or molecule and evaluate observables over that complex/molecule only. In contrast, global functions evaluate observables over the entire domain.

Any actions to be performed on the model must be listed after the model specification (i.e., the `end model` directive). Common actions include generating a network by iterative application of the rule set to the set of initial seed species, simulating a network (either deterministically or stochastically), performing a "network-free" simulation, changing or resetting the values of species populations or rate parameters, reading in a previously-generated network, and exporting the model and/or network in various file formats (e.g., SBML [74], M-file [75]). If multiple actions are listed in sequence then each is performed using the final system state from the previous action as the initial state for the current action (unless the system is explicitly reset to the initial conditions). For example, one might run an initial equilibration simulation in the absence of ligand, then add a dose of ligand and run a second simulation to investigate the dose-response behavior of the system. We refer the reader to Refs. [60–62] for further details regarding model specification and action syntax in BioNetGen.

1.4.9 A Survey of Rule-based Languages and software

A number of rule-based languages, in addition to BNG, are described in the literature. In this section, I will briefly survey some of these languages.

The κ-calculus is formal RBM language similar to BNGL. In contrast to BNGL, κ does not natively support identical component types (i.e. sites) within a molecule (i.e. agent). Furthermore, κ does not formally make the distinction between inter- and intra-molecular reactions. The simplicity of κ lends itself to formal analysis. In practice, these limitations are side-stepped by meta-κ, an extension of the language κ language that "compiles" to pure κ. A network-free simulator, KaSim, is available for simulating κ. Exact model reduction methods are also available κ models [76].

The Stochastic Simulation Compiler (SSC) is a rule-based language and software platform. SSC generates a reaction network and compiles a custom SSA simulator to achieve maximum performance [69]. The SSC language is similar to BNGL and κ, but the approach to reaction rules is somewhat different. In BNGL and κ, graph transformations are inferred from the differences between reactant patterns and product patterns. This requires a convention for mapping molecules on the reactant side to their counterparts on the product side. In contrast, SSC rules are specified by a set of reactant patterns and a set of transformations. Thus, the transformation is explicit, rather than implied. SSC also supports a coarse-grained representation of compartments and space via sub-volume discretization.

Simmune is a platform for multi-scale simulation of cellular systems [77,78]. At the small scale, simmune implements rule-based methods for biochemical interactions. Biochemical signals are coupled to a coarse-grained, dynamic representation of cellular morphology. Cell morphology is modeled as a cellular Potts models. Although RBM features in Simmune are conceptually similar to BNGL and κ, models are specified using a graphical user interface rather than a text-based language.

BioCham [79] is language for biochemical modeling with string-based formalism. Proteins are represented by string tokens and complexes are formed by string concatenation. Despite the formal differences, BioCham is a rule-based language in that reaction rules describe classes of reactions. Reaction rules are implemented by substring matching and transformation of the parent string. The BioCham platform implements a number of analytic tools, including bifurcation analysis and formal verification of temporal logic.

Relatively new to the scene, ML-rules is a multi-level rule-based language [80]. Objects in ML-rules can be nested arbitrarily to construct systems with multiple levels, such as extracellular space, cells, and organelles. Object nesting is dynamic and governed by rules that permit movement between levels, creation of new levels, and fusion of objects on the same level. ML-rules extends RBM to biological processes such as receptor-mediated endocytosis and the cell division.

1.5 THE LIMITS OF RULE-BASED MODELING

RBM languages provide a solution to the problem of combinatorial complexity at the level of model description, while network-free methods resolve the simulation problem. But the scale of modeling enabled by RBM also introduces new challenges. A few of those challenges, which provide motivation for the remainder of this work, are presented in this section.

1.5.1 Biochemistry is compartmentalized, RBM 1.0 is not

Biochemistry takes place in the compartmentalized structure of cells. All cells have membranes that separate the outside world from the inside. Eukaryotic cells are further partitioned into cytosol, nucleus, and other organelles. The compartmental structure also exists at the level of the organism, which is composed of organs and tissues separated by endothelial and epithelial barriers. Signaling pathways cross multiple levels of cellular structure. Consider EGF receptor signaling, described earlier. The EGF molecule is present in the extracellular space, but is detected by a transmembrane receptor. The receptor recruits and activate proteins in the cytosol. The EGF signal cascades through the MAP kinase pathway, activating transcription factors that are transported into the nucleus and regulator gene expression.

Membranes play a key role in cellular biochemistry. In addition to dividing the cell into compartments, membranes also mediate communication between the compartments via transmembrane proteins. Membranes also serve as a surface for localizing macromolecules, which can lead to enhanced reaction rates due to proximity. Despite the importance of membranes and compartmentalization, the first generation of RBM languages have limited facilities for representing membranes, or localizing molecules and reactions to specific compartments [5]. This motivates the development of *compartmental BNGL* (cBNGL), an extension of BNGL for compartmental modeling (Chapter 3).

1.5.2 The curse of cooperativity

Reaction rules describe a class of biochemical reactions with a common transformation and rate law. Allostery—energetic cooperativity between distant sites on macromolecules—is a ubiquitous feature of biological systems [81]. In the presence of cooperativity, reactions that share a transformation but differ in context will have different rate constants. Thus, it is not possible to lump such reactions into a single rule. This is the curve of cooperativity: it is everywhere and it complicates everything.

Consider receptor dimerization reactions in three different contexts.

```
R  + R  <-> RR   kp1,km1
LR + R  <-> LRR  kp2,km2
LR + LR <-> LRRL kp3,km3
```

The reactions describe receptor-receptor binding, but differ in ligand context. If the ligand binding site is allosterically coupled to the receptor dimerization site, then the rate constants

[5]Since work on cBNGL was initiated in 2008, a number of additional rule-based languages with compartmental features have appeared in the literature, including SSC [69] and ML-rules [80].

for each reaction will be different. Thus, the reactions cannot be described by a single reaction rule with a common rate constant[6]. The situation is further complicated by constraints among the rates constants introduced by thermodynamics cycles in the reaction network.

For the construction of simple models, cooperativity can be handled manually. But the problem is exceedingly difficult for large, combinatorially complex models. Energy-based modeling, a recent development in RBM [49], provides a partial solution by modeling cooperativity directly and constructing contextual rule variations in a principled manner. Building on this foundation, Chapter 2 presents a general framework and software implementation for energy-based modeling in BNG.

1.5.3 Network-free simulation of large systems is expensive

Network-free (NF) methods enable simulation of models with combinatorially large reaction networks. But since NF is intrinsically discrete stochastic, each molecule in the system must be represented individually. Thus, systems with a large number of particles require proportionally large memory resources. Although seldom a limiting factor for single pathway models, memory requirements will become prohibitive as the field moves away from single pathway models toward multi-pathway and whole cell models. Multi-cellular models will require resources on yet another order of magnitude.

A number of strategies, including multi-scale modeling and model reduction techniques, can reduce the cost of simulation. However, the potential for reduction in computational cost of NF simulations has not been fully exploited. This motivates the development of the Hybrid Particle/Population method in Chapter 4. This method substantially reduces memory requirements of NF simulations by lumping species with high copy number.

1.5.4 Other limitations not addressed in this dissertation

Although RBM is a powerful and flexible framework, it is not applicable to all types of systems. RBM represents the connectivity of molecules via edges in a graph. Thus, RBM can be

[6] It is possible to describe all three reactions with a single rule if the rate is given by a *local function*, an advanced concept introduced by Sneddon et al. [53]. However, local functions will not resolve the interdependencies among the rates induced by thermodynamic cycles.

viewed as “topological” modeling since the connectivity of molecules, but not the geometric relationships, are described. Encoding distance and angles in current RBM frameworks is impractical, at best. Thus, RBM is inadequate for modeling systems where geometry plays an important role. For example, although RBM can represent the connectivity of amino acids in a protein, representing angles and distances required for a molecular dynamic simulation is not practical. Similarly, it is difficult to represent steric hindrance or rigidity that prevents certain types of intramolecular bonds. Although future research may integrate RBM with geometric models, this limitation is not addressed in the present work.

RBM also assumes that the state of a macromolecule can be adequately represented by a discrete site structure. Thus, The modeling process begins with a choice of the discrete structure for each macromolecule in the system. This information can be inferred from crystallized structure or other experimental studies; however, at present there is no systemic method for selecting the site structure. Furthermore, the modeling process may be confounded if the sites in the biological entity are not accurately described by a collection of discrete sites. Although it may be possible to approximate the behavior of a protein with a finer-grained discrete representation, this approach is both tedious and subject to combinatorial explosion of influences among the sites. Again, this limitation is left for future work in the field.

2.0 MODELING ENERGY: A PATTERN-BASED APPROACH

2.1 MOTIVATION FOR ENERGY-BASED MODELING

A *reaction rule* in a rule-based model (RBM) encodes a class of reactions that share a common transformation, such as bond formation or post-translational modification, and a common rate law. The reactions differ only in the larger context of the reacting complexes, but this does not influence the rate of reaction. If many reactions can be represented by a single reaction rule, then the reaction network is highly compressible. The amount of compression depends on the degree of kinetic homogeneity that the modeler (or the data) is willing to accept.

The fundamental assumption of RBM is summarized in the maxim: "don't write, don't care". In other words, if a molecule component does not appear in the reaction rule, then its state does not influence the rate of reaction. *Reaction context* is the set of components named in the rule, but not modified. Reducing the reaction context in a rule typically increases the number of reactions described by the rule, while reducing the number of kinetic parameters in the model. Thus, there is a close relationship between reaction context and model complexity. If a class of reactions with a common rate law can explain the data as well as independent rate laws, then the common rate law is preferred. Therefore, eliminating reaction context (as much as the data will allow) is an attempt to follow the principle of Occam's Razor. In the language of RBM, if two models explain the data, the one with least reaction context is preferred.

Unfortunately, reaction context plays a very important role in biophysical models. Allostery, i.e. cooperativity between distant sites on macromolecules, is a common feature in biological systems. Allostery is mediated by a variety of mechanisms, such as ligand binging [82], covalent modification of protein residues [83–85], and conformational state changes [86, 87]. Allostery is known or postulated to play in role in various biophysical systems, including hemoglobin/oxygen binding [82], ligand-induced EGF receptor dimerization [88], and G-protein coupled receptors [89–91].

Allosteric mechanisms, which are fundamentally related to reaction context, require the reaction network to be partitioned into finer classes. Consequently, the modeler is forced to write a large number of related reaction rules with different context and kinetic rates. I will refer to the complexity induced by reaction context with the term *contextual complexity*. It is important to distinguish this from the distinct concept of *combinatorial complexity*. Combinatorial complexity is the explosion of the reaction network size due to combinations of multi-domain protein interactions, e.g. polymerization, scaffold assemblies, etc. In contrast, contextual complexity is the explosion in the number of reaction rules required to represent a reaction network due to reaction context. RBM effectively compresses combinatorially complex networks so long as contextual complexity is minimal. As contextual complexity increases, the benefits of RBM diminish. Under this reality, is RBM useful modeling paradigm for systems with cooperative mechanisms?

In this chapter, I will address the problem of contextual complexity by coupling rule-based modeling with an energy-based formalism. Energy-based approaches were first merged with RBM by Ollivier et al. [49]. In that work, the authors developed a formalism for modeling allostery mediated by conformational change. The energy-based approach to RBM, or EBM for brevity, was later generalized to arbitrary pattern graphs by Vincent Danos et al. [92, 93]. Under generalized EBM, energy is assigned to patterns, i.e. molecular motifs. These *energy patterns* form the basis of a free energy accounting system. Free energy of species formation, and subsequently the free energy change of a reaction, is computed by counting free energy motifs in the species and summing the free energy contributions. Given a set of free energy inequalities, Danos showed that it is a straightforward task to verify that a kinetic model satisfies those constraints. However, the existence of free energy assignments that are compatible with a given kinetic model is, in general, undecidable. Nonetheless, Danos suggested that kinetic models that are compatible with free energy assignments could be obtained *by construction*. The Allosteric Network Compiler (ANC) [49] demonstrates the construction approach in the limited context of allostery via conformational change.

In this chapter, I develop a general approach to constructing kinetic models that are compatible with free energy assignments *by construction*. Under my approach, reaction rules are decomposed into two components: *atomic* transformations (i.e. bond formation, post-translational site modification, etc.) and a set of *energy patterns*. Energy patterns refer to motifs in molecular complexes (bonds, states, pairs of bonds, etc.) and form the building blocks of a free energy accounting system for molecular complexes. Cooperativity among components, i.e. reaction context, is encoded in the set of energy patterns. Reactions are generated by applying the atomic transformations to the initial species set (the usual approach) and then kinetics are derived from the change in free energy of the reaction, as computed from the energy patterns. This approach has several advantages over traditional RBM. Most notably, (i) EBM typically requires fewer rules, (ii) rules have less context, and (iii) the resulting reaction network is guaranteed to satisfy detailed balance. Thus, the EBM approach is a promising solution to the problem of contextual complexity.

This chapter begins with an simple example of contextual complexity in a familiar hemoglobin model. Then I will briefly present the theory of free energy and detailed balance in the context of chemical reaction systems. I then proceed to develop the formal underpinnings of energy-based modeling, and demonstrate how the approach manages contextual complexity and guarantees detailed balance. A motivating example that illustrates the method will be developed in parallel with EBM. Then I present several advances in the energy-based approach, beyond the work of Danos and Ollivier; specifically, a generalization of energy-based kinetics that permits pure catalysis, and a method for integrating non-equilibrium processes such as ATP driven reactions. Finally, I present software for EBM within the BioNetGen platform, which is the major contribution in this chapter.

2.1.1 Contextual complexity: hemoglobin example

Consider a simple model of hemoglobin/oxygen binding. We will treat the hemoglobin tetramer as a single object with four identical binding sites for oxygen and two conformations, *tense* (T) and *relaxed* (R). We will assume that oxygen can bind to hemoglobin in either tense or relaxed conformations, although binding is more favorable in the tense state. Finally, we assume that hemoglobin exists in an equilibrium of mixed tense and relaxed conformations, even in the absence of oxygen. This is the Monod-Wyman-Changeaux model of hemoglobin [82, 83].

In the BioNetGen language (BNGL), the molecule types are:

```
begin molecule types
   # Hemoglobin tetramer
   Hgb(c~R~T,o2,o2,o2,o2)
   # Free oxygen
   O2(hgb)
end molecule types
```

where the `c` component represents the conformational state and the `o2` components represent the oxygen-binding heme group in the Hemoglobin subunits.

The system can be described by seven reaction rules: two for oxygen binding and five for conformation changes.

```
begin reaction rules
   # binding
   Hgb(c~R,o2) + O2(hgb) <-> Hgb(c~R,o2!1).O2(hgb!1)  kp1, km1
   Hgb(c~T,o2) + O2(hgb) <-> Hgb(c~T,o2!1).O2(hgb!1)  kp2, km2
   # conformation change
   Hgb(c~R,o2,o2,o2,o2)          <-> Hgb(c~T,o2,o2,o2,o2)          kp3, km3
   Hgb(c~R,o2!+,o2,o2,o2)        <-> Hgb(c~T,o2!+,o2,o2,o2)        kp4, km4
   Hgb(c~R,o2!+,o2!+,o2,o2)      <-> Hgb(c~T,o2!+,o2!+,o2,o2)      kp5, km5
   Hgb(c~R,o2!+,o2!+,o2!+,o2)    <-> Hgb(c~T,o2!+,o2!+,o2!+,o2)    kp6, km6
   Hgb(c~R,o2!+,o2!+,o2!+,o2!+)  <-> Hgb(c~T,o2!+,o2!+,o2!+,o2!+)  kp7, km7
end reaction rules
```

The first thing we notice is the five different rules that correspond to the conformational change from relaxed to tense. Each rule represents the same transformation in different context. This is an example of *contextual complexity*. Not only is it tedious to enumerate the rules, but each contextual variant introduces additional parameters. Although it is tempted to simply assign the same kinetic rates to each equation, this will effectively decouple oxygen binding from conformational state, which defeats the purpose of the model. Parameterization is further complicated by hidden constraints in the set of parameters due to “loops” in the reaction network. These constraints, required by the principle of detailed balance will be described in the next section.

To simplify the discussion, I will introduce a shorthand representation for the species in the model system. Each hemoglobin species will be represented by a letter describing the conformational state (R or T) and a number $(0, 1, 2, 3, 4)$ representing the number of bound oxygen. For example, a hemoglobin tetramer in the tense conformation with two bound oxygen will be denoted T2. Finally, free oxygen is represented by the letter O2.

The reaction network implied by this rule model contains 11 species and 13 reversible reactions. The species are: O2, R0, R1, R2, R3, R4, T0, T1, T2, T3, T4. The reactions are as follows,

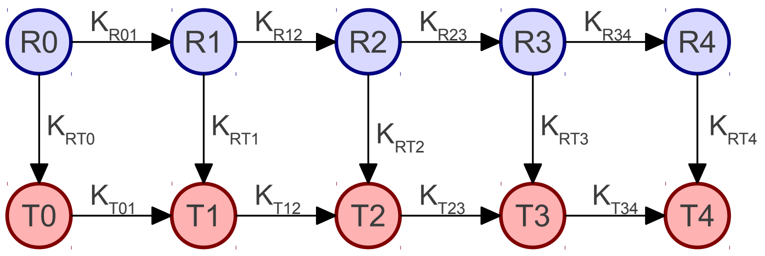
```
begin reactions
   R0 + O2 <-> R1  4*kp1, km1
   R1 + O2 <-> R2  3*kp1, 2*km1
   R2 + O2 <-> R3  2*kp1, 3*km1
   R3 + O2 <-> R4  1*kp1, 4*km1
   T0 + O2 <-> T1  4*kp2, km2
   T1 + O2 <-> T2  3*kp2, 2*km2
   T2 + O2 <-> T3  2*kp2, 3*km2
   T3 + O2 <-> T4  1*kp2, 4*km2
   R0      <-> T0  kp3, km3
   R1      <-> T1  kp4, km4
   R2      <-> T2  kp5, km5
   R3      <-> T3  kp6, km6
   R4      <-> T4  kp7, km7
end reactions
```

where the rate constant multipliers are due to the number of free oxygen sites on the hemoglobin (forward reaction) or the number of occupied heme sites (reverse reaction). Figure 2.1 shows the possible states of hemoglobin tetramers and the allowed transitions between states. There are a number of loops in the state transition diagram. As we will see in the next section, these loops introduce parameter constraints.

2.2 FREE ENERGY PRINCIPLES FOR BIOCHEMICAL SYSTEMS

This section introduces the basics of free energy for biochemical kinetics. The first sections introduce the concepts of energy, entropy, and free energy for systems in equilibrium. The discussion is informal with an emphasis on developing an intuition for free energy as probability. Next, Gibbs' free energy is presented along with its relationship to reaction equilibrium in biochemical systems. This leads to the principle of detailed balance, which

Figure 2.1: **State transition diagram for hemoglobin model.** Each circle represents a possible state for a Hemoglobin tetramer. Possible state transitions are indicated by arrows labeled by the respective equilibrium constants. Although state transitions are bidirectional, the direction of the arrow indicates the frame of reference for the equilibrium constants.



places constraints on the parameterization of kinetic models.

This brief introduction to energy, entropy and free energy draws inspiration from *Statistical Physics of Biomolecules* by Daniel M. Zuckerman [94], chapter 3 in particular. The reader is encouraged to read this text if further background on statistical mechanics is required. For further depth on statistical physics in the context of biochemical systems, the reader is directed to *Chemical Biophysics* by Daniel A. Beard and Hong Qian [95].

2.2.1 Energy

Energy is a quantity associated with a specific state of a biochemical system. The energy of a state is proportional to the logarithm of the probability that the system, at equilibrium, will be in that state. (The system is at equilibrium if the probability distribution is stationary.) If x is a state, then:

$$p(x) \propto \exp \frac{-E(x)}{k_B T} \tag{2.1}$$

where $E(x)$ is the energy of the state, k_B is the Boltzmann constant, T is the absolute temperature, and $p(x)$ is the probability of the state. If a system can take on N discrete states labeled $(s_1, s_2, \ldots, s_N)$, then the *partition function* for the system is defined to be:

$$Z = \sum_{n=1}^{N} \exp \frac{-E(s_n)}{k_B T}. \tag{2.2}$$

The normalized probability of a state can be obtained from the energy and the partition function as follows:

$$p(s) = \frac{1}{Z} \exp \frac{-E(s)}{k_B T}. \tag{2.3}$$

The partition function plays an important role in statistical mechanics, but for purposes here it is sufficient to view Z as a normalization factor.

Consider a system with N particles, where each can take one of two states ($x_i \in \{0, 1\}$). Assuming the probability for particle i is independent of particle j, for all $i \neq j$, the probability of system state $\vec{x} = (x_1, x_2, \ldots, x_N)$ is given by the product of the individual probabilities:

$$p(\vec{x}) = \prod_{n=1}^{N} \frac{1}{Z} \exp \frac{-E(x_n)}{k_B T} = \frac{1}{Z^N} \exp \frac{-\sum_{n=1}^{N} E(x_n)}{k_B T}. \tag{2.4}$$

In words, the energy of the system is given by the sum of the energy over the particles.

2.2.2 Entropy

Entropy is a measure of disorder in a system. Consider a box with N particles. Suppose a molecule can take on two conformations, `A` and `B`, with equal probability. Suppose k such molecules populate our system. The entropy of this system state is proportional to the logarithm of the number of ways N particles can be arranged so that k have conformation `A` and $N - k$ have conformation `B`. More precisely, the entropy is given by

$$S = k_B \log \Omega_{N,k}, \tag{2.5}$$

where S is the entropy, k_B is the Boltzmann constant, and Ω is the number of configurations. The number of configurations, in this case, is given by the the number of ways to choose k objects from a set of N, i.e. $\binom{N}{k} = \frac{N!}{k!(N-k)!}$. If $k = 0$ or $k = N$, the number of configurations is 1 and the entropy is 0. If N and k are sufficiently large, Sterling's approximation can be used to estimate the entropy:

$$\begin{aligned} S &= k_B \log \Omega(N,k) \;=\; k_B \log \frac{N!}{k!(N-k)!} \;=\; k_B \left(\log N! - \log k! - \log(N-k)!\right) \\ &\approx k_B \left((N \log N - N) - (k \log k - k) - ((N-k)\log(N-k) - (N-k))\right) \\ &= k_B \left(N \log \frac{N}{N-k} + k \log \frac{N-k}{k} \right). \end{aligned} \tag{2.6}$$

To find the maximum of the entropy, differentiate by k and solve for the critical points to obtain

$$\frac{dS}{dk} = 0 \iff \frac{N-k}{k} = 1 \iff k = \frac{N}{2}. \tag{2.7}$$

Therefore, the entropy of the system is maximized when half the particles have conformation `A` and half `B`. Since our intuition tells us the molecules will be divided about equally between `A` and `B`, we may anticipate that maximizing the entropy has a relationship to most likely system configurations.

Before moving on, I will introduce some terminology. A *macrostate* is a subset of all system *microstates* that are indistinguishable. In the example above, the microstates of the system are given by the vectors $\vec{x} = (x_1, x_2, \ldots, x_N$, where x_i is the conformation of particle n. A microstate $\vec{x}$ belongs to the macrostate k if $|\{i : x_i = \mathtt{B}\}| = k$. The microstates belonging to macrostate k are indistinguishable since, for purposes of well-mixed chemical kinetics, the identity of the k molecules in configuration `B` has no effect on the system dynamics.

More generally, a macrostate may be composed of microstates with different energies (i.e. different probabilities). Then the entropy of a macrostate is given by:

$$S = -k_B \sum_{n=1}^{N} p_n \log p_n. \tag{2.8}$$

where p_n is the probability of microstate n. This formulation works even if the microstates have different energies. But for our purposes, it is usually assumed that each microstate in the macrostate has the same energy. Note that if all N states have the same probability, then the above formula reduces to our first definition $S = k_B \log \Omega$.

2.2.3 Free energy

Suppose a macrostate X is defined by a set of microstates $\{x_1, x_2, \ldots, x_N\}$, where the energy of each microstate is constant: $E(x_i) = E_0$. Now we ask what is the probability that the system state is any of the microstates in X? Or, put another way, what is the probability that the system belongs to macrostate X? The probability of any microstate is proportional to $\exp \frac{-E_0}{k_B T}$. Since the microstates are disjoint, the probabilities are additive, therefore:

$$p(X) = \frac{1}{Z} \sum_{n=1}^{N} \exp \frac{-E_0}{k_B T}, \tag{2.9}$$

where Z is the partition function for the set of system microstates. Since the energy of each microstate is constant, we can rewrite the sum as:

$$p(X) \;=\; \frac{N}{Z} \exp \frac{-E_0}{k_B T} \;=\; \frac{1}{Z} \exp \frac{-(E_0 - k_B T \log N)}{k_B T}. \tag{2.10}$$

Since N is the number of microstates in S, we see that the energy term in the numerator has been replaced with the quantity $E_0 - TS(X)$, where $S(X)$ is the entropy of macrostate X. This leads to the definition of free energy for a macrostate X where each microstate has the same energy E_0:

$$F(X) := E_0 - TS(X). \tag{2.11}$$

Thus, the free energy $F(X)$ can be interpreted as the logarithm of the probability that the system state belongs to the macrostate X.

More generally, the free energy of a macrostate is the expected energy of the system conditioned on the state belonging to a macrostate X.

2.2.3.1 Free energy example: isomerization

Suppose a system consists of N molecules that take on one of two discrete conformational states, i.e. $x_n \in \{\mathtt{A}, \mathtt{B}\}$. Let $E_\mathtt{A}$ and $E_\mathtt{B}$ be the energies associated with a molecule is state A and B, respectively. The set of possible microstate microstates is given by the set $\{\vec{x} \in \{\mathtt{A}, \mathtt{B}\}^N\}$. The macrostate k is the subset of microstates where exactly k molecules have conformation B. Recalling that the energy of a system of particles is given by the sum of the energy of individual molecules, the free energy of macrostate k is given by:

$$\begin{aligned} F(k) &= (N-k)E_\mathtt{A} + kE_\mathtt{B} - TS_k = (N-k)E_\mathtt{A} + kE_\mathtt{B} - k_BT\log\Omega_k \\ &= (N-k)E_\mathtt{A} + kE_\mathtt{B} - k_BT\log\binom{N}{k}. \end{aligned} \tag{2.12}$$

So we see that the probability of the system being in macrostate k depends on both the energy of isomerization and the entropy of the system.

2.2.3.2 Difference in free energy is all that matters

Suppose macrostates X_1, X_2, …, X_N partition the microstates of a system. Let F_1, F_2, …, F_N be the free energy of the macrostates. We will attempt to show only differences in free energy matter, since shifting the free energy terms by a constant factor has no effect on the probabilities of the macrostates. Suppose the free energies are shifted by a constant value C, then:

$$\begin{aligned} p(X_i|C) &= \frac{\exp\frac{-(F_i+C)}{k_BT}}{\sum_{n=1}^N \exp\frac{-(F_n+C)}{k_BT}} = \frac{\exp\frac{-C}{k_BT}\exp\frac{-F_i}{k_BT}}{\exp\frac{-C}{k_BT}\sum_{n=1}^N \exp\frac{-F_n}{k_BT}} \\ &= \frac{\exp\frac{-F_i}{k_BT}}{\sum_{n=1}^N \exp\frac{-F_n}{k_BT}} = p(X_i|C=0). \end{aligned} \tag{2.13}$$

So the probabilities of the macrostates are not altered by a constant shift in free energy. Consequently, we are free to choose a reference state and arbitrarily set the free energy to zero.

Consider the previous example, if the reference state is $S_{ref} = (N_0, k_0) = (1, 0)$, then the relative free energy of $S = (N, k)$ is:

$$\begin{aligned} \Delta F(N,k) :=&\ F(N,k) - F(1,0) \\ =&\ \left((N-k)E_{\mathtt{A}} + kE_{\mathtt{B}} - k_BT\log\binom{N}{k}\right) - \left(E_{\mathtt{A}} - k_BT\log\binom{1}{1}\right) \\ =&\ (N-k)E_{\mathtt{A}} + (k-1)E_{\mathtt{B}} - k_BT\log\binom{N}{k}. \end{aligned} \tag{2.14}$$

Since the reference is arbitrary, we can choose $E_{\mathtt{A}}$ such that $F(1,0) = 0$. Then $\Delta F(N,k) = F(N,k)$.

The difference in free energy of the macrostate states (N,k) and $(N,k+1)$ is

$$\begin{aligned} \Delta F_{k,k+1} :=&\ F(N,k+1) - F(N,k) \\ =&\ \left((N-k-1)E_{\mathtt{A}} + (k+1)E_{\mathtt{B}} - k_BT\log\binom{N}{k+1}\right) \\ &\qquad - \left((N-k)E_{\mathtt{A}} + kE_{\mathtt{B}} - k_BT\log\binom{N}{k}\right) \\ =&\ E_{\mathtt{B}} - E_{\mathtt{A}} - k_BT\log\left(\frac{k!(N-k)!}{(k+1)!(N-k-1)!}\right) \\ =&\ \left(E_{\mathtt{B}} - k_BT\log(N-k)\right) - \left(E_{\mathtt{A}} - Tk_B\log(k+1)\right). \end{aligned} \tag{2.15}$$

2.2.3.3 Free energy of formation In the previous section we derived a formula for the change in free energy when one molecule changes configuration in a system of N particles. The formula can be divided into two terms, one for the "reactant", i.e. the molecule in configuration `A` prior to the change, and one for the "product", i.e. the molecule in configuration `B` subsequent to the change. Each term can be interpreted as the free energy cost of one molecule in that configuration, adjusted for the current quantity of molecules in that configuration. This motivates the concept of *free energy of formation*, which is defined to be:

$$\Delta F_f(X) \ :=\ \Delta F_f^{\circ}(X) \ +\ k_BT\log(n_X/n_0), \tag{2.16}$$

where $\Delta F_f^{\circ}(X)$ is the *standard free energy of formation* constant for a species X, n_X is the number of particles of species X, and n_0 is a reference quantity for the standard free energy of formation.

2.2.4 Gibbs free energy

An NPT system has a fixed number of particles, constant pressure, and constant temperature. These conditions often apply to biochemical or cell culture experiments, which are typically performed in a test tube under atmospheric pressure and room temperature. In an NPT system, free energy takes the form:

$$\Delta G := \Delta H - T\Delta S, \tag{2.17}$$

where S is entropy as before, and H is enthalpy, which is analogous to energy for an NPT system.

The Gibbs free energy of formation of a species X is defined to be:

$$\Delta G_f(X) \; := \; \Delta G_f^\circ(X) + RT\log([X]/c_0) \; = \; \Delta H_f^\circ(X) - RT\Delta S_f^\circ(X) + RT\log([X]/c_0), \tag{2.18}$$

where $\Delta H_f^\circ(X)$ is the standard heat of formation, $\Delta S_f^\circ(X)$ is the standard entropy, $[X]$ is the concentration of X, and c_0 is a reference concentration. The free energy of formation can be interpreted as the free energy cost of constructing one unit of species X from atomic constituents (i.e. the reference) in an NPT system with concentration $[X]$.

The Gibbs' free energy is perhaps the most important concept in chemical physics: an NPT system tends towards minimizing the Gibbs' free energy. Recalling that free energy is related to the probability of a state, there is an intuitive connection between free energy and equilibrium. The system prefers to reside in high probability states. Since the free energy is proportional to the negative of the logarithm of probability, it makes sense that the system would seek out states with minimum free energy.

2.2.4.1 Reaction equilibrium The change in free energy due to a reaction is the vital piece of information required to determine whether a reaction will proceed (on average) in the forward or reverse direction. If the change is negative, then the reaction proceeds forward. When the change is zero, the reaction is at *equilibrium* and has no tendency to move forward or reverse. A negative change in free energy can be viewed as a "flow" towards higher probability states. When the change in free energy is zero, then the reaction has reached a local maximum in the probability distribution.

The change in free energy of a reaction can be calculated by computing the difference between the free energy of formation of the products and reactants:

$$\Delta G_{\texttt{rxn}} \;=\; \sum_p \nu_p \left(\Delta G_f^\circ(p) + RT\log([p]/c_0)\right) - \sum_r \nu_r \left(\Delta G_f^\circ(r) + RT\log([r]/c_0)\right), \quad (2.19)$$

where p ranges over the products, r ranges over the reactants, and the ν's are stoichiometric coefficients.

When reactants and products are at standard concentration c_0, the change in free energy reduces to:

$$\Delta G^\circ_{\texttt{rxn}} = \sum_p \nu_p \Delta G_f^\circ(p) - \sum_r \nu_r \Delta G_f^\circ(r). \quad (2.20)$$

This quantity is called the *standard free energy change* of the the reaction. Equation 2.19 can be reformulated using the standard free energy change:

$$\begin{aligned}\Delta G_{\texttt{rxn}} &= \Delta G^\circ_{\texttt{rxn}} + \sum_p \nu_p RT\log([p]/c_0) - \sum_r \nu_r RT\log([r]/c_0)\\ &= \Delta G^\circ_{\texttt{rxn}} + RT\log\frac{\prod_p([p]/c_0)^{\nu_p}}{\prod_r([r]/c_0)^{\nu_r}}. \end{aligned} \quad (2.21)$$

So we see that the change in free energy is a constant and a term that accounts for non-standard concentrations of reactants and products. The *reaction quotient* is the ratio of product and reactant concentrations inside the logarithm:

$$Q := \frac{\prod_p([p]/c_0)^{\nu_p}}{\prod_r([r]/c_0)^{\nu_r}}. \quad (2.22)$$

The reaction is at equilibrium when the change in free energy is zero. Setting the free energy change to zero, the relationship between equilibrium and Q is

$$\Delta G_{\texttt{rxn}} = 0 \iff \Delta G^{\circ}_{\texttt{rxn}} = -RT \log Q \iff \exp \frac{-\Delta G^{\circ}_{\texttt{rxn}}}{RT} = Q. \tag{2.23}$$

So the reaction quotient at equilibrium is related to the standard change in free energy via the Boltzmann factor. The value of Q at equilibrium is usually denoted by K_{eq}, the equilibrium constant.

$$\exp \frac{-\Delta G^{\circ}_{\texttt{rxn}}}{RT} = K_{eq}. \tag{2.24}$$

2.2.4.2 Detailed balance At equilibrium, the net flux across each reaction is zero in an NPT system. This is the principle of *detailed balance*: the forward rate of each reaction is balanced with the reverse rate. We will see in this section that satisfying detailed balance introduces constraints among the equilibrium constants in the reaction network. And as a further consequence, the kinetics parameters are constrained.

Consider a "loop" of reactions such that the system state after firing the sequence is the same as the original state. Suppose r_1, r_2, $\ldots$, r_N are a sequence of reactions in an NPT system, and let $\vec{s}_i$ be the stoichiometry vector for reaction i. If a sequence of reactions has the property:

$$\sum_{i=1}^{N} \vec{s}_i = \vec{0}, \tag{2.25}$$

then the reaction sequence forms a *loop* in the reaction network. If a reaction sequence forms a loop in the reaction network, then we will see that the change in Gibbs' free energy due to firing the reactions is zero.

$$\sum_{i=1}^{N} \vec{s}_i = \vec{0} \implies \sum_{i=1}^{N} \Delta G^{\circ}_{\texttt{rxn}}(i) = 0, \tag{2.26}$$

where $\Delta G^{\circ}_{\texttt{rxn}}(i)$ is the standard free energy change of reaction i. In words, Gibbs free energy is conserved around loops in the reaction network.

Equation 2.26 is a simple consequence of the relationship between the standard change in free energy and the reaction quotient at equilibrium. For precision, let's define the reactant vector $\vec{r}_i$, where the $r_{i,j}$ is the number of species j consumed by reaction i, and the product vector $\vec{p_i}$, where $p_{i,j}$ is the number of species j produced by reaction i. The reactant and product vectors are related to the stoichiometry vector: $\vec{s_i} = \vec{p_i} - \vec{r_i}$. Now let's consider the product of the equilibrium constants:

$$\begin{aligned}\prod_{i=1}^{N} K_{eq,i} &= \prod_{i=1}^{N} \frac{\prod_{j=1}^{N_s}([x_j]/c_0)^{p_{i,j}}}{\prod_j([x_j]/c_0)^{r_{i,j}}} = \prod_{i=1}^{N}\prod_{j=1}^{N_s}([x_j]/c_0)^{p_{i,j}-r_{i,j}} = \prod_{i=1}^{N}\prod_{j=1}^{N_s}([x_j]/c_0)^{s_{i,j}} \\ &= \prod_{j=1}^{N_s}([x_j]/c_0)^{\sum_{i=1}^{N} s_{i,j}} = \prod_{j=1}^{N_s}([x_j]/c_0)^0 = 1, \qquad (2.27)\end{aligned}$$

where N_s is the number of species involved in the reaction loop. So the product of equilibrium constants around a loop is equal to one. Now let's use the relation between equilibrium constants and standard change in free energy:

$$1 = \prod_{i=1}^{N} K_{eq,i} = \prod_{i=1}^{N} \exp\frac{-\Delta G^\circ_{\texttt{rxn}}(i)}{RT} \iff 0 = \sum_{i=1}^{N} \Delta G^\circ_{\texttt{rxn}}(i). \qquad (2.28)$$

Therefore, the sum of standard change in free energy is zero around any loop in the reaction network. It also follows that the change in free energy for non-standard concentrations equals zero. Starting with the change Gibbs' free energy calculated from the free energy of formation, we obtain:

$$\begin{aligned}\sum_{i=1}^{N} \Delta G_{\texttt{rxn}}(i) &= \sum_{i=1}^{N}\sum_{j=1}^{N_s} \vec{s}_{i,j}\left(\Delta G^\circ_f(j) + RT\log([x_j]/c_0)\right) \\ &= \sum_{i=1}^{N}\sum_{j=1}^{N_s} \vec{s}_{i,j}\Delta G^\circ_f(j) + RT\sum_{i=1}^{N}\sum_{j=1}^{N_s} \vec{s}_{i,j}\log([x_j]/c_0) \\ &= \sum_{i=1}^{N} \Delta G^\circ_{\texttt{rxn}}(i) + RT\sum_{j=1}^{N_s}\left(\sum_{i=1}^{N} \vec{s}_{i,j}\right)\log([x_j]/c_0) = 0. \qquad (2.29)\end{aligned}$$

2.2.4.3 Detailed balance in the hemoglobin model The Hemoglobin state transition digram in Figure 2.1 has 4 simple loops. Each simple loop introduces one additional constraint on the equilibrium parameters. Since the oxygen binding sites are independent given the conformation of Hemoglobin (and concentration of free oxygen), we have $K_{\texttt{R01}} = K_{\texttt{R12}} = K_{\texttt{R23}} = K_{\texttt{R34}}$ and $K_{\texttt{T01}} = K_{\texttt{T12}} = K_{\texttt{T23}} = K_{\texttt{T34}}$. So there are $13 - 6 = 7$ equilibrium parameters and an additional 4 constraints due to the loop. Therefore, the model has 3 free equilibrium parameters. Choosing the free parameters $K_{\texttt{R01}}, K_{\texttt{RT1}}, K_{\texttt{T01}}$, the remaining parameters are computed as follow:

$$\begin{aligned}
K_{\texttt{RT1}} &= K_{\texttt{T01}} K_{\texttt{RT0}} / K_{\texttt{R01}} \\
K_{\texttt{RT2}} &= K_{\texttt{T01}}^2 K_{\texttt{RT0}} / K_{\texttt{R01}}^2 \\
K_{\texttt{RT3}} &= K_{\texttt{T01}}^3 K_{\texttt{RT0}} / K_{\texttt{R01}}^3 \\
K_{\texttt{RT4}} &= K_{\texttt{T01}}^4 K_{\texttt{RT0}} / K_{\texttt{R01}}^4 \qquad (2.30)
\end{aligned}$$

2.2.4.4 Elementary kinetics and its connection to detailed balance So far, we have elaborated the theory for systems at equilibrium. But is this relevant for systems that are far away from equilibrium? If most biochemical models include processes that are driven by external energy, e.g. ATP, why should we care about equilibrium principles? The answer lies in the connection between detailed balance and elementary kinetics, the most common theoretical model governing the kinetics of chemical reactions.

An elementary reactions is a chemical reaction in which the reactant species interact to form the products in one step, i.e. there are no intermediate steps in the reaction mechanism. The kinetic rate of an elementary reaction is given by the product of (i) a *microscopic* rate constant, (ii) a combinatorial factor that counts the number potential reactant sets, and (iii) a path factor describing the number of "reaction pathways" that a set of reactants may follow. In general, if $\vec{r}$ is the reactant vector, the elementary reaction rate is:

$$\texttt{rate} = k\eta \prod_{i=1}^{N_s} \prod_{j=1}^{r_i} \frac{n_i - (j-1)}{j}, \qquad (2.31)$$

where n_i is the number of molecules of species i, r_i is the reactant stoichiometry of species i, k is the *microscopic* rate constant (per unit time), and η is the path factor.

For example, the bimolecular reaction `A + B -> AB k` has the elementary rate:

$$\texttt{rate} = k n_A n_B, \tag{2.32}$$

where the number of ways to select an A,B pair to react is $n_A n_B$. In contrast, the reaction `A + A -> AA k` has the elementary rate:

$$\texttt{rate} = k\frac{n_A(n_A-1)}{2}, \tag{2.33}$$

since the number of ways to select an A,A pair is given by $\binom{n_A}{2} = \frac{n_A(n_A-1)}{2}$.

Most commonly, bulk rate constants, i.e. per concentration per time for bimolecular reactions, are reported in the literature. However, a rate constant in "microscopic" units, i.e. the propensity for an A,B particle pair to interact per unit time in the reactor volume, is required for discrete stochastic simulations. The microscopic constant is obtained from the bulk rate constant by dividing by the product of Avogadro's number and the reactor volume, $N_A V$.

At equilibrium, the reverse rate balances the forward rate so there is no net flux through the reaction. Consider the generalized elementary reaction:

$$\sum_{i=1}^{N_s} \texttt{x}_\texttt{i}{}^{r_i} \;\rightarrow\; \sum_{j=1}^{N_s} \texttt{x}_\texttt{j}{}^{p_j} \quad k_+, k_- \tag{2.34}$$

where $\vec{r}$ and $\vec{p}$ are the reactant and product vectors. At equilibrium the forward and reverse rates are balanced:

$$k_+\eta_+ \prod_{i=1}^{N_s} \frac{\prod_{j=1}^{r_i} n_i - (j-1)}{r_i!} \;=\; k_-\eta_- \prod_{i=1}^{N_s} \frac{\prod_{j=1}^{p_i} n_i - (j-1)}{p_i!}. \tag{2.35}$$

Rearranging terms, we obtain:

$$\frac{k_+\eta_+ \prod_i r_i!}{k_-\eta_- \prod_i p_i!} \;=\; \frac{\prod_i \prod_{j=1}^{p_i} n_i - (j-1)}{\prod_i \prod_{j=1}^{r_i} n_i - (j-1)}. \tag{2.36}$$

For large species counts, the ratio of weighted rate constants is approximately proportional to the reactant quotient:

$$(N_A V)^{\sum_i (r_i - p_i)} \frac{k_+\eta_+ \prod_i r_i!}{k_-\eta_- \prod_i p_i!} \;=\; (N_A V)^{\sum_i (r_i - p_i)} \frac{\prod_i {n_i}^{p_i}}{\prod_i {n_i}^{r_i}} \;=\; \frac{\prod_i \left(\frac{n_i}{N_A V}\right)^{p_i}}{\prod_i \left(\frac{n_i}{N_A V}\right)^{r_i}} \;=\; Q. \tag{2.37}$$

Hence, at equilibrium we have:

$$K_{eq} = (N_A V)^{\sum_i (r_i - p_i)} \frac{k_+ \eta_+ \prod_i r_i!}{k_- \eta_- \prod_i p_i!}. \tag{2.38}$$

Since the path factor is a constant, the equilibrium constant constrains the ratio of the elementary kinetic rate constants. Since the product of equilibrium constants around a loop is one, and the ratio of rate constants is proportional to the equilibrium constant, we have:

$$1 = \prod_{i=1}^{N} K_{eq,i} = \prod_{i=1}^{N} (N_A V)^{\sum_j (r_j - p_j)} \frac{k_{i+} \eta_{i+} \prod_j r_{i,j}!}{k_{i-} \eta_{i-} \prod_j p_{i,j}!} = \prod_{i=1}^{N} \frac{k_{i+} \eta_{i+} \prod_j r_{i,j}!}{k_{i-} \eta_{i-} \prod_j p_{i,j}!}. \tag{2.39}$$

So even when the system is away from the equilibrium, the elementary rates are constrained by principles of free energy.

Finally, if the change in free energy around each loop is zero, then detailed balance is automatically satisfied. For simplicity, absorb the multiplicity and combinatorial factors into the rate constant. Then for any reaction i in a loop:

$$\begin{aligned} \frac{k'_{i-}}{k'_{i+}} = \frac{1}{K_{eq,i}} = \prod_{j \neq i} K_{eq,j} = \prod_{j \neq i} \frac{k'_{j+}}{k'_{j-}} = \prod_{j \neq i} \frac{\prod_l {n_l}^{p_{j,l}}}{\prod_l {n_l}^{r_{j,l}}} = \frac{\prod_l {n_l}^{\sum_{j \neq i} p_{j,l}}}{\prod_l {n_l}^{\sum_{j \neq i} r_{j,l}}} = \frac{\prod_l {n_l}^{r_{i,l}}}{\prod_l {n_l}^{p_{i,l}}} \\ \Longrightarrow \; k'_{i+} \prod_l {n_l}^{r_{i,l}} = k'_{-} \prod_l {n_l}^{p_{i,l}}. \end{aligned} \tag{2.40}$$

2.2.4.5 Methods for satisfying detailed balance Various methods exist for satisfying detailed balance in reaction networks. Colquhoun et al. [96] presented a method that relies on finding the minimum cycle basis of the state graph. The computational cost is $> O((n+m)^3)$, where n is the number of undirected edges and m the number vertices [97]. Thus the cost is prohibitive for large reaction networks found in combinatorial networks. Yang et al. [97] reduced the cost by identifying the *minimal reaction scheme*, i.e. the sub-network excluding energy-driven and irreversible reactions, as well as reactions in acyclic pathways. Szederkenyi and Hangos developed a polynomial-time method posed as a linear programming task [98]. Gorban and Yablonsky extended detailed balance methods to networks with irreversible reactions. Danos et al. [99], working in a restricted class of Petri net, provided criterion for verification of detailed balance in a class of Petri nets.

The methods above attempt to satisfy or verify detailed balance in a reaction network. In contrast, the Allosteric Network Compiler (ANC) is a rule-based modeling language with semantics for allosteric mechanisms [49]. During network generation, kinetic rates are customized for each reaction to account for cooperative interactions. Thus, the reactions derived from a rule are guaranteed to satisfy detailed balance.

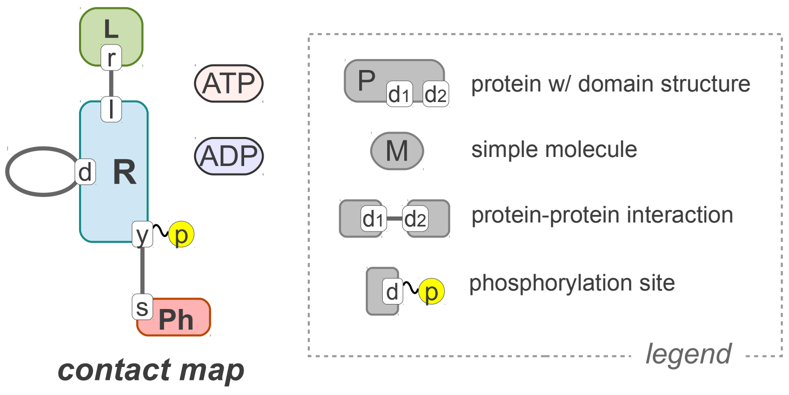
2.3 MOTIVATING EXAMPLE

This chapter will develop energy-based modeling in parallel with a model of ligand-induced receptor dimerization and phosphorylation. Henceforth, we will refer to this as the *receptor activation* model. Figure 2.2 shows the *contact map* for the receptor activation model. A contact maps shows the molecular domain structure, protein-protein interactions, and potential post-translational modifications. The receptor activation pathway begins with ligand binding the extracellular domain of the transmembrane receptor. Ligand binding drives the receptors to form dimers, and the kinase domain of a dimerized receptor trans-phosphorylates a residue on its partner receptor. Phosphorylation consumes one ATP, however we assume that the concentration of ATP is held constant in the cytosol. (The receptor is said to be *active* when it is phosphorylated.) Finally, a cytosolic phosphotase binds to the phosphorylated receptor at catalyzes dephosphorylation, returning the receptor to its inactive form. The ligand-receptor module is based on the Macdonald-Pike model of cooperativity in EGF ligand binding and receptor dimerization [88]. The complete model is listed in Appendix C.

The receptor activation model will motivate the methods, illustrating the benefits in terms of simplified model specification and automatic satisfaction of thermodynamic balance constraints. The receptor activation EBM is specified by 5 reaction rules (ligand binding, receptor dimerization, transphosphorylation, phosphotase binding, and dephosphorylation) and 8 energy patterns. The reaction rules have *minimal context*, meaning that only the sites of the reaction center are queried by the rules. The cooperativity between sites, such as the cooperativity of ligand binding and receptor dimerization, is encoded in the energy patterns.

In contrast, an equivalent plain BNGL model requires 10 reaction rules (not shown).

Figure 2.2: Contact map of example energy model: receptor activation.



Not only does the EBM require fewer rules, but each rule has minimal context and thus is simpler to write and read. Furthermore, 3 thermodynamic balance constraints are implied by the reaction rules. Whereas the modeler must manage the constraints manually in the plain BNGL model, these are handled automatically in the energy BNGL model. Managing thermodynamics constraints is especially challenging for larger models. The reaction network generated by the energy model, or the equivalent plain BNGL model, includes 50 species and 316 reactions.

2.4 THE THEORY OF ENERGY-BASED MODELING

In this section, I will develop the theory of energy-based modeling. First, we present a method of accounting free energy of species by pattern matching. Then the change in free energy of reactions are computed from the free energy of the reactant and product species. Next, kinetic parameters are derived from the change in free energy and an activation energy parameter. Finally, a software implementation in BioNetGen, *energy BNGL* or *eBNGL* is described.

The union of free energy principles and rule-based models was first described by Ollivier, et al. [49] and implemented in the Allosteric Network Compiler (ANC). Ollivier's method was generalized to arbitrary sources of cooperativity via pattern matching by Danos et al. [93]. The present work builds on this foundation by presenting (i) a general implementation of energy modeling in BNG, (ii) fully energy-based kinetics, including pure catalysis, (iii) non-equilibrium processes, (iv) a treatment of ring formation in oligomers.

2.4.1 Free energy accounting: a pattern-based approach

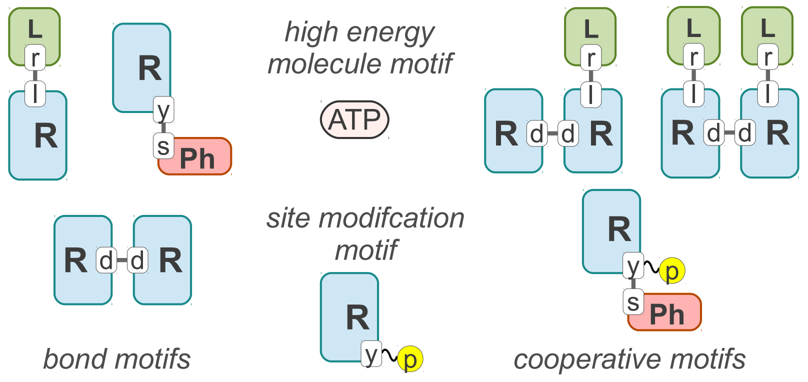
Energy patterns are molecular motifs where free energy is "stored". For example, free energy can be stored in a bond, `L(r!1).R(l!1)`; a molecule, `ATP()`; or a post-translational site modification, `R(y~P!?)`; and so forth. Free energy can also be stored in cooperativity between pairs of bonds, `L(r!1).R(l!1,d!2).R(d!2)`; bonds and site modifications, `R(y~P!1).Ph(s!1)`; etc. In general, free energy can be stored in arbitrary patterns. Each energy pattern is associated with a parameter called the *standard free energy of pattern formation*, or *pattern energy* for brevity. This parameter quantifies the amount of free energy required to form 1 unit of the pattern under standard conditions.

Consider a simple system with molecule types `A(b)` and `B(a)`. Suppose free energy is stored in the bond motif `A(b!1).B(a!1)`, with the pattern energy $\Delta G_f^\circ = -45\texttt{kJ/mol}$. If we begin with 1 molar `A(b)`, `B(a)` and `A(b!1).B(a!1)`, then the free energy cost of forming `A(b!1).B(a!1)` is -45 kJ/mol.

2.4.1.1 Computing species free energy

The *standard free energy of species formation*, or *species energy* for brevity, quantifies the free energy cost of forming the species from "atomic" constituents under standard conditions. Most often, our atomic components are not physical atoms, but rather the smallest unit of resolution in the model. In most cases, the "atoms" will be individual, unbound and unmodified macromolecules or other simple molecules like ATP. The species energy is a function of the energy patterns embedded in the molecular structure of the species. The energy is computed by counting the number of energy pattern matches in the species then computing the sum over matches weighted by the corresponding pattern energy. For example, if $\Delta G_f^\circ(\texttt{bond})$ is the pattern energy associated

Figure 2.3: Energy patterns in example model: receptor activation model.



with a type of bond, and the bond pattern is found 3 times in the species, then the pattern contributes $3 \cdot \Delta G_f^\circ(\texttt{bond})$ units of free energy to the species.

More formally, let $(P_n, \Delta G_f^\circ(n))_{n=1}^{N_P}$ be an indexed set of BNGL patterns paired with pattern energy parameters. Let x be a species and let $\texttt{emb}(P, x)$ be the set of embeddings of pattern P in the graph representing species x (see Appendix B for more details on graph formalism). Then the standard free energy of species formation is given by:

$$G_f^\circ(x) = \sum_{n=1}^{N} |\texttt{emb}(P_n, x)| \cdot \Delta G_f^\circ(n), \tag{2.41}$$

where $|\texttt{emb}(P_n, x)|$ is the number of elements in the set of embeddings, i.e. the number time the motif matches the species.

Figure 2.4 illustrates the process of computing species energy from pattern matches. In this example, the standard free energy of species formation is computed for a phosphorylated receptor with bound phosphotase. Three energy patterns are embedded in this species: the phosphorylated receptor motif (RyP), the receptor-phosphotase motif (R_Ph), and phosphorylated receptor-phosphotase motif (Rp_Ph). Since each energy pattern matches the species exactly once, the standard free energy of formation is computed by the sum of the pattern energy parameters weighted by unity: $G_f^\circ(\texttt{Rp}) + G_f^\circ(\texttt{R\_Ph}) + G_f^\circ(\texttt{RyP\_Ph})$.

Figure 2.4: Species free energy of formation: an illustration.

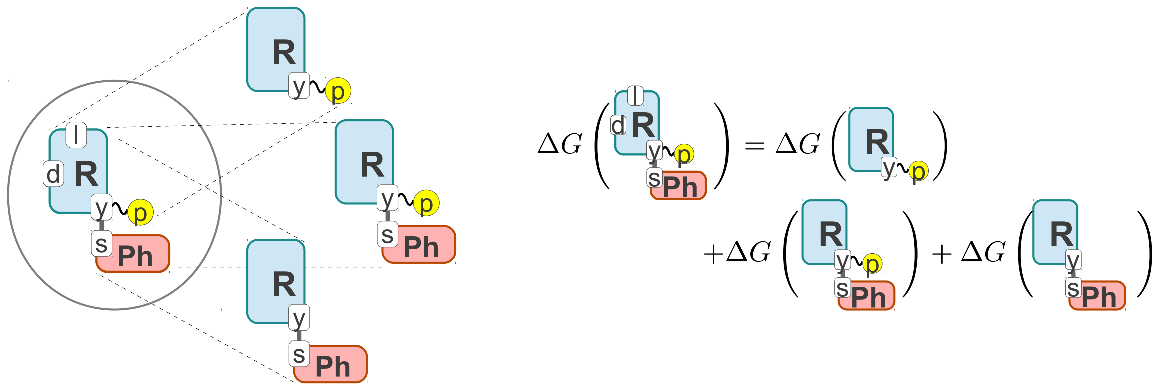


Table 2.1 lists energy patterns defined in the receptor activation model. There are several types of motifs represented in the model. Free energy is stored in three bond motifs: the ligand-receptor bond, receptor dimerization bonds, and the interaction of the phosphotase (`Ph`) with the receptor tyrosine residue. Additionally, energy is stored in the phosphorylated receptor tyrosine residue and ATP molecules. Finally, free energy is stored in cooperativity among these motifs: association of a ligand to a receptor dimer, the association of two ligands to a receptor dimer, and doubly ligated receptor dimers. Figure 2.3 illustrates the energy patterns contained in the model.

Each energy pattern is associated with a non-dimensional *standard free energy of pattern formation* parameter. If ΔG_f° has units of kJ/mol, the unitless parameter is obtained by dividing by the product of temperature (K) and the Universal Gas Constant (kJ/mol/K). However, if ΔG_f° has units of kJ/molecule, the Boltzmann constant k_B (kJ/K) substitutes for the Universal Gas Constant.

The reference free energy for the system is the ensemble of "atomized" proteins in the unbound, unmodified state (`L(r)`, `R(l,y 0)`, `Ph(y)`) and the `ADP()` molecule. The free energy of formation of a species is the difference in energy of the species (as computed from energy patterns) and its atomic components in the reference state. The `ATP()` free energy is with respect to the free energy of `ADP()`.

Table 2.1: Energy patterns block for receptor activation model.

```
begin energy patterns
   # bond energy
   L(r!1).R(l!1)    G_LR/RT
   R(d!1).R(d!1)    G_RR/RT
   R(y!1).Ph(y!1)   G_RPh/RT
   # state energy
   R(y~P!?)         G_RyP/RT
   # molecule energy
   ATP()            G_ATP/RT
   # cooperativity
   R(y~P!1).Ph(y!1)                      G_RyP_Ph/RT
   L(r!1).R(l!1,d!2).R(d!2)              G_LRR/RT
   L(r!1).R(l!1,d!2).R(l!3,d!2).L(r!3)   G_LRRL/RT
end energy patterns
```

At minimum, an energy model should define energy patterns for bonds and site modifications (e.g. conformational change, post-translational site modification, etc.). Energy patterns are not required for sites in the unbound, unmodified state since this is assumed to be the reference. If an energy pattern is not defined for a bond or site modification, it is assumed that the change in free energy is zero with respect to the reference. Usually it is not necessary to associate an energy pattern with the molecule itself, unless it is consumed or produced during the simulation. For example, free energy can associated with the high-energy molecule ATP, which is consumed by many enzymatic reactions. (We will later resolve the dangling issue of non-equilibrium processes such as those that consume ATP.) In the receptor activation model, `ADP()` does not require an energy pattern since the free energy of `ATP()` is defined relative to `ADP()`.

2.4.1.2 Computing change in free energy due to reaction The change in free energy due to a reaction is computed from the difference in free energy of the product and reactant species. If $\vec{r}_\mu$, $\vec{p}_\mu$ are the reactant and product vectors for a reaction μ, then the standard change in free energy is given by the difference in the standard free energy of formation of the reactants and products (see Equation 2.20):

$$\Delta G_f^\circ(\mu) \;=\; \sum_{i=1}^{N_s} p_{\mu,i}\Delta G_f^\circ(x_i) - \sum_{i=1}^{N_s} r_{\mu,i}\Delta G_f^\circ(x_i) \;=\; \sum_{i=1}^{N_s} s_{\mu,i} G_f^\circ(x_i). \tag{2.42}$$

where i ranges over species, x_i is species i, and $\vec{s}_\mu$ is the stoichiometry vector.

The free energy change can be reformulated in terms of the change in the pattern matches. Plugging in Equation 2.41 for $\Delta G_f^\circ(x_i)$ yields:

$$\Delta G_f^\circ(\mu) \;=\; \sum_{i=1}^{N_s} s_{\mu,i} \sum_{j=1}^{N_p} |\texttt{emb}(P_j, x_i)| \cdot G_f^\circ(P_j) \;=\; \sum_{j=1}^{N_p} \sigma_{\mu,j} G_f^\circ(P_j), \tag{2.43}$$

where $\sigma_{\mu,j} = \sum_{i=1}^{N_s} s_{\mu,i}|\texttt{emb}(P_j, x_i)|$ is the *pattern stoichiometry* of reaction μ. In words, the standard change in free energy is a linear combination of pattern energies with coefficients given by the pattern stoichiometry of the rule.

2.4.1.3 Free energy is conserved around loops in the reaction network Now we will show that the change in free energy around a loop in the reaction network is zero. Suppose $\mu_1, \mu_2, \ldots, \mu_{N_r}$ is a sequence of reactions such that the final system state is the same as the initial. More precisely, if $\vec{s}_n$ is the stoichiometry vector of reaction μ_n, and $\sum_{n=1}^{N} \vec{s}_n = \vec{0}$, then it follows that the standard change in free energy around the loop is zero:

$$\begin{aligned}\sum_{n=1}^{N_r} \Delta G_f^\circ(\mu_n) \;&=\; \sum_{n=1}^{N_r}\sum_{i=1}^{N_s} s_{n,i} \sum_{j=1}^{N_p} |\texttt{emb}(P_j, x_i)| \cdot G_f^\circ(P_j) \\ &=\; \sum_{i=1}^{N_s}\left(\sum_{n=1}^{N_r} s_{n,i}\right) \sum_{j=1}^{N_p} |\texttt{emb}(P_j, x_i)| \cdot G_f^\circ(P_j) \;=\; 0. \end{aligned} \tag{2.44}$$

Therefore, the pattern-based accounting system guarantees that all loops in the reaction network conserve free energy. In a later section we will derive kinetic parameters that are compatible with the change in free energy and show that this procedure guarantees detailed balance.

2.4.2 Energy-based kinetics

The Arrhenius theory of reaction rates postulates that a rate constant has the form

$$k = C \exp \frac{-E_A}{RT}, \tag{2.45}$$

where E_A is the *activation energy* and C is a constant. The theory assumes that a high energy *transition state* separates the reactant state from the product state. Assuming quasi-equilibrium between the reactants and the transition state, the reactants occupy the transition state with probability

$$\begin{aligned} p(\texttt{trans}) &= \frac{\exp\left(-(\Delta G_f(\texttt{reac}) + E_A)/RT\right)}{\exp\left(-\Delta G_f(\texttt{reac})/RT\right) + \exp\left(-(\Delta G_f(\texttt{reac}) + E_A)/RT\right)} \\ &= \frac{\exp(-E_A/RT)}{1 + \exp(-E_A/RT)}. \end{aligned} \tag{2.46}$$

Assuming that the fraction of time spent in the transition state is small, this is approximately:

$$p(\texttt{trans}) \approx \exp \frac{-E_A}{RT}. \tag{2.47}$$

Similar to the Michaelis-Menten equation, we now assume that the rate of reaction is proportional to the fraction of time spent in the transition state. This yields the Arrhenius rate in Equation 2.45.

Linear transition state theory [100], or linear TST, assumes that the activation energy is linearly related to the standard change in free energy of a reaction. This assumption leads an activation energy the following form:

$$E_A := E_0 + \phi \Delta G^\circ_{\texttt{rxn}}, \tag{2.48}$$

where E_0 and ϕ are free parameters. The parameters may be treated as global or local to a subset of reactions, depending on the scope of the linear assumption. Ollivier et al. adopted linear TST for the Allosteric Network Compiler. The scope of ϕ as a global or local parameter was explored, and it was observed that a global ϕ was sufficient for test models and, as an added benefit, reduced the likelihood of overfitting data. ANC does not explicitly define the E_0, rather this is inferred for each rule from a pair of baseline kinetic parameters,

k_+, k_-. Although linear TST restricts the kinetic space, the assumption has been shown to be adequate for a number of applications, e.g, [101].

The activation energy is typically set with respect to the forward reactions, i.e. the difference between the transition state energy and the free energy of the reactants. The activation energy for the reverse reaction is related to the forward reaction as follows:

$$E_A^- \;=\; E_A^+ + \Delta G^\circ_{\mathtt{rxn}} = E_0 + (\phi - 1)\Delta G^\circ_{\mathtt{rxn}}. \tag{2.49}$$

Plugging the formula for E_A^+ and E_A^- into the Arrhenius rate law, we find that the ratio of forward and reverse rate constants is equal to the Equilibrium constant, as required by equilibrium principles:

$$\frac{k_+}{k_-} = \frac{C \exp\left(-(E_0 + \phi\Delta G^\circ_{\mathtt{rxn}})/RT\right)}{C \exp\left(-(E_0 + (\phi - 1)\Delta G^\circ_{\mathtt{rxn}})/RT\right)} = \exp\frac{-\Delta G^\circ_{\mathtt{rxn}}}{RT}. \tag{2.50}$$

Therefore, we find it appropriate to adopt a convention for computing the kinetic rate constants as follows:

$$\begin{aligned} k_+ &= C \exp\frac{-(E_0 + \phi\Delta G^\circ_{\mathtt{rxn}})}{RT} \\ k_- &= C \exp\frac{-(E_0 + (\phi - 1)\Delta G^\circ_{\mathtt{rxn}})}{RT}, \end{aligned} \tag{2.51}$$

where E_0 and C are rule-specific parameters with scope of all reactions generated by the rule, and ϕ is usually a global system parameter (it may be desirable at times to define a unique ϕ for each reaction rule). Since biochemical systems are usually simulated at constant temperature, we may absorb the constant C into the Boltzmann factor. Letting $E_0' := E_0 - RT\log C$, we can rewrite the kinetic constants:

$$\begin{aligned} k_+ &= \exp\left(-\frac{E_0' + \phi\Delta G^\circ_{\mathtt{rxn}}}{RT}\right) \\ k_- &= \exp\left(-\frac{E_0' + (\phi - 1)\Delta G^\circ_{\mathtt{rxn}}}{RT}\right), \end{aligned} \tag{2.52}$$

Consequently, it is usually sufficient to work with the single parameter $E_0' := E_0 - RT\log C$ rather than E_0 and C individually. We will usually omit the *prime* from E_0' since it usually will not lead to any ambiguity.

The E_0 parameter is interpreted as the activation energy for a reaction where $\Delta G^\circ_{\texttt{rxn}} = 0$. The ϕ parameter is the *rate distribution* parameter. It determines how changes in $\Delta G^\circ_{\texttt{rxn}}$ are distributed between the forward and reverse rates. If $\phi = 1$, then the reverse rate is held constant while the forward rate adjusts to changes in $\Delta G^\circ_{\texttt{rxn}}$. When $\phi = 0$, the forward rate is held constant and only the reverse rate changes. At the special value $\phi = 1/2$, the rate parameters are treated symmetrically, in the sense that $\frac{k'_+}{k_+} = \frac{k_-}{k'_-}$. Furthermore, when $\phi = 1/2$, E_0 can be interpreted as the average of the forward and reverse activation energies:

$$\begin{aligned}\frac{E_A^+ + E_A^-}{2} &= \frac{(E_0 + \phi\Delta G^\circ_{\texttt{rxn}}) + (E_0 + (\phi - 1)\Delta G^\circ_{\texttt{rxn}})}{2} \\ &= \frac{(2E_0 + (2\phi - 1)\Delta G^\circ_{\texttt{rxn}}}{2} = E_0. \end{aligned} \tag{2.53}$$

The role of ϕ is illustrated in Figure 2.5. If we draw a line connecting the free energy of the reactants to the free energy of the reactants, the transition state energy is E_0 energy units above a point on this line set by the parameter ϕ. By adjusting ϕ, the transition state is defined with respect to the reactants, the products, or something in between.

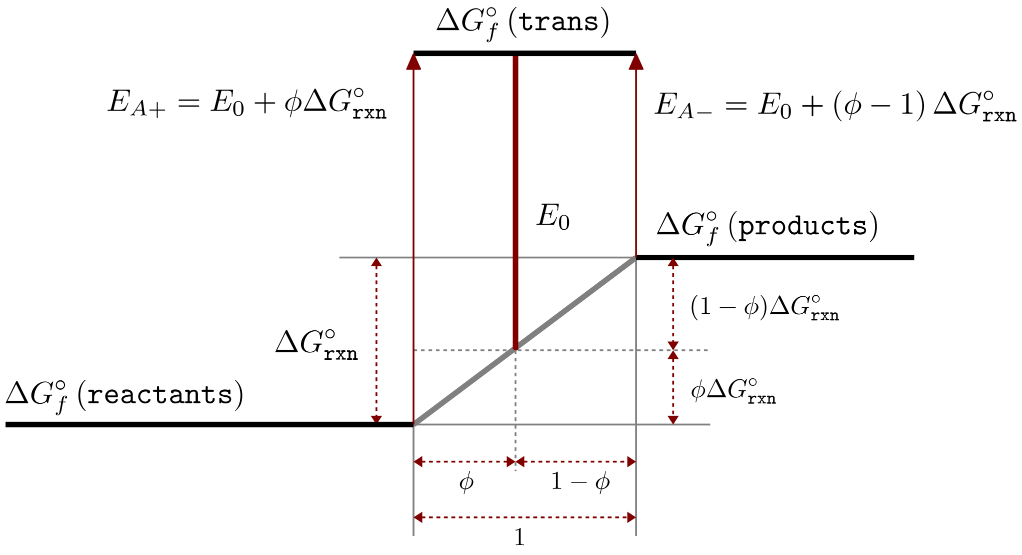
Choosing E_0 for a reaction rule may be non-intuitive for a modeler accustomed to kinetic parameters. It may be more natural to choose a forward rate constant k_{+0} for the reaction rule when no additional context is considered (i.e. compute the change in free energy for the rule, rather than any specific reaction). Letting $\Delta G^\circ_{\texttt{rule}}$ be the standard change in free energy of the rule, the baseline forward activation energy is $E_A^{+0} = E_0 + \phi\Delta G^\circ_{\text{rule}}$. Thus, we can choose k_{+0} and ϕ and compute E_0 as follows:

$$E_0 = -RT\log\left(k_{+0}/C\right) - \phi\Delta G^\circ_{\texttt{rule}}. \tag{2.54}$$

2.4.2.1 The limits of linear transition state theory A key limitation of the linear transition state assumption is that reactions with the same change in free energy will always have the same kinetic rate constants:

$$\begin{aligned}&\Delta G^\circ_{\texttt{rxn},i} = \Delta G^\circ_{\texttt{rxn},j} \\ &\implies\ E_A^{+i} = E_0 + \phi\Delta G^\circ_{\texttt{rxn},i} = E_A^{+j} \implies\ k_{+,i} = C\exp -E_A^{+i}/RT = k_{+,j}.\end{aligned} \tag{2.55}$$

Figure 2.5: **Geometric interpretation of transition state energy.** The transition state energy of a reaction is a function of the free energy change due to the reaction and two additional parameters E_0 and ϕ. This figure illustrates a geometric interpretation of the transition state energy. Let the x-axis correspond to a reaction coordinate where 0 is the reactant state and 1 is the product state. Let the y-axis be the free energy coordinate. A line is drawn between the points $\left(0, \Delta G^\circ_f(\texttt{reactants})\right)$ and $\left(1, \Delta G^\circ_f(\texttt{products})\right)$. The ϕ parameter selects a point along this line given by $\left(\phi, \Delta G^\circ_f(\texttt{reactants}) + \phi \Delta G^\circ_{\texttt{rxn}}\right)$. Finally, the E_0 parameter sets the "height" of the transition state at a fixed distance above the ϕ point, specifically: $\Delta G^\circ_f(\texttt{trans}) = \Delta G^\circ_f(\texttt{reactants}) + \phi \Delta G^\circ_{\texttt{rxn}} + E_0$.


$\Delta G^\circ_f\,(\texttt{trans})$
$E_{A+} = E_0 + \phi \Delta G^\circ_{\texttt{rxn}}$
$E_{A-} = E_0 + (\phi - 1)\,\Delta G^\circ_{\texttt{rxn}}$
E_0
$\Delta G^\circ_f\,(\texttt{products})$
$(1-\phi)\Delta G^\circ_{\texttt{rxn}}$
$\Delta G^\circ_{\texttt{rxn}}$
$\phi \Delta G^\circ_{\texttt{rxn}}$
$\Delta G^\circ_f\,(\texttt{reactants})$
ϕ
$1-\phi$
1

Therefore, pure catalytic effects, where the activation energy is lowered without shifting the equilibrium, are not permissible within linear TST. For example, consider the reactions `S(a~0) <-> s(a~P) kp,km` and `S(a~0).E <-> S(a~P).E b*kp,b*km`, where `E` is an enzyme and $b > 1$. Both reactions have the same equilibrium, but the reaction with enzyme occurs at a faster rate. Such reaction pairs, generated by the same rule, are not possible with the linear TST framework.

The Allosteric Network Compiler permits a workaround since the baseline forward and reverse rate constants are both free parameters for each reaction rule. The modeler writes two separate rules, one for the uncatalyzed reaction and one for the enzyme catalyzed reaction, with different baseline kinetic parameters. While serviceable, this approach risks the integrity of detailed balance in the resulting reaction network. The modeler must verify that no violations of detailed balance have been introduced in the reaction network. For a simple model, this will be a straightforward task. But for complex models, it may be quite challenging.

I propose an alternative approach that is energy-balanced by construction and so avoids the possibility of violating detailed balance. Let us generalize the baseline energy constant to a function of the reactants, $E_0(\vec{r})$, for the forward reaction, and a function of the products, $E_0(\vec{p})$, for the reverse reaction. Now the forward and reverse rate constants are computed as follows:

$$\begin{aligned} k_+ &= C \exp\left(-\frac{E_0(\vec{r}) + \phi \Delta G^\circ_{\texttt{rxn}}}{RT}\right) \\ k_- &= C \exp\left(-\frac{E_0(\vec{p}) + (\phi - 1) \Delta G^\circ_{\texttt{rxn}}}{RT}\right). \end{aligned} \tag{2.56}$$

But we immediately encounter a problem. The ratio of rate constants will not necessarily equal the equilibrium constant

$$\frac{k_+}{k_-} = \exp \frac{-\Delta G^\circ_{\texttt{rxn}} + (E_0(\vec{p}) - E_0(\vec{r}))}{RT} = K_{eq} \iff E_0(\vec{r}) = E_0(\vec{p}). \tag{2.57}$$

Consequently, we must restrict E_0 to the class of functions that evaluate to the same value on $\vec{r}$ and $\vec{p}$. Formally, if $\mathbb{N}^{N_s}$ is the set of all species vectors for the model system, and $\vec{s}$ is the stoichiometry vector of the reaction rule, then $E_0 \in \{f : \mathbb{N}^{N_s} \to \mathbb{R} \mid f(\vec{y}) = f(\vec{y} + \vec{s}) \forall \vec{y}\}$.

The set of constant functions is a trivial example of such a function. The set of functions that depends only on the reaction context also satisfies this definition.

Given a reaction rule and a set of pattern-parameter pairs $(P_i, E_i)_{i=1}^{I}$, suppose that none of the P_i overlaps the reaction center. Then E_0 is computed as follows:

$$E_0(\vec{r}) = \sum_{n=1}^{N_s} \sum_{i=1}^{I} E_i \cdot |\mathtt{emb}(P_i, x_n)|. \tag{2.58}$$

Since the patterns do not overlap the reaction center, it follows that $E_0(\vec{r} + \vec{s}) = E_0(\vec{r})$. Therefore, this class of activation energy functions is compatible with the reaction equilibrium constants.

In eBNG, the baseline activation energy E_0 may be set to a constant math expression or any global or local function (i.e. a function of pattern matches). Prior to network generation, functions are inspected for overlap with the reaction center. If possible overlap is detected, network generation is aborted with an error message to the modeler. Thus, the limitations of linear TST are removed without risk of violating detailed balance[1].

See Section 2.5.8.1 for an model system that illustrates local activation energy functions.

2.4.3 Non-equilibrium reactions: adding free energy to the system

While NPT assumptions are applicable to a number of biochemical models, in general this is a very limiting assumption. Most cellular signaling pathways, such as EGFR, involve enzymatic reactions that consume energy in the form or ATP. The concentration of ATP is held approximately constant in the cell by regulatory mechanisms. The production of ATP is driven by the consumption of energy-rich substrates, e.g. glucose, etc., obtained from the extracellular environment. At the scale of the test tube, the system is NPT if we assume reagents are neither added or removed during the experiment. However, it is impractical to model a system that includes the extracellular environment, nutrient uptake, central metabolism, and so forth. Instead, the model is usually built on an implicit assumption that the concentration of ATP, ADP, and so forth are held constant by some external

[1]At time of writing, local and global functions are supported by eBNG, but the check for overlap is not yet implemented.

mechanism. Hence, the system under study is not NPT, since an external free energy supply is continuously pumped into the system.

Is it reasonable then, for models that include an external source of free energy, to enforce the principle of detailed balance? And if so, how is free energy introduced to the system? A simple thought experiment demonstrates that detailed balance principles do apply to such systems, at least under the regime of elementary kinetics. Suppose the cell maintains an ATP/ADP gradient by consuming energy obtained from the environment. Then imagine that the cell finds itself starved of nutrients. Eventually, the cell would consume all its energy reserves and ATP/ADP concentration gradient would be lost. Gradually the cellular environment would tend toward an equilibrium. When equilibrium is achieved, detailed balance must be satisfied. Hence, since the parameters are independent of concentration, the elementary kinetic parameters of the model must satisfy any constraints imposed by detailed balance.

So we have established that, ideally, elementary kinetic parameters should satisfy detailed balance constraints, even for models that receive an external supply of free energy. But how can we model the flow of external free energy into the system? The answer is right before us, we simply need to make the implicit assumptions explicit. Let us suppose that an external energy sources establishes a non-equilibrium ratio of a high-energy molecule, e.g. ATP, and its low energy catabolite, e.g. ADP. Next, write mass-balanced, reversible reaction rules that explicitly consume the high-energy molecule and produce the low-energy molecule. For example, consider a phosphorylation reaction that consumes an ATP molecules. Such a reaction is typically written `R(y~0) <-> R(y~P) kp,km`, with the consumption of ATP implicit. We will make this assumption explicit by writing the mass balanced equation `R(y~0) + ATP + H2O <-> R(y~P) + ADP + Pi kp',km'`. To simplify the example, let us omit the water and phosphate molecules: `R(y~0) + ATP <-> R(y~P) + ADP kp',km'`. The rates of the reaction are given by:

$$\begin{aligned} \text{rate}_+ &= k'_+[\mathtt{R(y} \sim \mathtt{0)}][\mathtt{ATP}] \\ \text{rate}_- &= k'_-[\mathtt{R(y} \sim \mathtt{P)}][\mathtt{ADP}]. \end{aligned} \tag{2.59}$$

The rate constants are related to the non-balanced rate constants by the relationship

$$\begin{aligned} k_+ &= k'_+[\texttt{ATP}] \\ k_- &= k'_-[\texttt{ADP}]. \end{aligned} \tag{2.60}$$

Therefore the true equilibrium constant is related to the *steady-state constant* by:

$$K_{ss} = \frac{k_+}{k_-} = \frac{k'_+[\texttt{ATP}]}{k'_-[\texttt{ADP}]} = \frac{[\texttt{ATP}]}{[\texttt{ADP}]}K_{eq} = rK_{eq}, \tag{2.61}$$

where $r := \frac{[\texttt{ATP}]}{[\texttt{ADP}]}$ is the ratio of ATP to ADP. As long as $r \gg 1$, the quasi-steady state of the reaction is far from equilibrium. But eventually r will decrease to equilibrium unless an external source replenishes the ATP and deplenishes the ADP. This can be achieved by fixing the concentration of ATP and ADP.

When the external energy consumption is treated implicitly, we find that the product of equilibrium constants around reaction network loops is no longer unity:

$$\prod_{n=1}^{N}\frac{k_{-,n}}{k_{+,n}} = \prod_{n=1}^{N}\frac{r_n k'_{-,n}}{k'_{+,n}} = \prod_{n=1}^{N} r_n \prod_{n=1}^{N} K_{eq,n} = \prod_{n=1}^{N} r_n, \tag{2.62}$$

where r_n is the quotient of concentrations of the implicit reactants and products. Note that if one ATP is consumed per circuit, this corresponds to external free energy input of $\Delta G^\circ_f(\texttt{ATP} \to \texttt{ADP}) + RT\log r$ per circuit, i.e. the cost of assembling ATP from ADP under the conditions of r.

In conclusion, non-equilibrium biochemical processes can be implemented in a simple fashion by (i) writing mass balanced rules, (ii) selecting kinetic parameters that satisfy detailed balance, and (iii) fixed the concentration of high-energy molecules. In addition to satisfying thermodynamic constraints, this approach has a few more advantages. First, the detailed balance constraints eliminate a few free parameters. Second, the amount of free energy "consumed" by an ATP-driven reaction is limited by the change in free energy of ATP to ADP conversion. Third, mass balanced equations are true to the spirit of separating the model from any approximations or coarse-graining prior to simulation. Finally, since the reactions are reversible, it is possible to drive the reaction in the reverse direction if free energy change of the reaction becomes negative, this may be important in some processes, e.g. metabolism.

2.4.4 Ring closure

There are two general types of binding reactions: bimolecular and unimolecular. A bimolecular binding reaction involves two disconnected complexes forming a new bond to form a single, larger complex. A unimolecular binding reaction involves a single complex in which two distant sites form a new bond resulting in a *ring* structure. In the context of polymerization, a bimolecular binding reaction is a *chain elongation*, while a unimolecular reaction corresponds to *ring formation*. In terms of free energy, these reactions behave differently, even if the types of binding sites are the same in both cases. Typically the ring closure reaction is more favorable than chain elongation. This is due to the reduction in entropy required to form a single complex from two separate complexes. Prior to chain elongation, two complexes are free to diffuse in the reactor volume; subsequent to binding, the degree of diffusional freedom is reduces since the position of the complexes has become coupled.

I will attempt to describe the distinction between chain elongation and ring closure using the definition of free energy. Suppose that the binding reaction can be divided into two steps. First, the freely diffusing molecules form an encounter complex, i.e. the molecules are close enough to interact via electrostatic, hydrophobic, or other forces. Let us choose a radius r_e and suppose the molecules have formed the encounter complex if the distance between centers of mass is less than r_e. The change in free energy due to formation of the encounter complex is primarily due to the change in entropy. In the second step, the molecules in the encounter complex transition to the bound complex, i.e. the molecules "snap" together to form a tight bond. The change in free energy due to the second step is mostly a change of internal enthalpy and also internal entropy (a reduction of the internal flexibility of molecules in the encounter complex). The change in free energy of the total reaction is now:

$$\Delta G^\circ_{\texttt{elongation}} \;=\; \Delta G^\circ_1 + \Delta G^\circ_2 \;=\; (\Delta H^\circ_1 - T\Delta S^\circ_1) + (\Delta H^\circ_2 - T\Delta S^\circ_2)\,. \tag{2.63}$$

Assuming that the change of enthalpy in step 1 is much less than the other terms, the total change in free energy is approximately:

$$\Delta G^\circ_{\texttt{elongation}} \;\approx\; -T\Delta S^\circ_1 + \Delta G^\circ_2. \tag{2.64}$$

Finally, since the change in entropy in step 1 is proportional to the log of the ratio of effective volume of the encounter complex $V_e = \frac{4}{3}\pi r_e^3$ and reactor volume, V_r, we have:

$$\Delta G^\circ_{\texttt{elongation}} \approx -RT \log \frac{V_e}{V_r} + \Delta G^\circ_2. \tag{2.65}$$

This equation states that the chain elongation reaction has one free energy contribution due to the reduction in degrees of translational freedom, and a second term due to the the transition from the encounter complex to the bound state. Since $V_e < V_r$, the change in entropy is positive (as expected) and increases with the size of the reactor. However, the second term is independent of the reactor volume.

Let us proceed with the assumption that the change in free energy due to ring formation is approximated by ΔG°_2, and the free energy change due to chain elongation is the sum of contributions from forming the encounter complex and bond formation. Thus we arrived at a relationship between elongation and ring closure:

$$\Delta G^\circ_{\texttt{elongation}} \approx -RT \log \frac{V_e}{V_r} + \Delta G^\circ_{\texttt{ring}}. \tag{2.66}$$

Of course this is an approximation, but this formulation provides intuition for the relationship between chain elongation and ring formation.

This leads us to a conundrum. Consider the following model:

```
begin energy patterns
   A(x!1).B(y!1)  dG_bond
end energy patterns
begin reaction rules
   # bimolecular binding
   A(x) + B(y) <->  A(x!1).B(y!1)  Arrhenius(E0,phi)
   # unimolecular binding
   A(x).B(y) <->  A(x!1).B(y!1)  Arrhenius(E0,phi)
end reaction rules
```

The first reaction rule is a type of chain elongation, while the second reaction rule is a ring formation. Despite the topological difference, reactions generated from both rules will have the same change in free energy: `dG_bond`. This is an unsatisfying result, and so we must find another source of free energy to differentiate chain elongation and ring formation. There are two ways to proceed: add the ring complex to the list of energy patterns, or decompose the change in free energy for bimolecular reactions into internal and diffusional free energy changes.

Let us proceed with the first approach and associate a free energy with ring structures. Suppose, for example, molecules `A` and `B` (above) form a ring structure together with a third molecule `S`. We can assign a free energy to the ring motif, with a negative value to promote stability of the ring structure:

```
begin energy patterns
   A(x!1).B(y!1)                        dG_bond
   A(x!1,s!3).B(y!1,s!2).S(a!3,b!2)   dG_ring
end energy patterns
```

Using the relationship between free energy of chain elongation and ring formation (Equation 2.66), a first estimation for the free energy associated with the ring is proportional to the log of the ratio of the reactor volume to the encounter complex volume:

$$\text{dG\_ring} = RT \log\left(\frac{V_e}{V_r}\right). \tag{2.67}$$

Since $V_e < V_r$, it follows that dG_ring < 0 and the ring is more stable than the chain. Based on Equation 2.52, the rate constants for chain elongation are related to those for ring formation as follows:

$$\begin{aligned} k_{+,\text{elong}} &= k_{+,\text{ring}} \left(\frac{V_e}{V_r}\right)^{\phi} \\ k_{\text{-},\text{elong}} &= k_{\text{-},\text{ring}} \left(\frac{V_e}{V_r}\right)^{\phi-1}. \end{aligned} \tag{2.68}$$

If $\phi = 1$, the unbinding rates for rings and chains are the same. In that case, the entropic free energy change due to encounter complex formation is absorbed entirely in the forward rate constant. However, if $\phi \neq 1$, the free energy of forming the encounter complex is distributed between the forward and reverse rates. Here, the half life of a bond in a ring is different than that in a chain. If $\phi < 1$, the half life of a bond in a ring is longer than the equivalent bond in a chain.

There is currently no facility in eBNG for directly associating a free energy change with the molecularity of a reaction rule.

2.4.5 Comparison to ANC

The Allosteric Network Compiler (ANC) [49] is a pioneering rule-based platform with formal handling of allosteric cooperativity. While ANC is the foundation for eBNG, its limitations are also the motivation for design choices in eBNG. This section outlines the major differences between ANC and eBNG.

In ANC, cooperativity is modeled by pairwise site interactions within a molecule. Biologically, this is motivated by allosteric mechanisms such as conformational switching and post-translational modification. From the perspective of energy pattern formalism, this is equivalent to restricting energy patterns to pairs of sites within a molecules. Thus, ANC is not a generalized energy-modeling platform. Energy BNG extended the ANC concept to arbitrary energy patterns. Any molecular motif can be associated with a free-energy term. Thus it is possible to implement 3$^{\text{rd}}$ and higher order cooperativity. Additionally, cooperative mechanisms can span multiple molecules.

ANC associates independent forward and reverse kinetic constants to each reaction rule. These constants determine the baseline kinetics, i.e. the reaction rate when no cooperative modifiers are present. The independent rate constants can be problematic if a model includes more than one rule involving the same reaction center. In such a model, it is possible that the choice of rate constants will lead to violations of detailed balance. Thus, ANC cannot guarantee detailed balance is all cases. Energy BNG eliminates independent rate constants in favor of an activation energy. Thus, the kinetics of eBNG are entirely posed in the Arrhenius framework, rather than a mixture of kinetics and linear TST. As a consequence, an eBNG model is guaranteed to satisfy detailed balance when multiple rules act on the same reaction center. However, detailed balance can be violated intentionally by mixing traditional BNGL reaction rules with Arrhenius rate laws. This is useful for implementing energy- driven or irreversible reactions.

The kinetic modifiers in ANC are equivalent to linear transition state theory. As a consequence, reactions with the same change in free energy will have the same kinetics. Thus, it is not possible to implement pure catalytic enhancements (i.e. enhanced rate of reaction without a change in equilibrium) unless the uncatalyzed and catalyzed reactions are implemented by separate rules. This approach, however, can be dangerous due to the possibility of violating detailed balance. Energy BNG implements catalytic modifiers through the functional activation energy mechanism. Baseline activation energy is function of the local reaction context. Thus, enzyme binding, conformational switching, and post-translational modifications can influence the energy barrier independent of changes in reaction free energy.

2.5 CONSTRUCTING ENERGY-BASED MODELS WITH BIONETGEN

Energy BNG (eBNG) is the software implementation of the theory described in the previous section. Energy BNG is a superset of the BioNetGen language, hence any feature of BioNetGen may be included in an eBNG model. However, there is no guarantee that an energy-based model mixed with plain BNGL will satisfy detailed balance. This section describes the process of constructing an energy-based model in eBNG using the receptor activation model (Figure 2.2) as an illustration.

2.5.1 Block structure

An energy-based model, like other BioNetGen models, is specified in a plain text file with `.bngl` extension. A model begins with the directive `begin model` and ends with `end model`. The model is composed of a set of blocks, which describe objects of a given type. Each block starts with the directive `begin <blockname>` on a new line, followed by a number of object definitions, and ends with `end <block name>`. All blocks available in a plain `.bngl` are available in energy models. In addition, energy models should include the `energy patterns` block. The allowed blocks are, in recommended order: `parameters`, `molecule types`, `compartments`, `seed species`, `observables`, `functions`, `reaction rules`, and `energy patterns`. Blocks may be omitted if no object of the corresponding type is defined in the model. At the end of the model, a set of actions may be specified, such as `simulate()`, that are performed after the model is parsed by BioNetGen.

With the exception of the new `energy patterns` block, the syntax for each block is the same as plain BNGL. The reader is referred to the BNG book chapters [60, 61] for further documentation.

2.5.2 Preliminaries

Since energy features were not available prior to BNG-2.2.4, we will require version 2.2.4 or greater for the receptor activation model. This is accomplished by calling the `version` action at the beginning of the model.

```
version("2.2.4")
```

It is useful to tell BioNetGen the quantity units of the reference concentration for standard free-energy. BNG is capable of adjusting free-energy parameters for non-standard conditions using compartment volumes and the number of molecules per quantity unit. For example, if the reference quantity is 1 mole, there are N_A molecules per reference quantity. If the reference is 1 molecule, there is 1 molecule per reference quantity. The quantity units are specified by setting the option `NumberPerQuantityUnit`:

```
setOption("NumberPerQuantityUnit",6.0221e23)
```

2.5.3 Model parameters

Model parameters, as well as math expressions derived from parameters, are defined in the `parameters` block. An energy model will typically be parameterized with free-energy parameters, rather than kinetic parameters. But otherwise, the block is no different than a plain BNGL model. Some of the parameters for the receptor activation model are listed in Table 2.2. Free-energy parameters will be described later, when energy patterns and reaction rules are presented.

2.5.4 Compartments

Compartments are recommended, but not required in an energy model. The benefit of using compartments, even for models with only one compartment, is that eBNGL is able

Table 2.2: Parameters block for receptor activation model.

```
begin parameters
   # fundamental constants
   RT  2.577     # kJ/mol
   NA  6.022e23  # /mol
   PI  3.142     # Pi, no units
   # Geometry parameters
   rad_cell   1e-4  # radius of cell, dm
   cell_dens  1e9   # density of cells, /L
   width_PM   1e-7  # effective width of membrane, dm
   # Compartment volumes
   volEC  1/cell_dens               # vol. extracellular space, L
   volPM  4*PI*rad_cell^2*width_PM  # virtual vol. of plasma membrane, L
   volCP  4/3*PI*rad_cell^3         # vol. of cytoplasm, L
   # initial concentrations
   conc_L_0      20e-9   # mol/L
   count_R_0     24000   # molecules/cell
   conc_Ph_0     10e-9   # mol/L
   conc_ATP_0    1.0e-3  # mol/L
   conc_ADP_0    0.1e-3  # mol/L
   ...
end parameters
```

Table 2.3: Compartments block for receptor activation model.

```
begin compartments
   EC  3  volEC      # extra-cellular space
   PM  2  volPM  EC  # plasma membrane
   CP  3  volCP  PM  # cytoplasm
end compartments
```

to adjust the free-energy terms for non-standard concentrations. The receptor activation model has three compartments: extracellular space, plasma membrane, and cytoplasm. The compartments block for this model is shown in Table 2.3.

2.5.5 Molecule types and seed species

The first step in building an energy model, as is the case for rule-based models in general, is describing the types of molecules that will populate the system. It is recommended, but not mandatory, to name each molecule type in the `molecule types` block. Furthermore, the domain structure of macromolecules, including binding domains, potential post-translational modifications, and conformational states, should be described. The procedure for specifying molecule types is the same for energy models as plain BNGL. The contact map in figure 2.2 shows the domain structure of molecules in the receptor activation model.

Next, the initial population of molecular species is specified in the `seed species` block. Each line in the seed species block describes the structure of a molecular species (i.e. species *pattern*) and the initial quantity. It is recommended that seed species quantities have units of *molecule counts* rather than concentrations. This will permit simulation of the model as a continuous system (i.e. ODE) or a discrete stochastic system (i.e. SSA).

In the receptor activation example, there are 5 molecule types: the ligand molecule type, which has a receptor binding domain; the membrane receptor molecule type, with a ligand-binding domain, a dimerization domain, an a phosphorylation site; a cytosolic phosphotase (Ph), with receptor interaction site; and ATP and ADP, which are both simple molecules

Table 2.4: Molecule types block for receptor activation model.

```
begin molecule types
   L(r)             # ligand
   R(l,d,y~0~P)     # trans-membrane receptor
   Ph(y)        # phosphotase
   ATP()            # ATP
   ADP()            # ADP
end molecule types
```

without domain structure. The molecule types block is shown in Table 2.4.

The initial system state is described in the `seed species` block. In the receptor activation model, unbound, unphosphorylated transmembrane receptor is placed in the plasma membrane (PM); unbound phosphotase, ATP, and ADP are placed in the cytoplasm (CP), and free ligand is placed in the extracellular space (EC). The quantity of each species is specified with units of *molecule counts*. The seed species block is shown in Table 2.5.

Observe that `ATP` and `ADP` are prefixed by the `$` symbol in Table 2.5. The `$` symbol indicates that the quantity of the species in held constant during the simulation. `ATP` and `ADP` quantities are held constant in this model to mimic the cellular environment in which concentration of ATP and ADP are tightly regulated.

Table 2.5: Seed species block for receptor activation model.

```
begin seed species
   L@EC(r)          conc_L_0*NA*volEC
   R@PM(l,d,y~0)    count_R_0
   Ph@CP(y)         conc_Ph_0*NA*volCP
   $ATP@CP()        conc_ATP_0*NA*volCP      # ATP quantity held constant
   $ADP@CP()        conc_ADP_0*NA*volCP      # ADP quantity held constant
end seed species
```

Table 2.6: Observables block for receptor activation model.

```
begin observables
   Molecules  Lfree    L(r)
   Molecules  Ltotal   L()
   Molecules  Rbound   L(r!1).R(l!1)
   Molecules  Rdimer   R(d!1).R(d!1)
   Molecules  RyP      R(y~P!?)
   Molecules  Rtotal   R()
   Molecules  RyP_Ph   R(y!1).Ph(y!1)
   Molecules  PhTotal  Ph()
end observables
```

2.5.6 Model outputs: observables

The `observables` block in an energy model has the same syntax and function as plain BNGL. Each item in the observables block specifies an output of the model. Observables are defined by a set of patterns that match species in the system. The value of the observable is the sum over patterns of all matches weighted by the species population.

The receptor activation model includes a variety of observables, such as free ligand (`LFree`), dimerized receptor (`Rdimer`), and phosphorylated receptor (`RyP`). Although the total ligand and receptor are expected to be constant during a simulation, observables `Ltotal` and `Rtotal` are included in order to verify mass conservation in the model. The observables block for the receptor activation model is listed in Table 2.6.

2.5.7 Energy patterns

Energy patterns and the associated standard free energy of pattern formation parameters are specified in the `energy patterns` block. The syntax of each energy pattern is

```
Pattern Expression
```

where `Pattern` is a BNGL pattern graph that describes a molecule motif and `Expression` is a parameter name or constant math expression that specifies the standard free energy of pattern formation. For in-depth discussion on choosing energy patterns, see Section 2.4.1.1.

2.5.8 Reaction rules

Reaction rules in an energy model are are similar a standard rule, but with a key distinction: the reactions generated by the rule do not necessarily share the same rate constant. Thus, an energy-based rule is like a *template* for reaction rules, where the rate constant for each specific rule is derived from analysis of the reaction specific context. More specifically, reaction rate constants are derived from the standard change in free energy of the reaction, along with an activation energy parameter and rate distribution parameter. These interpretation of these parameters was discussed in Section 2.4.2.

Energy-based rules have several advantages over plain rules. First, an energy-based rule provides a mechanism for lumping a set of rules that have the same *reaction center*. Consequently, an energy model has fewer rules than an equivalent plain RBM. Second, since the context is accounted for later, an energy-based rule usually does include any additional context. This simplifies the process of writing each rule. Finally, the method for computing change in free energy guarantees that free energy is conserved around loops and detailed balance is satisfied. In sum, energy-based rules result in a model with fewer rules that are simple to read and write and satisfy detailed balance constraints.

Energy-based rules are implemented in eBNG by assigning the *Arrhenius* rate law type. All energy-based rules (with exceptions for computational efficiency) should be reversible, since we assume the system has an equilibrium. Unlike a plain BNGL rule, a single Arrhenius rate law is sufficient for the forward and reverse rule. Consider the following ligand-receptor binding rule in the receptor activation model.

```
L(r) + R(l) <-> L(r!1).R(l!1) Arrhenius(phi,E0_LR/RT).
```

The reactant patterns and product patterns follow the same syntax conventions as all BNGL rules. This is an energy-based rule since it has the Arrhenius rate law, which is sufficient for the forward and reverse reactions. Parameters for the Arrhenius rate law will be discussed in the next section. In consideration of the energy patterns defined for the model, this energy-based rule encodes three plain reaction rules, each corresponding to a different reaction context and rate constant.

```
L(r) + R(l,d)          <-> L(r!1).R(l!1,d)             kp1,km1
L(r) + R(l,d!1).R(l)   <-> L(r!2).R(l!2,d!1).R(l,d!1)  kp2,km2
L(r) + R(l,d!1).R(l!2,d!1).L(r!2) \
               <-> L(r!3).R(l!3,d!1).R(l!2,d!1).L(r!2)  kp3,km3
```

All the rules have the same reaction center: the `l` site of the receptor binds (or unbinds) the `r` site of the ligand. But the rules differ in context: ligand binding an undimerized receptor, ligand binding an un-ligated receptor dimer, and ligand binding a singly-ligated receptor dimer. Note that eBNG does not explicitly enumerate the set of reaction rules encoded by an energy rule; instead, the reactions are generated directly. The task of inferring the minimal set of reaction rules that are equivalent to the energy-based rule is left for the future.

Reaction rules with elementary or functional rate laws may be mixed with energy-based rules. However, there is no guarantee that reactions generated by such rules will satisfy detailed balance. Therefore, it is best to avoid non-energy-based rules unless the reaction consumes or produces energy external to the model system (e.g. degradation, synthesis, ATP consuming reactions) and the modeler wishes to treat the reaction outside the free energy framework. But, as discusses previous, it also possible to naturally integrate energy consuming rules into the energy framework.

The reaction rules block with the complete set of energy-based rules for the receptor activation model is listed in Table 2.7. Note that this model is purely energy-based and therefore detailed balance will be satisfied.

2.5.8.1 Catalytic enhancement via functional activation energy

In this section, I will illustrate a reaction rule that encodes both catalyzed and non-catalyzed reactions. We will temporarily deviate from the receptor activation example and consider the following model system:

```
begin seed species
   S(a~0~P,e)   S0
   E(s)         E0
end seed species
begin energy patterns
   S(e!1).E(s!1)  G_S_E/RT
   S(a~P)         G_SaP/RT
end energy patterns
begin observables
   Molecules  SE  S(e!1).E(s!1)
end observables
begin reaction rules
   S(e) + E(s) <-> S(e!1).E(s!1)  Arrhenius( E1, phi )
   S%x(a~0)x <-> S%x(a~P)       Arrhenius( E2-E2cat*SE(x), phi )
end reaction rules
```

The second reaction rule describes the site modification `S(a~0) -> S(a~P)`. Two reactions are generated from this rule, an uncatalyzed reaction and an enzyme catalyzed reaction.

```
   S(a~0,e) <-> S(a~P,e)  Arrhenius( E2, phi )
   S(a~0,e!1).E(s!1) <-> S(a~P,e!1).E(s!1)  Arrhenius( E2 + E2cat, phi )
```

Both reactions have the same change in free energy, `G_SaP/RT`. Under linear TST, both reactions would be assigned the same rate. But the activation energy is defined by a local function `E2+E2cat*SE(x)` that queries for the presence of the enzyme. If the enzyme is present, the activation barrier is lowered by `E2cat`. Since the enzyme is present on both sides of the reaction, the local function evaluates to the same value for the reactants and the products.

2.5.9 Energy parameters

The kinetics of an energy model are derived from three types of parameters: *standard free-energy of pattern formation* (for brevity, *pattern free-energy*), *baseline activation energy*, and *rate distribution* parameters (see Sections 2.4.1 and 2.4.2). The pattern free-energy parameters are paired with energy patterns in the `energy patterns` block. Baseline activation energy and rate distribution parameters are specified in the `Arrhenius` type rate laws associated with each reaction rule. Energy and kinetic parameters for the receptor activation model are listed in Table 2.8.

2.5.10 Simulating energy-based models

Energy-based models can be simulated using any of the network-based methods supported in BioNetGen. Prior to simulation the reaction network must be generated by calling the `generate_network` action. After network generation, the model system can be simulated as a continuous, deterministic system with `simulate_ode`; or as a discrete stochastic system with `simulate_ssa`. Alternatively, a custom model integrator can be exported to MATLAB as either an `m-file` script or `MEX` code by calling `writeMfile()` or `writeMexfile()`, respectively. The `m-file` integrator is implemented via MATLAB's `ode23s` integrator, while the `MEX` code integrator is based on the SUNDIALS CVODE library. The `MEX` code integrator is typically much faster, and is recommended for heavy computational tasks such as parameter estimation.

Network-free simulators (e.g. NFsim) do not support energy BNGL models. Support for energy models may be become available in the future; but at present, network-free methods for energy models do not exist.

2.6 MODEL SELECTION AND CALIBRATION WITH THE ENERGY LASSO

In the previous section we assumed that the relevant energy patterns were known in advance. In practice, the structural factors contributing to free energy may not be well understood. Thus we need a method to select energy patterns and calibrate the associated parameters. In this section, I present the *Energy Lasso*, a method for inferring energy patterns from experimental data.

The reader may recognize this as a model selection problem. Given a base model without cooperative interactions, what set of cooperativity energy patterns provides the best explanation of the experimental data? Although adding more energy patterns will improve the fitness of the model, it also introduces extra parameters that increase the likelihood of overfitting the data. The ideal model includes just enough energy patterns to explain the experimental data, but no more.

2.6.1 Selecting energy patterns

Suppose we begin with a set of P energy patterns that could be present in the system. One approach is to consider all 2^P variations of the model corresponding to all subsets of energy patterns. The *Bayes factor* is the probability of a model given the experimental data. It provides the basis for rigorous Bayesian model selection [102, 103]. We could proceed by computing the Bayes factor for each model variant and choosing the model with the greatest probability. This approach has been applied to biological systems where the number of variants under consideration is relatively small [16]. But this computationally intensive method does not scale well as the number of energy patterns increases. For example, if we consider 10 energy patterns, then over 1000 Bayes factors must be computed. Clearly, an alternative is required if we are to consider a large number of energy patterns.

LASSO is a method for simultaneous model selection and parameter calibration [104]. Rather than consider each discrete model variation separately, each possible mechanism is considered simultaneously. The strength of each mechanism is governed by an associated parameter. When the parameter is set to zero, the mechanism is effectively absent from the model. Each parameter has a Laplace prior probability, which is centered on zero and provides a preference for zero-valued parameters. But the prior also has large tails, which permits the parameter to deviate far from zero if the data likelihood improves sufficiently. Thus, the LASSO drives parameters to zero, effectively disabling a mechanism, unless the experimental data provides strong support for a non-zero value.

To provide more rigor, an overview of Bayesian probability is required. Bayesian parameter estimation begins with Bayes' rule, which states that the probability that a parameter set generated a given set of experimental observations is proportional to the product of the prior probability of the parameters and the likelihood of the experimental observations assuming the parameter set generated the data:

$$p(\vec{\theta}|X) \;=\; \frac{p(X|\vec{\theta}) \cdot p(\vec{\theta})}{p(X)}, \tag{2.69}$$

where $\vec{\theta}$ is a vector of model parameters and X is the experimental data.

For LASSO, the prior probability is given by the Laplace distribution:

$$p(\vec{\theta}|\lambda) \;=\; \prod_p \exp\left(-\frac{|\theta_p|}{\lambda^{-1}}\right), \tag{2.70}$$

where p is an index that ranges over the number of parameters. The log-posterior probability then takes the form:

$$\log p(\vec{\theta}|X,\lambda) \;=\; \log p(X|\vec{\theta}) + \log p(\vec{\theta}|\lambda) - \log p(X|\lambda). \tag{2.71}$$

Since the last term is constant with respect to $\vec{\theta}$, the log posterior is proportional to:

$$\log p(\vec{\theta}|X,\lambda) \;\propto\; \log p(X|\vec{\theta}) - \lambda \sum_p |\theta_p| \,. \tag{2.72}$$

So we see the log-likelihood of the data is penalized by a term that depends on the absolute value of the parameters weighted by λ. Thus, LASSO is also known as L1-regularization, since it is equivalent to a penalty proportional to the absolute value of parameters.

Given a model and experimental data, the posterior distribution can be sampled by Markov chain Monte Carlo (MCMC) methods. The Metropolis algorithm [105] is a simple but effective algorithm for sampling parameters from the posterior distribution. More sophisticated sampling methods, such as *simulated annealing* [106, 107] and *parallel tempering* [108, 109], are often employed since the chain converges towards the stationary distribution more rapidly in many practical applications. Once the parameter distribution is obtained, predictions of model behavior with uncertainty estimates can be obtained by simulating the model with parameter samples from the posterior. Under the combination of LASSO and Bayesian parameter estimation, important model mechanisms are revealed by significant differences between the Laplace prior and the posterior distribution of a parameter.

2.6.2 The energy lasso method

The *energy lasso* combines energy-based modeling with the LASSO approach to model selection. The modeler begins with a base model without cooperative interactions. Next, a set of cooperativities are postulated in terms of energy patterns that refer to two or more molecular sites. Finally, the model is calibrated to experimental data using the LASSO method, where Laplace priors are associated with each parameter associated with an energy pattern. The LASSO selects the energy patterns that are supported by the experimental data. Inspection of the posterior parameter distributions reveals the important mechanisms. Specifically, an energy pattern is important if the the posterior distribution has escaped the envelope of the Laplace prior.

The energy lasso provides a systematic approach to selection of energy patterns and calibration of parameters that does not depend on the biases of the modeler. Due to the computational cost of other approaches, a modeler is limited to consideration of a few cooperative mechanisms while others are excluded from the model. If the model output is sensitive to these choices, the drawn conclusions will incomplete (at best) or invalid (at worst). The energy lasso, in contrast, permits consideration of a large number of mechanisms and allows the experimental data to determine which will be included in the final model.

For example, in a model of the EGF receptor, Kholodenko et al. [17] assumed that only one cytosolic protein can bind a receptor at a time. This is equivalent to a large negative cooperativity between the simultaneous binding partners. In contrast, Blinov et al. [52] assumed that binding of intracellular factors occurs independently (i.e. no cooperativity). Both models made assumptions about the nature of the cooperativity that may influence the results and conclusions. Under the energy lasso, an energy pattern could be associated with simultaneous binding patterns and the calibration/selection procedure will determine whether the cooperativity is present or not.

The energy lasso enables consideration of many cooperative mechanisms and reduces modeler bias. But it does introduce its own systematic bias via the Laplace prior. Since parameters are penalized for deviating from zero, the parameter distribution is skewed relative to the distribution obtained from only the data likelihood. Despite this, the energy lasso bias should be preferable to making arbitrary choices about cooperative mechanisms. The energy lasso is also preferable to calibrating the most complex variant of the model without regularization, since the abundance of free parameters may lead to overfitting [110].

Table 2.7: Reaction rules block for receptor activation model.

```
begin reaction rules
   # ligand binding
   L(r) + R(l)  <-> L(r!1).R(l!1)    Arrhenius(phi,E0_LR/RT)
   # receptor dimerization
   R(d) + R(d)  <-> R(d!1).R(d!1)    Arrhenius(phi,E0_RR/RT)
   # Ph binding
   R(y) + Ph(y) <-> R(y!1).Ph(y!1)  Arrhenius(phi,E0_RPh/RT)
   # trans-phosphorylation
       R(d!1).R(d!1,y~0) + ATP  \
   <-> R(d!1).R(d!1,y~P) + ADP  Arrhenius(phi,E0_catRR/RT)
   # dephosphorylation by phosphatase
       R(y~P!1).Ph(y!1)  \
   <-> R(y~0!1).Ph(y!1)  Arrhenius(phi,E0_catPh/RT)
end reaction rules
```

Table 2.8: Free energy and activation energy parameters for the receptor activation model.

```
begin parameters
   ...
   # standard free energy of formation, kJ/mol
   G_LR         -47.5
   G_RR         -11.9
   G_RPh        -41.5
   G_RyP         51.1
   G_ATP         51.1
   G_RyP_Ph      5.9
   G_LRR        -5.9
   G_LRRL       -11.9
   # baseline activation energy, kJ/mol
   E0_LR        -11.9
   E0_RR        -5.9
   E0_RPh       -17.8
   E0_catRR     -11.9
   E0_catPh      5.9
   # rate distribution parameter, no units
   phi  0.5
end parameters
```

3.0 MODELING COMPARTMENTAL BIOCHEMICAL SYSTEMS

Rule-based modeling (RBM) was motivated by the multi-domain structure of biological macromolecules and the local nature of interactions. The compartmental topology of the cell, however, can also have profound effects on the regulation of cellular processes by controlling both the reactions that can occur and the rate at which they do so. In this chapter we describe compartmental BNGL (cBNGL), which extends BNGL to enable explicit modeling of the compartmental organization of the cell and its effects on system dynamics[1]. By making localization a queryable attribute of both molecules and species and introducing appropriate volumetric scaling of reaction rates, the effects of compartmentalization can be naturally modeled using rules. These properties enable the construction of new rule semantics that include both universal rules, those defining interactions that can take place in any compartment in the system, and transport rules, which enable movement of molecular complexes between compartments.

3.1 INTRODUCTION

Proteins in cellular regulatory systems can interact in a combinatorial number of ways to generate myriad protein complexes [44]. These interactions, because of the multicomponent composition of proteins, can be modeled by rules that specify the classes of reactions that

[1]This chapter is a adapted from a conference proceeding with co-authors Leonard Harris and James Faeder [111]. My contribution to the work includes design of the language extension, software implementation, and composition of the draft manuscript. Leonard Harris contributed substantially to manuscript composition and is co-first-author. James Faeder contributed to design of the language extension, software implementation, and manuscript revisions.

can occur and define reaction networks that account comprehensively for the consequences of protein-protein interactions [54]. The assumption underlying this modeling approach, which is consistent with the modularity of regulatory proteins [40], is that interactions are governed by local context, that is, properties of the components of the interacting molecules that are proximal to the site of interaction. As long as this is the case, rule-based models can achieve a compact description of a network with a large (even infinite) number of complexes [54].

The complex spatial topology of the cell, however, can also have profound effects on the regulation of cellular processes by controlling both the reactions that can occur and the rate at which they do so. Membranes are key players not only because they separate molecules in different compartments but also because they mediate the flow of material and information from one compartment to another [112]. Chemical reactions that occur at a cell membrane also have an enhanced rate because of the drastically smaller volume of the membrane compartment, a thin fluid layer roughly 100-1000 times smaller than the volume of the cell [113, 114]. The spatial localization of a reacting species in a biochemical network is thus a critical property that affects its reactivity. A rule-based model of the network must therefore describe, either implicitly or explicitly, its effects on the constituent reactions. The main goal of this paper is to describe an extension to an existing rule-based modeling language for biochemical systems that explicitly represents the topology of cellular compartments, the localization of species to specific compartments, and the effects of localization on biochemical reaction rates.

BioNetGen is an open-source software package that provides tools and a language [the BioNetGen language (BNGL)] for rule-based modeling of biochemical systems [60]. The syntax and semantics of the language are formally rooted in graph theory [66], but the language itself is simple, intuitive, and accessible to modelers with a wide range of mathematical backgrounds. BNGL is similar to the κ-calculus, which has also been developed as a language for rule-based modeling [68]. A review of tools for rule-based modeling can be found in [54], although a number of new entries have appeared more recently [69, 76–78, 80, 115–117].

BNGL does not currently allow explicit representation of cellular compartments and does not systematically account for the effects of spatial localization either in the selection of species that can undergo reactions or in the calculation of reaction rates. The effects of localization can be modeled in an ad hoc manner but it obscures the generality of molecular interactions, may require additional restrictions [such as the `include` and `exclude` commands [50]], and may require extensive enumeration of complexes in the rules. In this paper, we describe an extension of BNGL, which we call *compartmental* BNGL (cBNGL), that enables explicit modeling of the compartment topology of the cell and its effects on system dynamics. We show that by introducing a compartment topology, making localization a queryable attribute of both molecules and species, and introducing appropriate volumetric scaling of reaction rates the effects of compartmentalization can be naturally included in a rule-based model. The outline of the remainder of the chapter is as follows. In Sec. 3.2, we motivate our introduction of cBNGL with a schematic cell regulatory model that captures many essential features of intracellular biochemistry. The compartmental extension of BNGL is then described in Sec. 3.4. In Section 3.5, we summarize the strengths and weaknesses of cBNGL and compare it to various related approaches.

3.2 A COMPARTMENTAL MODEL OF THE CELL

The function of a signal transduction pathway is to detect an extracellular signal, relay this information inside the cell, and induce an appropriate change in cell function [112]. In Figure 3.1, we present a model of receptor-mediated signaling coupled with nuclear transport and transcriptional gene regulation that highlights the role of compartmental localization and transport. This example model will be used throughout the chapter to motivate and illustrate cBNGL.

Signaling is initiated when an extracellular ligand (L) is detected by membrane-localized receptors (R) that bind the ligand (R_1; Fig. 3.1). Because L can bind to itself (R_2), ligand-bound receptors can dimerize (also R_2) and be brought inside the cell by endocytosis ($R_{3–5}$), a process in which a small region of the plasma membrane is pinched off, forming a small bubble called an endosome [112]. Localizing a receptor complex to an endosome has the effect of trapping the ligand and receptor molecules in a small volume, which reduces entropy and enhances the free energy of binding. Note that during receptor internalization the receptor

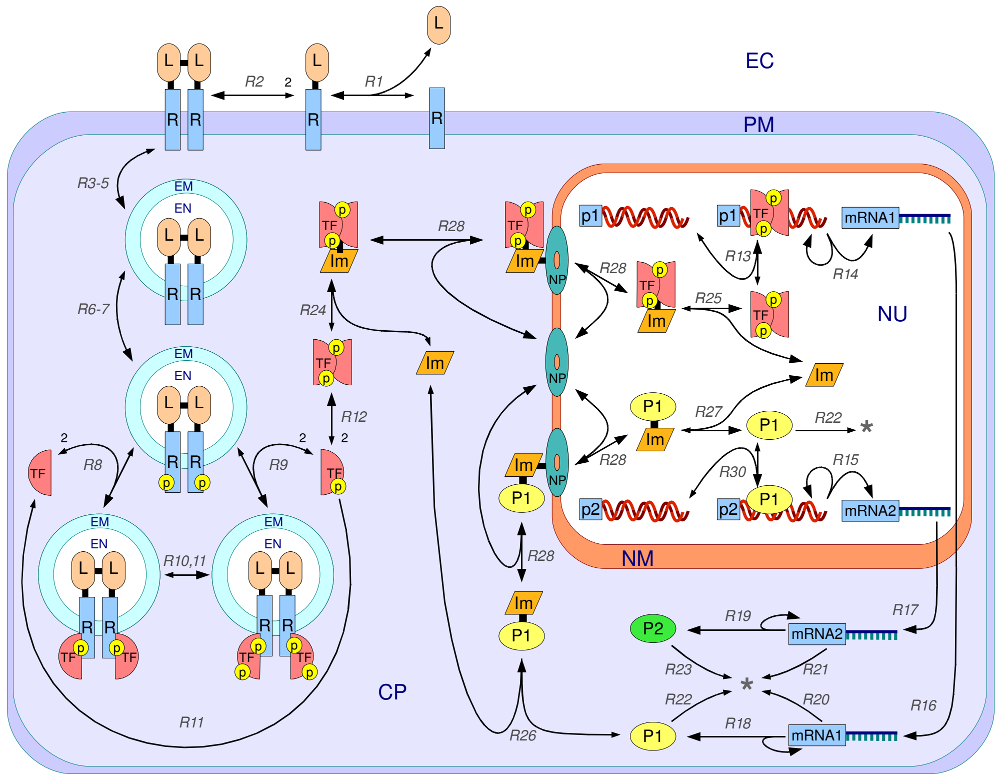


Figure 3.1: A compartmental model of the cell. The model couples simplified processes of signal transduction, nuclear transport and transcriptional regulation in a single eukaryotic cell. The system consists of four volume compartments: extracellular (EC), cytosol (CP), endosomal (EN) and nuclear (NU). These are separated by three membrane surfaces: plasma (PM), endosomal (EM) and nuclear (NM). The model is presented as a pathway that proceeds from ligand (L) binding to expression of protein P2. The underlying rule-based model defines a set of 354 reactions between 78 species. Bonds between molecules are shown as black lines. Black arrows between species represent reactions. Gray arrow labels correspond to the rule number that describes the reaction (see model files at www.bionetgen.org/wsc09). Black integer-valued arrow labels represent reaction stoichiometry (if not equal to unity). DNA promoters are pictured as a double helix icon.

complex moves from PM to EM, bound ligands move from EC to EN, and receptor-bound molecules in CP remain in CP.

Receptor dimerization brings the cytoplasmic domains of receptors into close proximity, allowing transautophosphorylation of a tyrosine amino acid by the receptors' catalytic domains ($R_{6\text{–}7}$). Inactive transcription factor (TF) can bind phosphorylated receptors (R_8), leading to transautophosphorylation of TF in complexes containing receptor dimers ($R_{10\text{–}11}$). Phosphorylated TF tends to unbind from the complex (R_9) and has a high affinity for forming dimers (R_{12}). Dimerized TF forms an active transcription factor that is escorted into the nucleus, through a nuclear pore, by an importin (Im) molecule (R_{24}, R_{28}). Inside the nucleus, the TF dimer binds to a promoter on DNA activating transcription of mRNA (R_{13}), which is transported out of the nucleus to the cytosol (R_{16}) and translated into P1 (R_{18}). Cytosolic P1 is also escorted into the nucleus by Im (R_{28}), where it binds a second promoter to activate transcription (R_{15}, R_{17}, R_{19}) and expression (R_{30}) of a second protein (P2).

3.3 REPRESENTING COMPARTMENTS IN PLAIN BNGL

This section will briefly overview BNGL and highlight aspects relevant to compartmentalizing the reaction network. BNGL has been more thoroughly elsewhere in this document (Section 1.4.3).

Molecules, the basic building blocks of a BioNetGen model, are declared in the `molecule types` block. Molecules may contain components, which represent the functional elements of molecules and may bind other components, either in the same molecule or another. Components may be associated with state variables with a finite set of possible values, each representing a conformational or chemical state of a component, such as phosphorylation status. The name of the molecule type is given first followed by a comma-separated list of its components in parenthesis. The allowed values of state variables are indicated by '∼' followed by a value, as in `L(r,d,loc~EC~EN)`, which declares a ligand molecule `L` that contains a receptor binding component `r`, a dimerization component `d`, and a location component `loc` that takes on values EC or EN.

The `seed species` block defines the species initially present in the system. For example, the line

```
L(r,d,loc~EC)   Lig0
```

specifies that the initial amount of free ligand monomers in EC is `Lig0`, a model parameter. Molecular complexes may also be specified, as in

```
L(r,d!1,loc~EC).L(r,d!1,loc~EC),
```

where a bond linking the `d` components of each `L` molecule is indicated by a shared bond label, `!1`.

The `reaction rules` block contains rules that define how molecules interact. A rule is comprised of a set of reactant patterns, a transformation arrow, a set of product patterns, and a rate law. A pattern is a set of molecules that select species through a mapping operation [66]. The match of a molecule in a pattern to a molecule in a species depends only on the components specified in the pattern (including wildcards), so that one pattern may select many different species. The '`+`' operator separates two reactant patterns that must map to distinct species (i.e., they may not reside in the same complex). The '`.`' operator separates molecules that are part of the same species. The transformation arrow may be either unidirectional (`->`) or bidirectional (`<->`). Five basic types of operations are carried out by the rules in the example system of Sec. 3.2: binding and unbinding of two molecules through a specified pair of components, transformation of a state variable, synthesis, and degradation. An example of a binding rule is

```
R8: R(tf~pY) + TF(d~Y,r,loc~CP) <-> \
        R(tf~pY!1).TF(d~Y,r!1,loc~CP)   kp_R_TF/vol_CP, km_R_TF
```

where the underlined portions identify the reaction center, which is the set of components modified by the rule. (`R8` indicates that this is rule #8 in the BNGL model file.) This rule specifies that *any* R molecule containing an unbound, phosphorylated `tf` component may bind to a `r` component of an unphosphorylated TF in CP. Here, the `tf` component of `R` is bound to the `r` component of `TF` by the addition of an edge labeled `1`, indicated by the two bond labels (`!1`) in the products.

`kp_R_TF/vol_CP` and `km_R_TF` specify the rate constants in the forward and reverse directions. (Assuming that the bimolecular rate constant is given in standard units of M^{-1} s^{-1}, `kp_R_TF` is that value divided by Avogadro's number.) In this case, the rate constant for the forward direction is a formula comprised of a bimolecular rate constant divided by the volume of CP. Parameters in BNGL have no explicitly defined units and bimolecular rate constants are generally given on a *per molecule per cell* basis. BNGL supports elementary (`Ele`) rate laws as well as two non-elementary types, Michaelis-Menten (`MM`) and saturation (`Sat`) [60].

Several examples are provided in the model files at `www.bionetgen.org/wsc09`.

3.4 COMPARTMENTAL BNGL

As discussed in Sec. 3.1, compartmental models such as the one in Fig. 3.1 can, in principle, be modeled using standard BNGL. For example, we have seen in Sec. 3.3 that in order to specify location in BNGL a component (`loc`) can be added to the component lists for molecules. Rules can then be written with the appropriate location states as context for the interacting molecules. In the common case where identical interactions can take place in different compartments, this means that multiple versions of the same interaction rule must be enumerated. Each will differ by only the values of the location labels and, in the case of bimolecular interactions, by the volume-dependent rate constant. Besides being tedious, this approach obscures the generality of interactions specified by a rule. In more complex cases, for example when there are many different ways that a particular molecule can be tethered to a membrane, the required enumeration will be prone to error; it is exactly such enumeration that the rule-based approach was developed to avoid. A similar situation can arise in transport reactions that depend only on the presence of a particular molecule or component state; a different rule is required for each possible stoichiometry of the transported complex. In cBNGL, this enumeration is avoided by introducing localization as a property of both molecules and species, with the localization of a species being a derived property of the localization of its constituent molecules. Species localization also permits restricting the scope of rule application to reactants in the same or adjacent compartments and the determination of volume-dependent rate constants for bimolecular and higher-order reactions. The local nature of molecular bonds also imposes natural topological constraints on complexes. If a molecule is tethered to the plasma membrane, then it cannot be transported to the nucleus without first breaking the tether. Rule application is restricted to adhere to these constraints. In the following, we describe cBNGL in detail. The complete syntax for cBNGL in extended Backus-Naur form is included in Appendix A.

3.4.1 Units

Species counts are assumed to be in population units or moles, not concentrations, in keeping with standard BNGL [60]. Rate constants for bimolecular reactions, however, are assumed to be given in units of *volume*/*time*[2] and Michaelis constants for `MM` and `Sat` rules in units of $volume^{-1}$ (this assumes that the appropriate factor of Avogadro's number, N_A, is included in the value, e.g., k_{bi}/N_A and $K_M \times N_A$). This allows for the specification of universal reaction rules that apply across compartments and whose rates are automatically scaled by the appropriate compartment volume (see Sec. 3.4.5.1).

3.4.2 Compartment topology

Cellular topology, as depicted in Fig. 3.2A, implies that the compartment graph, in which nodes represent compartments and directed edges represent containership, is a tree (Figure 3.2B). The structure is essentially the same as the compartment structure used in the Systems Biology Markup Language (SBML) [118]. Rules for defining compartment topologies in cBNGL are as follows:

1. A surface compartment must either be an outermost compartment (i.e., no containing compartment) or be contained by a single volume compartment.
2. A volume compartment must either be an outermost compartment or be contained by a single surface compartment.

[2] If molecule counts are considered explicitly, bimolecular rate constants for cBNGL have units of /*time*/(#/*volume*). However, we usually operate under the convention that *counts* are unitless.

3. A surface compartment may contain only a single volume compartment.
4. A volume compartment may contain *multiple* surface compartments.

Compartments in cBNGL are declared in a `compartments` block analogous to other blocks used in BNGL (see Sec. 3.3). The syntax of each line in the block is

```
[index] compartment_name dimension volume [containing_compartment]
```

where the square brackets denote optional arguments. `compartment_name` is a standard BNGL name (see Faeder et al. [60]), `dimension` is either 2 or 3 depending on whether the compartment is a surface (e.g., plasma membrane) or a volume (e.g., cytoplasm) and `volume` is the compartment volume, in units consistent with those used for bimolecular rate constants in the `parameters` block. For a surface compartment, the volume is the product of the surface area and an effective width, which accounts for the enhancement of a reaction rate relative to its value in three-dimensional space [113]. `containing_compartment` is the name of the parent compartment, if applicable (e.g., CP is contained by PM). The full topology specification for the example system of Sec. 3.2 is shown in Fig. 3.2C.

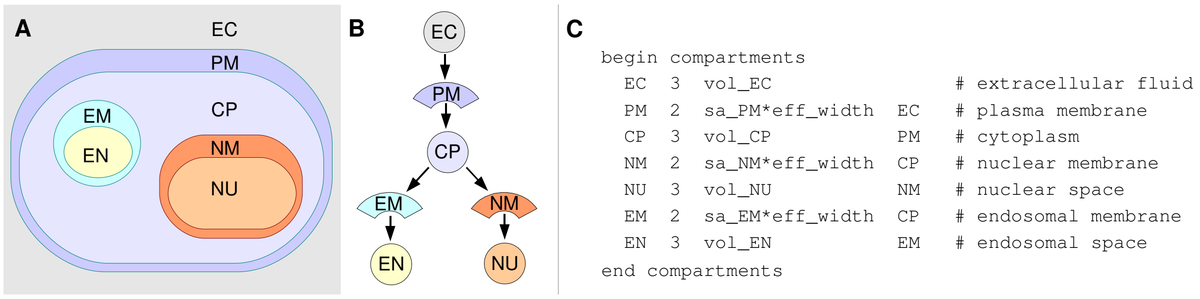
3.4.3 Molecule location

Each instance of a molecule in cBNGL has a compartment attribute, obviating the need for the location (`loc`) components of Sec. 3.3. Ligand molecules, for example, are declared in the `molecule types` block of a cBNGL input file as `L(r,d)`. In species, molecules must be given a location, specified using what we call postfix notation, e.g., `L(r,d)@EC`, which specifies a free ligand molecule in EC. Postfix notation may also be used in patterns to specify the location of a molecule, as in `L(r)@EC`. This will match ligand molecules in EC with a free `r` site.

3.4.4 Species location

Complexes can be built in cBNGL from molecules in the same way as in standard BNGL. For example, a freely diffusing ligand dimer in EC is specified as

```
L(r,d!1)@EC.L(r,d!1)@EC.
```



```
begin compartments
  EC  3  vol_EC                # extracellular fluid
  PM  2  sa_PM*eff_width  EC   # plasma membrane
  CP  3  vol_CP           PM   # cytoplasm
  NM  2  sa_NM*eff_width  CP   # nuclear membrane
  NU  3  vol_NU           NM   # nuclear space
  EM  2  sa_EM*eff_width  CP   # endosomal membrane
  EN  3  vol_EN           EM   # endosomal space
end compartments
```

Figure 3.2: **Compartment topology of cell model in cBNGL** (A) Illustration of a cell model with extracellular (EC), cytoplasmic (CP), nuclear (NU) and endosomal (EN) volumes and plasma (PM), endosomal (EM) and nuclear (NM) membrane surfaces. (B) Representation of the compartment topology by a directed graph. Volumes are represented by circles and membranes by arcs. A directed edge points from C1 to C2 if compartment C1 is immediately outside of compartment C2. (C) cBNGL specification of the topology shown in B.

Complexes can comprise molecules in adjacent compartments, as in the pattern

`L(r!1)@EC.R(l!1)@PM`,

but are currently not allowed to span multiple surface compartments. Complexes may then span, at most, a single surface compartment and its two adjacent volume compartments. The localization of a species is a global property of a species that is based on the location of each element comprising the species.

Definition 1 (Species localization).

A species `X` is said to be localized to a volume compartment `V` if, and only if, all of the molecules in `X` reside in `V`. Conversely, `X` is said to be localized to a surface compartment `S` if any molecule in `X` resides in `S`. Species cannot be formed that span multiple surface compartments.

Species can be referenced in cBNGL using a prefix notation that specifies their location in line with the above definition. For example, the ligand dimer species above can be written alternatively as `@EC:L(r,d!1).L(r,d!1)`. Prefix and postfix notations can also be used together, as in

```
@PM:R(l!3,tf~pY).L(r!3,d!1)@EC.L(r!2,d!1)@EC.R(l!2,tf~Y)
```

where both R molecules are located in PM. More examples of species and patterns in cBNGL are shown in Fig. 3.3.

3.4.5 Reaction rules

Rules in cBNGL are written using the same basic syntax as in standard BNGL (Sec. 3.3). The key difference is the possible specification of compartment localization for molecules or species in the reactant and product patterns. When compartments are omitted a rule is considered universal, acting on all sets of matching reactants that are in the same or adjacent compartments. If compartments are specified, we refer to the rule as scope-restricted. Incorporating compartments into the modeling language also requires the introduction of new types of rules for modeling compartment-to-compartment transport. Currently, transport rules are always scope-restricted because both the source and destination compartments must be specified. Lifting this restriction will require more extensive modification of the current BNGL syntax.

3.4.5.1 Universal reaction rules The utility of universal rules is that they simplify the specification of compartmental models. As discussed above, specifying such models in standard BNGL requires use of location components and enumeration of different versions of the same rule in different compartments. This is avoided in cBNGL by using universal rules. In addition, cBNGL automatically applies the restriction that reactants be in the same or adjacent compartments and applies the correct volumetric scaling to rate constants and other reaction parameters for bimolecular and higher-order reactions. No scaling is required for first order reactions. For elementary bimolecular reactions with the rate constant given in

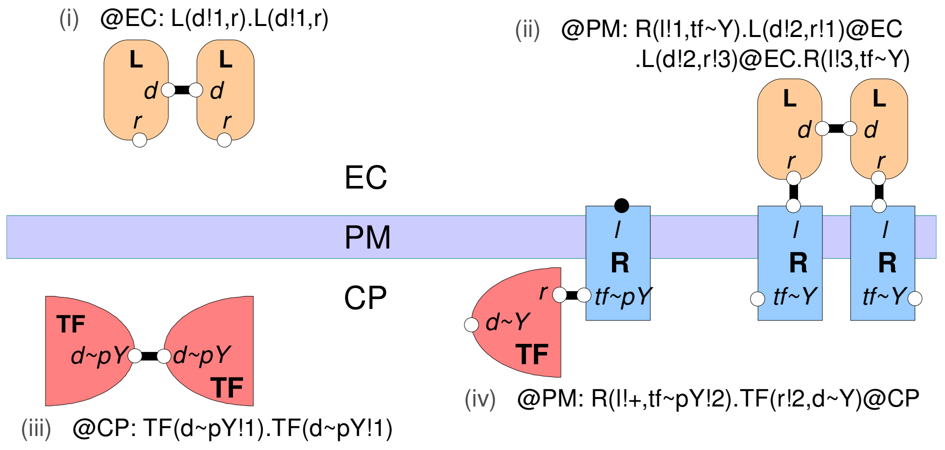

(i) @EC: L(d!1,r).L(d!1,r)
L
d
r
L
d
r
(ii) @PM: R(l!1,tf~Y).L(d!2,r!1)@EC
.L(d!2,r!3)@EC.R(l!3,tf~Y)
EC
PM
CP
TF
d~pY
d~pY
TF
(iii) @CP: TF(d~pY!1).TF(d~pY!1)
l
R
tf~pY
r
d~Y
TF
(iv) @PM: R(l!+,tf~pY!2).TF(r!2,d~Y)@CP
l
R
tf~Y
l
R
tf~Y

Figure 3.3: **Species and patterns in cBNGL.** (i) Species declaration for a ligand dimer localized in the extra-cellular space (EC) using compartment prefix notation. (ii) Species declaration of a receptor-dimer complex. The species is localized to the plasma membrane (PM) but two of the member molecules are in the EC. (iii) A pattern that matches all TF dimers that are phosphorylated at site `d` and localized to the cytosol (CP). This pattern is not a species since components `r` and `im` of TF are not declared. (iv) A pattern that matches complexes localized to the PM containing R and TF with additional context. Site `l` is shown as a filled black circle, indicating that R must be bound to some, unspecified molecule through site `l`.

units of *volume*/*time*, the rate constant is divided by the volume of the reactant compartment with the highest dimension [50, 113]. (The obvious generalization for higher-order reactions, dividing by the product of the $N-1$ highest-order compartment volumes, is implemented but its use is *not recommended*.) Note that volumetric scaling is performed for *all* rules not just universal ones.

An example of a universal rule from the cBNGL specification of the compartmental model in Sec. 3.2 is ligand dimerization:

```
R2: L(d) + L(d) <-> L(d!1).L(d!1)   kp_LL, km_LL
```

Figure 3.4A illustrates the six instances of this rule along with the appropriate scaling factors. Universal rules can also describe the synthesis of molecules, as in

```
R14: DNA(p1!+) -> DNA(p1!+) + mRNA1()   k_transcribe
```

The created molecule is placed is in the same volume as the reactant or, for bimolecular reactions, in the same volume used for scaling the rate constant (higher-order synthesis reactions are not currently supported). In the above rule, mRNA1 is placed in the same compartment as the DNA, which is always NU. Specifying a different localization for mRNA1 would override this behavior.

3.4.5.2 Scope-restricted rules Scope restriction limits application of a rule to species in a particular compartment through the specification of localization in reactant patterns. An example of a scope-restricted reaction rule in the compartmental model of Sec. 3.2 is

```
R11:  TF(d~pY)@CP -> TF(d~Y)@CP   k_TF_dephos
```

which specifies that dephosphorylation of transcription factors can only take place in the cytoplasm, which might be the case if an implicitly modeled phosphatase were localized there.

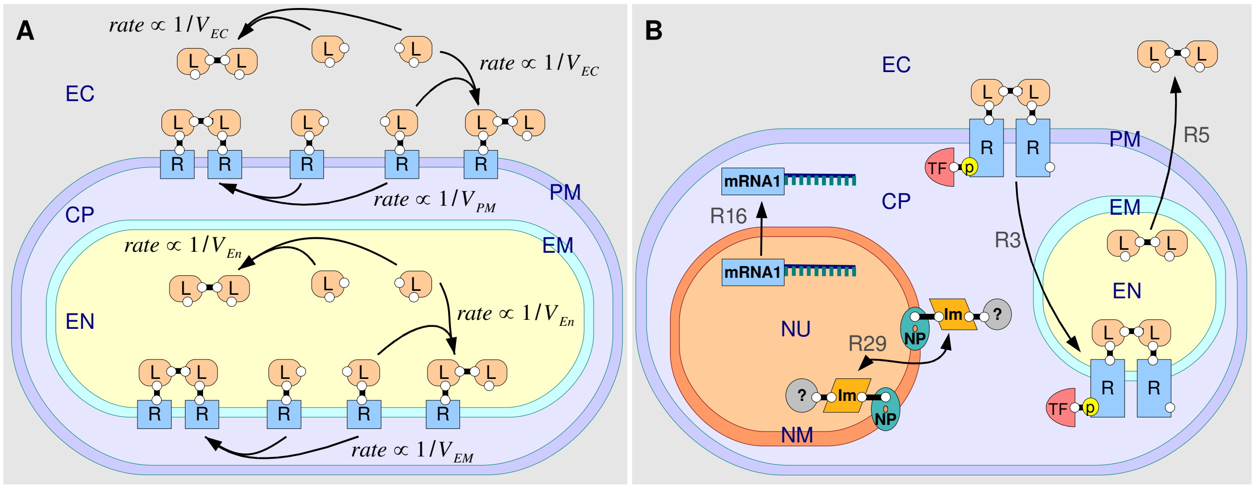


Figure 3.4: **Universal rules and transport in cBNGL** (A) Six instances of the universal rule for ligand dimerization. The single rule describes: (i) free ligand dimerization in the extracellular space (EC) and (ii) in the endosomal space (EN); (iii) free ligands binding to receptor-bound ligands at the plasma membrane (PM) and (iv) at the endosomal membrane (EM); and (v) receptor-bound ligand dimerization at PM and (vi) at EM. The rates for the individual reaction instances are automatically scaled by the volume of the reaction compartment (see Sec. 3.4.5.1). (B) Four types of transport reactions. (i) `R5: @EN:L -> @EC:L`. Volume-to-volume species transport. Free ligand dimers are recycled from the endosome to the extracellular space. (ii) `R3: @PM:R.R -> @EM:R.R`. Surface-to-surface species transport. Receptor-dimer complexes are internalized by endocytosis. Molecules in the dimer complex are transported along with the dimer. Molecules in PM are transported to EM, molecules in the adjacent EC are transported to EN and molecules in CP remain in CP. (iii) `R16: mRNA@NU -> mRNA@CP`. Single molecule transport. (iv) `R29: Im@CP.NP <-> Im@NU.NP`. Molecule transport with cargo. Importin bound to a nuclear pore is transported into the nucleus along with any bound cargo (specified by `MoveConnected` keyword; not shown).

In scope-restricted rules it is possible for a reactant pattern to match a species that upon application of the rule would create a product that does not match the product pattern. For example, consider a scope-restricted version of the ligand dissociation rule:

```
@PM:L(d!1).L(d!1) -> @PM:L(d) + @PM:L(d)
```

The pattern on the left-hand side matches receptor-ligand complexes that contain either one or two membrane-bound receptor molecules. Only those containing two receptor molecules, however, will result in two product species localized to PM after the bond between ligands is broken. BioNetGen checks that products of rule application match the product patterns, aborting the application if they do not.

3.4.5.3 Transport rules There are currently four classes of transport rule supported by cBNGL (see Fig. 3.4B): (i) molecule transport, (ii) cargo-carrying (trafficking) transport, (iii) volume-to-volume species transport, and (iv) surface-to-surface species transport.

Molecule transport is the simplest of these: the compartment designations of explicitly specified molecules are changed upon application of the rule. cBNGL allows molecules to transport volume-to-volume across a surface, volume to adjacent surface, and surface to adjacent volume. Surface-to-surface molecule transport is not permitted. Application of a molecule transport rule is rejected if the transport results in a bond that spans non-adjacent compartments. The rule in the compartmental model of Sec. 3.2 that transports mRNA from the nucleus to the cytoplasm is

```
R16: mRNA1@NU -> mRNA1@CP   k_mRNA_to_CP
```

In cargo-carrying transport, movement of a specified molecule causes simultaneous transport of connected molecules that are in the same compartment. Many cellular systems contain molecules that can bind and transport cargo between compartments, and these may bind a variety of molecules and complexes. In nuclear transport, for example, importin proteins bind to different protein molecules and escort them through the nuclear pore [112]. This type of molecule transport behavior is designated by adding the `MoveConnected` keyword to a molecule transport rule, as in the example

```
R29: Im(fg!1)@CP.NP(fg!1) <-> Im(fg!1)@NU.NP(fg!1) \
        k_Im_cross_NP, k_Im_cross_NP   MoveConnected
```

Here, an importin molecule in CP, bound through its `fg` site to a nuclear pore (NP), is transported into and out of NU along with its cargo, which can be a TF dimer or a P1. Thus, the `MoveConnected` keyword allows one to write a single rule that encompasses all possibilities without enumerating them. A precise description of the effect of a `MoveConnected` declaration requires the following definition:

Definition 2 (Compartment-connected component).
For a molecule `M` in compartment `C`, the compartment-connected component is the set of all molecules within the complex `X`, of which `M` is a member, that are connected to `M` by a path fully contained within `C`.

Only molecules in the compartment-connected component of the explicitly transported molecule(s) in a rule are co-transported. Molecules that are in the same compartment as the trafficking molecule but are connected to it through a path that passes through another compartment are not co-transported. This situation may arise because trafficking molecules often bind to surface molecules that may already be bound to other molecules in the same compartment as the trafficking molecule. Species transport rules change the locations of all molecules within a species rather than just a single molecule. These rules are specified using prefix compartment notation. Volume-to-volume species transport is straightforward: the compartment designations for all molecules in the volume-localized species are changed to the same destination compartment. An example is ligand recycling from EN to EC (see Fig. 3.4B):

```
R5: @EN:L -> @EC:L   k_recycle
```

Surface-to-surface species transport is more involved because species localized to surfaces can contain molecules that reside in adjacent volume compartments. These molecules must be correctly assigned to a new compartment during transport. In cBNGL, surface-to-surface transport rules map molecules in volume compartments in a manner consistent with endo-

and exocytosis (see Fig. 3.1). During endocytosis, molecules exterior to the invaginating membrane move into the newly-formed compartment, whereas molecules interior to the invaginating membrane remain in the same compartment. Surface-to-surface species transport is defined as follows:

Definition 3 (Surface-to-surface species transport). *A species transport rule* `@S1:P -> @S2:P`*, where* `P` *is any pattern and* `S1`*,* `S2` *are surface compartments separated by volume compartment* `V`*, transports the molecules in the complex that matches* `P` *as follows:*

1. *Molecules located in the originating surface* `S1` *move into the destination surface* `S2`*.*
2. *Molecules located in the volume* `V` *remain in* `V`
3. *Molecules located in* `V1` *(the compartment adjacent to* `S1` *but not* `S2`*) move into compartment* `V2` *(the compartment adjacent to* `S2` *but not* `S1`*)*

This type of transport is restricted to surface pairs that share an adjacent volume compartment. In the example of Sec. 3.2, receptor endocytosis (internalization) is modeled by the rule

```
R3: @PM:Rec.Rec -> @EM:Rec.Rec   k_R_endo
```

3.4.6 Comparison with BNGL

We have developed both BNGL and cBNGL versions of the compartmental model of Sec. 3.2. Each produces a network of 78 species and 354 reactions, giving identical numerical results (data not shown). The BNGL specification requires 45 rules whereas the cBNGL specification requires 30 rules, with most of the difference coming from the rules for ligand dimerization, receptor-ligand binding, endosome recycling and nuclear transport. Complete input files for this model system along with instructions for downloading a cBNGL version of BioNetGen can be found at `www.bionetgen.org/wsc09`.

3.5 DISCUSSION

cBNGL introduces a compartment topology composed of membranes and volumes, a new molecule property, *molecule localization*, which ties molecules to a compartment, and a new global property of species, *species localization*, which allows queries to determine whether a species is tethered to a membrane or freely diffusing in solution. The new `@` syntax allows a modeler to construct species and pattern matches in the context of the compartment topology. The new concepts of species and molecule localization provide a basis for the construction of new semantics for universal and transport rules.

Universal rules, which do not query localization explicitly, describe reactions that can occur between any set of matching species that are able to interact. Such rules reflect the usual biological scenario where the co-localization of reactants is a sufficient condition for the reaction to proceed. In cases where a reaction occurs in specific locations, the modeler may scope restrict a pattern in a rule to a single compartment by adding location context. The compartment to use in scaling of bimolecular reaction rates is determined from the dimensionality of the reactant compartments: a surface compartment is used only when both reactants are localized to the membrane; otherwise, the three-dimensional compartment is used. In both cases the rate is divided by the compartment volume, which naturally yields a large rate enhancement when both reactants are localized to the membrane [113, 114].

Transport rules allow the expression of a wide variety of biological transport phenomena. Molecule transport enables the description of simple diffusion across a membrane, translocation mediated by a membrane transporter, and insertion of a molecule into a membrane. Species transport rules allow the transport of an entire species based on a pattern match to part of the species complex. Volume-to-volume species transport rules allow concise representation of biological scenarios where transport across a membrane is facilitated by an escort molecule. Cargo-carrying transport coupled with a binding reaction at the membrane permits chaperone mediated species transport with a saturable rate. Surface-to-surface species transport rules allow transport of membrane-bound complexes to and from a contained membrane compartment in a manner the preserves complex structure and correct topological relationships. A common example of such a process in biology is receptor-mediated endocytosis, where an active receptor complex initiates vesicle formation and internalization.

3.5.1 Related work

Formal modeling in cellular systems has focused on three related, but largely disjoint, areas of biological expression: biochemistry of structured molecules, membrane-mediated biochemistry, and dynamic membrane systems. Rule-based modeling languages κ-calculus and BNG focus on the biochemistry of structured molecules but lack a natural approach to compartments and membranes [60, 68]. The Stochastic Simulation Compiler (SSC), another rule-based platform, allows static compartments, diffusion between compartments, and modeling of spatial and geometric effects, but lacks the notion of a membrane [119]. BioCham, a modeling platform with facilities for model checking, describes biochemistry through rules over unstructured objects and includes compartments and basic transport [79].

Several platforms have been constructed with a focus on membrane-mediated reactions. Cytosim, a formal language for describing reactions in the context of a membrane structure, includes a syntax and rule set that describe integral and peripheral membrane proteins, membrane recruitment reactions, and transport [120]. Cytosim's reactions, however, are limited to transformations of unstructured objects. Little b, a modular framework for biological modeling, includes rule-based modeling features and static membrane structures [117]. Molecules can be localized to the cytosolic or extracellular face of membranes and interact with molecules in the adjacent volume. Rules are provided for basic molecular transport reactions. Simmune, a multiscale platform tying rule-based molecular interactions to macroscopic cell behavior, models cells as a plasma membrane containing cytosolic compartment [115]. Membrane proteins in Simmune may have cytosolic and extracellular domains that interact with the adjoining volumes. While Simmune includes dynamic cell division and death, the topology of the cell is fixed and cellular models that required nested compartments for organelles and not handled.

Cardelli pioneered the formal description of dynamic membrane systems with the introduction of Brane Calculi [121, 122], which formally express biological membrane processes, including division, fusion and phagocytosis, but do not include description of biochemical reactions. BioAmbients, based on the stochastic π-calculus and ambients calculus, provides a language and simulation platform for dynamic compartments with structured molecular interactions [123]. It thus begins to bridge the biology of membranes and molecules, but treatment of membrane-mediated biochemistry is lacking. Similarly, Beta Workbench implements dynamic compartments through the abstraction of beta binders, an interface that wraps a collection of biological processes and controls external communication [116]. Nesting of beta binders is not permitted, so modeling cells with organelle structures is beyond its scope.

Recent efforts in the process algebra literature have made considerable progress in merging rule-based modeling with dynamical membrane systems. Laneve and colleagues [124] proposed bio-κ, a restricted variant of the κ-calculus combined with syntax and semantics for dynamic membrane reactions. Membranes can contain molecules and interact with adjacent volumes. While the rule-based capability of bio-κ is limited, the biological expressivity includes phagocytosis, fusion, fission and molecule translocation. Damgaard et al. [125] developed C-calculus, which introduces the concept of a channel, a type of bond that connects topologically related volumes. C-calculus is able to express complex biological phenomena that combine membrane and molecule interactions, such as clathrin-dependent vesicle formation. It can also describe chaperone-mediated transport. Though rich in their expressive capabilities, the primary shortcomings of these approaches are their current lack of freely-available end-user simulation software.

In Table 3.1, we compare the capabilities of cBNGL with the various modeling approaches cited above.

Although none of the platforms cover the full range of capabilities, cBNGL has the advantage of being able to describe a wide variety of biological phenomena associated with compartmentalization and membranes in a language that is fully compatible with the freely-available BioNetGen suite of modeling and simulation tools.

Table 3.1: **Comparison of modeling software for compartmental biochemical systems.**

Platform	Rule-based	Species transport	Membranes	Dynamic compartments	Software
Beta Workbench [116]	✓		limited	✓	✓
bio-κ [124]	✓		✓	✓	
BioAmbients [123]	✓			✓	✓
BioCham [79]	limited			✓†	✓
Brane calculi [121, 122]			✓	✓	
C-calculus [125]	✓	✓	✓	✓	
cBNGL [111]	✓	✓	✓		✓
Cyto-Sim [120]			✓		✓
Little b [117]	✓		✓		✓
Simmune [115]	✓		limited	✓	✓
SSC [119]	✓				✓

† Allows compartment volumes to change dynamically but cannot create nor delete compartments.

3.5.2 Limitations

While cBNGL encompasses a wide variety of membrane and transport phenomenon, there are important limitations. The compartment topology is static, so cell division, vesicle budding and fusion are not described. Thus, surface-to-surface transport rules capture only the average rates of molecular transport and not the turnover of individual vesicles. Compartment volumes are also fixed, so dynamic changes in volume cannot be described. Furthermore, transport reactions are not universal but tied to specific, named compartments. In a biological setting, transport typically depends on the presence of specific channels, pores or transport proteins in the separating membrane rather than the physical properties of the specific compartments. As a practical matter, we also note that our implementation of cBNGL for the particle-based simulation methods referred to in Sec. 3.1 is in progress. Future work will address each of these issues.

4.0 HYBRID PARTICLE/POPULATION SIMULATION METHOD

Network-based approaches to modeling and simulating biochemical reaction systems, such as those based on ordinary differential equations, require enumerating all species and reactions that can potentially exist in a system. The applicability of these approaches is limited, however, by the problem of combinatorial complexity, an explosion in the number of species and reactions due to protein-protein interactions and post-translational modifications in biochemical systems. Rule-based modeling (RBM) avoids this problem by representing molecules as structured objects and encoding their interactions as pattern-based rules. Rule-based models can, on the one hand, be used to generate fully-enumerated reaction networks or, alternatively, simulated directly using particle-based kinetic Monte Carlo methods. The latter "network-free" approach produces exact stochastic trajectories with a computational cost that is independent of network size. However, memory and run time costs increase linearly (or worse) with the number of particles being simulated, limiting the size of system that can be feasibly handled. Here, we present a hybrid particle/population simulation method that combines the best attributes of both the network-based and network-free approaches[1]. The method takes as input a rule-based model augmented with a user-specified subset of species to treat as population variables rather than as particles. The model is then transformed by a process of partial network expansion into a form that can be simulated using a population-adapted network-free simulator. We have implemented the transformation method within the open-source rule-based modeling platform BioNetGen and the resulting

[1] This chapter is a adapted from a manuscript with co-authors Leonard Harris and James Faeder. My contribution to the work includes development of the method, software implementation, analysis, and composition of the draft manuscript. Leonard Harris and James Faeder contributed to algorithm design and composition of the final manuscript. The manuscript was under review when this dissertation was deposited. A preprint is available in the arXiv (http://arxiv.org/abs/1301.6854).

hybrid model can be simulated using the particle-based simulator NFsim. Benchmark tests show that significant memory savings can be achieved using the new approach and a monetary cost analysis provides a practical measure of its utility. We also consider the possibility of applying accelerated-stochastic simulation methods, such as τ leaping, to the population subnetwork of the hybrid model in order to improve run time efficiency.

4.1 INTRODUCTION

4.1.1 Rule-based modeling

Cell signaling encompasses the collection of cellular processes that sample the extracellular environment, process and transmit that information to the interior of the cell, and regulate cellular responses. In a typical scenario, molecules outside of the cell bind to cognate receptors on the cell membrane, resulting in conformational change or clustering of receptors. A complex series of protein binding and biochemical events then occurs, ultimately leading to the activation or deactivation of proteins that regulate gene expression or other cellular processes.

A typical signaling protein possesses multiple interaction sites with activities that can be modified by direct chemical modification and by the effects of modification or interaction at other sites. This complexity at the protein level leads to combinatorial complexity at the level of signaling networks [44], posing a major barrier to the development of detailed and comprehensive models using standard approaches that require explicit enumeration of all species and reactions in a network [44,54,126]. This problem of scalability has motivated the development of rule-based modeling languages, such as the BioNetGen language (BNGL) [60, 127] and Kappa [67,68], which provide a rich yet concise description of signaling proteins and their interactions. The biochemical state-space explosion problem is avoided by representing interacting molecules as structured objects and using rules to encode their interactions. In the graph-based formalisms of BioNetGen and Kappa, molecules are represented as graphs and biochemical interactions by graph-rewriting rules. A rule can define either a single reaction or a class of reactions that share a common set of transformations (e.g., formation of a bond between molecules) and system-state requirements (e.g., that one or more components have a particular covalent modification). Rules are *local* in the sense that only the properties of the reactants that are transformed or are required for the transformation to take place affect their ability to react.

An important characteristic of rule-based models is that they can encode both finite and *infinite* reaction networks. If the network is finite and "not too large" then the network can be generated algorithmically [59–61, 66, 127] and simulated using a variety of network-based simulation methods (e.g., Refs. [25,37,38]). However, the rule-based methodology also provides a formalism for simulating models with prohibitively large or infinite state spaces. Such a "network-free" approach involves representing molecular complexes as particles and applying rule transformations to those particles at runtime using a kinetic Monte Carlo update scheme [55,57]. This avoids enumeration of combinatorially-large state spaces because only the current configuration of the system and its potential transformations are tracked. As a result, network-free methods can efficiently simulate complex systems beyond the reach of network-based methods [53, 55, 57]. However, because every molecule in the system is instantiated, network-free methods may require large amounts of computational memory, a potential limiting factor for simulating large systems such as the regulatory networks of a whole cell [5, 128]. A typical eukaryotic cell, for example, contains on the order of 10^3–10^4 protein-coding genes, 10^4–10^5 mRNA molecules, and 10^9–10^{10} protein molecules [8, 129], along with much larger numbers of metabolites, lipids, and other small molecules. Simulating a cell at this level of detail using network-free methods would be impractical. There is a need, therefore, to develop approaches that reduce the memory requirements of network-free methods.

4.1.2 Computational complexity

Network-based exact-stochastic simulation methods, like Gillespie's stochastic simulation algorithm (SSA) [25, 27, 28], treat species as lumped variables with a counter (i.e., a population). Several variants with different time complexities exist. The cost scaling in the number of reactions is of special interest for combinatorial models, where the number of reaction can become very large. The direct method [25] has linear time cost (per event) in the number of reactions. The Gibson-Bruck “next reaction” method [29] has logarithmic cost under the assumption that the number of dependencies per reaction is bounded by d. However, d often increases with the number of reactions. In that case, the scaling is $\mathrm{O}\big(d \log R\big)$, where R is the number of reactions. The logarithmic classes approach [30] has constant cost under the same assumptions (d is bounded); but when d is not bounded, the scaling is linear in d. The total time cost per simulation is given by the product of the time cost per event and the number of events. The number of events depends on the number of reactions, rate constants, number of particles, and volume. There is no simple generalization, but the total number of events scales linearly with simulated volume if total concentrations are fixed, i.e. particle number increases proportionally with volume. For “typical” models, the direct method with optimizations often outperforms methods with lower asymptotic complexity [31]. The space cost of SSA methods is either $\mathrm{O}\big(R\big)$ if all reactions are updated at each step (direct method), or $\mathrm{O}\big(d \cdot R\big)$ if a dependency tree is implemented (Gibson-Bruck). Most notable for our purposes, the space cost is constant in the number of particles.

Network-free methods, on the other hand, have a time complexity that depends on the number of rules rather than the number of reactions. The time cost per reaction event is approximately linear in the number of rules [53]. This explains their utility for models that encode very large or infinite networks with a relatively small number of rules. However, being exact-stochastic methods themselves, their total time complexity remains linear (or worse for gel-phase systems [55,130]) as the simulated volume increases (again, total concentrations are fixed, i.e. the number of particles increases proportionally with the reactor size). Moreover, since network-free methods represent every molecule in a system as an individual particle, the space complexity is linear in the number of particles. Thus, the efficient time complexity for network-free algorithms with respect to network size is gained at the expense of an increase in space complexity with respect to particle number. This limits the size of system that can be feasibly simulated using network-free methods. In Table 4.1, we summarize the time and space complexities of the SSA and network-free algorithms with respect to particle number and network/rule set size.

Table 4.1: **Time and space complexity for the SSA and network-free algorithms.** Scaling is shown with respect to particle number, P, and number of reactions, R, or rules, $\widetilde{R}$. Time complexity is shown per event. Total time cost of a simulation depends on the product of cost per event and number of events. The number of events in fixed length simulation depends on a number of factors, including rate constants, number of particles, and reactor volume; thus, it is difficult to generalize. Complexity of SSA also depends on d, the average number of dependencies per reaction. Since d often increases with R in combinatorial models, the assumptions of logarithmic and constant time algorithms do not always hold.

	Particles (P)	**Rxns** (R) *or* **Rules** ($\widetilde{R}$)
SSA *Time (per event)*	$O(1)$	$O(R)$[a], $O(d \log R)$[b], $O(d)$[c]
SSA *Space*	$O(1)$	$O(R)$[d], $O(d \cdot R)$[e]
NF *Time (per event)*	$O(1)$, $O(P)$[f]	$O(\widetilde{R})$
NF *Space*	$O(P)$	$O(\widetilde{R})$

[a]direct method [25, 31]
[b]Gibson-Bruck [29]
[c]logarthmic classes [30]
[d]no dependency graph
[e]dependency graph
[f]polymerizing systems in gel phase [55, 130] (see Figure 4.4B).

4.1.3 Combining network-based and network-free approaches

The key idea pursued in this work is that memory consumption can be reduced in network-free simulators by representing large numbers of identical molecular complexes as single variables with population counters rather than as many individual particles. However, retaining the ability to address combinatorial complexity requires retaining the particle representation for complexes that are comprised of many molecules and/or have complex substructures. Here, we present an approach, termed the hybrid particle/population (HPP) simulation method, that accomplishes this. Given a user-defined set of species to treat as population variables, the HPP method partially expands the network around the population species and then simulates the partially-expanded model using a population-adapted particle-based method. By treating complex species as structured particles, HPP capitalizes on the near-constant time complexity with respect to network size characteristic of the network-free approach. However, for the subset of species treated as population variables, we take advantage of the constant memory requirements of the network-based methods.

The paper is organized as follows. First, we provide brief descriptions of the example systems and performance metrics that we use to quantify the utility of the new method. We then present the HPP simulation method, which is the main contribution of this paper. Results of various performance analyses follow, which demonstrate that significant reductions in memory usage are possible using the new approach with little effect on simulation run time. Finally, we conclude with a discussion of the practical consequences of our work and some possible enhancements to the method.

4.2 METHODS

4.2.1 Example models

We have tested the performance of the HPP method by applying it to four example models, summarized in Table 4.2 and discussed in further detail below. All of the models are biologically relevant and are either taken directly from the literature or are based on models that are

Table 4.2: **Summary of example models used to test HPP performance.** Number of particles is for a NFsim simulation of a full cell volume ($f = 1$). Fractional cell volumes as low as 0.001 and as high as 1 are used in the performance analyses (see below for details).

Model	Rules	Reactions	Species	Particles ($f = 1$)	Population species	Rules after PNE	`t_end` (s)
TLBR [45, 53, 130]	4	∞	∞	5.3×10^6	2	9	500
Actin [53, 131]	21	∞	∞	1.2×10^6	2	25	1000
FcϵRI [47, 53, 132]	24	58 276	3744	6.9×10^6	1 / 6	25 / 38	2400
EGFR [19, 52, 133]	113	415 858	18 950	2.2×10^6	29	159	1200

taken from the literature. Complete BNGL model files, as well as HPP (partially-expanded) versions, are provided as Texts S4–S12 of the supporting information.

4.2.1.1 Trivalent-ligand bivalent-receptor The trivalent-ligand bivalent-receptor model (TLBR) is a simplified representation of receptor aggregation following multivalent ligand binding. TLBR has biological relevance to antigen-antibody interaction at the cell surface, where bivalent IgE-FcϵRI receptor complexes aggregate in the presence of multivalent antigen [45]. A theoretical study of the TLBR system was presented by Goldstein and Perelson [45], who derived analytical conditions for a solution-gel phase transition in terms of binding equilibrium constants, free ligand concentration, and receptors per cell. A more recent study considered the effects of steric constraints and ring closure on the solution-gel phase transition in the TLBR system [130].

Despite its simplicity, the TLBR system experiences a state-space explosion near the solution-gel phase boundary. A computational study by Sneddon et al. [53] reproduced the analytical results of Goldstein and Perelson using the BNGL-compliant network-free simulator NFsim [53]. Due to large excesses of ligand and receptor under certain conditions, TLBR is a natural test case for HPP. We simulated the TLBR system using HPP with free ligand and receptor treated as population species. All simulations were performed with parameters as defined in Monine et al. [130], which lie within the solution-gel coexistence region. A cell-scale simulation assumed 1 nl extracellular volume per cell (10^6 cells/ml) with 8.3 nM ligand and 3×10^5 receptors per cell. Simulations were performed at fractional cell volumes, f, ranging from 0.001 to 0.1 with a lumping rate constant `k_lump` = 10 000/s (see below). Results are shown in Figure 4.4.

4.2.1.2 Actin polymerization Actin polymerization plays a key role in cell morphology and motility [134, 135]. Roland et al. [131] presented a dynamic model of actin polymerization featuring filament elongation by monomer addition, stabilization by ATP hydrolysis, and severing mediated by actin depolymerizing factor (ADF)/cofilin. Sneddon et al. [53] presented a rule-based formulation of the Roland et al. model and replicated their results using NFsim. The model features an excess of actin monomer and ADF molecules. Therefore, we speculated that substantial memory reduction is possible using the hybrid approach. We applied HPP to the Sneddon et al. rule-based model of actin dynamics (hereafter referred to as the Actin model) with actin monomer and ADF treated as population species. A cell-scale simulation assumed 1 pl intracellular volume with 1 μM actin monomer and 1 μM ADF/cofilin. Simulations were performed at fractional cell volumes, f, ranging from 0.01 to 1 with a lumping rate constant `k_lump` = 10 000/s. Results are shown in Figure 4.5.

4.2.1.3 FcϵRI signaling FcϵRI is a membrane receptor that binds IgE antibodies. Signaling through FcϵRI regulates basophilic histamine release in response to IgE antibody-antigen interaction [136]. Faeder et al. [46,47] developed a rule-based model of FcϵRI receptor assembly and activation in which receptor dimerization/clustering is mediated by chemically cross-linked IgE, which serve as multivalent ligands. Dimerized receptors are also transphosphorylated, leading to Syk and Lyn recruitment and phosphorylation. Sneddon et al. [53] presented several extensions of the Faeder et al. model, including the *gamma2* variant with two γ phosphorylation sites. Particle-based NFsim simulations of the *gamma2* model were found to be substantially faster than network-based SSA simulations.

Due to the excess of free ligand, the HPP method was applied to the *gamma2* model to reduce memory consumption. The method was applied with two different sets of population species. In the first case, only free ligand was treated as a population species (FcϵRI:1). In the second, cytosolic Lyn and all four phosphorylation states of cytosolic Syk were also treated as populations (FcϵRI:6). A cell-scale simulation assumed 1 pl intracellular volume with 1 nl extracellular space per cell (10^6 cells/ml), 10 nM ligand, and 4×10^5 receptors per cell. Simulations were performed at fractional cell volumes, f, ranging from 0.001 to 0.1 with a lumping rate constant `k_lump`$=$10 000/s. Results are shown in Figure 4.6.

4.2.1.4 EGFR signaling

A model of signaling through the epidermal growth factor receptor (EGFR), beginning with ligand binding and concluding with nuclear phospho-ERK activity, was constructed by combining three existing models: (i) a rule-based model of EGFR complex assembly [52]; (ii) a Ras activation model [133]; (iii) a pathway model of Raf, MEK and ERK activation [19]. Ras activation was coupled to the EGFR complex assembly by treating receptor-recruited Sos as the Ras GEF. Activated Ras was coupled to the Raf/MEK/ERK cascade through RasGTP-Raf binding and subsequent phosphorylation of Raf. Parameters for the combined model were obtained from the respective models. However, parameters governing Ras-GEF (i.e., Sos) activity had to be changed from their original values [133] in order to account for the known GEF-mediated activation of Ras [137]. Specifically, we used $K_{M,\mathrm{GDP}} = K_{M,\mathrm{GTP}} = 1.56\times10^{-7}$ M and $D = 1000$ (unitless).

Free EGF and Raf-, MEK-, and ERK-based species were treated as population species in the hybrid variant. Ras-based species were also treated as populations except for those that include a Sos molecule. A cell-scale simulation assumed 0.94 pl cytosolic and 0.22 pl nuclear volume, with 0.94 pl extracellular space, 10 nM ligand, and 4×10^5 receptors per cell. Simulations were performed at fractional cell volumes, f, ranging from 0.01 to 1 with a lumping rate constant `k_lump`$=$100 000/s. Results are shown in Figure 4.7.

4.2.2 Performance metrics

HPP was evaluated for peak memory use, CPU run time, and accuracy as compared to particle-based NFsim simulations. For models where network generation is possible (FcϵRI and EGFR), comparisons were also made to SSA simulations (as implemented in BioNetGen [60]). All benchmarks were performed on a 2 $\times$ Intel Xeon E5520 @ 2.27 GHz (8 cores, 16 threads, x86_64 instruction set) with 74 GB of RAM running the GNU/Linux operating system. To ensure that each process had access to 100% of the compute cycles of a thread, no more than 12 simulations were benchmarked simultaneously. In all cases, reported results are based on $\geq$7 independent simulation runs.

4.2.2.1 Peak memory Peak memory usage was evaluated by peak virtual memory allocation reported by the operating system with the command: `cat /proc/<PID>/status`. For all benchmark models, peak memory was achieved early in the simulation and remained steady throughout (data not shown).

4.2.2.2 CPU run time CPU run time was evaluated using clock time as a metric. Clock time was recorded using the `Time::HiRes` Perl module. Run time included initialization as well as the simulation phase. Partial network expansion for HPP simulations was a one time cost, typically a few seconds, and was not included in the benchmark.

4.2.2.3 Accuracy Simulation accuracy was quantified using several approaches. For all models, the total number of reaction firings was recorded for each simulation run (firings of population-mapping rules were subtracted from the total in HPP simulations). Based on 40 independent samples each of NFsim and HPP (and SSA for the FcϵRI model), we tested the null hypothesis that none of the methods produce a larger number of firings under the Mann-Whitney U test [138, 139]. Distributions were visualized as box plots showing minimum values, maximum values, and quartiles.

For the TLBR and Actin models, equilibrium distributions for key observables were also compared. These include the number of receptor clusters in the TLBR model and the length of actin polymers in the Actin model. 10 000 samples were collected over 100 000 seconds of simulated time. Distributions were compared by binning samples (20 bins) and performing a two-sample chi-squared test [140]. For the FcϵRI and EGFR models, trajectories for key observables were collected from 40 simulation runs each. Moving averages and 5–95% frequency envelopes were plotted for sampling windows of 10 s. Due to complications of autocorrelation, a statistical test was not applied to the dynamic trajectory comparison. Instead, the results were plotted for inspection by eye.

4.2.3 Software implementation

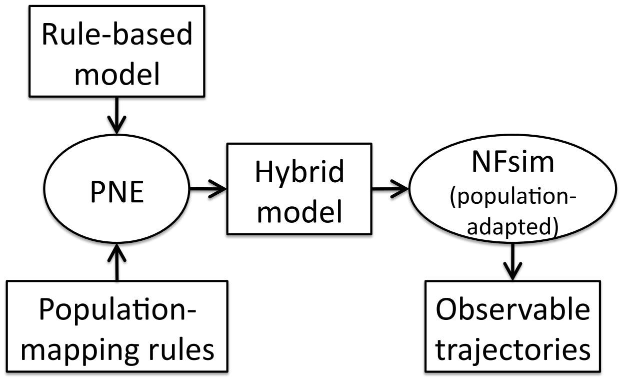
HPP and NFsim simulations were run using NFsim version 1.11, which is available for download at `http://emonet.biology.yale.edu/nfsim`. All simulations (SSA included) were invoked through BioNetGen version 2.2.4, which implements the hybrid model generator and is distributed with NFsim 1.11 (see Refs. [60, 61] and Secs. S3.1.5 and S4.1 of Text S1 for details on running simulations with BioNetGen). NFsim and BioNetGen source code are available at `http://code.google.com/p/nfsim` and `http://code.google.com/p/bionetgen`, respectively. The most recent implementation of the hybrid model generator can be found in the `BioNetGen-2.2.4-stable` distribution (see `generate_hybrid_model()` in the `BNGAction.pm` module at `http://code.google.com/p/bionetgen/source/browse/bng2/Perl2/BNGAction.pm`). Additional documentation for BioNetGen can be found at `http://bionetgen.org`.

4.3 RESULTS

4.3.1 A hybrid particle/population simulation approach

In this section, we first present an algorithm for transforming a rule-based model into a partial-network system. We then describe a population-adapted network-free simulation protocol that can be applied to that system. We refer to the combination of these methods as the hybrid particle/population (HPP) simulation method. The basic work flow is

Figure 4.1: **Basic workflow of the HPP simulation method.** Given a rule-based model and a user-specified set of population-mapping rules (which define the population species), partial network expansion (PNE) is performed to generate a hybrid version of the original model. The hybrid model is then passed to the population-adapted version (1.11 or later) of the network-free simulator NFsim, which generates the time-evolution trajectories for all observable quantities specified in the original model.



provided in Figure 4.1. The transformation method has been implemented within the open-source rule-based modeling package BioNetGen [60, 61, 127] and the resulting hybrid model can be simulated using version 1.11 (or later) of the network-free simulator NFsim [53]. The reduction in computational memory requirements offered by HPP is an important step towards the practical simulation of systems approaching the size and complexity of a whole cell [5, 128].

For convenience, we will adhere in this paper to the BNGL syntax, which is summarized in Section 1.4.3 of the introductory chapter. A complete description of the syntax and semantics of BNGL can be found in Refs. [60, 61, 111]. For the theoretical foundations of BioNetGen see Ref. [66] and Appendix B.

4.3.1.1 Population species and population-mapping rules

Given a rule-based model, the first step in the HPP approach is to select a subset of species to treat as "lumped" population variables. Currently, this is left as a task for the modeler (an optimal strategy for selecting species for population lumping is a topic for future research; see "Discussion"). Generally speaking, however, species that maintain a large population throughout the duration of the simulation are good candidates. It is also best to choose species that contain only a small number of components and/or participate in only a few reaction rules, since the time complexity of the simulation algorithm is strongly correlated with the size of the expanded rule set (Table 4.1). A receptor with many phosphorylation and binding sites, for example, is usually best treated as a particle because treating it as a population species would require enumerating a large number of species and reactions through partial network expansion (Section 4.3.1.2).

The next step is to map each (structured) species selected for lumping to an associated *unstructured* species, which we refer to as a *population species*. Population species differ from standard BNGL species in that they possess a property, called a *count*, which records their current population. The mapping is accomplished using a *population-mapping rule*, which follows the same syntactic conventions as a standard BNGL rule. For example,

```
Egf(r) -> pop_Egf() k_lump
```

maps the unbound EGF ligand `Egf(r)` to the unstructured population species `pop_Egf()`. The rate parameter here is termed the *lumping rate constant*. The basic idea is that during the course of a simulation complexes may dissociate and release instances of structured species that have been selected for lumping, `Egf(r)` in this case. The population-mapping rules gather these instances back up into the population species. The lumping rate constant thus determines how long these unlumped particles persist in the system. Its value should ideally be set to infinity as the HPP method is formally exact only for an infinite lumping rate. However, if infinity is not an option in the network-free simulator being used (currently it is not in NFsim) then setting it to a value that is "large" with respect to the model dynamics will suffice (see Figs. 4.4–4.7, panels C and D). Note that the action of the above rule is to delete the `Egf(r)` molecule and *increment by one* the counter of the `pop_Egf()` molecule (only a single instance of a population species ever exists in the simulation environment).

4.3.1.2 Partial network expansion The ultimate goal of the HPP method is to replace in the simulation environment large numbers of indistinguishable particles with small numbers of lumped objects containing population counters (the population species), thus significantly reducing memory usage. In order to accomplish this without losing information regarding the dynamics of the system, we must partially expand the rule set until all reactions in which the population species appear as reactants are enumerated. We refer to this procedure as partial network expansion (PNE).

The PNE algorithm is comprised of three basic steps, applied to each reaction rule in a model:

1. For each reactant pattern, identify all matches of the pattern into the (structured) species selected for lumping. Also collect a self-match of the reactant pattern *unless* it equals a species selected for lumping (see below).
2. Derive an expanded set of rules by applying the original rule to all possible combinations (the Cartesian product) of the reactant pattern matches collected in step 1.
3. For each derived rule, replace each reactant and product pattern that equals a species selected for lumping with its unstructured population counterpart.

The result is an expanded rule set consisting of three general types of rules: (i) particle rules, comprised of reactant patterns that exclusively match particles; (ii) mixed particle/population rules, which operate on a mixture of particles and population species; (iii) pure population *reactions*, which operate exclusively on population species. The expanded rule set has the property that every possible action of the original rules on the population species is enumerated while actions on particle objects remain pattern-based (i.e., non-enumerated). A formal basis for the PNE algorithm, with pseudocode, is provided in Appendix E.

Note that the set of particle rules in the expanded rule set is just the set of reaction rules in the original model, less those that exclusively match structured species selected for lumping. This is a consequence of excluding self-matches in Step 1 of the PNE algorithm for reactant patterns that equal structured species selected for lumping. The reason why this is done is that, for large `k_lump`, there will *almost never* exist any instances of the structured species in the system. As mentioned above, the role of the population-mapping rules is to gather instances of structured species selected for lumping up into the population species. A large value of `k_lump` means that this will be accomplished almost instantaneously. As such, rules involving those reactant patterns need not be retained since they will (essentially) never fire. Removing them decreases the size of the partially-expanded rule set and improves the efficiency of the method. Note, however, that for the sake of completeness we have implemented an "exact" PNE method in BioNetGen (version 2.2.3 or later) that retains *all* of the self-matches and, hence, gives exact results for any value of `k_lump` (see Appendix F for instructions on how to call this method). For a select number of examples, we have confirmed that both methods give statistically identical results for sufficiently large `k_lump` and that the exact method is less efficient (data not shown).

PNE is best illustrated through an example. In Figure 4.2, we present a simple rule-based model of receptor activation (for brevity, parameters, initial populations, and output observables are omitted; see Appendix G.1 for the complete model in BNGL format). The model includes a ligand, `L`, its cognate receptor, `R`, and three cytosolic proteins, `A`, `B`, and `C`, that are recruited to the phosphorylated receptor. The 16 rules (six unidirectional and five reversible), describing ligand-receptor binding, receptor phosphorylation/dephosphorylation, and protein recruitment, encode a reaction network comprised of 56 species and 287 reactions. In applying the HPP method, eight species are selected for lumping: free ligand, free `A`, `B` and `C`, and complexes of `A`, `B` and `C` that exclude the receptor. Receptor complexes are treated as particles because there are many possible receptor configurations (48 total).

In Figure 4.3, a step-by-step application of PNE to rule 11f (forward) of Figure 4.2 is presented. First, both reactant patterns are matched to the structured species selected for lumping. Reactant pattern 1 has one match, while reactant pattern 2 has two. Note that since neither reactant pattern exactly equals a species (i.e., is isomorphic to one) the self match (identity automorphism) is added to the reactant match list in both cases. This is required for generating mixed particle-population rules, where one reactant is an unstructured population species and a second is a structured particle.

Next, the rule is applied to each possible reactant set (the Cartesian product of the reactant match lists). This results in a set of six derived rules. The structured species are then replaced in these rules by their associated unstructured population species, resulting in

Figure 4.2: **Simple receptor activation model in BNGL format.** Abridged; see Appendix G.1 for the complete model.

	Molecule Type	*Description*
1	`L(r)`	extracellular ligand
2	`R(l,a~0~P,b~0~P)`	membrane receptor with two phosphorylation sites
3	`A(r,b~0~P)`	cytosolic molecule (recruits to phosphorylated receptor)
4	`B(r,c)`	cytosolic molecule (recruits to phospho-receptor or phospho-A)
5	`C(b)`	cytosolic molecule (binds to B)

	Reaction Rule	*Rate constant(s)*	*Description*
1	`L(r) + R(l) <-> L(r!1).R(l!1)`	kp1,km1	receptor-ligand binding
2	`L(r!1).R(l!1,a~0) -> L(r!1).R(l!1,a~P)`	k2	site `a` phosphorylation
3	`L(r!1).R(l!1,b~0) -> L(r!1).R(l!1,b~P)`	k2	site `b` phosphorylation
4	`R(a~P) -> R(a~0)`	k3	site `a` dephosphorylation
5	`R(b~P) -> R(b~0)`	k3	site `b` dephosphorylation
6	`R(a~P) + A(r) <-> R(a~P!1).A(r!1)`	kp4,km4	phosphorylated `R` binding `A`
7	`R(b~P) + B(r) <-> R(b~P!1).B(r!1)`	kp5,km5	phosphorylated `R` binding `B`
8	`B(c) + C(b) <-> B(c!1).C(b!1)`	kp6,km6	`B` and `C` binding
9	`R(a~P!1).A(r!1,b~0) -> R(a~P!1).A(r!1,b~P)`	k7	recruited `A` phosphorylation
10	`A(b~P) -> A(b~0)`	k8	`A` dephosphorylation
11	`A(b~P) + B(r) <-> A(b~P!1).B(r!1)`	kp9,km9	phosphorylated `A` binding `B`

Population-mapping rules:

	Structured species		*Population species*	*Lumping rate constant*
1	`L(r)`	`->`	`P1()`	`k_lump`
2	`A(r,b~0)`	`->`	`P2()`	`k_lump`
3	`A(r,b~P)`	`->`	`P3()`	`k_lump`
4	`A(r,b~P!1).B(r!1,c)`	`->`	`P4()`	`k_lump`
5	`A(r,b~P!1).B(r!1,c!2).C(b!2)`	`->`	`P5()`	`k_lump`
6	`B(r,c)`	`->`	`P6()`	`k_lump`
7	`B(r,c!1).C(b!1)`	`->`	`P7()`	`k_lump`
8	`C(b)`	`->`	`P8()`	`k_lump`

Figure 4.3: **Partial network expansion (PNE) applied to Rule 11f of Figure 4.2.**

Reaction rule: `A(b~P) + B(r) -> A(b~P!1).B(r!1) kp9`

Description: "Bind component sites b~P and r"

1: Find Reactant Pattern (RP) Matches

	Matches to RP 1	*Population species*		*Matches to RP 2*	*Population species*
1	`A(b~P)`	identity automorphism	1	`B(r)`	identity automorphism
2	`A(r,b~P)`	`P3()`	2	`B(r,c)`	`P6()`
			3	`B(r,c!1).C(b!1)`	`P7()`

2: Rule Expansion: Apply rule to each reactant set in the Cartesian product of reactant matches

1	`A(b~P) + B(r)`	`->`	`A(b~P!1).B(r!1)`	kp9
2	`A(b~P) + B(r,c)`	`->`	`A(b~P!1).B(r!1,c)`	kp9
3	`A(b~P) + B(r,c!1).C(b!1)`	`->`	`A(b~P!2).B(r!2,c!1).C(b!1)`	kp9
4	`A(r,b~P) + B(r)`	`->`	`A(r,b~P!1).B(r!1)`	kp9
5	`A(r,b~P) + B(r,c)`	`->`	`A(r,b~P!1).B(r!1,c)`	kp9
6	`A(r,b~P) + B(r,c!1).C(b!1)`	`->`	`A(r,b~P!2).B(r!2,c!1).C(b!1)`	kp9

3: Rule Rewriting: Substitute structured species graphs with unstructured population species

1	`A(b~P) + B(r)`	`->`	`A(b~P!1).B(r!1)`	kp9
2	`A(b~P) + P6()`	`->`	`A(b~P!1).B(r!1,c)`	kp9
3	`A(b~P) + P7()`	`->`	`A(b~P!2).B(r!2,c!1).C(b!1)`	kp9
4	`P3() + B(r)`	`->`	`A(r,b~P!1).B(r!1)`	kp9
5	`P3() + P6()`	`->`	`P4()`	kp9
6	`P3() + P7()`	`->`	`P5()`	kp9

one pure particle rule (the original rule), three mixed particle/population rules, and two pure population reactions. Including the population-mapping rules, the hybrid model contains a total of 42 rules, more than the original 16 but significantly fewer than the 287 reactions of the fully expanded network. The complete partially-expanded HPP model for this system can be found in Appendix G.2.

4.3.1.3 Population-adapted network-free simulation Since an HPP model is properly a rule-based model, network-free simulation algorithms are readily adapted to the HPP method by the addition of: (i) a population count property for each molecule object; (ii) a transformation that performs population increments and decrements; (iii) a method for calculating population-weighted propensities (rates). The population-weighted propensity of an expanded rule $\widetilde{R}_\mu$ is given by the formula

$$a_\mu = \frac{k_\mu}{s_\mu} \prod_{r=1}^{M_\mu} \left(\sum_{x=1}^{X} \rho(x) \eta_{\mu,r}(x) \right), \tag{4.1}$$

where k_μ is the rate constant, s_μ is the symmetry factor (see Ref. [60, *Note 4.21*]), M_μ is the number of reactant patterns in the rule (i.e., the *molecularity*), X is the total number of complexes in the system, $\rho(x)$ is the population of complex x (unity in the case of particles), and $\eta_{\mu,r}(x)$ is the number of matches of reactant pattern r into complex x.

Note that in the case of symmetric population reactions [e.g., `pop_A() + pop_A() -> A(a!0).A(a!0)`], the possibility of a null event must be calculated in order to prevent self reactions. This is accomplished by rejecting the event with probability $1/\rho(x)$. Furthermore, since population species have zero components, if complex x is a population species and $\eta_{\mu,r}(x) = 1$ (can only equal 0 or 1), then $\eta_{\mu,r}(y) = 0$ for all $y \neq x$. This property is useful because it guarantees that a reactant pattern matches either particles or population species exclusively, never a mixture of both. Thus, once a rule has been selected to fire, the particles to participate in that rule can be selected from a uniform distribution rather than from a population-weighted distribution.

Figure 4.4: **HPP performance analysis for the TLBR model.** (A) peak memory usage (*left*: absolute, *right*: relative to NFsim); (B) CPU run time (*left*: absolute, *right*: relative to NFsim); (C) number of reaction events fired during a simulation ($f = 0.01$); (D) equilibrium distribution of number of clusters ($f = 0.01$).

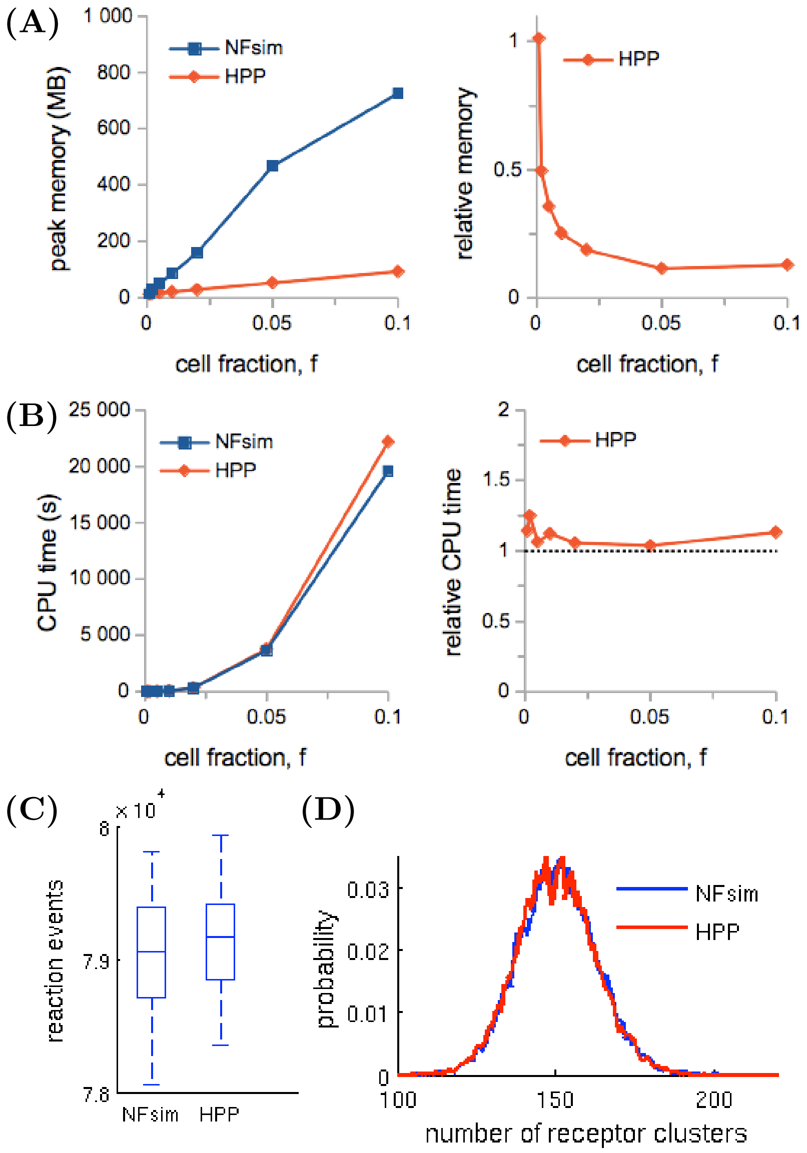


Figure 4.5: **HPP performance analysis for the actin polymerization model.** (A) peak memory usage (*left*: absolute, *right*: relative to NFsim); (B) CPU run time (*left*: absolute, *right*: relative to NFsim); (C) number of reaction events fired during a simulation ($f = 0.01$); (D) equilibrium distribution of actin polymer lengths ($f = 0.01$).

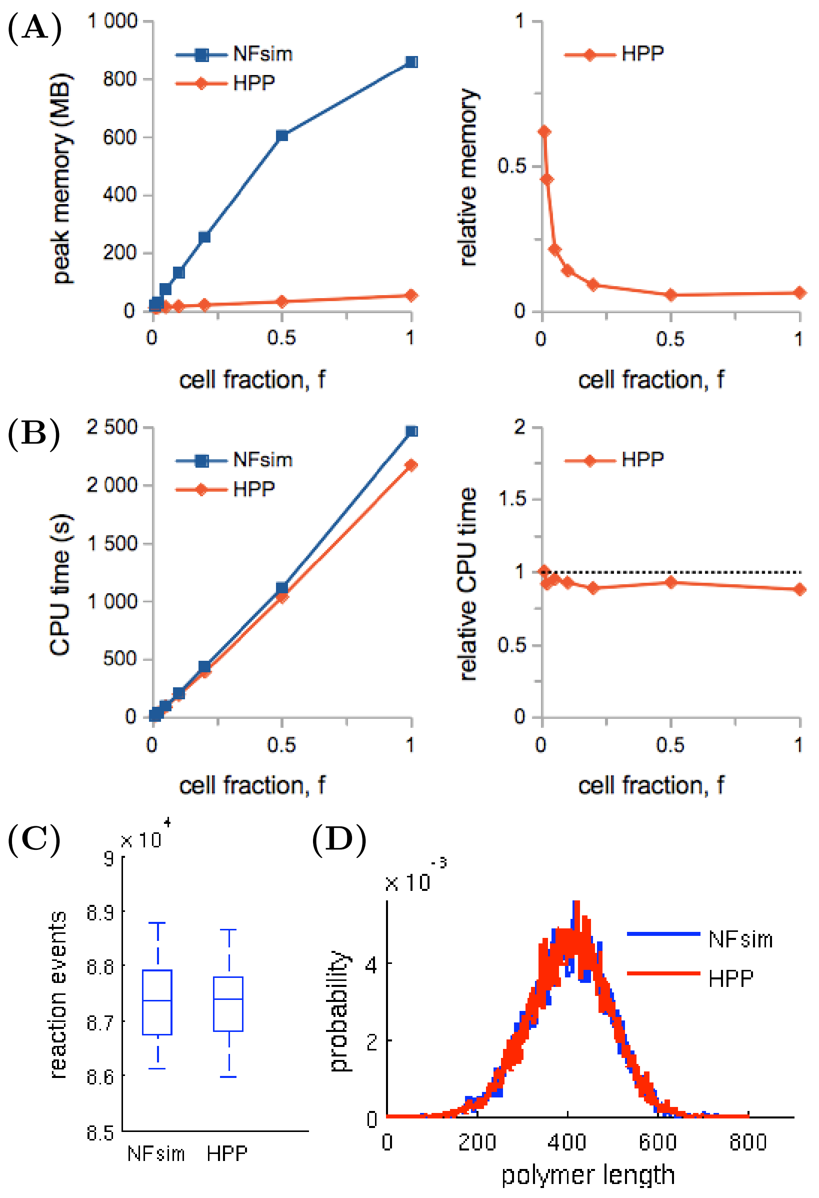


Figure 4.6: **HPP performance analysis for the FcϵRI signaling model.** (A) peak memory usage (*left*: absolute, *right*: relative to NFsim); (B) CPU run time (*left*: absolute, *right*: relative to NFsim); (C) number of reaction events fired during a simulation ($f = 0.01$); (D) time courses (means and 5–95% envelopes; $f = 0.01$) for γ-phosphorylated receptor (*top*) and receptor-recruited, α-phosphorylated Syk (*bottom*). SSA time courses are virtually indistinguishable and have been omitted for clarity.

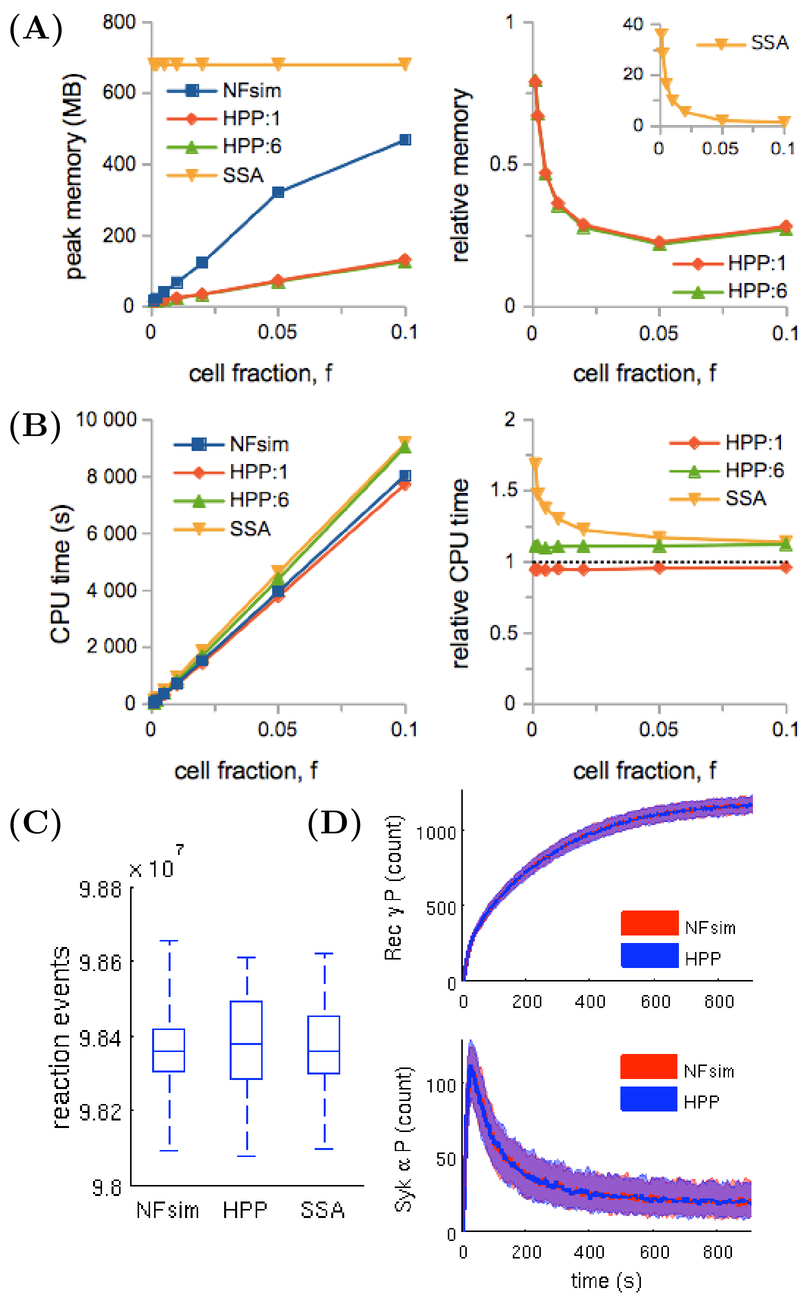
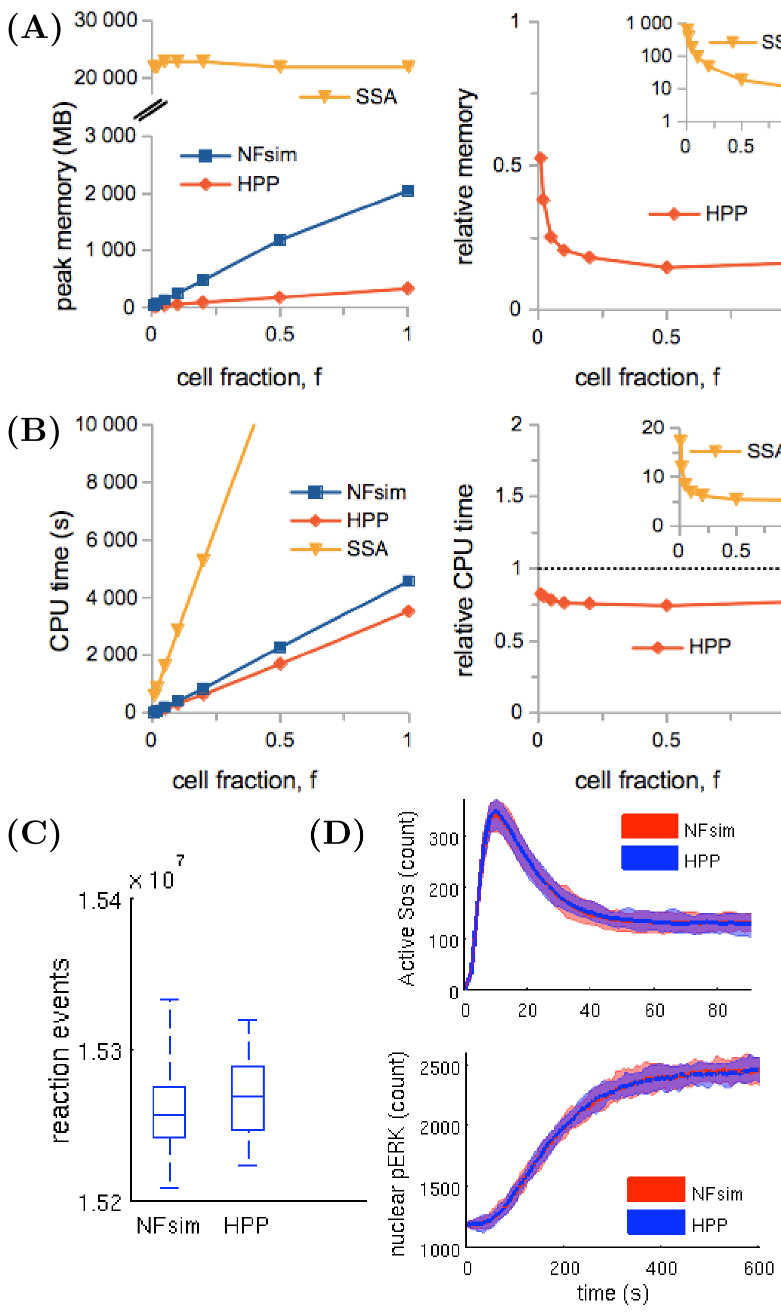


Figure 4.7: **HPP performance analysis for the EGFR signaling model.** (A) peak memory usage (*left*: absolute, *right*: relative to NFsim); (B) CPU run time (*left*: absolute, *right*: relative to NFsim); (C) number of reaction events fired during a simulation ($f = 0.05$); (D) time courses (means and 5–95% envelopes; $f = 0.05$) for activated Sos (*top*) and phosphorylated ERK (*bottom*). Due to high computational expense, SSA statistics were not collected in (C) and (D).



4.3.2 Performance analyses

4.3.2.1 Peak memory use and CPU run time Figures 4.4–4.7, panels A, show absolute and relative (with respect to NFsim) peak memory use as a function of cell fraction, f, for all models considered. We see that in all tested cases HPP requires less memory than NFsim. For NFsim, we see the expected linear relationship (Table 4.1) between peak memory use and system size (see below). For HPP, peak memory use also scales linearly but with a smaller slope. This is expected because as the system size increases a portion of the added particles, and hence memory cost, is always absorbed by the population subnetwork. We also see that the relative memory benefits of HPP go to zero at the smallest cell fractions. This is because opportunities for compression decrease as species populations approach zero. In cases where network generation is possible (FcϵRI, Figure 4.6A; EGFR, Figure 4.7A), we see for the SSA the expected constant relationship between memory usage and system size (Table 4.1). The SSA also requires more memory than both NFsim and HPP for all cell fractions considered. This is due to the high memory cost of the dependency update graph [31] (standard in SSA implementations), which scales with the product of the number of reactions in the network and the average connectivity per reaction.

Note that, for NFsim, the second-to-last points in the absolute memory usage plots (panels A, *left*) actually lie slightly above the linear trend in all cases. This is due to the fact that memory is allocated in NFsim in logarithmically growing intervals (a common approach in software design) up to a threshold, after which allocation occurs in intervals of a constant size. Basically, at the second-to-largest volumes memory allocation is still in the logarithmic phase, where greater excesses of memory are likely to be allocated (the largest volumes are in the constant phase). To address this, we have rerun simulations for all models at all cell fractions using a custom compiled version of NFsim that allocates memory in smaller intervals. The data confirm that the memory scaling for NFsim with system size is in fact linear (data not shown).

Figures 4.4–4.7, panels B, show absolute and relative (with respect to NFsim) CPU run times as a function of cell fraction. Generally speaking, HPP and NFsim run times are comparable in all cases, indicating that the reductions in memory use seen in Figs. 4.4–

4.7, panels A, are not achieved at the cost of increased run times. In fact, HPP is slightly faster than NFsim in most cases. This is because operations on population species (e.g., increment/decrement) are less costly than the graph operations applied to particles (e.g., subgraph matching). Also note in Figure 4.4B the expected quadratic relationship between run time and system size for the TLBR model (Table 4.1), which is due to the solution-gel phase transition [55, 130]. In Figs. 4.6B and 4.7B, we see that the SSA is slower than both NFsim and HPP for all cell fractions considered. The difference is most pronounced at small cell fractions and is much more significant for EGFR than for FcϵRI. This is as expected since previous work [53] has shown that network-free methods significantly outperform network-based stochastic methods when population levels are small and network sizes are large (the EGFR network is significantly larger than the FcϵRI network; Table 4.2).

Finally, we see in Figure 4.6B that the CPU run time increases as we increase the number of species treated as populations in the FcϵRI model, even though the memory usage remains constant (Figure 4.6A). This is interesting because it suggests that the FcϵRI:1 variant, with free ligand as the only population species, is near-optimally lumped for the cell fractions considered. Evidently, the population levels of the lumped cytosolic Lyn and Syk species in the FcϵRI:6 variant never get large enough to provide significant memory savings (any savings that are achieved are offset by the additional mixed rules and reactions added to the rule set).

4.3.2.2 Accuracy Figures 4.4–4.7, panels C, show distributions of the number of reaction firings per simulation run for each of the simulation methods considered. For all models, the distributions, as illustrated by box plots, are similar for NFsim, HPP, and SSA (the latter for FcϵRI only; Figure 4.6C). The two-sided Mann-Whitney U test [138, 139] was unable to reject the null hypothesis at the 5% significance level for all models (TLBR: $p = 0.25$; Actin: $p = 0.90$; FcϵRI: $p = 0.27$; EGFR: $p = 0.07$), providing strong evidence that HPP produces statistically identical numbers of reaction firings to both NFsim and SSA.

Figures 4.4–4.7, panels D, compare distributions obtained from NFsim and HPP simulations of all models. In Figure 4.4D, we show equilibrium distributions of the number of receptor clusters in the TLBR model ($f = 0.01$). In Figure 4.5D, equilibrium distributions of polymer lengths in the Actin model are shown ($f = 0.01$). In both cases, we could not reject the null hypothesis of the equivalence of the NFsim and HPP distributions (TLBR: $p = 0.50$; Actin: $p = 0.66$). In Figure 4.6D, time courses for γ-phosphorylated receptor and receptor-recruited, α-phosphorylated Syk are shown ($f = 0.01$). In Figure 4.7D, time courses for membrane-recruited (active) SOS and nuclear phospho-ERK are presented ($f = 0.05$). Although we did not perform any statistical tests, visual inspection of the trajectories shows that in all cases the NFsim and HPP results are virtually identical.

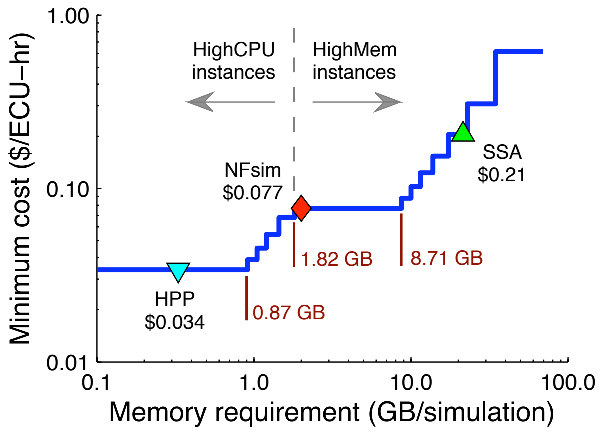
4.4 DISCUSSION

We have presented a hybrid particle/population simulation approach for rule-based models of biological systems. The HPP approach is applied in two stages (Figure 4.1): (i) transformation of a model into an equivalent hybrid form by partially expanding the network around a selected set of population species; (ii) simulation of the transformed model using a population-adapted network-free simulator. The method is formally exact for an infinite population lumping rate constant, but can produce statistically exact results in practice provided that a sufficiently large value is used (Figs. 4.4–4.7, panels C and D). As currently implemented, the primary advantage of the HPP method is in reducing memory usage during simulation (Figs. 4.4–4.7, panels A). Importantly, this is accomplished with little to no impact on simulation run time (Figs. 4.4–4.7, panels B).

4.4.1 Monetary cost analysis

In order to frame our results within a real-world context, we have estimated the cost of simulation based on hourly rates of *on-demand instances* on the Amazon Elastic Compute Cloud (EC2). In Figure 4.8, we show the hourly cost (per "effective compute unit") of simulation as a function of required memory per simulation (details of the calculation can be found in Appendix D). Also included in the plot are values for HPP (0.3 GB), NFsim (2.1 GB), and SSA (22.0 GB) simulations of the EGFR model at cell fraction $f = 1$ (Figure 4.7A).

Figure 4.8: **Cost of running simulations on the Amazon Elastic Compute Cloud (EC2).** The minimum cost as a function of memory requirement was calculated based on January 2012 pricing of all *Standard*, *High-CPU*, and *High-Memory* EC2 instances [141]. Also included are values for NFsim, HPP, and SSA simulations of the EGFR model at cell fraction $f = 1$.



The EC2 offers various "instance types" with different CPU and memory resources, including *High-CPU* and *High-Memory* varieties. Our calculations show that below 1.82 GB of required memory High-CPU instances are the most cost effective. Above this threshold High-Memory instances are the better option. We see that the HPP simulation falls below this cutoff while both NFsim and SSA lie above. There is a quantifiable benefit, therefore, to reducing memory usage in this case. HPP simulations on the EC2 would be $\sim$2.5 and $\sim$33 times less expensive than NFsim and SSA, respectively (HPP is slightly faster than NFsim and significantly faster than SSA; see Figure 4.7B). The whole-cell EGFR model thus provides a tangible illustration of the benefits of reducing memory load for dynamic simulations of large biochemical models, approaching the scale of a whole cell.

These benefits are further accentuated by the fact that common model analysis techniques such as parameter estimation (e.g., Ref. [142]), sensitivity analysis (e.g., Ref. [143]), and statistical model checking (e.g., Ref. [144]), typically entail running a large number (thousands or more) of independent simulation trials. These trials can be distributed across a computer cluster and run independently, which is efficient because there is little need for interprocess communication. However, typical hardware configurations have a fixed number of processor cores with a large, common pool of memory. If each independent process consumes a large amount of memory, then it may be impossible to fully utilize each CPU core. Reducing memory requirements can thus also improve the effective use of cluster and cloud-based computing resources.

Finally, realistic simulations of whole-cell dynamics require accounting for the highly crowded and inhomogeneous environment of the cell interior. A common approach is to discretize a system into multiple well-mixed subvolumes, with reactions taking place within, and diffusion occurring between, each subvolume [145, 146]. However, doing so can significantly increase the number of objects (species, reactions, rules, etc.) that must be represented and retained in memory during the course of a simulation. The ability to reduce memory usage is thus of particular value in the case of spatial simulations as well. We expect that HPP can be successfully applied to spatial models without modification and can provide several-fold reductions in memory use without substantial impact on run time.

4.4.2 Future directions

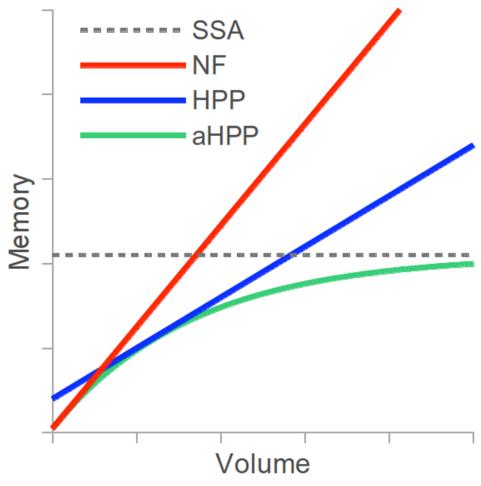
We have shown that peak memory use for HPP scales linearly with system size with a slope that is smaller than for NFsim (Figs. 4.4–4.7, panels A). Moreover, when network generation is possible, HPP memory use at the system sizes considered is significantly less than for SSA, which has approximately constant scaling (Figs. 4.6A and 4.7A). However, the linear scaling of HPP and constant scaling of SSA means that with further increases in the system size there will invariably come a point where HPP memory use exceeds that of SSA. Intuitively, we understand that this is because species that are rare at small volumes, and hence chosen to be treated as particles, become plentiful at large volumes. Since we require that population species be selected *a priori*, there is no way to address this problem currently without intervention by the user.

We propose to develop an enhanced version of HPP that tracks the populations of particle species and automatically lumps them into population species when their number exceeds a certain threshold. PNE would be performed every time a new species is lumped, meaning that the memory load would never exceed that of the fully-expanded network. In Figure 4.9, we provide a qualitative sketch of how the memory usage of such an “automated HPP” (aHPP) method would scale with system size for a model with a finite network. Included for comparison are HPP, NFsim, and SSA scalings as well. For infinite networks (such as TLBR and Actin), we expect aHPP memory use to scale somewhere between constant and linear (no worse than HPP) at large volumes, depending on the model.

Furthermore, the cost analysis of Figure 4.8 illustrates that if, in addition to reducing memory usage, simulation run times can be decreased, then even greater monetary benefits are possible. There has been much research into methods for accelerating stochastic simulations of chemical reaction networks. These include τ leaping [28], originally proposed by Gillespie [32] and improved upon by Gillespie and others [33, 34, 37, 147–151]. The HPP method provides a unique opportunity for the application of τ-leaping approaches because, unlike in pure particle-based methods, there exists a partial network of reactions that act on population species. We believe that it would be relatively straightforward to integrate a τ-leaping method into the HPP by applying it exclusively to the population subnetwork while retaining the network-free approach in the particle-based component. We have recently implemented a τ-leaping variant in BioNetGen, known as the partitioned-leaping algorithm [37], and are actively working on integrating it with the HPP.

Importantly, the integrated τ-leaping/HPP method will involve calculating different time steps for the particle and population subcomponents of the system and setting the global time step, τ, to the smaller of the two. As such, if the fastest dynamics in the model are contained within the particle-based component, then accelerations in the population subnetwork will be limited. However, if the fast dynamics are due to the presence of a large population of identical particle species (i.e., suboptimal lumping), then automated lumping, i.e., via aHPP, could alleviate the problem. This further emphasizes the value of the proposed aHPP method. Optimizing both memory usage and speed of simulation will not only decrease the time to results but will impact, in a tangible way, the cost of doing computational research.

Figure 4.9: **Memory use vs. simulated volume for different simulation methods, including a hypothetical "automated HPP" (aHPP).** For finite networks, aHPP memory use plateaus once the entire reaction network has been generated. For infinite networks, the scaling at large volumes should fall somewhere between constant and linear (no worse than HPP) depending on the model.



5.0 APPLICATION: SEPSIS AND HEMOADSORPTION TREATMENT

Sepsis is a serious illness characterized by the combination of infection and systemic inflammation. Hemoadsorption (HA) is an experimental treatment for sepsis in which the patient's blood is circulated through a device packed with porous beads that adsorb a range of small molecules, including inflammatory cytokines. HA has been shown to be effective at improving survival in a rat model of sepsis [152, 153]; however, the mechanism of efficacy remains uncertain. In this chapter, I present models of sepsis and HA and attempt to elucidate a casual mechanism for treatment efficacy[1].

I proposed a causal mechanism for improved survival of sepsis with HA treatment. The hypothesis supposes that inflammatory mediators produced at the site of infection escape into the circulation, where they are transported to distant organs. Distant organs become inflamed, leading to widespread neutrophil recruitment and a reduction in circulating neutrophil levels. Lower circulating levels reduces availability of neutrophils at the site of infection. Consequently, the infection is not adequately suppressed, eventually resulting in death. HA prevents this cascade by removing inflammatory mediators from the blood, reducing systemic inflammation and preventing the drop in circulating neutrophils. Higher circulating levels leads to improved recruitment to the site of infection and promotes clearance of pathogen.

To determine the plausibility of the hypothesis, I developed a compartmental rule-based model of sepsis and HA (sepsis model 1, Section 5.3). The model includes three organ compartments, peritoneum, blood, and other tissue, that are populated by pathogen,

[1]Models and analysis presented in this chapter were developed under the supervision of Gilles Clermont and Robert S. Parker. Experimental work was performed by our collaborators in the laboratory of Dr. John Kellum. The statistical analysis herein is my own and may differ slightly from any experimental publications that result from their work.

macrophage, neutrophils. A pro-inflammatory cytokine and its inhibitor regulate the recruitment of neutrophils, which eliminate the pathogen. The HA device is modeled as a first-order removal of cytokine from the blood compartment. A set of parameters was identified for which HA treatment rescues the *in silico* animal from septic death. Simulations predicated that HA efficacy was associated with improved circulating neutrophils levels, lower systemic neutrophil recruitment, and increased neutrophil recruitment to the site of infection.

Drawing on these simulation results and other lines of evidence, experimental collaborators devised a set of experiments to measure inflammatory mediators and white blood cells in the lung, blood, and peritoneal compartment of septic rats. Experiments indicated that HA leads to significantly reduced neutrophil infiltration in the lung and, although the difference did not reach a significance threshold, increased recruitment to the peritoneum. Contrary to our simulations, circulating neutrophils were not depleted in sham animals, nor increased with HA treatment.

Since experimental results did not support a drop in circulating blood neutrophils, the original hypothesis was modified. Based on input from experimental colleagues, I hypothesized that the ratio of local to systemic chemokine was the key driver for neutrophil recruitment. Reduction of chemokines in the blood via HA increases the ratio, leading to better recruitment to the site of infection and improved survival. At the same time, lower levels of blood cytokines reduces systemic inflammation and contributes to improved survival.

To test the modified hypothesis, I formulated a model that incorporates the chemokine ratio mechanism (sepsis model 2, Section 5.5) and calibrated to experimental data. Simulations predict that HA leads to a short-term drop in circulating cytokines and chemokines, which rebound to normal levels within a few hours. Despite the fast rebound, the effect on neutrophil recruitment was sufficient to rescue the *in silico* animal from a septic death trajectory. However, the model was unable to reproduce the increase in peritoneal neutrophils observed in laboratory experiments. Thus, I conclude that while the chemokine ratio hypothesis is a plausible mechanism for improved survival with HA, further laboratory and modeling work is required to elucidate the biological mechanisms underlying sepsis and HA treatment.

5.1 SEPSIS AND INFLAMMATION: AN INTRODUCTION

Inflammation is the body's natural and essential response to infection and trauma. Inflammation recruits an innate immune response to fight off infection and initiates the adaptive immune response [154]. Macrophage cells reside in tissues and serve as sentinels. Macrophage detect pathogen and cellular damage signals through surface receptors that are complementary to conserved bacterial proteins (such as LPS) and host proteins associated with cell damage [155]. Upon detection, macrophage secrete a variety of pro-inflammatory cytokines and chemoattractants into the local interstitial space. TNF and interleukin-1 (IL-1) are two important pro-inflammatory cytokines. TNF and IL-1 alert nearby cells to the presence of danger.

Pro-inflammatory signals trigger changes in the local vasculature. Nearby blood vessels dilate and become more permeable. This physiological change increases blood flow and increases the influx of proteins that target pathogen. Endothelial cells, which express receptors for TNF and IL-1, begin to express neutrophils adhesion molecules [156]. Neutrophils are a white blood cell that circulates in the blood stream [157]. Neutrophils express molecules that bind to activated endothelial cells, causing the neutrophil to roll slowly along the vessel. Chemokines, including (IL-8), trigger tight binding of neutrophils to the endothelium. After tight binding, neutrophils migrate into the inflamed tissue following chemokine signals.

After migration, neutrophils bind pathogen via surface receptor, and then internalize and digest the pathogen (a process called phagocytosis). Neutrophils also release a variety of toxic chemicals into the interstitium that inhibit bacterial growth, but also cause damage to local tissues. Tissue damage may lead to further inflammation (positive feedback), but also triggers anti-inflammatory events to minimize host damage.

Pro-inflammatory signals trigger a subsequent wave of anti-inflammatory signals. TNF, for example, binds to receptors expressed on macrophage and activates the production of the potent anti-inflammatory IL-10 [158]. IL-10 binds to receptors on a variety of cells, including macrophage, and inhibits the production of pro-inflammatory signals. This negative feedback plays an important rule in limiting inflammation.

5.1.1 Sepsis

Properly regulated inflammation is localized to the site of infection and trauma. Due to infection in the blood or other pathological conditions, acute inflammation can become systemic, spreading throughout the body. Sepsis, a systemic inflammatory response triggered by infection, is one such pathology [159]. An estimated 751,000 incidents of severe sepsis occur annually in the United States with 51% of cases receiving treatment in the ICU and 28.6% mortality [160]. Severe sepsis may lead to the failure of multiple organ systems [160, 161]. The cascade of organ failures may occur even while the infection is suppressed by antibiotics, demonstrating that uncontrolled inflammation is destructive to organ tissues.

The mainstay of sepsis treatment has been source control, antibiotic treatment, fluid resuscitation and organ support. Recent recommendations added very few additional options to this therapeutic approach [162]. Numerous attempts at controlling the ill effects of the inflammatory response in sepsis using immunomodulation such as anti-inflammatory treatments (*e.g.* anti-TNF antibodies, low dose corticosteroids, etc.) have largely failed to improve patient outcome despite promise in pre-clinical trials; hence, there is ongoing controversy as to the merit of immunomodulation in severe sepsis [162, 163] and reticence to pursue development of interventions based on simple rationales. An opportunity exists for model-based intervention design in sepsis and other complex diseases where standard approaches have not delivered expected advances.

5.1.2 Cecal-Ligation and Puncture: animal model of sepsis

Cecal-ligation and puncture (CLP) is a laboratory model of clinical sepsis [164]. The cecum of an animal, typically a rodent, is surgically ligated and punctured with a needle [165]. The punctured cecum initiates a polymicrobial infection in the peritoneal cavity of the animal, leading to sepsis, shock, and typically death [166]. CLP has been described as the “gold standard” of animal sepsis models, although sensitivity to the parameters of the procedure result in poor reproducibility if the protocol is not performed consistently [166].

5.1.3 Hemoadsorption: a potential treatment for sepsis

Hemoadsorption (HA) is a blood purification treatment that has been shown to improve short term survival in septic rat models [152, 153]). A portion of the septic animal's blood is circulated through an *ex vivo* circuit including the HA device: a column packed with small adsorptive beads. Key inflammatory cytokines responsible for the propagation of the inflammatory response, including TNF, interleukin(IL)-6, and IL-10, are captured by the device [152]. A calibrated model of cytokine capture was presented by DiLeo et al. [167,168] There is preliminary evidence [169] that existing HA devices directly interact with circulating WBCs, the distal effectors of systemic inflammation. Research is ongoing to characterize the specific nature of the interaction between the HA device and subpopulations of WBCs.

5.2 MODELING INFLAMMATION, A SURVEY

The complexity of the inflammatory response, and the lack of clinical progress in treating sepsis, make it an ideal candidate for systems modeling. A survey of inflammation modeling approaches is presented here, with a focus on host-pathogen models of acute inflammation.

5.2.1 Equation models

The simplest models of sepsis are abstract representations of pro and anti-inflammatory feedback loop as small systems of ordinary differential equations [20,21,23]. These abstract models reproduce multiple physiological outcomes observed in clinical sepsis, including healthy resolution, septic death, and aseptic death. Though useful for theoretical purposes, calibration to experimental data is not possible since biological observables are not represented in the model.

Phenomenological models of inflammation with variables corresponding to biological entities have been constructed using larger systems (8-18 equations) of ODEs [22, 170]. The inclusion of biological observables provides a quantitative connection between the model and biology that is not possible with abstract models. The mathematical form is based on heuristics, often Michaelis-Menten or Hill-type functions, inferred from biological relations reported in the literature. Model parameters are calibrated to experimental measurements obtained in vivo. This type of model is useful for engineering applications, such as process control, where calibration and efficient simulation are required. The number of parameters in these models is large compared to the available in vivo data. Integrating supplemental data from in vitro or in vivo experiments would be ideal, but the phenomenological nature of these models limits the applicability of data obtained under different experimental conditions. ODE models are amenable to dynamic analysis techniques, but typically restricted to a few parameters of interest.

The ODE models described above do not represent spatial aspects of sepsis. Sepsis, however, is a process that occurs in the context of a complex physiological space. To reflect this fact, a course-grained representation of organ physiology was implemented in a compartmental ODE model of acute inflammation presented by Reynolds [171]. The model represents organs as well-mixed compartments connected by a vascular compartment. Cell migration and cytokine diffusion are included as mathematical transport terms that link compartments. The coarse representation of space sacrifices spatial detail to obtain rapid simulations. The Reynolds model inspired the compartmental model presented later in this chapter (Sec. 5.3).

Compartmentalized ODEs can be written by hand, but the task becomes tedious as the number of compartments increase. Reactions are typically common to many or all compartments, requiring the modeler to write similar equations repeatedly. This avoid the tedious and error-prone process of writing compartmentalized ODEs, the compartmental model can be specified with a set of universal reaction rules and a compartmental topology. Then the reaction network can be algorithmically constructed via the network generation algorithm. Compartmental BNGL, presented in Chapter. 3, is one such framework for compartmental rule-based modeling [111].

5.2.2 Agent-based models

Agent-based models (ABM) are a common choice for modeling spatial phenomenon. IN contrast to ODE models, which lump populations into a single variable, agent models represent members of the population as individuals, i.e. agents. ABMs are useful for simulating systems where the state space dimensionality is too large for network based simulation methods. A typical agent simulation includes a 2-D or 3-D space in which agents move about. One or more cell types are represented as agents, each with its own list of instructions that define motion and other cell behaviors. Additionally, background layers may be implemented to simulate biochemical gradients. Simulations are typically implemented with a fixed-time step sequence rather than exact stochastic methods. ABMs have been used to study acute inflammation [172], the formation of diabetic foot ulcers [173] and the formation of granulomas in tuberculosis [174].

ABMs have disadvantages in certain situations. Spatial simulations are often burdened by tedious computations of diffusion and agent movements. If spatial details are not the primary concern of the model, this computational overhead may be unsatisfactory. Agent models are fundamentally stochastic; therefore multiple simulations are required to determine the average behavior of the model and its variance. This compounds the computational burden, especially during parameter estimation and sensitivity analysis. ABM, which lacks the concept of a bond, cannot represent complexes of agents unless the modeler defines an agent class for each possible complex. This is a severe limitation when modeling systems with combinatorial possibilities for binding configurations (e.g. signal transduction).

5.2.3 Discrete-state models

Discrete-state models are simple and often useful for qualitative modeling. A set of discrete variables represent biological entities or behaviors, e.g. phosphorylated protein, or cell migration. In a two-state model, a 0 value indicates that the biological referent is inactive, e.g. transcription factor is not activated, while a value of 1 indicates activity, e.g. the transcription factor is active. The state of each variable is computed by a logical formula in terms of influencing variables. Discrete state models can be simulated, without further specification, using an iterative update scheme. These models can encode dynamic behaviors typical of equation models, including bistability and oscillations. But the connection to continuous time is tenuous. Discrete-state modeling has been applied to a variety of biological phenomena. Thakar et al. [175], for example, constructed a boolean model of host-level inflammation that was able to reproduce a number of qualitative observations.

The Kauffman-Glass [176] treatment uses a discrete-state model to induce a piecewise linear system (PLS) with continuous dynamics. This scheme requires a set of threshold parameters, and kinetic parameters for each partition of state-space. Dynamics are linear in each partition, but the kinetic parameters switch when the trajectory crosses a partition boundary. The PLS approach handles continuous variables and time in a more natural way than discrete models, but at the expense of additional parameters.

5.3 A RULE-BASED MODEL OF SEPSIS WITH HEMOADSORPTION

Two experiments reported in the literature motivated the development of a compartmental model of sepsis. Kellum and colleagues studied the efficacy of extra-corporeal blood purification via a hemoadsorption (HA) device for the treatment of sepsis in a CLP rat model. In the study, rats were subjected to CLP and either received 4 hours of HA treatment beginning 18 hours post CLP or sham treatment with an empty cartridge. HA treated animals had a higher survival rate and lower levels of circulating cytokines in comparison to sham treated animals [152, 153]. Despite the efficacy of HA treatment, it is unclear why cytokine capture improves survival. The authors propose that broad-spectrum filtration of cytokines reduces systemic inflammation and the risk for multiple organ dysfunction and death.

In a second study, Alves-Filho et al. compared cytokine levels and neutrophil recruitment in lethal and sublethal models of peritoneal sepsis in rats. The study found that lethal sepsis, in comparison to sublethal sepsis, is associated with increased levels of circulating cytokines, increased neutrophil recruitment to the lung, and decreased neutrophil counts in the peritoneum [177].

Building on the work of Alves-Filho et al., I developed a mechanistic hypothesis for the efficacy of HA treatment. In response to CLP, high levels of cytokines are produced and *spill over* from the site of infection into the blood. High levels of cytokines in the blood lead to poor neutrophil recruitment to the site of infection and high levels of system neutrophil recruitment. The immediate effect of HA treatment is a reduction of cytokine levels in the blood. Following the logic of Alves-Filho, this restores neutrophil recruitment to the site of infection and minimizes systemic recruitment. Thus, clearance of the infection is improved while damage to organs from systemic inflammation is minimized.

The hypothesis as stated so far is incomplete, as the link between high levels of cytokine and reduced recruitment to the site of infection is unclear. There are several possible hypotheses, but I proceeded with perhaps the simplest hypotheses: excess cytokine in the blood activates endothelium systemically, leading to wide-spread, non-specific neutrophil recruitment to tissue and depletion of circulating neutrophils. Due to lower levels of circulating neutrophils in the blood, fewer neutrophils are available for recruitment to the site of infection.

5.3.1 Model construction

In order to test the hypothesis, I developed a model of sepsis with blood, peritoneum and "other tissue" compartments. These three compartments are the minimum required to represent infection and systemic inflammation: a site of infection, distant tissues that may become inflamed, and the blood circulation, which transports mediators between tissues. The central focus of the model is recruitment of neutrophils from the blood to inflamed tissues and the subsequent elimination of pathogen by phagocytosis. Recruitment is mediated by cytokines produced in response to pathogen and the endothelial cells activated by those cytokines. I assume that the adaptive immune response is not relevant on the time scale of experiments (3 days) and may be excluded from the model.

This model, titled *sepsis model 1*, is summarized in Figure 5.1. The model is constructed in compartmental BNGL (cBNGL), a rule-based language. Tissues are represented by compartments, while cells and signaling molecules are treated as *agents* in those compartments[2].

[2]I will use the term *agent* throughout this section since the usual BNGL terminology, i.e. *molecule*, is confusing in this context.

Sepsis model 1 is composed of 27 reaction rules, corresponding to a reaction network with 134 species and 545 reactions. The model has a total of 27 free parameters. Reaction rules are depicted graphically in Figure 5.2 and the complete BNGL source code is listed in Appendix H.1.

5.3.1.1 Compartments

The model consists of three organs: blood, peritoneum, and *other tissue*. Each organ is represented by a 3-dimensional compartment in cBNGL: `B`, `P`, and `T`, respectively. The peritoneum and other tissue are separated from the blood by endothelial layers, which are represented by 2-dimensional compartments: `eP` and `eT`. The peritoneal compartment is the site of bacterial infection, while *other tissue* represents an organ that is free of active infection but may be affected by systemic inflammation. For example, *other tissue* may represent the lung. The model assumes that infection is contained in the peritoneum and does not spread to the blood or other tissue.

The volume of the organ compartments are constant during the simulation. The volume of blood in a 250g rat is about 14 ml and, assuming hematocrit of .45, the plasma volume is 7.8 ml. The effective volume of the peritoneum and other tissue are set by tunable parameters. The surface area of each endothelial layer is proportional to the volume of the respective organ compartment.

Compartments are assumed to be well-mixed. Although organs are in fact heterogeneous and diffusion limited, the well-mixed assumption is reasonable since (i) available data is not spatially resolved and (ii) every subvolume of an organ is close to the blood supply and thus similarly coupled to the blood compartment. Other modelers have chosen to apply the well-mixed assumption in similar contexts.

5.3.1.2 Agents

The model is populated by four types of cells: pathogen, tissue macrophage, endothelial cells, and neutrophils. Each cell type is represented by a BNGL agents: `Path`, `Mphage`, `Endo` and `Neutr`. The behavior of each agent is governed by a set of rules, described in the next section. The components of each cell type represent sites for cell-cell interaction, ligand binding sites, or physiological states (such as *activated* or *inactive*. The agent type block (i.e. "molecule types") is listed below.

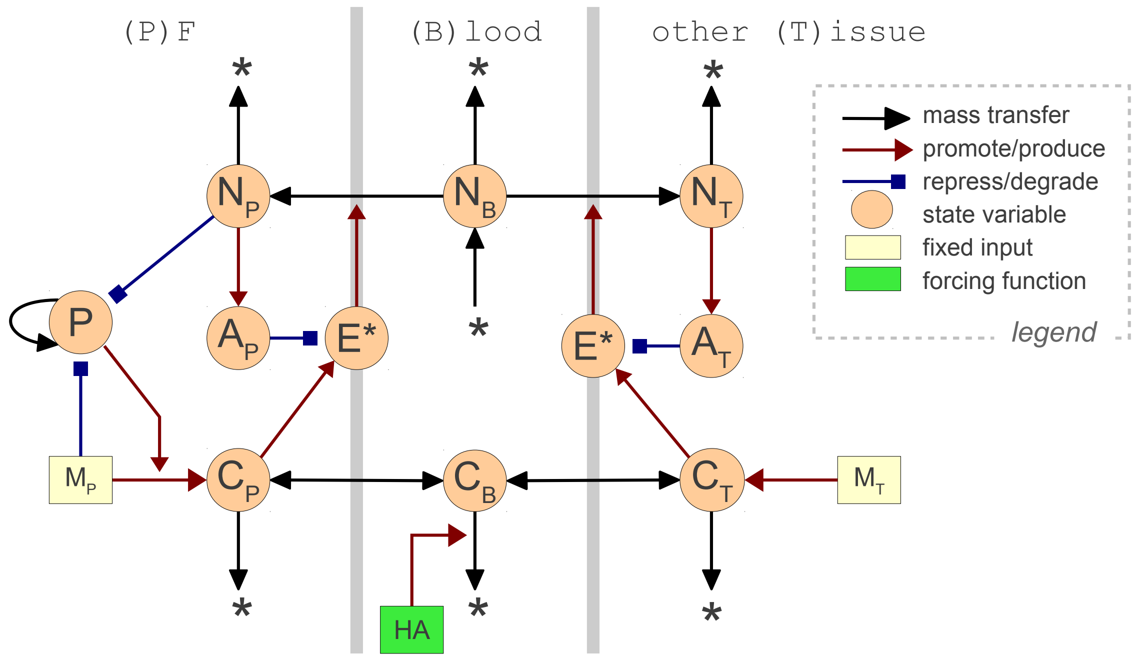


Figure 5.1: Schematic of sepsis model 1. The agents are macrophage, `Mphage` (M); endothelial cells, `Endo` (E); neutrophils, `Neutr` (N); bacterial pathogen, `Path` (P); pro-inflammatory cytokine, `IL1` (C); and IL-1 receptor antagonist, `IL1ra` (A). See Section 5.3.1 for a complete description of the model.

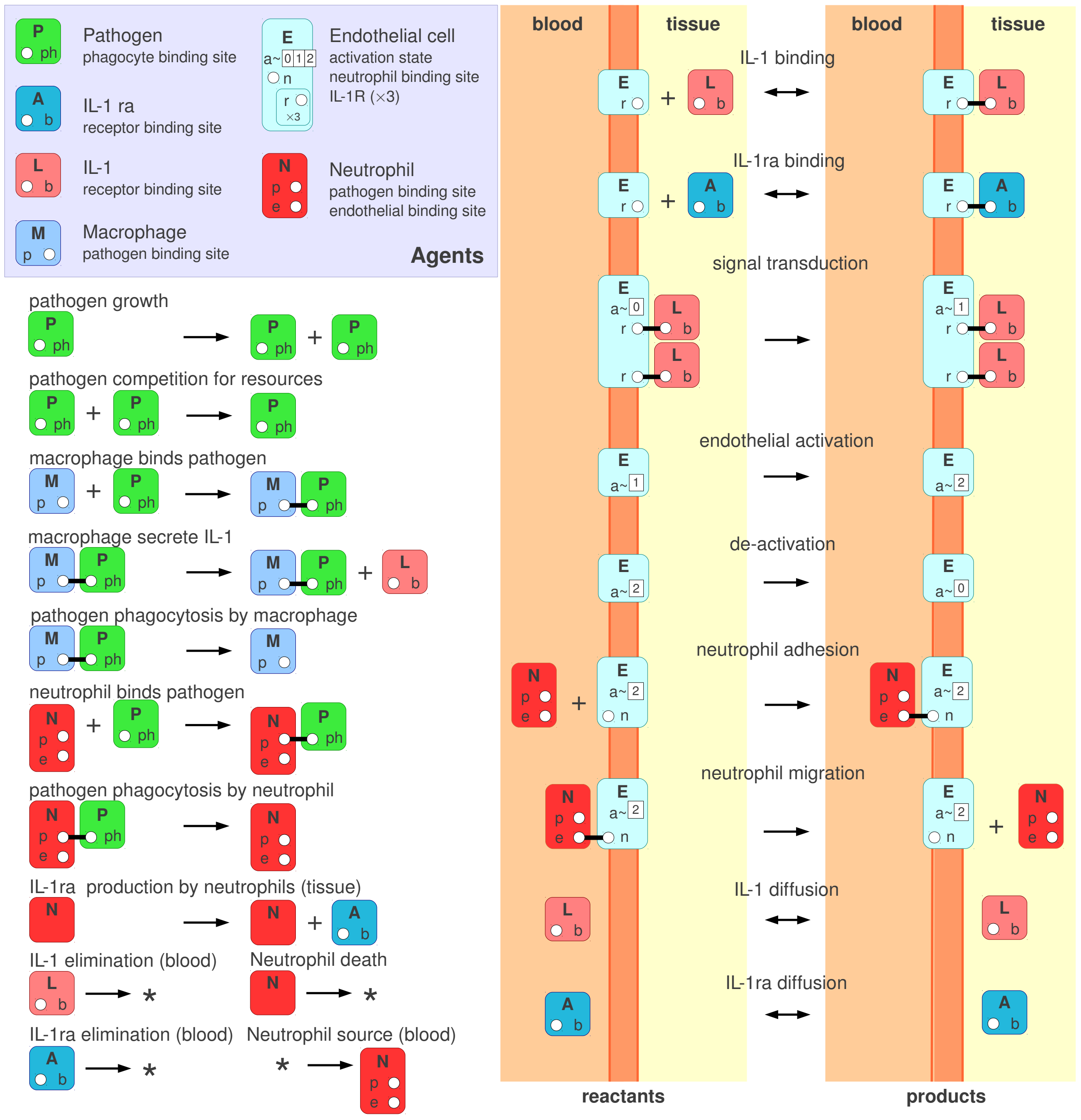

P ph Pathogen phagocyte binding site
A b IL-1 ra receptor binding site
L b IL-1 receptor binding site
M p Macrophage pathogen binding site
E a~0 1 2 n r ×3 Endothelial cell activation state neutrophil binding site IL-1R (×3)
N p e Neutrophil pathogen binding site endothelial binding site
Agents
pathogen growth
pathogen competition for resources
macrophage binds pathogen
macrophage secrete IL-1
pathogen phagocytosis by macrophage
neutrophil binds pathogen
pathogen phagocytosis by neutrophil
IL-1ra production by neutrophils (tissue)
IL-1 elimination (blood)
Neutrophil death
IL-1ra elimination (blood)
Neutrophil source (blood)
blood
tissue
IL-1 binding
IL-1ra binding
signal transduction
endothelial activation
de-activation
neutrophil adhesion
neutrophil migration
IL-1 diffusion
IL-1ra diffusion
reactants
products

Figure 5.2: A complete list of rules for sepsis model 1, depicted graphically. Each rule is composed of a set of *reactant patterns* on the left-hand side and the *product patterns* on the right-hand side. Reversible rules are indicated with a double arrow. If a component of an agent is not shown in the reactant pattern, then the state of that component does not influence the reaction.

```
begin molecule types
   Path(ph)                  # pathogen
   Mphage(p)                 # macrophage
   IL1(b)                    # ligand (IL1)
   IL1ra(b)                  # receptor antagonist (IL1ra)
   Neutr(p,e)                # neutrophil
   Endo(n,a~0~1~2,r,r,r)     # endothelial cell
end molecule types
```

Pathogen are restricted to the peritoneal compartment, where the infection is initiated at $t = 0$ with a dose pathogen. The `Path` agent contains one component `ph` that serves as a site for cell-cell interaction between the pathogen and phagocytic host cells.

Tissue macrophage, represented by the `Mphage` agent, are present in the peritoneum and other tissue, but absent from the blood. The total concentration of `Mphage` is constant during model simulations. `Mphage` has one component, `p`, that serves as site for cell-cell interaction with pathogen.

Endothelial cells, `Endo`, reside in the endothelial barriers of the peritoneum and other tissue. Like tissue macrophage, the total surface concentration is held constant during the simulation. The agent type for endothelial cells is `Endo(a~0~1~2,n,r,r,r)`. The `a` site represents the activation state of the endothelial cell. A state value of 0 represents *inactive*, 1 represents *activating*, and 2 represents a fully *activated* cell. The `n` component serves as a site for cell-cell interaction with neutrophils in the blood. Endothelial cells only interact with neutrophils after activation by `IL1` [178]. The r site represents a surface receptors that binds `IL1` and `IL1ra`. The cell contains 3 receptor sites, which permits a graded response depending on the level of ligand saturation. Endothelial activation is initiated when at least two receptor sites are occupied by `IL1`. A endothelial cell, in fact, contains thousands of receptors, but it is not feasible to represent the cell in this detail.

Neutrophils, `Neutro`, initially populate the blood compartment, which contains a source of neutrophil. Unlike other cells in the model, neutrophils can migrate to other compartments. The neutrophil molecule type is `Neutr(p,e)`, where `p` is a site for cell-cell interaction with `Path` and `e` is a site for cell-cell interaction with `Endo` cells.

The model contains two small signaling molecules: Interleukin-1, `IL1`, and its receptor antagonist, `IL1ra`. Interleukin-1 is pro-inflammatory cytokine that mediates the activation of endothelial cells [179]. IL-1ra binds to the same receptor as IL-1 and blocks signaling [180, 181]. The agent types are `IL1(b)` and `IL1ra(b)`, where `b` is the site for receptor binding. `IL1` and `IL1ra` can reside in the peritoneum, blood, and other tissue. Both molecules can "leak" between the compartments, which is the primary mechanism for the spread of inflammation in the model. Although, in fact, a large number of cytokines and chemokines organize the inflammatory response, I chose to focus on a single pro-inflammatory mediator and its *anti-inflammatory* receptor antagonist. This choice was made for two reasons. First, the network of interactions among mediators is complex and not fully understood. Consequently, calibrating a model with multiple cytokines and chemokines would not be possible without a large amount of data. Second, since the model is exploratory, a single pro-inflammatory mediator and its anti-inflammatory partner may serve as an abstract representation of the complete pro/anti-inflammatory circuitry.

5.3.1.3 Rules

The complete set of agent rules for the sepsis model are depicted in Figure 5.2. I will describe the rules for each agent type in the model.

Pathogen grow according to a logistic rate law. Logistic growth is implemented with two rules: one describing pathogen growth via cell division at a rate proportional the pathogen population, and a second describing population pressure. Pathogen are captured and eliminated by local macrophage and neutrophils (described later).

```
# pathogen rules
Path(ph) -> Path(ph) + Path(ph)   sP_P   # growth
Path(ph) + Path(ph) -> Path(ph)   dP_P   # population pressure
```

The total quantity of tissue macrophage is held constant during the simulation. Tissue macrophage irreversibly bind pathogen according to a second-order process and eliminate the pathogen at a first order rate. While bound to pathogen, macrophage produce cytokines in the local compartment.

```
# macrophage rules
Path(ph) + Mphage(p) -> Path(ph!1).Mphage(p!1)                bMP
Mphage(p!1).Path(ph!1) -> Mphage(p!1).Path(ph!1) + IL1(b)   sL_M
Path(ph!1).Mphage(p!1) -> Mphage(p)                          dP_M
```

The total quantity of endothelial is held constant during a simulation. The receptor sites on endothelial cells bind reversibly to `IL1` and `IL1ra` ligand in the peritoneal or tissue compartment adjacent to the the endothelial layer. Binding follows a bimolecular rate law and each receptor site has identical kinetics. When receptor `IL1` site occupancy is two or greater, endothelial cells enter the *delayed activation* state in a first order rate. (`IL1` does not contribute to activation and thus acts as a competitive inhibitor.) Once in the delayed activation state, the endothelial cell becomes activated at a first order rate. Activated endothelial cells bind irreversibly to neutrophils in the blood compartment according to a bilinear rate law and mediated neutrophil migration into tissue. Activated cells return to the inactive state at a first order rate.

```
# endothelial rules
IL1(b)@P   + Endo(r) <-> IL1(b!1)@P.Endo(r!1)     bLE, uLE
IL1(b)@T   + Endo(r) <-> IL1(b!1)@T.Endo(r!1)     bLE, uLE
IL1ra(b)@P + Endo(r) <-> IL1ra(b!1)@P.Endo(r!1)   bLE, uLE
IL1ra(b)@T + Endo(r) <-> IL1ra(b!1)@T.Endo(r!1)   bLE, uLE
Endo(a~0,r!1,r!2).IL1(b!1).IL1(b!2) \
 -> Endo(a~1,r!1,r!2).IL1(b!1).IL1(b!2)  aE_L  # activation, step 1
Endo(a~1) -> Endo(a~2)  aE                     # activation, step 2
Endo(a~2) -> Endo(a~0)  dE                     # deactivation
Endo(n,a~2) + Neutro(e)@B -> Endo(n!1,a~2).Neutro(e!1)@B  bEN
Endo(n!1)@eP.Neutro(e!1)@B -> Endo(n)@eP + Neutro(e)@P    tN_E
Endo(n!1)@eT.Neutro(e!1)@B -> Endo(n)@eT + Neutro(e)@T    tN_E
```

Neutrophils are produced in the blood compartment at a constant rate. As above, neutrophil bind irreversibly to activated endothelial cells according to a bilinear rate law. Neutrophils bound to endothelial cells migrate into the adjacent compartment under a first order rate law. Since neutrophils only bind to endothelial cells from the blood compartment, neutrophil trafficking is one-way from blood to peritoneum or other tissue. Neutrophils bind to pathogen in the local compartment under a bilinear rate law, and, as above, the pathogen are eliminated in a first order rate after binding. Neutrophils decay in a first order rate that is constant and independent of compartment. Neutrophil that have migrated to the peritoneum or other tissue produce `IL1ra` in the local compartment at a first order rate.

```
# neutrophil rules
Path(ph) + Neutro(p,e) -> Path(ph!1).Neutro(p!1,e)  NP
Path(ph!1).Neutro(p!1) -> Neutro(p)                 dP_N
Neutro@P -> Neutro@P + IL1ra(b)@P  sA_N
Neutro@T -> Neutro@T + IL1ra(b)@T  sA_N
0 -> Neutro(p,e)@B  sN_B
Neutro -> 0         dN DeleteMolecules
```

`IL1` is produced at a first order rate by macrophage that are bound to pathogen. `IL1` "leaks" between peritoneum and blood, and between other tissue and blood in a first order rate proportional to surface area of the relevant endothelial barrier. Forward and reverse rates are constrained such that equal concentrations are achieved in adjacent compartments at equilibrium. In the blood, cytokine is eliminated at a first-order rate to model the elimination of cytokine by the kidney, liver and other mechanisms. As above, cytokine in the peritoneum and other tissue bind reversibly to receptor sites on endothelial cells.

IL1ra is produced at a first order rate by neutrophils in the peritoneum and lung compartment. As with IL1, IL1ra leaks between the peritoneum and blood, and between other tissue and blood in a first order rate proportional to surface area of the relevant endothelial barrier. IL1ra in the peritoneum and other tissue binds irreversibly to receptor sites on endothelial cells, but does not active endothelial cells. IL1ra, like IL1 is eliminated in the blood according to a first order rate law.

```
#IL1, IL21ra rules
IL1(b)@P    <-> IL1(b)@B      tL_PB, tL_BP   # IL1 leak P<->B
IL1(b)@T    <-> IL1(b)@B      tL_TB, tL_BT   # IL1 leak T<->B
IL1ra(b)@P <-> IL1ra(b)@B    tL_PB, tL_BP   # IL1ra leak T<->B
IL1ra(b)@T <-> IL1ra(b)@B    tL_TB, tL_BT   # IL1ra leak T<->B
IL1(b)@B    -> 0  dL                         # IL1 blood elim.
IL1ra(b)@B -> 0  dL                         # IL1ra blood elim.
```

5.3.2 Calibration and simulation

5.3.2.1 CLP protocol Cecal-ligation and puncture (CLP) was simulated by initializing the model system with a non-zero quantity of pathogen in the peritoneum at time zero. The severity of was adjusted by increasing or decreasing the quantity of initial pathogen at time zero. Pathogen is assumed to be contained in the peritoneum during the course of the simulation.

5.3.2.2 HA protocol HA treatment was modeled as a first-order cytokine (IL1 and IL1ra) elimination rate in the blood compartment. Although HA treatment in the laboratory was limited to a 4 hour regimen, simulations were performed under the assumption of continual HA. Given a typical HA flow of 0.8 ml/min [153] and hematocrit of .45, the rate of plasma flow through the device is about 26.4 ml/hr. Assuming 7.8 ml of plasma in a 250g rat [182], the flow rate per unit plasma is 3.38 /hr. The cytokine capture efficiency of the

HA device is known to drop in a time-dependent fashion [167, 168], but 100% capture was chosen as a first approximation suitable for this exploratory work. Since the natural half-life of cytokine in the blood is on the order of 30 minutes [183] (elimination rate 1.4 /hr), the HA device increases the elimination of cytokine by about two to three fold.

5.3.2.3 Calibration Parameters were based on estimates obtained in experimental literature, where available. Remaining parameters were selected from physiologically-reasonable ranges. The complete set of parameter values are listed in the model in Appendix H.1. The model was based on a 250g rat with 74.2 ml of interstitial fluid, including 7.8 ml of blood plasma [182]. Of the non-plasma interstitial fluid, 7.8 ml were assigned to the peritoneal fluid, `P`, and the remaining to other tissue `T`. Concentration units were arbitrary units /ml for cells, and pM for IL-1 and IL-1ra. Based on a pharmacokinetic study of TNF$-\alpha$ in rats that reported half-life in the range of 15-23 minutes [183], baseline elimination rate of IL-1 and IL-1ra were set to 1.4 /hr. IL-1 and IL-1ra dissociation constants were 150 pM [180]. Neutrophil decay rate corresponded to a half-life of 12 hours, since neutrophil life span was reported to be 8 hours in the blood and longer in tissues [184]. Pathogen capture rate per neutrophil was about 12 /hr for the pathogen doses simulated. Upon capturing a pathogen, phagocytosis completes with half-life of 15 minutes. Macrophage/pathogen interaction parameters were set to the same values as neutrophil/pathogen parameters. Pathogen growth rate corresponded to a doubling-time of 4 hours.

5.3.2.4 Simulation The reaction network of the cBNGL model generated in BioNetGen version `2.2.4`. The reaction network was then translated to a system of ODEs via the BioNetGen `writeMexfile()` method and integrated with SUNDIALS CVODE 2.6.0 using the stiff solver option. Subsequent analysis was performed in MATLAB.

5.3.2.5 Model outcomes For a given severity of CLP, P_0, and blood cytokine elimination, dL, a simulation was classified as *non-lethal* if the concentration of pathogen fell below 1 a.u./ml prior by day 7 post-CLP and remained below that threshold through day 10. Otherwise the simulation was classified as *lethal*. For the selected parameter set, an *aseptic lethal* condition, i.e. pathogen falls returns to zero while inflammation remains high, was not observed. Hence there was no need to define an aseptic lethal group.

5.3.3 Model simulation results

Sepsis model #1 was simulated with increasing initial pathogen dose to reproduce the results Alves-Filho et al. reported for lethal and sublethal sepsis. Neutrophil recruitment and cytokine levels versus initial pathogen dose are shown in Figure 5.3. Panel **A** shows maximum neutrophil recruitment to the peritoneum and other tissue, and the minimum level of circulating neutrophils during the course of a 7 day simulated septic event. Peak neutrophil recruitment to the peritoneum occurs at pathogen dose $P_0 \approx 300$ a.u./ml, and then decreases slightly as pathogen increases towards the lethal threshold at $P_0 = 1638$ a.u./ml. Panel **B** shows the maximum cytokine concentration in the peritoneum, blood, and other tissue over the course of the simulation. Cytokine levels increased in all organ compartments with increasing pathogen dose, until cytokine levels plateau at the lethal threshold. Simulated results for both cytokine and neutrophils agree qualitatively with experimental measurements obtained in lethal and sublethal murine CLP models [177]. However, the reduction in peritoneal neutrophils in lethal versus sublethal sepsis was about 4-fold in experiments, but only about 15% in sepsis model 1. This suggests that additional mechanisms, such as defective neutrophil rolling and adhesion [177], may be play an important role in the observed results. Nonetheless, model results indicate that circulating neutrophil depletion is a possible contributor to decreased peritoneal neutrophil recruitment in lethal sepsis.

Next, sepsis model 1 was simulated with increasing blood cytokine elimination rate in order to explore the effect of HA treatment on neutrophil recruitment, cytokines, and survival. Neutrophil recruitment and cytokine levels versus blood cytokine elimination rate are shown in Figure 5.4. All simulations were performed with initial pathogen dose $P_0 = 1638$a.u./ml. Baseline blood cytokine elimination rate was $dL = 1.4/\text{hr}$, corresponding to a natural half-life of 30 minutes in the blood. HA was simulated by increasing the blood cytokine elimination rate. Panel **A** shows maximum neutrophil recruitment to the peritoneum and other tissue, and minimum level of circulating neutrophils, versus the blood cytokine elimination rate as a

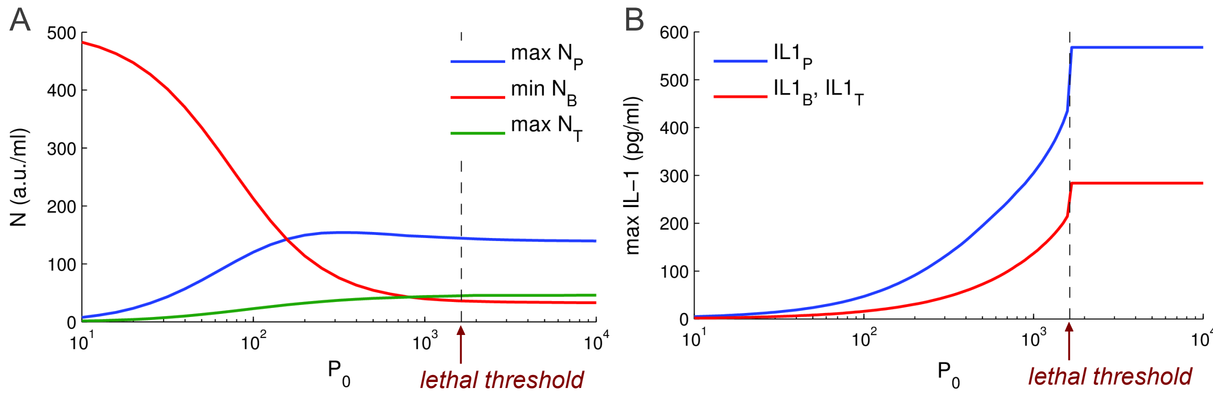


Figure 5.3: Neutrophil recruitment and IL-1 increases in the peritoneum and other tissue, while levels of circulating neutrophils decrease, as initial pathogen dose, P_0, increases. (**A**) Maximum neutrophil recruitment to peritoneum, *max* N_P, maximum neutrophil recruitment to other tissue, max N_T, and minimum circulating neutrophils, min N_B. (**B**) Maximum interleukin-1 in peritoneum, blood, and other tissue. Blood cytokine elimination rate was fixed at $dL = 1.4$ /hr. The lethal threshold (dashed line) is the value of P_0 above which the simulation is lethal.

fraction of baseline dL/dL_0. Neutrophils recruited to the peritoneum increase, while recruitment to other tissues decreases, as the blood cytokine elimination rate increases. Due to a reduction in systemic neutrophil recruitment, the pool of circulating neutrophils maintains a higher level with increasing elimination rate. The increase in neutrophil recruitment in the peritoneum is a consequence of higher levels of circulating neutrophils. Panel **B** shows maximum cytokine levels in the peritoneum, blood, and other tissues. In all cases, maximum cytokine levels decrease with increasing blood cytokine elimination rate, with a large drop in cytokine levels near the lethal threshold. Note that the pathogen dose is lethal when the fractional elimination rate falls below 1.

These results show that an increase in blood cytokine elimination is a possible mechanism for improved survival in septic animals treated with an HA device. Figure 5.5 shows pathogen and peritoneal neutrophil trajectories for various blood cytokine elimination rates with a fixed initial pathogen dose, $P_0 = 1638$ a.u./ml. In panel **A**, the pathogen trajectory tends towards zero for higher elimination rates, but tends to a lethal outcome for lower elimination rates. Panel **B** shows that peritoneal neutrophil recruitment increases with higher elimination rates, demonstrating the mechanism for improved pathogen elimination. Finally, Figure 5.6. shows the minimum lethal pathogen dose as a function of the blood cytokine elimination rate. The lethal pathogen dose is monotonic increasing for the range of elimination rates simulated.

Together, these results demonstrate that the hypothesized HA mechanism is sufficient to explain the improved survival of HA treated animals, increased neutrophil recruitment to the peritoneum, and recruitment to other tissues. The model predicts that this mechanism is predicated on a decrease in the level of circulating neutrophils.

5.4 FROM THE LABORATORY: ORGAN-SPECIFIC DATA

The simulation results presented above, along with other lines of evidence [177, 185], motivated experimental collaborator Zhi-Yong Peng, et al., working in lab of Dr. John Kellum, to measure white blood cell, chemokine, and cytokine levels in the blood, lung and peritoneum

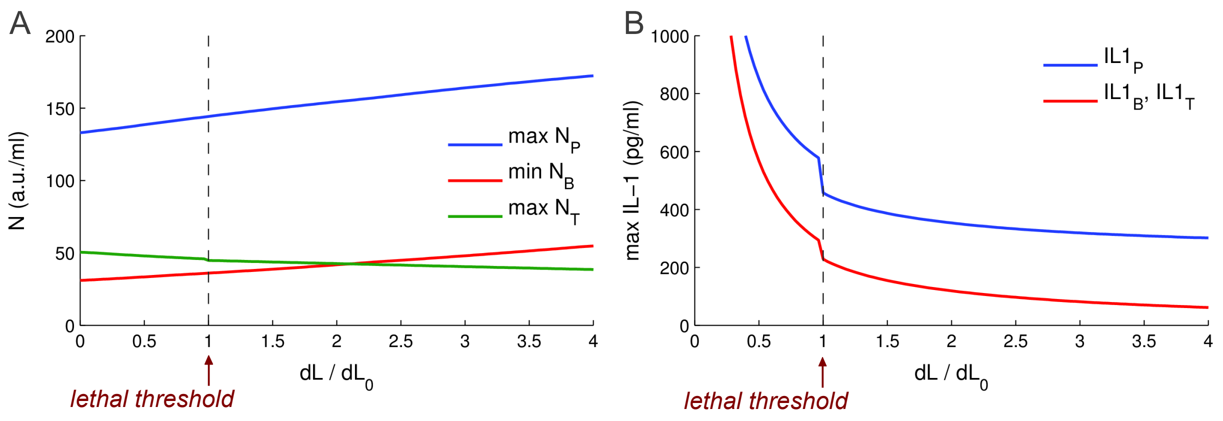


Figure 5.4: **Neutrophils and IL-1 vs. Blood Cytokine Eliminate Rate.** Neutrophil recruitment to the peritoneum and levels of circulating neutrophils increase, while recruitment to other tissues decreases, with an increase in the rate of blood cytokine elimination. Improved recruitment to the lung is a consequence of higher levels of circulating neutrophils, which is in turn a consequence of lower cytokine levels in other tissue. (**A**) Maximum neutrophil recruitment to peritoneum, max N_P, maximum neutrophil recruitment to other tissue, max N_T, and minimum circulating neutrophils, min N_B. (**B**) Maximum interleukin-1 in peritoneum, blood, and other tissue. Blood cytokine elimination rate is expressed as a fraction of the baseline rate, $dL_0 = 1.4$ /hr. All simulations performed with $P_0 = 1638$ a.u./ml. The lethal threshold (dashed line) is the value of dL/dL_0 above which the simulation is lethal.

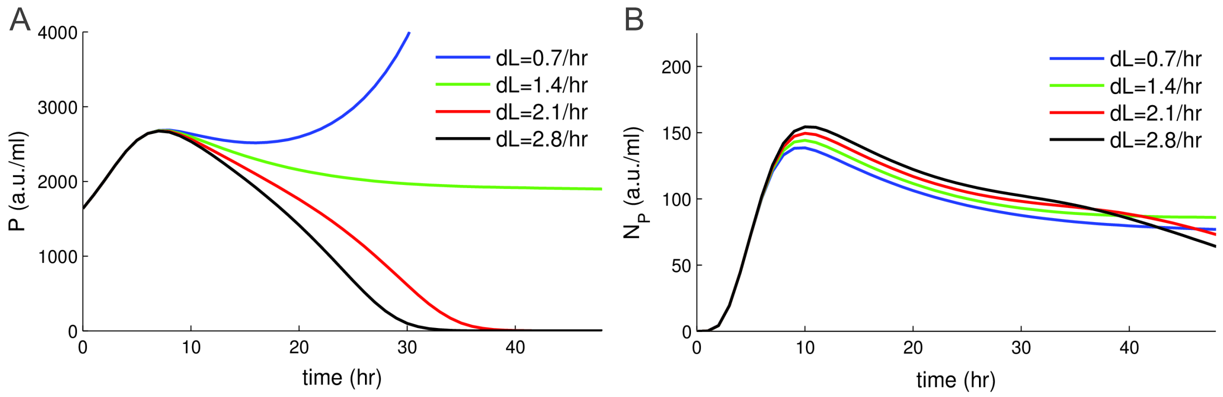


Figure 5.5: Increasing the blood cytokine elimination rate, dL, rescues an otherwise lethal simulation. Improved pathogen elimination is a consequence of improved neutrophil recruitment to the peritoneum with higher dL. (**A**) Pathogen trajectories and (**B**) peritoneal neutrophils for various blood cytokine elimination rates. Blood cytokine elimination rate is expressed as a fraction of the baseline rate, $dL_0 = 1.4$ /hr. All simulations performed with $P_0 = 1638$ a.u./ml.

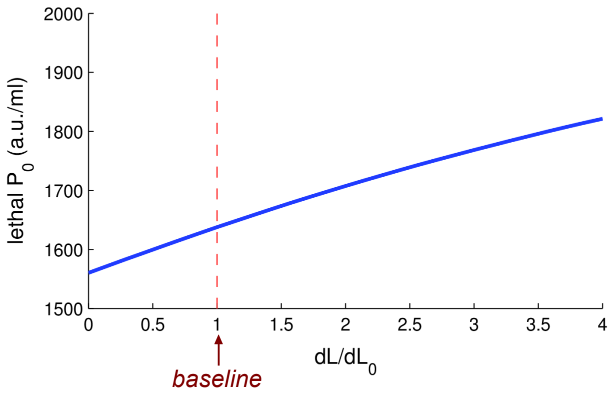


Figure 5.6: The minimum lethal pathogen dose increases with blood cytokine elimination rate, dL.

of rats subjected to CLP and treated with either HA or a sham device. I will briefly describe their methods and results here. Male Sprague-Dawley rats (450-500g) were subjected to a "less-lethal" CLP protocol [153]. At 18 hours post-CLP, animals received either HA or sham treatment for 4 hours. The HA device was a 1 ml cartridge containing CytoSorb polymer beads, with flow rate 0.8-1.0 ml/min. Blood was drawn at 18, 22 and 72 hours after CLP. Animals were sacrificed at 72 hours and fluid was sampled from the lung by bronchoalveolar lavage (BAL) with 15 ml of saline, and peritoneal fluid (PF) by aspiration of the peritoneal cavity with 30 ml of saline. A panel of cytokines and chemokines was measured via multiplex Luminex assays. White blood cells (WBC) were isolated in BAL and PF and the percent polymorphonuclear leukocytes (PMN) and monocytes were assessed by flow cytometry. PMN were also isolated and counted directly in PF and blood at 72 hours. Bacteria culture was obtained from PF and blood samples at 72 hours. Further details of the experiments were reported at Critical Care Congress [186] and a manuscript is in preparation.

All measurements were log transformed prior to computing mean and standard error. Measurements below the detection limit of the assay were replaced with numbers drawn from a uniform distribution over $[\mathrm{LDL}/2, \mathrm{LDL}]$, where LDL is the lower detection limit. Statistical significance was assessed by two-sample Student's t-test with threshold $p = 0.05$.

5.4.1 Experimental results

IL-6, TNF-α, IL-10, and MCP1 were significantly lower in the blood at 72 hours in HA versus sham treated animals (Figure 5.7). Surprisingly, none of the assayed cytokines and chemokines were statistically different at the 22 hour time point that coincides with the end of HA treatment. Cytokines and chemokines tended to be lower in HA at the 22 hour time point, but the difference was not statistically significant. A similar result was obtained in a second study [187]. As expected, there was no statistical difference in cytokines and chemokines at the 18 hour time point coinciding with the start of HA.

IL-6, IL-10, CXCL1 (KC), MCP1, and MIP2 were significantly lower in BAL samples at 72 hours in HA versus sham treated animals (Figure 5.7). Cytokines and chemokines in PF tended to be higher in HA treated animals, but the difference was not significant. These results suggest that inflammation in the lung, but not the peritoneum, is reduced in the HA treated animals.

The ratio of cytokines and chemokines in the PF and BAL to the blood was computed and tested for differences between HA and sham treated animals. The ratio of IL-6, IL-10, CXCL1, and MIP2 in PF to blood was significantly increased at 72 hours in HA treated versus sham animals. The ratio of IL6 and MCP1 in BAL to blood was significantly decreased, while the same ratio of TNF-α was significantly increased. Some evidence suggests the difference in chemokine concentration between tissue and blood is an important factor in recruitment of WBC to inflamed tissue [185].

Since measurements of cytokines and chemokines were very noisy, and since the trends across them were similar, the data was standardized and lumped to determine the overall trend in the data. Each cytokine and chemokine was standardized by its mean and standard deviation across all time points and organs. The standardized data for all cytokines and chemokines was lumped. Mean and standard error of the lumped data was then computed for each time point and tissue. Figure 5.7, lower right panel, shows the lumped cytokine/chemokine data. Lumped cytokine/chemokine was significantly lower in BAL and blood samples at 72 hours in HA versus sham treated animals. No statistical difference was observed at the 22 hour time point, although the mean value was lower in HA treated animals.

PMN counts in BAL were significantly lower in HA versus sham treated animals (Figure 5.8). PMN counts in the blood and PF were higher in HA treated animals, although the difference was not statistically significant. Monocytes counts in PF were significantly higher in HA treated animals, but not statistically different in blood or BAL. These results suggest that WBC recruitment to the peritoneum is preserved or increased somewhat in HA treated animals, while systemic recruitment is reduced. Since circulating WBC are not different between sham and HA treated animals, it appears that depletion of circulating WBC is not an important factor in HA device efficacy.

Bacterial culture in the PF and blood at 72 hours was not significantly different in HA and sham treated animals (Figure 5.8), though average bacterial counts in the PF were higher in sham versus HA. Bacterial counts in the blood were on the order of 10^2-10^3 /ml

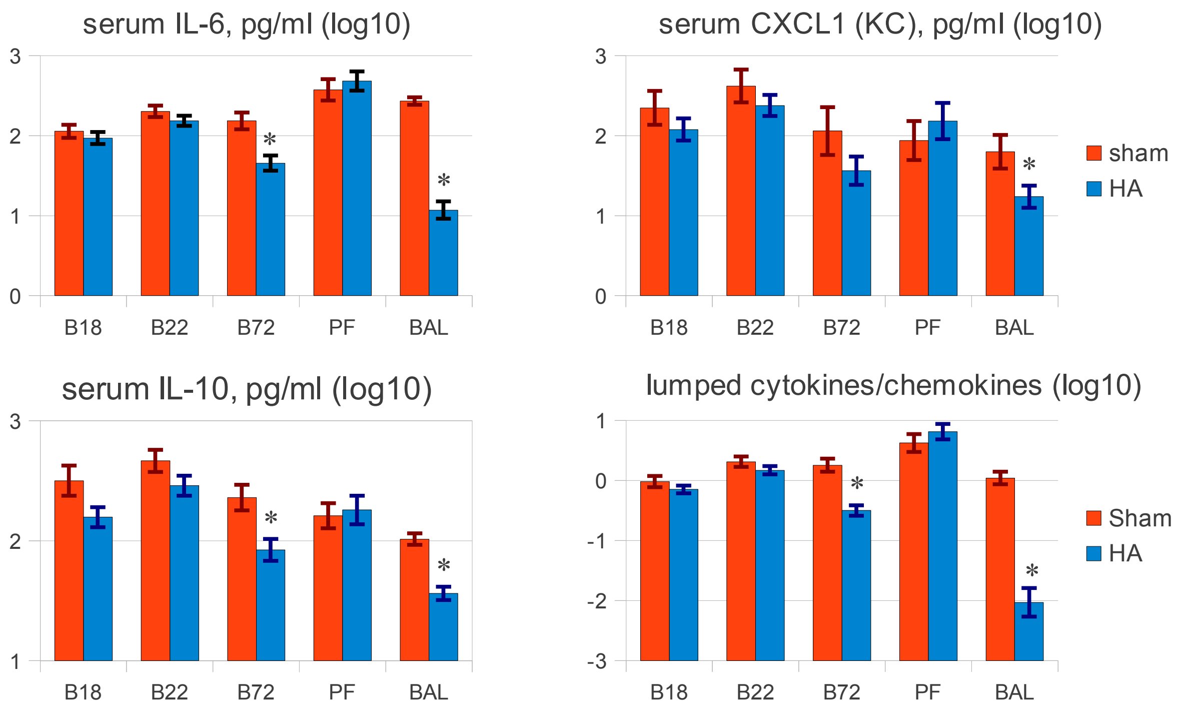


Figure 5.7: Experimental measurements: cytokine and chemokine measurements obtained from a CLP rat model with HA or sham treatment. Blood was sampled at 18, 22 and 72 hours post CLP (B18,B22,B72). Measurements in the lung (BAL) and peritoneal fluid (PF) were obtained at 72 hours post CLP. Data was log transformed prior to analysis. Mean and standard error are shown. Asterisk indicates a significant p-value (0.05) under the Student's t-test. Data for each cytokine and chemokine was standardized and then averaged to show the overall trend (lower right).

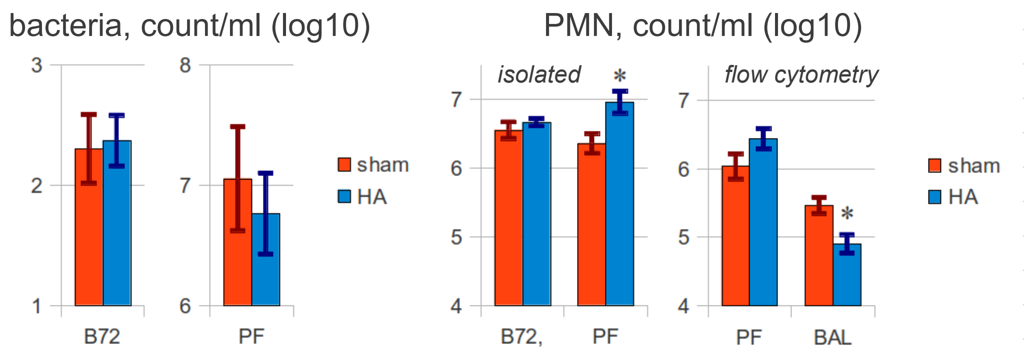


Figure 5.8: Experimental measurements: neutrophil and bacteria cells counts obtained from a CLP rat model with HA or sham treatment. Measurements were obtained 72 hours post CLP in the lung (BAL), peritoneal fluid (PF) and blood (B72). Data was log transformed prior to analysis. Mean and standard error are shown. Asterisk indicates a significant p-value (0.05) under the student's t-test.

compared to 10^6-10^7 /ml in PF.

Put together, the organ-level data shows that HA treatment decreases systemic inflammation as measured by cytokines, chemokines, and neutrophil recruitment. This is accompanied by sustained or increased response at the site of infection, the peritoneum.

5.4.2 Comparison to sepsis model 1

Some of the model 1 predictions were validated by experiment, but others were not supported. The model predicted a decline in cytokines and neutrophils in tissues distant to the site of infection in HA treated animals. This was confirmed by experiment, which found significantly lower neutrophil counts, cytokines levels, and chemokine levels in BAL. The model also predicted an increase in neutrophils and a drop in cytokines in PF for HA versus sham. Experiments failed to find a significant difference in neutrophil counts, cytokines or chemokines in PF, although the average counts were higher in HA. Since the model predicted a small change in PF neutrophils, it is not surprising that the difference was not statistically significant. However, it is difficult to reconcile the model prediction of decreased PF cytokines with experimental measurements that favor an increase. Therefore, I conclude that the model failed to predict the PF cytokine response to HA treatment.

Model analysis suggested that improved pathogen clearance in HA was a consequence of a partial reversal of circulating neutrophil depletion. Experimental data did not support circulating neutrophil depletion or a difference in circulating levels between HA and sham treated animals. Circulating neutrophils and monocytes in the blood were not significantly different between HA and sham. Although baseline neutrophil counts were not measured, levels of circulating neutrophils were in the range reported in healthy animals [182]. Therefore, experiments did not support the proposed mechanism.

Surprisingly, experiments failed to detect a statistically significant difference in cytokines or chemokines at the 22 hour time point. The 22 hour time point coincides with the termination of HA treatment. Since it is known that the HA device captures cytokines and chemokines, it was expected that HA treated animals would have significantly lower levels of cytokines and chemokines immediately following treatment. Since this was not the case, it is unclear how HA treatment leads to reduced systemic inflammation and improved long term survival. It is possible that cytokines and chemokines are moderately lower, but statistical significance was not achieved due to a high signal to noise ratio in experimental assays. Alternatively, an unmeasured inflammatory mediator may be responsible for the efficacy of HA [187]. It is also possible that a mechanism other than capture of soluble signaling molecules is responsible, e. g. the HA device changes the phenotype of WBCs by some unknown mechanism.

In conclusion, sepsis model 1 predicted the response to HA treatment in the lung, but failed to predict other features. Furthermore, a key component of the proposed mechanism—reversal of circulating neutrophil depletion—was not supported by data. Therefore, model 1 must be rejected and modified.

5.5 A CALIBRATED MODEL OF SEPSIS WITH HEMOADSORPTION

The mechanism of sepsis model 1 was based on a hypothesis that depends on depletion of circulating neutrophils in sham treated animals that is partially reversed by HA treatment. Since experiments did not support either the depletion of neutrophils or a significant difference between HA and sham, the model was deemed to be inadequate. Therefore, a new model, based on a new hypothesis, was required.

The foundation of sepsis model 2 is the hypothesis that the ratio of local chemokine levels to systemic levels (e.g. blood) is the key driver of neutrophil recruitment. Under this hypothesis, reducing chemokine levels in the blood via HA treatment increases the ratio of PF chemokines to blood chemokines. Consequently, neutrophil recruitment to the site of infection is improved. The importance of the ratio of local to systemic chemokines was reported by Call, et al. [185], while Dr. John Kellum proposed it as a potential mechanism for the efficacy of the HA device. For brevity, we will refer to this as the *chemokine ratio* hypothesis.

This chemokine ratio hypothesis is promising, on one hand, since it does not rely on depletion of circulating neutrophils. It is less promising, on the other hand, since it depends on the reduction of blood chemokines by the HA device. Experiments did not identify a statistically significant difference in any of the measured blood chemokines at the time point immediately following HA treatment (22 hours). Therefore it is questionable whether the HA device is able to change the chemokine ratio as required. It is possible that the HA device does reduce blood chemokines, but the difference is modest and thus difficult to achieve statistical significance. Alternately, cytokines and chemokine levels might rebound quickly after termination of treatment. If this is the case, then survival in the CLP must be sensitive to a temporary decrease in the chemokine ratio. I hoped to determine if this explanation was feasible by constructing a model that implements the ratio hypothesis.

Sepsis Model 2 was constructed to take advantage of the organ-level data obtained from CLP experiments. The model was constructed with three compartments representing the blood, PF, and lung. The compartments were populated by bacteria, macrophage, neutrophils, and a cytokine/chemokine. Since the data available is sparse (three time points in the blood, one in the lung, one in PF), I attempted to construct the simplest model suitable for testing the ratio hypothesis. As a consequence of this design choice, only a single abstract molecule was chosen to represent the battery cytokines and chemokines measured. This choice was thought to be reasonable since the sparsity of the data suggests that calibrating a more model complex model would inevitably lead to overfitting. Furthermore, the experimental data reveals a similar quantitative trend for the cytokines and chemokines measured (see Figure 5.7, lower right panel). Measured cytokines and chemokines were standardized by mean and standard deviation and then lumped according to time point and compartment to establish the quantitative values for the abstract cytokine/chemokine agent.

5.5.1 Model construction

An overview of Sepsis model 2 is shown in Figure 5.9. The model was constructed with the principles described above. To summarize: (1) neutrophil recruitment is driven by the local to systemic chemokine ratio, (2) circulating neutrophils are not depleted, (3) experimentally measured species are the basis for the model, and (4) the model is kept as simple as possible. The model was specified as a rule-based model in compartmental BioNetGen language [111]. The model consists of 3 compartments and 5 agents, whose dynamics are governed by 16 rules. The corresponding reaction network has 11 species and 20 reactions. The model required 22 parameters, of which 7 are fixed prior to calibration (depending on the model variant). The model is described in detail in the following sections. The complete BNGL model is listed in Appendix H.2.

5.5.1.1 Compartments The model is composed of three compartments: peritoneum, `P`; blood, `B`; and lung, `L`. The compartments are assumed to be well-mixed, as was the case in Model 1. The endothelium is not explicitly represented in this model, instead traffic across the endothelium is represented by phenomenological terms. It is assumed that pathogen are isolated to the peritoneum. The model is based on a 500g rat with 15.6 ml of plasma [182]. The BNGL compartments block is below.

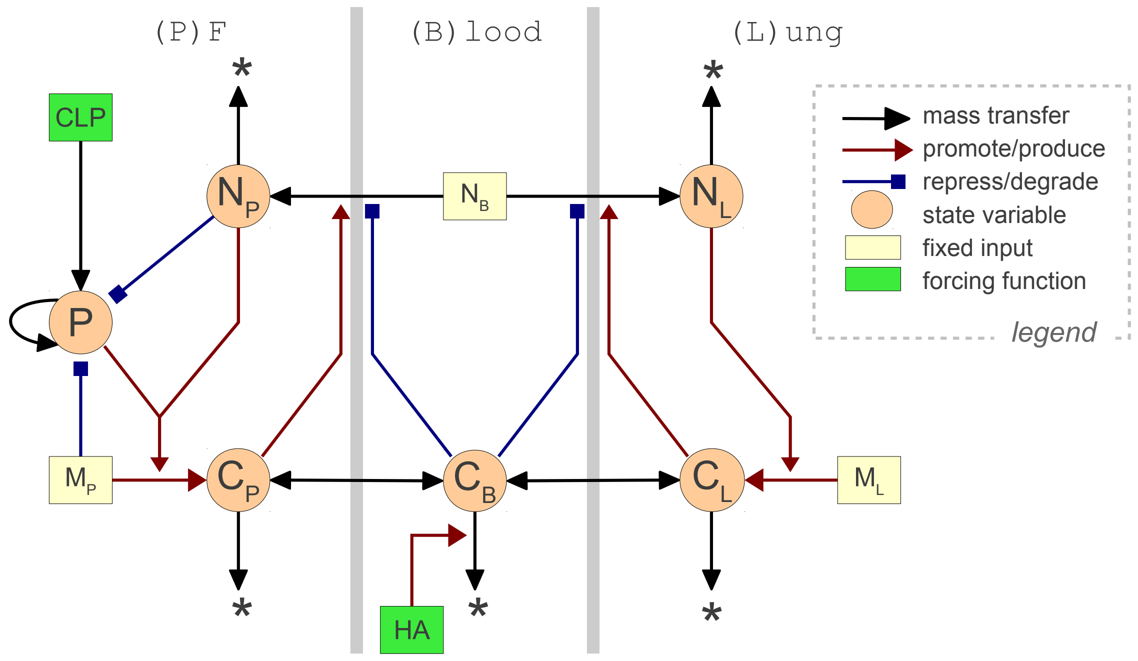


Figure 5.9: **Schematic of sepsis model 2.** The model has three compartments: peritoneal fluid (PF), blood (B), and lung (L). Compartments are populated with pathogen, `Path` (P); macrophage, `Mphage` (M); neutrophils, `Neutr` (N); and a lumped chemokine/cytokine, `Cyto` (C). Circles represent agents, boxes are constant inputs or forcing functions, arrows show mass transfer or regulation.

```
begin compartments
   B  3  vB    # blood
   P  3  vP    # peritoneum
   L  3  vL    # lung
end compartments
```

5.5.1.2 Agents Model compartments are populated by five types of agent: a pathogen source, `CLP`; pathogen, `Path`; macrophage `Mphage`; neutrophils `Neutro`, and the abstract cytokine/chemokine `Cyto`. The `CLP` agent has two states, `s~0` and `s~1`. `CLP` is initialized in state `s~0` (potential), progresses to `s~1` (active pathogen source), and is then eliminated (CLP wound closes). `Path` is initialized at zero and is restricted to compartment `P` during simulations. `Mphage` are present in compartments `P` and `L`, with their populations fixed during simulation. `Neutro` are initially populated in compartment `B`, where the population is held constant, but can migrate into `P` and `L` in response to `Cyto`. The BNGL "molecule types" block, which defines the agents, is listed below:

```
begin molecule types
    CLP(s~0~1)  # pathogen source
    Path()      # pathogen
    Cyto()      # cytokine/chemokine
    Mphage()    # tissue macrophage
    Neutro()    # neutrophil
end molecule types
```

5.5.1.3 Rules `CLP` agents, which are a source of `Path`, are initialized in compartment `P`. The `CLP` agent progresses through 2 stages and is then eliminated from the system. Elimination of `CLP` represents closure or "healing" of the wound. A two-stage process was chosen to achieve a longer time course than the exponential decay of a single-stage process.

`Path` are introduced to the peritoneum by active `CLP`, i.e. state `s~1`, and grow according to a logistic rate law. `Path` are eliminated via phagocytosis by local `Mphage` and recruited `Neutr`, where the rate is linear in `Mphage` and `Neutr` and saturable in `Path`. The model assumes pathogen are contained in compartment `P`, i.e. the infection is local and does not progress to bacteremia. BNGL rules governing `CLP` and `Path` dynamics are listed below. Note that `PathP`, which appears in the logistic growth rate, is a BNGL observable that counts `Path@P`, i.e. pathogen in the peritoneal compartment.

```
# CLP and pathogen rules
CLP(s~0) -> CLP(s~1)         dclp                      # CLP progression
CLP(s~1) -> 0                dclp                      # CLP wound closure
CLP(s~1) -> CLP(s~1) + Path  kpclp                     # pathogen source
Path -> Path + Path          kp*(1-(PathP/vP)/pmax)   # logistic growth
Mphage + Path -> Mphage      pn/(xpn + Path/vP)        # phagocyt. by Mphage
Neutro + Path -> Neutro      kpn/(xpn + Path/vP)       # phagocyt. by Neutro
```

Mphage populations are fixed during the simulation, so there are no rules describing their population dynamics. Mphage produce Cyto according to a second-order Hill function that depends on two terms: local Path and Neutr. Cyto production in response to Neutr is a surrogate for the inflammatory response to tissue damage caused by neutrophil activity, such as the release of radical oxygen species. The rationale for coarse-graining this process was the lack of experimentally measured intermediates.

Cyto "leaks" between the tissue compartments and the blood according to a first order rate law. The forward and reverse leak rates are constrained such that concentrations in neighboring compartments are equalized at equilibrium. Cyto is eliminated in the blood and tissue according to a first order rate law. Separate rate constants are specified for elimination in the blood, where the liver and kidney actively remove cytokine and chemokines, and tissue. Elimination rate in the blood can be enhanced by extracorporeal methods, e.g. HA treatment. The rules governing cytokine dynamics are listed below (PathP, NeutroP, and NeutrL are observables the count Path@P, Neutr@P, and Neutr@L, respectively).

```
# cytokine rules
Mphage@P -> Mphage@P + Cyto@P  f1()                # cytokine prod.
Mphage@L -> Mphage@L + Cyto@L  f2()
Cyto@P <-> Cyto@B              kcBT, kcBT*vP/vB    # cytokine transport
Cyto@L <-> Cyto@B              kcBT, kcBT*vL/vB
Cyto@P -> 0                    dcT                 # cytokine elim.
Cyto@B -> 0                    dcB
Cyto@L -> 0                    dcT

# where:
f1() = kc/mT0*((PathP/vP/xcp)^2 + (NeutroP/vP/xcn)^2) \
            /(1 + (PathP/vP/xcp)^2 + (NeutroP/vP/xcn)^2)
f2() = kc/mT0*(NeutroL/vL)^2/((xcn^2 + NeutroL/vL)^2)
```

Neutr initially populate compartment B, where the population is fixed. Neutr migrate to compartments P and L in response to local Cyto; however, migration is inhibited by the presence of Cyto in compartment B. The rate function for recruitment is linear in tissue Cyto and inhibited in the denominator by Cyto in B. An additional constant term *incB* appears in the denominator to prevent divergence to infinity at CytoB $= 0$. Neutr are eliminated at a first-order rate that is independent of compartment. The BNGL rules for neutrophil dynamics are listed below.

```
# neutrophil dynamics
Neutro@B -> Neutro@P  f3p()   # recruitment to Peritoneum
Neutro@B -> Neutro@L  f4p()   # recruitment to Lung
Neutro -> 0           dn      # neutrophil death

# where:
f3() = (knc*incB/xnc)*(CytoP/vP)/(incB + (CytoB/vB))*vP/(nB0*vB)
f4() = (knc*incB/xnc)*(CytoL/vL)/(incB + (CytoB/vB))*vL/(nB0*vB)
```

An alternative model considered the possibility that the HA device captures blood neutrophils in addition to cytokines and chemokines. *Ex vivo* studies found that HA filters neutrophils from blood [169], though this has not yet been observed *in vivo*. Since blood neutrophil capture would have no effect if the population is fixed, a source of neutrophils in the blood was added to the model and the population was permitted to vary dynamically. The source is regulated by negative feedback which helps to stabilize circulating neutrophil levels during inflammation. The variant of the model with blood neutrophil dynamics is labeled sepsis model $2'$.

```
# blood neutrophil source, sepsis model 2'
0 -> Neutro@B    dn*nB0*(1 + nBfbk*(1 - (NeutroB/vB)/nB0))
```

5.5.2 Calibration and simulation

5.5.2.1 CLP protocol CLP was modeled by an abstract CLP agent placed in the peritoneal compartment of the sepsis model. The CLP agent was initialized at $t = 0$ to 1 a.u./ml (a.u.=*arbitrary unit*) in state s~0. The parameters *dclp* and *kpclp*, the CLP progression and Path source rate constants, were treated as free parameters and calibrated. Since the severity of CLP *in vivo* depends on the number of punctures and the gauge of the needle [188], the initial value of CLP was adjusted to produce more or less severe sepsis *in silico*.

5.5.2.2 HA protocol HA was modeled as a first order elimination of the lumped cytokine/chemokine, Cyto, from the blood compartment. HA was initiated 18 hours post-CLP by increasing the blood `Cyto` elimination rate, dcB, and then terminated after 4 hours of treatment by reseting dcB to baseline. HA supplemented the elimination rate by 1.69/hr, which was computed under the assumption of a 0.8 ml/min flow rate, 0.45 hematocrit, 15.6 ml of plasma, and 100% capture rate.

5.5.2.3 Calibration The model was calibrated to experimental data presented in section 5.4. Data was log-transformed prior to further analysis. Mean and standard error were computed for bacteria and PMN for each time point and tissue. Each cytokine (IL-6, TNF, IL-1β, IL-10) and chemokine (KC, MIP1, and MCP2) was standardized against its mean and standard deviation across all time points and tissues. Standardized cytokines and chemokines were then lumped together at each time point and location prior to calculation of mean and standard error. `Path`, `Neutr`, and `Cyto` were respectively calibrated to bacteria, PMN, and lumped cytokines/chemokines. In total, calibration data consisted of 2 species (`Neutr`,`Cyto`) at 3 time points in `B`, 3 species (`Path`,`Neutr`,`Cyto`) at 1 time point in `PF`, and 2 species (`Neutr`,`Cyto`) at 1 time point in `L`.

Since the data was sparse (18 points) compared to the number of free parameters (15), a Bayesian approach was chosen for calibration. Under the Bayesian approach, parameters are sampled from the posterior distribution of parameters. Model predictions are then computed as averages over the ensemble of model predictions generated from the parameters sets. The key idea is that if many parameter sets explain that data well, then the predictions should be based on the average across all those parameter sets. This approach has been successfully applied to a variety of biological modeling problems [16, 189, 190].

Since model parameters are positive and range over orders of magnitude, all parameters were log-transformed (base 10) prior to calibration. To reduce the likelihood of overfitting, independent Laplace priors were defined for each free parameter. Laplace priors are the Bayesian equivalent of L1-regularization (i.e. LASSO) [104]. Location parameters of the Laplace distributions, μ_p, were based literature ranges, if available, and biological intuition otherwise. A single scale parameter, $b = 0.088$, was selected for all parameters by running trial calibrations with various scales and choosing the maximum scale that did not permit large trajectory oscillations (evidence of overfitting). This corresponds to an L1 penalty of $\lambda = 11.4$. In addition to Laplace prior, some parameters were restricted to biologically feasible ranges. Table 5.1 lists the model parameters, units, and prior distributions.

The log-prior probability for Laplace priors was computed by the formula:

$$\log P\left(\vec{\theta}\right) = \sum_{p=1}^{N_P} \frac{-|\theta_p - \mu_p|}{b}, \tag{5.1}$$

where N_P is the number of parameters.

Sham and HA trajectories were generated for each parameter set $\vec{\theta}$. Simulated trajectories were log transformed prior to comparison with calibration data. Concentrations in compartments `P` and `L` were "diluted" to achieve consistency with experimental methods (PF was aspirated with 30 ml saline, BAL was performed with 15 ml saline). Log-likelihood was computed as summation over time and location of the weighted square difference between model trajectory and calibration data:

$$\log \mathcal{L}(\vec{\theta}|X, \Sigma) = \sum_t \sum_i \frac{-\left(\log y_{t,i}(\vec{\theta}) - x_{t,i}\right)^2}{2{\sigma_{t,i}}^2}, \tag{5.2}$$

where $y_{t,i}(\theta)$ is the model state of output i at time t, and $x_{t,i}$ and $\sigma_{t,i}$ are the mean and standard error of the log experimental data for output i at time t.

The unnormalized log-posterior probability was computed by the sum of the log-prior and log-likelihood for sham and HA treatments:

$$\log P(\vec{\theta}|X, \Sigma) \;\propto\; \log \mathcal{L}(\vec{\theta}|X, \Sigma, \text{sham}) + \log \mathcal{L}(\vec{\theta}|X, \Sigma, \text{HA}) + \log P(\vec{\theta}). \tag{5.3}$$

Table 5.1: Parameter values, ranges, and priors for sepsis model 2.

parameter	units	value/distribution/range	notes
vP	ml	15.6	interstitial volume of `P`
vB	ml	15.6	plasma volume of `B` (500 g rat) [182, 191])
vL	ml	15.6	interstitial volume of `L`
$mT0$	10^3/ml	10^2	init. `Mphage` [192]
$B0$	10^3/ml	3.6×10^3	init. `Neutr@B` [182]
$CLP0$	a.u./ml	1	init. CLP (arbitrary unit)
$dclp$	/hr	log-Laplace$(.12, b)$	`CLP` progression rate
$kpclp$	10^3/hr	log-Laplace$(5 \times 10^3, b)$	`Path` arrival rate due to `CLP`
kp	/hr	log-Laplace$(.12, b)$	`Path` growth rate [193]
$pmax$	10^3/ml	10^6	maximum `Path`
kpn	/hr	log-Laplace$(3, b)$	phagocytosis, max [193]
xpn	10^3/ml	log-Laplace$(10^3, b)$	phagocyt., half-max `Path` [193]
kc	/ml/hr	log-Laplace$(6, b)$	`Cyto` production, max
xcp	10^3/ml	log-Laplace$(10^3, b)$	`Cyto` production, half-max `Path`
xcn	10^3/ml	log-Laplace$(10^3, b)$	`Cyto` production, half-max `Neutro`
dcT	/hr	log-Laplace$(.347, b)$ $.029 < \theta < .116$	`Cyto` elim. in `P`, `L` [194]
dcB	/hr	log-Laplace$(1.109, b)$ $.69 < \theta < 2.77]$	`Cyto` elim. in `B` [183]
$kcBT$	/hr	log-Laplace$(.347, b)$ $.116 < \theta < 1.39$	`Cyto` transport rate [194]
dn	/hr	log-Laplace$(.058, b)$ $.029 < \theta < .116$	`Neutro` death rate [184]
knc	10^3/hr/ml	log-Laplace$(300, b)$	`Neutro` migration, max
xnc	/ml	log-Laplace$(1, b)$	migration, half-max `Cyto`
$incB$	/ml	log-Laplace$(1, b)$	migration, half-max `Cyto@B` inhibition

5.5.2.4 Dynamic behavior constraints In addition to Laplace priors, parameters were constrained to regions where certain dynamical behaviors are satisfied. These behaviors were deemed to be biological necessities, and thus parameter sets that did not produce those behaviors were excluded from the calibration.

The local tissue macrophage should be sufficient to clear a small dose pathogen insult without recruiting neutrophils from the circulation. Mathematically, this behavior is guaranteed by showing that the `Path-Mphage` subsystem is stable at $\texttt{Path} = 0, \texttt{Mphage} = \mathit{mT0}$. Since the population of `Mphage` is fixed, I need only demonstrate that $\texttt{Path} = 0$ is a fixed point and the derivative of `Path` is negative at 0. This is guaranteed if the inequality holds:

$$\mathit{kp} < \frac{\mathit{kpn} \cdot \mathit{mT0}}{\mathit{xpn}},$$

where *kp* is the pathogen growth rate, *kpn* is the maximum rate of pathogen elimination per macrophage, *mT0* is the concentration of tissue macrophage, and *xpn* is the concentration of `Path` for half-maximum elimination rate.

The local macrophage response should also be insufficient to restrain pathogen growth (without aid of neutrophils) for sufficiently large source of of pathogen. If this were not the case, then neutrophil recruitment would be unnecessary. Mathematically, this is guaranteed by showing that there exists a pathogen source rate such that the derivative of `Path` is always increasing if `Neutro` is held at zero. This holds if:

$$\mathit{kpclp} > \mathit{kpn} \cdot \mathit{mT0},$$

where $\overset{\frown}{\text{k}}$pclp is the source of pathogen due to `CLP`. Note that for this calculation I have assumed `CLP` is fixed at 1.

The positive feedback mechanism between `Neutro` and `Cyto` should have two stable fixpoints when `Path` and `Cyto@B` are fixed at zero. The low fixed point corresponds to the resting state of the inflammatory system. The high fixed point represents a self-reinforcing inflammation that can persist in the absence of infection [20, 23, 195]. It is straightforward to show that the `Neutro-Cyto` subsystem always has a stable fixed point at zero. Since the fixed point equation is cubic equation, it either has 1 or 3 real solutions. Existence of 3 solutions is guaranteed if the nullclines have a non-zero crossing, which is guaranteed if:

$$2xcn < \frac{kc}{dcT}\frac{knc}{xnc \cdot dn}$$

Phase-plane analysis shows that if 3 fixpoints exist, then the upper fixed point is stable and the middle fixed point is a saddle. Thus the inequality is sufficient to guarantee bistability.

Since HA improves survival in laboratory experiments, the calibrated model should also "rescue" trajectories on an otherwise lethal course. This property was enforced heuristically since an analytic condition for rescue is unknown. In HA simulations `Path@P`, `Cyto@P` and `Neutr@P` must fall below $\sim$ 20% of peak experimental values by day 7 and $\sim$ 5% by day 10. This qualitative constraint is more questionable than the previous constraints since HA survival in the laboratory was not complete.

5.5.2.5 Sampling the posterior parameter distribution Samples were generated from the posterior parameter distribution using a variant of the Metropolis algorithm [105, 196]. The Metropolis algorithm, which belongs to the class of Markov chain Monte Carlo (MCMC) methods, generates a chain of parameter samples whose distribution converges to the correct posterior distribution [197, 198]. Briefly, at each step of the algorithm a new sample is generated from a *proposal* distribution. The proposal sample is accepted based on the ratio of its posterior probability to that of the previous sample. This ratio can be computed from the unnormalized posterior probability, which is straightforward calculation. If the ratio is greater than 1, the proposal is automatically accepted. Otherwise, the sample is accepted with a probability given by the ratio. If the proposal is rejected, the original sample is retained. The algorithm may be viewed as probabilistic hill-climbing: the chain tends to climb towards regions of higher probability, but there is a chance of moving back down to explore areas of low probability.

The parallel tempering method was used to generating parameter samples [108, 109]. Parallel tempering is a variant of the Metropolis that runs several MCMC chains in parallel with different "temperatures". Temperature is a parameter that determines strictness in accepting less probable samples. In a high-temperature chain, relative probability is less important and the entropy, or “width”, of the parameter distribution dominates. At lower temperatures, the relative probability becomes more important. Periodically the parallel MCMC chains attempt to exchange samples. The exchange probability is based on the product of the difference in log-probability and difference in inverse temperature. Since the different chains are exploring the distribution at different resolutions and exchanging information, the parallel tempering algorithm tends to converge more rapidly to the true distribution.

Parallel tempering was performed with 4 chains and 25 Metropolis steps between attempts at swapping chains. The proposal distribution was Gaussian with diagonal covariance matrix. During a burn-in phase of 125,000 steps, the proposal step size and temperature chains were adjusted periodically to improve step and swap acceptance rates. The target acceptance rate was 24%. After the burn-in phase, step size and temperature were fixed. The auto-adjust procedure tended to produce good acceptance rate (data not shown). Following burn-in, 5 million steps were performed. Samples were thinned to 1 in 25 steps, for a total of 200,000 samples. Parallel tempering with auto-adjust was implemented in MATLAB. Code is available to the public at `http://code.google.com/p/ptempest/`.

5.5.2.6 Simulation The reaction network was generated in BioNetGen version `2.2.4`. The reaction network was subsequently translated to a system of ODEs via the BioNetGen `writeMexfile()` action and integrated with SUNDIALS CVODE 2.6.0 [38] using the stiff solver option. Absolute and relative tolerances for integration were 10^{-6} and 10^{-7}, respectively. Subsequent analysis was performed in MATLAB.

5.5.2.7 Trajectory classification Model trajectories were classified as *septic lethal*, *aseptic lethal*, or *sublethal* based on the concentration of `Path@P`, `Cyto@P`, and `Neutro@P` at day 21. To define the threshold, trajectories were generated from 500 parameter sets drawn from the posterior distribution, with initial `CLP` ranging from .5–1.5 a.u./ml and either sham or HA treatments at 0–1.6 ml/min. Histograms of `Path@P`, `Cyto@P`, and `Neutro@P` at day 21 revealed two prominent modes: high and low `Path@P`. These were labeled *septic lethal* and *non-septic*. The `non-septic` group was composed of a major mode with low `Cyto@P`,`Neutro@P`; and a minor mode with moderate `Cyto@P`,`Neutro@P`. These modes were labeled *non-lethal* and *aseptic lethal*. Thresholds discriminating between the modes were selected by inspection: `Path@P` $> 10^6$ /ml for *septic lethal*; `Neutro@P` $> 3^5$ /ml and `Path@P` $< 10^6$ /ml for *aseptic lethal*; and `Neutro@P` $< 3^5$ /ml and `Path@P` $< 10^6$ /ml for *non-lethal*. For subsequent analysis, *septic lethal* and *aseptic lethal* trajectories were combined into a single *lethal* category.

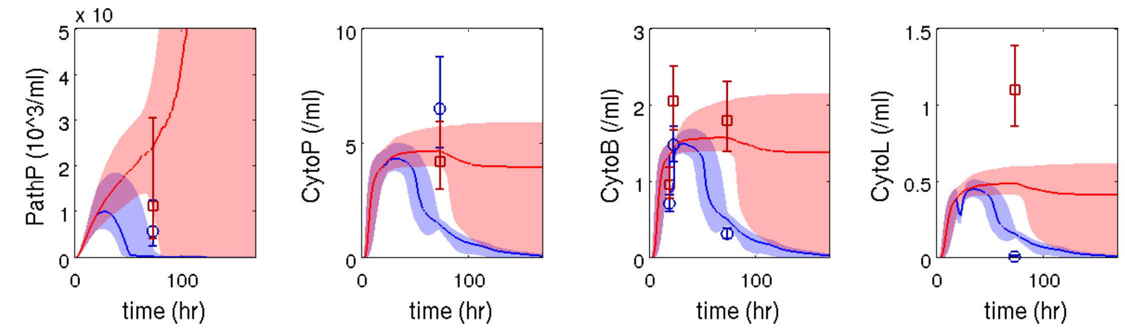
5.5.3 Model simulation results

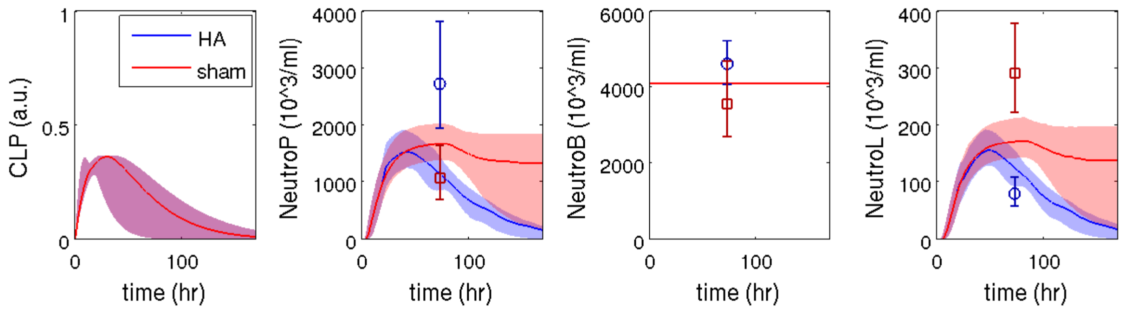
Sepsis model 2 was calibrated to experimental data using Bayesian methods. Distributions of model trajectories were visualized by sampling parameters for the posterior distribution, simulating the trajectories, and then plotting the median trajectory along with the 16th and 84th percentiles. The distributions of sham and HA trajectories are shown in Figure 5.10a. In HA simulations, pathogen trajectories returned to a low level by the end of the simulation, as required during calibration. In contrast, sham distributions were a mixture of pathogen trajectories that either return to a low level or grow out of control. Thus, the postulated HA mechanism is plausible rationale for improved survival in the simulated cohorts.

While HA is able to rescue simulated animals from septic death, the quality of the model fit is poor. `CytoP` and `NeutrP` were elevated in HA versus sham experiments at 72 hours, but in model fits the HA trajectory is lower than sham. `CytoL` and `NeutrL` were elevated in HA versus sham at 72 hours in both model and experiment, however the quantitative fit between model and experiment is poor.

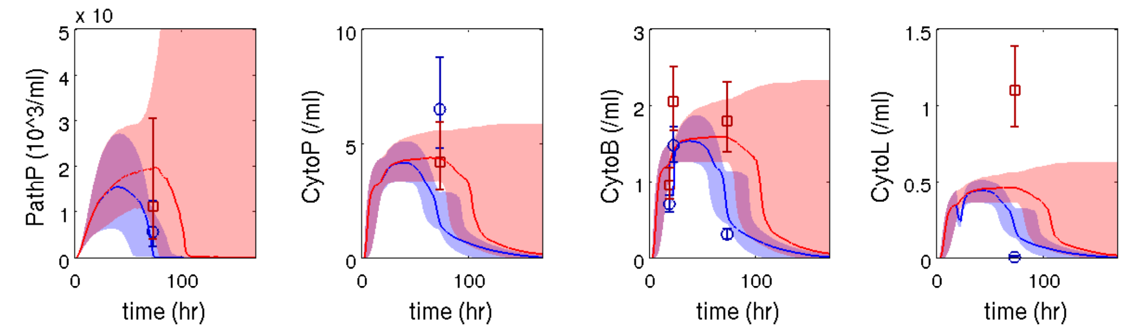
The poor fit was surprising since the number of experimental data points (18) is sparse in comparison to the number of free parameters (15). This suggests that the model structure may be an insufficient representation of the biological process. However, it is important to note that measurements were very noisy, and differences between `CytoP`, `NeutrP` in HA and sham did not reach the level of significance.

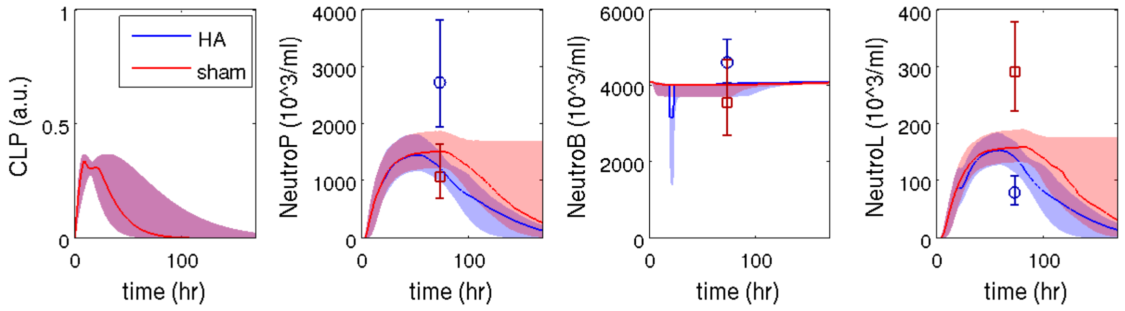
To assess whether poor fits were due to the strength of the Laplace prior, calibrations were performed with prior distributions of varying width. The λ parameter controls the prior strength; a smaller value tends yields better fits but also increases the likelihood of





(a) sepsis model 2: base





(b) sepsis model 2′: blood neutrophil dynamics and capture

Figure 5.10: Trajectory distributions for sepsis model 2. Median trajectories and 16th and 84th percentiles are shown for HA (blue) and sham (red).

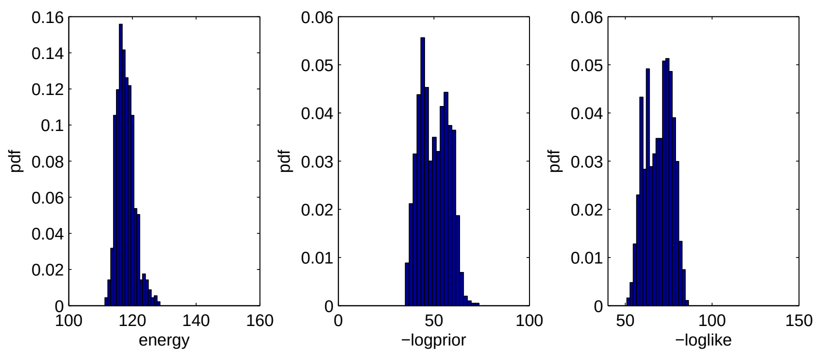
overfitting. Trajectories from calibrations with λ=5.7, 7.1, and 9.5 showed oscillatory behavior, a symptom of overfitting. Thus, I concluded that $\lambda = 11.4$ was necessary to control overfitting.

Distributions of parameter energy (unnormalized log-posterior), log-prior, and log-likelihood are shown in Figure 5.11a. Energy is the negative sum of log-prior and log-likelihood. The most probable log-likelihood was ~ 58, while the most probable log-prior was ~ 52. Although log-likelihood is somewhat smaller than log-prior, the similarity in magnitude shows that the prior plays a large role in the posterior distribution. Since reducing the prior strength leads to oscillation, this suggests that more data points are desirable.

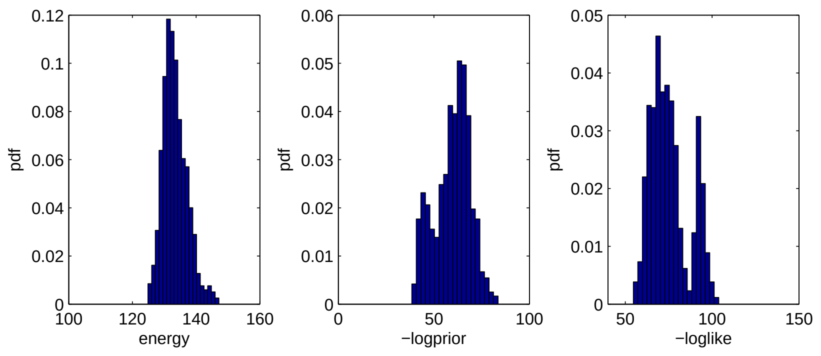
Marginal posterior parameter distributions ($\lambda = 11.4$) are shown in Figure 5.12a. The Laplace prior is plotted along with the posterior histogram to reveal differences. Several posterior distributions drifted away from the prior, demonstrating the importance of these parameters in achieve a good fit. Key parameters include *xcp* and *xcn* (pathogen and neutrophil concentration corresponding to half-maximum cytokine production), *dcB* (blood cytokine elimination rate), *dn* (neutrophil death rate), *kp* (pathogen growth rate), *kpn* (pathogen elimination rate due to neutrophils), *kc* (max cytokine production rate), and *knc* (cytokine concentration corresponding to half-maximum neutrophil recruitment). Parameters that were less important for obtaining a good fit include *kpclp* (pathogen source due to `CLP`), *xpn* (pathogen population corresponding to half-maximum elimination rate), *dcT* (tissue cytokine elimination rate), *kcBT* (cytokine transport rate between blood and tissue), and *incB* (blood cytokine concentration corresponding to half-maximum inhibition of neutrophil recruitment).

The posterior distribution *dclp* is bimodal, revealing two distinct `CLP` progressions (fast and slow) that fit the data with more-or-less equal quality. Figure 5.13 compares the trajectory distributions of the slow and fast `CLP` progressions. The primary difference is observed the `PathP` trajectory. `PathP` peaks at higher levels in fast versus slow `CLP`. A comparison of marginal posterior parameter distributions for fast and slow progression `CLP` is shown in Figure 5.14.

Efficacy of HA treatment was evaluated *in silico* by plotting percent lethality versus CLP severity, CLP_0, for sham treatment and HA treatment with varying degree of intensity. HA

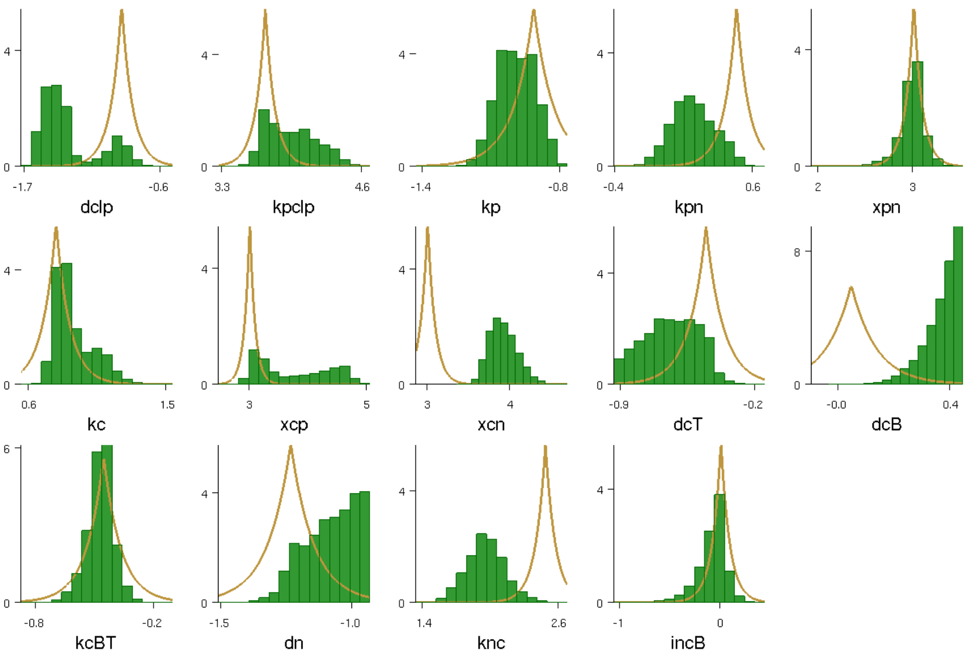


(a) sepsis model 2, original

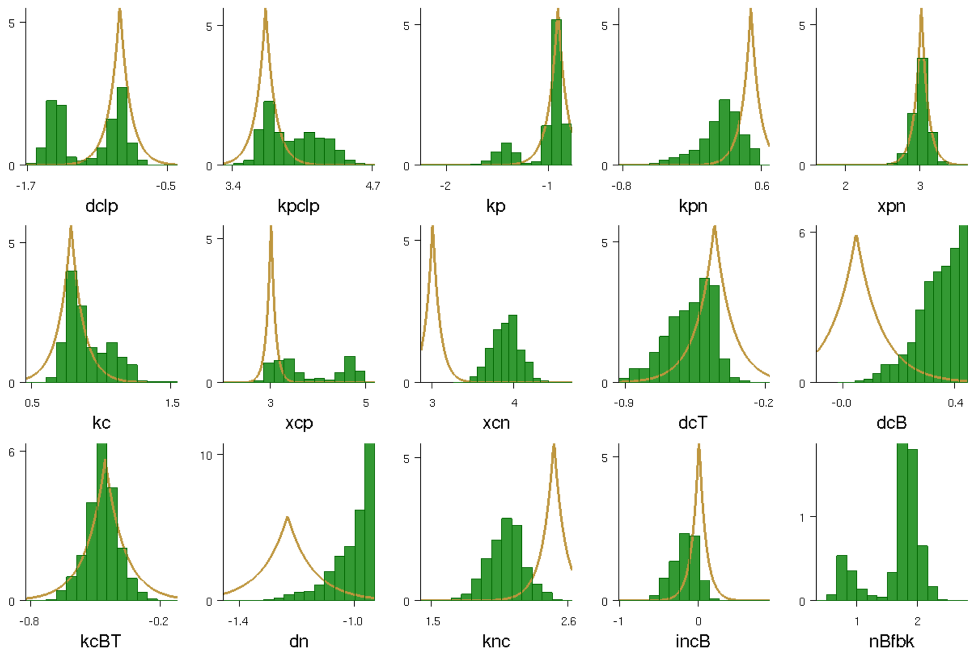


(b) sepsis model 2′, blood neutrophil dynamics and capture

Figure 5.11: Distribution of parameter fitness for sepsis model 2. Total energy (log-posterior) and its log-likelihood and log-prior components are shown.

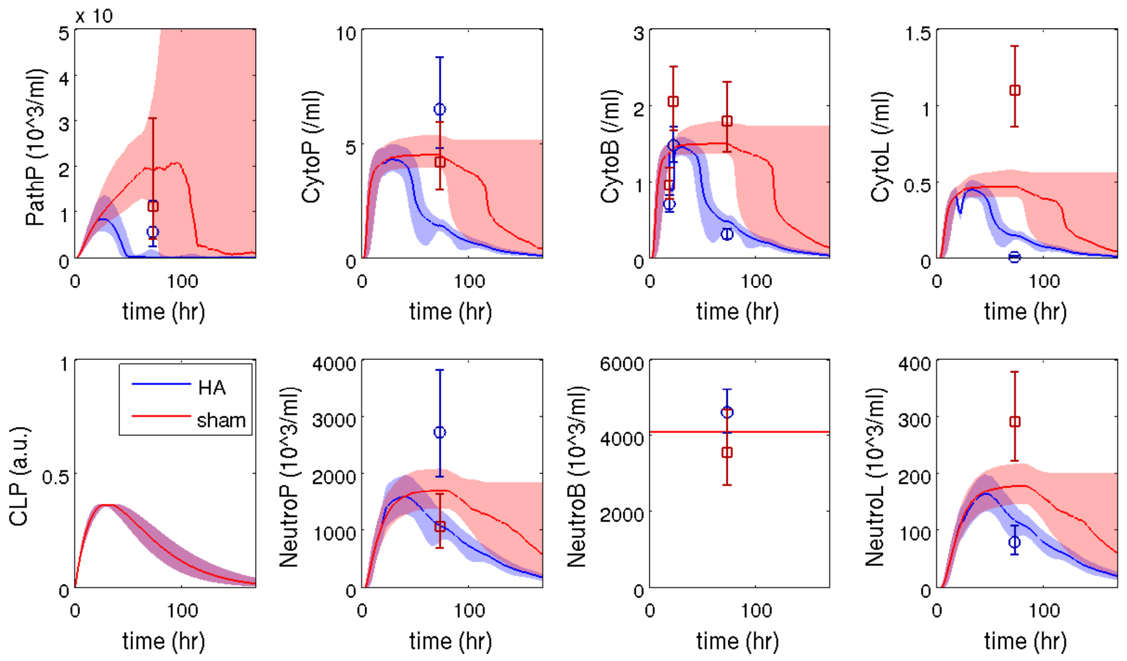


(a) sepsis model 2: base

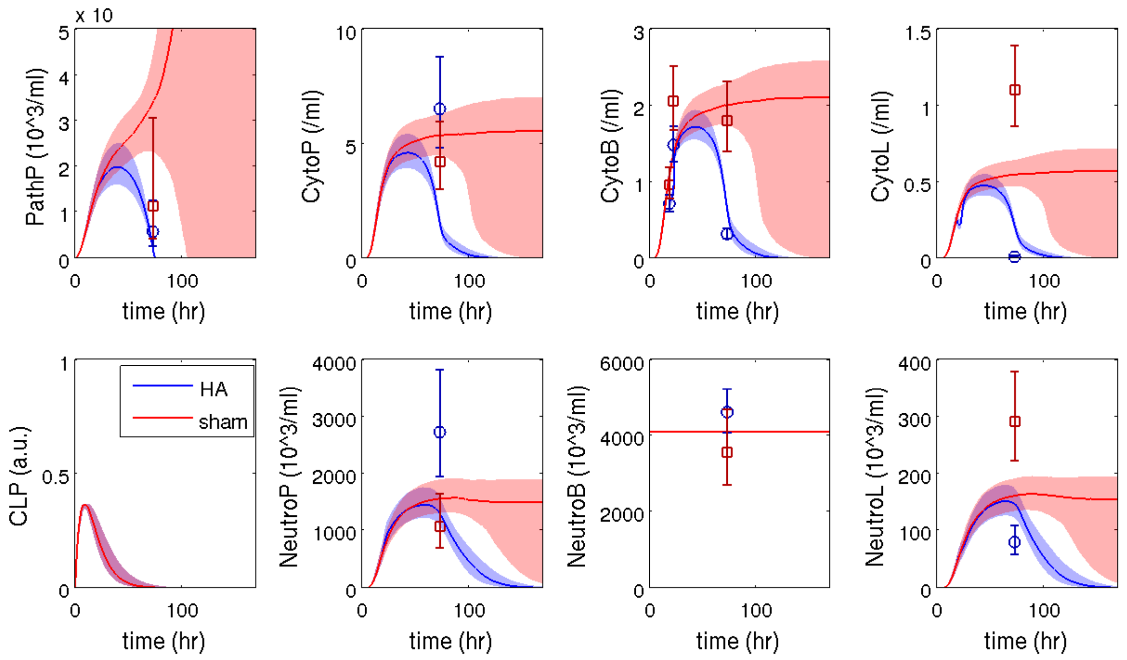


(b) sepsis model 2′: blood neutrophil dynamics and capture

Figure 5.12: Marginal posterior parameter distributions for sepsis model 2 and 2′. Line plots show the Laplace prior. Parameters values are plotted on a log scale (base 10).



(a) slow progression CLP



(b) fast progression CLP

Figure 5.13: Comparison of trajectory distributions for slow and fast CLP progression.

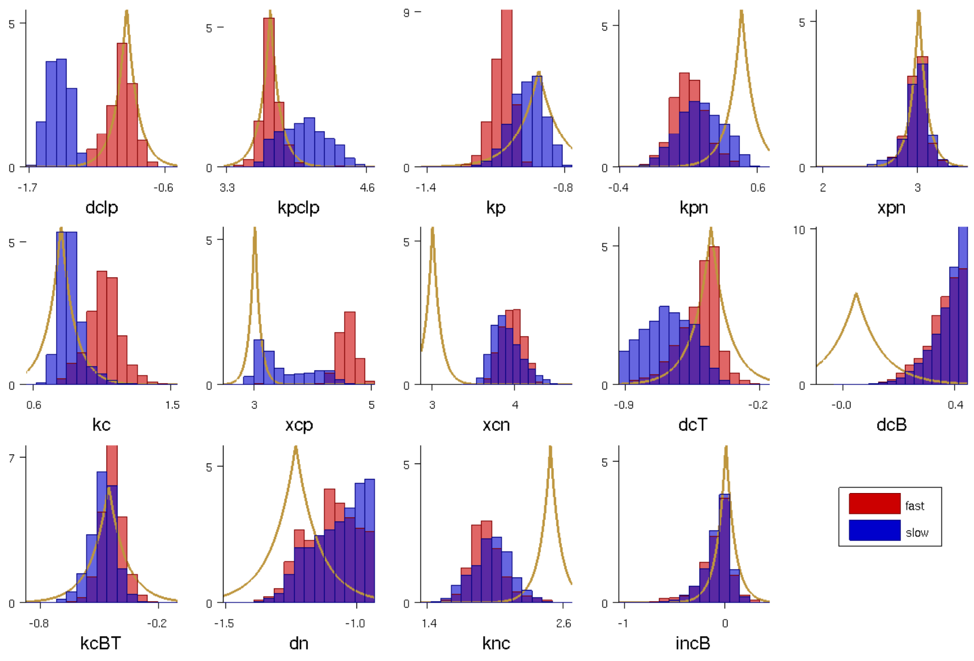


Figure 5.14: Comparison of marginal posterior parameter distributions for slow and fast CLP progression. Line plots show the Laplace prior. Parameters values are plotted on a log scale (base 10).

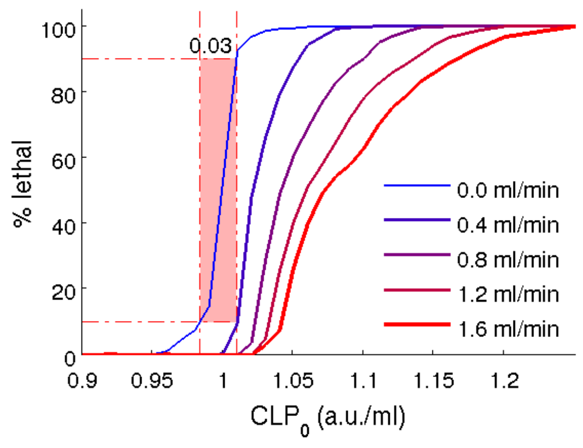
intensity was adjusted by increasing the flow rate through the device (0.4 - 2.0 ml/min). Percent lethality was computed by drawing $N = 500$ parameter sets from the posterior sample distribution and classifying the resultant trajectories as lethal or sublethal. The results are shown in Figure 5.15a.

Percent lethality, as predicted by the model, is very sensitive to CLP severity near $CLP_0 = 1$ a.u/ml and insensitive otherwise. At baseline CLP severity, a 2% change in severity increases lethality from under 10% to over 90%. This is illustrated in Figure 5.15 by a shaded red box. Experiments have demonstrated that CLP lethality is sensitive to variables such as the needle gauge and the number of punctures. However, the experimental sensitivity appears to be much lower than model sensitivity. Ebong et al. [188] found that 18-, 21-, and 25-gauge punctures induce 100%, 50%, and 5% lethality, respectively. A 21-gauge needle has a cross-sectional area 2.5-fold large than 25-gauge needle. Assuming that CLP severity is linearly related to cross-sectional area of the puncture, this suggests that a 2-fold increase in severity increases the lethality by about 50%. Hence, the extreme sensitivity of lethality to to CLP (at baseline $CLP_0 = 1$) is not supported by experiment.

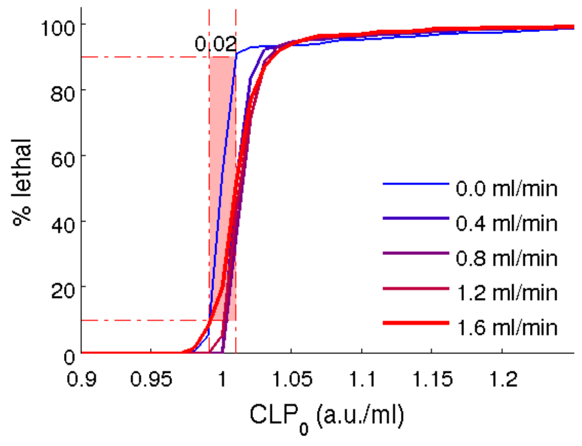
HA treatment (0.8 ml/min) improves the 50% lethal dose (LD50) from 1 a.u./ml to 1.03 a.u./ml, an increase of 3% (Figure 5.15). Doubling the HA flow rate to 1.6 ml/min further increases the LD50 to 1.06 a.u/ml, a 6% increase. Lethality is sensitive to HA only in a narrow range of CLP severity from 0.97 to 1.1. Thus, efficacy of HA is not robust in the model. In contrast, laboratory experiments suggest that improved survival with HA treatment is robust in a rats subjected to CLP [153,186]. Thus, the model failed to replicate the robust response to HA that has been observed in the laboratory.

5.5.3.1 Neutrophil dynamics and capture Since sepsis model 2 fits were not satisfactory, an alternate model was considered. The alternative model, labeled sepsis model $2'$, includes blood neutrophil dynamics. Additionally, the HA device captures circulating neutrophils in addition to cytokine and chemokines. Model $2'$ was calibrated using the same methods described previously.

Trajectory distributions for model $2'$ are shown in Figure 5.10b. Blood neutrophil levels dropped during HA, but promptly rebounded to normal levels. Infection and inflammation



(a) sepsis model 2: base



(b) sepsis model 2′: blood neutrophil dynamics and capture

Figure 5.15: Predicted lethality versus CLP severity with and without HA treatment. Percent lethality is plotted against CLP severity, CLP_0, for sham treatment (0 ml/min) and HA treatment of variable intensity (0.4-2.0 ml/min). Shaded box shows values of CLP_0 corresponding to 10% and 90% lethality with sham treatment.

in the median sham treated animal resolved to a healthy state, in contrast to model 2 where the median sham trajectory led to septic death. Otherwise, trajectory distributions are qualitatively similar to model 2 and have the same shortcomings. Specifically, `CytoP` and `NeutroP` levels are lower in HA versus sham, while experiments display the opposite trend. `CytoL` and `NeutroL` are lower in HA versus sham, as shown in experiments, but do not fit experimental data well.

Efficacy of HA treatment was evaluated, as before, by plotting percent lethality versus CLP severity for sham and a range of HA intensities (0.4-2.0 ml/min). Results are shown in Figure 5.15. Lethality is very sensitive to CLP severity in a narrow range near $CLP_0 = 1$, increasing from 10% to 90% lethality with only a 3% change in severity (shaded red box), and insensitive otherwise. HA treatment at 0.8 ml/min improves survival in a narrow range from $CLP_0 = 0.99$ to 1.05 and is ineffective elsewhere. Thus, HA is not a robust treatment in model $2'$. As discussed above, laboratory experiments indicate that lethality is less sensitive to changes in CLP severity and HA efficacy is more robust than predicted by model.

Distributions of parameter energy, log-prior, and log-likelihood are shown in Figure 5.11b. Average energy and negative log-likelihood for model $2'$ is higher than model 2. Therefore, since model $2'$ also has one more free parameter, model 2 is the preferred model in terms of fitness and number of parameters.

5.6 DISCUSSION

I constructed models of sepsis with hemoadsorption treatment in order to understand the causal mechanism for improved survival observed in experiments. The first model (Section 5.3) implemented a mechanism wherein systemic cytokines activated endothelial cells, leading to widespread neutrophil recruitment, depletion of circulating pools, and inadequate neutrophil recruitment to the site of infection. By removing cytokines from the circulation, HA reduced systemic recruitment, leading to improved circulating neutrophil levels, higher recruitment to site of infection, and clearance of pathogen. In preliminary work, a set of parameters were identified where HA treatment rescued the *in silico* animal from a lethal outcome. The model predicted higher peak neutrophils in the peritoneum, reduced neutrophil and cytokine levels in the lung, and higher circulating neutrophils in HA treated animals.

Experimental collaborators collected organ-specific data from septic rats with HA or sham treatment. As predicted, cytokines and neutrophil counts were significantly lower at 72 hours in the lung of HA versus sham animals. Cytokines in the blood at 72 hours were also significantly lower in HA treated animals. However, there was no statistical difference between circulating neutrophils, peritoneal neutrophils, and peritoneal cytokines in HA versus sham treated animals. Since experiments did not support a key component of the mechanism, i.e. a substantial drop in circulating neutrophils in sham treated animals, the first model was rejected.

A second model (Section 5.5) was constructed with an alternative hypothesis for HA efficacy. In the second model, neutrophil recruitment was dependent on the ratio of local to systemic chemokines. The HA device removes chemokines from the blood, increasing the ratio, and improving neutrophil recruitment to the site of infection. The model was calibrated to experimental data using Bayesian methods. Trajectory distributions were plotted and demonstrated improved survival in HA treated animals. HA treatment led to a reduction in inflammatory mediators in the blood and neutrophils in the lung and blood at 72 hours. Nonetheless, the model fit was poor. Notably, inflammatory mediators and neutrophils in the peritoneum were lower in HA than sham, despite experimental measurements suggesting the opposite trend.

Sensitivity analysis found that the model lethality is very sensitive to CLP at a critical threshold and otherwise insensitive. Similarly, HA treatment was effective over a small range of CLP severity and otherwise ineffective. These results stand in contrast to laboratory studies that show lethality increases gradually with the gauge of the puncture needle [188] and that improved survival is a robust feature of HA.

Since the second model did not adequately reproduce experimental measurements, I analyzed the regulatory structure of the model in search of an explanation. Figure 5.16 shows the regulatory interactions and feedback loops in sepsis model 2. HA treatment has a negative influence on `Cyto@B` (step 1). Since `Cyto@B` negatively influences `Neutro`, the short-term effect of HA on neutrophil recruitment is positive (step 2a). At the same time, reduction of `Cyto@B` reduces the positive influence on `Cyto@P` and `Cyto@L` (step 2b). `Neutro` and `Cyto` participate in a positive feedback loop that acts as bistable switch. Increased `Neutro` pushes the switch towards the elevated inflammation state, while reduced `Cyto` pushes the switch toward rest. In lieu of quantitative analysis, the net effect of `Cyto@B` on the bistable switch is indeterminate (step 3).

Suppose the net effect of reduced `CytoB` is an increase in `Neutr` and `Cyto` levels relative to sham. If the increase is too large, the system becomes trapped in the elevated inflammation state, a lethal outcome. Since HA improves survival, this cannot be the case. So the system must stay in the basin that returns to rest in the absence of pathogen. An increase in `Neutro` exerts a negative influence on `Path` (step 4), and `Path` levels decline relative to sham. Reduced pathogen lowers the positive regulation of `Cyto` (step 5) and, assuming the boost in `Neutro` was sufficient to put `Path` on downward trend, the `Cyto-Neutr` switch begins a return to rest (step 6). This sequence of events leads to improved survival, as desired. `Cyto` and `Neutr` are temporarily elevated during step 3, but improved pathogen clearance accelerates the return to rest at steps 4-6. Therefore, the regulatory structure of the model does not support a long term increase in peritoneal inflammation, unless the elevated inflammation state is reached.

Alternatively, assume the net effect of reduced `CytoB` is a short-term decrease in `Neutr` and `Cyto` levels relative to sham. If the system is approaching the elevated inflammation state, this effect may push the system back into the rest basin. This would lead to improved survival, but also to reduced `Neutr` and `Cyto` in PF. Again, the model does not permit a long term increase in `Neutr` and `Cyto` due to HA treatment.

There is only one scenario where the model supports a long term increase of `Neutr` and `Cyto` in HA relative to sham. This is case where the sham trajectory is tending towards healthy resolution, but a short term increase in `Neutr` and `Cyto` due to HA pushes the system into the elevated inflammation state. Since HA increases lethality in this case, it is not compatible with reports of improved survival with HA.

In experiments, `NeutrP` and `CytoP` were elevated in HA compared to sham at 72 hours. This is more than 2 days after HA treatment ends. Since the model regulatory structure

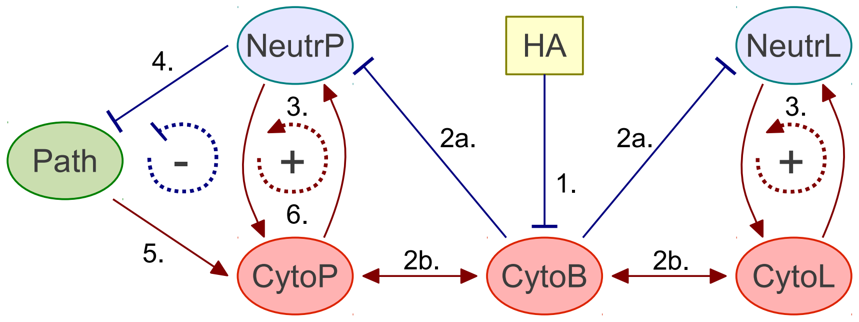


Figure 5.16: **Regulatory influences in sepsis model 2 and the mechanism of HA.** Positive influences are indicated by red arrows, negative influences by blue bars. Feedback loops are shown with dotted arrows and bars. Numbered influences show the influence of HA flowing through the system (see main text). HA negatively influences `CytoB` (1). Since `CytoB` has a positive influence on `Cyto`, but a negative influence on `Neutr`, the net effect on the `Cyto-Neutr` positive feedback loop is uncertain (3). If the net effect is positive, it cannot be so strong as to push the system to the elevated inflammation state (since HA improves survival). The resultant increase in `Neutr` negatively influences `Path` (4) and, in turn, the positive regulation of `Cyto` by `Path` is reduced (5) and the system tends towards rest (6). Alternatively, if the net effect of HA on `Cyto-Neutr` is negative, the system can be rescued from an elevated inflammatory state and pushed into the basin of rest. In either case, the `Cyto` and `Neutr` will be elevated for only a short period of time.

does not support a long-term increase in `Neutr` and `Cyto` in HA, model 2 is an improbable explanation for the data (improbable in the sense that the range of parameters where the short-term effects last for 2+ days is likely to be be vanishingly small).

It is worth noting that experimental measurements differences between HA and sham in PF did not reach the level of significance. So it is conceivable, but unlikely, that `NeutrP` and `CytoP` levels are indeed somewhat lower in HA at the 72 hour time point. Confirmatory experiments would determine if the first experimental results are reliable.

Nonetheless, discrepancies between experimental data and models suggest that further work is needed to understand sepsis and HA. Alternative mechanisms should be considered. For example, Alves-Filho, et al. found evidence that chemokine receptors are downregulated on neutrophils in severe sepsis via a TLR2 dependent mechanism [199]. However, it is unclear why HA treatment would impact TLR2 signaling. It may also prove useful to include macrophage dynamics or independent pro- and anti-inflammatory cytokines, but calibrating such a complex model to sparse and noisy data is questionable. Development of less-invasive assays (permitting higher time resolution sampling) with lower noise should be a priority for both modelers and experimentalists.

APPENDIX A

BNGL SYNTAX

This appendix presents BNGL syntax, including comparments and energy patterns, in Extended Backus-Naur Form (EBNF). EBNF symbols are summarized in Table A1. The BNGL specification listed here is a guideline, and not necessarily authoritative.

Comments are initiated by the `#` character and continue to the end of the line. Line continuation is signified by \, which must be the last non-whitespace character on the line. The following specification assumes that comments and line continuations have been handled in preprocessing. Optional white space is also omitted from the syntax to improve clarity.

The following non-terminals are not defined below, but the standard definitions may be assumed: `Letter`, `Digit`, `Real`, `PositiveInteger`, `NaturalNumber` $(0, 1, \ldots)$, and `String`.

Table A1: Summary of symbols in Extended Backus-Naur Form

description	*symbol*
definition	`=`
concatenation	`,`
termination	`;`
alternation (or)	`|`
optional	`[ ... ]`
one-or-more	`{ ... }`
terminal string	`" ... "`
comment	`(* ... *)`

```
# BNGL syntax with compartments and energy patterns
# (Extended Backus-Naur Form)

WS = {" "|"\t"};
NewLine = {"\n"};

Name = Letter, [{Letter|Digit|"_"}];
State = "∼",(Letter|Digit),[{Letter|Digit|"_"}] | "∼?";
Bond = "!",{Digit} | "!?"  | "!+";
Tag = "%",(Letter|Digit),[{Letter|Digit|"_"}];
Compartment = "@",Name;
LineLabel = {Digit},WS | Name,":",[WS];

ComponentType = Name, [{"∼",State}];

Component = Name, [{"∼",State | "!",Bond | "%",Tag}];

MoleculeType = Name, ["(", [ComponentType, [{",",ComponentType}]], ")"];

Molecule =
    Name, [{Tag|Compartment}], ["(",[Component,[{",",Component}]],")"]
  | Name, ["(",[Component,[{",",Component}]],")"], [{Tag|Compartment}];

SimpleMolecule = Name, [{Tag|Compartment}], ["()"]
               | Name, ["()"], [{Tag|Compartment}];

PatternMods = {"$" | "{matchOnce}"};

PatternQuantifier = ("<"|"<="|"=="|">="|">"),NaturalNumber;

Pattern = "0"
        | [{Tag|Compartment},(":"|"::")], [PatternMods],
          Molecule, [{".",Molecule}], [PatternQuantifier];

Species = "0"
        | [{Tag|Compartment},(":"|"::")], [$], Molecule, [{".",Molecule}];

SimpleSpecies = [{Tag|Compartment},(":"|"::")], SimpleMolecule;

RuleModifier =
    "DeleteMolecules" | "MoveConnected" | "TotalRate"
  | "exclude_reactants","(",PositiveInteger,Pattern,[{",",Pattern}],")"
  | "include_reactants","(",PositiveInteger,Pattern,[{",",Pattern}],")"
  | "exclude_products","(",PositiveInteger,Pattern,[{",",Pattern}],")"
  | "include_products","(",PositiveInteger,Pattern,[{",",Pattern}],")";

RateLaw =
    MathExpression
  | "Sat","(",MathExpression,",",MathExpression,")"
  | "MM","(",MathExpression,",",MathExpression,")"
  | "Hill","(",MathExpression,",",MathExpression,",",MathExpression,")"
  | "Arrhenius","(",MathExpression,",",MathExpression,")";
```

```
(*optional whitespace is permitted in definitions below this point*)

UniRule = Pattern,[{"+",Pattern}], "->",Pattern,[{"+",Pattern}],
          WS,RateLaw,[{WS,RuleModifier}];

RevRule = Pattern,[{"+",Pattern}],"<->",Pattern,[{"+",Pattern}],
          WS,RateLaw,",",RateLaw,[{WS,RuleModifier}];

Rule = UniRule | RevRule;

PopulationMap =
    Species, "->", SimpleSpecies, WS, RateLaw,, [{WS,RuleModifier}];

Reaction = NaturalNumber,[{",",NaturalNumber}],
           WS, NaturalNumber,[{",",NaturalNumber}],
           (Real|Real,"*",Name);

Observable =
    ("Molecules"|"Species"), WS, Name, WS, Pattern, [{",",Pattern}];

Group = Name, WS, [NaturalNumber,"*"], NaturalNumber,
        [{",", [NaturalNumber,"*"], NaturalNumber}];

MathExpression = Real | Id | "(",MathExpression,")"
               | UnaryOperator,MathExpression
               | MathExpression,BinaryOperator,MathExpression
               | Id,"(",[MathExpression, [{",",MathExpression}]],")";

ParameterDefn = Id, (WS|"="), MathExpression;

FunctionDefn =
    Id, ["(", [Name,[{",",Name}]], ")"], (WS|"="), MathExpression;

Args = Real | String | Args,",",Args;
     | "{", Name,"=>",Args, [{",", Name,"=>",Args}], "}"
     | "[", (Real|String), [{",", (Real|String)}], "]";

Option = Name, "(",[Args],")",[";"];
Action = Name, "(",[Args],")",[";"];

(* block defintions *)
ParameterBlock = "begin parameters", NewLine,
                 {[LineLabel], ParameterDefn, NewLine},
                 "end molecule types", NewLine;

MoleculeTypeBlock = "begin molecule types", NewLine,
                    {[LineLabel], MoleculeType, NewLine},
                    "end molecule types", NewLine;

CompartmentBlock = "begin compartments", NewLine,
                   {   [LineLabel], Compartment, WS, ("2"|"3"),
                       WS, MathExpression, [WS, Name], NewLine    },
                   "end comparments types", NewLine;
```

```
SpeciesBlock = "begin seed species", NewLine,
               {[LineLabel], Species, WS, MathExpression, NewLine},
               "end seed species", NewLine
             | "begin species", NewLine,
               {[LineLabel], Species, WS, MathExpression, NewLine},
               "end seed species", NewLine;

ObservableBlock = "begin observables", {NewLine},
                  {[LineLabel], Observable, NewLine},
                  "end observables", NewLine;

GroupBlock = "begin groups", NewLine,
             {[LineLabel], Group, NewLine},
             "end groups", NewLine;

EnergyPatternBlock =
    "begin energy patterns", NewLine,
    {[LineLabel], Pattern, [WS], MathExpression, NewLine},
    "end energy patterns", NewLine;

FunctionBlock = "begin functions", NewLine,
                {[LineLabel], FunctionDefn, NewLine},
                "end functions", NewLine;

ReactionRuleBlock = "begin reaction rules", NewLine,
                    {[LineLabel], Rule, NewLine},
                    "end reaction rules", NewLine;

ReactionBlock = "begin reactions", NewLine,
                {[LineLabel], Reaction, NewLine},
                "end reactions", NewLine;

PopulationMapBlock = "begin population maps", NewLine,
                     {[LineLabel], PopulationMap, NewLine},
                     "end parameters", NewLine;

ActionBlock = "begin actions", NewLine,
              {[LineLabel], Action, NewLine},
              "end actions", NewLine;

Block = ParameterBlock     | MoleculesTypeBlock | CompartmentBlock
      | SpeciesBlock       | ObservableBlock    | GroupBlock
      | EnergyPatternBlock | FunctionBlock      | ReactionRuleBlock
      | ReactionBlock      | PopulationMapBlock | ActionBlock;

Model = [{Option}],
        (   {Block}
          | "begin model", NewLine,
            {Block}
            "end model", NewLine
        ),
        [{Action}];
```

APPENDIX B

GRAPH-BASED FORMALISM FOR BIONETGEN

Here, I present the graph-theoretic formalism of rule-based modeling in BNGL. This material is adapted from a previous graph-based description of BNGL by Blinov et al. [66] and also draws inspiration from the κ-calculus graph formalism of Winskel and co-workers [200]. The formalism of Blinov et al. is oriented towards a population-based approach. The terminology of the present work diverges to some degree in order to support a particle-based perspective.

Although similar to κ, BNGL requires a distinct formalism since BNGL: (i) permits multiple identical components in a molecule, and (ii) makes a formal distinction between inter- and intra-molecular reactions via the `+` and `.` symbols.

B.1 PATTERN GRAPHS

Pattern graphs are objects that describe substructures, or "motifs," within molecular complexes. Similar to the concept of a functional group in organic chemistry, a pattern graph specifies a molecular motif, or *moiety*, that participates in a characteristic class of reactions. Any instance of a pattern graph within a complex will participate in its characteristic reaction without regard to the composition and structure of the surrounding complex.

Definition 4 (Pattern Graph)**.** *Given a countable index set* $\mathbb{I}$*, a set of type labels* Ω_T*, and* *a set of state labels Ω_S, a pattern graph is defined by the tuple:*

$$P \models (M \subset \mathbb{I},\ C \subset M \times \mathbb{I},\ B \subset C \times C_{\pm}, \texttt{type} : M \cup C \to \Omega_T,\ \texttt{state} : C \to \Omega_S \cup \{\emptyset, ?\})$$

where M is the set of **molecules***, C is the set of* **components***, B is the set of* **component bonds** *(edges), $C_{\pm} = C \cup \{+, -\}$ is the set of components augmented with the* **non-specific bond** *wildcard* '+' *and the* **null bond** '−'*,* `type` *is a function that assigns a type label to each molecule and component, and* `state` *is a function that assigns a state label to each component* ('∅' *is the* **null state** *and* '?' *is the* **unspecified state** *wildcard*)*. Furthermore, P satisfies the following: (i) at most one bond per component; (ii) bonds are undirected; (iii) no self bonds. The set of all pattern graphs is denoted by $\mathbb{P}$.*

Each component in C is contained by a unique molecule in M. If $c = (m, n) \in C$, then component c is *contained* by molecule m. Components can be thought of as the nodes of the pattern graph, bonds as edges between nodes, and molecules as groups of components.

Definition 5 (Connected Pattern Graph)**.** *A pattern graph P is* **connected** *if every pair of molecules in the pattern graph is connected by a sequence of component bonds.*

B.1.1 Example: representation of a BNGL pattern as a pattern graph

Consider the BNGL pattern

`L(r!1).R(l!1,a~P).`

Let us represent this pattern formally as a pattern graph. We begin by assigning a unique index to each molecule (from left to right) and a unique ordered pair of indices to each component (also from left to right), where the first element of the pair is the index of the molecule containing the component:

$$\texttt{L}_1\texttt{(r}_{(1,1)}\texttt{!1).R}_2\texttt{(l}_{(2,1)}\texttt{!1,a}_{(2,2)}\texttt{\sim P).}$$

Now we can write the corresponding pattern graph tuple $P = (M, C, B, \texttt{type}, \texttt{state})$, where

$$
\begin{aligned}
M &= \{\ 1,\ 2\ \} \\
C &= \{\ (1,1),\ (2,1),\ (2,2)\ \} \\
B &= \{\ \big((1,1),(2,1)\big),\big((2,1),(1,1)\big),\big((2,2),-\big)\ \} \\
\texttt{type} &= \{\ 1 \mapsto \text{“}\texttt{L}\text{”},\ 2 \mapsto \text{“}\texttt{R}\text{”},\ (1,1) \mapsto \text{“}\texttt{r}\text{”},\ (2,1) \mapsto \text{“}\texttt{l}\text{”},\ (2,2) \mapsto \text{“}\texttt{a}\text{”}\ \} \\
\texttt{state} &= \{\ (1,1) \mapsto \emptyset,\ (2,1) \mapsto \emptyset,\ (2,2) \mapsto \text{“}\texttt{P}\text{”}\ \}
\end{aligned}
$$

Note that the bond from component `r` to component `l` appears in B twice, once for each ordering of the components. This is simply a convention for indicating that the bond is undirected. It is also clear that P is a connected pattern graph because molecules `L` and `R` are connected through this bond.

B.2 COMPLEX GRAPHS AND ENSEMBLES OF COMPLEXES

A complex graph, or simply a *complex*, is a pattern graph that defines an assembly of connected molecules with a precise configuration of bonds and states. While a general pattern graph describes parts of a molecular complex, a complex graph describes a complete instance of a molecular complex. In other words, there is no ambiguity, such as unspecified state (‘`?`’) or non-specific bond (‘`+`’) wildcards.

Definition 6 (Complex Graph)**.** *A complex graph is a connected pattern graph where: (i) every component state is defined (no state wildcards); (ii) every component has a null bond or a proper bond (no bond wildcards).*

The state of a model system is described by an ensemble of complexes.

Definition 7 (Ensemble of Complexes)**.** *An ensemble of complexes is a finite set of disjoint complexes.*

Intuitively, an ensemble of complexes is a “particle-based” representation of a system, where each particle corresponds to a distinct molecule in one of the complexes.

B.3 PATTERN GRAPH ISOMORPHISM AND SPECIES

It is useful to define the notion of equivalence of pattern graphs. We will say that two pattern graphs are equivalent, or *isomorphic*, if they have the same composition and structure.

Definition 8 (Pattern Graph Isomorphism)**.** *Pattern graphs x and y are* **isomorphic***, written $x \cong y$, if there exists a* **one-to-one and onto** *map ϕ of molecules and components in x to those in y that preserves the following properties: (i) graph structure; (ii) component containership; (iii) type; (iv) states. A mapping with these properties is called an* **isomorphism***.*

Since isomorphism is an equivalence relation on the set of complexes, we can partition any ensemble into classes of isomorphic complexes, which we call *species*. This partition is useful because all complexes within a class have the same composition and structure and, therefore, each member has the same reaction motifs and propensities to participate in reaction rules.

Definition 9 (Species)**.** *Given a complex graph y, the species $[y]$ is the set of all complexes that are isomorphic to y. The set of all species is denoted by $\mathbb{S}$.*

Though $[y]$ is properly a set, it can be represented by any complex $y' \in [y]$.

If X is an ensemble of complexes, then a similar but distinct concept is that of the species $[y]$ *with respect to* X, written $[y|X]$. This is the subset of complexes in X that are isomorphic to y, i.e., $\{x \in X \mid x \cong y\}$. The *population* of a species with respect to an ensemble X is the number of complexes in $[y|X]$, i.e., $\rho([y|X]) = |\{x \in X \mid x \cong y\}|$. Thus, a species with respect to an ensemble can be represented by a complex and a population attribute: $[y|X] \equiv (y, \rho([y|X]))$. The set of all species with a population attribute is denoted $\mathbb{S}^\dagger = \mathbb{S} \times \mathbb{N}_{\geq 0}$. Under this notation, an ensemble of complexes X can be represented by a set of species with population counters,

$$S(X) = \{(s, \rho([s|X])) : s \in X/\cong\},$$

where $X/\cong$ is the quotient set of X by $\cong$.

We may choose to dispense with the reference to X entirely and directly represent a system as a *species ensemble*.

Definition 10 (Species Ensemble). *A species ensemble S is a set of species with population counters. More precisely,*

$$S \subset \{(s, N_s) \mid s \in \mathbb{S},\ N_s \in \mathbb{N}_{\geq 0}\}.$$

If $(s, N_s) \in S$, then we will abuse notation slightly and write $s \in S$ and define $\rho(s) = N_s$. In contrast to an ensemble of complexes, a species ensemble is a "population-based" representation of a system since complexes belonging to the same species are lumped together into a single object.

B.4 EMBEDDINGS OF PATTERN GRAPHS

An embedding, or "pattern match," is a map from a pattern to an instance of that pattern within a complex or, more generally, another pattern.

Definition 11 (Embedding). *Given patterns p and q, an* **embedding** *of p in q is a* **one-to-one** *mapping ϕ from the molecules and components of p into those of q that preserves the following properties: (i) graph structure* **compatibility***; (ii) component containership; (iii) type; (iv) state* **compatibility***. The non-specific bond '`+`' is compatible with any bond except the null bond '`-`'. The unspecified state wildcard '`?`' is compatible with any state.*

An embedding of pattern p in pattern q is denoted by $\phi_{p,q}$. If X is an ensemble of complexes, then $\texttt{emb}(p, X)$ is the set of all embeddings of pattern p into the complexes of X. Similarly, the set of embeddings from pattern p into *any* other pattern graph is denoted by $\texttt{emb}(p, \mathbb{P})$. Embeddings of a pattern graph into a species, e.g., $\phi \in \texttt{emb}(p, s)$, where $s \in \mathbb{S}$, should be interpreted as embeddings into the pattern graph representing the species.

It is often useful to refer to the pattern graph or species that contains the specific instance of an embedding. We call this the *target* of the embedding.

Definition 12 (Target). *Given an embedding ϕ, the* **target** *of ϕ, written $\texttt{targ}(\phi)$, is the pattern graph or species that contains the* **image** *of ϕ.*

B.5 REACTION RULES

A reaction rule represents a class of potential transformations on an ensemble of complexes.

Definition 13 (Reaction Rule)**.** *A reaction rule is defined by the tuple*

$$r \models \left((R_v)_{v=1}^{V},\ (P_w)_{w=1}^{W},\ \psi,\ k \right),$$

where V is the number of **reactant** *patterns, W is the number of* **product** *patterns, (R_v) are the reactant pattern graphs, (P_w) are the product pattern graphs, ψ is the* **component correspondence map***, and k is the* **propensity**,[1] *which may be given by a non-negative real number or function.*

The reactant pattern graphs describe motifs in complexes where a reaction event may occur. The product patterns describe the structure at the site of the motif subsequent to transformation under the action of the rule. The component correspondence map is a one-to-one, partial mapping of molecules and components in (R_v) to those of (P_w) that satisfies several properties (see below). The propensity k is defined such that $k{\cdot}dt$ is the probability for each rule instance (described next) to undergo transformation in the next infinitesimal time interval dt [25, 28]. In addition to a simple non-negative constant, the propensity may be a function of the ensemble of complexes (a "global function"), the target of a reactant pattern embedding (a "local function"), or both (a "composite function") [56].

The component correspondence map, ψ, defines how molecules and components in the reactant patterns correspond to those in the product patterns. The map is partial because a subset of molecules and components may be deleted or synthesized by the action of the rule. If a reactant molecule is not in the domain of ψ, $\texttt{dom}(\psi)$, then the molecule and its components are deleted. Similarly, if a product molecule is not in the image of ψ, $\texttt{im}(\psi)$, then the molecule and its components are synthesized. The differences in the bond and state configurations

[1]The traditional definition of the "propensity" is as the product of a constant, c, and a *combinatorial factor*, $h(\mathbf{X})$, which is a function of the system state $\mathbf{X}$ and represents the number of different ways in which specific instances of the reactant species can react [25, 28]. Here, we are dispensing with the combinatorial factor and using the term "propensity" to refer to a constant *or function* that is analogous to c in the traditional definition. We sometimes refer to this quantity as an "instance rate." Note that from a particle-based perspective, each complex can be thought of as a unique species with a unit population, i.e., $h(\mathbf{X}) = 1$. Thus, our definition of the propensity is simply a special case of the traditional definition.

of corresponding components imply a set of pattern graph transformations. For example, if two components in the reactant patterns do not share a bond edge but the corresponding components in the product patterns do, then a "bond addition" transformation is implied by the rule.

Definition 14 (Component Correspondence Map)**.** *Given reactant patterns* $(R_v)_{v=1}^{V}$ *and product patterns* $(P_w)_{w=1}^{W}$*, a component correspondence map* ψ *is a* **one-to-one, partial** *mapping from the molecules and components of* (R_v) *to those of* (P_w)*,*

$$\psi : \bigcup_{v} (M_{R_v} \cup C_{R_v}) \;\; \rightarrow \;\; \bigcup_{w} (M_{P_w} \cup C_{P_w}) \, ,$$

that satisfies the following properties: (i) molecules map to molecules and components map to components; (ii) types are preserved; (iii) molecule substructures are preserved; (iv) no new wildcards appear in the product graphs.

It is useful to define the notion of the "reaction center" of a rule, which is the subset of molecules and components in the reactant patterns that are modified by the action of the rule.

Definition 15 (Reaction Center)**.** *The reaction center of a rule* r*, written* $\texttt{RC}(r)$*, is the subset of reactant molecules not in the domain of* ψ *(deleted molecules)* **plus** *the subset of reactant components in the domain of* ψ *where the corresponding product component: (i) is bound if the reactant component is unbound; (ii) is unbound if the reactant component is bound; (iii) has a different state than the reactant component.*

Conversely, molecules and components in the reactant patterns of a rule that are *not* within the reaction center are called the "reaction context." In BNGL, the reaction context represents additional conditions necessary for a transformation encoded in a rule to take place. By convention, the reaction context does not contribute to the propensity of a rule, i.e., multiple matches to the context does not increase the value of the propensity (multiple matches to the center *does*). As such, it is necessary to equate embeddings where the images of the reaction center are the same even if the context differs.

Definition 16 (Embedding Equivalence)**.** *Given a rule r, with reaction center $\mathtt{RC}(r)$, the embeddings $\phi, \phi' \in \mathtt{emb}(R_v, \mathbb{P})$ are equivalent with respect to $\mathtt{RC}(r)$ if $\phi|_{\mathtt{RC}(r)} = \phi'|_{\mathtt{RC}(r)}$, where $\phi|_{\mathtt{RC}(r)}$ is the restriction of the embedding map ϕ to the molecules and components in $\mathtt{RC}(r)$.*

The equivalence class of embedding ϕ with respect to $\mathtt{RC}(r)$ is written $[\phi]$. When the context is clear, we will usually omit reference to $\mathtt{RC}(r)$. Furthermore, since the embedding class is represented by any of its members, we will usually omit the square braces. Note that all embeddings in an equivalence class have the same target. The set of all equivalence classes of embeddings of pattern R is denoted by $\mathtt{emb}(R, \mathbb{P}) / \mathtt{RC}(r)$.

In order to apply the transformation(s) encoded by a reaction rule, it is necessary to have an embedding of each reactant pattern into a target. A *rule instance* is a collection of embedding classes, one for each reactant pattern, whose targets can be transformed by the action of a rule.

Definition 17 (Rule Instance)**.** *Given reactant patterns $(R_v)_{v=1}^{V}$, a rule instance is a tuple of embedding classes $([\phi_v]) \in \prod_{v=1}^{V} \mathtt{emb}(R_v, \mathbb{P}) / \mathtt{RC}(r)$.*

The targets of a rule instance are denoted by the tuple $\mathtt{targ}(([\phi_v])) = \{\mathtt{targ}(\phi_v)\}_{v=1}^{V}$. Note that the targets are not necessarily distinct.

The *action* of a rule is defined by an operator that maps rule instances to products. The products are constructed by transforming the embedding targets according to the graph operations implied by the differences between the reactant patterns and the product patterns via the component correspondence map.

Definition 18 (Action of a Rule on a Rule Instance)**.** *Given a rule r with reactant patterns $(R_v)_{v=1}^{V}$, the* **action** *of r is given by an operator that maps rule instances to tuples of products,*

$$r[\,] : \prod_{v=1}^{V} \mathtt{emb}(R_v, \mathbb{P}) / \mathtt{RC}(r) \;\rightarrow\; (\mathbb{P})^{<\infty},$$

where $(\mathbb{P})^{<\infty}$ is the set of finite sequences of pattern graphs (although sometimes the pattern graph is the representative of a species).

The details of this operator are beyond the scope of the present work (see Sec. 3.1 of Blinov et al. [66] for a discussion within the context of network generation). It is sufficient for our purposes to simply state that we have a unique operator that maps rule instances to products.

It may seem awkward that the action of a rule maps embeddings to products rather than targets to products. However, this distinction is important because the same targets may be transformed into different products depending on the specific embedding into those targets. Thus, the action of a rule depends on both the targets and the embeddings, but since the targets are implicit in the choice of embeddings it is unnecessary to explicitly mention them in the definition. Also note that the number of products produced by the action of a rule is not necessarily equal to the number of product patterns in the rule. This is generally rectified by a post-action rejection step. For example, during network generation BioNetGen rejects all reactions generated from a rule that violate the product-side molecularity of that rule [60] (see below for further discussion).

If the targets of a rule instance belong to an ensemble of complexes, then we can naturally define the action of the rule on the ensemble. The instance targets are removed from the ensemble and replaced by the products generated by the action of the rule on the instance.

Definition 19 (Action of a Rule on an Ensemble of Complexes)**.** *Given an ensemble of complexes X, a rule r, and a rule instance $(\phi_v) \in \prod_v \mathtt{emb}(R_v, X)/\,\mathtt{RC}(r)$, the action of r on X* **through** *(ϕ_v), written $r[(\phi_v)](X)$, is*

$$X \mapsto (X \setminus \mathtt{targ}((\phi_v))) \cup r[(\phi_v)],$$

where $r[(\phi_v)]$ is the action of r on (ϕ_v).

The action of a rule on a species ensemble S is defined similarly except that targets and products belonging to the species in S are handled by modification of the population counter rather than set deletion and addition.

Definition 20 (Action of a Rule on a Species Ensemble)**.** *Given a species ensemble S, a rule r, and a rule instance $(\phi_v) \in \prod_v \mathtt{emb}(R_v, S)/\,\mathtt{RC}(r)$, the action of r on S through (ϕ_v) is*

$$S \mapsto \Big\{(s, N'_s) \mid s \in S, N'_s = N_s - \big|\{v \in \{1 \dots V_r\} \mid \mathtt{targ}(\phi_v) = s\}\big| + \big|\{w \in \{1 \dots W_r\} \mid p_w = s\}\big|\Big\}$$

where (p_w) are the products of the action $r[(\phi_v)]$.

Given an ensemble of complexes X, the set of all rule instances for rule r in X is defined to be

$$\mathtt{rinst}\big(r, X\big) = \prod_{v=1}^{V} \mathtt{emb}(R_v, X) / \,\mathtt{RC}(r).$$

This set, along with the rule action operator $r[\;]$, encapsulates every possible transformation on an ensemble of complexes by a reaction rule, and forms the basis for the NF simulation approach [53, 55, 56]. The set of all rule instances for a species ensemble S is defined analogously.

A few notes about rule instance propensities. The propensity of a rule instance is the probability of the rule acting on the instance in the infinitesimal time interval dt. There are several factors that determine this probability: (i) k_r, the propensity of the *rule*; (ii) the symmetry factor of the rule; (iii) the targets of the rule instance; (iv) the products of the action. The symmetry factor of a rule is a positive constant associated with the intrinsic symmetry of the reactant and product patterns of the rule. Its computation is the beyond the scope of this document (see Blinov et al. [66] for a discussion). The targets of the rule instance influence the propensity in two ways. First, the targets of the embeddings must be distinct complexes, i.e., the set of targets must have the same molecularity as the rule. If the targets are not distinct then the propensity is zero. Second, if the rule propensity is a *local function* (see Sneddon et al. [53]), then the local structure of the target can influence the propensity. The products can also influence the propensity, but only as a post-action rejection. While it can be shown that the products, in an aggregate sense, always contain embeddings of the product patterns, there is no guarantee that the products will have the same molecularity as the product patterns. After the products are constructed by the action of the rule, the molecularity of the products can be compared to that of the rule. If the molecularity is different, then the products are rejected and the targets restored. Since compliance with product molecularity is handled as a rejection step, the propensity of a rule instance is actually an upper bound on the true propensity.

Definition 21 (Propensity of a Rule Instance for an Ensemble of Complexes)**.** *Given a rule r with propensity k_r and symmetry constant s_r, an ensemble of complexes X, and a rule instance (ϕ_v), the propensity of the rule instance is*

$$a\left((\phi_v)\right) = \frac{k_r}{s_r} \prod_{v=1}^{V} \max\left\{0,\ 1 - \sum_{j<v} \delta(\phi_j, \phi_v)\right\},$$

where $\delta(\phi_j, \phi_v) = 1$ if $\texttt{targ}(\phi_j) = \texttt{targ}(\phi_v)$ and 0 otherwise.

Similarly, for a species ensemble,

Definition 22 (Propensity of a Rule Instance for a Species Ensemble)**.** *Given a rule r with propensity k_r and symmetry constant s_r, a species ensemble S, and a rule instance (ϕ_v), the propensity of the rule instance is*

$$a\left((\phi_v)\right) = \frac{k_r}{s_r} \prod_{v=1}^{V} \max\left\{0,\ \rho(\texttt{targ}(\phi_v)) - \sum_{j<v} \delta(\phi_j, \phi_v)\right\}.$$

Note that the propensity of a rule instance for an ensemble of complexes is just a special case of this definition [i.e., $\rho(\texttt{targ}(\phi_v)) = 1$ if $\texttt{targ}(\phi_v)$ is a complex $x \in X$].

B.6 BNGL MODELS

An *observable* defines a measurable output of a model system. Observables are composed of a set of pattern graphs, each corresponding to a molecular motif that is detectable by an experimental assay [60].

Definition 23 (Observable)**.** *An observable is defined by the tuple*

$$\mathcal{O} \models \left(\texttt{name}, \texttt{type} \in \{\text{"Molecules"}, \text{"Species"}\}, (P_v)_{v=1}^{V}\right),$$

where `name` *is a label for the observable,* `type` *is a property that determines how the observable is calculated, and (P_v) is a tuple of pattern graphs.*

The *population* of an observable is a measure of the number of embeddings of the pattern graphs within the system. The type of the observable, "Molecule" or "Species", determines the method of computing the population (see Faeder et al. [60]). For an ensemble of complexes X, the population of an observable is

$$\rho(\mathcal{O}|X) = \begin{cases} \sum_{v=1}^{V} |\texttt{emb}(P_v, X)| & : \text{ type} = \text{"Molecules"} \\ \\ \sum_{x \in X} H\Big(\sum_{v=1}^{V} |\texttt{emb}(P_v, x)|\Big) & : \text{ type} = \text{"Species"} \end{cases},$$

where $H(z) = 1$ if $z \geq 1$ and 0 otherwise. Similarly, for a species ensemble S,

$$\rho(\mathcal{O}|S) = \begin{cases} \sum_{s \in S} \rho(s) \sum_{v=1}^{V} |\texttt{emb}(P_v, s)| & : \text{ type} = \text{"Molecules"} \\ \\ \sum_{s \in S} \rho(s)\, H\Big(\sum_{v=1}^{V} |\texttt{emb}(P_v, s)|\Big) & : \text{ type} = \text{"Species"} \end{cases}.$$

A BNGL model completely specifies a dynamic system, including the initial system configuration, the reaction rules that govern the dynamics, and a set of observables that define the outputs of the system.

Definition 24 (Model)**.** *A BNGL model is defined by the tuple*

$$\mathcal{M} \models (\texttt{seed\_species}, \texttt{observables}, \texttt{reaction\_rules})\,,$$

where `seed_species` *is a subset of* $\mathbb{S}^\dagger$ *that specifies the initial state of the model system,* `observables` *is a set of observables, and* `reaction_rules` *is a set of reaction rules that govern the dynamics of the system.*

In practice, a BNGL model may also include parameters, molecule types (component-level type graphs), and function definitions [60], but we will omit these here for simplicity. Note that while seed species are always specified by a species ensemble in a model, the representation within a simulator may be as an ensemble of complexes (particle representation), a species ensemble (population representation), or a *hybrid ensemble* (see below).

APPENDIX C

RECEPTOR ACTIVATION MODEL

The complete receptor activation model, used as an illustration of energy modeling (Chapter 2), is listed here. The model file is also included in the `Models2` subfolder of BioNetGen distribution 2.2.4.

```
# Receptor activation model for energy BNGL
begin model

# requires BioNetGen version >= 2.2.4
version("2.2.4")
# Quantities have units in moles, so set this to Avogadro's Number
setOption("NumberPerQuantityUnit",6.0221e23)

begin parameters
   # fundamental constants
   RT  2.577      # kJ/mol
   NA  6.022e23   # /mol
   PI  3.142      # Pi, no units
   # Geometry parameters
   rad_cell   1e-4  # radius of cell, dm
   cell_dens  1e9   # density of cells, /L
   width_PM   1e-7  # effectice width of membrane, dm
   # Compartment volumes
   volEC  1/cell_dens               # vol. extracellular space, L
   volPM  4*PI*rad_cell^2*width_PM  # virtual vol. of plasma membrane, L
   volCP  4/3*PI*rad_cell^3         # vol. of cytoplasm, L
   # initial concentrations
   conc_L_0      20e-9   # mol/L
   count_R_0     24000   # molecules/cell
   conc_Ph_0     10e-9   # mol/L
   conc_ATP_0    1.0e-3  # mol/L
   conc_ADP_0    0.1e-3  # mol/L
   # standard free energy of formation, kJ/mol
   G_LR         -47.5
   G_RR         -11.9
   G_RPh         -41.5
   G_RyP          51.1
   G_ATP          51.1
   G_RyP_Ph       5.9
   G_LRR         -5.9
   G_LRRL        -11.9
   # baseline activation energy, kJ/mol
   E0_LR         -11.9
   E0_RR         -5.9
   E0_RPh        -17.8
   E0_catRR      -11.9
   E0_catPh       5.9
   # rate distribution parameter, no units
   phi  0.5
end parameters
begin compartments
   EC  3  volEC       # extracellular space
   PM  2  volPM  EC   # plasma membrane
   CP  3  volCP  PM   # cytoplasm
end compartments
begin molecule types
   L(r)            # ligand
   R(l,d,y~0~P)    # trans-membrane receptor
   Ph(y)       # phosphotase
   ATP()           # ATP
   ADP()           # ADP
end molecule types
begin seed species
   L@EC(r)         conc_L_0*NA*volEC
   R@PM(l,d,y~0)   count_R_0
   Ph@CP(y)        conc_Ph_0*NA*volCP
   $ATP@CP()       conc_ATP_0*NA*volCP      # ATP quantity held constant
   $ADP@CP()       conc_ADP_0*NA*volCP      # ADP quantity held constant
end seed species
begin observables
   Molecules  Lfree    L(r)
   Molecules  Ltotal   L()
   Molecules  Rbound   L(r!1).R(l!1)
   Molecules  Rdimer   R(d!1).R(d!1)
   Molecules  RyP      R(y~P!?)
   Molecules  Rtotal   R()
   Molecules  RyP_Ph   R(y!1).Ph(y!1)
   Molecules  PhTotal  Ph()
end observables
begin energy patterns
   # bond energy
   L(r!1).R(l!1)   G_LR/RT
   R(d!1).R(d!1)   G_RR/RT
   R(y!1).Ph(y!1)  G_RPh/RT
   # state energy
   R(y~P!?)        G_RyP/RT
   # molecule energy
   ATP()           G_ATP/RT
   # cooperativity
   R(y~P!1).Ph(y!1)                        G_RyP_Ph/RT
   L(r!1).R(l!1,d!2).R(d!2)                G_LRR/RT
   L(r!1).R(l!1,d!2).R(l!3,d!2).L(r!3)  G_LRRL/RT
end energy patterns
begin reaction rules
   # ligand binding
   L(r) + R(l)  <-> L(r!1).R(l!1)   Arrhenius(phi,E0_LR/RT)
   # receptor dimerization
   R(d) + R(d)  <-> R(d!1).R(d!1)   Arrhenius(phi,E0_RR/RT)
   # Ph binding
   R(y) + Ph(y) <-> R(y!1).Ph(y!1)  Arrhenius(phi,E0_RPh/RT)
   # trans-phosphorylation
       R(d!1).R(d!1,y~0) + ATP  \
   <-> R(d!1).R(d!1,y~P) + ADP  Arrhenius(phi,E0_catRR/RT)
   # dephosphorylation by phosphtase
       R(y~P!1).Ph(y!1)  \
   <-> R(y~0!1).Ph(y!1)  Arrhenius(phi,E0_catPh/RT)
end reaction rules
end model

# generate reaction network..
generate_network({overwrite=>1})

# simulate ODE system to steady state..
simulate({method=>"ode",t_end=>12,n_steps=>120,atol=>1e-3,rtol=>1e-7})
```

APPENDIX D

THE MONETARY COST OF SIMULATION MEMORY

We assume that each simulation is run on one virtual core (with possibly more than one simulation running per core) and there is no memory sharing among simulations. Let $\mathcal{I}$ be an Amazon EC2 instance type, $m_{\mathcal{I}}$ the total instance memory, $v_{\mathcal{I}}$ the number of virtual cores in the instance, $u_{v_{\mathcal{I}}}$ the number of compute units per virtual core, $C_{\mathcal{I}}$ the cost of the instance per unit time, and N the number of simultaneous simulations running on an instance. For each configuration pair $(\mathcal{I}, N)$, with $N \in \{1, 2, \ldots\}$, we will compute the cost (per unit time) per "effective" compute unit (ECU). We will then obtain the minimum cost by finding that (those) configuration(s) (multiple instances may have the same cost) with the lowest cost *under the constraint* that the instance(s) has (have) sufficient memory to run N simulations.

Under our assumption that a simulation is run on one virtual core, the average number of compute units available per simulation is the minimum of the number of compute units per core and the total number of compute units divided by the number of running simulations,

$$u_s(\mathcal{I}, N) = u_{v_{\mathcal{I}}} \cdot \min\{1, v_{\mathcal{I}}/N\}.$$

The number of ECUs is the number of running simulations multiplied by the average number of compute units per simulation,

$$\epsilon(\mathcal{I}, N) = N \cdot u_s(\mathcal{I}, N) = u_{v_{\mathcal{I}}} \cdot \min\{N, v_{\mathcal{I}}\}.$$

Note that the number of ECUs is equal to the total number of compute units if $N \geq v_{\mathcal{I}}$ and less if $N < v_{\mathcal{I}}$. The intuition here is that if the memory required per simulation limits the number of simulations that can be run simultaneously on an instance to fewer than the number of virtual cores, then we must treat the instance as if it had fewer virtual cores since some cannot be utilized.

The cost (per unit time) per ECU for a given configuration is then the cost of the instance divided by the number of ECUs,

$$C(\mathcal{I}, N) = \frac{C_{\mathcal{I}}}{\epsilon(\mathcal{I}, N)} = \frac{C_{\mathcal{I}}}{u_{v_{\mathcal{I}}} \cdot \min\{N, v_{\mathcal{I}}\}}.$$

For a given configuration, the memory available per simulation is

$$m(\mathcal{I}, N) = m_{\mathcal{I}}/N.$$

Assuming that each simulation run requires m_s units of memory, the minimum cost per ECU is obtained by finding the lowest configuration cost under the constraint that the configuration has sufficient memory available,

$$C_{\min}(m_s) = \min_{\text{all}\,(\mathcal{I},N)} \{C(\mathcal{I}, N) : m_s \leq m(\mathcal{I}, N)\}.$$

We need only consider $N \leq v_{\mathcal{I}}$ when searching for the minimum cost because $\epsilon(\mathcal{I}, N)$ is constant $(= u_{v_{\mathcal{I}}} v_{\mathcal{I}})$ for $N \geq v_{\mathcal{I}}$. $C_{\min}(m_s)$ has been calculated considering all Amazon EC2 *Standard*, *High-CPU*, and *High-Memory* instance types [141] and plotted in Fig. 8 of the main text. The configuration(s) associated with the minimum cost is (are) simply the set

$$\chi(m_s) = \{(\mathcal{I}, N) : C(\mathcal{I}, N) = C_{\min}(m_s) \;\wedge\; m_s \leq m(\mathcal{I}, N)\}.$$

Example: The Amazon EC2 *High-Memory Double Extra Large* instance has four virtual cores with 3.25 compute units each, 34.2 GB of total memory, and a cost of \$1.00/hr (January 2012 on-demand pricing, Linux operating system) [141]:

$$v_{\mathcal{I}} = 4,$$
$$u_{v_{\mathcal{I}}} = 3.25,$$
$$m_{\mathcal{I}} = 34.2,$$
$$C_{\mathcal{I}} = 1.00.$$

The number of compute units available per simulation is

$$u_s(\mathcal{I}, N) = 3.25 \cdot \min\{1, 4/N\},$$

and the number of ECUs is

$$\epsilon(\mathcal{I}, N) = 3.25 \cdot \min\{N, 4\}.$$

This gives a cost per ECU of

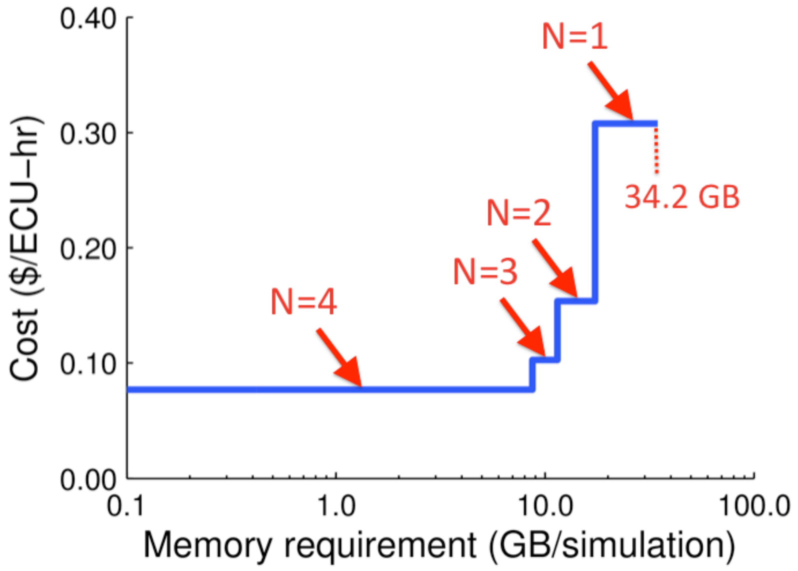
$$C(\mathcal{I}, N) = \frac{1.00}{3.25 \cdot \min\{N, 4\}},$$

with memory availability per simulation of

$$m(\mathcal{I}, N) = 34.2/N.$$

For this instance type, we show in Fig. D1 the minimum effective cost, $C^{\mathcal{I}}_{\min}(m_s)$, versus memory per simulation, m_s. Up to $m_s = 8.6$ GB it is possible to run four simultaneous simulations at a cost of $\sim$\$0.08/ECU-hr. We can run three simulations at $\sim$\$0.10/ECU-hr for $8.6 < m_s \leq 11.4$ GB, two simulations at $\sim$\$0.15/ECU-hr for $11.4 < m_s \leq 17.1$ GB, and one simulation at $\sim$\$0.31/ECU-hr for $17.1 < m_s \leq 34.2$ GB. Above $m_s = 34.2$ GB we obviously cannot run any simulations because the memory requirement exceeds the total memory available in the instance.

Figure D1: **Minimum cost of effective compute unit vs. memory required.** Results are shown for the Amazon EC2 *High-Memory Double Extra Large*. *Effective compute units* is the product of compute units per core and available CPU cores, less any idle cores due to insufficient memory.



APPENDIX E

FORMAL BASIS FOR HPP METHOD

E.1 FORMAL BASIS FOR THE PARTIAL NETWORK EXPANSION (PNE) ALGORITHM

E.1.1 Hybrid ensembles

HPP simulation requires the notion of a *hybrid ensemble*. A hybrid ensemble is a particle/population representation of a system, since part of the system is represented as individual particles and the remainder as lumped populations.

Definition 25 (Hybrid Ensemble)**.** *A hybrid ensemble is the pair (X, S), where X is an ensemble of complexes and S is a species ensemble.*

For convenience, if (X, S) is a hybrid ensemble and $x \in X$, then we define $\rho(x) = 1$. The action of a rule on a hybrid ensemble is defined by combining Defs. 19 and 20.

Definition 26 (Action of a Rule on a Hybrid Ensemble)**.** *Given a hybrid ensemble (X, S), a rule r, and a rule instance $(\phi_v) \in \prod_v \texttt{emb}(R_v, X \cup S)/\texttt{RC}(r)$, the action of r on (X, S) through (ϕ_v) is*

$$\begin{aligned} X &\mapsto \Big(X \setminus \texttt{targ}((\phi_v))\Big) \cup \Big(r[(\phi_v)] \setminus S\Big), \\ S &\mapsto \Big\{(s, N'_s) \mid s \in S, \\ &\qquad\quad N'_s = N_s - \big|\{v \in \{1 \ldots V_r\} \colon \texttt{targ}(\phi_v) = s\}\big| + \big|\{w \in \{1 \ldots W_r\} \colon p_w = s\}\big|\Big\} \end{aligned}$$

where (p_w) are the products of the action $r[(\phi_v)]$.

The set of all rule instances for a hybrid ensemble (X, S) is defined analogously to that for an ensemble of complexes,

$$\texttt{rinst}\big(r, X \cup S\big) = \prod_{v=1}^{V} \texttt{emb}(R_v, X \cup S) / \, \texttt{RC}(r).$$

The propensity of a rule instance for a hybrid ensemble is the same as defined in Def. 22. Finally, the population of an observable for a hybrid ensemble (X, S) is just the sum of the observable counts for the complexes and the species, i.e.,

$$\rho(\mathcal{O}|X,S) = \begin{cases} \sum_{v=1}^{V} |\texttt{emb}(P_v, X)| + \sum_{s \in S} \rho(s) \sum_{v=1}^{V} |\texttt{emb}(P_v, s)| \\ \qquad\qquad\qquad\qquad\qquad\qquad : \text{ type} = \text{“Molecules”} \\ \\ \sum_{x \in X} H\Big(\sum_{v=1}^{V} |\texttt{emb}(P_v, x)|\Big) + \sum_{s \in S} \rho(s)\, H\Big(\sum_{v=1}^{V} |\texttt{emb}(P_v, s)|\Big) \\ \qquad\qquad\qquad\qquad\qquad\qquad : \text{ type} = \text{“Species”} \end{cases}.$$

E.1.2 Child rules and rule restriction

A reaction rule r' is said to be a *child* of reaction rule r if the rule instances of r' are a subset of those of r and the action of r' is equivalent to r on that subset. In formal terms, there must be a rule instance $(\phi_v) \in \prod_v \texttt{emb}(R_v, R'_v)$ and a tuple of embeddings $(\phi_w) \in \prod_w \texttt{emb}(P_w, P'_w)$. Furthermore, it must follow that the component correspondence maps commute, i.e. $\phi' \circ \psi(x) = \psi' \circ \phi(x)$ for all $x \in \texttt{dom}(\psi)$, where ϕ and ϕ' are the unions of the respective embeddings. A child rule can be constructed from the pairing of a rule r with a rule instance (ϕ_v). The child rule has reactant patterns given by the targets of (ϕ_v) and product patterns constructed by the action of the rule on the instance.

Definition 27 (Restriction of a Rule by a Rule Instance)**.** *The* **child rule** *induced by the restriction of rule r by a rule instance (ϕ_v) is defined to be*

$$r|(\phi_v) = \big(\,\texttt{targ}((\phi_v))\,,\, r[(\phi_v)],\, \psi',\, k_r\,\big)\,,$$

where $r[(\phi_v)]$ are the products of the action of r on (ϕ_v) and ψ' is obtained from a **single pushout** *of ϕ and ψ (see Refs. [66, 201]). In general, the targets may be species or pattern graphs.*

Given a parent rule p, child rules r and r' are equivalent if the reactant patterns are isomorphic, the product patterns are isomorphic, and the component correspondence maps commute with the isomorphisms.

Definition 28 (Child Rule Equivalence)**.** *Child rules r and r' are equivalent, written $r \cong r'$, if the reactant patterns are isomorphic via mappings (ϕ_v), the product patterns are isomorphic via mappings (ϕ'_w), and the following hold:*

- $\phi(\texttt{dom}(\psi)) = \texttt{dom}(\psi')\,,$
- $\forall z \in \texttt{dom}(\psi) :\; \psi' \circ \phi(z) = \phi' \circ \psi(z),$

where ϕ and ϕ' are the unions of the embedding maps.

The class of all child rules equivalent to r is denoted by $[r]$. The equivalence relation partitions the set of child rules into classes. If R is a set of child rules, then the set of all equivalence classes in R is denoted $R/\cong$. Child rules in the same class can be treated as a single entity with a *multiplicity factor* that ensures that the propensity of the class is equal to the sum of the individual propensities. A proof is beyond the scope of this document.

E.1.3 Population-mapping rules

A species is "unstructured" if it is represented by a complex consisting of a single molecule that contains no components. Given a set of structured species S and a set of unstructured species U, where the types in U do not overlap with those in S, an "unstructured representation" of the species in S is a one-to-one mapping $\lambda : S \to U$. In the population-adapted NF simulator, complexes belonging to an unstructured species can be treated as a population (a species ensemble) rather than as individual particles (an ensemble of complexes). Thus, structured species can be treated as populations if an unstructured representation is provided *and* the reaction rules are partially expanded so that the dynamics of the unstructured species are equivalent to those of the structured counterpart. This process is accomplished by the PNE algorithm.

Prior to performing PNE, the modeler defines a set of *population-mapping rules*. Each mapping in an unstructured representation implies a population-mapping rule, and conversely a set of population-mapping rules implies an unstructured representation. Formally, a population-mapping rule is a reaction rule that matches complexes belonging to a single species and transforms them into the unstructured representation.

Definition 29 (Population-mapping rule)**.** *Given an unstructured representation $\lambda : S \to U$, a population-mapping rule for species $s \in S$ is a reaction rule*

$$l_s \models \big((s),\, (\lambda(s)),\, \varnothing,\, k_{\text{lump}} \big)\,,$$

where $k_{\text{lump}} \geq 0$ is the "lumping rate constant."

E.1.4 The PNE algorithm

Network expansion is the process of enumerating all of the specific reactions (at the resolution of species) implied by a set of reaction rules. The rule set produced by the PNE algorithm is a *partial* network expansion in the sense that the reactions are enumerated only with respect to the population species. Interactions among particles and between particles and population species are encoded by a set of child rules derived from the original rule set. Thus, there are three types of reaction rules among the set of children: (i) rules that govern the interaction of particles; (ii) mixed particle-population rules that govern the interaction of particles and population species; (iii) population *reactions* that govern the interactions among population species. Rules of type (i) are just duplicates of the original reaction rules. Rules of type (iii) are proper reactions since each reactant pattern matches exactly one species. Rules of type (ii) and (iii) together encapsulate every possible interaction among the population species and between population species and reactant motifs within particles.

In this sense, PNE is a hybrid of network-based and network-free methods: particles interact within an NF framework, populations interact within a network-based framework, and particle-population interactions are handled in a hybrid manner. Pseudocode for the PNE algorithm is presented as Algorithm 4. The method takes as input a BNGL model and a set of population-mapping rules and returns a model with a partially expanded rule set (an HPP model) that is dynamically equivalent to the original model for sufficiently large k_{lump} (or for any k_{lump} if the "exact" option is selected; see Algorithm 4).

When the HPP model is loaded into the population-adapted NF simulator, all of the unstructured species in U are represented as a species ensemble (i.e., as populations). The structured species, however, are represented as an ensemble of complexes X (i.e., as particles). Therefore, the system representation is a hybrid ensemble $(X, U^{\dagger})$. The propensities for rule instances are computed according to Def. 22. However, rule instances in which two or more reactant patterns have the same target are given the same propensity as other instances and then rejected in a post-sampling step. Furthermore, at simulation runtime, population-mapping rules serve the practical purpose of matching unlumped particles belonging to a population species, deleting the particle, and incrementing the population counter of the corresponding unstructured representation. Runtime lumping is required since the PNE algorithm is not guaranteed to identify every possible way that particle-particle or particle-population events can yield a product belonging to the set of population species (e.g., through complex dissociation). However, PNE does enumerate every way that a population species can participate in a reaction as a reactant.

E.2 EQUIVALENCE OF THE PNE RULE SET AND THE ORIGINAL RULE SET

We claim that PNE yields a system that is equivalent to the original model system in the sense that for any ensemble of complexes the rule set generated by PNE encompasses the same set of rule instances. We will outline a proof for the basic case where symmetry can be ignored.

Algorithm 4: Partial Network Expansion

```
input  :  model, a BNGL model (Def. 24);    $k_{\text{lump}}$, a lumping rate constant
          $\lambda : S \to U$, unstructured representation of the species in $S$ (Sec. E.1)
output:  hpp_model, the output model
// Transform seed species in $S$ into the unstructured representation
foreach $(s', n_{s'}) \in$ model.seed_species do
    if $s' \in S$ then
        add $(\lambda(s'), n_{s'})$ to hpp_model.seed_species
    else
        add $(s', n_{s'})$ to hpp_model.seed_species
foreach $s \in S$ do
    if $\nexists (s', n_{s'}) \in$ hpp_model.seed_species such that $s' \cong s$ then
        add $(\lambda(s), 0)$ to hpp_model.seed_species
// Reformulate observables (Def. 23) in terms of unstructured representation
foreach (name, type, $(P_v)$) $\in$ model.observables do
    $(P'_v) = (P_v)$ if type = "Molecules" then
        foreach pair $s \in S$, $P_v \in (P_v)$ do
            add $|\mathtt{emb}(P_v, s)|$ copies of $\lambda(s)$ to $(P'_v)$
    else
        foreach $s \in S$ do
            if $\cup_v \mathtt{emb}(P_v, s) \neq \varnothing$ then
                add $\lambda(s)$ to $(P'_v)$
    add (name, type, $(P'_v)$) to hpp_model.observables
// Expand reaction rules (Def. 13) around the species in $S$
foreach $r := \big( (R_v),\ (P_w),\ \psi,\ k \big) \in$ model.reaction_rules do
    // 1: Gather reactant pattern embeddings  foreach $R_v \in (R_v)$ do
        if $\nexists s \in S$ such that $s \cong R_v$ OR if "exact" method is desired then
            add $\mathtt{id}_{R_v,R_v}$ to embeddings$[v]$ // i.e., add the identity map of the reactant pattern
        add $\mathtt{emb}(R_v, S)/\mathtt{RC}(r)$ to embeddings$[v]$ // Def. 16
    // 2: Construct a child rule for each rule instance in the Cartesian product of embeddings
    foreach rule instance $(\phi_v) \in \prod_{v=1}^{V}$ embeddings$[v]$ do // Def. 17
        $\big( (R'_v),\ (P'_w),\ \psi',\ k \big)\ := \ r \mid (\phi_v)$ // Def. 27
        // 3: Replace instances of structured population species with unstructured counterparts
        foreach $R'_v \in (R'_v)$ do
            if $\exists s \in S$ such that $s = R'_v$ then // equality is required!
                substitute $\lambda(s)$ for $R'_v$
        foreach $P'_w \in (P'_w)$ do
            if $\exists s \in S$ such that $s \cong P'_w$ then // isomorphism is sufficient here
                substitute $\lambda(s)$ for $P'_w$
        add $\big( (R'_v),\ (P'_w),\ \psi',\ k \big)$ to child_rules
    // Compute equivalence classes of child rules (Def. 28) and add to hpp_model
    add (child_rules$/\cong$) to hpp_model.reaction_rules
// Add population-mapping rules (Def. 29)
foreach $s \in S$ do
    add $\big( (s),\ (\lambda(s)),\ \varnothing,\ k_{\text{lump}} \big)$ to hpp_model.reaction_rules
```

Let X be an ensemble of complexes and r a rule with reactant patterns $R_1, R_2, \ldots, R_N$. Furthermore, let $\mathfrak{s}$ be a set of "structured" species, $\mathfrak{u}$ be a set of "unstructured" species, and $m : \mathfrak{s} \to \mathfrak{u}$ be a one-to-one correspondence of structured species in $\mathfrak{s}$ and unstructured species in $\mathfrak{u}$. Each unstructured species in $\mathfrak{u}$ is complex consisting of a single molecule, whose type is not in the original model, that has no components.

Suppose $X_{\text{po}} \subseteq \{x \in X : \exists s \in \mathfrak{s} \text{ such that } x \cong s\} \subseteq X$, and let $X_{\text{pa}} = X \setminus X_{\text{po}}$. let $H = (X_{\text{po}}, S_{\mathfrak{s}}(X_{\text{pa}})$ be the representation of X as a hybrid ensemble. In words, X_{po} is a subset of the complexes in X that will be represented by an ensemble of species, and X_{pa} is the subset of complexes represented as an ensemble of complexes. Take note that we have not assumed that all complexes belonging to species in $\mathfrak{s}$ are in X_{po}.

We begin by partitioning the set of rule instances of r in X into the various classes of particle-particle interactions, particle-population interactions, and population-population interactions. To be precise, let $(Y_n)_{n=1}^N \in \prod_{n=1}^N \{X_{\text{pa}}, X_{\text{po}}\}$ and define the subset of rule instances where embedding n targets a complex in Y_n

$$\text{rinst}(r;(Y_n)) = \Big\{ \Big((\phi_n), k_r \Delta\big((\phi_n)\big) \Big) : \forall n \, (\phi_n \in \texttt{emb}(R_n, Y_n)) \Big\}.$$

Observe that the subsets $\texttt{rinst}\big(r;(Y_n)\big)$ partition the set of rule instances since:

$$\bigcup_{(Y_n) \in \prod_{n=1}^N \{X_{\text{pa}}, X_{\text{po}}\}} \texttt{rinst}\big(r;(Y_n)\big) = \texttt{rinst}\big(r, X\big)$$

and

$$(Y_n) \neq (Z_n) \implies \texttt{rinst}\big(r;(Y_n)\big) \cap \texttt{rinst}\big(r;(Z_n)\big) = \varnothing.$$

Since the general notation is tedious, let us illustrate with a bimolecular reaction rule (which is also the first interesting case):

$$\begin{aligned}
\texttt{rinst}\big(r, X\big) = \; & \texttt{rinst}\big(r;(X_{\text{pa}}, X_{\text{pa}})\big) \cup \texttt{rinst}\big(r;(X_{\text{pa}}, X_{\text{po}})\big) \\
& \cup \texttt{rinst}\big(r;(X_{\text{po}}, X_{\text{pa}})\big) \cup \texttt{rinst}\big(r;(X_{\text{po}}, X_{\text{po}})\big) \\
= \; & \big\{ \big(\phi_1, \phi_2, k\Delta\big((\phi_1,\phi_2)\big)\big) : \phi_1 \in \texttt{emb}(R_1, X_{\text{pa}}), \phi_2 \in \texttt{emb}(R_2, X_{\text{pa}}) \big\} \bigcup \\
& \big\{ \big(\phi_1, \phi_2, k\Delta\big((\phi_1,\phi_2)\big)\big) : \phi_1 \in \texttt{emb}(R_1, X_{\text{po}}), \phi_2 \in \texttt{emb}(R_2, X_{\text{pa}}) \big\} \bigcup \\
& \big\{ \big(\phi_1, \phi_2, k\Delta\big((\phi_1,\phi_2)\big)\big) : \phi_1 \in \texttt{emb}(R_1, X_{\text{pa}}), \phi_2 \in \texttt{emb}(R_2, X_{\text{po}}) \big\} \bigcup \\
& \big\{ \big(\phi_1, \phi_2, k\Delta\big((\phi_1,\phi_2)\big)\big) : \phi_1 \in \texttt{emb}(R_1, X_{\text{po}}), \phi_2 \in \texttt{emb}(R_2, X_{\text{po}}) \big\}.
\end{aligned}$$

Notice that the the first term corresponds to rule instances among particles in X_{pa}, the second and third terms correspond to mixed interactions among particles in X_{pa} and populations of X_{po}, and the the fourth term corresponds to interactions among populations in X_{po}.

Next, let us group rule instances by the species of the target complexes in X_{po}:

$$\begin{aligned}\texttt{rinst}\big(r, X\big) = \ \dots \ = & \\ & \big\{\big(\phi_1, \phi_2, k\Delta\big((\phi_1,\phi_2)\big)\big)\,|\phi_1 \in \texttt{emb}(R_1, X_{\mathrm{pa}})\,, \phi_2 \in \texttt{emb}(R_2, X_{\mathrm{pa}})\big\} \bigcup \\ & \bigcup_{s\in\mathfrak{s}} \big\{\big(\phi_1, \phi_2, k\Delta\big((\phi_1,\phi_2)\big)\big)\,|\phi_1 \in \texttt{emb}(R_1, [s|X_{\mathrm{po}}])\,, \phi_2 \in \texttt{emb}(R_2, X_{\mathrm{pa}})\big\} \bigcup \\ & \bigcup_{s\in\mathfrak{s}} \big\{\big(\phi_1, \phi_2, k\Delta\big((\phi_1,\phi_2)\big)\big)\,|\phi_1 \in \texttt{emb}(R_1, X_{\mathrm{pa}})\,, \phi_2 \in \texttt{emb}(R_2, [s|X_{\mathrm{po}}])\big\} \bigcup \\ & \bigcup_{s_1\in\mathfrak{s}} \bigcup_{s_2\in\mathfrak{s}} \big\{\big(\phi_1, \phi_2, k\Delta\big((\phi_1,\phi_2)\big)\big)\,|\phi_1 \in \texttt{emb}(R_1, [s_1|X_{\mathrm{po}}])\,, \phi_2 \in \texttt{emb}(R_2, [s_2|X_{\mathrm{po}}])\big\},\end{aligned}$$

recalling that $\texttt{emb}(R_n, [s|X_{\mathrm{po}}])$ is the set of embeddings from R_n into any complex $x \in X_{\mathrm{po}}$ such that $x \cong s$.

And now we further partition the rule instances by the specific embeddings:

$$\begin{aligned}\texttt{rinst}\big(r, X\big) = \ \dots \ = & \\ & \{(\phi_1, \phi_2, k\delta) : \phi_1 \in \texttt{emb}(R_1, X_{\mathrm{pa}})\,, \phi_2 \in \texttt{emb}(R_2, X_{\mathrm{pa}})\} \bigcup \\ & \bigcup_{s\in\mathfrak{s}} \ \bigcup_{\phi_1\in\texttt{emb}(R_1,s)} \ \bigcup_{x_1\in X_{\mathrm{po}}} \{(\texttt{id}_{s,x_1} \circ \phi_1, \phi_2, k\delta) : \phi_2 \in \texttt{emb}(R_2, X_{\mathrm{pa}})\} \bigcup \\ & \bigcup_{s\in\mathfrak{s}} \ \bigcup_{\phi_2\in\texttt{emb}(R_2,s)} \ \bigcup_{x_2\in X_{\mathrm{po}}} \{(\phi_1, \texttt{id}_{s,x_2} \circ \phi_2, k\delta) : \phi_1 \in \texttt{emb}(R_1, X_{\mathrm{pa}})\} \bigcup \\ & \bigcup_{s_1\in\mathfrak{s}} \ \bigcup_{\phi_1\in\texttt{emb}(R_1,s_1)} \ \bigcup_{x_1\in X_{\mathrm{po}}} \ \bigcup_{s_2\in\mathfrak{s}} \ \bigcup_{\phi_2\in\texttt{emb}(R_1,s_1)} \ \bigcup_{x_2\in X_{\mathrm{po}}} \{(\texttt{id}_{s_1,x_1} \circ \phi_1, \texttt{id}_{s_2,x_2} \circ \phi_2, k\delta)\}\,,\end{aligned}$$

where $\texttt{id}_{s,x}$ is the map from the complex representing species s into complex x (which belongs to s) induced by an isomorphism of s and x.

Next, we can reformulate our rule instances in terms hybrid rule instances of r in the hybrid ensemble representation, $H = (X_{\text{pa}}, S(X_{\text{po}}))$:

$$\begin{aligned}\texttt{rinst}\big(r, X\big) = \;\ldots\; \equiv\; & \big\{\big(\phi_1, \phi_2, k\eta\big(\phi_1,\phi_2\big)\big) : \phi_1 \in \texttt{emb}(R_1, X_{\text{pa}})\,, \phi_2 \in \texttt{emb}(R_2, X_{\text{pa}})\big\} \bigcup \\ & \bigcup_{s\in S(X_{\text{po}})} \bigcup_{\phi_1\in\texttt{emb}(R_1,s)} \{\big(\phi_1, \phi_2, k\eta\big(\phi_1,\phi_2\big)\big) : \phi_2 \in \texttt{emb}(R_2, X_{\text{pa}})\} \bigcup \\ & \bigcup_{s\in S(X_{\text{po}})} \bigcup_{\phi_2\in\texttt{emb}(R_2,s)} \{\big(\phi_1, \phi_2, k\eta\big(\phi_1,\phi_2\big)\big) : \phi_1 \in \texttt{emb}(R_1, X_{\text{pa}})\} \bigcup \\ & \bigcup_{s_1\in S(X_{\text{po}})} \bigcup_{\phi_1\in\texttt{emb}(R_1,s_1)} \bigcup_{s_2\in S(X_{\text{po}})} \bigcup_{\phi_2\in\texttt{emb}(R_2,s_2)} \{\big(\phi_1, \phi_2, k\eta\big(\phi_1,\phi_2\big)\big)\}.\end{aligned}$$

Recall that if s is a species, then $\texttt{emb}(R_2, s)$ is the set of embeddings into *a representative complex* for species s, and *not* the set of embeddings into *all complexes* belonging to species s. The population of complexes belonging to species s is factored into the η function.

The reader may have noticed that we have a set of hybrid rule instances, but we have not specified the corresponding hybrid rules. Now we equate each of the above hybrid rule instance with the equivalent hybrid rule that generates those instances.

$$\begin{aligned}\texttt{rinst}\big(r, X\big) = \;\ldots\; \equiv\; & \texttt{rinst}\big(r|(\texttt{id}_{R_1}, \texttt{id}_{R_2}), H\big) \bigcup \\ & \bigcup_{s\in\mathfrak{s}} \bigcup_{\phi_1\in\texttt{emb}(R_1,s)} \texttt{rinst}\big(r|(\phi_1, \texttt{id}_{R_2}), H\big) \bigcup \\ & \bigcup_{s\in\mathfrak{s}} \bigcup_{\phi_2\in\texttt{emb}(R_2,s)} \texttt{rinst}\big(r|(\texttt{id}_{R_1}, \phi_2), H\big) \bigcup \\ & \bigcup_{s_1\in\mathfrak{s}} \bigcup_{\phi_1\in\texttt{emb}(R_1,s_1)} \bigcup_{s_2\in\mathfrak{s}} \bigcup_{\phi_2\in\texttt{emb}(R_2,s_2)} \texttt{rinst}\big(r|(\phi_1, \phi_2), H\big)\,.\end{aligned}$$

Regrouping terms, we find the the hybrid rules are generated from the restriction of rule r by bundles in the cartesian product of two embedding sets

$$\texttt{rinst}\big(r, X\big) = \;\ldots\; \equiv \left\{\begin{aligned} & \texttt{rinst}\big(r|(\phi_1, \phi_2), H\big) : \\ & \phi_1 \in \{\texttt{id}_{R_1}\} \bigcup_{s_1\in\mathfrak{s}} \texttt{emb}(R_1, s_1)\,,\; \phi_2 \in \{\texttt{id}_{R_2}\} \bigcup_{s_2\in\mathfrak{s}} \texttt{emb}(R_2, s_2)\end{aligned}\right\}.$$

The embedding sets are precisely those used in the first step of PNE: the identity embedding of the reactant pattern in itself along with every embedding of the reactant pattern in the structured population species.

Therefore, the PNE rule set given by hybrid rules

$$\left\{ r|(\phi_1,\phi_2) : \phi_1 \in \{\mathtt{id}_{R_1}\} \bigcup_{s_1 \in \mathfrak{s}} \mathtt{emb}(R_1, s_1)\,,\ \phi_2 \in \{\mathtt{id}_{R_2}\} \bigcup_{s_2 \in \mathfrak{s}} \mathtt{emb}(R_2, s_2) \right\},$$

operating on the hybrid ensemble H, is equivalent to the orignal rule r operating on the ensemble of complexes X.

The final step of PNE is substituting instances of structured population species in reaction rules with the corresponding unstructured species. Since the map between structured and unstructured representations is one-to-one and onto, there is no loss of fidelity due to sustitution of one for the other.

E.2.1 The lumping rate constant and exactness of PNE

The above proof does not assume that all complexes with species in $\mathfrak{s}$ belong to X_{po}. Consequently, the HPP simulation is equivalent to the network-free simulation even if the lumping rate is finite. However, equivalence does depend upon inclusion of all particle-particle and particle-population rules. As noted in the main text, PNE excludes such rules from the final set if the reactant pattern is isomorphic to a population species. This is done for pragmatic reasons, since those rules add extra overhead to the simulation. If the lumping rate is sufficiently large, lumpable particles are rapidly converted to the population form and the results are sufficiently accurate to appear statistically identical. This being the case, we felt the overhead of extra rules was a greater burden than a slight loss in accuracy. If loss in accuracy is a concern, the lumping rate may be set to an infinite value, *i.e. instantaneous lumping.*

APPENDIX F

GENERATING AND SIMULATING HPP MODELS WITH BNG/NFSIM

From a BioNetGen user's perspective, once a set of species has been selected for lumping, converting a standard BNGL model into an HPP model requires only two additional steps. First, the population-mapping rules (see "Population species and population-mapping rules" in the main text) must be specified within a `population maps` block of the BNGL model file. Note that this is the *only* place where the population species should be specified, they should not be included in the `molecule types` or `seed species` blocks (Sec. 1.4.8). The lumping rate constant should be set to a large value (NFsim does not currently support infinite rates) and can be specified either as a number or as a reference to a parameter in the `parameters` block.

Next, the `generate_hybrid_model()` action must be invoked after the model specification. Arguments are passed to `generate_hybrid_model()` in a comma-separated list enclosed by curly braces and each argument/value pair has the form `argument=>value`. Allowed arguments are:

- `overwrite=>0/1`: Disable/enable automatic overwriting of any existing hybrid model (default: `0`).
- `execute=>0/1`: Disable/enable execution of actions after generation of hybrid model (default: `0`).
- `verbose=>0/1`: Disable/enable verbose output during hybrid model generation (default: `0`).

- `suffix=>"string"`: Set the HPP model suffix to value of "`string`" (default: "`hpp`").
- `actions=>["action1","action2",...]`: Define a list of actions (as strings) to be included at the end of the HPP model file.
- `exact=>0/1`: Disable/enable exact mode (default: `0`); *BioNetGen 2.2.3 or later.*

A call to `generate_hybrid_model()` instructs BioNetGen to generate and write the HPP model to the file `[model]_[suffix].bngl`, where `[model]` is the original BNGL filename and `[suffix]` is "`hpp`" by default and can be changed by the user using the `suffix` argument.

In generating the HPP model, BioNetGen automatically appends all of the population species defined in the `population maps` block to the original `molecule types` block, each followed by a `population` keyword to distinguish it as a population type. The `seed species` block is modified by replacing any structured species selected for lumping with their population species counterparts. The model observables are also modified to account for the introduction of the population species, which is straightforward. Observables are collections of one or more patterns [60, 61]. Therefore, each structured species selected for lumping is matched against each pattern in each observable and if a match is found then the associated population species is added to the observable list. Finally, PNE is performed to obtain the expanded rule set, to which the population-mapping rules are appended in the `reaction rules` block. A before-and-after illustration of this process can be seen by comparing the original (Text S2) and partially-expanded (Text S3) versions of the simple receptor activation model considered in "Partial network expansion" and Figs. 2 and 3 of the main text.

If `execute=>1` is passed to `generate_hybrid_model()`, BioNetGen will additionally perform any actions on the HPP model that are listed in the `actions` argument. For example, a NFsim simulation can be run through BioNetGen by calling the `simulate_nf()` action [61, 62], e.g.,

```
generate_hybrid_model({execute=>1, \
                    actions=>["simulate_nf({t_end=>10,n_steps=>100})"]}).
```

Alternatively, a XML encoding of the HPP model can be obtained by invoking the `writeXML()` action:

```
generate_hybrid_model({execute=>1,actions=>["writeXML()"]}).
```

This allows NFsim simulations to be run in standalone mode (see Refs. [53, 73]). Note that NFsim version 1.11 or later is required to run HPP simulations.

APPENDIX G

HPP ILLUSTRATIVE EXAMPLE, COMPLETE BNGL SOURCE

G.1 ORIGINAL MODEL

```
## recAct_example.bngl
##
## Simple model for illustration of HP/P method.
##
## This model is a simplified variant of the EGF receptor
## activation model published in:
##   ML Blinov, JR Faeder, B Goldstein, WS Hlavacek.
##   "A network model of early events in epidermal growth factor
##     receptor signaling that accounts for combinatorial complexity."
##   BioSystems 83, 136-151 (2006).
begin model
begin parameters
    # fraction of cell, no units
    f       0.10
    # Avogadro's number, /mol
    NA      6.0221e23
    # Volume, liters
    V       1e-12*f
    # initial species counts
    L0      500e-9*NA*V
    R0      100e-9*NA*V
    A0      100e-9*NA*V
    B0      100e-9*NA*V
    C0      100e-9*NA*V
    BC0     100e-9*NA*V
    # rate constants, units /s
    kp1     10e6/(NA*V)
    km1     1.0
    k2      1.0
    k3      1.0
    kp4     10e6/(NA*V)
    km4     1.0
    kp5     10e6/(NA*V)
    km5     1.0
    kp6     10e6/(NA*V)
    km6     1.0
    k7      1.0
    k8      1.0
    kp9     10e6/(NA*V)
    km9     1.0
    # population lumping parameter
    klump  10000
end parameters
begin molecule types
    L(r)                # ligand molecule
    R(l,a~0~P,b~0~P)    # transmembrane receptor
    A(r,b~0~P)          # cytosolic mediator A
    B(r,c)              # cytosolic mediator B
    C(b)                # cytosolic mediator C
end molecule types
begin seed species
    L(r)                 L0
    R(l,a~0,b~0)         R0
    A(r,b~0)             A0
    B(r,c)               B0
    C(b)                 C0
    B(r,c!1).C(b!1)      BC0
end seed species
begin observables
    Molecules    LR       L(r!1).R(l!1)
    Molecules    Rp       R(a~P!?)  R(b~P!?)
    Molecules    Ap       A(b~P!?)
    Molecules    RC       R(b~P!1).B(r!1,c!2).C(b!2),\
                            R(a~P!1).A(r!1,b~P!2).B(r!2,c!3).C(b!3)
    # total counts (conserved)
    Molecules    Ltot     L()
    Molecules    Rtot     R()
    Molecules    Atot     A()
    Molecules    Btot     B()
    Molecules    Ctot     C()
end observables
begin reaction rules
    # ligand receptor binding
    L(r) + R(l)  <->  L(r!1).R(l!1)     kp1, km1
    # ligand induced receptor phosphorylation
    L(r!1).R(l!1,a~0)  ->  L(r!1).R(l!1,a~P)     k2
    L(r!1).R(l!1,b~0)  ->  L(r!1).R(l!1,b~P)     k2
    # receptor dephosphorylation
    R(a~P)  ->  R(a~0)     k3
    R(b~P)  ->  R(b~0)     k3
    # phosphorylated receptor binding A
    R(a~P) + A(r)  <->  R(a~P!1).A(r!1)     kp4, km4
    # phosphorylated receptor binding B
    R(b~P) + B(r)  <->  R(b~P!1).B(r!1)     kp5, km5
    # B binding C
    B(c) + C(b)  <->  B(c!1).C(b!1)    kp6, km6
    # bound A phosphorylation
    R(a~P!1).A(r!1,b~0)  ->  R(a~P!1).A(r!1,b~P)    k7
    # A dephosphorylation
    A(b~P)  ->  A(b~0)    k8
    # phosphorylated A binding B
    A(b~P) + B(r)  <->  A(b~P!1).B(r!1)  kp9, km9
end reaction rules
begin population maps
    L(r)                             ->  P1()    klump
    A(r,b~0)                         ->  P2()    klump
    A(r,b~P)                         ->  P3()    klump
    A(r,b~P!1).B(r!1,c)              ->  P4()    klump
    A(r,b~P!1).B(r!1,c!2).C(b!2)     ->  P5()    klump
    B(r,c)                           ->  P6()    klump
    B(r,c!1).C(b!1)                  ->  P7()    klump
    C(b)                             ->  P8()    klump
end population maps
end model

## model actions ##
saveConcentrations()
# simulate NF
setParameter("f",0.01)
resetConcentrations()
simulate_nf({suffix=>"nf",t_end=>40,n_steps=>120,gml=>1000000,verbose=>1})

# generate hybrid particle/population model and simulate
setParameter("f",0.01)
resetConcentrations()
generate_hybrid_model({overwrite=>1,verbose=>1,execute=>1,\
    actions=>["simulate_nf({t_end=>40,n_steps=>120,\
                gml=>1000000,verbose=>1})"]})

# generate reaction network and simulate SSA
generate_network({overwrite=>1})
setParameter("f",0.01)
resetConcentrations()
simulate_ssa({suffix=>"ssa",t_end=>40,n_steps=>120})
```

G.2 HPP MODEL

```
# recAct_example_hpp.bngl
#
# Created by BioNetGen 2.2.3
substanceUnits("Number")

begin model
begin parameters
  f      0.01
  NA     6.0221e23
  V      1e-12*f
  L0      (500e-9*NA)*V
  R0      (100e-9*NA)*V
  A0      (100e-9*NA)*V
  B0      (100e-9*NA)*V
  C0      (100e-9*NA)*V
  BC0     (100e-9*NA)*V
  kp1     10e6/(NA*V)
  km1     1.0
  k2      1.0
  k3      1.0
  kp4     10e6/(NA*V)
  km4     1.0
  kp5     10e6/(NA*V)
  km5     1.0
  kp6     10e6/(NA*V)
  km6     1.0
  k7      1.0
  k8      1.0
  kp9     10e6/(NA*V)
  km9     1.0
  klump   10000
end parameters
begin molecule types
  A(r,b~0~P)
  P5()                population
  P7()                population
  P2()                population
  P1()                population
  B(r,c)
  P6()                population
  C(b)
  P3()                population
  R(l,a~0~P,b~0~P)
  P8()                population
  P4()                population
  L(r)
end molecule types
begin observables
  Molecules LR L(r!1).R(l!1)
  Molecules Rp R(a~P!?), R(b~P!?)
  Molecules Ap A(b~P!?), P3(), P4(), P5()
  Molecules RC R(b~P!1).B(r!1,c!2).C(b!2), \
               R(a~P!1).A(r!1,b~P!2).B(r!2,c!3).C(b!3)
  Molecules Ltot L() P1()
  Molecules Rtot R()
  Molecules Atot A() P2() P3() P4() P5()
  Molecules Btot B() P4() P5() P6() P7()
  Molecules Ctot C() P5() P7() P8()
end observables
begin species
  P1()            L0
  R(a~0,b~0,l)    R0
  P2()            A0
  P6()            B0
  P8()          C0
  P7()          BC0
  P3()          0
  P4()          0
  P5()          0
end species
begin reaction rules
  Rule1_v1:  P1() + R(l) -> L(r!1).R(l!1)  kp1
  Rule1r_v1:  L(r!1).R(l!1) -> P1() + R(l)  km1 DeleteMolecules
  Rule2_v1:  L(r!1).R(a~0,l!1) -> L(r!1).R(a~P,l!1)  k2
  Rule3_v1:  L(r!1).R(b~0,l!1) -> L(r!1).R(b~P,l!1)  k2
  Rule4_v1:  R(a~P) -> R(a~0)  k3
  Rule5_v1:  R(b~P) -> R(b~0)  k3
  Rule6_v1:  R(a~P) + A(r) -> A(r!1).R(a~P!1)  kp4
  Rule6_v2:  R(a~P) + P2() -> A(b~0,r!1).R(a~P!1)  kp4
  Rule6_v3:  R(a~P) + P3() -> A(b~P,r!1).R(a~P!1)  kp4
  Rule6_v4:  R(a~P) + P4() -> A(b~P!1,r!2).B(c,r!1).R(a~P!2)  kp4
  Rule6_v5:  R(a~P) + P5() -> A(b~P!1,r!2).B(c!3,r!1).C(b!3).R(a~P!2)  kp4
  Rule6r_v1:  A(r!1).R(a~P!1) -> R(a~P) + A(r)  km4
  Rule7_v1:  R(b~P) + B(r) -> B(r!1).R(b~P!1)  kp5
  Rule7_v2:  R(b~P) + P6() -> B(c,r!1).R(b~P!1)  kp5
  Rule7_v3:  R(b~P) + P7() -> B(c!1,r!2).C(b!1).R(b~P!2)  kp5
  Rule7r_v1:  B(r!1).R(b~P!1) -> R(b~P) + B(r)  km5
  Rule8_v1:  B(c) + P8() -> B(c!1).C(b!1)  kp6
  Rule8_v2:  P4() + P8() -> P5()  kp6
  Rule8_v3:  P6() + P8() -> P7()  kp6
  Rule8r_v1:  B(c!1).C(b!1) -> B(c) + P8()  km6 DeleteMolecules
  Rule8r_v2:  P5() -> P4() + P8()  km6
  Rule8r_v3:  P7() -> P6() + P8()  km6
  Rule9_v1:  A(b~0,r!1).R(a~P!1) -> A(b~P,r!1).R(a~P!1)  k7
  Rule10_v1:  A(b~P) -> A(b~0)  k8
  Rule10_v2:  P3() -> P2()  k8
  Rule11_v1:  A(b~P) + B(r) -> A(b~P!1).B(r!1)  kp9
  Rule11_v2:  A(b~P) + P6() -> A(b~P!1).B(c,r!1)  kp9
  Rule11_v3:  A(b~P) + P7() -> A(b~P!1).B(c!2,r!1).C(b!2)  kp9
  Rule11_v4:  P3() + B(r) -> A(b~P!1,r).B(r!1)  kp9
  Rule11_v5:  P3() + P6() -> P4()  kp9
  Rule11_v6:  P3() + P7() -> P5()  kp9
  Rule11r_v1:  A(b~P!1).B(r!1) -> A(b~P) + B(r)  km9
  Rule11r_v2:  P4() -> P3() + P6()  km9
  Rule11r_v3:  P5() -> P3() + P7()  km9
  MapRule0:  L(r) -> P1()  klump
  MapRule1:  A(b~0,r) -> P2()  klump
  MapRule2:  A(b~P,r) -> P3()  klump
  MapRule3:  A(b~P!1,r).B(c,r!1) -> P4()  klump
  MapRule4:  A(b~P!1,r).B(c!2,r!1).C(b!2) -> P5()  klump
  MapRule5:  B(c,r) -> P6()  klump
  MapRule6:  B(c!1,r).C(b!1) -> P7()  klump
  MapRule7:  C(b) -> P8()  klump
end reaction rules
end model

## model actions ##
simulate_nf({t_end=>40,n_steps=>120,gml=>1000000,verbose=>1})
```

APPENDIX H

SEPSIS MODELS, COMPLETE BNGL SOURCE

H.1 SEPSIS MODEL 1

```
# Sepsis Model #1, BNGL format
version("2.2.3")
begin model
begin parameters
   # compartment volumes. See "the Laboratory Rat". Sharp, et al. 1998.
   # 250 g rat, extracell. fluid: 74.2 ml (incl. plasma), plasma: 7.8 ml
   volB   7.8             # ml
   volP   7.8             # ml
   volT   58.6            # ml
   # initial conecentrations
   P0     2000            # arbitrary units (a.u.)/ml
   M0     50              # a.u./ml
   E0     50              # a.u./ml
   N0     500             # a.u./ml
   # pathogen
   sP_P   0.168           # pathogen growth rate, /hr
   maxP   1e5             # max pathogen, a.u./ml
   dP_P   sP_P/maxP       # pathogen competition, /(a.u./ml)/hr
   # macrophage/pathogen
   bMP    0.006           # path-macro binding rate, /(a.u./ml)/hr
   sL_M   7.98            # IL1 production by macroph, pmol/hr/a.u.
   dP_M   2.76            # path. elimination rate, after binding, /hr
   # neutrophil/pathogen
   bNP    0.006           # path-neutr binding rate, /(a.u./ml)/hr
   dP_N   2.76            # path. elimination rate, after binding, /hr
   sA_N   6.0             # IL1ra production by neutro., pmol/hr/a.u.
   # neutrophil source/death
   sN_B   30.0            # source of neutrophils in blood, a.u./hr
   dN     0.06            # decay rate of neutr. /hr
   # IL1,IL1ra receptor binding
   bLE    0.0046          # IL1-IL1R binding rate, /pM/hr (Kd = 150pM)
   uLE    0.70              # IL1-IL1R unbinding rate, /hr
   # endothelial cells
   aE_L   600.              # endo. activation rate, step 1, /hr (fast)
   aE     0.348             # endo. activation rate, step 2, /hr (slow)
   dE     0.174             # rate of endo. de-activation, /hr
   # neutrophil recruitment
   bEN    0.00498           # endo-neutr binding rate, /(a.u./ml)/hr
   tN_E   4.14              # migration rate, /hr
   # cytokine "leak" between tissues to blood
   tL_PB  1.4               # transport rate IL1,IL1ra from P to B, /hr
   tL_TB  tL_PB             # transport rate IL1,IL1ra from P to B, /hr
   tL_BP  volP/volB*tL_PB   # transport rate IL1,IL1ra from B to P /hr
   tL_BT  volT/volB*tL_TB   # transport rate IL1,IL1ra from B to T /hr
   # cytokine elimination
   dL     1.4               # elimination rate IL1,IL1ra, /hr
end parameters
begin compartments
   B   3  volB          # blood
   eP  2  volP/10  B    # peritoneal endothelium
   P   3  volP     eP   # peritoneum
   eT  2  volT/10  B    # tissue endothelium
   T   3  volT     eT   # tissue
end compartments
begin molecule types
   Path(ph)                  # pathogen
   Mphage(p)                 # macrophage
   IL1(b)                    # ligand (IL1)
   IL1ra(b)                  # receptor antagonist (IL1ra)
   Neutro(p,e)               # neutrophil
   Endo(n,a~0~1~2,r,r,r)     # endothelial cell
end molecule types
begin seed species
   @P: Path(ph)              P0*volP
   @P: Mphage(p)             M0*volP
   @T: Mphage(p)             M0*volT
   @eP: Endo(n,a~0,r,r,r)    E0*volP
   @eT: Endo(n,a~0,r,r,r)    E0*volT
   @B: Neutro(p,e)           N0*volB
end seed species
begin observables
   Molecules Path_P          @P:Path()
   Molecules Neutr_P         @P:Neutro()
   Molecules Neutr_B         @B:Neutro()
   Molecules Neutr_T         @T:Neutro()
   Molecules IL1_P           @P:IL1(b)
   Molecules IL1_B           @B:IL1(b)
   Molecules IL1_T           @T:IL1(b)
   Molecules IL1ra_P         @P:IL1ra(b)
   Molecules IL1ra_B         @B:IL1ra(b)
   Molecules IL1ra_T         @T:IL1ra(b)
   Molecules Act_Endo_P      @eP:Endo(a~2)
   Molecules Total_Endo_P    @eP:Endo()
   Molecules Act_Endo_T      @eT:Endo(a~2)
   Molecules Total_Endo_T    @eT:Endo()
```

```
end observables
begin reaction rules
   # logistic pathogen growth
   Path(ph) -> Path(ph) + Path(ph)   sP_P
   Path(ph) + Path(ph) -> Path(ph)   dP_P
   # macrophage binding P, cytokine prod., and phagocytosis
   Path(ph) + Mphage(p) -> Path(ph!1).Mphage(p!1)                bMP
   Mphage(p!1).Path(ph!1) -> Mphage(p!1).Path(ph!1) + IL1(b)  sL_M
   Path(ph!1).Mphage(p!1) -> Mphage(p)                          dP_M
   # neutrophil binds P and phagocytoses
   Path(ph) + Neutro(p,e) -> Path(ph!1).Neutro(p!1,e)  NP
   Path(ph!1).Neutro(p!1) -> Neutro(p)                 dP_N
   # neutrophils secrete IL1ra after migration
   Neutro@P -> Neutro@P + IL1ra(b)@P   sA_N
   Neutro@T -> Neutro@T + IL1ra(b)@T   sA_N
   # source and decay of neutrophils
   0 -> Neutro(p,e)@B   sN_B
   Neutro -> 0          dN DeleteMolecules
   # binding of IL1,IL1ra to endothelial cells
   IL1(b)@P   + Endo(r) <-> IL1(b!1)@P.Endo(r!1)    bLE, uLE
   IL1ra(b)@P + Endo(r) <-> IL1ra(b!1)@P.Endo(r!1)  bLE, uLE
   IL1(b)@T   + Endo(r) <-> IL1(b!1)@T.Endo(r!1)    bLE, uLE
   IL1ra(b)@T + Endo(r) <-> IL1ra(b!1)@T.Endo(r!1)  bLE, uLE
   # activation/deactivation of endothelial cells
   Endo(a~0,r!1,r!2).IL1(b!1).IL1(b!2) \
    -> Endo(a~1,r!1,r!2).IL1(b!1).IL1(b!2)  aE_L
   Endo(a~1) -> Endo(a~2)  aE
   Endo(a~2) -> Endo(a~0)  dE
   # binding of neutrophils to activated endothelium
   Endo(n,a~2) + Neutro(e)@B -> Endo(n!1,a~2).Neutro(e!1)@B  bEN
   # migration of neutrophils across enothelium
   Endo(n!1)@eP.Neutro(e!1)@B -> Endo(n)@eP + Neutro(e)@P  tN_E
   Endo(n!1)@eT.Neutro(e!1)@B -> Endo(n)@eT + Neutro(e)@T  tN_E
   # leaking IL1,IL1ra to blood
   IL1(b)@P   <-> IL1(b)@B     tL_PB, tL_BP
   IL1(b)@T   <-> IL1(b)@B     tL_TB, tL_BT
   IL1ra(b)@P <-> IL1ra(b)@B   tL_PB, tL_BP
   IL1ra(b)@T <-> IL1ra(b)@B   tL_TB, tL_BT
   # elimination of IL1,IL1ra from blood
   IL1(b)@B   -> 0  dL
   IL1ra(b)@B -> 0  dL
end reaction rules
end model
```

H.2 SEPSIS MODEL 2

```
# Sepsis Model #2A, BNGL format
# Neutrophil recruitment is linear in tissue cytokine
#  with inhibition by blood cytokine
version("2.2.3")
begin model
begin parameters
```

```
   # volume params
   vP      15.6   # ml, interstitial volume, Peritoneal compartment
   vB      15.6   # ml, plasma volume, Blood compartment
   vL      15.6   # ml, interstitial volumne, Lung compartment
   # init conditions
   mT0     1e2    # 10^3/ml, resident macrophage in tissue
   nB0     4.5e3  # 10^3/ml, baseline circulating neutrophils
   CLP0    1      # a.u./ml, initial CLP condition
   # CLP healrate
   dclp    0.12   # /hr, CLP progression rate
   # pathogen params
   kpclp   4200   # 10^3/hr, source of pathogen from CLP
   kp      0.174  # /hr, pathogen growth parameter
   pmax    1e6    # 10^3/ml, maximum pathogen (logistic form)
   kpn     3      # /hr, max elimination rate of pathogen (per neutrophil)
   xpn     1e3    # 10^3/ml, conc. of pathogen resulting in half-max elim.
   # cytokine params
   kc      6      # /ml/hr, max production of cytokine
   xcp     1e3    # 10^3/ml, conc. of pathogen for half-max cytokine prod.
   xcn     1e3    # 10^3/ml, conc. of neutro. for half-max cytokine prod.
   dcT     0.36   # /hr, cytokine elimination rate in tissue
   dcB     1.0    # /hr, elimination rate in blood
   kcBT    0.36   # /hr, cytokine transport rate between blood to tissue
   # neutrophil params
   dn      0.06   # /hr, death rate of neutrophils
   knc     300    # 10^3/hr/ml, max neutrophil migration (/ml of tissue)
   xnc     1      # /ml, conc. of tissue cytokine for half-max neutr. migr.
   incB    1      # /ml, conc. of blood cytokine for
                  #   half-max inhibition of neutrophil migr.
end parameters
begin molecule types
   Path()
   Cyto()
   Mphage()
   Neutro()
   CLP(s~0~1)
end molecule types
begin compartments
   B  3  vB    # blood
   P  3  vP    # peritoneum
   L  3  vL    # lung
end compartments
begin seed species
   Path@P       0
   Cyto@P       0
   Cyto@B       0
   Cyto@L       0
   Mphage@P     mT0*vP
   Mphage@L     mT0*vL
   Neutro@P     0
   $Neutro@B    nB0*vB
   Neutro@L     0
   CLP@P(s~0)   CLP0*vP
end seed species
```

```
begin observables
   Molecules PathP    Path@P
   Molecules CytoP    Cyto@P
   Molecules CytoB    Cyto@B
   Molecules CytoL    Cyto@L
   Molecules CLP      CLP@P(s~1)
   Molecules NeutroP  Neutro@P
   Molecules NeutroB  Neutro@B
   Molecules NeutroL  Neutro@L
end observables
begin reaction rules
   # CLP "healing"
   CLP(s~0) -> CLP(s~1)  dclp
   CLP(s~1) -> 0         dclp
   # source of pathgoen
   CLP(s~1) -> CLP(s~1) + Path  kpclp
   # logistic growth
   Path -> Path + Path  kp*(1-(PathP/vP)/pmax)
   # phagocytosis by resident phagocytes
   Mphage + Path -> Mphage  kpn/(xpn + PathP/vP)
   # phagocytosis by recruited phagocytes
   Neutro + Path -> Neutro  kpn/(xpn + PathP/vP)
   # cytokine production (due to Pathogen presence or Neutrophil activity)
   Mphage@P -> Mphage@P + Cyto@P \
       kc/mT0*((PathP/vP/xcp)^2 + (NeutroP/vP/xcn)^2) \
       /(1 + (PathP/vP/xcp)^2 + (NeutroP/vP/xcn)^2)
   Mphage@L -> Mphage@L + Cyto@L \
       kc/mT0*(NeutroL/vL)^2/((xcn^2 + NeutroL/vL)^2)
   # cytokine "leak" into blood compartment
   Cyto@P <-> Cyto@B    kcBT, kcBT*vP/vB
   Cyto@L <-> Cyto@B    kcBT, kcBT*vL/vB
   # cytokine elimination
   Cyto@P -> 0  dcT
   Cyto@B -> 0  dcB
   Cyto@L -> 0  dcT
   # neutrophil recruitment
   Neutro@B -> Neutro@P \
       (knc*incB/xnc)*(CytoP/vP)/(incB + (CytoB/vB))*vP/(nB0*vB)
   Neutro@B -> Neutro@L \
       (knc*incB/xnc)*(CytoL/vL)/(incB + (CytoB/vB))*vL/(nB0*vB)
   # neutrophil elimination
   Neutro -> 0  dn
end reaction rules
end model
```

BIBLIOGRAPHY

[1] Hiroaki Kitano. Systems biology: A brief overview. *Science*, 295(5560):1662–1664, March 2002.

[2] Eric J. Deeds, Jean Krivine, Jerome Feret, Vincent Danos, and Walter Fontana. Combinatorial complexity and compositional drift in protein interaction networks. *PLoS ONE*, 7(3), March 2012.

[3] Robert C. Hsueh, Madhusudan Natarajan, Iain Fraser, Blake Pond, Jamie Liu, Susanne Mumby, Heping Han, Lily I. Jiang, Melvin I. Simon, Ronald Taussig, and Paul C. Sternweis. Deciphering signaling outcomes from a system of complex networks. *Sci. Signal.*, 2(71):ra22, 2009.

[4] Madhusudan Natarajan, Keng-Mean Lin, Robert C. Hsueh, Paul C. Sternweis, and Rama Ranganathan. A global analysis of cross-talk in a mammalian cellular signalling network. *Nat Cell Biol*, 8(6):571–580, June 2006.

[5] J. R. Karr, J. C. Sanghvi, D. N. Macklin, M. V. Gutschow, J. M. Jacobs, B. Bolival Jr., N. Assad-Garcia, J. I. Glass, and M. W. Covert. A whole-cell computational model predicts phenotype from genotype. *Cell*, 150:389–401, 2012.

[6] John J Tyson, Katherine C Chen, and Béla Novák. Sniffers, buzzers, toggles and blinkers: dynamics of regulatory and signaling pathways in the cell. *Current Opinion in Cell Biology*, 15(2):221–231, 2003.

[7] Uri Alon. Network motifs: theory and experimental approaches. *Nature Reviews Genetics*, 8(6):450–461, June 2007.

[8] U. Alon. *An Introduction to Systems Biology: Design Principles of Biological Circuits.* Chapman & Hall/CRC, New York, 2006.

[9] R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon. Network motifs: Simple building blocks of complex networks. *Science*, 298(5594):824–827, October 2002.

[10] Péter Érdi and János Tóth. *Mathematical Models of Chemical Reactions: Theory and Applications of Deteministic and Stochastic Models*. Manchester University Press, Manchester, UK, 1989.

[11] Katherine C. Chen, Attila Csikasz-Nagy, Bela Gyorffy, John Val, Béla Novák, and John J. Tyson. Kinetic analysis of a molecular model of the budding yeast cell cycle. *Molecular Biology of the Cell*, 11(1):369–391, January 2000.

[12] Katherine C. Chen, Laurence Calzone, Attila Csikasz-Nagy, Frederick R. Cross, Béla Novák, and John J. Tyson. Integrative analysis of cell cycle control in budding yeast. *Molecular Biology of the Cell*, 15(8):3841–3862, August 2004.

[13] John J Tyson and Bela Novak. Regulation of the eukaryotic cell cycle: Molecular antagonism, hysteresis, and irreversible transitions. *Journal of Theoretical Biology*, 210(2):249–263, 2001.

[14] Béla Novák and John J Tyson. A model for restriction point control of the mammalian cell cycle. *Journal of Theoretical Biology*, 230(4):563–579, 2004.

[15] John G. Albeck, John M. Burke, Bree B. Aldridge, Mingsheng Zhang, Douglas A. Lauffenburger, and Peter K. Sorger. Quantitative analysis of pathways controlling extrinsic apoptosis in single cells. *Molecular Cell*, 30(1):11–25, apr 2008.

[16] Hoda Eydgahi, William W. Chen, Jeremy L. Muhlich, Dennis Vitkup, John N. Tsitsiklis, and Peter K. Sorger. Properties of cell death models calibrated and compared using bayesian approaches. *Molecular Systems Biology*, 9(1), 2013.

[17] B N Kholodenko, O V Demin, G Moehren, and J B Hoek. Quantification of short term signaling by the epidermal growth factor receptor. *The Journal of biological chemistry*, 274(42):30169–30181, 1999.

[18] B. Schoeberl, C. Eichler-Jonsson, E. D. Gilles, and G. Mller. Computational modeling of the dynamics of the MAP kinase cascade activated by surface and internalized EGF receptors. *Nature biotechnology*, 20(4):370, 2002.

[19] Aki Fujioka, Kenta Terai, Reina E Itoh, Kazuhiro Aoki, Takeshi Nakamura, Shinya Kuroda, Eisuke Nishida, and Michiyuki Matsuda. Dynamics of the Ras/ERK MAPK cascade as monitored by fluorescent probes. *J. Biol. Chem.*, 281:8917–8926, 2006.

[20] Angela Reynolds, Jonathan Rubin, Gilles Clermont, Judy Day, Yoram Vodovotz, and G Bard Ermentrout. A reduced mathematical model of the acute inflammatory response: I. derivation of model and analysis of anti-inflammation. *Journal of Theoretical Biology*, 242(1), 2006.

[21] Judy Day, Jonathan Rubin, Yoram Vodovotz, Carson C Chow, Angela Reynolds, and Gilles Clermont. A reduced mathematical model of the acute inflammatory response

II. capturing scenarios of repeated endotoxin administration. *Journal of Theoretical Biology*, 242(1):237–256, 2006.

[22] Carson C Chow, Gilles Clermont, Rukmini Kumar, Claudio Lagoa, Zacharia Tawadrous, David Gallo, Binnie Betten, John Bartels, Gregory Constantine, Mitchell P Fink, Timothy R Billiar, and Yoram Vodovotz. The acute inflammatory response in diverse shock states. *Shock*, 24(1):74–84, 2005.

[23] Rukmini Kumar, Gilles Clermont, Yoram Vodovotz, and Carson C. Chow. The dynamics of acute inflammation. *Journal of Theoretical Biology*, 230(2):145–155, 2004.

[24] Matthias Ruth and Bruce Hannon. Law of mass action. In *Modeling Dynamic Biological Systems*, Modeling Dynamic Systems, pages 61–64. Springer New York, January 1997.

[25] D. T. Gillespie. A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. *J. Comput. Phys.*, 22:403–434, 1976.

[26] Ina Koch. Petri nets a mathematical formalism to analyze chemical reaction networks. *Molecular Informatics*, 29(12):838843, 2010.

[27] Daniel T. Gillespie. Exact stochastic simulation of coupled chemical reactions. *J. Phys. Chem.*, 81(25):2340–2361, 1977.

[28] D. T. Gillespie. Stochastic simulation of chemical kinetics. *Annu. Rev. Phys. Chem.*, 58:35–55, 2007.

[29] M. A. Gibson and J. Bruck. Efficient exact stochastic simulation of chemical systems with many species and many channels. *J. Phys. Chem. A*, 104:1876–1889, 2000.

[30] Alexander Slepoy, Aidan P. Thompson, and Steven J. Plimpton. A constant-time kinetic Monte Carlo algorithm for simulation of large biochemical reaction networks. *J. Chem. Phys.*, 128:205101, 2008.

[31] Y. Cao, H. Li, and L. Petzold. Efficient formulation of the stochastic simulation algorithm for chemically reacting systems. *J. Chem. Phys.*, 121:4059–4067, 2004.

[32] D. T. Gillespie. Approximate accelerated stochastic simulation of chemically reacting systems. *J. Chem. Phys.*, 115:1716–1733, 2001.

[33] D. T. Gillespie and L. R. Petzold. Improved leap-size selection for accelerated stochastic simulation. *J. Chem. Phys.*, 119:8229–8234, 2003.

[34] Yang Cao, D T Gillespie, and L R Petzold. Efficient step size selection for the tau-leaping simulation method. *J. Chem. Phys.*, 124:044109, 2006.

[35] Yang Cao, Daniel T. Gillespie, and Linda R. Petzold. The slow-scale stochastic simulation algorithm. *The Journal of Chemical Physics*, 122(1):014116–18, January 2005.

[36] Leonard A. Harris, Aaron M. Piccirilli, Emily R. Majusiak, and Paulette Clancy. Quantifying stochastic effects in biochemical reaction networks using partitioned leaping. *Phys. Rev. E*, 79:051906, 2009.

[37] Leonard A. Harris and Paulette Clancy. A "partitioned leaping" approach for multiscale modeling of chemical reaction dynamics. *J. Chem. Phys.*, 125:144107, 2006.

[38] Alan C Hindmarsh, Peter N Brown, Keith E Grant, Steven L Lee, Radu Serban, Dan E Shumaker, and Carol S Woodward. SUNDIALS: suite of nonlinear and differential/algebraic equation solvers. *ACM Trans. Math. Software*, 31:363–396, 2005.

[39] Bard Ermentrout. XPPAUT. In N. Le Novre, editor, *Computational Systems Neurobiology*, pages 519–531. Springer Netherlands, January 2012.

[40] T. Pawson and P. Nash. Assembly of cell regulatory systems through protein interaction domains. *Science*, 300(5618):445–452, 2003.

[41] Wendell A Lim. The modular logic of signaling proteins: building allosteric switches from simple binding domains. *Current Opinion in Structural Biology*, 12(1):61–68, 2002.

[42] Robert G. Smock and Lila M. Gierasch. Sending signals dynamically. *Science*, 324(5924):198–203, 2009.

[43] Drew Endy and Roger Brent. Modelling cellular behaviour. *Nature*, 409(6818):391–395, 2001.

[44] W. S. Hlavacek, J. R. Faeder, M. L. Blinov, A. S. Perelson, and B. Goldstein. The complexity of complexes in signal transduction. *Biotechnol. Bioeng.*, 84:783–794, 2003.

[45] B Goldstein and A S Perelson. Equilibrium theory for the clustering of bivalent cell surface receptors by trivalent ligands. Application to histamine release from basophils. *Biophys. J.*, 45:1109–1123, 1984.

[46] Byron Goldstein, James R. Faeder, William S. Hlavacek, Michael L. Blinov, Antonio Redondo, and Carla Wofsy. Modeling the early signaling events mediated by FcϵRI. *Mol. Immunol.*, 38:1213–1219, 2002.

[47] James R Faeder, William S Hlavacek, Ilona Reischl, Michael L Blinov, Henry Metzger, Antonio Redondo, Carla Wofsy, and Byron Goldstein. Investigation of early events in FcϵRI-mediated signaling using a detailed mathematical model. *J. Immunol.*, 170:3769–3781, 2003.

[48] Franck Toledo and Geoffrey M. Wahl. Regulating the p53 pathway: in vitro hypotheses, in vivo veritas. *Nature Reviews Cancer*, 6(12):909–923, 2006.

[49] Julien F Ollivier, Vahid Shahrezaei, and Peter S Swain. Scalable rule-based modelling of allosteric proteins and biochemical networks. *PLoS Computational Biology*, 6(11):e1000975, 2010.

[50] Dipak Barua, James R Faeder, and Jason M Haugh. A bipolar clamp mechanism for activation of Jak-family protein tyrosine kinases. *PLoS Comput. Biol.*, 5:e1000364, 2009.

[51] Pawe Kocieniewski, James R. Faeder, and Tomasz Lipniacki. The interplay of double phosphorylation and scaffolding in MAPK pathways. *Journal of Theoretical Biology*, 295(0):116–124, February 2012.

[52] M. L. Blinov, James R. Faeder, B. Goldstein, and W. S. Hlavacek. A network model of early events in epidermal growth factor receptor signaling that accounts for combinatorial complexity. *BioSyst.*, 83:136–151, 2006.

[53] M. W. Sneddon, J. R. Faeder, and T. Emonet. Efficient modeling, simulation and coarse-graining of biological complexity with NFsim. *Nat. Meth.*, 8:177–183, 2011.

[54] W. S. Hlavacek, J. R. Faeder, M. L. Blinov, R. G. Posner, M. Hucka, and W. Fontana. Rules for modeling signal-transduction systems. *Sci. STKE*, 2006:re6, 2006.

[55] J. Yang, M. I. Monine, J. R. Faeder, and W. S. Hlavacek. Kinetic Monte Carlo method for rule-based modeling of biochemical networks. *Phys. Rev. E*, 78:031910, 2008.

[56] M. W. Sneddon. *Overcoming complexity in systems biology modeling and simulation.* PhD thesis, Yale University, 2012.

[57] V. Danos, J. Feret, W. Fontana, and J. Krivine. Scalable simulation of cellular signalling networks. *Lect. Notes Comput. Sci.*, 4807:139–157, 2007.

[58] Joshua Colvin, Michael I Monine, James R Faeder, William S Hlavacek, Daniel D Von Hoff, and Richard G Posner. Simulation of large-scale rule-based models. *Bioinformatics*, 25:910–917, 2009.

[59] J. R. Faeder, M. L. Blinov, B. Goldstein, and W. S. Hlavacek. Rule-based modeling of biochemical networks. *Complexity*, 10:22–41, 2005.

[60] J. R. Faeder, M. L. Blinov, and W. S. Hlavacek. Rule-based modeling of biochemical systems with BioNetGen. In *Methods in Molecular Biology*, volume 500, pages 113–167. Humana Press, Clifton, N.J., 2009.

[61] J. A. P. Sekar and J. R. Faeder. Rule-based modeling of signal transduction: A primer. In *Methods In Molecular Biology*, volume 880, pages 139–218. Humana Press, Clifton, N.J., 2012.

[62] http://www.bionetgen.org.

[63] M.L. Blinov, O. Ruebenacker, and I.I. Moraru. Complexity and modularity of intracellular networks: a systematic approach for modelling and simulation. *IET Syst. Biol.*, 2:363–368, 2008.

[64] I.I. Moraru, J.C. Schaff, B. M. Slepchenko, M.L. Blinov, F. Morgan, A. Lakshminarayana, F. Gao, Y. Li, and L.M. Loew. Virtual cell modelling and simulation software environment. *IET Systems Biology*, 2(5):352–362, 2008.

[65] Boris M Slepchenko, James C Schaff, Ian Macara, and Leslie M Loew. Quantitative cell biology with the virtual cell. *Trends in Cell Biology*, 13(11):570–576, 2003.

[66] M. L. Blinov, J. Yang, J. R. Faeder, and W. S. Hlavacek. Graph theory for rule-based modeling of biochemical networks. *Lect. Notes Comput. Sci.*, 4230:89–106, 2006.

[67] V. Danos and C. Laneve. Formal molecular biology. *Theor. Comput. Sci.*, 325(1):69–110, 2004.

[68] V. Danos, J. Feret, W. Fontana, R. Harmer, and J. Krivine. Rule-based modelling of cellular signalling. *Lect. Notes Comput. Sci.*, 4703:17–41, 2007.

[69] Mieszko Lis, Maxim N Artyomov, Srinivas Devadas, and Arup K Chakraborty. Efficient stochastic simulation of reaction-diffusion processes via direct compilation. *Bioinformatics (Oxford, England)*, 25(17):2289–2291, 2009.

[70] A. G. Batzer, D. Rotin, J. M. Ureña, E. Y. Skolnik, and J. Schlessinger. Hierarchy of binding sites for Grb2 and Shc on the epidermal growth factor receptor. *Mol. Cell Biol.*, 14:5192–5201, 1994.

[71] Y. Okabayashi, Y. Kido, T. Okutani, Y. Sugimoto, K. Sakaguchi, and M. Kasuga. Tyrosines 1148 and 1173 of activated human epidermal growth factor receptors are binding sites of Shc in intact cells. *J. Biol. Chem.*, 269:18674–18678, 1994.

[72] N. W. Lemons, B. Hu, and W. S. Hlavacek. Hierarchical graphs for rule-based modeling of biochemical systems. *BMC Bioinformatics*, 12:45, 2011.

[73] http://www.nfsim.org.

[74] http://sbml.org.

[75] http://www.mathworks.com/products/matlab/.

[76] Jérôme Feret, Vincent Danos, Jean Krivine, Russ Harmer, and Walter Fontana. Internal coarse-graining of molecular systems. *Proc. Natl. Acad. Sci. U.S.A.*, 106:6453–6458, 2009.

[77] Fengkai Zhang, Bastian R. Angermann, and Martin Meier-Schellersheim. The simmune modeler visual interface for creating signaling networks based on bi-molecular interactions. *Bioinformatics*, 29(9):1229–1230, May 2013.

[78] Bastian R. Angermann, Frederick Klauschen, Alex D. Garcia, Thorsten Prustel, Fengkai Zhang, Ronald N. Germain, and Martin Meier-Schellersheim. Computational modeling of cellular signaling processes embedded into dynamic spatial contexts. *Nature Methods*, 9(3):283–289, March 2012.

[79] Laurence Calzone, Francois Fages, and Sylvain Soliman. BIOCHAM: an environment for modeling biological systems and formalizing experimental knowledge. *Bioinformatics*, 22:1805–1807, 2006.

[80] Carsten Maus, Stefan Rybacki, and Adelinde M. Uhrmacher. Rule-based multi-level modeling of cell biological systems. *BMC Systems Biology*, 5(1):166, October 2011.

[81] K. Gunasekaran, Buyong Ma, and Ruth Nussinov. Is allostery an intrinsic property of all dynamic proteins? *Proteins: Structure, Function, and Bioinformatics*, 57(3):433443, 2004.

[82] Jacque Monod, Jeffries Wyman, and Jean-Pierre Changeux. On the nature of allosteric transitions: A plausible model. *Journal of Molecular Biology*, 12(1):88–118, 1965.

[83] D. E. Koshland, G. Nmethy, and D. Filmer. Comparison of experimental binding data and theoretical models in proteins containing subunits*. *Biochemistry*, 5(1):365–385, 1966.

[84] Judith Hbezfeld and H.Eugene Stanley. A general approach to co-operativity and its application to the oxygen equilibrium of hemoglobin and its effectors. *Journal of Molecular Biology*, 82(2):231–265, 1974.

[85] Eric R. Henry, Stefano Bettati, James Hofrichter, and William A. Eaton. A tertiary two-state allosteric model for hemoglobin. *Biophysical Chemistry*, 98(12):149–164, 2002.

[86] Brian F. Volkman, Doron Lipson, David E. Wemmer, and Dorothee Kern. Two-state allosteric behavior in a single-domain signaling protein. *Science*, 291(5512):2429–2433, 2001.

[87] Sho Asakura and Hajime Honda. Two-state model for bacterial chemoreceptor proteins: The role of multiple methylation. *Journal of Molecular Biology*, 176(3):349–367, 1984.

[88] Jennifer L. Macdonald and Linda J. Pike. Heterogeneity in EGF-binding affinities arises from negative cooperativity in an aggregating system. *Proceedings of the National Academy of Sciences*, 105(1):112–117, 2008.

[89] Kristen L. Pierce, Richard T. Premont, and Robert J. Lefkowitz. Seven-transmembrane receptors. *Nature Reviews Molecular Cell Biology*, 3(9):639–650, 2002.

[90] T Gudermann, F Kalkbrenner, and G Schultz. Diversity and selectivity of receptor-g protein interaction. *Annual Review of Pharmacology and Toxicology*, 36(1):429–459, 1996.

[91] Jyrki P. Kukkonen, Johnny Nsman, and Karl E.O. kerman. Modelling of promiscuous receptorGi/Gs-protein coupling and effector response. *Trends in Pharmacological Sciences*, 22(12):616–622, 2001.

[92] Giorgio Bacci, Vincent Danos, and Ohad Kammar. On the statistical thermodynamics of reversible communicating processes. In Andrea Corradini, Bartek Klin, and Corina Crstea, editors, *Algebra and Coalgebra in Computer Science*, number 6859 in Lecture Notes in Computer Science, pages 1–18. Springer Berlin Heidelberg, 2011.

[93] P Swain, A Weisse, J Ollivier, O Oury, V Danos, F Boyde, E Deeds, and R Harmer. Energy as syntax. Lecture, March 2011. http://homepages.inf.ed.ac.uk/vdanos/pdfs/eas.pdf [Accessed 13 June 2013].

[94] Daniel M. Zuckerman. *Statistical Physics of Biomolecules: An Introduction*. CRC Press, Boca Raton, Fl, 2010.

[95] Daniel A. Beard and Hong Qian. *Chemical Biophysics: Quantitative Analysis of Cellular Systems*. Cambridge University Press, Cambridge, 2008.

[96] David Colquhoun, Kathryn A. Dowsland, Marco Beato, and Andrew J.R. Plested. How to impose microscopic reversibility in complex reaction mechanisms. *Biophysical Journal*, 86(6):3510–3518, June 2004.

[97] Jin Yang, William J. Bruno, William S. Hlavacek, and John E. Pearson. On imposing detailed balance in complex reaction mechanisms. *Biophysical Journal*, 91(3):1136–1141, 2006.

[98] Gbor Szederknyi and Katalin M. Hangos. Finding complex balanced and detailed balanced realizations of chemical reaction networks. *Journal of Mathematical Chemistry*, 49(6):1163–1179, 2011.

[99] Vincent Danos and Nicolas Oury. Equilibrium and termination II: the case of petri nets. *submitted*. http://www.pps.univ-paris-diderot.fr/~danos/pdf/et2.pdf.

[100] J. E. Leffler. Parameters for the description of transition states. *Science*, 117(3039):340–341, 1953.

[101] W. A. Eaton, E. R. Henry, and J. Hofrichter. Application of linear free energy relations to protein conformational changes: the quaternary structural change of hemoglobin. *Proceedings of the National Academy of Sciences*, 88(10):4472–4475, 1991.

[102] Robert E. Kass and Adrian E. Raftery. Bayes factors. *Journal of the American Statistical Association*, 90(430):773–795, 1995.

[103] Andrew Gelman and Xiao-Li Meng. Simulating normalizing constants: from importance sampling to bridge sampling to path sampling. *Statistical Science*, 13(2):163–185, 1998.

[104] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B. Methodological*, 58(1):267–288, 1996.

[105] Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21(6):1087–1092, jun 1953.

[106] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, May 1983.

[107] V. ern. Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm. *Journal of Optimization Theory and Applications*, 45(1):41–51, January 1985.

[108] David J Earl and Michael W Deem. Parallel tempering: theory, applications, and new perspectives. *Physical chemistry chemical physics: PCCP*, 7(23):3910–3916, dec 2005.

[109] Robert H. Swendsen and Jian-Sheng Wang. Replica monte carlo simulation of spin-glasses. *Physical Review Letters*, 57(21):2607–2609, nov 1986.

[110] Gerd Gigerenzer and Henry Brighton. Homo heuristicus: Why biased minds make better inferences. *Topics in Cognitive Science*, 1(1):107143, 2009.

[111] L. A. Harris, J. S. Hogg, and J. R. Faeder. Compartmental rule-based modeling of biochemical systems. In M. D. Rossetti, R. R. Hill, B. Johansson, A. Dunkin, and R. G. Ingalls, editors, *Proceedings of the 2009 Winter Simulation Conference*, pages 908–919, 2009.

[112] B. Alberts, A. Johnson, J. Lewis, M. Raff, K. Roberts, and P. Walter. *Molecular Biology of the Cell.* Garland Science, 4th edition, 2002.

[113] J. M. Haugh and D. A. Lauffenburger. Physical modulation of intracellular signaling processes by locational regulation. *Biophys. J.*, 72:2014–2031, 1997.

[114] B. N. Kholodenko, J. B. Hoek, and H. V. Westerhoff. Why cytoplasmic signalling proteins should be recruited to cell membranes. *Trends Cell Biol.*, 10:173–178, 2000.

[115] Martin Meier-Schellersheim, Xuehua Xu, Bastian Angermann, Eric J. Kunkel, Tian Jin, and Ronald N. Germain. Key role of local regulation in chemosensing revealed by a new molecular interaction-based modeling method. *PLoS Comput Biol*, 2(7):710–724, 2006.

[116] Lorenzo Dematte, Corrado Priami, and Alessandro Romanel. The Beta Workbench: A computational tool to study the dynamics of biological systems. *Brief. Bioinform.*, 9:437–449, 2008.

[117] Aneil Mallavarapu, Matthew Thomson, Benjamin Ullian, and Jeremy Gunawardena. Programming with models: Modularity and abstraction provide powerful capabilities for systems biology. *J. R. Soc. Interface*, 6:257–270, 2009.

[118] M. Hucka, S. Hoops, S. M. Keating, N. Le Novère, S. Sahle, and D. Wilkinson. Systems Biology Markup Language (SBML) Level 2: Structures and facilities for model definitions. *Nature Precedings*, 2008. http://dx.doi.org/10.1038/npre.2008.2715.1.

[119] Mieszko Lis, Maxim N Artyomov, Srinivas Devadas, and Arup K Chakraborty. Efficient stochastic simulation of reaction-diffusion processes via direct compilation. *Bioinformatics (Oxford, England)*, 25(17):2289–2291, 2009.

[120] Sean Sedwards and Tommaso Mazza. Cyto-Sim: a formal language model and stochastic simulator of membrane-enclosed biochemical processes. *Bioinformatics*, 23:2800–2802, 2007.

[121] L. Cardelli. Brane calculi. In *Computational Methods in Systems Biology*, pages 257–278. Springer Berlin Heidelberg, 2005.

[122] L. Cardelli. Bitonal membrane systems. *Theor. Comput. Sci.*, 404:5–18, 2008.

[123] Aviv Regev, Ekaterina M. Panina, William Silverman, Luca Cardelli, and Ehud Shapiro. BioAmbients: an abstraction for biological compartments. *Theor. Comput. Sci.*, 325:141–167, 2004.

[124] C. Laneve and F. Tarissan. A simple calculus for proteins and cells. *Theor. Comput. Sci.*, 404:127–141, 2008.

[125] T. C. Damgaard, V. Danos, and J. Krivine. A language for the cell. Technical Report TR-2008-116, IT University of Copenhagen, Rued Langgaards Vej 7, DK-2300 Copenhagen V, 2008.

[126] B. B. Aldridge, J. M. Burke, D. A. Lauffenburger, and P. K. Sorger. Physicochemical modelling of cell signalling pathways. *Nat. Cell. Biol.*, 8:1195–1203, 2006.

[127] M. L. Blinov, J. R. Faeder, B. Goldstein, and W. S. Hlavacek. BioNetGen: software for rule-based modeling of signal transduction based on the interactions of molecular domains. *Bioinformatics*, 17:3289–3291, 2004.

[128] M Tomita. Whole-cell simulation: a grand challenge of the 21st century. *Trends Biotechnol.*, 19:205–210, 2001.

[129] Uri Moran, Rob Phillips, and Ron Milo. Snapshot: Key numbers in biology. *Cell*, 141:1262–1262.e1, 2010.

[130] M. I. Monine, R. G. Posner, P. B. Savage, J. R. Faeder, and W. S. Hlavacek. Modeling multivalent ligand-receptor interactions with steric constraints on configurations of cell-surface receptor aggregates. *Biophys. J.*, 98:48–56, 2010.

[131] Jeremy Roland, Julien Berro, Alphée Michelot, Laurent Blanchoin, and Jean-Louis Martiel. Stochastic severing of actin filaments by actin depolymerizing factor/cofilin controls the emergence of a steady dynamical regime. *Biophys. J.*, 94:2082–2094, 2008.

[132] Byron Goldstein, James R. Faeder, and William S. Hlavacek. Mathematical and computational models of immune-receptor signalling. *Nat. Rev. Immunol.*, 4:445–456, 2004.

[133] Edward C Stites, Paul C Trampont, Zhong Ma, and Kodi S Ravichandran. Network analysis of oncogenic Ras activation in cancer. *Science*, 318:463–467, 2007.

[134] Thomas D Pollard and Gary G Borisy. Cellular motility driven by assembly and disassembly of actin filaments. *Cell*, 112:453–465, 2003.

[135] Catherine I Lacayo, Zachary Pincus, Martijn M VanDuijn, Cyrus A Wilson, Daniel A Fletcher, Frank B Gertler, Alex Mogilner, and Julie A Theriot. Emergence of large-scale cell morphology and movement from local actin filament growth dynamics. *PLOS Biol.*, 5:e233, 2007.

[136] Kelly D. Stone, Calman Prussin, and Dean D. Metcalfe. IgE, mast cells, basophils, and eosinophils. *J. Allergy Clin. Immun.*, 125:S73–S80, 2010.

[137] A. F. Overbeck, T. R. Brtva, A. D. Cox, S. M. Graham, S. Y. Huff, R. Khosravi-Far, L. A. Quilliam, P. A. Solski, and Der C. J. Guanine nucleotide exchange factors: activators of Ras superfamily proteins. *Mol. Reprod. Dev.*, 42:468–476, 1995.

[138] Frank Wilcoxon. Individual comparisons by ranking methods. *Biometrics Bull.*, 1:80–83, 1945.

[139] H. B. Mann. On a test of whether one of two random variables is stochastically larger than the other. *Ann. Math. Stat.*, 18:50–60, 1947.

[140] Karl Pearson. On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling. *Philos. Mag.*, 50:157–175, 1900.

[141] http://aws.amazon.com/ec2/ [Accessed Jan 2013].

[142] Darrell Whitley. A genetic algorithm tutorial. *Stat. Comput.*, 4:65–85, 1994.

[143] F. Campolongo, J. Cariboni, and A. Saltelli. An effective screening design for sensitivity analysis of large models. *Environ. Modell. Softw.*, 22:1509–1518, 2007.

[144] E. M. Clarke, J. R. Faeder, C. L. Langmead, L. A. Harris, S. K. Jha, and A. Legay. Statistical model checking in *BioLab*: Applications to the automated analysis of T-cell receptor signaling pathway. *Lect. Notes Comput. Sci.*, 5307:231–250, 2008.

[145] J. Elf and M. Ehrenberg. Spontaneous separation of bi-stable biochemical systems into spatial domains of opposite phases. *Syst. Biol.*, 1:230–236, 2004.

[146] D. Bernstein. Simulating mesoscopic reaction-diffusion systems using the Gillespie algorithm. *Phys. Rev. E*, 71:041103, 2005.

[147] T. Tian and K. Burrage. Binomial leap methods for simulating stochastic chemical kinetics. *J. Chem. Phys.*, 121:10356–10364, 2004.

[148] A. Chatterjee, D. G. Vlachos, and M. A. Katsoulakis. Binomial distribution based τ-leap accelerated stochastic simulation. *J. Chem. Phys.*, 122:024112, 2005.

[149] A. Auer, P. Chatelain, and P. Koumoutsakos. R-leaping: Accelerating the stochastic simulation algorithm by reaction leaps. *J. Chem. Phys.*, 125:084103, 2006.

[150] X. Cai and Z. Xu. K-leap method for accelerating stochastic simulation of coupled chemical reactions. *J. Chem. Phys.*, 126:074102, 2007.

[151] D. F. Anderson. Incorporating postleap checks in tau-leaping. *J. Chem. Phys.*, 128:054103, 2008.

[152] John A Kellum, Mingchen Song, and Ramesh Venkataraman. Hemoadsorption removes tumor necrosis factor, interleukin-6, and interleukin-10, reduces nuclear factor-kappaB DNA binding, and improves short-term survival in lethal endotoxemia. *Critical Care Medicine*, 32(3):801–805, 2004.

[153] Zhi-Yong Peng, Melinda J Carter, and John A Kellum. Effects of hemoadsorption on cytokine removal and short-term survival in septic rats. *Critical Care Medicine*, 36(5):1573–1577, 2008.

[154] C. A Janeway, P. Travers, M. Walport, and M. J Shlomchik. *Immunobiology: the immune system in health and disease.* Taylor & France, 6th edition, 2004.

[155] J. Cohen. The immunopathogenesis of sepsis. *Nature*, 420(6917):885891, 2002.

[156] Jordan S. Pober and William C. Sessa. Evolving functions of endothelial cells in inflammation. *Nat Rev Immunol*, 7(10):803–815, 2007.

[157] Carl Nathan. Neutrophils and immunity: challenges and opportunities. *Nat Rev Immunol*, 6(3):173–182, 2006.

[158] Philip O. Scumpia and Lyle L. Moldawer. Biology of interleukin-10 and its regulatory roles in sepsis syndromes. *Critical Care Medicine*, 33(Suppl):S468–S471, 2005.

[159] R C Bone, R A Balk, F B Cerra, R P Dellinger, A M Fein, W A Knaus, R M Schein, and W J Sibbald. Definitions for sepsis and organ failure and guidelines for the use of innovative therapies in sepsis. *Chest*, 101(6):1644–1655, 1992.

[160] D C Angus, W T Linde-Zwirble, J Lidicker, G Clermont, J Carcillo, and M R Pinsky. Epidemiology of severe sepsis in the united states: analysis of incidence, outcome, and associated costs of care. *Critical Care Medicine*, 29(7):1303–1310, 2001.

[161] Suveer Singh and Timothy Evans. Organ dysfunction during sepsis. *Intensive Care Medicine*, 32(3):349–360, mar 2006.

[162] R. Dellinger, Mitchell Levy, Jean Carlet, Julian Bion, Margaret Parker, Roman Jaeschke, Konrad Reinhart, Derek Angus, Christian Brun-Buisson, Richard Beale, Thierry Calandra, Jean-Francois Dhainaut, Herwig Gerlach, Maurene Harvey, John Marini, John Marshall, Marco Ranieri, Graham Ramsay, Jonathan Sevransky, B. Thompson, Sean Townsend, Jeffrey Vender, Janice Zimmerman, and Jean-Louis Vincent. Surviving sepsis campaign: International guidelines for management of severe sepsis and septic shock: 2008. *Intensive Care Medicine*, 34(1):17–60, 2008.

[163] Jean-Louis Vincent, Qinghua Sun, and Marc-Jacques Dubois. Clinical trials of immunomodulatory therapies in severe sepsis and septic shock. *Clinical Infectious Diseases*, 34(8):1084–1093, 2002.

[164] M.P. Fink. Animal models of sepsis and its complications. *Kidney Int.*, 74(8):991–993, oct 2008.

[165] William J Hubbard, Mashkoor Choudhry, Martin G Schwacha, Jeffrey D Kerby, 3rd Rue, Loring W, Kirby I Bland, and Irshad H Chaudry. Cecal ligation and puncture. *Shock (Augusta, Ga.)*, 24 Suppl 1:52–57, December 2005.

[166] Daniel Rittirsch, L. Marco Hoesel, and Peter A. Ward. The disconnect between animal models of sepsis and human sepsis. *Journal of Leukocyte Biology*, 81(1):137–143, January 2007.

[167] Morgan DiLeo, John Kellum, and William Federspiel. A simple mathematical model of cytokine capture using a hemoadsorption device. *Annals of Biomedical Engineering*, 37(1):222–229, 2009.

[168] Morgan V. DiLeo, James D. Fisher, and William J. Federspiel. Experimental validation of a theoretical model of cytokine capture using a hemoadsorption device. *Annals of Biomedical Engineering*, 37(11):2310–2316, 2009.

[169] Thomas Rimmel, Ata Murat Kaynar, Joseph N. McLaughlin, Jeffery V. Bishop, Morgan V. Fedorchak, Anan Chuasuwan, Zhiyong Peng, Kai Singbartl, Daniel R. Frederick, Lin Zhu, Melinda Carter, William J. Federspiel, Adriana Zeevi, and John A. Kellum. Leukocyte capture and modulation of cell-mediated immunity during human sepsis: an ex vivo study. *Critical Care*, 17(2):1–13, April 2013.

[170] Silvia Daun, Jonathan Rubin, Yoram Vodovotz, Anirban Roy, Robert Parker, and Gilles Clermont. An ensemble of models of the acute inflammatory response to bacterial lipopolysaccharide in rats: Results from parameter space reduction. *Journal of Theoretical Biology*, 253(4):843–853, 2008.

[171] Angela Reynolds. *Mathematical Models of Acute Inflammation and a Full Lung Model of Gas Exchange Under Inflammatory Stress*. PhD thesis, University of Pittsburgh, 2008.

[172] Gary An. In silico experiments of existing and hypothetical cytokine-directed clinical trials using agent-based modeling. *Critical Care Medicine*, 32(10):2050–2060, 2004.

[173] Qi Mi, Beatrice Rivire, Gilles Clermont, David L. Steed, and Yoram Vodovotz. Agent-based model of inflammation and wound healing. *Wound Repair and Regeneration*, 15(5):671–682, 2007.

[174] Denise E Kirschner, Stewart T Chang, Thomas W Riggs, Nicolas Perry, and Jennifer J Linderman. Toward a multiscale model of antigen presentation in immunity. *Immunological Reviews*, 216:93–118, 2007.

[175] Juilee Thakar, Mylisa Pilione, Girish Kirimanjeswara, Eric T Harvill, and Rka Albert. Modeling systems-level regulation of host immune responses. *PLoS Comput Biol*, 3(6):e109, 2007.

[176] L Glass and S A Kauffman. The logical analysis of continuous, non-linear biochemical control networks. *Journal of Theoretical Biology*, 39(1):103–129, 1973.

[177] Jos C Alves-Filho, Andressa de Freitas, Momtchilo Russo, and Fernando Q Cunha. Toll-like receptor 4 signaling leads to neutrophil migration impairment in polymicrobial sepsis. *Critical Care Medicine*, 34(2):461–470, 2006.

[178] RP Schleimer and BK Rutledge. Cultured human vascular endothelial cells acquire adhesiveness for neutrophils after stimulation with interleukin 1, endotoxin, and tumor-promoting phorbol diesters. *J Immunol*, 136(2):649–654, January 1986.

[179] M P Bevilacqua, J S Pober, M E Wheeler, R S Cotran, and M A Gimbrone. Interleukin 1 acts on cultured human vascular endothelium to increase the adhesion of polymorphonuclear leukocytes, monocytes, and related leukocyte cell lines. *The Journal of Clinical Investigation*, 76(5):2003–2011, November 1985.

[180] D. J. Dripps, E. Verderber, R. K. Ng, R. C. Thompson, and S. P. Eisenberg. Interleukin-1 receptor antagonist binds to the type II interleukin-1 receptor on b cells and neutrophils. *Journal of Biological Chemistry*, 266(30):20311–20315, October 1991.

[181] William P. Arend, Mark Malyak, Carla J. Guthridge, and Cem Gabay. Interleukin-1 receptor antagonist: role in biology, 1998.

[182] Patrick E. Sharp and Marie C. La Regina. *The Laboratory Rat*. CRC Press, Boca Raton, FL, 2002.

[183] B. L. Ferraiolo, J. McCabe, S. Hollenbach, B. Hultgren, R. Pitti, and H. Wilking. Pharmacokinetics of recombinant human tumor necrosis factor-alpha in rats. effects

of size and number of doses and nephrectomy. *Drug Metabolism and Disposition*, 17(4):369–372, July 1989.

[184] Chrisovalantou Cheretakis, Roland Leung, Chun Xiang Sun, Yigal Dror, and Michael Glogauer. Timing of neutrophil tissue repopulation predicts restoration of innate immune protection in a murine bone marrow transplantation model. *Blood*, 108(8):2821–2826, October 2006.

[185] Douglas Ruben Call, Jean Ann Nemzek, Samuel John Ebong, Gerald Lee Bolgos, David Eric Newcomb, and Daniel George Remick. Ratio of local to systemic chemokine concentrations regulates neutrophil recruitment. *The American Journal of Pathology*, 158(2):715–721, February 2001.

[186] Zhiyong Peng, Jeffery Bishop, Xiaoyan Wen, Feihu Zhou, Ata Murat Kaynar, and John Kellum. Hemoadsorption modulates leukocyte trafficking in different compartments during sepsis. *Critical Care Medicine*, 39(12):3, 2011.

[187] Zhi-Yong Peng, Hong-Zhi Wang, Melinda J. Carter, Morgan V. Dileo, Jeffery V. Bishop, Fei-Hu Zhou, Xiao-Yan Wen, Thomas Rimmel, Kai Singbartl, William J. Federspiel, Gilles Clermont, and John A. Kellum. Acute removal of common sepsis mediators does not explain the effects of extracorporeal blood purification in experimental sepsis. *Kidney International*, 81(4):363–369, February 2012.

[188] Samuel Ebong, Douglas Call, Jean Nemzek, Gerald Bolgos, David Newcomb, and Daniel Remick. Immunopathologic alterations in murine models of sepsis of increasing severity. *Infection and Immunity*, 67(12):6603–6610, December 1999. PMID: 10569781.

[189] David J. Klinke. An empirical bayesian approach for model-based inference of cellular signaling networks. *BMC Bioinformatics*, 10(1):371, 2009.

[190] Tian-Rui Xu, Vladislav Vyshemirsky, Amlie Gormand, Alex von Kriegsheim, Mark Girolami, George S Baillie, Dominic Ketley, Allan J Dunlop, Graeme Milligan, Miles D Houslay, and Walter Kolch. Inferring signaling pathway topologies from multiple perturbation measurements of specific biochemical species. *Science signaling*, 3(134):ra20, 2010.

[191] H. B. Lee and M. D. Blaufox. Blood volume in the rat. *J Nucl Med*, 26(1):72–76, January 1985.

[192] Shannon Copeland, H. Shaw Warren, Stephen F. Lowry, Steve E. Calvano, and Daniel Remick. Acute inflammatory response to endotoxin in mice and humans. *Clinical and Diagnostic Laboratory Immunology*, 12(1):60–67, January 2005.

[193] Yongmei Li, Arthur Karlin, John D. Loike, and Samuel C. Silverstein. A critical concentration of neutrophils is required for effective bacterial killing in suspension. *Proceedings of the National Academy of Sciences*, 99(12):8289–8294, June 2002.

[194] D G Garlick and E M Renkin. Transport of large molecules from plasma to interstitial fluid and lymph in dogs. *The American journal of physiology*, 219(6):1595–1605, December 1970.

[195] Yoram Vodovotz, Gregory Constantine, Jonathan Rubin, Marie Csete, Eberhard O. Voit, and Gary An. Mechanistic simulations of inflammation: Current state and future prospects. *Mathematical Biosciences*, 217(1):1–10, January 2009.

[196] W. K. Hastings. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57(1):97–109, apr 1970.

[197] Christian P Robert and George Casella. *Monte Carlo Statistical Methods*. Springer, New York, NY, 2nd edition, 2010.

[198] W R Gilks, S Richardson, and D J Spiegelhalter. *Markov Chain Monte Carlo in Practice*. Chapman & Hall/CRC, Boca Raton, FL, 1996.

[199] Jose C. Alves-Filho, Andressa Freitas, Fabricio O. Souto, Fernando Spiller, Heitor Paula-Neto, Joao S. Silva, Ricardo T. Gazzinelli, Mauro M. Teixeira, Sergio H. Ferreira, and Fernando Q. Cunha. Regulation of chemokine receptor by toll-like receptor 2 is critical to neutrophil migration and resistance to polymicrobial sepsis. *Proceedings of the National Academy of Sciences*, 106(10):4018–4023, March 2009.

[200] V. Danos, J. Feret, W. Fontana, R. Harmer, J. Hayman, J. Krivine, C. Thompson-Walsh, and G. Winskel. Graphs, rewriting and causality in rule-based models. *submitted*. http://www.cl.cam.ac.uk/~jmh93/papers/biomod.pdf.

[201] H. Ehrig, R. Heckel, M. Korff, M. Löwe, L. Ribeiro, A. Wagner, and A. Corradini. Algebraic approaches to graph transformation part II: Single pushout approach and comparison with double pushout approach. In *Handbook of Graph Grammars and Computing by Graph Transformation*, volume 1, pages 247–312. World Scientific, Singapore, 1996.