Efficient modeling, simulation and coarse-graining of biological complexity with NFsim

Michael W Sneddon[1,2], James R Faeder[3] & Thierry Emonet[1,2,4]

Managing the overwhelming numbers of molecular states and interactions is a fundamental obstacle to building predictive models of biological systems. Here we introduce the Network-Free Stochastic Simulator (NFsim), a general-purpose modeling platform that overcomes the combinatorial nature of molecular interactions. Unlike standard simulators that represent molecular species as variables in equations, NFsim uses a biologically intuitive representation: objects with binding and modification sites acted on by reaction rules. During simulations, rules operate directly on molecular objects to produce exact stochastic results with performance that scales independently of the reaction network size. Reaction rates can be defined as arbitrary functions of molecular states to provide powerful coarse-graining capabilities, for example to merge Boolean and kinetic representations of biological networks. NFsim enables researchers to simulate many biological systems that were previously inaccessible to general-purpose software, as we illustrate with models of immune system signaling, microbial signaling, cytoskeletal assembly and oscillating gene expression.

A cornerstone of scientific discovery is the hypothesis-driven experiment. However, in recent years high-throughput experimental methods have led to a marked increase in the scale and complexity of the biological questions that have been investigated, making it difficult to formulate hypotheses[1] and to anticipate results before committing resources to experiments. Computational methods offer the promise of transforming quantitative experimental data into predictive and testable models of cellular behavior[2,3], but the complexities of biological systems[4–7] are exposing the limitations of standard modeling techniques. Here we address two major limitations that stand in the way of widespread adoption of modeling and simulation tools by biologists: the inability to deal with the combinatorial complexity of molecular interactions, and the limited flexibility of simulation tools for coarse-graining biochemical processes.

Combinatorial complexity arises frequently in biochemical pathways because molecules tend to interact in a multitude of distinct combinations[8]. Consider a signaling protein that can be competitively phosphorylated at multiple sites[6]. Each site can be in one of four states: phosphorylated, unphosphorylated, bound to a phosphatase or bound to a kinase. Accounting for the combination of possible states generates ~4^n unique chemical species and reactions, where n is the number of sites (**Fig. 1a**,**b**). Standard simulation methods have computational and memory requirements that scale with network size (**Fig. 1c**) and thereby impose an inherent limit on the complexity of systems that can be handled[8]. Even a powerful computer running software[9,10] designed to handle large reaction networks could not simulate the multisite phosphorylation model when the number of sites exceeded eight (**Fig. 1c**).

Solving the combinatorial challenge is imperative, because it emerges throughout biology, for example, whenever details of signaling complexes, polymerization or post-translational modifications are considered. Combinatorial complexity arises even in relatively simple systems where most molecular interactions and reaction rates have been characterized. As a result, this form of complexity is among the main limiting factors in modeling systems as diverse as the MAP kinase cascade[4], cytoskeletal assembly[5], bacterial chemotaxis[7] and amyloid fibril growth[11]. In all these cases, modelers are forced either to make simplifications to conform to standard modeling conventions or to build their own problem-specific software that is difficult to validate, reuse and extend[4,6,11–13].

A second technical hurdle lies in simulating biochemical mechanisms in an approximate or coarse-grained manner. Coarse-graining is necessary in situations where gaps exist in our mechanistic knowledge and is useful when certain reactions occur rapidly or are inconsequential to system output[14]. Coarse-graining strategies are also central to efforts to bridge the divide between '-omics'-level networks and predictive models of dynamic cellular behavior[15]. A limited number of simulators currently support basic approximations in the form of mathematical expressions, such as the Michaelis-Menten function for enzymatic activity[16,17]. However, no general simulator can handle more complicated approximations such as those needed to model long-range cooperativity in large signaling complexes[12,18] or to simulate logical interaction networks simultaneously with detailed reaction dynamics.

Rule-based modeling offers a promising approach to combinatorial complexity and coarse-grained simulation[8,9,19]

[1]Department of Molecular, Cellular and Developmental Biology, Yale University, New Haven, Connecticut, USA. [2]Interdepartmental Program in Computational Biology and Bioinformatics, Yale University, New Haven, Connecticut, USA. [3]Department of Computational and Systems Biology, University of Pittsburgh School of Medicine, Pittsburgh, Pennsylvania, USA. [4]Department of Physics, Yale University, New Haven, Connecticut, USA. Correspondence should be addressed to T.E. (thierry.emonet@yale.edu).

RECEIVED 8 SEPTEMBER; ACCEPTED 3 DECEMBER; PUBLISHED ONLINE 26 DECEMBER 2010; DOI:10.1038/NMETH.1546

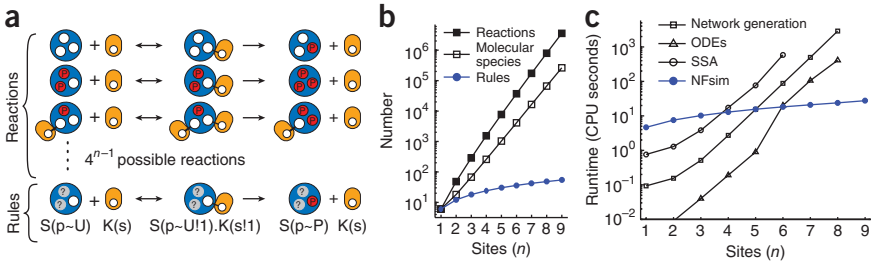


Figure 1 | Combinatorial complexity in multisite phosphorylation and NFsim performance scaling. (**a**) A substrate protein (S, blue) can be phosphorylated (P, red) at multiple independent sites by a kinase (K, yellow). A single rule, indicating that a site named 'p' on 'S' must be in the unphosphorylated state 'U' for the event to occur, can represent 4^{n-1} different reactions, where n is the number of phosphorylation sites on the substrate protein. (**b**) The number of reactions and chemical species that need to be accounted for grows exponentially with n. Rules and parameters grow only linearly with n. (**c**) The runtime performance of NFsim (blue filled circles) for this model is compared to the runtime of optimized ODE (triangles) and SSA (open circles) simulators[9,10]. Network generation (squares) depicts the computational cost of transforming a rule-based model into a set of reactions that can be simulated by ODE or SSA simulators.

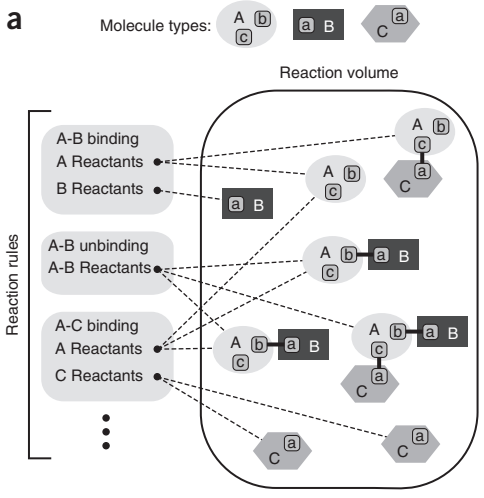
(**Supplementary Notes 1** and **2**). Instead of listing all chemical species that could potentially exist in a system, rule-based models identify only the types of molecules that exist, and then use rules to specify when and how chemical reactions occur (**Fig. 1a**). Whereas the numbers of reactions and chemical species grow exponentially with additional molecular details, the number of rules and parameters grow linearly (**Fig. 1b**). Rules can be used to generate the full network of potential reactions[9,19,20], which can then be simulated with ordinary differential equations (ODEs) or stochastic methods. However, a set of rules often gives rise to an intractable number of distinct reaction combinations, precluding the use of standard methods to simulate them. So although rule-based modeling languages can represent many systems, the simulation of rule-based models remains challenging.

Here we present an open-source modeling platform called NFsim that efficiently simulates rule-based models and permits flexible coarse-graining of reaction mechanisms using arbitrary mathematical or conditional functions. NFsim generalizes an agent-based kinetic Monte Carlo method that has been shown to circumvent the combinatorial bottleneck in simulations[21,22]. By putting substantial effort into optimizing the algorithmic data structures and operations of this method, we achieved a speed increase of at least 1–2 orders of magnitude over the few existing tools that are also based on this approach (**Supplementary Note 3** and **Supplementary Table 1**). In addition, existing agent-based simulators for rule-based models[21,23–25] lack coarse-graining capabilities and so cannot simulate many systems that show complex interactions, including most of the example models presented here. NFsim models are specified using an extended version of the BioNetGen Language (BNGL)[9], a widely used language for rule-based modeling that we have extended here to support advanced coarse-graining features (**Supplementary Note 2**). This latest version of BNGL is backward-compatible with models specified in previous versions of the language. NFsim also includes a set of Matlab-based utilities to run parameter scanning, perform data fitting and analyze simulation results. The software, user manual and practical example models are available at http://www.nfsim.org/.

RESULTS

NFsim

To overcome combinatorial complexity, NFsim represents each molecule as a distinct software object or agent and uses rules to describe reactions. Molecular agents have labeled domains that can each have a state value, for example, phosphorylated, methylated or activated (**Fig. 2a**). Domains also serve as binding sites for other molecules and thereby allow the assembly of arbitrarily large complexes. Rather than tracking all possible chemical species, NFsim follows only the molecular configurations that exist at a given time. Rules then track



b

```
begin parameters
   # define constant parameters of the model
   k1 0.02
   . . .
end parameters

begin molecule types
   # declare the types of molecules that exist
   A(b,c)
   B(a)
   C(a)
end molecule types

begin observables
   # patterns define basic simulation output
   Molecules AB_complex A(b!1).B(a!1)
end observables

begin reaction rules
   # rules define molecular interactions
   A(b) + B(a) -> A(b!1).B(a!1) k1
   A(b!1).B(a!1) -> A(b) + B(a) k2
   A(c) + C(a) -> A(c!1).C(a!1) k3
   . . .
end reaction rules
```

Figure 2 | Schematic overview of NFsim. (**a**) Rules keep track of the molecular agents that can participate in a reaction (dashed lines) by matching possible reactants to user-defined reactant patterns. At each simulation step, a rule event is generated, reactant molecules are selected and transformed, and the set of possible reactants for each rule is updated (see Online Methods). In addition to the binding and unbinding events, rules can also specify reversible reactions, domain state changes, molecular synthesis and molecular degradation. (**b**) Example BNGL file specifying the model depicted in **a**. Lines that begin with # are user comments ignored by NFsim. Parameters and rate constants are defined in the parameters block. Molecule types 'A', 'B' and 'C' are defined with a set of labeled domains 'a', 'b' and 'c'. Observables specify molecular patterns that provide simulation output, such as the pattern 'AB_complex'. Molecular domains are referenced in rules to define how molecules interact. 'A(b!1).B(a!1)' denotes that a bond connects domain 'b' of molecule 'A' to domain 'a' of molecule 'B'. Here binding of 'A' to 'B' is declared to be independent of the binding of 'A' to 'C', because in the reactant pattern of the rule, the site named 'c' of molecule 'A' is omitted. Reaction rates are given as parameters 'k1', 'k2' and 'k3'.

possible molecular events and interactions without ever having to exhaustively enumerate all reaction combinations. Rules are specified in terms of general patterns, which use wild cards to identify potential reactant molecules (**Fig. 2b**). With this underlying computational representation, a relatively small set of rules and agents can account for all possible reactions and chemical species in a system, even if the number of possible reactions or species is infinite.

To advance a simulation efficiently in time, we generalized a rule-based version of Gillespie's stochastic simulation algorithm (SSA)[22,26] (see Online Methods). Our method is guaranteed to produce the same results as the exact SSA by cycling over three primary steps. First, NFsim calculates the probability or propensity for each rule to take effect given the current molecular states. Second, it samples the time to the next reaction event and selects the corresponding reaction rule. Finally, NFsim executes the selected reaction by applying the rule and updating the molecular agents accordingly. This core simulation method was rigorously validated by comparing output for a series of models to that of a validated SSA implementation (**Supplementary Notes 4–10** and **Supplementary Figs. 1–5**).

NFsim efficiently simulates large reaction networks

To measure the efficiency of the method, we compared the runtime performance of NFsim with those of standard ODE and SSA approaches[9] by simulating the dynamics of proteins that can be phosphorylated at multiple, independent sites (**Fig. 1a**). As the number of sites increased, the growing combinatorial complexity of possible molecular states and interactions slowed ODE and SSA performance in proportion to the size of the reaction network (**Fig. 1c**). Eventually, memory requirements made ODE and SSA simulation impossible. The performance of NFsim, on the other hand, scaled linearly with the number of rules regardless of the number of reactions. For proteins with as few as six phosphorylation sites, NFsim outperformed both ODE and SSA simulations.

We obtained similar performance results for more complicated systems such as signaling through the Fcε receptor complex (FcεRI)[27]. This pathway has a central role in inducing the inflammatory response of the immune system to allergens (**Fig. 3a**, **Supplementary Table 2** and **Supplementary Note 8**). The dynamics of FcεRI signaling are difficult to study because the multivalent ligands cross-link the receptors into large aggregates[28]. As in the multi-site phosphorylation model, NFsim outperformed ODEs and the SSA as the models became more complex and the number of reactions and chemical species increased (**Fig. 3b**).

Predicting the distribution of protein aggregate sizes and configurations is also important, but standard methods limit the size of the protein complexes that form. For example, the largest complex considered in the FcεRI models consists of ten molecules[27]. NFsim did not impose such limits, as we showed with simulations of the trivalent-ligand, bivalent-receptor (TLBR) model[28], a general model of immunoreceptor aggregation based on FcεRI signaling (**Fig. 3c**,**d** and **Supplementary Note 5**). With the TLBR model, we also found that for a fixed rule set, NFsim performance was independent of the reaction network size and had a memory requirement that was linearly proportional to the number of molecules (**Supplementary Fig. 6**).

Model building and parameter estimation with NFsim

In mixed modeling and experimental settings, it is necessary to compare model predictions directly with experimental data and then to use experimental data to constrain and estimate model parameters. NFsim provides a number of practical features and Matlab-based tools to support and accelerate this model-building process, including a simple scripting language to change parameters mid-simulation and an interactive simulation session to assess reactions as they occur. Although parameter estimation of stochastic models is still an open problem[29], NFsim also includes a set of Matlab-based tools that enables users to perform parameter scanning and estimation of NFsim models with the standard fitting routines provided by Matlab. If necessary, users can also modify these scripts to design and implement more elaborate parameter estimation strategies.

We applied the parameter-scanning and estimation tools to the TLBR receptor aggregation model[28] (**Fig. 3c**,**d** and **Supplementary**

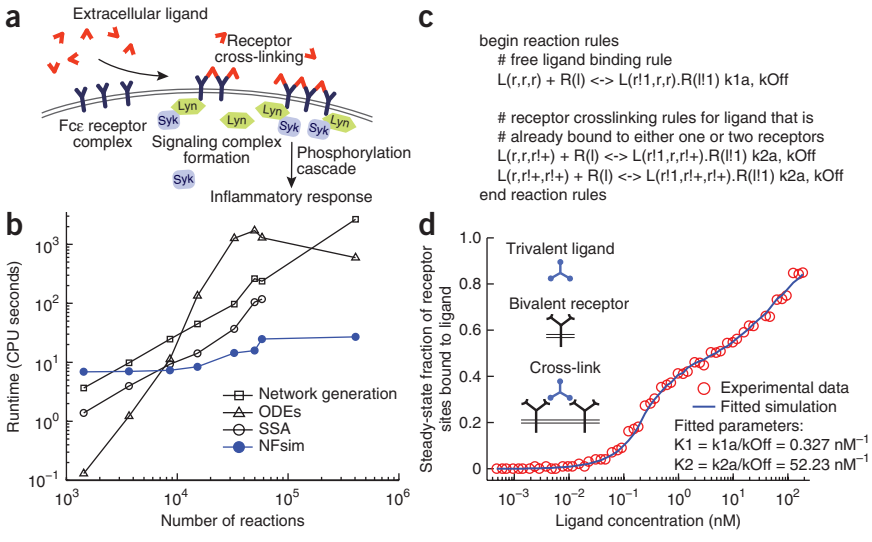


Figure 3 | Simulation performance and parameter estimation for receptor aggregation models. (**a**) Schematic of the early events of Fcε receptor (FcεRI) signaling. (**b**) The runtime performance of NFsim (filled circles) for a compendium of eight FcεRI signaling models of increasing reaction network size[27] as compared to ODE simulation (triangles), SSA simulation (open circles) and reaction network generation (squares). We used an optimized ODE solver[10] that can activate sparse-matrix representations and adaptive time-steps, leading to the apparent plateau in ODE performance. The largest model could not be simulated with the SSA because total computer memory (4 GB) was exhausted. (**c**) The trivalent-ligand, bivalent-receptor (TLBR) model serves as a simplified representation of FcεRI aggregation and is readily encoded in three BNGL rules, where the syntax 'r!+' indicates that the molecular domain named 'r' must be bound. (**d**) We used the NFsim suite of Matlab-based utilities to fit the TLBR model parameters to published flow cytometry data[30] that measured steady-state receptor-ligand binding.

Fig. 7). Recent flow cytometry experiments measured the relative amount of fluorescently labeled trivalent ligand bound to Fcε receptors of mast cells[30]. We first used the parameter-scanning capabilities of NFsim to automate multiple simulation runs where the initial ligand concentration was varied to compare model results directly with the flow cytometry data. We then fit the two free parameters of the TLBR model to the experimental binding data using a standard Matlab-based, nonlinear least-squares fitting routine. The binding constants obtained were within the range of values estimated in the original study[30], which previously required a problem-specific simulator and custom fitting code to estimate parameters (**Supplementary Note 6**).

NFsim tracks molecular connectivity in large complexes

When NFsim simulates reactions, it assumes that the reactants are well mixed. However, a key feature of NFsim is its ability to track and output the connectivity of large molecular aggregates and polymers. This feature is crucial for studies of dynamic aggregation and polymerization processes in which a spatially resolved reaction-diffusion simulation is too costly or not necessary. In addition, the complexity of molecular interactions in these processes often forces researchers either to make unwanted simplifications or to create custom software[11,13,31]. Here we show how NFsim can scale to larger, more complicated models and generate molecular connectivity output by simulating a rule-based model of cytoskeletal actin filament assembly, disassembly and branching.

As a starting point, we built on two recent models of actin polymerization[13,31]. The first model was used to investigate the stochastic severing of actin filaments in a dynamic steady-state regime of constant polymerization compensated by cofilin-mediated disassembly[13]. However, this study required custom simulation code and did not consider the reactions mediated by the Arp2/3 complex that form branched structures. The second model was used to investigate the branching process of actin filaments[31], but used a differential equation approach and so could not follow the distribution of filament lengths or the topology of actin structures. We combined these two models into a single rule-based model, which we then extended to account for the termination of filament elongation when capping protein binds filament ends (**Fig. 4a**, **Supplementary Note 9** and **Supplementary Tables 3–7**). Our extended model reproduced the results of the original modeling studies (**Supplementary Fig. 4**).

We then compared predictions from our model to an independent set of total internal reflection fluorescence (TIRF) microscopy experiments that measured filament branching in flow chambers[32] (**Fig. 4b,c**). To calibrate the model, we used NFsim's parameter-scanning routine to estimate the base rate of filament nucleation under the *in vitro* conditions. We also used NFsim's scripting function to simulate the addition and removal of reagents through the flow chamber (**Supplementary Note 9**).

Although the simulations of actin polymerization are not spatially resolved, the analysis tools of NFsim together with existing knowledge of the structure of actin filaments[5] allowed us to reconstruct and visualize the topology of interactions (**Fig. 4d** and **Supplementary Videos 1** and **2**). In these simulations, the average filament length was 136 subunits (~0.35 μm), which is consistent with *in vivo* observations of filaments at the leading edge of motile cells[5] (**Fig. 4e,f**). We then investigated other important aspects of the process, such as the distribution of branch numbers in steady-state actin structures exposed to severing reactions (**Fig. 4g,h**). We found that individual connected structures had only 3.7 branches on average but could grow to 40 or more branches over minutes (**Fig. 4g,h**). Thus, although actin filament structures tended to be sparsely connected by branch points, the system still had the capacity to rapidly assemble extended structures, which probably has an important role in enabling motile cells to respond quickly to stimuli.

Coarse-graining with functional rate laws

A key aspect of biological modeling involves deciding which processes need to be explicitly simulated and which can be safely

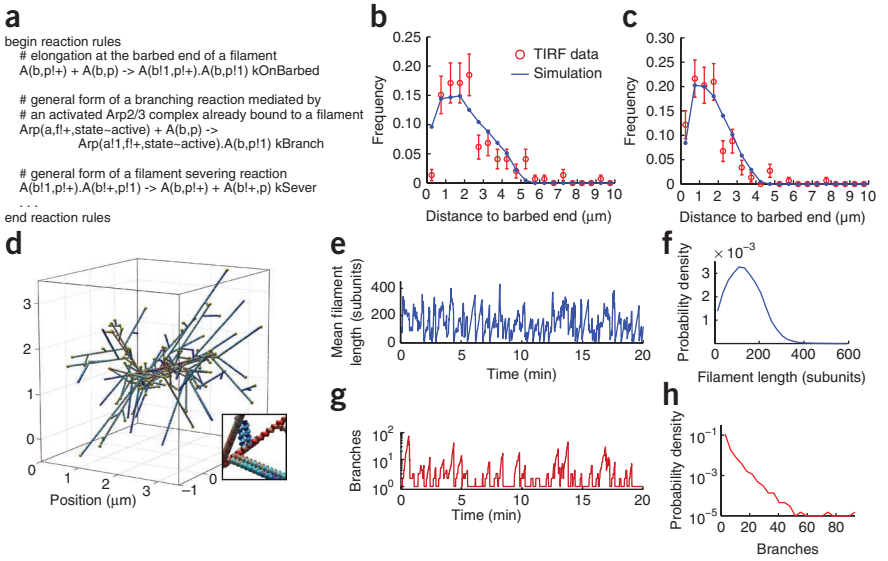


Figure 4 | Tracking molecular connectivity during simulation of cytoskeletal actin polymerization. (**a**) Simplified versions of the rules that model actin polymerization, branching and severing reactions. (**b**,**c**) Comparison of simulations with TIRF microscopy experiments that monitored filament branching in flow cells[32]. (**b**) Distributions of the positions of branching events taking place on preexisting mother filaments. Distances are measured relative to the position of the barbed end of the mother filament at the time of Arp2/3 addition. (**c**) Distributions of the positions of branching events taking place on portions of mother filaments that elongated after addition of Arp2/3. Distances are measured from the branching point to the position of the growing barbed end of the mother filament at the time of branching. Error bars denote s.e.m. (n = 146 for **b**; n = 154 for **c**). (**d**) Three-dimensional visualizations of the molecular connectivity generated using NFsim's output options depicting ATP-actin subunits (blue), ADP-P_i actin subunits (cyan), ADP actin subunits (red) and filament ends that are capped by capping protein (yellow). Inset shows a close-up of the visualization. (**e**,**f**) Simulated trajectory of the mean filament length of a connected actin structure (**e**) and corresponding steady-state distribution (**f**). (**g**,**h**) Simulated trajectory of the mean number of branches in a connected actin structure (**g**) and corresponding steady-state distribution (**h**).

approximated or ignored. In NFsim, users can coarse-grain complex reaction dynamics by defining reaction rates as arbitrary mathematical or conditional expressions. There are three different functional constructs than can be used in NFsim to define the rate law of a reaction: global functions, local functions and conditional expressions.

Global functions are defined in terms of time-dependent variables that are global to the entire system, such as total molecule counts. Although some other simulators support global functions[16,17], such functions are new to rule-based languages. Local functions are defined in terms of time-dependent variables that are local to individual molecular complexes. 'Local' in this sense refers to the connectivity of the molecules and not the spatial location of molecules. Local functions allow users to write a single mathematical function that is automatically evaluated separately for every molecular instance. Conditional functions allow the definition of logical switches (for example, turning enzymes 'on' or 'off'), piecewise linear functions and conditional dependencies similar to those used in Boolean networks. Implementation of functional and conditional rate laws required the development of new bookkeeping procedures to maintain algorithmic efficiency (Online Methods).

We illustrate the application of local and global functions with a model of the *Escherichia coli* chemotaxis signaling system, which bacteria use to navigate toward attractants and away from repellants[7] (**Supplementary Note 10**). Chemoreceptors embedded in the bacterial cell membrane assemble into highly cooperative signaling teams containing multiple receptor dimers (**Fig. 5a**), which control the activity of a histidine kinase, CheA. Binding of attractant or repellant molecules to individual receptors triggers a concerted and amplified response from the entire signaling team. The resulting change in kinase activity modifies the activity of the response regulator CheY. CheY diffuses through the cell and relays the signal to the flagellar motors, which drive the motion of the cell. Adaptation to persistent stimuli is mediated by two enzymes, CheR and CheB, that methylate and demethylate the receptors. Importantly, the rates of methylation and demethylation of an individual receptor depend on the conformation of that receptor, which in turn depends on the active or inactive state of other receptors in the same signaling team. Previously, the details of receptor cooperativity could be simulated only by building custom simulators[12,33].

Experimental and theoretical studies have shown that the activity of each signaling team can be represented in a coarse-grained manner using a Monod-Wyman-Changeux model[12,33]. In NFsim this is achieved by defining a local function, 'pOn', that depends on the methylation and ligand binding state of each receptor in a given signaling team (**Fig. 5b**). The argument 'x' refers to the particular complex over which pOn must be evaluated. The pOn function can then be used to represent how the rate of methylation of one receptor depends on the activity of its signaling team. This is illustrated in the block of code in **Figure 5b**, where the rate of the methylation reaction is multiplied by pOn(x). The '%x ' prefixed to 'Rec(m~?)' tells NFsim to evaluate the function pOn(x) on the entire complex (signaling team) that is connected to the indicated receptor. The local-function syntax also allows the arrangement of receptors in a signaling team to be modified without changing rule definitions. This allowed us to extend the model to reflect recent data about the topology of signaling teams from cryo-electron microscopy experiments (**Supplementary Note 10**). The extended model reproduced the wide dynamic range and sensitivity of bacteria to chemical attractants (**Fig. 5c**).

Thermal fluctuations and upstream signaling cause the flagellar motor to spontaneously transition between alternating rotational states. A coarse-grained, two-state model captures the key dynamic behavior of the motor response[34], which we can encode in NFsim using global functions (**Fig. 5d** and **Supplementary Note 11**). Note that the functions 'kPlus()' and 'kMinus()' are defined without arguments because they are global functions that are evaluated over the entire system. We then used the model to fit experimental measurements of the single-cell motor response curve[35] (**Fig. 5e**).

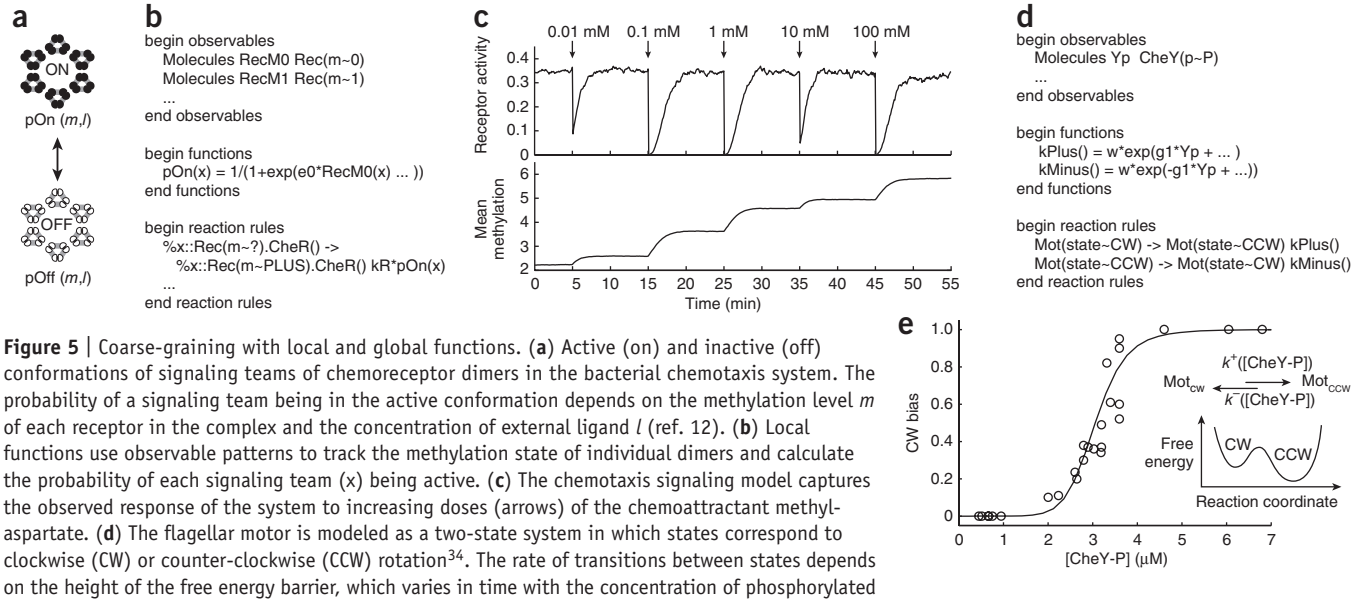


Figure 5 | Coarse-graining with local and global functions. (**a**) Active (on) and inactive (off) conformations of signaling teams of chemoreceptor dimers in the bacterial chemotaxis system. The probability of a signaling team being in the active conformation depends on the methylation level *m* of each receptor in the complex and the concentration of external ligand *l* (ref. 12). (**b**) Local functions use observable patterns to track the methylation state of individual dimers and calculate the probability of each signaling team (x) being active. (**c**) The chemotaxis signaling model captures the observed response of the system to increasing doses (arrows) of the chemoattractant methyl-aspartate. (**d**) The flagellar motor is modeled as a two-state system in which states correspond to clockwise (CW) or counter-clockwise (CCW) rotation[34]. The rate of transitions between states depends on the height of the free energy barrier, which varies in time with the concentration of phosphorylated CheY ([CheY-P]). This model is specified with an observable pattern that tracks CheY-P numbers and global functions that define the rate at which the motor switches states. (**e**) The model simulated with NFsim (line) captures the probability of being in the CW rotational state (CW bias) as a function of [CheY-P] as measured in single cell experiments[35] (circles); inset shows the free energy diagram governing the transitions between CW and CCW rotations.

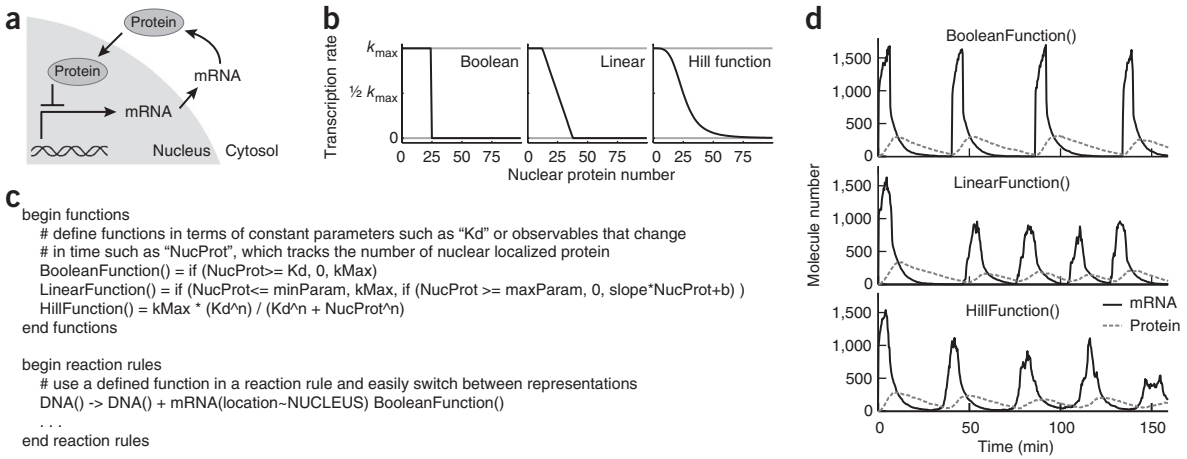


Figure 6 | Achieving multiple levels of resolution with conditional and functional rate-law expressions. (**a**) Schematic diagram of a biochemical system that can oscillate owing to negative feedback with a time delay that arises from nuclear shuttling and protein synthesis. For the system to oscillate, the negative feedback on the promoter activity must show some nonlinearity. (**b**) Nonlinearity represented by a simplified Boolean ON-OFF switch, a piecewise linear response or a Hill function. (**c**) Functions written in NFsim describing each of these coarse-grained representations. Conditional expressions, which can be arbitrarily nested as shown in the definition of the linear approximation, are interpreted as "if [*Condition*], then use [*RateExpression1*], else [*RateExpression2*]". (**d**) Time courses for mRNA (black solid line) and protein (gray dashed line) levels for the different coarse-grained representations.

Merging Boolean logic with kinetic reaction rules

A pressing challenge in computational biology lies in incorporating static, large-scale molecular interaction data with dynamic models of biochemical processes[15]. The difficulty arises because high-throughput '-omics' methods typically produce binary ON-OFF information, but standard dynamic modeling methods require knowledge of reaction mechanisms and rates. To provide the flexibility needed to merge different types of biochemical representations, NFsim supports the definition of reaction rates as conditional expressions of the form: "if [Condition], then use [Rate Expression 1], else use [Rate Expression 2]". Conditional expressions can be nested to construct complex logical sequences of control over reaction rules.

We illustrate conditional rules by simulating a prototypical model of oscillating gene expression driven by a negative feedback loop coupled to a time delay[36] (**Fig. 6a** and **Supplementary Note 12**). As a first approximation to describe the activity of the promoter, a Boolean ON-OFF switch might be used (**Fig. 6b**,**c**). As more detailed experimental information is obtained, however, our coarse-graining strategy allows users to update this approximation incrementally and to use, for instance, a piecewise linear model or a Hill function. As expected[36,37], the choice of approximation considerably affected the predicted dynamics of the system (**Fig. 6d**).

DISCUSSION

A fundamental strength of NFsim is the ability to add new molecular details to a model without modifying existing reactions or significantly affecting computational performance. Therefore, users can approach biological problems incrementally by starting with as few free parameters as possible, and can still rapidly incorporate new experimental information as needed. Although NFsim can simulate many models that were previously intractable, standard methods are generally more efficient for systems that have limited numbers of molecular states. Accordingly, NFsim is fully integrated with BioNetGen so that modelers can use standard simulation methods whenever these are possible or more efficient. Eliminating the need to rewrite models for different simulation methods greatly accelerates the model-building process. Furthermore, by using BNGL for model specification, NFsim models can be shared easily between researchers and expanded quickly in future work. Coarse-graining capabilities add to this versatility by allowing modelers to switch between detailed and simplified representations and to leverage logical interaction rules obtained from high-throughput studies in kinetic biochemical models. Finally, because NFsim uses agent-based software architecture, it provides a general framework for implementing models with agent-based or executable biology approaches[2]. The core features and performance of NFsim make it immediately useful to biologists who want to build, study and share executable models of complex biological systems.

METHODS

Methods and any associated references are available in the online version of the paper at http://www.nature.com/naturemethods/.

Note: Supplementary information is available on the Nature Methods website.

ACKNOWLEDGMENTS

We thank J. Berro, E. De La Cruz and T. Pollard for help with the actin assembly model, R. Gutenkunst, C. Henry, J. Hogg, G. Jentsch, W. Pontius and F. Xia for general testing, J. Hogg for assistance in extending BioNetGen, and R. Alexander, G. Altan-Bonnet, P. Cluzel, N. Frankel, L. Harris, W. Hlavacek and G. Jentsch for comments on the manuscript. Supported by the US National Science Foundation (CCF-0829836 to M.W.S. and T.E.; CCF-0829788 to J.R.F.).

AUTHOR CONTRIBUTIONS

M.W.S. wrote the software and performed all simulations. M.W.S., J.R.F. and T.E. designed the algorithms and research and wrote the manuscript.

COMPETING FINANCIAL INTERESTS

The authors declare no competing financial interests.

Published online at http://www.nature.com/naturemethods/.
Reprints and permissions information is available online at http://npg.nature.com/reprintsandpermissions/.

1. Anonymous. Defining the scientific method. *Nat. Methods* **6**, 237 (2009).

2. Fisher, J. & Henzinger, T.A. Executable cell biology. *Nat. Biotechnol.* **25**, 1239–1249 (2007).
3. Kitano, H. Computational systems biology. *Nature* **420**, 206–210 (2002).
4. Chen, W.W. *et al.* Input-output behavior of ErbB signaling pathways as revealed by a mass action model trained against dynamic data. *Mol. Syst. Biol.* **5**, 239 (2009).
5. Pollard, T.D. & Borisy, G.G. Cellular motility driven by assembly and disassembly of actin filaments. *Cell* **112**, 453–465 (2003).
6. Thomson, M. & Gunawardena, J. Unlimited multistability in multisite phosphorylation systems. *Nature* **460**, 274–277 (2009).
7. Wadhams, G.H. & Armitage, J.P. Making sense of it all: bacterial chemotaxis. *Nat. Rev. Mol. Cell Biol.* **5**, 1024–1037 (2004).
8. Hlavacek, W.S. *et al.* Rules for modeling signal-transduction systems. *Sci. STKE* **2006**, re6 (2006).
9. Faeder, J.R., Blinov, M.L. & Hlavacek, W.S. Rule-based modeling of biochemical systems with BioNetGen. *Methods Mol. Biol.* **500**, 113–167 (2009).
10. Hindmarsh, A.C. *et al.* SUNDIALS: Suite of nonlinear and differential/algebraic equation solvers. *ACM Trans. Math. Softw.* **31**, 363–396 (2005).
11. Knowles, T.P.J. *et al.* An analytical solution to the kinetics of breakable filament assembly. *Science* **326**, 1533–1537 (2009).
12. Hansen, C.H., Endres, R.G. & Wingreen, N.S. Chemotaxis in *Escherichia coli*: a molecular model for robust precise adaptation. *PLOS Comput. Biol.* **4**, e1 (2008).
13. Roland, J., Berro, J., Michelot, A., Blanchoin, L. & Martiel, J.L. Stochastic severing of actin filaments by actin depolymerizing factor/cofilin controls the emergence of a steady dynamical regime. *Biophys. J.* **94**, 2082–2094 (2008).
14. Rao, C.V. & Arkin, A.P. Stochastic chemical kinetics and the quasi-steady-state assumption: application to the Gillespie algorithm. *J. Chem. Phys.* **118**, 4999–5010 (2003).
15. Hyduke, D.R. & Palsson, B.Ø. Towards genome-scale signalling-network reconstructions. *Nat. Rev. Genet.* **11**, 297–307 (2010).
16. Hoops, S. *et al.* COPASI—a COmplex PAthway SImulator. *Bioinformatics* **22**, 3067–3074 (2006).
17. Ramsey, S., Orrell, D. & Bolouri, H. Dizzy: stochastic simulation of large-scale genetic regulatory networks. *J. Bioinform. Comput. Biol.* **3**, 415–436 (2005).
18. Hazelbauer, G.L., Falke, J.J. & Parkinson, J.S. Bacterial chemoreceptors: high-performance signaling in networked arrays. *Trends Biochem. Sci.* **33**, 9–19 (2008).
19. Danos, V., Feret, J., Fontana, W., Harmer, R. & Krivine, J. Rule-based modelling of cellular signalling. *Lect. Notes Comput. Sci.* **4703**, 17–41 (2007).
20. Lok, L. & Brent, R. Automatic generation of cellular reaction networks with Moleculizer 1.0. *Nat. Biotechnol.* **23**, 131–136 (2005).
21. Danos, V., Feret, J., Fontana, W. & Krivine, J. Scalable simulation of cellular signaling networks. *Lect. Notes Comput. Sci.* **4807**, 139–157 (2007).
22. Yang, J., Monine, M.I., Faeder, J.R. & Hlavacek, W.S. Kinetic Monte Carlo method for rule-based modeling of biochemical networks. *Phys. Rev. E* **78**, 031910 (2008).
23. Colvin, J. *et al.* Simulation of large-scale rule-based models. *Bioinformatics* **25**, 910–917 (2009).
24. Colvin, J. *et al.* RuleMonkey: software for stochastic simulation of rule-based models. *BMC Bioinformatics* **11**, 404 (2010).
25. Morton-Firth, C.J., Shimizu, T.S. & Bray, D. A free-energy-based stochastic simulation of the tar receptor complex. *J. Mol. Biol.* **286**, 1059–1074 (1999).
26. Gillespie, D.T. A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. *J. Comput. Phys.* **22**, 403–434 (1976).
27. Faeder, J.R. *et al.* Investigation of early events in FcεRI-mediated signaling using a detailed mathematical model. *J. Immunol.* **170**, 3769–3781 (2003).
28. Goldstein, B. & Perelson, A.S. 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).
29. Poovathingal, S.K. & Gunawan, R. Global parameter estimation methods for stochastic biochemical systems. *BMC Bioinformatics* **11**, 414 (2010).
30. Monine, M.I., Posner, R.G., Savage, P.B., Faeder, J.R. & Hlavacek, W.S. Modeling multivalent ligand-receptor interactions with steric constraints on configurations of cell-surface receptor aggregates. *Biophys. J.* **98**, 48–56 (2010).
31. Beltzner, C.C. & Pollard, T.D. Pathway of actin filament branch formation by Arp2/3 complex. *J. Biol. Chem.* **283**, 7135–7144 (2008).
32. Amann, K.J. & Pollard, T.D. Direct real-time observation of actin filament branching mediated by Arp2/3 complex using total internal reflection fluorescence microscopy. *Proc. Natl. Acad. Sci. USA* **98**, 15009–15013 (2001).
33. Mello, B.A. & Tu, Y. An allosteric model for heterogeneous receptor complexes: Understanding bacterial chemotaxis responses to multiple stimuli. *Proc. Natl. Acad. Sci. USA* **102**, 17354–17359 (2005).
34. Tu, Y. & Grinstein, G. How white noise generates power-law switching in bacterial flagellar motors. *Phys. Rev. Lett.* **94**, 208101–208104 (2005).
35. Cluzel, P., Surette, M. & Leibler, S. An ultrasensitive bacterial motor revealed by monitoring signaling proteins in single cells. *Science* **287**, 1652–1655 (2000).
36. Novak, B. & Tyson, J.J. Design principles of biochemical oscillators. *Nat. Rev. Mol. Cell Biol.* **9**, 981–991 (2008).
37. Mather, W., Bennett, M.R., Hasty, J. & Tsimring, L.S. Delay-induced degrade-and-fire oscillations in small genetic circuits. *Phys. Rev. Lett.* **102**, 068105 (2009).

ONLINE METHODS

Implementation details. NFsim was implemented in ANSI/ISO standard C++ and runs on all major operating systems, including Windows, Mac and Linux. The source code of NFsim and all accompanying Matlab-based utilities is provided under the GNU general public license (GPL) online from the NFsim website (http://www.nfsim.org/). Technical documentation, example models and a comprehensive user manual are also provided online. BioNetGen, written in Perl, was extended to support the new version of BNGL presented here and comes packaged with NFsim.

Core simulation algorithm. At initialization, all simulation objects representing molecules, rules, and functions are created according to a general BNGL input file through an intermediate and equivalent XML specification format that was designed in this work. Rule objects are used to determine the initial set of molecules that can participate in each rule (**Fig. 2a**). Rule objects also calculate their reaction propensity based on the number of possible reactants, rate constants and a corresponding rate law. Then, NFsim iterates over the following steps until the user-specified simulation end time is reached.

Step 1 is to sample the time of the next reaction event. The waiting time before the next rule-generated event follows a Poisson distribution[22]. Thus, the timing of events is sampled with the same approach used by the SSA[26]. Namely, if there are m rules with reaction propensities r_i with $i = 1,\ldots,m$, the duration of time τ before the next event is sampled as

$$\tau = -(1/r_{\text{tot}}) \cdot \ln(\rho_1)$$

where r_{tot} is the sum of the propensities of each of the m rules and ρ_1 is a uniformly distributed random number on the interval (0,1).

Step 2 is to select the next rule to fire. NFsim provides two methods for selecting the next reaction event. The default method, analogous to the SSA direct method[26], selects the next event by finding the smallest integer J such that

$$\sum_{j=1}^{J} r_j > \rho_2 \cdot r_{\text{tot}}$$

where ρ_2 is a uniformly distributed random number on the interval (0,1). The default method incurs an $O(m)$ computational cost where m is the number of rules. An alternative method operates by segregating rules based on propensities into logarithmic classes, and uses a von Neumann rejection scheme for event selection[38]. The logarithmic classes method has a computational cost $O(\log_2 m)$ equivalent to the scaling of the Gibson-Bruck method[39] but incurs an overhead cost for maintenance of the logarithmic classes.

Step 3 is to select the reacting molecules. For basic rate laws, reactants have an equal probability of participating in a rule and are chosen accordingly. Rules can also occur with a rate that depends functionally on local context, which requires additional computation to select reactants in proportion to individual propensities (see section on functionally defined rate laws below).

Step 4 is to determine whether reactants satisfy nonlocal connectivity constraints. In NFsim it is possible to simulate reactions that take place between two sites on the same molecular complex, which result in the formation of intramolecular bonds. Distinguishing between inter- and intramolecular bonds is costly, but is usually most efficiently accomplished after an event has been selected[22]. If such constraints are not satisfied, then the event is rejected, the reaction rule is not applied, simulation time is still advanced and the program returns to step 1.

Step 5 is to transform selected reactants. Rule objects directly transform reactant molecules into the product species.

Step 6 is to update reactant lists and generate output. Molecular agents update their reactant-rule references to account for their ability to participate in a new set of rules. Simulation output is generated if needed during this step.

Step 7 is to update rule propensities to reflect the new state of the system and advance the simulation time.

Coarse-graining with functionally defined rate laws. NFsim allows users to define mathematical expressions in terms of the number of molecules that match particular patterns and constant parameters in the system. Mathematical expressions are parsed and compiled into bytecode for efficient evaluation using the muParser library (http://muparser.sourceforge.net/). All basic mathematical operators, including trigonometric functions, are supported. In addition, conditional statements can be used to logically control how the expression is evaluated. Expressions can be used to define the rate law of a reaction in two ways. First, the function can reference global patterns that count molecules throughout the entire system. Second, the function can reference local patterns that count the number of matching molecules over a single connected complex or molecule. Conditional statements are evaluated as either global or local rate laws depending on the variables referenced.

Simulation of global rate laws is straightforward and has been implemented in other simulation platforms[16,17]. Briefly, the global function is re-evaluated if needed after each simulation step and subsequently used to update the rate law and propensity of the dependent reactions. Simulation of local functions is much more challenging because patterns must be identified and tracked separately for each molecular complex or molecule in the system. As the formation of molecular complexes is a dynamic process, the value of local functions will change frequently during simulation. We call a reaction rule that depends on this type of local pattern a distribution of rates (DOR) rule because each possible reactant can participate in the same rule with a different rate, generating a distribution of rates which must be considered. For a DOR rule with m possible reactant molecules, the contribution of the single reactant molecule i (with $i = 1,\ldots,m$) to the total propensity of the DOR rule is $k{\cdot}n{\cdot}f(x_i)$, where $f(x_i)$ is the functional expression evaluated on x_i, the local set of molecules connected to reactant i. Here k is a rate constant and n is the number of possible reactant partners in a bimolecular rule, or 1 in the case of unimolecular rules. The total propensity of a DOR rule is the sum of the propensity contributions of each of the m reactant molecules:

$$k \cdot n \cdot \sum_{i=1}^{m} f(x_i)$$

The DOR rule maintains the total propensity value during simulation as individual agents are updated. Individual agents that participate in a DOR rule keep a reference to the value of the local function evaluated over its local complex so that updates are efficient.

A DOR rule calculates its total propensity and is selected to fire in step 2 of the core algorithm in the same manner as all other rules. If a DOR event is fired, the reactant molecule that participates must be randomly selected in proportion to its individual propensity.

doi:10.1038/nmeth.1546

To perform the selection efficiently, NFsim uses a binary search tree to organize the propensities of each reactant that participates in the given DOR rule. Therefore, applying step 3 of the core algorithm on DOR rules incurs an $O(\log_2 m)$ cost, where m is the number of matches to the reactant pattern used in the rule.

38. Fricke, T. & Wendt, D. The Markoff automaton: a new algorithm for simulating the time-evolution of large stochastic dynamic systems. *Int. J. Mod. Phys. C* **6**, 277–306 (1995).
39. Gibson, M.A. & Bruck, J. Efficient exact stochastic simulation of chemical systems with many species and many channels. *J. Phys. Chem. A* **104**, 1876–1889 (2000).