Hierarchical Optimization of Biochemical Networks

Nisha Ann Viswan, Alexandre Tribut, Manvel Gasparyan, Ovidiu Radulescu, Upinder S Bhalla

Hierarchical optimization of biochemical networks

Nisha Ann Viswan[1,3], Alexandre Tribut[2,4], Manvel Gasparyan[2], Ovidiu Radulescu[2,*], Upinder S. Bhalla[1,*]

1 National Centre for Biological Sciences, Tata Institute of Fundamental Research, Bangalore, India.
2 Laboratory of Pathogens and Host Immunity, University of Montpellier, CNRS and INSERM, Montpellier, France.
3 The University of Trans-Disciplinary Health Sciences and Technology, Bangalore, India.
4 Ecole Centrale de Nantes, Nantes, France.

* To whom correspondence should be addressed: bhalla@ncbs.res.in, ovidiu.radulescu@umontpellier.fr.

Abstract

Biological signalling systems are complex, and efforts to build mechanistic models must confront a huge parameter space, indirect and incomplete data, and frequently encounter multiscale and multiphysics phenomena. We present HOSS, a framework for Hierarchical Optimization of Systems Simulations, to address such problems. HOSS operates by breaking down extensive systems models into individual pathway blocks organized in a nested hierarchy. At the first level, dependencies are solely on signalling inputs, and subsequent levels rely only on the preceding ones. We demonstrate that each independent pathway in every level can be efficiently optimized. Once optimized, its parameters are held constant while the pathway serves as input for succeeding levels. We develop an algorithmic approach to identify the necessary nested hierarchies for the application of HOSS in any given biochemical network. Furthermore, we devise two parallelizable variants that generate numerous model instances using stochastic scrambling of parameters during initial and intermediate stages of optimization. Our results indicate that these variants produce superior models and offer an estimate of solution degeneracy. Additionally, we showcase the effectiveness of the optimization methods for both abstracted, event-based simulations and ODE-based models.

Keywords: systems biology, mechanistic models, optimization, modularity, AutoML.

Author summary

Biochemical pathway models integrate quantitative and qualitative data to understand cell functioning, disease effects, and to test treatments in silico. Constructing and optimizing these models is challenging due to the complexity and multitude of variables and parameters involved. Although hundreds of biochemical models have been developed and are available in repositories, they are rarely reused. To enhance the utilization of these models in biomedicine, we propose HOSS, an innovative hierarchical model optimization method. HOSS takes advantage of the modular structure of pathway models by breaking down large mechanistic computational models into smaller modules. These modules are then optimized progressively, starting with input modules

and following causality paths. This method significantly reduces the computational burden as each step involves solving a simpler problem. By making the optimization process more manageable, HOSS accelerates the lifecycle of biochemical models and promotes their broader use in biomedical research and applications.

Introduction

Many large biochemical pathway models have been developed since the early days of systems biology. These models take many different formalisms, including visual representations of data, such as protein interaction networks [1], and executable models like ordinary differential equations [2], boolean models [3], and more recently, HillTau abstractions [4]. Among executable models, ODEs provide accurate representation of pathway dynamics, but incorporate many unknown parameters.

Two key advances have opened up the possibility of scaling up systems models substantially, in terms of complexity and reproducibility. First, there is now a rich ecosystem of data resources and data mining resources, both from structured databases and from the much broader but unstructured scientific literature. These approaches have already been used to scale up pathway diagrams and interaction networks [5]. Second, the advent of numerous high-throughput methods such as phosphoproteomics, imaging, and mass spectrometry promise far larger and internally consistent datasets (see [6]) than the extant patchwork of precise but once-off biochemical experiments performed by individual laboratories.

However, in spite of a few attempts [7, 8], the model development process is far from being automatic and standardized. In particular, parameter optimization methods have been implemented in a fragmented manner [9–17]. This is in part due to the very wide diversity of experimental inputs used to constrain such models, but also due to the inherent contradictions and incompleteness of the parameter constraints. For example, to compensate for the incompleteness of specific datasets and further constrain the model, it is not uncommon to amalgamate the findings from various publications. These data sources may utilize experimental preparations that differ significantly or even involve different classes of organisms [2]. This practice introduces inconsistent experimental inputs into the model. Consequently, model development is highly idiosyncratic, and different modelers may arrive at quite distinct models or parameter sets despite drawing on similar data sources.

There have been previous ambitious efforts to systematically funnel many experimental inputs into detailed and biologically driven models [18]. Such efforts require the integration of large-scale systematic data gathering with data management and modeling (e.g, SPEDRE [19]). The current paper focuses on standardizing the calibration and optimization stages of model development, given a large but incomplete set of experimental data. We build on our recently developed framework (FindSim [20]) for curating a very wide range of biochemical and physiological experiments, representing it in a consistent format, and using such curated experiment definitions to drive multiscale models. In principle, each new experiment should improve our understanding of biological systems, and thus help us to refine models of these systems. This amounts to a multi-parameter optimization problem. Its result should be a model that fits experiments as well as possible within the limitations of the model, while incorporating expert evaluation of the relative reliability of different experiments.

We formalize and implement a general methodology for solving multi-parameter optimization problems by leveraging the modularity property of biochemical networks. These networks consist of groups of species and biochemical reactions that function autonomously. Utilizing modularity for stepwise parametric optimization is a natural approach and has been applied to Petri Net models of signaling pathways [21]. However,

this application lacked generality and was not algorithmically formulated. In this work, we present innovative algorithmic approaches for systematically performing modular decomposition of biochemical networks, described by various methods including ordinary differential equations and event-based modeling. Additionally, we connect the modular approach to hierarchical optimization, offering fully automated methods to handle data and models in a hierarchical manner.

Inspired by game theory and now with multiple applications in science and engineering, hierarchical optimization decomposes a complex optimization problem into several coupled simpler problems [22, 23]. Although NP-hard in general, hierarchical optimization becomes easier for nested hierarchies, where lower levels depend on fewer parameters than the upper levels. We refer to such a method as nested hierarchical optimization and provide algorithmic solutions for implementing it in pathways.

We report the development of an optimization pipeline, HOSS, implementing nested hierarchical optimization. We illustrate its use on an extant database of over 100 experiment definitions in the domain of synaptic signalling to improve the parameterization of a set of models of major signalling pathways involved in synaptic signalling and cell proliferation. HOSS utilizes FindSim [20] in order to consistently evaluate models based on a specified set of experiments.

We show how our hierarchical approach addresses many of the challenges of parametric optimization problems, and outperforms a *flat* (i.e., non hierarchical single stage) optimization approach in efficiency, structure, and accuracy.

Our pipeline implements tools and standard formats for automatically handling models, data and machine learning (ML) scenarios. All our models are encoded using the Systems Biology Markup Language (SBML), a well-established format in Systems Biology. Both data and optimization choices, including flat and hierarchical optimization, with the definition of submodels in the hierarchical case, are encoded using JavaScript Object Notation (JSON) files. The goal is to make ML more reproducible and accessible to non-experts, while increasing productivity for experts. This situates our effort within the field of Automated Machine Learning (AutoML), a relatively new area that makes advanced ML techniques more accessible and accelerates research processes in computational biology [24].

Methods

Mathematical formalism

Objective (cost) function

A popular choice of objective function is the log likelihood. For data with normally distributed deviations, it reads:

$$L(\theta, s, \sigma) = \frac{1}{2} \sum_{i=1}^{N} \left[\log(2\pi\sigma_i^2) + \left(\frac{y_i - s_i x_i(\theta)}{\sigma_i} \right)^2 \right], \tag{1}$$

where y_i and $x_i(\theta)$ are observed and predicted concentrations of the i^{th} observed species, respectively, and θ are kinetic parameters. Parameters s_i are scaling parameters, accounting for the fact that the measurements are not absolute and σ_i are standard deviation parameters (see [25]).

In the HOSS calculations we perform two levels of scoring. First, for each experiment for which the sub-model is tested, we obtain a normalized root-mean-square cost similar to the above calculation, except it is normalized to the maximum of the experiment readout for molecule y_i among all the observations of the same variable at

different times or in different conditions:

$$\mathrm{NRMS}(\theta) = \sqrt{\frac{1}{N_d}\sum_{i,k}\left(\frac{y_{ik} - x_i(t_k,\theta)}{m_i}\right)^2}, \tag{2}$$

where m_i is the maximum value of the observed variable y_i and $x_i(t_k,\theta)$ is its predicted value at the time t_k, and N_d is the number of data points (terms in the sum).

To handle multiple data sets and multi-objective optimization we adopt a weighted sum approach. We define the weighted normalized cost, that combines values of (2) obtained in multiple datasets:

$$\mathrm{WNRMS}(\theta) = \sqrt{\frac{\sum_j w_j(\mathrm{NRMS}_j(\theta))^2}{\sum_j w_j}}, \tag{3}$$

where w_j, and NRMS_j are positive weights, and normalized root mean costs of individual datasets, respectively.

Flat and Hierarchical optimization

Parameter optimization involves minimization of an objective function $f : S \subset \mathbb{R}^n \to \mathbb{R}$, where S is a space of constraints, $\boldsymbol{p} \in S$ a vector of parameters. The flat method consists of solving the problem:

$$\min_{\boldsymbol{p}\in S} f(\boldsymbol{p}). \tag{4}$$

There are many methods to solve (4). In our framework we use multistart optimization, by launching local search procedures from randomly chosen starting points generated uniformly in logarithmic scale:

$$\boldsymbol{p} = \tilde{\boldsymbol{p}}\exp(\log(\boldsymbol{a}) + \log(\boldsymbol{b}/\boldsymbol{a})\boldsymbol{U}), \tag{5}$$

where $\tilde{\boldsymbol{p}}$ is a nominal guess, $\boldsymbol{U} = (U_1, U_2, \ldots, U_n)$ a vector of random, independent variables whose distribution is uniform over $[0,1]$ or standard normal, $\boldsymbol{a} = (a_1, a_2, \ldots, a_n)$ and $\boldsymbol{b} = (b_1, b_2, \ldots, b_n)$ are vectors of positive scales, such that $0 < a_i < 1 < b_i$ for $1 \le i \le n$. All the vector multiplications in (5) are elementwise. By using this procedure, the range of start parameters is from $\bar{p}_i a_i$ to $\bar{p}_i b_i$.

We refer to this procedure as *parameter scrambling*. Despite its simplicity, multistart optimization with logarithmic sampling has proven to be effective in benchmarks of biochemical pathways [26].

In hierarchical optimization [22], K sub-problems, each one defined by an objective function $f_i : S \longrightarrow \mathbb{R},\ i = 1, \ldots, K$, are solved iteratively. Parameters of the problem are grouped in K groups $\boldsymbol{p} = (p_1, \ldots, p_K)$, where $p_i \in \mathbb{R}^{n_i}$ for $1 \le i \le K$ and $n = n_1 + \ldots + n_K$. We look for $\boldsymbol{p}^* = (p_1^*, \ldots, p_K^*) \in S \subset \mathbb{R}^n$, solution of:

$$\begin{aligned}
&\min_{p_K} f_K(p_1^*, \ldots, p_{K-1}^*, p_K) && \text{where } p_{K-1}^* \text{ solves}\\
&\min_{p_{K-1}} f_{K-1}(p_1^*, \ldots, p_{K-2}^*, p_{K-1}, p_K) && \text{where } p_{K-2}^* \text{ solves}\\
&\quad\vdots && \quad\vdots\\
&\min_{p_2} f_2(p_1^*, p_2, \ldots, p_K) && \text{where } p_1^* \text{ solves}\\
&\min_{p_1} f_1(p_1, p_2, \ldots, p_K) && p \text{ subject to } S.
\end{aligned} \tag{6}$$

The case $K = 2$ is known as bilevel optimization [23]. In this case the optimization of f_1 is called lower-level problem, whereas the optimization of f_2 is the upper-level problem.

The problem (6) is difficult, because each individual problem has to be solved for multiple values of the remaining variables, all subjected to the constraints S. Indeed, it has been proved that even apparently simple bilevel optimization problems are NP-hard [23]. However, the solution of bilevel optimization is straightforward if the lower-level problem has unique analytic solution. In this case, the naive algorithm, utilizes the solution of the lower-level problem to eliminate p_1 and reduce the upper-level optimization to minimizing a composed function that depends on p_2 only, is effective. Bilevel optimization with analytic solution for the lower-level problem has already been used for systems biology models. In this case the lower-level parameters are the scaling and standard deviation parameters s_i, σ_i introduced in (1), that can be optimized by analytic formulas, see [25].

Another simple hierarchical optimization case is when the functions $f_i, 1 \leq i \leq K$ depend on nested sets of parameters and the set of constraints factorizes $S = S_1 \times S_2 \times \ldots \times S_K$. Then, (6) reads:

$$\begin{aligned}
&\min_{p_K \in S_K} f_K(p_1^*, \ldots, p_{K-1}^*, p_K) && \text{where } p_{K-1}^* \text{ solves} \\
&\min_{p_{K-1} \in S_{K-1}} f_{K-1}(p_1^*, \ldots, p_{K-2}^*, p_{K-1}) && \text{where } p_{K-2}^* \text{ solves} \\
&\quad \vdots && \quad \vdots \\
&\min_{p_2 \in S_2} f_2(p_1^*, p_2) && \text{where } p_1^* \text{ solves} \\
&\min_{p_1 \in S_1} f_1(p_1)
\end{aligned} \tag{7}$$

We call the problem (7), **nested hierarchical optimization**. Nested hierarchical optimization can be solved iteratively, starting with the last problem in (7).

Nested hierarchical decompositions

A biochemical model is defined by a set of reactions $\mathcal{R}$ and a set of species $\mathcal{S}$. We also define the stoichiometric matrix $\boldsymbol{S}$, whose elements S_{ij} represent the number of molecules of the species i produced (if $S_{ij} > 0$) or consumed (if $S_{ij} < 0$) by the reaction j. Furthermore, the reaction rate vector $\boldsymbol{R}(\boldsymbol{x}, \boldsymbol{p}) = (R_1(\boldsymbol{x}, \boldsymbol{p}), \ldots, R_r(\boldsymbol{x}, \boldsymbol{p}))$ is a function of species concentrations $\boldsymbol{x} = (x_1, \ldots, x_N)$ and kinetic parameters $\boldsymbol{p}$. Each reaction j is characterized by a parameter vector $\boldsymbol{p}_j$, therefore we have $\boldsymbol{p} = (\boldsymbol{p}_1, \ldots, \boldsymbol{p}_r)$.

Species concentrations evolve in time as a result of chemical reactions. These define a semiflow (time dependent mapping of the species concentrations, enabling the computation of future concentrations based on the present ones) $\boldsymbol{\phi}(t, \boldsymbol{x}; \boldsymbol{p}), t \geq 0$ such that $\boldsymbol{x}(t) = \boldsymbol{\phi}(t, \boldsymbol{x}_0; \boldsymbol{p})$ represents the species concentration vector starting from initial values $\boldsymbol{x}(0) = \boldsymbol{x}_0$. The semiflow results from the integration of ODEs in chemical kinetics models or from the simulation of event driven dynamics in HillTau abstractions [4].

Some species forming a subset $BS \subset \mathcal{S}$ are buffered, and their concentrations are kept constant.

Our construction relies on the following concept. We call **autonomous pair**, a pair of reaction and species subsets $(I, J), I \subset \mathcal{S}, J \subset \mathcal{R}$ that satisfy:

1. if a species is in the subset I, then all the reactions consuming or producing this species are in the corresponding reaction subset J, namely if $i \in I$ then $j \in J$ whenever $S_{ij} \neq 0$.

2. if a reaction is in the subset J, then all the species on which the reaction rate depends are in the corresponding species subset I, unless these species are buffered, i.e. if $j \in J$ then $i \in I$ whenever $\frac{\partial R_j}{\partial x_i} \neq 0$ and $i \notin BS$.

Let $\boldsymbol{x}_I$ be the concentration vector of the species in I and $\boldsymbol{p}_J$ the kinetic constants of the reactions in J. From the above definition it follows that $\boldsymbol{x}_I$ can be computed at any positive time t by a semiflow depending only on the parameters $\boldsymbol{p}_J$, namely $\boldsymbol{x}_I(t) = \boldsymbol{\phi}_I(t, \boldsymbol{x}_I(0); \boldsymbol{p}_J)$. Consider the data subset D_I, consisting of observations $\boldsymbol{y}_I$ of the species $\boldsymbol{x}_I$ only. Then, the objective function measuring the difference between observed and predicted values of $\boldsymbol{x}_I$ depends only on $\boldsymbol{p}_J$, namely

$$f_J(\boldsymbol{p}_J) = \sum_{i \in I} \sum_k (y_{ik} - \phi_i(t_k, \boldsymbol{x}_I(0); \boldsymbol{p}_J))^2. \tag{8}$$

Suppose now that we find species and reaction subsets,

$$\begin{aligned} I_1 \subset I_2 \subset \ldots \subset I_K = \mathcal{S}, \\ J_1 \subset J_2 \subset \ldots \subset J_K = \mathcal{R}, \end{aligned} \tag{9}$$

such that (I_k, J_k) are autonomous pairs for all $1 \leq k \leq K$.

We call (9) a **nested hierarchical decomposition**. Optimization of the objective functions (8) can then be done hierarchically, as in (7).

Constructing nested hierarchical decompositions using the interaction graph

Let us define the interaction digraph as $\mathcal{I} = (V, E)$, where V is the set of vertices (all species) and E is the set of edges. A pair of species $(i, j) \in E$ defines an edge from i to j if and only if there is a reaction that consumes or produces the species j, and its rate depends on the concentration of the species i. This graph is used to define causality relations between species, namely we say that j is causal to i, $j \rightsquigarrow i$, if j is connected to i by a path in $\mathcal{I}$. All the species j causal to i are needed for computing the time evolution of x_i.

Strongly connected components (SCC) of $\mathcal{I}$ are subsets $K \subset V$ such that $j \rightsquigarrow i$ and $i \rightsquigarrow j$ for all $i, j \in K$, maximal with respect to this property. In this paper, we refer to SCCs as **blocks**. Blocks form a partition of the species set I. This partition can be used to define a SCC quotient graph as follows: blocks are vertices of the SCC quotient graph, and two blocks are connected if there is one species in one block connected in the interaction graph to a species in the other block. The SCC quotient graph is always acyclic (see Figure 1 and [27]).

The following property is important for building nested hierarchical decompositions.

Property: For any block K and any subset I of an autonomous pair (I, J), one has either $K \subset I$ or $K \cap I = \emptyset$.

Thus, we can build a nested hierarchical decomposition by using the blocks and the quotient graph. The first level subset I_1 is the union of blocks that are roots of the quotient graph, i.e. blocks having no incoming connections. The corresponding reaction subset J_1 is made of all reactions producing or consuming species from I_1. The next level I_2 is obtained by adding to the roots all the blocks receiving direct connections only from the roots, and so on and so forth. Algorithmically, one must associate a hierarchical level l to each block, defined as the length of the longest path from the roots to the block (see Figure1). Then, the set I_l is the union of all blocks with a hierarchical level smaller than l.

Although the nested hierarchical decomposition (9) is not unique, the decomposition obtained by this procedure is unique and has the advantage of minimality. More precisely, I_1 is the minimal subset containing the root blocks, such that (I_1, J_1) is

autonomous. I_2 is the minimal subset containing the species I_1 and all the species receiving direct interactions only from I_1.

The quotient graph also provides a useful data structure for parallel optimization of the parameters. Thus, each tree originating from a root corresponds to terms in the objective function that can be optimized independently of the others.

Hierarchical decompositions with feedback

In some signaling pathway models, downstream molecules regulate upstream ones through feedback [28]. This can result in all species influencing each other, forming a single block where hierarchical and flat optimizations are equivalent. However, even in these cases, we can identify smaller blocks and decompose the network hierarchically. The autonomy of these resulting blocks is only approximate, but it allows us to benefit from hierarchical optimization. An iterative approach, starting with approximate hierarchical optimization and continuing with flat optimization, is a good option in this scenario.

We summarize the hierarchical decomposition procedure in the case with feedback, leaving the details to a separate paper.

In the presence of feedback, two concepts are key for the hierarchical decomposition.

The first concept is *r-causality*. A species j is r-causal to a species i, $j \overset{r}{\rightsquigarrow} i$, if j is connected to i by a path in the interaction graph $\mathcal{I}$, of length smaller than or equal to r. The introduction of r-causality imposes an upper limit on the length of paths connecting species in the interaction graph. By taking the value of r sufficiently large one can thus break feedback loops.

The other concept is *agony*, a measure used to quantify the hierarchical organization of directed networks [29,30]. It helps in identifying and evaluating the hierarchical structure within a network by penalizing the inconsistencies present in the hierarchy. More precisely, integer scores representing the level in the hierarchy are associated to each species in the network. Then agony is a function of all these scores and of the interaction graph, that penalizes the edges for a node with high level to a node with lower level. The set of scores that minimizes agony is then used to define the hierarchy.

Our procedure to compute the hierarchical decomposition of a network with feedback is as follows:

- First define r-blocks, such as maximal subsets such that any species is r-causal to any other.
- Because r-causality is not an equivalence relation, r-blocks can overlap. Generate a consolidated r-block partition (also named r-SCC partition) by agglutinating r-blocks that overlap.
- Use the r-SCC partition to define a r-quotient graph in the same way as the quotient graph was defined from the SCC partition. The nodes of the r-quotient graph are the consolidated r-blocks.
- Use agony to define hierarchical levels in the r-quotient graph.

Another strategy would be to apply agony directly to the interaction graph. However, the computational burden is reduced, and the optimization result is robust by using the r-quotient graph instead. The value of r has to be chosen not too large to avoid one r-block that contains all the species, and not too small to avoid many r-blocks that contain just one species. For the models studied in this paper r equal to one or two is good enough to avoid having just one block or many blocks containing just one species.

The hierarchical decomposition algorithms used in this paper were implemented in Python (see HiNetDecom in the Availability section).

The application of this procedure to signalling networks with feedback is illustrated in the Figure 2.

Signal back-propagation, reduced Michaelis-Menten mechanisms, and irreversible reactions

Although in a signalling cascade the signal usually propagates in only one directions, there are situations when both forward and backward propagation are possible. Similar to the case of feedback, minimal blocks and autonomous pairs can encompass the entire pathway in this instance as well. Signal backpropagation can occur due to enzyme sequestration, a phenomenon in which the active enzyme from an upstream tier of the signalling cascade is sequestered as part of the enzyme-substrate complex [31]. The back-propagation phenomenon disappears under quasi-steady state (QSS) conditions, when the enzyme-substrate complexes have low concentrations [32, 33]. As signalling pathways models often assume QSS, it is useful to have a tool that reduces mass action models by eliminating complexes. The reduced models can then be decomposed hierarchically using methods proposed above. As a result of the reduction, the hierarchical decomposition is improved: some large blocks split into smaller ones. This holds whenever there is sequestration related back-propagation, whether this is accompanying by feedback or not.

The QSS reduction of Michaelis-Menten mechanism is based on identifying in the reaction network of motifs of the type:

$$S_i + E_i \underset{k_i^-}{\overset{k_i^+}{\rightleftharpoons}} ES_i \xrightarrow{k_{\text{cat}}^i} P_i + E_i,$$

and find all motifs that the same enzyme E_i.

Let I_i be the subset of reactions using the same enzyme E_i, i.e. $E_j = E_i,\ \forall j \in I_i$. Then $\forall j \in I_i$ the Michaelis-Menten mechanism is replaced by a single reaction

$$S_j \xrightarrow{V_j} P_j,$$

with the rate

$$V_j = k_{\text{cat}}^j \frac{E_i S_j / k_m^j}{1 + \sum\limits_{l \in I_i} S_l / k_m^l},$$

where $k_m^j = (k_j^- + k_{\text{cat}}^j)/k_j^+$.

The rates of the reduced reaction depends on the concentration of the substrate S_j, but also on enzyme E_i and on the substrates $S_k, k \in I_i, k \neq j$, that should be added to the list of modifiers of this reaction. The corresponding reduction algorithm was implemented in Python.

The resulting decomposition may be useful even if QSS conditions are not rigorously satisfied. In this case, the hierarchical levels are autonomous only approximately, but the hierarchical optimization may still be better than the plain optimization.

Other sources of signal backpropagation are the reversible reactions connecting species from different levels of the hierarchy. A reversible reaction allows the propagation of the signal in both directions and establishes interaction graph connections in both directions between reactants and products. However, some reversible reaction effectively function in only one direction. We say that a reaction is *forward irreversible* if $R_+ >> R_-$ where R_+, R_- are the forward and backward reaction rates. This condition can be verified using numerical simulations or any information about the orders of magnitude of the kinetic constants and concentrations of reactants and products. When it is satisfied we can consider that the reaction is irreversible.

As an illustration, we tested by simulation the forward irreversibility in the signaling model b2AR-PKA (see Table 1). We found several forward irreversible reactions, but the reaction $CaMCa3 + Ca \rightleftharpoons CaMCa4$ is particularly important for the directionality of the signal propagation. By considering this reaction to be forward irreversible, a large block containing $AC1$,$AC2$ and CaM splits into two blocks, one of rank one containing CaM and the other of rank two containing $CaMCa4$ (see Figure 2). Indeed, the signal propagates from $CaMCa3$ to $CaMCa4$ and not backwards.

The HOSS Optimization framework

The HOSS software is designed to orchestrate complex, multi-level hierarchical optimizations. To do this it deploys numerous individual optimization steps, each of which fits a subset of a model to a number of individual experiments (Figure 3 A). HOSS works on signalling and other models which are subdivided into blocks, typically individual signalling pathways in a signalling network. The blocks are organized into a hierarchy informed by the above mathematical formalism, where each level depends only on signalling input coming from preceding levels, and blocks within a level are independent of each other (Figure 3 B). During operation, HOSS reads a configuration file in JSON format, which specifies the metadata and overall optimization parameters, such as optimization algorithm and tolerance (Figure 3 B). The configuration file further specifies a weighted set of experimental protocols defined in the FindSim JSON format [20]. Finally, within each block it identifies which parameters are to be adjusted, and optionally their bounds. HOSS calls the FindSim utility [20] to set the parameter vector, and to compute the objective (cost) function giving the accuracy of the model fit for each experiment. The default objective function is the normalized root-mean-square difference between experimental data and simulation readout (2). When several experiments pertaining to different readouts, or datasets of different origins are available for the same model, a consolidated objective function is obtained by combining individual objective functions scaled by weights (3). This consolidated objective function is used in the optimization algorithm which is provided by scipy.minimize. HOSS can employ nested parallelization by simultaneously running FindSim on each experiment within a block, and independently optimizing each block on different processes. For the purposes of subsequent discussion, we refer to the optimization routine (provided by scipy.optimize) as the optimization *algorithm*, and the hierarchical optimization program (provided by HOSS) as the HOSS *method*.

FindSim is the Framework for Integration of Neuronal Data and SIgnalling Models [20]. Briefly, it does three things: 1) reads a model and tweaks its parameters, 2) reads the definition of an experiment and runs it on the model, and 3) compares the output of the model with data from an experiment (Figure 3 D, E, F). FindSim is agnostic to model definition format and simulator. It currently works with the HillTau [4] format and simulator, and with ODE and mass action models specified in SBML and other formats, and solved using the MOOSE simulator [34]. FindSim utilizes a JSON format file to specify experiment inputs and readouts. Crucially, an experiment defined in FindSim format can be applied to completely different models even using different modelling formalisms, provided the input and output entities are common. We illustrate these capabilities below. In the context of HOSS, we use FindSim on four kinds of experiments applicable to cellular signalling: dose-response, time-series, bar-charts and direct parameter estimates (Figure 3 D, E, F). FindSim has additional capabilities to handle common electrophysiological experiments [35, 36] but these are not used in the current study.

Large Models overview

For the purposes of this report, we model two signalling pathways in two formalisms each (Figure 4 A-D). The pathways are the beta-adrenergic receptor activation of protein kinase A (the b2AR pathway) and the epidermal growth factor activation of MAPK/ERKII (the EGFR pathway). The reaction topologies of these pathways are based on and simplified from [2]. The two formalisms are HillTau [4], which is an abstracted reduced signalling model specification which maintains direct experimental mapping of selected parameters such as concentrations and rates; and well-mixed chemical kinetics specified in SBML or other compatible formats. In the current study, SBML models are based on ODE dynamics. However, in HillTau, species dynamics are not computed using ODEs, but by a hybrid system. The composition of the models is reported in Table 1.

Pathway	Formalism	Number of species	Number of reactions	Number of parameters
EGFR-MAPK	HillTau	14	7	29
b2AR-PKA	HillTau	21	12	37
EGFR-MAPK	ODE	36	22	54
b2AR-PKA	ODE	53	40	93

Table 1. Composition of large test models used in this study.

Experimental database

We have used manual curation of the experimental literature to build up a repository of over 350 signalling experiments with a focus on synaptic signalling pathways. There are two key characteristics of this dataset, which drives several of the design choices in HOSS. First, the number of experiments pertaining to each pathway is limited, and considerably below the number of parameters even for HillTau models (Figure 5 A, B). Second, there are frequently overlapping experiments which disagree on the quantitative values of readouts (Figure 5 C, D).

Results

Hierarchical optimization outperforms flat optimization for a paradigmatic model with synthetic datasets

In order to illustrate and test the hierarchical optimization method we first use a paradigmatic model of the MAPK signalling cascade, introduced by Huang and Ferrel [37]. The SBML model is available in the Biomodels [38] database. The corresponding ODE system can be found in ODEbase [39] database https://www.odebase.org/detail/1330. The original SBML model consists of mass-action elementary reactions. Because of multiple Michaelis-Menten mechanisms sharing the same enzyme there is back-propagation of the signal and the application of the hierarchical decomposition algorithm to this model results in only one autonomous pair that includes the entire model.

By applying the QSS reduction transformation, the 22 ODEs in the ODEbase model are simplified to 8 differential equations. Notably, 4 species exclusively function as enzymes, and are considered buffered after the transformation (MAPKKK activator, MAPKKK inactivator, MAPKKPase, MAPKPase). As shown in Figure 6 A, the reduced MAPK model lends itself to a hierarchical cascade with 3 levels.

We tested hierarchical optimization using time series produced in [40], consisting of 10 *in silico* experiments. Each experiment employed a different concentration of

MAPKKK activator. For the flat optimization we used 12 distinct starting points log-uniformly distributed in a hypercube with edges $[10^{-10}, 10]$, and for hierarchical optimization (with parameter scrambling) we used 12 starting points per level. Figure 6 B shows that flat optimization takes longer compared to hierarchical optimization in terms of total duration. Additionally, the hierarchical optimization outperforms classical optimization significantly in terms of the objective function value (Figure 6 C).

Black-box, non-gradient optimization methods work well for flat optimization.

To scale up our analysis to moderately large models, we utilized the HOSS pipeline on a set of four signalling pathways as described in Table 1. Notably all details required for execution of the optimization pipeline, such as applicable experiments (FindSim files in JSON format), experiment weights, parameter lists, and parameter bounds were incorporated into the HOSS files. Thus a single command triggers execution of a complex pipeline, and a single file orchestrates all the data, models, optimization options, and parameter specification. As a reference, we first ran the HOSS pipeline using flat (non-hierarchical) optimization on the models, employing a number of standard optimization methods in the scipy.minimize library (Figure 7 A). Our initial models were initially parameterized manually using inspection of a limited subset of experiments. Following the flat optimization, all of the algorithms produced better fitting models than the start models. This was reflected in the modest improvement in the model-fitting objective function, which we refer to as cost (Figure 7 B). We found that COBYLA (black-box, non-gradient algorithm based on linear programming and linearization of the problem and constraints) and SLSQP (iterative quadratic programming method, also using linearized constraints) were considerably faster to converge than gradient algorithms such as BFGS (quasi-Newton algorithm based on an approximated inverse Hessian matrix) (Figure 7 C). COBYLA was more reliable in producing small costs. A possible explanation for this effect is the conflict within the multiple datasets used in the weighted cost (3). This conflict may lead to ill-conditioned Hessians and degenerate quadratic approximations of the cost functions, which disadvantage the BFGS and SLSQP algorithms. Accordingly we used COBYLA for subsequent hierarchical optimization runs.

Hierarchical optimization is more efficient than flat optimization for biochemical models with real datasets

We next tested the HOSS pipeline for hierarchical optimization (Figure 8 A). We have shown above that nested hierarchical optimization is more efficient than flat optimization for fitting a medium-sized model with synthetic data. Our results here show that this efficiency carries over to complex real-world cases involving large models (Figure 4), large but incomplete datasets (Figure 5), and noisy and sometimes inconsistent data (Figure 5). We implemented hierarchical optimization in HOSS as schematized in (Figure 8 A).

The signalling reactions from Figure 4 were manually subdivided into individual pathways reflecting their biological organization. Within the HOSS configuration file for each model, the pathways were placed in a hierarchy which reflected their position in the signalling cascade (e.g., Figure 3 C, Figure 8 B). We again tested three different algorithms for optimization: BFGS, COBYLA and SLSQP. We found that hierarchical optimization worked for all algorithms, though COBYLA gave smaller costs than BFGS and SLSQP in most cases (Figure 8 B). The runtimes followed the same pattern as for

flat optimization, that is, BFGS > COBYLA > SLSQP. We then compared how hierarchical optimization performed compared to flat optimization (Figure 8 D E). HOSS gave smaller or comparable costs to flat optimization in all except the ODE-based EGFR model, labeled D4-EGFR. We speculate that a loop unrolling pass would improve the EGFR pathway cost, since there is a feedback loop in the EGFR pathway which violates the hierarchy assumptions. Notably, the runtime for hierarchical optimization was considerably faster in all cases.

Multistart methods yield lower optimization cost: initScram method.

As the basic HOSS algorithm may be susceptible to local minima, we implemented a version which generated a large number of initial models with parameters randomized in a log-normal distribution of half-width scramble Range (scramRange, defined as $b = 1/a = \text{scramRange} > 1$ in (5)) (Figure 9 A, B). This is a known approach, with roots in simulated annealing methods [26, 41, 42]. We extended the HOSS framework to overlay model parameter scrambling and process farming onto the hierarchical optimization method. This is an embarrassingly parallel problem and each of the optimization processes could run in parallel. In the course of these runs we identified one necessary refinement to the algorithm. In some cases, a subset of the initial models took an enormously long time to converge. Thus we implemented a timeout for each elementary minimization run. This may slightly reduce the number of completed runs, but frequently led to considerable improvement in runtime. In an analogy with simulated annealing, we asked if successive rounds of optimization would find still lower minima. We found that multiple rounds of optimization tended to converge rapidly (Figure 9 C). Hence in most cases a single optimization step should suffice.

The optimization costs resulting from a typical run with 200 initial models fell into a distribution which depended both on model and on scramRange (Figure 9 D, E). As expected, the width of the cost distribution increased with scramRange. The best fits were at the left of the distribution and in these examples were were obtained with a scramRange of ∼5.0, that is, log-normal random scaling from 1/5 to 5-fold of each initial parameter (Figure 9 D, E). The costs for these fits were considerably lower than those obtained with plain HOSS. To relate the NRMS divergence between parameters to scrambleRange, we generated a set of models at a series of scrambleRange values, and computed NRMS between each population (Figure 9 F). Interestingly, the best few models (lowest costs) were not necessarily very similar in their parameters. We did a normalized RMS comparison of parameters of the top 10 D4-b2AR models and found no obvious clusters (Figure 9 G). Using the relationship from (Figure 9 F), we observed that the NRMS range of ∼1.0, as seen in these best 10 models, corresponded to a scrambleRange of ∼2.0. This means that the parameters of these models differed by as much as a factor of two. As another measure of the parameter similarity of 'good' models, we plotted the distribution of (model parameter) / (mean parameter) across all parameters taken from the best 25% of models, that is, those whose costs were in the lowest quartile (Figure 9 H, I). We found that this clustered around one, suggesting that there is indeed a global optimum to which most models converge. Note that this parameter distribution is narrower with a broad tail, as compared to the source model parameter distribution from (Figure 9 B).

Multi-stage Monte-Carlo yields further improvements of the optimization cost: hossMC method.

As a final refinement of our code-base, we implemented a similar model-scrambling step within each stage of the HOSS algorithm (Figure 10 A). Thus, each subset of the model was subject to scrambling to give S variants (S∼200 for a full run). These S variants were individually optimized in an elementary minimization step similar to a single stage in the original HOSS method (Figure 8 A). If there were multiple model subsets within a given level of the HOSS hierarchy, each was subject to this process to give S optimized variants. The best of each subset were then recombined so as to obtain the top N solutions for a given level. Typical values for N were ∼10. These top N sub-models were then used as separate starting points for further scrambled models for the next level of HOSS, such that we again had S variants to optimize. After the program ran through all levels, we had a set of the best-fitting N models obtained by the overall pipeline. This method generated excellent fits to the data, slightly better than the previous multi-start method initScram (Figure 10 B). Wallclock time was similar to that of the initScram method provided there were enough CPU cores available to run all the steps in parallel (Figure 10 C). The total CPU time for both randomized methods was also quite similar (Figure 10 D).

To summarize the performance of the four methods employed here (flat, hoss, initScram and hossMC), we compared three metrics across the four optimization methods in the HOSS framework. The metrics were the final cost (Figure 10 B), wallclock time (Figure 10 C), and total CPU time (Figure 10 D). As detailed above, the hossMC method was most effective but most CPU-costly, followed closely both in time and model fitting cost by the initScram method. The plain HOSS method was uniformly the fastest, but its optimization costs did not compare well with the two multi-start methods (initScram and hossMC) for any of our models. The conventional flat method is not a good choice by any criterion.

Discussion and conclusion

We have developed a pipeline for hierarchically optimizing large signalling models with hundreds of parameters. We show that hierarchical optimization gives better model fits, and does so faster than conventional flat optimization. We extend this approach to two further methods which use Monte Carlo sampling of multiple parameter start points to give still better final models.

Model provenance and modelling disease variants

Complex biological models, and signalling models in particular, frequently draw upon diverse sources of data. Such models are often hand-tuned, and such tuning may be very effective because it draws upon expert intuition and implicit knowledge about the behaviour of familiar pathways. However, many model parameters are adopted from the literature without clearly documenting the parameter optimization procedures or the data used in these procedures. This makes model provenance problematic. How did the modeller end up with a particular set of parameters? The HOSS framework introduces model optimization pipelines that are efficient, scalable, repeatable and above all, transparent. The development of a well-structured optimization configuration format in HOSS ensures that all experimental data and model choices, their weights, and all hyperparameter selections are as clearly defined as the algorithms and the simulators. This emphasis on provenance is designed to place the HOSS framework in line with FAIR principles [43]. We highlight two use cases to illustrate how HOSS supports reuse.

First, model rederivation: A different scientist may feel that some of the original experiments should be considered more authoritative than others. This can be done simply by assigning a greater numerical weight to the selected experiments, rerunning the pipeline, and seeing what changes in the resultant optimized model. Similarly, a researcher could include some new experiments into the dataset against which the model is to be optimized. This simplicity of model derivation brings a more data-driven flavor to debates over model assumptions and how well they represent the known experimental literature. As HOSS is agnostic to model formalism, it follows that these comparisons could even extend over distinct models implemented with different formalisms (e.g., HillTau vs mass action chemistry). Although not yet implemented, the same principles may apply to optimizing qualitative models such as Boolean networks.

Second, The HOSS structure is highly effective for model specialization. A researcher may wish to make a family of models for different disease mutations, based on a dataset of readouts for experiments in a set of mutant animal or cell lines. Using the HOSS pipeline, it is straightforward to replace the original (wild-type) experiments with the respective mutant line experiments, rerun the optimization, and obtain disease-specific models. Thus the HOSS framework encourages best practice in developing complex models which can be easily reused.

Large models and large datasets

HOSS is scalable. This is in large part due to the efficiency of the hierarchical optimization core method we have described. Based on this, we have shown that even large models can be optimized quickly. Beyond this, HOSS organizes systems optimization problems in a modular manner which scales well with complex models and datasets. As a key part of this, HOSS organizes models into hierarchies, within which data, parameter choices, and multiple optimization stages of a pipeline can be triggered using a single command. Thus, once it is set up, a HOSS optimization run does not require many steps of inspection and tweaking by the investigator, and is simple to incrementally extend with new experiments and updated models. Rerunning a pipeline is trivial, and is limited only by computational resources. Several tools also provide model optimization (e.g., COPASI [41])

Model building is not limited just by resources and datasets, but also by how manageable is the organization of the dataset. The traditional way to associate model parameters with experiments is to provide citations (e.g., refs: DOQCS [44], BioModels [38], ODEbase [39]). This is neither complete, due to the previously mentioned lack of documentation, nor automated, because every iteration of the model would, in principle, require human intervention to produce or find new data, reorganize it and reparametrize the models. Several efforts have sought to reorganize experimental data into a standardized machine-readable format [45, 46], and HOSS uses the FindSim format to do so [20]. The organization of a HOSS pipeline lends itself to version control, since every component of the pipeline is a file in a standard location and standard format. Specifically, the HOSS configuration file is in JSON, the model definition files may be SBML or HillTau, and the experiment specification files are FindSim JSON files.

HOSS encourages the clear subdivision of models and experiments into groupings around individual signalling steps, such as the activation of a kinase by its immediate second messengers. This has implications for experimental design geared to tightly defining large signalling systems, because it lays out the kinds of experiments that are needed to achieve full coverage of all the reaction steps. Notably, we find that two kinds of experiments can greatly tighten parameters: local experiments that probe input-output properties of a given signalling step, and readouts of all key intermediates along a receptor-driven pathway to ensure that signal propagation remains intact.

Model degeneracy, granularity, and completeness

In cellular signaling models, it is now clear that many parameter combinations may yield the same input-output properties. The origins of this *degeneracy* could be epistemic and due to data incompleteness [47], but also biological; being a feature of cells themselves, that can perform the same biological function in multiple ways [48–50]. The HOSS framework may provide a useful tool to study such degeneracy. Most directly, the two Monte Carlo methods supported by HOSS (initScram and hossMC) generate multiple 'good' models, which can be tested for degeneracy (e.g., Figure 9). Because HOSS is agnostic to model detail, simulator, and model formalism, it also lends itself to asking how model granularity affects degeneracy. We have previously suggested that it is useful to develop a family of models at different resolution for any given signalling system [33, 51, 52]. HOSS is well equipped to facilitate this, as it can use the same experimental dataset for models at different detail. We demonstrate this in the model choices in this paper, since the D3 models using HillTau, and the D4 models using the MOOSE simulator [34], are parameterized using overlapping sets of experiments, separated only by the fact that some experiments in the D4 set depend on molecules that are not defined in the simpler D3 models. Model completeness, referring to how well a model incorporates all necessary details to accurately representing a system or phenomenon, is quite difficult to ascertain in biology as it is in all scientific fields confronting theory and experiments [53]. Several methods have attempted to explore model topology space along with parameters [7, 8, 54, 55], but HOSS supports a more pragmatic interpretation: Is a model complete enough to account for a given set of observations? It does so by trying a large number of possible parameter sets and seeing whether any of these initial conditions result in well-fitting models. A failure to do so suggests that the model topology may need to be reconsidered. We have previously illustrated the behavior of a series of models of activity-driven synaptic signalling at different levels of granularity, and show that more detailed models fit additional features of the response [4]. However, an overly detailed model can lead to overfitting if the data is not sufficiently rich. We suggest that multi-grain hierarchical approaches, including automated model granularity (level of detail) selection, may represent a future evolution of hierarchical optimization.

Code availability

Code locations:
HOSS https://github.com/BhallaLab/HOSS,
FindSim https://github.com/BhallaLab/FindSim,
HillTau https://github.com/BhallaLab/HillTau,
MOOSE https://github.com/BhallaLab/moose-core,
HiNetDecom https://github.com/Computational-Systems-Biology-LPHI/HiNetDecom.

Acknowledgments

This work has been supported by CEFIPRA grant 68T08-3 to OR and USB. NAV has been supported by the Department of Biotechnology, Government of India, under the fellowship No. DBT/2018/NCBS/998. USB and NCBS-TIFR receive the support of the Department of Atomic Energy, Government of India, under Project Identification No. RTI 4006. We thank Pawan Kumar for help with supervision of AT, and Clement Raspail for providing a new, improved version of the Michaelis-Menten reduction code.

Author contributions

NV developed the database of experiments, and provided initial models for the b2AR and EGFR pathways. AT developed the algorithms and wrote the first versions of codes for nested hierarchical composition and QSS reduction, analysed the MAPK example and benchmarked early versions of the hierarchical decomposition on Biomodels database. MG developed and implemented new algorithms for automatic hierarchical decomposition based on r-SCC and finalized the benchmarking. OR developed the mathematical framework, designed the research, and wrote the paper. USB designed the research, implemented the code for the HOSS framework, ran the simulations, and wrote the paper.

References

1. Kanehisa M, Goto S. KEGG: Kyoto encyclopedia of genes and genomes. Nucleic acids research. 2000;28(1):27–30.
2. Bhalla US, Iyengar R. Emergent properties of networks of biological signaling pathways. Science. 1999;283(5400):381–387.
3. Abou-Jaoudé W, Traynard P, Monteiro PT, Saez-Rodriguez J, Helikar T, Thieffry D, et al. Logical modeling and dynamical analysis of cellular networks. Frontiers in genetics. 2016;7:94.
4. Bhalla US. HillTau: A fast, compact abstraction for model reduction in biochemical signaling networks. PLoS Computational Biology. 2021;17(11):e1009621.
5. Gyori BM, Bachman JA. From knowledge to models: Automated modeling in systems and synthetic biology. Current Opinion in Systems Biology. 2021;28:100362. doi:10.1016/j.coisb.2021.100362.
6. Nyman E, Stein RR, Jing X, Wang W, Marks B, Zervantonakis IK, et al. Perturbation biology links temporal protein changes to drug responses in a melanoma cell line. PLoS computational biology. 2020;16(7):e1007909.
7. Büchel F, Rodriguez N, Swainston N, Wrzodek C, Czauderna T, Keller R, et al. Path2Models: large-scale generation of computational models from biochemical pathway maps. BMC Systems Biology. 2013;7(1):116. doi:10.1186/1752-0509-7-116.
8. Gyori BM, Bachman JA, Subramanian K, Muhlich JL, Galescu L, Sorger PK. From word models to executable models of signaling networks using automated assembly. Molecular Systems Biology. 2017;13(11):954. doi:10.15252/msb.20177651.
9. Tasaki S, Nagasaki M, Oyama M, Hata H, Ueno K, Yoshida R, et al. Modeling and estimation of dynamic EGFR pathway by data assimilation approach using time series proteomic data. Genome Informatics. 2006;17(2):226–238.
10. Quach M, Brunel N, d'Alché Buc F. Estimating parameters and hidden variables in non-linear state-space models based on ODEs for biological networks inference. Bioinformatics. 2007;23(23):3209–3216.
11. Banga JR. Optimization in computational systems biology. BMC systems biology. 2008;2(1):1–7.

12. Chou IC, Voit EO. Recent developments in parameter estimation and structure identification of biochemical and genomic systems. Mathematical biosciences. 2009;219(2):57–83.

13. Kravaris C, Hahn J, Chu Y. Advances and selected recent developments in state and parameter estimation. Computers & chemical engineering. 2013;51:111–123.

14. Nim TH, Luo L, Clément MV, White JK, Tucker-Kellogg L. Systematic parameter estimation in data-rich environments for cell signalling dynamics. Bioinformatics. 2013;29(8):1044–1051.

15. Rodriguez-Fernandez M, Rehberg M, Kremling A, Banga JR. Simultaneous model discrimination and parameter estimation in dynamic models of cellular systems. BMC systems biology. 2013;7:1–14.

16. Fröhlich F, Kaltenbacher B, Theis FJ, Hasenauer J. Scalable parameter estimation for genome-scale biochemical reaction networks. PLoS computational biology. 2017;13(1):e1005331.

17. Loskot P, Atitey K, Mihaylova L. Comprehensive review of models and methods for inferences in bio-chemical reaction networks. Frontiers in genetics. 2019; p. 549.

18. Gilman AG, Simon MI, Bourne HR, Harris BA, Long R, Ross EM, et al. Overview of the alliance for cellular signaling. Nature. 2002;20:703–706.

19. Nim TH, White JK, Tucker-Kellogg L. SPEDRE: a web server for estimating rate parameters for cell signaling dynamics in data-rich environments. Nucleic acids research. 2013;41(W1):W187–W191.

20. Viswan NA, HarshaRani GV, Stefan MI, Bhalla US. FindSim: a framework for integrating neuronal data and signaling models. Frontiers in neuroinformatics. 2018;12:38.

21. Koh G, Teong HFC, Clement MV, Hsu D, Thiagarajan P. A decompositional approach to parameter estimation in pathway modeling: a case study of the Akt and MAPK pathways and their crosstalk. Bioinformatics. 2006;22(14):e271–e280.

22. Anandalingam G, Friesz T. Hierarchical optimization: An introduction. Annals of Operations Research. 1992;34(1):1–11. doi:10.1007/BF02098169.

23. Liu R, Gao J, Zhang J, Meng D, Lin Z. Investigating Bi-Level Optimization for Learning and Vision From a Unified Perspective: A Survey and Beyond. IEEE Transactions on Pattern Analysis and Machine Intelligence. 2021;44:10045–10067.

24. Valeri JA, Soenksen LR, Collins KM, Ramesh P, Cai G, Powers R, et al. BioAutoMATED: An end-to-end automated machine learning tool for explanation and design of biological sequences. Cell Systems. 2023;14(6):525–542.

25. Loos C, Krause S, Hasenauer J. Hierarchical optimization for the efficient parametrization of ODE models. Bioinformatics. 2018;34(24):4266–4273. doi:10.1093/bioinformatics/bty514.

26. Villaverde AF, Fröhlich F, Weindl D, Hasenauer J, Banga JR. Benchmarking optimization methods for parameter estimation in large kinetic models. Bioinformatics. 2019;35(5):830–838.

27. Bloem R, Gabow HN, Somenzi F. An algorithm for strongly connected component analysis in n log n symbolic steps. In: International Conference on Formal Methods in Computer-Aided Design. Springer; 2000. p. 56–73.

28. Bhalla US, Ram PT, Iyengar R. MAP kinase phosphatase as a locus of flexibility in a mitogen-activated protein kinase signaling network. Science. 2002;297(5583):1018–1023.

29. Gupte M, Shankar P, Li J, Muthukrishnan S, Iftode L. Finding hierarchy in directed online social networks. In: Proceedings of the 20th international conference on World wide web; 2011. p. 557–566.

30. Tatti N. Faster way to agony: Discovering hierarchies in directed graphs. In: Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2014, Nancy, France, September 15-19, 2014. Proceedings, Part III 14. Springer; 2014. p. 163–178.

31. Ventura AC, Sepulchre JA, Merajver SD. A hidden feedback in signaling cascades is revealed. PLoS computational biology. 2008;4(3):e1000041.

32. Briggs GE, Haldane JBS. A note on the kinetics of enzyme action. Biochemical journal. 1925;19(2):338.

33. Radulescu O, Gorban AN, Zinovyev A, Noel V. Reduction of dynamical biochemical reactions networks in computational biology. Frontiers in Genetics. 2012;3:131. doi:10.3389/fgene.2012.00131.

34. Ray S, Bhalla US. PyMOOSE: interoperable scripting in Python for MOOSE. Frontiers in Neuroinformatics. 2008;0. doi:10.3389/neuro.11.006.2008.

35. Bhalla US, Bower JM. Exploring parameter space in detailed single neuron models: simulations of the mitral and granule cells of the olfactory bulb. Journal of neurophysiology. 1993;69(6):1948–1965.

36. Brown SA, Moraru II, Schaff JC, Loew LM. Virtual NEURON: a strategy for merged biochemical and electrophysiological modeling. Journal of computational neuroscience. 2011;31:385–400.

37. Huang CY, Ferrell JE. Ultrasensitivity in the mitogen-activated protein kinase cascade. Proceedings of the National Academy of Sciences. 1996;93(19):10078–10083. doi:10.1073/pnas.93.19.10078.

38. Malik-Sheriff RS, Glont M, Nguyen TVN, Tiwari K, Roberts MG, Xavier A, et al. BioModels — 15 years of sharing computational models in life science. Nucleic Acids Research. 2020;48(D1):D407–D415. doi:10.1093/nar/gkz1055.

39. Lüders C, Sturm T, Radulescu O. ODEbase: A Repository of ODE Systems for Systems Biology. Bioinformatics Advances. 2022;2(1). doi:10.1093/bioadv/vbac027.

40. Henriques D, Villaverde AF, Rocha M, Saez-Rodriguez J, Banga JR. Data-driven reverse engineering of signaling pathways using ensembles of dynamic models. PLOS Computational Biology. 2017;13(2):1–25. doi:10.1371/journal.pcbi.1005379.

41. Elsts A, Pentjuss A, Stalidzans E. SpaceScanner: COPASI wrapper for automated management of global stochastic optimization experiments. Bioinformatics. 2017;33(18):2966–2967.

42. Dai Z, Lai L. Differential simulated annealing: a robust and efficient global optimization algorithm for parameter estimation of biological networks. Molecular BioSystems. 2014;10(6):1385–1392.

43. Eriksson O, Bhalla US, Blackwell KT, Crook SM, Keller D, Kramer A, et al. Combining hypothesis-and data-driven neuroscience modeling in FAIR workflows. Elife. 2022;11:e69013.

44. Sivakumaran S, Hariharaputran S, Mishra J, Bhalla US. The Database of Quantitative Cellular Signaling: management and analysis of chemical kinetic models of signaling networks. Bioinformatics. 2003;19(3):408–415.

45. Novère NL, Finney A, Hucka M, Bhalla US, Campagne F, Collado-Vides J, et al. Minimum information requested in the annotation of biochemical models (MIRIAM). Nature biotechnology. 2005;23(12):1509–1515.

46. Saez-Rodriguez J, Goldsipe A, Muhlich J, Alexopoulos LG, Millard B, Lauffenburger DA, et al. Flexible informatics for linking experimental data to mathematical models via DataRail. Bioinformatics. 2008;24(6):840–847.

47. Hüllermeier E, Waegeman W. Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods. Machine Learning. 2021;110:457–506.

48. Ludwig MZ, Palsson A, Alekseeva E, Bergman CM, Nathan J, Kreitman M. Functional evolution of a cis-regulatory module. PLoS biology. 2005;3(4):e93.

49. Maleszka R, Mason PH, Barron AB. Epigenomics and the concept of degeneracy in biological systems. Briefings in Functional Genomics. 2014;13(3):191–202.

50. Prinz AA. Degeneracy rules! The Journal of Physiology. 2017;595(8):2409.

51. Radulescu O, Gorban AN, Zinovyev A, Lilienbaum A. Robust simplifications of multiscale biochemical networks. BMC Systems Biology. 2008;2(1):86. doi:10.1186/1752-0509-2-86.

52. Viswan NA, Bhalla US. Understanding molecular signaling cascades in neural disease using multi-resolution models. Current Opinion in Neurobiology. 2023;83:102808.

53. Hofman JM, Watts DJ, Athey S, Garip F, Griffiths TL, Kleinberg J, et al. Integrating explanation and prediction in computational social science. Nature. 2021;595(7866):181–188.

54. Flöttmann M, Schaber J, Hoops S, Klipp E, Mendes P. ModelMage: a tool for automatic model generation, selection and management. Genome Informatics. 2008;20:52–63.

55. Fröhlich F, Loos C, Hasenauer J. Scalable inference of ordinary differential equation models of biochemical processes. Gene regulatory networks: methods and protocols. 2019; p. 385–422.

56. Mukhin YV, Garnovsky EA, Ullian ME, Garnovskaya MN. Bradykinin B2 receptor activates extracellular signal-regulated protein kinase in mIMCD-3 cells via epidermal growth factor receptor transactivation. Journal of Pharmacology and Experimental Therapeutics. 2003;304(3):968–977.

57. Saito T, Okada S, Ohshima K, Yamada E, Sato M, Uehara Y, et al. Differential activation of epidermal growth factor (EGF) receptor downstream signaling pathways by betacellulin and EGF. Endocrinology. 2004;145(9):4232–4243.

58. Kholodenko BN, Demin OV, Moehren G, Hoek JB. Quantification of short term signaling by the epidermal growth factor receptor. Journal of Biological Chemistry. 1999;274(42):30169–30181.

59. Solberg R, Taskén K, Wen W, Coghlan VM, Meinkoth JL, Scott JD, et al. Human regulatory subunit RIβ of cAMP-dependent protein kinases: expression, holoenzyme formation and microinjection into living cells. Experimental cell research. 1994;214(2):595–605.

60. Wolter S, Golombek M, Seifert R. Differential activation of cAMP-and cGMP-dependent protein kinases by cyclic purine and pyrimidine nucleotides. Biochemical and biophysical research communications. 2011;415(4):563–566.

61. Hasler P, Moore JJ, Kammer GM. Human T lymphocyte cAMP-dependent protein kinase: subcellular distributions and activity ranges of type I and type II isozymes. The FASEB journal. 1992;6(9):2735–2741.

Figures

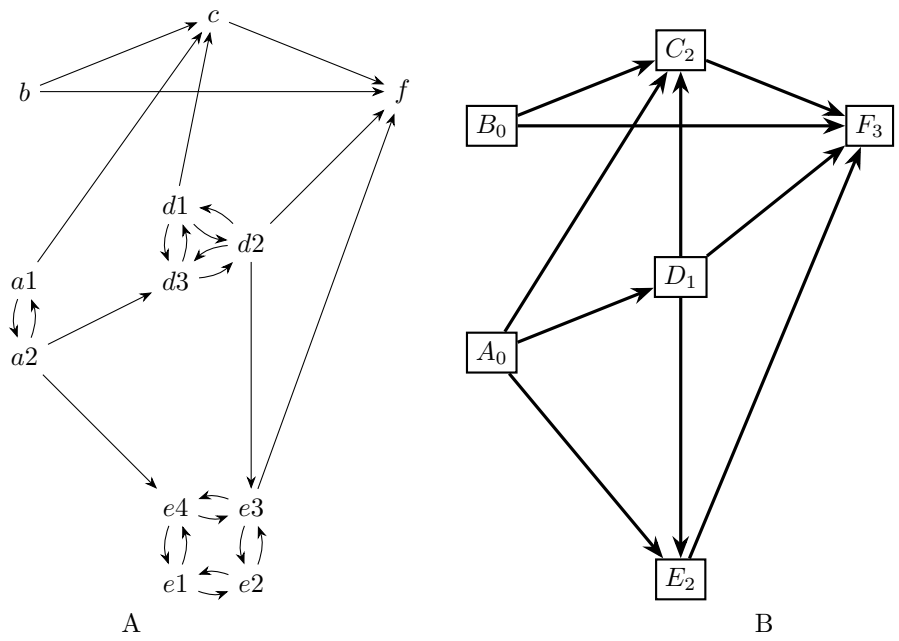

Fig 1. Interaction and quotient graphs used for the hierarchical decomposition. A) The interaction graph is a directed graph whose nodes are biochemical species. One source species acts on a target species if there is a reaction consuming or producing the target, whose rate depends on the concentration of the source. The strongly connected components (SCC) are maximal sets of nodes such that there are paths connecting each node to any other node. This graph has six SCCs: $A_0 = \{a_1, a_2\}$, $B_0 = \{b\}$, $D_1 = \{d_1, d_2, d_3\}$, $C_2 = \{c\}$, $E_2 = \{e_1, e_2, e_3, e_4\}$, $F_3 = \{f\}$. B) The quotient graph is an acyclic directed graph, whose vertices are the SCC of the interaction graph. Two SCC are connected in the quotient graph if there is a species in one connected to a species in the other. The hierarchy level of a block (SCC) is the length of the longest path in the quotient graph, connecting a root to the block. In this example, blocks A, B are roots and have level 0, block D has level 1, blocks C, E have level 2 and F have level 3.

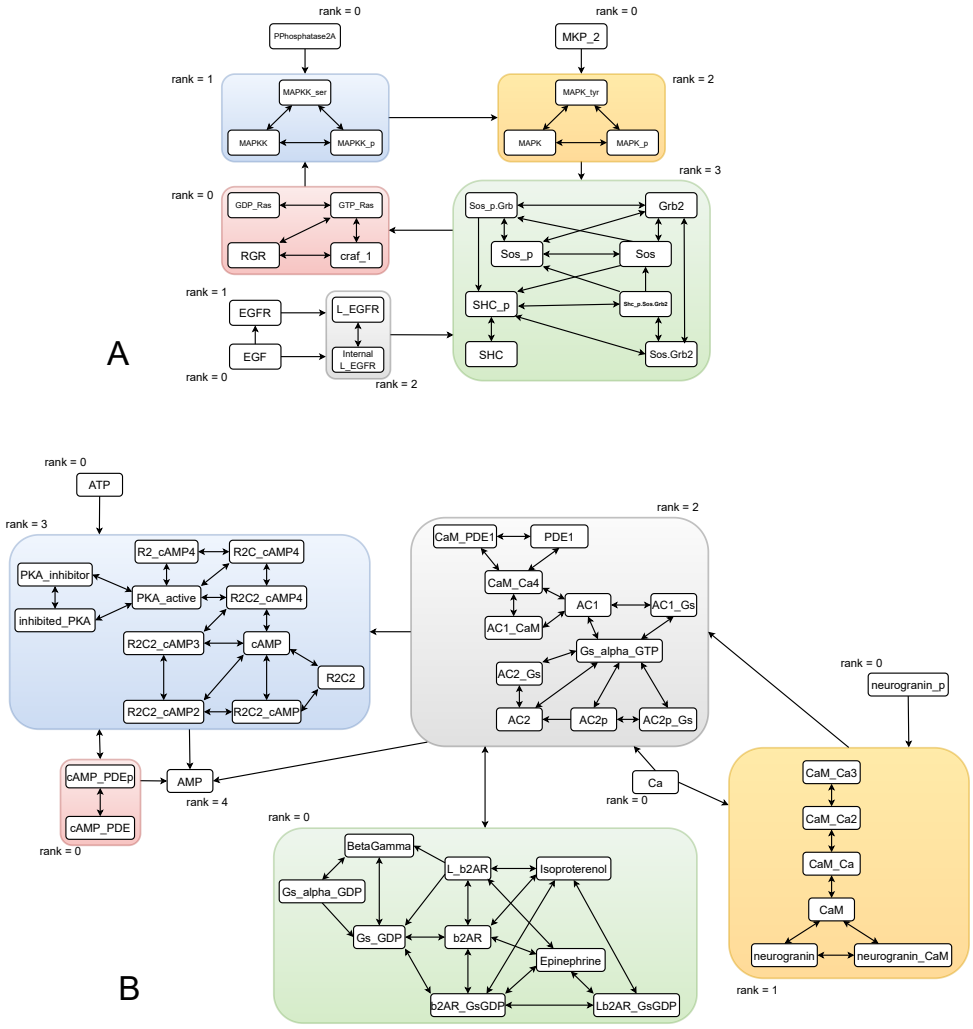

Fig 2. Blocks and quotient graph computed by automated hierarchical decomposition: r-blocks with for the EGFR model (A) and for the b2AR model (B) after Michaelis-Menten type reduction. One has the same blocks for $r = 1, 2$. In the model EGFR we have considered that the reaction $EGFR + EGF \rightleftharpoons LEGFR$ is forward irreversible. In the model b2AR we have considered that the reaction $CaMCa3 + Ca \rightleftharpoons CaMCa4$ is forward irreversible. The forward irreversibility conditions were verified by numerical simulations.

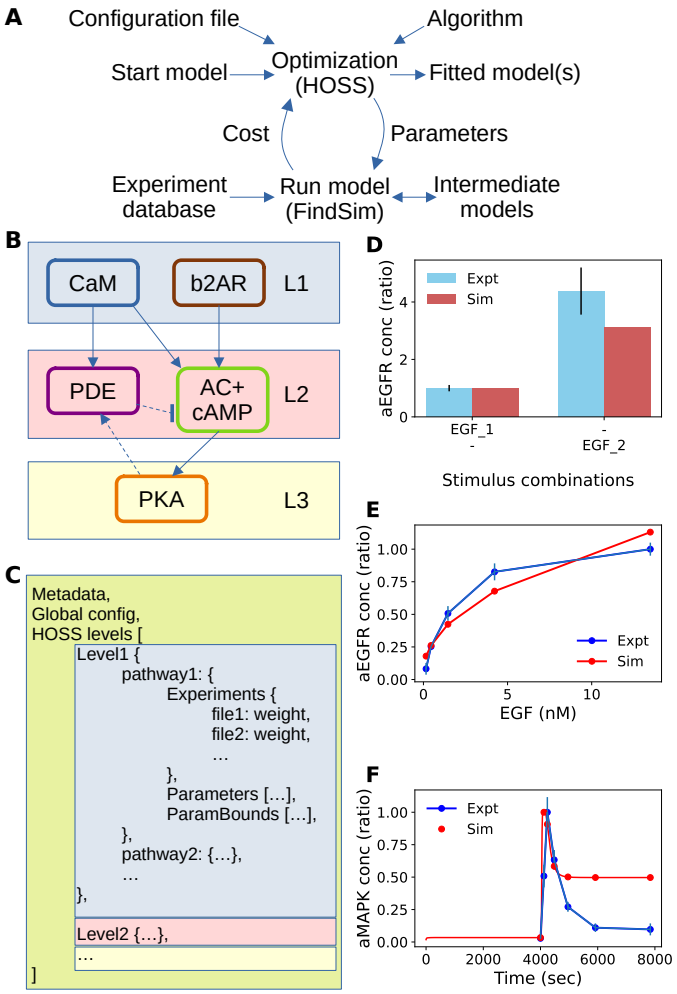

Fig 3. Optimization framework.
A: Schematic of optimization pipeline. HOSS follows a pipeline defined in a JSON configuration file to apply the specified algorithm at each stage of the optimization. HOSS orchestrates the operations of FindSim which takes a model, modifies its parameters, runs the model against specified experiments, and returns a cost representing the minimal distance between data and model predictions. This cost is used by the algorithm. B: Typical model decomposition into levels. L1 depends only on inputs, L2 depends only on L1 and inputs, and L3 depends on all upstream pathways. Within a level we can have multiple signalling blocks provided they do not depend on each other. However, we may have cross-interactions or feedback (arrows with dashed lines), which may require the pipeline to repeat one or more levels. C: pseudocode for definition of the HOSS pipeline. Within each level we can have multiple pathways, each of which needs a list of experiments, parameters and optionally parameter bounds. Colors map to corresponding levels of the model from panel B. D, E, F: Typical examples of experiments defined in FindSim format and run using FindSim to obtain optimization costs. In all these cases EGF is used as an input to the EGFR pathway. D: Bar chart. Here EGF is provided at baseline level (0.1 nM, named EGF_1) and at stimulus level (1.5625 nM, named EGF_2), and the resultant level of activated EGF receptor (aEGFR) is found. E: Dose-response. Here EGF is provided at a series of fixed input levels, and the steady-state levels of aEGFR are measured F: Time-series. Here a 7.8125 nM step stimulus of EGF is applied at t=4000s, and the level of activated MAPK is read out.

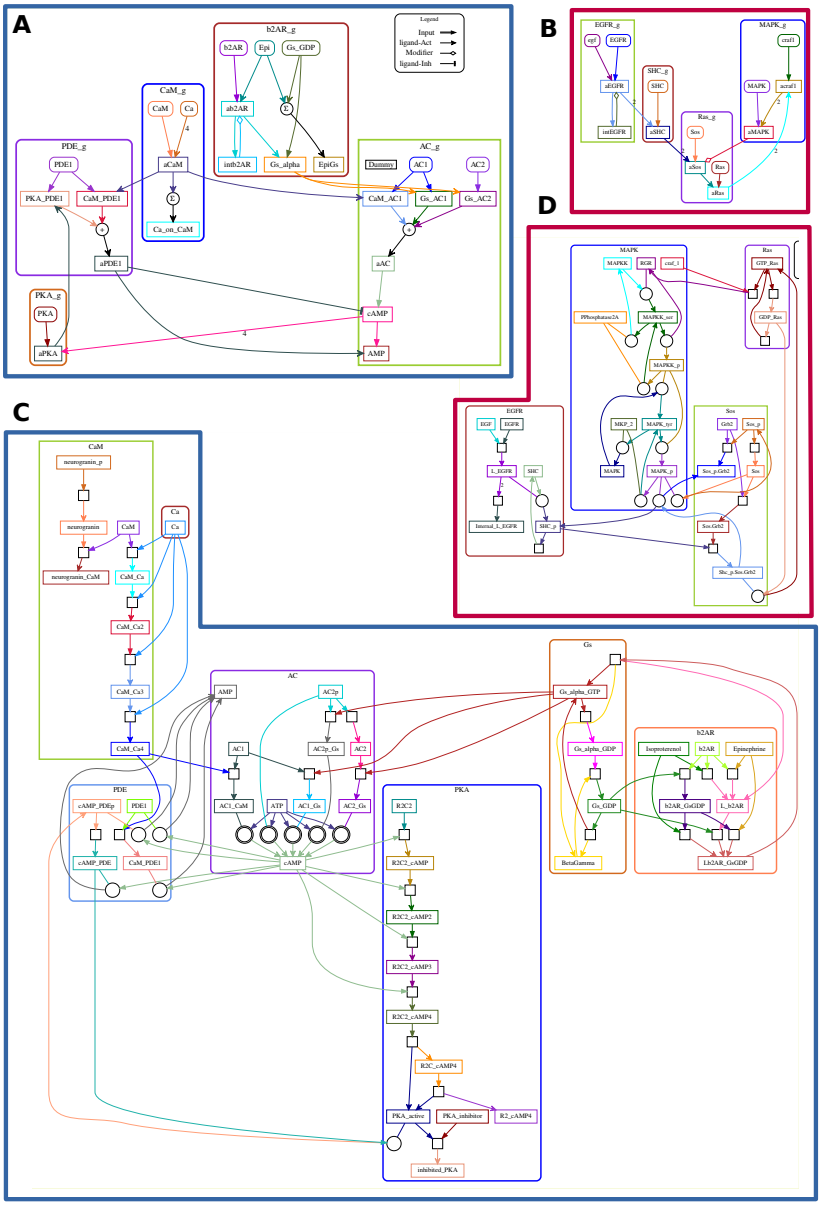

Fig 4. Models used in current study. A: Beta-2 adrenergic receptor pathway leading to Protein Kinase A activation (b2AR pathway), implemented in HillTau format. B: Epidermal growth factor receptor pathway leading to Mitogen-Activated Protein Kinase activation (EGFR pathway), implemented in HillTau format. C: b2AR pathway implemented in ODE format compatible with SBML. D: EGFR pathway implemented in ODE format. Note that the ODE format implementations are more chemically detailed, but retain overlap with HillTau implementations for several key readouts.

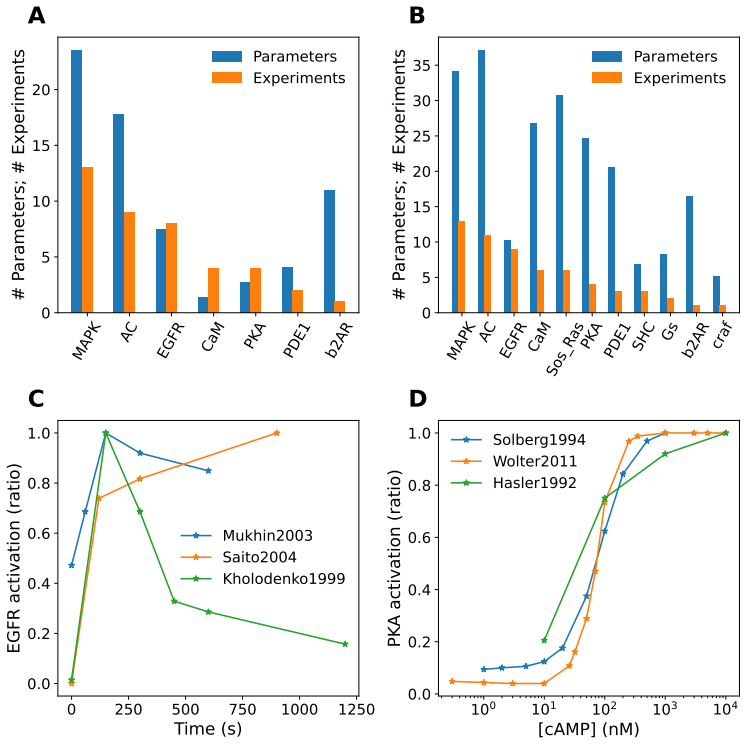

Fig 5. Features of experimental database. A, B: Number of parameters (blue) and number of experiments (orange) to constrain them, for different blocks in the model, sorted in order of decreasing number of experiments. In almost all cases the number of experiments falls well short of the number of parameters, that is, the model is underconstrained. A: Reduced (HillTau) models. B: ODE (SBML) models. C, D: Experiments may be inconsistent. C: Three time-series experiments for EGFR activation following a pulse of EGF, normalized to maximal response. These experiments were performed on different cell lines and not surprisingly, the time-courses differ [56–58]. D: Three dose-response experiments for PKA activation by cAMP. These experiments use purified preparations and despite somewhat different conditions the Kd is quite similar [59–61].

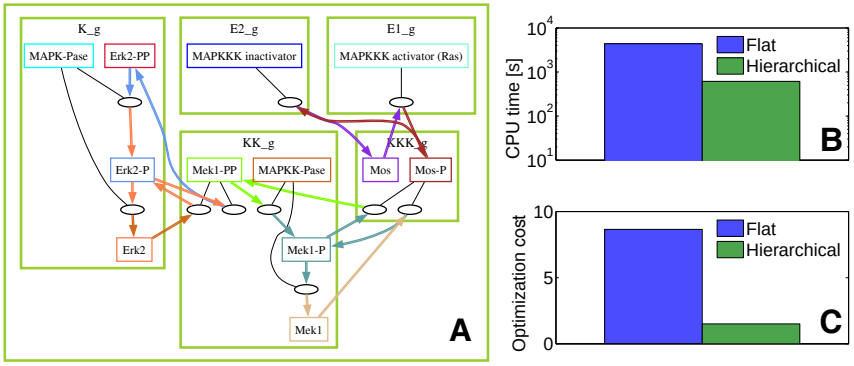

Fig 6. Optimization of the reduced MAPK model. A) Blocks in the reduced model; a three level nested hierarchy is defined as follows: level 1 (E1_g, E2_g, KKK_g), level2 (E1_g, E2_g, KKK_g, KK_g), level3 (E1_g, E2_g, KKK_g, KK_g, K_g). B,C) Performance of flat and hierarchical optimization.

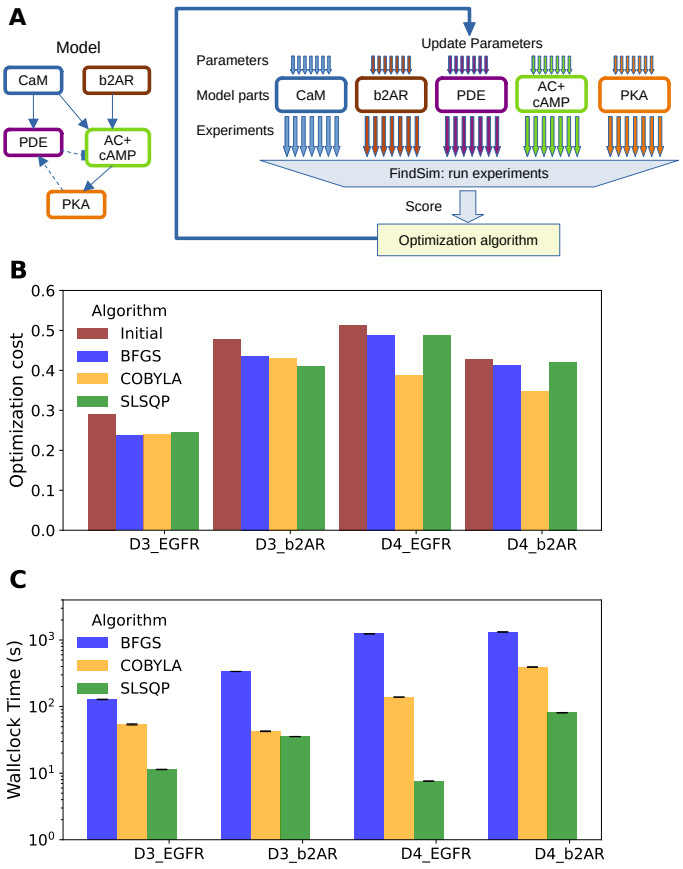

Fig 7. Flat optimization method. A: Schematic of flat optimization: Left: Model. Right: Method. All model subsets and all experiments are run in parallel. The resulting costs are combined into a single value used by the optimizer. The optimizer adjusts all parameters across model subsets, for each iteration. B: Barchart of costs for the four different models, comparing the initial cost with the final cost obtained using three different algorithms (BFGS, COBYLA, SLSQP) from the scipy.minimize library. BFGS is a gradient descent algorithm. Note that SLSQP sometimes does not converge to a low cost. C: Barchart of runtimes for the different algorithms. BFGS is always slower. Although SLSQP is typically the fastest algorithm, it sometimes produces high costs as seen in panel B.

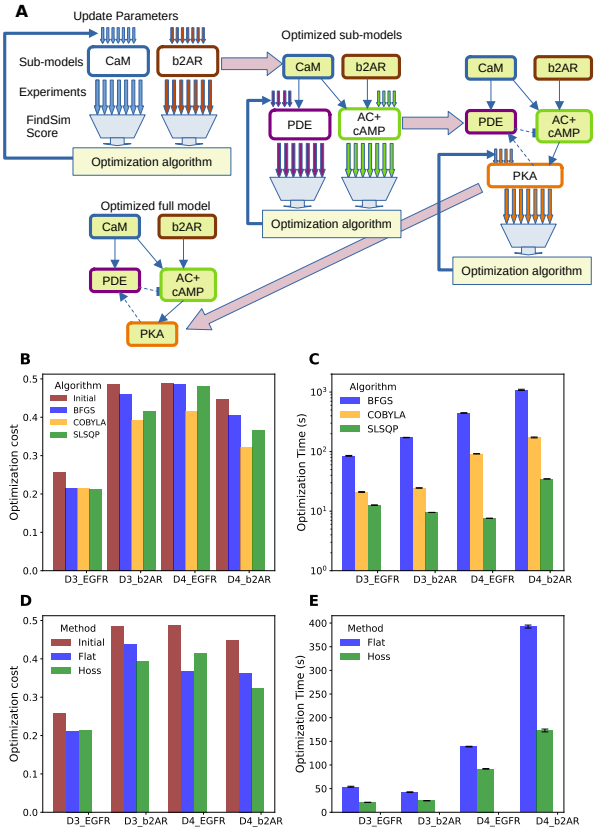

Fig 8. Hierarchical Optimization on large models. A: Schematic of hierarchical optimization as implemented in HOSS. First, the upper level of the model hierarchy is optimized, in this case the CaM and b2AR sub-models. Each is individually optimized, and any of the standard algorithms such as BFGS or COBYLA may be employed. Experiments specific to each sub-model are used to compute individual costs and independently update the sub-model parameters. Then, these sub-models are held fixed and the next level of the hierarchy is optimized (PDE and AC+cAMP sub-models). Finally, the lowest level of hierarchy (PKA) is optimized. With this the entire optimization is complete. B: Barchart of costs for the four different models, comparing the initial cost with the final cost obtained using three different algorithms from the scipy.minimize library. C: Barchart of runtimes for the different algorithms. As in the flat method, SLSQP is the fastest. D: Hierarchical optimization vs flat costs using COBYLA. With a single exception, HOSS gives lower or comparable costs. This exception is likely due to relaxation of hierarchy assumptions due to feedback. E: Timing of optimizations run using hierarchical optimization vs flat optimization timing using COBYLA. Hierarchical optimization is faster.

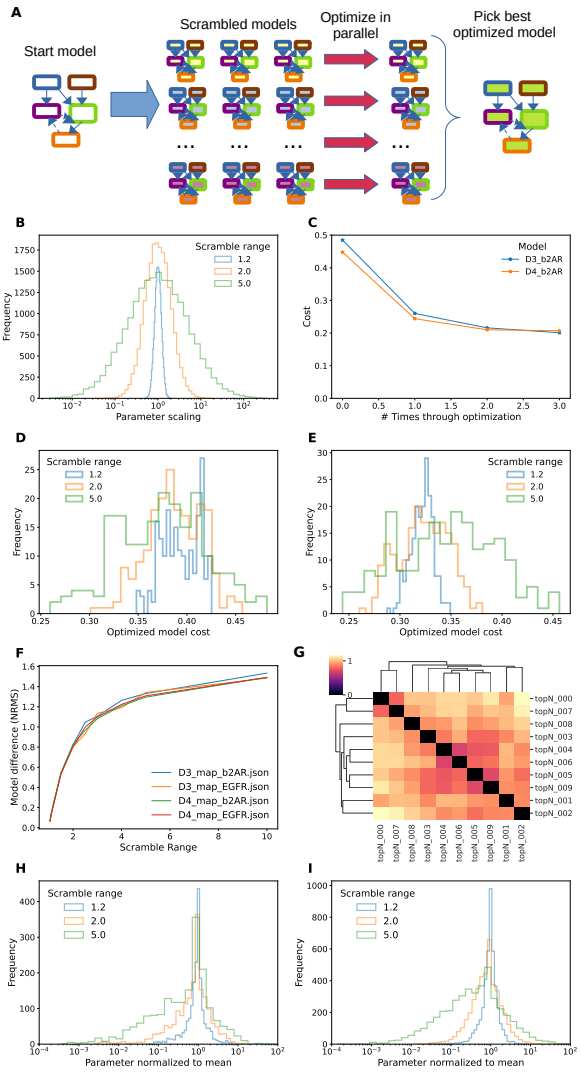

Fig 9. InitScram method. A. Schematic of method. B. log-normal distribution of parameter scaling from reference values for scrambleRange (SR) of 1.2, 2.0, and 5.0. C. Improvement of fit over successive optimizations. A second optimization produces a small improvement, and little improvement results from a third round. D. Cost distributions for three values of SR, D3 b2AR model. Note that the peaks are similar but the widths greater for larger SR, hence there are parameter sets with smaller costs (left tail of distribution) for large SR. E. Cost distributions for 3 values of SR, D4 b2AR model. Here the peaks of the cost distribution moves to the left with smaller SR. F. Mapping between parameter scrambling range and NRMS metric for similarity of models shows that this is independent of model. G. Model optimization cost cluster-map for top 10 optimized D4 b2AR models. H: Distribution of parameter scaling for optimized D3-b2AR models, normalized to mean of respective parameter for the best 10 models from that run. The optimized parameters converge very closely to the best 10 means. I: Distribution of parameter scaling for optimized D4-b2AR models. Here the tails of the distributions are somewhat wider, but there is still a narrow peak around 1.0 showing convergence from different start points. Note that peaks are narrower than the initial parameter ranges from panel B.

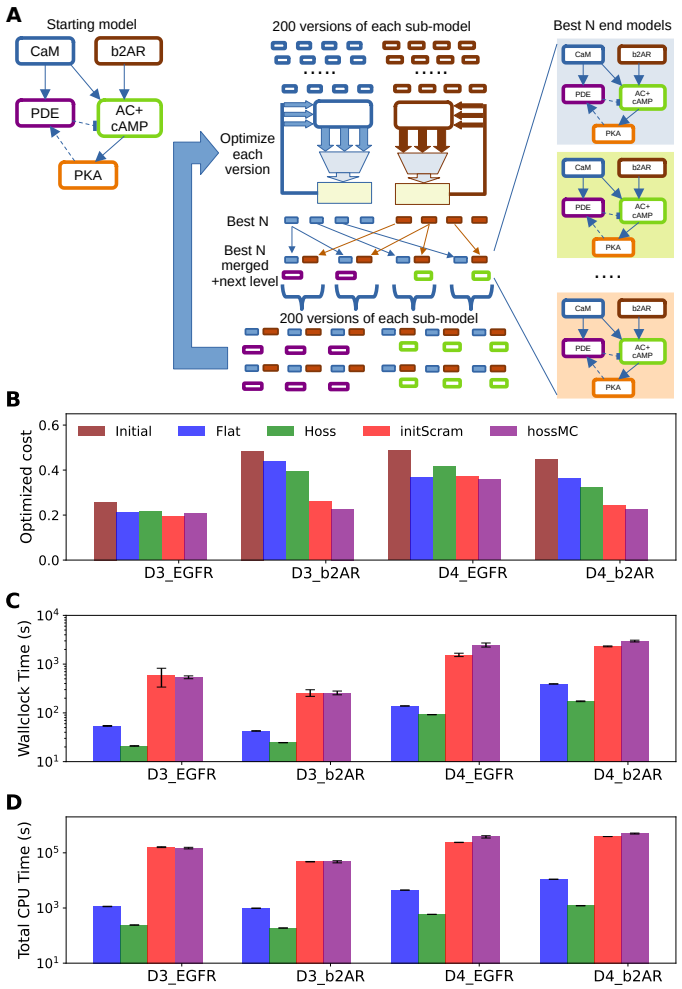

Fig 10. hossMC method A. Schematic of method. The model subsets in the first hierarchical level of the model are each scrambled 200 times, and each such starting point is optimized. The best N models (N=5) are taken from each sub-model and recombined to obtain the overall best N models for the first level. These are then merged with a sub-model from the next level, and these N models are then used as starting points for another round of model scrambling. The 200 scrambled models are again individually optimized as before, and the cycle repeats till we have optimized all levels. The best N merged models are provided as solutions. B, C, D: Comparing the 4 methods (flat, HOSS, initScram and hossMC). B: Optimization costs, including the initial cost for reference. The InitScram and hossMC methods worked the best. C. Wallclock time. The plain HOSS method was fastest. The two randomized methods initScram and hossMC were run on 24 processes, but still were much slower because they performed 200 repeats of all optimizations. D. Total CPU time. Here we factor in the number of processes and the parallelization of experiment cost estimation. By this metric, the HOSS method is substantially better than any other, and the two multistart methods are much more computationally costly.

Thesaurus

Autonomous pair: submodel consisting of subset of species and subset of reactions whose time evolution can be autonomously computed; if a species is in the subset of species, then also all the species casual to it are in.
Causality: a source species is causal to a target species if there is a path in the interaction graph from the source to the target.
r-causality: a source species is r-causal to a target species if the source is related to the target by an interaction graph path containing r arcs or less.
Optimization cost: distance between data and model predictions. *Synonyms*: **optimization score**.
Cost function: cost dependence on the model's parameters. *Synonyms*: **objective function, loss function.**
Block: strongly connected component of the interaction graph. *Synonyms*: **strongly connected component (SCC)**.
r-block: maximal subset of species such that any two species from it are r-causal one to another.
Evaluation: computing the cost for a model.
Flat optimization: minimizing the cost function non-hierarchically. *Synonyms*: **plain optimization**.
Hierarchical level: integer defining the position in the hierarchy. *Synonyms*: **rank**.
Hierarchical optimization: minimizing the cost function sequentially by starting with lower level parameters and proceeding to parameters having higher level in the hierarchy.
Interaction graph: directed graph whose nodes are the species, and a source and target species are connected by an arc if the production or consumption of the target depends on the source's concentration.
Quotient graph: acyclic directed graph whose vertices are blocks and two blocks are connected if one species in one is connected to a species in the other.
r-quotient graph: directed graph whose vertices are r-SCC and two r-SCC are connected if one species in one is connected to a species in the other. The r-quotient graph can contain cycles.
r-strongly connected component (r-SCC): union of non-disjoint r-blocks. *Synonyms*: **consolidated r-block**.