Scalable Nonlinear Programming Framework for Parameter Estimation in Dynamic Biological System Models

Sungho Shin[1], Ophelia Venturelli[1,2], Victor M. Zavala*[1],

1 Department of Chemical and Biological Engineering,
University of Wisconsin-Madison, Madison, Wisconsin, USA
2 Department of Biochemistry,
University of Wisconsin-Madison, Madison, Wisconsin, USA

* zavalatejeda@wisc.edu

Abstract

We present a nonlinear programming (NLP) framework for the scalable solution of parameter estimation problems that arise in dynamic modeling of biological systems. Such problems are computationally challenging because they often involve highly nonlinear and stiff differential equations as well as many experimental data sets and parameters. The proposed framework uses cutting-edge modeling and solution tools which are computationally efficient, robust, and easy-to-use. Specifically, our framework uses a time discretization approach that: i) avoids repetitive simulations of the dynamic model, ii) enables fully algebraic model implementations and computation of derivatives, and iii) enables the use of computationally efficient nonlinear interior point solvers that exploit sparse and structured linear algebra techniques. We demonstrate these capabilities by solving estimation problems for synthetic human gut microbiome community models. We show that an instance with 156 parameters, 144 differential equations, and 1,704 experimental data points can be solved in less than 3 minutes using our proposed framework (while an off-the-shelf simulation-based solution framework requires over 7 hours). We also create large instances to show that the proposed framework is scalable and can solve problems with up to 2,352 parameters, 2,304 differential equations, and 20,352 data points in less than 15 minutes. Competing methods reported in the computational biology literature cannot address problems of this level of complexity. The proposed framework is flexible, can be broadly applied to dynamic models of biological systems, and enables the implementation of sophisticated estimation techniques to quantify parameter uncertainty, to diagnose observability/uniqueness issues, to perform model selection, and to handle outliers.

Author summary

Constructing and validating dynamic models of biological systems spanning biomolecular networks to ecological systems is a challenging problem. Here we present a scalable computational framework to rapidly infer parameters in complex dynamic models of biological systems from large-scale experimental data. The framework was applied to infer parameters of a synthetic microbial community model from large-scale time series data. We also demonstrate that this framework can be used to analyze parameter uncertainty, to diagnose whether the experimental data are sufficient to

uniquely determine the parameters, to determine the model that best describes the data, and to infer parameters in the face of data outliers.

Introduction

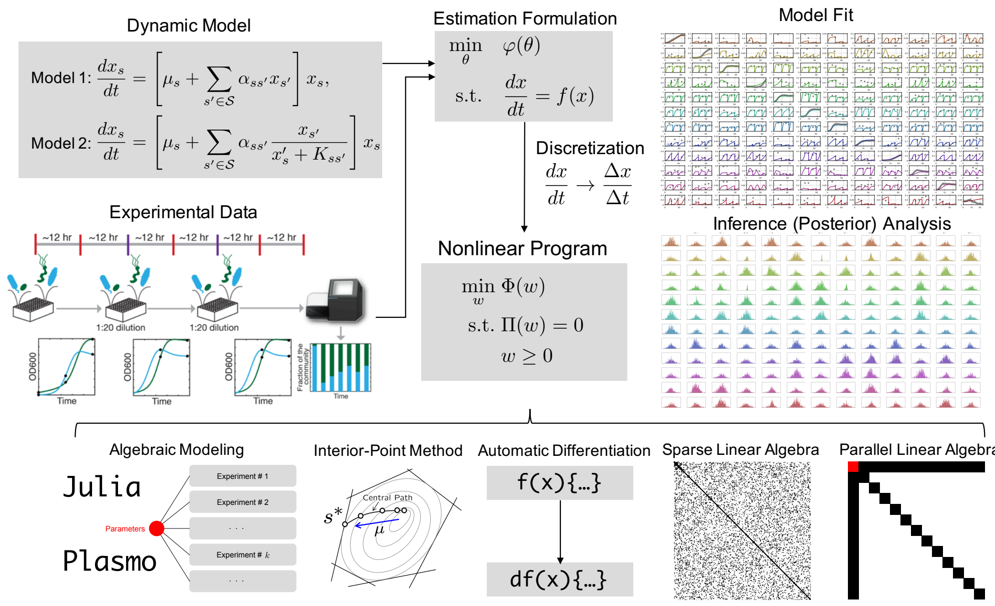
Dynamic Model
Model 1: $\frac{dx_s}{dt} = \left[\mu_s + \sum_{s' \in \mathcal{S}} \alpha_{ss'} x_{s'}\right] x_s,$
Model 2: $\frac{dx_s}{dt} = \left[\mu_s + \sum_{s' \in \mathcal{S}} \alpha_{ss'} \frac{x_{s'}}{x'_s + K_{ss'}}\right] x_s$
Experimental Data
~12 hr
1:20 dilution
OD600
Time
Fraction of the community
Estimation Formulation
$\min_\theta \ \varphi(\theta)$
s.t. $\frac{dx}{dt} = f(x)$
Discretization
$\frac{dx}{dt} \to \frac{\Delta x}{\Delta t}$
Nonlinear Program
$\min_w \ \Phi(w)$
s.t. $\Pi(w) = 0$
$w \geq 0$
Model Fit
Inference (Posterior) Analysis
Algebraic Modeling
Julia
Plasmo
Parameters
Experiment # 1
Experiment # 2
Experiment # k
Interior-Point Method
Central Path
$s*$
μ
Automatic Differentiation
f(x){…}
df(x){…}
Sparse Linear Algebra
Parallel Linear Algebra

Fig 1. Illustration of the proposed estimation framework. Mathematical models for biological systems are often expressed as systems of differential equations with parameters that need be estimated from experimental data. We formulate the estimation problem using a maximum a posteriori (MAP) formulation. This yields optimization problems constrained by differential equations that are transformed into fully algebraic nonlinear programs by using discretization schemes. The resulting NLPs can be easily implemented in algebraic modeling languages such as `JuMP` and `Plasmo.jl` that compute derivatives automatically and that are interfaced to powerful interior-point optimization solvers that exploit sparsity and structure to achieve high computational efficiency. The proposed framework is scalable, robust, easy-to-use, and flexible. These capabilities facilitate high-level tasks such as identification of parameter observability/uniqueness issues, model selection, and uncertainty quantification.

Dynamic modeling is essential for understanding the behavior of biological systems. Systems of interest in this domain include microbial communities and microbiome, gene regulatory networks, and metabolic pathways [1–3]. An important task that arises in modeling studies is validation against experimental data by using parameter estimation techniques. This task is computationally challenging because of the need to solve optimization problems constrained by differential equations. Challenges arise from the dimensionality, nonlinearity, and stiffness of the dynamic model, from the incomplete observation of the system states, from the need to estimate many parameters, and from the need to handle a large number of experimental data sets.

Extensive research on solution methods for estimation problems with differential equations has been reported in the computational biology literature (see [4,5] for comprehensive reviews). These methods target maximum likelihood estimation formulations (which are derived from Bayesian principles). In these formulations, one aims to find parameters that maximize the likelihood function. The most used strategy to handle such formulations is the so-called *simulation-based* approach. Here, the idea is to perform repetitive simulations of the dynamic model at different trial parameter values to identify a set of parameters that maximizes the likelihood. The trial parameter values are updated using derivative-based or derivative-free search schemes [6–8]. While the simulation-based approach is intuitive, repetitive solutions of large dynamic models is computationally expensive and differential equation solvers can fail at trial parameter values that are non-physical or that trigger unstable responses. In addition, techniques to compute first and second order derivatives for derivative-based schemes (e.g., finite differences, forward and adjoint sensitivities) involve intrusive procedures and are often limited to first-order derivatives [7]. The need for derivatives can be bypassed by using derivative-free search schemes [9, 10], which are widely popular in computational biology. Such methods include simulated annealing [11], genetic algorithms [12, 13], particle swarms [14], approximate Bayesian computation [15, 16], and various other methods [17, 18]. Derivative-free schemes do not scale well in the number of parameters (a larger number of trial parameter values often need to be explored compared to derivative-based schemes). Moreover, second order derivative information is needed to determine if the parameter estimates are unique/observable given available experimental data [19, 20]. The uniqueness/observability test of the parameter estimates is based on curvature information of the likelihood function at the solution.

Simulation-based estimation frameworks previously reported in the computational biology literature have focused on problems that usually contain less than 100 parameters [10, 15, 21]. To the best of our knowledge, the largest estimation problem solved using a simulation-based framework contains 3,780 data points and 1,801 parameters [7]. Such a problem was solved (to partial optimality) using a derivative-based search scheme that uses first-order derivative information (using an adjoint method) and required over 5 hours of computing time. The scalability limitations of simulation-based approaches present an important obstacle in considering models of higher fidelity, in exploiting high-throughput experimental data, in analyzing parameter uncertainties, and in implementing sophisticated techniques such as ensemble modeling.

In this work, we propose a nonlinear programming (NLP) framework for solving estimation problems with embedded dynamic models [22, 23]. The framework is based on a *direct transcription* approach wherein the dynamic model is converted into a large set of algebraic equations by applying time-discretization techniques. The algebraic equations are then embedded directly as constraints in the optimization problem (a nonlinear program-NLP). The NLPs arising from time discretization are of high dimension (easily reaching hundreds of thousands to millions of variables and constraints) but are also *sparse* and *structured*. Moreover, by transforming the dynamic model into algebraic equations, it becomes possible to use automatic differentiation techniques available in modern modeling languages to compute first and second derivatives. Exploitation of sparsity and structure, together with the availability of derivative information, enable the solution of estimation problems with complex dynamic models and efficient handling of many parameters and experimental data sets.

Discretization-based estimation approaches have been widely studied in diverse fields such as chemical engineering [24–26] and aerospace engineering [27–29] (see [30] for a comprehensive review) but less so in computational biology. A major factor that has hindered wider adoption is the lack of easy-to-use computational frameworks that facilitate access to non-expert users. In this work, we demonstrate that modern modeling and solution tools can be combined to create scalable, robust, easy-to-use, and flexible frameworks. We demonstrate the benefits by solving challenging estimation problems arising in nonlinear microbial community models.

The proposed framework enables the implementation of higher level tasks such as observability analysis and uncertainty quantification. Uncertainty quantification (UQ) seeks to characterize parameter posterior distribution, which is necessary to obtain confidence levels/regions and parameter correlation information. Conventionally, UQ is performed by using second order derivative (Hessian) information of the likelihood function to construct an approximate parameter posterior covariance matrix [23, 31] or by using a Markov-Chain-Monte-Carlo (MCMC) techniques [32, 33]. The Hessian-based approach is scalable but it requires intrusive computation (cannot be automatically computed by the solver) and does not capture well the effect of nonlinearities and physical constraints [31]. In MCMC, one samples parameters from the prior parameter density and compares the associated model outputs with experimental data to decide whether to accept that sample or not. By repeating these accept/reject steps one can construct an approximate parameter posterior. MCMC is rather easy to implement (it is not intrusive) but, being simulation-based, also suffers from potential failures of the differential equation solver at non-physical parameter samples, it does not scale well with the number of parameters, and convergence issues might be encountered.

In this work, we propose to overcome some of these challenges by using a randomized maximum a posteriori (rMAP) framework [34–37]. This method computes approximate samples from the parameter posterior distribution by performing random perturbations on the experimental data and by re-solving the estimation problem. This allows exploration of the parameter space more efficiently compared to the MCMC scheme because each sample can be computed in parallel (MCMC is sequential). Moreover, the rMAP approach is non-intrusive, can capture nonlinear and physical constraints effects, and avoids potential failures of differential equation solvers. The proposed estimation framework is flexible and can easily accommodate advanced estimation formulations. To demonstrate this, we implement formulations that use different prior regularization schemes and k-max norms (the mean of a specified fraction of largest values) to mitigate large outliers [38, 39].

The paper is organized as follows. The methods section provides a general form for the estimation problem under study and discusses how this can be cast as a sparse NLP by using time-discretization techniques. Furthermore, we introduce basic concepts behind NLP solvers that exploit sparsity and structures at the linear algebra level. In addition, we discuss rMAP and outlier mitigation schemes. In the results section, we demonstrate that the proposed framework can handle challenging estimation problems arising in microbial community models.

Methods

Estimation for Dynamic Models

We consider estimation problems of the following form:

$$\begin{aligned}
\min_{\theta} \quad & \varphi(\theta) + \sum_{k\in\mathcal{K}} \varphi_k(\bar{\eta}_k, \eta_k) && \text{(1a)}\\
\text{s.t.} \quad & \dot{x}_k(t) = f_k(x_k(t), \theta),\ k \in \mathcal{K},\ t \in [0, T_k] && \text{(1b)}\\
& x_k(0) = x_k^0, \quad k \in \mathcal{K} && \text{(1c)}\\
& \bar{\eta}_k(t) = \phi_k(x_k(t), \theta), \quad k \in \mathcal{K},\ t \in \mathcal{T}_k && \text{(1d)}\\
& 0 \le h_k(x_k(t), \theta), \quad k \in \mathcal{K},\ t \in \mathcal{T}_k. && \text{(1e)}
\end{aligned}$$

Here, $\mathcal{K} := \{0, 1 \ldots K\}$ is the *set of experiments* and $\mathcal{T}_k := \{t_1, t_2, \cdots, t_{n_k}\}$ is the set of measurement (sampling) times in experiment $k \in \mathcal{K}$. Time is denoted as $t \in [0, T_k]$, where $T_k \in \mathbb{R}_+$ is the duration for experiment $k \in \mathcal{K}$. The variable vector

$x_k : \mathbb{R} \to \mathbb{R}^{n_{x_k}}$ are the differential *state* time trajectories, $\theta \in \mathbb{R}^{n_\theta}$ are the model *parameters*, $\bar{\eta}_k(t) \in \mathbb{R}^{n_{\eta_k}}$ are the model predicted *outputs* with corresponding *experimental observations* $\eta_k(t) \in \mathbb{R}^{n_{\eta_k}}$ at time $t \in \mathcal{T}_k$ and experiment $k \in \mathcal{K}$, and $x_k^0 \in \mathbb{R}^{n_k}$ are initial conditions for experiment $k \in \mathcal{K}$. For convenience in the notation, we define the output vectors $\bar{\eta}_k = (\bar{\eta}_k(t_1), \cdots, \bar{\eta}_k(t_{n_k}))$ and the experimental output vectors $\eta_k = (\eta_k(t_1), \cdots, \eta_k(t_{n_k}))$ for experiment $k \in \mathcal{K}$ as well as the total output vector $\bar{\eta} = (\bar{\eta}_1, \cdots, \bar{\eta}_K)$ and the total experimental output vector $\eta = (\eta_1, \cdots, \eta_K)$. The vector function $f_k(\cdot)$ denotes the dynamic model mapping, $\varphi(\cdot)$ and $\varphi_k(\cdot)$ are objective function mappings, $\phi_k(\cdot)$ is the state-to-output mapping, and $h_k(\cdot)$ is the constraint mapping. All the mappings are assumed to be at least twice continuously differentiable with respect to all the arguments. The estimation formulation (1a) captures all the features that we need to demonstrate the capabilities of our proposed framework. Our framework, however, can also accommodate more general features; for instance, the initial conditions (1c) can be also considered as unknown variables that need to be estimated and we can define non-additive objective functions that penalize large errors.

Problem (1) can be derived from Bayesian principles. To see this and introduce some useful notation, we start from Bayes theorem:

$$p(\theta \,|\, \eta) = \frac{p(\eta|\theta)p(\theta)}{p(\eta)}. \tag{2}$$

Here, $p(\theta \,|\, \eta)$ is the parameter posterior density (i.e., the parameter density given knowledge on the outputs), $p(\eta \,|\, \theta)$ is the output posterior density (i.e., the outputs given knowledge on the parameters), $p(\theta)$ is the prior density (i.e., parameter density before knowledge of the output), and $p(\eta)$ is the output marginal density. In a maximum a posteriori (MAP) formulation, the goal is to find the parameters that maximize $p(\theta \,|\, \eta)$. Because $p(\eta)$ does not depend on θ, this can also be achieved by maximizing $p(\eta|\theta)p(\theta)$. Maximizing $p(\eta|\theta)p(\theta)$ is equivalent to maximizing the log-likelihood function $L(\theta) = \log(p(\theta)) + \log p(\eta|\theta)$. If the outputs from the experiments $k \in \mathcal{K}$ are independent (which is usually the case), we have that $p(\eta|\theta) = \prod_{k\in\mathcal{K}} p(\eta_k \,|\, \theta)$ and thus:

$$L(\theta) = \log p(\theta) + \sum_{k\in\mathcal{K}} \log p(\eta_k|\theta). \tag{3}$$

The observed outputs are random variables that are usually considered to be Gaussian and thus $\eta_k|\theta \sim \mathcal{N}(\bar{\eta}_k, \Sigma_k)$, where Σ_k is the covariance matrix. The prior density $p(\theta)$ is also often assumed to be Gaussian and thus $\theta \sim \mathcal{N}(\bar{\theta}, \Sigma_\theta)$, where $\bar{\theta}$ is the mean of the prior distribution and Σ_θ is its covariance. With this, we obtain:

$$-\log p(\theta) = \frac{n_\theta}{2}\log 2\pi + \frac{1}{2}\log\det\Sigma_\theta + \frac{1}{2}(\theta - \bar{\theta})^T\Sigma_\theta^{-1}(\theta - \bar{\theta}) \tag{4a}$$

$$-\log p(\eta_k|\theta) = \frac{n_{\eta_k}}{2}\log 2\pi + \frac{1}{2}\log\det\Sigma_k + \frac{1}{2}(\eta_k - \bar{\eta}_k)^T\Sigma_k^{-1}(\eta_k - \bar{\eta}_k). \tag{4b}$$

By comparing (3) with (1a) we can see that minimizing $\varphi(\theta) + \sum_{k\in\mathcal{K}} \varphi_k(\eta_k, \bar{\eta}_k)$ is equivalent to minimizing $-L(\theta)$. Here, the dynamic model together with the state-to-output mapping defines a *parameter-to-output mapping* of the form $\bar{\eta}_k := m_k(\theta)$. In a simulation-based estimation approach, the mapping $m_k(\theta)$ is computed by simulating the dynamic model (1b)-(1c) at a trial value θ using a differential equation solver and by evaluating the outputs at the sampling times $t \in \mathcal{T}_k$ using (1d). In a discretization-based approach, the mapping $m_k(\theta)$ is not computed explicitly (but we use it here as a mathematical representation that is used to explain some relevant concepts). Constraints (1e) restrict the parameter space to be explored.

After dropping all constant terms in the likelihood function we obtain:

$$\varphi(\theta) = \frac{1}{2}(\theta - \bar{\theta})^T \Sigma_\theta^{-1} (\theta - \bar{\theta}) \tag{5a}$$

$$\varphi_k(\bar{\eta}_k, \eta_k) = \frac{1}{2}(\eta_k - \bar{\eta}_k)^T \Sigma_k^{-1} (\eta_k - \bar{\eta}_k). \tag{5b}$$

The function $\varphi(\theta)$ is usually known as the *prior* term and provides a *regularization* effect that stabilizes the solution of the estimation problem when the parameters cannot be uniquely infered from the available data [40–43]. This regularization term arises from the prior density $p(\theta)$ and provides a mechanism to encode knowledge on the parameters. Assuming that the prior density is Gaussian gives rise to a prior term that is defined by a weighted L2 norm. Recently, the machine learning community has also proposed the use of regularization terms that use L1 norms (e.g., $\varphi(\theta) = \|\theta - \bar{\theta}\|_1$). The L1 norm induces sparsity in the parameters and corresponds to assuming that the prior density is Laplacian. One can also show that an L1 norm acts as an exact penalty function and implicitly induces constraints on the parameters. Similarly, one can also use the inequality constraints $h_k(\cdot)$ to directly embed physical knowledge in the MAP formulation (e.g., concentrations can only be positive).

From (5) we see that $\varphi_k(\cdot)$ are squared error terms and thus the MAP problem minimizes the *sum* of the squared errors across all experiments $k \in \mathcal{K}$. This approach offers limited control on large errors that might result from data outliers. Here, we propose to use a k-max norm to mitigate these issues. Our proposal is based on the observation that a k-max norm is equivalent to a conditional-value-at-risk (CVaR) norm [38, 39, 44]. The CVaR$_\beta$ norm of a vector $\mathbf{e} = (e_1, \cdots, e_K)$ with components $e_k = \varphi_k(\bar{\eta}_k, \eta_k)$ is defined as the average of the β-fraction of largest elements of the vector (where $\beta \in [0, 1]$ is a parameter that defines the size of the fraction) [44]. One can show that, when $\beta \to 1$, the CVaR norm is the largest fitting error and, when $\beta \to 0$, the CVaR norm is the sum of fitting errors (as in the standard MAP formulation). A key computational property of the CVaR norm is that it can be formulated as an standard optimization problem. In particular, the MAP problem with a CVaR error norm can be expressed as:

$$\min_{\theta,\gamma} \quad \varphi(\theta) + K\left(\gamma + \frac{1}{(1-\beta)}\sum_{k=1}^{K}[e_k - \gamma]_+\right) \tag{6a}$$

$$\text{s.t.} \quad (1b) - (1e) \tag{6b}$$

where $[\cdot]^+ = \max(0, \cdot)$ is the max function and γ is an auxiliary variable [39].

Nonlinear Programming Formulation

To solve the MAP problem (1) we approximate the differential equations by using a discretization scheme. This enables the use of computationally efficient NLP solvers and facilitates high-level UQ and observability monitoring tasks.

Time Discretization

Discretization schemes such as Euler, Runge-Kutta, and orthogonal collocation are commonly used to transform differential equations into algebraic ones. Orthogonal collocation is often preferred because accurate approximations can be obtained with few discretization points [22]. To simplify the presentation we use an implicit Euler scheme, which can be shown to be a special type of an orthogonal collocation scheme (i.e., it is a one-point Radau collocation scheme). We discretize the time domain $[0, T_k]$ into a set of intervals with fixed discrete-time points $\{t_0, t_1, \cdots, t_{N_k}\}$ for each experiment $k \in \mathcal{K}$ (where $t_0 = 0$ and $t_{N_k} = T_k$). The associated index set is represented by $\mathcal{N}_k := \{1, \cdots, N_k\}$. By applying an implicit Euler scheme, the dynamic model (1b) is converted into a set of nonlinear algebraic equations of the form:

$$x_k(t_{j+1}) = x_k(t_j) + (t_{j+1} - t_j) f_k(x_k(t_{j+1}), \theta), \; k \in \mathcal{K}, \, j \in \mathcal{N}_k \tag{7a}$$
$$x_k(0) = x_k^0, \; k \in \mathcal{K}. \tag{7b}$$

With this, we can approximate the MAP problem (1) using the NLP:

$$\min_{\theta, x, \bar{\eta}} \quad \varphi(\theta) + \sum_{k \in \mathcal{K}} \varphi_k(\bar{\eta}_k, \eta_k) \tag{8a}$$
$$\text{s.t.} \quad x_k^{j+1} = x_k^j + (t_{j+1} - t_j)\, f_k(x_k^{j+1}, \theta), \; k \in \mathcal{K}, \, j \in \mathcal{N}_k \tag{8b}$$
$$x_k(0) = x_k^0, \quad k \in \mathcal{K} \tag{8c}$$
$$\bar{\eta}_k(t) = \phi_k(x_k(t), \theta), \; k \in \mathcal{K}, \, t \in \mathcal{T}_k \tag{8d}$$
$$0 \leq h_k(x_k^j, \theta), \; k \in \mathcal{K}, \, j \in \mathcal{N}_k. \tag{8e}$$

Here, we use $x_k^j = x_k(t_j)$ as short-hand notation to represent states at time t_j and experiment k. For convenience, we express the NLP in the following abstract form:

$$\min_{w} \Phi(w) \tag{9a}$$
$$\text{s.t.} \; \Pi(w) = 0 \tag{9b}$$
$$w \geq 0. \tag{9c}$$

where $w \in \mathbb{R}^n$ is a large-dimensional vector containing all the discrete-time states x_k^j, parameters θ, and additional auxiliary variables. The mapping $\Phi : \mathbb{R}^n \to \mathbb{R}$ is the objective function and $\Pi : \mathbb{R}^n \to \mathbb{R}^m$ are equality constraints that contain algebraic equations obtained fro discretization of the dynamic model and other auxiliary equations. General inequality constraints can be transformed into equality constraints and simple non-negativity bounds by using auxiliary slack variables (i.e., $0 \leq h_k(x_k^j, \theta)$ can be written as $s_k = h_k(x_k^j, \theta)$ with $s_k \geq 0$).

A useful representation of the NLP results from noticing that the parameters θ are the only complicating (coupling) variables across experiments $k \in \mathcal{K}$. Consequently, we can express the NLP in the *structured* from [45]:

$$\min_{\theta, w_0 \dots w_K} \quad \Phi(\theta) + \sum_{k \in \mathcal{K}} \Phi_k(w_k, \theta) \tag{10a}$$
$$\text{s.t.} \; \Pi_k(w_k, \theta) = 0, \quad k \in \mathcal{K} \tag{10b}$$
$$w_k \geq 0, \quad k \in \mathcal{K}. \tag{10c}$$

Here, the variable vector w_k contains all the discrete-time states and auxiliary variables of experiment $k \in \mathcal{K}$, $\Phi(\cdot)$ is the prior term, $\Phi_k(\cdot)$ is the contribution of experiment $k \in \mathcal{K}$ to the likelihood function, and $\Pi_k(\cdot)$ contains the discretized dynamic model equations and auxiliary equations for experiment $k \in \mathcal{K}$. As we discuss next, this representation can be used to derive parallel solution approaches.

Interior-Point Solvers

The NLPs that result from time discretization exhibit a high degree of *algebraic sparsity* (only a few variables appear in each constraint) and are highly structured. Sparsity and structure permeates down to linear algebra operations performed inside the optimization solver. This is sharp contrast to the simulation-based approach, which induces dense linear algebra operations in the space of the parameters θ. Most modern large-scale NLP solvers such as `Ipopt` and `Knitro` seek to exploit sparsity and structure at the linear algebra level to achieve high computational efficiency [46, 47]. Interior point solvers, in particular, provide a flexible framework to do this. These solvers replace the variable bounds by using a logarithmic barrier function. In the context of NLP (9), this results in a logarithmic barrier subproblem of the form:

$$\min_{w} \; \Phi(w) - \mu \sum_{i=1}^{n} \log w^{(i)} \tag{11a}$$

$$\text{s.t. } \Pi(w) = 0 \tag{11b}$$

where $\mu \in \mathbb{R}_+$ is the so-called barrier parameter. The logarithmic term becomes large as $w^{(i)}$ approaches the boundary of the feasible region. This ensures that variables remain in the *interior* of the feasible region (hence the origin of the term *barrier*). A key observation is that one can recover a solution of the original NLP (9) by solving a sequence of barrier problems for decreasing values of μ [48]. An important property of interior-point methods is that the original NLP with bounds is converted into a sequence of NLPs with equality constraints. This removes the combinatorial complexity of identifying the set of bounds that are active or inactive at the solution (a bottleneck in active-set solvers).

Sparse Linear Algebra

Interior-point methods enable efficient linear algebra implementations. To explain how this is done, we note that the optimality conditions of the barrier problem are given by the following set of nonlinear equations:

$$\nabla_w \Phi(w) + \nabla_w \Pi(w)^T \lambda - \nu = 0 \tag{12a}$$

$$\Pi(w) = 0 \tag{12b}$$

$$V\,W\mathbf{1} = \mu, \tag{12c}$$

where, $\lambda \in \mathbb{R}^m$ are the Lagrange multipliers of the equality constraints, ν are the Lagrange multipliers of the bound constraints, $V = \text{diag}(\nu)$ and $W = \text{diag}(w)$ are diagonal matrices, and $\mathbf{1}$ is a vector of all ones.

By applying Newton's method to (12), we obtain the following linear algebra system:

$$\underbrace{\begin{bmatrix} H(w^\ell, \lambda^\ell) + \kappa_w \mathbb{I} & \nabla_w \Pi(w^\ell) \\ \nabla_w \Pi(w^\ell) & \end{bmatrix}}_{M^\ell(\kappa_w)} \begin{bmatrix} \Delta w^\ell \\ \Delta \lambda^\ell \end{bmatrix} = - \begin{bmatrix} \nabla_w \mathcal{L}(w^\ell, \lambda^\ell) \\ \Pi(w^\ell) \end{bmatrix}. \tag{13}$$

Here, ℓ is the Newton iteration index, Δw^ℓ is the search direction for the primal variables, $\Delta \lambda^\ell$ is the search direction for the dual variables, and $H(w^\ell, \lambda^\ell) = \nabla_{ww} \mathcal{L}(w^\ell, \lambda^\ell) + (W^\ell)^{-1} V^\ell$ is the Hessian of the Lagrange function $\mathcal{L}(w^\ell, \lambda^\ell) := \Phi(w^\ell) - \mu \sum_{i=1}^{n} \log w^\ell_{(i)} + \Pi(w^\ell)^T \lambda^\ell$. The matrix $M^\ell(\kappa_w)$ is known as the *augmented matrix.* The Newton step computation in a simulation-based approach operates only in the space of the parameters θ (the states are implicitly eliminated by simulation). In the time discretization approach, the Newton search is in the space of both the discretized states and parameters (contained in the high-dimensional variable vector w). Interestingly, however, the augmented matrix found in typical applications is highly sparse (with less than 1% of its entries are non-zero) [23].

The constant $\kappa_w \in \mathbb{R}_+$ is a *Hessian regularization parameter* which plays a key role in the context of parameter estimation. In particular, one can prove that the augmented matrix $M^\ell(\kappa_w)$ is non-singular (and thus the linear algebra system has a unique solution) if and only if the reduced Hessian matrix $Z^T H(w^\ell, \lambda^\ell) Z$ is positive definite and the Jacobian matrix $\nabla_w \Pi(w^\ell)$ has full row rank. Here, the matrix $Z \in \mathbb{R}^{n_\theta \times n_\theta}$ is such that its columns span the null-space of the Jacobian (i.e., $\nabla_w \Pi(w^\ell) Z = 0$). Moreover, the matrix Z is of the same dimension as the number of degrees of freedom (in our context this is precisely the number of parameters). When the reduced Hessian is positive definite (i.e., all its eigenvalues are positive) and the Jacobian has full row rank, one can prove that the Newton step of the primal variables Δw^ℓ obtained from the solution of (12) is a descent direction for the objective function (i.e., $(\Delta w^\ell)^T \nabla_w \Phi(w^\ell) < 0$) when the constraints are close to being satisfied (i.e., $\Pi(w^\ell) \approx 0$). This is key because it indicates that the Newton step improves the objective function (in our context, the negative likelihood function). This property cannot be guaranteed when the reduced Hessian is not positive definite. When such a situation is encountered, one can increase the regularization parameter κ_w until the reduced Hessian is positive definite and a descent direction is obtained. This approach is closely connected to the Levenberg-Marquardt method used in simulation-based estimation approaches (in which one regularizes the Hessian of the negative likelihood function as $-\nabla_{\theta\theta} L(\theta^\ell) + \kappa_\theta \mathbb{I}$) [49]. Another key observation is that, when the reduced Hessian is positive definite at the solution w^*, the estimated parameters are *unique*. This provides an indication that the experimental data is sufficiently informative to identify the parameters uniquely (i.e., the parameters are observable). We note that using a prior term $\varphi(\theta)$ in the MAP formulation has the effect of adding the positive definite matrix Σ_θ^{-1} to the reduced Hessian. This artificially regularizes the problem (as is done in the Levenberg-Marquardt scheme by adding the term $\kappa_w \mathbb{I}$). Consequently, when testing for observability/uniqueness, it is necessary to drop the prior term from the MAP formulation. Testing for observability also requires exact second order derivative information because the Hessian is needed. In the time-discretization approach, such information can be obtained directly from algebraic modeling languages. Simulation-based solution approaches often cannot check observability of the parameters (computing second derivatives using adjoint and sensitivity schemes is complicated).

Computing the eigenvalues of the reduced Hessian to check for positive definiteness is expensive. Interestingly, one can also determine if the reduced Hessian is positive definite by using inertia information of the augmented matrix $M^\ell(\kappa_w)$. The inertia of a matrix M is denoted as $\text{Inertia}(M) = \{n_+, n_-, n_0\}$ where n_+, n_-, and n_0 are the number of positive, negative, and zero eigenvalues of matrix M, respectively. One can prove that the reduced Hessian matrix is positive definite if $\text{Inertia}(M^\ell(\kappa_w)) = \{n, m, 0\}$, where we recall that n is the dimension of the variable vector w and m is the number of constraints. Notably, one can obtain the inertia of $M^\ell(\kappa_w)$ without computing the eigenvalues of the matrix. This is done by using modern sparse symmetric factorization routines such as `MA57` or `Pardiso` [48]. Such routines factorize the matrix $M^\ell(\kappa_w)$ as LBL^T where L is a lower triangular matrix and B is a matrix with 1×1 and 2×2 blocks in the diagonal. One can show that the number of positive and negative eigenvalues of $M^\ell(\kappa_w)$ are the number of positive and negative eigenvalues of B (which are easy to determine).

Modern interior-point solvers are equipped with highly sophisticated safeguarding techniques that enable the solution of highly nonlinear problems. A powerful approach is called a filter line-search method, in which one seeks to find a step-size κ such that the trial Newton iteration $w^{\ell+1} = w^\ell + \kappa \Delta w^\ell$ either decreases the objective function or the constraint violation $\|\Pi(w^\ell)\|$. If the step is accepted, the current values for the objective and constraint violation $(\Phi(w^\ell), \Pi(w^\ell))$ are stored in a filter (a history of previous successful iterations). At the next iterate, one requires that the Newton step is not in the filter and that it improves either the objective or the constraint violation.

This rather simple strategy is extremely effective in practice.

We highlight the fact that the proposed discretization approach bypasses the need to repetitively simulate the dynamic model (the discretized dynamic model contained in $\Phi(w)$ is solved progressively by Newton's method). This brings substantial computational savings. Moreover, since the discrete-time model is solved at the solution w^* of the NLP (9), we have that the discrete-time states $\{x_k^j\}^*$ approximate the state trajectories $x(t)$, $t \in \mathcal{T}_k, k \in \mathcal{K}$. In the absence of inequality constraints, one can also show that the reduced Hessian $Z^T H(w^*, \lambda^*) Z$ approximates the Hessian of the negative log-likelihood function $\nabla_{\theta\theta} L(\theta^*)$ in a neighborhood of w^* (which contains θ^*). We thus have that the reduced Hessian approximates the inverse parameter covariance matrix V_θ^{-1}. When inequality constraints are present, some of the parameters or state variables might hit their physical bounds and this deteriorates the approximation. When the parameters are not unique (the reduced Hessian has zero eigenvalues), the parameter covariance matrix is singular.

Structured Linear Algebra

A key advantage of using interior-point solvers is that they enable *modular linear algebra* implementations. For instance, the multi-experiment structure of problem (10) permeates down to the linear algebra system, to give a system of the form:

$$\left[\begin{array}{c|cccc} K_\theta & B_1^T & B_2^T & \dots & B_k^T \\ \hline B_1 & K_1 & & & \\ B_2 & & K_2 & & \\ \vdots & & & \ddots & \\ B_k & & & & K_k \end{array}\right] \begin{bmatrix} \Delta\theta \\ \Delta w_1 \\ \Delta w_2 \\ \vdots \\ \Delta w_K \end{bmatrix} = - \begin{bmatrix} r_\theta \\ r_1 \\ r_2 \\ \vdots \\ r_K \end{bmatrix}, \tag{14}$$

where $\Delta\theta$ is the Newton step for the parameters and $\Delta w_k = (\Delta x_k, \Delta\lambda_k)$ is the Newton step for variables in experiment k. The above system is said to have a block-bordered diagonal (BBD) structure. Here, we have that:

$$K_\theta = H_\theta, \qquad K_k = \begin{bmatrix} H_k & J_k^T \\ J_k & \end{bmatrix}, \qquad B_k^T = \left[Q_k^T \; T_k^T\right], \tag{15}$$

where $J_k = \nabla_{w_k}\Pi_k$, $T_k = \nabla_\theta \Pi_k$, $H_\theta = \nabla_{\theta\theta}\mathcal{L} + \kappa_w \mathbb{I}$, $H_k = \nabla_{w_k w_k}\mathcal{L} + W_k^{-1} V_k + \kappa_w \mathbb{I}$, $Q_k = \nabla_{\theta x_k}\mathcal{L}$, $r_\theta = \nabla_\theta \mathcal{L}$, and $r_k = \nabla_{w_k}\mathcal{L}$.

The BBD matrix is a permutation of the augmented matrix $M^\ell(\kappa_w)$ (obtained by ordering variables by experiment). The BBD matrix can thus be expressed as $P^T M^\ell(\kappa_w) P$ where P is a permutation matrix. The permutation does not affect the eigenvalues of the matrix. The BBD system (14) can be solved in parallel by using a Schur complement decomposition approach [45, 50]. This requires the solution of the linear algebra systems:

$$\left(K_\theta - \sum_{k\in\mathcal{K}} B_k K_k^{-1} B_k^T\right) \Delta w_\theta = -r_\theta + \sum_{k\in\mathcal{K}} K_k^{-1} B_k r_k \tag{16a}$$

$$K_k \Delta w_k = -r_k - B_k^T \Delta\theta, \; k \in \mathcal{K}. \tag{16b}$$

Here, $S = K_\theta - \sum_{k\in\mathcal{K}} B_k K_k^{-1} B_k^T$ is the *Schur complement matrix* which has the same dimension as the number of degrees of freedom (in our case the number of parameters). The key observation is that the experiment matrices K_k can be factorized by using an LBL^T factorization (by using `MA57` or `PARDISO`) *in parallel*. As a result, Schur decomposition can achieve high computational efficiency in estimation problems with many experimental data sets.

When using a Schur decomposition, one can estimate the inertia of the BBD matrix by using Haynsworth's formula:

$$\text{Inertia}(M^{\ell}(\kappa_w)) = \sum_{k\in\mathcal{K}} \text{Inertia}(K_k) + \text{Inertia}\left(K_\theta - \sum_{k\in\mathcal{K}} B_k K_k^{-1} B_k^T\right). \tag{17}$$

We recall that $n = n_\theta + \sum_k n_k$ and $m = \sum_k m_k$. Consequently, if we have that Inertia(K_k)=$\{n_k, m_k, 0\}$ for all $k \in \mathcal{K}$ then Inertia$(M^\ell(\kappa_w)) = \{n, m, 0\}$ if and only if Inertia$(S) = \{n_\theta, 0, 0\}$ (i.e., the Schur complement is positive definite). One can obtain the inertia of the blocks K_k and S using LBL^T factorization. This allows us to test observability of the parameters.

Uncertainty Quantification

The estimation problem under the MAP framework gives the values of the parameters θ^* that maximize the parameter posterior density. However, a characterization of the entire posterior is necessary to assess parameter uncertainty. The posterior covariance may be approximated from the reduced Hessian at the solution of the problem w^* and the covariance matrix can be used to determine ellipsoidal level sets of the posterior (confidence regions). This approach, however, might fail to capture nonlinear and constraint effects [23]. In this work, we circumvent these issues by using a randomized maximum a posteriori (rMAP) approach. Under this method, the posterior distribution is explored by using random perturbations on the experimental data (which can be easily parallelized). The rMAP framework can also deliver *approximate samples* from the parameter posterior distribution and implicitly captures nonlinear and constraint effects. To show this, we use the implicit mapping representation $\bar{\eta}_k = m_k(\theta)$. Under this representation, the posterior density (2) can be expressed as:

$$p(\theta \mid \eta) = \frac{1}{p(\eta)} \exp\left((\theta - \bar{\theta})^T \Sigma_\theta^{-1} (\theta - \bar{\theta}) + \sum_{k\in\mathcal{K}} (m_k(\theta) - \eta_k)^T \Sigma_k^{-1} (m_k(\theta) - \eta_k)\right) \tag{18a}$$

$$= \frac{1}{p(\eta)} \exp\left(-\frac{1}{2}(m(\theta) - \hat{\eta})^T \Sigma^{-1} (m(\theta) - \hat{\eta})\right) \tag{18b}$$

where $m(\theta) := (\theta, m_1(\theta), \cdots, m_K(\theta))$, $\Sigma := \text{diag}(\Sigma_\theta, \Sigma_1, \Sigma_2, \cdots, \Sigma_K)$, and $\hat{\eta} = (\bar{\theta}, \eta)$. Here, we redefine $\eta \leftarrow \hat{\eta}$ to enable compact notation. Since θ^* is a solution of the MAP problem, we have that:

$$\theta^* = \underset{\theta}{\text{argmin}}\ (m(\theta) - \eta)^T \Sigma^{-1} (m(\theta) - \eta). \tag{19}$$

If the mapping $m_k(\cdot)$ is continuously differentiable, we have that:

$$m(\theta) = m(\theta^*) + \nabla m(\theta^*)(\theta - \theta^*) + O(\|\theta - \theta^*\|_2^2). \tag{20}$$

To enable compact notation we define $\eta^* = m(\theta^*)$ and $\nabla m^* = \nabla m(\theta^*)$. We have that θ^* satisfies the stationary condition of (19):

$$(\nabla m^*)^T \Sigma^{-1} (\eta^* - \eta) = 0. \tag{21}$$

We use (20) to obtain a second-order Taylor approximation of the posterior as:

$$\begin{aligned}
p(\theta \mid \eta) &\approx \frac{1}{p(\eta)} \exp\left(-\frac{1}{2}(\eta^* - \eta + \nabla m^*(\theta-\theta^*))^T \Sigma^{-1} (\eta^* - \eta + \nabla m^*(\theta-\theta^*))\right) \\
&= \frac{\exp\left((\eta^*-\eta)^T\Sigma^{-1}(\eta^*-\eta)\right)}{p(\eta)} \exp\left(-\frac{1}{2}(\theta-\theta^*)^T \nabla m^{*T} \Sigma^{-1} \nabla m^* (\theta-\theta^*)\right) \\
&\propto \exp\left(-\frac{1}{2}(\theta-\theta^*)^T \nabla m^{*T} \Sigma^{-1} \nabla m^* (\theta-\theta^*)\right)
\end{aligned} \tag{22}$$

This implies that the posterior is *approximately* represented as:

$$\theta \mid \eta \sim \mathcal{N}\left(\theta^*, \left(\nabla m^{*T}\Sigma^{-1}\nabla m^*\right)^{-1}\right). \tag{23}$$

We recall that the output error is Gaussian and we can thus write $\eta = m(\theta) + \epsilon$ with $\epsilon \sim \mathcal{N}(0, \Sigma)$. We now consider the MAP problem with randomly perturbed data:

$$\tilde{\theta} = \underset{\theta}{\operatorname{argmin}}\ (m(\theta) - (\eta+\epsilon))^T\Sigma^{-1}(m(\theta)-(\eta+\epsilon)) \tag{24}$$

and note that

$$\begin{aligned}
&(m(\theta) - (\eta+\epsilon))^T\Sigma^{-1}(m(\theta)-(\eta+\epsilon)) \\
&= (\eta^* - \eta + \nabla m^*(\theta-\theta^*) - \epsilon)^T\Sigma^{-1}(\eta^* - \eta + \nabla m^*(\theta-\theta^*) - \epsilon) + O(\|\theta-\theta^*\|_2^2) \\
&= (\theta-\theta^*)^T\nabla m^{*T}\Sigma^{-1}\nabla m^*(\theta-\theta^*) + 2(\theta-\theta^*)^T\nabla m^{*T}\Sigma^{-1}\epsilon + O(\|\theta-\theta^*\|_2^2) + C,
\end{aligned} \tag{25}$$

where C is a constant. Consequently, for sufficiently small ϵ, we can linearize the mapping $m(\cdot)$ to obtain an approximate solution of (24) of the form:

$$\tilde{\theta} \approx \theta^* + \left(\nabla m^{*T}\Sigma^{-1}\nabla m^*\right)^{-1}\nabla m^{*T}\Sigma^{-1}\epsilon. \tag{26}$$

Here, we observe that the right-hand side of (26) is Gaussian with mean θ^* and covariance:

$$\begin{aligned}
&\left(\left(\nabla m^{*T}\Sigma^{-1}\nabla m^*\right)^{-1}\nabla m^{*T}\Sigma^{-1}\right)\Sigma\left(\left(\nabla m^{*T}\Sigma^{-1}\nabla m^*\right)^{-1}\nabla m^{*T}\Sigma^{-1}\right)^T \\
&= \left(\nabla m^{*T}\Sigma^{-1}\nabla m^*\right)^{-1}\left(\nabla m^{*T}\Sigma^{-1}\nabla m^*\right)\left(\nabla m^{*T}\Sigma^{-1}\nabla m^*\right)^{-1} \\
&= \left(\nabla m^{*T}\Sigma^{-1}\nabla m^*\right)^{-1}.
\end{aligned} \tag{27}$$

We thus have that:

$$\tilde{\theta} \sim \mathcal{N}\left(\theta^*, \left(\nabla m^{*T}\Sigma^{-1}\nabla m^*\right)^{-1}\right), \tag{28}$$

Consequently, solving (24) provides an approximate sample from the posterior distribution $p(\theta \mid \eta)$. The sampling procedure (24) is accurate up to second order. To obtain an exact sampling from the posterior, one needs to implement a rigorous MCMC scheme. The MCMC scheme removes the bias that appears in the rMAP sample density, which results from the second order approximation [34, 36]. Several works in the literature, however, report that accurate posterior densities can be obtained using an rMAP scheme [35, 51, 52]. We also note that the rMAP scheme implicitly captures nonlinearities and physical constraints when computing samples from the posterior. In particular, solving the perturbed problem (24) corresponds to solving the MAP problem (1) with randomly perturbed data and the MAP problem enforces constraints and handles the full nonlinear model.

Algebraic Modeling Platforms

Having an algebraic representation of the estimation problem has many practical and computational advantages. In particular, one can implement the estimation problem in easy-to-use and open-source modeling languages such as `JuMP` [53], `Plasmo.jl` [54], and `Pyomo` [55, 56]. These modeling languages are equipped with automatic differentiation techniques that compute exact first and second derivatives. Derivative information is communicated to optimization solvers without any user intervention. Modern algebraic modeling languages such as `Plasmo.jl` and `Pyomo` also allow users to convey structural information to the solvers. This is beneficial in the case of parameter estimation, where the structure can be exploited to enable parallelism and the use of high-performance computing clusters. In our framework, we use the modeling language `Plasmo.jl` to express multi-experiment estimation problems as graphs. Our implementation using `Plasmo.jl` is illustrated in Fig 2. The full `Julia` script is available at `https://github.com/zavalab/JuliaBox/tree/master/MicrobialPLOS`. We highlight that the same script can be used to solve the estimation problem using a general NLP solver such as `Ipopt` on a single-processor computer or with a structure-exploiting parallel NLP solver such as `PIPS-NLP` on multiple parallel computing processors (this might be a multi-core computing server or a large-scale computing cluster). This allows users with limited knowledge on scientific computing to gain access to advanced high-performance computing capabilities.

```
# Call Libraries
using JuMP, Plasmo, Ipopt

data=GetData() # Load data

# Create a graph for estimation model
graph=Plasmo.PlasmoGraph()
m_parent=Parent_Model() # Create parent model
node_parent=add_node!(graph,m_parent) # Connect parent to graph model

# Define array of children models
m_children=Array{JuMP.Model,1}(nExp)
node_children=Array{NodeOrEdge,1}(nExp)

# For each experiment,
for i in 1:nExp

m_children[i]=Exp_Model(data) # Create children model
node_children=add_node!(graph,m)) # Connect children to graph model

# Linking constraints (parameters common across experiments)
@linkconstraint(graph,[j in data[i].species,
                       k in [0 ; data[i].species]],
                          m_parent[:a][j,k]==m[:a][j,k])
end

# Solve problem with Ipopt
status=Plasmo.solve(graph)

 # Solve problem with PIPS-NLP
status= Plasmo.pipsnlp_solve(graph)
```

Fig 2. Snippet of parameter estimation implementation in `Plasmo.jl`

Results

Human gut microbial communities and microbiomes are highly dynamic networks coupled by positive or negative interactions and numerous feedback loops that display complex behaviors [57–60]. The generalized Lotka-Volterra (gLV) model provides a useful approach to capture such behavior [61–65]. Specifically, gLV captures single species growth rates and intra-species and inter-species positive and negative interactions. We apply the proposed NLP framework to estimate the growth and interaction parameters of the gLV model from experimental data collected in [60]. The microbial species involved in the experiments are shown in Table 1. Experiments were designed to study the synthetic ecology encompassing 12 prevalent human-associated intestinal species.

Table 1. List of microbial species used in case study.

Label	Full name	Abbreviation
1	Blautia hydrogenotrophica	BH
2	Collinsella aerofaciens	CA
3	Bacteroides uniformis	BU
4	Prevotella copri	PC
5	Bacteroides ovatus	BO
6	Bacteroides vulgatus	BV
7	Bacteroides thetaiotaomicron	BT
8	Eggerthella lenta	EL
9	Faecalibacterium prausnitzii	FP
10	Clostridium hiranonis	CH
11	Desulfovibrio piger	DP
12	Eubacterium rectale	ER

The gLV model is given by:

$$\frac{dx_s}{dt} = \left(\mu_s + \sum_{s' \in \mathcal{S}} \alpha_{ss'} x_{s'}\right) x_s, \;\; s \in \mathcal{S} \qquad \text{(Model 1)}$$

where $\mathcal{S} = \{1, 2, \cdots, S\}$ is the set of microbial species, $x_s : \mathbb{R} \to \mathbb{R}$ is the trajectory of the abundance of species $s \in \mathcal{S}$, μ_s is the growth rate of species s, and $\alpha_{ss'}$ is the interaction parameter that captures the effect of the abundance of species s' on the growth rate of species s. Species s and species s' are referred to as recipient species and donor species, resepectively.

The parameters (growth rates and interaction) cannot be calculated directly from first-principles and must be estimated from experimental data. The means of the prior densities of the parameters are assumed to be $\bar{\mu}_s = \bar{\alpha}_{ss'} = 0$ and their standard deviations are assumed to be $\sigma_\mu = \sigma_\alpha = 1/\sqrt{50}$. Such values are empirically determined by selecting the standard deviation values that give biologically feasible parameter estimates (the range of biologically feasible parameter values are $0.09 < \mu_s < 2.1$, $-10 < \alpha_{ij} < 10$, and $-10 < \alpha_{ii} < 0$). The variances for the output measurements are assumed to be $\sigma_{k,s}(t) = 0.05 \max(0.1, \eta_{k,s}(t))$. There are a total of 156 parameters including 12 monospecies growth rate and 144 interaction parameters (12 x 12). The set of experiments $\mathcal{K}$ includes 12 monospecies experiments and 66 pairwise community experiments (total of $K = 78$ experiments). The estimation problem contains a total of 144 differential equations (i.e. the model is a system of ordinary differential equations on $\mathbb{R}^{144}$). The computational characteristics of the estimation problem are summarized in Table 2 (labeled as P1).

Table 2. Characteristic of estimation problems used in scalability studies.

	Original	**Original+Synthetic**	**Synthetic**			
Label	**P1**	**P2**	**S1**	**S2**	**S3**	**S4**
Number of Species	12	12	12	24	36	48
Number of Diff. Equations	144	1,584	144	576	1,296	2,304
Number of Parameters	156	156	156	600	1,332	2,352
Number of Experiments	78	858	78	300	666	1,176
Number of Data Points	1,704	18,744	1,632	5,568	11,808	20,352
Number of NLP Variables	91,644	1,006,524	83,004	340,824	773,460	1,380,912
Number of NLP Constraints	91,488	1,006,368	82,848	340,224	772,128	1,378,560

Problem P1 includes the original experimental data [60] for a 12-species microbial community (consisting of 12 mono-species and 66 pairwise community experiments). The sampling frequency and the experiment duration of the mono-species experiments are 30 minutes and 24 hours, respectively. The sampling frequency in the pairwise community experiments is 12 hours and the experiment duration range from 60 to 72 hours. In the pairwise experiments, the media are diluted by 1/20 once every 24 hours. The dynamic model is discretized using an implicit Euler scheme with 5 equally-spaced discretization points (monospecies experiments) and 120 equally-spaced discretization points (pairwise experiments). The data used in P2 includes the original data of P1 and 10 additional synthetic data sets obtained with random data perturbations. Data for problems S1-S4 is synthetic and is obtained by running a simulation of the community model with fixed parameters and by adding 5% noise to the outputs. The data sampling characteristics (e.g., frequency, duration, dilution patterns) are the same as those of P1. The parameter values used for simulations are randomly generated from $\mu_s \sim \mathcal{N}(0.3, 0.1^2)$, $\alpha_{ss} \sim \mathcal{N}(-1, 0.1^2)$, and $\alpha_{ss'} \sim \mathcal{N}(0, 0.1^2)$.

Observability and Regularization

Parameter observability was checked by solving the MAP formulation for P1 (which uses the available experimental data) and by checking the inertia of the augmented system at the solution (reported by `Ipopt`). Here, we omitted the prior regularization term $\varphi(\cdot)$. We found that parameters obtained from P1 are *not unique* (not observable from the available data). Moreover, we found that the estimated parameter values without regularization have unrealistic (non-physical) values, see Fig 3 (a). This observation justifies the need to use prior information. The results obtained by adding L1 and L2 priors to the MAP formulation are presented in Fig 3 (b,c). Unique parameter estimates were found when L1 or L2 priors were used. We also found that the L1 prior induces sparser solutions (many parameters are zero). For the remainder of the results, we use the formulation with an L2 prior.

Model Fitting

Model validation was performed by assessing the goodness of fit to the experimental data (Fig 4). We can see that the model is capable of fitting most of the data points, but there are a number of experiments where the model prediction deviates significantly from the experimental data (such experiments are highlighted with red boxes). Furthermore, we can observe outliers at single data points (highlighted with red circles). Poor fitting can be caused by either bad local minima (the optimization solver finds a local optimal solution rather than the global optimal solution) or by a structural model error (the model structure is incapable of capturing the actual behavior of the system). To avoid bad local minima, we solved the MAP problem with multiple starting points. Such an approach increases the probability to find the global optimum, but obtaining a rigorous certificate of a global minimum is computationally challenging (rigorous global optimization techniques are currently not scalable to large problems). We found that the use of multiple starting points does not improve the model fit. Consequently, we

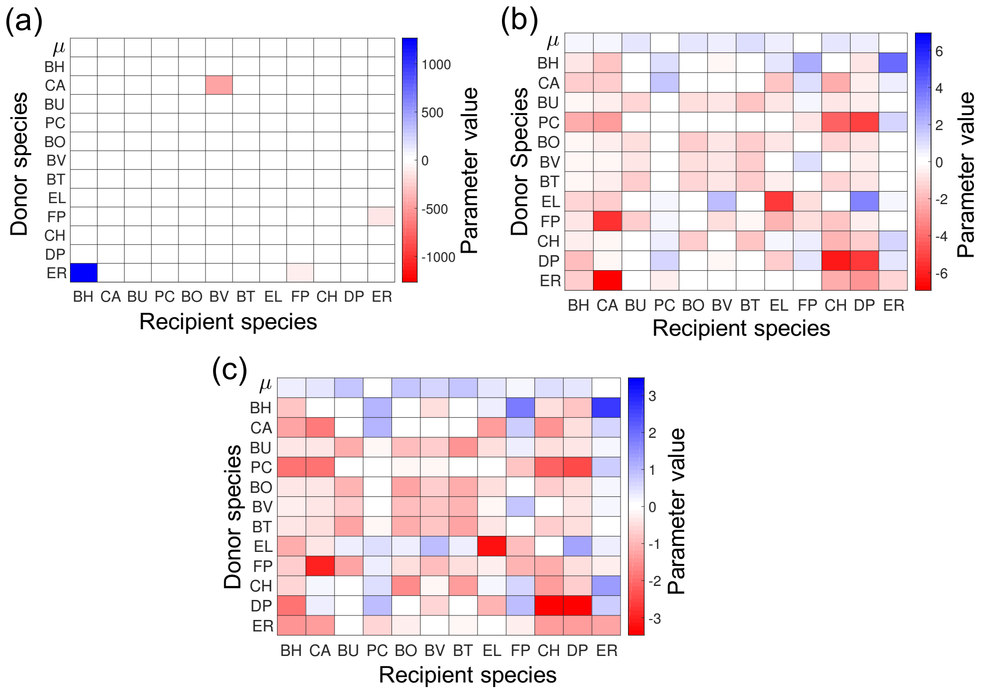

Fig 3. Parameter estimates with MAP formulations (Model 1). (a) Estimates using MAP formulation with no prior information. **(b)** Estimates using L1 prior. **(c)** Estimates using L2 prior. (a-c) The first row shows values for the growth rate parameters μ_s and the rest of the rows show values for the interaction parameters $\alpha_{ss'}$. The species name corresponding to s and s' are presented on the x and y axes. Recipient and donor species are on the x and y-axis, respectively.

attribute fitting errors to the model structure itself. In particular, the gLV model neglects various physical and biological phenomena such as lag phase or interaction coefficients that change as a function of time [60]. To investigate structural errors, we solved the MAP problem with a variant of the gLV model. In particular, we investigated the saturable gLV model [66,67] (Model 2):

$$\frac{dx_s}{dt} = \left(\mu_s + \sum_{s'\in\mathcal{S}} \frac{\alpha_{ss'} x_{s'}}{K_{ss'} + x_{s'}}\right) x_s, \qquad \text{(Model 2)}$$

where $K_{ss'} > 0$ are additional interaction parameters. The saturable model exhibits a higher degree of nonlinearity than the gLV model and includes S^2 more parameters (the number of degrees of freedom increases from $S^2 + S$ to $2S^2 + S$). As a result, the saturable gLV model provides more flexibility to improve model fitting. The model fitting obtained with the saturable gLV form is illustrated in Fig 5. As can be seen, significant improvements; in particular, the overall fitting error

$$\sum_{k\in\mathcal{K}} \sum_{s\in\mathcal{S}_k} \sum_{t\in\mathcal{T}_k} \frac{1}{2} \left(\frac{\eta_{k,s}(t) - \bar{\eta}_{k,s}(t)}{\sigma_{k,s}(t)}\right)^2, \qquad (29)$$

was reduced by 30%. Increasing the number of degrees of freedom can cause overfitting, however, and this can make the model less predictive. Consequently, there is a trade-off between fit and predictability.

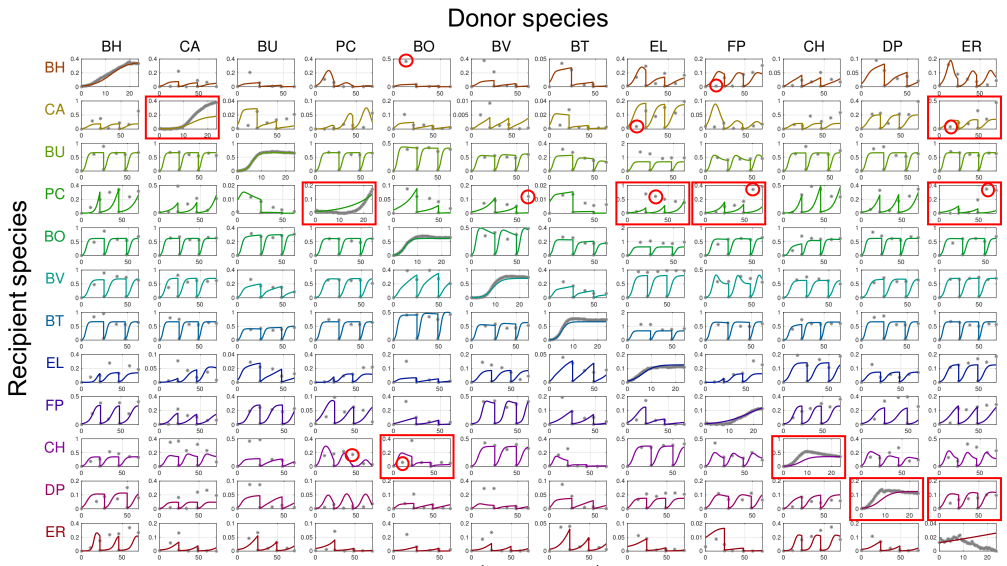

Fig 4. Fitting of experimental data (Model 1). Subplots show the measured and predicted species abundance in the microbial community. The subplots on the diagonal show fitting for mono-species experiments. Subplots on the i-th row and j-th column shows fitting for the corresponding pairwise culture (the abundance of species i in the presence of species j). Recipient and donor species are listed in rows and columns, respectively. For each subplot, the y-axis represents the absolute abundance in the community based on relative abundance multiplied by total biomass (OD600) and the x-axis represents the experiment time in hours. Data points are denoted by grey dots and dynamic model trajectories are denoted by solid lines. The data points highlighted with red circles are data points corresponding to the ten largest errors $(1/2)\left(\eta_{k,s}(t)-\bar{\eta}_{k,s}(t)\right)^2/\sigma_{k,s}(t)^2$. The subplots highlighted with red boxes are subplots for the experiments with the ten largest total prediction errors $\sum_{t\in\mathcal{T}_k}(1/2)\left(\eta_{k,s}(t)-\bar{\eta}_{k,s}(t)\right)^2/\sigma_{k,s}(t)^2$

.

Bayesian information criteria (BIC) is an approach for model selection [68–70]. BIC can be used as a score that represents the likelihood of the model that considers not only model fitting but also the number of parameters. In particular, the model with the smallest BIC can be considered as the most likely model. In BIC, one aims to compare the posterior probability of the models given the experimental observation. The posterior probabilities can be indirectly compared by comparing the marginal probability $p(\eta)=\int p(\eta|\theta)p(\theta)d\theta$ of the experimental observation for each model. By applying a Laplace approximation, one can derive a reasonable approximation of $-2\log p(\eta)$ as follows.

$$-2\log p(\eta)\approx \text{BIC} := -2\log p(\eta|\theta^*)+n_\theta \log n_\eta \tag{30}$$

Recall that n_η is the dimension of the output vector η (the number of experimental observations) and θ^* is the optimal parameter estimate. The log-likelihood term $-2\log p(\eta|\theta^*)$ corresponds to the prediction error (the squared errors across organisms and data sets) value of the MAP problem and is obtained while solving the problem. As can be seen from (30), the number of parameters are penalized in the score. Thus, the effect of overfitting that comes from increasing the number of parameters can be prevented. Our results indicate that the BIC of the gLV model is 44,798 and the BIC of the saturable model is 31, 572. This implies that the saturable model is the more likely model based on the available experimental data. We highlight, however, that the Laplace approximation assumes independent and identically distributed sampling and a sufficient number of samples (which does not strictly hold for our case). Model selection based on BIC with specialized experimental data is an interesting direction of future work.

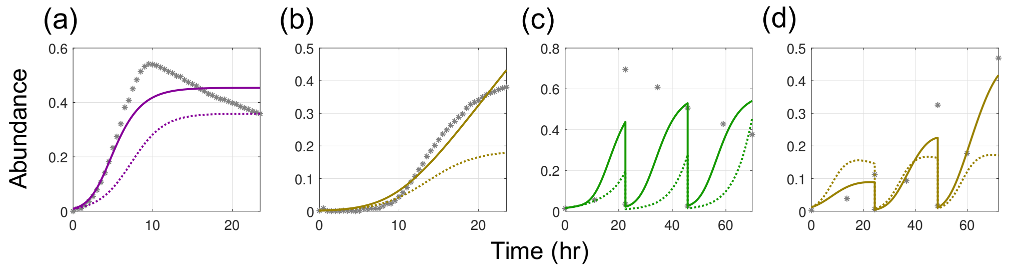

Fig 5. Improvement of model fitting with saturable gLV model (Model 2). Model fitting for 4 experiments selected among the experiments with 10 largest total prediction errors. The gLV model fits (dotted line) are compared with those of the saturable model (solid line). **(a)** Model fits to monospecies experiment with CH. **(b)** Model fits to monospecies experiment with CA. **(c)** Model fits to pairwise community experiment with PC in the presence of EL. **(d)** Model fits for pairwise community experiment with CA in the presence of ER.

To mitigate outliers, we solved the MAP problem with a CVaR norm and $\beta = 0.9$ (to penalize the 10% largest errors). The model fitting obtained with the CVaR formulation is shown in the supporting information and relevant results are summarized in Fig 6. It can be observed that the fitting errors for the outliers obtained with the standard MAP formulation are reduced. The effect of the CVaR formulation is also evident when analyzing the prediction error histogram (see Fig 7). In particular, we observe that the tail of high prediction errors becomes less pronounced under the CVaR formulation. In particular, the mean of the 10% largest errors decreases by 18% (from 167.81 to 137.04). On the other hand, it can also be observed that the mean error increases under CVaR and that the tail of small errors shrinks. This illustrates the fundamental trade-off that usually arises in robust statistics. The behavior induced with CVaR aids estimator performance because it prevents overfitting experimental data sets.

Inference (Posterior) Analysis

We used rMAP to assess the uncertainty of the 156 parameters estimated from P1 using the available experimental data. To do so, we draw data perturbations as $\eta_{k,s}(t) \leftarrow \eta_{k,s}(t) + \epsilon_{k,s}(t)$ with $\epsilon_{k,s}(t) \sim \mathcal{N}(0, \sigma_{k,s}(t)^2)$. We solved 500 MAP problems to obtain parameter samples and use this to approximate the covariance matrix for the posterior. The marginals for the posterior are shown in Fig 10 and the standard deviations are shown in Fig 8 (a). A large standard deviation indicates that the estimated parameter value is not reliable. We note that about half of the parameters can be estimated reliably while the other half exhibit significant uncertainty. This indicates that more experimental data should be obtained. From the sample covariance, we generated 95% ellipsoidal confidence regions for each pair of parameters. The correlation plots of μ_s against $\alpha_{ss'}$ for $s, s' \in \mathcal{S}$ are shown in Fig 11 and the Pearson correlation coefficients are shown in Fig 8 (b). In an ideal case, the parameters should be uncorrelated because data should be sufficient to estimate each parameter reliably.

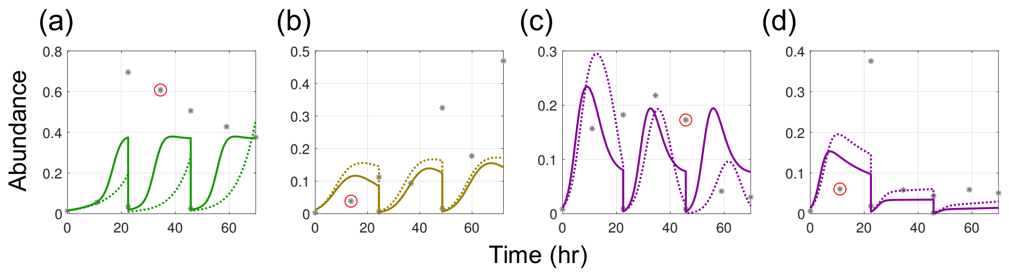

Fig 6. Improvement of model fitting by using CVaR (k-max) MAP formulation (Model 1). Model fits for 4 experiments selected among the experiments with 10 largest prediction errors $(1/2)\left(\eta_{k,s}(t) - \bar{\eta}_{k,s}(t)\right)^2 / \sigma_{k,s}(t)^2$. The model fits from standard MAP formulation (dotted line) are compared with the model fits from CVaR formulation with $\beta = 0.9$ (solid line). **(a)** gLV model fits to pairwise community experiment of PC in the presence of EL. **(b)** gLV model fits to pairwise community experiment of CA in the presence of ER. **(c)** gLV model fits to pairwise community experiment with CH in the presence of PC. **(d)** gLV model fits to pairwise community experiment with CH in the presence of BO.

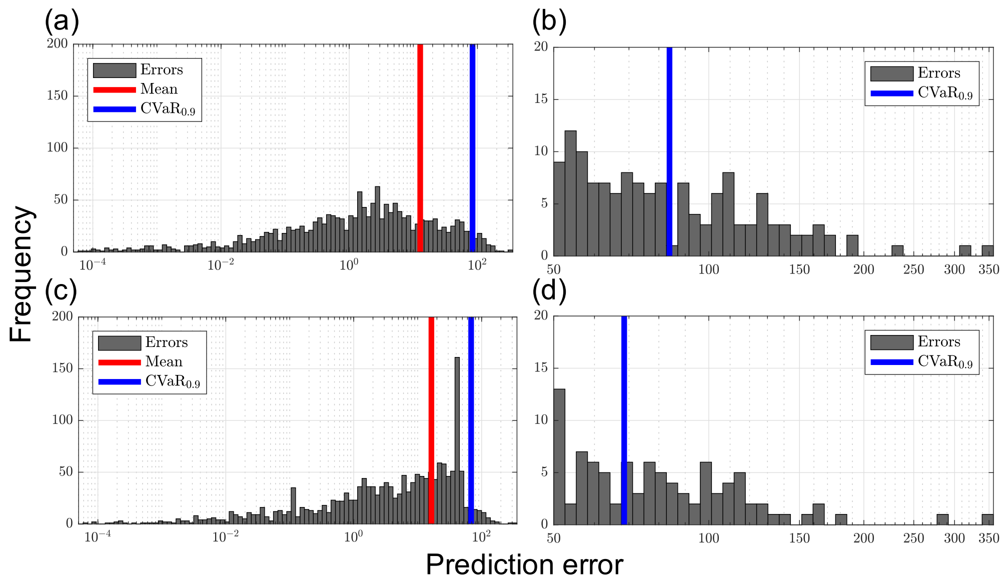

Fig 7. Histograms of fitting errors (Model 1). (a) Error histogram for the standard MAP formulation. **(b)** Tail region of (a). **(c)** Error histogram for CVaR formulation. **(d)** Tail region of (c). (a-d) The x-axis represents the value of prediction error evaluated at the solution and the y-axis represents the frequency. The red and blue line represent quantiles: the overall mean of prediction errors (red) and the mean of largest 10% errors (blue).

Using our data set, however, we can observe strong correlations between the parameters μ_s and α_{ss} in Fig 11, and strong positive and negative correlations can also be found in Fig 8 (b).

Furthermore, since the whole approximate distribution is obtained in the inference analysis based on rMAP framework, we can perform more sophisticated analysis on the characteristics of the distribution. In particular, one can investigate third and fourth moments (Fig 9) to examine the skewness and the kurtosis of the distribution. Such information can be used to investigate the deviation of the posterior distribution from the normal distribution. If the posteriors are stritly normally distributed, the third and fourth moments should be zero and $3\sigma^4$, respectively. However, we can observe that many posterior distributions deviate from such expectations. Thus, we can see that some of the distributions are not close to the normal distribution.

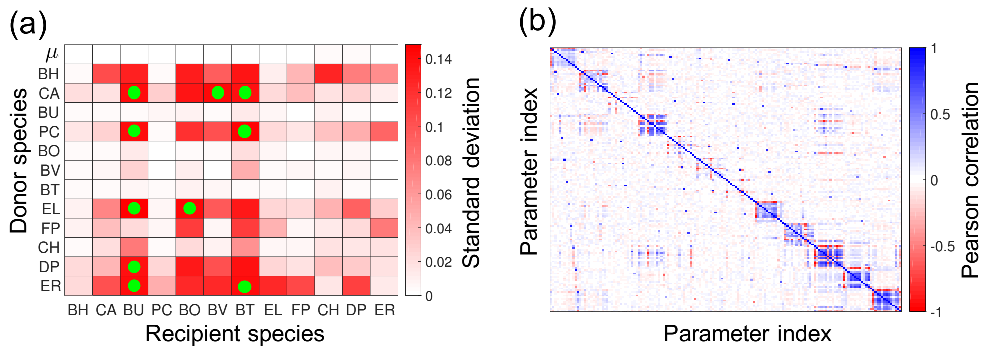

Fig 8. Parameter uncertainty and correlations (Model 1). (a) Heat map presents the standard deviation for the parameter posterior density. The first row shows the standard deviations of the growth rate parameters μ_s and the rest of the rows show the standard deviations of the interaction coefficients $\alpha_{ss'}$. Recipient and donor species are on the x and y-axis, respectively. The data points highlighted with green circles are data points corresponding to the ten largest standard deviations. **(b)** The heat map represents the Pearson correlation coefficients of the poterior distributions. The x-axis and the y-axis represents the index of parameters where the parameter vector is constructed as $\theta = (\mu_1, \alpha_{11} \cdots \alpha_{1S}, \cdots, \mu_S, \alpha_{S1}, \cdots \alpha_{SS})$. The block on the i-th row and j-th column is the Pearson correlation between the i-th and j-th component of parameter vector θ.

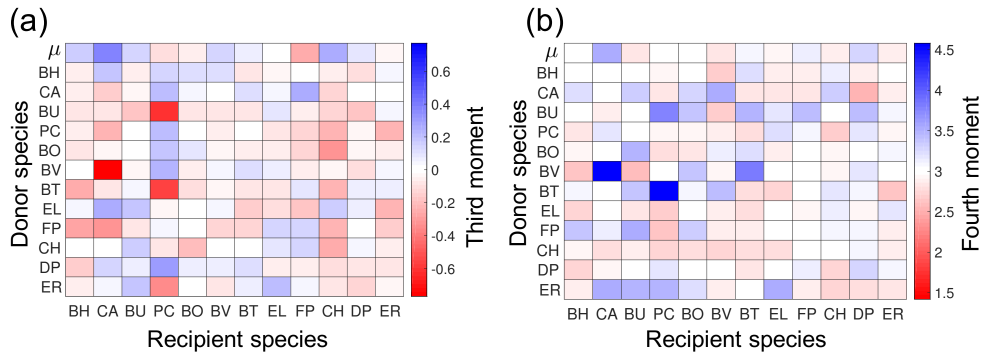

Fig 9. Third and fourth momentum of posterior (Model 1). (a) Heat map presents the third momentum of the parameter posterior density (normalized by σ^3).**(b)** The heat map represents the fourth momentum of the poterior density (normalized by σ^4). (a-b) The first row shows the standard deviations of the growth rate parameters μ_s and the rest of the rows show the standard deviations of the interaction coefficients $\alpha_{ss'}$. Recipient and donor species are on the x and y-axis, respectively.

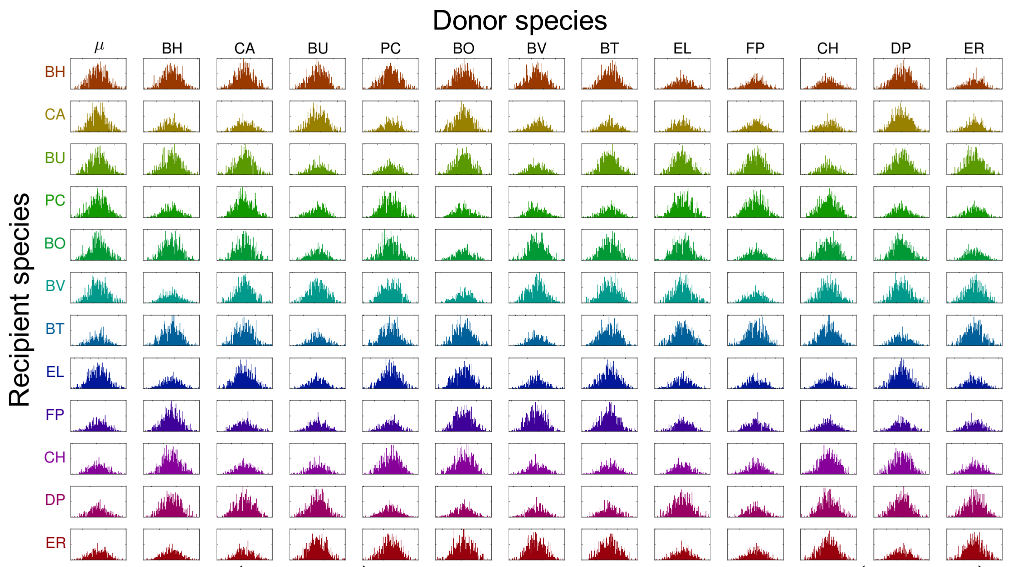

Fig 10. Posterior (marginal) densities for estimated parameters (Model 1). Each subplot shows the histogram of the samples from the approximate parameter posterior. The x-axis represents the values of the estimated parameters and the y-axis represents the frequencies. The subplots on the first column show the distribution of the growth rates μ_s and the rest of the subplots show distributions of the interaction parameters $\alpha_{ss'}$. Recipient and donor species are listed in rows and columns, respectively. The x-axis is scaled to show $\mu \pm 3\sigma$ where the μ is the mean and the σ is the standard deviation of the posterior distribution.

Computational Scalability

We assessed the computational scalability of the estimation framework by analyzing problems with different sizes and characteristics. Problem P1 was implemented in the algebraic modeling platform `JuMP` and solved with the NLP solver `Ipopt` configured with the sparse linear solver `MA57`. Problem P1 with gLV model and L2 prior was solved in 134 seconds and 78 NLP iterations on a standard computing server with an Intel(R) Xeon(R) CPU E5-2698 v3 processor running at 2.30GHz. Problem P1 with gLV model and L1 prior was solved in 219 seconds and 68 NLP iterations with the same hardware. A comparable problem requires over 7 hours to solve using a simulation-based approach implemented in `Matlab` and that uses finite differences to obtain first derivatives [60]. Despite the significant gains in computational performance obtained with `Ipopt`, its solution time scales nearly quadratically with the number of data sets. To overcome this scalability issue, we compared the performance of the serial solver `Ipopt` against that of the parallel solver `PIPS-NLP` (which uses a Schur complement decomposition to perform linear algebra operations). To test the scalability of `PIPS-NLP`, we generated a larger version of the estimation problem (labeled as P2). This problem is created by adding synthetic data sets. The NLP corresponding to P2 has over one million variables and constraints (but the number of parameters is the same as that of P1). This problem was implemented in `Plasmo.jl`. The benefit of using a parallel approach is clearly seen in Fig 12. Here, we highlight that `PIPS-NLP` solved P2 in less than 10 minutes and 94 NLP iterations using 16 cores while `Ipopt` requires around 30 minutes and 67 iterations. Furthermore, `IPOPT` found a different local solution and the solution from `PIPS-NLP` had a better objective value. Fig 12(b) also shows that `PIPS-NLP` achieves nearly

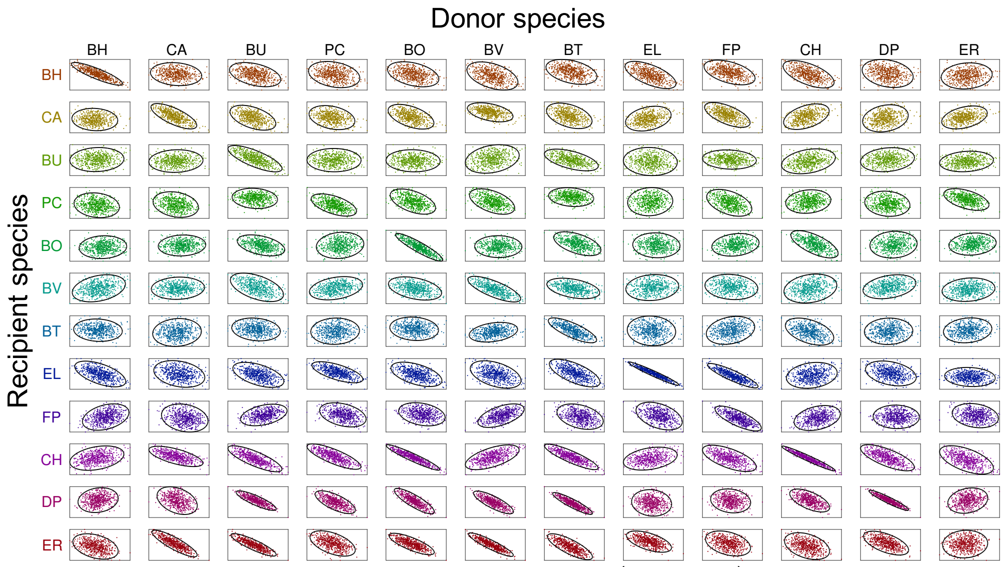

Fig 11. Correlations between parameter pairs (Model 1). Each subplot shows the 95% confidence regions (solid ellipses) of the approximate parameter posterior distributions and the sample points (dots). The subplots on the s-th row and s'-th column show the correlation of μ_s and $\alpha_{ss'}$. Recipient and donor species are listed in rows and columns, respectively. Only a representative subset of parameter pairs is presented (there are a total 12,090 pairs).

perfect strong scaling (speedup increases linearly with the number of cores).

In rMAP-based uncertainty quantification, the main computational challenge was the repetitive solution of the optimization problems. However, such challenge can be overcome by using the existing solution information. The required number of iterations in NLP solver can be greatly reduced when a good starting point (initial guess of the solution) is available (often referred to as *warm start*). Since only small modifications are made to the original problem to formulate the rMAP problem, the NLP solution of rMAP problem is very similar to that of the original problem. Thus, by warm-starting the NLP with the original NLP solution, the computational efforts to solve rMAP problem can be significantly reduced. In particular, most rMAP sampling problem was solved in less than 10 NLP iterations while the original problem required NLP 78 iterations.

We also assessed computational capability in estimation problems with the larger number of species in the microbial community (which increases the number of differential equations and parameters). Here, we generated synthetic data using simulations for larger communities. The generated data are summarized in Table 2. The number of the parameters and of data points scales nearly quadratically with respect to the size of the community. The computation times are shown in Fig 13. The results indicate that, by using `PIPS-NLP`, one can solve estimation problems with up to 48 species in *less than 15 minutes and 40 NLP iterations* (using 12 parallel computing cores). We highlight that, to the best of our knowledge, problem S4 is the largest estimation problem reported in computational biology literature. This problem contains 2,304 differential equations, 2,352 parameters, and 20,352 data points. The corresponding NLP contains 1.3 million variables and constraints.

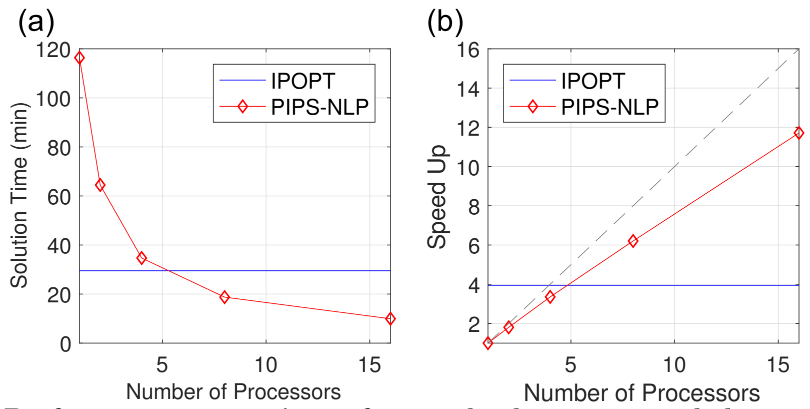

Fig 12. Performance comparison of general solver `Ipopt` and the structure-exploiting solver `PIPS-NLP` (Model 1). (a) Solution time for P2 using `Ipopt` and `PIPS-NLP`. The y-axis shows the solution time and the x-axis shows the number of cores used. For `Ipopt` the single core solution time is given by the horizontal blue line. **(b)** The y-axis represents the speed-up (the single-core solution time divided by the multi-core solution time). The blue line is the single-core solution time of `PIPS-NLP` divided by the single-core solution time of `Ipopt`. The grey dashed line represents the strong scaling line.

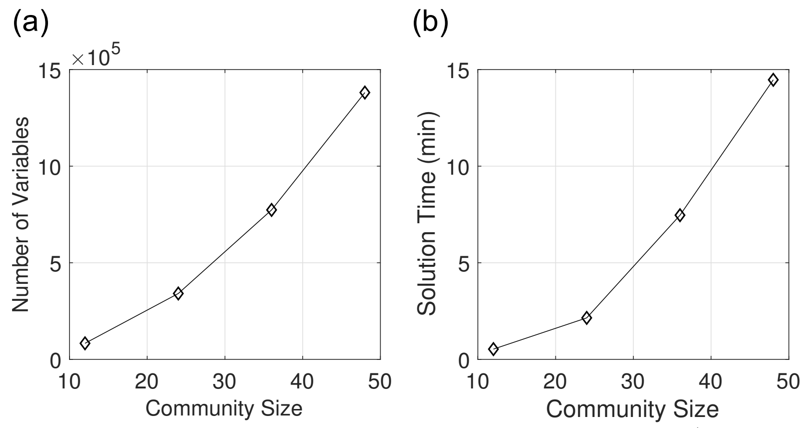

Fig 13. Computational scalability with larger communities (Model 1). (a) Number of variables against community size (total number of species). **(b)** The computation times for problems S1-S4 (see Table 2). The problems were solved with `PIPS-NLP` on 12 parallel cores (Intel(R) Xeon(R) CPU E5-2698 v3 processor running at 2.30GHz).

Concluding Remarks

The high computational efficiency achieved with the proposed framework can enable kinetic modeling of complex biological systems ranging from biomolecular networks to high-dimensional microbial communities [71]. Indeed, the proposed framework can be used to construct and analyze high-fidelity models of whole-cells or microbiomes [72, 73]. In particular, these methods can be applied to develop predictive dynamic models of multi-gene synthetic circuits interacting with host-cell processes for accurately predicting cell growth and synthetic circuit activity [74] or kinetic models of metabolite transformations driving community dynamics. These methods will advance our capability of integrating mechanistic modeling frameworks with large-scale experimental data. Furthermore, uncertainty quantification and observability analysis can provide valuable information to guide and accelerate experimental data collection. These capabilities are also essential in diagnosing structural model errors. The proposed framework uses state-of-the-art and easy-to-use modeling and solution tools that can be broadly applied to diverse biological systems and accessible to a wide range of users. Together, these advances will ultimately transform biology into a predictive and model-guided discipline.

References

1. Venturelli OS, Zuleta I, Murray RM, El-Samad H. Population diversification in a yeast metabolic program promotes anticipation of environmental shifts. PLoS biology. 2015;13(1):e1002042.

2. Friedman J, Gore J. Ecological systems biology: The dynamics of interacting populations. Current Opinion in Systems Biology. 2017;1:114–121.

3. Venayak N, Anesiadis N, Cluett WR, Mahadevan R. Engineering metabolism through dynamic control. Current opinion in biotechnology. 2015;34:142–152.

4. Ashyraliyev M, Fomekong-Nanfack Y, Kaandorp JA, Blom JG. Systems biology: Parameter estimation for biochemical models. FEBS Journal. 2009;276(4):886–902. doi:10.1111/j.1742-4658.2008.06844.x.

5. Sun J, Garibaldi JM, Hodgman C. Parameter estimation using metaheuristics in systems biology: A comprehensive review. IEEE/ACM Transactions on Computational Biology and Bioinformatics. 2012;9(1):185–202. doi:10.1109/TCBB.2011.63.

6. Raue A, Schilling M, Bachmann J, Matteson A, Schelke M, Kaschek D, et al. Lessons learned from quantitative dynamical modeling in systems biology. PloS one. 2013;8(9):e74335.

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

8. Leppavuori JT, Domach MM, Biegler LT. Parameter estimation in batch bioreactor simulation using metabolic models: Sequential solution with direct sensitivities. Industrial & Engineering Chemistry Research. 2011;50(21):12080–12091.

9. Mendes P, Kell D. Non-linear optimization of biochemical pathways: applications to metabolic engineering and parameter estimation. Bioinformatics (Oxford, England). 1998;14(10):869–883.

10. Moles CG, Mendes P, Banga JR. Parameter estimation in biochemical pathways: a comparison of global optimization methods. Genome research. 2003;13(11):2467–2474.

11. Kirkpatrick S, Gelatt CD, Vecchi MP, et al. Optimization by simulated annealing. science. 1983;220(4598):671–680.

12. Kikuchi S, Tominaga D, Arita M, Takahashi K, Tomita M. Dynamic modeling of genetic networks using genetic algorithm and S-system. Bioinformatics. 2003;19(5):643–650.

13. Tominaga D, Koga N, Okamoto M. Efficient numerical optimization algorithm based on genetic algorithm for inverse problem. In: Proceedings of the 2nd Annual Conference on Genetic and Evolutionary Computation. Morgan Kaufmann Publishers Inc.; 2000. p. 251–258.

14. Yang XS. Nature-inspired metaheuristic algorithms. Luniver press; 2010.

15. Toni T, Welch D, Strelkowa N, Ipsen A, Stumpf MP. Approximate Bayesian computation scheme for parameter inference and model selection in dynamical systems. Journal of the Royal Society Interface. 2009;6(31):187–202.

16. Toni T, Stumpf MP. Simulation-based model selection for dynamical systems in systems and population biology. Bioinformatics. 2009;26(1):104–110.

17. Balsa-Canto E, Peifer M, Banga JR, Timmer J, Fleck C. Hybrid optimization method with general switching strategy for parameter estimation. BMC systems biology. 2008;2(1):26.

18. Vaz AIF, Vicente LN. A particle swarm pattern search method for bound constrained global optimization. Journal of Global Optimization. 2007;39(2):197–219.

19. Zavala VM, Biegler LT. Optimization-based strategies for the operation of low-density polyethylene tubular reactors: Moving horizon estimation. Computers & Chemical Engineering. 2009;33(1):379–390.

20. López-Negrete R, Biegler LT. A moving horizon estimator for processes with multi-rate measurements: A nonlinear programming sensitivity approach. Journal of Process Control. 2012;22(4):677–688.

21. Lillacci G, Khammash M. Parameter estimation and model selection in computational biology. PLoS computational biology. 2010;6(3):e1000696.

22. Biegler LT. Nonlinear programming: concepts, algorithms, and applications to chemical processes. vol. 10. Siam; 2010.

23. Zavala VM. Computational strategies for the optimal operation of large-scale chemical processes. Carnegie Mellon University; 2008.

24. Albuquerque JS, Biegler LT. Decomposition algorithms for on-line estimation with nonlinear DAE models. Computers & chemical engineering. 1997;21(3):283–299.

25. Leibman M, Edgar T, Lasdon L. Efficient data reconciliation and estimation for dynamic processes using nonlinear programming techniques. Computers & chemical engineering. 1992;16(10-11):963–986.

26. Tjoa IB, Biegler LT. Simultaneous solution and optimization strategies for parameter estimation of differential-algebraic equation systems. Industrial & Engineering Chemistry Research. 1991;30(2):376–385.

27. Betts JT. Optimal interplanetary orbit transfers by direct transcription. Journal of the Astronautical Sciences. 1994;42(3):247–268.

28. Betts JT, Cramer EJ. Application of direct transcription to commercial aircraft trajectory optimization. Journal of Guidance, Control, and Dynamics. 1995;18(1):151–159.

29. Bottasso CL, Croce A. Optimal control of multibody systems using an energy preserving direct transcription method. Multibody System Dynamics. 2004;12(1):17–45.

30. Biegler LT. An overview of simultaneous strategies for dynamic optimization. Chemical Engineering and Processing: Process Intensification. 2007;46(11):1043–1053.

31. Pirnay H, López-Negrete R, Biegler LT. Optimal sensitivity based on IPOPT. Mathematical Programming Computation. 2012;4(4):307–331.

32. Chib S, Greenberg E. Understanding the metropolis-hastings algorithm. The american statistician. 1995;49(4):327–335.

33. Gamerman D, Lopes HF. Markov chain Monte Carlo: stochastic simulation for Bayesian inference. Chapman and Hall/CRC; 2006.

34. Wang K, Bui-Thanh T, Ghattas O. A randomized maximum a posteriori method for posterior sampling of high dimensional nonlinear Bayesian inverse problems. SIAM Journal on Scientific Computing. 2018;40(1):A142–A171.

35. Oliver DS. Metropolized Randomized Maximum Likelihood for sampling from multimodal distributions. arXiv preprint arXiv:150708563. 2015;.

36. Bardsley JM, Solonen A, Haario H, Laine M. Randomize-Then-Optimize: a Method for Sampling From Posterior Distributions in Nonlinear Inverse Problems. Siam Journal on Scientific Computing. 2014;36(4):A1895–A1910. doi:10.1137/140964023.

37. Oliver DS, He N, Reynolds AC. Conditioning permeability fields to pressure data. In: ECMOR V-5th European Conference on the Mathematics of Oil Recovery; 1996.

38. Rockafellar RT, Uryasev S. Optimization of conditional value-at-risk. Journal of risk. 2000;2:21–42.

39. Rockafellar RT, Uryasev S. Conditional value-at-risk for general loss distributions. Journal of banking & finance. 2002;26(7):1443–1471.

40. Boyd S, Vandenberghe L. Convex optimization. Cambridge university press; 2004.

41. Tikhonov A. Numerical methods for the solution of ill-posed problems;.

42. Golub GH, Van Loan CF. Matrix computations. vol. 3. JHU Press; 2012.

43. Tibshirani R. Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society Series B (Methodological). 1996; p. 267–288.

44. Pavlikov K, Uryasev S. CVaR norm and applications in optimization. Optimization Letters. 2014;8(7):1999–2020. doi:10.1007/s11590-013-0713-7.

45. Zavala VM, Laird CD, Biegler LT. Interior-point decomposition approaches for parallel solution of large-scale nonlinear parameter estimation problems. Chemical Engineering Science. 2008;63(19):4834–4845.

46. Wächter A, Biegler LT. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. Mathematical Programming. 2006;106(1):25–57. doi:10.1007/s10107-004-0559-y.

47. Byrd RH, Nocedal J, Waltz RA. KNITRO: An integrated package for nonlinear optimization. In: Large-scale nonlinear optimization. Springer; 2006. p. 35–59.

48. Zavala VM, Biegler LT. Nonlinear programming strategies for state estimation and model predictive control. In: Nonlinear model predictive control. Springer; 2009. p. 419–432.

49. Bard Y. Nonlinear parameter estimation. 1974;.

50. Kang J, Chiang N, Laird CD, Zavala VM. Nonlinear programming strategies on high-performance computers. In: Decision and Control (CDC), 2015 IEEE 54th Annual Conference on. IEEE; 2015. p. 4612–4620.

51. Emerick AA, Reynolds AC. Investigation of the sampling performance of ensemble-based methods with a simple reservoir model. Computational Geosciences. 2013;17(2):325–350.

52. Gao G, Zafari M, Reynolds AC, et al. Quantifying uncertainty for the PUNQ-S3 problem in a Bayesian setting with RML and EnKF. In: SPE reservoir simulation symposium. Society of Petroleum Engineers; 2005.

53. Dunning I, Huchette J, Lubin M. JuMP: A Modeling Language for Mathematical Optimization. arXiv:150801982 [mathOC]. 2015;59(2):1–25. doi:10.1137/15M1020575.

54. Jalving J, Abhyankar S, Kim K, Hereld M, Zavala VM. A graph-based computational framework for simulation and optimisation of coupled infrastructure networks. IET Generation, Transmission & Distribution. 2017;11(12):3163–3176.

55. Hart WE, Laird CD, Watson JP, Woodruff DL, Hackebeil GA, Nicholson BL, et al. Pyomo–optimization modeling in python. vol. 67. 2nd ed. Springer Science & Business Media; 2017.

56. Hart WE, Watson JP, Woodruff DL. Pyomo: modeling and solving mathematical programs in Python. Mathematical Programming Computation. 2011;3(3):219–260.

57. Huttenhower C, Gevers D, Knight R, Abubucker S, Badger JH, Chinwalla AT, et al. Structure, function and diversity of the healthy human microbiome. Nature. 2012;486(7402):207.

58. Tropini C, Earle KA, Huang KC, Sonnenburg JL. The Gut microbiome: connecting spatial organization to function. Cell host & microbe. 2017;21(4):433–442.

59. Earle KA, Billings G, Sigal M, Lichtman JS, Hansson GC, Elias JE, et al. Quantitative imaging of gut microbiota spatial organization. Cell host & microbe. 2015;18(4):478–488.

60. Venturelli OS, Carr AC, Fisher G, Hsu RH, Lau R, Bowen BP, et al. Deciphering microbial interactions in synthetic human gut microbiome communities. Molecular Systems Biology. 2018;14(6). doi:10.15252/msb.20178157.

61. Lotka AJ. Elements of physical biology. Science Progress in the Twentieth Century (1919-1933). 1926;21(82):341–343.

62. Volterra V. Variations and fluctuations of the number of individuals in animal species living together. ICES Journal of Marine Science. 1928;3(1):3–51.

63. Stein RR, Bucci V, Toussaint NC, Buffie CG, Rätsch G, Pamer EG, et al. Ecological modeling from time-series inference: insight into dynamics and stability of intestinal microbiota. PLoS computational biology. 2013;9(12):e1003388.

64. Mounier J, Monnet C, Vallaeys T, Arditi R, Sarthou AS, Hélias A, et al. Microbial interactions within a cheese microbial community. Applied and environmental microbiology. 2008;74(1):172–181.

65. Widder S, Allen RJ, Pfeiffer T, Curtis TP, Wiuf C, Sloan WT, et al. Challenges in microbial ecology: building predictive understanding of community function and dynamics. The ISME journal. 2016;10(11):2557.

66. Momeni B, Xie L, Shou W. Lotka-Volterra pairwise modeling fails to capture diverse pairwise microbial interactions. Elife. 2017;6.

67. Thébault E, Fontaine C. Stability of ecological communities and the architecture of mutualistic and trophic networks. Science. 2010;329(5993):853–856.

68. Konishi S, Kitagawa G. Information criteria and statistical modeling. Springer Science & Business Media; 2008.

69. Chen J, Chen Z. Extended Bayesian information criteria for model selection with large model spaces. Biometrika. 2008;95(3):759–771.

70. Schwarz G, et al. Estimating the dimension of a model. The annals of statistics. 1978;6(2):461–464.

71. Stanford NJ, Lubitz T, Smallbone K, Klipp E, Mendes P, Liebermeister W. Systematic construction of kinetic models from genome-scale metabolic networks. PloS one. 2013;8(11):e79195.

72. Karr JR, Sanghvi JC, Macklin DN, Gutschow MV, Jacobs JM, Bolival Jr B, et al. A whole-cell computational model predicts phenotype from genotype. Cell. 2012;150(2):389–401.

73. Macklin DN, Ruggero NA, Covert MW. The future of whole-cell modeling. Current opinion in biotechnology. 2014;28:111–115.

74. Weiße AY, Oyarzún DA, Danos V, Swain PS. Mechanistic links between cellular trade-offs, gene expression, and growth. Proceedings of the National Academy of Sciences. 2015; p. 201416533.