Supporting Information

Babtie et al. 10.1073/pnas.1414026112

SI Materials and Methods

GP Regression. We use GP regression, a nonparametric Bayesian method for nonlinear regression, to model the concentration of each species as a function of time, $x_n(t)$, and the corresponding derivatives, $\dot{x}_n(t)$, from time course data for each species. For our application, data are simulated from the initial candidate ODE model we wish to analyze for sensitivity to topological alterations. We used MATLAB functions from the GPML Toolbox (1, 2) to infer hyperparameters and fit the GP regression models.

A GP is a collection of random variables, any finite subset of which follows a multivariate Gaussian distribution (2). For GP regression we assume a GP prior over a function, denoted, e.g.,

$$x_n(\mathbf{t}) \sim \mathcal{GP}\left(m(\mathbf{t}), k(\mathbf{t}, \mathbf{t}')\right), \qquad \text{[S1]}$$

where $m(\mathbf{t})$ is a mean function for the values taken by variable x_n at times $\mathbf{t}$ and $k(\mathbf{t}, \mathbf{t}')$ is a covariance function. We use a zero-mean function and a squared covariance function,

$$k(t_i, t_j) = \sigma_f^2 \exp\left(-\frac{(t_i - t_j)^2}{2\ell^2}\right), \qquad \text{[S2]}$$

where σ_f and l are hyperparameters defining the distribution. We assume the data are subject to normally distributed noise with constant variance σ_ϵ^2, thus inducing a GP prior over the observed outputs for species n, $y_n(\mathbf{t})$,

$$y_n(\mathbf{t}) \sim \mathcal{GP}\left(m_y(\mathbf{t}), k_y(\mathbf{t}, \mathbf{t}')\right), \qquad \text{[S3]}$$

with $m_y(t_i) = m(t_i)$ and $k_y(t_i, t_j) = k(t_i, t_j) + \sigma_\epsilon^2 \delta_{ij}$, where δ_{ij} is the Kronecker delta function.

Given the assumed GP prior and noise model we can write the joint distribution,

$$\begin{bmatrix} \mathbf{y}_n \\ \mathbf{x}_n^* \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mathbf{m} \\ \mathbf{m}_* \end{bmatrix}, \begin{pmatrix} K + \sigma_\epsilon^2 I & K_{o*} \\ K_{*o} & K_{**} \end{pmatrix}\right), \qquad \text{[S4]}$$

where $\mathbf{y}_n = [y_n(t_1), \ldots, y_n(t_s)]^T$ is a set of observed outputs at times $t_1, \ldots, t_s$; $\mathbf{x}_n^* = [x_n(t_1^*), \ldots, x_n(t_r^*)]^T$ for any finite set of time points $t_1^*, \ldots, t_r^*$; $\mathbf{m} = [m(t_1), \ldots, m(t_s)]^T$ and $\mathbf{m}_* = [m(t_1^*), \ldots, m(t_r^*)]^T$ are vectors specified by the mean function $m(t)$; I is the $s \times s$ identity matrix; and entries in the covariance matrices are given by

$$\begin{aligned} K_{ij} &= k(t_i, t_j), \\ (K_{o*})_{ij} &= k(t_i, t_j^*), \\ (K_{*o})_{ij} &= k(t_i^*, t_j), \\ (K_{**})_{ij} &= k(t_i^*, t_j^*). \end{aligned}$$

We can specify the likelihood $p(\mathbf{y}_n)$ as

$$p(\mathbf{y}_n) = \frac{1}{(2\pi)^{s/2}\left|K + \sigma_\epsilon^2 I\right|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{y}_n - \mathbf{m})^T \left(K + \sigma_\epsilon^2 I\right)^{-1} (\mathbf{y}_n - \mathbf{m})\right). \qquad \text{[S5]}$$

Following the method of Rasmussen and Williams (2), we determine values for the GP hyperparameters $(\sigma_f, l, \sigma_\epsilon)$ by maximizing the likelihood function with respect to these parameters.

We obtain the posterior distribution for our function $x_n(\mathbf{t})$ by updating the GP prior using the observed dataset $\mathbf{y}_n$. From the joint distribution in Eq. **S4** we can specify the GP posterior for $x_n(t)$ conditioned on the observed data:

$$\left[x_n(t_1^*), \ldots, x_n(t_r^*)\right]^T \Big| \mathbf{y}_n \sim \mathcal{N}\left(\mathbf{m}_{post}, K_{post}\right), \qquad \text{[S6]}$$

where

$$\mathbf{m}_{post} = \mathbf{m}_* + K_{*o}\left(K + \sigma_\epsilon^2 I\right)^{-1}(\mathbf{y}_n - \mathbf{m}),$$

$$K_{post} = K_{**} - K_{*o}\left(K + \sigma_\epsilon^2 I\right)^{-1} K_{o*}.$$

Using this approach, we can sample realizations of the function $x_n(t)$ at any chosen time point.

Similarly, we can specify the joint distribution of the corresponding derivative $\dot{x}_n(t)$ and the observed data $\mathbf{y}_n$ to obtain the GP posterior distribution for the derivative,

$$\left[\dot{x}_n(t_1), \ldots, \dot{x}_n(t_s)\right]^T \Big| \mathbf{y}_n \sim \mathcal{N}\Big(L_{DF}\left(K + \sigma_\epsilon^2 I\right)^{-1}\mathbf{y}_n, M - L_{DF}\left(K + \sigma_\epsilon^2 I\right)^{-1} L_{FD}\Big), \qquad \text{[S7]}$$

where covariance matrix entries are defined by

$$K_{ij} = k(t_i, t_j), \qquad \text{[S8]}$$

$$(L_{FD})_{ij} = \mathrm{cov}\left(x_n(t_i), \dot{x}_n(t_j)\right) = \frac{d}{dt_j} k(t_i, t_j) = \frac{(t_i - t_j)}{\ell^2} K_{ij}, \qquad \text{[S9]}$$

$$(L_{DF})_{ij} = \mathrm{cov}\left(\dot{x}_n(t_i), x_n(t_j)\right) = \frac{d}{dt_i} k(t_i, t_j) = \frac{(t_j - t_i)}{\ell^2} K_{ij}, \qquad \text{[S10]}$$

$$(M)_{ij} = \mathrm{cov}\left(\dot{x}_n(t_i), \dot{x}_n(t_j)\right) = \frac{d^2}{dt_i dt_j} k(t_i, t_j) = \left(\frac{1}{\ell^2} - \frac{(t_i - t_j)^2}{\ell^4}\right) K_{ij}. \qquad \text{[S11]}$$

Here we assumed a zero-mean function for $m(t)$ and only considered the time points $t_1, \ldots, t_s$ of the observed data points to simplify the notation. Samples of the derivative function at other time points were obtained by calculating the corresponding covariance matrix entries.

Simulation Parameters for Synthetic Datasets. Data in Fig. 3 were simulated from model A with parameters $s_n = 0.2$, $\beta_{nk} = 2$, $m_{nk} = 5$, and $\theta_{nk} = 1.5$ for all n, k; and values for γ_n given by the nth component of vector $\boldsymbol{\gamma} = [0.9, 0.9, 0.7, 1.5, 1.5]$. Initial concentrations of species in the system were set to $\mathbf{x}(0) = [1, 0.5, 1, 0.5, 0.5]$.

Trajectories in Fig. S2*B* were simulated from model A with parameters s_n and γ_n given by the nth components of vectors $\mathbf{s} = [0.5, 0.5, 0.2, 0.2, 0.2]$ and $\boldsymbol{\gamma} = [0.9, 0.9, 0.7, 0.5, 1.3]$, respectively, and the parameters associated with interactions set to $(\beta_{15}, \beta_{21}, \beta_{31}, \beta_{41}, \beta_{43}, \beta_{52}, \beta_{54}) = (2, 1.5, 3, 1, 2, 2, 2)$, $(\theta_{15}, \theta_{21}, \theta_{31}, \theta_{41}, \theta_{43}, \theta_{52}, \theta_{54}) = (1, 1, 1, 1, 2, 1.5, 1.5)$, and $(m_{15}, m_{21}, m_{31}, m_{41}, m_{43}, m_{52}, m_{54}) = (3, 2, 2, 2, 3, 1, 4)$. Initial species concentrations were set to $\mathbf{x}(0) = [0.1, 0.5, 1, 0.5, 0.5]$ for condition 1 and $\mathbf{x}(0) = [0.1, 0.1, 0, 3, 2.5]$ for condition 2.

Fig. 4*A* data were simulated from model B with parameters r_n given by $\mathbf{r} = [0.3, 0.7, 0.5, 0.4, 0.4]$, and $(a_{13}, a_{15}, a_{24}, a_{42}, a_{43}, a_{52}) = (0.4, 0.7, 1.5, 1.4, 0.7, 1.2)$. Initial species populations were set to $\mathbf{x}(0) = [0.2, 0.5, 0.2, 0.2, 0.3]$.

Data used for Bayesian inference (Fig. S4) were simulated from model C with parameters $\mathbf{r} = [0.3, 0.7, 0.5, 0.4, 0.4]$ and $(a_{12}, a_{14}, a_{21}, a_{23}, a_{31}, a_{34}, a_{41}, a_{45}, a_{51}, a_{54}) = (0.4, 0.3, 1.5, 1.4, 0.7, 1.2, 0.6, 1.5, 1.1, 0.2)$.

During gradient-matching parameter estimation, the allowed values for parameters were constrained by the limits $0.1 \le s_n \le 1$, $0.1 \le \gamma_n \le 2$, $0.5 \le \beta_{nk} \le 4$, $0.2 \le \theta_{nk} \le 3$, and $0.7 \le m_{nk} \le 5$ for gene regulatory network models, and $0.1 \le r_n \le 2$ and $0.1 \le a_{nk} \le 5$ for competitive population dynamics models.

Parameter Inference. We used one of the following methods (as stated in the main text) to infer model parameters from the synthetic or experimental datasets. In all cases we specified likelihoods by assuming Gaussian noise with fixed variance.

Maximum likelihood estimation plus parametric bootstrap. Maximum likelihood estimates for the parameters were obtained from the original dataset and used to simulate trajectories for all species; replicate datasets were generated based on these trajectories assuming additive Gaussian noise. We obtained parameter estimates from each replicate dataset using constrained optimization to generate approximate sampling distributions for each parameter (3).

Nested sampling. Nested sampling is an algorithm developed by Skilling (4) to estimate the evidence for a particular model which also provides samples from the posterior distribution. We used a C implementation of the algorithm (5) with uniform priors for all parameters and a random walk sampling algorithm.

Metropolis–Hastings. This is a Markov chain Monte Carlo method that enables sampling from the joint posterior distribution (6, 7); we used a Gaussian transition kernel to generate parameter proposals, and uniform priors for all parameters.

Laplace approximation. This method approximates the posterior by a multivariate normal probability density function. Although unlikely to be a good global approximation of the posterior, it may nevertheless provide a good local approximation in the region of the estimated parameter vector (obtained using constrained optimization techniques, as in "*Maximum likelihood estimation plus parametric bootstrap*" above). We used the R implementation provided in the LaplacesDemon package (8).

SI Results

Automated Model Generation and Ranking. As described in the main text, we construct and rank all possible component equations that describe the dynamics of each species in a system using gradient-matching parameter estimation. Fig. S2*A* illustrates the rankings of the 165 possible component equations calculated using the oscillating GP regression model trajectories displayed in Fig. 3 (main text), and the rules described in the accompanying section of the main text (titled "*Automated Model Generation and Ranking*"). We combine component equations for each species to create a set of coupled ODEs describing the dynamics of the complete system. As shown in Fig. 3 (main text), the best model accurately captures the desired dynamics, whereas lower-ranked models deviate from these trajectories. The "top-ranked ODE" in Fig. 3 was created by combining the top-ranked component model for each species, whereas the exemplar "lower-ranked ODE" was constructed by combining the eighth-ranked component equations. For clarity, in Fig. 3 we only show simulations from two possible ODE models, but there are many alternative highly ranked models that also produce the desired dynamics displayed by the best model.

The relative rankings of models depend on the particular dataset to which they are fitted during this parameter estimation step. If we have multiple simulations from our candidate model, corresponding to the known behavior of the modeled system under different experimental conditions, we can use this information to reduce the set of compatible models. For example, Fig. S2*B* shows trajectories simulated from model A using identical parameter values but two different initial conditions (see *SI Materials and Methods* for simulation parameters). As before, we generate the same $33 \times 5 = 165$ component equations, which combine to give $33^5 = 3.9 \times 10^7$ possible complete ODE models, and rank these complete models under each condition (Fig. S2 *C* and *D*). Although there is some correlation between the rankings, adding additional datasets clearly identifies a smaller group of models with dynamics consistent with the data-generating model.

Parametric Bootstrap Distributions. Fig. S3 extends Fig. 4*B* from the main text, by showing bootstrap distributions for all parameters present in the true model.

Bayesian Inference. Additional Bayesian inference results are given here for the best close models selected by nested sampling—those with estimated evidence (Table S1) greater than or equal to the true model (model C, Fig. 2), and differing by just a single edge. Posterior samples were obtained using two algorithms, Metropolis–Hastings and nested sampling, with the same artificial dataset.

Selection of Models for Yeast Gene Expression Data. We constructed an initial candidate model for the dynamics of clustered yeast gene expression profiles (data from ref. 9) using the network inference approach described by Lu et al. (10), as implemented in D-NetWeaver (11). We sampled ODE models with X edges randomly rewired, relative to this initial candidate model, to find alternative models with consistent dynamics. To do this, X nonzero entries in connectivity matrix A (each corresponding to an interaction) were chosen at random to delete, and replaced by new interactions absent from the initial model; in this way model complexity (measured as the total number of edges and parameters) remained constant across all tested models.

We sampled 5×10^4 rewired models for each of $X = 1, 2, 3, 5, 10, 20,$ or 30 rewired edges, and estimated the associated parameters by gradient matching [using GP regression estimates of concentrations $\hat{\mathbf{x}}(t)$ and derivatives $\dot{\hat{\mathbf{x}}}(t)$, calculated from trajectories simulated from the initial candidate model]. The top 50 models within each rewiring category were selected based on log-likelihood values calculated using the gradient-matching parameter estimates (θ_{GM}); this provided a group of 350 models, in addition to our initial candidate model, for further analysis.

To assess the robustness of parameter estimation to topological alterations, we then used constrained optimization (using the fmincon function in MATLAB, with parameter bounds of ± 5, and initializing the algorithm at the corresponding gradient-matching parameter estimates, θ_{GM}) to obtain maximum likelihood estimates of the parameters for each of the 351 models, by fitting to the cluster means. We denote these estimates by θ_{OPT1}.

To allow for the possibility that, for some of the models, the optimization algorithm may only have converged to a local optimum, we performed a second round of optimization for each model, using alternative starting points. To obtain "good" alternative starting points for each model, we made use of the set of all 351 estimates θ_{OPT1} (one for each model), and took the median for each parameter (calculated from θ_{OPT1}, over all of the 351 models in which it appears). We denote the resulting estimates by θ_{OPT2}.

Finally, for each model we chose whichever of θ_{OPT1} and θ_{OPT2} had the higher likelihood value (in most cases this was θ_{OPT1}, but in a few cases θ_{OPT2} provided a marginal improvement).

Despite performing a restart of the optimization algorithm, we still cannot be entirely certain that the final parameter estimate

for each model corresponds to the true global maximum likelihood estimate. This problem is ubiquitous in maximum likelihood estimation problems. As a result of this, we cannot strictly say that we are performing a topological sensitivity analysis of the true (global) maximum likelihood parameter estimate. Instead, we are performing a TSA of the parameter estimates provided by an algorithm that targets the true maximum likelihood estimate (MLE). This accurately reflects what will generally be possible in practice, because (for any realistic practical problem) all that will be available is the output of such an algorithm, which cannot be guaranteed to be equal to the global MLE. TSA as performed here then lends increased credibility to those model features that are supported by all or, more likely, the majority of models. Any aspect that is only displayed by a few of the models, by contrast, ought to either be viewed with skepticism or be investigated further, ideally in carefully designed experiments (12, 13).

1. Rasmussen CE, Nickisch H (2010) Gaussian processes for machine learning (GPML) toolbox. *J Mach Learn Res* 11:3011–3015.
2. Rasmussen CE, Williams CKI (2006) *Gaussian Processes for Machine Learning* (MIT Press, Cambridge, MA).
3. Murphy KP (2012) *Machine Learning. A Probabilistic Perspective* (MIT Press, Cambridge, MA).
4. Skilling J (2006) Nested sampling for general Bayesian computation. *Bayesian Anal* 1(4):833–860.
5. Johnson R, Kirk P, Stumpf MPH (2014) SYSBIONS: Nested sampling for systems biology. *Bioinformatics*, 10.1093/bioinformatics/btu675.
6. Hastings WK (1970) Monte Carlo sampling methods using Markov chains and their applications. *Biometrika* 57(1):97–109.
7. Wilkinson DJ (2011) *Stochastic Modelling for Systems Biology* (CRC Press, Boca Raton, FL), 2nd Ed.
8. Statisticat LLC (2014) LaplacesDemon: Complete Environment for Bayesian Inference. www.bayesian-inference.com/software, R package Version 14.06.23.
9. Spellman PT, et al. (1998) Comprehensive identification of cell cycle-regulated genes of the yeast *Saccharomyces cerevisiae* by microarray hybridization. *Mol Biol Cell* 9(12):3273–3297.
10. Lu T, Liang H, Li H, Wu H (2011) High dimensional ODEs coupled with mixed-effects modeling techniques for dynamic gene regulatory network identification. *J Am Stat Assoc* 106(496):1242–1258.
11. Wu S, Liu Z-P, Qiu X, Wu H (2014) Modeling genome-wide dynamic regulatory network in mouse lungs with influenza infection using high-dimensional ordinary differential equations. *PLoS ONE* 9(5):e95276.
12. Liepe J, Filippi S, Komorowski M, Stumpf MPH (2013) Maximizing the information content of experiments in systems biology. *PLOS Comput Biol* 9(1):e1002888.
13. Silk D, Kirk PDW, Barnes CP, Toni T, Stumpf MPH (2014) Model selection in systems biology depends on experimental design. *PLOS Comput Biol* 10(6):e1003650.

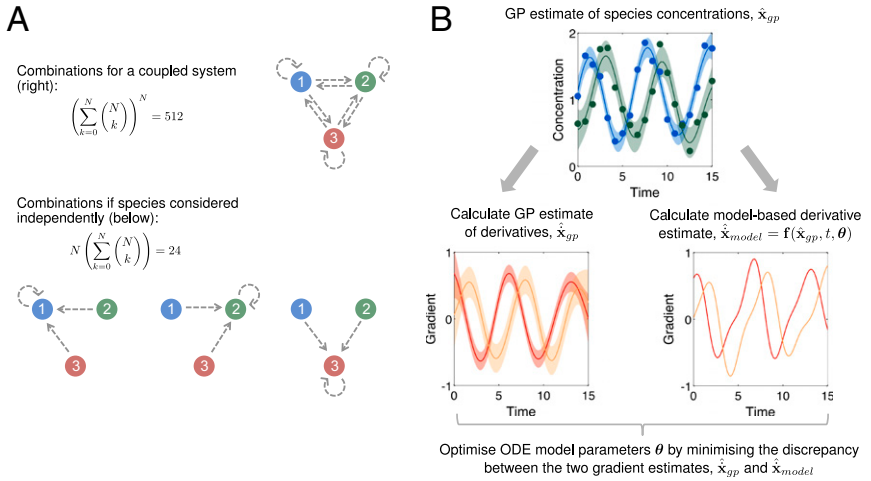

Fig. S1. Reducing combinatorial complexity using gradient-matching parameter estimation. (*A*) Considering the regulation of each species independently reduces the search space of possible models. For a system with 3 species and interactions possible between any pair of species (including self-interaction) there are 512 possible topologies for the complete network. If we always consider the complete ODE system (i.e., the complete set of parent sets, $\mathcal{C}$) we would need to test 512 models to do an exhaustive search. We can reduce this search space by considering the possible parent sets $\mathrm{Pa}(x_n)$ for each species n independently; for this example there are 8 possible parent sets for each species so we only need to test 24 models to search all possible network topologies. We obtain the overall network topology by combining a selected parent set for each species to obtain the complete set $\mathcal{C}$. (*B*) Overview of the gradient-matching parameter estimation method. GP regression models are fitted to time course data (circles) for all species, in this case two, to provide estimates of species concentrations $\hat{\mathbf{x}}_{gp}(t)$. GP estimates for the corresponding derivatives $\dot{\hat{\mathbf{x}}}_{gp}(t)$ are also calculated. A second model-derived estimate of the derivatives $\dot{\hat{\mathbf{x}}}_{model}(t)$ is calculated using the GP estimates of species concentrations and the ODE model, $\mathbf{f}(\hat{\mathbf{x}}_{gp}, t, \boldsymbol{\theta})$. ODE model parameters $\boldsymbol{\theta}$ are estimated by minimizing the discrepancy between the data-driven and model-driven estimates of the derivatives ($\dot{\hat{\mathbf{x}}}_{gp}$ and $\dot{\hat{\mathbf{x}}}_{model}$, respectively) using a constrained optimization algorithm or linear regression as appropriate (the fmincon or fitlm MATLAB functions, respectively).

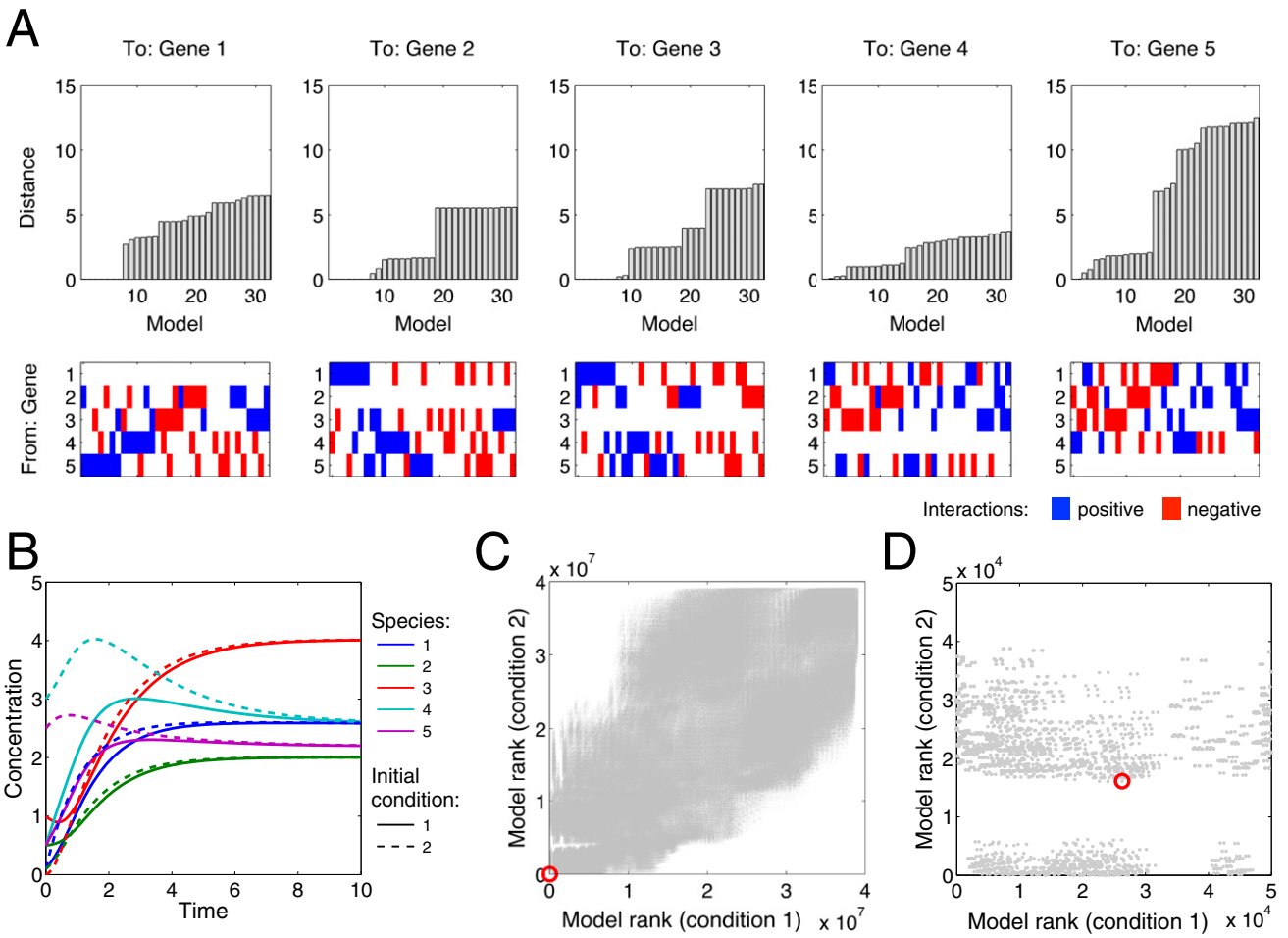

Fig. S2. Example of automated model generation and ranking for a five-species gene regulatory network (model A). (*A*) Rankings of models describing the regulation of each species 1–5, calculated using the GP regression models illustrated in Fig. 3 of the main text. For each species, 33 models were tested during an exhaustive search of all possible models within the chosen rules—a maximum of 2 parents, and 2 possible types of interaction (activation or inhibition). Bar graphs show the minimized distance obtained during gradient-matching parameter estimation for each model. The edges present in each model are indicated by the colored plots below the bar graphs (activating interactions are shown in blue, and inhibiting interactions in red). (*B*) Trajectories simulated from model A using two different initial conditions (*SI Materials and Methods*). Data for all species at 40 equally spaced time points were used to test potential ODE models. (*C*) Comparison of complete ODE model rankings for each initial condition dataset. Each model is indicated by a gray dot, with the ranks of the true model shown by a red circle. Models were ranked by AIC_c weights calculated for all possible complete ODE models (total number of possible models is $33^5 = 3.9 \times 10^7$). (*D*) Enlarged section of Fig. S2*C* to show the ranks of the model used for simulation. Note that, as in this example, the true model may not rank first because parameter estimation relies on the GP regression mean estimates $\hat{x}_n(t)$ and not the exact noiseless values $x_n(t)$.

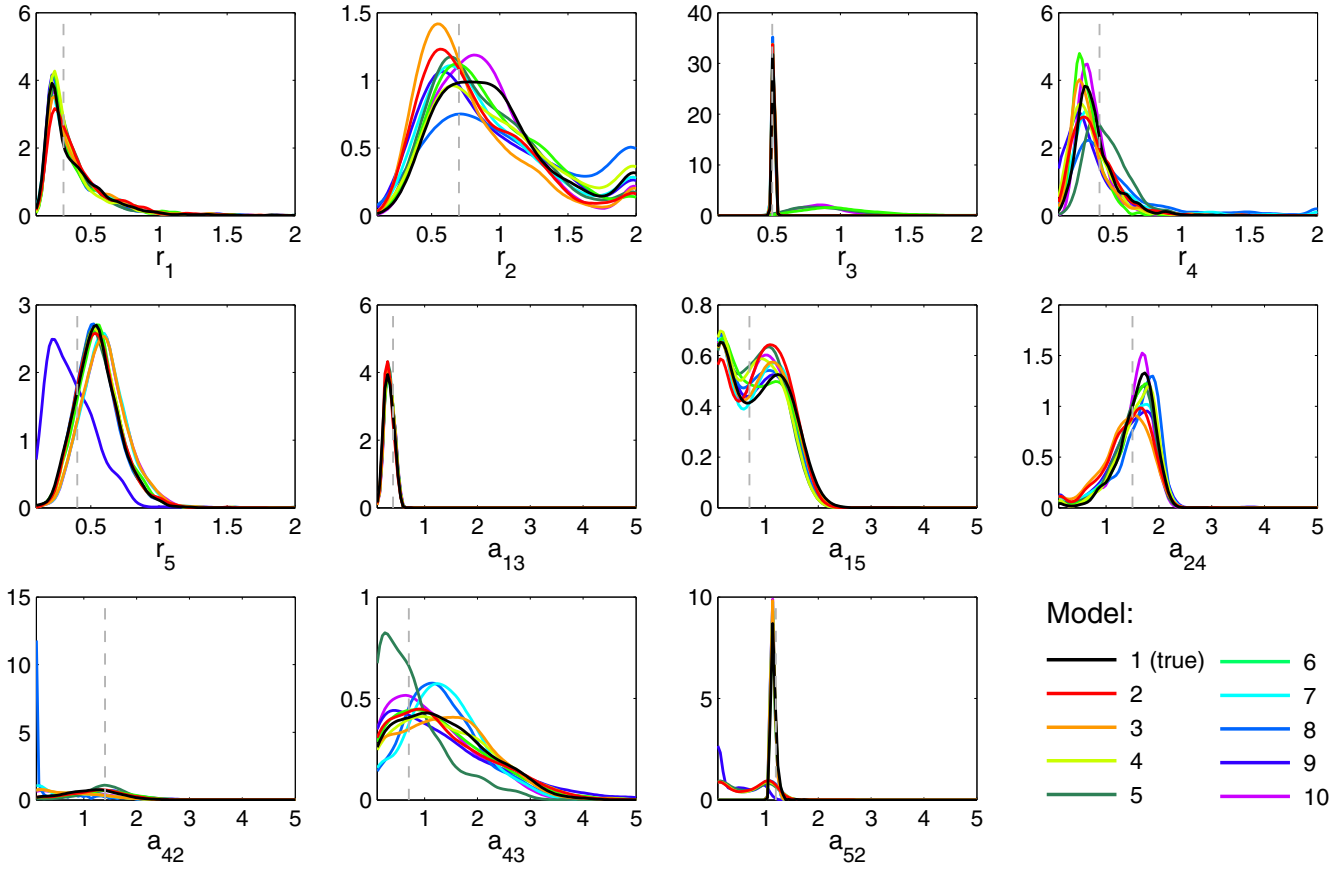

Fig. S3. Comparison of parametric bootstrap distributions obtained for the parameters present in the true model (model 1) using the top 10 complete ODE models (ranked by AIC_c values). Solid lines indicate kernel density estimates of the distributions obtained for each of the alternative model structures; vertical gray lines indicate the true model parameters used to simulate the noisy dataset. Horizontal axes are limited to the parameter ranges allowed during constrained optimization.

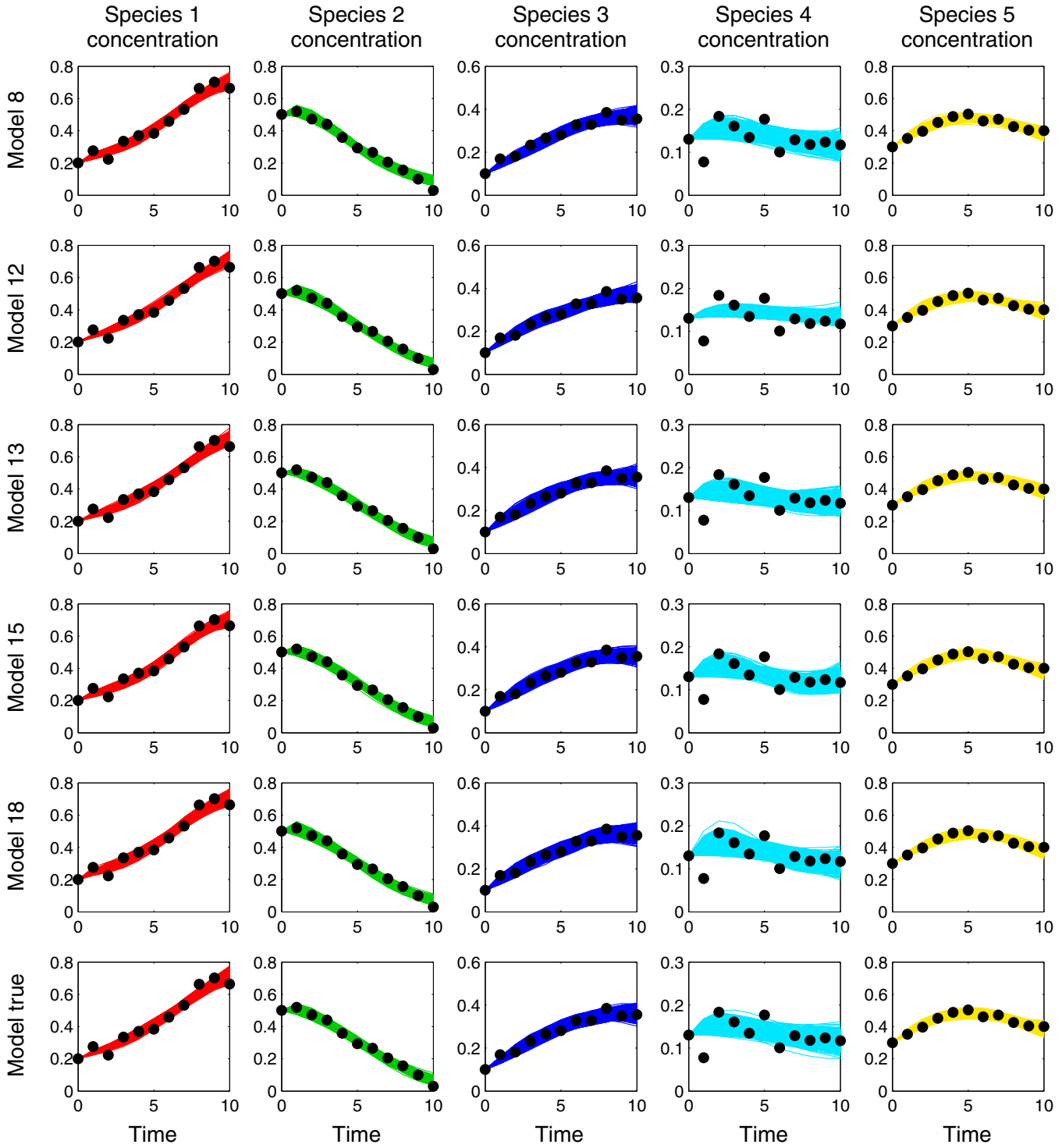

Fig. S4. Nested sampling posterior predictive trajectories. Concentration trajectories for the five species in model C were simulated using the posterior samples obtained by nested sampling for each selected alternative model. Graphs are arranged with the data for each species in a different column, and each model in a different row; black circles show the noisy synthetic dataset D_C used for inference.

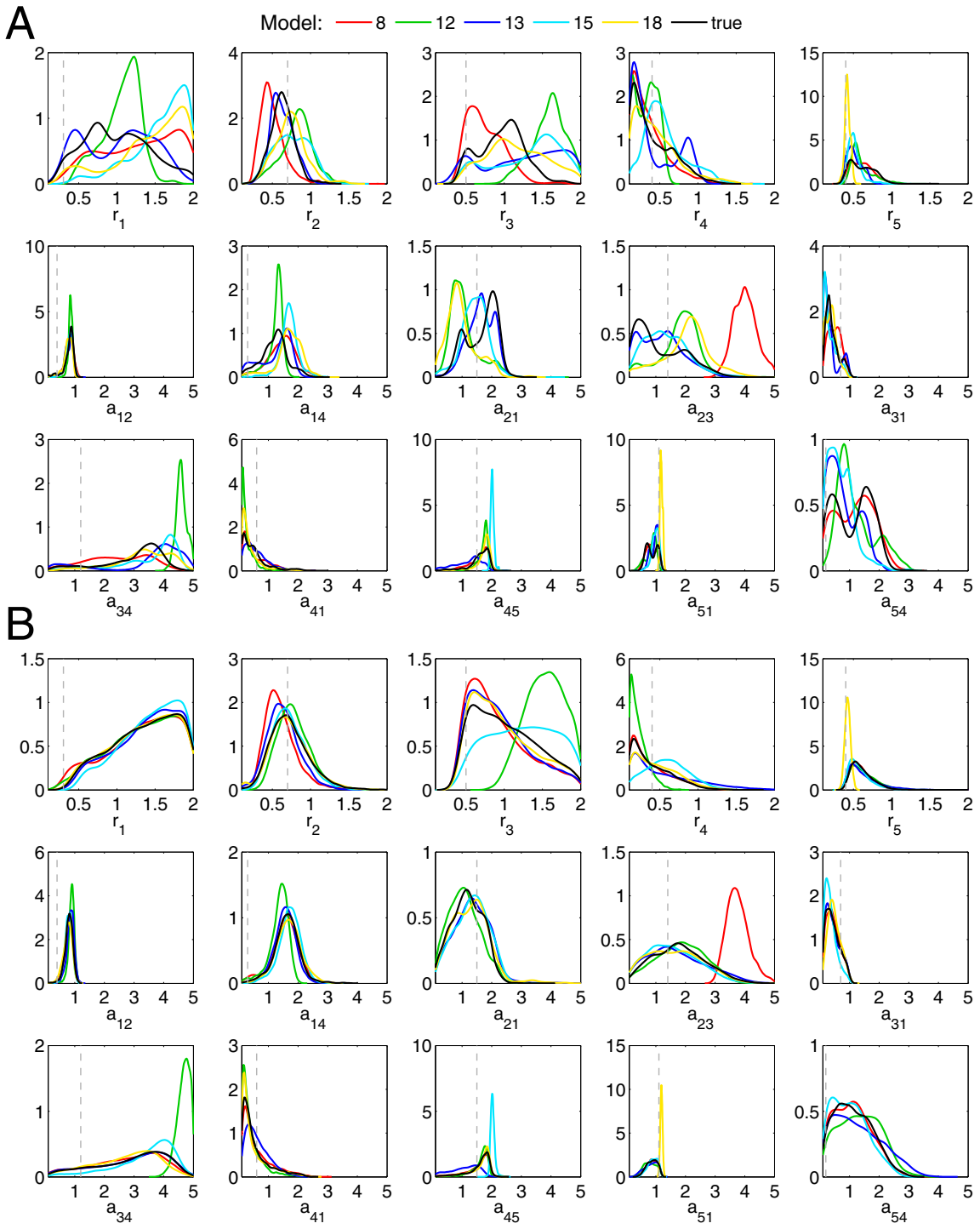

Fig. S5. Comparison of marginal posterior parameter distributions obtained for different models using (*A*) nested sampling or (*B*) Metropolis–Hastings algorithms. Kernel density estimates (colored solid lines) are shown for the marginal posterior distributions estimated using each of the selected models for parameters present in the true model; true parameter values are indicated by dashed gray lines.

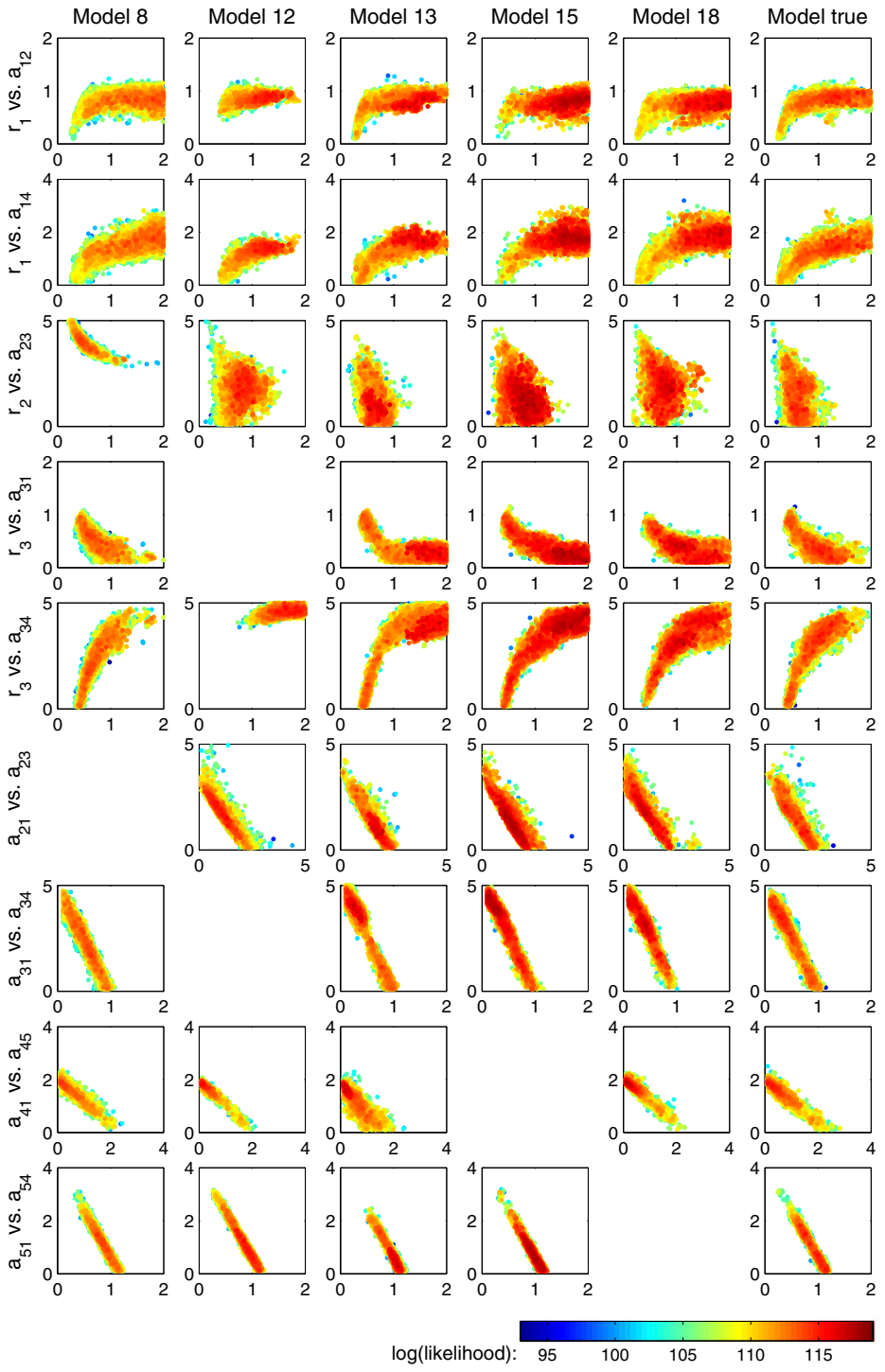

Fig. S6. Comparison of nested sampling pairwise posterior parameter estimates for different models illustrating observed dependencies. Example bivariate scatter plots for the posterior samples obtained for the best models using nested sampling. Each sample is shown by a circle, with the color corresponding to the likelihood value (ranging from blue to red in ascending order); missing plots indicate that a particular model does not contain that pair of parameters. Consistent pairwise parameter dependencies were also inferred using the Metropolis–Hastings algorithm.

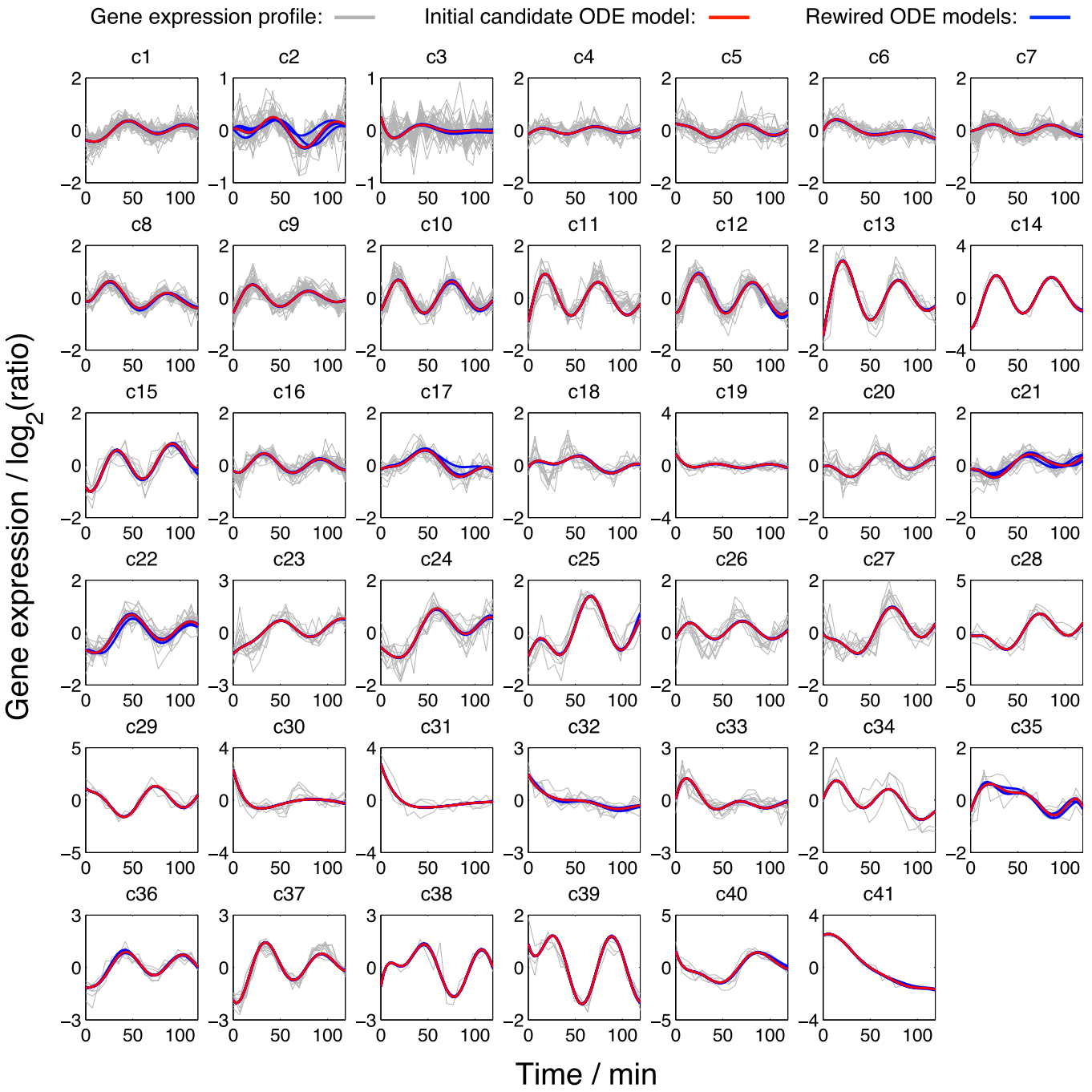

Fig. S7. Comparison of selected linear ODE models describing the dynamics of cell-cycle regulated yeast genes. Clustered gene expression profiles for yeast genes are shown in gray (α-factor synchronized expression data from ref. 9). Maximum likelihood point estimates were obtained by fitting models to the mean expression profile for each gene cluster (clusters labeled "c1–c41"); simulated trajectories from the optimized models are shown in red (initial candidate model) and blue (seven selected rewired models). The rewired models include the best model from each tested level of rewiring (models with 1, 2, 3, 5, 10, 20, or 30 edges rewired at random relative to the initial candidate model).

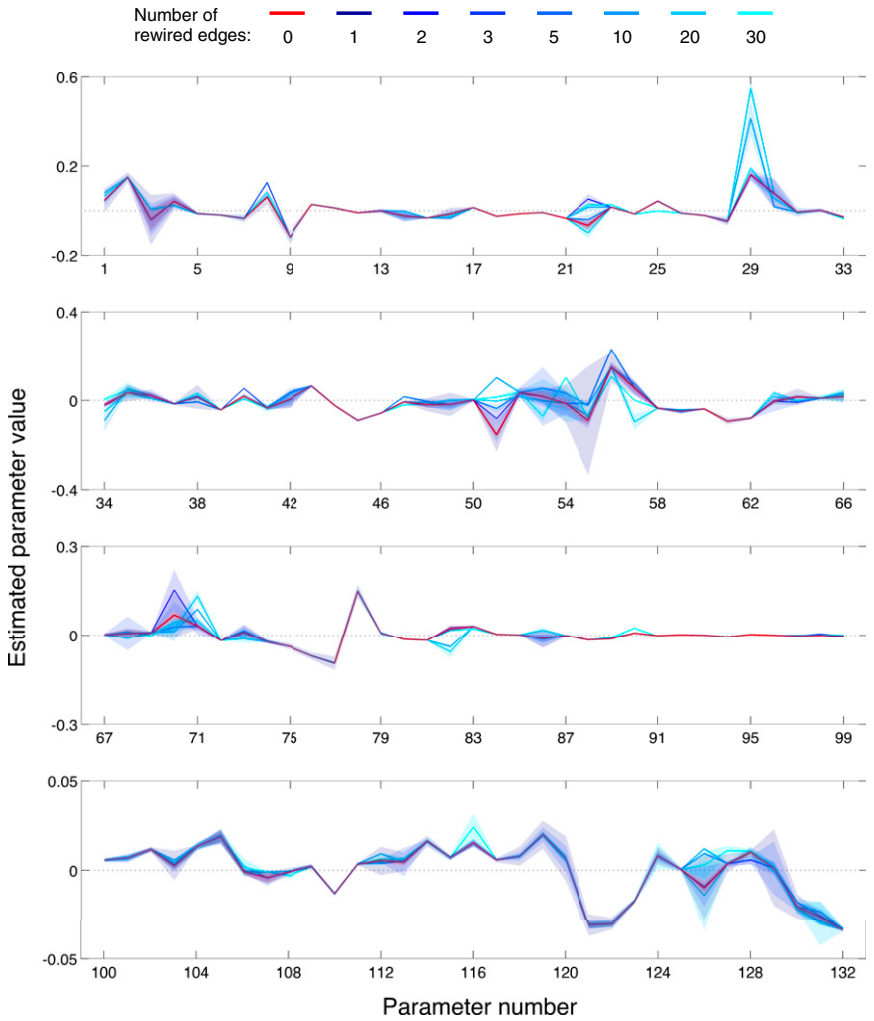

Fig. S8. Comparison of parameter estimates obtained using alternative models for clustered yeast gene expression data. Point estimates were obtained using maximum likelihood estimation for each of the eight selected models (initial candidate and seven rewired models). Graphs show the values estimated using the different models for 132 equivalent parameters that are common to all selected models (i.e., parameters corresponding to the same interaction or constant term). Each line indicates the optimized values obtained for these parameters using one of the models; shaded regions indicate 95% confidence intervals obtained using a Laplace approximation (8)—we have chosen to represent discrete data as continuous in this case for clarity (overlaid error bars would be hard to distinguish).

Table S1. Nested sampling evidence estimates for the best models (those with evidence greater than or equal to the true model)

Model	Evidence, ln Z ($\pm$1 SD)
15	81.9 ($\pm$0.1)
13	81.5 ($\pm$0.1)
12	80.2 ($\pm$0.1)
18	80.1 ($\pm$0.1)
True	79.4 ($\pm$0.1)
8	79.4 ($\pm$0.1)