Model Discrimination Using Data Collaboration†

Ryan Feeley,*,‡,§ Michael Frenklach,*,‡ Matt Onsum,‡ Trent Russi,‡ Adam Arkin,|| and Andrew Packard‡

Department of Mechanical Engineering, University of California, Berkeley, California 94720-1740, and Howard Hughes Medical Institute, Department of Bioengineering, University of California, Berkeley, California 94710, and Physical Biosciences Division, Lawrence Berkeley National Laboratory, Berkeley, California 94720

Received: November 1, 2005; In Final Form: January 11, 2006

This paper introduces a practical data-driven method to discriminate among large-scale kinetic reaction models. The approach centers around a computable measure of model/data mismatch. We introduce two provably convergent algorithms that were developed to accommodate large ranges of uncertainty in the model parameters. The algorithms are demonstrated on a simple toy example and a methane combustion model with more than 100 uncertain parameters. They are subsequently used to discriminate between two models for a contemporarily studied biological signaling network.

1. Introduction

The scientific method of establishing a reaction mechanism consists of performing pertinent experiments, postulating a reaction mechanism, and comparing the predictions of the mathematical model describing this mechanism to the experimental observations. Often, there are several competing hypotheses advanced to explain the same observed phenomena, and one is faced with the problem of discriminating among them.

Discrimination or selection among competing reaction models begins with consideration of mutual consistency of the individual thermochemical assignments and their harmony with the kinetic theory.[1] One then asks how well each reaction model describes given experimental observations. The usual mathematical approach to answer the latter question evokes a variation of the least-squares or statistical inference.[2−4]

Recently, we formulated a methodology of *data collaboration*[5−7] aimed at the analysis of complex systems with differential-equation models and heterogeneous datasets and demonstrated this formulation on complex reaction systems. The formulation is based on the concept of a dataset,[6,7] which formally integrates pertinent experimental data and mathematical process models. One of the questions addressed was dataset consistency—motivating a measure that quantifies the degree to which the process model predicts the data.[7]

In the present study we apply the measure of consistency to the problem of model discrimination. Previously we have examined datasets for which a good amount of detail on the process models was available. In the model discrimination setting, less confidence is placed on the individual candidate models. This often manifests itself in large intervals of uncertainty in model parameters. We demonstrate two new algorithms that allow our previously introduced techniques to more readily accommodate such large parameter ranges.

We begin by briefly reviewing the basics of the data collaboration method. After that we formulate the model discrimination problem and describe the applicable mathematical details and numerical techniques. We then present three examples of the approach, one being a toy example and two considering real-world problems, one from the field of combustion chemistry and the other from systems biology. We conclude with a brief summary. The notations used in this work are listed in Table 1.

2. Data Collaboration

Data collaboration is a method that unites process models and associated admissible parameter values with experimental data and accompanying uncertainties. Heterogeneous data obtained from various sources are integrated through models that depend on common parameters. This ensemble is called a dataset. Optimization driven techniques can then be applied to dataset analysis, including performing dataset based predictions,[5] determining dataset consistency,[7] and the here addressed topic, discriminating between candidate system models.

2.1. Dataset. As before,[7] we associate with each experiment a dataset unit, (d, u, M), where d is the measured value, u the reported uncertainty, and M a mathematical model of the experiment. The true value of the experimental observable, y, satisfies $|d - y| \leq u$. The model M depends on only the parameter vector to generate a prediction for y, and is often developed from a more general system model by fixing the initial conditions and inputs to the conditions of the particular experiment.

In the following, the collection of dataset units comprise the dataset D. The constant m will denote the number of units in a dataset and the subscript e will be employed to index dataset units and their associated components; n will signify the number of parameters relevant for a particular dataset. The boldface $\mathbf{x}$ will denote a value of the n-dimensional parameter vector, and the ith component of this vector will be indicated by x_i. Thus, $\mathbf{x} = (x_i)_{i=1}^{n}$. The prior information concerning the parameter vector is encapsulated by a set H containing all allowable values. We assume H can be specified by componentwise confinement of the parameter vector to bounded intervals, $\mathrm{H} = \{\mathbf{x}: \alpha_i \leq x_i \leq \beta_i\}$.

† Part of the special issue "David M. Golden Festschrift".

* Corresponding author. E-mail: myf@me.berkeley.edu.

‡ Department of Mechanical Engineering, University of California, Berkeley.

§ E-mail: rfeeley@me.berkeley.edu.

|| Howard Hughes Medical Institute, Department of Bioengineering, University of California, Berkeley, and Physical Biosciences Division, Lawrence Berkeley National Laboratory.

10.1021/jp056309s CCC: $33.50
Published on Web 03/07/2006

TABLE 1: Notation and Symbols

symbol	datatype	description
M_{sys}	mathematical function	candidate kinetic reaction model; has dependency on parameters and experimental conditions
e	integer	generic index used to reference dataset units
(d_e, u_e, M_e)	dataset unit	recorded information from an experimental observation
D	dataset	collection of m dataset units
Y_e	(textual) description	description of the observable of the eth dataset unit
y_e	scalar	true value of the observable of the eth dataset unit
d_e	scalar	measured data of the eth dataset unit
u_e	scalar	upper bound on the uncertainty of d_e
M_e	mathematical function	parametrized experiment model that predicts y_e (possibly derived from M_{sys})
$\mathbf{x}$	n-dimensional vector	value of the model parameters
H	n-dimensional rectangle	set of admissible parameter values
S_e	mathematical function	surrogate model that approximates M_e over H
b_e	scalar	upper bound on $\max_{\mathbf{x}\in H}\|M_e(\mathbf{x}) - S_e(\mathbf{x})\|$
C_D	scalar	consistency measure of the dataset D (see eq 1)
$\bar{C}_D$	scalar	upper bound on the consistency measure C_D
$\underline{C}_D$	scalar	lower bound on the consistency measure C_D

2.2. Dataset Consistency Measure. In prior study,[7] we introduced a consistency measure C_D of a dataset

$$C_D = \text{maximum value of } \gamma \text{ subject to the constraints:}$$
$$\begin{bmatrix} \mathbf{x} \in H \\ d_e - (u_e - \gamma) \le M_e(\mathbf{x}) \le d_e + (u_e - \gamma), \quad \text{for } e = 1, ..., m \end{bmatrix} \tag{1}$$

The dataset is *consistent* if there exists a parameter vector $\mathbf{x}$ in the set H at which all m dataset units satisfy $|M_e(\mathbf{x}) - d_e| \le u_e$. The constraints in eq 1 indicate one can subtract C_D from each u_e and find a parameter vector at which all model evaluations match their respective data within these perturbed uncertainties. Consequently, a dataset with nonnegative C_D is consistent.

The formulation of a consistency measure is not a radical departure from the familiar weighted least-squares technique used to solve the inverse problem that fits parametrized models to data. The chief difference is specification by the former of the maximum permissible error level a priori through u_e. Operationally, computing the consistency measure involves minimizing the maximum model/data mismatch, with the consideration that mismatch below the maximum permissible error level is somewhat acceptable. Section 1 of the Appendix provides a brief derivation linking the consistency measure to least-squares analysis.

3. Model Selection and Discrimination

Given a physical system upon which various experimental measurements have been performed, one is interested in a model to make inferences from the data. In this work, we consider the situation where the researcher must select the most appropriate system model from a collection of competing candidate models.

The appropriateness of a candidate is customarily appraised by optimizing an objective or cost function that may incorporate model/data mismatch, parsimony, model robustness, and deviation of parameters from accepted values. This general optimization paradigm includes weighted least-squares approaches, maximum likelihood based methods, and various statistical strategies incorporating information theoretic criteria.[2−4,8,9] It is well-known that standard search techniques may not discover the global solution to a nonlinear optimization problem. In fact, for nonconvex problems, verifying that a solution returned by such a procedure is even locally optimal is computationally complex (Horst et al.[10] show this to be an NP-hard problem). In parameter estimation, local solutions are often accepted because they provide a "fit" model. However, in a model selection exercise, local solutions may falsely rank the candidates, so when possible, global methods should be considered.

Using the dataset consistency concept introduced in section 2, we approach the discrimination problem in the following way. Model fitness is ranked by incorporating each candidate in a distinct dataset (each describing the same experiments but employing a different system model) and evaluating C_D for each. The largest value of C_D indicates the superior candidate. As with any ranking, if there is little difference between the top performers, prudent selection, perhaps requiring additional experiments or statistical characterization of experimental errors, is needed. The C_D-based model selection approach requires only that a dataset incorporating each candidate model be formed and that each candidate is accompanied by a set of admissible parameter values.

A key characteristic of C_D is that it corresponds to a global optimum and is computable in practice, using optimization techniques we used in the past and expand in this work. For a given hypothetical model, the consistency measure approach minimizes the worst-case data/model mismatch. As such, the method is quick to point out defects (evidenced by dataset inconsistency) in a hypothetical model. This comes at the expense of high sensitivity to insufficient error bounds assigned to unsuspected outliers. The primary contribution of this work is not specification of an infallible objective function for model discrimination (which is perhaps unachievable given the breadth of applications) but instead the description of our global solution techniques. The methods for evaluating eq 1 that we present in the next section may be readily modified to incorporate alternative objective functions that are less sensitive to outliers, penalize the number of model parameters, or have other circumstance driven characteristics.

4. Solution Methods

4.1. Dataset Creation. Consider a system for which various quantities of interest Y_e ($e = 1, ..., m$) have been measured. Let M_{sys} be a candidate system model that is capable, after specification of initial conditions, inputs, and auxiliary output functions, of forming a parameter dependent prediction of each Y_e. From M_{sys} we then derive an experiment model M_e for each Y_e. As a brief example of the technique, suppose M_{sys} has a state-space representation

$$\frac{d}{dt} z_t = f(\mathbf{x}, z_t) \tag{2}$$

where $\mathbf{x}$ signifies a particular value for the uncertain parameters

and z_t denotes the state vector at time t. Suppose Y_e is specified as *the value of* z_{10} *when the initial condition* $z_0 = 0$. The model M_e is then the relation between $\mathbf{x}$ and z_{10} when the system is initialized with zero initial conditions. Obviously, more elaborate experimental conditions or output functions mapping the state trajectory to a particular experimental observable are possible.

For each of the m experimental observations, M_e is combined with the measured data d_e and the bound on the observational uncertainty u_e to realize a dataset unit (M_e, d_e, u_e). The behavior of M_{sys} at the experimentally exercised conditions is completely captured by the models M_e ($e = 1, ..., m$). Consequently, operations on the dataset, together with information on the admissible parameters of M_{sys}, can determine the compatibility of M_{sys} and the experimental data. In this work the compatibility is indicated by the consistency measure C_D (eq 1) of the dataset.

4.2. Computation of C_D. A constrained optimization with linear objective and general form constraints, such as that which defines the consistency measure in eq 1, resides in the computational complexity class termed NP-hard. In fact, adding a single indefinite quadratic constraint to a linear program (an optimization with linear objective and constraint functions) results in an NP-hard optimization.[11] Efficient (i.e., polynomial-time) solution methods for this class of problems remain undiscovered and are widely believed to not exist.[12] We approach the difficult optimization of eq 1 by approximating each process model that appears in the constraints and solving a related optimization involving these approximations and associated fitting errors. While these refinements do not alter the complexity class of the problem and incur the added difficulty of approximation, they appear to be effective in practice.

We have developed two algorithms to calculate the dataset consistency measure that rigorously treat the model approximation step. The first is a branch and bound algorithm that computes a lower bound $\underline{C}_D$ and upper bound $\bar{C}_D$ that satisfy $\underline{C}_D \le C_D \le \bar{C}_D$. These bounds may be brought arbitrarily close to each other in finitely many iterations, effectively determining C_D to any required tolerance. The second algorithm implements a trust-region strategy. This is a local search technique, and as such, can only compute a lower bound $\underline{C}_D$ on C_D. This lower bound may be useful for comparative purposes. For example, in the biological model discrimination case of section 5.3, we rank two models by showing $\underline{C}_D$ for one model was greater than $\bar{C}_D$ for the other.

4.2.1. Surrogate Model Approximations. The approximation to an experiment model M_e is called a *surrogate model* (the terms *meta-model* and *response surface* also appear in the literature) and is denoted S_e. For clarity, we restrict our discussion to quadratic surrogate models; however, the techniques generalize to polynomial surrogates of higher order. Employing surrogate models in the optimization routines offers three benefits. First, evaluation of an ordinary differential equation (ODE) based model invariably requires a computer simulation that is subject to deterministic errors of roundoff and inexact numerical integration. While the true model behavior may be smooth, this inexact evaluation introduces low-amplitude high-frequency ripples in the objective and/or constraint functions. This "noise" cripples gradient- or Hessian-based optimization routines (e.g., Newton or quasi-Newton approaches). Fitting quadratic surrogate models to experiment model evaluations tends to dampen such ill behavior. The second benefit is the (relative) ease with which off-the-shelf constrained optimization software can handle smooth algebraic functions—surrogate model evaluation and gradient assessment is computationally inexpensive. Last, tractable relaxations exist for optimizations involving polynomials.[34]

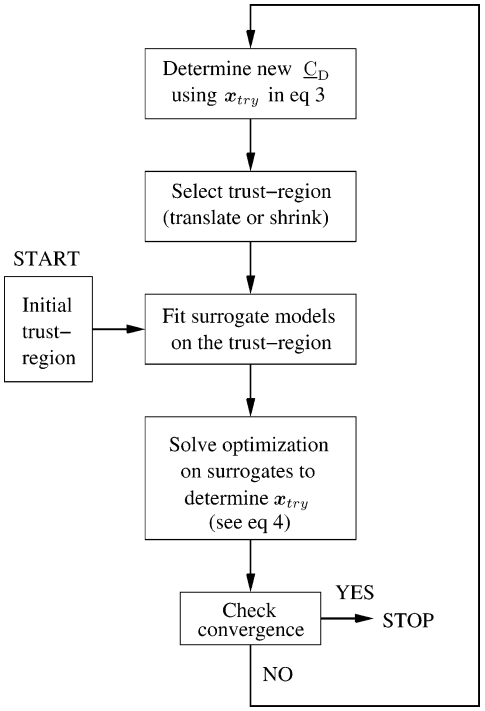

Figure 1. Program flow of the trust-region algorithm discussed in section 4.2.3. The algorithm is found in section A.3.

The approximation S_e is obtained using the response surface methodology.[13−16] This approach uses regression and computer experiments (i.e., numerical evaluations of the dynamic experiment model $M_e(\mathbf{x})$) at optimally selected combinations of the parameter values to construct the surrogate model. The entire set of parameter combinations is called a computer experiment design. These designs originate from a rigorous analysis of variance, with the objective of minimizing the number of computer experiments to be performed to gain the required information. The residuals from the regression are monitored to assess the quality of fit.

The model fit of a polynomial surrogate is improved if the degree is increased or if the approximation is sought over a smaller parameter domain. The later is exploited by the routines we present in sections 4.2.2 and 4.2.3, both based on intelligently sectioning the parameter domain to reduce the fitting error. A more complete discussion of the optimizations that result with this surrogate fitting procedure, their properties, and recently discovered solution techniques for higher order polynomial surrogates, can be found in Seiler et al.[17] and references therein.

4.2.2. Branch and Bound Algorithm for C_D. Branch and bound is a widely used technique for global optimization over a bounded or finite domain. The method is employed in numerous applications, including integer and mixed-integer linear programming and nonlinear programming.[18] The technique resolves a difficult optimization by recursively partitioning the domain over which the problem is posed into successively smaller disjoint subdivisions and examining related "easier" problems over each subdivision. As indicated by the name, the method involves two basic operations. *Branching* consists of dividing members of the subdivision collection (which may be represented as nodes in a binary tree diagram whose root node corresponds to the original problem domain; see Figure 3) into a larger collection of smaller subdivisions. *Bounding* refers to performing (presumably) tractable computations that determine

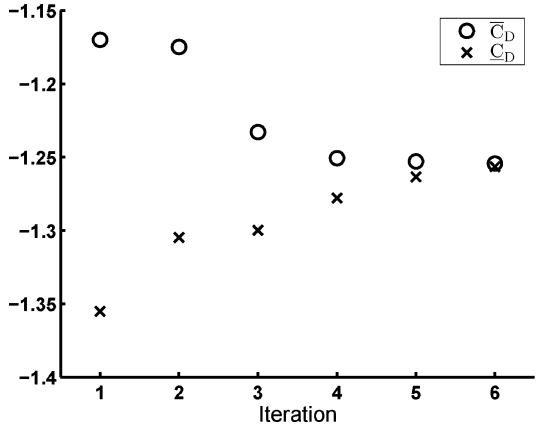

Figure 2. Convergence of the bounds on C_D for the toy example of section 5.1. Six iterations were used to compute C_D within a tolerance of 0.005. Notice $\bar{C}_D$ from the first iteration is negative, so the hypothetical model is invalidated after one iteration.

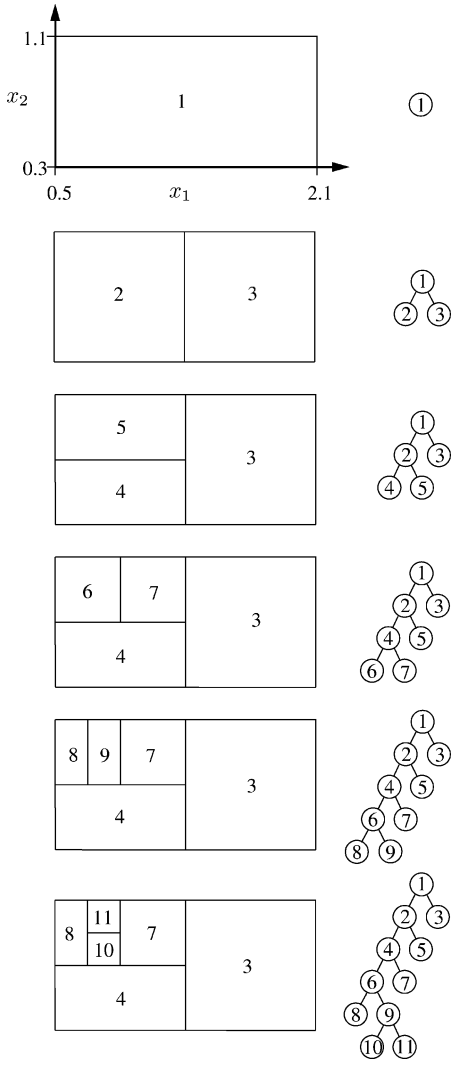

Figure 3. Nodes visited by six iterations of the branch and bound procedure as applied to the toy example of section 5.1. At algorithm termination, each node has been pruned except node 11, which contains the optimal solution.

an interval that contains the maximum (when the method is applied to a maximization) of the objective function over each subdivision. This interval is described by an upper bound $\Phi_{ub}(H_l)$ and a lower bound $\Phi_{lb}(H_l)$, where H_l denotes the particular subdivision. Any subdivision whose upper bound is below the largest lower bound among the current subdivisions cannot contain the global maximum and is discarded. This discardation step, which is traditionally called *pruning*, does not change the worst-case time complexity of the algorithm. However, in practice it imparts considerable time savings. A subdivision is *solved* if its upper and lower bounds are within a specified tolerance of one another. The procedure terminates when all undivided subdivisions (the leaves in the tree diagram representation) are either solved or pruned.

A detailed description of the branch and bound algorithm and proof of convergence to C_D is provided in section A.2 of the Appendix. Here it is pertinent to mention the method requires an upper bound on the surrogate modeling error. Specifically for each e, we assume there is a known constant b_e such that $\max_{\mathbf{x}\in H}|M_e(\mathbf{x}) - S_e(\mathbf{x})| \le b_e$. In practice, it is not possible to know such a bound with certainty. However, b_e can be estimated by assessing $|M_e - S_e|$ both at the sample points used in the fit and at any additional points generated expressly for that purpose. This estimate can be improved with local search. Another salient point is that the worst-case algorithm performance, measured in number of iterations required for termination, grows exponentially with problem size. This handicap is expected since as mentioned in section 4.2.1, consistency measure determination resides in a complexity class for which no known polynomial-time algorithm exists. Potentially exponential time performance is typical of branch and bound procedures and global methods in general.

4.2.3. Trust-Region Algorithm for C_D. The branch and bound procedure determines the consistency measure of a dataset by first examining the entirety of the parameter domain and then considering subdivisions. This creates practical difficulties in the initial iterations because the polynomial surrogate models must approximate the experiment models over the entire parameter domain. When this fit is poor (which indicates the experiment model has nonpolynomial behavior on the entire domain of H), determining an optimal approximation and accurately assessing fitting error requires numerous model evaluations. As an alternative, in this section we introduce a trust-region based algorithm that produces a lower bound on the consistency measure. This technique creates piecewise fits over subsets of H, analogous to the technique employed by the PRISM method.[19] This is not a global optimization method and produces only a lower bound $\underline{C}_D$.

The dataset consistency measure has the equivalent definition

$$C_D = \max_{\mathbf{x}\in H} h(\mathbf{x}), \quad \text{where } h(\mathbf{x}) = \min_{e=1,\ldots,m} \{u_e - |M_e(\mathbf{x}) - d_e|\} \quad (3)$$

that is better suited for the discussion of this section. The trust-region algorithm is initialized by selecting a rectangular subset H_{tr} of H that is sufficiently small for quadratic surrogates to accurately fit each experiment model. H_{tr} is called the *trust-region* because it is the domain over which we tentatively trust the validity of the approximations. Evaluating h in eq 3 at the center point $\mathbf{x}_c$ of H_{tr} provides an initial lower bound on C_D. Next we use a local search to compute

$$\mathbf{x}_{try} = \operatorname*{argmax}_{\mathbf{x}\in H_{tr}} \min_{e=1,\ldots,m} \{u_e - |S_e(\mathbf{x}) - d_e|\}, \quad (4)$$

where for $e = 1, \ldots, m$, S_e is a surrogate model for M_e that is valid over the trust-region H_{tr}. This provides a trial point $\mathbf{x}_{try}$ at which h can again be evaluated. If $h(\mathbf{x}_{try}) > h(\mathbf{x}_c)$, the lower bound on C_D has improved, so H_{tr} is replaced with a new slightly

larger trust-region centered about $\mathbf{x}_{try}$ and the procedure is repeated. If not, this indicates poor surrogate approximation over H_{tr}, so H_{tr} is made smaller, model fits are regenerated, and a new trial point is computed via eq 4. The schematic in Figure 1 depicts the flow of the algorithm.

Trust-region methods have enjoyed considerable research activity; see Conn et al.[20] for a comprehensive review. In Elster and Neumaier[21] the idea is applied to noisy objective functions. Our implementation breaks from tradition in that the function approximations are made using the response surface methodology discussed in section 4.2.1 rather than from the first few terms of the Taylor approximation. This produces approximations with larger regions of validity and correspondingly larger trust-regions.

The primary benefit of the trust-region algorithm over off-the-shelf constrained optimization software lies in the use of algebraic surrogate models in place of the ODE-based experiment models. By design, the surrogate models only involve parameters that have reasonable impact on the model behavior (so-called active variables), so the optimizations on the surrogate models involve fewer variables and are better conditioned. Also, the Jacobian and Hessian of the surrogate models may be cheaply computed. Additionally, model approximation has a smoothing affect that alleviates the aforementioned ripples that may be introduced into the experiment models by numerical integration. While these ripples can be attenuated by stringent integration tolerances, the trust-region method bypasses that additional computational expense.

5. Examples of Applications

5.1. Toy Example. Our first case is a toy problem that is similar to example 4 from the work of Prajna.[22] The example features two parameters and is used to illustrate the branch and bound algorithm for C_D in a situation where parameter domain subdivisions and surrogate models are easily visualized. Consider a system with the hypothetical differential equation model

$$\frac{d}{dt}z(t) = -x_1 z(t)^3 \tag{5}$$

From prior information it is known that $0.5 \le x_1 \le 2.1$. We suppose three measurements have been taken on this system, one establishing that the initial state $z(0)$ lies in the interval [0.3, 1.1] another determining $z(2)$ lies in [0.55, 0.65], and the third finding $z(4)$ in [0.2, 0.3]. With these data, Prajna used a novel barrier function approach[22] to show the hypothetical model was invalid. We use the branch and bound algorithm introduced in section 4.2.2 to determine a consistency measure for this model/data ensemble.

We now assemble a dataset incorporating the hypothetical model. First consider the measurement of $z(2)$. A predictive model for the state at time $t = 2$ requires knowledge of both the differential equation (eq 5) and the state at some time point. Conveniently, knowledge that the initial state lies in the interval [0.3, 1.1] is available. We incorporate this measurement as an additional uncertain parameter, defining $x_2 = z(0)$, and noting $0.3 \le x_2 \le 1.1$. A model for $z(2)$ as a function of the uncertain parameters x_1 and x_2 is then

$$M(x_1, x_2) = z(2), \quad \text{subject to } z(0) = x_2, \quad \frac{d}{dt}z(t) = -x_1 z(t)^3 \tag{6}$$

From this we assemble a dataset unit $U_1 = (d_1, u_1, M_1)$ where $d_1 = 0.6$, $u_1 = 0.05$, and M_1 is described by eq 6. The dataset unit U_2 for the measurement of $z(4)$ is defined similarly. The prior information is the set $H = \{(x_1, x_2) \in R^2: 0.5 \le x_1 \le 2.1 \text{ and } 0.3 \le x_2 \le 1.1\}$.

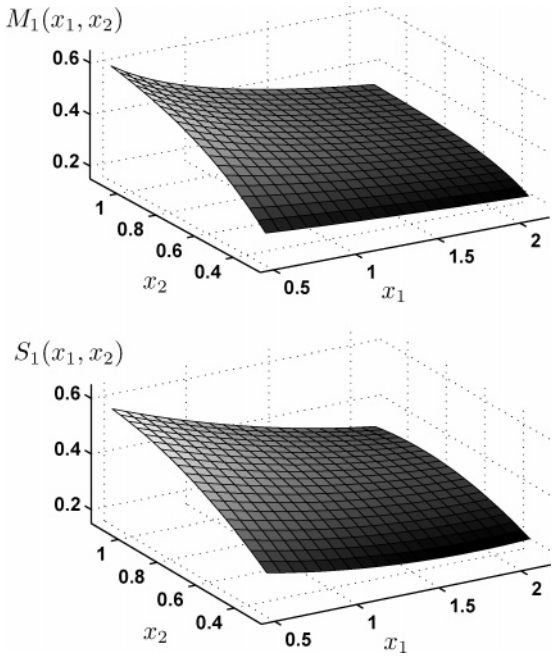

Figure 4. Response surface of M_1(top frame) and the quadratic approximation S_1(bottom frame). The approximation satisfies $|S_1(\mathbf{x}) - M_1(\mathbf{x})| \le 0.0125$ for all $\mathbf{x}$ in H. The branch and bound algorithm partitions H into subdivisions, reducing model approximation errors.

TABLE 2: Iteration Summary for Toy Branch and Bound Example

iteration no.	node no.	$\Phi_{lb}(H_l)$	$\Phi_{ub}(H_l)$	$b_1{}^l(\times 10^{-3})$	$b_2{}^l(\times 10^{-3})$
$k = 1$	$l = 1$	−1.355	−1.170	12.5	10.8
$k = 2$	$l = 2$	−1.305	−1.175	8.30	7.44
	$l = 3$	−2.872	−2.806	5.89	3.89
$k = 3$	$l = 4$	−1.638	−1.558	2.50	2.82
	$l = 5$	−1.300	−1.233	3.30	2.43
$k = 4$	$l = 6$	−1.278	−1.251	1.19	0.795
	$l = 7$	−1.510	−1.486	0.637	0.318
$k = 5$	$l = 8$	−1.469	−1.443	0.538	0.474
	$l = 9$	−1.264	−1.253	0.464	0.238
$k = 6$	$l = 10$	−1.364	−1.362	0.110	0.0778
	$l = 11$	−1.256	−1.254	0.0925	0.0717

Six iterations of the branch and bound algorithm presented in section 4.2.2 determined C_D for this dataset to be −1.254 with a tolerance $\epsilon = 0.005$. Figure 2 displays convergence of the upper bound $\bar{C}_D$ and lower bound $\underline{C}_D$ with iteration number. These bounds draw together as H is subdivided, reflecting a reduction in surrogate model fitting error. The subdivision of H generated by the algorithm is illustrated in Figure 3. Figure 4 displays a graph of M_1 and the quadratic approximation S_1 that was fit over H. The approximation is reasonably accurate, but visibly differs from M_1. As H is subdivided by the algorithm, the accuracy of this approximation will improve. This is depicted in Table 2, which contains both the surrogate fitting error bounds and Φ_{lb} and Φ_{ub} evaluated at each subdivision. 336 evaluations of the experiment models M_1 and M_2 were required to compute C_D to this tolerance. However the procedure demonstrated C_D was negative in the first iteration, which required only 48 evaluations of the two models.

5.2. GRI-Mech Dataset. We demonstrate the trust-region algorithm on a real-world example, the GRI-Mech 3.0 dataset,[23] taken from the field of combustion chemistry. It is a collabora-

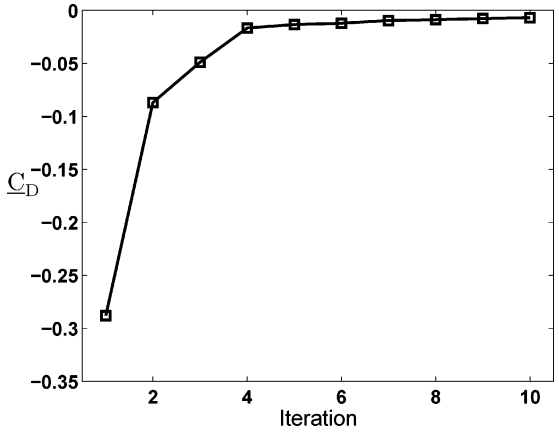

Figure 5. $\underline{C}_D$ vs iteration count for the GRI-Mech 3.0 dataset described in section 5.2.

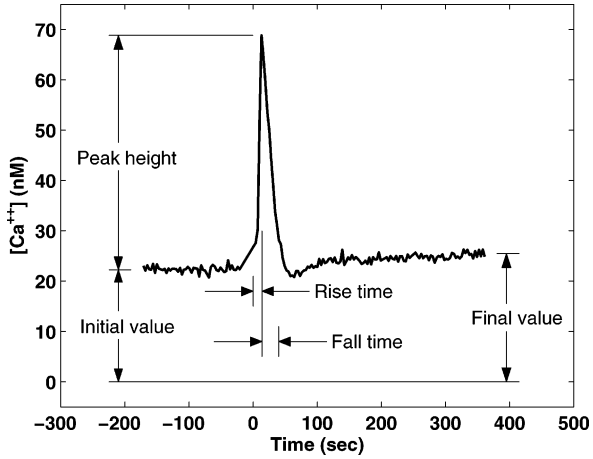

Figure 6. Representative time-trace of the calcium data, indicating the five derived data features. This plot was obtained at a ligand dose of 250 nM C5a.[26]

tive data repository for the development of detailed kinetic models for pollutant formation in the combustion of natural gas, and was used in our previous studies.[5−7] In the present case, we use the GRI-Mech 3.0 dataset to test the trust-region algorithm for $\underline{C}_D$ in a situation where we know the correct answer a priori. The example provides a frame of reference from which we can assess the algorithm's performance on the biological application of the next subsection that features a large H and ODE-based experiment models.

Briefly, the GRI-Mech 3.0 dataset is composed of 77 dataset units and 102 active parameters (see refs 6, 7, and 23 for details). High fidelity quadratic approximations for the kinetic model at the conditions of each dataset unit are available, and for the purpose of this example, we treat these as the "true" models M_e ($e = 1, ..., 77$). In a prior study examining this dataset with the quadratic models,[7] we employed this constrained optimization routine FMINCON in the MATLAB programming environment to determine $\underline{C}_D = -0.0065$ at $u_e = 0.08$ (other means found $\bar{C}_D = -0.00\overline{33}$). We validated the trust-region algorithm described in section 4.2.3 on this dataset, using linear surrogate models to approximate the quadratic experiment models. The resulting lower bound on C_D essentially duplicated the lower bound found in our prior work by examining the quadratic models directly, determining $\underline{C}_D = -0.0068$ in 13 iterations. This value of $\underline{C}_D$ indicates that addition of 0.0068 to each u_e makes the dataset consistent. Additionally, we note that in only four iterations, the algorithm improved $\underline{C}_D$ from -0.288 to -0.017. Figure 5 shows how this lower bound on C_D increases with iteration count.

5.3. Calcium Mobilization Model Discrimination. We now describe a real-world example for which we do not know the results a priori. It is taken from a biological signaling application that triggered development of the methods described in this work. We consider two proposed models and show that one better reproduces the experimental data.

In the response to an extracellular stimulus (e.g., presence of a signaling chemical, electric potential, or mechanical stimulation), eukaryotic cells display a variety of responses, including chemotaxis (movement along the concentration gradient of a chemotactic agent), phagocytosis (engulfing foreign matter), altered gene expression, and secretion. These cellular behaviors are triggered through multilayered signaling pathways that involve a myriad of signaling molecules. Signal transduction along such pathways is a key regulatory mechanism for most biological processes and is an area of active research. A growing body of quantitative cellular signaling data is being generated by the biological community (though it is often difficult to cross-compare data among different laboratories). One large effort, the Alliance for Cellular Signaling (AfCS), is dedicated to developing an extensive model of G-protein coupled receptor (GPCR)-mediated signal transduction in a murine macrophage cell line.[24] Application of one or more natural or pharmaceutical receptor ligands to the outside of the cell invokes a complex response in this chemically complex pathway.

The release of calcium from internal stores into the cytoplasm is a central secondary response that is experimentally accessible in living cells through calcium sensitive fluorescent dyes. Different combinations and timing of ligand application lead to different complex patterns of calcium response. The action of calcium, which occurs in a diverse array of cell types (cardiac, skeletal muscle, endothelial, pancreatic, etc.) is not fully understood, but considered to be a ubiquitous signaling phenomena.[25] It is the subject of numerous experimental and theoretical reports and has provided a fruitful area for biological modelers. One difficulty in modeling the phenomena is the variety of responses that occur across cell types, necessitating cell type specific models. It is a hope of researchers that the underlying mechanisms of the calcium release in different cell types have broad similarity and share common biomolecular components. This would allow structurally similar models (with perhaps differing parameter values) to mathematically describe the process.

Using AfCS calcium time-trace data,[26] we applied the techniques presented in this work to discriminate between two previously published calcium response models. The calcium time-traces were collected in triplicate at each of six extracellular dosage levels of the ligand C5a (an anaphylatoxin known to stimulate calcium signaling in macrophage cells). The dosage levels were: 25, 50, 100, 250, 500, and 1000nM. For each dosage level, we encapsulated the time-trace data with five scalar features: initial value, final value, peak height relative to initial value, rise time from initial value to peak, and time to fall from the peak to 5% of the offset between the peak and the final value. This generated a dataset comprised of 30 units, one from each feature/dosage combination. The five features are illustrated with a representative time-trace of the C5a-induced calcium response in Figure 6. The model for each dataset unit was a special case of the system model, obtained by specifying the initial state, input dosages, and a mathematical function relating the model predicted calcium trajectory to the relevant feature.

We now discuss our results with two previously published calcium models. However, we need to mention that these

TABLE 3: Iteration Summary for Wiesner Branch and Bound Example

iteration no.	$\bar{C}_D$
1	1.213
2	1.1715
3	1.1320
4	0.8556
5	0.6289
6	0.3907
7	0.3832
8	−0.1539

literature models were formulated for cell types differing from those on which the AfCS data was obtained and are simplified representations of the actual biochemical reaction networks. Therefore, our results should be viewed an example of our techniques and a demonstration of their potential to answer current biological questions rather than an endorsement or denunciation of a particular model.

5.3.1. A Case of an Invalidated Model. We consider first a model published in 1996 by Wiesner et al.[27] Their model describes the calcium response in umbilical vein endothelial cells to extracellular thrombin (a protease active in the blood clotting response) concentrations. This dynamic model is presented as eight coupled ODE's involving 27 uncertain parameters. The paper includes parameter values that provide an adequate fit to the thrombin-induced calcium data considered in the report. Our objective was to determine whether this model could explain the AfCS macrophage calcium data after modest adjustments in parameter values.

Initial inspection showed that dramatic changes in parameter values would be required for this model to duplicate the initial and final values of the AfCS calcium data. Scaling the model states could remedy this lack of fit, so we focused our analysis on the remaining three dynamic features: peak height, rise time, and fall time. Therefore, we used 18 of our original 30 dataset units (three features for six ligand doses). The final consideration for our investigation was the choice of H, i.e., the specification of parameter values we would admit into the analysis. Since our goal was to make only modest adjustment to the parameter values published in the Wiesner paper, we established H as a hypercube centered at the published parameter values with edge lengths varying from 20% of the center value to an order of magnitude, as we deemed biologically appropriate.

With this setup, eight iterations of the branch and bound procedure presented in section 4.2.2 determined a negative upper bound on the consistency measure of this model/data/parameter domain ensemble. The model was thus invalid, so having no need to more accurately compute C_D, we terminated the algorithm execution. Table 3 records the refinement of the upper bound on the consistency measure as a function of iteration number.

5.3.2. A Case of a Validated Model. The other calcium model we considered was presented in 2003 by Lemon et al.[28] This work was of a more theoretical nature and focused on modeling the calcium response of generic nonexcitable cells. The model was comprised of eight coupled ODE's depending on 34 uncertain parameters. In the manuscript, Lemon validated the model against published experimental data for the calcium response of human 1321N1 astrocytoma cells induced by the nucleotide UTP. Parameter values were either gleaned from the literature or adjusted to fit this published dataset.

We formed a dataset coupling the Lemon model to the same 18 feature/dosage combinations of the AfCS calcium data used with the Wiesner model. We then employed the trust-region algorithm presented in section 4.2.3 on these dataset and found a lower bound $\underline{C}_D = 0.131$. Compared with C_D for the Wiesner model, which had a negative upper bound, the positive lower bound from the trust-region algorithm demonstrates the Lemon model better explains the data and is thus a superior candidate by the C_D selection criterion. Figure 7 displays the improvement of $\underline{C}_D$ with iteration count.

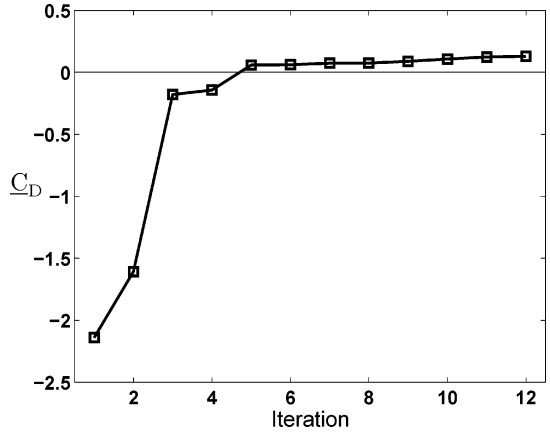

Figure 7. $\underline{C}_D$ vs iteration count for the Lemon model based calcium dataset described in section 5.3.2. A positive value of $\underline{C}_D$ is achieved, indicating this dataset is consistent. The Lemon model is then a superior candidate relative to the Wiesner model, which displayed a negative $\bar{C}_D$ (and thus was inconsistent).

There are two probable reasons why the Lemon model scored a higher C_D than the Wiesner model. First, we suspect that the Lemon model could more easily explain the data since it had seven more parameters than the Wiesner model. And second, recent experiments performed by the AfCS have shown the importance of receptor desensitization in the calcium response. The Lemon model incorporated receptor desensitization while the Wiesner model did not. In a future paper, we will explore if the addition of a receptor desensitization mechanism significantly improves the C_D of the Wiesner model.

6. Summary

We introduced a practical approach to discriminate among models constrained by experimental data. The technique utilizes a dataset consistency measure C_D, which is a computable global optimum that quantifies model/data mismatch. We presented two algorithms to compute or bound C_D. The first was a subdivision approach based on the well-known branch and bound global optimization technique. At its core, this is an intelligent divide-and-conquer strategy. The second approach we introduced employed a trust-region local search that culls small subsets of the parameter domain for parameter values that provide adequate data fit.

This consistency measure-based technique was effective on two real-world problems of appreciable complexity, involving tens to hundreds of model parameters. The first of these examples was from field of combustion chemistry. We studied this system in the past and here used it to validate our newly developed algorithms. The second example is the cellular signal transduction application that motivated us to to develop the methods described herein. For this problem, we discriminated between two published models and demonstrated that one more faithfully reproduced experimental data. The methods are easily parallizable, and provide a practical use of computational power to aid both the modeler and experimentalist.

Acknowledgment. David Golden was one of the principal investigators of the GRI-Mech project and currently is one of

the leading figures of the Process Informatics Model (PrIMe) initiative. Many years of collaboration, discussion, and personal friendship with one of the authors (M.F.) contributed to the philosophical aspects of scientific inference that the present approach pursues. The work was supported by the NSF, Information Technology Research Program, Grant No. CTS-0113985, and the NSF initiative on Collaborative Research: Cyberinfrastructure and Research Facilities, Grant No. CHE-0535542.

Appendix

A.1. Relation between C_D and Least Squares. In the notation of data collaboration, the weighted least-squares parameter estimation problem is the unconstrained optimization

$$\min_{\mathbf{x}\in \mathrm{H}} [\sum_{e=1}^{m} w_e (M_e(\mathbf{x}) - d_e)^2]^{0.5} \tag{A1}$$

where w_e ($e = 1, ..., m$) is a user chosen weight.

Omitting the weights for clarity, eq A1 has a solution equivalent to that of the constrained problem

$$\min_{\mathbf{x}\in \mathrm{H}} ||\boldsymbol{\delta}||_2$$

$$\text{subject to:} \quad -\delta_e \le M_e(\mathbf{x}) - d_e \le \delta_e, \quad e = 1, ..., m \tag{A2}$$

where $||\boldsymbol{\delta}||_2$ ($=\Sigma_{e=1,...,m}\delta_e^2]^{0.5}$) is the two-norm of $\boldsymbol{\delta}$, the vector of δ_e's. Relating this to C_D in eq 1 of the main text, when the constraints $|M_e(\mathbf{x}) - d_e| \le u_e$ have been scaled so that for $e = 1, ..., m$, the uncertainties u_e have a common magnitude $\tilde{u}$,

$$C_D = \tilde{u} - \min_{\mathbf{x}\in \mathrm{H};\boldsymbol{\delta}} ||\boldsymbol{\delta}||_\infty$$

$$\text{subject to:} \quad -\delta_e \le M_e(\mathbf{x}) - d_e \le \delta_e, \quad e = 1, ..., m \tag{A3}$$

where $||\boldsymbol{\delta}||_\infty$ ($= \max_{e=1,...,m}\delta_e$) is the infinity-norm of $\boldsymbol{\delta}$. The similarity between eq A2 and eq A3 is apparent—both drive down a norm of the model/data mismatch vector $(|M_e(\mathbf{x}) - d_e|)_{e=1}^m$.

A.2. Branch and Bound Algorithm for C_D. For presentation of the branch and bound algorithm, it is convenient to form functions capturing the constraints on the model parameters imparted by the dataset. Let

$$f_i(\mathbf{x}) = \begin{cases} M_e(\mathbf{x}) - d_e + u_e & \text{for } i = 2e-1 \\ -M_e(\mathbf{x}) + d_e + u_e & \text{for } i = 2e \end{cases} \tag{A4}$$

The constraints on $\mathbf{x}$ implicit in the dataset then become $0 \le f_i(\mathbf{x})$ for $i = 1, ..., 2m$. We denote by H_l and $\mathbf{x}_{c_l}$ a generic rectangular subset of the parameter domain H and its corresponding center point. The symbol S_e^l represents a surrogate model for M_e fit over H_l. Let

$$g_i^l(\mathbf{x}) = \begin{cases} S_e^l(\mathbf{x}) - d_e + u_e & \text{for } i = 2e-1 \\ -S_e^l(\mathbf{x}) + d_e + u_e & \text{for } i = 2e \end{cases} \tag{A5}$$

As in the main text, we use $\underline{C}_D$ and $\bar{C}_D$ to denote a lower and upper bound respectively on C_D.

Consider the function

$$\Phi(\mathrm{H}_l) = \max_{\mathbf{x}\in \mathrm{H}_l} \min_{i=1,...,2m} f_i(\mathbf{x}) \tag{A6}$$

Comparison with eq 1 or eq 3 of the main text shows $C_D = \Phi(\mathrm{H})$. We use the branch and bound algorithm described in section A.2.1 to determine $\Phi(\mathrm{H})$ within a specified tolerance ϵ. The principle difference between our branch and bound implementation and those commonly appearing in the literature lies in our employment of polynomial approximations g_i^l to the constraint functions f_i. This refinement enables the time-saving heuristics discussed in section A.2.2. Our base algorithm depends on two assumptions:

(a1) For $i = 1, ..., 2m$, an error bound b_i^l is available such that $\mathbf{x}$ in H_l implies $|f_i(\mathbf{x}) - g_i^l(\mathbf{x})| \le b_i^l$.

(a2) For $i = 1, ..., 2m$, the rate of change of f_i (e.g., the derivative if f_i is differentiable) is bounded by a known constant ρ_i, i.e., for all $\mathbf{x},\mathbf{y}\in\mathrm{H}, ||f_i(\mathbf{x}) - f_i(\mathbf{y})|| \le \rho_i||\mathbf{x} - \mathbf{y}||$. In section A.2.2, we propose a heuristic modification to the base algorithm that renders the second assumption (which is required for provable convergence) unnecessary in practice.

Proposition 1. *Suppose there are functions Φ_{lb} and Φ_{ub} defined on all rectangles $\mathrm{H}_l \subset \mathrm{H}$ satisfying the following points:*

1. For every $\mathrm{H}_l \subset \mathrm{H}$, $\Phi_{\mathrm{lb}}(\mathrm{H}_l) \le \Phi(\mathrm{H}_l) \le \Phi_{\mathrm{ub}}(\mathrm{H}_l)$.

2. For every $\epsilon > 0$, there exists $\delta > 0$ such that for any $\mathrm{H}_l \subset \mathrm{H}$, $diag(\mathrm{H}_l) < \delta$ implies $\Phi_{\mathrm{ub}}(\mathrm{H}_l) - \Phi_{\mathrm{lb}}(\mathrm{H}_l) \le \epsilon$, where $diag(\mathrm{H}_l)$ denotes the length of the maximal diagonal of the n-dimensional rectangle H_l.

Then the algorithm provided in section A.2.1 will compute $\Phi(\mathrm{H})$ to any requested tolerance in finitely many iterations.

Proof. Let $\bar{\sigma}(\mathrm{H}_l)$, $\underline{\sigma}(\mathrm{H}_l)$ denote the length of the longest and shortest edge of H_l, respectively. Define

$$\mathrm{r} = \frac{\bar{\sigma}(\mathrm{H})}{\underline{\sigma}(\mathrm{H})} \tag{A7}$$

The partitioning rule for rectangles in the algorithm of section A.2.1 divides a rectangle along its longest side, so for any H_l,

$$1 \le \frac{\bar{\sigma}(\mathrm{H}_l)}{\underline{\sigma}(\mathrm{H}_l)} \le \max\{r,2\} \tag{A8}$$

The expression for the volume of an n-dimensional rectangle together with eq A8 gives

$$\mathrm{vol}(\mathrm{H}_l) \ge \bar{\sigma}(\mathrm{H}_l)\underline{\sigma}(\mathrm{H}_l)^{n-1} \ge \frac{\bar{\sigma}(\mathrm{H}_l)^n}{\max\{r,2\}^{n-1}} \ge \frac{\bar{\sigma}(\mathrm{H}_l)^n}{\max\{r,2\}^n} \tag{A9}$$

Combining eqs A9 with the geometrical consequence $\sqrt{n}\bar{\sigma}(\mathrm{H}_l) \ge \mathrm{diag}(\mathrm{H}_l)$, we have

$$\mathrm{diag}(\mathrm{H}_l) \le \sqrt{n}(\mathrm{vol}(\mathrm{H}_l))^{1/n} \max\{r,2\} \tag{A10}$$

For fixed k, H is the union of the disjoint rectangles $(\mathrm{H}_l)_{l\in L_k}$, so there exits l' such that $\mathrm{vol}(\mathrm{H}_{l'}) \le \mathrm{vol}(\mathrm{H})/k$, which together with eq A10 yields

$$\mathrm{diag}(\mathrm{H}_{l'}) \le \sqrt{n}\left(\frac{\mathrm{vol}(\mathrm{H})}{k}\right)^{1/n} \max\{r,2\} \tag{A11}$$

Consequently, for large k, the partition of H must include a rectangle of small diagonal.

We now show that given some positive tolerance ϵ, there is some iteration count K such that $\mathrm{UB}_K - \mathrm{LB}_K \le \epsilon$. Invoking condition 2 of this proposition, we may select a positive δ such that $\mathrm{diag}(\mathrm{H}_l) \le 2\delta$ implies $\Phi_{\mathrm{ub}}(\mathrm{H}_l) - \Phi_{\mathrm{lb}}(\mathrm{H}_l) \le \epsilon$. Choose sufficiently large k so that

$$\sqrt{n}\left(\frac{\mathrm{vol}(\mathrm{H})}{k}\right)^{1/n} \max\{r,2\} \le \delta \tag{A12}$$

By the argument preceding eq A11, for some $l' \in L_k$, $\mathrm{diag}(H_{l'}) \leq \delta$. Then by a simple geometric argument, the rectangle $H_{l''}$, one of whose halves is $H_{l'}$, must satisfy $\mathrm{diag}(H_{l''}) \leq 2\delta$. Thus, $\Phi_{ub}(H_{l''}) - \Phi_{lb}(H_{l''}) \leq \epsilon$. By the partition rule of the algorithm, that $H_{l''}$ was divided indicates that for some $K \leq k$, $UB_K = \Phi_{ub}(H_{l''})$. Since $LB_k = \max_{l \in L_k} \Phi_{lb}(H_l)$, it then follows that

$$UB_K - LB_K = \Phi_{ub}(H_{l''}) - LB_K \leq \Phi_{ub}(H_{l''}) - \Phi_{lb}(H_{l''}) \leq \epsilon \quad \text{(A13)}$$

Thus, in K or fewer iterations, the algorithm will achieve the tolerance ϵ and terminate.

In our base branch and bound algorithm, we employ the bounding functions

$$\Phi_{lb}(H_l) = \min_{i=1,\ldots,2m} f_i(\mathbf{x}_{c_l}), \text{ and}$$
$$\Phi_{ub}(H_l) = \min_{i=1,\ldots,2m} \{g_i^l(\mathbf{x}_{c_l}) + b_i^l + 0.5\rho_i \,\mathrm{diag}(H_l)\} \quad \text{(A14)}$$

We now show these functions fulfill the two hypotheses of Proposition 1.

Proposition 2. *The functions defined in eq A14 satisfy condition 1 of Proposition 1.*

Proof. The left inequality of condition 1 is clear, as $\Phi(H_l)$ involves a maximization over all $\mathbf{x} \in H$, while $\Phi_{lb}(H_l)$ considers only the single point $\mathbf{x}_{c_l}$. To verify the right inequality of the condition, let $H_l \subset H$ be arbitrary, and take $i \in \{1, \ldots, 2m\}$. By assumption a1, $f_i(\mathbf{x}_{c_l}) \leq g_i^l(\mathbf{x}_{c_l}) + b_i^l$ and by assumption a2, $\max_{\mathbf{x} \in H_l} f_i(\mathbf{x}) \leq f_i(\mathbf{x}_{c_l}) + 0.5\rho_i \,\mathrm{diag}(H_l)$. Consequently, for $i = 1, \ldots, 2m$,

$$\max_{\mathbf{x} \in H_l} f_i(\mathbf{x}) \leq g_i^l(\mathbf{x}_{c_l}) + b_i^l + 0.5\rho_i \,\mathrm{diag}(H_l) \quad \text{(A15)}$$

Finally we note

$$\Phi(H_l) = \max_{\mathbf{x} \in H_l} \min_{i=1,\ldots,2m} f_i(\mathbf{x}) \leq \min_{i=1,\ldots,2m} \max_{\mathbf{x} \in H_l} f_i(\mathbf{x}) \leq \min_{i=1,\ldots,2m} \{g_i^l(\mathbf{x}_{c_l}) + b_i^l + 0.5\rho_i \,\mathrm{diag}(H_l)\} = \Phi_{ub}(H_l) \quad \text{(A16)}$$

which completes the proof.

Proposition 3. *The functions defined in eq A14 satisfy condition 2 of Proposition 1.*

Proof. From assumption a1 and the definition of Φ_{ub},

$$\Phi_{ub}(H_l) \leq \min_{i=1,\ldots,2m} \{f_i(\mathbf{x}_{c_l}) + 2b_i^l + 0.5\rho_i \,\mathrm{diag}(H_l)\} \quad \text{(A17)}$$

For $i = 1, \ldots, 2m$, assumption a2 and the observation g_i^l must approximate f_i at least as well as a constant function equaling $f_i(\mathbf{x}_{c_l})$, gives $b_i^l \leq 0.5\rho_i \,\mathrm{diag}(H_l)$. We then have

$$\Phi_{ub}(H_l) \leq \min_{i=1,\ldots,2m} \{f_i(\mathbf{x}_{c_l}) + 1.5\rho_i \,\mathrm{diag}(H_l)\} \leq \min_{i=1,\ldots,2m} f_i(\mathbf{x}_{c_l}) + 1.5\,\mathrm{diag}(H_l) \max_{i=1,\ldots,2m} \rho_i = \Phi_{lb}(H_l) + 1.5\rho_i \,\mathrm{diag}(H_l) \max_{i=1,\ldots,2m} \rho_i \quad \text{(A18)}$$

Hence

$$\delta = \frac{\epsilon}{1.5 \max_{i=1,\ldots,2m} \rho_i} \quad \text{(A19)}$$

meets the challenge of condition 2.

A.2.1. Base Branch and Bound Algorithm.

Initialization:

Choose an objective function tolerance $\epsilon > 0$.

Set $k \leftarrow 1$, (iteration index)

$H_1 \leftarrow H$,

$UB_1 \leftarrow \Phi_{ub}(H_1)$, (global upper bound)

$LB_1 \leftarrow \Phi_{lb}(H_1)$, (global lower bound)

$N \leftarrow 1$, (number of subdivisions)

$L_1 \leftarrow \{1\}$. (list of subdivisions to explore)

while $UB_k - LB_k > \epsilon$

Pick $l^* \in L_k$ such that $UB_k = \Phi_{ub}(H_{l^*})$.

Partition H_{l^*} into H_{N+1} and H_{N+2} using a cutting plane orthogonal to and centered about its longest edge.

Set $k \leftarrow k + 1$,

$L_k \leftarrow (L_{k-1} \backslash \{l^*\}) \cup \{N + 1, N + 2\}$

$LB_k \leftarrow \max_{l \in L_k} \Phi_{lb}(H_l)$,

$UB_k \leftarrow \max_{l \in L_k} \Phi_{ub}(H_l)$.

Delete from L_k all indices l for which $\Phi_{ub}(H_l) < LB_k$ or $\Phi_{ub}(H_l) - \Phi_{lb}(H_l) \leq \epsilon$.

Set $N \leftarrow N + 2$.

end while

Set $\underline{C}_D \leftarrow LB_k$, $\bar{C}_D \leftarrow UB_k$.

This algorithm calculates the consistency measure within ϵ and can be terminated before convergence if only the sign of the measure is of interest. As is typical of branch and bound procedures, the worst-case performance, measured in number of iterations required for termination, grows exponentially with problem size. This worst-case performance is to be expected since as mentioned in section 4.2.1, consistency measure determination resides in a complexity class for which no known polynomial-time algorithm exists. Despite this handicap, heuristic improvements can be made to the bounding functions Φ_{ub} and Φ_{lb} that yield substantial (although generally unprovable) improvements in practice. The next section focuses on heuristics that appear to work well on mechanistic modeling inspired problems.

A.2.2. Upper Bound Refinement. In the algorithm presented, Φ_{ub} depends on the constants ρ_i that bound the rates of change of the constraint functions f_i. In practice, these bounds are rarely known, and must be estimated with generous safety factor. As an alternative, notice assumption a1 ensures

$$\Phi(H_l) \leq \max_{\mathbf{x} \in H_l} \min_{i=1,\ldots,2m} \{g_i^l(\mathbf{x}) + b_i^l\} \quad \text{(A20)}$$

When the surrogate models are quadratic polynomials, an optimization technique known as the *S-procedure* in the systems and control community[30] readily upper bounds the right side of eq A20, in turn yielding a upper bound on $\Phi(H_l)$. The S-procedure lower bound problem can be arrived via the standard Lagrange relaxation to the right side of eq A20 together with matrix algebra (see ref 31 or the appendix of ref 7) and results in a convex optimization problem known as a semidefinite program (SDP). An SDP in a modest number (hundreds) of variables and constraints is a readily solved convex optimization problem.[31] The S-procedure derived upper bound is typically a substantial improvement over the Φ_{ub} and its use eliminates our requirement of a known bound on the rate of change of each f_i.

Improvements over the S-procedure upper bound are available through what we broadly refer to as "higher-order relaxations". While a typical Lagrange relaxation of a constrained maximization searches for nonnegative scalar multipliers, these higher-order formulations search for nonnegative polynomial multipliers, using sum-of-squares (SOS) programming. SOS programming can also accommodate higher-order polynomial surrogate models. Like the S-procedure, SOS problems are convex, but often feature large numbers of variables. While theoretically tractable, stable algorithms for solving are in their infancy and are an area of active research.[32,33] Further details regarding SOS programming are available in Parrilo.[34]

Lower Bound Refinement. Consider the function

$$h(\mathbf{x}) = \min_{i=1,\ldots,2m} f_i(\mathbf{x}) \tag{A21}$$

and observe

$$\Phi(H_l) = \max_{\mathbf{x}\in H_l} h(\mathbf{x}) \text{ and } \Phi_{lb}(H_l) = h(\mathbf{x}_{c_l}) \tag{A22}$$

An obvious improvement over evaluating h at the center point $\mathbf{x}_{c_l}$ of H_l is to maximize h over H_l using a local search. Alternatively, if the time required to perform a local search on functions involving the process models (i.e., f_i) is prohibitive, one can consider optimizations involving the surrogate models (which have been developed for the upper bound computation). Under the assumption $|f_i(\mathbf{x}) - g_i^l(\mathbf{x})| \le b_i^l$ for all $\mathbf{x}\in H_l$, the following two quantities lower bound $\Phi(H_l)$.

$$\Phi_{lb,a}(H_l) = \max_{\mathbf{x}\in H_l} \gamma \text{ subject to } \gamma \le g_i^l(\mathbf{x}) - b_i^l,\ i = 1, \ldots, 2m \tag{A23}$$

$$\Phi_{lb,b}(H_l) = h(\mathbf{x}^*) \text{ where } \mathbf{x}^* = \operatorname*{argmax}_{\mathbf{x}\in H_l} \gamma \text{ s.t. } \gamma \le g_i^l(\mathbf{x}),\ i = 1, \ldots, 2m \tag{A24}$$

Roughly the same computational time is required to evaluate $\Phi_{lb,a}$ or $\Phi_{lb,b}$ so we compute both and take the larger one as the lower bound for H_l. This local search over the surrogate models usually generates a sufficiently better bound than Φ_{lb} to warrant the increased computational expense.

A.3. Trust-Region Algorithm. To initialize the algorithm, first select a cubical subset of H over which surrogate models of the selected form well approximate the process models. This can be accomplished by attempting fits over successively smaller subsets of H until adequate approximations are obtained. The algorithm can make amends if the initial subset is poorly chosen, so this step is not critical.

Initialization:

Choose an objective function tolerance $\epsilon > 0$.

Choose an initial trust-region H_0 with center point $\mathbf{x}_{c_0}$.

Select a subset growth factor $\gamma > 1$.

Set $k \leftarrow 1$, (iteration index)

$LB_{-1} \leftarrow -\infty$, (lower bound on the consistency measure)

$LB_0 \leftarrow h(\mathbf{x}_{c_0})$ (see equation eq 3).

while $LB_k - LB_{k-1} > \epsilon$

Set *validtry* $\leftarrow$ *false*

while *validtry* == *false*.

Over H_k, approximate each model M_e with a surrogate model S_e^k.

Set $\mathbf{x}_{try} \leftarrow \operatorname*{argmax}_{\mathbf{x}\in H_k} \min_{e=1,\ldots,m}\{u_e - |S_e^k(\mathbf{x}) - d_e|\}$.

Set $LB_{try} \leftarrow h(\mathbf{x}_{try})$

if $LB_{try} > LB_k$

Set $k \leftarrow k + 1$, $LB_k \leftarrow LB_{try}$, $\mathbf{x}_{c_k} \leftarrow \mathbf{x}_{try}$,

validtry $\leftarrow$ *true*,

$H_k \leftarrow (\mathbf{x}_{c_k} - \mathbf{x}_{c_{k-1}}) + \gamma H_{k-1}$.

(translate and grow trust-region)

else

Set $H_k \leftarrow 1/\gamma H_k$. (shrink trust-region)

end if

end while

end while

Set $\underline{C}_D \leftarrow LB_k$.

References and Notes

(1) Golden, D. M.; Manion, J. A. *Adv. Chem. Kinet. Dyn.* **1992**, *1*, 187−274.

(2) Bard, Y. *Nonlinear Parameter Estimation;* Academic Press: Orlando, FL, 1974.

(3) Akaike, H. *IEEE Trans. Automat. Control* **1974**, *19*, 716−723.

(4) Gelman, A.; Carlin, J. B.; Stern, H. S.; Rubin, D. B. *Bayesian Data Analysis;* Chapman and Hall: London, 2003.

(5) Frenklach, M.; Packard, A.; Seiler, P. In *Proceedings of the American Control Conference*; IEEE: New York, 2002; pp 4135−4140.

(6) Frenklach, M.; Packard, A.; Seiler, P.; Feeley, R. *Int. J. Chem. Kinet.* **2004**, *36*, 57−66.

(7) Feeley, R.; Seiler, P.; Packard, A.; Frenklach, M. *J. Phys. Chem. A* **2004**, *108*, 9573−9583.

(8) Burnham, K. P.; Anderson, D. R. *Model Selection and Multimodel Inference;* Springer: New York, 2002.

(9) MacKay, D. J. C. *Information Theory, Inference, and Learning Algorithms;* Cambridge University: New York, 2003.

(10) Horst, R.; Pardalos, P. M.; Thoai, N. V. *Introduction to Global Optimization;* Kluwer Academic: New York, 2000.

(11) Sahni, S. *SIAM J. Comput.* **1974**, *3*, 262−279.

(12) Papadimitriou, C. H. *Computational Complexity;* Addison-Wesley: Reading, MA, 1994.

(13) Frenklach, M. Modeling. Chapter 7 In *Combustion Chemistry*; Gardiner, J., Ed.; Springer-Verlag: New York, 1984.

(14) Box, G. E. P.; Hunter, W. G.; Hunter, J. S. *Statistics for Experimenters. An Introduction to Design, Data Analysis, and Model Building*; Wiley: New York, 1978.

(15) Box, G. E. P.; Draper, N. R. *Empirical Model-Building and Response Surfaces;* Wiley: New York, 1987.

(16) Myers, R. H.; Montgomery, D. C. *Response Surface Methodology: Process and Product in Optimization Using Designed Experiments*; Wiley Series in Probability and Statistics; John Wiley & Sons: New York, 2002.

(17) Seiler, P.; Frenklach, M.; Packard, A.; Feeley, R. *Eng. Optim.* in press.

(18) Lawler, E.; Wood, D. *Operations Res.* **1966,** *14,* 699−719.

(19) Tonse, S. R.; Moriarty, N. W.; Brown, N. J.; Frenklach, M. *Isr. J. Chem.* **1999**, *39*, 97−106.

(20) Conn, A. R.; Gould, N. I. M.; Toint, P. L. *Trust-Region Methods;* SIAM and MPS: Philadelphia, PA, 2000.

(21) Elster, C.; Neumaier, A. *Computing* **1997,** *58,* 31−46.

(22) Prajna, S. *Proceedings of 42nd IEEE Conference on Decision and Control*; IEEE: New York, 2003; pp 2884−2888.

(23) Smith, G. P.; Golden, D. M.; Frenklach, M.; Moriarty, N. W.; Eiteneer, B.; Goldenberg, M.; Bowman, C. T.; Hanson, R. K.; Song, S.; W. C. Gardiner, J.; Lissianski, V. V.; Qin, Z. GRI-Mech 3.0. http://www.me.berkeley.edu/gri_mech/.

(24) Taussig, R.; Ranganathan, R.; Ross, E. M.; Gilman, A. G. *Nature (London)* **2002**, *420*, 703−706.

(25) Berridge, M. J.; Lipp, P.; Bootman, M. D. *Nature Rev. Mol. Cell. Biol.* **2000**, *1*, 11−21.

(26) Alliance for Cellular Signaling. http://www.signaling-gateway.org.

(27) Wiesner, T. F.; Berk, B. C.; Nerem, R. M. *Am. J. Physiol.* **1996**, *270,* C1556−C1569.

(28) Lemon, G.; Gibson, W.; Bennett, M. *J. Theor. Biol.* **2003**, *223*, 93−111.

(29) Balakrishnan, V.; Boyd, S.; Balemi, S. *Int. J. Robust Nonlin. Control.* **1991**, 295−317.

(30) Boyd, S.; El Ghaoui, L.; Feron, E.; Balakrishnan, V. *Linear Matrix Inequalities in System and Control Theory*; Studies in Applied Mathematics 15*;* SIAM: Philadelphia, PA, 1994.

(31) Boyd, S.; Vandenberghe, L. *Convex Optimization*; Cambridge University: New York, 2004.

(32) Prajna, S.; Papachristodoulou, A.; Seiler, P.; Parrilo, P. A. SOSTOOLS: Sum of squares optimization toolbox for MATLAB, 2004.

(33) Henrion, D.; Lasserre, J. B. *ACM Transact. Math. Software* **2003**, *29,* 165−194.

(34) Parrilo, P. A. *Math. Programming Ser. B* **2003**, *96*, 293−320.