Using Dynamic Active Subspaces to Construct Surrogate Models for Calibrating Tumor Growth Models to Data

P.N. Vu and A.L. Lewis

Abstract - While the use of complex ODE models describing tumor growth and tumor-immune interactions may be desirable for accurately representing the biological mechanisms that govern tumor growth, the high-dimensionality of the parameter space can make it difficult to uniquely identify optimal parameter estimates during model calibration. Construction of a surrogate model that can accurately approximate the quantity of interest with fewer parameters may enable us to fit this alternate model to the data in place of the original ODE system. In this investigation, we consider how active subspaces may be used to construct time-dependent surrogate models with reduced parameter dimension, and analyze the trade-offs in parameter identifiability, model fit error, and computational run-time to provide a roadmap for ensuring an identifiable parameter set during model calibration.

Keywords : mathematical oncology; model calibration; surrogate models; dynamic active subspaces; activity scores

Mathematics Subject Classification (2020) **:** 62H99; 00A71; 62H25; 65C50

1 Introduction

Though cancer is believed to have appeared as early as 1200 BC in ancient Egypt [1], effective cancer treatment only emerged in the early 20th century. These days, chemotherapy, surgery, and radiotherapy are prevalent cancer treatments available [2], though targeted drug therapeutics are often unsuccessful due to the emergence of resistance to treatment among tumor cells [3]. Immunotherapy treatments, such as T-cell transfer therapy, immune checkpoint inhibitors, monoclonal antibody treatment, and cancer-targeting vaccines, are currently under development in clinical trials [4, 5]. However, we are still far from fully understanding the interactions between the immune system and tumor growth dynamics. Experimentally validated mathematical models can provide an analytic framework in which we can gain better insight into how tumor cells grow in the presence of the immune system.

Various models using differential equations have been put forth to study tumor growth [6, 7, 8]. Many of these models are simple, containing only a few parameters and considering tumor cell growth in isolation. Others are more complex and incorporate interactions of the tumor cells with the tumor microenvironment and immune system. While

complex ODE models are able to more accurately capture the physical reality of the biological system, they come with a significant drawback: the sparsity of available tumor population data makes calibration of these complex models more difficult, due to the high-dimensionality of the parameter space. The inability to uniquely infer all model parameters has serious ramifications for the level of uncertainty in the resulting model trajectory and the ability of the model to make useful predictions at future times.

With this in mind, we consider how one might build a surrogate model as an approximation to an ODE system and calibrate this surrogate model in place of the original. We use the tumor-immune ODE model developed in [9] as a test model, and use tools from active subspace methodology to construct a surrogate model to approximate our quantity of interest (QoI) from the ODE system. Active subspace methods use the gradient of the QoI to identify a subspace of prominent directions upon which the function changes the most. Directions in which the QoI is least sensitive to perturbations can then be disregarded in order to reduce the dimension of the problem [10]. Essentially, the active subspace method aims to reduce the number of parameters in the model by rotating the coordinate system to align with linear combinations of the standard basis vectors that are most influential to the output. Because these important directions can still incorporate information from all of the original parameters, the active subspace method is more powerful and general than a parameter subset selection method, in which only a strict subset of the original parameters, ranked according to some sensitivity metric, are retained. This rotation does comes with a price, however: defining new parameters as linear combinations of the original parameters can make it difficult to interpret their biological meaning. However, a new model can then be trained with dependence only upon these active directions, allowing for better computational efficiency and presenting a promising new outlook for the model calibration problem.

Active subspace methods have been employed in many applications with scalar-valued quantities of interest—see [10] for several examples. More recently, work has been done to extend this methodology to a time-dependent framework, as in [11], where the authors use dynamic active subspaces to build a surrogate model that approximates T-cell count in the progression of HIV over time, or in [12], where the authors use metrics derived from active subspaces to quantify the time-dependent sensitivity of voltage and capacity to various input parameters in a lithium ion battery model. We build upon these works, using the procedure of [11] to construct a surrogate model and the sensitivity metrics of [13, 12] to perform parameter selection. We then consider how such a surrogate model may be employed for the purposes of model calibration.

In Section 2, we briefly discuss the model from [9] upon which we will test our procedure, then restate the highlights of the active subspace methodology and describe the basics of our model calibration procedure. Section 3 contains the results of our investigation, comparing the model calibration results across three models: the original ODE system, the constructed surrogate model, and a reduced form of the surrogate model that contains fewer parameters to be estimated. Finally, we summarize our work, discuss limitations, and suggest possible avenues for future exploration in Section 4.

2 Methods

In this section we introduce the system of ordinary differential equations that we will use to analyze tumor growth, restate the highlights of the active subspace method for clarity, and discuss our method for calibrating the various models to synthetic data.

2.1 Model explanation and synthetic data construction

In this investigation, we use the experimentally validated model published in [9] to demonstrate our proposed methodology. This model describes growth of a solid tumor over time—assuming that the tumor grows logistically in the absence of interference—and accounts for interactions between tumor cells and the immune system, summarized by natural killer (NK) cells and $CD8^+$ T-cells. NK cells are large granular lymphocytes which are assumed to always be active, representing the innate immune system. By contrast, $CD8^+$ T-cells represent the tumor-specific immunity; they are cytotoxic to tumor cells, but must first be recruited to the tumor site once tumor cells are detected. In the model, it is assumed that both NK and $CD8^+$ cells can kill tumor cells, but both will be deactivated after some number of encounters with tumor cells, a mechanism representing immune exhaustion. The model system is given in Equation (1), where $T(t)$ represents the tumor cell population, $N(t)$ represents the total level of NK cell effectiveness (i.e., the active NK cell population), and $L(t)$ is the total level of $CD8^+$ cell effectiveness, each at a given time t. Further details regarding the functional forms of the chosen terms can be found in [9]. For all simulations in this investigation, we use initial values of $T(0) = 10^4$ cells, $N(0) = 10^3$ cells, and $L(0) = 0$, representing the fact that we begin with only an innate immune response.

$$\begin{aligned} \frac{dT}{dt} &= aT(1-bT) - cNT - D \\ \frac{dN}{dt} &= \sigma - fN + \frac{gT^2}{h+T^2}N - pNT \\ \frac{dL}{dt} &= -mL + \frac{jD^2}{k+D^2}L - qLT + rNT \\ D &= d\ \frac{(L/T)^\lambda}{(L/T)^\lambda + s}T. \end{aligned} \tag{1}$$

This model provides an excellent setting for us to investigate potential dimension reduction; the model includes three state variables (T, N, and L), and a set of 16 parameters, whose biological descriptions and estimated nominal values are listed in Table 1. As described in [9], these values were chosen based on a combination of model fits to mouse and human data and references from the literature. A sample simulation of system (1) illustrating the dynamics of all three populations using the parameters listed in Table 1 is shown in Figure 1(a).

Though system (1) tracks the changes in all three cell populations over time, it is unlikely that we will ever possess experimental or clinical data for all three populations.

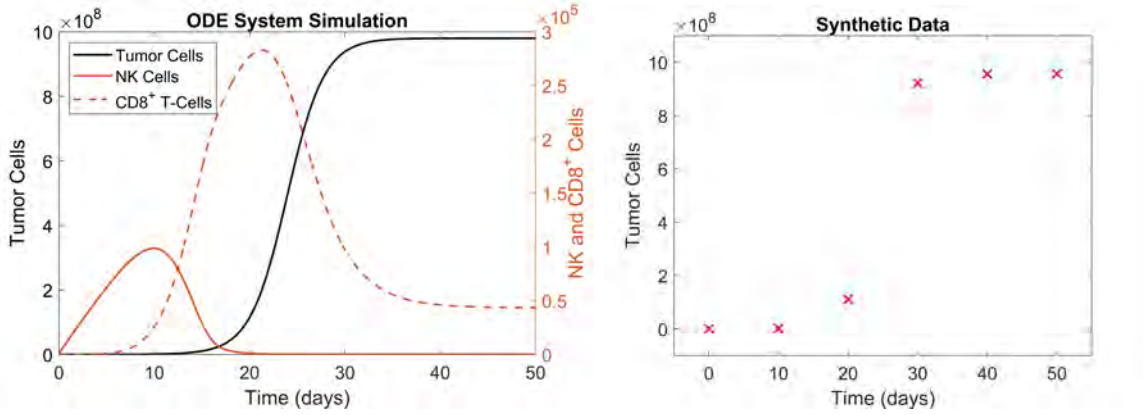
Param.	Value	Units	Description
a	5.14×10^{-1}	day^{-1}	Tumor growth rate
b	1.02×10^{-9}	cell^{-1}	b^{-1} is the carrying capacity
c	3.23×10^{-7}	$\text{cell}^{-1}\ \text{day}^{-1}$	Fractional non-ligand-transduced tumor cell kill by NK cells
d	4.23	day^{-1}	Saturation level of fractional tumor cell kill by $CD8^+$ T cells
σ	1.3×10^4	cells day^{-1}	Constant source of NK cells
λ	1.43	None	Exponent of fractional tumor cell kill by $CD8^+$ T-cells
f	4.12×10^{-2}	day^{-1}	Death rate of NK cells
g	2.5×10^{-2}	day^{-1}	Max NK cell recruitment rate by non-ligand-transduced tumor cells.
h	2.02×10^7	cell^2	Steepness coefficient of the NK cell recruitment curve
j	3.75×10^{-2}	day^{-1}	Maximum $CD8^+$ T-cell recruitment rate
k	2.0×10^7	cell^2	Steepness coefficient of the $CD8^+$ T-cell recruitment curve
m	2.00×10^{-2}	day^{-1}	Death rate of $CD8^+$ T-cells
q	3.42×10^{-10}	$\text{cell}^{-1}\ \text{day}^{-1}$	$CD8^+$ T-cell inactivation rate by tumor cells
p	1.00×10^{-7}	$\text{cell}^{-1}\ \text{day}^{-1}$	NK cell inactivation rate by tumor cells
s	3.6×10^{-1}	None	Steepness coefficient of the Tumor-($CD8^+$ T-cell) competition term
r	1.1×10^{-7}	$\text{cell}^{-1}\ \text{day}^{-1}$	Rate of $CD8^+$ T-cell stimulation due to NK-induced tumor cell death

Table 1: Estimated parameter values and parameter descriptions

More likely, we would have access only to tumor volume scans at several points in time, and all model parameters would have to be estimated using only information about the observable quantity $T(t)$. Thus, for the remainder of this investigation, we will focus on $T(t)$ as our *quantity of interest* (QoI). To mimic the model calibration procedure in a clinical setting, we produce a set of synthetic data for model calibration by using the simulation for $T(t)$ in Figure 1(a), computing $T(t_i)$ for $t_i = 0, 10, 20, 30, 40, 50$ days, and adding up to 5% noise to each data point to represent possible measurement error:

$$T_{\text{data}}(t_i) = (1 + \varepsilon_i)\ T_{\text{simulation}}(t_i), \text{ for } t_i = 0, 10, \ldots, 50,$$

where $\varepsilon_i \sim \mathcal{U}[-0.05, 0.05]$. Figure 1(b) shows the produced synthetic data that will be used to calibrate all models in Section 3.



(a) (b)

Figure 1: (a) Simulation of system (1) using the parameter values reported in Table 1. (b) Synthetic data for use in model calibration.

2.2 Active subspace methodology

Full details regarding the construction and estimation of the active subspace are provided in [10]. Here, we restate the relevant highlights for clarity.

2.2.1 Defining active subspaces

Let $f(\mathbf{x})$, for $f : \mathbb{R}^m \to \mathbb{R}$, denote a scalar-valued quantity of interest which depends upon a vector of parameters $\mathbf{x} \in \mathbb{R}^m$ drawn from a specified probability density $\rho(\mathbf{x})$ such that $\rho : \mathbb{R}^m \to \mathbb{R}$. We assume that all parameters have been normalized (i.e., centered at zero and rescaled to have equal ranges), in order to ensure that all parameters are weighted equally in the analysis. As detailed in [10], common choices for $\rho(\mathbf{x})$ include the uniform hypercube $\mathcal{U}[-1,1]^m$ and the multivariate standard Gaussian density $\mathcal{N}(0,1)^m$.

Assuming that f is differentiable with gradient vector $\nabla_{\mathbf{x}} f \in \mathbb{R}^m$, we define the $m \times m$ symmetric, positive semidefinite matrix $\mathbf{C}$ as the expected value of the outer product of the gradient with itself:

$$\mathbf{C} = \int (\nabla_{\mathbf{x}} f)\,(\nabla_{\mathbf{x}} f)^T\,\rho(\mathbf{x})\,d\mathbf{x}, \tag{2}$$

whose elements are the average product of partial derivatives,

$$\mathbf{C}_{ij} = \int \left(\frac{\partial f}{\partial x_i}\right)\left(\frac{\partial f}{\partial x_j}\right)\rho(\mathbf{x})\,d\mathbf{x}, \qquad i,j = 1,...,m, \tag{3}$$

When $i = j$, the elements on the main diagonal of $\mathbf{C}$ give an indication of how much f perturbs with respect to parameter x_i. Likewise, the off-diagonal elements represent "pairwise" contributions of both parameters x_i and x_j to changes in the QoI. As discussed in [10], $\mathbf{C}$ can be interpreted as the uncentered covariance matrix of the gradient vector $\nabla_{\mathbf{x}} f$ having density $\rho(\mathbf{x})$.

Being symmetric, $\mathbf{C}$ admits a real eigenvalue decomposition

$$\mathbf{C}{=}\mathbf{W}\Lambda\mathbf{W}^T, \tag{4}$$

with $\mathbf{W} = [\mathbf{w}_1, \mathbf{w}_2, ..., \mathbf{w}_m]$ being the orthonormal matrix of eigenvectors corresponding to the matrix $\Lambda{=}\ \mathrm{diag}(\lambda_1, \lambda_2, ..., \lambda_m)$ of eigenvalues in decreasing order. It can be shown that each eigenvalue λ_i is the mean-squared directional derivative of f with respect to its corresponding eigenvector $\mathbf{w}_i$ [10]; i.e.

$$\lambda_i = \int \left((\nabla_{\mathbf{x}} f)^T \mathbf{w}_i\right)^2 \rho(\mathbf{x})\, d\mathbf{x}, \qquad \text{for } i = 1, 2, ..., m. \tag{5}$$

That is, the eigenvalue λ_i quantifies how much f perturbs on the direction specified by $\mathbf{w}_i$. Furthermore, when $\lambda_i \approx 0$, f barely perturbs along the direction $\mathbf{w}_i$; this can be exploited to reduce the dimension of the parameter space.

Accordingly, we look for a natural gap—should one occur—appearing between two consecutive eigenvalues in Λ, say λ_n and λ_{n+1}. Then, we partition the eigenvalues and eigenvectors according to this gap:

$$\Lambda = \begin{bmatrix} \Lambda_1 & \\ & \Lambda_2 \end{bmatrix}, \mathbf{W} = [\mathbf{W}_1 \quad \mathbf{W}_2] \tag{6}$$

where $\Lambda_1 = \mathrm{diag}(\lambda_1, \lambda_2, ..., \lambda_n)$ contains the first n largest eigenvalues in decreasing order (by definition, $n \leq m$), and $\mathbf{W}_1$ is the $m \times n$ matrix containing the corresponding first n eigenvectors. As suggested in [10], we traditionally exploit the largest naturally occurring gap in the eigenvalues, while recognizing that a trade-off exists between the goals of reducing the dimension of the parameter space (which favors using fewer eigenvalues), and maintaining accuracy in the eventual surrogate model (for which reserving more dimensions may be more desirable). Coleman et al. [14] further explore the pros and cons of several approaches for determining the dimension of the active subspace, including a comparison of the gap-based approach, an error-based approach, and an energy-based approach similar to that used in principle component analysis (PCA).

Having partitioned the eigenvalues and eigenvectors based on the largest gap in the eigenvalues, we define the *active* variables $\mathbf{y}$ and *inactive* variables $\mathbf{z}$, which are linear combinations of the original parameters:

$$\mathbf{y} = \mathbf{W}_1^T \mathbf{x} \in \mathbb{R}^n \qquad \text{and} \qquad \mathbf{z} = \mathbf{W}_2^T \mathbf{x} \in \mathbb{R}^{m-n}. \tag{7}$$

We define the *active subspace* to be the range of the eigenvectors in $\mathbf{W}_1$, and the *inactive subspace* to be the range of the eigenvectors in $\mathbf{W}_2$. Since $\mathbf{W}$ is an orthonormal matrix, we can write

$$\mathbf{x} = \mathbf{W}\mathbf{W}^T\mathbf{x} = \mathbf{W}_1\mathbf{W}_1^T\mathbf{x} + \mathbf{W}_2\mathbf{W}_2^T\mathbf{x} = \mathbf{W}_1\mathbf{y} + \mathbf{W}_2\mathbf{z}. \tag{8}$$

Then, since f is relatively insensitive to changes in the inactive variables $\mathbf{z}$, we can approximate f by

$$f(\mathbf{x}) = f(\mathbf{W}_1\mathbf{y} + \mathbf{W}_2\mathbf{z}) \approx f(\mathbf{W}_1\mathbf{y}).$$

This approximation allows us to construct a surrogate model $g(\mathbf{y})$, which depends only on the active variables $\mathbf{y}$, such that $f(\mathbf{x}) \approx g(\mathbf{y})$. The construction of a surrogate model will be described in Section 2.2.4.

2.2.2 Estimating the active subspace

In reality, it is difficult to compute $\mathbf{C}$ directly because it requires integration across a moderate- to high-dimensional parameter space. Instead, we can employ the sampling-based Algorithm 1.1 from [10] to estimate the active subspace, range($\mathbf{W_1}$). We restate this procedure in Algorithm 1 below.

Algorithm 1 Estimating the active subspace (Algorithm 1.1, [10])

1: Draw independent samples $\{\mathbf{x}_i\}_{i=1}^M$ according to the sampling density $\rho(\mathbf{x})$.
2: For each sample $\mathbf{x}_i$, compute $\nabla_{\mathbf{x}} f_i = \nabla_{\mathbf{x}} f(\mathbf{x}_i)$ and the quantity of interest $q_i = f(\mathbf{x}_i)$.
3: Construct the matrix

$$\hat{\mathbf{C}} = \frac{1}{M} \sum_{i=1}^{M} \nabla_{\mathbf{x}} f_i \nabla_{\mathbf{x}} f_i^T = \hat{\mathbf{W}} \hat{\mathbf{\Lambda}} \hat{\mathbf{W}}^{\mathbf{T}},$$

where $\hat{\mathbf{W}}$ is the matrix of eigenvectors, and $\hat{\mathbf{\Lambda}} = \text{diag}(\hat{\lambda}_1, \ldots, \hat{\lambda}_m)$ is the diagonal matrix of eigenvalues ordered in decreasing order.
4: Partition $\hat{\mathbf{W}} = [\hat{\mathbf{W}}_\mathbf{1}\ \hat{\mathbf{W}}_\mathbf{2}]$ according to the natural split in the eigenvalues. The active subspace can then be approximated by range($\hat{\mathbf{W}}_\mathbf{1}$).

In practice, the number of gradient samples employed is chosen to be at least $M = \alpha k \log(m)$, where α is an oversampling factor chosen from the interval $[2, 10]$ and k is chosen to be one greater than the largest dimension that can be tolerated for the active subspace (often determined by computational budget). This choice of M is motivated by the convergence theorems presented in [10].

If the gradient vector is difficult to compute manually or beyond computing capabilities, then a finite difference scheme can be employed to compute $\nabla_{\mathbf{x}} f_i$ in Algorithm 1—see [10] for further details.

2.2.3 Sensitivity metrics from the active subspace

Because the change in f with respect to each direction in the active subspace is encoded in the pairs $(\lambda_j, \mathbf{w}_j)$ for $j = 1, \ldots, n$, we can use the information gathered from the eigenpairs of $\mathbf{C}$ to define a sensitivity metric for the original parameter set $\mathbf{x}$ [13]. We define the *activity score* of parameter x_i as

$$A_i = \sum_{j=1}^{n} \lambda_j \mathbf{w}_{i,j}^2. \tag{9}$$

That is, we sum the contributions from parameter x_i over all directions in the active subspace, weighted according to the eigenvalue associated with each. In short, the activity

scores defined above quantify the importance of each of our original parameters within the active subspace itself.

We can use these activity scores to rank the importance of each of the original parameters with respect to the chosen quantity of interest $f(\mathbf{x})$. This is the basis for parameter *subset* reduction, wherein the most influential parameters are estimated while the rest are fixed at a nominal value or removed from the model. This tool will be used in Section 3.3 to reduce the parameter dimension of our surrogate model.

2.2.4 Constructing surrogate models

Having estimated the active subspace, range($\mathbf{W_1}$), in Algorithm 1, we seek to construct a surrogate model $g(\mathbf{y})$, which can be used as an approximation to $f(\mathbf{x})$ but depends only on the n active variables. Such surrogate models are often termed *response surfaces*, and may be constructed using radial basis functions, polynomial interpolation, or regression procedures [10]. Throughout this investigation, we will construct our response surfaces using polynomial regression. Algorithm 2 details the construction of the response surface.

Algorithm 2 Constructing a response surface on the active variables

1: Use Algorithm 1 to compute $\hat{\mathbf{W}}_\mathbf{1}$.
2: Generate a set of training samples $\{\mathbf{x}_j\}_{j=1}^N$, and evaluate $f(\mathbf{x}_j)$ at each.
3: For each $j = 1, 2, ..., N$, compute the corresponding active parameter set $\mathbf{y}_j = \hat{\mathbf{W}}_1{}^T \mathbf{x}_j$.
4: Fit a polynomial response surface of order p, denoted $g(\mathbf{y})$, to the pairs $\{(\mathbf{y}_j, f(\mathbf{x}_j))\}_{j=1}^N$.

In Step 2 of Algorithm 2, the number of training samples, N, should be sufficiently large to enable a polynomial regression of order p on the n active variables. In Step 4, it can be helpful to first visualize the relationship between the active variables and the quantity of interest via a scatter plot of $\mathbf{y}_j = \hat{\mathbf{W}}_1^T \mathbf{x}_j$ versus $f(\mathbf{x}_j)$ for $j = 1, \ldots, N$—termed a *sufficient summary plot* (SSP)—provided that the dimension of the active subspace permits such a visualization. We use the trend observed in the SSP to choose a polynomial order p, then employ the MATLAB function `polyfit` to find the best-fit polynomial of degree p to fit the training points.

2.2.5 Extension to a time-dependent quantity of interest

Until now, we have only worked with a scalar QoI, $f : \mathbb{R}^m \to \mathbb{R}$. For our tumor growth application, however, we are interested in a time-dependent quantity of interest, $T(t)$, representing the size of the tumor population at time t. Thus, we need to extend the active subspace methodology to analyze a time-dependent QoI, denoted $f(\mathbf{x}, t)$. We will follow the procedure outlined by Loudon et al. in [11].

We begin by redefining our matrix $\mathbf{C}$ from Equation (3) as a time-dependent matrix:

$$\mathbf{C}(t) = \int \nabla f(\mathbf{x}, t) \nabla f(\mathbf{x}, t)^T \rho(\mathbf{x})\, d\mathbf{x} = \mathbf{W}(t) \Lambda(t) \mathbf{W}(t)^T \tag{10}$$

It is assumed that the probability density function on the parameter space, $\rho(\mathbf{x})$, does not change over time. By the above definition, the *active subspace* spanned by $\mathbf{W}_1(t)$ is time-dependent; that is, the combination of parameters that contribute most to changes in the quantity of interest may vary over time.

We can analyze how the relationship between the active variables and the quantity of interest changes over time by looking at the sufficient summary plots at a set of discrete times $\{t_k\}_{k=1}^T$. For each discrete time point, we can construct a response surface $g(\mathbf{y}(t_k))$. If a pattern persists across all of the response surfaces—for example, all of the SSPs can be characterized by a response surface of the same order polynomial—then we can assemble all of the response surfaces into a global function, $h(\mathbf{y}(t), t)$, whose form incorporates this trend.

For example, if each sufficient summary plot can be characterized by a quadratic response surface, $g(\mathbf{y}(t_k)) = a_2(t_k)\mathbf{y}(t)^2 + a_1(t_k)\mathbf{y}(t) + a_0(t_k)$ for $k = 1, \ldots, T$, then we could construct a global function

$$h(\mathbf{y}(t), t) = \alpha_2(t)\mathbf{y}(t)^2 + \alpha_1(t)\mathbf{y}(t) + \alpha_0(t),$$

where $\alpha_i(t)$ for $i = 0, 1, 2$ are time-dependent functions which summarize how the constant coefficients $a_2(t_k)$, $a_1(t_k)$, and $a_0(t_k)$ change as t_k ranges from t_1 to t_T. That is, for the set $\{a_0(t_k)\}_{k=1}^T$, we can find a functional form $\alpha_0(t)$ such that $\alpha_0(t_k) \approx a_0(t_k)$ for all k; likewise, we do the same for the sets $\{a_1(t_k)\}_{k=1}^T$ and $\{a_2(t_k)\}_{k=1}^T$. In this investigation, we use cubic splines to construct the time-dependent polynomial coefficients, using the MATLAB function `spline`. The procedure for constructing such a global function in general is summarized in Algorithm 3.

Algorithm 3 Constructing a global, time-dependent polynomial surrogate model

1: For each t_k, $k = 1, \ldots, T$, compute $\hat{\mathbf{W}}_\mathbf{1}(t_k)$ using Algorithm 1.
2: For each t_k, $k = 1, \ldots, T$, use Algorithm 2 to fit a polynomial response surface of order p,

$$g(\mathbf{y}(t_k)) = a_p(t_k)\mathbf{y}(t_k)^p + a_{p-1}(t_k)\mathbf{y}(t_k)^{p-1} + \cdots + a_0(t_k),$$

by determining the best-fit constant coefficients $a_p(t_k), a_{p-1}(t_k), \ldots, a_0(t_k)$.
3: Construct a global polynomial function

$$h(\mathbf{y}(t), t) = \alpha_p(t)\mathbf{y}(t)^p + \alpha_{p-1}(t)\mathbf{y}(t)^{p-1} + \cdots + \alpha_0(t),$$

where $\alpha_i(t)$ for $i = 0, \ldots, p$ represents a cubic spline functional fit to the constant coefficients $a_i(t_k)$ for $k = 1, \ldots, T$.

We note that the construction described in Algorithm 3 requires that the response surfaces $g(\mathbf{y}(t_k))$ all be of the same form (e.g., polynomials of the same degree). Global surrogate models can be built in a piecewise manner for scenarios in which this is not the case; see [11] for an example of a model in which the functional form of the response surface changes over time.

Finally, we note that the activity score sensitivity metric defined in Equation (9) can be extended to a time-dependent scenario by computing a parameter's activity score at each discrete time step t_k and then summing the activity scores over all time steps, $k = 1, \ldots, T$. Assuming that the time steps are evenly-spaced, this is essentially equivalent to employing a time-averaged sensitivity metric. For a parameter x_i, we define the overall activity score to be

$$A_{i,\text{Total}} = \sum_{k=1}^{T} A_i(t_k) = \sum_{k=1}^{T} \sum_{j=1}^{n} \lambda_j(t_k) \mathbf{w}_{i,j}(t_k)^2. \tag{11}$$

2.3 Model Calibration

Our model in Equation (1) contains sixteen parameters, a moderate- to high-dimensional problem for model calibration. One of the aims in the field of uncertainty quantification is to uniquely estimate the set of optimal parameters given a set of data; that is, we look for the set of parameter values that is most likely to have produced the given data set. Because this procedure requires working backwards to measure the model inputs from the data representing the quantity of interest, we call it an *inverse problem*.

Conventionally, both frequentist and Bayesian perspectives have been employed to solve the inverse problem. In the former, parameters are assumed to be fixed but unknown point estimates. From a Bayesian perspective, however, parameters are considered to be random variables whose associated distributions can be updated to reflect new information: i.e., acquisition of new data. In this study, we use the Bayesian framework for parameter estimation. One benefit of using a Bayesian method is the construction of a *posterior density* for each parameter, which quantifies the level of uncertainty in that particular input. These posterior densities can then be propagated through the model to quantify how uncertainties in the model inputs may impact our level of certainty in the model output. We discuss the basics of the Bayesian Metropolis algorithm employed for parameter estimation below, and refer the interested reader to [15, 16] for more details.

For each parameter, we begin with a uniform (i.e., non-informative) prior distribution. The Metropolis algorithm employs Monte Carlo sampling to sample from the admissible parameter space, constructing a Markov chain of parameter candidates with each candidate chosen from a multivariate Gaussian proposal function centered at the previously accepted candidate. The optimization of the parameter estimates is based upon computation of the likelihood function; if we assume that errors are independent and identically distributed as $\varepsilon \sim \mathcal{N}(0, \sigma^2)$—where σ^2 represents the error variance—then this likelihood function takes the form

$$\mathcal{L}(\theta|v) = \frac{1}{(2\pi\sigma^2)^{N/2}} \exp\left(-\frac{1}{2\sigma^2} \sum_{i=1}^{N} [v_i - f(\theta, t_i)]^2 \right),$$

where v_i represents the ith observed data point and $f(\theta, t_i)$ represents the model quantity of interest evaluated at parameter set θ and time t_i, for $i = 1, \ldots, N$. With this

formulation, maximizing the likelihood function is equivalent to minimizing the sum-of-squares error between the model predictions and the observed data. For each proposed parameter candidate in the Markov chain, we compute the likelihood function; if the function increases compared to the evaluation at the previous candidate, we accept this new candidate as the next parameter set in our chain. Else, we reject the candidate with a probability based upon the ratio of the proposed candidate's likelihood to that of the previous candidate—see [15] for further details. The stationary distribution to which the Markov parameter chain converges is equivalent to the posterior distribution of the parameters given the observed data. The mode of each distribution is considered the *maximum a posteriori* (MAP) estimate for that parameter, and is used for further model analysis.

During parameter estimation, it is also worthwhile to investigate whether such a "best" set of parameter values is unique. Given a quantity of interest $f(\mathbf{x})$ which depends on parameter vector $\mathbf{x}$, we define the parameter set $\mathbf{x}$ to be *identifiable* at $\mathbf{x}^*$ if $f(\mathbf{x}) = f(\mathbf{x}^*)$ implies $\mathbf{x} = \mathbf{x}^*$ for all $\mathbf{x}$ in the admissible parameter space. One method by which to assess whether a parameter is identifiable is to consider the posterior density of that parameter. Assuming an initial uniform prior distribution for parameters, an identifiable parameter will have a posterior density with a unimodal peak, with a range that is greatly narrowed from the given flat, uniform prior distributions. On the contrary, non-identifiable parameters often have multimodal or flat posterior densities that are relatively unchanged from the uniform priors. A model is only considered identifiable if all model parameters are identifiable, and obtaining an identifiable parameter set is not guaranteed during model calibration; see [15] for further information.

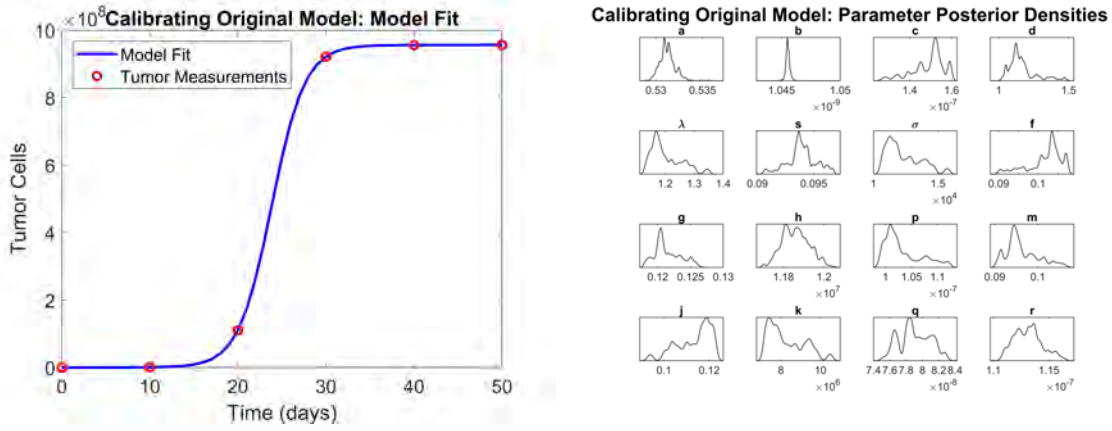
For the purposes of building a model with reasonable predictive power, we desire that all model parameters be identifiable. Else, propagating wide, uninformed posterior densities for non-identifiable parameters through the model results in a large degree of uncertainty in our final model trajectory. Thus, a major question of interest for us will be to determine the maximum number of identifiable parameters for a model, and we will eliminate all others from consideration.

3 Results

We now present the results of our investigation. Recall, our quantity of interest is the number of tumor cells at time t over a period of 50 days, denoted by $T(t)$, for the model described in Equation (1). We use MATLAB 2022a to run simulations in parallel with 32 cores using Lafayette College's High Performance Computing cluster for the construction of the global surrogate model. For all numerical solutions of the ODE system in (1), we use the ODE solver `ode45` with a time step of 10^{-1}. Model calibration is performed on a MacBook Pro with 16GB of memory, using the Metropolis algorithm code provided in the MATLAB MCMC toolbox [16].

3.1 Calibrating the original model

We first attempt to calibrate the full model, estimating all sixteen parameters from the system in (1) by fitting the model trajectory for $T(t)$ to the synthetic data shown in Figure 1(b). The results of our model calibration are shown in Figure 2. While we can achieve an excellent model fit to the data, as shown in Figure 2(a), the posterior densities in Figure 2(b) make it clear that the parameter estimates used to generate that model fit are not unique. Because the posterior densities are non-smooth and multimodal, this suggests that there are many different parameter combinations that can yield the same optimal model fit. Thus, we conclude that the set of all sixteen parameters is non-identifiable, motivating our search for a alternative model in the following section. We note also that calibrating the full model is time-consuming, since every evaluation of the sum-of-squares function requires solving the ODE system before comparing the model predictions for $T(t)$ to the data. Specifically, running the Metropolis algorithm for the full model with a total chain length of 100,000 parameter candidates requires 35 minutes of computational time.



(a) (b)

Figure 2: Calibrating the full model (1) to synthetic data. (a) Model fit to synthetic data, and (b) parameter posterior densities.

3.2 Surrogate model construction and calibration

To address the issue of parameter non-identifiability when calibrating the original model, we use active subspace methodology to exploit prominent linear combinations of the original sixteen parameters to build a lower-dimensional surrogate model. We construct the active subspace every 0.5 days, starting at time $t = 0$, using $M = 500$ samples to estimate $\mathbf{C}(t)$. The sampled parameters for the gradient matrix computations are uniformly sampled from the hypercube $[-1, 1]^{16}$, then scaled and shifted so that the sampling means are the nominal values listed in Table 1 and the ranges are the 2.5% neighborhood around each mean, following the procedure in [11]. At each step t_k, a plot

of the eigenvalue matrix $\Lambda(t_k)$ indicates a significant gap between the first and second eigenvalue. Thus, we conclude that an active subspace of dimension $n = 1$ should be sufficient to characterize the majority of the dynamics in our quantity of interest. To confirm, we plot the sufficient summary plot for each time t_k, $k = 1, \ldots, T$. In all cases, the relationship between the active variable $y = \mathbf{W}_1(t_k)^T\mathbf{x}$ and the quantity of interest $f(\mathbf{x}, t_k)$ exhibits a strong linear relationship, indicating that (a) a single dimension is enough to capture the relationship, and (b) the relationship can be characterized by a linear model. (Note that since y is now 1-dimensional, we drop the vector notation used in Section 2.) The sufficient summary plots for days 10, 20, 30, 40, and 50 are shown in Figure 3(a)-(e), respectively.

Because all of the SSPs display a univariate linear relationship, we construct a surrogate model of the form

$$T(t) \approx h(y(t), t) = \alpha_1(t)y(t) + \alpha_0(t), \tag{12}$$

where $y(t) = \mathbf{W}_1(t)^T\mathbf{x}$. In this setting, $\alpha_1(t)$ represents a time-dependent function describing the slope of each SSP over time, and $\alpha_0(t)$ describes the behavior of the intercept of the SSPs over time. Plots of the SSP slopes and intercepts with respect to time are shown in Figure 4. To describe the functional forms of these time-dependent coefficients, we use a cubic spline interpolation—utilizing the `spline` function in MATLAB—to fit curves for $\alpha_1(t)$ and $\alpha_0(t)$ to the coefficients collected from the SSPs. It should be noted that, due to the global nature of the active subspace method, these spline functions are valid for simulating model trajectories for any set of parameter values from inside the 2.5% range specified for the gradient sampling; that is, the coefficients of these spline functions do not need to be re-estimated for each simulation.

To verify the accuracy of the surrogate model with regards to approximating the quantity of interest from the original model in Equation (1), we plot the QoI over time calculated both by solving (1) and by using the surrogate model $h(y(t), t)$ in Figure 5, for a set of parameter values generated randomly from the ranges described above. The surrogate model trajectory is constructed in a discrete manner, where at each time step t_k we use the spline functions to determine $\alpha_1(t_k)$ and $\alpha_0(t_k)$, and calculate the active variable $y(t_k)$ by projecting the centered and scaled set $\mathbf{x}$ into the active subspace defined by $\mathbf{W}_1(t_k)$. As seen in Figure 5, the surrogate model provides a very close approximation to $T(t)$. Furthermore, the simulation of the surrogate model does not require solving an ODE system, and is therefore more computationally efficient.

However, while our surrogate model provides a good approximation to our quantity of interest—and increases computational efficiency for model simulations—performing model calibration with this surrogate model still requires the estimation of all sixteen original model parameters, since the full set $\mathbf{x}$ is required in order to determine the active variable $y(t)$ at any given time step. The results of the model calibration for $h(y(t), t)$ are shown in Figure 6. The vector $\mathbf{x}$ is estimated within the scaled and centered space of $[-1, 1]^m$, after which all parameter chains are mapped back into the original parameter space defined above.

While the model fit to data is still excellent (see Figure 6(a)), we do not see any sig-

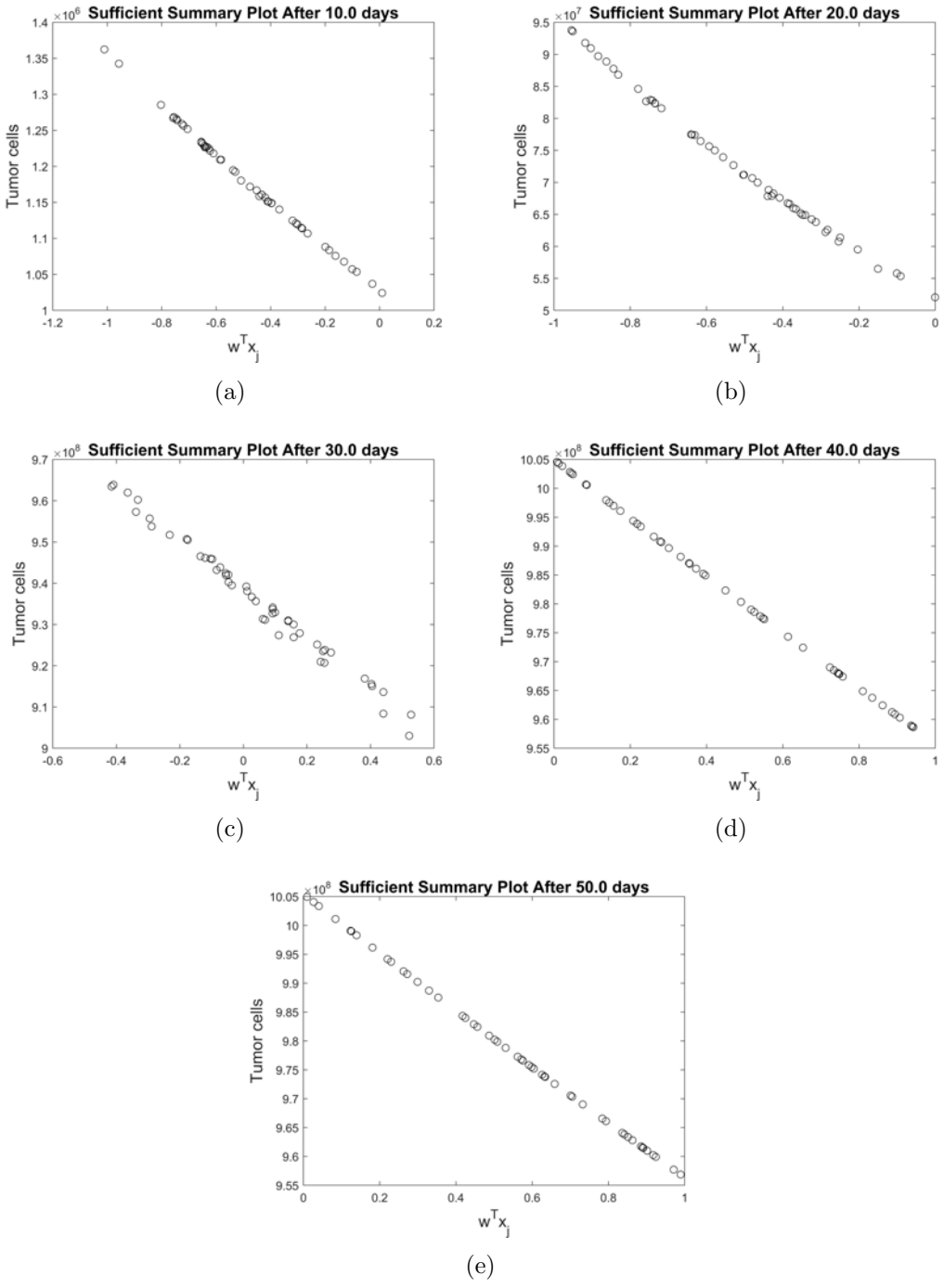
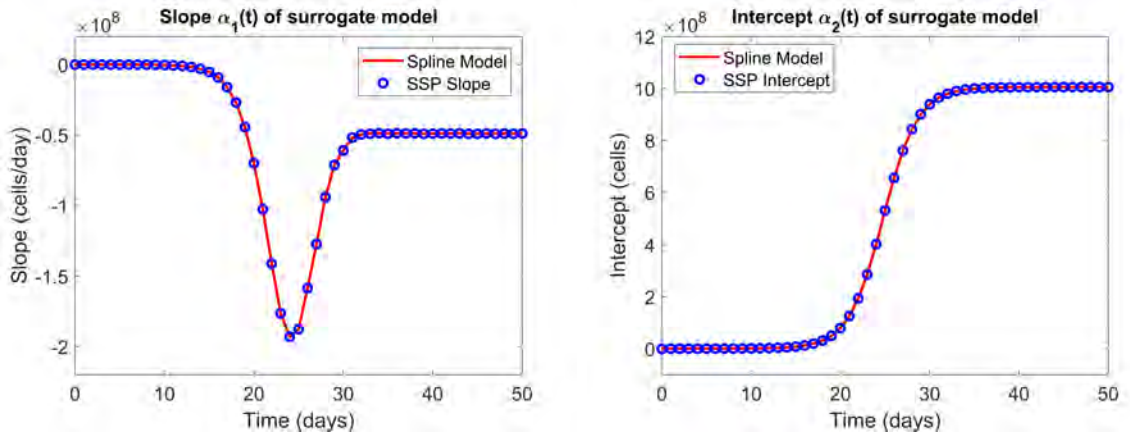


Figure 3: Sufficient summary plots at days 10, 20, 30, 40, and 50.

nificant improvement in the identifiability of the full parameter set. Figure 6(b) shows the posterior densities for each of the sixteen model parameters; most are wide and uninformed, with accepted parameter candidates stretching across the entire permissible



(a) (b)

Figure 4: (a) Slope and (b) intercept plots.

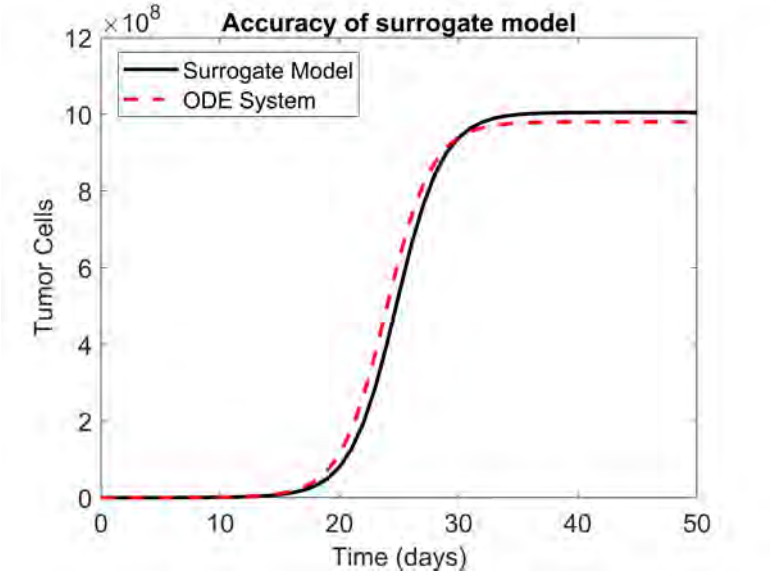
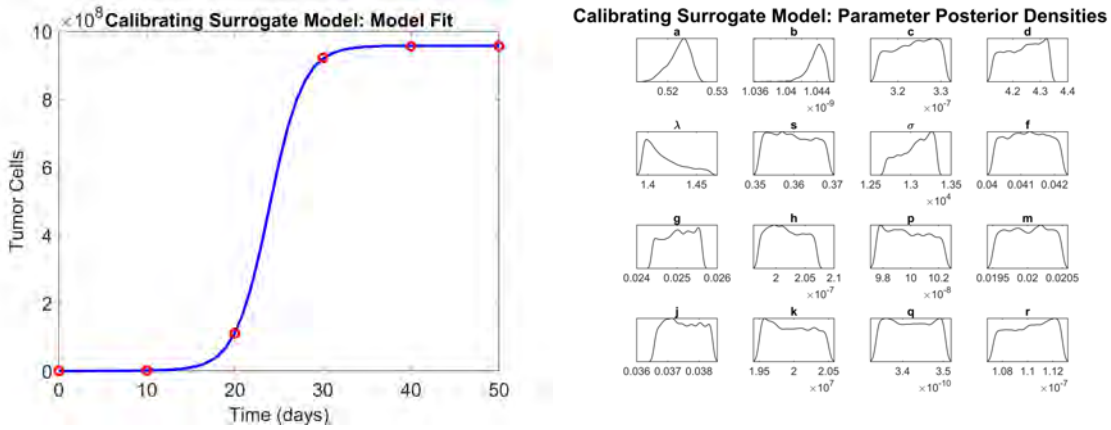


Figure 5: Demonstrating the accuracy of the surrogate model as an approximation to $T(t)$ from Equation (1).

range. Our next step is to determine how we might reduce the number of parameters to be estimated, such that the set of those estimated may be identifiable.

3.3 Reduced surrogate model construction and calibration

We now seek to reduce the parameter dimension of the surrogate model $h(y(t), t)$ built in Section 3.2 by using the time-averaged activity scores defined in Equation (11) to determine a subset of the original parameters can be uniquely identified. Whereas the construction of our surrogate model relies on *subspace-based* reduction, this selection of



(a) (b)

Figure 6: Calibrating the surrogate model (12) with 16 parameters to data. (a) Model fit to synthetic data, and (b) parameter posterior densities.

original model parameters is now equivalent to employing parameter *subset* selection. Based on the time-averaged activity scores, we can rank the sixteen parameters in decreasing order according to how influential they are to our quantity of interest. The logarithms of the time-averaged activity scores for all parameters (normalized to have a maximum value of one) are shown in Figure 7—the scores are shown in log scale in order to better differentiate between their magnitudes. Note that, by definition (11), all activity scores are positive; however, the score for h is so small that the normalized log value turns out to be negative. We observe that parameter a is the most influential. Recall, in our ODE system, a controls the growth rate of the tumor; thus, it is unsurprising that this parameter has a large impact on the total tumor volume at any given time. The second most influential parameter is b, representing the inverse of the tumor carrying capacity. Rounding out the top three influential parameters is λ, which controls how the kill rate of tumor cells by $CD8^+$ immune cells is affected by the immune-to-tumor cell ratio. While other parameters may be important to the system (1) as a whole, they are determined to play less of a role in influencing $T(t)$, our quantity of interest.

We are interested in defining a *reduced* surrogate model—that is, a surrogate model of a lower parameter dimension—by identifying the largest subset of most influential parameters that will yield an identifiable model. Let θ_k denote the parameter set containing only the k most influential parameters, as defined by their activity scores ranking. For example, $\theta_3 = [a, b, \lambda]$, as discussed above. We define $\mathbf{x}_{\text{red},k}$ to be a vector of length m, with the $m - k$ least influential parameters replaced by zeros. That is,

$$\mathbf{x}_{\text{red},k} = [\mathbf{1}_{\theta_k}(x_1) \cdot x_1,\ \mathbf{1}_{\theta_k}(x_2) \cdot x_2, \ldots,\ \mathbf{1}_{\theta_k}(x_m) \cdot x_m],$$

where the indicator function $\mathbf{1}_{\theta_k}(x_i)$ is defined as

$$\mathbf{1}_{\theta_k}(x_i) = \begin{cases} 1 & \text{if } x_i \in \theta_k \\ 0 & \text{if } x_i \notin \theta_k. \end{cases}$$

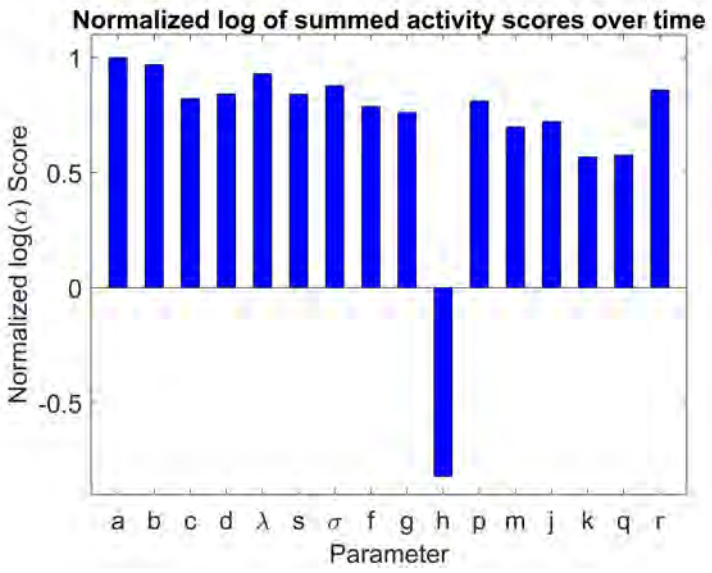


Figure 7: Normalized log of activity scores for all parameters summed over time steps $t = 0, 1, \ldots, 50$.

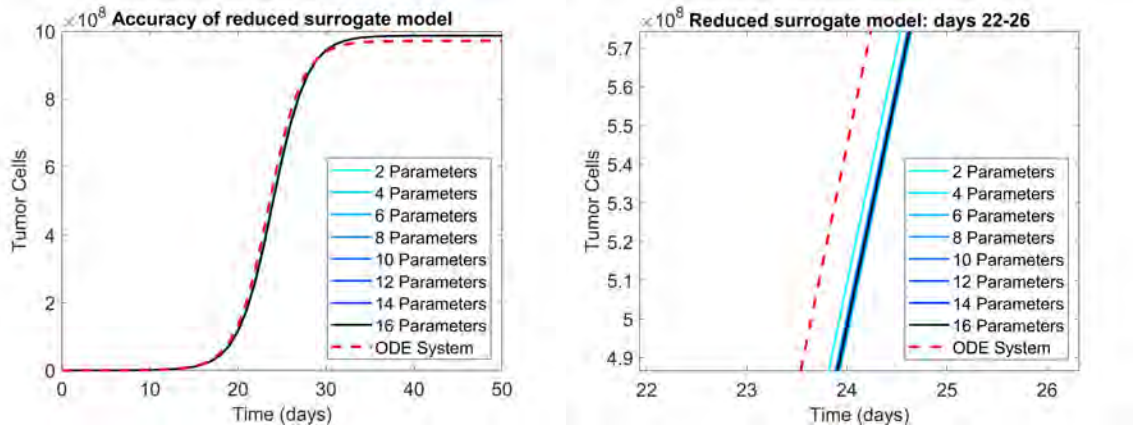
We then define the reduced surrogate model with k parameters to be

$$h_{\text{red},k}(y_{\text{red},k}(t), t) = \alpha_1(t)\ y_{\text{red},k}(t) + \alpha_0(t), \tag{13}$$

where $y_{\text{red},k}(t) = W_1(t)^T \mathbf{x}_{\text{red},k}$. In essence, this fixes the $m - k$ least influential parameters at the mean values of their proposed ranges when constructing the active variable y, since setting the scaled parameter equal to zero is equivalent to setting the natural parameter value equal to its mean once transformed back into the original parameter space. Accordingly, we expect that when one retains the majority of the m parameters in the reduced surrogate model, the approximation of $T(t)$ will still be relatively accurate; as the number of retained parameters decreases, the accuracy of the approximation may deteriorate slightly. This trend is illustrated in Figure 8(a), where simulations are shown for the original model (1), the surrogate model with sixteen parameters (12), and the reduced surrogate models using $k = 2, 4, \ldots, 14$ most influential parameters, using the same randomly generated parameters as in Figure 5. Figure 8(b) shows a close-up of the simulations to allow better visualization of how the loss of parameters impacts the accuracy of the surrogate model.

In order to determine the maximum number of parameters that can be uniquely identified in our reduced surrogate model, we calibrate the reduced surrogate model $h_{\text{red},k}(y_{\text{red},k}(t), t)$ for $k = 1, \ldots, 16$ (recall: $k = 16$ represents the surrogate model that was calibrated in Section 2.2.4). For each calibration, we consider the resulting posterior densities for all estimated parameters, and determine which parameters are identifiable, as indicated by a clear unimodal peak in the distribution. The results of our identifiability analysis for the reduced surrogate models are summarized in Figure 9.

The results show that $k = 3$ is the maximum number of parameters that can be



(a) (b)

Figure 8: Comparison of reduced surrogate models and original ODE system (1): (a) Full time scale, and (b) Close-up of days [22,26].

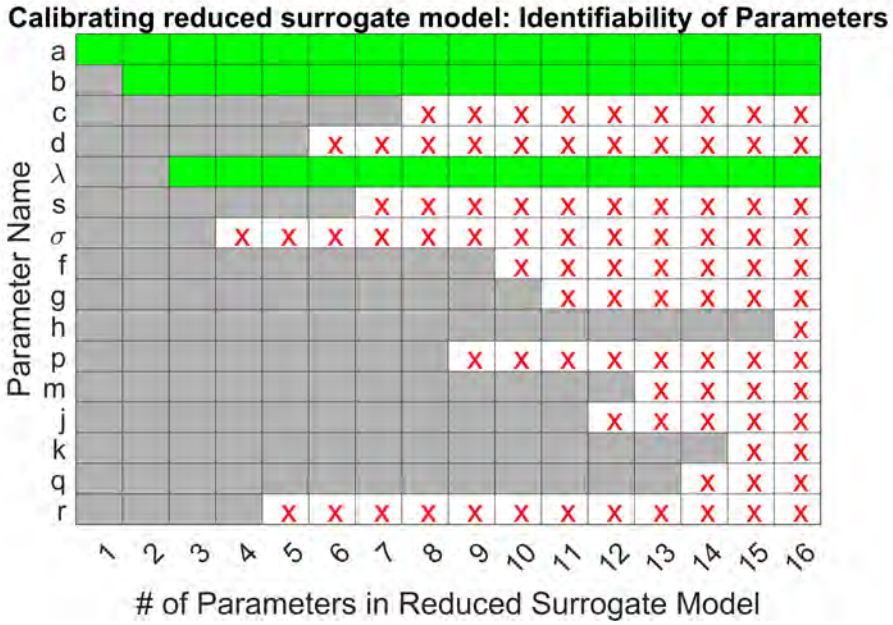
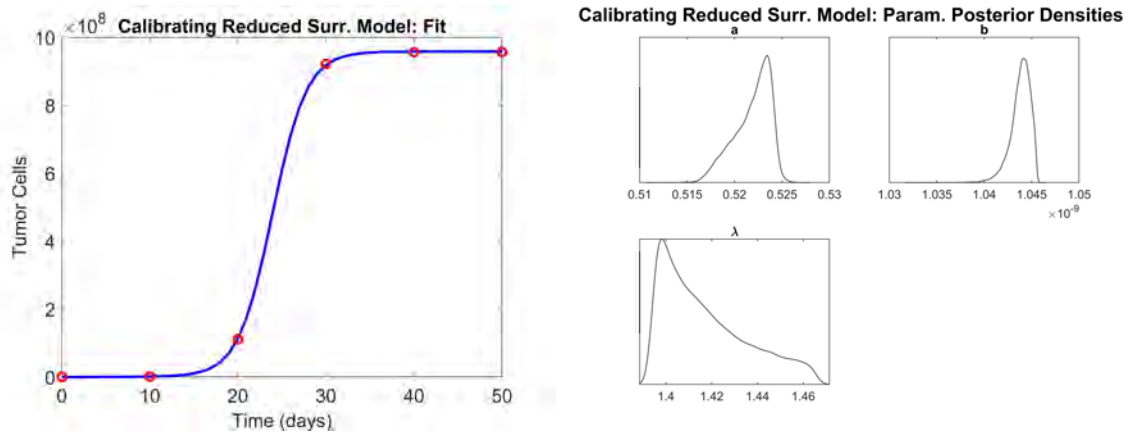


Figure 9: Identifiability of model parameters for each of the 16 surrogate models. Green indicates that a parameter was identifiable, a red × indicates a non-identifiable parameter, and gray indicates that the parameter was not utilized for that particular reduced model.

uniquely identified during model calibration, namely a, b, and λ, whose meanings are discussed above. The model fit and parameter posterior densities for the calibration of $h_{\text{red},3}(y_{\text{red},3}(t), t)$ are shown in Figure 10. The *maximum a posteriori* (MAP) estimate is easily identified in the posterior densities, indicating the value of the optimal point estimate for each parameter.



(a) (b)

Figure 10: Calibrating the reduced surrogate model (13) with three parameters to data. (a) Model fit to synthetic data, and (b) parameter posterior densities.

Because decreasing the number of parameters to be estimated reduces the flexibility of the model trajectory, there exists a trade-off between the number of estimated parameters and the resulting sum-of-squares error between the model predictions and the synthetic data. This trade-off is summarized in Figure 11. We note that the SSE does not improve drastically after $k = 3$, providing further support for our use of $h_{\text{red},3}(y_{\text{red},3}(t), t)$, since using more parameters gains us very little in terms of model accuracy and costs us significantly in terms of parameter identifiability.

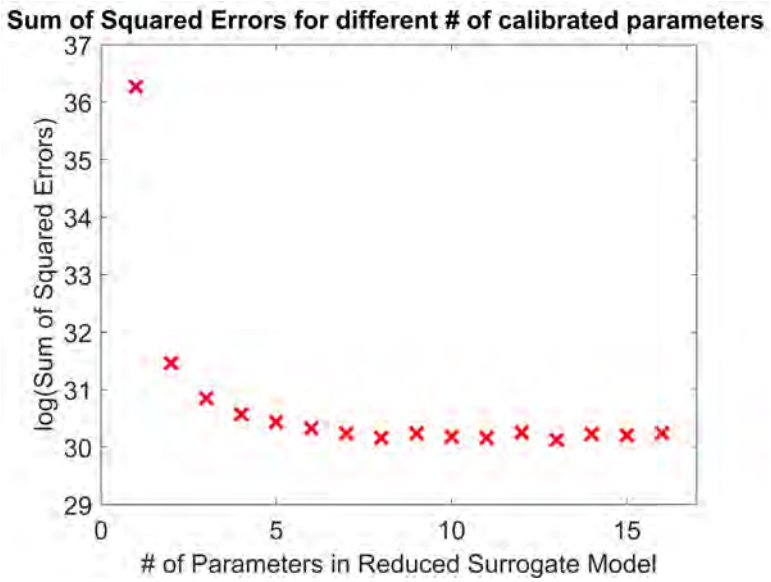


Figure 11: Log of the sum-of-squared errors between model trajectory and synthetic data for each of the 16 possible surrogate models.

3.4 Comparison of models

We have compared three models for our QoI; in terms of decreasing complexity, we have the original ODE system with 16 parameters (Section 3.1), the surrogate model with 16 parameters (Section 3.2), and the reduced surrogate model with three parameters (Section 3.3). We employed a Metropolis algorithm to estimate the parameters and produce the best curve fit for the QoI, and used the posterior densities of the parameters to determine which parameters were uniquely identifiable. In addition to considering the issue of identifiability, we also recorded the CPU running time and calculated the sum-of-squared errors of the best curve fits for each model. The results of these three metrics are summarized in Table 2.

Model	**Curve fit SSE**	**Calibration Time**	**Identifiable**
ODE system	7.50×10^{10}	35.30 minutes	No
Surrogate model	1.36×10^{13}	3.14 minutes	No
Reduced surrogate model	2.50×10^{13}	3.97 minutes	Yes

Table 2: Comparison of model calibration procedure.

Since the synthetic data used for model fitting was generated from the ODE system itself, it is unsurprising that the sum-of-squared errors for the ODE system is numerically superior to those of the surrogate models, though we note that all three models are able to provide an excellent visual fit to the data, as shown in Figures 2(a), 6(a), and 10(a). In terms of computational efficiency, the two surrogate models far outstrip the original model, since there is no need to solve an ODE system for each evaluation of the sum-of-squares function in the Metropolis algorithm. Finally, we note that while information criteria such as the Akaike Information Criterion (AIC) or Bayesian Information Criterion (BIC) are often employed to choose between models—rewarding a good model fit but penalizing the use of two many extraneous parameters—the fact that only the reduced surrogate model permits a fully identifiable parameter set suggests that such a comparison should not determine which model we select. Our primary consideration was to find a model with identifiable parameters, and the reduced surrogate model with three parameters is the only contender that meets this criterion. Thus, we conclude that if the goal is to calibrate a model to use for later analysis by estimating all model parameters—with an emphasis on reducing uncertainty in the model inputs in order to better control the model output—use of the reduced surrogate model is suggested.

4 Conclusion

The calibration of tumor growth models to experimental or clinical data is of utmost importance for validation of the models and subsequent use for prediction. However, clinical data is often limited in quantity, making estimation of the full parameter set infeasible. In this investigation, we considered alternate approaches to calibrating the full

model, by constructing surrogate models which can approximate the quantity of interest but contain fewer parameters to be estimated.

By analyzing which linear combinations of the original model parameters are most influential in driving changes in the quantity of interest, we can build surrogate models that can capture the dynamics of a more complex ODE system with far less computational power and, in some cases, a reduced parameter set. Following the procedure of [11], we have illustrated how to construct a global surrogate model that can approximate the tumor population from a three-compartment ODE system describing tumor-immune interactions by using dynamic active subspaces. We then considered how such a surrogate model might be calibrated to synthetic data, and discussed the trade-offs between parameter identifiability, model fit error, and computational run time.

In particular, we found that using a reduced surrogate model—which uses subspace-based reduction for model construction and then subset-based reduction for parameter selection—can allow for the generation of identifiable parameter estimates. In addition, using this model is computationally superior to the original model in terms of run-time. However, we note that by "removing" the contributions of uninfluential parameters from the model, we are essentially fixing those values at the means of their proposed parameter ranges. Further work should be conducted to determine whether this is the best choice; that is, the dependency of the model calibration results upon the values chosen for the means of the fixed parameters remains an open question and should be studied further. We also remark that one could render the original ODE system identifiable by similarly fixing parameters at their mean values; however, a thorough sensitivity analysis would be required in advance in order to determine which parameters should be fixed and which should be estimated. In the reduced surrogate modeling approach, this ranking information is already freely available via the activity scores, requiring no further computational power than has already been utilized to construct the model.

Because the surrogate models are constructed by determining significant contributions from linear combinations of the original model parameters and then exploiting structural relationships between these linear combinations and the quantity of interest, these models should be considered *phenomonological*, in the sense that the model form is chosen to optimize model fit with no focus on accurately representing the underlying biological mechanisms. Thus, while this model has been shown to be a good explanatory model for the tumor population over the considered time interval, further work is required to determine whether the model can accurately predict future behavior beyond 50 days, since we can no longer guarantee that model outputs will behave within biological constraints. Furthermore, we note that all results from this investigation hold for the single set of synthetic data plotted in Figure 1(b), but that we cannot guarantee the same results would be obtained from a data set generated from different parameters than those listed in Table 1.

Finally, we remark that the approach used here employs parameter estimation on (a subset of) the original parameter space. An interesting alternative would be to calibrate the model by estimating the rotated active parameters $\mathbf{y}$ and then transforming the resulting parameter chains back into the original parameter space to yield estimates for

$\mathbf{x}$, as done in [17]. This method would enable calibration to be conducted on a lower-dimensional space while still allowing for the possibility that all of the original parameters could be informed. However, the procedure in [17] was employed only for scalar-value quantities of interest; additional work is currently being done to extend this framework to vector-valued QoIs such as that considered in this investigation.

A Sample Code

Below, we include sample code for the construction and calibration of the surrogate model with 16 parameters. Note that the calibration code can be easily altered to calibrate any of the reduced surrogate models by setting all parameters not to be estimated equal to zero. In order to run the calibration code, one must first download the MCMC toolbox code from [16].

```
%% TASK 1: Construct global surrogate model

% Nominal parameter values (from de Pillis et al.)
a = 5.14*10^-1;
b = 1.02*10^-9;
c = 3.23*10^-7;
d = 4.23;
sigma = 1.3*10^4;
lambda = 1.43;
f = 4.12*10^-2;
g = 2.5*10^-2;
h = 2.02*10^-7;
j = 3.75*10^-2;
k = 2*10^7;
m = 2*10^-2;
q = 3.42*10^-10;
p = 1*10^-7;
s = 3.6*10^-1;
r = 1.1*10^-7;

% Create parameter vector and bounds (plus/minus 2.5%)
x_pars = [a; b; c; d; lambda; s; sigma; f; g; h; p; m; j; k; q; r];

lb = x_pars-.025*x_pars;
ub = x_pars+.025*x_pars;

% Initial conditions
T0 = 10^4;
N0 = 10^3;
L0 = 0;
initial_values = [T0; N0; L0];

ssp_period = 1; %How often to generate sufficient summary plots
ss_period = 0.5; %How often to compute active subspace
```

```
% Preparation for calculating active subspace
M = 500; %number of gradient samples at each time step
G = zeros(16,M); %allocate storage for gradient matrix
h_fd = 10^-2; %step size for finite differences
evec = zeros(16,16,50/ssp_period+1); %storage for eigenvectors
sing = zeros(16,M,50/ssp_period+1); %storage for singular values
activity = zeros(1,16,50/ssp_period+1); %storage for activity scores

% Loop over time from day 1 to 50
for t=1:(50/ss_period)
    fprintf('Building gradient of QoI at time t=%.1f days\n',t*ss_period);

    % Compute gradient matrix at time t
    for i=1:M
        % Draw random samples from [-1,1]
        x_r = -1+2*rand([16,1]);
        % Rescale random parameters to reside in natural parameter ranges
        x = lb+.5*(x_r+1).*(ub-lb);

        % Computing gradients
        for j=1:16
            x_h = zeros(16,1);
            x_h(j,1) = 1;
            perturb_r = x_r+h_fd*x_h;
            perturb = lb+.5*(perturb_r+1).*(ub-lb);
            G(j,i) = (1/sqrt(M))*(ODEModel(perturb,initial_values,t*ss_period)...
            -ODEModel(x,initial_values,t*ss_period))/h_fd;
        end
    end

    % Compute singular value decomposition of G
    [U,S,~] = svd(G);

    if real(U(1,1)) >0 %align eigenvectors in same direction across time steps
        U = -U;
    end

    % Saving singular values and eigenvalues for choosing dimensions
    if floor(t/(ssp_period/ss_period))==t/(ssp_period/ss_period)
        evec(:,:,t/(ssp_period/ss_period)+1)=U;
        sing(:,:,t/(ssp_period/ss_period)+1)=S;
        a_score = zeros(1,16);
        for j=1:16
            a_score(1,j) =  S(1,1)^2*U(j,1)^2; %activity score for 1D active subspace
        end
        activity(:,:,t/(ssp_period/ss_period)+1)=a_score;
    end

    % Build sufficient summary plot to evaluate efficacy of active subspace
    W_1 = U(:,1); %dimension of subspace, according to eigenvalues
    n_train = 50; %number of training points
    y_train = zeros(1,n_train);
```

```
    q_train = zeros(1,n_train);
    for a=1:n_train
        % Compute active variables at training samples
        x_train_r = -1+2*rand([16,1]);
        y_train(1,a) = W_1'*x_train_r;
        % Compute QoI at training samples
        x_train = lb+.5*(x_train_r+1).*(ub-lb);
        q_train(a) = ODEModel(x_train,initial_values,t*ss_period);
    end

    % Plot sufficient summary plots
    slope = zeros(1,50/ssp_period+1); %storage for slope coefficients
    yintercept = zeros(1,50/ssp_period+1); %storage for intercept coefficients

    if floor(t/(ssp_period/ss_period))==t/(ssp_period/ss_period)
        p = polyfit(y_train,q_train,1); %Fit linear response surface to training pts
        slope(1,t*ss_period/ssp_period+1) = p(1);
        yintercept(1,t*ss_period/ssp_period+1) = p(2);

        figure
        plot(y_train,q_train,'ko')
        xlabel('w_1^Tx_j')
        ylabel('ODEModel cells')
        title(sprintf('Sufficient Summary Plot After %.1f days', t*ss_period))
    end
end

% Consider how slope and intercept change over time
p_slope = spline(0:ssp_period:50,slope);
figure
plot(0:ssp_period:50,ppval(p_slope,0:ssp_period:50),'r-',0:ssp_period:50,slope,'bo')
ylabel('Slope (cells/days)')
xlabel('Time (days)')
set(gca,'fontsize',14)
legend('Spline Model','Slope of SSP')

p_yinterceptfit = spline(0:ssp_period:50,yintercept);
figure
plot(0:ssp_period:50,ppval(p_yinterceptfit,0:ssp_period:50),'r-',...
0:ssp_period:50,yintercept,'bo')
ylabel('Intercept (cells)')
xlabel('Time (days)')
set(gca,'fontsize',14)
legend('Spline Model','Intercept of SSP')


%% TASK 2: Calibrate surrogate model to data

% Load data
data.xdata = [0; 10; 20; 30; 40; 50]; %data point every ten days
data.ydata = [9.6211e3 1.1752e6 1.1094e8 9.2171e8 9.5605e8 9.5661e8]; %tumor cells
```

```
global evec p_slope p_yinterceptfit

params = repmat(0,1,16);
lb = repmat(-1,1,16);
ub = repmat(1,1,16);

%Run Metropolis algorithm to get Bayesian posteriors; sample each from
%[-1,1] uniform prior
params1 = {
    {'a',params(1),lb(1),ub(1)}
    {'b',params(2),lb(2),ub(2)}
    {'c', params(3),lb(3),ub(3)}
    {'d', params(4),lb(4),ub(4)}
    {'\lambda',params(5),lb(5),ub(5)}
    {'s',params(6),lb(6),ub(6)}
    {'\sigma',params(7),lb(7),ub(7)}
    {'f',params(8),lb(8),ub(8)}
    {'g',params(9),lb(9),ub(9)}
    {'h',params(10),lb(10),ub(10)}
    {'p',params(11),lb(11),ub(11)}
    {'m',params(12),lb(12),ub(12)}
    {'j',params(13),lb(13),ub(13)}
    {'k',params(14),lb(14),ub(14)}
    {'q',params(15),lb(15),ub(15)}
    {'r',params(16),lb(16),ub(16)}
    };

model.ssfun = @sse_tumorimmune;
options.updatesigma = 1;
no_smps = 50000;

%burn-in period
options.nsimu = no_smps;
[results,chain,s2chain] = mcmcrun(model,data,params1,options);

%official run
options.nsimu = no_smps;
[results,chain,s2chain,ss2chain] = mcmcrun(model,data,params1,options,results);

% Find the optimal parameters based on minimum SSE
ind = find(ss2chain == min(ss2chain));  ind = ind(1);
bayesparams = chain(ind,:)

% Plot graph to compare fit
    t = 0:1:50;
    figure
    cells = zeros(length(t),1);
    for i_cells = 1:length(t)
        U = real(evec(:,:,i_cells));
        y_test = U(:,1)'*bayesparams';
        % Generate surrogate model at optimal parameter set
        cells(i_cells) = ppval(p_slope,t(i_cells))*y_test...
```

```
        +ppval(p_yinterceptfit,t(i_cells));
    end

    plot(t,cells,'-b','Linewidth',2)
    hold on
    plot(data.xdata,data.ydata,'or','Linewidth',2)
    ylabel('Tumor Cells','Fontsize',14)
    xlabel('Time (days)','FontSize',14);
    set(gca,'fontsize',14)


    %Convert chains back to natural parameter range
    for p = 1:16
        chain(:,p) = lb(p)+.5*(chain(:,p)+1)*(ub(p)-lb(p));
    end

    paramLabels = {'a','b','c','d','\lambda','s','\sigma','f','g','h',...
    'p','m','j','k','q','r'};

    %Look at the chains
    figure(2)
    mcmcplot(chain,[],paramLabels,'chainpanel')

    %Look at the posterior densities
    figure(3)
    mcmcplot(chain,[],paramLabels,'denspanel')


%% HELPER FUNCTIONS

function tumor = ODEModel(params, initConds, time)

[t,V] = ode45(@(t,V) TNL_System(t,V,params),0:.1:50,initConds);
idx = find(t==time);
tumor = V(idx,1);

end


function tumorode = TNL_System(t,V,x)

% x: parameter space [a,b,c,d,lambda,s,sigma,f,g,h,p,m,j,k,q,r]
% V: dependent variable space [T,N,L]

tumorode = zeros(3,1);

% ODE system from "A Validated Mathematical Model of Cell-Mediated Immune
% Response to Tumor Growth" - Pillis, Radunskaya, Wiseman
D_tumorode = x(4,1)*V(1)/(1+x(6,1)/(V(3)/V(1))^x(5,1));
tumorode(1) = x(1,1)*V(1)*(1-x(2,1)*V(1))-x(3,1)*V(2)*V(1)-D_tumorode;
tumorode(2) = x(7,1)-V(2)*x(8,1)+V(2)*x(9,1)*V(1)^2/(V(1)^2+x(10,1))...
-x(11,1)*V(2)*V(1);
```

```
tumorode(3) = -x(12,1)*V(3)+V(3)*x(13,1)*D_tumorode^2/(D_tumorode^2+x(14,1))...
-x(15,1)*V(3)*V(1)+x(16,1)*V(2)*V(1);

end


function SSE = sse_tumorimmune(params, data)

    global evec p_slope p_yinterceptfit
    time = 0:1:50;
    cells = zeros(length(time),1);
    for i_cells = 1:length(time)
        U = real(evec(:,:,i_cells));
        y_test = U(:,1)'*params';
        % Build surrogate model at current parameter candidate
        cells(i_cells,1) = ppval(p_slope,time(i_cells))*y_test...
        +ppval(p_yinterceptfit,time(i_cells));
    end

    if length(cells(:,1))<51 %Model output is ill-behaved
        SSE = 1000; %penalty to get it away from samples that are forcing...
        solution to blow up
    else
        idx = [1 11 21 31 41 51]; %use only the points that match our data times
        SSE = sum(sum((data.ydata-cells(idx,1)).^2));
    end

end
```

Acknowledgments

P.N. Vu completed this research project under the mentorship of A.L. Lewis as part of the Lafayette College EXCEL Scholars Program in Summer 2022.

References

[1] M. Binder, C. Roberts, N. Spencer, D. Antoine, C. Cartwright, On the Antiquity of Cancer: Evidence for Metastatic Carcinoma in a Young Man from Ancient Nubia (c. 1200BC), *PLoS One*, **9** (2014).

[2] M. Arruebo, N. Vilaboa, B. Sáez-Gutierrez, J. Lambea, A. Tres, M. Valladares, A. González-Fernández, Assessment of the evolution of cancer treatment therapies, *Cancers (Basel)*, **3** (2011), 3279–3300.

[3] X. Sun, B. Hu, Mathematical modeling and computational prediction of cancer drug resistance, *Brief. Bioinform.*, **19** (2018), 1382–1399.

[4] P.J. DeMaria, M. Bilusic, Cancer Vaccines, *Hematol./Oncol. Clin. N. Am.*, **15** (2001), 741–773.

[5] National Cancer Institute, Immunotherapy for Cancer, available online at the URL: https://www.cancer.gov/about-cancer/treatment/types/immunotherapy#what-is-the-current-research-in-immunotherapy.

[6] E.A. Sarapata, L.G. de Pillis, *Bull. Math. Biol.*, **76** (2014), 2010–2024.

[7] L.N. Ghaffari, C.M.L. Loeffler, J. Grajek, K. Stankova, A.T. Pearson, H.S. Muti, C. Trautwein, H. Enderling, J. Poleszczuk, J.N. Kather, Classical mathematical models for prediction of response to chemotherapy and immunotherapy, *PLoS Comput Biol*, **18** (2022).

[8] A. Yin, D.J.A.R. Moes, J.G.C. van Hasselt, J.J. Swen, H-J Guchelaar, A review of mathematical models for tumor dynamics and treatment resistance evolution of solid tumors, *CPT Pharmacometrics Syst. Pharmacol.*, **8** (2019), 720–737.

[9] L.G. de Pillis, A.E. Radunskaya, C.L. Wiseman, A validated mathematical model of cell-mediated immune response to tumor growth, *Cancer Res.*, **65** (2005), 7950–7958.

[10] P.G. Constantine, *Active Subspaces: Emerging Ideas for Dimension Reduction in Parameter Studies*, SIAM-Society for Industrial and Applied Mathematics, 2015.

[11] T. Loudon, S. Pankavich, Mathematical analysis and dynamic active subspaces for a long term model of HIV, *Math Biosci Eng.*, **14** (2017), 709–733.

[12] P.G. Constantine, A. Doostan, Time-dependent global sensitivity analysis with active subspaces for a lithium ion battery model, *CoDA 2016 Special Issue: Selected Papers from the Conference on Data Analysis 2016*, **10** (2016), 243–262.

[13] P.G. Constantine, P. Diaz, Global sensitivity metrics from active subspaces, *Reliab. Eng. Syst*, **162** (2017), 1–13.

[14] K.D. Coleman, A.L. Lewis, R.C. Smith, B.J. Williams, M. Morris, B. Khuwaileh, Gradient-free construction of active subspaces for dimension reduction in complex models with applications to neutronics, *SIAM/ASA J Uncertain*, **7** (2018).

[15] R.C. Smith, *Uncertainty Quantification: Theory, Implementation, and Applications*, SIAM - Society for Industrial and Applied Mathematics, 2013.

[16] M. Laine, MCMC Toolbox for MATLAB, available online at the URL: `https://mjlaine.github.io/mcmcstat/`.

[17] A.L. Lewis, R.C. Smith, B.J. Williams, Bayesian calibration on active subspaces, *Proceedings of the American Control Conference*, (2017).

Phuong Nam Vu
Lafayette College
730 High St
Easton, PA, 18042, USA
E-mail: `vunp@lafayette.edu`

Allison L. Lewis
Lafayette College
730 High St
Easton, PA, 18042, USA
E-mail: `lewisall@lafayette.edu`

Received: September 27, 2022 **Accepted**: January 18, 2023
Communicated by Yi Grace Wang