2021

Mathematical Modeling of Lung Inflammation: Macrophage Polarization and Ventilator-Induced Lung Injury with Methods for Predicting Outcome

Sarah B. Minucci
Virginia Commonwealth University

Mathematical Modeling of Lung Inflammation: Macrophage Polarization and Ventilator-Induced Lung Injury with Methods for Predicting Outcome

A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy at Virginia Commonwealth University.

by

Sarah Bartlett Minucci
M.S., Virginia Commonwealth University, 2017
B.S. & B.A., Lee University, 2016

Under the supervision of Angela M. Reynolds, Ph.D.
Associate Professor, Department of Mathematics & Applied Mathematics
Virginia Commonwealth University

Virginia Commonwealth University
Richmond, Virginia
Spring 2021

Acknowledgements

To my amazing advisor, Dr. Angela Reynolds - thank you for your time, your expertise, your mentorship. I can't begin to express how lucky I feel to have been your student. I hope you use the Magnolia soft pretzel recipe for years to come.

To Dr. Rebecca Heise and lab (especially Mike and Franck) - it has been an absolute pleasure collaborating with you. Thank you for giving me such a concrete way to motivate the math I do.

To my other committee members - Dr. Reed Ogrosky, Dr. Laura Ellwein Fix, and Dr. David Edwards: thank you for your time and expertise in helping me achieve this goal. I've greatly appreciated your mentorship throughout my time at VCU. Your encouragement has made me feel like a real mathematician.

To the entire VCU Department of Mathematics & Applied Mathematics and Department of Statistics & Operations Research, particularly Dr. Cheng Ly, Dr. Rebecca Segal, and Dr. D'Arcy Mays - thank you for all of the advice and the invaluable opportunities you were able to make happen for me.

Thank you to my math/stat peers at VCU - Jamie, Elora, Rebecca, Kelly, Michelle, Josh, Kyle, Jeff, and many others. You not only made the difficult days bearable, but every day fun. I would not have made it through this program without you.

Thank you to Dr. Debra Gladden, Dr. Blayne Carroll, Dr. Laura Singletary, Dr. Caroline Maher-Boulis, and the rest of the Lee University Math Department for not only preparing me extremely well for graduate school, but for life. You are the reason for my

passionate theology of work and math.

To Dr. Stephen Merrill at Marquette University - you were the perfect person to introduce me to mathematical biology. Thank you for your guidance and teaching me about research as an act of worship.

To Dr. Kyle Hocking and everyone at VoluMetrix - thank you for allowing me to be a part of your team. It was an invaluable experience and I could not have asked for a better group of people to learn from.

To Dr. Rachel Jennings - thank you for being a wonderful mentor. You were placed in my math journey at exactly the right time, and your career advice has been instrumental. I'm happy to report I won't have to sleep on your couch after graduation.

To my City Church family - thank you for keeping me grounded, giving me a wonderful community, and reminding me of my chief end. Thank you especially to Emily, Jason, Hank, & Essie and Emily & Gabriel for being my home away from home.

To the sweetest, most loyal friends a girl could ask for - Ainsley, Rachel, Jordan, Sarah H., Hannah, Sarah R., Audrey, thank you for being my cheerleaders.

Gram & Papa, thank you for watching so many of my talks and presentations, and for your constant encouragement. Julia & Allie, thank you for creating the best vacations for me when I needed an escape - it helps when two of your best friends are your sisters.

Finally, to my parents, Steve and Janice - it goes without saying, but I would not be here without you. Thank you for your continual support in every way.

Table of Contents

List of Figures

List of Tables

Abstract

MATHEMATICAL MODELING OF LUNG INFLAMMATION: MACROPHAGE POLARIZATION AND VENTILATOR-INDUCED LUNG INJURY WITH METHODS FOR PREDICTING OUTCOME

By Sarah Minucci, Ph.D.

A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy at Virginia Commonwealth University.

Virginia Commonwealth University, 2021.

Director: Angela Reynolds, Ph.D., Associate Professor, Department of Mathematics and Applied Mathematics

Lung insults, such as respiratory infections and lung injuries, can damage the pulmonary epithelium, with the most severe cases needing mechanical ventilation for effective breathing and survival. Furthermore, despite the benefits of mechanical ventilators, prolonged or misuse of ventilators may lead to ventilation-associated/ventilation-induced lung injury (VILI). Damaged epithelial cells within the alveoli trigger a local immune response. A key immune cell is the macrophage, which can differentiate into a spectrum of phenotypes ranging from pro- to anti-inflammatory. To gain a greater understanding of the mechanisms of the immune response in the lungs and possible outcomes, we developed several mathematical models of interactions between immune system components and site of damage while accounting for macrophage polarization. We analyzed these models to highlight the parameters and corresponding biological mechanisms that drive outcome and to make predictions about lung health.

We developed a set of ordinary differential equations (ODEs) to model VILI and utilized parameter sampling to evaluate how baseline immune state and lung health, as well as response to tissue damage, affect post-ventilation outcomes. We used a variety of methods to analyze the resulting parameter sets, transients, and outcomes. Analysis showed that parameters and properties of transients related to epithelial repair and M1 activation are important factors. We then used this collection of parameter sets to generate synthetic data and developed algorithms that utilize this collection to predict lung health outcomes based on early time-point data. Our results were comparable to logistic regression and random forest classification methods, and we performed several case studies to highlight how our methods can be used.

Finally, we used different modeling techniques, ODE modeling and agent-based modeling (ABM), to simulate the spectrum of macrophage activation to general pro- and anti-inflammatory stimuli on an individual cell level. The ODE model includes two hallmark pro- and anti-inflammatory signaling pathways and the ABM incorporates similar M1-M2 rules but in a spatio-temporal platform. We then performed simulations with various initial conditions to replicate different experimental setups. Comparing the two models' results sheds light on the important features of each modeling approach. In the future, when more data is available, these features can be considered when choosing techniques to best fit the needs of the modeler and application.

Chapter 1

Introduction

Inflammation in the lungs is a response to invading pathogens, such as bacteria and viruses, or to other types of insults, including smoking and mechanical ventilation. The immune response is a vital process to remove microorganisms and damaged tissue and promote repair, but failure of this system can lead to chronic inflammation and organ failure. Inflammation is tightly regulated, involving many complex mechanisms, the details of which are still incompletely understood. In particular, more research is needed to understand the spectrum of activation within macrophages, cells crucial to the immune response and involved in the pathogenesis of many diseases [11].

Mathematical modeling provides a platform through which to simulate biological processes, understand and analyze the underlying mechanisms, generate and test hypotheses, and make predictions. The goal of this work is to apply mathematical modeling and analysis methods to develop a greater understanding of lung inflammation in response to mechanical ventilation and general inflammatory stimuli. We aim to explore the driving mechanisms behind varied immune responses and predict outcome, highlighting the role of the spectrum of macrophage activation, from classically activated M1 to alternatively activated M2. Figure 1.1 shows how the three main chapters of this work are linked together in the context of lung inflammation.

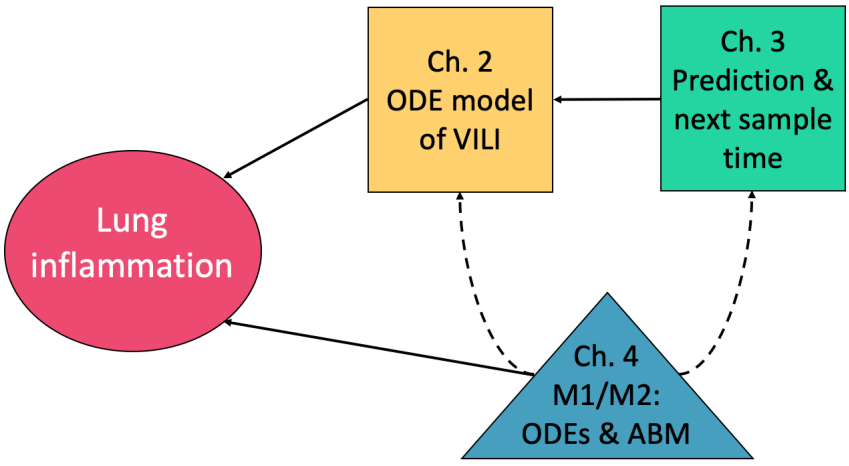


Figure 1.1: A visual representation of how Chapters 2, 3, and 4 relate to our overall goal of understanding lung inflammation. Chapter 2 is specific to VILI, and Chapter 3 builds directly off this work. Chapter 4 examines M1/M2 activation in the context of general inflammatory stimuli, but can be used and adapted to inform more specific types of insults, such as VILI.

This work begins with a set of ordinary differential equations (ODEs) modeling VILI in Chapter 2. We developed this model based on known biological interactions and mechanisms, accounting for macrophage phenotype. Due to a limited amount of data currently available, we used Latin hypercube sampling to develop a collection of parameter sets, each of which produced unique dynamics. We then relied on parameter sensitivity and statistical/machine learning methods to gain a greater understanding of the immune response to VILI and the mechanisms that drive post-ventilation outcomes. We also hypothesized interventions and performed these interventions on a case study. This chapter is a standalone journal article [85] and builds the framework for the next chapter. It contains a detailed background on lung inflammation along with an explanatory figure.

In Chapter 3, we extend the results of our model from Chapter 2. From the parameter collection produced by Latin hypercube sampling, we generated synthetic data. We then used this data to develop an algorithm to predict whether the lung health of an *in silico* patient would worsen after ventilation and a period of recovery due to a severe response to ventilation. Our results compared well to current classification methods, and we extended the

use of our algorithm to determine the next time at which a sample should be taken to obtain the most useful information. The corresponding parameters also provided information about possible outcomes; we developed a process through which we can supplement inconclusive results by examining parameter values. We showed overall accuracy for the entire collection as well as results for selected cases.

Chapters 2 and 3 focus on the immune response to mechanical ventilation; Chapter 4 examines the spectrum of macrophage activation in response to more general inflammatory stimuli using two different modeling approaches. To determine the importance of including subcellular signaling in a model, we developed a system of ODEs representing two hallmark signaling pathways within a macrophage, initiated by signaling proteins TNFα and IL-10. The former is related to the pro-inflammatory (M1) response and the latter to a regulatory (M2b) response. We extended this model to incorporate multiple macrophages and cell lifespan. The second is an agent-based model (ABM) which also utilizes M1-M2 activation rules, but in a spatio-temporal platform. We tuned the model parameters to calibrate the models to each other in the context of a single macrophage. We then performed simulations with various initial conditions to replicate different experimental setups. Comparing the two models' results sheds light on the important features of the modeling approaches and future directions when more data is available.

We conclude in Chapter 5 with a summary of our results, relevance for current experimental processes, and future directions for modeling of lung inflammation.

Chapter 2

Mathematical modeling of ventilator-induced lung inflammation

2.1 Introduction

Inflammation occurs in the lungs when an immune response is initiated to eliminate an insult. Types of insults include inhaled pathogens, such as influenza, Mycobacterium tuberculosis, SARS-CoV-2, and other harmful particles. In the most severe cases this leads to acute respiratory distress syndrome (ARDS). Due to respiratory failure associated with ARDS, the clinical intervention is the use of mechanical ventilation (MV) [139].

Despite the benefits of MV, prolonged or misuse of these ventilators may lead to VILI. In this work we will focus on the alveolar tissue damage associated with MV and resulting immune cell recruitment. The damage caused to alveolar sacs (clusters of alveolar cells) during MV can lead to volutrauma (extreme stress/strain), barotrauma (air leaks), atelectrauma (repeated opening and closing of alveoli), and biotrauma (general severe inflammatory response). If the trauma increases, it can lead to multi-system organ failure [45, 118].

It has also been shown that the inflammatory response of the elderly is altered in the lungs and other areas [103, 108]. As compared to younger mice, increased levels of circulating

inflammatory cytokines and altered macrophage function have been reported in old mice [16]. A 2003-2008 study conducted at Bridgeport Hospital reported that 4,238 out of 9,912 (42.8%) patients received MV for a median of two days. Mortality or discharge to extended-care facilities increased for each decade of age greater than 65 years [39]. Most recently, severe forms of COVID-19, a highly infectious respiratory disease caused by the novel coronavirus SARS-CoV-2, can lead to respiratory failure and death [64]. Studies report varying but overall relatively high rates of mechanical ventilation in response to COVID-19 [20, 71, 145]. The case fatality rate for COVID-19 patients over 70 years old and over 80 years old was around 50.8% and 14.8% of the total number of deaths, respectively [144]. This is in agreement with other studies reporting higher rates of severe outcomes in patients with COVID-19 aged 65 or older [15].

The change in the inflammatory response with patient age combined with the increased need for ventilation and increased mortality rate among the elderly stress the need to investigate the influence of aging in VILI. We used mathematical modeling to investigate the role of the pulmonary innate immune response and interventions to alleviate ventilator-induced damage. At this stage of exploration of VILI, we focus on epithelial damage and immune system interactions. VILI is complex and the final injury pathways may involve pre-existing or evolving co-morbidities. However, we developed this model to explore the contribution of epithelial damage to the development of VILI in isolation.

It is difficult to clinically isolate the local epithelial and inflammatory response in the lung during VILI, and *in silico* modeling of experimental data from animal experiments or human cell lines may help us to understand this complex condition. *In silico* approaches provide the ability to explore immune responses by including various nonlinear dynamics and feedback loops in order to shed light on the specific mechanisms and interactions that drive diseases and generate hypotheses [37]. The framework we have built here addresses VILI with various parameters and initial conditions that can be narrowed in future studies with data from different age groups and/or insults to explore dynamics and driving factors

in various diseases related to age and/or outcome.

We adapted a model developed by Torres et al. for the innate immune response to bacteria, which accounts for macrophage polarization along the pro- to anti- inflammatory spectrum, by including epithelial dynamics and damage-induced recruitment of immune cells [128]. We used this model to understand the mechanisms by which the immune system responds to damaged epithelial cells and the sensitivity of lung health to components of this complex process. We began by analyzing the epithelial subsystem mathematically, since this component of the model was not in Torres et al.. We performed a fixed point analysis and bifurcation diagrams for this subsystem, which is included in the supplementary material (Appendix A.1). We combined the epithelial subsystem with the Torres et al. model by adapting the immune cell dynamics such that they are triggered by epithelial cell damage rather than an infection.

The resulting model is a system of nonlinear ordinary differential equations with a substantial number of parameters. We allowed the parameters in the model to vary over specified ranges using Latin hypercube sampling to simulate the variety of immune system dynamics that may be observed. We organized parameter sets into three categories, healthy, moderate damage, and severe damage, based on the percentage of healthy epithelial cells. The breakdown of the sets into these categories is shown in Figure 2.4, which we describe in greater detail in the following sections. To determine what is driving differences in lung health immediately after ventilation as well as after a recovery phase, we used a variety of methods to analyze the resulting dynamics: 1) comparison of parameters associated with different outcomes, 2) random forest decision tree algorithm, which parses through the variety of predictors that may be particularly important in the immune response to VILI and 3) parameter sensitivity with eFAST, a variance-based method.

2.1.1 Biological background

The alveolar epithelium consists of alveolar type I and type II cells. Alveolar type I cells make up about 95% of the alveolar surface and are primarily responsible for facilitating gas exchange. Type II cells cover the other 5% of the surface and are important in the innate immune response. In the presence of damage, these cells proliferate to repair the epithelium and can also differentiate to type I cells [79, 81].

The immune response is divided into innate (non-specific) and adaptive (acquired) responses. The adaptive immune response involves cells that are effective at fighting specific pathogens, whereas the innate immune response lacks specificity and allows the host to respond to a variety of insults. Two of the most important innate immune cells are neutrophils and macrophages, which can be tissue-specific or recruited to the site upon insult. Some of the important features of the immune response to lung damage are illustrated in Figure 2.1.

Neutrophils respond quickly to pro-inflammatory signals sent from damaged epithelial cells and other resident cells. A small amount of neutrophils are found in the lungs in homeostasis [61]. Neutrophils have phagocytic capabilities in the presence of invading pathogens, but in the case of VILI without infection neutrophils recruit other immune cells such as macrophages through the production of pro-inflammatory agents such as proteinases and cytokines and contribute to the removal of damaged or dead tissue. An overabundance of neutrophils and their byproducts can cause further unnecessary damage [44]. Neutrophils are relatively short-lived; they become apoptotic and are removed by macrophages [61] or become necrotic in an uncontrolled death resulting in the release of cytotoxic material [93].

Phenotypes of macrophages can range from "pro-inflammatory" (M1) to "anti-inflammatory" (M2) based on their activators and byproducts [88, 135]. Their pro-inflammatory behavior includes destroying pathogens, consuming damaged cells, and amplification of signaling. Their anti-inflammatory response, which counteracts pro-inflammatory behavior, promotes repair by producing anti-inflammatory cytokines and removing apoptotic neutrophils. A single macrophage may produce both pro-inflammatory and anti-inflammatory signals con-

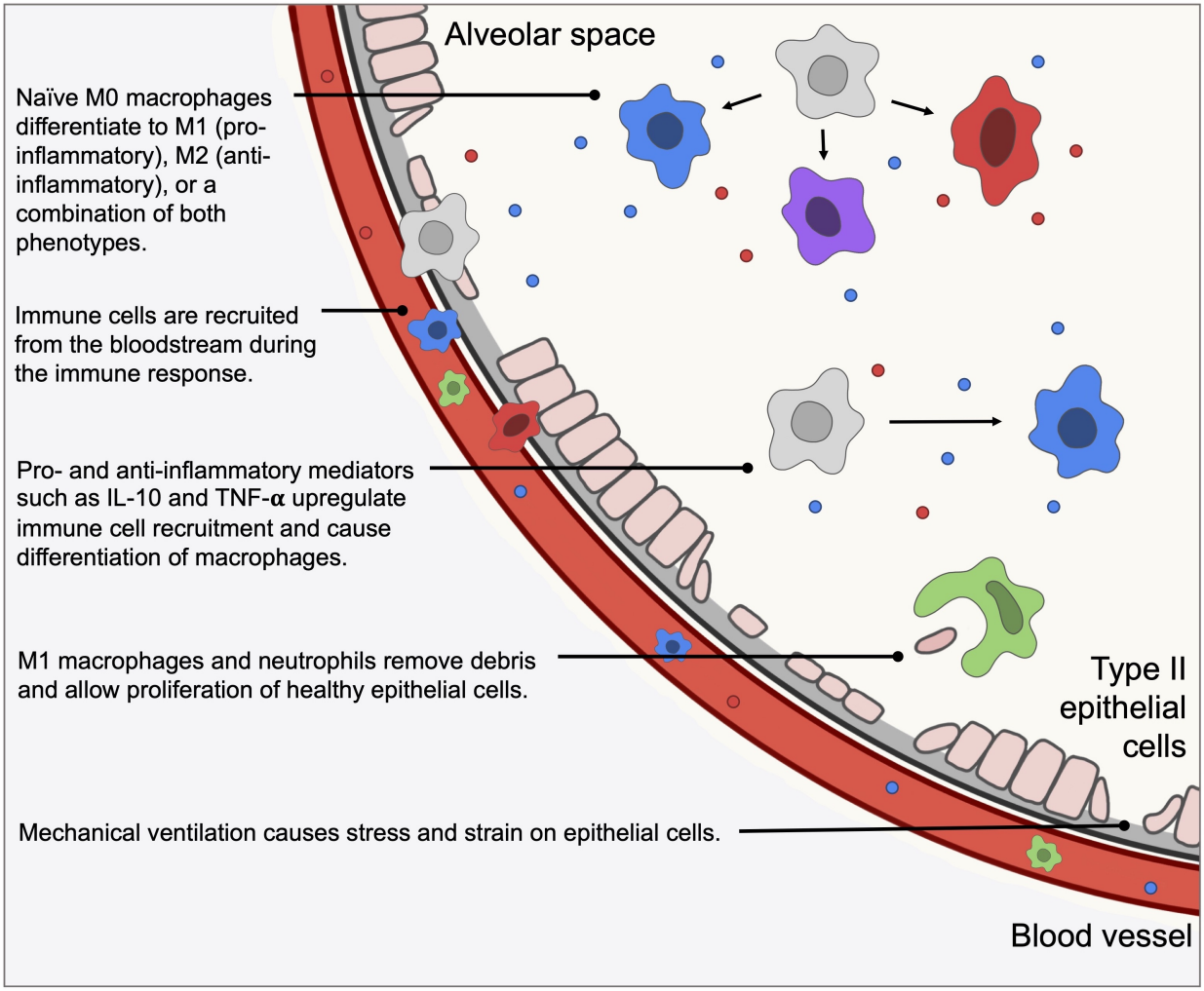


Figure 2.1: An illustration of some of the important biological mechanisms and interactions included in our model, which is described in the following sections.

currently [11].

An imbalance in the pro- and anti-inflammatory responses can cause complications for the individual during various injuries and insults. Also, macrophages play a significant role in the impact of aging on the immune response [16, 67, 72]. Therefore, to develop interventions to mitigate the effects of VILI, it is important to study the immune response to lung injury and the interplay between various types of cells.

2.1.2 Mathematical background

Mathematical modeling is used to capture the complexities of the immune response to epithelial cell damage, including important feedback loops and nonlinearities. Analyzing the resulting model gives insight into the driving mechanisms of this system. An *in silico* approach allows for simulation of scenarios and hypotheses for new interventions, especially when *in vivo* and *in vitro* experiments to explore possible interventions to improve patient outcomes are difficult to perform.

Many models have examined the within-host immune response to bacterial and viral infections, such as influenza, tuberculosis, pneumonia [10, 17, 18, 58, 91, 120] and, most recently, COVID-19 [35, 47, 98]. Additionally, models related to non-infectious injury such as smoking and asthma [14, 22, 42, 102] and general inflammatory stress [103, 111] have been developed. We have published a review of mathematical models that focus generally on the immune response in the lungs [86].

Models of MV and VILI generally deal with the mechanics of the airways, including airflow, pressure, and gas exchange to inform and optimize machine settings and assess stress [7, 24, 46, 57, 62, 75, 100, 106, 124]. Fluid-structure interactions (FSI) can be incorporated into such models [5, 56, 100]. Aghasafari et al. [5] and Ibrahim et al. [52] incorporate the epithelium and immune cells into cellular automata models linked to tissue-scale mechanics. Previous models that include epithelial damage have been developed for wound healing [101], infection [31, 87], and other applications using a variety of methods [119, 132]. Several infec-

tion models identify parameters related to bacterial growth that delineate between healthy and infected states, or high and low pathogenicity [86].

Models have also been developed to understand and analyze the subcellular pathways that govern the phenotype switch that macrophages undergo from pro-inflammatory to anti-inflammatory, as well as other important subcellular pathways [40, 73, 94, 122, 137, 149]. Other mathematical models have described macrophage polarization in the context of infection [27], cancer [70], and other injuries [36, 53]. However, to our knowledge, no mathematical models have described M1/M2 interactions specifically in the context of VILI. We modeled the inflammatory response to VILI, specifically the resulting damage to epithelial cells, using a set of coupled ODEs, which we describe further in the following section. Systems of ODEs are often used to model complex biological systems because of their ability to reproduce a variety of dynamics with reasonable computation times.

To perform a global assessment of a large parameter space such as that described in this work, other methods are needed aside from traditional parameter estimation techniques. Latin hypercube sampling (LHS), a Monte Carlo-based method, evenly samples the parameter space and can quantify uncertainty in model output. Sensitivity analysis methods identify how changes in parameters affect model output. Partial rank correlation measures the linear relationship between input and output and is useful in cases where the relationship is monotonic; when monotonicity is not the case, variance-based techniques such as the Sobol method and eFAST are advantageous [76]. These and similar methods have been applied to various models of inflammation, infection, and others to explore parameter space and identify sensitive parameters that contribute to damage, disease, and recovery [80, 127, 143, 148].

2.2 Methods & Model Development

The primary focus of this model is to examine the effects of damage on the alveolar epithelium, in particular alveolar type II cells, since they are responsible for restoration of

the epithelium. The physical forces of ventilation such as overdistention and tears in the epithelial membrane cause epithelial cells to release various mediators that elicit an immune response [118]. Epithelial barrier damage is one of the main features of VILI [30], and the extent to which the alveolar epithelium is damaged is a useful indicator of the overall effects of a lung insult [79, 138]. We began with a small three-dimensional system of differential equations of epithelial cell dynamics and analyzed this model using stability analysis and bifurcations (Appendix A.1). This became the basis for our lung compartment dynamics in our multi-compartmental model for ventilator-induced lung injury.

The full model is a system of coupled ordinary differential equations based on the interactions between immune cells, epithelial cells, and other mediators in the alveoli, shown in Fig 2.2. This model captures dynamics in two compartments, the local lung and the blood. Damaged lung epithelial cells release mediators that activate local cells and recruit nonresident immune cells from the bloodstream. These activated cells interact with the lung epithelial cells.

2.2.1 Model equations

A system of ODEs was used to model these interactions because of its ability to capture distinct nonlinearities and feedback loops with relatively few computational requirements. However, one of the drawbacks of an ODE model is that it assumes a well-mixed environment, in which all elements of the model are evenly distributed throughout the given space. One way to include aspects of the spatial heterogeneity without explicitly modeling space is to use a compartmental model. Each compartment represents a well-mixed environment and, when biologically appropriate, cells and mediators can move between compartments. The model includes a variable for each cell or mediator for each compartment in which it can be located.

Here we chose to model two compartments. The first is the site of inflammation in the lungs, specifically the epithelial cells which provide a barrier lining in the alveolar region

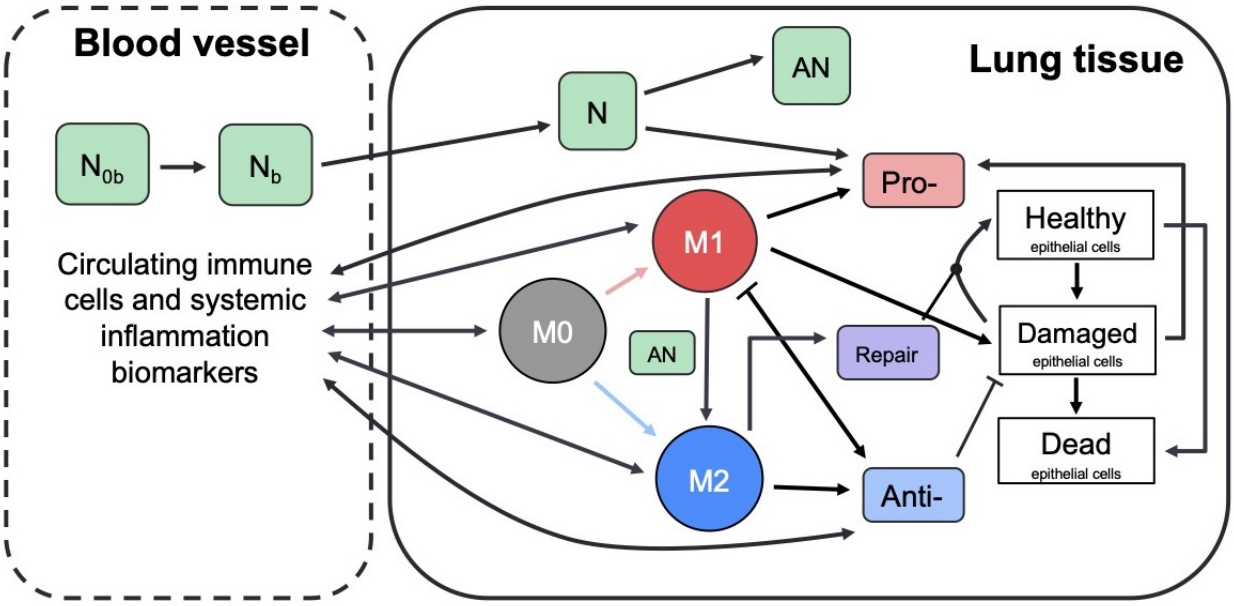


Figure 2.2: **Schematic describes interactions between immune system components.** Green boxes represent neutrophils: unactivated and activated neutrophils in the bloodstream (N_{0b} and N_0, respectively), activated and apoptotic neutrophils at the site of inflammation (N and AN, respectively). Circles represent M0, M1, and M2 macrophages, which perform a number of roles including removing debris and producing pro- and anti-inflammatory mediators. White boxes represent healthy, damaged, and dead epithelial cells/empty space. Pro- and anti-inflammatory mediators (red and blue boxes) recruit and activate immune cells (red and blue arrows indicate activation by pro- and anti-inflammatory mediators, respectively). Repair mediators (purple box) promote repair of damaged epithelial cells. Dynamics between cells and mediators in the blood (not shown) are similar to the detailed dynamics shown for local inflammation in lung tissue.

Bloodstream	Lung	Description
	E_h	Healthy epithelial cells
	E_d	Damaged epithelial cells
	E_e	Dead epithelial cells/empty space
p_b	p	Pro-inflammatory mediators
a_b	a	Anti-inflammatory mediators
M_{0b}	M_0	Naive macrophages
M_{1b}	M_1	M1 pro-inflammatory macrophages
M_{2b}	M_2	M2 anti-inflammatory macrophages
N_{0b}		Unactivated neutrophils
N_b		Activated neutrophils
	N	Neutrophils
	AN	Apoptotic neutrophils
	R	Repair mediators

Table 2.1: State variables for the model. Variables in both columns represent cells or mediators that diffuse between the two compartments.

of the lung. The second compartment is the adjacent blood vessel that provides additional immune support to the site of damage. Differentiating between these two compartments allows us to determine the concentrations of various immune cells and other mediators in each separate area. This is necessary so that in future work this model can be calibrated with blood and local data, which are often measured by different means and with different frequencies.

Fig 2.2 gives a detailed breakdown of the dynamics in the lung. The dynamics are similar for the same cells and mediators in the blood. Cell types tracked in each compartment are stated in Table 2.1. In the following subsections, we develop the equations for these variables. The parameters used in the equations are given in Table 2.2 with their description and range used during parameter sampling. Since data is not yet available to estimate these model parameters, we use Latin hypercube sampling and exploratory simulations to determine initial acceptable ranges. This process is described in further detail in Section 2.2.2.

Name	Description	Range used
$a_{b\infty}$	Relative effectiveness of a_b at inhibiting M_{0b} differentiation to M_{1b}	$[0.29, 67.35]$
a_∞	Relative effectiveness of a at inhibiting M_0 differentiation to M_1	$[0.13, 72.08]$
b_d	Baseline decay of damaged cells	$[1.06 \times 10^{-5}, 0.07]$
b_p	Baseline self-resolving repair of epithelial cells	$[0, 6.20]$
b_r	Baseline repair of damaged cells	$[9.79 \times 10^{-3}, 4.47]$
d_a	Rate of diffusion for a	$[0.19, 177.98]$
d_p	Rate of diffusion for p	$[0.34, 2.3 \times 10^3]$
d_{m0}	Rate of diffusion for M_0	$[0.24, 275.55]$
d_{m1}	Rate of diffusion for M_1	$[2.75 \times 10^{-3}, 19.8]$
d_{m2}	Rate of diffusion for M_2	$[0.14, 143.36]$
k_{am1}	Production rate of a by M_{1b} & M_1	$[0.01, 18.01]$
k_{am2}	Production rate of a by M_{2b} & M_2	$[2.43 \times 10^{-3}, 1.67]$
k_{an}	Rate at which neutrophils become apoptotic	$[0.01, 50.04]$
k_{anm1}	Rate of M_1 phagocytosis of AN	$[1.32 \times 10^{-3}, 0.69]$
k_{anm2}	Rate of M_2 phagocytosis of AN	$[2.71 \times 10^{-3}, 7.36]$
k_{em1}	Rate of phagocytosis of damaged cells by M_1	$[0.01, 16.03]$
k_{en}	Rate of phagocytosis of damaged cells by N	$[0.01, 16.03]$
k_{ep}	Rate of self-resolving repair mediated by p	$[0, 4.30]$
k_{er}	Rate of repair of damaged cells by R	$[1.47 \times 10^{-3}, 1.08]$
x_{er}	Regulates effectiveness of repair of damaged cells by R (Hill-type constant)	$[7.23 \times 10^{-3}, 4.13]$
k_{m0a}	Rate of differentiation of M_0 by a	$[0.01, 89.07]$
x_{m0a}	Regulates effectiveness of differentiation of M_0 by a (Hill-type constant)	$[0.16, 136.83]$
k_{m0ab}	Rate of differentiation of M_{0b} by a_b	$[1.15, 436.59]$
x_{m0ab}	Regulates effectiveness of a_b differentiation of M_{0b} (Hill-type constant)	$[0.16, 83.97]$
k_{m0ad}	Rate of recruitment of M_{0b} by a_b	$[0.34, 181.89]$
x_{m0ad}	Regulates effectiveness of recruitment of M_{0b} by a_b (Hill-type constant)	$[0.01, 27.6]$
k_{m0p}	Rate of differentiation of M_0 by p	$[8.99 \times 10^{-3}, 37.2]$
x_{m0p}	Regulates effectiveness of differentiation of M_0 by p (Hill-type constant)	$[1.17, 1.14 \times 10^4]$
k_{m0pb}	Rate of differentiation of M_{0b} by p_b	$[0.05, 89.96]$
x_{m0pb}	Regulates effectiveness of differentiation of M_{0b} by p_b (Hill-type constant)	$[41.51, 2.92 \times 10^4]$
k_{m0pd}	Rate of recruitment of M_{0b} by p_b	$[4.57 \times 10^{-3}, 53.97]$
x_{m0pd}	Regulates effectiveness of recruitment of M_{0b} by p_b (Hill-type constant)	$[0.24, 180.74]$
k_{m1p}	Rate of recruitment of M_{1b} by p_b	$[0.2, 92.81]$
x_{m1p}	Regulates effectiveness of recruitment of M_{1b} by p_b (Hill-type constant)	$[9.8 \times 10^{-3}, 1.69]$
k_{m2a}	Upregulation of M_{2b} recruitment by a	$[0.1, 219.93]$
x_{m2a}	Regulates effectiveness of M_{2b} recruitment by a (Hill-type constant)	$[0.08, 94.84]$
k_{m2r}	Upregulation of M_{2b} recruitment by R	$[3.61 \times 10^{-3}, 20.11]$
x_{m2r}	Regulates effectiveness of M_{2b} recruitment by R (Hill-type constant)	$[0.01, 18.70]$
k_{man}	Rate of M_1 switch to M_2 by AN	$[0.01, 27.08]$
k_{mne}	Rate of collateral damage to epithelial cells by macrophages and neutrophils	$[1.12 \times 10^{-3}, 5.17]$
x_{mne}	Regulates effectiveness of macrophages and neutrophils to damage epithelial cells (Hill-type constant)	$[0.03, 41.06]$
k_n	Rate of migration of N_b to lung	$[2.39 \times 10^{-3}, 3.54]$
k_{n0p}	Rate of activation of N_b by p	$[0.01, 5.58]$
x_{n0p}	Regulates effectiveness of activation of N_b by p (Hill-type constant)	$[0.03, 142.56]$
k_{pe}	Production rate of p by E_d	$[44.02, 1.12 \times 10^4]$
k_{pm1}	Production rate of p by M_1 & M_{1b}	$[0.24, 412.22]$
k_{pn}	Production rate of p and p_b by neutrophils	$[1.67 \times 10^{-3}, 2.95]$
k_{rm2}	Production rate of R by M_2	$[0.02, 40.97]$
μ_a	Decay rate of a	$[5.16 \times 10^{-4}, 5.08]$
μ_{ab}	Decay rate of a_b	$[0.04, 12.86]$
μ_p	Decay rate of p	$[2.76 \times 10^{-3}, 41.04]$
μ_{pb}	Decay rate of p_b	$[4.79 \times 10^{-4}, 3.71]$
μ_{m0}	Decay rate of M_0	$[0.01, 42.67]$
μ_{m0b}	Decay rate of M_{0b}	$[7.66 \times 10^{-3}, 329.59]$
μ_{m1}	Decay rate of M_1	$[8.2 \times 10^{-3}, 10.16]$
μ_{m1b}	Decay rate of M_{1b}	$[0.03, 60.32]$
μ_{m2}	Decay rate of M_2	$[0.27, 135.37]$
μ_{m2b}	Decay rate of M_{2b}	$[0.02, 16.51]$
μ_{nb}	Decay rate of N_b	$[2.49 \times 10^{-3}, 6.03]$
μ_{n0b}	Decay rate of N_{0b}	$[3.94 \times 10^{-6}, 2.1 \times 10^{-3}]$
μ_n	Decay rate of N	$[8 \times 10^{-3}, 4.32]$
μ_R	Decay rate of R	$[0.72, 761.75]$
s_a	Source rate of background a_b	$[5.75 \times 10^{-3}, 1.11]$
s_d	Rate of damage from ventilator	0.75
s_m	Source rate of M_{0b}	$[1.28, 1.14 \times 10^3]$
s_n	Source rate of N_{0b}	$[0.22, 225.45]$
s_p	Source rate of background p_b	$[6.5 \times 10^{-4}, 9.4]$

Table 2.2: Model parameters with short descriptions and ranges used in LHS.

Epithelial cells

We define the local lung epithelial cells to be a simplified approximation of the entire alveolar space, and track time in hours. We modeled this "local space" as the percentage of cells in three subpopulations, healthy (E_h), damaged (E_d), and dead epithelial cells/empty space needing be replaced by healthy cells (E_e). Thus, E_h, E_d, and E_e are dimensionless and $E_e + E_h + E_d = 1$. We depict these populations using Eqs (2.1), (2.2), and (2.3). These equations track proliferation and interactions between the epithelial and immune cells that are recruited in response to VILI. The first term in Eq (2.1) is a logistic growth, representing epithelial cells that spread and replicate to fill E_e. The factors $E_h + E_d$ and E_e delineate the areas taken up by cells and the empty space that can be filled by new cells. This term appears negated in Eq (2.3), modeling the removal of empty space. The proliferation rate is assumed to be b_p at baseline and it is modulated at a rate proportional to the pro-inflammatory mediator level, $k_{ep}p$. Nearby epithelial cells and progenitor cells, stem cells that can differentiate into specific types of epithelial cells only, perform this task. These cells spread and replicate to fill the empty space left by dead epithelial cells [25, 41, 48]. In this model we do not account for the progenitor cells. Therefore, we attribute all proliferation to the local epithelial cells.

The next term in Eq (2.1) and the first term of Eq (2.2) represents repair of damaged cells back to a healthy state. Epithelial cells are prone to self-repair [25], represented by a baseline rate b_r, and repair at a faster rate in the presence of repair mediators variable R, which tracks the level of mediators that promote epithelial repair such as fibronectin and other epithelial growth factors [43, 48, 113]. The third term in Eq (2.1) and second in Eq (2.2) represents collateral damage to epithelial cells by the influx and activity of the immune system. This mechanism is modeled via a nonlinear term, which is dependent on macrophage and neutrophil levels [4, 63, 92]. We also model damage due to ventilator-induced injury as $s_d E_h$, the fourth term in Eq (2.1) and fifth term in Eq (2.2), in which injury occurs at a rate proportional to the amount of healthy epithelial cells at a given time. This general term

covers over-distention for any mode of ventilation.

$$\frac{dE_h}{dt} = \overbrace{(b_p + k_{ep}p)(E_h + E_d)E_e}^{\text{Proliferation of healthy cells, upregulated by PIM}} + E_d\left(\overbrace{b_r}^{\text{Baseline repair}} + \overbrace{\frac{k_{er}R}{x_{er}+R}}^{\text{Upregulation via repair mediators}}\right) - \overbrace{E_h\left(\frac{k_{mne}(M_1+N)^2}{x_{mne}^2+(M_1+N)^2}\right)}^{\text{Damage via M1 \& neutrophils}} - \overbrace{s_dE_h}^{\text{Damage from ventilator}} \tag{2.1}$$

$$\frac{dE_d}{dt} = -\overbrace{E_d\left(b_r + \frac{k_{er}R}{x_{er}+R}\right)}^{\text{Baseline repair}} + \overbrace{E_h\left(\frac{k_{mne}(M_1+N)^2}{x_{mne}^2+(M_1+N)^2}\right)}^{\text{Damage via M1 \& neutrophils}} - \overbrace{k_{em1}M_1E_d}^{\text{Phagocytosis of damaged cells by M1}}\overbrace{\left(\frac{1}{1+\left(\frac{a}{a_\infty}\right)^2}\right)}^{\text{Inhibition by AIM}} - \overbrace{k_{en}NE_d}^{\text{Phagocytosis of damaged cells by N}} + \overbrace{s_dE_h}^{\text{Damage from ventilator}} - \overbrace{b_dE_d}^{\text{Death}} \tag{2.2}$$

$$\frac{dE_e}{dt} = -\overbrace{(b_p + k_{ep}p)(E_h + E_d)E_e}^{\text{Proliferation of healthy cells, upregulated by PIM}} + \overbrace{k_{em1}M_1E_d}^{\text{Phagocytosis of damaged cells by M1}}\overbrace{\left(\frac{1}{1+\left(\frac{a}{a_\infty}\right)^2}\right)}^{\text{Inhibition by AIM}} + \overbrace{k_{en}NE_d}^{\text{Phagocytosis of damaged cells by N}} + \overbrace{b_dE_d}^{\text{Death}} \tag{2.3}$$

M1 macrophages and neutrophils clear debris from the inflammation site to make room for healthy epithelial cells to divide and fill the empty space [25, 41, 61]. The third and fourth terms in Eq (2.2) and second and third in Eq (2.3) represent this phagocytosis of damaged cells by M1 macrophages and activated neutrophils, respectively. Regulation of M1 is modeled by the last multiplier in the term, representing inhibition by anti-inflammatory mediators (AIM), such as IL-10 [4, 48, 54]. The negative feedback loop of AIM inhibiting further pro-inflammatory functions occurs multiple times in our model. We will heretofore refer to this multiplier as inhibition by AIM. Depending on the compartment, the term may

utilize the variable a_b (bloodstream) or a (local). The anti-inflammatory and regulatory role of M2 macrophages and the balance between M1 and M2 phenotypes is critical for a successful and rapid recovery [48, 135]. The last term of Eqs (2.2) and (2.3), $b_d E_d$, represents the death of E_d (negative in Eq 2.2) and the associated gain in the E_e population (positive in Eq 2.3)).

Pro- and anti-inflammatory mediators

As a signal to the immune cells, damaged epithelial cells release pro-inflammatory cytokines and other mediators, including TNFα and matrix metalloproteinases (MMPs) [25, 41, 88]. In our equations, we track these pro-inflammatory mediators (PIM) in both compartments: p in the lungs and p_b in the blood. The release of PIM by damaged epithelial cells leads to diffusion of PIM into the bloodstream to recruit additional immune cells [41]. Movement from one compartment to another is assumed to be due to passive diffusion driven by the difference of the PIM concentrations between both compartments, first term in Eqs (2.4) and (2.5). This type of diffusion term will be used for all variables in our model that move bidirectionally from one compartment to the other.

M1 macrophages produce PIM, which upregulate the activation and migration of macrophages to the site of injury; see the second term in Eqs (2.4) and (2.5) [48, 88]. The macrophage population self-regulates by releasing AIM such as IL-10, thus inhibiting further production of PIM [73]. Therefore the production terms for PIM by M1 macrophages in both the blood and lung compartments include an inhibition multiplier. Therefore, the rate of PIM production by M1 macrophages decreases with increased concentrations of a_b or a.

Neutrophils are also important producers of pro-inflammatory mediators such as TNFα, IL-1, IL-6, LTB4, and chemokines, which stimulate the activation of macrophages toward an M1 phenotype [44, 61, 63, 113, 125], third term in Eqs (2.4) and (2.5). Low levels of PIM exist in the absence of damage, accounted for by the source term s_p in the second to last term of Eq (2.4) [9, 130]. The final terms of this equation and Eq (2.5) model the natural

decay of these mediators.

$$\frac{dp_b}{dt} = \overbrace{d_p(p - p_b)}^{\text{Diffusion}} + \overbrace{k_{pm1}M_{1b}}^{\text{Production via M1}} \overbrace{\left(\frac{1}{1 + \left(\frac{a_b}{a_{b\infty}}\right)^2}\right)}^{\text{Inhibition by AIM}} + \overbrace{k_{pn}N_b}^{\text{Production via neutrophils}} + \overbrace{s_p}^{\text{Background production}} - \overbrace{\mu_{p_b}p_b}^{\text{Decay}} \tag{2.4}$$

$$\frac{dp}{dt} = -\overbrace{d_p(p - p_b)}^{\text{Diffusion}} + \overbrace{k_{pm1}M_1}^{\text{Production via M1}} \overbrace{\left(\frac{1}{1 + \left(\frac{a}{a_\infty}\right)^2}\right)}^{\text{Inhibition by AIM}} + \overbrace{k_{pn}N}^{\text{Production via neutrophils}} + \overbrace{k_{pe}E_d}^{\text{Production via ep. damage}} - \overbrace{\mu_p p}^{\text{Decay}} \tag{2.5}$$

Anti-inflammatory mediators, such as the anti-inflammatory signaling caused by IL-4 and IL-10 [95], are represented by Eq (2.6) in the bloodstream and Eq (2.7) at the site of damage. The first term in each equation models diffusion. AIM are released by both M1 and M2 macrophages [48, 54, 88], shown in the second and third terms of Eqs (2.6) and (2.7). Similarly to p_b, background levels of a_b are present in the absence of an immune response, represented by term four in Eq (2.6) [9]. Natural decay of AIM is accounted for by the last term in each equation.

$$\frac{da_b}{dt} = \overbrace{d_a(a - a_b)}^{\text{Diffusion}} + \overbrace{k_{am1}M_{1b}}^{\text{Production via M1}} + \overbrace{k_{am2}M_{2b}}^{\text{Production via M2}} + \overbrace{s_a}^{\text{Background production}} - \overbrace{\mu_{a_b}a_b}^{\text{Decay}} \tag{2.6}$$

$$\frac{da}{dt} = \overbrace{-d_a(a - a_b)}^{\text{Diffusion}} + \overbrace{k_{am1}M_1}^{\text{Production via M1}} + \overbrace{k_{am2}M_2}^{\text{Production via M2}} - \overbrace{\mu_a a}^{\text{Decay}} \tag{2.7}$$

M0 macrophages

M0 macrophages, also called naive or undifferentiated, are present both locally and in the blood. The diffusion term, seen in the first term of Eqs (2.8) and (2.9), represents movement between compartments. The baseline diffusion between compartments is modeled in the same manner as with other variables, but the rate at which this diffusion occurs is modulated by mediators. Increased PIM and AIM levels cause undifferentiated macrophages in the bloodstream to be recruited at a higher rate to the damaged site, where they differentiate and perform phagocytic, pro-inflammatory, and pro-resolving roles [88]. This increased flux between compartments due to the presence of p_b and a_b is modeled by adding to the baseline diffusion rate (d_{m0}). The added term is a Michaelis-Menten-type term to capture the increasing rate as mediators rise, with a maximum rate at which these cells can diffuse, $(d_{m0} + k_{m0pd} + k_{m0ad})$.

MV induces epithelial cells to produce pro-inflammatory mediators such as TNFα, chemokines, and interleukins (ILs) [45]. Undifferentiated macrophages receive these signals and differentiate into the M1 phenotype [134]. Eq (2.8) accounts for activation to M1 and M2 in the bloodstream by PIM and AIM, respectively, given a high enough concentration of these mediators [4]. Although there is still debate on the types of macrophages that exist in the bloodstream after being released from the bone marrow, there is evidence that populations of both M1 and M2 exist in the bloodstream before being recruited to the site of injury [54, 88]. Thus, we include this process in our equations in the second term of Eq (2.8). Undifferentiated macrophages in the bloodstream can change phenotype to M1 or M2 after interacting with PIM or AIM, respectively, modeled by a Hill-type term. This nonlinearity accounts for the sufficient amount of PIM or AIM needed to activate M_0 and that this process saturates to a maximum rate of k_{m0pb} and k_{m0ab} for activation by pro- and anti-inflammatory mediators, respectively.

The second term in Eq 2.9 represents activation of undifferentiated macrophages in the lung compartment to the pro-inflammatory phenotype, downregulated by the anti-

inflammatory response through an inhibition multiplier. In this term, M2 macrophages can also be activated directly from the naive phenotype by various repair and anti-inflammatory mediators involved in the repair of epithelial cells [41, 48].

Using the same inhibition multiplier as previously, AIM inhibit differentiation to M1 as part of their regulatory role in the inflammatory process, although a complete understanding of these mechanisms is yet to be uncovered [41, 73, 88]. In the absence of injury, lungs contain a low number of undifferentiated macrophages which patrol the surrounding area [25]. "Patrolling" macrophages are also prevalent in the bloodstream. The third term in Eq (2.8) represents a constant source of undifferentiated macrophages into circulation [48]. We also account for natural decay of all macrophage phenotypes in the last term of Eqs (2.8) through (2.13).

$$\begin{aligned}\frac{dM_{0b}}{dt} =& \overbrace{(M_0 - M_{0b})\left(d_{m0} + \frac{k_{m0pd}p_b}{x_{m0pd} + p_b} + \frac{k_{m0ad}a_b}{x_{m0ad} + a_b}\right)}^{\text{Diffusion, upregulated by PIM \& AIM}} \\ &- M_{0b}\left[\overbrace{\left(\frac{k_{m0pb}p_b^2}{x_{m0pb}^2 + p_b^2}\right)}^{\substack{\text{Differentiation}\\ \text{to M1 via PIM}}}\overbrace{\left(\frac{1}{1 + \left(\frac{a_b}{a_{b\infty}}\right)^2}\right)}^{\substack{\text{Inhibition}\\ \text{by AIM}}} + \overbrace{\left(\frac{k_{m0ab}a_b^2}{x_{m0ab}^2 + a_b^2}\right)}^{\substack{\text{Differentiation}\\ \text{to M2}}}\right] \\ &+ \overbrace{s_m}^{\text{Source}} - \overbrace{\mu_{M0_b}M_{0b}}^{\text{Decay}} \end{aligned} \tag{2.8}$$

$$\begin{aligned}\frac{dM_0}{dt} =& -\overbrace{(M_0 - M_{0b})\left(d_{m0} + \frac{k_{m0pd}p_b}{x_{m0pd} + p_b} + \frac{k_{m0ad}a_b}{x_{m0ad} + a_b}\right)}^{\text{Diffusion, upregulated by PIM \& AIM}} \\ &- M_0\left[\overbrace{\left(\frac{k_{m0p}p^2}{x_{m0p}^2 + p^2}\right)}^{\substack{\text{Differentiation}\\ \text{to M1 via PIM}}}\overbrace{\left(\frac{1}{1 + \left(\frac{a}{a_\infty}\right)^2}\right)}^{\substack{\text{Inhibition}\\ \text{by AIM}}} + \overbrace{\left(\frac{k_{m0a}a^2}{x_{m0a}^2 + a^2}\right)}^{\substack{\text{Differentiation}\\ \text{to M2}}}\right] \\ &- \overbrace{\mu_{M_0}M_0}^{\text{Decay}} \end{aligned} \tag{2.9}$$

M1 macrophages

Similarly to naive macrophages, M1 macrophages move between compartments. The presence of pro-inflammatory mediators, which act as recruiters, increases the rate of diffusion, shown in the first term of Eq (2.10) [88]. The second term represents differentiation from the naive state, as described for the associated term in M_0.

Macrophages exhibit high plasticity, and based on the mediators and other immune cells they encounter, they can switch phenotype and perform different or enhanced functions; this plasticity is not yet fully understood [4, 48]. M1 macrophages are primarily responsible for producing PIM, thereby recruiting other immune cells to the damaged area [54]. M2 macrophages are considered pro-resolving and downregulate PIM. Both M1 and M2 macrophages phagocytize apoptotic cells such as neutrophils [113]. The shift from an overall pro-inflammatory phase to an anti-inflammatory phase in the course of the immune response is highly dependent upon a shift in macrophage behavior, specifically the shift from a mainly M1 response to a mainly M2 response [41, 54, 88].

One of the primary ways this shift is achieved is through the inhibition of M0 to M1 differentiation by anti-inflammatory mediators, as described previously. Additionally, when pro-inflammatory macrophages phagocytize apoptotic neutrophils, they shift towards a more anti-inflammatory phenotype. This results in suppression of the release of pro-inflammatory mediators and production of pro-resolving mediators [63, 92]. We account for this shift by including the third term in Eq (2.11) to account for M1 macrophages shifting to the M2 phenotype when they phagocytize apoptotic neutrophils. This term is proportional to the term in the apoptotic neutrophil equation, Eq (2.17), modeling the phagocytosis of apoptotic neutrophils by $M1$. This term also includes inhibition of M1 function by AIM. It has been shown in some studies that M2 macrophages can switch to an M1 phenotype [49], although this idea is not currently widely accepted. Thus, we choose to include only the shift from M1 to M2.

$$\frac{dM_{1b}}{dt} = \overbrace{(M_1 - M_{1b})\left(d_{m1} + \frac{k_{m1p}p_b}{x_{m1p} + p_b}\right)}^{\text{Diffusion, upregulated by PIM}} + M_{0b}\overbrace{\left(\frac{k_{m0pb}p_b^2}{x_{m0pb}^2 + p_b^2}\right)}^{\substack{\text{Differentiation}\\ \text{to M1}}}\overbrace{\left(\frac{1}{1 + \left(\frac{a_b}{a_{b\infty}}\right)^2}\right)}^{\substack{\text{Inhibition}\\ \text{by AIM}}} - \overbrace{\mu_{M1_b}M_{1b}}^{\text{Decay}} \tag{2.10}$$

$$\frac{dM_1}{dt} = -\overbrace{(M_1 - M_{1b})\left(d_{m1} + \frac{k_{m1p}p_b}{x_{m1p} + p_b}\right)}^{\text{Diffusion, upregulated by PIM}} + \overbrace{M_0\left(\frac{k_{m0p}p^2}{x_{m0p}^2 + p^2}\right)}^{\substack{\text{Differentiation}\\ \text{to M1 via PIM}}}\overbrace{\left(\frac{1}{1 + \left(\frac{a}{a_\infty}\right)^2}\right)}^{\substack{\text{Inhibition}\\ \text{by AIM}}} - \overbrace{k_{man}(k_{anm1}ANM_1)}^{\substack{\text{M1 switch to M2}\\ \text{by phagocytosis}}}\overbrace{\left(\frac{1}{1 + \left(\frac{a}{a_\infty}\right)^2}\right)}^{\substack{\text{Inhibition}\\ \text{by AIM}}} - \overbrace{\mu_{M_1}M_1}^{\text{Decay}} \tag{2.11}$$

M2 macrophages

M2 macrophages, associated with an anti-inflammatory response, can be activated directly from undifferentiated macrophages by specific anti-inflammatory signals in addition to switching phenotype from M1. They diffuse between compartments modeled in the first terms of Eqs (2.12) and (2.13). M2 macrophages produce anti-inflammatory mediators which recruit and promote differentiation to more M2 macrophages, described in the second term of both equations. They release cytokines that trigger the repair phase of the immune response [48, 88]. This repair phase includes repair mediators (discussed in Eq (2.18)), which play a direct role in the reconstruction of healthy epithelial cells and resolution of damage [48].

$$\frac{dM_{2b}}{dt} = \overbrace{(M_2 - M_{2b})\left(d_{m2} + \frac{k_{m2r}R}{x_{m2r}+R} + \frac{k_{m2a}a}{x_{m2a}+a}\right)}^{\text{Diffusion}} + \overbrace{M_{0b}\left(\frac{k_{m0ab}a_b^2}{x_{m0ab}^2 + a_b^2}\right)}^{\substack{\text{Differentiation}\\ \text{to M2}}} - \overbrace{\mu_{M2_b} M_{2b}}^{\text{Decay}} \tag{2.12}$$

$$\frac{dM_2}{dt} = -\overbrace{(M_2 - M_{2b})\left(d_{m2} + \frac{k_{m2r}R}{x_{m2r}+R} + \frac{k_{m2a}a}{x_{m2a}+a}\right)}^{\text{Diffusion}} + \overbrace{M_0\left(\frac{k_{m0a}a^2}{x_{m0a}^2 + a^2}\right)}^{\substack{\text{Differentiation}\\ \text{to M2}}} + \overbrace{k_{man}(k_{anm1}ANM_1)}^{\substack{\text{M1 switch to M2}\\ \text{by phagocytosis}}} \overbrace{\left(\frac{1}{1+\left(\frac{a}{a_\infty}\right)^2}\right)}^{\substack{\text{Inhibition}\\ \text{by AIM}}} - \overbrace{\mu_{M_2} M_2}^{\text{Decay}} \tag{2.13}$$

Neutrophils

Neutrophils are considered the first responders to injury [41, 44]. Generated in the bone marrow, free-flowing neutrophils, described as N_{0b}, circulate in the lung vasculature at baseline levels [61] and are represented by the first term in Eq (2.14). In the presence of injury, neutrophils are activated and recruited to the damaged site through pro-inflammatory mediators such as TNFα, IL-1β, and other chemokines and cytokines [44, 125]. This recruitment is represented by the first term in Eqs (2.14) and (2.15). On the other hand, anti-inflammatory mediators, including macrophage-produced resolvins and protectins, inhibit further recruitment of neutrophils [92]. Similarly to the differentiation of macrophages, it is assumed that a neutrophils activation is nonlinear and that it saturates. Therefore, a Hill-type term with a maximum rate of k_{n0p} and a constant of x_{n0p} is used to model activation of neutrophils by PIM. To model the inhibition of neutrophil activation by AIM, we include the same inhibition multiplier as previously described. The effectiveness of AIM to inhibit this process is controlled by $a_{b\infty}$. We also account for intrinsic decay of neutrophils in the last term of Eqs (2.14) through (2.16).

$$\frac{dN_{0b}}{dt} = -\overbrace{N_{0b}\left(\frac{k_{n0p}p_b^2}{x_{n0p}^2+p_b^2}\right)}^{\text{Activation by PIM}}\overbrace{\left(\frac{1}{1+\left(\frac{a_b}{a_{b\infty}}\right)^2}\right)}^{\text{Inhibition by AIM}} + \overbrace{s_N}^{\text{Source}} - \overbrace{\mu_{N_{0b}}N_{0b}}^{\text{Decay}} \tag{2.14}$$

$$\frac{dN_b}{dt} = \overbrace{N_{0b}\left(\frac{k_{n0p}p_b^2}{x_{n0p}^2+p_b^2}\right)}^{\text{Activation by PIM}}\overbrace{\left(\frac{1}{1+\left(\frac{a_b}{a_{b\infty}}\right)^2}\right)}^{\text{Inhibition by AIM}} - \overbrace{k_nN_b}^{\text{Migration}} - \overbrace{\mu_{N_b}N_b}^{\text{Decay}} \tag{2.15}$$

Neutrophils go through a multi-step process of rolling along and subsequently adhering to the surface of the endothelium. Then neutrophils transmigrate to the injury site either through or between endothelial cells [44, 61]. This process is assumed to be driven not by a concentration difference in neutrophils between the compartments but rather is a direct consequence of activation. Therefore, neutrophil transmigration, the first term in Eq (2.16), is modeled from the bloodstream to the site of injury by a linear term with rate k_n.

Activated neutrophils that have transmigrated through the endothelium and reached the site of injury release pro-inflammatory mediators, as discussed previously in Eq (2.5). During infection, neutrophils play an important role by phagocytizing pathogens [63], but during VILI a main role of neutrophils is the recruitment of macrophages, particularly to promote a more pro-inflammatory environment for the clearance of damaged and dead cells [44].

Neutrophils become apoptotic, modeled by the second term of Eq (2.16) [41]. In this state, they are phagocytized by M1 and M2 macrophages (second and third terms of Eq (2.17), respectively) and no longer contribute to the production of PIM [61, 113, 123]. Phagocytosis by M1 macrophages is inhibited by AIM using our standard functional form for the inhibition multiplier. AIM do not inhibit phagocytosis by M2 macrophages since AIM support the function of anti-inflammatory cells.

$$\frac{dN}{dt} = \overbrace{k_n N_b}^{\text{Migration}} - \overbrace{k_{an} N}^{\text{Transition to apoptotic}} - \overbrace{\mu_n N}^{\text{Decay}} \tag{2.16}$$

$$\frac{dAN}{dt} = \overbrace{k_{an} N}^{\text{Transition to apoptotic}} - \overbrace{k_{anm1} AN M_1}^{\text{Phagocytosis by M1}} \overbrace{\left(\frac{1}{1 + \left(\frac{a}{a_\infty}\right)^2} \right)}^{\text{Inhibition by AIM}} - \overbrace{k_{anm2} AN M_2}^{\text{Phagocytosis by M2}} \tag{2.17}$$

Repair mediators

The direct contribution of alveolar macrophages to the repair of epithelial cells is not completely understood, although macrophage involvement in the repair process has been widely demonstrated [48]. M2 macrophages produce various mediators such as prostaglandin E_2, chemokines such as CCL2, TGF-β, fibronectin 1 and other epithelial growth factors [43, 48, 113] that promote repair of epithelial cells and recruit fibroblasts, key cells involved in tissue repair [105]. We do not model each of these components, instead grouping them together in one variable called R, which can be thought of as the downstream effects of fibroblasts and other mediators. If the recovery phase is the focus of a future study this model could be adapted to include these dynamics explicitly. The production of R by M2 macrophages is modeled by the first term in Eq (2.18). The second term models intrinsic decay of these mediators.

$$\frac{dR}{dt} = \overbrace{k_{rm2} M_2}^{\text{Upregulation by M2}} - \overbrace{\mu_R R}^{\text{Decay}} \tag{2.18}$$

These equations form a system of ODEs that captures the most important aspects of the immune response to VILI. In the following sections we describe various computational approaches used to explore parameter space, determine the parameters the model is most sensitive to and establish influential predictors of model outcomes. We end with case stud-

ies in which we modulated particular parameters and then evaluated long-term epithelial damage.

2.2.2 Sampling method for parameters: Latin hypercube sampling

Because of the large number of variables and parameters, mathematical and statistical techniques needed to be used to analyze the system and find parameter sets that generate a variety of dynamics and outcomes of immune cell populations included in this model. At this stage we analyzed the model with various parameters without utilizing data; in future work this model can be coupled with ventilation experiments to narrow parameter ranges. As an initial step we determined initial conditions and parameters for this model through LHS, which generates random, unique parameter sets according to user-defined distributions [83]. As suggested by Marino et al. [76], we initially chose uniform distributions since we had no prior knowledge of the parameter values, and sampled on a logarithmic scale to cover a span of several magnitudes. For LHS with uniform distributions assumed for each parameter, to generate n desired parameter sets, the algorithm splits the determined range into n evenly-spaced subintervals and each interval is sampled exactly once [76]. We also sampled using log-normal distributions for each parameter with the same means and variances as the uniform distributions to see whether restriction of the parameter space by bounded intervals, as enforced by the uniform distributions, affected our results. We sampled using log-normal, rather than normal, distributions to ensure the parameters were positive.

Using MATLAB functions adapted from Kirschner [59], all parameters were sampled except the rate of damage s_d due to ventilation to ensure the same insult during all simulations. We began to explore parameter space by sampling near transients associated with different outcomes. Ranges were set such that that the resulting sampling gave rise to a variety of behaviors and outcomes. Table 2.2 shows the final ranges used for the LHS sweep that constructed the collection of parameter sets used in this work. Using LHS in these ranges we generated 100,000 parameter sets. Future work could calibrate parameter sets to data from

different experimental or clinical groups and then use the analysis methods in this manuscript to compare dynamics and parameters that drive differences between experimental or clinical groups.

2.2.3 Parameter Set Collections: Healthy, Moderate Damage, & Severe Damage

Our goal was to understand the effects of baseline lung health, represented by initial conditions and unique parameter sets, on the response to ventilation and post-ventilation recovery. Therefore we needed to start our simulation from initial conditions associated with a steady state, so that when ventilation was simulated we were seeing changes in the dynamics only due to the ventilator. For all 100,000 parameter sets we simulated the model for 800 hours, without any ventilation ($s_d = 0$), to determine if a numerical steady-state condition was reached in the absence of ventilation. Our numerical steady state condition was that the l^2-norm of the difference between each meshpoint in the last 100 hours of the simulation and the last point (hour 800) was less than 0.1. By examining simulation results, we confirmed that this ensured minimal change in all variables at the tail end of the simulation.

Three different initial conditions were used with the sampled parameter sets for the 800-hour, non-vent simulation, in order to find sets that have steady states. The first set of initial conditions was associated with initial simulations used to develop the sampling ranges, but was not associated with a particular set in our final round of sampling. The second initial condition we chose was associated with an insult to the epithelial cells with no initial immune response, all variables set to zero except for $E_h(0) = 0.75$ and $E_d(0) = 0.25$. The third and final initial condition had all variables set to zero except for $M1(0) = 50$, which is starting with an activated immune response and healthy tissue. If our numerical steady-state condition was satisfied with any of these initial conditions, the parameter set was accepted and the associated initial conditions were set to the variable values at 800 hours. A total of 27,836 sets satisfied our numerical steady state condition. Any parameter set that

did not result in an equilibrium state by 800 hours from these three initial conditions was not simulated with ventilation. Since we did not perform a complete analysis on all 100,000 sets, we do not mathematically conclude that the remaining parameters cannot reach a steady state. However, given the robustness of the resulting dynamics and the number of parameter sets that reached our condition for numerical steady state, we assumed that actual biological dynamics were well represented by simulating these 27,836 unique parameter sets. The same process was applied to the log-normally-distributed collection of parameter sets, generating a total of 33,812 sets that reached a steady state. Results throughout this manuscript were similar for the sets generated using log-normal distributions, see Section A.2.

Some parameter sets gave rise to a steady state with associated initial conditions where the percent of empty space in the epithelium was significantly high. Therefore, we eliminated some sets based on their initial condition for E_e (empty/dead cells). In this paper we focus on the 24,432 parameters sets that had a steady state for $E_e < 50\%$ and show a summary of all results for steady states with $E_e < 25\%$ and $E_e < 75\%$ in Section A.2. We did not find any major differences when varying this inclusion threshold.

These 24,432 sets were then simulated for 200 hours with ventilation for the first two hours (a nonzero damage rate), a duration comparable with murine experiments [121, 141]. Given that all mice do not survive ventilation, we adjusted our model to account for extensive lung damage due to ventilation, leading to severe inflammation. Without this adjustment, the model assumes survival in all scenarios and allows for a recovery phase. Instead, we assumed that a high percentage of empty space E_e is not survivable; therefore, we set a threshold for E_e. Ideally this threshold's value would be derived from data. However, in the absence of data related to epithelial integrity, we used a threshold of 75% given that we had set a threshold of 50% for $E_e(0)$. These two thresholds combined map the arbitrary 0 to 100% epithelial population to metrics of overall lung health. E_e more than 50% without ventilation is not survivable and more than 75%, even with MV, is not survivable. Therefore, if E_e rises above 75% at any time, variables are set to 0 at that time.

Simulations were separated into three different categories based on percentage of healthy epithelial cells at time of classification T:

- $E_h(T) \geq 90\%$: Healthy epithelial cells sufficiently cover the alveoli to function normally
- $50\% \leq E_h(T) < 90\%$: Moderate tissue damage
- $0\% \leq E_h(T) < 50\%$: Severe tissue damage

Sets were classified into the three categories based on their initial conditions and again at two other time points: immediately after ventilation (2 hours) and after ventilation with a recovery period (200 hours). Classification at these two time points allowed us to understand the immediate effects of VILI as well as the long-term effects after a period of recovery. These parameter sets, their corresponding transients, and the outcomes they generate were used to develop a collection of parameter sets representing a variety of immune system dynamics. The collection was then used to analyze outcomes in terms of their associated transient variables and underlying parameters.

2.2.4 eFAST

We used several tools to perform a sensitivity analysis of model parameters. A common method is calculating partial rank correlation coefficients (PRCCs), but results are only reliable for monotonic relationships between parameters and variables. Our model output does not fit this criteria. Marino et al. suggest the extended Fourier amplitude sensitivity test (eFAST), a variance-based method for non-linear, non-monotonic relationships [76]. The greatest drawback of eFAST compared to PRCC is the computation time.

eFAST varies parameters and the resulting variation in model output is calculated using statistical variance. The algorithm varies each parameter at different frequencies by creating a sinusoidal function, called a search curve, and then sampling parameter values along the function. Fourier analysis measures the influence of the parameter's frequency on model output. First-order sensitivity S_i for a parameter i is calculated by varying only i and leaving

the rest constant. Total-order sensitivity S_{Ti} is calculated by varying i using a unique, higher frequency and varying the other parameters using lower non-unique frequencies. This total-order sensitivity captures non-linear interactions between parameters in addition to changes in model output. We implemented the method by Marino et al. [76] to calculate S_i and S_{Ti} and determined the statistical significance of each parameter. A "dummy parameter" was included in the parameter set and its eFAST index was compared to the other parameters found in the model.

MATLAB functions by Kirschner [59] to perform eFAST are available online. We obtained 257 values of each parameter on a search curve and repeated this process for five unique search curves since different ones can generate slightly different samples. Sensitivity can be calculated at specific time points for the desired variable.

2.2.5 Random forest decision tree

Aside from more conventional sensitivity analysis measures, we chose a few alternative methods that require less computation time and can include other features of the model besides parameters. One of these alternatives is a random forest decision tree [65, 66]. Each parameter set in the collection has a number of predictors and outputs: parameters and any other characteristics from the transients that can be quantified. The decision tree algorithm determines the parameter/predictor values that best partition the collection into categories of healthy, moderate damage, and severe damage. Each member of the collection answers a series of questions, i.e. nodes on the tree, based on the predictor values of that parameter set, eventually being classified into a particular outcome. This process is repeated to obtain a "forest" of decision trees.

Since a decision tree simply takes value for each predictor and is not dependent on the model itself, measures besides parameters can be used. We included supplementary predictors calculated from the transients. These predictors are: maximum and minimum M1 and M2 (percent of total macrophages and raw values), time at which M1 and M2

maximums occur (M1/M2 peak time), ratio of M1 peak to M1 initial condition, percent M2 macrophages at 10 hours, ratio of E_h initial condition to E_h at 30 minutes, 2 hours, and 6 hours, and the difference between E_h initial condition and E_h at 200 hours. Adding these predictors allowed for the possibility that the best classifiers of outcome could be not only parameters but also properties of the transients. This knowledge could provide additional information about metrics for experimentalists and clinicians to track in order to identify early warning signs for undesirable results.

One metric generated by the random forest is the importance value of each parameter or characteristic, calculated from the Gini Index [6]. The importance value is a measure of how important any given parameter was in determining the outcome of each set in the collection. Because of the large number of parameters in the model, this can provide intuition about which parameters and other characteristics of the transients are most influential in determining outcomes. The R and MATLAB code used for this method are provided in Sections A.4-A.6.

2.3 Results

Our aim is to understand how recruitment of the immune response and its interactions with epithelial cells translate to specific outcomes and what dynamics are driving this process. Using the techniques described in the previous section, we determined predictors of outcome and/or processes that could be targeted to modulate outcome.

2.3.1 Sample Transients and Collection Breakdown

This model can generate a variety of dynamics, similar to the mixed responses of patients on a ventilator [99]. Our model generates a variety of dynamics which reflects this spectrum of responses. There is significant variability between outcomes as well as within them. Fig 2.3 shows examples of these different dynamics for healthy epithelial cells and M0, M1, and M2

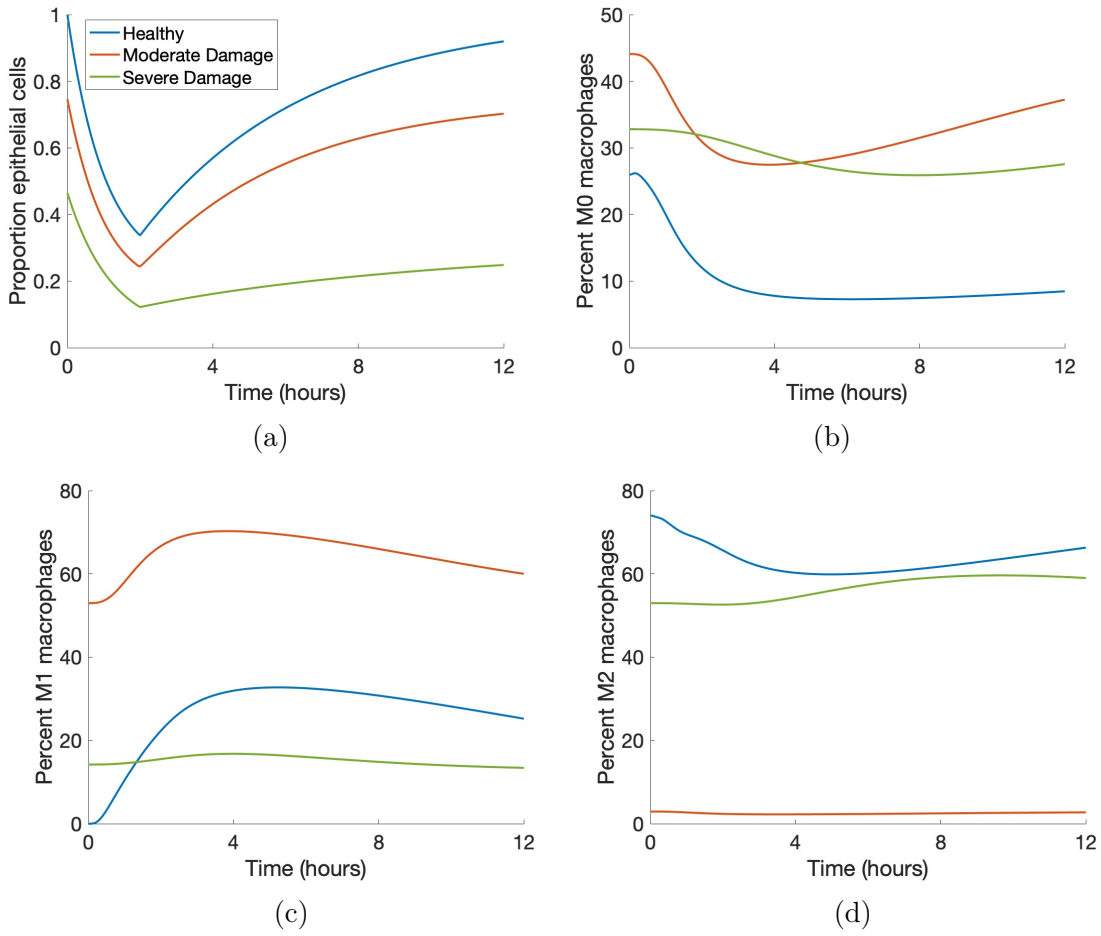


Figure 2.3: **Sample simulations show the variety of model-generated dynamics.** Blue, orange, and green curves indicate healthy, moderate damage and severe damage outcomes, respectively. (a) Proportion healthy epithelial cells. (b) Percent M0 macrophages. (c) Percent M1 macrophages. (d) Percent M2 macrophages.

macrophages using a case of each of the three outcomes as determined at 200 hours: healthy, moderate damage, and severe damage. All three can be classified as "severe damage" at 2 hours. Each case has a unique set of initial conditions and parameters, giving rise to three very different immune responses and epithelial cell health. Simulations were run in MATLAB using the code provided in Section A.4.

We generated 100,000 parameter sets using LHS with parameter ranges given in Table 2.2. Fig 2.4 shows the breakdown of these parameter sets based on whether or not the dynamics led to a steady-state system and whether the steady state value had $E_e \leq 50\%$ in the absence of ventilation. Additionally, Figure 2.4 shows the classifications of each parameter

Total LHS runs: 100,000

Not steady-state: 77,568

Steady-state: 24,432		
Classification after 2 hours		
	Start	End
H:	16,833 (14,260)	2,573 (0)
M:	5,382 (3,387)	10,116 (8,121)
S:	2,217 (0)	11,743 (9,526)
Classification after 200 hours		
	Start	End
H:	16,833 (635)	16,198 (0)
M:	5,382 (572)	5,104 (294)
S:	2,217 (0)	3,130 (913)

Figure 2.4: **Results of 100,000 LHS runs grouped by classification.** Parameter sets are broken down by their associated initial conditions (Start) and ending states (End) and by category healthy (H), moderate damage (M), or severe damage (S). Numbers in parentheses in the IC columns are the number of simulations that started in the category associated with that row and change their state after ventilation. Numbers in parentheses in the ES columns are the number of simulations that ended in the category associated with that row, but were not in that category before ventilation. The first three rows in the table show classification immediately after a 2-hour period of ventilation. The last three rows show classification after 200 hours (a 2-hour vent and period of recovery). All parameter sets are associated with a steady-state solution with $E_e < 50\%$.

set based on their associated initial conditions (before ventilation), immediately after the 2-hour ventilation, and after the 2-hour ventilation with a recovery period (200 hours total). The top number in each box is the total number of parameter sets in that category, and that number is further broken down by the category in which they started (column 1) and ended (column 2). The number in parentheses in the first column is the number of sets that started in that category but ended in a different one. Conversely, the number in parentheses in the second column shows the sets that ended in a certain outcome but did not start there. These numbers serve as a summary of how damage may affect outcome directly after ventilation as well as after a recovery period for the variety of behaviors in the collection of parameter sets. We will analyze all 24,432 sets that reach steady state (with steady state $E_e < 50\%$) to understand the full array of responses that could occur.

2.3.2 Determining Predictors and Driving Dynamics

In this section we examine and compare the results of multiple methods that determine the parameters and other predictors that help differentiate or predict model dynamics and outcomes.

Correlations and Significance Testing Highlight Specific Parameters

As an initial step towards understanding relationships between parameters and model output, we calculated the correlations of parameters and predictors with one another for each outcome. There were some correlations between predictors that were very high, but were measuring similar things; for example, maximum M1 and minimum M1. We excluded these since they did not provide new or useful information. Aside from these, there were only a few correlations between parameters or between parameters and predictors that were higher than $R = 0.3$. The pair with the highest correlation, for outcome determined at both 2 and 200 hours, is shown in Fig 2.5 using a random sample from each classification group for better visibility of the points. For k_{mne}, the rate of collateral damage to epithelial cells by macrophages and neutrophils, parameter sets that resulted in moderate and severe damage outcomes had a significant correlation with the E_h ratio at 0.5 hours. The E_h ratio and k_{mne} had the following correlations for each group with classification at 2 hours: healthy $R = 0.24$ (not shown), moderate damage $R = 0.43$, and severe damage $R = 0.73$. For classification at 200 hours, the correlations were: healthy $R = 0.1$ (not shown), moderate damage $R = 0.66$, and severe damage $R = 0.87$. These high correlations suggest that the parameter k_{mne} may play a key role in determining outcome, which we explored further in the following sections.

We also performed hypothesis testing for predictors. We were not able to use ANOVA, a common statistical model used to examine the difference between group means, because the resulting distributions for the accepted parameter sets were not necessarily normal. The Kruskal-Wallis test is an alternative to ANOVA when the variable distributions are not normal [84]. We categorized all parameter sets by their outcome (healthy, moderate damage,

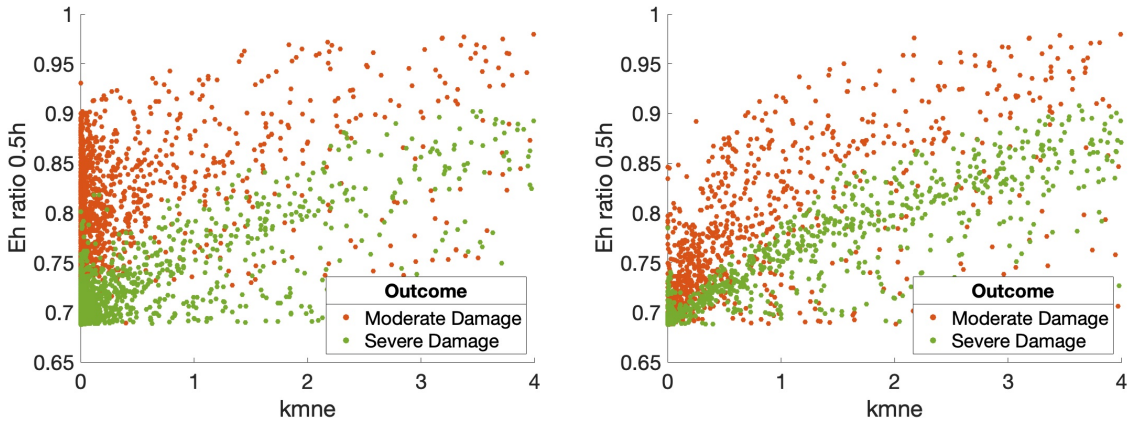


Figure 2.5: **Scatter plot of predictors with notable correlations.** Parameter k_{mne} (rate of collateral damage to epithelial cells by macrophages and neutrophils) versus ratio of E_h at 0.5 hours to initial E_h values. (a) Outcome was determined at 2 hours. Correlations: resolved to healthy $R = 0.24$ (not shown); moderate damage $R = 0.43$; severe damage $R = 0.73$. (b) Outcome was determined at 200 hours. Correlations: resolved to healthy $R = 0.1$ (not shown); moderate damage $R = 0.66$; severe damage $R = 0.87$. Points are a random sample of the total points.

severe damage) and compared them. If any of the three groups had a statistically significant difference (p-value less than 0.05), a Wilcoxon test was performed on each pair (healthy and moderate damage, healthy and severe damage, moderate and severe damage) to determine which groups were different from one another. P-values for the Kruskal-Wallis and Wilcoxon tests were adjusted using the Benjamini–Hochberg procedure to control for the false discovery rate [8]. Knowledge of which parameters and other predictors were different between groups based on outcome provides insight into predicting outcomes and which predictors might best influence the immune response to damage.

When classification occurred at 2 hours, 52 out of 81 parameters and other predictors returned results for a statistically significant difference between at least two groups and 30 gave statistically significant differences between all three groups. For classification at 200 hours, statistically significant differences occurred for at least two groups and all three groups for 40 and 13 predictors, respectively. Table 2.3 shows a summary of the results from the various methods used to examine predictors' significance in determining model output.

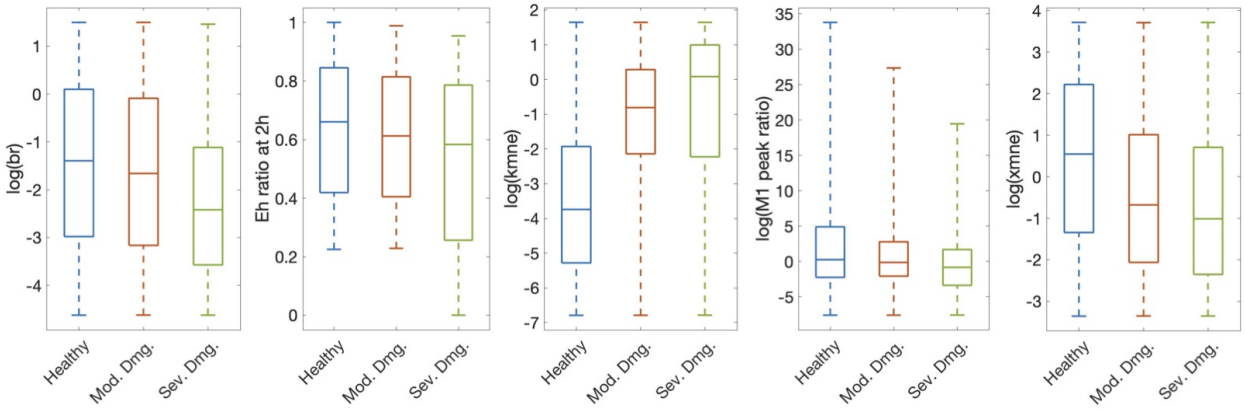


Figure 2.6: **Predictors selected by significance testing show visible differences between injury groups.** Subset of parameters and predictors that showed a statistically significant difference between all three outcomes determined at 200 hours: healthy, moderate damage, and severe damage, as determined by the Kruskal-Wallis and Wilcoxon tests. These five predictors were also statistically significant when classification occurred at 2 hours. All are shown on a log scale for better visibility. Parameters/predictors: b_r, baseline repair rate of damaged cells; E_h ratio at 2h, ratio of E_h at 2 hours to E_h initial condition; k_{mne}, rate of collateral damage to epithelial cells by macrophages and neutrophils; M1 peak ratio, ratio of M1 maximum to initial condition; x_{mne}, regulates effectiveness of macrophages and neutrophils to damage epithelial cells (Hill-type constant).

Columns 1 and 3 of Table 2.3 show the predictors in which all three groups were different from one another for both classification times, as determined by the Kruskal-Wallis and Wilcoxon tests. Results in other columns are described in the following sections. Box plots of a subset of predictors in which all three groups were different are shown in Fig 2.6 to help visualize these differences.

Parameter Sensitivity with eFAST

We calculated eFAST indexes for E_h at 30 minutes, 2 hours (end of ventilation), 6 hours, and 200 hours (time at which outcome is determined). We included a few early time points since we are looking for parameters that could suggest early interventions to mitigate possible negative outcomes. We calculated first-order and total-order sensitivities S_i and S_{Ti}, respectively. Fig 2.7 shows results for the parameters with p-value < 0.05. Parameters x_{mne} (Hill-type constant for effectiveness of macrophages and neutrophils in damaging epithelial

Classification after 2 hours		Classification after 200 hours		eFAST (Ordered)			
Sig. Testing	Random Forest	Sig. Testing	Random Forest				
(Not ordered)	(Ordered output)	(Not ordered)	(Ordered output)	0.5h	2h	6h	200h
E_h ratio 2h	E_h ratio 2h	k_{mne}	k_{mne}	k_{en}	k_{en}	x_{mne}	x_{mne}
E_h ratio 0.5h	E_h ratio 0.5h		E_h ratio 6h	k_{pe}	b_r		
E_h ratio 6h	E_h ratio 6h	x_{mne}	x_{mne}	μ_p	μ_p		
k_{mne}	k_{mne}	E_h ratio 2h	E_h ratio 2h	x_{mne}	k_{pe}		
b_r	b_r	Min M1	Min M1	k_n	k_n		
k_{ep}	k_{ep}	E_h ratio 0.5h	E_h ratio 0.5h	k_{ep}	x_{mne}		
	x_{mne}	Min M2	Min M2	b_r	k_{ep}		
Min M1	Min M1	M2% at 10h	M2% at 10h	k_{an}			
k_{en}	k_{en}	b_r	b_r				
s_n	s_n		k_{en}				
Max M1		M1 peak time					
Min M1%		k_{ep}					
k_{an}		M1 peak ratio					
Max M1%		μ_p					
k_{em1}		k_{em1}					
M1 peak time							
k_{am1}							
μ_{na}							
Max M2%							
k_n							
μ_p							
Min M2%							
a_∞							
s_m							
M2% at 10h							
μ_{ab}							
k_{pm1}							
b_p							
k_{nn}							
x_{m0ab}							
μ_{m1b}							

Table 2.3: Summary of three different methods used to determine the most influential predictors, including parameters and other factors. Columns 1-4 show results for all 24,432 parameter sets. Columns 1-2 show results for analysis methods with classification into three categories (healthy, moderate damage, severe damage) after 2 hours, and columns 3-4 show results for classification after 200 hours. Columns 1 & 3: significance testing results for predictors in which all three groups are statistically different (p-value < 0.05). For ease of comparison between columns, the predictor is listed next to its counterpart in the ordered random forest list, if listed in that column. Column 2 & 4: average importance values determined by random forest decision trees. The top ten are ordered from highest to lowest importance. Columns 5-8: first-order eFAST results (ordered by p-value, with p-value < 0.05) for four time points.

cells), b_r (baseline repair of damaged cells), and k_{en} (phagocytosis of damaged cells by N) were sensitive for several time points. There were no parameters with total-order sensitivity p-value < 0.05 for 6 hours. Parameters with a significant S_i may be better candidates for treatment than those with a significant S_{Ti} because first-order sensitivity measures sensitivity of E_h based only on fluctuations in a single parameter. For this reason and since many of the same parameters are significant for first-order and total-order sensitivity, we show results for first-order sensitivity in Columns 5-8 of Table 2.3, ordered from lowest p-value to highest and for the four time points specified.

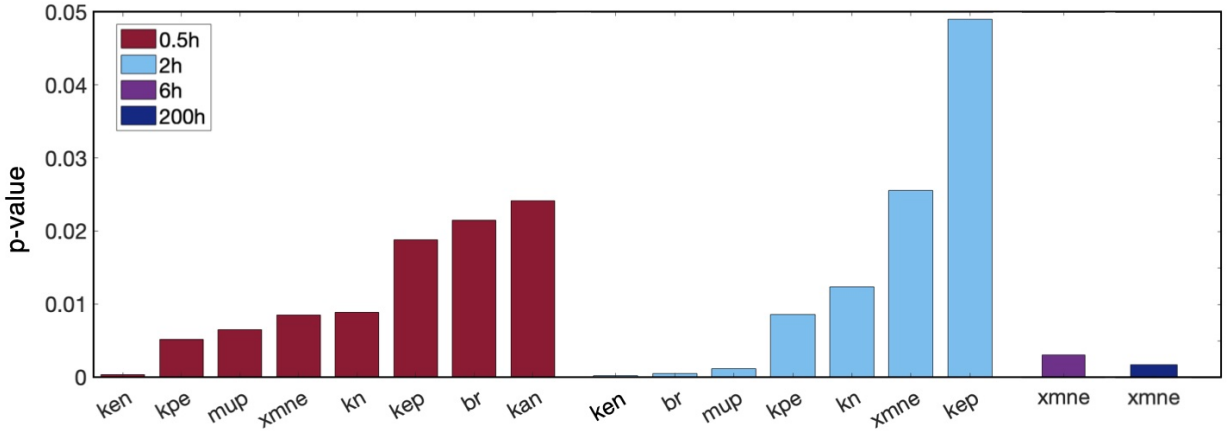
Random forest algorithm to determine predictors

To offset any unusual results generated by the randomness of the decision tree algorithm, we replicated the process of randomly selecting a training set and generating importance values from the random forest 1000 times. Fig 2.8 shows the average and standard deviations of the top ten importance values generated for both 2-hour and 200-hour classifications.

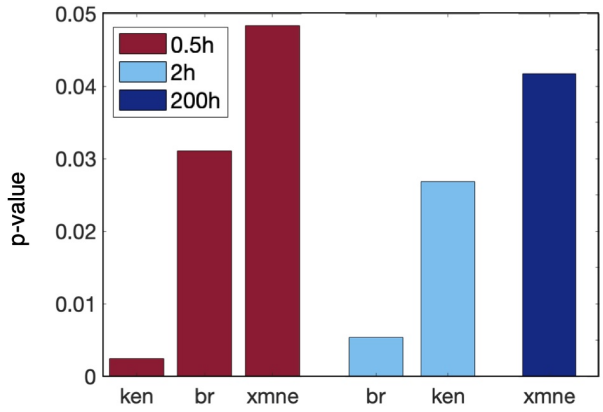
Many of the same predictors are seen in both 2 and 200-hour outcomes, though in a different order. Notice that the standard deviations in both figures are small and support that the predictors remain the same across multiple random forest simulations. Furthermore, several of the top ten predictors were found to be significant by the Kruskal-Wallis test, and b_r, x_{mne}, and k_{en} are shared by random forest and eFAST. (see Table 2.3). The consistency of the importance of these parameters and predictors using different methods supports the idea that they play a significant role in the sensitivity of model output and determining or differentiating outcomes, both immediately after ventilation and after a period of recovery, though they may be more important at specific times.

2.3.3 Modulating recovery: a case study of select transients

Fig 2.9 shows four examples of transients that started in one category and ended in another after ventilation plus a recovery period. We used the information gained in the parameter



(a)



(b)

Figure 2.7: **Parameter sensitivity analysis shows which parameters most influence model output.** Parameters determined by eFAST to be most sensitive, with p-values calculated by comparing eFAST sensitivity indexes to a dummy variable. Results are given for each of the time points tested: 0.5 (red), 2 (blue), 6 hours (purple), 200 hours (navy). (a) First-order sensitivity, also shown in Table 2.3. (b) Total-order sensitivity. Results at 6 hours are not shown as there were no statistically significant parameters at that time point. Parameters: k_{en}, rate of phagocytosis of damaged cells by N; k_{pe}, production rate of p by E_d; μ_p, decay rate of p; x_{mne}, regulates effectiveness of macrophages and neutrophils to damage epithelial cells (Hill-type constant); k_n, rate of migration of N_b to lung; k_{ep}, rate of self-resolving repair mediated by p; b_r, baseline repair rate of damaged cells; k_{an}, rate at which neutrophils become apoptotic.

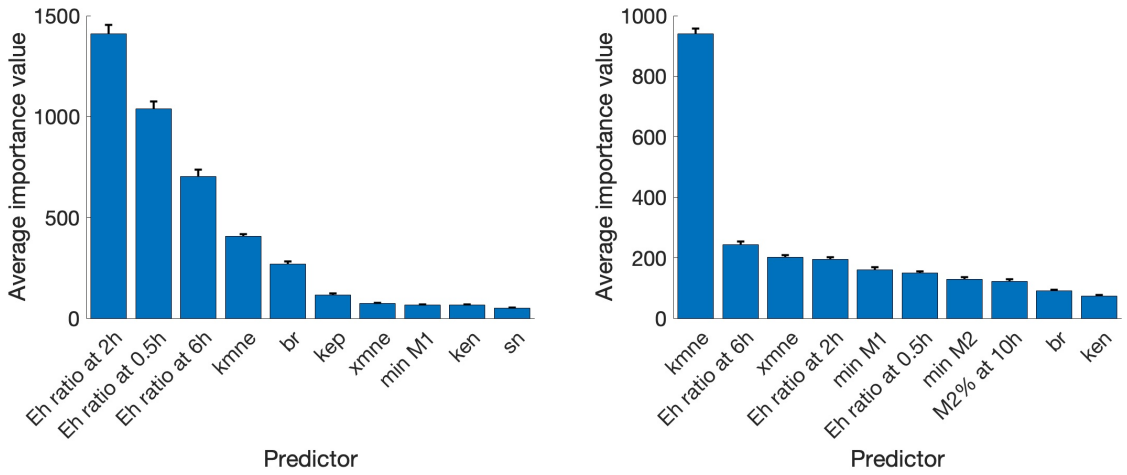
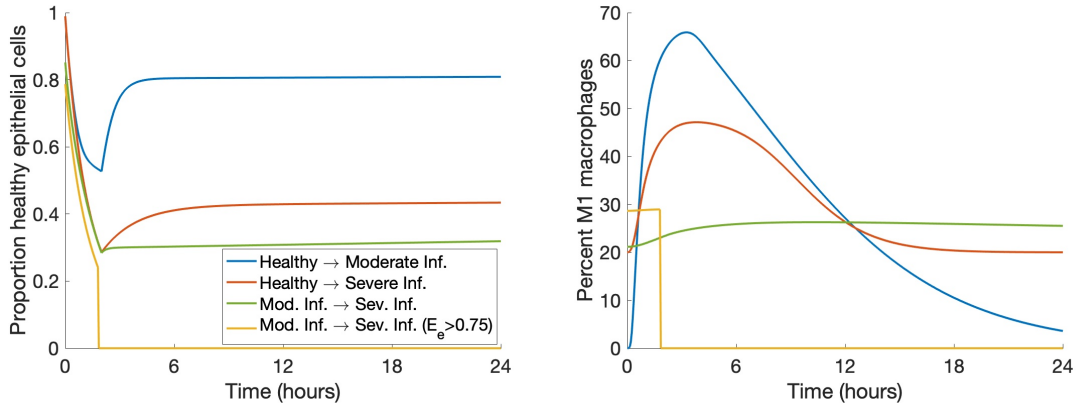


Figure 2.8: **Random forest decision tree selects top indicators of outcome.** Mean and standard deviation of importance values for the top ten highest predictors from 1000 random forest decision trees. Results with classification at (a) 2 hours and (b) 200 hours. Parameters: k_{mne}, rate of collateral damage to epithelial cells by macrophages and neutrophils; b_r, baseline repair rate of damaged cells; k_{ep}, rate of self-resolving repair mediated by $p.$; x_{mne}, regulates effectiveness of macrophages and neutrophils to damage epithelial cells (Hill-type constant); k_{en}, rate of phagocytosis of damaged cells by N; s_n, source rate of N_{0b}. E_h ratio at 0.5, 2, and 6h represents the ratio of E_h at those time points to its initial condition.



(a) (b)

Figure 2.9: **Some parameter sets generate transients that end in a worse outcome.** (a) Transients of E_h that started in one category and ended in a different one. (b) Corresponding transients of $M1$. We included examples of all possible worsening changes in classification as well as a case in which all variables were set to zero due to $E_e > 0.75$ at some time.

analysis to identify key targets for interventions that could modulate damage, especially in the case of a patient starting in one state and ending in a different, negative outcome even after a recovery period. The goal was to return the percentage of healthy epithelial cells to its original steady-state earlier, since the inability to recover from a 2-hour vent after 200 hours or more could be detrimental to long-term health.

Our analysis showed that the parameters k_{mne}, the rate of collateral damage by macrophages and neutrophils to epithelial cells, x_{mne}, the Hill-type constant which regulates the effectiveness of macrophages and neutrophils in damaging epithelial cells, b_r, the rate of self-repair of healthy epithelial cells, and k_{en}, the rate of phagocytosis of damaged cells by neutrophils, were some of the most influential parameters and thus could inform targets for intervention. Furthermore, in the previous section, we obtained results for classification at 2 hours and 200 hours, showing how parameter sensitivity differs between time points. Thus, we examined interventions beginning at several time points (see Fig 2.10).

We intervened in a case that started healthy and ended in moderate damage. Note in Fig 2.10, the original E_h transient began recovery to healthy after the two-hour ventilation

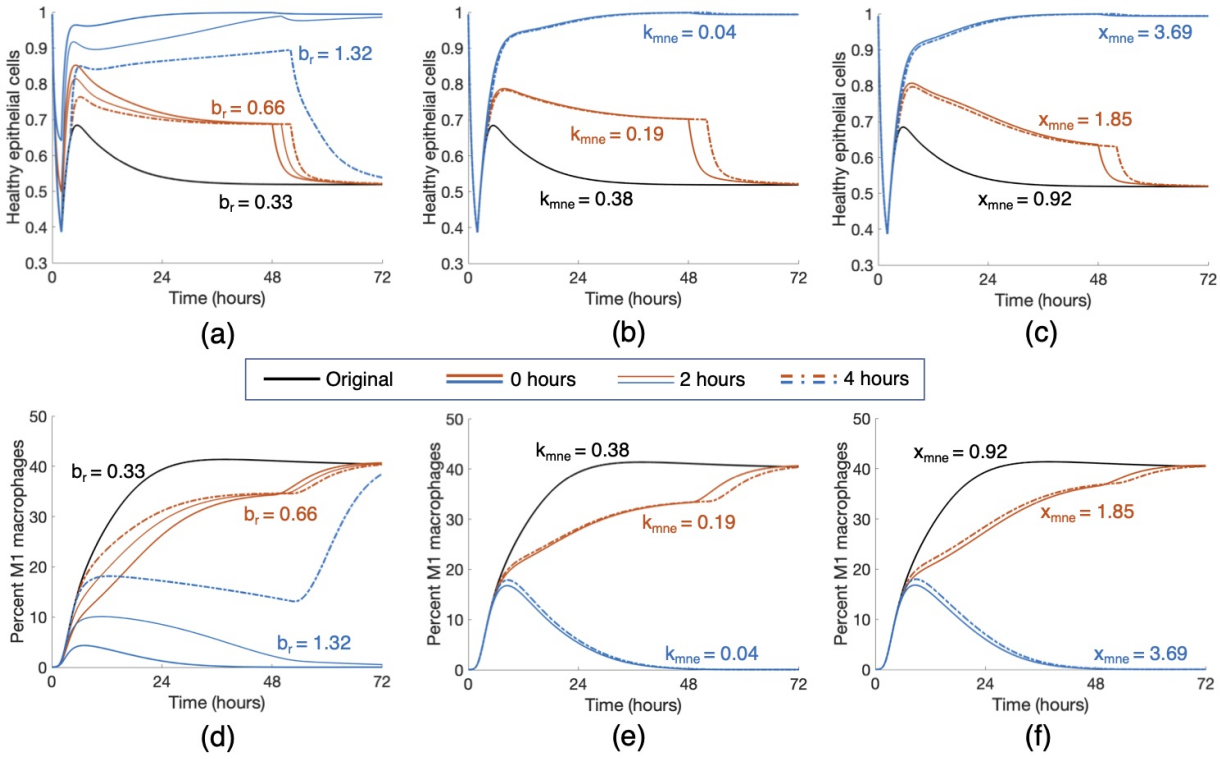


Figure 2.10: **Modulating parameters based on parameter analysis improved outcome in case study.** Starting with a parameter set that gave rise to an E_h transient that started healthy and ended in a moderate damage state, we applied various treatment strategies by changing three key parameters, b_r (rate at which healthy epithelial cells self-repair), k_{mne} (rate of collateral damage to epithelial cells by macrophages and neutrophils), and x_{mne} (Hill-type constant which regulates the effectiveness of macrophages and neutrophils in damaging epithelial cells). Results for various changes are shown for healthy epithelial cells (a, b, c) and percent of M1 macrophages (d, e, f). Treatment began at 0, 2, or 4 hours after the start of ventilation, denoted by solid, dotted, and dot-dashed lines, respectively, and lasted for 48 hours. The original parameter values are $b_r = 0.33$, $k_{mne} = 0.38$, and $x_{mne} = 0.92$. Black transients show the original dynamics without intervention. Orange transients show moderate treatment for each parameter, which was found to be insufficient in mediate the injury. Blue transients show stronger treatments, which were sufficient to bring about resolution for some intervention times.

period, but by the end of the 200-hour period, was at a lower E_h value than it was initially. This was coupled with a transient for M1 in which the pro-inflammatory phenotype increases significantly and then stays in the 40-45% range.

Increasing b_r by various amounts had increasingly positive effects on long-term epithelial health. Lower values of b_r increased E_h slightly and an earlier intervention generated a higher peak of E_h around five hours, but did not continue increasing at this rate regardless of intervention time. If b_r was increased substantially for a significant duration of treatment time, healthy epithelial cells reached the healthy steady-state after ventilation and did not decrease again. Shown in Figures 2.10a and 2.10d, doubling b_r to 0.66 was not enough to generate recovery, but increasing b_r by a factor of four to 1.32 did result in a healthy outcome. For an insufficient treatment duration and value of b_r, levels of E_h were higher until treatment ended and then decreased back to the same level as the original simulation. For a long enough treatment duration, the proportion of healthy epithelial cells remained high even after treatment ended. For $b_r = 0.66$, the intervention time did not improve health in the long run, whereas for $b_r = 1.32$, intervention at either 0 or 2 hours was sufficient to bring about recovery while intervention at 4 hours was not.

The parameter k_{mne} has an inverse relationship with epithelial health; thus, decreasing the parameter provided better results. Decreasing k_{mne} slightly increased the rate of recovery but not enough to change the outcome to resolved. However, with a significant enough decrease of k_{mne}, M1 activation peaked around hour 10 and decreased back to its original level. The original simulation shows M1 activation leveling off at a high percentage of activation (Fig 2.10e). The modulated return to baseline levels was paired with a healthy outcome for epithelial cells (Fig 2.10b). For higher values of k_{mne}, results were about the same for any intervention time 4 hours or less after the beginning of ventilation. Note in Fig 2.10 that the time at which intervention begins mattered somewhat for changes in b_r but not for k_{mne}. Figures 2.10b and 2.10e show that half of the original value of k_{mne} (0.38 to 0.19) was not low enough to change the outcome; multiplying by a factor of 0.1 to $k_{mne} = 0.04$, on the

other hand, was sufficient to change the outcome to healthy.

We also increased the parameter x_{mne}. Increasing this value caused the presence of macrophages and neutrophils to be less effective in damaging epithelial cells. Similarly to the other treatments, sufficient changes to x_{mne} brought about long-term recovery and the time at which intervention began was not as important. Figures 2.10c and 2.10f show doubling x_{mne} to 1.85 was insufficient to change the outcome, and increasing x_{mne} by a factor of four to 3.69 was sufficient.

Finally, we increased k_{en}. This increased the rate at which neutrophils phagocytize damaged cells, making room for new, healthy cells. Interestingly, although k_{en} was shown to be an important parameter in our analysis, even increasing the parameter by a factor of ten to 1.52 was insufficient to make any real changes in the epithelial and macrophage populations. Since there was no significant change, we do not show this treatment in Fig 2.10.

We also examined the results of combination therapy that could include regulation of two or three parameters. Together, changes in parameter values that would be insufficient on their own were able to regulate macrophage activation and bring epithelial cells back to a healthy state. Additionally, higher values of b_r and x_{mne} and lower values of k_{mne} precipitated a quicker recovery from damage. Intervention time was important for parameter values near the threshold, but not for parameter values sufficiently above or below the threshold. Intervention time may make a difference in the ending values of E_h or $M1$, depending on the parameters. Many combinations could be formulated; Fig 2.11 shows two cases in which two parameter changes were insufficient to bring about recovery individually but were sufficient when combined. The orange curves show $b_r = 0.99$ and $k_{mne} = 0.19$ and the blue curves show $x_{mne} = 2.31$ and $k_{en} = 1.52$, which brought about long-term recovery for all three intervention times.

For other cases starting in a healthy state and ending in moderate or severe damage, a high enough b_r can bring about resolution in some cases. In general, earlier intervention

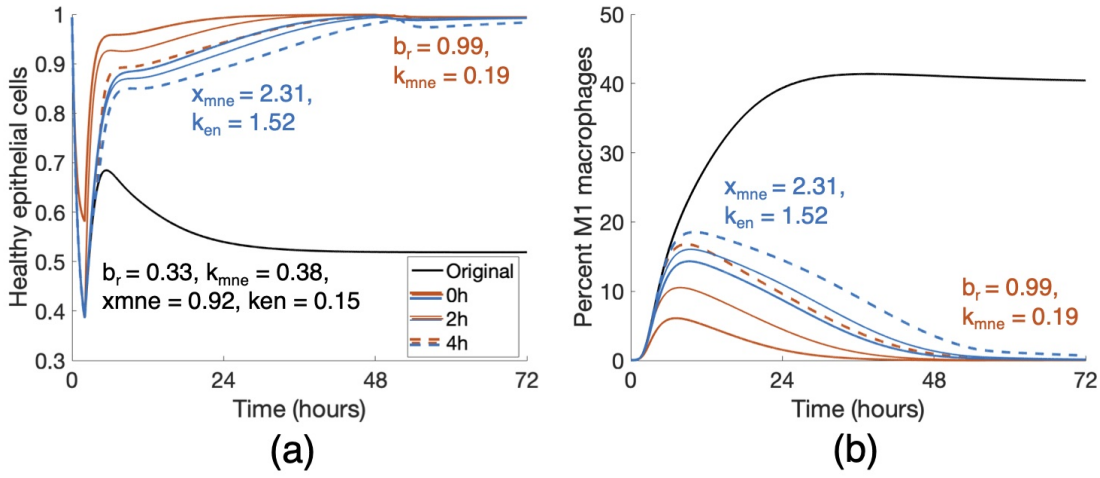


Figure 2.11: **Treatment by combining parameter changes can result in a positive outcome.** Changes in b_r, k_{mne}, x_{mne} and k_{en} that were insufficient on their own (Fig 2.10) resulted in a change in outcome when combined. Orange curves show a combination treatment of $b_r = 0.99$ and $k_{mne} = 0.19$ and blue curves show that of $x_{mne} = 2.31$ and $k_{en} = 1.52$. Duration of treatment in each case was 48 hours, and all intervention times (0, 2, and 4 hours) were successful in a long-term recovery.

times resulted in a faster rate of recovery, but there were varied responses to changes in k_{mne}, x_{mne}, and k_{en}. Even for transients with similar E_h and $M1$ dynamics, reactions to interventions may be different, reinforcing the uniqueness of each parameter set, mirroring the variety of patient responses to MV.

2.4 Discussion

MV is a widely-used short-term life support technique. However, despite its life-saving uses, it often comes with serious complications. Decades of work have contributed to our understanding of the physiology and management of MV, though additional research is needed to best care for patients during and after the period of ventilation, including interventions that target inflammation triggered by ventilator-induced lung injury [99]. Within the immune response, the spectrum of macrophage activation has been a recently growing field of research [11, 128] and with recent findings regarding differences in macrophage polarization linked to age [12, 16], a better understanding of and treatment for VILI is of great concern.

Additionally, mortality rates for MV patients increase with age [26, 117]. Mathematical models have studied a host of causes of lung inflammation, including bacterial and viral infections and allergic reactions [86]. Our model includes macrophage polarization with a more detailed epithelial subsystem to model ventilator-induced lung injury. These features provide a better understanding of how the components of the immune response, including those associated with the different macrophage phenotypes and baseline lung health (steady state values), play a role in post-ventilation outcomes both immediately after ventilation as well as after a period of recovery.

Our approach of developing a collection of parameter sets and identifying the important parameters is a first step in uncovering the driving mechanisms behind VILI and how they contribute to outcomes. Analysis of the model showed that properties and parameters related to epithelial repair and M1 activation and de-activation were especially predictive of outcome. We used b_r, the rate of self-repair of epithelial cells, k_{mne}, the rate at which macrophages and neutrophils cause collateral damage to epithelial cells, x_{mne}, the Hill-type coefficient that regulates the effectiveness of that collateral damage, and k_{en}, the rate of phagocytosis of damaged epithelial cells by neutrophils, to simulate treatments for a parameter set in the collection that started healthy and ended in a moderate damage outcome. We found that modulating b_r is effective in most cases, and the other four can be helpful in some. The chosen case responded differently to treatments and these were paired with varied M1 activation dynamics, indicating that macrophage activation is tied to epithelial health in VILI.

The epithelial subsystem in this model is a simplified version of epithelial cell dynamics that reduces complexity by not accounting for each individual cell and all possible damage levels. We used three categories to model epithelial cell states where a damaged epithelial cell corresponds to increased production of pro-inflammatory mediators. Using this model with data will require alignment of these variables with experimental measurements of lung health. Future iterations of this model would ideally be calibrated with M1/M2 activation and lung

epithelial data in the context of VILI derived from clinical samples. However this would likely need measurements of macrophage phenotypes and epithelial health at multiple time points from various age groups. Until these types of clinical and experimental measures are available, biologically relevant dynamics could be determined using inflammatory biomarkers and macrophage recruitment from cell and tissue experimental models of VILI [109, 129, 140]. For example, Valentine et al. [131] recorded inflammatory gene expression and monocyte recruitment in response to *in vitro* mechanical stretch.

Another area of study is determining if and when the model is bistable, identifying mechanisms that can transition trajectories from one steady state to another, and establishing when this is biologically relevant with regard to treatment. This would help address why some virtual cases can recover with a short intervention time while others need indefinite treatment. Additionally, this model can be expanded to include other types of injury and/or the comorbidities that lead to needing MV, such as a bacterial or viral infection or ARDS, or coupled with other previously published models, to study the interactions between the different types of injury and how they contribute to patient outcome.

In conclusion, our model contributes to the current understanding of the immune response in the lungs, and is an important first step for VILI. Our parameter analysis using a variety of methods provides new insight into potential interventions during and after ventilation to mediate VILI. Experimental data will greatly improve our ability to suggest treatments. Furthermore, the model can be extended to address specific diseases.

2.5 Acknowledgments

This work was supported by the National Science Foundation via award CMMI-1351162 and by the National Institutes of Health via award R21HL146250 (R.H.).

Chapter 3

VILI Model: Methods for Outcome Prediction

3.1 Introduction

In recent years, virtual cohorts have become a way to guide patient interventions in a more personalized manner and reduce the expense and risk of experimental and clinical studies [21]. In this way, a single model can accommodate a variety of patient dynamics and responses by changing model parameters. However, the methods used to obtain patient-specific information are highly specific to the type of injury, data available, and the model itself.

In Chapter 2, we utilized our ODE model and a collection of parameter sets to represent the variety of dynamics observed in patients. We then used this information to determine the parameters to which model output was most sensitive. In this chapter, we extend the use of our model and parameter collection to predict severe responses to ventilation and determine the next best time to obtain new data.

Previously, the Maximally Informative Next Experiment (MINE) algorithm was introduced by Dong et al. [33] to guide experimental design such that model uncertainty is reduced. The MINE approach considers the variance between predicted model outputs; the

larger the variance at a future time point, the more knowledge can be gained from experiments at that time point. The MINE algorithm was developed specifically for systems of ODEs and has been extended and applied to various models [29, 32, 82].

Currently, a sufficient amount of data is not available for our model to utilize a formal MINE approach. However, with the same principles, we used an alternative to parameter variance, since our parameter space is large and exact parameter values are currently unknown. Thus, using principles similar to the MINE approach and synthetic data generated from the collection of parameter sets in Chapter 2, we developed a process through which we can predict changes in epithelial health after ventilation, determine potential next sampling times, and provided recommendations for future efforts to obtain experimental data. When more data is available, this same process can be used iteratively with experiments.

3.2 Methods

3.2.1 Generation of synthetic data & outcome classification

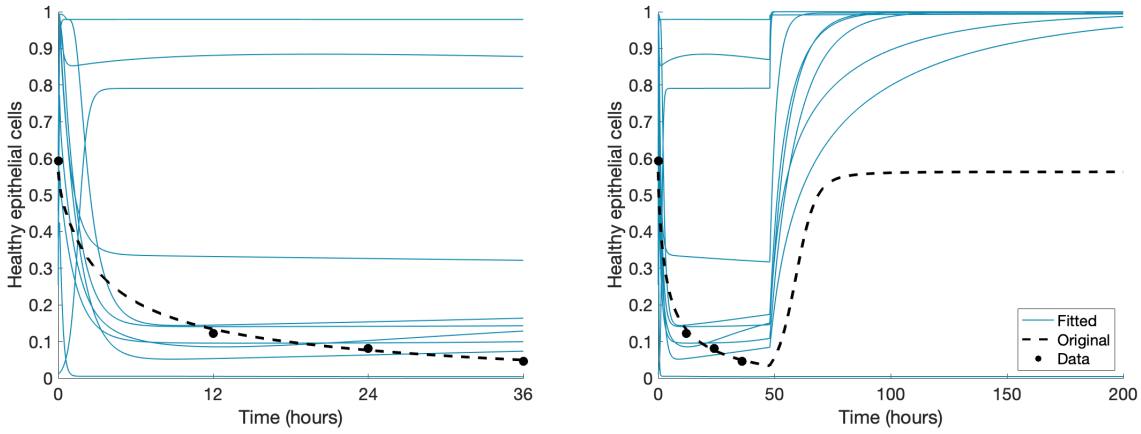
We first obtained synthetic data from the ODE model-generated dynamics corresponding to each parameter set in the collection. We mimicked a clinical setup such that ventilation occurred for 48 hours, and samples were collected every 12 hours during that period. Simulating a 48-hour ventilation resulted in numerical errors for some parameter sets; these cases were removed for a total of 24,170 sets with $E_e(0) < 0.5$, consistent with our methods in Chapter 2. To obtain synthetic data, values were "collected" from transients generated from each prameter set in the collection at these time points and perturbed with 10% uniform noise. In a clinical setting, data can potentially be recorded for macrophage and pro- and anti-inflammatory mediators as well as some measurement of epithelial health. Therefore, the synthetic data we generated included all of these variables, for both the site of inflammation and bloodstream compartments. We used various combinations of available synthetic data throughout the rest of this chapter when testing the predictive ability of our methods.

In Chapter 2, the outcome was based on the ending value of E_h, the proportion of health epithelial cells. In this chapter, we still utilized this number but in relation to its initial condition; the set was categorized as having a severe response to ventilation, or having a "worsening" outcome, if the value of E_h at 200 hours (48-hour vent plus a recovery period) was at least 20% less than its initial condition. If the set was less than 20% lower, it was categorized as not changing. We chose to examine not just E_h at the end of the simulation as in Chapter 2, but rather classify whether the condition worsened to evaluate VILI. We wanted to be able to predict whether a patient will deviate from their initial state after ventilation and a period of recovery. Also in Chapter 2, if $E_e > 0.75$ at any time during the simulation, all variables were set to zero to preserve a reasonable threshold of survival during ventilation. In this chapter, we felt that it did not make sense to set the variables to zero and then generate the data; this would skew calculations used in our algorithm and affect the machine learning methods because of the sharp change in the variables at that time. To maintain a reasonable threshold of survival, we set $E_h = 0$ at the end of the simulation for any transient that had $E_e > 0.75$ at any time. Then the outcome is determined based on this ending value in comparison to its initial condition.

3.2.2 Initial parameter estimation

One of the most common methods for estimating parameters is through curve-fitting algorithms such as nonlinear least squares. As a proof of concept, we first fit our model to synthetic data (without noise) using a nonlinear least-squares method which takes an initial guess. Since least-squares methods are sensitive to initial guesses, we obtained 30 fits using 30 initial guesses randomly generated between the upper and lower bounds. Bounds for parameters were obtained from the values in Table 2.2 and for initial conditions from zero to the maximums generated by the collection of parameter sets in Chapter 2. We also ensured that initial conditions for E_h, E_d, and E_e summed to one.

We estimated all parameters and initial conditions using synthetic data from variables



(a) (b)

Figure 3.1: Results of parameter estimation using a least-squares method. Points represent synthetic data, and solid lines represent the transients generated from fitting the original parameters and initial conditions to synthetic data. Solid lines represent the 10 parameter fits with the lowest norm of the residuals out of the 30 initial guesses. Transients show E_h scaled by $E_h + E_d + E_e$ for (a) the four time points at which data was input into the parameter fit process and (b) the complete 200-hour simulation.

representing healthy epithelial cells, M0, M1, M2 macrophages, and pro-inflammatory and anti-inflammatory mediators in the alveolar space and the bloodstream at four early time points. Results for E_h are shown in Figure 3.1.

As can be seen in Figure 3.1, the parameter estimation routine did not return satisfactory fits. This may be due to the large ranges for initial conditions and parameters and the large number of variables and time points the algorithm had to fit. Another drawback of using a least-squares parameter estimation is that the parameters which best fit the data may result in a system whose dynamics are not driven by the damage itself. In Aim I, we discussed the value of our model in exploring the development of VILI in isolation, without considering comorbidities such as infection. We established a numerical steady-state criteria such that the dynamics seen were only due to ventilator-induced damage. When performing parameter estimation through an algorithm such as least-squares, the dynamics generated by the estimated parameters cannot be guaranteed to arise only from damage due to ventilation.

Furthermore, due to the large number of parameters and variables compared to data

available, the results of a least squares fit like the one seen in Figure 3.1 do not consistently reflect the dynamics of the true parameter set. To ensure damage-driven changes in the components of our model, we used the collection of 24,170 parameter sets developed in the previous aim. All of these parameter sets satisfy the necessary condition that in the absence of damage, all variables remain at a numerical steady-state.

In Aim I, simulations were performed with a two-hour vent time which reflects murine experiments [121, 141]. In this aim, we changed the vent time to 48 hours, representing a timeline more appropriate to a clinical setting. In the following sections, we propose a method using our collection of parameter sets and corresponding dynamics with the goal of accurately predicting a worsening outcome due to VILI.

3.2.3 Development of RES algorithm

Our algorithm is based on having a large collection of *in silico* immune system data for which the outcome is already known. Then for an incoming data set, a subset of this collection that behaves similarly to the data is selected, and predicts the outcome based on what is already known about this subset.

We obtain this subset through calculating a relative error metric for each variable v, with data points x_0 and their corresponding synthetic data point x generated from the collection, accounting for all time points t that are available. We then sum over all variables for which samples have been collected. We call this a relative error sum (RES), calculated as follows:

$$RES = \sum_{v} \sqrt{\sum_{t} \left(\frac{x - x_0}{x_0} \right)^2}$$

Therefore each member of the collection has an RES metric in comparison to the data set. The n parameter sets with the smallest RES are selected to predict the outcome. The data is predicted to an outcome if a certain percentage of the selected subset reaches that outcome; this decision threshold can be varied, which will be discussed below.

3.2.4 Possible combinations of data

Different clinical situations or experiments may necessitate different types of data collection. We wanted to examine the success of our algorithm in the context of multiple cases of data availability, and comment on which combinations of variables and time points may provide the best predictions. Therefore, we examined seven different cases of data availability in the form of model variables. All cases contain samples taken at 0, 12, 24, and 36 hours unless otherwise specified, and we performed our prediction methods on these combinations of data:

1. E_h, macrophages and inflammatory mediators from the alveolar space and bloodstream.
2. E_h, macrophages and inflammatory mediators from the alveolar space and bloodstream. Does not include samples at $t = 0$ hours.
3. Macrophages and inflammatory mediators from the alveolar space and bloodstream.
4. Macrophages and inflammatory mediators from the bloodstream.
5. E_h and macrophages from the alveolar space and bloodstream.
6. Macrophages from the alveolar space and bloodstream.
7. Macrophages from the bloodstream.

We will refer to these numbers throughout this chapter in reference to the types of data availability.

3.2.5 Varying the decision threshold

The parameter sets selected by the RES formula are then used to predict the outcome of the given data set. At the very least, more than half of the selected sets should have a particular outcome in order to predict that the data also has this outcome. However, in a clinical setting, a threshold of 50% is likely not enough for a confident prediction and

justification for intervention. Therefore, we explored varying the decision threshold such that 60%, 70%, or some other percent of the selected subset was necessary to classify the data into a particular outcome. This also introduced the possibility that the algorithm may generate a subset in which the percentage of sets in one outcome or the other does not reach the desired threshold, resulting in an inconclusive outcome. We investigated additional ways to support outcome prediction in inconclusive cases, discussed in the following sections.

3.2.6 Comparison of RES to other classification methods

A main purpose of the RES process was to classify a data set into one of two categories: E_h changes by a sufficiently large amount (20% or more), or E_h does not change. Other classification methods exist, so we compared the success of our methods to these. First, we implemented a random forest decision tree, a machine learning method used in Chapter 2. Instead of using the random forest to identify important parameters, as we did in the previous chapter, we used it to predict classification into one of the two categories using the available data. We first selected a subset of the collection as a training set, and the rest as a test set. The random forest was created from the training set, and then used to predict the outcome of each member of the test set. Second, we used logistic regression, where input (data) is continuous but output (classification) is binary [96]. We also used a training and test set for logistic regression.

Additionally, the ranges of the variables themselves are large and different for each variable, so they were scaled before training and testing by subtracting the mean of each variable and dividing by the standard deviation.

Considering class imbalances

One of the main challenges associated with classification using machine learning methods is the class imbalance problem. It is important to note that out of the 24,170 members of the collection, only 13.75% decrease by 20% or greater after a 200-hour simulation. When

a machine learning algorithm such as a decision tree method trains a model, it is motivated to correctly classify the group that has a significantly greater sample size so that it has a higher percentage accuracy. This results in the group that has a smaller sample size being disproportionately misclassified.

There are a number of ways to circumnavigate this issue; two of the most common approaches are [68]:

1. Impose a higher cost when a member of the smaller group is misclassified. Many functions have been developed to determine these cost values, or the cost can be factors proportional to the total number of cases in each group such that the the cost factor for the smaller group is larger than the cost factor for the larger group.

2. Construct the model based on training samples of each group so that the samples are the same or similar sizes.

We applied the latter method for its ease of implementation, comparing results when class imbalances were considered and when they were not. When considering class imbalances, we constructed a training set by randomly selecting 1000 samples from the "change" group and 1000 from the "no change" group. The test set was the entire collection. When not taking class imbalances into account, the training set consisted of a random sample of the entire collection, as described above.

3.2.7 Next sample time algorithm

A feature of our process is that not only does it predict a data set's outcome given the time points available, but it is extended to determine the next best time to sample, to gain a more confident prediction or a new prediction in the case of an inconclusive result when there is not sufficient information. This process takes the subset selected by the RES algorithm and a set of possible time points at which a new sample can be taken. To demonstrate the process,

we chose a set of possible new time points to be every six hours after the original samples were taken, through 96 hours. To select the next sample time, the steps are as follows:

1. Obtain the subset of n parameter sets and corresponding transients with the smallest RES.

2. For each member of the subset, get value of E_h at each possible next sample time, as if the sample were taken at that time point.

3. At each possible sample time, calculate the variance of the n E_h values.

4. Obtain the possible sample time at which the variance is the highest - choose this as the next sample time.

The concept behind selecting the next sample time is choosing when variance among potential predictions is highest; thus, when the sample is actually taken, there may be less uncertainty regarding the prediction that the algorithm makes [82].

In a clinical or experimental setting, obtaining a next sample time is useful when the current data does not provide enough information about whether or not to intervene; this case is reflected in the data considered "inconclusive" by the RES algorithm. Therefore, we used the next sample time algorithm to choose a time and simulated collecting data at that point. Then, the RES algorithm was employed again, where a smaller subset of the original set selected by the RES algorithm determined the outcome based on the additional data point.

3.2.8 Using parameters to support classification

The benefit of having a collection of parameter sets and their corresponding transients is that not only can information be gained from the variables' dynamics, but also from the parameters themselves, as we showed in Chapter 2 through various analysis methods including random forest and eFAST.

In this chapter, we used parameter values as a supplement to classification when the process resulted in an inconclusive decision. Future work could involve using parameter values in conjunction with the RES algorithm, but in this work we ranked decisions by the RES algorithm as final if they did not return inconclusive. We used random forest to calculate the importance values for the parameters when outcome was determined by whether E_h worsened, not by the E_h ending value as in Chapter 2. The six parameters with the highest importance values were b_p, k_{mne}, x_{mne}, k_{en}, k_{ep}, and μ_p. We tested various combinations of these parameters as predictors to examine their effectiveness in predicting outcome.

Since these six parameters were shown in Aim I to be good predictors of outcome, our method used the means of the parameter values associated with the subset of selected sets to predict outcome depending on whether it lies above or below certain means. These means were compared to the means of the overall collection. For each of the six parameter listed above, three overall means were calculated: 1) mean over all parameters in the collection, 2) mean over parameters associated with a "change" outcome, and 3) mean over parameters associated with a "no change" outcome. We calculated their arithmetic means as well as their geometric means since parameters were sampled on a logarithmic scale. We compared this classification method with random forest and logistic regression methods using the parameter values as predictors.

In all cases except the geometric mean of μ_p, the means associated with "no change" and "change" outcomes were on opposite sides of the total mean. For example, the arithmetic mean of k_{ep} associated with "no change" was greater than the overall mean, and the mean associated with "change" was less. This provides intuition about how comparing individual parameter values with these means could help classify into one outcome or the other.

Our goal was to increase confidence in the algorithm's prediction, so instead of classifying based only on the overall mean, which could result in greater overlap between the two classification groups, we classified data only if the parameter was above or below the "change"

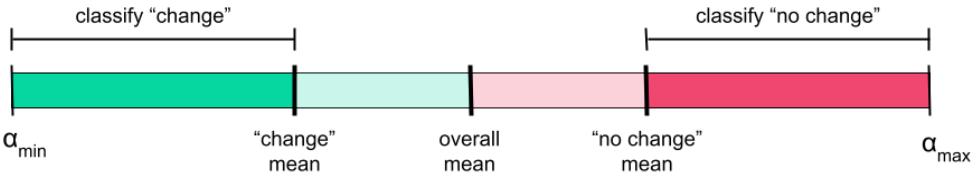


Figure 3.2: Example of how our algorithm predicts outcome based on a parameter value. For some parameter α, if the parameter value associated with a certain data set is less than the mean associated with a "change" outcome, E_h is predicted to change by greater than 20%. If the parameter is greater than the mean associated with a "no change" outcome, E_h is predicted not to change.

or "no change" means. See Figure 3.2 for a visualization.

Not only can individual parameters aid in prediction, but groups of parameters can provide additional information about the outcome. We performed our prediction method on all combinations of the six parameters, such that if a majority of parameter values predicted an outcome, the data set was classified as having that outcome. Comparing the results of each parameter combinations can show which parameters - together and separately - hold the greatest predictive power.

3.3 Results

In this section, we examine the results of our algorithm applied to the collection of parameter sets developed in Chapter 2. Our goal is to use this collection and their resulting transients to predict worsening outcomes using early time points, and recommend informative next sample times to reduce harm to the subject or patient and decrease cost. Figure 3.3 shows a diagram of the process used to make predictions. For every synthetic data set, the RES algorithm made an initial prediction. If that prediction was inconclusive, we performed either the next sample time algorithm or parameter means method to decrease the number of inconclusive results and increase accuracy.

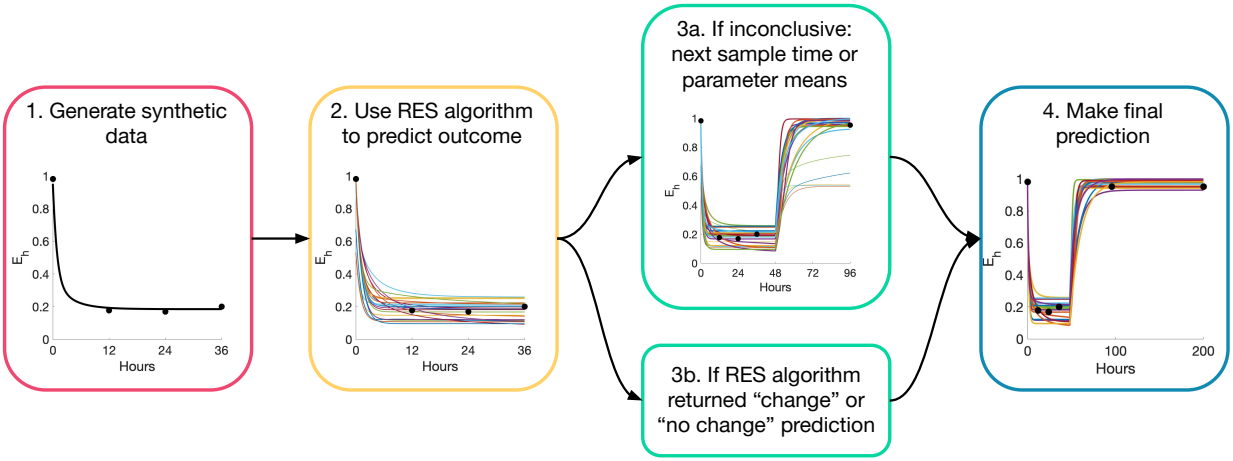


Figure 3.3: Diagram of the process used in this section to obtain predictions of whether a synthetic data set indicated a severe response to ventilation.

3.3.1 RES method compared to other classification methods

We first sought to evaluate the accuracy of our method in comparison to two established classifications methods, random forest decision tree and logistic regression. Figure 3.4 shows how the RES algorithm, using selected subsets of 10, 20, or 30, compared with the established methods. We also show the corresponding accuracy with percent false positives and false negatives in Table 3.1. We set the decision threshold to 50%, where if 50% or more of the RES sets reached a "change" outcome, the data set was predicted to have a "change" outcome. Otherwise, it was classified as having no change. This way, data sets could not be classified as inconclusive. Changing the decision threshold is explored in the next section.

We fit the random forest and logistic regression models first without taking class imbalances into account. We randomly chose 70% of the collection as training data, and tested on the remaining 30%. Second, we took class imbalances into account by using a training set that consisted of 1000 sets with a "change" outcome and 1000 with a "no change" outcome. We used the entire collection as a test set to prevent too large or too small of one outcome to skew results.

Taking into account class imbalances resulted in a higher percentages of the smaller "change" group being classified correctly, but unsurprisingly at the expense of correctly-

Percent accuracy					
Data type	$ns = 10$	$ns = 20$	$ns = 30$	Rand For	Log Reg
1	96.01	96.00	96.02	97.75	97.47
2	95.25	95.15	94.99	97.44	97.03
3	88.61	88.29	88.34	88.86	86.20
4	88.62	88.27	88.23	88.59	86.22
5	96.78	96.63	96.70	97.80	97.29
6	87.25	86.99	86.89	87.04	86.20
7	87.29	86.91	86.82	87.10	86.29
Percent false positive (falsely identified as needing intervention)					
	$ns = 10$	$ns = 20$	$ns = 30$	Rand For	Log Reg
1	2.54	2.39	2.25	1.06	2.01
2	3.16	3.13	3.24	1.24	2.11
3	2.07	1.42	1.13	13.59	10.15
4	1.82	1.13	0.85	13.72	10.12
5	1.35	1.29	1.13	1.09	1.95
6	0.74	0.31	0.24	13.73	12.61
7	0.64	0.22	0.14	13.68	12.44
Percent false negative (not identifying a needed intervention)					
	$ns = 10$	$ns = 20$	$ns = 30$	Rand For	Log Reg
1	1.45	1.61	1.73	1.46	0.24
2	1.59	1.73	1.77	1.73	0.45
3	9.33	10.29	10.53	0.20	1.00
4	9.56	10.59	10.92	0.05	1.28
5	1.87	2.09	2.17	1.62	0.25
6	12.01	12.71	12.88	0.06	0.35
7	12.06	12.87	13.04	0.04	0.46

Table 3.1: Accuracy, false positives, and false negatives for our RES algorithm with 10, 20, or 30 selected subsets, compared to random forest and logistic regression results.

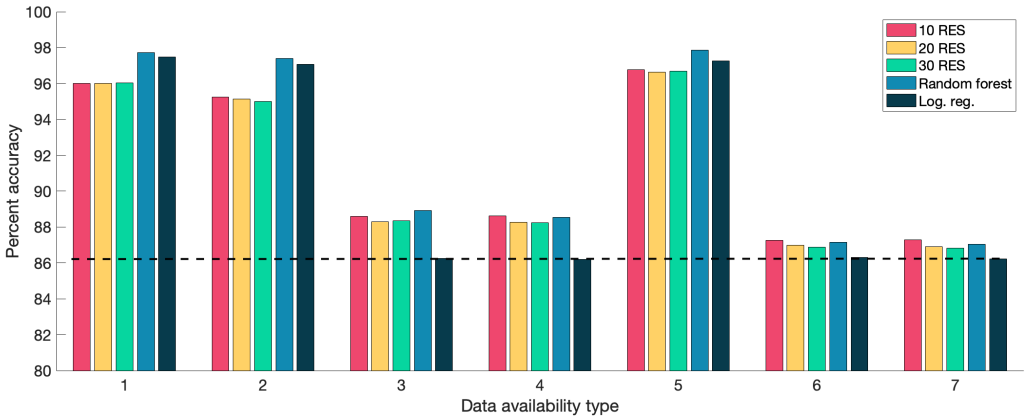


Figure 3.4: Percent accuracy of each approach to predicting outcome. Types of data availability are as follows: 1) E_h, macrophages, and mediators in bloodstream and alveolar space, 2) Same as 1, but without samples at $t = 0$ hours, 3) Macrophages and mediators from alveolar space and bloodstream, 4) Macrophages and mediators from bloodstream, 5) E_h and macrophages from alveolar space and bloodstream, 6) Macrophages from alveolar space and bloodstream, 7) Macrophages from bloodstream. 10, 20, and 30 RES represent selecting subsets of the smallest 10, 20, and 30 relative error sums to predict outcome. Dashed black line at 86.25% represents the percentage of sets that have a "no change" outcome. Values of percent accuracy, false positives, and false negatives are shown in Table 3.1.

predicted "no change" data sets. The accuracy of these models was significantly lower than that of the models that did not take into account class imbalances, so we only show results for models that did not take class imbalances into account.

The vast majority of synthetic data sets resulted in a "no change" outcome. If we were to predict that all cases, had a "no change" outcome, we would be correct 86.25% of the time, represented by the dashed black line in Figure 3.4. This means that nearly 15% of cases would be misclassified; we used our algorithm to not only predict the "no change" sets correctly, but also improve early identification of "change" outcomes. We included this benchmark in the figure because we sought predictive methods that achieved a percent accuracy of greater than 86.25%.

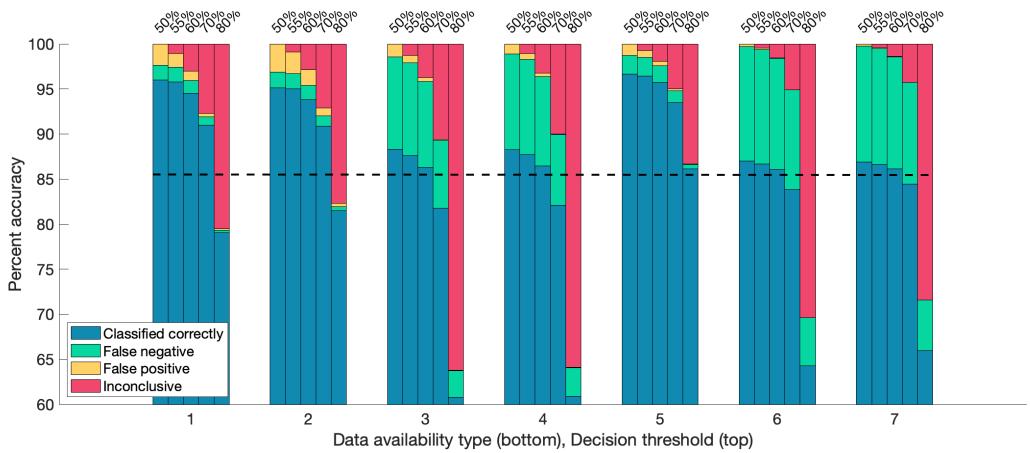
As can be seen in Figure 3.4, the RES algorithm performed well in comparison to the machine learning methods, slightly better in some cases than logistic regression and slightly less effective than random forest. The size of the subsets selected in the RES algorithm (10, 20, or 30) did not make a significant difference in the results. There was, however, a

noticeable difference in results between data availability types 1, 2, 5 and 3, 4, 6, 7. The former group contains some measure of E_h whereas the latter does not, suggesting that having a measure of E_h in the data set improves prediction. This is not surprising since E_h is used in determining the outcome.

More surprisingly, though, was that the accuracy rate of the RES algorithm was higher for data availability type 5 than for 1 and 2, even though type 5 does not include data on pro- and anti-inflammatory mediators whereas 1 and 2 do. Similarly, adding these mediators as predictors in random forest and logistic regression increased accuracy by less than one percent. This decrease in accuracy by RES and only minor improvement by the other methods may be due to the high variability of the p_b, a_b, p, and a transients. Given experimental data, acceptable dynamics for these variables could lead to reduced variability allowing them to aid in prediction.

Another unexpected result was that including initial conditions, i.e. data at hour 0, did not significantly improve accuracy of any of the classification methods. Examining the difference between data availability types 1 and 2, where type 2 is the same variables as 1 but without data at hour 0, accuracy differed by less than half a percent for all methods shown in Figure 3.4; in fact, RES with a selected subset of 10 and logistic regression had slightly better accuracy when not including initial conditions.

These findings support the use of the RES algorithm, as its results were comparable to well-established classification methods and also provides the basis for the algorithm that chooses the next sample time. Furthermore, it provides insight into the types of data that should be collected and when. Since collecting samples can be expensive and invasive, it is useful to choose only what is necessary, and *in silico* modeling provides the ability to test different combinations of data and guide experimental design.



(a)

Figure 3.5: Results of RES method when varying decision threshold from 50% to 80%. The method was performed for all seven data data availability types by selecting a subset of 20 to determine outcome. Y-axis begins at 60% to highlight differences between decision thresholds. Dashed black line at 86.25% represents the percentage of sets that have a "no change" outcome.

3.3.2 Varying the decision threshold

In the previous section, the decision threshold for the RES algorithm was 50%, such that if 50% or more of the subset had a certain outcome, the data was predicted to have that outcome. This means that for a decision threshold of 50%, every data set was guaranteed to be classified, but this does not provide much confidence in the results. Therefore, we explored the effects of increasing the decision threshold such that a greater percentage of the selected subset must have a certain outcome to classify a data set. We began with a simple majority and increased to 80%. Figure 3.5 shows the results for all seven data availability types. Results from selected subsets of 10, 20, and 30 were all similar; thus, we only show results from 20 sets here.

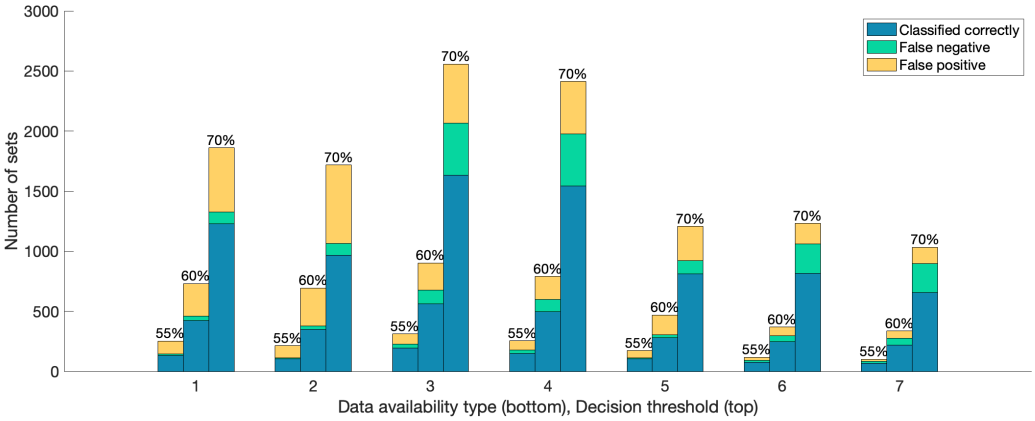
For every increase in the decision threshold, fewer data sets were misclassified; however, the cost was that more sets were considered inconclusive, where neither outcome had a percentage of the selected subset that reached the threshold. This may be preferred in a clinical setting, where a greater degree of confidence is needed when considering additional intervention. The greatest number of misclassifications occurred with the lowest decision threshold,

50%, paired with no inconclusive sets. As the decision threshold increased, the total number classified both correctly and incorrectly decreased and the number of inconclusive sets increased. There was a large increase in inconclusive sets between 70 and 80%. For data availability types 1, 2, and 5, the total percentage of misclassified sets show about half false positive and half false negative. On the other hand, for types 3, 4, 6, and 7, nearly all of the misclassified sets were false negatives. This is most likely due to the much larger number of sets that do not change than the number that do. A benefit of this process is that the threshold can be adjusted based on the needs of the situation; for example, if a greater cost occurs for a misclassified set than an inconclusive one.

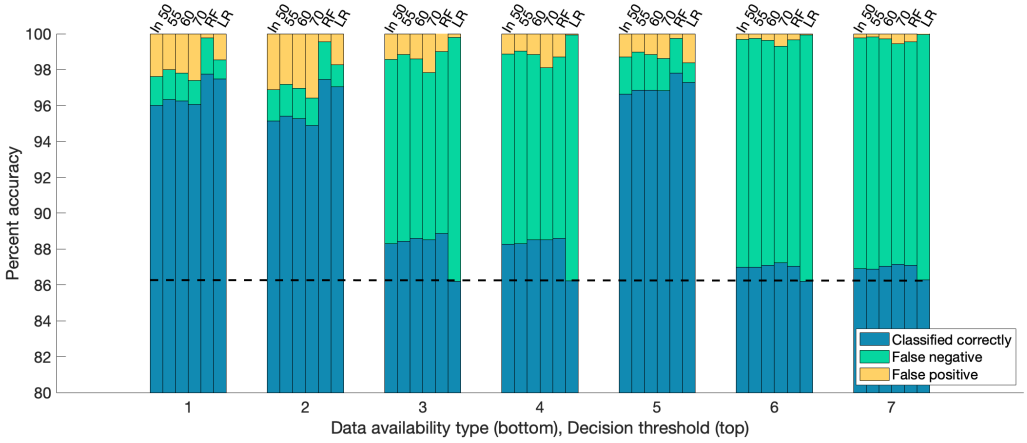
3.3.3 Next sample time & new predictions

When available data does not provide convincing evidence of one outcome or the other, an additional sample later in time may be useful. Using an extension of the RES algorithm, we performed additional predictions on sets that were considered inconclusive by using our "next sample time" process described in the Methods section. Using a selected subset of 10 gave slightly higher success rates when predicting the new outcome, whereas selected subset of 30 had a lower number of inconclusive sets. A selected subset of 20 balanced both of these considerations, so we show results for 20 in Figure 3.6.

The bar graph shows stark differences between the number of inconclusive sets determined after the first prediction by the three decision thresholds shown: 55, 60, and 70%. For all data availability types with a 55% decision threshold, less than 400 out of 24,170 were originally inconclusive. To ensure no inconclusive sets after the next sample time determined by the algorithm, sets were classified as having a "change" outcome if 50% or more of the new subset had a "change" outcome. The number of sets that came out inconclusive after the first round of classificaiton increased as the decision threshold increased, up to nearly 3000 for some of the data availability types. Although the number of inconclusive sets determined by the RES algorithm was much smaller for 55% than 60% and 70%, when examined as



(a)



(b)

Figure 3.6: Results for sets that were first classified as inconclusive, shown as (a) number of sets that were originally classified as inconclusive, and (b) percent of total sets after initial prediction and, if needed, prediction after next sample time. These results were then classified based on a new selected subset of half the original subset. Next sample time with original decision thresholds of 55, 60, and 70% were compared to initial RES prediction with a decision threshold of 50% (In 50) and random forest and logistic regression results. The dashed black line represents the percentage of sets that have a "no change" outcome.

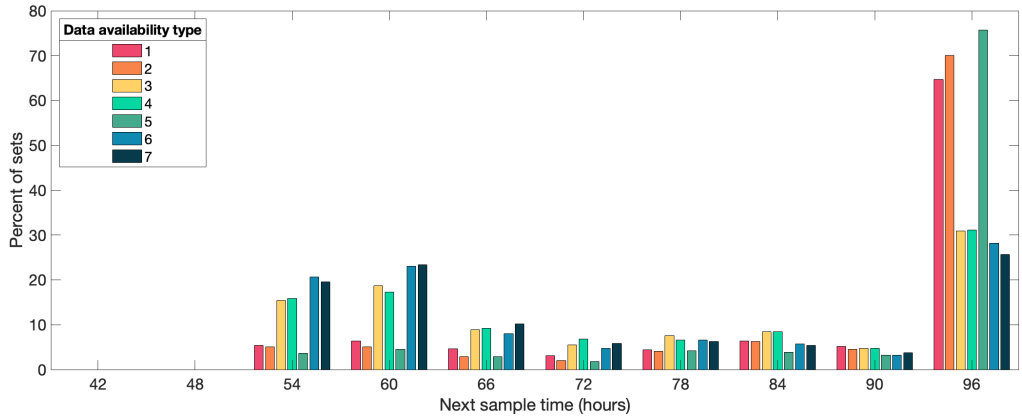
percentages they performed similarly, shown in Figure 3.6(b). After selecting a next sample time, they also performed with similar accuracy to the original 50% decision threshold. We also noticed that both our algorithm and the machine learning methods were more prone to predicting false negatives in data availability types 3, 4, 6, and 7; this is likely due to the disproportionately large number of sets in the group of sets that do not change outcome.

We were also interested in the times the algorithm chose to be the next sample time, and whether there were any patterns. We compared these times between data availability types, decision thresholds, and overall. Figure 3.7 shows these results, first with the frequencies were broken down by data availability type and decision threshold, then totaling the results from different decision thresholds and data availability types. We found that 42 and 48 hours were never chosen as next sample times. 54 and 60 hours were selected sometimes, and the probability of choosing time points after that generally decreased until 96 hours, which was chosen at the highest rate.

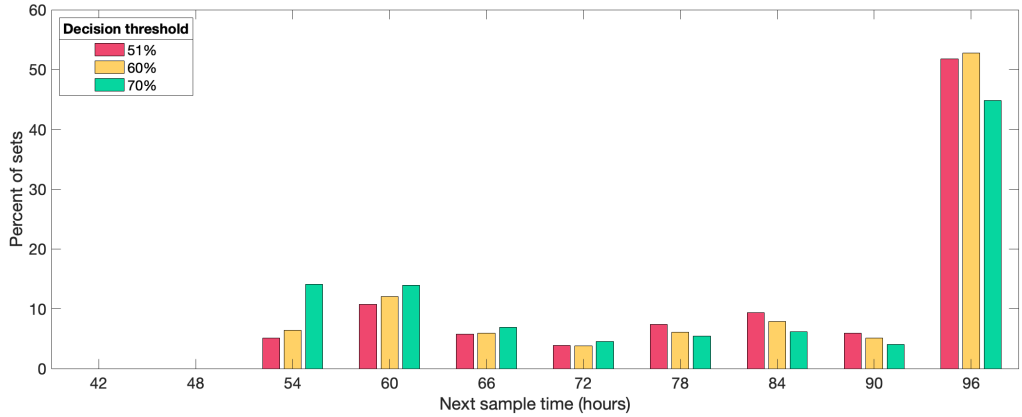
There were some specific differences regarding the type of data available and decision threshold. First, a notable observation was that for a decision threshold of 70% (Figure 3.7b), sample times of 54 and 60 hours were chosen slightly more often than for the other two thresholds. Second, data availability types 1, 2, and 5 (Figure 3.7c) showed 96 hours to be selected around 75% of the time, with 54 and 60 hours selected at about the same rate as 66-90 hours, whereas types 3, 4, 6, and 7 showed 54, 60, and 96 hours to be selected at nearly equal rates. As previously mentioned, types 1, 2, and 5 contain data from E_h whereas 3, 4, 6, and 7 do not. It follows that the closest sample time to the end of the simulation, which determines the outcome group, provided the most information about the classification when E_h was available. Without E_h, there was greater dispersion across time points.

3.3.4 Using parameters to support classification

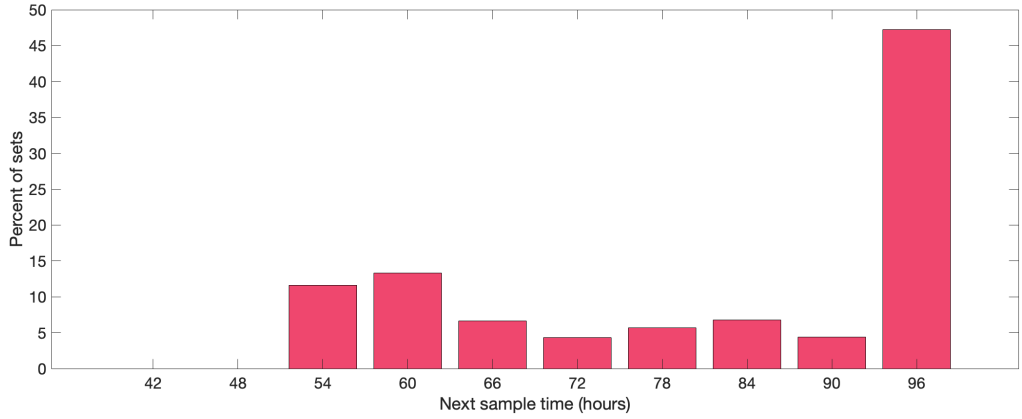
An additional step in classifying inconclusive data sets is taking advantage of our knowledge of the parameter values corresponding to the transients that were selected from the RES



(a)



(b)



(c)

Figure 3.7: Results of the RES algorithm selecting the next sample time based on maximum variance. (a) Results across all decision thresholds, separated by data availability type. (b) Results across all data availability types, separated by decision threshold. (c) Percent of sets across all decision thresholds (60-80%) and data availability types (1-7).

algorithm. Since this method is based on the parameter means, the geometric or arithmetic means can be used to classify sets. We include results for both since parameters were originally sampled on a logarithmic scale.

Similarly to the next sample time algorithm, we used the method to predict the outcomes of parameter sets in the collection that were first predicted as inconclusive, testing all combinations of the six parameters, $k_{ep},\ b_p,\ k_{mne},\ \mu_p,\ k_{en}$ and x_{mne}. Because the process purposefully does not classify all parameters, many of the sets were determined by the algorithm to be inconclusive. However, most of the parameter combinations that were not inconclusive had a very high successful prediction rate. We show two parameter combinations in Figure 3.8: k_{ep} and x_{mne}, and $k_{ep},\ k_{mne},\ \mu_p,\ k_{en}$ and x_{mne}.

We tested our algorithm on all data availability types and a selected subset of 20. The parameter means method assigned a prediction to the data set if 55% or more of the parameters in a specific combination of the six parameters predicted either "change" or "no change." Figure 3.8 shows a comparison of our parameter means method, using geometric or arithmetic means, to results of random forest and logistic regression using synthetic data as well as the original RES prediction (without taking class imbalances into account). Since data types 1, 2, and 5 behaved similarly and 3, 4, 6, and 7 behaved similarly, we show one from each group. We also included the percentage of sets classified as a "no change" outcome, 86.25%, shown by the dashed black line. As we explained above, a success rate higher than this is desired.

Figure 3.8 reveals that the parameter means method results in a high percentage of inconclusive predictions, but that misclassification rates are low. Decision thresholds of 55-60% show marginal differences between original RES predictions (column names 4, 5) and arithmetic and geometric mean predictions (columns Ar, Geo 4 and 5). For higher decision thresholds of 70-80%, increased accurate predictions using arithmetic and geometric means can be seen compared to the original RES prediction. The two parameter combinations have similar results, and data availability type 5 performs better than type 4 in every case,

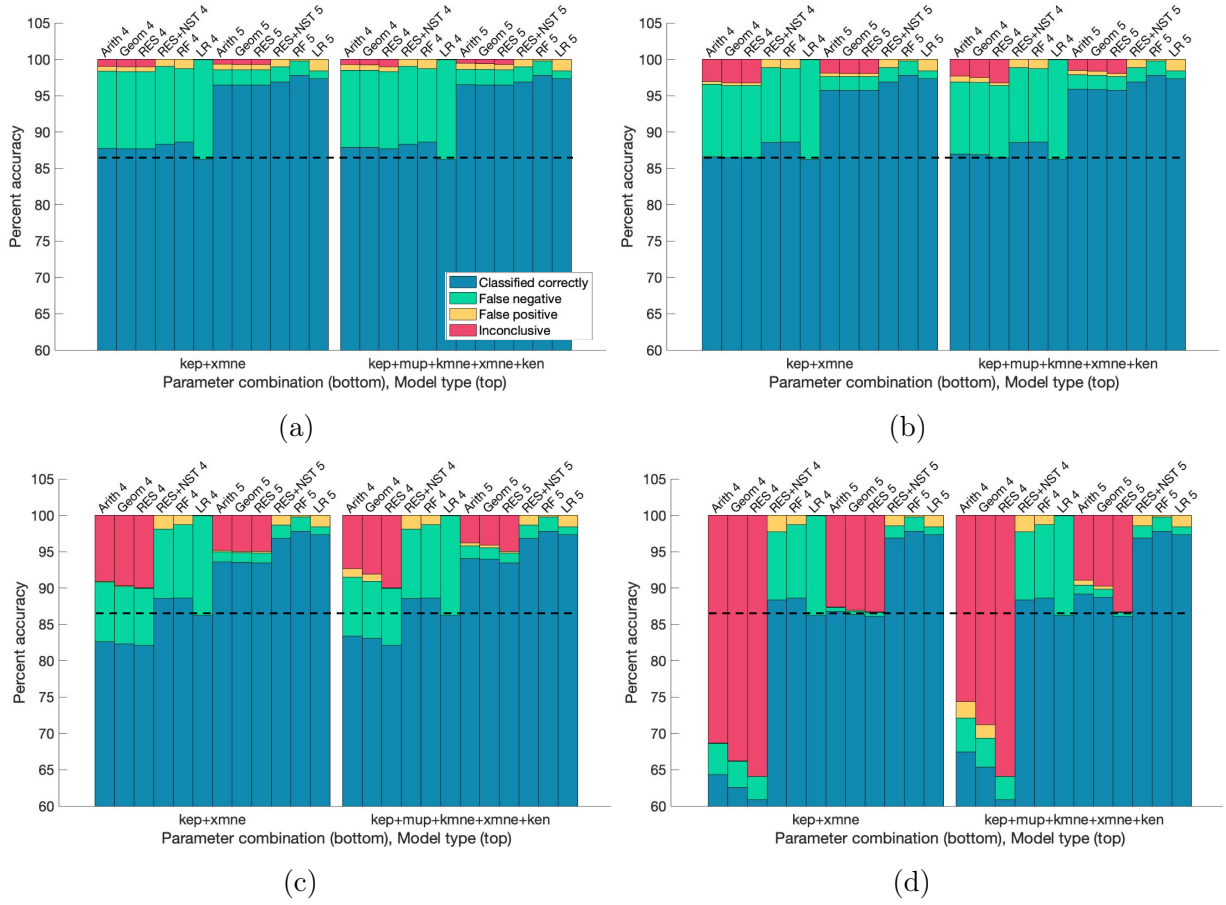


Figure 3.8: Results from predictions using multiple methods described in this chapter for two combinations of parameters (apply only to parameter means method, the other methods' results are consistent between the two combinations) and two data availability types (types 4 and 5). The parameter combinations are k_{ep} and x_{mne} (left) and k_{ep}, k_{mne}, μ_p, k_{en} and x_{mne} (right). From left to right: prediction via RES plus arithmetic (Arith) and geometric (Geom) means method, prediction via RES only (RES), prediction via RES plus next sample time method (RES+NST), random forest and logistic regression with data types 4 and 5 as input (RF, LR). Panels show four different decision thresholds: (a) 55%, (b) 60%, (c) 70%, (d) 80%. The dashed black line represents the percentage of sets that have a "no change" outcome.

and comparable to random forest and logistic regression. However, this accuracy had the cost of significantly fewer classifications due to the high number of inconclusive sets. This parameter means exercise shows that the parameters can be useful in predicting outcome in some cases, though future work is needed to decrease the number of inconclusive results.

3.3.5 Comparison of methods

As seen in Figure 3.8, our results show many variations, including number of selected subsets, decision threshold, and type of data available. We also show how taking an additional step of either selecting a next sample time or utilizing the parameter values of the selected sets can aid in a more confident prediction. In comparison to the established classification methods random forest and logistic regression, our methods performed well. In particular, our RES algorithm plus next sample time predicted with accuracy comparable to random forest and logistic regression for all decision thresholds studied in this work; for some data availability types, our algorithm performed better. RES alone, RES plus next sample time, and RES plus parameter means also performed well when E_h data was included in the data set, though more work should be done to improve prediction for all of our methods when E_h is not included and to decrease the number of inconclusive sets predicted by the parameter means method.

3.4 Case studies

Thus far, we have shown results for the entire collection. In this section, we examine a few parameter sets that represent cases where different predictions methods may be used. This is to demonstrate how our processes could be used for an individual data set.

3.4.1 Case study 1: "no change" outcome

First, we examined a case in which E_h was confidently predicted as not worsening. Using the RES algorithm with an 80% decision threshold, all data types correctly predicted no change from the E_h initial condition; furthermore, any decision threshold lower than 80% will also give the same prediction. We highlight types 1 and 3 in this and the following case studies as examples of data that includes E_h and does not include E_h. Figure 3.9 shows the 20 transients to which the data was closest to, as defined by having the smallest RES values; note that the two variables shown are only a subset of the total data used by the RES algorithm to find the closest matches.

The transients for M1 in Figure 3.9(b) and (d) are very similar, since M1 data was included in both data types. However, E_h is fairly different between the two types and we did not include data in panel (c) since E_h data is not included in type 3. Therefore, the range of possible E_h transients is wider. Even without E_h data, however, the RES algorithm predicted this case correctly. In Figure 3.9, bold lines show sets classified as having a worsening outcome. Using data type 1, none of the sets have this outcome, and with type 3, only 3 out of 20 sets have this outcome. Therefore, with an 80% decision threshold, both correctly predict that this case does not have a severe response to ventilation.

In this case, the pre-ventilation state, measured by initial condition for E_h, was high and the algorithm predicted correctly that after ventilation, E_h will return to this state. Based on the algorithm's prediction that the post-ventilation state will not worsen, we would not recommend additional intervention. Because the algorithm predicted with high confidence that this case will not worsen, an additional sample at a later time was deemed unnecessary.

3.4.2 Case study 2: "change" outcome

Next, we demonstrate a case that worsens after ventilation and period of recovery. A decision threshold of 70% results in data types 1-5 predicting a worsening outcome, and 6-7 inconclusive. A decision threshold of 60% results in all data availability types predicting a

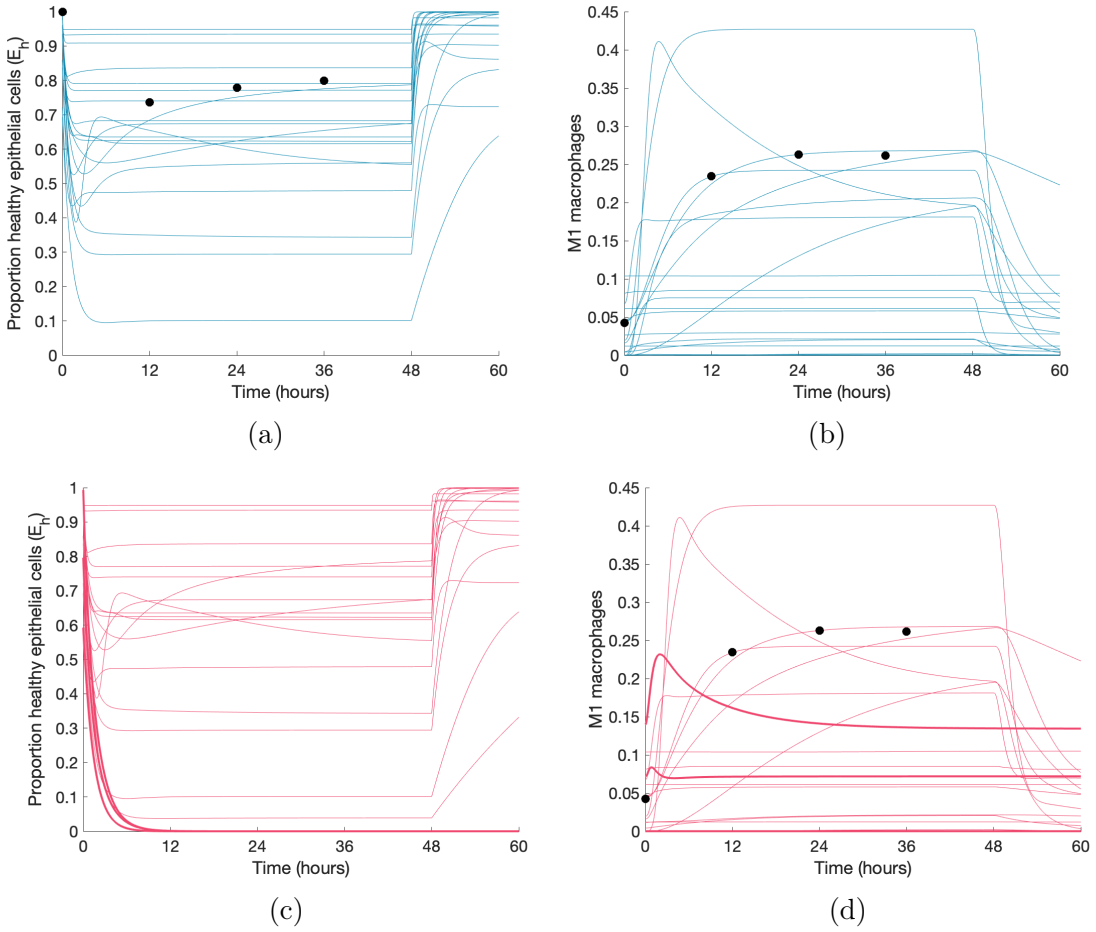


Figure 3.9: Case 1: Data used in RES algorithm (black points) to find the 20 sets that are closest to the data. Blue curves in (a) and (b) are the sets found when data availability type 1 was used, and pink curves in (c) and (d) show results from type 3. Bold lines show sets classified as having a worsening outcome.

worse outcome. Figure 3.10 shows the subset of transients selected to predict the data set's outcome. Out of the 20 selected sets, 20 in type 1 and 16 in type 3 predicted a worsening outcome, resulting in a "change" outcome for up to an 80% decision threshold. Bolded curves in the figure represent the sets that are classified as worsening. Similarly to Case 1, M1 transients were similar between the two data availability types, but E_h transients were very different since E_h data is not available for type 3. In Chapter 2 we discussed how dysregulation of the M1/M2 response can affect the epithelium, and the low amount of M1 in this case could be influencing the poor outcome seen in E_h data in Figure 3.10(a).

We show the dynamics out to 200 hours to highlight the contrast between the variability in the E_h transients selected for type 1, panel (a), and those for type 3, panel (c). Since E_h data was not included for type 3, higher variability can be seen in panel (c) than in (a). Nonetheless, many of the sets in (c) are decreasing from a high percentage of E_h to a lower one, indicating a "change" outcome; in particular, see bolded curves in Figure 3.10, representing sets with a worsening outcome. We hypothesize that this is due to the corresponding variables that do have data and the underlying mechanisms of the dynamical system that lead to a severe response in E_h. Since the outcome was predicted with a high decision threshold, we did not proceed with an additional sample point.

3.4.3 Case 3: inconclusive prediction with next sample time needed

The third case we examined was one in which the RES algorithm initially gave an inconclusive result, but selecting a next sample time provided the correct prediction. In this case, RES with data availability types 1 and 3 gave an inconclusive result with a decision threshold of 70%. Bold transients in Figure 3.11 show the selected sets that end in a worse outcome. Given this result, we wanted to sample again at a later time to obtain more information.

The next sample time algorithm determined 54 hours to be the best sample time for both data availability types 1 and 3. This time was commonly selected based on our results for the entire collection of parameter sets. After sampling at 54 hours and predicting again,

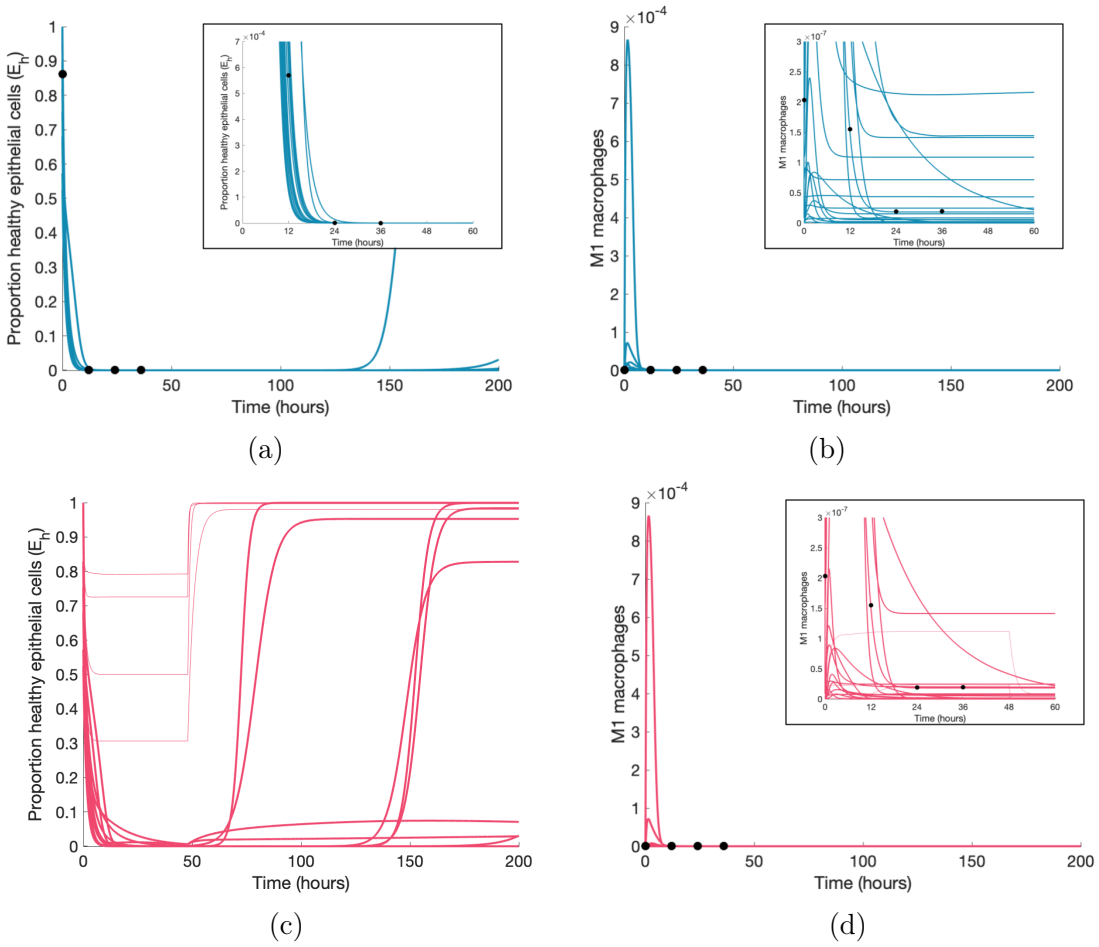


Figure 3.10: Case 2: Data used in RES algorithm (black points) to find the 20 sets that are closest to the data. Blue curves in (a) and (b) are the sets found when data availability type 1 was used, and pink curves in (c) and (d) show results from type 3. Insets are included when necessary to show data with smaller x and y ranges. Bold lines show sets classified as having a worsening outcome.

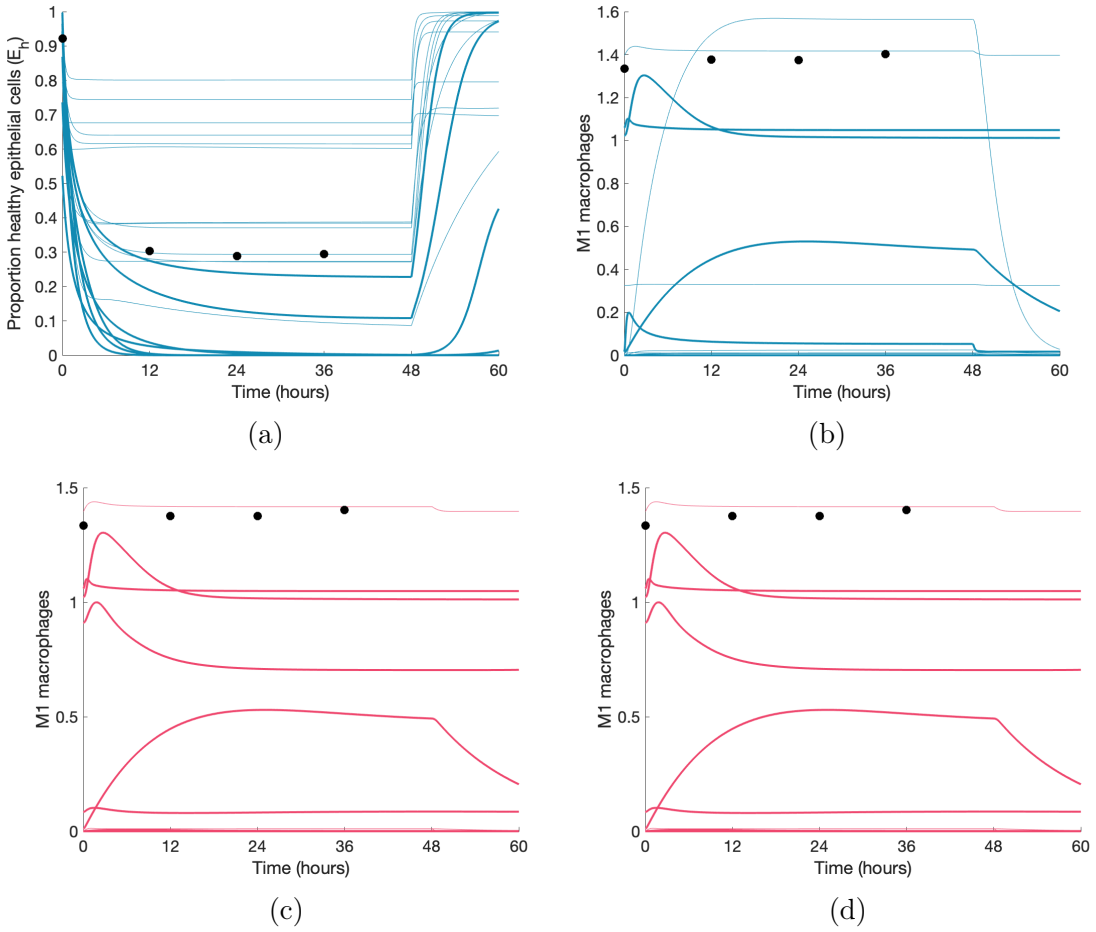


Figure 3.11: Case 3: Data used in RES algorithm (black points) to find the 20 sets that are closest to the data. Blue curves in (a) and (b) are the sets found when data availability type 1 was used, and pink curves in (c) and (d) show results from type 3. Sets that end in a worsening outcome are bold.

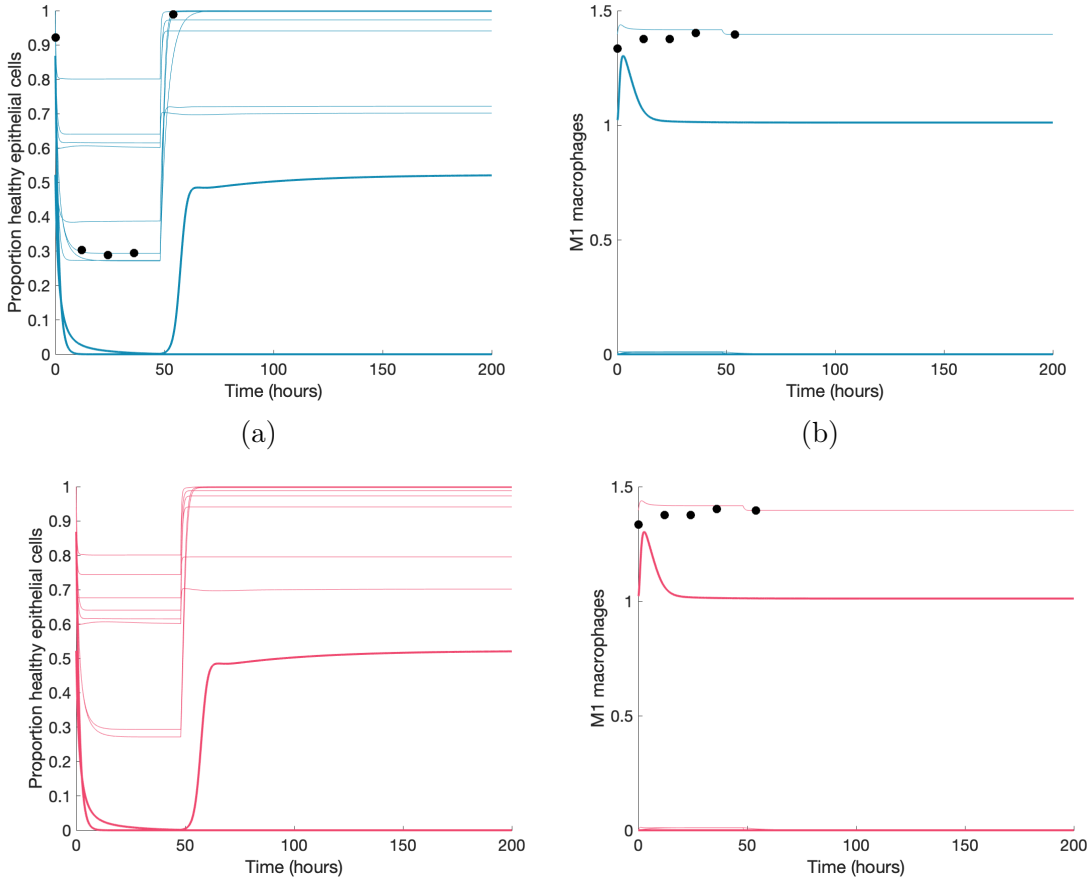


Figure 3.12: Case 3: Data used in RES algorithm (black points) to find the 10 sets that are closest to the data after selecting a next sample time. Blue curves in (a) and (b) are the sets found when data availability type 1 was used, and pink curves in (c) and (d) show results from type 3. Sets that end in a worsening outcome are bold.

both resulted in a correct prediction of "no change." Figure 3.12 shows the subset of original sets selected to predict this case. The additional time point for E_h at 54 hours reveals that there was a sharp increase in E_h after ventilation, representing a recovery back to its initial condition.

3.4.4 Case 4: inconclusive result

As shown in the figures that summarize the entire collection, our algorithm is not always accurate. In this section we show a case in which a next sample time incorrectly predicts the

outcome. Figure 3.13 shows the selected sets after the initial inconclusive prediction and a next sample time, predicting a "change" outcome when the true outcome was "no change."

Figure 3.13(a) shows that at hour 84, the time point selected as the next sample time, E_h increases back up to its original magnitude and many of the selected transients also increase. However, based on our criteria that if $E_e > 0.75$ at any time the outcome is set to worsening, many of the selected transients result in a worsening outcome. Based on the E_h data at 12, 24, and 36 hours, the case does not correspond to $E_e > 0.75$ but some transients in the selected subset may have slightly lower E_h and result in higher E_e values. These high E_e values can be seen in Figure 3.13(e), where many reach above 0.75 at some point during ventilation. This results in a inconsistent prediction.

We then used the parameter means process to obtain a prediction. This method does not use data; only the parameter values corresponding to the selected subsets. We found in the previous section that parameter combinations with more successful predictions, not including inconclusive results, were $k_{ep} + x_{mne}$ and $k_{ep} + k_{mne} + \mu_p + k_{en} + x_{mne}$. The parameter means method can be based on geometric or arithmetic means; we tried both. For the combination of k_{ep} and x_{mne}, the prediction was still inconclusive using arithmetic means but with geometric means, the process correctly predicted a "no change" outcome. The combination of k_{ep}, k_{mne}, μ_p, k_{en} and x_{mne} predicted the opposite: geometric means predicted inconclusive and arithmetic means predicted "no change."

These four case studies show some of the possibilities that can occur when using the methods developed in this chapter. As seen in the results for the entire collection of synthetic data, our algorithms generally perform well, such as Cases 1 and 2, but may also result in inconclusive or incorrect predictions, as in Cases 3 and 4. Further work should be done to determine the most reliable methods when the parameter means method is inconclusive or provides contrasting results based on mean type (geometric or arithmetic) or contrasting to a next sample time prediction. The differences between the data availability types showed that having E_h in the data set was useful but not always necessary in correctly predicting the

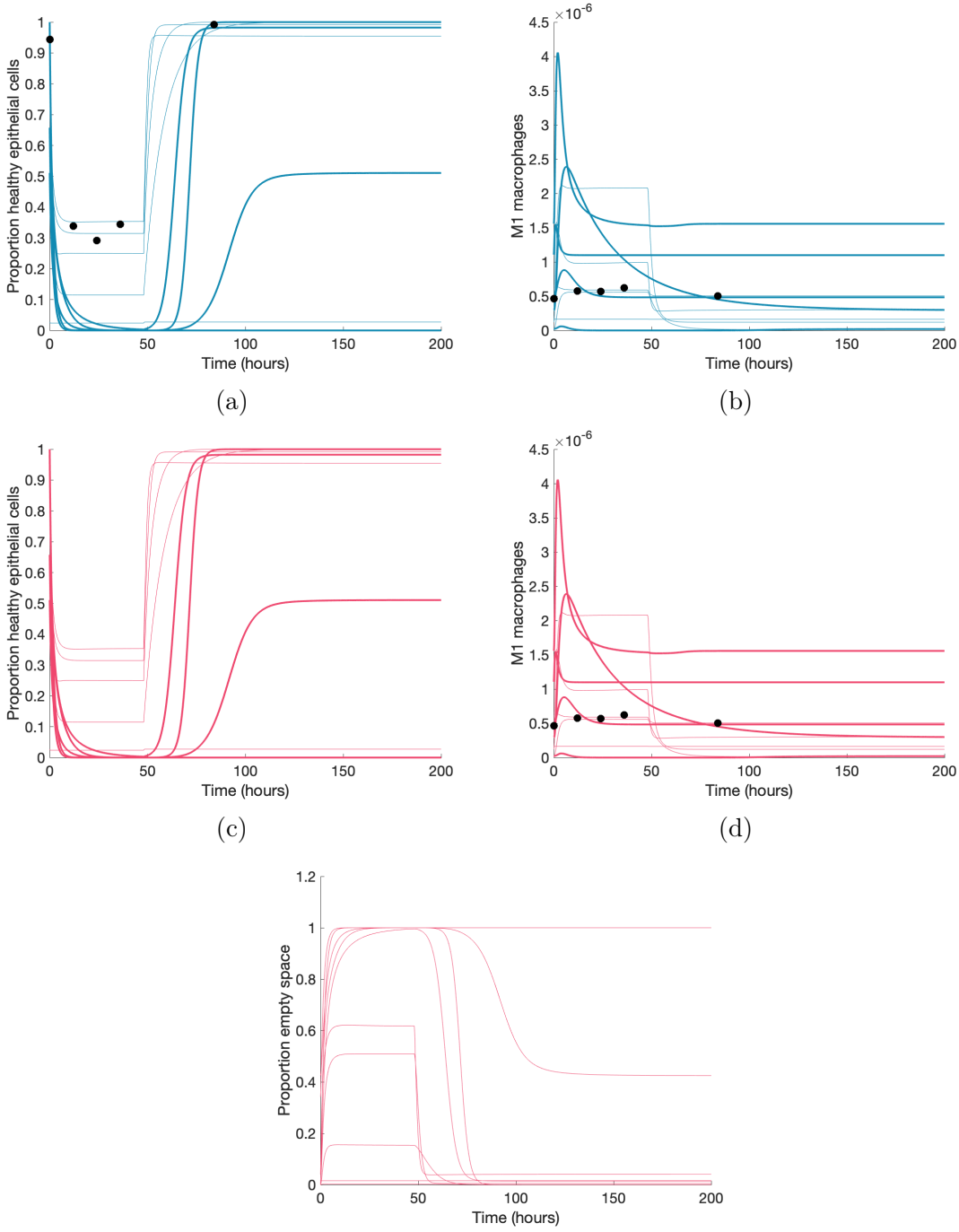


Figure 3.13: Case 4: Data used in RES algorithm (black points) to find the 10 sets that are closest to the data after selecting a next sample time. Blue curves in (a) and (b) are the sets found when data availability type 1 was used, and pink curves in (c) and (d) show results from type 3. Sets that end in a worsening outcome are bold.

outcome. In Figures 3.9-3.13, the range of values that the algorithm selected show that this process does not find exact fits in the collection because of the number of variables involved in the process and the variability in the transients, but that a subset of close yet inexact fits can still be useful for predicting.

3.5 Discussion

This chapter shows the many possibilities of how a collection of data sets, such as the one synthetically generated by the parameter sets from Chapter 2, can be used to aid in predicting outcome and determining the next best time to sample. This *in silico* modeling and prediction could save time and money in experimental and clinical settings. However, real data should be collected to effectively use the algorithms described here. The results from our methods also bring up some questions. We found that data that includes E_h performed similarly (types 1, 2, and 5), and differently from those that do not include E_h, which performed similarly (types 3, 4, 6, and 7). The groups that performed similarly have different forms, such as taken from the alveolar space vs. the bloodstream or macrophages vs. pro- and anti-inflammatory mediators.

Despite these different variables, accuracy was similar. In fact, Figure 3.4 shows that with E_h data, not including pro- and anti-inflammatory mediators increased the accuracy rate for the RES algorithm. We hypothesize that this is due to the large variability in the dynamics of PIM and AIM produced by the collection of parameter sets. These mediators are produced by several types of cells in the ODE model (see Eqs (2.4), (2.5), (2.6), and (2.7) in Chapter 2) and have wide parameter ranges for the associated production rates. Therefore the synthetic data produced by these dynamics may also have high variability. Using this algorithm on experimental data could help to better distinguish the effectiveness of the different cells and mediators sampled in predicting outcome, indicating which types of data and sample times provide the most information. This could aid future experimental

design.

Other statistical methods could be used to aid or modify the current process. For example, the next sample time algorithm utilizes the variance of E_h at various possible sample times; however, variance can be susceptible to outliers. Using more robust measures of dispersion such as the interquartile range may be useful in obtaining the next best sample time and thus better predictions [74].

Furthermore, having parameter sets based on real data would be useful in the parameter means method. Although we established in Chapter 2 what we believed to be reasonable parameter ranges for obtaining a variety of dynamics, these ranges are not based on data. We hypothesize that obtaining ranges informed by data would improve the accuracy of both the parameter means method and the RES algorithm. In the future, more work could be done to understand the difference between cutoffs determined by the geometric means and those determined by arithmetic means.

Since these methods only require a set of ODEs and a collection of parameter sets, this process could be applied to any model. The outcome could be determined by a relative decrease in a particular variable like we did in this chapter, but could also distinguish between growth or decay of a bacterial population, damage categories as in Chapter 2, or any other type of classification.

Chapter 4

Macrophage Phenotype Polarization

4.1 Introduction

As discussed in previous chapters, the plasticity of macrophages allows them to perform many roles in response to an injury or infection, and they have a significant impact on the overall ability of the immune system to resolve the insult [11]. Several models have been published that include macrophage polarization, including ODE models of subcellular signaling and simplified M1/M2 activation. Maiti et al. [73] and Moya et al. [90] focused on the subcellular signaling pathways of NF-κB/TNFα and STAT3/IL-10, respectively. Frank et al. [40] and Zhao et al. [149] developed two-dimensional ODE models with M1 and M2 activation as the state variables. Rex et al. [110] used a Boolean model to select genes related to M1/M2 dynamics and developed an ODE modeling the dynamics of those genes. Additionally, some modeling efforts of macrophage plasticity incorporate spatial dynamics. Agent-based models that include M1/M2 phenotypes have been developed in the context of tuberculosis [58] and Nickaeen et al. [94] developed a PDE model of M1/M2 macrophages in response to high levels of IL4 or LPS/IFNγ.

In this chapter, we propose two models of the immune response to lung inflammation that build upon previous modeling work to examine the spectrum of macrophage activation in

greater detail. Whereas the model introduced in Chapter 2 examines the immune system's response to damaged epithelium in response to VILI, the ODE model proposed in this chapter tracks M1 and M2 activation via subcellular signaling pathways in response to general inflammatory stimuli. This model is an extension of work by Maiti et al. [73]; we added details of the IL-10 pathway not yet included in Maiti et al. by adapting and extending equations from Moya et al. [90], including both pro- and anti-inflammatory feedback loops and their interactions. The model consists of ten macrophages, each of which has a set of equations modeling its subcellular pathways. These ten macrophages are linked by external TNFα and IL-10, which can be both introduced into the system at various times and produced by the macrophages themselves. In our ABM, we incorporated various mediators with a spectrum of M1/M2 polarization and spatial dynamics. In this model, macrophages can become more activated towards an M1 or M2 phenotype based on their local patch environment, and perform a variety of roles depending on their activation levels. Both models account for macrophage cell cycle using randomly generated lifespans for each macrophage.

Based on data from Maiti et al. [73], we calibrated the models to each other by simulating a single macrophage with both pro- and anti-inflammatory stimuli. Through this initial scenario, we found that modeling the SOCS regulatory feedback loop is important in the definitive resolution of inflammation. We then simulated additional scenarios highlighting the effects of incorporating cell lifespan, recruitment, and various types of external stimuli and initial conditions. Comparison of these scenarios between the ODE model and ABM revealed overall similar behavior of M1 and M2 activation across two very different modeling approaches, suggesting that detailed subcellular pathway modeling is not necessary to achieve complex interplay between M1 and M2 polarization.

In the following sections, we describe the models in detail, the calibrating experiment, and the comparison of various simulated scenarios.

4.2 Methods

4.2.1 ODE subcellular macrophage model

Biological summary

There are several main interactions involved in cell signaling pathways that we include in our model. First, extracellular signals such as TNFα and IL-10 bind to and unbind from their receptors on the cell surface [73, 150]. Receptors transmit signals to other proteins within the cell, which may become activated or phosphorylated [1]. These complexes induce activation of transcription factors, proteins that are responsible for translocating to the nucleus, where they control the transcription of specific genes in the DNA into mRNA. mRNA then undergoes translation in the cytosol, where the protein corresponding to the gene is assembled according to the mRNA sequence [3]. We also account for degradation of various components. We model this process using the law of mass action unless otherwise specified. Details for these interactions are given in the following sections.

TNFα triggers a signaling pathway that leads to activation of the transcription factor NFκB and the subsequent shift to an M1 phenotype [135]. This results in the production of additional TNFα and IL-10 as well as other proteins. Alternatively, IL-10 activates the transcription factor STAT3 through the Jak-STAT pathway, giving rise to M2-type activation [19]. To capture the interactions between these pathways, we developed an ODE model, adapted from Maiti et al. [73] that includes these hallmark signaling pathways. This involves subcellular interactions between receptors and proteins in the cytosol and nucleus of the macrophage.

The model by Maiti et al. [73] initiates their signaling cascade with LPS, a molecule found in Gram-negative bacteria used to experimentally induce an immune response. IKK, a protein whose role is to regulate phosphorylation of IκBα, is activated by both LPS and TNFα. Since we model general lung inflammation in this chapter, we do not rely on activation of the M1 pathway by LPS; rather, we focus on activation via TNFα. Maiti et al.

[73] include production of IL-10 and STAT3; our model extends this by including additional components of the Jak-STAT pathway and the negative feedback loops required to resolve the immune response. In the following sections, we note specifically which equations and terms are novel to our model.

LPS binds to its receptor, TLR4, which activates neutral IKK. IKK then phosphorylates IκBα in the IκBα-NFκB complex, freeing NFκB to translocate to the nucleus. In the absence of a stimulus, IκBα sequesters NFκB to prevent it from causing the production of unnecessary proteins [50, 97]. Transcription factor NFκB initiates transcription of TNFα, IL-10, A20, and IκBα mRNA, resulting in their translation and protein production [116]. As part of a negative feedback loop that prevents excessive production of these proteins, A20 inactivates active IKK and IκBα sequesters unbound NFκB [89]. TNFα and IL-10 are secreted from the cell.

Extracellular TNFα binds to its receptor, activating neutral IKK [116]. IL-10 also binds to its receptor, and JAK and Tyk tyrosine kinases, whose main function is to activate STAT3, bind to this complex as well [133]. Without all of these components, STAT3 cannot be phosphorylated and control transcription of key genes in the nucleus. The IL-10-Jak-Tyk complex activates STAT3, which translocates to the nucleus and initiates the production of IL-10, SOCS1, and SOCS3 [19]. Both SOCS1 and SOCS3 are part of negative feedback loops that bring about resolution of both the M1 and M2 pathways. SOCS3 inhibits transcription of TNFα mRNA and both SOCS1 and SOCS3 inhibit activation of STAT3 [23, 104]. IL-10 also inhibits activation of IKK [34].

Eqs (4.2) through (4.27) are from Maiti et al. [73] unless otherwise noted, and the model variables we added are shown in Eqs (4.28) through (4.38). Figure 4.1 summarizes these interactions, described in more detail in the equations. This schematic differs from that in Chapter 2 wherein the interactions described in Chapter 2 account for tissue-level dynamics, where cells and extracellular signals interact in the bloodstream and site of injury to perform various functions. Here, the schematic describes interactions between receptors, transcription

factors, and other proteins within the cell in response to detection of extracellular signals on the cell surface. Table 4.1 lists the parameters used in the model and their descriptions. Code for these equations can be found in Section B.2.

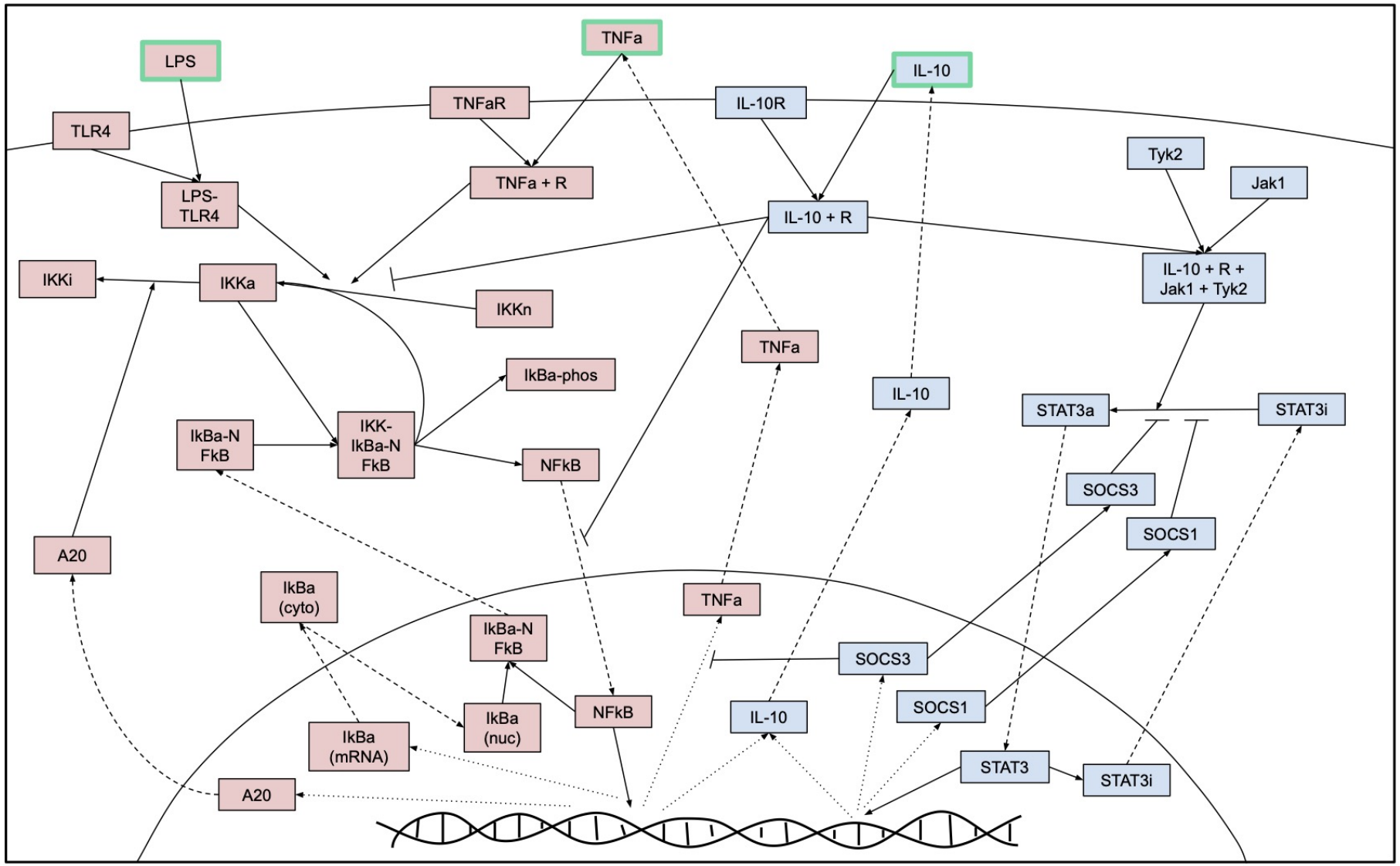


Figure 4.1: Schematic of interactions within a macrophage and with external stimuli LPS, TNFα, and IL-10. Dashed lines represent interactions that involve movement between the cytosol and nucleus, dotted lines represent transcription processes in the nucelus, and solid lines represent all other interactions. Red boxes represent components that are primarily associated with the pro-inflammatory/M1 pathway and blue boxes with the anti-inflammatory/M2 pathway.

	Parameter	Description	Value
1.	a_{trans}	Rate at which $A20$ is translated by $NF\kappa B$	11.338
2.	c_{tf}	Maximum $NF\kappa B$ concentration in nucleus	0.114
3.	$c_{tfstat3}$	Maximum $STAT3$ concentration in nucleus	0.0669
4.	e_{ki}	Rate at which $I\kappa B\alpha$ is imported outside nucleus	2.172×10^{-4}
5.	e_{ni}	Rate at which $I\kappa B\alpha$-NFκB is exported outside nucleus	0.157
6.	i_{ki}	Rate at which $I\kappa B\alpha$ is imported into nucleus	0.0155
7.	$il10_{max}$	$IL10/IL10R$ maximum concentration	5.523×10^{-5}
8.	i_{ln}	Rate at which $NF\kappa B$ is imported into the nucleus	0.0021
9.	k_{bal}	Component balance for TNFα and IL-10	0.0018
9.	$k_{ikbatrans}$	Rate at which $I\kappa B\alpha$ is translated by $NF\kappa B$	0.179
10.	k_{dega20}	Rate at which $A20$ decays	6.227×10^{-4}
11.	$k_{degikba}$	Rate at which phosphorylated $I\kappa B\alpha$ decays	2.232×10^{-4}
12.	$k_{degtnfa}$	Rate at which extracellular $TNF\alpha$ is degraded	1.209×10^{-4}
13.	k_{f1}	Rate at which LPS binds to its receptor	0.275
14.	k_{f3}	Rate at which $TNF\alpha$ binds to its receptor	0.040
15.	k_{f4}	Rate at which $I\kappa B\alpha$ and $NF\kappa B$ associate	0.0023×10^{-4}
16.	k_{fi}	Rate at which IKK is activated	0.093
17.	k_{ilc}	Rate at which $IL10_{cyto}$ moves outside the cell	1.681×10^{-4}
18.	k_{iljb}	Rate at which $JAK1$ and $Tyk2$ are recruited to the $IL10$ complex	0.0078
19.	k_{iliju}	Rate at which $JAK1$ and $Tyk2$ unbind from the $IL10$ complex	0.0246
20.	k_{ilm}	Rate at which IL_{mRNA} move from the nucleus to the cytosol	0.335
21.	k_{ilnf}	Rate at which $IL10_{mRNA}$ is transcribed by $NF\kappa B$	0.234
22.	k_{ilrb}	Rate at which $IL10_{ext}$ binds to its receptor	0.0079
23.	k_{ilru}	Rate at which $IL10_{ext}$ unbinds from its receptor	4.225×10^{-4}
24.	k_{ilsn}	Rate at which $IL10_{mRNA}$ is transcribed by $STAT3$	0.939
25.	k_{in}	Inhibition by IL-10: $\max\left(1-\frac{IL10/R}{IL10/R_{max}},0\right)$	Varies
26.	k_{k1}	Rate at which IKK is inactivated by $A20$	0.0335
27.	k_{k3}	Rate at which IKK associates with $I\kappa B\alpha - NF\kappa B$	0.940
28.	k_{r1}	Rate at which LPS dissociates from its receptor	1.804×10^{-5}
29.	k_{r3}	Rate at which $TNF\alpha$ dissociates from its receptor	0.0032
30.	k_{s1}	Rate at which $SOCS1_{mRNA}$ moves into the cytosol	1.0192
31.	k_{s1st}	Rate at which $SOCS1_{mRNA}$ is transcribed by $STAT3$	1.970
32.	k_{s3}	Rate at which $SOCS3_{mRNA}$ moves into the cytosol	0.0047
33.	k_{s3st}	Rate at which $SOCS3_{mRNA}$ is transcribed by $STAT3$	2.701
34.	k_{sa}	Rate at which activated STAT3 moves into nucelus	5.227×10^{-5}
35.	k_{sec}	Rate at which $TNF\alpha$ is secreted from the cytosol outside the cell	1.694×10^{-4}
36.	k_{sni}	Rate at which activated $STAT3$ in the nucleus becomes deactivated	8.902×10^{-5}
37.	$k_{snicyto}$	Rate at which inactivated $STAT3$ in the nucleus moves into the cytosol	0.0083
38.	k_{stat}	Rate at which IL-10 complex activates STAT3	0.0094
39.	$k_{tnfatrans}$	Rate at which $TNF\alpha$ is translated by $NF\kappa B$	0.389
40.	k_v	Nuclear:cytoplasmic ratio (volume)	1.042
41.	μ_{a20m}	Decay rate of $A20_{mrna}$	0.0114
42.	μ_{ilc}	Decay rate of $IL10_{cyto}$	0.0067
43.	μ_{ile}	Decay rate of $IL10_{ext}$	7.105×10^{-5}
44.	μ_{ilm}	Decay rate of $IL10_{mRNA}$	0.0234
45.	μ_{s1c}	Decay rate of $SOCS1_{cyto}$	3.591
46.	μ_{s1m}	Decay rate of $SOCS1_{mRNA}$	0.139
47.	μ_{s3c}	Decay rate of $SOCS3_{cyto}$	0.110
48.	μ_{s3m}	Decay rate of $SOCS3_{mRNA}$	0.0717
49.	μ_{tnc}	Decay rate of $TNF\alpha_{cyto}$	0.0080
50.	μ_{tnm}	Decay rate of $TNF\alpha_{mrna}$	0.0125
51.	p	Transcription parameter	0.0371
52.	s_m	Rate at which $NF\kappa B$ transcribes mRNA	0.237
53.	$SOCS3_\infty$	Relative effectiveness of $SOCS3_{cyto}$ at inhibiting TNFα transcription	10.609
54.	$SOCS_\infty$	Relative effectiveness of $SOCS1_{cyto}$ and $SOCS3_{cyto}$ at inhibiting activation of STAT3	21.933
55.	t_{i3}	Rate at which $IKK/I\kappa B\alpha/NF\text{-}\kappa B$ is broken down	3.515×10^{-6}

Table 4.1: List of parameter estimates from preliminary fit for the subcellular pathways model.

LPS

Maiti et al. began the model through initiation by LPS, a major component of bacteria identified by the macrophage. LPS is represented as a constant input into the system, shown in Eq (4.1). When LPS is detected by TLR4, its receptor, they form a complex denoted $LPS/TLR4$, shown in Eqs (4.2) and (4.3). Components connected by a forward slash, such

as $LPS/TLR4$, represent a complex; otherwise, variables side by side are multiplied together. We will use this convention in the equations described throughout this section.

$$\frac{dLPS}{dt} = \overbrace{0}^{\text{Constant input}} \tag{4.1}$$

$$\frac{dTLR4}{dt} = \overbrace{-k_{f1}LPS\ TLR4}^{\text{LPS binds to receptor}} + \overbrace{k_{r1}LPS/TLR4}^{\text{LPS unbinds from receptor}} \tag{4.2}$$

$$\frac{dLPS/TLR4}{dt} = \overbrace{k_{f1}LPS\ TLR4}^{\text{LPS binds to receptor}} - \overbrace{k_{r1}LPS/TLR4}^{\text{LPS unbinds from receptor}} \tag{4.3}$$

IκBα kinase

IκBα kinase (IKK) is represented in three distinct states: neutral, active, and inactive, shown in Eqs (4.4), (4.5), and (4.6), respectively. The binding of LPS and TNFα to their respective receptors triggers the activation of neutral IKK, represented by the first term in Eqs (4.4) and (4.5). As part of a negative feedback loop for the pro-inflammatory response, IL-10 inhibits neutral IKK from activating. Maiti et al. describes this inhibition in the first term of Eqs (4.5) and (4.5) through the parameter k_{in}, where

$$k_{in} = \max\left(1 - \frac{IL10/R}{IL10/R_{max}}, 0\right).$$

Active IKK phosphorylates the IKK-IκBα-NFκB complex (second term in Eq (4.5)). Phosphorylation causes the complex to break down, releasing a neutral form of IKK, shown in the second term of Eq (4.4). Finally as part a negative feedback loop to prevent an overactive pro-inflammatory response, the protein A20 inactivates active IKK, the last term of Eq (4.5) and Eq (4.6).

$$\frac{dIKK_n}{dt} = -\overbrace{k_{fi}k_{in}(LPS/TLR4 + TNF\alpha/R)IKK_n}^{\text{IKK activation}} + \overbrace{t_{i3}IKK/I\kappa B\alpha/NF\kappa B_{cyto}}^{\text{Complex breaks down}} \quad (4.4)$$

$$\frac{dIKK_a}{dt} = \overbrace{k_{fi}k_{in}(LPS/TLR4 + TNF\alpha/R)IKK_n}^{\text{IKK activation}} - \overbrace{k_{k3}k_{in}IKK_aI\kappa B\alpha/NF\kappa B_{cyto}}^{\text{IKK binds to I}\kappa\text{ B}\alpha/\text{NF}\kappa\text{B}}$$

$$- \overbrace{k_{k1}IKK_aA20_{cyto}}^{\text{A20 deactivates IKK}} \quad (4.5)$$

$$\frac{dIKK_i}{dt} = \overbrace{k_{k1}IKK_aA20_{cyto}}^{\text{A20 deactivates IKK}} \quad (4.6)$$

IκBα

In a resting state, IκBα sequesters free NFκB by associating into a complex, shown in the first term of Eq (4.7). This process also occurs in the nucleus, from which the complex can move to the cytosol (second term of Eq (4.7)). Activated IKK phosphorylates the complex, represented by the third term in Eq (4.7). The binding of active IKK to IκBα-NFκB (first term of Eq (4.8)) causes all three components to separate, modeled by the second term of Eq (4.8): NFκB is released, IκBα is degraded, and IKK returns to a neutral state.

$$\frac{dI\kappa B\alpha/NF\kappa B_{cyto}}{dt} = \overbrace{k_{f4}NF\kappa B_{cyto}I\kappa B\alpha_{cyto}}^{\text{Association}} + \overbrace{e_{ni}I\kappa B\alpha/NF\kappa B_{nuclear}k_v}^{\text{Moves outside nucleus}}$$

$$- \overbrace{k_{k3}k_{in}IKK_aI\kappa B\alpha/NF\kappa B_{cyto}}^{\text{IKK binds to I}\kappa\text{B}\alpha/\text{NF}\kappa\text{B}} \quad (4.7)$$

$$\frac{dIKK_a/I\kappa B\alpha/NF\kappa B_{cyto}}{dt} = \overbrace{k_{k3}k_{in}IKK_aI\kappa B\alpha/NF\kappa B_{cyto}}^{\text{IKK binds to I}\kappa\text{B}\alpha/\text{NF}\kappa\text{B}} - \overbrace{t_{i3}IKK/I\kappa B\alpha/NF\kappa B_{cyto}}^{\text{Complex breaks down}} \quad (4.8)$$

Eqs (4.9) through (4.12) show the various states of the inhibitory protein IκBα. NFκB

promotes the transcription of IκBα mRNA, shown in the first term of Eq (4.9). Subsequent translation of the protein and decay of the mRNA are described in the first term of Eq (4.10) and the second term of Eq (4.9), respectively. As previously described, the second term of Eq (4.10) represents IκBα sequestering free NFκB in the cytosol. In a resting cell, excess IκBα is distributed evenly between the cytosol and nucleus; thus, the last two terms of Eq (4.10) show import and export of IκBα between the two compartments [69]. The parameter k_v accounts for the nuclear-cytoplasmic ratio to account for the size of the cell's cytoplasm in relation to its nucleus. The release of NF-κB from the IκBα-NF-κB complex by active IKK results in the phosphorylation of IκBα and its subsequent degradation, shown in the two terms of Eq (4.12).

$$\frac{dI\kappa B\alpha_{mrna}}{dt} = \overbrace{\frac{s_m pNF\kappa B_{nuclear}}{c_{tf} + NF\kappa B_{nuclear}}}^{\text{Transcription via NF}\kappa\text{B}} - \overbrace{\mu_{ilm} I\kappa B\alpha_{mrna}}^{\text{Decay}} \tag{4.9}$$

$$\begin{aligned}\frac{dI\kappa B\alpha_{cyto}}{dt} =& \overbrace{k_{ikbatrans} I\kappa B\alpha_{mrna}}^{\text{Translation}} - \overbrace{k_{f4} NF\kappa B_{cyto} I\kappa B\alpha_{cyto}}^{\text{Association}} \\ & - \overbrace{i_{ki} I\kappa B\alpha_{cyto}}^{\text{Import to nucleus}} + \overbrace{e_{ki} I\kappa B\alpha_{nuclear} k_v}^{\text{Export from nucleus}}\end{aligned} \tag{4.10}$$

$$\frac{dI\kappa B\alpha_{nuclear}}{dt} = \overbrace{- k_{f4} NF\kappa B_{nuclear} I\kappa B\alpha_{nuclear}}^{\text{Association}} + \overbrace{\frac{i_{ki}}{k_v} I\kappa B\alpha_{cyto}}^{\text{Import to nucleus}} - \overbrace{e_{ki} I\kappa B\alpha_{nuclear}}^{\text{Export from nucleus}} \tag{4.11}$$

$$\frac{dI\kappa B\alpha_{phospho}}{dt} = \overbrace{t_{i3} IKK_a / I\kappa B\alpha / NF\kappa B_{cyto}}^{\text{IKK releases NF}\kappa\text{B}} - \overbrace{k_{degikba} I\kappa B\alpha_{phospho}}^{\text{Decay}} \tag{4.12}$$

NFκB

The protein NFκB is released from the complex (first term of Eq (4.13)) and translocates to the nucleus, represented by the second term of Eq (4.13) [69]. NFκB activates the transcription of several genes, including TNFα and IL-10, A20, and IκBα. IκBα sequesters nuclear NFκB (last term in Eq (4.14) and first term in Eq (4.15)) before the complex moves back

into the cytosol, shown in the last term of Eq (4.15).

$$\frac{dNF\kappa B_{cyto}}{dt} = \overbrace{t_{i3}IKK_a/I\kappa B\alpha/NF\kappa B_{cyto}}^{\text{IKK releases NF}\kappa\text{B}} - \overbrace{i_{ln}k_{in}NF\kappa B_{cyto}}^{\text{Moves to nucleus}} - \overbrace{k_{f4}NF\kappa B_{cyto}\ I\kappa B\alpha_{cyto}}^{\text{I}\kappa\text{B}\alpha\text{ sequesters NF}\kappa\text{B}} \tag{4.13}$$

$$\frac{dNF\kappa B_{nuclear}}{dt} = \overbrace{\frac{i_{ln}k_{in}NF\kappa B_{cyto}}{k_v}}^{\text{Moves to nucleus}} - \overbrace{k_{f4}NF\kappa B_{nuclear}I\kappa B\alpha_{nuclear}}^{\text{I}\kappa\text{B}\alpha\text{ sequesters NF}\kappa\text{B}} \tag{4.14}$$

$$\frac{dI\kappa B\alpha/NF\kappa B_{nuclear}}{dt} = \overbrace{k_{f4}NF\kappa B_{nuclear}I\kappa B\alpha_{nuclear}}^{\text{I}\kappa\text{B}\alpha\text{ sequesters NF}\kappa\text{B}} - \overbrace{e_{ni}I\kappa B\alpha/NF\kappa B_{nuclear}}^{\text{Moves outside nucleus}} \tag{4.15}$$

TNFα

One of the main targets of gene expression of NFκB is the pro-inflammatory cytokine TNFα. The first term of Eq (4.16) represents transcription of mRNA. There is evidence that Suppressor of Cytokine Signaling 3 (SOCS3), discussed in further detail below, plays a role in regulating the pro-inflammatory response by inhibiting TNFα mRNA and protein production, although the exact mechanisms by which this occurs is still unclear [28, 104]. We included a multiplier, not in the original equation by Maiti et al., in this first term to represent inhibition of mRNA production by SOCS3. After transcription and translation, TNFα is secreted from the cell (first two terms of Eq (4.17)). The parameter k_{bal} represents a component balance for TNFα as it moves from the cytosol to the supernatant.

Extracellular TNFα binds to its receptor on the cell surface, represented by the second term in Eq (4.18). In some cases the cytokine unbinds from its receptor, accounted for by the second term in Eq (4.18). Once inside the cell, either after binding to its receptor or being translocated from the nucleus, TNFα performs several important roles. Shown in the first term of Eq (4.4), TNFα bound to its receptor upregulates activation of IKK, which then precipitates further NFκB transcription.

$$\frac{dTNF\alpha_{mrna}}{dt} = \overbrace{\frac{s_m p\ NF\kappa B_{nuclear}}{c_{tf} + NF\kappa B_{nuclear}}}^{\text{Transcription via NF}\kappa\text{B}} \overbrace{\left(\frac{1}{1 + \left(\frac{SOCS3_{cyto}}{SOCS3_{\infty}}\right)^2}\right)}^{\text{Inhibition by SOCS3}} - \overbrace{\mu_{tnm} TNF\alpha_{mrna}}^{\text{Decay}} \tag{4.16}$$

$$\frac{dTNF\alpha_{cyto}}{dt} = \overbrace{k_{tnfatrans} TNF\alpha_{mrna}}^{\text{Translation}} - \overbrace{k_{sec} TNF\alpha_{cyto}}^{\text{Secreted from cell}} - \overbrace{\mu_{tnc} TNF\alpha_{cyto}}^{\text{Decay}} \tag{4.17}$$

$$\frac{dTNF\alpha_{ext}}{dt} = \overbrace{k_{sec} k_{bal} TNF\alpha_{cyto}}^{\text{Secreted from cell}} - \overbrace{k_{f3} TNF\alpha_{ext} TNF\alpha R}^{\text{TNF}\alpha\text{ binds to receptor}} + \overbrace{k_{r3} TNF\alpha/R}^{\text{TNF}\alpha\text{ unbinds from receptor}}$$

$$- \overbrace{k_{degtnfa} TNF\alpha_{ext}}^{\text{Decay}} \tag{4.18}$$

$$\frac{dTNF\alpha R}{dt} = - \overbrace{k_{f3} TNF\alpha_{ext} TNF\alpha R}^{\text{TNF}\alpha\text{ binds to receptor}} + \overbrace{k_{r3} TNF\alpha/R}^{\text{TNF}\alpha\text{ unbinds from receptor}} \tag{4.19}$$

$$\frac{dTNF\alpha/R}{dt} = \overbrace{k_{f3} TNF\alpha_{ext} TNF\alpha R}^{\text{TNF}\alpha\text{ binds to receptor}} - \overbrace{k_{r3} TNF\alpha/R}^{\text{TNF}\alpha\text{ unbinds from receptor}} \tag{4.20}$$

A20

As mentioned previously, A20 is another NFκB-responsive gene responsible for deactivating IKK, which blocks NFκB translocation to the nucleus. Eq (4.21) shows transcription and subsequent degradation of A20 mRNA. Eq (4.22) shows translation of the protein in the cytosol, and A20 decays at rate k_{dega20}, second term in Eq 4.22.

$$\frac{dA20_{mrna}}{dt} = \overbrace{\frac{s_m\ p\ NF\kappa B_{nuclear}}{c_{tf} + NF\kappa B_{nuclear}}}^{\text{Transcription via NF}\kappa\text{B}} - \overbrace{\mu_{a20m} A20_{mrna}}^{\text{Degradation}} \tag{4.21}$$

$$\frac{dA20_{cyto}}{dt} = \overbrace{a_{trans} A20_{mrna}}^{\text{Translation}} - \overbrace{k_{dega20} A20_{cyto}}^{\text{Decay}} \tag{4.22}$$

IL-10

A hallmark of the anti-inflammatory response is the cytokine IL-10. Its gene is a target of NFκB transcription and is involved in the regulation of the pro-inflammatory response. Some events related to IL-10 production and function are included in the model by Maiti et al. [73], but we expand the model to include a fuller view of the role of IL-10 and an important pathway it activates.

Extracellular IL-10 can bind to and unbind from its receptor IL-10R, as modeled by the first two terms in Eq (4.23) [90]. For simplicity, we assume the total number of receptors is conserved. The first term in Eq (4.25) describes upregulation of the IL-10 gene by transcription factors NFκB and STAT3. Maiti et al. include the constants 0.4 and 0.6 such that NFκB is responsible for 40% of the transcription rate and STAT3 is responsible for the other 60%. The nonlinear terms represent maximum possible rates of IL-10 transcription, since space in the nucleus is limited. IL-10 is translated from its mRNA and secreted from the cell (first two terms of Eq (4.26)). The third term in Eq (4.23) includes a component balance k_{bal} between the cytosol and supernatant. Baseline degradation rates for extra- and intracellular IL-10 and IL-10 mRNA is included in Eqs (4.23), (4.26), and (4.25), respectively.

$$\frac{dIL10_{ext}}{dt} = -\overbrace{k_{ilrb}IL10_{ext}IL10R}^{\text{Binds to receptor}} + \overbrace{k_{ilru}IL10/R}^{\text{Unbinds from receptor}} + \overbrace{k_{ilc}k_{bal}IL10_{cyto}}^{\text{Moves outside cell}} - \overbrace{\mu_{ile}IL10_{ext}}^{\text{Decay}} \tag{4.23}$$

$$\frac{dIL10R}{dt} = -\overbrace{k_{ilrb}IL10_{ext}IL10R}^{\text{IL-10 binds to receptor}} + \overbrace{k_{ilru}IL10/R}^{\text{IL-10 unbinds from receptor}} \tag{4.24}$$

$$\frac{dIL10_{mRNA}}{dt} = \overbrace{0.4k_{ilnf}p\frac{NF\kappa B}{c_{tf}+NF\kappa B} + 0.6k_{ilsn}p\frac{STAT3_n}{c_{tfstat3}+STAT3_n}}^{\text{Gene transcription}} - \overbrace{\mu_{ilm}IL10_{mRNA}}^{\text{Decay}} \tag{4.25}$$

$$\frac{dIL10_{cyto}}{dt} = \overbrace{k_{ilm}IL10_{mRNA}}^{\text{Translation}} - \overbrace{k_{ilc}IL10_{cyto}}^{\text{Moves outside cell}} - \overbrace{\mu_{ilc}IL10_{cyto}}^{\text{Decay}} \tag{4.26}$$

JAK-STAT signaling

Aside from inhibitory functions, IL-10 signaling initiates the JAK-STAT signaling pathway, a primary mechanism through which the immune response mediates inflammation [107]. The protein tyrosine kinases JAK1 and Tyk2 are recruited to the IL-10/IL-10 receptor complex, shown in the third term of Eq (4.27). This creates a new complex, $IL10/R/JAK1/Tyk2$, Eq (4.30) [115]. The second term accounts for the possibility that the complex may break apart. JAK1 (Eq (4.28)) and Tyk2 (Eq (4.29)) concentrations are conserved, assuming enzyme-type dynamics. In light of the many components involved in creating this complex, we explored incorporating the various combinations of the binding steps, such as the individual receptor components, each of which bind to a specific tyrosine kinase. In the end, we decided to model the recruitment of JAK1 and Tyk2 to the IL-10/IL-10 receptor complex as one step; this still captures the appropriate dynamics without adding more parameters and equations. The last two terms of Eq (4.27) and all of Eqs (4.28) through (4.30) are our additions to the original model by Maiti et al., with terms representing activation of STAT3 through the Jak-STAT pathway adapted from Moya et al. [90].

$$\frac{dIL10/R}{dt} = \overbrace{k_{ilrb} IL10_{ext} IL10R}^{\text{IL-10 binds to receptor}} - \overbrace{k_{ilru} IL10/R}^{\text{IL-10 unbinds from receptor}}$$

$$- \overbrace{k_{iljb} IL10/R\ JAK1\ Tyk2}^{\text{Recruitment of } JAK1 \text{ and } Tyk2} + \overbrace{k_{ilju} IL10/R/JAK1/Tyk2}^{\text{Dissociation of } JAK1 \text{ and } Tyk2} \quad (4.27)$$

$$\frac{dJAK1}{dt} = - \overbrace{k_{iljb} IL10/R\ JAK1\ Tyk2}^{\text{Recruitment of } JAK1 \text{ and } Tyk2} + \overbrace{k_{ilju} IL10/R/JAK1/Tyk2}^{\text{Dissociation of } JAK1 \text{ and } Tyk2} \quad (4.28)$$

$$\frac{dTyk2}{dt} = - \overbrace{k_{iljb} IL10/R\ JAK1\ Tyk2}^{\text{Recruitment of } JAK1 \text{ and } Tyk2} + \overbrace{k_{ilju} IL10/R/JAK1/Tyk2}^{\text{Dissociation of } JAK1 \text{ and } Tyk2} \quad (4.29)$$

$$\frac{dIL10/R/JAK1/Tyk2}{dt} = \overbrace{k_{iljb} IL10/R\ JAK1\ Tyk2}^{\text{Recruitment of } JAK1 \text{ and } Tyk2} - \overbrace{k_{ilju} IL10/R/JAK1/Tyk2}^{\text{Dissociation of } JAK1 \text{ and } Tyk2)} \quad (4.30)$$

The IL-10/IL-10 receptor/JAK1/Tyk2 complex serves as a temporary docking station for inactive Signal Transducer and Activator of Transcription 3 (STAT3) [112]. Upon recruitment to the complex, STAT3 is activated and undergoes homodimerization, shown in the first term of Eq (4.31). Maiti et al. modeled the recruitment and activation of STAT3 through binding of STAT3 to the IL-10/IL-10R complex without Jak1 and Tyk2. We also included a multiplier representing inhibition by Suppressors of Cytokine Signaling 1 and 3 (SOCS1 and SOCS3), two IL-10 responsive genes as well as the second term of Eq (4.33) and Eq (4.34) which allow for the conservation of STAT3 in the model. SOCS1 inhibits JAK1 function by binding its SH2 domain to JAK1, preventing STAT3 from docking to the IL-10 complex. SOCS3 performs a similar role but docks to the receptor; since we do not model at the level of detail of specific binding locations, we model this inhibition as having the same result, which is preventing STAT3 from activating [23, 126, 147].

STAT3 translocates to the nucleus (second term of Eq (4.32)) and controls transcription of several IL-10 responsive genes. The main inhibitor of STAT3 function is PIAS3. The protein binds to activated STAT3, preventing further transcription [146]. We model this by including

a deactivation term with rate k_{sni}, shown in the second term of Eq (4.33). Assuming enyzme-type dynamics for all states of STAT3, the transcription factor is conserved, and deactivated nuclear STAT3 returns to the cytosol in the last term of Eq (4.34).

$$\frac{dSTAT3_i}{dt} = -\overbrace{2k_{stat}IL10/R/JAK1/Tyk2\ STAT3_i^2}^{\text{STAT3 activation}}\overbrace{\left(\frac{1}{1+\left(\frac{SOCS1_{cyto}+SOCS3_{cyto}}{SOCS_\infty}\right)^2}\right)}^{\text{Inhibition by SOCS1/3}} + \overbrace{k_{snicyto}STAT3_{ni}}^{\text{Moves to cytosol}} \tag{4.31}$$

$$\frac{dSTAT3_a}{dt} = \overbrace{k_{stat}IL10/R/JAK1/Tyk2\ STAT3_i^2}^{\text{STAT3 activation}}\overbrace{\left(\frac{1}{1+\left(\frac{SOCS1_{cyto}+SOCS3_{cyto}}{SOCS_\infty}\right)^2}\right)}^{\text{Inhibition by SOCS1/3}} - \overbrace{k_{sa}STAT3_a}^{\text{Moves to nucleus}} \tag{4.32}$$

$$\frac{dSTAT3_n}{dt} = \overbrace{k_{sa}STAT3_a}^{\text{Moves to nucleus}} - \overbrace{k_{sni}STAT3_n}^{\text{Deactivation}} \tag{4.33}$$

$$\frac{dSTAT3_{ni}}{dt} = \overbrace{k_{sni}STAT3_n}^{\text{Deactivation}} - \overbrace{k_{snicyto}STAT3_{ni}}^{\text{Moves to cytosol}} \tag{4.34}$$

SOCS

The inclusion of SOCS, represented in Eqs (4.35) through (4.38), is also novel to our model as compared to that by Maiti et al. Suppressors of Cytokine Signaling 1 and 3 (SOCS1, SOCS3) are upregulated via STAT3 transcription and translation, first two terms of Eqs (4.35) and (4.36), respectively [19, 51]. The last terms of these two equations represent natural degradation of the mRNA.

$$\frac{dSOCS1_{mRNA}}{dt} = \overbrace{k_{s1st}STAT3_n}^{\text{Gene transcription}} - \overbrace{k_{s1}SOCS1_{mRNA}}^{\text{Translation}} - \overbrace{\mu_{s1m}SOCS1_{mRNA}}^{\text{Decay}} \tag{4.35}$$

$$\frac{dSOCS3_{mRNA}}{dt} = \overbrace{k_{s3st}STAT3_n}^{\text{Gene transcription}} - \overbrace{k_{s3}SOCS3_{mRNA}}^{\text{Translation}} - \overbrace{\mu_{s3m}SOCS3_{mRNA}}^{\text{Decay}} \tag{4.36}$$

$$\frac{dSOCS1_{cyto}}{dt} = \overbrace{k_{s1}SOCS1_{mRNA}}^{\text{Translation}} - \overbrace{\mu_{s1c}SOCS1_{cyto}}^{\text{Decay}} \tag{4.37}$$

$$\frac{dSOCS3_{cyto}}{dt} = \overbrace{k_{s3}SOCS3_{mRNA}}^{\text{Translation}} - \overbrace{\mu_{s3c}SOCS3_{cyto}}^{\text{Decay}} \tag{4.38}$$

We used this model that includes both pro- and anti-inflammatory signaling pathways to provide a fuller picture of the spectrum of activation that can occur within a macrophage. In the following pages, we discuss how this model was implemented and compared to the ABM.

4.2.2 Parameters & initial conditions for ODE model

We used data from Maiti et al. [73] to obtain initial parameter values as a starting point. This was not a complete parameter estimation (calculating sensitivities, etc.), but rather a first step in obtaining parameter values and initial conditions that produce dynamics that are roughly expected and have the correct scales. Due to challenges with processed data provided by the authors, we did not rely on fits to their data. Instead, we ensured that parameter values produced similar behavior between single and multiple macrophage simulations.

Since the model simulations by Maiti et al. [73] were initialized with LPS, once the final parameter set was obtained the model was run for 1,000 hours with no LPS using the code provided in Section B.2. The ending values of these simulations for each variable were determined to be the baseline initial conditions, representing a state of no macrophage activation.

4.2.3 Modeling multiple macrophages

The equations described in the section above represent the pathways in a single macrophage. To model recruitment and cell lifespan, we extended the model such that the equations are copied ten times to represent ten macrophages. These macrophages share the same extracellular components: LPS, IL-10, and TNFα. Figure 4.2 shows a visualization of this compartmental model. Furthermore, each macrophage is randomly assigned a lifespan, 12 ± 3 hours. At the end of each cell's lifespan, the variables in the signaling pathway are returned to a naive state to represent the recruitment of a naive cell.

Our aim in constructing a model of multiple macrophages was to examine how macrophages in close proximity behave in response to extracellular stimuli while still utilizing the ODE structure. Resulting dynamics of variables that exist in each model can be viewed separately or averaged together to obtain the average behavior across all macrophages.

4.2.4 Agent-based M1/M2 model

Our ABM pro- and anti-inflammatory mediators (PIM and AIM, respectively), M0, M1, and M2 macrophages, and SOCS on a 40-by-40 grid, implemented using object-oriented programming in MATLAB (code provided in Section B.1). Macrophages are mobile agents with M1/M2 activation and SOCS levels as associated attributes. Each macrophage may take up one patch, and pro- and anti-inflammatory mediators are measured by amount on each patch, diffusing across the grid over time. We do not specifically model TNFα and IL-10 but rather group together general mediators with pro- and anti-inflammatory roles. The model can be initialized with varying levels of any of these components and simulated to obtain the resulting dynamics. The model performs a series of steps to recruit macrophages, determine M1/M2 activation, and produce and inhibit pro- and anti-inflammatory mediators and SOCS. Each macrophage has levels of M1 and M2 activation, where $0 \leq \mathrm{M1} + \mathrm{M2} \leq 1$, and these activation levels are updated based on the surrounding levels of pro- and anti-inflammatory mediators. Figure 4.3 summarizes the steps taken during every iteration of

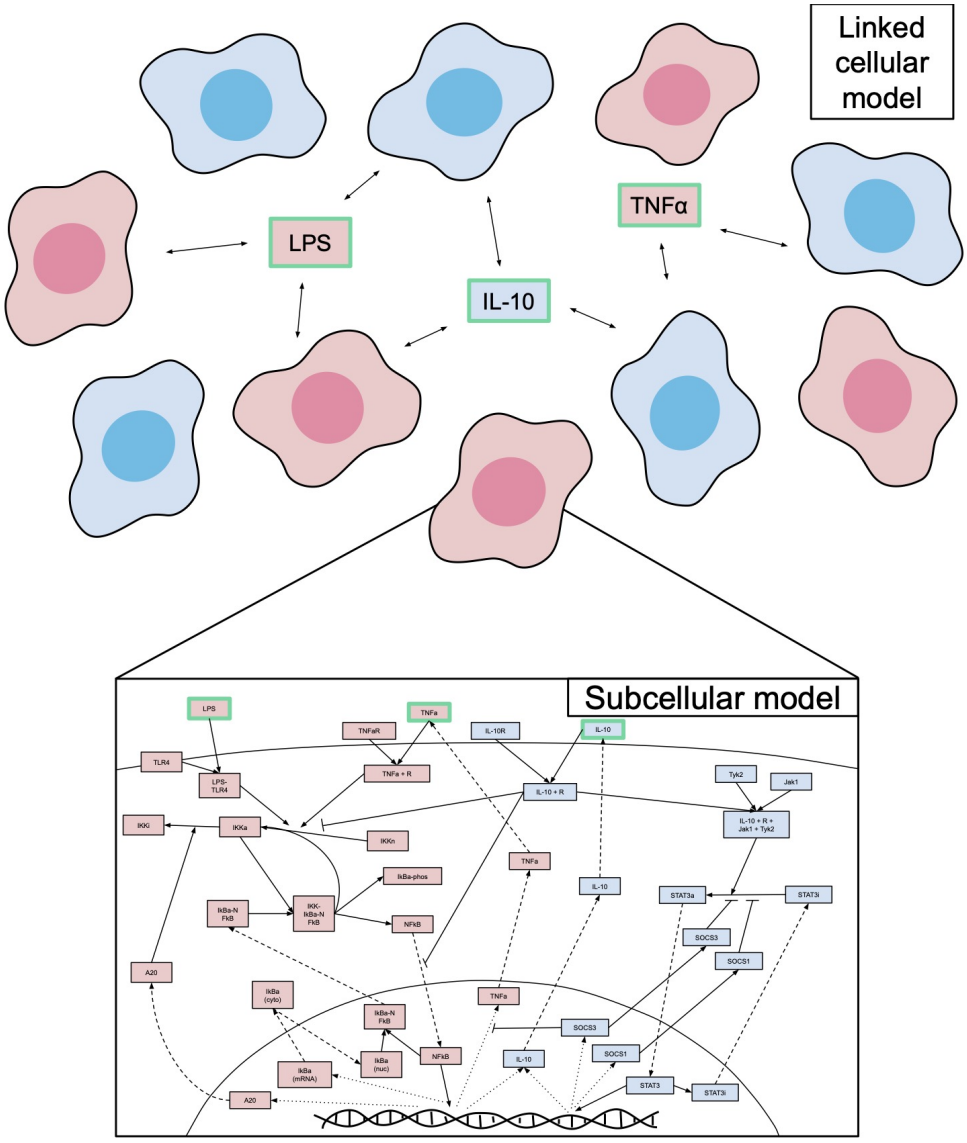


Figure 4.2: A representation of the multiple macrophages ODE model, in which each macrophage is a compartment with its own set of subcellular signaling pathways, and all ten macrophages share external stimuli LPS, TNFα, and IL-10.

the simulation, where each iteration represents 20 minutes. These steps are based on the same interactions described in the ODE model.

4.2.5 Calibrating experiment and scenarios

To be able to compare the models to each other, we implemented the same scenario, which we called a "calibrating experiment," in each model and tuned the ABM results so that PIM & AIM and M1 & M2 activation results were similar to their corresponding components in the ODE model, since the ODE model parameters were already set (see Section 4.2.2 for process). These ODE model components were extracellular TNFα & IL-10 and TNFα mRNA & IL-10 mRNA, respectively. We chose M1 and M2 activation to be represented by TNFα and IL-10 mRNA, respectively, since mRNA is produced via downstream signaling initiated by the surrounding environment and also results in specific proteins that are secreted from the cell. Thus, mRNA associated with the cell's phenotype both reflects and drives macrophage polarization.

Tuning parameters so that the ABM and ODE model returned similar dynamics in the calibrating experiment allowed us to obtain similar behavior at baseline and compare the results of more complicated experiments. We chose this scenario to be a single macrophage with a high pro-inflammatory stimulus and without cell death. In the ODEs, initial conditions were established such that all variables are at baseline levels, to represent an M0 macrophage. TNFα, the variable representing a pro-inflammatory stimulus in the ODE model, was set to 10 pg/mL, consistent with experimental methods [60].

For a naive macrophage in the ABM, we used a 3-by-3 grid so that the cell could move but interact only with the mediators in its immediate proximity. A naive macrophage in this model is defined as having activation M1 + M2 < 0.25; M1 and M2 activation were randomly chosen with bounds that satisfy this condition. Pro-inflammatory mediators do not have specific units but after exploratory simulations, we considered a concentration of 30 in the center space of the grid to be sufficient to mount an inflammatory response. ABM

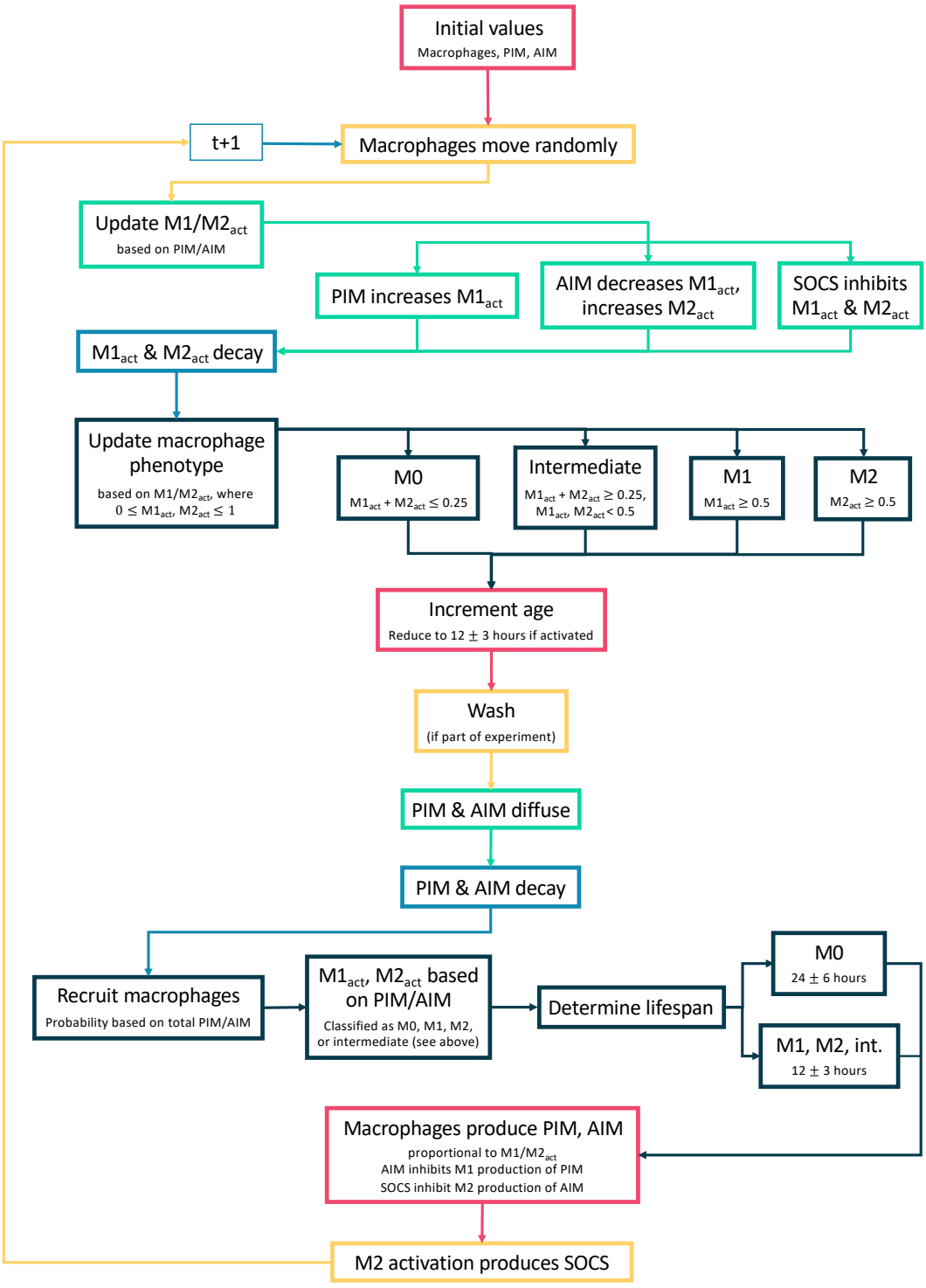


Figure 4.3: Description of steps in ABM for each iteration of the simulation.

parameters were tuned manually to match the dynamics observed in the ODE.

Through simulating the calibrating experiment and scenarios, described below, we found that receptor-bound TNFα and IL-10 in the ODE model played an important role in the resulting dynamics. Many modelers do not model changes in cytokine levels due to binding to receptors, assuming this amount is negligible. However, we found that this is not the case in our ODE model, and explicitly modeling receptors makes a difference in dynamics. Receptors were not explicitly modeled in the ABM; macrophage activation is based solely on the surrounding PIM and AIM. This can create a disparity in the amount of PIM and AIM that are compared between the two models. In Figures 4.4 and 4.5 and in our results, we showed two cases of the ODE model: when only extracellular TNFα and IL-10 are considered, and when both extracellular and receptor-bound TNFα and IL-10 are considered. We also discussed differences between these two cases.

In the calibrating experiment, we set the ODE and ABM parameters and initial conditions such that a single macrophage would exhibit similar M1 and M2 behavior when initialized with PIM (process described in Methods section). Figure 4.4 shows the results of this simulation. All transients are normalized for comparison because the units in the models vary. To do this, we scaled each transient by its maximum. The results of the ABM in Figure 4.4 is the result of 50 simulations; on the other hand, the ODE model with a single macrophage is deterministic and thus only one simulation is necessary.

M2 activation occurs slightly earlier in the ABM than in the ODE, but we concluded that the results were similar enough to proceed with comparisons. Adding receptor-bound TNFα and IL-10 to their extracellular counterparts did not make a significant difference in the results. We also considered the magnitudes of M1 and M2 activation in relation to each other, shown in Figure 4.5. M1 and M2 activation in the ABM are, by definition, bound between 0 and 1. To compare with the ABM, we scaled TNFα and IL-10 mRNA in the ODE by the maximum of TNFα. Peak M2 activation in both the ABM and ODE are about half the peak M1 activation. This shows important dynamics observed in both models, which

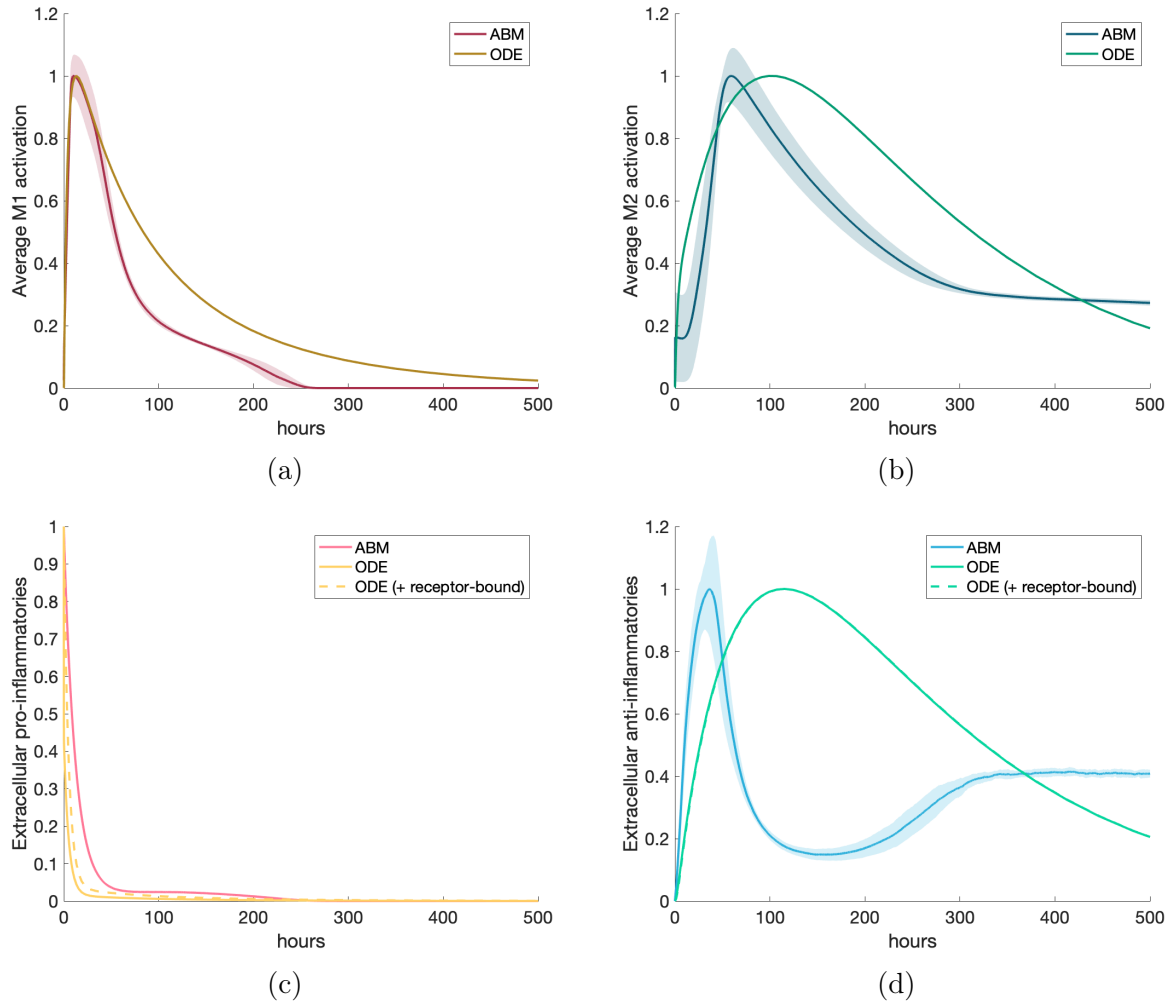


Figure 4.4: Calibrating experiment: single macrophage activated by a pro-inflammatory stimulus. ABM and ODE results are shown on the same plots for comparison. All transients are scaled by their maximums. Dotted lines represent extracellular TNFα or IL-10 with receptor-bound TNFα or IL-10, respectively. (a) M1 activation, (b) M2 activation, (c) pro-inflammatory mediators, (d) anti-inflammatory mediators.

illustrates the strength of the calibration.

Once the parameters were set and the calibrating experiment was simulated, we changed the initial conditions to represent six additional scenarios, which will be described in greater detail below. First, we used the same single-macrophage model as described above but with an anti-inflammatory stimulus. Then, using the 40-by-40 grid for the ABM and ten-macrophage model for the ODE, we incorporated recruitment/turnover and cell lifespan. For these larger models, we simulated the following scenarios, the results of which will be

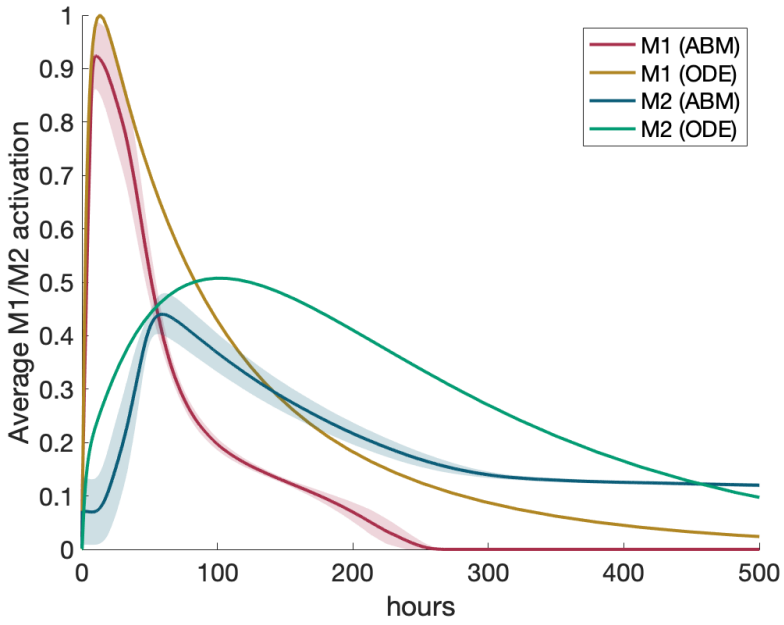


Figure 4.5: Calibrating experiment: M1 and M2 activation resulting from the calibrating experiment. ODE results are scaled by the maximum M1 activation to compare to activation in the ABM, which is bound by 0 and 1.

discussed in the following section:

1. Naive macrophages with large pro-inflammatory stimulus
2. Naive macrophages with large anti-inflammatory stimulus
3. M1 macrophages with anti-inflammatory stimulus
4. Half M1, half M2 macrophages
5. Pro-inflammatory stimulus, wash at hour 12, then anti-inflammatory stimulus

4.3 Results

We simulated equivalent scenarios in an ODE model and an agent-based model of M1/M2 activation in response to general inflammatory stimuli. In this section we compare the results of the two models to shed light on the benefits of each model type and, in particular,

examine whether the incorporation of a spatial component through an ABM or the incorporation of hallmark signaling pathways through an ODE improve the value of the models in understanding immune system dynamics. All results shown are the average of 50 simulations except the single-macrophage simulations, which is deterministic since age is not a factor. Code is provided in Sections B.1 and B.2.

4.3.1 Scenario 1: macrophage with anti-inflammatory stimulus

For the first scenario, we used the same structure of a single macrophage as in the calibrating experiment. Instead of a pro-inflammatory stimulus, we used an anti-inflammatory stimulus. Figure 4.6 shows the results of this simulation. AIM and M2 activation behave roughly the same; in the ABM, M2 activation decreases slightly slower than in the ODE. For the ABM, in both the calibrating experiment and this scenario, there is a slight increase in AIM later in time. This may be due to the small amount of SOCS left at this time, allowing AIM to increase slightly before decaying completely due to decreasing M2 activation. Including receptor-bound mediators in the ODE reveals a slower decrease in AIM over time but overall similar behavior to the ABM. A small increase in M1 activation and PIM also occurs later in time in the ODE model; this is due to trace amounts of NFκB in the baseline levels of the cell that result in a small amount of TNFα production downstream. Additionally, we noted that simulating a single macrophage in the ABM shows consistent results for each of the 50 simulations, since the shaded regions around the curves, representing standard deviation, are very small or nearly zero.

Figure 4.6(c) shows that some PIM is produced in the ABM due to a small percentage of M1 activation existing in the naive macrophages (see Figure 4.3 to see how naive macrophages are defined), but both models show a decrease to zero in the presence of a large concentration of AIM. Similarly to Figure 4.5 in the calibrating experiment, we show M1 activation in relation to M2 activation in Figure 4.7 to better visualize the magnitude of the pro-inflammatory response, which is very small in relation to the much larger anti-

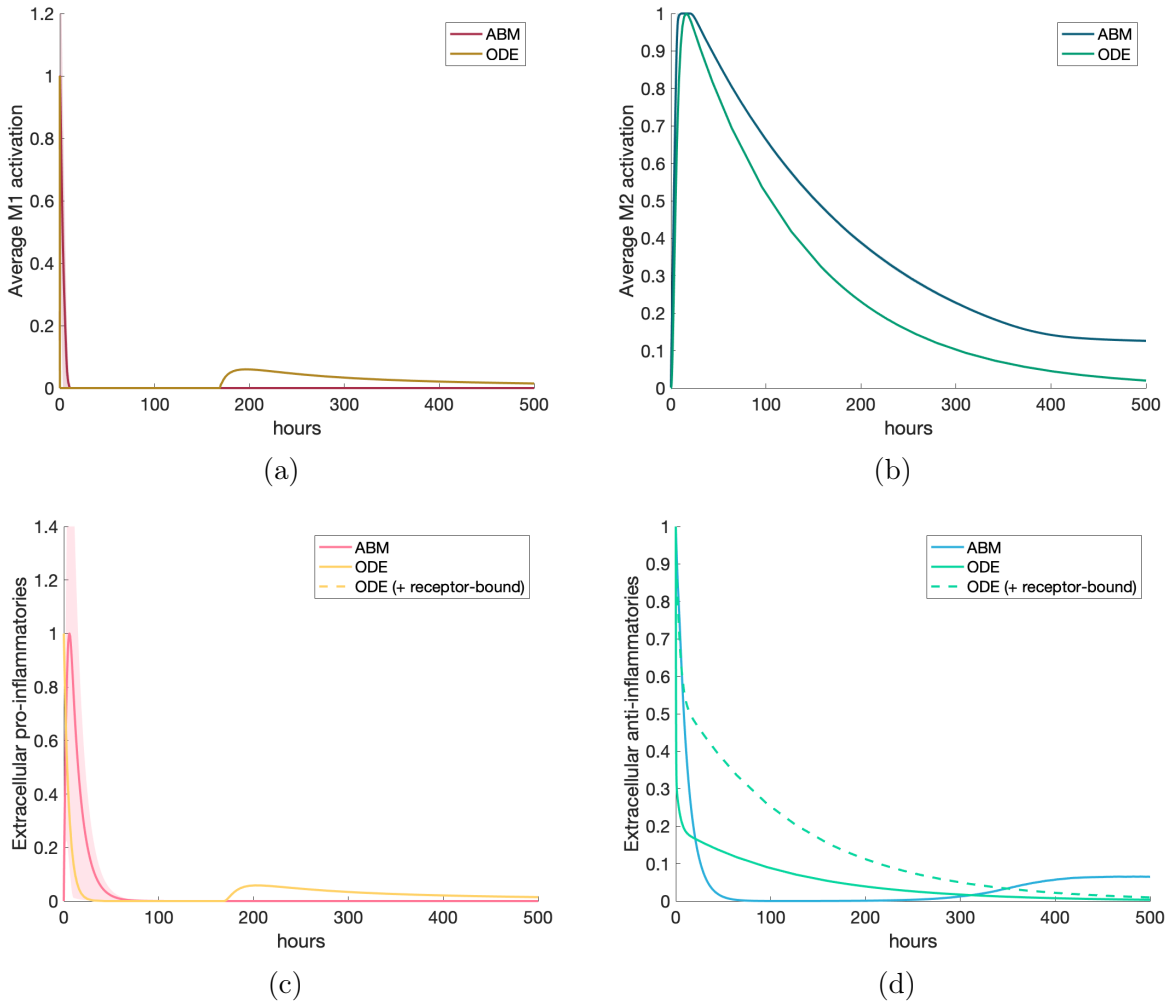


Figure 4.6: Scenario 1: Simulation of single response to an anti-inflammatory stimulus. All transients are scaled individually by their maximums. (a) M1 activation, (b) M2 activation, (c) PIM, (d) AIM.

inflammatory stimulus.

4.3.2 Scenario 2: multiple macrophages with pro-inflammatory stimulus

We then introduced recruitment/turnover and cell lifespan. In the ABM, the grid was expanded to 40-by-40 with ten M0 macrophages initially, and the recruitment feature was turned on. Naive and activated macrophages were randomly assigned lifespans of 24 ± 6 and 12 ± 3 hours, respectively. In the ODE, all ten macrophage compartments were utilized

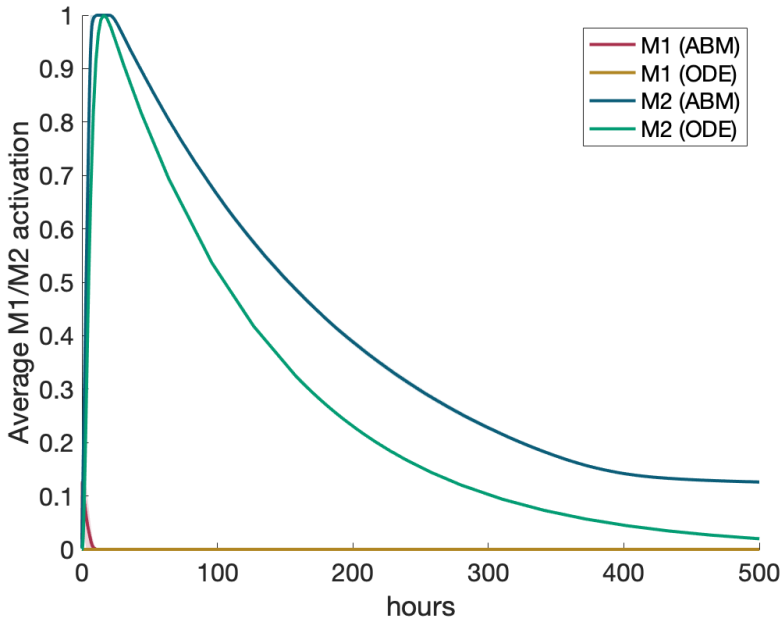


Figure 4.7: Scenario 1: M1 and M2 activation resulting from the calibrating experiment. ODE and ABM results are scaled by the maximum M2 activation to compare to maximum M1 activation, which is nearly nonexistent in comparison to M2.

and had lifespans of 12 ± 3 hours. In this scenario, we introduced a large pro-inflammatory stimulus into the model. Results are shown in Figure 4.8.

In this scenario, M1 and M2 activation in the ODE occur before the ABM, despite similar dynamics for the anti-inflammatory mediators between the two models (panel (d)). Including receptor-bound TNFα (Figure 4.8(c)) makes a significant difference in the dynamics. Our ODE model shows that when naive macrophages are introduced into an environment with a high concentration of TNFα, receptors quickly bind to free TNFα. Therefore, extracellular TNFα in the ODE did not compare well to PIM in the ABM, since receptors are not modeled in the ABM. When receptor-bound TNFα was added to the PIM total shown in Figure 4.8, the dynamics matched up almost perfectly to the ABM. Figure 4.8(d) shows almost no difference between extracellular IL-10 only and extracellular IL-10 with receptor-bound IL-10, suggesting that accounting for both populations matters more when a large amount of extracellular mediators is introduced rather than the resulting dynamics are observed over time. Standard deviations, shown as the shaded regions in the figures, are also higher than

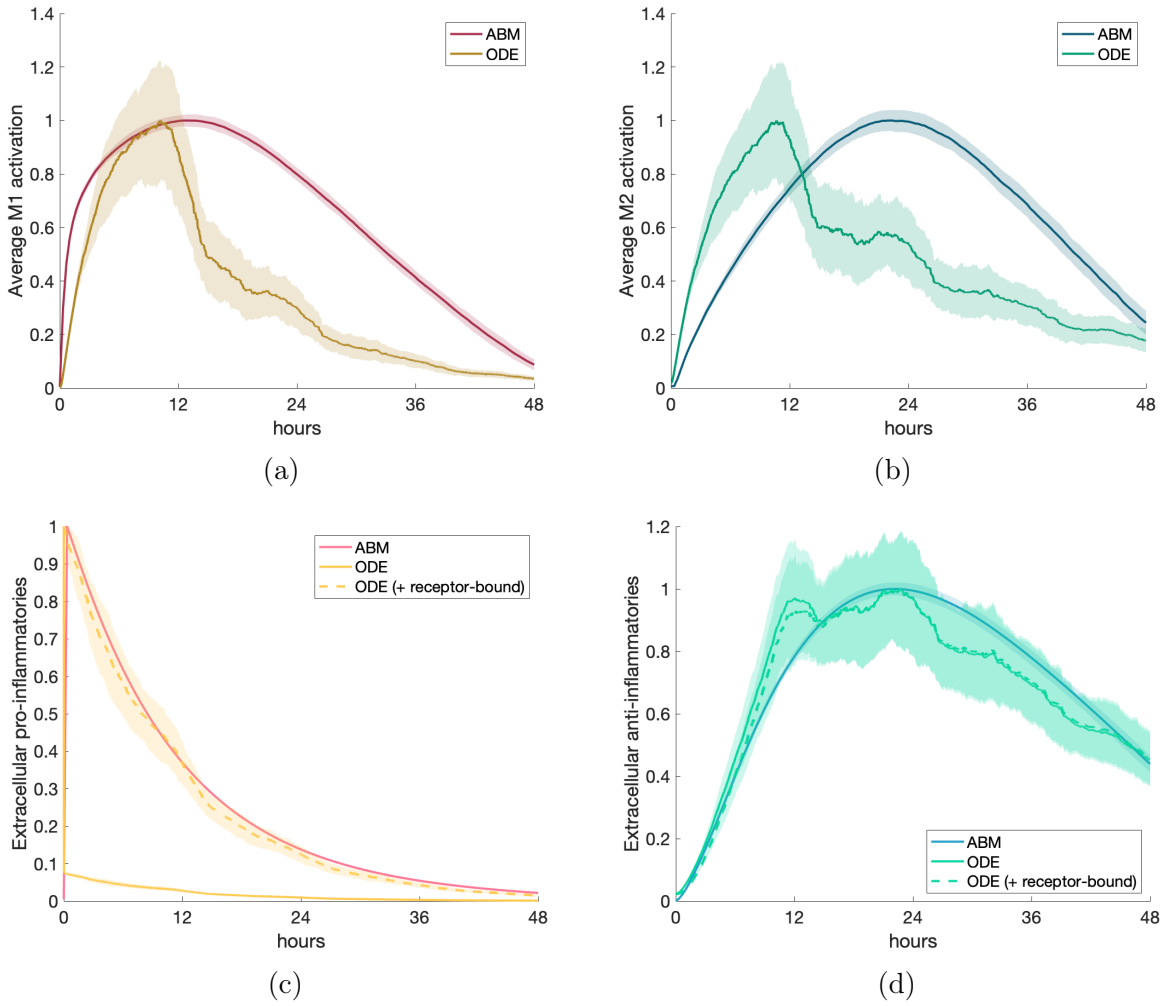


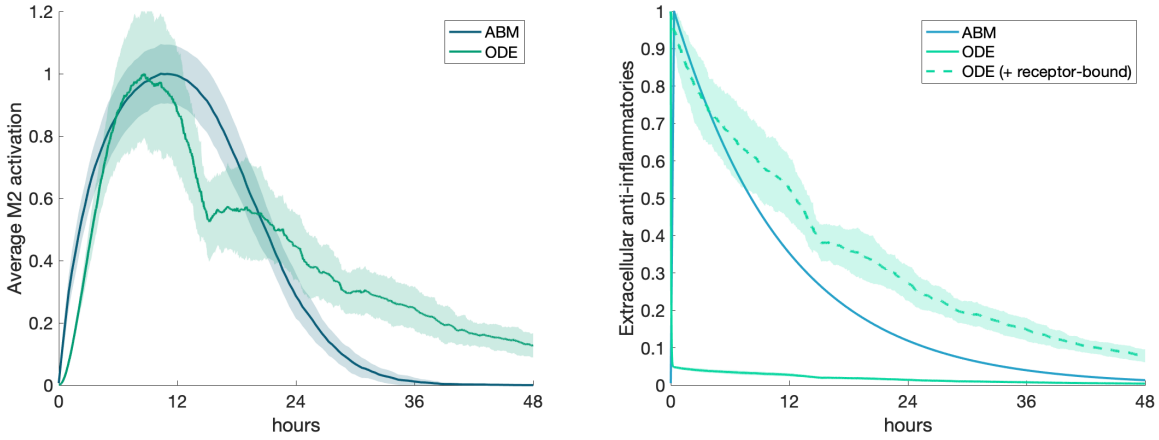
Figure 4.8: Scenario 2: M1/M2 response to model of multiple macrophages, activated by an initial amount of pro-inflammatory mediators. All transients are scaled individually by their maximums. (a) M1 activation, (b) M2 activation, (c) PIM, (d) AIM.

the single-macrophage simulations, since cell lifespan and recruitment provide additional randomness.

4.3.3 Scenario 3: multiple macrophages with anti-inflammatory stimulus

The same initial conditions were used for this scenario as in the previous one, except instead of a pro-inflammatory stimulus, an anti-inflammatory stimulus was introduced into the system. Results are shown in Figure 4.9. PIM and M1 activation were very small compared to AIM

and M2 activation, so we do not show their dynamics.



(a) (b)

Figure 4.9: Scenario 3: M1/M2 response to M1 macrophages activated by an initial amount of anti-inflammatory mediators. All transients are scaled individually by their maximums. (a) M2 activation, (b) AIM.

M2 activation in the two models are very similar, with the ODE showing a slightly longer tail after the peak of activation. This is paired with a slower decrease of IL-10 (AIM) when receptor-bound IL-10 is taken into account, similarly to PIM in the previous scenario. For a large anti-inflammatory stimulus, similar dynamics are observed between both models when the ODE transient includes receptor-bound IL-10.

4.3.4 Scenario 4: M1 macrophages with anti-inflammatory stimulus

Next we examined what would happen to an M1 environment when an anti-inflammatory stimulus is introduced into the system. We first needed to determine what this M1 environment would look like as initial conditions that could be used to begin the simulation.

For the ODE, we set all ten macrophages to an M1 phenotype based on the maximum activation that occurs in the calibrating experiment. This maximum occurs around hour 13, so we used the variable values at this time as the initial conditions for all macrophages. We then added a high concentration of IL-10 (the same amount as in Scenario 3) and ran the simulation.

For the ABM, M1 macrophages are defined as having $M1_{act} > 0.5$ and produce pro- and anti-inflammatory mediators proportional to their activation. To account for recruitment, the equivalent of which in the ODE model is turnover to naive initial conditions, we introduced into the system the number of M1 macrophages at the time when M1 activation was at its highest in Scenario 2. We find that this occurred roughly at hour 12, when there were 205 macrophages. We used this number of M1 macrophages as the initial conditions, along with the same amount of anti-inflammatory mediators as in Scenario 3. We performed two simulations to account for receptor-bound TNFα - in the first simulation, we started without any extracellular TNFα. Second, we considered no TNFα to also include no receptor-bound TNFα. Figure 4.10 shows the results for average activation and extracellular mediators.

M2 activation is similar, with the tail of M2 activation and AIM slightly longer in the ODE than the ABM. AIM have a similar response as in the previous scenario, such that with a large anti-inflammatory stimulus, including receptor-bound IL-10 in the AIM improve the ODE model's similarity to the ABM dynamics. Similarly, including receptor-bound TNFα in the total PIM matches ABM dynamics better, though in this case PIM production increases at a slightly higher rate in the ABM. Also, M1 activation shows a small rebound before it decreases to zero. Since AIM do not stimulate the pro-inflammatory signaling pathway but rather inhibit it, this rebound may be due to residual NFκB and TNFα in the cytosol and nucleus of the M1 macrophages, taking some time to make its way downstream before being used to produce a small amount of extracellular TNFα.

4.3.5 Scenario 5: half M1 and half M2

We then observed the results of initializing the models to a state of high activation such that half of the macrophages present were activated to an M1 phenotype and half were M2.

For the ODEs, we used the same initial conditions for M1 macrophages as in Scenario 4, and used a similar method to obtain initial conditions for M2. Hour 16 in Scenario 1 was the time around which peak M2 activation occurs. Five macrophages had M1 initial conditions

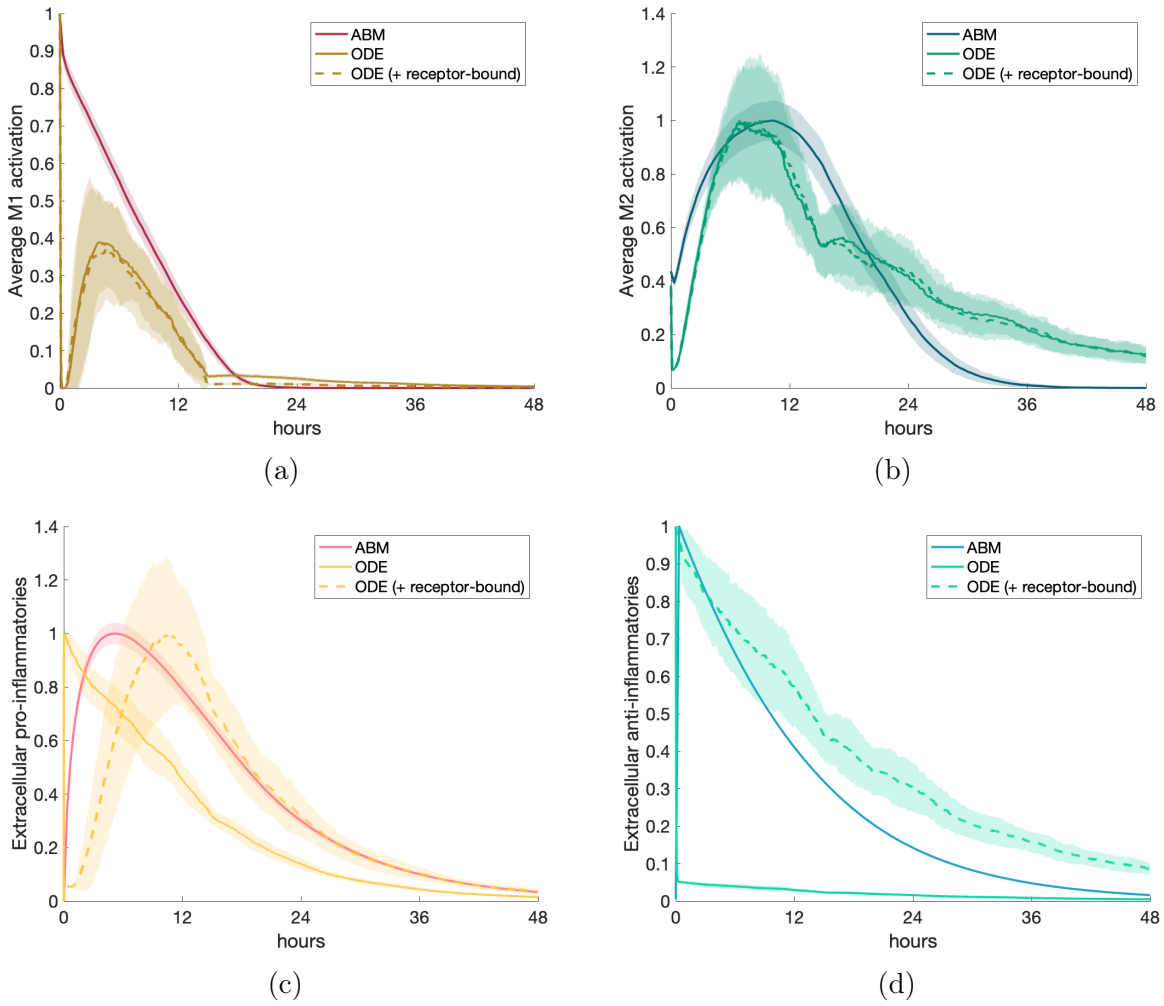


Figure 4.10: Scenario 4: M1/M2 response to anti-inflammatory stimulus introduced into an M1-polarized system. Transients are scaled individually by their maximums. (a) M1 activation, (b) M2 activation, (c) PIM, (d) AIM.

and the other five had M2 initial conditions.

For the ABM, we used a total of 200 macrophages to represent a state of high activation, similar to the maximum amount of macrophages in Scenario 4. Half were defined as M1 and half as M2. Figure 4.11 shows the results for M1 and M2 activation and for pro- and anti-inflammatory mediators. When receptor-bound mediators were not taken into account, only extracellular TNFα and IL-10 were set to zero at the beginning of the simulation. When receptor-bound mediators were considered part of the overall TNFα and IL-10 concentrations, they were also set to zero.

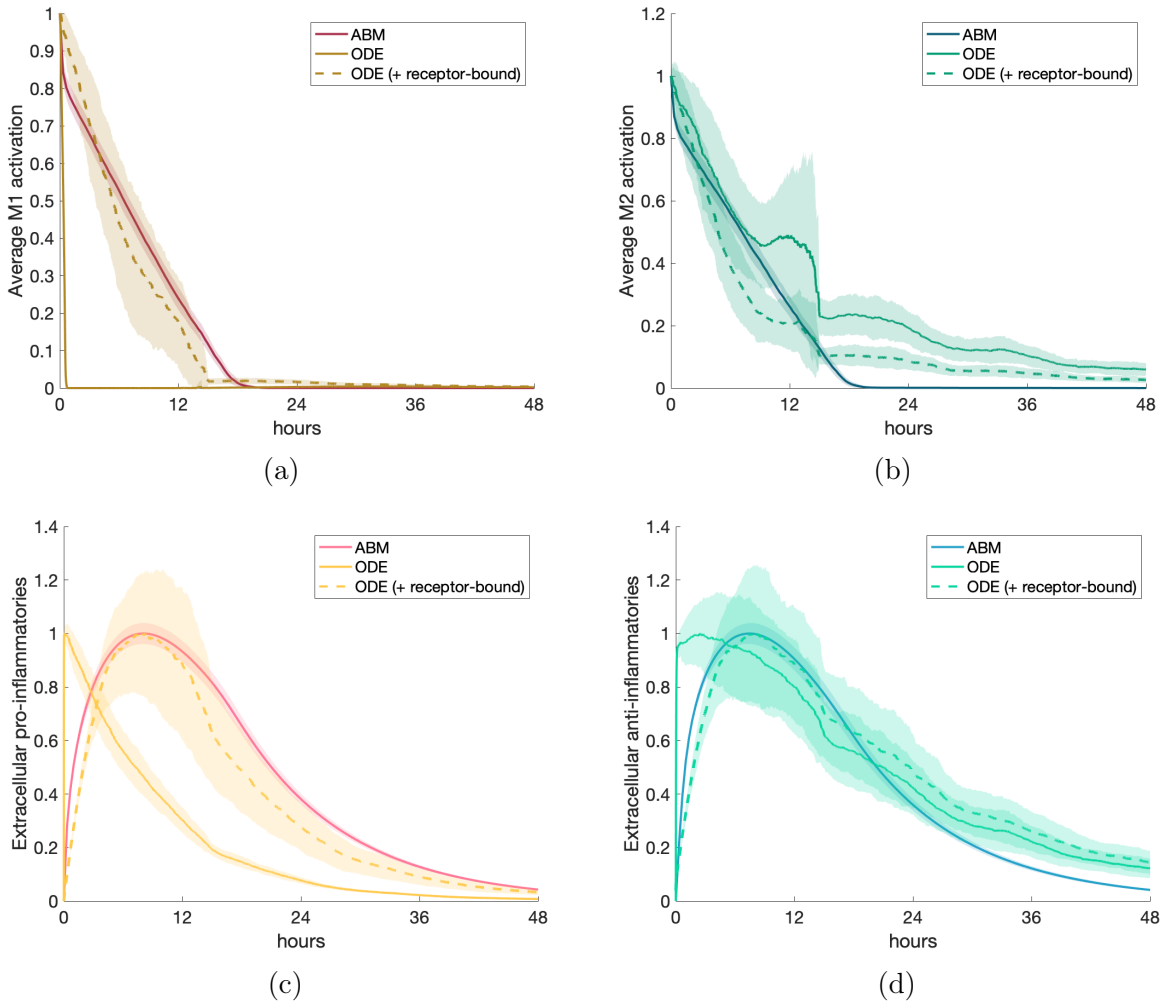


Figure 4.11: Scenario 5: M1/M2 response to a state of activation in which half of the macrophages present are M1 macrophages and half are M2. Transients are scaled individually by their maximums. (a) M1 activation, (b) M2 activation, (c) PIM, (d) AIM.

The ODE model results that included receptor-bound mediators in the total were more similar to ABM results, reflected in all four panels of Figure 4.11. M1 and M2 activation decay at similar rates due to low production of mediators, and the maximum PIM and AIM show similar behavior as well. The ODE model has consistently shown a longer tail in the overall anti-inflammatory response, both in AIM and M2 activation.

4.3.6 Scenario 6: PIM activation with wash and anti-inflammatory stimulus

It is common in experimental setups to perform a wash, where cells are treated with a stimulus, then "washed" with a solution to remove external mediators [136]. We replicated this experiment by beginning with the same initial conditions as in Scenario 2: 10 naive macrophages and a pro-inflammatory stimulus. Then at hour 12, the cells, at whatever state they were in at that time, were "washed" such that PIM and AIM were set to zero and a high amount of AIM was added (same as initial amount in Scenario 3). In the case of considering receptor-bound mediators, receptor-bound TNFα was also set to zero at hour 12. Results are shown in Figure 4.12. Since the times at which M1 and M2 activation is affected most by the wash is different for the ABM versus the ODE model, we compare experiments to examine how they differ from a control, where there is no wash and no AIM added at 12 hours. Therefore, we show four cases, all of which are initialized with PIM: 1) PIM with no later intervention, 2) no wash, AIM added at 12 hours, 3) wash with AIM added, 4) wash with no AIM added. In the future, experiments could be performed with data collected when the models' dynamics differ significantly in order to select which model best replicates the experimental results.

Figure 4.12 shows that M1 and M2 activation in the ABM responds similarly regardless of the experiment, whereas they have more distinct results in the ODE simulations. The ODE model has a more immediate response to the AIM than the wash, shown in the sharp changes at hour 12 for the blue and yellow curves in panels (b) and (d). On the other hand, activation in the ABM does not show these sharp changes; rather, they are more gradual even though large jumps are reflected in the PIM and AIM dynamics. Incorporating the receptor-bound mediators into the extracellular AIM and PIM in the ODE simulations shows nearly an exact match with the ABM results in panels (e) through (h). Furthermore, in both model types, M1 activation generally peaks before M2 activation. One noticeable difference is that with the wash, no AIM experiment, more time is needed in the ABM to

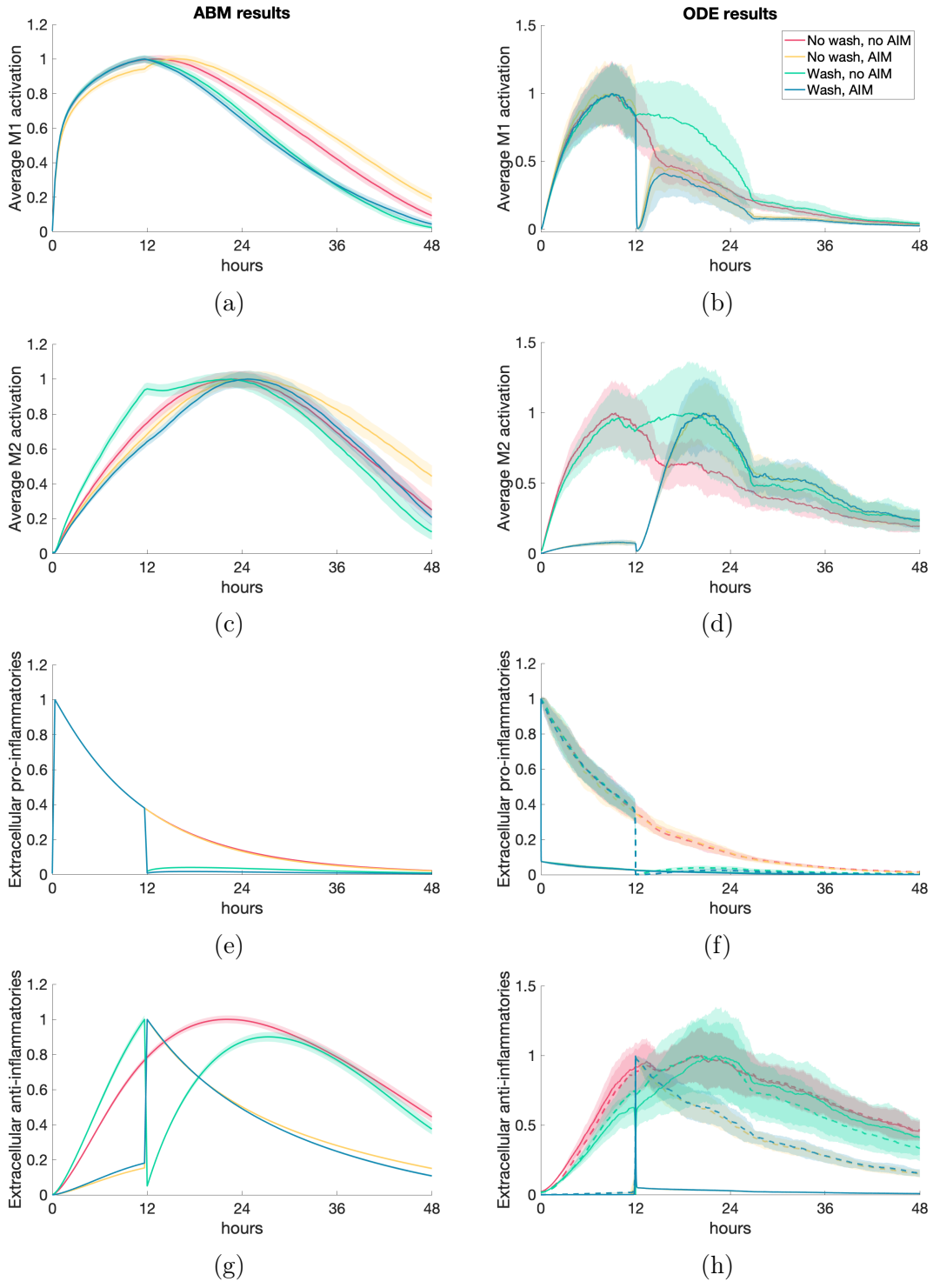


Figure 4.12: Scenario 6: M1/M2 response to an initial pro-inflammatory stimulus and either wash or no wash, with AIM added or not added at hour 12. Transients are scaled individually by their maximums. Column 1: ABM results. Column 2: ODE results. (a, b) M1 activation, (c, d) M2 activation, (e, f) PIM, (g, h) AIM.

return to its original levels whereas the ODE model shows a faster rebound. Examining these four scenarios allowed us to observe how the models respond to different variations of stimuli and pinpoint the sensitivity of both models to these stimuli. It could also aid in selecting the best way to create an *in silico* representation of an experiment such as a wash.

4.4 Discussion

With still much unknown about M1-M2 polarization and the important role it plays in the pathogenesis of many diseases [11], our modeling approaches and scenarios contribute to the body of knowledge surrounding macrophage polarization by providing a comparison of *in silico* platforms to test hypotheses and highlight mechanisms that may be necessary or unnecessary to include in future models.

By using the same basic principles of M1/M2 activation, interaction with mediators, and cell lifespan, our two distinctly different models provided surprisingly similar results after tuning to a common calibrating experiment. In particular, peak times and overall shapes of the transients were similar in most cases. Whereas our ODE model accounted for relatively detailed subcellular signaling, where each term represented a different interaction within the cell as well as with extracellular mediators, our ABM simplified the interactions to reflect similar roles of M1/M2 activation without the detail of individual mechanisms and interactions. Rather, only M1/M2 activation and mediators were measured in the model.

A common difference between models was a longer tail of M2 activation and AIM activity across several scenarios. This was also seen in the calibrating scenario, where M2 activation decreases more quickly in the ABM. Future work could include finer tuning of the parameters to better align the model results.

Another thread throughout this chapter is the consideration of receptor-bound TNFα and IL-10 in the ODE model. In most scenarios, especially those with high amounts of one mediator (Scenarios 2-4) or both (Scenarios 5-6), incorporating receptor-bound mediators

into the overall concentration of mediators improved similarity to the ABM results. Though this disparity was initially unexpected, it was not surprising since the ABM does not explicitly model receptors such that extracellular mediators are not removed from the population when they interact with a macrophage. Due to the significant difference when taking into account receptors versus not taking them into account, future changes to the ABM may involve accounting for receptor-bound mediators by explicitly including receptors or PIM and AIM in the extracellular population could be decreased when they come into contact with a macrophage, representing binding to receptors.

We also wanted to examine the differences that incorporating space (ABM) or detailed subcellular signaling (ODEs) would make in the resulting dynamics. A notable difference between the two models is seen in Figure 4.10(a), where residual amounts of M1-related variables such as intracellular forms of NFκB and TNFα resulted in a small downstream bump in M1 activation in the ODE, whereas the ABM, which does not account for these variables, showed a more gradual, constant decrease of M1 activation to zero. Another significant difference was observed in the "wash" experience in Scenario 6, where the ODE model had a greater sensitivity to immediate changes in PIM and AIM than the ABM. In the ABM, rules of macrophage activation are defined such that activation decreases gradually when a stimulus is not present, whereas in the ODE model the explicit transcription of mRNA responds directly and more immediately to a lack of extracellular mediators. Interestingly, this discrepancy did not significantly affect the other scenarios. This is an area of future investigation, especially if these models could be validated with experimental data. Overall, the incorporation of multi-step subcellular signaling was not very important since the ABM did not include subcellular signaling and we obtained similar dynamics from both models.

We did not observe significant differences regarding the spatial dynamics of the ABM versus the well-mixed assumption of the ODE model, although this was not a focus of our analysis. It has been shown in previous ABMs involving macrophages, such as modeling granuloma formation in tuberculosis [77], that incorporating the ability of macrophages to

interact on a spatial level and gather together is important to the immune response. Future simulations and scenarios could involve putting initial amounts of PIM and AIM on different areas within the grid or in different patterns to observe more carefully how space plays a role in M1/M2 activation.

Based on our findings from comparing the two models, we recommend a focus on the main interactions of extracellular mediators and macrophages, where M1/M2 polarization can occur on a continuous spectrum, reflecting the current knowledge and modeling practices of macrophage activation [78, 88, 149]. Important feedback loops in the pro- and anti-inflammatory phases of the immune response are: the positive feedback loop of M1 activation, upregulation of M2 via M1, and the negative feedback loop in which M2 decreases both M1 and itself. Initially, our ABM did not include SOCS, a family of intracellular proteins produced by the IL-10 pathway to regulate itself. Without this regulatory feedback loop, M2 activation and AIM did not decrease back to its initial state, but when we added SOCS to the ABM, we obtained the expected dynamics such that the calibrating experiment results of the ABM were similar to the ODE model, which did include SOCS. Whether these interactions and feedback loops are modeled explicitly through signaling pathways or through general rules was less important for our purposes, as our results from the two approaches were similar, as long as they were included in some manner.

Future work necessary to confirm our hypotheses via the scenarios described above is to fit both models, especially the calibrating experiment, to additional data. The only data used so far was an initial parameter estimation of the ODE model parameters based on LPS-induced dynamics. More sophisticated parameter estimation methods, such as obtaining correlations between parameters and a sensitivity analysis, would be useful due to the large number of parameters in the model. Furthermore, currently both models are meant to represent the immune response to a general insult. These models can be adapted to incorporate the key players and mechanisms involved in specific injuries such as bacterial or viral infections, mechanical ventilation, or smoking/COPD.

Chapter 5

Discussion

5.1 Conclusion

Infections and other insults associated with the lungs are still some of the top causes of death worldwide [2]. Despite decades of research surrounding lung inflammation and damage, there is still much that is unknown about the mechanisms of inflammation and repair in the lungs, especially relating to the M1/M2 spectrum of activation. In the context of VILI, the need to test new ventilation strategies that mitigate VILI before becoming standard practice arises from the combination of patient-specific needs and the many configurations of the ventilator itself [118]. Another example of the need for new treatments can be seen in treatment of the respiratory infection tuberculosis, for which there is currently not a comprehensive vaccine and antibiotic resistance is a growing problem [58]. *In silico* modeling is a useful tool to simulate these complex interactions, test hypotheses, and make predictions. Furthermore, a variety of methods, including statistical and machine learning techniques, can aid in analyzing our models in the absence of experimental data. In this work, we utilized various types of *in silico* modeling with standard analysis methods and ones developed on our own to contribute to a greater understanding of lung inflammation.

To understand VILI and identify the key mechanisms driving outcome, we developed

an ODE model that includes an epithelial subsystem and accounts for M1/M2 polarization. Through LHS, we generated a collection of parameter sets that provided a wide variety of dynamics. Using hypothesis testing, a random forest algorithm, and the variance-based sensitivity analysis method eFAST, we identified the parameters and other predictors that contributed most to changes in overall lung health. We found that the key parameters corresponded to mechanisms of epithelial repair and M1 activation. We then hypothesized interventions based on these mechanisms and modulated epithelial damage in a case study.

We explored the usefulness of the collection of parameter sets produced from LHS and their corresponding model-generated dynamics. By creating a large synthetic data set from these transients, we were able to develop an algorithm based on relative error between the collection and each synthetic data set to predict whether the individual's condition would worsen after ventilation and a period of recovery. We also extended this algorithm to choose a next sample time that would provide the most information to determine outcome, and employed the parameter values themselves to supplement prediction in some cases. There are many different ways our algorithms can be tuned based on the needs of the user, such as the availability of different kinds of data and the thresholds for determining outcome. Thus far, our prediction processes compares fairly well to current classification methods using the synthetic data as a proof of concept. However, our methods should be calibrated to experimental data to be truly useful in a real-world setting and improve the accuracy of our predictions.

As previously stated, a coordinated M1/M2 response is necessary to correctly respond to and resolve damage in the lung. We built upon previous models to develop a system of ODEs that represents M1 and M2 subcellular signaling pathways within a macrophage. We also applied the same principles in an ABM, adding a spatio-temporal component to the macrophage dynamics. Both of these models represented a generic inflammatory response in the lungs. We developed these models to include cell lifespan and M1/M2 dynamics on both an individual-cell level and a tissue-scale level. In this way, macrophages interact with

their environment. We compared these different modeling approaches in various scenarios and found that including the details of the subcellular signaling pathway were not necessary in capturing the nonlinear dynamics of the M1/M2 response as long as the feedback loops regulating this immune response were incorporated into the model. We also identified how differences between the two models, such as explicitly including receptors, can affect output and should be considered when modeling.

Overall, this work contributes to the current body of knowledge surrounding mathematical modeling of lung inflammation, and in particular the role of the spectrum of macrophage activation. Due to the lack of data currently available, sophisticated computational methods were necessary to gain insight into the immune response. Through these methods we were able to identify important mechanisms, hypothesize and test interventions, and provide suggestions for future modeling efforts.

5.2 Future directions

A current limitation of our work is a lack of data, especially M1/M2 data in response to VILI. Once this is available, it will be possible to perform formal parameter estimation methods, calculate correlations between parameters, and obtain sensitivity analysis results. Furthermore, a true cohort could be established to aid in future predictions of outcome. With a bank of previous data, the algorithms we developed in Chapter 3 could be validated and used in real time for experimental design and in clinical settings to recommend sample times and determine whether additional intervention is needed, both during and after ventilation. The calibrating experiment and scenarios in Chapter 4 could also be fit to data. This would shed more light on M1/M2 dynamics on an individual cell level and aid in understanding the collective behavior of the tightly regulated interactions and roles of these cells.

All of these modeling approaches can be adapted or coupled with other models to replicate the dynamics of specific diseases. We discussed in Chapter 2 that although we focused our

modeling on the immune system dynamics that resulted only from ventilation, there are a number of insults and comorbidities that induce the need for ventilation in the first place. Since the algorithms developed in Chapter 3 are essentially classification methods based on an ODE model and collection of parameter sets, these same algorithms can be applied to other models to obtain predictions of different outcomes. Furthermore, for our general inflammation models in Chapter 4, specific mechanisms and interactions can be included to reproduce infection, particle inhalation through smoking, or other initiators of the immune response.

The methods used in this work, such as LHS, hypothesis testing, and eFAST, were appropriate for our purposes but alternative sampling and parameter sensitivity analysis methods exist. For example, Halton sequences are quasi-random but often cover parameter space more evenly [142], and partial rank correlation coefficients and the Sobol method are alternatives to eFAST [76, 114]. Additionally, incorporating other statistical methods could be useful. When performing hypothesis tests, interactions between parameters can be taken into account [13]. Furthermore, in Chapter 3, there may be other combinations of parameters not included in the collection that best fit a data set. Instead of the RES algorithm, LHS could be used as an experimental design to fill the parameter space. Then, using a linear model or Gaussian process model, parameter values could be selected to optimize the fit to data [55]. Future work could explore whether these and other analysis methods provide more information or aid in more accurate predictions.

Additional directions for future work are related to the properties of the models themselves. In Chapters 2 and 3, we examined how outcomes differed between initial conditions and states after ventilation plus a recovery period. After exploratory simulations, we observed that it is possible for many of the cases that worsen after 200 hours in comparison to their pre-ventilation state to eventually return to their initial conditions, but only after several thousand hours, which may be clinically unrealistic. We performed a bifurcation analysis for the epithelial subsystem, showing stable and unstable steady-states at zero or

nonzero points for E_h and E_d, but what about the stability of the larger system? Understanding the stability of the model as a whole may provide additional insight into the reasons why some individuals can recovery quickly after ventilation while others do not; this is of particular interest in the context of inflammaging in VILI, where older individuals do not recover as well from ventilation as younger individuals.

Furthermore, the ABM incorporates recruitment of macrophages, diffusion of extracellular mediators, and random movement of macrophages. However, the spatial dynamics of this model were not fully explored in Chapter 4, and could be useful in understanding how different spatial configurations of the immune system components might affect the M1/M2 response. Future simulations could include placing macrophages and extracellular mediators in different locations on the grid and examining the resulting patterns.

In conclusion, through mathematical modeling of lung inflammation and incorporating macrophage polarization, we have been able to develop a better understanding of these complex interactions and recommend targets for intervention. In the future, we will be able to validate our models and strengthen results through fitting to data, which can be guided by our modeling efforts described in this work.

Appendix A

Chapter 2 supplementary material

A.1 Epithelial subsystem

We began with a small three-dimensional system of differential equations of epithelial cell dynamics, shown in Eqs (A.1)-(A.3). We then performed a bifurcation analysis to gain an initial understanding of the effects of the immune response on the epithelium. In this section, we examine steady-states that arise from mechanisms specific to VILI.

E_h is the proportion of the local space filled by healthy cells, E_d is the proportion of the local space filled by damaged cells, and E_e represents dead cells or empty "space" that can be replaced/filled with healthy cells. We define this "local space" to be a simplified approximation of the entire alveolar space. Each term represents a biological event explained by the brackets above the term. This first model includes only the baseline abilities of epithelial cells to proliferate and repair themselves in the presence of sustained damage. We do not explicitly model proliferating and non-proliferating cells; the parameter p_e is modulated to reflect the general mechanism by which neighboring epithelial cells renew surrounding "space" (tracked by E_e).

$$\frac{dE_h}{dt} = \overbrace{p_e(E_h + E_d)(E_e)}^{\text{Proliferation}} + \overbrace{rE_d}^{\text{Repair}} - \overbrace{sE_h}^{\text{Damage}} \tag{A.1}$$

$$\frac{dE_d}{dt} = -\overbrace{rE_d}^{\text{Repair}} - \overbrace{bE_d}^{\text{Death}} + \overbrace{sE_h}^{\text{Damage}} \tag{A.2}$$

$$\frac{dE_e}{dt} = -\overbrace{p_e(E_h + E_d)(E_e)}^{\text{Proliferation}} + \overbrace{bE_d}^{\text{Death}} \tag{A.3}$$

Ventilator-induced injury is represented by the rate s, and causes healthy epithelial cells to become damaged. This general term covers over-distension for any mode of ventilation. Some damaged cells, depending on the severity of damage, have the ability to repair themselves, returning from the E_d state back to E_h, represented by a baseline repair rate r [25]. Damaged cells may also decay naturally at a rate b.

The first terms in Eq (A.1) for E_h, and Eq (A.3) for E_e, account for proliferation of the healthy and damaged cells into empty space. Note that total local space is conserved: $E_e + E_h + E_d = 1$. Therefore, we can define $E_e = 1 - (E_h + E_d)$ and rewrite this term, where it becomes the standard logistic growth with a carrying capacity of 1, associated with 100% of space being filled. Thus, E_h, E_d, and E_e are dimensionless and we determine time to be in hours. Eliminating E_e gives rise to a two-dimensional system, Eqs (A.4)-(A.5).

$$\frac{dE_h}{dt} = \overbrace{p_e(E_h + E_d)(1 - (E_h + E_d))}^{\text{Proliferation}} + \overbrace{rE_d}^{\text{Repair}} - \overbrace{sE_h}^{\text{Damage from ventilator}} \tag{A.4}$$

$$\frac{dE_d}{dt} = -\overbrace{rE_d}^{\text{Repair}} - \overbrace{bE_d}^{\text{Death}} + \overbrace{sE_h}^{\text{Damage from ventilator}} \tag{A.5}$$

Stability analysis revealed that in the absence of ventilator-induced damage ($s = 0$) and with all positive parameters, $(0, 0)$ is a saddle node and $(0, 1)$ is a stable equilibrium with eigenvalues $\lambda_1 = -r - b$ and $\lambda_2 = -p_e$. Given a nonzero initial condition for damaged cells,

the epithelial cells subsystem will resolve to the fully repaired fixed point $(0, 1)$.

In the presence of sustained ventilator-induced damage ($s > 0$), the E_d nullcline switches from a vertical line to a line with slope $(r+b)/s$. The second equilibrium point changes from $(0, 1)$ to

$$(E_d^*, E_h^*) = \left(\frac{s^2(p_e - b) + p_e s(b + r)}{p_e(b^2 + r^2 + s^2 + 2br + 2bs + 2rs)}, \right.$$
$$\left. \frac{(r + b)[s(p_e - b) + p_e(b + r)]}{p(b^2 + r^2 + s^2 + 2br + 2bs + 2rs)} \right)$$

Therefore in the presence of sustained damage, there no longer exists an equilibrium associated with full recovery.

In this section, we focused on the existence of bifurcations rather than the specific values and time scales at which they occur; therefore the parameter values and ranges were chosen to highlight the presence of these bifurcations. Further consideration for parameter ranges is discussed in Sections 2 and 3. Based on initial exploration of the parameter space, E_h and E_d seemed most responsive to changes in p_e. There may be other bifurcation parameters; we chose p_e as examples (Figures A.1).

A bifurcation diagram for p_e, shown in Fig A.1, has one transcritical bifurcation at $p_e^* = 0.497$. The bifurcation diagrams in this manuscript were created using XPPAUT [38] with code included in Section A.3. In this figure, we show the proportion of space occupied by healthy epithelial cells as a percentage, which is E_h multiplied by 100. The second equilibrium for values of p_e below the bifurcation is not included in the diagram, since it is non-biological (negative E_h). For small values of p_e, the ability of healthy cells to proliferate and replace dead cells was insufficient and damage caused both healthy and damaged cells to approach 0%. On the other hand, for values of p_e larger than p_e^*, the system approached the stable nonzero equilibrium (E_d^*, E_h^*), which was closer to $(0, 1)$ for higher values of p_e even in the presence of sustained damage.

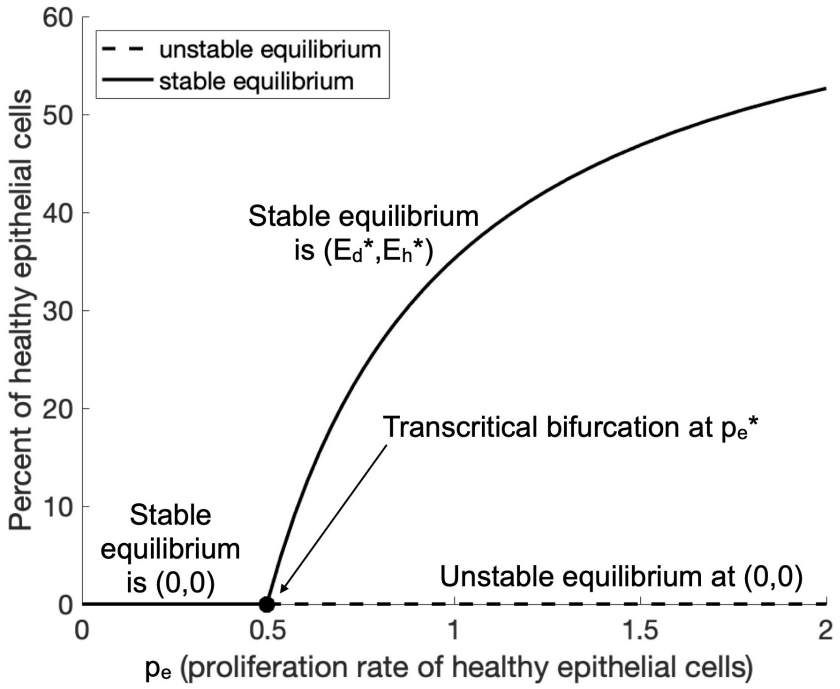


Figure A.1: **The epithelial subsystem generated a transcritical bifurcation for the parameter p_e.** Bifurcation diagram for the proliferation parameter p_e for the epithelial system with VILI and no immune response. Other parameters were set to $r = 2.6$, $s = 0.22$, and $b = 0.74$. The unstable equilibrium below $p_e < p_e^* = 0.497$ is not included in the figure, since it is not biologically relevant.

A.1.1 Fixed immune response

Next we examinde the roles of immune cells, especially neutrophils and macrophages, by adding several terms to Eqs (A.1) and (A.2). We first focused on dynamics with a fixed immune response, because when we worked with the full model (Section 2.2), we only considered parameter sets that gave rise to steady-state solutions in the absence of ventilator-induced damage. Therefore, we decided to start our model development by analyzing E_h and E_d with immune cells as parameters before including their full dynamics. The modifications are shown in Eqs (A.6) and (A.7).

$$\frac{dE_h}{dt} = \overbrace{p_e(E_h+E_d)(1-(E_h+E_d))}^{\text{Proliferation}} + \overbrace{rE_d}^{\text{Repair}} - \overbrace{sE_h}^{\substack{\text{Damage from}\\ \text{ventilator}}} - \overbrace{nE_h}^{\substack{\text{Damage via}\\ \text{M1 \& neutrophils}}} \tag{A.6}$$

$$\frac{dE_d}{dt} = -\overbrace{rE_d}^{\text{Repair}} - \overbrace{bE_d}^{\text{Death}} + \overbrace{sE_h}^{\substack{\text{Damage from}\\ \text{ventilator}}} + \overbrace{i_mE_h}^{\substack{\text{Damage via}\\ \text{M1 \& neutrophils}}} - \overbrace{mE_d}^{\substack{\text{Phagocytosis}\\ \text{of damaged}\\ \text{cells by M1}}} \tag{A.7}$$

The physical presence of immune cells, especially first-responder neutrophils, causes small-scale collateral damage as they clear debris [92] and can be especially deleterious if the response is overzealous [44]. This biological event is modeled as the last term in Eq (A.6) with cells switching from a healthy to a damaged state at the rate i_m. M1 macrophages aid in the clearance of damaged cells to make room for replacement by new, healthy cells through subcellular signaling and phagocytosis [4, 41]. The last term in Eq (A.7) represents this loss of damaged cells.

The stability analysis was similar to that from the model without the immune response, with additional parameters $m,\ i_m$ that could shift steepness of the nullcline or the speed at which the system approached or diverged from an equilibrium. The parameter p_e once again

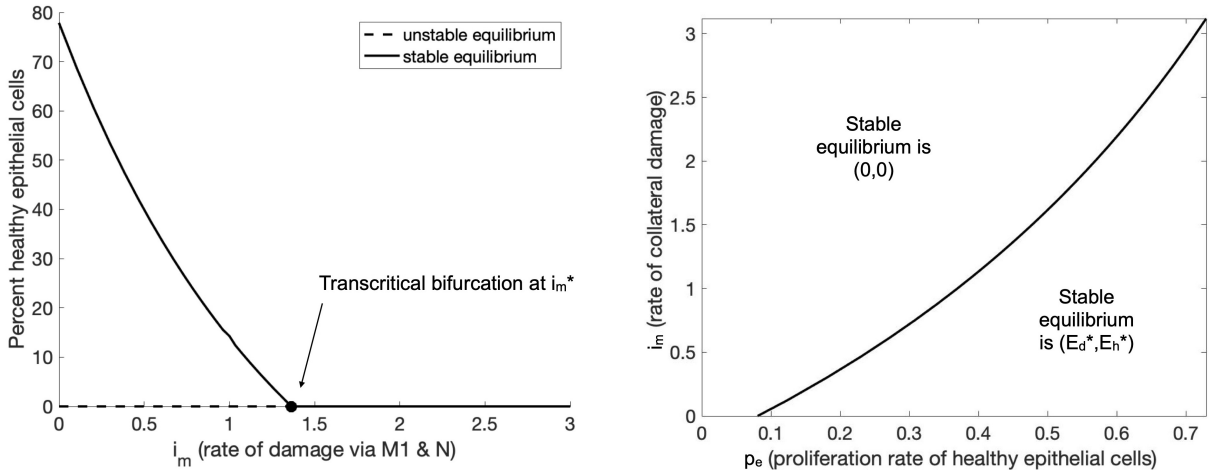
played an important role in the stability of the two critical points, $(0,0)$ and

$$(E_d^*, E_h^*) = \left(\frac{(i_m+s)[(i_m+s)(p_e-b-m)+p_e(b+m+i_m)]}{p_e(b+m+i_m+r+s)^2}, \right.$$
$$\left. \frac{(b+m+r)[(i_m+s)(p_e-b-m)+p_e(b+m+i_m)]}{p_e(b+m+i_m+r+s)^2} \right)$$

There was a transcritical bifurcation when the value of p_e was varied; given its similarity to Fig A.1, it is not shown here. For the same parameter values as in Fig A.1 ($r = 2.6, s = 0.22, b = 0.74$) with $m = 0.92$ and $i_m = 1.6$ added, we obtained the same $p_e^* = 0.497$. The main difference between these models is that the transcritical bifurcation point p_e^* may be lower because of the damage resulting from macrophages and neutrophils, represented by m and i_m. The rate of proliferation of healthy cells may need to be higher to counteract these effects.

The bifurcation diagram for scaled E_h versus i_m also had a transcritical bifurcation (see Fig A.2a). For sufficiently low values of i_m, the nonzero critical point was stable, but for values above $i_m^* = 1.364$, $(0,0)$ was the stable equilibrium. Additionally, the two-parameter stability diagram shows a curve which separates the p_e/i_m-space into two stability regimes (see Fig A.2b). For high enough values of i_m and low enough values of p_e, the system went to zero for both variables. Biologically, this corresponds to a situation in which the ability of epithelial cells to proliferate is low and there are high levels of immune cells. On the other hand, with low levels of immune cells and a higher proliferation rate, the system limits to the nonzero equilibrium. It should be noted that for a large enough p_e, it would take an extremely high value of i_m to overpower proliferation and make $(0,0)$ the stable critical point. In the full system, the initial conditions for our simulations have similar properties to the type of steady state in the non-zero stable equilibrium region of Fig A.2b. Varying levels of baseline inflammation exist given differences in patients' age and past medical history.

These simple models provide a framework for the dynamics of the epithelium in response to damage and an introductory look into the influence of the immune response. However,



(a) (b)

Figure A.2: **Variations on the epithelial subsystem revealed a transcritical bifurcation and two-parameter bifurcation.** (a) Bifurcation diagram for epithelial subsystem when varying n. Other parameter values were set to $r = 2.6,\ p_e = 0.45,\ s = 0.22,\ b = 0.74,\ i_m = 1.6,\ m = 0.92$. (b) Two-parameter plot showing values of p_e and i_m which caused the subsystem to have either a zero or nonzero stable equilibrium.

there are many more complex, nonlinear interactions and events involved in VILI which we explored in the full model in Section 2.2.

A.2 Analysis results for different sampling techniques

The next pages show the results of our analysis methods for LHS-generated parameter sets using log-uniform and log-normal distributions, with three different exclusion criteria based on E_e initial condition. We found that our methods did not differ significantly among these sets.

Cohort Breakdown

A summary of the initial sates and outcomes and how they change, depending on the maximum initial amount of Ee allowed (exclusion group), time at which outcome is determined, and type of sampling distribution.

Numbers in parentheses are the number of sets that leave the state and enter the state at the end of the simulation for initial condition (IC) and ending state (ES), respectively.

	Log-uniform, 200h			Log-uniform, 2h	Log-normal
Initial condition criteria:	Ee(0)<75%	Ee(0)<50%	Ee(0)<25%	Ee(0)<50%	Ee(0)<50%
Total number of sets that reached steady-state:	24798	24432	23517	22432	33256
Healthy IC:	16833 (635)	16833 (635)	16833 (635)	16833 (14260)	21403 (37)
Health ES:	16198 (0)	16198 (0)	16198 (0)	2573 (0)	21373 (7)
Moderate damage IC:	5382 (572)	5382 (572)	4697 (265)	5382 (3387)	10892 (155)
Moderate damage ES:	5105 (295)	5104 (294)	4726 (294)	10116 (8121)	10771 (34)
Moderate damage IC:	2583 (1)	2217 (0)	1987 (0)	2217 (0)	961 (0)
Moderate damage ES:	3495 (913)	3130 (913)	2593 (606)	11743 (9526)	1112 (151)

Top Correlations

The parameters that have the highest correlation with parameters and other predictors, for each exclusion group and outcome classification.

	Log-uniform			Log-uniform, 2h	Log-normal
Criteria:	Ee(0)<75%	Ee(0)<50%	Ee(0)<25%	Ee(0)<50%	Ee(0)<50%
	kmne, Eh ratio 0.5h			kmne, Eh ratio 0.5h	kmne, Eh ratio 0.5h
Healthy	0.1	0.1	0.1	0.24	0.06
Moderate damage	0.66	0.66	0.66	0.43	0.47
Severe damage	0.55	0.87	0.86	0.73	0.67
	br, Eh ratio 0.5h			br, Eh ratio 0.5h	br, Eh ratio 0.5h
Healthy	0.29	0.29	0.29	-0.04	0.19
Moderate damage	0.42	0.42	0.43	0.27	0.23
Severe damage	0.05	0.18	0.22	0.32	0.22
	sm, max M2			sm, max M2	sm, max M2
Healthy	0.32	0.32	0.32	0.28	0.48
Moderate damage	0.31	0.31	0.31	0.32	0.51
Severe damage	0.4	0.29	0.3	0.32	0.52

Significance Testing

Parameters and other predictors that show a statistically significant difference (p-value<0.05) between all three outcome classifications, using Kruskal-Wallis and Wilcoxon tests.

	Log-uniform			Log-uniform, 2h	Log-normal
Criteria:	Ee(0)<75%	Ee(0)<50%	Ee(0)<25%	Ee(0)<50%	Ee(0)<50%
Significant predictors:	kmne	kmne	kmne	kep	kmne
	xmne	xmne	xmne	br	xmne
	M2% at 10h	M2% at 10h	br	Eh ratio at 0.5h	Eh ratio at 2h
	min M2	min M2	M2% at 10h	Eh ratio at 2h	mup
	M1 peak time	br	ken	Eh ratio at 6h	ken
	Eh ratio at 2h	Eh ratio at 2h	min M2	min M1	kep
	min M1	M1 peak time	min M1	ken	M1 peak ratio
	kep	kep	Eh ratio at 2h	sn	br
	br	min M1	M1 peak ratio	max M1	M1 peak time
	M1 peak ratio	M1 peak ratio	kep	min M1%	min M1%
	Eh ratio at 0.5h	Eh ratio at 0.5h	Eh ratio at 0.5h	kan	M2 peak time
	mup	mup	kem1	max M1%	kem1
	kem1	kem1	bp	kem1	sn
	kpe			M1 peak time	M2% at 10h
				kam1	ainf
				muna	
				max M2%	
				kn	
				mup	
				min M2%	
				ainf	
				sm	
				M2% at 10h	
				muab	
				kpm1	
				kmne	
				bp	
				knn	
				xm0ab	
				mum1b	

Random Forest Decision Tree

Ten highest average importance values, as determined by 1000 random forests.

	Log-uniform, 200h			Log-uniform, 2h	Log-normal
Criteria:	Ee(0)<75%	Ee(0)<50%	Ee(0)<25%	Ee(0)<50%	Ee(0)<50%
Top ten, in order:	kmne	kmne	kmne	Eh ratio at 2h	kmne
	Eh ratio at 6h	Eh ratio at 6h	Eh ratio at 6h	Eh ratio at 0.5h	xmne
	Eh ratio at 2h	xmne	xmne	Eh ratio at 6h	Eh ratio at 2h
	xmne	Eh ratio at 2h	Eh ratio at 2h	kmne	Eh ratio at 0.5h
	min M1	min M1	Eh ratio at 0.5h	br	Eh ratio at 6h
	Eh ratio at 0.5h	Eh ratio at 0.5h	min M1	kep	mup
	min M2	min M2	min M2	xmne	min M1
	M2% at 10h	M2% at 10h	M2% at 10h	min M1	ken
	br	br	br	ken	min M1%
	ken	ken	ken	sn	kep

A.3 Code: XPP file

The following is an .ode file that can be input into XPP to obtain the bifurcations described in Section A.1.

```
# VILI epithelial subsystem, Minucci et al


# the parameters:
p N=0, R=2.6, P=0.45, S=0.22, B=0.74, M=0
### with immune response:
### N=1.6, M=0.92


# the system:
h'=-N*h+R*d+P*(h+d)*(1-h-d)-S*h
d'=-M*d-R*d+N*h+S*h-B*d


# initial conditions:
h(0)=1
d(0)=0


done
```

A.4 Code: ODE model equations

The following MATLAB function contains the equations for the full model presented in Section 2.2.

```
1 function [dxdt] = model_equations(t,y,param)
2 % Equations for compartmental model of immune response to VILI
```

```
% Minucci et al. | 2021

vent_time=2;

% parameters

dp = param(1); % PIM diffusion
da = param(2); % AIM diffusion
dm0 = param(3); % M0 diffusion
dm1 = param(4); % M1 diffusion
dm2 = param(5); % M2 diffusion
xm0pb = param(6); % regulates differentiation of M0b by pb
xm0ab = param(7); % regulates differentiation of M0b by pb
xm0pd =param(8); % regulates recruitment of M0b by pb
xm0ad = param(9); % regulates recruitment of M0b by ab
xm1p = param(10); % regulates recruitment of M1b by pb
kpm1 = param(11); % production of p by M1
kpe = param(12); % production of p by damaged cells
kam1 = param(13); % production of a by M1
kam2 = param(14); % production of a by M2
xm0p = param(15); % regulates differentiation of M0 by p
xm0a = param(16); % regulates differentiation of M0 by a
km1m2 = param(17); % dummy variable (used for eFAST)
xer = param(18); % regulates repair of damaged cells by R
kep = param(19); % self-resolving repair mediated by p
km2r = param(20); % upregulation of M2 recruitment by R
km2a = param(21); % upregulation of M2 recruitment by AIM
```

```
xnup = param(22); % regulates activation of neutrophils by PIM
kpn = param(23); % production of PIM by neutrophils
kn = param(24); % migration of activated neutrophils to lung
kman = param(25); % upregulation of M1 switch by AN
kan = param(26); % neutrophils become apoptotic
knn = param(27); % neutrophils become necrotic
kanm1 = param(28); % phagocytosis by M1
kanm2 = param(29); % phagocytosis by M2
km0pb = param(30); % differentiation of M0b by pb
km0ab = param(31); % differentiation of M0b by ab
km0pd = param(32); % recruitment of M0b by pb
km0ad = param(33); % recruitment of M0b by ab
km1p = param(34); % recruitment of M1b by pb
km0p = param(35); % differentiation of M0 by p
km0a = param(36); % differentiation of M0 by a
krm2 = param(37); % production of R by M2
ker = param(38); % repair of damaged cells by R
kem1 = param(39); % further damage by M1
knup = param(40); % activation of neutrophils by PIM
abinf = param(41); % maximum amount of ab for inhibition
ainf = param(42); % maximum amount of a for inhibition
mupb = param(43); % decay rate of pb
muab = param(44); % decay rate of ab
mum0b = param(45); % decay rate of M0b
mum1b = param(46); % decay rate of M1b
mum2b = param(47); % decay rate of M2b
mup = param(48); % decay rate of p
```

```
mua = param(49); % decay rate of a
mum0 = param(50); % decay rate of M0
mum1 = param(51); % decay rate of M1
mum2 = param(52); % decay rate of M2
muR = param(53); % decay rate of R
munu = param(54); % decay rate of Nu
muna = param(55); % decay rate of Na
sm = param(56); % source of M0b
sn = param(57); % source of Nu
bd = param(58); % baseline decay of damaged cells
br = param(59); % baseline repair of damaged cells
bp = param(60); % baseline self-resolving repair of epithelial cells
sd=param(61); % constant damage rate
kmne=param(62); % damage to healthy epithelial cells due to M1 & N
sp=param(63); % source of PIM
sa=param(64); % source of AIM
xmne=param(65); % regulates damage to healthy epithelial cells due to M1 & N
xm2r=param(66); % regulates effectiveness of M2b recruitment by R
xm2a=param(67); % regulates effectiveness of M2b recruitment by a
ken=param(68); % rate of phagocytosis of damaged cells by neutrophils

% rename variables
pb=y(1);
ab=y(2);
m0b=y(3);
m1b=y(4);
m2b=y(5);
```

```
nu=y(6);
na=y(7);
p=y(8);
a=y(9);
m0=y(10);
m1=y(11);
m2=y(12);
n=y(13);
an=y(14);
R=y(15);
eh=y(16);
ed=y(17);
ee=y(18);

%%%%%%%%%%% Equations
dxdt = zeros(length(y),1);
% 1. pb / PIM
dxdt(1) = sp + dp*(p - pb) + kpm1*m1b*(1/(1+(ab/abinf)^2)) + kpn*na - mupb*
    pb;
% 2. ab / AIM
dxdt(2) = sa + da*(a - ab) + kam1*m1b + kam2*m2b - muab*ab;
% 3. M0b
dxdt(3) = sm - m0b*(((km0pb*pb^2)/(xm0pb^2+pb^2))*(1/(1+(ab/abinf)^2)) + ...
    ((km0ab*ab^2)/(xm0ab^2+ab^2))) + (m0-m0b)*(dm0+((km0pd*pb)/(xm0pd+pb))
        ...
    + ((km0ad*ab)/(xm0ad+ab))) - mum0b*m0b;
% 4. M1b
```

```
dxdt(4) = m0b*(km0pb*pb^2/(xm0pb^2+pb^2))*(1/(1+(ab/abinf)^2)) +...
    (m1-m1b)*(dm1+(km1p*pb/(xm1p+pb)))-mum1b*m1b;
% 5. M2b
dxdt(5) = m0b*(km0ab*ab^2/(xm0ab^2+ab^2)) + (m2-m2b)*(dm2+(km2r*R/(xm2r+R))
    ...
    +(km2a*a/(xm2a+a))) - mum2b*m2b;
% 6. Nu
dxdt(6) = sn - nu*((knup*pb^2)/(xnup^2+pb^2))*(1/(1+(ab/abinf)^2))-munu*nu;
% 7. Na
dxdt(7) = nu*((knup*pb^2)/(xnup^2+pb^2))*(1/(1+(ab/abinf)^2)) - kn*na - muna
    *na;

% LUNG COMPARTMENT

% 8. p / PIM
dxdt(8) = -dp*(p-pb) + kpm1*m1*(1/(1+(a/ainf)^2)) + kpe*ed + kpn*n - mup*p;
% 9. a / AIM
dxdt(9) =-da*(a-ab) + kam1*m1 + kam2*m2 - mua*a;
% 10. M0
dxdt(10) = -(m0-m0b)*(dm0+((km0pd*pb)/(xm0pd+pb))+((km0ad*ab)/(xm0ad+ab)))
    ...
    -m0*((km0p*p^2/(xm0p^2+p^2))*(1/(1+(a/ainf)^2))+(km0a*a^2/(xm0a^2+a^2)))
        -mum0*m0;
% 11. M1
dxdt(11) = m0*(km0p*p^2/(xm0p^2+p^2))*(1/(1+(a/ainf)^2))...
    - (m1-m1b)*(dm1+(km1p*pb/(xm1p+pb)))...
    - (kman*(kanm1*an*m1))*(1/(1+(a/ainf)^2)) -mum1*m1;
```

```
% 12. M2
dxdt(12) = m0*((km0a*a^2)/(xm0a^2+a^2)) ...
    - (m2-m2b)*(dm2+(km2r*R/(xm2r+R))+(km2a*a/(xm2a+a))) ...
    + (kman*(kanm1*an*m1))*(1/(1+(a/ainf)^2)) - mum2*m2;
% 13. N
dxdt(13) = kn*na - kan*n - knn*n;
% 14. AN
dxdt(14) = kan*n - kanm1*an*m1*(1/(1+(a/ainf)^2)) - kanm2*an*m2;
% 15. R
dxdt(15) = krm2*m2-muR*R;
% 16. Eh
if t<=vent_time % 2-hour damage
    dxdt(16) = -eh*(kmne*(m1+n)^2/(xmne^2+(m1+n)^2)) + ed*(br+(ker*R/(xer+R)
        ))...
        - sd*eh + (bp+kep*p)*(eh+ed)*ee;
else
    dxdt(16) = -eh*(kmne*(m1+n)^2/(xmne^2+(m1+n)^2)) + ed*(br+(ker*R/(xer+R)
        ))...
        + (bp+kep*p)*(eh+ed)*ee;
end
% 17. Ed
if t<=vent_time % 2-hour damage
    dxdt(17) = -ed*kem1*m1*(1/(1+(a/ainf)^2)) - ed*ken*n...
        - ed*(br+(ker*R/(xer+R))) + eh*(kmne*(m1+n)^2/(xmne^2+(m1+n)^2))...
        + sd*eh - bd*ed;
else
    dxdt(17) = -ed*kem1*m1*(1/(1+(a/ainf)^2)) - ed*ken*n...
```

```
        - ed*(br+(ker*R/(xer+R))) + eh*(kmne*(m1+n)^2/(xmne^2+(m1+n)^2))...
         - bd*ed;
end
% 18. Ee
dxdt(18)=ed*kem1*m1*(1/(1+(a/ainf)^2)) + bd*ed + ed*ken*n...
    - (bp+kep*p)*(eh+ed)*ee;
end
```

A.5 Code: Random forest

The following script can be run in R to obtain the average importance values for all parameters and other predictors from 1000 runs of the random forest decision tree algorithm. This process is described in Section 2.2.5 and results are shown in Figure 2.8.

```
#### Minucci et al. | Mathematical Modeling of Ventilator-Induced Lung Inflammation
#### random forest with LHS data - run several and find average importance values
require(randomForest)

## Loading required package:  randomForest
## randomForest 4.6-14
## Type rfNews() to see new features/changes/bug fixes.

########################################
######### LHSproc should have 87 columns
########################################

parnames=c('dp','da','dm0','dm1','dm2','xm0pb','xm0ab','xm0pd','xm0ad','xm1p',
           'kpm1','kpe','kam1','kam2','xm0p','xm0a','km1m2','xer','kep ',
```

```
            'km2r','km2a','xnup','kpn','kn','kman','kan','knn','kanm1',
            'kanm2','km0pb','km0ab','km0pd','km0ad','km1p','km0p','km0a',
            'krm2','ker','kem1','knup','abinf','ainf','mupb','muab',
            'mum0b','mum1b','mum2b','mup','mua','mum0','mum1','mum2','muR',
            'munu','muna','sm','sn','bd','br','bp','sd','kmne','sp',
            'sa','xmne','xm2r','xm2a','ken',
            'starting.state','M2.percent.at.t=10','max.M1.percent','max.M2.percent',
            'min.M1.percent','min.M2.percent','max.M1','max.M2','min.M1',
            'min.M2','M1.peak.time','M2.peak.time','M1.peak.ratio','Eh.difference',
            'Eh.ratio.0.5h','Eh.ratio 2h','Eh.ratio.6h','Eh.end.value','outcome')

# insert your own file here
params<-read.table("LHS_rf_example.txt",
   header=FALSE, sep=",", quote="", col.names = parnames)
params$starting.state <- factor(params$starting.state)
params$outcome <- factor(params$outcome)

# remove some predictors related to classification
partest=params
partest$Eh.difference <- NULL
partest$km1m2 <- NULL # dummy parameter
partest$sd <- NULL # not varied
partest$Eh.end.value <- NULL
partest$starting.state <- NULL

#####################################################################
nchar<-81; # total columns minus one (outcome is response variable)
```

```r
#set.seed(29)

# set number of runs and create arrays for saving data
nruns=5 # will take the average of nruns importance values
importance.all<-array(0, dim=c(nchar,nruns))

for(i in 1:nruns){
  print(i) # tells what iteration you're on (out of nruns)
  train = sample(1:nrow(partest), 50) # example, training set should be >50
  rf.par = randomForest(outcome~., data = partest, subset = train) # default mtry
  importance.rf=importance(rf.par)
  importance.all[,i]<-round(importance(rf.par), 2) # look at variable importance
}
```

```
## [1] 1
## [1] 2
## [1] 3
## [1] 4
## [1] 5
```

```r
# save this file and import into MATLAB script to make plot
# write.table(importance.all,"FILE PATH/importance_values_from_R.txt",
#  sep="\t", row.names=FALSE)
```

A.6 Code: plot random forest results

This MATLAB script loads the file generated in Section A.5 and plots the top ten average importance values.

```
% Minucci et al. | Mathematical Modeling of Ventilator-Induced Lung Inflammation
% finds & plots top 10 importance values

% load file
rf=readtable('importance_values_from_R');
rf=table2array(rf);

% first load importance values
im_vals=rf;
im_vals=mean(im_vals,2);
im_stdev=std(rf,0,2);
% importance value names

im_names={'dp','da','dm0','dm1','dm2','xm0pb','xm0ab','xm0pd','xm0ad','xm1p',...
    'kpm1','kpe','kam1','kam2','xm0p','xm0a','xer','kep','km2r','km2a',...
    'xnup','kpn','kn','kman','kan','knn','kanm1','kanm2','km0pb','km0ab',...
    'km0pd','km0ad','km1p','km0p','km0a','krm2','ker','kem1','knup','abinf',...
    'ainf','mupb','muab','mum0b','mum1b','mum2b','mup','mua','mum0','mum1',...
    'mum2','muR','munu','muna','sm','sn','bd','br','bp','kmne',...
    'sp','sa','xmne','xm2r','xm2a','ken','M2% at 10h','max M1%',...
```

```
    'max M2%','min M1%','min M2%','max M1','max M2',...
    'min M1','min M2','M1 peak time','M2 peak time','M1 peak ratio',...
    'Eh ratio at 0.5h','Eh ratio at 2h','Eh ratio at 6h'};

% find max values
[top_im_vals,im_index]=maxk(im_vals,10);
top_im_names=im_names(im_index);

% plot
x = categorical(top_im_names);
x = reordercats(x,top_im_names);
y = top_im_vals;
figure
hold on
bar(x,y);
er = errorbar(x,y,[],im_stdev(im_index),'linewidth',2);
er.Color = [0 0 0];
er.LineStyle = 'none';
set(gca,'fontsize',16)
ylabel('Average importance value')
xlabel('Predictor')
```

Appendix B

Chapter 4 supplementary material

B.1 Code: agent-based model

The following MATLAB scripts and functions, writted in object-oriented programming notation, are used to obtain results for the agent-based model described in Section 4.2.4. The main scripts which include the parameters and toggles for recruitment, wash, age, plots, etc. are `main_abm.m`, `MacModel.m`, and `MyRules`. To run, enter `model=main_abm;` into the command line.

B.1.1 main_abm.m

```
% main_abm - intro to OOP
classdef main_abm < handle

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

properties (SetAccess = public)
    Models = {};
    Rule;
```

```
9       rule;
10      generations;
11      GenerationSize = 20; %duration in minutes, must also change in
            InflammatoryDataFitting.m
12      Runs = 1;
13      hours = 50;
14      gridSize = 120;
15  end
16
17  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
18
19  methods
20
21  function sim = main_abm()
22      %size, runs, generations
23      sim.generations = sim.hours*60/sim.GenerationSize;
24      for i = 1: sim.Runs
25          sim.Models{i} = MacModel(sim.gridSize);
26          sim.Models{i}.setGeneration(sim.generations);
27          sim.Rule = MyRules(sim.Models{i});
28      end
29      sim.generations = sim.Models{1}.MaxGenerations;
30
31      sim.setParameters();
32      sim.run();
33  end
34
```

```
35 function setParameters(this)
36     % young
37     for i = 1:length(this.Models)
38         % no parameters needed yet
39     end
40 end
41
42 function run(this)
43     display('running ...');
44     for i = 1:length(this.Models)
45         this.Models{i}.run();
46         model = this.Models{i};
47
48         % Append rows of data to matrices for each outcome.
49         this.Rule.Results{model.Outcome}.Runs = this.Rule.Results{model.
               Outcome}.Runs + 1;
50         this.Rule.Results{model.Outcome}.ImmuneCellCounts = this.Rule.
               Results{model.Outcome}.ImmuneCellCounts + model.Rules{1}.
               ImmuneCellCounts;
51         this.Rule.Results{model.Outcome}.ImmuneCellCountsSquared = this.Rule
               .Results{model.Outcome}.ImmuneCellCountsSquared + (model.Rules
               {1}.ImmuneCellCounts .^ 2);
52         this.Rule.Results{model.Outcome}.IntermediateCounts = this.Rule.
               Results{model.Outcome}.IntermediateCounts + model.Rules{1}.
               IntermediateCounts;
53         this.Rule.Results{model.Outcome}.IntermediateCountsSquared = this.
               Rule.Results{model.Outcome}.IntermediateCountsSquared + (model.
```

```
        Rules{1}.IntermediateCounts .^ 2);
54      this.Rule.Results{model.Outcome}.M1Counts = this.Rule.Results{model.
        Outcome}.M1Counts + model.Rules{1}.M1Counts;
55      this.Rule.Results{model.Outcome}.M1CountsSquared = this.Rule.Results
        {model.Outcome}.M1CountsSquared + (model.Rules{1}.M1Counts .^ 2);
56      this.Rule.Results{model.Outcome}.M2Counts = this.Rule.Results{model.
        Outcome}.M2Counts + model.Rules{1}.M2Counts;
57      this.Rule.Results{model.Outcome}.M2CountsSquared = this.Rule.Results
        {model.Outcome}.M2CountsSquared + (model.Rules{1}.M2Counts .^ 2);
58      this.Rule.Results{model.Outcome}.TotalMacs = this.Rule.Results{model
        .Outcome}.TotalMacs + model.Rules{1}.TotalMacs;
59      this.Rule.Results{model.Outcome}.TotalMacsSquared = this.Rule.
        Results{model.Outcome}.TotalMacsSquared + (model.Rules{1}.
        TotalMacs .^ 2);
60      this.Rule.Results{model.Outcome}.ProInflammatoryCounts = this.Rule.
        Results{model.Outcome}.ProInflammatoryCounts + model.Rules{1}.
        ProInflammatoryCounts;
61      this.Rule.Results{model.Outcome}.ProInflammatoryCountsSquared = this
        .Rule.Results{model.Outcome}.ProInflammatoryCountsSquared + (
        model.Rules{1}.ProInflammatoryCounts .^ 2);
62      this.Rule.Results{model.Outcome}.AntiInflammatoryCounts = this.Rule.
        Results{model.Outcome}.AntiInflammatoryCounts + model.Rules{1}.
        AntiInflammatoryCounts;
63      this.Rule.Results{model.Outcome}.AntiInflammatoryCountsSquared =
        this.Rule.Results{model.Outcome}.AntiInflammatoryCountsSquared +
        (model.Rules{1}.AntiInflammatoryCounts .^ 2);
```

```
64          this.Rule.Results{model.Outcome}.SOCSCounts = this.Rule.Results{
               model.Outcome}.SOCSCounts + model.Rules{1}.SOCSCounts;
65          this.Rule.Results{model.Outcome}.SOCSCountsSquared = this.Rule.
               Results{model.Outcome}.SOCSCountsSquared + (model.Rules{1}.
               SOCSCounts .^ 2);
66          this.Rule.Results{model.Outcome}.AverageM1Activation = this.Rule.
               Results{model.Outcome}.AverageM1Activation + model.Rules{1}.
               AverageM1Activation;
67          this.Rule.Results{model.Outcome}.AverageM1ActivationSquared = this.
               Rule.Results{model.Outcome}.AverageM1ActivationSquared + (model.
               Rules{1}.AverageM1Activation .^ 2);
68          this.Rule.Results{model.Outcome}.AverageM2Activation = this.Rule.
               Results{model.Outcome}.AverageM2Activation + model.Rules{1}.
               AverageM2Activation;
69          this.Rule.Results{model.Outcome}.AverageM2ActivationSquared = this.
               Rule.Results{model.Outcome}.AverageM2ActivationSquared + (model.
               Rules{1}.AverageM2Activation .^ 2);
70          this.Rule.Results{model.Outcome}.RecruitedCells = this.Rule.Results{
               model.Outcome}.RecruitedCells + model.Rules{1}.RecruitedCells;
71          this.Rule.Results{model.Outcome}.RecruitedCellsSquared = this.Rule.
               Results{model.Outcome}.RecruitedCellsSquared + (model.Rules{1}.
               RecruitedCells .^ 2);
72          this.Rule.Results{model.Outcome}.ProbRecruited = this.Rule.Results{
               model.Outcome}.ProbRecruited + model.Rules{1}.ProbRecruited;
73          this.Rule.Results{model.Outcome}.ProbRecruitedSquared = this.Rule.
               Results{model.Outcome}.ProbRecruitedSquared + (model.Rules{1}.
               ProbRecruited .^ 2);
```

```
74
75      end
76
77      % calculate averages
78          this.Rule.Results{model.Outcome}.ImmuneCellCounts = this.Rule.
                Results{model.Outcome}.ImmuneCellCounts / this.Rule.Results{model
                .Outcome}.Runs;
79          this.Rule.Results{model.Outcome}.ImmuneCellCountsSquared = this.Rule
                .Results{model.Outcome}.ImmuneCellCountsSquared / this.Rule.
                Results{model.Outcome}.Runs;
80          this.Rule.Results{model.Outcome}.IntermediateCounts = this.Rule.
                Results{model.Outcome}.IntermediateCounts / this.Rule.Results{
                model.Outcome}.Runs;
81          this.Rule.Results{model.Outcome}.IntermediateCountsSquared = this.
                Rule.Results{model.Outcome}.IntermediateCountsSquared / this.Rule
                .Results{model.Outcome}.Runs;
82          this.Rule.Results{model.Outcome}.M1Counts = this.Rule.Results{model.
                Outcome}.M1Counts / this.Rule.Results{model.Outcome}.Runs;
83          this.Rule.Results{model.Outcome}.M1CountsSquared = this.Rule.Results
                {model.Outcome}.M1CountsSquared / this.Rule.Results{model.Outcome
                }.Runs;
84          this.Rule.Results{model.Outcome}.M2Counts = this.Rule.Results{model.
                Outcome}.M2Counts / this.Rule.Results{model.Outcome}.Runs;
85          this.Rule.Results{model.Outcome}.M2CountsSquared = this.Rule.Results
                {model.Outcome}.M2CountsSquared / this.Rule.Results{model.Outcome
                }.Runs;
```

```
86          this.Rule.Results{model.Outcome}.TotalMacs = this.Rule.Results{model
                .Outcome}.TotalMacs / this.Rule.Results{model.Outcome}.Runs;
87          this.Rule.Results{model.Outcome}.TotalMacsSquared = this.Rule.
                Results{model.Outcome}.TotalMacsSquared / this.Rule.Results{model
                .Outcome}.Runs;
88          this.Rule.Results{model.Outcome}.ProInflammatoryCounts = this.Rule.
                Results{model.Outcome}.ProInflammatoryCounts / this.Rule.Results{
                model.Outcome}.Runs;
89          this.Rule.Results{model.Outcome}.ProInflammatoryCountsSquared = this
                .Rule.Results{model.Outcome}.ProInflammatoryCountsSquared / this.
                Rule.Results{model.Outcome}.Runs;
90          this.Rule.Results{model.Outcome}.AntiInflammatoryCounts = this.Rule.
                Results{model.Outcome}.AntiInflammatoryCounts / this.Rule.Results
                {model.Outcome}.Runs;
91          this.Rule.Results{model.Outcome}.AntiInflammatoryCountsSquared =
                this.Rule.Results{model.Outcome}.AntiInflammatoryCountsSquared /
                this.Rule.Results{model.Outcome}.Runs;
92          this.Rule.Results{model.Outcome}.SOCSCounts = this.Rule.Results{
                model.Outcome}.SOCSCounts / this.Rule.Results{model.Outcome}.Runs
                ;
93          this.Rule.Results{model.Outcome}.SOCSCountsSquared = this.Rule.
                Results{model.Outcome}.SOCSCountsSquared / this.Rule.Results{
                model.Outcome}.Runs;
94          this.Rule.Results{model.Outcome}.AverageM1Activation = this.Rule.
                Results{model.Outcome}.AverageM1Activation / this.Rule.Results{
                model.Outcome}.Runs;
```

```
95          this.Rule.Results{model.Outcome}.AverageM1ActivationSquared = this.
               Rule.Results{model.Outcome}.AverageM1ActivationSquared / this.
               Rule.Results{model.Outcome}.Runs;
96          this.Rule.Results{model.Outcome}.AverageM2Activation = this.Rule.
               Results{model.Outcome}.AverageM2Activation / this.Rule.Results{
               model.Outcome}.Runs;
97          this.Rule.Results{model.Outcome}.AverageM2ActivationSquared = this.
               Rule.Results{model.Outcome}.AverageM2ActivationSquared / this.
               Rule.Results{model.Outcome}.Runs;
98          this.Rule.Results{model.Outcome}.RecruitedCells = this.Rule.Results{
               model.Outcome}.RecruitedCells / this.Rule.Results{model.Outcome}.
               Runs;
99          this.Rule.Results{model.Outcome}.RecruitedCellsSquared = this.Rule.
               Results{model.Outcome}.RecruitedCellsSquared / this.Rule.Results{
               model.Outcome}.Runs;
100         this.Rule.Results{model.Outcome}.ProbRecruited = this.Rule.Results{
               model.Outcome}.ProbRecruited / this.Rule.Results{model.Outcome}.
               Runs;
101         this.Rule.Results{model.Outcome}.ProbRecruitedSquared = this.Rule.
               Results{model.Outcome}.ProbRecruitedSquared / this.Rule.Results{
               model.Outcome}.Runs;
102
103         % convert to vectors
104         %%% healthy outcome
105         avgm0count_h = [this.Models{i}.InitialImmuneCount (this.Rule.Results
               {1}.ImmuneCellCounts)];
106         avgintcount_h = [0 (this.Rule.Results{1}.IntermediateCounts)];
```

```
107         avgm1count_h = [this.Models{i}.InitialM1Count (this.Rule.Results{1}.
                M1Counts)];
108         avgm2count_h = [this.Models{i}.InitialM2Count (this.Rule.Results{1}.
                M2Counts)];
109         avgtotalmacs_h = [avgm0count_h(1)+avgm1count_h(1)+avgm2count_h(1) (
                this.Rule.Results{1}.TotalMacs)];
110         avgpimcount_h = [this.Models{i}.InitialPIM (this.Rule.Results{1}.
                ProInflammatoryCounts)];
111         avgaimcount_h = [this.Models{i}.InitialAIM (this.Rule.Results{1}.
                AntiInflammatoryCounts)];
112         avgsocscount_h = [this.Models{i}.InitialSOCS (this.Rule.Results{1}.
                SOCSCounts)];
113         avgm1act_h = [mean(mean(this.Models{i}.InitialM1ActivationLattice))
                (this.Rule.Results{1}.AverageM1Activation)];
114         avgm2act_h = [mean(mean(this.Models{i}.InitialM2ActivationLattice))
                (this.Rule.Results{1}.AverageM2Activation)];
115         avgrecruit_h = [0 (this.Rule.Results{1}.RecruitedCells)];
116         avgprobrecruit_h = [0 (this.Rule.Results{1}.ProbRecruited)];
117
118         avgm0count_sq_h = [avgm0count_h(1)^2 (this.Rule.Results{1}.
                ImmuneCellCountsSquared)];
119         avgintcount_sq_h = [avgintcount_h(1)^2 (this.Rule.Results{1}.
                IntermediateCountsSquared)];
120         avgm1count_sq_h = [avgm1count_h(1)^2 (this.Rule.Results{1}.
                M1CountsSquared)];
121         avgm2count_sq_h = [avgm2count_h(1)^2 (this.Rule.Results{1}.
                M2CountsSquared)];
```

```
122         avgtotalmacs_sq_h = [avgtotalmacs_h(1)^2 (this.Rule.Results{1}.
                TotalMacsSquared)];
123         avgpimcount_sq_h = [avgpimcount_h(1)^2 (this.Rule.Results{1}.
                ProInflammatoryCountsSquared)];
124         avgaimcount_sq_h = [avgaimcount_h(1)^2 (this.Rule.Results{1}.
                AntiInflammatoryCountsSquared)];
125         avgsocscount_sq_h = [avgsocscount_h(1)^2 (this.Rule.Results{1}.
                SOCSCountsSquared)];
126         avgm1act_sq_h = [avgm1act_h(1)^2 (this.Rule.Results{1}.
                AverageM1ActivationSquared)];
127         avgm2act_sq_h = [avgm2act_h(1)^2 (this.Rule.Results{1}.
                AverageM2ActivationSquared)];
128         avgrecruit_sq_h = [avgrecruit_h(1)^2 (this.Rule.Results{1}.
                RecruitedCellsSquared)];
129         avgprobrecruit_sq_h = [avgprobrecruit_h(1)^2 (this.Rule.Results{1}.
                ProbRecruitedSquared)];
130
131         %%% inflamed outcome
132         avgm0count_i = [this.Models{i}.InitialImmuneCount (this.Rule.Results
                {2}.ImmuneCellCounts)];
133         avgintcount_i = [0 (this.Rule.Results{2}.IntermediateCounts)];
134         avgm1count_i = [this.Models{i}.InitialM1Count (this.Rule.Results{2}.
                M1Counts)];
135         avgm2count_i = [this.Models{i}.InitialM2Count (this.Rule.Results{2}.
                M2Counts)];
136         avgtotalmacs_i = [avgm0count_i(1)+avgm1count_i(1)+avgm2count_i(1) (
                this.Rule.Results{2}.TotalMacs)];
```

```
137         avgpimcount_i = [this.Models{i}.InitialPIM (this.Rule.Results{2}.
                ProInflammatoryCounts)];
138         avgaimcount_i = [this.Models{i}.InitialAIM (this.Rule.Results{2}.
                AntiInflammatoryCounts)];
139         avgsocscount_i = [this.Models{i}.InitialSOCS (this.Rule.Results{2}.
                SOCSCounts)];
140         avgm1act_i = [mean(mean(this.Models{i}.InitialM1ActivationLattice))
                (this.Rule.Results{2}.AverageM1Activation)];
141         avgm2act_i = [mean(mean(this.Models{i}.InitialM2ActivationLattice))
                (this.Rule.Results{2}.AverageM2Activation)];
142         avgrecruit_i = [0 (this.Rule.Results{2}.RecruitedCells)];
143         avgprobrecruit_i = [0 (this.Rule.Results{2}.ProbRecruited)];
144
145         avgm0count_sq_i = [avgm0count_i(1)^2 (this.Rule.Results{2}.
                ImmuneCellCountsSquared)];
146         avgintcount_sq_i = [avgintcount_i(1)^2 (this.Rule.Results{2}.
                IntermediateCountsSquared)];
147         avgm1count_sq_i = [avgm1count_i(1)^2 (this.Rule.Results{2}.
                M1CountsSquared)];
148         avgm2count_sq_i = [avgm2count_i(1)^2 (this.Rule.Results{2}.
                M2CountsSquared)];
149         avgtotalmacs_sq_i = [avgtotalmacs_i(1)^2 (this.Rule.Results{2}.
                TotalMacsSquared)];
150         avgpimcount_sq_i = [avgpimcount_i(1)^2 (this.Rule.Results{2}.
                ProInflammatoryCountsSquared)];
151         avgaimcount_sq_i = [avgaimcount_i(1)^2 (this.Rule.Results{2}.
                AntiInflammatoryCountsSquared)];
```

```
152         avgsocscount_sq_i = [avgsocscount_i(1)^2 (this.Rule.Results{2}.
                SOCSCountsSquared)];
153         avgm1act_sq_i = [avgm1act_i(1)^2 (this.Rule.Results{2}.
                AverageM1ActivationSquared)];
154         avgm2act_sq_i = [avgm2act_i(1)^2 (this.Rule.Results{2}.
                AverageM2ActivationSquared)];
155         avgrecruit_sq_i = [avgrecruit_i(1)^2 (this.Rule.Results{2}.
                RecruitedCellsSquared)];
156         avgprobrecruit_sq_i = [avgprobrecruit_i(1)^2 (this.Rule.Results{2}.
                ProbRecruitedSquared)];
157
158         % compute standard deviations
159         %%% healthy outcome
160         sdm0count_h = sqrt(avgm0count_sq_h - avgm0count_h.^2);
161         sdintcount_h = sqrt(avgintcount_sq_h - avgintcount_h.^2);
162         sdm1count_h = sqrt(avgm1count_sq_h - avgm1count_h.^2);
163         sdm2count_h = sqrt(avgm2count_sq_h - avgm2count_h.^2);
164         sdtotalmacs_h = sqrt(avgtotalmacs_sq_h - avgtotalmacs_h.^2);
165         sdpimcount_h = sqrt(avgpimcount_sq_h - avgpimcount_h.^2);
166         sdaimcount_h = sqrt(avgaimcount_sq_h - avgaimcount_h.^2);
167         sdsocscount_h = sqrt(avgsocscount_sq_h - avgsocscount_h.^2);
168         sdm1act_h = sqrt(avgm1act_sq_h - avgm1act_h.^2);
169         sdm2act_h = sqrt(avgm2act_sq_h - avgm2act_h.^2);
170         sdrecruit_h = sqrt(avgrecruit_sq_h - avgrecruit_h.^2);
171         sdprobrecruit_h = sqrt(avgprobrecruit_sq_h - avgprobrecruit_h.^2);
172
173         %%% inflamed outcome
```

```
174         sdm0count_i = sqrt(avgm0count_sq_i - avgm0count_i.^2);
175         sdintcount_i = sqrt(avgintcount_sq_i - avgintcount_i.^2);
176         sdm1count_i = sqrt(avgm1count_sq_i - avgm1count_i.^2);
177         sdm2count_i = sqrt(avgm2count_sq_i - avgm2count_i.^2);
178         sdtotalmacs_i = sqrt(avgtotalmacs_sq_i - avgtotalmacs_i.^2);
179         sdpimcount_i = sqrt(avgpimcount_sq_i - avgpimcount_i.^2);
180         sdaimcount_i = sqrt(avgaimcount_sq_i - avgaimcount_i.^2);
181         sdsocscount_i = sqrt(avgsocscount_sq_i - avgsocscount_i.^2);
182         sdm1act_i = sqrt(avgm1act_sq_i - avgm1act_i.^2);
183         sdm2act_i = sqrt(avgm2act_sq_i - avgm2act_i.^2);
184         sdrecruit_i = sqrt(avgrecruit_sq_i - avgrecruit_i.^2);
185         sdprobrecruit_i = sqrt(avgprobrecruit_sq_i - avgprobrecruit_i.^2);
186
187         % save some results for healthy - can do the same for inflamed
188         this.Rule.Results{1}.avgm1act = avgm1act_h;
189         this.Rule.Results{1}.avgm2act = avgm2act_h;
190         this.Rule.Results{1}.avgpimcount = avgpimcount_h;
191         this.Rule.Results{1}.avgaimcount = avgaimcount_h;
192         this.Rule.Results{1}.sdm1act = sdm1act_h;
193         this.Rule.Results{1}.sdm2act = sdm2act_h;
194         this.Rule.Results{1}.sdpimcount = sdpimcount_h;
195         this.Rule.Results{1}.sdaimcount = sdaimcount_h;
196
197         t = 1:this.generations;
198         t = t .* (this.GenerationSize/60);
199         t=[0 t]; % include initial conditions in plot
200
```

```
201         % plot results
202         if(model.togglePlot) % full macrophage model
203             %%% healthy outcome
204             figure('name','Results: Healthy Outcome','Position'
                    ,[100,200,1000,800]);
205             subplot(2,3,1)
206             boundedline(t,avgpimcount_h, sdpimcount_h);ylabel('Pro-
                    inflammatory count');xlabel('hours');
207             title('Healthy outcome')
208
209             subplot(2,3,2)
210             boundedline(t,avgaimcount_h, sdaimcount_h);ylabel('Anti-
                    inflammatory count');xlabel('hours');
211
212             subplot(2,3,3)
213             boundedline(t,avgm1act_h, sdm1act_h,'r',t,avgm2act_h, sdm2act_h,
                    'b');ylabel('Average activation');xlabel('hours');
214             legend('M1','M2')
215
216             subplot(2,3,4)
217             boundedline(t,avgrecruit_h,sdrecruit_h);ylabel('Macrophages
                    recruited');xlabel('hours');
218
219             subplot(2,3,5)
220             boundedline(t,avgprobrecruit_h,sdprobrecruit_h);ylabel('Average
                    probability of mac recruitment');xlabel('hours');
221
```

```
222             subplot(2,3,6)
223             boundedline(t,avgm1count_h,sdm1count_h,'r',t,avgm2count_h,
                    sdm2count_h,'b',t,avgintcount_h,sdintcount_h,'y');ylabel('
                    Macrophage count');xlabel('hours');
224             legend('M1','M2','Intermediate')
225
226             %%% inflamed outcome
227             figure('name','Results: Inflamed Outcome','Position'
                    ,[200,200,1000,800]);
228             subplot(2,3,1)
229             boundedline(t,avgpimcount_i, sdpimcount_i);ylabel('Pro-
                    inflammatory count');xlabel('hours');
230             title('Inflamed outcome')
231
232             subplot(2,3,2)
233             boundedline(t,avgaimcount_i, sdaimcount_i);ylabel('Anti-
                    inflammatory count');xlabel('hours');
234
235             subplot(2,3,3)
236             boundedline(t,avgm1act_i, sdm1act_i,'r',t,avgm2act_i, sdm2act_i,
                    'b');ylabel('Average activation');xlabel('hours');
237             legend('M1','M2')
238
239             subplot(2,3,4)
240             boundedline(t,avgrecruit_i,sdrecruit_i);ylabel('Macrophages
                    recruited');xlabel('hours');
241
```

```
242             subplot(2,3,5)
243             boundedline(t,avgprobrecruit_i,sdprobrecruit_i);ylabel('Average
                    probability of mac recruitment');xlabel('hours');
244
245             subplot(2,3,6)
246             boundedline(t,avgm1count_i,sdm1count_i,'r',t,avgm2count_i,
                    sdm2count_i,'b',t,avgintcount_i,sdintcount_i,'y');ylabel('
                    Macrophage count');xlabel('hours');
247             legend('M1','M2','Intermediate')
248
249             figure
250             boundedline(t,avgm1act_h, sdm1act_h,'r',t,avgm2act_h, sdm2act_h,
                    'b');ylabel('Average activation');xlabel('hours');
251             legend('M1','M2')
252             set(gca,'fontsize',16)
253         end
254
255         % plot results
256         if(model.togglePlotSingleMac) % single macrophage model
257             %%% healthy outcome
258             figure('name','Results: Healthy Outcome','Position'
                    ,[100,200,1000,800]);
259             subplot(2,3,1)
260             boundedline(t,avgpimcount_h, sdpimcount_h);ylabel('Pro-
                    inflammatory count');xlabel('hours');
261             title('Healthy outcome')
262
```

```
263             subplot(2,3,2)
264             boundedline(t,avgaimcount_h, sdaimcount_h);ylabel('Anti-
                    inflammatory count');xlabel('hours');
265
266             subplot(2,3,3)
267             boundedline(t,avgm1act_h, sdm1act_h,'r',t,avgm2act_h, sdm2act_h,
                    'b');ylabel('Average activation');xlabel('hours');
268             legend('M1','M2')
269
270             subplot(2,3,4)
271             boundedline(t,avgsocscount_h,sdsocscount_h);ylabel('Total SOCS')
                    ;xlabel('hours');
272
273             subplot(2,3,5) % make sure no macs were recruited
274             boundedline(t,avgrecruit_h,sdrecruit_h);ylabel('Macrophages
                    recruited');xlabel('hours');
275         end
276
277         if model.toggleLayeredFigure
278             %%% healthy outcome
279             figure('name','Results: Healthy Outcome')
280             a=area(t,[avgm0count_h; avgm1count_h; avgintcount_h;
                    avgm2count_h]');
281             a(1).FaceColor = [199 199 199]/255; % M0
282             a(2).FaceColor = [255 133 194]/255; % M1
283             a(3).FaceColor = [255 253 128]/255; % intermediate
284             a(4).FaceColor = [161 176 255]/255; % M2
```

```
285             xlabel('hours')
286             ylabel('Macrophages')
287             legend('M0','M1','Intermediate','M2')
288             set(gca,'fontsize',16)
289             %%% inflamed outcome
290             figure('name','Results: Inflamed Outcome')
291             a=area(t,[avgm0count_i; avgm1count_i; avgintcount_i;
                    avgm2count_i]');
292             a(1).FaceColor = [199 199 199]/255; % M0
293             a(2).FaceColor = [255 133 194]/255; % M1
294             a(3).FaceColor = [255 253 128]/255; % intermediate
295             a(4).FaceColor = [161 176 255]/255; % M2
296             xlabel('hours')
297             ylabel('Macrophages')
298             legend('M0','M1','Intermediate','M2')
299         end
300
301         % plot results
302         if(model.SingleMacWriteUpFigures) % single macrophage model
303             %%% healthy outcome
304             figure
305             boundedline(t,avgpimcount_h, sdpimcount_h,'r');ylabel('Pro-
                    inflammatory count');xlabel('Time (hours)');
306             set(gca,'fontsize',16)
307
308             figure
```

```
309             boundedline(t,avgaimcount_h, sdaimcount_h,'b');ylabel('Anti-
                    inflammatory count');xlabel('Time (hours)');
310             set(gca,'fontsize',16)
311
312             figure
313             boundedline(t,avgm1act_h, sdm1act_h,'r');ylabel('Average M1
                    activation');xlabel('Time (hours)');
314             set(gca,'fontsize',16)
315
316             figure
317             boundedline(t,avgm2act_h, sdm2act_h,'b');ylabel('Average M2
                    activation');xlabel('Time (hours)');
318             set(gca,'fontsize',16)
319
320             figure
321             boundedline(t,avgsocscount_h,sdsocscount_h);ylabel('Total SOCS')
                    ;xlabel('Time (hours)');
322             set(gca,'fontsize',16)
323         end
324
325 end
326
327 end
328
329 end
```

B.1.2 MacModel.m

```
classdef MacModel < handle


properties (SetAccess = public)
    CurrentGeneration = 0;
    PreviousGeneration = 0;
    MaxGenerations;
    GenerationSize = 20; %duration in minutes, must also change in main_abm,
        MyRules
    ImmuneLattice;
    ProInflammatoryLattice;
    AntiInflammatoryLattice;
    SOCSLattice;
    M1ActivationLattice;
    M2ActivationLattice;
    InitialM1ActivationLattice;
    InitialM2ActivationLattice;
    ImmuneCellCount;
    InitialImmuneAge;
    ImmuneAge;
    InitialImmuneMatrix;
    InitialImmuneCount=0;
    InitialM1Count=100;
    InitialM2Count=100;
    InitialIntCount=0;
    InitialPIM = 0;
    InitialAIM = 0;
```

```
26      InitialSOCS = 0;
27      AgeMeanM0 = 24; % change here & in MyRules.m
28      AgeStDevM0 = 6; % change here & in MyRules.m
29      AgeMeanActivated = 12; % change here & in MyRules.m
30      AgeStDevActivated = 3; % change here & in MyRules.m
31      Outcome = Outcomes.Healthy;
32      ShowLattices = false;
33      toggleImmune = true; % from previous version, always true
34      ToggleRecruitment = true; % recruit macrophages from outside grid
35      toggleWash = false; % wash: reset extracellular environment at some
            point
36      togglePlot = true;
37      togglePlotSingleMac = false;
38      toggleLayeredFigure = false;
39      SingleMacWriteUpFigures = false;
40      InitialMatrix;
41      Rules;
42      RuleSet = {'MyRules'};
43      Debug = false;
44  %                   M0          blank   problem  M1           problem  M2
                problem  intermediate   problem
45      ImmuneColorMap = [255 255 255; 0 0 0; 0 255 0; 255 133 194; 0 255 0; 161
            176 255; 0 255 0; 255, 253, 128; 0 255 0] / 255;
46  end
47
48
49  methods
```

```
50
51  function model = MacModel(size)
52      % Create macrophage matrix
53      model.RandomizeMatrix(size);
54      model.InitialMatrix = ones(size,size);
55      % Create all the rule objects.
56      for i = 1:length(model.RuleSet)
57          model.Rules{i} = feval(str2func(model.RuleSet{i}), model);
58      end
59      model.reset();
60  end
61
62  function RandomizeMatrix(this,size)
63      this.InitialImmuneMatrix = ones(size/3,size/3);
64      this.InitialImmuneAge = ones(size/3,size/3);
65      this.InitialImmuneMatrix = this.InitialImmuneMatrix.*2;
66      this.InitialM1ActivationLattice = zeros(size/3,size/3);
67      this.InitialM2ActivationLattice = zeros(size/3,size/3);
68      %Add random initial immune cells
69      n_m0=0;
70      n_m1=0;
71      n_m2=0;
72      % set up M0
73      while n_m0 < this.InitialImmuneCount
74          i = randi([1 size/3]);
75          j = randi([1 size/3]);
76          this.InitialImmuneMatrix(i,j) = 1;
```

```
77          n_m0 = n_m0 + 1;
78          this.InitialImmuneAge(i,j) = round(random_in_range(0,(this.AgeMeanM0
                +this.AgeStDevM0)*60/this.GenerationSize));
79  %         this.InitialImmuneAge(i,j) = (round(this.AgeStDevM0*randn(1)) +
    this.AgeMeanM0)*60/this.GenerationSize;
80          this.InitialM1ActivationLattice(i,j)=random_in_range(0,0.25);
81          this.InitialM2ActivationLattice(i,j)=random_in_range(0,0.25-this.
                InitialM1ActivationLattice(i,j));
82      end
83
84      % set up M1
85      while n_m1 < this.InitialM1Count
86          i = randi([1 size/3]);
87          j = randi([1 size/3]);
88          if this.InitialImmuneMatrix(i,j)~=1
89              this.InitialImmuneMatrix(i,j) = 4;
90              n_m1 = n_m1 + 1;
91          this.InitialImmuneAge(i,j) = round(random_in_range(0,(this.
                AgeMeanActivated+this.AgeStDevActivated)*60/this.GenerationSize))
                ;
92  %         this.InitialImmuneAge(i,j) = (round(this.AgeStDevActivated*randn
    (1)) + this.AgeMeanActivated)*60/this.GenerationSize;
93          this.InitialM1ActivationLattice(i,j)=random_in_range(0.5,1);
94          this.InitialM2ActivationLattice(i,j)=random_in_range(0,0.49);
95          end
96      end
97
```

```
98      % set up M2
99      while n_m2 < this.InitialM2Count
100         i = randi([1 size/3]);
101         j = randi([1 size/3]);
102         if this.InitialImmuneMatrix(i,j)~=1 || this.InitialImmuneMatrix(i,j)
                ~=4
103             this.InitialImmuneMatrix(i,j) = 6;
104             n_m2 = n_m2 + 1;
105         this.InitialImmuneAge(i,j) = round(random_in_range(0,(this.
                AgeMeanActivated+this.AgeStDevActivated)*60/this.GenerationSize))
                ;
106 %         this.InitialImmuneAge(i,j) = (round(this.AgeStDevActivated*randn
    (1)) + this.AgeMeanActivated)*60/this.GenerationSize;
107         this.InitialM2ActivationLattice(i,j)=random_in_range(0.5,1);
108         this.InitialM1ActivationLattice(i,j)=random_in_range(0,0.49);
109         end
110     end
111     % set up intermediate
112     while n_m2 < this.InitialM2Count
113         i = randi([1 size/3]);
114         j = randi([1 size/3]);
115         if this.InitialImmuneMatrix(i,j)~=1 || this.InitialImmuneMatrix(i,j)
                ~=4
116             this.InitialImmuneMatrix(i,j) = 6;
117             n_m2 = n_m2 + 1;
118         this.InitialImmuneAge(i,j) = round(random_in_range(0,(this.
                AgeMeanActivated+this.AgeStDevActivated)*60/this.GenerationSize))
```

```
            ;
%         this.ImmuneAge(i,j) = (round(this.AgeStDevActivated*randn(1)) +
    this.AgeMeanActivated)*60/this.GenerationSize;
        this.InitialM1ActivationLattice(i,j)=random_in_range(0,0.49);
        this.InitialM2ActivationLattice(i,j)=random_in_range(max([0, 0.25-
            this.InitialM1ActivationLattice(i,j)]),0.49);
        end
    end

    this.ImmuneLattice = this.InitialImmuneMatrix;
end

function setGeneration(this, n)
    this.MaxGenerations = n;
end

function reset(this)
    this.ImmuneLattice = this.InitialImmuneMatrix;
    this.ImmuneAge = this.InitialImmuneAge;
    this.ProInflammatoryLattice = zeros(size(this.InitialImmuneMatrix));
    this.ProInflammatoryLattice(14:27,14:27) = this.InitialPIM;
    this.AntiInflammatoryLattice = zeros(size(this.InitialImmuneMatrix));
    this.AntiInflammatoryLattice(14:27,14:27) = this.InitialAIM;
    this.SOCSLattice = zeros(size(this.InitialImmuneMatrix));
    this.SOCSLattice(2,2) = this.InitialSOCS;
    this.M1ActivationLattice = this.InitialM1ActivationLattice;
    this.M2ActivationLattice = this.InitialM2ActivationLattice;
```

```
143     this.CurrentGeneration = 0;
144     this.PreviousGeneration = 0;
145 
146     for i = 1:length(this.Rules)
147         rule = this.Rules{i};
148         rule.reset();
149     end
150 end
151 
152 function run(this)
153     % Run the simulation using the rules in the model's RuleSet
154     % parameter.
155 
156     % Splitting up this figure into separate figures since subimage
157     % requires the Image Processing Toolkit.
158 
159     if this.ShowLattices
160         immune_h = figure;
161         set(immune_h, 'Name', 'Macrophages', ...
162                       'OuterPosition', [100 100 300 300]);
163         title('Immune Cells')
164 
165         proinflammatory_h = figure;
166         set(proinflammatory_h, 'Name', 'Pro-inflammatory Mediators', ...
167                       'OuterPosition', [400 100 300 300]);
168         title('Pro-inflammatory Cells')
169 
```

```
170         antiinflammatory_h = figure;
171         set(antiinflammatory_h, 'Name', 'Anti-inflammatory Mediators', ...
172                     'OuterPosition', [700 100 300 300]);
173         title('Anti-inflammatory Cells')
174 
175         m1activation_h = figure;
176         set(m1activation_h, 'Name', 'M1 Activation', ...
177                     'OuterPosition', [100 600 300 300]);
178         title('M1 Activation')
179 
180         m2activation_h = figure;
181         set(m2activation_h, 'Name', 'M2 Activation', ...
182                     'OuterPosition', [400 600 300 300]);
183         title('M2 Activation')
184     end
185 
186     if this.Debug
187         pause on;
188     else
189         pause off;
190     end
191 
192     for generation = 1:this.MaxGenerations
193         this.CurrentGeneration = generation;
194         this.PreviousGeneration = generation-1;
195 
196         if this.ShowLattices
```

```
197             % Update the immune cell lattice window
198             set(0, 'CurrentFigure', immune_h);
199             image(this.ImmuneLattice);
200             colormap(this.ImmuneColorMap);
201             axis square;
202             set(gca, 'XTick', [], ...
203                      'YTick', [], ...
204                      'XTickLabel', '', ...
205                      'YTickLabel', '');
206             title(sprintf('Total time (Hours) %f', generation/(60/this.
                   GenerationSize))); %MUST CHANGE IF YOU CHANGE GENERATION SIZE
207
208             set(0, 'CurrentFigure', proinflammatory_h);
209             imagesc(this.ProInflammatoryLattice,[0 4]);
210             colormap(autumn);
211             axis square;
212             set(gca, 'XTick', [], ...
213                      'YTick', [], ...
214                      'XTickLabel', '', ...
215                      'YTickLabel', '');
216
217             set(0, 'CurrentFigure', antiinflammatory_h);
218             imagesc(this.AntiInflammatoryLattice,[0 4]);
219             colormap(cool);
220             axis square;
221             set(gca, 'XTick', [], ...
222                      'YTick', [], ...
```

```
223                 'XTickLabel', '', ...
224                 'YTickLabel', '');
225
226         set(0, 'CurrentFigure', socs_h);
227         imagesc(this.SOCSLattice,[0 4]);
228         colormap(parula);
229         axis square;
230         set(gca, 'XTick', [], ...
231                 'YTick', [], ...
232                 'XTickLabel', '', ...
233                 'YTickLabel', '');
234
235         set(0, 'CurrentFigure', m1activation_h);
236         imagesc(this.M1ActivationLattice,[0 0.5]);
237         colormap(autumn);
238         axis square;
239         set(gca, 'XTick', [], ...
240                 'YTick', [], ...
241                 'XTickLabel', '', ...
242                 'YTickLabel', '');
243
244         set(0, 'CurrentFigure', m2activation_h);
245         imagesc(this.M2ActivationLattice,[0 0.5]);
246         colormap(cool);
247         axis square;
248         set(gca, 'XTick', [], ...
249                 'YTick', [], ...
```

```
                    'XTickLabel', '', ...
                    'YTickLabel', '');

            drawnow;

            if this.Debug
                pause;
            end
        end

        for i = 1:length(this.Rules)
            rule = this.Rules{i};
            rule.apply_rule();
        end

    end

    % After the run we call finalize() on each rule.
    for i = 1:length(this.Rules)
        rule = this.Rules{i};
        rule.finalize();
    end
end

end

end
```

B.1.3 MyRules.m

```
classdef MyRules < LungModelRule

properties (SetAccess = public)
    Model;
    GenerationSize = 20; %duration in minutes, must also change in sbm.m,
       MacModel.m
    ImmuneCellCounts;
    ImmuneCellCountsSquared;
    IntermediateCounts;
    IntermediateCountsSquared;
    M1Counts;
    M2Counts;
    M1CountsSquared;
    M2CountsSquared;
    TotalMacs;
    TotalMacsSquared;
    ProInflammatoryCounts;
    ProInflammatoryCountsSquared;
    AntiInflammatoryCounts;
    AntiInflammatoryCountsSquared;
    SOCSCounts;
    SOCSCountsSquared;
    AverageM1Activation;
    AverageM1ActivationSquared;
    AverageM2Activation;
    AverageM2ActivationSquared;
```

```
    RecruitedCells;
    RecruitedCellsSquared;
    ProbRecruited;
    ProbRecruitedSquared;
    Results;
    ProbImmuneCellArrives;

    % parameters
    AgeMeanM0 = 24; % change here & in MacModel.m
    AgeStDevM0 = 6; % change here & in MacModel.m
    AgeMeanActivated = 12; % change here & in MacModel.m
    AgeStDevActivated = 3; % change here & in MacModel.m

    ImmuneProInflammatoryRate = 0.35; % M1s produce pro-inflammatories
    ImmuneAntiInflammatoryRate = 0.55; % M2s produce anti-inflammatories
    ImmuneM1AntiInflammatoryRate=0.12; % M1s produce anti-inflammatories

    ProInflammatoryDecayRate = 0.03;
    AntiInflammatoryDecayRate = 0.03;
    SOCSDecayRate = 0.03;

    PIMNegativeFeedbackRate=0.002;
    AIMNegativeFeedbackRate=0.002;

    RecruitmentMMTerm = 30;
    AIMRecruitScale=0.25;
    PIMActivationScale=0.75;
```

```
    AIMActivationScale=0.75;
    AIMInfinity=4; % Recruitment: M1 activation inhibited by AIM

    M1ActivationRate=0.05; % M1 activation increased by PIM (0,1)
    M1ActHillParameter=1; % increase M1 expression via PIM
    M2ActScalar=0.06; % increase M2 activation by AIM
    M2ActHillParameter=0.85; % Hill: increase M2 activation via AIM
    M1AIMInfinity=0.05; % inhibition of M1s production of pro-inflammatories
        by AIM
    M1DecreaseViaAIM=0.01; % AIM decreases M1 activation
    M1DecreaseViaAIMHill=0.4; % Hill parameter - AIM decreases M1 activation

    SOCSProductionRate=4; % rate at which AIM produce SOCS
    AIMSOCSHill=3.5; % AIM production of SOCS, larger -> requires more
       accumulation to be effective
    M1SOCSInfinity=4; % SOCS inhibition of M1 activation
    M2SOCSInfinity=4; % SOCS inhibition of M2 activation
    AIMSOCSInfinity=0.3; % SOCS inhibition of AIM production

    WashTime=12; % hours
    WashAIM=30;%3.5;
end

methods

    function rule = MyRules(model)
        rule.Model = model;
```

```
78          rule.reset();
79
80          rule.Results = cell(1, length(enumeration('Outcomes')));
81
82          % We have to initialize all of the matrices where our counts
83          % get appended to. These don't get wiped out between runs.
84          % They stick around even after a reset().
85          for i = 1:length(enumeration('Outcomes'))
86              rule.Results{i}.Runs = 0;
87              rule.Results{i}.ImmuneCellCounts = zeros(1, model.MaxGenerations
                    );
88              rule.Results{i}.ImmuneCellCountsSquared = zeros(1, model.
                    MaxGenerations);
89              rule.Results{i}.IntermediateCounts = zeros(1, model.
                    MaxGenerations);
90              rule.Results{i}.IntermediateCountsSquared = zeros(1, model.
                    MaxGenerations);
91              rule.Results{i}.M1Counts = zeros(1, model.MaxGenerations);
92              rule.Results{i}.M1CountsSquared = zeros(1, model.MaxGenerations)
                    ;
93              rule.Results{i}.M2Counts = zeros(1, model.MaxGenerations);
94              rule.Results{i}.M2CountsSquared = zeros(1, model.MaxGenerations)
                    ;
95              rule.Results{i}.TotalMacs = zeros(1, model.MaxGenerations);
96              rule.Results{i}.TotalMacsSquared = zeros(1, model.MaxGenerations
                    );
```

```
97              rule.Results{i}.ProInflammatoryCounts = zeros(1, model.
                    MaxGenerations);
98              rule.Results{i}.ProInflammatoryCountsSquared = zeros(1, model.
                    MaxGenerations);
99              rule.Results{i}.AntiInflammatoryCounts = zeros(1, model.
                    MaxGenerations);
100             rule.Results{i}.AntiInflammatoryCountsSquared = zeros(1, model.
                    MaxGenerations);
101             rule.Results{i}.SOCSCounts = zeros(1, model.MaxGenerations);
102             rule.Results{i}.SOCSCountsSquared = zeros(1, model.
                    MaxGenerations);
103             rule.Results{i}.AverageM1Activation = zeros(1, model.
                    MaxGenerations);
104             rule.Results{i}.AverageM1ActivationSquared = zeros(1, model.
                    MaxGenerations);
105             rule.Results{i}.AverageM2Activation = zeros(1, model.
                    MaxGenerations);
106             rule.Results{i}.AverageM2ActivationSquared = zeros(1, model.
                    MaxGenerations);
107             rule.Results{i}.RecruitedCells = zeros(1, model.MaxGenerations);
108             rule.Results{i}.RecruitedCellsSquared = zeros(1, model.
                    MaxGenerations);
109             rule.Results{i}.ProbRecruited = zeros(1, model.MaxGenerations);
110             rule.Results{i}.ProbRecruitedSquared = zeros(1, model.
                    MaxGenerations);
111         end
112     end
```

```
113
114     function apply_rule(this)
115         % move macrophages, determine M1/M2 activation, update age
116         model = this.Model;
117         if(model.toggleImmune)
118             this.moveImmuneCells();
119         end
120
121 %       if wash: reset pro/anti-inflammatories
122         if(model.toggleWash) && (model.CurrentGeneration==this.WashTime*60/
                this.GenerationSize)
123             model.ProInflammatoryLattice = zeros(size(model.
                    ProInflammatoryLattice));
124             model.AntiInflammatoryLattice = zeros(size(model.
                    AntiInflammatoryLattice));
125             model.AntiInflammatoryLattice(14:27,14:27) = this.WashAIM;
126 %             model.AntiInflammatoryLattice(14:27,14:27) = model.
    AntiInflammatoryLattice(14:27,14:27) + this.WashAIM;
127         end
128
129         % diffuse pro- and anti-inflammatory mediators
130         model.ProInflammatoryLattice = diffuse(model.ProInflammatoryLattice)
                ;
131         model.AntiInflammatoryLattice = diffuse(model.
                AntiInflammatoryLattice);
132 %         model.SOCSLattice = diffuse(model.SOCSLattice);
133
```

```
134         % The amount of pro- and anti-inflammatories in each cell decays at
                a set rate
135         model.ProInflammatoryLattice = model.ProInflammatoryLattice .* (1 -
                this.ProInflammatoryDecayRate);
136         model.AntiInflammatoryLattice = model.AntiInflammatoryLattice .* (1
                - this.AntiInflammatoryDecayRate);
137         model.SOCSLattice = model.SOCSLattice .* (1 - this.SOCSDecayRate);
138
139         % recruit macrophages
140         if(model.toggleImmune)
141             %%% calculate the probability a macrophage will be recruited
142             if(model.ToggleRecruitment)
143                 this.ProbImmuneCellArrives = (model.ProInflammatoryLattice+
                        this.AIMRecruitScale*model.AntiInflammatoryLattice)
                        .^2./((model.ProInflammatoryLattice+this.AIMRecruitScale*
                        model.AntiInflammatoryLattice).^2+this.RecruitmentMMTerm
                        ^2);
144             else
145                 this.ProbImmuneCellArrives = zeros(size(model.ImmuneLattice)
                        );
146             end
147             this.ProbRecruited(model.CurrentGeneration)=mean(mean(this.
                    ProbImmuneCellArrives));
148             temp = (rand(size(model.ImmuneLattice)) < this.
                    ProbImmuneCellArrives) & (~ismember(model.ImmuneLattice,[1 4
                    6 8]));
```

```
149             this.RecruitedCells(model.CurrentGeneration)=sum(sum(temp)); %
                    number of macrophages recruited
150             % M1 & M2 activation
151             temp_m1act=model.ProInflammatoryLattice./(model.
                    ProInflammatoryLattice+this.PIMActivationScale).*(1/1+(model.
                    AntiInflammatoryLattice/this.AIMInfinity));
152             temp_m2act=model.AntiInflammatoryLattice./(model.
                    AntiInflammatoryLattice+this.AIMActivationScale);
153
154             % if M1act+M2act>1, scaling is needed
155             test=temp_m1act+temp_m2act>1;
156             temp_sum=temp_m1act+temp_m2act;
157             temp_m1act(test)=temp_m1act(test)./temp_sum(test);
158             temp_m2act(test)=temp_m2act(test)./temp_sum(test);
159
160             model.M1ActivationLattice(temp) = temp_m1act(temp);
161             model.M2ActivationLattice(temp) = temp_m2act(temp);
162             temp_m1=temp_m1act>0.5;
163             temp_m2=temp_m2act>0.5;
164             temp_int=((temp_m1act+temp_m2act)>=0.25) & (temp_m1act<0.5) & (
                    temp_m2act<0.5);
165             temp_naive=(temp_m1act+temp_m2act)<0.25;
166             % update immune state
167             model.ImmuneLattice(temp_naive & temp) = ImmuneStates_sbm.
                    M0Static;
168             model.ImmuneLattice(temp_m1 & temp) = ImmuneStates_sbm.M1Static;
169             model.ImmuneLattice(temp_m2 & temp) = ImmuneStates_sbm.M2Static;
```

```
170             model.ImmuneLattice(temp_int & temp) = ImmuneStates_sbm.
                    MIntStatic;
171             % define ages (naive/activated)
172             temp_age_act=ismember(model.ImmuneLattice,[4 6 8]); % activated
                    macrophages
173 %             ages=max([0,round((this.AgeStDevActivated + this.
    AgeMeanActivated).*randn(size(model.ImmuneLattice)).*60/this.
    GenerationSize)]);
174             ages=round(this.AgeStDevActivated.*randn(size(model.
                    ImmuneLattice)) + this.AgeMeanActivated).*60/this.
                    GenerationSize;
175             model.ImmuneAge(temp_age_act & temp)=ages(temp_age_act & temp);
176             temp_age_m0=model.ImmuneLattice==ImmuneStates_sbm.M0Static; %
                    activated macrophages
177 %             ages=max([0,round((this.AgeStDevM0 + this.AgeMeanM0).*randn(
    size(model.ImmuneLattice)).*60/this.GenerationSize)]);
178             ages=round(this.AgeStDevM0.*randn(size(model.ImmuneLattice)) +
                    this.AgeMeanM0).*60/this.GenerationSize;
179             model.ImmuneAge(temp_age_m0 & temp)=ages(temp_age_m0 & temp);
180             clear temp temp_m1act temp_m2act temp_age_act temp_age_m0 ages;
181
182             %%%% all immune cells recruit & produce inflammatories
183             temp=ismember(model.ImmuneLattice,[1 4 6 8]); % any kind of
                    macrophage, proportional to activation (see below)
184             % M1s produce pro-inflammatories, inhibited by AIM
185             model.ProInflammatoryLattice(temp) = model.
                    ProInflammatoryLattice(temp) + this.ImmuneProInflammatoryRate
```

```
            *normrnd(1,0.25,size(model.ImmuneLattice(temp))).*...
186             model.M1ActivationLattice(temp).*(1./(1+(model.
                    AntiInflammatoryLattice(temp)./this.M1AIMInfinity)));
187         % M1s produce anti-inflammatories
188         model.AntiInflammatoryLattice(temp) = model.
                AntiInflammatoryLattice(temp) + this.
                ImmuneM1AntiInflammatoryRate*normrnd(1,0.25,size(model.
                ImmuneLattice(temp))).*...
189             model.M1ActivationLattice(temp);
190         % M2s produce anti-inflammatories, inhibited by SOCS
191         model.AntiInflammatoryLattice(temp) = model.
                AntiInflammatoryLattice(temp) + this.
                ImmuneAntiInflammatoryRate*normrnd(1,0.25,size(model.
                ImmuneLattice(temp))).*...
192             model.M2ActivationLattice(temp).*1./(1+(model.SOCSLattice(
                    temp)./this.AIMSOCSInfinity).^3);
193         % SOCS produced by level of M2 activation
194         model.SOCSLattice(temp) = model.SOCSLattice(temp) + this.
                SOCSProductionRate*normrnd(1,0.25,size(model.ImmuneLattice(
                temp))).*...
195             model.M2ActivationLattice(temp).^2./(model.
                    M2ActivationLattice(temp)+this.AIMSOCSHill^2);
196         clear temp;
197     end
198
199     % Save the plot data
```

```
200         this.ImmuneCellCounts(model.CurrentGeneration) = sum(model.
               ImmuneLattice(:) == 1);
201         this.ProInflammatoryCounts(model.CurrentGeneration) = sum(model.
               ProInflammatoryLattice(:));
202         this.AntiInflammatoryCounts(model.CurrentGeneration) = sum(model.
               AntiInflammatoryLattice(:));
203         this.SOCSCounts(model.CurrentGeneration) = sum(model.SOCSLattice(:))
               ;
204         this.IntermediateCounts(model.CurrentGeneration) = sum(model.
               ImmuneLattice(:) == 8);
205         this.M1Counts(model.CurrentGeneration) = sum(model.ImmuneLattice(:)
               == 4);
206         this.M2Counts(model.CurrentGeneration) = sum(model.ImmuneLattice(:)
               == 6);
207         this.TotalMacs(model.CurrentGeneration) = sum(model.ImmuneLattice(:)
                == 1) + sum(model.ImmuneLattice(:) == 4) +...
208             sum(model.ImmuneLattice(:) == 6) + sum(model.ImmuneLattice(:) ==
                    8);
209         this.AverageM1Activation(model.CurrentGeneration) = mean(mean(model.
               M1ActivationLattice));
210         this.AverageM2Activation(model.CurrentGeneration) = mean(mean(model.
               M2ActivationLattice));
211
212         % Determine the model's outcome. If ??? then the outcome is Inflamed
               .
213             if this.ImmuneCellCounts(model.CurrentGeneration) + this.
                    IntermediateCounts(model.CurrentGeneration) + ...
```

```
                    this.M1Counts(model.CurrentGeneration) + this.M2Counts(
                        model.CurrentGeneration) >= 400
                model.Outcome = Outcomes.Inflamed;
            else
                model.Outcome = Outcomes.Healthy;
            end
    end

    function reset(this)
        model = this.Model;

        % Note that the Results DO NOT get reset.
        % Set up the arrays holding the transient plot data.
        this.ImmuneCellCounts = zeros(1, model.MaxGenerations);
        this.ProInflammatoryCounts = zeros(1, model.MaxGenerations);
        this.AntiInflammatoryCounts = zeros(1, model.MaxGenerations);
        this.SOCSCounts = zeros(1, model.MaxGenerations);
        this.IntermediateCounts = zeros(1, model.MaxGenerations);
        this.M1Counts = zeros(1, model.MaxGenerations);
        this.M2Counts = zeros(1, model.MaxGenerations);
        this.TotalMacs = zeros(1, model.MaxGenerations);
        this.AverageM1Activation = zeros(1, model.MaxGenerations);
        this.AverageM2Activation = zeros(1, model.MaxGenerations);
        this.RecruitedCells = zeros(1, model.MaxGenerations);
        this.ProbRecruited = zeros(1, model.MaxGenerations);
    end

```

```
240     function moveImmuneCells(this)
241         model = this.Model;
242         %change the space to where the immune cells moves to 3. If the
243         %cell cannot move, change the space its on to 3. at the end,
244         %change all 3s to 1s.
245         temp = randi([1,9],size(model.ImmuneLattice));
246         %1 2 3
247         %4 5 6
248         %7 8 9
249         [m,n] = size(model.ImmuneLattice);
250         ii=0;
251         jj=0;
252         for i = 1:m
253             for j = 1:n
254                 if (model.ImmuneLattice(i,j) == ImmuneStates_sbm.M0Static)
                        || ...
255                         (model.ImmuneLattice(i,j) == ImmuneStates_sbm.
                            M1Static) || ...
256                         (model.ImmuneLattice(i,j) == ImmuneStates_sbm.
                            M2Static) || ...
257                         (model.ImmuneLattice(i,j) == ImmuneStates_sbm.
                            MIntStatic)
258                     %move up
259                     if temp(i,j) < 3
260                         ii = i - 1;
261                     %move down
262                     elseif temp(i,j) > 6
```

```
263                     ii = i + 1;
264                 else
265                     ii = i;
266                 end
267                 %move right
268                 if mod(temp(i,j),3) == 0
269                     jj = j + 1;
270                 %move left
271                 elseif mod(temp(i,j),3) == 1
272                     jj = j - 1;
273                 else
274                     jj = j;
275                 end
276                 %adjust values at edges to spill over to other side
277                 if ii <= 0
278                     ii = m;
279                 end
280                 if ii > m
281                    ii = 1;
282                 end
283                 if jj <= 0
284                     jj = n;
285                 end
286                 if jj > n
287                     jj = 1;
288                 end
289                 % move cells
```

```
290                 if model.ImmuneLattice(ii,jj) == ImmuneStates_sbm.Empty
291                     % move age cell
292                     model.ImmuneAge(ii,jj) = model.ImmuneAge(i,j)-1;
293                     old_age=model.ImmuneAge(i,j);
294                     model.ImmuneAge(i,j) = 0;
295                     % move SOCS
296                     model.SOCSLattice(ii,jj) = model.SOCSLattice(i,j);
297                     model.SOCSLattice(i,j) = 0;
298
299                     % update M1/M2 activation
300                     pim_fun=@(x) x^2/(x^2+this.M1ActHillParameter^2);
301                     aim_fun=@(x,hill) x^2/(x^2+hill^2);
302                     % get old M1 activation
303                     oldm1act=model.M1ActivationLattice(i,j);
304                     model.M1ActivationLattice(i,j)=0; % macrophage no
                            longer there
305                     % get old M2 activation
306                     oldm2act=model.M2ActivationLattice(i,j);
307                     model.M2ActivationLattice(i,j)=0; % macrophage no
                            longer there
308
309                     % increase M1 expression via PIM, inhibited by SOCS
310                     model.M1ActivationLattice(ii,jj)=oldm1act+min([this.
                            M1ActivationRate*pim_fun(model.
                            ProInflammatoryLattice(ii,jj))*normrnd(1,0.25)...
311                          *1/(1+(model.SOCSLattice(ii,jj)/this.
                                 M1SOCSInfinity)^2), 1-oldm1act-oldm2act]);
```

```
312
313                         % decrease M1 expression via AIM
314                         model.M1ActivationLattice(ii,jj)=max([model.
                               M1ActivationLattice(ii,jj)-this.M1DecreaseViaAIM*
                               aim_fun(model.AntiInflammatoryLattice(ii,jj),this
                               .M1DecreaseViaAIMHill), 0]);
315
316                         % increase M2 expression via AIM, inhibited by SOCS
317                         model.M2ActivationLattice(ii,jj)=oldm2act+min([this.
                               M2ActScalar*model.AntiInflammatoryLattice(ii,jj)
                               ^4/(model.AntiInflammatoryLattice(ii,jj)^4+this.
                               M2ActHillParameter^4)*normrnd(1,0.25)...
318                             *1/(1+(model.SOCSLattice(ii,jj)/this.
                                   M2SOCSInfinity)^2), 1-oldm1act-oldm2act]);
319 %                         model.M2ActivationLattice(ii,jj)=oldm2act+min([
    this.M2ActScalar*aim_fun(model.AntiInflammatoryLattice(ii,jj),this.
    M2ActHillParameter)*normrnd(1,0.25)...
320
321                         % decrease M1 & M2 expression (natural decay)
322                         model.M1ActivationLattice(ii,jj)=model.
                               M1ActivationLattice(ii,jj).*(1-this.
                               PIMNegativeFeedbackRate);
323                         model.M2ActivationLattice(ii,jj)=model.
                               M2ActivationLattice(ii,jj).*(1-this.
                               AIMNegativeFeedbackRate);
324
325                         % make old space empty
```

```
326                     model.ImmuneLattice(i,j) = ImmuneStates_sbm.Empty;
327
328                     % change new state
329
330                     % was original state M0?
331                     oldstate=model.ImmuneLattice(ii,jj) ==
                            ImmuneStates_sbm.M0Moving;
332
333                     if model.M1ActivationLattice(ii,jj)>0.5
334                         model.ImmuneLattice(ii,jj) = ImmuneStates_sbm.
                                M1Moving;
335                         if oldstate==1 % if M0 -> M1, change age to 12
                                hours
336                             model.ImmuneAge(ii,jj)=min(old_age,(round(
                                    this.AgeStDevActivated.*randn(1,1) + this
                                    .AgeMeanActivated).*60/this.
                                    GenerationSize));
337                         end
338                     elseif model.M2ActivationLattice(ii,jj)>0.5
339                         model.ImmuneLattice(ii,jj) = ImmuneStates_sbm.
                                M2Moving;
340                         if oldstate==1 % if M0 -> M2, change age to 12
                                hours
341                             model.ImmuneAge(ii,jj)=min(old_age,(round(
                                    this.AgeStDevActivated.*randn(1,1) + this
                                    .AgeMeanActivated).*60/this.
                                    GenerationSize));
```

```
342                         end
343                     elseif model.M1ActivationLattice(ii,jj)+model.
                            M2ActivationLattice(ii,jj)>0.25
344                         model.ImmuneLattice(ii,jj) = ImmuneStates_sbm.
                                MIntMoving;
345                         if oldstate==1 % if M0 -> intermediate, change
                                age to 12 hours
346                             model.ImmuneAge(ii,jj)=min(old_age,(round(
                                    this.AgeStDevActivated.*randn(1,1) + this
                                    .AgeMeanActivated).*60/this.
                                    GenerationSize));
347                         end
348                     else
349                         model.ImmuneLattice(ii,jj) = ImmuneStates_sbm.
                                M0Moving;
350                     end
351
352                 end
353             end
354         end
355     end
356
357     %finalize cells from Moving to Static
358     % M0
359     temp = model.ImmuneLattice == ImmuneStates_sbm.M0Moving;
360     model.ImmuneLattice(temp) = ImmuneStates_sbm.M0Static;
361     % M1
```

```
362         temp = model.ImmuneLattice == ImmuneStates_sbm.M1Moving;
363         model.ImmuneLattice(temp) = ImmuneStates_sbm.M1Static;
364         % M2
365         temp = model.ImmuneLattice == ImmuneStates_sbm.M2Moving;
366         model.ImmuneLattice(temp) = ImmuneStates_sbm.M2Static;
367         % intermediate
368         temp = model.ImmuneLattice == ImmuneStates_sbm.MIntMoving;
369         model.ImmuneLattice(temp) = ImmuneStates_sbm.MIntStatic;
370         %kill cells
371         tempDeath = (model.ImmuneAge <= 0);
372         model.ImmuneLattice(tempDeath) = ImmuneStates_sbm.Empty;
373         model.M1ActivationLattice(tempDeath) = 0;
374         model.M2ActivationLattice(tempDeath) = 0;
375         model.SOCSLattice(tempDeath) = 0;
376         model.ImmuneAge(tempDeath) = 0;
377     end
378
379     function finalize(this)
380         model = this.Model;
381         % Append rows of data to matrices for each outcome.
382         this.Results{model.Outcome}.Runs = this.Results{model.Outcome}.Runs
                + 1;
383
384         this.Results{model.Outcome}.ImmuneCellCounts = this.Results{model.
                Outcome}.ImmuneCellCounts + model.Rules{1}.Results{model.Outcome
                }.ImmuneCellCounts;
```

```
385         this.Results{model.Outcome}.ImmuneCellCountsSquared = this.Results{
               model.Outcome}.ImmuneCellCountsSquared + (model.Rules{1}.Results{
               model.Outcome}.ImmuneCellCounts .^ 2);
386         this.Results{model.Outcome}.IntermediateCounts = this.Results{model.
               Outcome}.IntermediateCounts + model.Rules{1}.Results{model.
               Outcome}.IntermediateCounts;
387         this.Results{model.Outcome}.IntermediateCountsSquared = this.Results
               {model.Outcome}.IntermediateCountsSquared + (model.Rules{1}.
               Results{model.Outcome}.IntermediateCounts .^ 2);
388         this.Results{model.Outcome}.M1Counts = this.Results{model.Outcome}.
               M1Counts + model.Rules{1}.Results{model.Outcome}.M1Counts;
389         this.Results{model.Outcome}.M1CountsSquared = this.Results{model.
               Outcome}.M1CountsSquared + (model.Rules{1}.Results{model.Outcome
               }.M1Counts .^ 2);
390         this.Results{model.Outcome}.M2Counts = this.Results{model.Outcome}.
               M2Counts + model.Rules{1}.Results{model.Outcome}.M2Counts;
391         this.Results{model.Outcome}.M2CountsSquared = this.Results{model.
               Outcome}.M2CountsSquared + (model.Rules{1}.Results{model.Outcome
               }.M2Counts .^ 2);
392         this.Results{model.Outcome}.TotalMacs = this.Results{model.Outcome}.
               TotalMacs + model.Rules{1}.Results{model.Outcome}.TotalMacs;
393         this.Results{model.Outcome}.TotalMacsSquared = this.Results{model.
               Outcome}.TotalMacsSquared + (model.Rules{1}.Results{model.Outcome
               }.TotalMacs .^ 2);
394         this.Results{model.Outcome}.ProInflammatoryCounts = this.Results{
               model.Outcome}.ProInflammatoryCounts + model.Rules{1}.Results{
               model.Outcome}.ProInflammatoryCounts;
```

```
395         this.Results{model.Outcome}.ProInflammatoryCountsSquared = this.
               Results{model.Outcome}.ProInflammatoryCountsSquared + (model.
               Rules{1}.Results{model.Outcome}.ProInflammatoryCounts .^ 2);
396         this.Results{model.Outcome}.AntiInflammatoryCounts = this.Results{
               model.Outcome}.AntiInflammatoryCounts + model.Rules{1}.Results{
               model.Outcome}.AntiInflammatoryCounts;
397         this.Results{model.Outcome}.AntiInflammatoryCountsSquared = this.
               Results{model.Outcome}.AntiInflammatoryCountsSquared + (model.
               Rules{1}.Results{model.Outcome}.AntiInflammatoryCounts .^ 2);
398         this.Results{model.Outcome}.SOCSCounts = this.Results{model.Outcome
               }.SOCSCounts + model.Rules{1}.Results{model.Outcome}.SOCSCounts;
399         this.Results{model.Outcome}.SOCSCountsSquared = this.Results{model.
               Outcome}.SOCSCountsSquared + (model.Rules{1}.Results{model.
               Outcome}.SOCSCounts .^ 2);
400         this.Results{model.Outcome}.AverageM1Activation = this.Results{model
               .Outcome}.AverageM1Activation + model.Rules{1}.Results{model.
               Outcome}.AverageM1Activation;
401         this.Results{model.Outcome}.AverageM1ActivationSquared = this.
               Results{model.Outcome}.AverageM1ActivationSquared + (model.Rules
               {1}.Results{model.Outcome}.AverageM1Activation.^ 2);
402         this.Results{model.Outcome}.AverageM2Activation = this.Results{model
               .Outcome}.AverageM2Activation + model.Rules{1}.Results{model.
               Outcome}.AverageM2Activation;
403         this.Results{model.Outcome}.AverageM2ActivationSquared = this.
               Results{model.Outcome}.AverageM2ActivationSquared + (model.Rules
               {1}.Results{model.Outcome}.AverageM2Activation.^ 2);
```

```
404         this.Results{model.Outcome}.RecruitedCells = this.Results{model.
                Outcome}.RecruitedCells + model.Rules{1}.Results{model.Outcome}.
                RecruitedCells;
405         this.Results{model.Outcome}.RecruitedCellsSquared = this.Results{
                model.Outcome}.RecruitedCellsSquared + (model.Rules{1}.Results{
                model.Outcome}.RecruitedCells.^ 2);
406         this.Results{model.Outcome}.ProbRecruited = this.Results{model.
                Outcome}.ProbRecruited + model.Rules{1}.Results{model.Outcome}.
                ProbRecruited;
407         this.Results{model.Outcome}.ProbRecruitedSquared = this.Results{
                model.Outcome}.ProbRecruitedSquared + (model.Rules{1}.Results{
                model.Outcome}.ProbRecruited.^ 2);
408     end
409 end
410 end
```

B.1.4 Outcomes.m

```
1 classdef Outcomes < uint32
2     enumeration
3         Healthy  (1)
4         Inflamed (2)
5     end
6 end
```

B.1.5 ImmuneStates_sbm.m

```
classdef ImmuneStates_sbm < uint32
    enumeration
        M0Static      (1)
        Empty         (2)
        M0Moving      (3)
        M1Static      (4)
        M1Moving      (5)
        M2Static      (6)
        M2Moving      (7)
        MIntStatic    (8) % intermediate: between M1/M2
        MIntMoving    (9) % intermediate: between M1/M2
    end
end
```

B.1.6 LungModelRule.m

```
classdef LungModelRule < handle
    properties (Abstract)
        Model;
    end

    methods (Abstract, Static)
        apply_rule(this);
        finalize(this);
    end

    methods
```

```
12          function nbhd = neighborhood_count(this, mat, state)
13              % Count the number of cells that match a given state in each
14              % cell's Moore neighborhood.
15              [m, n] = size(mat);
16
17              up = [2:m 1];
18              down = [m 1:m-1];
19              left = [2:n 1];
20              right = [n 1:n-1];
21
22              nbhd = (mat(up, :) == state) + ...
23                     (mat(up, left) == state) + ...
24                     (mat(up, right) == state) + ...
25                     (mat(down, :) == state) + ...
26                     (mat(down, left) == state) + ...
27                     (mat(down, right) == state) + ...
28                     (mat(:, left) == state) + ...
29                     (mat(:, right) == state);
30          end
31
32          function diffused = diffuse(mat)
33              % Given a matrix containing a fluid intensity (0 to 1),
34              % diffuse the fluid across the matrix evenly.
35              [m, n] = size(mat);
36
37              up = [2:m 1];
38              down = [m 1:m-1];
```

```
            left = [2:n 1];
            right = [n 1:n-1];

            diffused = 1/9 * (mat(up, left) + ...
                       mat(up, :) + ...
                       mat(up, right) + ...
                       mat(:, left) + ...
                       mat(:, :) + ...
                       mat(:, right) + ...
                       mat(down, left) + ...
                       mat(down, :) + ...
                       mat(down, right));
        end
    end
end
```

B.1.7 diffuse.m

```
function diffused = diffuse(mat)
            % Given a matrix containing a fluid intensity (0 to 1),
            % diffuse the fluid across the matrix evenly.
            [m, n] = size(mat);

            up = [2:m 1];
            down = [m 1:m-1];
            left = [2:n 1];
            right = [n 1:n-1];
```

```
10
11              diffused = 1/9 * (mat(up, left) + ...
12                               mat(up, :) + ...
13                               mat(up, right) + ...
14                               mat(:, left) + ...
15                               mat(:, :) + ...
16                               mat(:, right) + ...
17                               mat(down, left) + ...
18                               mat(down, :) + ...
19                               mat(down, right));
20          end
```

B.1.8 boundedline.m

```
1  function varargout = boundedline(varargin)
2  %BOUNDEDLINE Plot a line with shaded error/confidence bounds
3  % Created by Kelly Kearney
4  % https://github.com/kakearney/boundedline-pkg
5  %
6  % [hl, hp] = boundedline(x, y, b)
7  % [hl, hp] = boundedline(x, y, b, linespec)
8  % [hl, hp] = boundedline(x1, y1, b1, linespec1,  x2, y2, b2, linespec2)
9  % [hl, hp] = boundedline(..., 'alpha')
10 % [hl, hp] = boundedline(..., ax)
11 % [hl, hp] = boundedline(..., 'transparency', trans)
12 % [hl, hp] = boundedline(..., 'orientation', orient)
13 % [hl, hp] = boundedline(..., 'cmap', cmap)
```

```
14 %
15 % Input variables:
16 %
17 %   x, y:       x and y values, either vectors of the same length, matrices
18 %               of the same size, or vector/matrix pair where the row or
19 %               column size of the array matches the length of the vector
20 %               (same requirements as for plot function).
21 %
22 %   b:          npoint x nsize x nline array.  Distance from line to
23 %               boundary, for each point along the line (dimension 1), for
24 %               each side of the line (lower/upper or left/right, depending
25 %               on orientation) (dimension 2), and for each plotted line
26 %               described by the preceding x–y values (dimension 3).  If
27 %               size(b,1) == 1, the bounds will be the same for all points
28 %               along the line.  If size(b,2) == 1, the bounds will be
29 %               symmetrical on both sides of the lines.  If size(b,3) == 1,
30 %               the same bounds will be applied to all lines described by
31 %               the preceding x–y arrays (only applicable when either x or
32 %               y is an array).  Bounds cannot include Inf, –Inf, or NaN,
33 %
34 %   linespec:   line specification that determines line type, marker
35 %               symbol, and color of the plotted lines for the preceding
36 %               x–y values.
37 %
38 %   'alpha':    if included, the bounded area will be rendered with a
39 %               partially–transparent patch the same color as the
40 %               corresponding line(s).  If not included, the bounded area
```

```
41 %               will be an opaque patch with a lighter shade of the
42 %               corresponding line color.
43 %
44 %   ax:         handle of axis where lines will be plotted.  If not
45 %               included, the current axis will be used.
46 %
47 %   transp:     Scalar between 0 and 1 indicating with the transparency or
48 %               intensity of color of the bounded area patch. Default is
49 %               0.2.
50 %
51 %   orient:     'vert': add bounds in vertical (y) direction (default)
52 %               'horiz': add bounds in horizontal (x) direction
53 %
54 %   cmap:       n x 3 colormap array.  If included, lines will be colored
55 %               (in order of plotting) according to this colormap,
56 %               overriding any linespec or default colors.
57 %
58 % Output variables:
59 %
60 %   hl:         handles to line objects
61 %
62 %   hp:         handles to patch objects
63 %
64 % Example:
65 %
66 % x = linspace(0, 2*pi, 50);
67 % y1 = sin(x);
```

```
% y2 = cos(x);
% e1 = rand(size(y1))*.5+.5;
% e2 = [.25 .5];
%
% ax(1) = subplot(2,2,1);
% [l,p] = boundedline(x, y1, e1, '-b*', x, y2, e2, '--ro');
% outlinebounds(l,p);
% title('Opaque bounds, with outline');
%
% ax(2) = subplot(2,2,2);
% boundedline(x, [y1;y2], rand(length(y1),2,2)*.5+.5, 'alpha');
% title('Transparent bounds');
%
% ax(3) = subplot(2,2,3);
% boundedline([y1;y2], x, e1(1), 'orientation', 'horiz')
% title('Horizontal bounds');
%
% ax(4) = subplot(2,2,4);
% boundedline(x, repmat(y1, 4,1), permute(0.5:-0.1:0.2, [3 1 2]), ...
%             'cmap', cool(4), 'transparency', 0.5);
% title('Multiple bounds using colormap');


% Copyright 2010 Kelly Kearney

%---------------------
% Parse input
```

```
95  %--------------------
96
97  % Alpha flag
98
99  isalpha = cellfun(@(x) ischar(x) && strcmp(x, 'alpha'), varargin);
100 if any(isalpha)
101     usealpha = true;
102     varargin = varargin(~isalpha);
103 else
104     usealpha = false;
105 end
106
107 % Axis
108
109 isax = cellfun(@(x) isscalar(x) && ishandle(x) && strcmp('axes', get(x,'type
        ')), varargin);
110 if any(isax)
111     hax = varargin{isax};
112     varargin = varargin(~isax);
113 else
114     hax = gca;
115 end
116
117 % Transparency
118
119 [found, trans, varargin] = parseparam(varargin, 'transparency');
120
```

```
121 if ~found
122     trans = 0.2;
123 end
124 
125 if ~isscalar(trans) || trans < 0 || trans > 1
126     error('Transparency must be scalar between 0 and 1');
127 end
128 
129 % Orientation
130 
131 [found, orient, varargin] = parseparam(varargin, 'orientation');
132 
133 if ~found
134     orient = 'vert';
135 end
136 
137 if strcmp(orient, 'vert')
138     isvert = true;
139 elseif strcmp(orient, 'horiz')
140     isvert = false;
141 else
142     error('Orientation must be ''vert'' or ''horiz''');
143 end
144 
145 
146 % Colormap
147 
```

```
148 [hascmap, cmap, varargin] = parseparam(varargin, 'cmap');
149
150 %Linewidth;
151 [found, width, varargin] = parseparam(varargin, 'linewidth');
152 if ~found
153 width = 1;
154 end
155
156
157 % X, Y, E triplets, and linespec
158
159 [x,y,err,linespec] = deal(cell(0));
160 while ~isempty(varargin)
161     if length(varargin) < 3
162         error('Unexpected input: should be x, y, bounds triplets');
163     end
164     if all(cellfun(@isnumeric, varargin(1:3)))
165         x = [x varargin(1)];
166         y = [y varargin(2)];
167         err = [err varargin(3)];
168         varargin(1:3) = [];
169     else
170         error('Unexpected input: should be x, y, bounds triplets');
171     end
172     if ~isempty(varargin) && ischar(varargin{1})
173         linespec = [linespec varargin(1)];
174         varargin(1) = [];
```

```
175     else
176         linespec = [linespec {[]}];
177     end
178 end
179 
180 %---------------------
181 % Reformat x and y
182 % for line and patch
183 % plotting
184 %---------------------
185 
186 % Calculate y values for bounding lines
187 
188 plotdata = cell(0,7);
189 
190 htemp = figure('visible', 'off');
191 for ix = 1:length(x)
192 
193     % Get full x, y, and linespec data for each line (easier to let plot
194     % check for properly-sized x and y and expand values than to try to do
195     % it myself)
196 
197     try
198         if isempty(linespec{ix})
199             hltemp = plot(x{ix}, y{ix});
200         else
201             hltemp = plot(x{ix}, y{ix}, linespec{ix});
```

```
202         end
203     catch
204         close(htemp);
205         error('X and Y matrices and/or linespec not appropriate for line
                plot');
206     end
207
208     linedata = get(hltemp, {'xdata', 'ydata', 'marker', 'linestyle', 'color'
            });
209
210     nline = size(linedata,1);
211
212     % Expand bounds matrix if necessary
213
214     if nline > 1
215         if ndims(err{ix}) == 3
216             err2 = squeeze(num2cell(err{ix},[1 2]));
217         else
218             err2 = repmat(err(ix),nline,1);
219         end
220     else
221         err2 = err(ix);
222     end
223
224     % Figure out upper and lower bounds
225
226     [lo, hi] = deal(cell(nline,1));
```

```
227     for iln = 1:nline
228
229         x2 = linedata{iln,1};
230         y2 = linedata{iln,2};
231         nx = length(x2);
232
233         if isvert
234             lineval = y2;
235         else
236             lineval = x2;
237         end
238
239         sz = size(err2{iln});
240
241         if isequal(sz, [nx 2])
242             lo{iln} = lineval - err2{iln}(:,1)';
243             hi{iln} = lineval + err2{iln}(:,2)';
244         elseif isequal(sz, [nx 1])
245             lo{iln} = lineval - err2{iln}';
246             hi{iln} = lineval + err2{iln}';
247         elseif isequal(sz, [1 2])
248             lo{iln} = lineval - err2{iln}(1);
249             hi{iln} = lineval + err2{iln}(2);
250         elseif isequal(sz, [1 1])
251             lo{iln} = lineval - err2{iln};
252             hi{iln} = lineval + err2{iln};
253         elseif isequal(sz, [2 nx]) % not documented, but accepted anyways
```

```
254             lo{iln} = lineval - err2{iln}(:,1);
255             hi{iln} = lineval + err2{iln}(:,2);
256         elseif isequal(sz, [1 nx]) % not documented, but accepted anyways
257             lo{iln} = lineval - err2{iln};
258             hi{iln} = lineval + err2{iln};
259         elseif isequal(sz, [2 1]) % not documented, but accepted anyways
260             lo{iln} = lineval - err2{iln}(1);
261             hi{iln} = lineval + err2{iln}(2);
262         else
263             error('Error bounds must be npt x nside x nline array');
264         end
265 
266     end
267 
268     % Combine all data (xline, yline, marker, linestyle, color, lower bound
269     % (x or y), upper bound (x or y)
270 
271     plotdata = [plotdata; linedata lo hi];
272 
273 end
274 close(htemp);
275 
276 % Override colormap
277 
278 if hascmap
279     nd = size(plotdata,1);
280     cmap = repmat(cmap, ceil(nd/size(cmap,1)), 1);
```

```
281     cmap = cmap(1:nd,:);
282     plotdata(:,5) = num2cell(cmap,2);
283 end
284 
285 
286 %---------------------
287 % Plot
288 %---------------------
289 
290 % Setup of x and y, plus line and patch properties
291 
292 nline = size(plotdata,1);
293 [xl, yl, xp, yp, marker, lnsty, lncol, ptchcol, alpha] = deal(cell(nline,1))
        ;
294 
295 for iln = 1:nline
296     xl{iln} = plotdata{iln,1};
297     yl{iln} = plotdata{iln,2};
298 %     if isvert
299 %         xp{iln} = [plotdata{iln,1} fliplr(plotdata{iln,1})];
300 %         yp{iln} = [plotdata{iln,6} fliplr(plotdata{iln,7})];
301 %     else
302 %         xp{iln} = [plotdata{iln,6} fliplr(plotdata{iln,7})];
303 %         yp{iln} = [plotdata{iln,2} fliplr(plotdata{iln,2})];
304 %     end
305 
```

```
306     [xp{iln}, yp{iln}] = calcpatch(plotdata{iln,1}, plotdata{iln,2}, isvert,
            plotdata{iln,6}, plotdata{iln,7});
307
308     marker{iln} = plotdata{iln,3};
309     lnsty{iln} = plotdata{iln,4};
310
311     if usealpha
312         lncol{iln} = plotdata{iln,5};
313         ptchcol{iln} = plotdata{iln,5};
314         alpha{iln} = trans;
315     else
316         lncol{iln} = plotdata{iln,5};
317         ptchcol{iln} = interp1([0 1], [1 1 1; lncol{iln}], trans);
318         alpha{iln} = 1;
319     end
320 end
321
322 % Plot patches and lines
323
324 [hp,hl] = deal(zeros(nline,1));
325
326 axes(hax);
327 hold all;
328
329 for iln = 1:nline
330     hp(iln) = patch(xp{iln}, yp{iln}, ptchcol{iln}, 'facealpha', alpha{iln},
            'edgecolor', 'none');
```

```
end

for iln = 1:nline
    hl(iln) = line(xl{iln}, yl{iln}, 'marker', marker{iln}, 'linestyle',
        lnsty{iln}, 'color', lncol{iln}, 'linewidth', width);
end

%---------------------
% Assign output
%---------------------

nargchk(0, 2, nargout);

if nargout >= 1
    varargout{1} = hl;
end

if nargout == 2
    varargout{2} = hp;
end

%---------------------
% Parse optional
% parameters
%---------------------

function [found, val, vars] = parseparam(vars, param)
```

```
357
358 isvar = cellfun(@(x) ischar(x) && strcmpi(x, param), vars);
359
360 if sum(isvar) > 1
361     error('Parameters can only be passed once');
362 end
363
364 if any(isvar)
365     found = true;
366     idx = find(isvar);
367     val = vars{idx+1};
368     vars([idx idx+1]) = [];
369 else
370     found = false;
371     val = [];
372 end
373
374 %------------------------------
375 % Calculate patch coordinates
376 %------------------------------
377
378 function [xp, yp] = calcpatch(xl, yl, isvert, lo, hi)
379
380 ismissing = any(isnan([xl;yl;lo;hi]),2);
381 if any(ismissing)
382
383 else
```

```
    if isvert
        xp = [xl fliplr(xl)];
        yp = [lo fliplr(hi)];
    else
        xp = [lo fliplr(hi)];
        yp = [yl fliplr(yl)];
    end
end
```

B.2 Code: ODE model

The following function contains the equations for the ten-macrophage ODE model described in Section 4.2.1.

```
function dx=rhs_crosstalk_compartmental(~,x,param,scale_flag)
% x, param: each column is a different compartment

if scale_flag==1
    scale=3600;
else
    scale=1;
end

%-------------- nominal parameter value --------------------
%%%% shared parameters
kdeg_tnfa =    param(1)*scale;
muile =        param(2)*scale;

```

```
%%%% compartmental parameters
a_trans_1 =      param(3)*scale;
a_trans_2 =      param(52+3)*scale;
a_trans_3 =      param(2*52+3)*scale;
a_trans_4 =      param(3*52+3)*scale;
a_trans_5 =      param(4*52+3)*scale;
a_trans_6 =      param(5*52+3)*scale;
a_trans_7 =      param(6*52+3)*scale;
a_trans_8 =      param(7*52+3)*scale;
a_trans_9 =      param(8*52+3)*scale;
a_trans_10 =     param(9*52+3)*scale;
ctf_1 =          param(4);
ctf_2 =          param(52+4);
ctf_3 =          param(2*52+4);
ctf_4 =          param(3*52+4);
ctf_5 =          param(4*52+4);
ctf_6 =          param(5*52+4);
ctf_7 =          param(6*52+4);
ctf_8 =          param(7*52+4);
ctf_9 =          param(8*52+4);
ctf_10 =         param(9*52+4);
ctf_stat3_1 =    param(5);
ctf_stat3_2 =    param(52+5);
ctf_stat3_3 =    param(2*52+5);
ctf_stat3_4 =    param(3*52+5);
ctf_stat3_5 =    param(4*52+5);
ctf_stat3_6 =    param(5*52+5);
```

```
ctf_stat3_7 =    param(6*52+5);
ctf_stat3_8 =    param(7*52+5);
ctf_stat3_9 =    param(8*52+5);
ctf_stat3_10 =   param(9*52+5); % 473
eki_1 =          param(6)*scale;
eki_2 =          param(52+6)*scale;
eki_3 =          param(2*52+6)*scale;
eki_4 =          param(3*52+6)*scale;
eki_5 =          param(4*52+6)*scale;
eki_6 =          param(5*52+6)*scale;
eki_7 =          param(6*52+6)*scale;
eki_8 =          param(7*52+6)*scale;
eki_9 =          param(8*52+6)*scale;
eki_10 =         param(9*52+6)*scale;
eni_1 =          param(7)*scale;
eni_2 =          param(52+7)*scale;
eni_3 =          param(2*52+7)*scale;
eni_4 =          param(3*52+7)*scale;
eni_5 =          param(4*52+7)*scale;
eni_6 =          param(5*52+7)*scale;
eni_7 =          param(6*52+7)*scale;
eni_8 =          param(7*52+7)*scale;
eni_9 =          param(8*52+7)*scale;
eni_10 =         param(9*52+7)*scale;
ikba_trans_1 =   param(8)*scale;
ikba_trans_2 =   param(52+8)*scale;
ikba_trans_3 =   param(2*52+8)*scale;
```

```
ikba_trans_4 =   param(3*52+8)*scale;
ikba_trans_5 =   param(4*52+8)*scale;
ikba_trans_6 =   param(5*52+8)*scale;
ikba_trans_7 =   param(6*52+8)*scale;
ikba_trans_8 =   param(7*52+8)*scale;
ikba_trans_9 =   param(8*52+8)*scale;
ikba_trans_10 =  param(9*52+8)*scale;
iki_1 =          param(9)*scale;
iki_2 =          param(52+9)*scale;
iki_3 =          param(2*52+9)*scale;
iki_4 =          param(3*52+9)*scale;
iki_5 =          param(4*52+9)*scale;
iki_6 =          param(5*52+9)*scale;
iki_7 =          param(6*52+9)*scale;
iki_8 =          param(7*52+9)*scale;
iki_9 =          param(8*52+9)*scale;
iki_10 =         param(9*52+9)*scale;
il10max_1 =      param(10);
il10max_2 =      param(52+10);
il10max_3 =      param(2*52+10);
il10max_4 =      param(3*52+10);
il10max_5 =      param(4*52+10);
il10max_6 =      param(5*52+10);
il10max_7 =      param(6*52+10);
il10max_8 =      param(7*52+10);
il10max_9 =      param(8*52+10);
il10max_10 =     param(9*52+10);
```

```
iln_1 =          param(11)*scale;
iln_2 =          param(52+11)*scale;
iln_3 =          param(2*52+11)*scale;
iln_4 =          param(3*52+11)*scale;
iln_5 =          param(4*52+11)*scale;
iln_6 =          param(5*52+11)*scale;
iln_7 =          param(6*52+11)*scale;
iln_8 =          param(7*52+11)*scale;
iln_9 =          param(8*52+11)*scale;
iln_10 =         param(9*52+11)*scale;
kdeg_a20_1 =     param(12)*scale;
kdeg_a20_2 =     param(52+12)*scale;
kdeg_a20_3 =     param(2*52+12)*scale;
kdeg_a20_4 =     param(3*52+12)*scale;
kdeg_a20_5 =     param(4*52+12)*scale;
kdeg_a20_6 =     param(5*52+12)*scale;
kdeg_a20_7 =     param(6*52+12)*scale;
kdeg_a20_8 =     param(7*52+12)*scale;
kdeg_a20_9 =     param(8*52+12)*scale;
kdeg_a20_10 =    param(9*52+12)*scale;
kdeg_ikba_1 =    param(13)*scale;
kdeg_ikba_2 =    param(52+13)*scale;
kdeg_ikba_3 =    param(2*52+13)*scale;
kdeg_ikba_4 =    param(3*52+13)*scale;
kdeg_ikba_5 =    param(4*52+13)*scale;
kdeg_ikba_6 =    param(5*52+13)*scale;
kdeg_ikba_7 =    param(6*52+13)*scale;
```

```
kdeg_ikba_8 =    param(7*52+13)*scale;
kdeg_ikba_9 =    param(8*52+13)*scale;
kdeg_ikba_10 =   param(9*52+13)*scale;
kf1_1 =          param(14)*scale;
kf1_2 =          param(52+14)*scale;
kf1_3 =          param(2*52+14)*scale;
kf1_4 =          param(3*52+14)*scale;
kf1_5 =          param(4*52+14)*scale;
kf1_6 =          param(5*52+14)*scale;
kf1_7 =          param(6*52+14)*scale;
kf1_8 =          param(7*52+14)*scale;
kf1_9 =          param(8*52+14)*scale;
kf1_10 =         param(9*52+14)*scale;
kf3_1 =          param(15)*scale;
kf3_2 =          param(52+15)*scale;
kf3_3 =          param(2*52+15)*scale;
kf3_4 =          param(3*52+15)*scale;
kf3_5 =          param(4*52+15)*scale;
kf3_6 =          param(5*52+15)*scale;
kf3_7 =          param(6*52+15)*scale;
kf3_8 =          param(7*52+15)*scale;
kf3_9 =          param(8*52+15)*scale;
kf3_10 =         param(9*52+15)*scale;
kf4_1 =          param(16)*scale;
kf4_2 =          param(52+16)*scale;
kf4_3 =          param(2*52+16)*scale;
kf4_4 =          param(3*52+16)*scale;
```

```
kf4_5 =          param(4*52+16)*scale;
kf4_6 =          param(5*52+16)*scale;
kf4_7 =          param(6*52+16)*scale;
kf4_8 =          param(7*52+16)*scale;
kf4_9 =          param(8*52+16)*scale;
kf4_10 =         param(9*52+16)*scale;
kfi_1 =          param(17)*scale;
kfi_2 =          param(52+17)*scale;
kfi_3 =          param(2*52+17)*scale;
kfi_4 =          param(3*52+17)*scale;
kfi_5 =          param(4*52+17)*scale;
kfi_6 =          param(5*52+17)*scale;
kfi_7 =          param(6*52+17)*scale;
kfi_8 =          param(7*52+17)*scale;
kfi_9 =          param(8*52+17)*scale;
kfi_10 =         param(9*52+17)*scale;
kilc_1 =         param(18)*scale;
kilc_2 =         param(52+18)*scale;
kilc_3 =         param(2*52+18)*scale;
kilc_4 =         param(3*52+18)*scale;
kilc_5 =         param(4*52+18)*scale;
kilc_6 =         param(5*52+18)*scale;
kilc_7 =         param(6*52+18)*scale;
kilc_8 =         param(7*52+18)*scale;
kilc_9 =         param(8*52+18)*scale;
kilc_10 =        param(9*52+18)*scale;
kiljb_1 =        param(19)*scale;
```

```
kiljb_2 =        param(52+19)*scale;
kiljb_3 =        param(2*52+19)*scale;
kiljb_4 =        param(3*52+19)*scale;
kiljb_5 =        param(4*52+19)*scale;
kiljb_6 =        param(5*52+19)*scale;
kiljb_7 =        param(6*52+19)*scale;
kiljb_8 =        param(7*52+19)*scale;
kiljb_9 =        param(8*52+19)*scale;
kiljb_10 =       param(9*52+19)*scale;
kilju_1 =        param(20)*scale;
kilju_2 =        param(52+20)*scale;
kilju_3 =        param(2*52+20)*scale;
kilju_4 =        param(3*52+20)*scale;
kilju_5 =        param(4*52+20)*scale;
kilju_6 =        param(5*52+20)*scale;
kilju_7 =        param(6*52+20)*scale;
kilju_8 =        param(7*52+20)*scale;
kilju_9 =        param(8*52+20)*scale;
kilju_10 =       param(9*52+20)*scale;
kilm_1 =         param(21)*scale;
kilm_2 =         param(52+21)*scale;
kilm_3 =         param(2*52+21)*scale;
kilm_4 =         param(3*52+21)*scale;
kilm_5 =         param(4*52+21)*scale;
kilm_6 =         param(5*52+21)*scale;
kilm_7 =         param(6*52+21)*scale;
kilm_8 =         param(7*52+21)*scale;
```

```
kilm_9 =         param(8*52+21)*scale;
kilm_10 =        param(9*52+21)*scale;
kilnf_1 =        param(22)*scale;
kilnf_2 =        param(52+22)*scale;
kilnf_3 =        param(2*52+22)*scale;
kilnf_4 =        param(3*52+22)*scale;
kilnf_5 =        param(4*52+22)*scale;
kilnf_6 =        param(5*52+22)*scale;
kilnf_7 =        param(6*52+22)*scale;
kilnf_8 =        param(7*52+22)*scale;
kilnf_9 =        param(8*52+22)*scale;
kilnf_10 =       param(9*52+22)*scale;
kilrb_1 =        param(23)*scale;
kilrb_2 =        param(52+23)*scale;
kilrb_3 =        param(2*52+23)*scale;
kilrb_4 =        param(3*52+23)*scale;
kilrb_5 =        param(4*52+23)*scale;
kilrb_6 =        param(5*52+23)*scale;
kilrb_7 =        param(6*52+23)*scale;
kilrb_8 =        param(7*52+23)*scale;
kilrb_9 =        param(8*52+23)*scale;
kilrb_10 =       param(9*52+23)*scale;
kilru_1 =        param(24)*scale;
kilru_2 =        param(52+24)*scale;
kilru_3 =        param(2*52+24)*scale;
kilru_4 =        param(3*52+24)*scale;
kilru_5 =        param(4*52+24)*scale;
```

```
kilru_6 =        param(5*52+24)*scale;
kilru_7 =        param(6*52+24)*scale;
kilru_8 =        param(7*52+24)*scale;
kilru_9 =        param(8*52+24)*scale;
kilru_10 =       param(9*52+24)*scale;
kilsn_1 =        param(25)*scale;
kilsn_2 =        param(52+25)*scale;
kilsn_3 =        param(2*52+25)*scale;
kilsn_4 =        param(3*52+25)*scale;
kilsn_5 =        param(4*52+25)*scale;
kilsn_6 =        param(5*52+25)*scale;
kilsn_7 =        param(6*52+25)*scale;
kilsn_8 =        param(7*52+25)*scale;
kilsn_9 =        param(8*52+25)*scale;
kilsn_10 =       param(9*52+25)*scale;
kk1_1 =          param(26)*scale;
kk1_2 =          param(52+26)*scale;
kk1_3 =          param(2*52+26)*scale;
kk1_4 =          param(3*52+26)*scale;
kk1_5 =          param(4*52+26)*scale;
kk1_6 =          param(5*52+26)*scale;
kk1_7 =          param(6*52+26)*scale;
kk1_8 =          param(7*52+26)*scale;
kk1_9 =          param(8*52+26)*scale;
kk1_10 =         param(9*52+26)*scale;
kk3_1 =          param(27)*scale;
kk3_2 =          param(52+27)*scale;
```

```
kk3_3 =          param(2*52+27)*scale;
kk3_4 =          param(3*52+27)*scale;
kk3_5 =          param(4*52+27)*scale;
kk3_6 =          param(5*52+27)*scale;
kk3_7 =          param(6*52+27)*scale;
kk3_8 =          param(7*52+27)*scale;
kk3_9 =          param(8*52+27)*scale;
kk3_10 =         param(9*52+27)*scale;
kr1_1 =          param(28)*scale;
kr1_2 =          param(52+28)*scale;
kr1_3 =          param(2*52+28)*scale;
kr1_4 =          param(3*52+28)*scale;
kr1_5 =          param(4*52+28)*scale;
kr1_6 =          param(5*52+28)*scale;
kr1_7 =          param(6*52+28)*scale;
kr1_8 =          param(7*52+28)*scale;
kr1_9 =          param(8*52+28)*scale;
kr1_10 =         param(9*52+28)*scale;
kr3_1 =          param(29)*scale;
kr3_2 =          param(52+29)*scale;
kr3_3 =          param(2*52+29)*scale;
kr3_4 =          param(3*52+29)*scale;
kr3_5 =          param(4*52+29)*scale;
kr3_6 =          param(5*52+29)*scale;
kr3_7 =          param(6*52+29)*scale;
kr3_8 =          param(7*52+29)*scale;
kr3_9 =          param(8*52+29)*scale;
```

```
kr3_10 =        param(9*52+29)*scale;
ks1_1 =         param(30)*scale;
ks1_2 =         param(52+30)*scale;
ks1_3 =         param(2*52+30)*scale;
ks1_4 =         param(3*52+30)*scale;
ks1_5 =         param(4*52+30)*scale;
ks1_6 =         param(5*52+30)*scale;
ks1_7 =         param(6*52+30)*scale;
ks1_8 =         param(7*52+30)*scale;
ks1_9 =         param(8*52+30)*scale;
ks1_10 =        param(9*52+30)*scale;
ks1st_1 =       param(31)*scale;
ks1st_2 =       param(52+31)*scale;
ks1st_3 =       param(2*52+31)*scale;
ks1st_4 =       param(3*52+31)*scale;
ks1st_5 =       param(4*52+31)*scale;
ks1st_6 =       param(5*52+31)*scale;
ks1st_7 =       param(6*52+31)*scale;
ks1st_8 =       param(7*52+31)*scale;
ks1st_9 =       param(8*52+31)*scale;
ks1st_10 =      param(9*52+31)*scale;
ks3_1 =         param(32)*scale;
ks3_2 =         param(52+32)*scale;
ks3_3 =         param(2*52+32)*scale;
ks3_4 =         param(3*52+32)*scale;
ks3_5 =         param(4*52+32)*scale;
ks3_6 =         param(5*52+32)*scale;
```

```
ks3_7 =          param(6*52+32)*scale;
ks3_8 =          param(7*52+32)*scale;
ks3_9 =          param(8*52+32)*scale;
ks3_10 =         param(9*52+32)*scale;
ks3st_1 =        param(33)*scale;
ks3st_2 =        param(52+33)*scale;
ks3st_3 =        param(2*52+33)*scale;
ks3st_4 =        param(3*52+33)*scale;
ks3st_5 =        param(4*52+33)*scale;
ks3st_6 =        param(5*52+33)*scale;
ks3st_7 =        param(6*52+33)*scale;
ks3st_8 =        param(7*52+33)*scale;
ks3st_9 =        param(8*52+33)*scale;
ks3st_10 =       param(9*52+33)*scale;
ksa_1 =          param(34)*scale;
ksa_2 =          param(52+34)*scale;
ksa_3 =          param(2*52+34)*scale;
ksa_4 =          param(3*52+34)*scale;
ksa_5 =          param(4*52+34)*scale;
ksa_6 =          param(5*52+34)*scale;
ksa_7 =          param(6*52+34)*scale;
ksa_8 =          param(7*52+34)*scale;
ksa_9 =          param(8*52+34)*scale;
ksa_10 =         param(9*52+34)*scale;
ksec_1 =         param(35)*scale;
ksec_2 =         param(52+35)*scale;
ksec_3 =         param(2*52+35)*scale;
```

```
ksec_4 =         param(3*52+35)*scale;
ksec_5 =         param(4*52+35)*scale;
ksec_6 =         param(5*52+35)*scale;
ksec_7 =         param(6*52+35)*scale;
ksec_8 =         param(7*52+35)*scale;
ksec_9 =         param(8*52+35)*scale;
ksec_10 =        param(9*52+35)*scale;
ksni_1 =         param(36)*scale;
ksni_2 =         param(52+36)*scale;
ksni_3 =         param(2*52+36)*scale;
ksni_4 =         param(3*52+36)*scale;
ksni_5 =         param(4*52+36)*scale;
ksni_6 =         param(5*52+36)*scale;
ksni_7 =         param(6*52+36)*scale;
ksni_8 =         param(7*52+36)*scale;
ksni_9 =         param(8*52+36)*scale;
ksni_10 =        param(9*52+36)*scale;
ksnicyto_1 =     param(37)*scale;
ksnicyto_2 =     param(52+37)*scale;
ksnicyto_3 =     param(2*52+37)*scale;
ksnicyto_4 =     param(3*52+37)*scale;
ksnicyto_5 =     param(4*52+37)*scale;
ksnicyto_6 =     param(5*52+37)*scale;
ksnicyto_7 =     param(6*52+37)*scale;
ksnicyto_8 =     param(7*52+37)*scale;
ksnicyto_9 =     param(8*52+37)*scale;
ksnicyto_10 =    param(9*52+37)*scale;
```

```
kstat_1 =        param(38)*scale;
kstat_2 =        param(52+38)*scale;
kstat_3 =        param(2*52+38)*scale;
kstat_4 =        param(3*52+38)*scale;
kstat_5 =        param(4*52+38)*scale;
kstat_6 =        param(5*52+38)*scale;
kstat_7 =        param(6*52+38)*scale;
kstat_8 =        param(7*52+38)*scale;
kstat_9 =        param(8*52+38)*scale;
kstat_10 =       param(9*52+38)*scale;
kv_1 =           param(39);
kv_2 =           param(52+39);
kv_3 =           param(2*52+39);
kv_4 =           param(3*52+39);
kv_5 =           param(4*52+39);
kv_6 =           param(5*52+39);
kv_7 =           param(6*52+39);
kv_8 =           param(7*52+39);
kv_9 =           param(8*52+39);
kv_10 =          param(9*52+39);
mua20m_1 =       param(40)*scale;
mua20m_2 =       param(52+40)*scale;
mua20m_3 =       param(2*52+40)*scale;
mua20m_4 =       param(3*52+40)*scale;
mua20m_5 =       param(4*52+40)*scale;
mua20m_6 =       param(5*52+40)*scale;
mua20m_7 =       param(6*52+40)*scale;
```

```
mua20m_8 =       param(7*52+40)*scale;
mua20m_9 =       param(8*52+40)*scale;
mua20m_10 =      param(9*52+40)*scale;
muilc_1 =        param(41)*scale;
muilc_2 =        param(52+41)*scale;
muilc_3 =        param(2*52+41)*scale;
muilc_4 =        param(3*52+41)*scale;
muilc_5 =        param(4*52+41)*scale;
muilc_6 =        param(5*52+41)*scale;
muilc_7 =        param(6*52+41)*scale;
muilc_8 =        param(7*52+41)*scale;
muilc_9 =        param(8*52+41)*scale;
muilc_10 =       param(9*52+41)*scale;
muilm_1 =        param(42)*scale;
muilm_2 =        param(52+42)*scale;
muilm_3 =        param(2*52+42)*scale;
muilm_4 =        param(3*52+42)*scale;
muilm_5 =        param(4*52+42)*scale;
muilm_6 =        param(5*52+42)*scale;
muilm_7 =        param(6*52+42)*scale;
muilm_8 =        param(7*52+42)*scale;
muilm_9 =        param(8*52+42)*scale;
muilm_10 =       param(9*52+42)*scale;
mus1c_1 =        param(43)*scale;
mus1c_2 =        param(52+43)*scale;
mus1c_3 =        param(2*52+43)*scale;
mus1c_4 =        param(3*52+43)*scale;
```

```
mus1c_5 =       param(4*52+43)*scale;
mus1c_6 =       param(5*52+43)*scale;
mus1c_7 =       param(6*52+43)*scale;
mus1c_8 =       param(7*52+43)*scale;
mus1c_9 =       param(8*52+43)*scale;
mus1c_10 =      param(9*52+43)*scale;
mus1m_1 =       param(44)*scale;
mus1m_2 =       param(52+44)*scale;
mus1m_3 =       param(2*52+44)*scale;
mus1m_4 =       param(3*52+44)*scale;
mus1m_5 =       param(4*52+44)*scale;
mus1m_6 =       param(5*52+44)*scale;
mus1m_7 =       param(6*52+44)*scale;
mus1m_8 =       param(7*52+44)*scale;
mus1m_9 =       param(8*52+44)*scale;
mus1m_10 =      param(9*52+44)*scale;
mus3c_1 =       param(45)*scale;
mus3c_2 =       param(52+45)*scale;
mus3c_3 =       param(2*52+45)*scale;
mus3c_4 =       param(3*52+45)*scale;
mus3c_5 =       param(4*52+45)*scale;
mus3c_6 =       param(5*52+45)*scale;
mus3c_7 =       param(6*52+45)*scale;
mus3c_8 =       param(7*52+45)*scale;
mus3c_9 =       param(8*52+45)*scale;
mus3c_10 =      param(9*52+45)*scale;
mus3m_1 =       param(46)*scale;
```

```
mus3m_2 =        param(52+46)*scale;
mus3m_3 =        param(2*52+46)*scale;
mus3m_4 =        param(3*52+46)*scale;
mus3m_5 =        param(4*52+46)*scale;
mus3m_6 =        param(5*52+46)*scale;
mus3m_7 =        param(6*52+46)*scale;
mus3m_8 =        param(7*52+46)*scale;
mus3m_9 =        param(8*52+46)*scale;
mus3m_10 =       param(9*52+46)*scale;
mutnc_1 =        param(47)*scale;
mutnc_2 =        param(52+47)*scale;
mutnc_3 =        param(2*52+47)*scale;
mutnc_4 =        param(3*52+47)*scale;
mutnc_5 =        param(4*52+47)*scale;
mutnc_6 =        param(5*52+47)*scale;
mutnc_7 =        param(6*52+47)*scale;
mutnc_8 =        param(7*52+47)*scale;
mutnc_9 =        param(8*52+47)*scale;
mutnc_10 =       param(9*52+47)*scale;
mutnm_1 =        param(48)*scale;
mutnm_2 =        param(52+48)*scale;
mutnm_3 =        param(2*52+48)*scale;
mutnm_4 =        param(3*52+48)*scale;
mutnm_5 =        param(4*52+48)*scale;
mutnm_6 =        param(5*52+48)*scale;
mutnm_7 =        param(6*52+48)*scale;
mutnm_8 =        param(7*52+48)*scale;
```

```
mutnm_9 =        param(8*52+48)*scale;
mutnm_10 =       param(9*52+48)*scale;
p_1 =            param(49);
p_2 =            param(52+49);
p_3 =            param(2*52+49);
p_4 =            param(3*52+49);
p_5 =            param(4*52+49);
p_6 =            param(5*52+49);
p_7 =            param(6*52+49);
p_8 =            param(7*52+49);
p_9 =            param(8*52+49);
p_10 =           param(9*52+49);
sm_1 =           param(50)*scale;
sm_2 =           param(52+50)*scale;
sm_3 =           param(2*52+50)*scale;
sm_4 =           param(3*52+50)*scale;
sm_5 =           param(4*52+50)*scale;
sm_6 =           param(5*52+50)*scale;
sm_7 =           param(6*52+50)*scale;
sm_8 =           param(7*52+50)*scale;
sm_9 =           param(8*52+50)*scale;
sm_10 =          param(9*52+50)*scale;
socs3inf_1 =     param(51);
socs3inf_2 =     param(52+51);
socs3inf_3 =     param(2*52+51);
socs3inf_4 =     param(3*52+51);
socs3inf_5 =     param(4*52+51);
```

```
socs3inf_6 =     param(5*52+51);
socs3inf_7 =     param(6*52+51);
socs3inf_8 =     param(7*52+51);
socs3inf_9 =     param(8*52+51);
socs3inf_10 =    param(9*52+51);
socsinf_1 =      param(52);
socsinf_2 =      param(52+52);
socsinf_3 =      param(2*52+52);
socsinf_4 =      param(3*52+52);
socsinf_5 =      param(4*52+52);
socsinf_6 =      param(5*52+52);
socsinf_7 =      param(6*52+52);
socsinf_8 =      param(7*52+52);
socsinf_9 =      param(8*52+52);
socsinf_10 =     param(9*52+52);
ti3_1 =          param(53)*scale;
ti3_2 =          param(52+53)*scale;
ti3_3 =          param(2*52+53)*scale;
ti3_4 =          param(3*52+53)*scale;
ti3_5 =          param(4*52+53)*scale;
ti3_6 =          param(5*52+53)*scale;
ti3_7 =          param(6*52+53)*scale;
ti3_8 =          param(7*52+53)*scale;
ti3_9 =          param(8*52+53)*scale;
ti3_10 =         param(9*52+53)*scale;
tnfa_trans_1 =   param(54)*scale;
tnfa_trans_2 =   param(52+54)*scale;
```

```
tnfa_trans_3 =   param(2*52+54)*scale;
tnfa_trans_4 =   param(3*52+54)*scale;
tnfa_trans_5 =   param(4*52+54)*scale;
tnfa_trans_6 =   param(5*52+54)*scale;
tnfa_trans_7 =   param(6*52+54)*scale;
tnfa_trans_8 =   param(7*52+54)*scale;
tnfa_trans_9 =   param(8*52+54)*scale;
tnfa_trans_10 =  param(9*52+54)*scale;

%----------------------- original model -------------------------------

%%%% shared variables
lps=    x(1);
il10ext=x(2);
tnfaext=x(3);

%%%% in each compartment
a20cyto_1=             x(4);
a20cyto_2=             x(1*36+4);
a20cyto_3=             x(2*36+4);
a20cyto_4=             x(3*36+4);
a20cyto_5=             x(4*36+4);
a20cyto_6=             x(5*36+4);
a20cyto_7=             x(6*36+4);
a20cyto_8=             x(7*36+4);
a20cyto_9=             x(8*36+4);
a20cyto_10=            x(9*36+4);
```

```
a20mrna_1=            x(5);
a20mrna_2=            x(1*36+5);
a20mrna_3=            x(2*36+5);
a20mrna_4=            x(3*36+5);
a20mrna_5=            x(4*36+5);
a20mrna_6=            x(5*36+5);
a20mrna_7=            x(6*36+5);
a20mrna_8=            x(7*36+5);
a20mrna_9=            x(8*36+5);
a20mrna_10=           x(9*36+5);
ikba_nfkbcyto_1=      x(6);
ikba_nfkbcyto_2=      x(1*36+6);
ikba_nfkbcyto_3=      x(2*36+6);
ikba_nfkbcyto_4=      x(3*36+6);
ikba_nfkbcyto_5=      x(4*36+6);
ikba_nfkbcyto_6=      x(5*36+6);
ikba_nfkbcyto_7=      x(6*36+6);
ikba_nfkbcyto_8=      x(7*36+6);
ikba_nfkbcyto_9=      x(8*36+6);
ikba_nfkbcyto_10=     x(9*36+6);
ikba_nfkbnuclear_1=   x(7);
ikba_nfkbnuclear_2=   x(1*36+7);
ikba_nfkbnuclear_3=   x(2*36+7);
ikba_nfkbnuclear_4=   x(3*36+7);
ikba_nfkbnuclear_5=   x(4*36+7);
ikba_nfkbnuclear_6=   x(5*36+7);
ikba_nfkbnuclear_7=   x(6*36+7);
```

```
ikba_nfkbnuclear_8=    x(7*36+7);
ikba_nfkbnuclear_9=    x(8*36+7);
ikba_nfkbnuclear_10=   x(9*36+7);
ikbacyto_1=            x(8);
ikbacyto_2=            x(1*36+8);
ikbacyto_3=            x(2*36+8);
ikbacyto_4=            x(3*36+8);
ikbacyto_5=            x(4*36+8);
ikbacyto_6=            x(5*36+8);
ikbacyto_7=            x(6*36+8);
ikbacyto_8=            x(7*36+8);
ikbacyto_9=            x(8*36+8);
ikbacyto_10=           x(9*36+8);
ikbamrna_1=            x(9);
ikbamrna_2=            x(1*36+9);
ikbamrna_3=            x(2*36+9);
ikbamrna_4=            x(3*36+9);
ikbamrna_5=            x(4*36+9);
ikbamrna_6=            x(5*36+9);
ikbamrna_7=            x(6*36+9);
ikbamrna_8=            x(7*36+9);
ikbamrna_9=            x(8*36+9);
ikbamrna_10=           x(9*36+9);
ikbanuclear_1=         x(10);
ikbanuclear_2=         x(1*36+10);
ikbanuclear_3=         x(2*36+10);
ikbanuclear_4=         x(3*36+10);
```

```
609 ikbanuclear_5=         x(4*36+10);
610 ikbanuclear_6=         x(5*36+10);
611 ikbanuclear_7=         x(6*36+10);
612 ikbanuclear_8=         x(7*36+10);
613 ikbanuclear_9=         x(8*36+10);
614 ikbanuclear_10=        x(9*36+10);
615 ikbaphospho_1=         x(11);
616 ikbaphospho_2=         x(1*36+11);
617 ikbaphospho_3=         x(2*36+11);
618 ikbaphospho_4=         x(3*36+11);
619 ikbaphospho_5=         x(4*36+11);
620 ikbaphospho_6=         x(5*36+11);
621 ikbaphospho_7=         x(6*36+11);
622 ikbaphospho_8=         x(7*36+11);
623 ikbaphospho_9=         x(8*36+11);
624 ikbaphospho_10=        x(9*36+11);
625 ikka_1=                x(12);
626 ikka_2=                x(1*36+12);
627 ikka_3=                x(2*36+12);
628 ikka_4=                x(3*36+12);
629 ikka_5=                x(4*36+12);
630 ikka_6=                x(5*36+12);
631 ikka_7=                x(6*36+12);
632 ikka_8=                x(7*36+12);
633 ikka_9=                x(8*36+12);
634 ikka_10=               x(9*36+12);
635 ikka_ikba_nfkbcyto_1=  x(13);
```

```
ikka_ikba_nfkbcyto_2=  x(1*36+13);
ikka_ikba_nfkbcyto_3=  x(2*36+13);
ikka_ikba_nfkbcyto_4=  x(3*36+13);
ikka_ikba_nfkbcyto_5=  x(4*36+13);
ikka_ikba_nfkbcyto_6=  x(5*36+13);
ikka_ikba_nfkbcyto_7=  x(6*36+13);
ikka_ikba_nfkbcyto_8=  x(7*36+13);
ikka_ikba_nfkbcyto_9=  x(8*36+13);
ikka_ikba_nfkbcyto_10= x(9*36+13);
ikki_1=                x(14);
ikki_2=                x(1*36+14);
ikki_3=                x(2*36+14);
ikki_4=                x(3*36+14);
ikki_5=                x(4*36+14);
ikki_6=                x(5*36+14);
ikki_7=                x(6*36+14);
ikki_8=                x(7*36+14);
ikki_9=                x(8*36+14);
ikki_10=               x(9*36+14);
ikkn_1=                x(15);
ikkn_2=                x(1*36+15);
ikkn_3=                x(2*36+15);
ikkn_4=                x(3*36+15);
ikkn_5=                x(4*36+15);
ikkn_6=                x(5*36+15);
ikkn_7=                x(6*36+15);
ikkn_8=                x(7*36+15);
```

```
ikkn_9=                 x(8*36+15);
ikkn_10=                x(9*36+15);
il10_il10r_1=           x(16);
il10_il10r_2=           x(1*36+16);
il10_il10r_3=           x(2*36+16);
il10_il10r_4=           x(3*36+16);
il10_il10r_5=           x(4*36+16);
il10_il10r_6=           x(5*36+16);
il10_il10r_7=           x(6*36+16);
il10_il10r_8=           x(7*36+16);
il10_il10r_9=           x(8*36+16);
il10_il10r_10=          x(9*36+16);
il10_rjt_1=             x(17);
il10_rjt_2=             x(1*36+17);
il10_rjt_3=             x(2*36+17);
il10_rjt_4=             x(3*36+17);
il10_rjt_5=             x(4*36+17);
il10_rjt_6=             x(5*36+17);
il10_rjt_7=             x(6*36+17);
il10_rjt_8=             x(7*36+17);
il10_rjt_9=             x(8*36+17);
il10_rjt_10=            x(9*36+17);
il10cyto_1=             x(18);
il10cyto_2=             x(1*36+18);
il10cyto_3=             x(2*36+18);
il10cyto_4=             x(3*36+18);
il10cyto_5=             x(4*36+18);
```

```
il10cyto_6=           x(5*36+18);
il10cyto_7=           x(6*36+18);
il10cyto_8=           x(7*36+18);
il10cyto_9=           x(8*36+18);
il10cyto_10=          x(9*36+18);
il10mrna_1=           x(19);
il10mrna_2=           x(1*36+19);
il10mrna_3=           x(2*36+19);
il10mrna_4=           x(3*36+19);
il10mrna_5=           x(4*36+19);
il10mrna_6=           x(5*36+19);
il10mrna_7=           x(6*36+19);
il10mrna_8=           x(7*36+19);
il10mrna_9=           x(8*36+19);
il10mrna_10=          x(9*36+19);
il10r_1=              x(20);
il10r_2=              x(1*36+20);
il10r_3=              x(2*36+20);
il10r_4=              x(3*36+20);
il10r_5=              x(4*36+20);
il10r_6=              x(5*36+20);
il10r_7=              x(6*36+20);
il10r_8=              x(7*36+20);
il10r_9=              x(8*36+20);
il10r_10=             x(9*36+20);
jak1_1=               x(21);
jak1_2=               x(1*36+21);
```

```
jak1_3=              x(2*36+21);
jak1_4=              x(3*36+21);
jak1_5=              x(4*36+21);
jak1_6=              x(5*36+21);
jak1_7=              x(6*36+21);
jak1_8=              x(7*36+21);
jak1_9=              x(8*36+21);
jak1_10=             x(9*36+21);
lps_tlr4_1=          x(22);
lps_tlr4_2=          x(1*36+22);
lps_tlr4_3=          x(2*36+22);
lps_tlr4_4=          x(3*36+22);
lps_tlr4_5=          x(4*36+22);
lps_tlr4_6=          x(5*36+22);
lps_tlr4_7=          x(6*36+22);
lps_tlr4_8=          x(7*36+22);
lps_tlr4_9=          x(8*36+22);
lps_tlr4_10=         x(9*36+22);
nfkbcyto_1=          x(23);
nfkbcyto_2=          x(1*36+23);
nfkbcyto_3=          x(2*36+23);
nfkbcyto_4=          x(3*36+23);
nfkbcyto_5=          x(4*36+23);
nfkbcyto_6=          x(5*36+23);
nfkbcyto_7=          x(6*36+23);
nfkbcyto_8=          x(7*36+23);
nfkbcyto_9=          x(8*36+23);
```

```
nfkbcyto_10=          x(9*36+23);
nfkbnuclear_1=        x(24);
nfkbnuclear_2=        x(1*36+24);
nfkbnuclear_3=        x(2*36+24);
nfkbnuclear_4=        x(3*36+24);
nfkbnuclear_5=        x(4*36+24);
nfkbnuclear_6=        x(5*36+24);
nfkbnuclear_7=        x(6*36+24);
nfkbnuclear_8=        x(7*36+24);
nfkbnuclear_9=        x(8*36+24);
nfkbnuclear_10=       x(9*36+24);
socs1cyto_1=          x(25);
socs1cyto_2=          x(1*36+25);
socs1cyto_3=          x(2*36+25);
socs1cyto_4=          x(3*36+25);
socs1cyto_5=          x(4*36+25);
socs1cyto_6=          x(5*36+25);
socs1cyto_7=          x(6*36+25);
socs1cyto_8=          x(7*36+25);
socs1cyto_9=          x(8*36+25);
socs1cyto_10=         x(9*36+25);
socs1mrna_1=          x(26);
socs1mrna_2=          x(1*36+26);
socs1mrna_3=          x(2*36+26);
socs1mrna_4=          x(3*36+26);
socs1mrna_5=          x(4*36+26);
socs1mrna_6=          x(5*36+26);
```

```
socs1mrna_7=          x(6*36+26);
socs1mrna_8=          x(7*36+26);
socs1mrna_9=          x(8*36+26);
socs1mrna_10=         x(9*36+26);
socs3cyto_1=          x(27);
socs3cyto_2=          x(1*36+27);
socs3cyto_3=          x(2*36+27);
socs3cyto_4=          x(3*36+27);
socs3cyto_5=          x(4*36+27);
socs3cyto_6=          x(5*36+27);
socs3cyto_7=          x(6*36+27);
socs3cyto_8=          x(7*36+27);
socs3cyto_9=          x(8*36+27);
socs3cyto_10=         x(9*36+27);
socs3mrna_1=          x(28);
socs3mrna_2=          x(1*36+28);
socs3mrna_3=          x(2*36+28);
socs3mrna_4=          x(3*36+28);
socs3mrna_5=          x(4*36+28);
socs3mrna_6=          x(5*36+28);
socs3mrna_7=          x(6*36+28);
socs3mrna_8=          x(7*36+28);
socs3mrna_9=          x(8*36+28);
socs3mrna_10=         x(9*36+28);
stat3a_1=             x(29);
stat3a_2=             x(1*36+29);
stat3a_3=             x(2*36+29);
```

```
stat3a_4=             x(3*36+29);
stat3a_5=             x(4*36+29);
stat3a_6=             x(5*36+29);
stat3a_7=             x(6*36+29);
stat3a_8=             x(7*36+29);
stat3a_9=             x(8*36+29);
stat3a_10=            x(9*36+29);
stat3i_1=             x(30);
stat3i_2=             x(1*36+30);
stat3i_3=             x(2*36+30);
stat3i_4=             x(3*36+30);
stat3i_5=             x(4*36+30);
stat3i_6=             x(5*36+30);
stat3i_7=             x(6*36+30);
stat3i_8=             x(7*36+30);
stat3i_9=             x(8*36+30);
stat3i_10=            x(9*36+30);
stat3n_1=             x(31);
stat3n_2=             x(1*36+31);
stat3n_3=             x(2*36+31);
stat3n_4=             x(3*36+31);
stat3n_5=             x(4*36+31);
stat3n_6=             x(5*36+31);
stat3n_7=             x(6*36+31);
stat3n_8=             x(7*36+31);
stat3n_9=             x(8*36+31);
stat3n_10=            x(9*36+31);
```

```
stat3ni_1=            x(32);
stat3ni_2=            x(1*36+32);
stat3ni_3=            x(2*36+32);
stat3ni_4=            x(3*36+32);
stat3ni_5=            x(4*36+32);
stat3ni_6=            x(5*36+32);
stat3ni_7=            x(6*36+32);
stat3ni_8=            x(7*36+32);
stat3ni_9=            x(8*36+32);
stat3ni_10=           x(9*36+32);
tlr4_1=               x(33);
tlr4_2=               x(1*36+33);
tlr4_3=               x(2*36+33);
tlr4_4=               x(3*36+33);
tlr4_5=               x(4*36+33);
tlr4_6=               x(5*36+33);
tlr4_7=               x(6*36+33);
tlr4_8=               x(7*36+33);
tlr4_9=               x(8*36+33);
tlr4_10=              x(9*36+33);
tnfa_tnfar_1=         x(34);
tnfa_tnfar_2=         x(1*36+34);
tnfa_tnfar_3=         x(2*36+34);
tnfa_tnfar_4=         x(3*36+34);
tnfa_tnfar_5=         x(4*36+34);
tnfa_tnfar_6=         x(5*36+34);
tnfa_tnfar_7=         x(6*36+34);
```

```
tnfa_tnfar_8=          x(7*36+34);
tnfa_tnfar_9=          x(8*36+34);
tnfa_tnfar_10=         x(9*36+34);
tnfacyto_1=            x(35);
tnfacyto_2=            x(1*36+35);
tnfacyto_3=            x(2*36+35);
tnfacyto_4=            x(3*36+35);
tnfacyto_5=            x(4*36+35);
tnfacyto_6=            x(5*36+35);
tnfacyto_7=            x(6*36+35);
tnfacyto_8=            x(7*36+35);
tnfacyto_9=            x(8*36+35);
tnfacyto_10=           x(9*36+35);
tnfamrna_1=            x(36);
tnfamrna_2=            x(1*36+36);
tnfamrna_3=            x(2*36+36);
tnfamrna_4=            x(3*36+36);
tnfamrna_5=            x(4*36+36);
tnfamrna_6=            x(5*36+36);
tnfamrna_7=            x(6*36+36);
tnfamrna_8=            x(7*36+36);
tnfamrna_9=            x(8*36+36);
tnfamrna_10=           x(9*36+36);
tnfar_1=               x(37);
tnfar_2=               x(1*36+37);
tnfar_3=               x(2*36+37);
tnfar_4=               x(3*36+37);
```

```
tnfar_5=              x(4*36+37);
tnfar_6=              x(5*36+37);
tnfar_7=              x(6*36+37);
tnfar_8=              x(7*36+37);
tnfar_9=              x(8*36+37);
tnfar_10=             x(9*36+37);
tyk2_1=               x(38);
tyk2_2=               x(1*36+38);
tyk2_3=               x(2*36+38);
tyk2_4=               x(3*36+38);
tyk2_5=               x(4*36+38);
tyk2_6=               x(5*36+38);
tyk2_7=               x(6*36+38);
tyk2_8=               x(7*36+38);
tyk2_9=               x(8*36+38);
tyk2_10=              x(9*36+38);
il10act_1=            x(39);
il10act_2=            x(1*36+39);
il10act_3=            x(2*36+39);
il10act_4=            x(3*36+39);
il10act_5=            x(4*36+39);
il10act_6=            x(5*36+39);
il10act_7=            x(6*36+39);
il10act_8=            x(7*36+39);
il10act_9=            x(8*36+39);
il10act_10=           x(9*36+39);

```

```
kin_1=(1-il10_il10r_1/il10max_1)*(il10_il10r_1<il10max_1);
kin_2=(1-il10_il10r_2/il10max_2)*(il10_il10r_2<il10max_2);
kin_3=(1-il10_il10r_3/il10max_3)*(il10_il10r_3<il10max_3);
kin_4=(1-il10_il10r_4/il10max_4)*(il10_il10r_4<il10max_4);
kin_5=(1-il10_il10r_5/il10max_5)*(il10_il10r_5<il10max_5);
kin_6=(1-il10_il10r_6/il10max_6)*(il10_il10r_6<il10max_6);
kin_7=(1-il10_il10r_7/il10max_7)*(il10_il10r_7<il10max_7);
kin_8=(1-il10_il10r_8/il10max_8)*(il10_il10r_8<il10max_8);
kin_9=(1-il10_il10r_9/il10max_9)*(il10_il10r_9<il10max_9);
kin_10=(1-il10_il10r_10/il10max_10)*(il10_il10r_10<il10max_10);

dx=zeros(9*36+39,1);

%%%% shared variables
% lps
dx(1) = 0;

% il10ext (formerly il10sup)
dx(2) = -kilrb_1*il10ext*il10r_1 - kilrb_2*il10ext*il10r_2...
        -kilrb_3*il10ext*il10r_3 - kilrb_4*il10ext*il10r_4...
        -kilrb_5*il10ext*il10r_5 - kilrb_6*il10ext*il10r_6...
        -kilrb_7*il10ext*il10r_7 - kilrb_8*il10ext*il10r_8...
        -kilrb_9*il10ext*il10r_9 - kilrb_10*il10ext*il10r_10...
    + kilru_1*il10_il10r_1 + kilru_2*il10_il10r_2...
    + kilru_3*il10_il10r_3 + kilru_4*il10_il10r_4...
    + kilru_5*il10_il10r_5 + kilru_6*il10_il10r_6...
    + kilru_7*il10_il10r_7 + kilru_8*il10_il10r_8...
```

```
    + kilru_9*il10_il10r_9 + kilru_10*il10_il10r_10...
    + kilc_1*il10cyto_1*(0.36/200) + kilc_2*il10cyto_2*(0.36/200)...
    + kilc_3*il10cyto_3*(0.36/200) + kilc_4*il10cyto_4*(0.36/200)...
    + kilc_5*il10cyto_5*(0.36/200) + kilc_6*il10cyto_6*(0.36/200)...
    + kilc_7*il10cyto_7*(0.36/200) + kilc_8*il10cyto_8*(0.36/200)...
    + kilc_9*il10cyto_9*(0.36/200) + kilc_10*il10cyto_10*(0.36/200)...
    - muile*il10ext;

% tnfaext
dx(3) = ksec_1*tnfacyto_1*0.36/200 + ksec_2*tnfacyto_2*0.36/200 ...
      + ksec_3*tnfacyto_3*0.36/200 + ksec_4*tnfacyto_4*0.36/200 ...
      + ksec_5*tnfacyto_5*0.36/200 + ksec_6*tnfacyto_6*0.36/200 ...
      + ksec_7*tnfacyto_7*0.36/200 + ksec_8*tnfacyto_8*0.36/200 ...
      + ksec_9*tnfacyto_9*0.36/200 + ksec_10*tnfacyto_10*0.36/200 ...
- kf3_1*tnfaext*tnfar_1 - kf3_2*tnfaext*tnfar_2 ...
- kf3_3*tnfaext*tnfar_3 - kf3_4*tnfaext*tnfar_4 ...
- kf3_5*tnfaext*tnfar_5 - kf3_6*tnfaext*tnfar_6 ...
- kf3_7*tnfaext*tnfar_7 - kf3_8*tnfaext*tnfar_8 ...
- kf3_9*tnfaext*tnfar_9 - kf3_10*tnfaext*tnfar_10 ...
+ kr3_1*tnfa_tnfar_1 + kr3_2*tnfa_tnfar_2 ...
+ kr3_3*tnfa_tnfar_3 + kr3_4*tnfa_tnfar_4 ...
+ kr3_5*tnfa_tnfar_5 + kr3_6*tnfa_tnfar_6 ...
+ kr3_7*tnfa_tnfar_7 + kr3_8*tnfa_tnfar_8 ...
+ kr3_9*tnfa_tnfar_9 + kr3_10*tnfa_tnfar_10 ...
- kdeg_tnfa*tnfaext;

% a20cyto
```

```
dx(4) =      a_trans_1*a20mrna_1-kdeg_a20_1*a20cyto_1;
dx(4+36) =   a_trans_2*a20mrna_2-kdeg_a20_2*a20cyto_2;
dx(4+36*2) = a_trans_3*a20mrna_3-kdeg_a20_3*a20cyto_3;
dx(4+36*3) = a_trans_4*a20mrna_4-kdeg_a20_4*a20cyto_4;
dx(4+36*4) = a_trans_5*a20mrna_5-kdeg_a20_5*a20cyto_5;
dx(4+36*5) = a_trans_6*a20mrna_6-kdeg_a20_6*a20cyto_6;
dx(4+36*6) = a_trans_7*a20mrna_7-kdeg_a20_7*a20cyto_7;
dx(4+36*7) = a_trans_8*a20mrna_8-kdeg_a20_8*a20cyto_8;
dx(4+36*8) = a_trans_9*a20mrna_9-kdeg_a20_9*a20cyto_9;
dx(4+36*9) = a_trans_10*a20mrna_10-kdeg_a20_10*a20cyto_10;

% a20mrna
dx(5) =      sm_1*p_1*(nfkbnuclear_1/(ctf_1+nfkbnuclear_1))-mua20m_1*
   a20mrna_1;
dx(5+36) =   sm_2*p_2*(nfkbnuclear_2/(ctf_2+nfkbnuclear_2))-mua20m_2*
   a20mrna_2;
dx(5+36*2) = sm_3*p_3*(nfkbnuclear_3/(ctf_3+nfkbnuclear_3))-mua20m_3*
   a20mrna_3;
dx(5+36*3) = sm_4*p_4*(nfkbnuclear_4/(ctf_4+nfkbnuclear_4))-mua20m_4*
   a20mrna_4;
dx(5+36*4) = sm_5*p_5*(nfkbnuclear_5/(ctf_5+nfkbnuclear_5))-mua20m_5*
   a20mrna_5;
dx(5+36*5) = sm_6*p_6*(nfkbnuclear_6/(ctf_6+nfkbnuclear_6))-mua20m_6*
   a20mrna_6;
dx(5+36*6) = sm_7*p_7*(nfkbnuclear_7/(ctf_7+nfkbnuclear_7))-mua20m_7*
   a20mrna_7;
dx(5+36*7) = sm_8*p_8*(nfkbnuclear_8/(ctf_8+nfkbnuclear_8))-mua20m_8*
```

```
    a20mrna_8;
dx(5+36*8) = sm_9*p_9*(nfkbnuclear_9/(ctf_9+nfkbnuclear_9))-mua20m_9*
    a20mrna_9;
dx(5+36*9) = sm_10*p_10*(nfkbnuclear_10/(ctf_10+nfkbnuclear_10))-mua20m_10*
    a20mrna_10;

% ikba_nfkbcyto
dx(6) =      kf4_1*nfkbcyto_1*ikbacyto_1+eni_1*ikba_nfkbnuclear_1*kv_1-kk3_1
    *kin_1*ikka_1*ikba_nfkbcyto_1;
dx(6+36) =   kf4_2*nfkbcyto_2*ikbacyto_2+eni_2*ikba_nfkbnuclear_2*kv_2-kk3_2
    *kin_2*ikka_2*ikba_nfkbcyto_2;
dx(6+36*2) = kf4_3*nfkbcyto_3*ikbacyto_3+eni_3*ikba_nfkbnuclear_3*kv_3-kk3_3
    *kin_3*ikka_3*ikba_nfkbcyto_3;
dx(6+36*3) = kf4_4*nfkbcyto_4*ikbacyto_4+eni_4*ikba_nfkbnuclear_4*kv_4-kk3_4
    *kin_4*ikka_4*ikba_nfkbcyto_4;
dx(6+36*4) = kf4_5*nfkbcyto_5*ikbacyto_5+eni_5*ikba_nfkbnuclear_5*kv_5-kk3_5
    *kin_5*ikka_5*ikba_nfkbcyto_5;
dx(6+36*5) = kf4_6*nfkbcyto_6*ikbacyto_6+eni_6*ikba_nfkbnuclear_6*kv_6-kk3_6
    *kin_6*ikka_6*ikba_nfkbcyto_6;
dx(6+36*6) = kf4_7*nfkbcyto_7*ikbacyto_7+eni_7*ikba_nfkbnuclear_7*kv_7-kk3_7
    *kin_7*ikka_7*ikba_nfkbcyto_7;
dx(6+36*7) = kf4_8*nfkbcyto_8*ikbacyto_8+eni_8*ikba_nfkbnuclear_8*kv_8-kk3_8
    *kin_8*ikka_8*ikba_nfkbcyto_8;
dx(6+36*8) = kf4_9*nfkbcyto_9*ikbacyto_9+eni_9*ikba_nfkbnuclear_9*kv_9-kk3_9
    *kin_9*ikka_9*ikba_nfkbcyto_9;
dx(6+36*9) = kf4_10*nfkbcyto_10*ikbacyto_10+eni_10*ikba_nfkbnuclear_10*kv_10
    -kk3_10*kin_10*ikka_10*ikba_nfkbcyto_10;
```

```
994
995 % ikba_nfkbnuclear
996 dx(7) =      kf4_1*nfkbnuclear_1*ikbanuclear_1-eni_1*ikba_nfkbnuclear_1;
997 dx(7+36) =   kf4_2*nfkbnuclear_2*ikbanuclear_2-eni_2*ikba_nfkbnuclear_2;
998 dx(7+36*2) = kf4_3*nfkbnuclear_3*ikbanuclear_3-eni_3*ikba_nfkbnuclear_3;
999 dx(7+36*3) = kf4_4*nfkbnuclear_4*ikbanuclear_4-eni_4*ikba_nfkbnuclear_4;
1000 dx(7+36*4) = kf4_5*nfkbnuclear_5*ikbanuclear_5-eni_5*ikba_nfkbnuclear_5;
1001 dx(7+36*5) = kf4_6*nfkbnuclear_6*ikbanuclear_6-eni_6*ikba_nfkbnuclear_6;
1002 dx(7+36*6) = kf4_7*nfkbnuclear_7*ikbanuclear_7-eni_7*ikba_nfkbnuclear_7;
1003 dx(7+36*7) = kf4_8*nfkbnuclear_8*ikbanuclear_8-eni_8*ikba_nfkbnuclear_8;
1004 dx(7+36*8) = kf4_9*nfkbnuclear_9*ikbanuclear_9-eni_9*ikba_nfkbnuclear_9;
1005 dx(7+36*9) = kf4_10*nfkbnuclear_10*ikbanuclear_10-eni_10*ikba_nfkbnuclear_10
     ;
1006
1007 % ikbacyto
1008 dx(8) =      ikba_trans_1*ikbamrna_1-kf4_1*nfkbcyto_1*ikbacyto_1-iki_1*
     ikbacyto_1+eki_1*ikbanuclear_1*kv_1;
1009 dx(8+36) =   ikba_trans_2*ikbamrna_2-kf4_2*nfkbcyto_2*ikbacyto_2-iki_2*
     ikbacyto_2+eki_2*ikbanuclear_2*kv_2;
1010 dx(8+36*2) = ikba_trans_3*ikbamrna_3-kf4_3*nfkbcyto_3*ikbacyto_3-iki_3*
     ikbacyto_3+eki_3*ikbanuclear_3*kv_3;
1011 dx(8+36*3) = ikba_trans_4*ikbamrna_4-kf4_4*nfkbcyto_4*ikbacyto_4-iki_4*
     ikbacyto_4+eki_4*ikbanuclear_4*kv_4;
1012 dx(8+36*4) = ikba_trans_5*ikbamrna_5-kf4_5*nfkbcyto_5*ikbacyto_5-iki_5*
     ikbacyto_5+eki_5*ikbanuclear_5*kv_5;
1013 dx(8+36*5) = ikba_trans_6*ikbamrna_6-kf4_6*nfkbcyto_6*ikbacyto_6-iki_6*
     ikbacyto_6+eki_6*ikbanuclear_6*kv_6;
```

```
dx(8+36*6) = ikba_trans_7*ikbamrna_7-kf4_7*nfkbcyto_7*ikbacyto_7-iki_7*
   ikbacyto_7+eki_7*ikbanuclear_7*kv_7;
dx(8+36*7) = ikba_trans_8*ikbamrna_8-kf4_8*nfkbcyto_8*ikbacyto_8-iki_8*
   ikbacyto_8+eki_8*ikbanuclear_8*kv_8;
dx(8+36*8) = ikba_trans_9*ikbamrna_9-kf4_9*nfkbcyto_9*ikbacyto_9-iki_9*
   ikbacyto_9+eki_9*ikbanuclear_9*kv_9;
dx(8+36*9) = ikba_trans_10*ikbamrna_10-kf4_10*nfkbcyto_10*ikbacyto_10-iki_10
   *ikbacyto_10+eki_10*ikbanuclear_10*kv_10;

% ikbamrna
dx(9) =      sm_1*p_1*(nfkbnuclear_1/(ctf_1+nfkbnuclear_1))-muilm_1*
   ikbamrna_1;
dx(9+36) =   sm_2*p_2*(nfkbnuclear_2/(ctf_2+nfkbnuclear_2))-muilm_2*
   ikbamrna_2;
dx(9+36*2) = sm_3*p_3*(nfkbnuclear_3/(ctf_3+nfkbnuclear_3))-muilm_3*
   ikbamrna_3;
dx(9+36*3) = sm_4*p_4*(nfkbnuclear_4/(ctf_4+nfkbnuclear_4))-muilm_4*
   ikbamrna_4;
dx(9+36*4) = sm_5*p_5*(nfkbnuclear_5/(ctf_5+nfkbnuclear_5))-muilm_5*
   ikbamrna_5;
dx(9+36*5) = sm_6*p_6*(nfkbnuclear_6/(ctf_6+nfkbnuclear_6))-muilm_6*
   ikbamrna_6;
dx(9+36*6) = sm_7*p_7*(nfkbnuclear_7/(ctf_7+nfkbnuclear_7))-muilm_7*
   ikbamrna_7;
dx(9+36*7) = sm_8*p_8*(nfkbnuclear_8/(ctf_8+nfkbnuclear_8))-muilm_8*
   ikbamrna_8;
dx(9+36*8) = sm_9*p_9*(nfkbnuclear_9/(ctf_9+nfkbnuclear_9))-muilm_9*
```

```
    ikbamrna_9;
dx(9+36*9) = sm_10*p_10*(nfkbnuclear_10/(ctf_10+nfkbnuclear_10))-muilm_10*
    ikbamrna_10;

% ikbanuclear
dx(10) =      -kf4_1*nfkbnuclear_1*ikbanuclear_1+iki_1/kv_1*ikbacyto_1-eki_1
    *ikbanuclear_1;
dx(10+36) =   -kf4_2*nfkbnuclear_2*ikbanuclear_2+iki_2/kv_2*ikbacyto_2-eki_2
    *ikbanuclear_2;
dx(10+36*2) = -kf4_3*nfkbnuclear_3*ikbanuclear_3+iki_3/kv_3*ikbacyto_3-eki_3
    *ikbanuclear_3;
dx(10+36*3) = -kf4_4*nfkbnuclear_4*ikbanuclear_4+iki_4/kv_4*ikbacyto_4-eki_4
    *ikbanuclear_4;
dx(10+36*4) = -kf4_5*nfkbnuclear_5*ikbanuclear_5+iki_5/kv_5*ikbacyto_5-eki_5
    *ikbanuclear_5;
dx(10+36*5) = -kf4_6*nfkbnuclear_6*ikbanuclear_6+iki_6/kv_6*ikbacyto_6-eki_6
    *ikbanuclear_6;
dx(10+36*6) = -kf4_7*nfkbnuclear_7*ikbanuclear_7+iki_7/kv_7*ikbacyto_7-eki_7
    *ikbanuclear_7;
dx(10+36*7) = -kf4_8*nfkbnuclear_8*ikbanuclear_8+iki_8/kv_8*ikbacyto_8-eki_8
    *ikbanuclear_8;
dx(10+36*8) = -kf4_9*nfkbnuclear_9*ikbanuclear_9+iki_9/kv_9*ikbacyto_9-eki_9
    *ikbanuclear_9;
dx(10+36*9) = -kf4_10*nfkbnuclear_10*ikbanuclear_10+iki_10/kv_10*ikbacyto_10
    -eki_10*ikbanuclear_10;

% ikbaphospho
```

```
dx(11) =      ti3_1*ikka_ikba_nfkbcyto_1-kdeg_ikba_1*ikbaphospho_1;
dx(11+36) =   ti3_2*ikka_ikba_nfkbcyto_2-kdeg_ikba_2*ikbaphospho_2;
dx(11+36*2) = ti3_3*ikka_ikba_nfkbcyto_3-kdeg_ikba_3*ikbaphospho_3;
dx(11+36*3) = ti3_4*ikka_ikba_nfkbcyto_4-kdeg_ikba_4*ikbaphospho_4;
dx(11+36*4) = ti3_5*ikka_ikba_nfkbcyto_5-kdeg_ikba_5*ikbaphospho_5;
dx(11+36*5) = ti3_6*ikka_ikba_nfkbcyto_6-kdeg_ikba_6*ikbaphospho_6;
dx(11+36*6) = ti3_7*ikka_ikba_nfkbcyto_7-kdeg_ikba_7*ikbaphospho_7;
dx(11+36*7) = ti3_8*ikka_ikba_nfkbcyto_8-kdeg_ikba_8*ikbaphospho_8;
dx(11+36*8) = ti3_9*ikka_ikba_nfkbcyto_9-kdeg_ikba_9*ikbaphospho_9;
dx(11+36*9) = ti3_10*ikka_ikba_nfkbcyto_10-kdeg_ikba_10*ikbaphospho_10;

% ikka
dx(12) =      kfi_1*kin_1*(lps_tlr4_1+tnfa_tnfar_1)*ikkn_1-kk3_1*kin_1*
   ikka_1*ikba_nfkbcyto_1-kk1_1*ikka_1*a20cyto_1;
dx(12+36) =   kfi_2*kin_2*(lps_tlr4_2+tnfa_tnfar_2)*ikkn_2-kk3_2*kin_2*
   ikka_2*ikba_nfkbcyto_2-kk1_2*ikka_2*a20cyto_2;
dx(12+36*2) = kfi_3*kin_3*(lps_tlr4_3+tnfa_tnfar_3)*ikkn_3-kk3_3*kin_3*
   ikka_3*ikba_nfkbcyto_3-kk1_3*ikka_3*a20cyto_3;
dx(12+36*3) = kfi_4*kin_4*(lps_tlr4_4+tnfa_tnfar_4)*ikkn_4-kk3_4*kin_4*
   ikka_4*ikba_nfkbcyto_4-kk1_4*ikka_4*a20cyto_4;
dx(12+36*4) = kfi_5*kin_5*(lps_tlr4_5+tnfa_tnfar_5)*ikkn_5-kk3_5*kin_5*
   ikka_5*ikba_nfkbcyto_5-kk1_5*ikka_5*a20cyto_5;
dx(12+36*5) = kfi_6*kin_6*(lps_tlr4_6+tnfa_tnfar_6)*ikkn_6-kk3_6*kin_6*
   ikka_6*ikba_nfkbcyto_6-kk1_6*ikka_6*a20cyto_6;
dx(12+36*6) = kfi_7*kin_7*(lps_tlr4_7+tnfa_tnfar_7)*ikkn_7-kk3_7*kin_7*
   ikka_7*ikba_nfkbcyto_7-kk1_7*ikka_7*a20cyto_7;
dx(12+36*7) = kfi_8*kin_8*(lps_tlr4_8+tnfa_tnfar_8)*ikkn_8-kk3_8*kin_8*
```

```
    ikka_8*ikba_nfkbcyto_8-kk1_8*ikka_8*a20cyto_8;
dx(12+36*8) = kfi_9*kin_9*(lps_tlr4_9+tnfa_tnfar_9)*ikkn_9-kk3_9*kin_9*
    ikka_9*ikba_nfkbcyto_9-kk1_9*ikka_9*a20cyto_9;
dx(12+36*9) = kfi_10*kin_10*(lps_tlr4_10+tnfa_tnfar_10)*ikkn_10-kk3_10*
    kin_10*ikka_10*ikba_nfkbcyto_10-kk1_10*ikka_10*a20cyto_10;

% ikka_ikba_nfkbcyto
dx(13) =      kk3_1*kin_1*ikka_1*ikba_nfkbcyto_1-ti3_1*ikka_ikba_nfkbcyto_1;
dx(13+36) =   kk3_2*kin_2*ikka_2*ikba_nfkbcyto_2-ti3_2*ikka_ikba_nfkbcyto_2;
dx(13+36*2) = kk3_3*kin_3*ikka_3*ikba_nfkbcyto_3-ti3_3*ikka_ikba_nfkbcyto_3;
dx(13+36*3) = kk3_4*kin_4*ikka_4*ikba_nfkbcyto_4-ti3_4*ikka_ikba_nfkbcyto_4;
dx(13+36*4) = kk3_5*kin_5*ikka_5*ikba_nfkbcyto_5-ti3_5*ikka_ikba_nfkbcyto_5;
dx(13+36*5) = kk3_6*kin_6*ikka_6*ikba_nfkbcyto_6-ti3_6*ikka_ikba_nfkbcyto_6;
dx(13+36*6) = kk3_7*kin_7*ikka_7*ikba_nfkbcyto_7-ti3_7*ikka_ikba_nfkbcyto_7;
dx(13+36*7) = kk3_8*kin_8*ikka_8*ikba_nfkbcyto_8-ti3_8*ikka_ikba_nfkbcyto_8;
dx(13+36*8) = kk3_9*kin_9*ikka_9*ikba_nfkbcyto_9-ti3_9*ikka_ikba_nfkbcyto_9;
dx(13+36*9) = kk3_10*kin_10*ikka_10*ikba_nfkbcyto_10-ti3_10*
    ikka_ikba_nfkbcyto_10;

% ikki
dx(14) =      kk1_1*ikka_1*a20cyto_1;
dx(14+36) =   kk1_2*ikka_2*a20cyto_2;
dx(14+36*2) = kk1_3*ikka_3*a20cyto_3;
dx(14+36*3) = kk1_4*ikka_4*a20cyto_4;
dx(14+36*4) = kk1_5*ikka_5*a20cyto_5;
dx(14+36*5) = kk1_6*ikka_6*a20cyto_6;
dx(14+36*6) = kk1_7*ikka_7*a20cyto_7;
```

```
dx(14+36*7) = kk1_8*ikka_8*a20cyto_8;
dx(14+36*8) = kk1_9*ikka_9*a20cyto_9;
dx(14+36*9) = kk1_10*ikka_10*a20cyto_10;

% ikkn
dx(15) =      -kfi_1*kin_1*(lps_tlr4_1+tnfa_tnfar_1)*ikkn_1+ti3_1*
   ikka_ikba_nfkbcyto_1;
dx(15+36) =   -kfi_2*kin_2*(lps_tlr4_2+tnfa_tnfar_2)*ikkn_2+ti3_2*
   ikka_ikba_nfkbcyto_2;
dx(15+36*2) = -kfi_3*kin_3*(lps_tlr4_3+tnfa_tnfar_3)*ikkn_3+ti3_3*
   ikka_ikba_nfkbcyto_3;
dx(15+36*3) = -kfi_4*kin_4*(lps_tlr4_4+tnfa_tnfar_4)*ikkn_4+ti3_4*
   ikka_ikba_nfkbcyto_4;
dx(15+36*4) = -kfi_5*kin_5*(lps_tlr4_5+tnfa_tnfar_5)*ikkn_5+ti3_5*
   ikka_ikba_nfkbcyto_5;
dx(15+36*5) = -kfi_6*kin_6*(lps_tlr4_6+tnfa_tnfar_6)*ikkn_6+ti3_6*
   ikka_ikba_nfkbcyto_6;
dx(15+36*6) = -kfi_7*kin_7*(lps_tlr4_7+tnfa_tnfar_7)*ikkn_7+ti3_7*
   ikka_ikba_nfkbcyto_7;
dx(15+36*7) = -kfi_8*kin_8*(lps_tlr4_8+tnfa_tnfar_8)*ikkn_8+ti3_8*
   ikka_ikba_nfkbcyto_8;
dx(15+36*8) = -kfi_9*kin_9*(lps_tlr4_9+tnfa_tnfar_9)*ikkn_9+ti3_9*
   ikka_ikba_nfkbcyto_9;
dx(15+36*9) = -kfi_10*kin_10*(lps_tlr4_10+tnfa_tnfar_10)*ikkn_10+ti3_10*
   ikka_ikba_nfkbcyto_10;

% il10_il10r
```

```
dx(16) =      kilrb_1*il10ext*il10r_1-kilru_1*il10_il10r_1 - kiljb_1*
   il10_il10r_1*jak1_1*tyk2_1 + kilju_1*il10_rjt_1;
dx(16+36) =   kilrb_2*il10ext*il10r_2-kilru_2*il10_il10r_2 - kiljb_2*
   il10_il10r_2*jak1_2*tyk2_2 + kilju_2*il10_rjt_2;
dx(16+36*2) = kilrb_3*il10ext*il10r_3-kilru_3*il10_il10r_3 - kiljb_3*
   il10_il10r_3*jak1_3*tyk2_3 + kilju_3*il10_rjt_3;
dx(16+36*3) = kilrb_4*il10ext*il10r_4-kilru_4*il10_il10r_4 - kiljb_4*
   il10_il10r_4*jak1_4*tyk2_4 + kilju_4*il10_rjt_4;
dx(16+36*4) = kilrb_5*il10ext*il10r_5-kilru_5*il10_il10r_5 - kiljb_5*
   il10_il10r_5*jak1_5*tyk2_5 + kilju_5*il10_rjt_5;
dx(16+36*5) = kilrb_6*il10ext*il10r_6-kilru_6*il10_il10r_6 - kiljb_6*
   il10_il10r_6*jak1_6*tyk2_6 + kilju_6*il10_rjt_6;
dx(16+36*6) = kilrb_7*il10ext*il10r_7-kilru_7*il10_il10r_7 - kiljb_7*
   il10_il10r_7*jak1_7*tyk2_7 + kilju_7*il10_rjt_7;
dx(16+36*7) = kilrb_8*il10ext*il10r_8-kilru_8*il10_il10r_8 - kiljb_8*
   il10_il10r_8*jak1_8*tyk2_8 + kilju_8*il10_rjt_8;
dx(16+36*8) = kilrb_9*il10ext*il10r_9-kilru_9*il10_il10r_9 - kiljb_9*
   il10_il10r_9*jak1_9*tyk2_9 + kilju_9*il10_rjt_9;
dx(16+36*9) = kilrb_10*il10ext*il10r_10-kilru_10*il10_il10r_10 - kiljb_10*
   il10_il10r_10*jak1_10*tyk2_10 + kilju_10*il10_rjt_10;

% il10_rjt (il10/il10r/jak1/tyk2)
dx(17) =      kiljb_1*il10_il10r_1*jak1_1*tyk2_1 - kilju_1*il10_rjt_1;
dx(17+36) =   kiljb_2*il10_il10r_2*jak1_2*tyk2_2 - kilju_2*il10_rjt_2;
dx(17+36*2) = kiljb_3*il10_il10r_3*jak1_3*tyk2_3 - kilju_3*il10_rjt_3;
dx(17+36*3) = kiljb_4*il10_il10r_4*jak1_4*tyk2_4 - kilju_4*il10_rjt_4;
dx(17+36*4) = kiljb_5*il10_il10r_5*jak1_5*tyk2_5 - kilju_5*il10_rjt_5;
```

```
dx(17+36*5) = kiljb_6*il10_il10r_6*jak1_6*tyk2_6 - kilju_6*il10_rjt_6;
dx(17+36*6) = kiljb_7*il10_il10r_7*jak1_7*tyk2_7 - kilju_7*il10_rjt_7;
dx(17+36*7) = kiljb_8*il10_il10r_8*jak1_8*tyk2_8 - kilju_8*il10_rjt_8;
dx(17+36*8) = kiljb_9*il10_il10r_9*jak1_9*tyk2_9 - kilju_9*il10_rjt_9;
dx(17+36*9) = kiljb_10*il10_il10r_10*jak1_10*tyk2_10 - kilju_10*il10_rjt_10;

% il10cyto
dx(18) =      kilm_1*il10mrna_1-kilc_1*il10cyto_1-muilc_1*il10cyto_1;
dx(18+36) =   kilm_2*il10mrna_2-kilc_2*il10cyto_2-muilc_2*il10cyto_2;
dx(18+36*2) = kilm_3*il10mrna_3-kilc_3*il10cyto_3-muilc_3*il10cyto_3;
dx(18+36*3) = kilm_4*il10mrna_4-kilc_4*il10cyto_4-muilc_4*il10cyto_4;
dx(18+36*4) = kilm_5*il10mrna_5-kilc_5*il10cyto_5-muilc_5*il10cyto_5;
dx(18+36*5) = kilm_6*il10mrna_6-kilc_6*il10cyto_6-muilc_6*il10cyto_6;
dx(18+36*6) = kilm_7*il10mrna_7-kilc_7*il10cyto_7-muilc_7*il10cyto_7;
dx(18+36*7) = kilm_8*il10mrna_8-kilc_8*il10cyto_8-muilc_8*il10cyto_8;
dx(18+36*8) = kilm_9*il10mrna_9-kilc_9*il10cyto_9-muilc_9*il10cyto_9;
dx(18+36*9) = kilm_10*il10mrna_10-kilc_10*il10cyto_10-muilc_10*il10cyto_10;

% il10mrna
dx(19) =      0.4*kilnf_1*p_1*(nfkbnuclear_1/(ctf_1+nfkbnuclear_1)) + 0.6*
   kilsn_1*p_1*(stat3n_1/(ctf_stat3_1+stat3n_1)) - muilm_1*il10mrna_1;
dx(19+36) =   0.4*kilnf_2*p_2*(nfkbnuclear_2/(ctf_2+nfkbnuclear_2)) + 0.6*
   kilsn_2*p_2*(stat3n_2/(ctf_stat3_2+stat3n_2)) - muilm_2*il10mrna_2;
dx(19+36*2) = 0.4*kilnf_3*p_3*(nfkbnuclear_3/(ctf_3+nfkbnuclear_3)) + 0.6*
   kilsn_3*p_3*(stat3n_3/(ctf_stat3_3+stat3n_3)) - muilm_3*il10mrna_3;
dx(19+36*3) = 0.4*kilnf_4*p_4*(nfkbnuclear_4/(ctf_4+nfkbnuclear_4)) + 0.6*
   kilsn_4*p_4*(stat3n_4/(ctf_stat3_4+stat3n_4)) - muilm_4*il10mrna_4;
```

```
dx(19+36*4) = 0.4*kilnf_5*p_5*(nfkbnuclear_5/(ctf_5+nfkbnuclear_5)) + 0.6*
    kilsn_5*p_5*(stat3n_5/(ctf_stat3_5+stat3n_5)) - muilm_5*il10mrna_5;
dx(19+36*5) = 0.4*kilnf_6*p_6*(nfkbnuclear_6/(ctf_6+nfkbnuclear_6)) + 0.6*
    kilsn_6*p_6*(stat3n_6/(ctf_stat3_6+stat3n_6)) - muilm_6*il10mrna_6;
dx(19+36*6) = 0.4*kilnf_7*p_7*(nfkbnuclear_7/(ctf_7+nfkbnuclear_7)) + 0.6*
    kilsn_7*p_7*(stat3n_7/(ctf_stat3_7+stat3n_7)) - muilm_7*il10mrna_7;
dx(19+36*7) = 0.4*kilnf_8*p_8*(nfkbnuclear_8/(ctf_8+nfkbnuclear_8)) + 0.6*
    kilsn_8*p_8*(stat3n_8/(ctf_stat3_8+stat3n_8)) - muilm_8*il10mrna_8;
dx(19+36*8) = 0.4*kilnf_9*p_9*(nfkbnuclear_9/(ctf_9+nfkbnuclear_9)) + 0.6*
    kilsn_9*p_9*(stat3n_9/(ctf_stat3_9+stat3n_9)) - muilm_9*il10mrna_9;
dx(19+36*9) = 0.4*kilnf_10*p_10*(nfkbnuclear_10/(ctf_10+nfkbnuclear_10)) +
    0.6*kilsn_10*p_10*(stat3n_10/(ctf_stat3_10+stat3n_10)) - muilm_10*
    il10mrna_10;

% il10r
dx(20) =      -kilrb_1*il10ext*il10r_1+kilru_1*il10_il10r_1;
dx(20+36) =   -kilrb_2*il10ext*il10r_2+kilru_2*il10_il10r_2;
dx(20+36*2) = -kilrb_3*il10ext*il10r_3+kilru_3*il10_il10r_3;
dx(20+36*3) = -kilrb_4*il10ext*il10r_4+kilru_4*il10_il10r_4;
dx(20+36*4) = -kilrb_5*il10ext*il10r_5+kilru_5*il10_il10r_5;
dx(20+36*5) = -kilrb_6*il10ext*il10r_6+kilru_6*il10_il10r_6;
dx(20+36*6) = -kilrb_7*il10ext*il10r_7+kilru_7*il10_il10r_7;
dx(20+36*7) = -kilrb_8*il10ext*il10r_8+kilru_8*il10_il10r_8;
dx(20+36*8) = -kilrb_9*il10ext*il10r_9+kilru_9*il10_il10r_9;
dx(20+36*9) = -kilrb_10*il10ext*il10r_10+kilru_10*il10_il10r_10;

% jak1
```

```
dx(21) =      - kiljb_1*il10_il10r_1*jak1_1*tyk2_1 + kilju_1*il10_rjt_1;
dx(21+36) =   - kiljb_2*il10_il10r_2*jak1_2*tyk2_2 + kilju_2*il10_rjt_2;
dx(21+36*2) = - kiljb_3*il10_il10r_3*jak1_3*tyk2_3 + kilju_3*il10_rjt_3;
dx(21+36*3) = - kiljb_4*il10_il10r_4*jak1_4*tyk2_4 + kilju_4*il10_rjt_4;
dx(21+36*4) = - kiljb_5*il10_il10r_5*jak1_5*tyk2_5 + kilju_5*il10_rjt_5;
dx(21+36*5) = - kiljb_6*il10_il10r_6*jak1_6*tyk2_6 + kilju_6*il10_rjt_6;
dx(21+36*6) = - kiljb_7*il10_il10r_7*jak1_7*tyk2_7 + kilju_7*il10_rjt_7;
dx(21+36*7) = - kiljb_8*il10_il10r_8*jak1_8*tyk2_8 + kilju_8*il10_rjt_8;
dx(21+36*8) = - kiljb_9*il10_il10r_9*jak1_9*tyk2_9 + kilju_9*il10_rjt_9;
dx(21+36*9) = - kiljb_10*il10_il10r_10*jak1_10*tyk2_10 + kilju_10*
   il10_rjt_10;

% lps_tlr4
dx(22) =      kf1_1*lps*tlr4_1-kr1_1*lps_tlr4_1;
dx(22+36) =   kf1_2*lps*tlr4_2-kr1_2*lps_tlr4_2;
dx(22+36*2) = kf1_3*lps*tlr4_3-kr1_3*lps_tlr4_3;
dx(22+36*3) = kf1_4*lps*tlr4_4-kr1_4*lps_tlr4_4;
dx(22+36*4) = kf1_5*lps*tlr4_5-kr1_5*lps_tlr4_5;
dx(22+36*5) = kf1_6*lps*tlr4_6-kr1_6*lps_tlr4_6;
dx(22+36*6) = kf1_7*lps*tlr4_7-kr1_7*lps_tlr4_7;
dx(22+36*7) = kf1_8*lps*tlr4_8-kr1_8*lps_tlr4_8;
dx(22+36*8) = kf1_9*lps*tlr4_9-kr1_9*lps_tlr4_9;
dx(22+36*9) = kf1_10*lps*tlr4_10-kr1_10*lps_tlr4_10;

% nfkbcyto
dx(23) =      ti3_1*ikka_ikba_nfkbcyto_1-iln_1*kin_1*nfkbcyto_1-kf4_1*
   nfkbcyto_1*ikbacyto_1;
```

```
dx(23+36) =   ti3_2*ikka_ikba_nfkbcyto_2-iln_2*kin_2*nfkbcyto_2-kf4_2*
   nfkbcyto_2*ikbacyto_2;
dx(23+36*2) = ti3_3*ikka_ikba_nfkbcyto_3-iln_3*kin_3*nfkbcyto_3-kf4_3*
   nfkbcyto_3*ikbacyto_3;
dx(23+36*3) = ti3_4*ikka_ikba_nfkbcyto_4-iln_4*kin_4*nfkbcyto_4-kf4_4*
   nfkbcyto_4*ikbacyto_4;
dx(23+36*4) = ti3_5*ikka_ikba_nfkbcyto_5-iln_5*kin_5*nfkbcyto_5-kf4_5*
   nfkbcyto_5*ikbacyto_5;
dx(23+36*5) = ti3_6*ikka_ikba_nfkbcyto_6-iln_6*kin_6*nfkbcyto_6-kf4_6*
   nfkbcyto_6*ikbacyto_6;
dx(23+36*6) = ti3_7*ikka_ikba_nfkbcyto_7-iln_7*kin_7*nfkbcyto_7-kf4_7*
   nfkbcyto_7*ikbacyto_7;
dx(23+36*7) = ti3_8*ikka_ikba_nfkbcyto_8-iln_8*kin_8*nfkbcyto_8-kf4_8*
   nfkbcyto_8*ikbacyto_8;
dx(23+36*8) = ti3_9*ikka_ikba_nfkbcyto_9-iln_9*kin_9*nfkbcyto_9-kf4_9*
   nfkbcyto_9*ikbacyto_9;
dx(23+36*9) = ti3_10*ikka_ikba_nfkbcyto_10-iln_10*kin_10*nfkbcyto_10-kf4_10*
   nfkbcyto_10*ikbacyto_10;

% nfkbnuclear
dx(24) =      iln_1*kin_1*nfkbcyto_1/kv_1-kf4_1*nfkbnuclear_1*ikbanuclear_1;
dx(24+36) =   iln_2*kin_2*nfkbcyto_2/kv_2-kf4_2*nfkbnuclear_2*ikbanuclear_2;
dx(24+36*2) = iln_3*kin_3*nfkbcyto_3/kv_3-kf4_3*nfkbnuclear_3*ikbanuclear_3;
dx(24+36*3) = iln_4*kin_4*nfkbcyto_4/kv_4-kf4_4*nfkbnuclear_4*ikbanuclear_4;
dx(24+36*4) = iln_5*kin_5*nfkbcyto_5/kv_5-kf4_5*nfkbnuclear_5*ikbanuclear_5;
dx(24+36*5) = iln_6*kin_6*nfkbcyto_6/kv_6-kf4_6*nfkbnuclear_6*ikbanuclear_6;
dx(24+36*6) = iln_7*kin_7*nfkbcyto_7/kv_7-kf4_7*nfkbnuclear_7*ikbanuclear_7;
```

```
dx(24+36*7) = iln_8*kin_8*nfkbcyto_8/kv_8-kf4_8*nfkbnuclear_8*ikbanuclear_8;
dx(24+36*8) = iln_9*kin_9*nfkbcyto_9/kv_9-kf4_9*nfkbnuclear_9*ikbanuclear_9;
dx(24+36*9) = iln_10*kin_10*nfkbcyto_10/kv_10-kf4_10*nfkbnuclear_10*
    ikbanuclear_10;

% socs1cyto
dx(25) =      ks1_1*socs1mrna_1 - mus1c_1*socs1cyto_1;
dx(25+36) =   ks1_2*socs1mrna_2 - mus1c_2*socs1cyto_2;
dx(25+36*2) = ks1_3*socs1mrna_3 - mus1c_3*socs1cyto_3;
dx(25+36*3) = ks1_4*socs1mrna_4 - mus1c_4*socs1cyto_4;
dx(25+36*4) = ks1_5*socs1mrna_5 - mus1c_5*socs1cyto_5;
dx(25+36*5) = ks1_6*socs1mrna_6 - mus1c_6*socs1cyto_6;
dx(25+36*6) = ks1_7*socs1mrna_7 - mus1c_7*socs1cyto_7;
dx(25+36*7) = ks1_8*socs1mrna_8 - mus1c_8*socs1cyto_8;
dx(25+36*8) = ks1_9*socs1mrna_9 - mus1c_9*socs1cyto_9;
dx(25+36*9) = ks1_10*socs1mrna_10 - mus1c_10*socs1cyto_10;

% socs1mrna
dx(26) =      ks1st_1*stat3n_1 - ks1_1*socs1mrna_1 - mus1m_1*socs1mrna_1;
dx(26+36) =   ks1st_2*stat3n_2 - ks1_2*socs1mrna_2 - mus1m_2*socs1mrna_2;
dx(26+36*2) = ks1st_3*stat3n_3 - ks1_3*socs1mrna_3 - mus1m_3*socs1mrna_3;
dx(26+36*3) = ks1st_4*stat3n_4 - ks1_4*socs1mrna_4 - mus1m_4*socs1mrna_4;
dx(26+36*4) = ks1st_5*stat3n_5 - ks1_5*socs1mrna_5 - mus1m_5*socs1mrna_5;
dx(26+36*5) = ks1st_6*stat3n_6 - ks1_6*socs1mrna_6 - mus1m_6*socs1mrna_6;
dx(26+36*6) = ks1st_7*stat3n_7 - ks1_7*socs1mrna_7 - mus1m_7*socs1mrna_7;
dx(26+36*7) = ks1st_8*stat3n_8 - ks1_8*socs1mrna_8 - mus1m_8*socs1mrna_8;
dx(26+36*8) = ks1st_9*stat3n_9 - ks1_9*socs1mrna_9 - mus1m_9*socs1mrna_9;
```

```
dx(26+36*9) = ks1st_10*stat3n_10 - ks1_10*socs1mrna_10 - mus1m_10*
    socs1mrna_10;

% socs3cyto
dx(27) =      ks3_1*socs3mrna_1 - mus3c_1*socs3cyto_1;
dx(27+36) =   ks3_2*socs3mrna_2 - mus3c_2*socs3cyto_2;
dx(27+36*2) = ks3_3*socs3mrna_3 - mus3c_3*socs3cyto_3;
dx(27+36*3) = ks3_4*socs3mrna_4 - mus3c_4*socs3cyto_4;
dx(27+36*4) = ks3_5*socs3mrna_5 - mus3c_5*socs3cyto_5;
dx(27+36*5) = ks3_6*socs3mrna_6 - mus3c_6*socs3cyto_6;
dx(27+36*6) = ks3_7*socs3mrna_7 - mus3c_7*socs3cyto_7;
dx(27+36*7) = ks3_8*socs3mrna_8 - mus3c_8*socs3cyto_8;
dx(27+36*8) = ks3_9*socs3mrna_9 - mus3c_9*socs3cyto_9;
dx(27+36*9) = ks3_10*socs3mrna_10 - mus3c_10*socs3cyto_10;

% socs3mrna
dx(28) =      ks3st_1*stat3n_1 - ks3_1*socs3mrna_1 - mus3m_1*socs3mrna_1;
dx(28+36) =   ks3st_2*stat3n_2 - ks3_2*socs3mrna_2 - mus3m_2*socs3mrna_2;
dx(28+36*2) = ks3st_3*stat3n_3 - ks3_3*socs3mrna_3 - mus3m_3*socs3mrna_3;
dx(28+36*3) = ks3st_4*stat3n_4 - ks3_4*socs3mrna_4 - mus3m_4*socs3mrna_4;
dx(28+36*4) = ks3st_5*stat3n_5 - ks3_5*socs3mrna_5 - mus3m_5*socs3mrna_5;
dx(28+36*5) = ks3st_6*stat3n_6 - ks3_6*socs3mrna_6 - mus3m_6*socs3mrna_6;
dx(28+36*6) = ks3st_7*stat3n_7 - ks3_7*socs3mrna_7 - mus3m_7*socs3mrna_7;
dx(28+36*7) = ks3st_8*stat3n_8 - ks3_8*socs3mrna_8 - mus3m_8*socs3mrna_8;
dx(28+36*8) = ks3st_9*stat3n_9 - ks3_9*socs3mrna_9 - mus3m_9*socs3mrna_9;
dx(28+36*9) = ks3st_10*stat3n_10 - ks3_10*socs3mrna_10 - mus3m_10*
    socs3mrna_10;
```

```
1258
1259 % stat3a (formerly stat3a)
1260 dx(29) =      kstat_1*il10_rjt_1*stat3i_1^2*(1/(1+((socs1cyto_1+socs3cyto_1)
        /socsinf_1)^2)) - ksa_1*stat3a_1;
1261 dx(29+36) =   kstat_2*il10_rjt_2*stat3i_2^2*(1/(1+((socs1cyto_2+socs3cyto_2)
        /socsinf_2)^2)) - ksa_2*stat3a_2;
1262 dx(29+36*2) = kstat_3*il10_rjt_3*stat3i_3^2*(1/(1+((socs1cyto_3+socs3cyto_3)
        /socsinf_3)^2)) - ksa_3*stat3a_3;
1263 dx(29+36*3) = kstat_4*il10_rjt_4*stat3i_4^2*(1/(1+((socs1cyto_4+socs3cyto_4)
        /socsinf_4)^2)) - ksa_4*stat3a_4;
1264 dx(29+36*4) = kstat_5*il10_rjt_5*stat3i_5^2*(1/(1+((socs1cyto_5+socs3cyto_5)
        /socsinf_5)^2)) - ksa_5*stat3a_5;
1265 dx(29+36*5) = kstat_6*il10_rjt_6*stat3i_6^2*(1/(1+((socs1cyto_6+socs3cyto_6)
        /socsinf_6)^2)) - ksa_6*stat3a_6;
1266 dx(29+36*6) = kstat_7*il10_rjt_7*stat3i_7^2*(1/(1+((socs1cyto_7+socs3cyto_7)
        /socsinf_7)^2)) - ksa_7*stat3a_7;
1267 dx(29+36*7) = kstat_8*il10_rjt_8*stat3i_8^2*(1/(1+((socs1cyto_8+socs3cyto_8)
        /socsinf_8)^2)) - ksa_8*stat3a_8;
1268 dx(29+36*8) = kstat_9*il10_rjt_9*stat3i_9^2*(1/(1+((socs1cyto_9+socs3cyto_9)
        /socsinf_9)^2)) - ksa_9*stat3a_9;
1269 dx(29+36*9) = kstat_10*il10_rjt_10*stat3i_10^2*(1/(1+((socs1cyto_10+
        socs3cyto_10)/socsinf_10)^2)) - ksa_10*stat3a_10;
1270
1271 % stat3i (formerly stat3cyto)
1272 dx(30) =      - 2*kstat_1*il10_rjt_1*stat3i_1^2*(1/(1+((socs1cyto_1+
        socs3cyto_1)/socsinf_1)^2)) + ksnicyto_1*stat3ni_1;
1273 dx(30+36) =   - 2*kstat_2*il10_rjt_2*stat3i_2^2*(1/(1+((socs1cyto_2+
```

```
   socs3cyto_2)/socsinf_2)^2)) + ksnicyto_2*stat3ni_2;
dx(30+36*2) = - 2*kstat_3*il10_rjt_3*stat3i_3^2*(1/(1+((socs1cyto_3+
   socs3cyto_3)/socsinf_3)^2)) + ksnicyto_3*stat3ni_3;
dx(30+36*3) = - 2*kstat_4*il10_rjt_4*stat3i_4^2*(1/(1+((socs1cyto_4+
   socs3cyto_4)/socsinf_4)^2)) + ksnicyto_4*stat3ni_4;
dx(30+36*4) = - 2*kstat_5*il10_rjt_5*stat3i_5^2*(1/(1+((socs1cyto_5+
   socs3cyto_5)/socsinf_5)^2)) + ksnicyto_5*stat3ni_5;
dx(30+36*5) = - 2*kstat_6*il10_rjt_6*stat3i_6^2*(1/(1+((socs1cyto_6+
   socs3cyto_6)/socsinf_6)^2)) + ksnicyto_6*stat3ni_6;
dx(30+36*6) = - 2*kstat_7*il10_rjt_7*stat3i_7^2*(1/(1+((socs1cyto_7+
   socs3cyto_7)/socsinf_7)^2)) + ksnicyto_7*stat3ni_7;
dx(30+36*7) = - 2*kstat_8*il10_rjt_8*stat3i_8^2*(1/(1+((socs1cyto_8+
   socs3cyto_8)/socsinf_8)^2)) + ksnicyto_8*stat3ni_8;
dx(30+36*8) = - 2*kstat_9*il10_rjt_9*stat3i_9^2*(1/(1+((socs1cyto_9+
   socs3cyto_9)/socsinf_9)^2)) + ksnicyto_9*stat3ni_9;
dx(30+36*9) = - 2*kstat_10*il10_rjt_10*stat3i_10^2*(1/(1+((socs1cyto_10+
   socs3cyto_10)/socsinf_10)^2)) + ksnicyto_10*stat3ni_10;

% stat3n (formerly stat3_stat3n)
dx(31) =      ksa_1*stat3a_1 - ksni_1*stat3n_1;
dx(31+36) =   ksa_2*stat3a_2 - ksni_2*stat3n_2;
dx(31+36*2) = ksa_3*stat3a_3 - ksni_3*stat3n_3;
dx(31+36*3) = ksa_4*stat3a_4 - ksni_4*stat3n_4;
dx(31+36*4) = ksa_5*stat3a_5 - ksni_5*stat3n_5;
dx(31+36*5) = ksa_6*stat3a_6 - ksni_6*stat3n_6;
dx(31+36*6) = ksa_7*stat3a_7 - ksni_7*stat3n_7;
dx(31+36*7) = ksa_8*stat3a_8 - ksni_8*stat3n_8;
```

```
dx(31+36*8) = ksa_9*stat3a_9 - ksni_9*stat3n_9;
dx(31+36*9) = ksa_10*stat3a_10 - ksni_10*stat3n_10;

% stat3ni
dx(32) =      ksni_1*stat3n_1 - ksnicyto_1*stat3n_1;
dx(32+36) =   ksni_2*stat3n_2 - ksnicyto_2*stat3n_2;
dx(32+36*2) = ksni_3*stat3n_3 - ksnicyto_3*stat3n_3;
dx(32+36*3) = ksni_4*stat3n_4 - ksnicyto_4*stat3n_4;
dx(32+36*4) = ksni_5*stat3n_5 - ksnicyto_5*stat3n_5;
dx(32+36*5) = ksni_6*stat3n_6 - ksnicyto_6*stat3n_6;
dx(32+36*6) = ksni_7*stat3n_7 - ksnicyto_7*stat3n_7;
dx(32+36*7) = ksni_8*stat3n_8 - ksnicyto_8*stat3n_8;
dx(32+36*8) = ksni_9*stat3n_9 - ksnicyto_9*stat3n_9;
dx(32+36*9) = ksni_10*stat3n_10 - ksnicyto_10*stat3n_10;

% tlr4
dx(33) =      -kf1_1*lps*tlr4_1+kr1_1*lps_tlr4_1;
dx(33+36) =   -kf1_2*lps*tlr4_2+kr1_2*lps_tlr4_2;
dx(33+36*2) = -kf1_3*lps*tlr4_3+kr1_3*lps_tlr4_3;
dx(33+36*3) = -kf1_4*lps*tlr4_4+kr1_4*lps_tlr4_4;
dx(33+36*4) = -kf1_5*lps*tlr4_5+kr1_5*lps_tlr4_5;
dx(33+36*5) = -kf1_6*lps*tlr4_6+kr1_6*lps_tlr4_6;
dx(33+36*6) = -kf1_7*lps*tlr4_7+kr1_7*lps_tlr4_7;
dx(33+36*7) = -kf1_8*lps*tlr4_8+kr1_8*lps_tlr4_8;
dx(33+36*8) = -kf1_9*lps*tlr4_9+kr1_9*lps_tlr4_9;
dx(33+36*9) = -kf1_10*lps*tlr4_10+kr1_10*lps_tlr4_10;

```

```
% tnfa_tnfar
dx(34) =      kf3_1*tnfaext*tnfar_1-kr3_1*tnfa_tnfar_1;
dx(34+36) =   kf3_2*tnfaext*tnfar_2-kr3_2*tnfa_tnfar_2;
dx(34+36*2) = kf3_3*tnfaext*tnfar_3-kr3_3*tnfa_tnfar_3;
dx(34+36*3) = kf3_4*tnfaext*tnfar_4-kr3_4*tnfa_tnfar_4;
dx(34+36*4) = kf3_5*tnfaext*tnfar_5-kr3_5*tnfa_tnfar_5;
dx(34+36*5) = kf3_6*tnfaext*tnfar_6-kr3_6*tnfa_tnfar_6;
dx(34+36*6) = kf3_7*tnfaext*tnfar_7-kr3_7*tnfa_tnfar_7;
dx(34+36*7) = kf3_8*tnfaext*tnfar_8-kr3_8*tnfa_tnfar_8;
dx(34+36*8) = kf3_9*tnfaext*tnfar_9-kr3_9*tnfa_tnfar_9;
dx(34+36*9) = kf3_10*tnfaext*tnfar_10-kr3_10*tnfa_tnfar_10;

% tnfacyto
dx(35) =      tnfa_trans_1*tnfamrna_1-ksec_1*tnfacyto_1-mutnc_1*tnfacyto_1;
dx(35+36) =   tnfa_trans_2*tnfamrna_2-ksec_2*tnfacyto_2-mutnc_2*tnfacyto_2;
dx(35+36*2) = tnfa_trans_3*tnfamrna_3-ksec_3*tnfacyto_3-mutnc_3*tnfacyto_3;
dx(35+36*3) = tnfa_trans_4*tnfamrna_4-ksec_4*tnfacyto_4-mutnc_4*tnfacyto_4;
dx(35+36*4) = tnfa_trans_5*tnfamrna_5-ksec_5*tnfacyto_5-mutnc_5*tnfacyto_5;
dx(35+36*5) = tnfa_trans_6*tnfamrna_6-ksec_6*tnfacyto_6-mutnc_6*tnfacyto_6;
dx(35+36*6) = tnfa_trans_7*tnfamrna_7-ksec_7*tnfacyto_7-mutnc_7*tnfacyto_7;
dx(35+36*7) = tnfa_trans_8*tnfamrna_8-ksec_8*tnfacyto_8-mutnc_8*tnfacyto_8;
dx(35+36*8) = tnfa_trans_9*tnfamrna_9-ksec_9*tnfacyto_9-mutnc_9*tnfacyto_9;
dx(35+36*9) = tnfa_trans_10*tnfamrna_10-ksec_10*tnfacyto_10-mutnc_10*
   tnfacyto_10;

% tnfamrna
dx(36) =      sm_1*p_1*(nfkbnuclear_1/(ctf_1+nfkbnuclear_1))*(1/(1+(
```

```
    socs3cyto_1/socs3inf_1)^2))-mutnm_1*tnfamrna_1;
dx(36+36) =   sm_2*p_2*(nfkbnuclear_2/(ctf_2+nfkbnuclear_2))*(1/(1+(
    socs3cyto_2/socs3inf_2)^2))-mutnm_2*tnfamrna_2;
dx(36+36*2) = sm_3*p_3*(nfkbnuclear_3/(ctf_3+nfkbnuclear_3))*(1/(1+(
    socs3cyto_3/socs3inf_3)^2))-mutnm_3*tnfamrna_3;
dx(36+36*3) = sm_4*p_4*(nfkbnuclear_4/(ctf_4+nfkbnuclear_4))*(1/(1+(
    socs3cyto_4/socs3inf_4)^2))-mutnm_4*tnfamrna_4;
dx(36+36*4) = sm_5*p_5*(nfkbnuclear_5/(ctf_5+nfkbnuclear_5))*(1/(1+(
    socs3cyto_5/socs3inf_5)^2))-mutnm_5*tnfamrna_5;
dx(36+36*5) = sm_6*p_6*(nfkbnuclear_6/(ctf_6+nfkbnuclear_6))*(1/(1+(
    socs3cyto_6/socs3inf_6)^2))-mutnm_6*tnfamrna_6;
dx(36+36*6) = sm_7*p_7*(nfkbnuclear_7/(ctf_7+nfkbnuclear_7))*(1/(1+(
    socs3cyto_7/socs3inf_7)^2))-mutnm_7*tnfamrna_7;
dx(36+36*7) = sm_8*p_8*(nfkbnuclear_8/(ctf_8+nfkbnuclear_8))*(1/(1+(
    socs3cyto_8/socs3inf_8)^2))-mutnm_8*tnfamrna_8;
dx(36+36*8) = sm_9*p_9*(nfkbnuclear_9/(ctf_9+nfkbnuclear_9))*(1/(1+(
    socs3cyto_9/socs3inf_9)^2))-mutnm_9*tnfamrna_9;
dx(36+36*9) = sm_10*p_10*(nfkbnuclear_10/(ctf_10+nfkbnuclear_10))*(1/(1+(
    socs3cyto_10/socs3inf_10)^2))-mutnm_10*tnfamrna_10;

% tnfar
dx(37) =      -kf3_1*tnfaext*tnfar_1+kr3_1*tnfa_tnfar_1;
dx(37+36) =   -kf3_2*tnfaext*tnfar_2+kr3_2*tnfa_tnfar_2;
dx(37+36*2) = -kf3_3*tnfaext*tnfar_3+kr3_3*tnfa_tnfar_3;
dx(37+36*3) = -kf3_4*tnfaext*tnfar_4+kr3_4*tnfa_tnfar_4;
dx(37+36*4) = -kf3_5*tnfaext*tnfar_5+kr3_5*tnfa_tnfar_5;
dx(37+36*5) = -kf3_6*tnfaext*tnfar_6+kr3_6*tnfa_tnfar_6;
```

```
dx(37+36*6) = -kf3_7*tnfaext*tnfar_7+kr3_7*tnfa_tnfar_7;
dx(37+36*7) = -kf3_8*tnfaext*tnfar_8+kr3_8*tnfa_tnfar_8;
dx(37+36*8) = -kf3_9*tnfaext*tnfar_9+kr3_9*tnfa_tnfar_9;
dx(37+36*9) = -kf3_10*tnfaext*tnfar_10+kr3_10*tnfa_tnfar_10;

% tyk2
dx(38) =      - kiljb_1*il10_il10r_1*jak1_1*tyk2_1 + kilju_1*il10_rjt_1;
dx(38+36) =   - kiljb_2*il10_il10r_2*jak1_2*tyk2_2 + kilju_2*il10_rjt_2;
dx(38+36*2) = - kiljb_3*il10_il10r_3*jak1_3*tyk2_3 + kilju_3*il10_rjt_3;
dx(38+36*3) = - kiljb_4*il10_il10r_4*jak1_4*tyk2_4 + kilju_4*il10_rjt_4;
dx(38+36*4) = - kiljb_5*il10_il10r_5*jak1_5*tyk2_5 + kilju_5*il10_rjt_5;
dx(38+36*5) = - kiljb_6*il10_il10r_6*jak1_6*tyk2_6 + kilju_6*il10_rjt_6;
dx(38+36*6) = - kiljb_7*il10_il10r_7*jak1_7*tyk2_7 + kilju_7*il10_rjt_7;
dx(38+36*7) = - kiljb_8*il10_il10r_8*jak1_8*tyk2_8 + kilju_8*il10_rjt_8;
dx(38+36*8) = - kiljb_9*il10_il10r_9*jak1_9*tyk2_9 + kilju_9*il10_rjt_9;
dx(38+36*9) = - kiljb_10*il10_il10r_10*jak1_10*tyk2_10 + kilju_10*
   il10_rjt_10;

% il10act: il10mrna produced by STAT3
dx(39) =       0.6*kilsn_1*p_1*(stat3n_1/(ctf_stat3_1+stat3n_1)) - muilm_1*
   il10act_1;
dx(39+36) =    0.6*kilsn_2*p_2*(stat3n_2/(ctf_stat3_2+stat3n_2)) - muilm_2*
   il10act_2;
dx(39+36*2) =  0.6*kilsn_3*p_3*(stat3n_3/(ctf_stat3_3+stat3n_3)) - muilm_3*
   il10act_3;
dx(39+36*3) =  0.6*kilsn_4*p_4*(stat3n_4/(ctf_stat3_4+stat3n_4)) - muilm_4*
   il10act_4;
```

```
1384 dx(39+36*4) =  0.6*kilsn_5*p_5*(stat3n_5/(ctf_stat3_5+stat3n_5)) - muilm_5*
        il10act_5;
1385 dx(39+36*5) =  0.6*kilsn_6*p_6*(stat3n_6/(ctf_stat3_6+stat3n_6)) - muilm_6*
        il10act_6;
1386 dx(39+36*6) =  0.6*kilsn_7*p_7*(stat3n_7/(ctf_stat3_7+stat3n_7)) - muilm_7*
        il10act_7;
1387 dx(39+36*7) =  0.6*kilsn_8*p_8*(stat3n_8/(ctf_stat3_8+stat3n_8)) - muilm_8*
        il10act_8;
1388 dx(39+36*8) =  0.6*kilsn_9*p_9*(stat3n_9/(ctf_stat3_9+stat3n_9)) - muilm_9*
        il10act_9;
1389 dx(39+36*9) =  0.6*kilsn_10*p_10*(stat3n_10/(ctf_stat3_10+stat3n_10)) -
        muilm_10*il10act_10;
1390 end
```

Bibliography

[1] Cell Signaling | Learn Science at Scitable, . URL https://www.nature.com/scitable/topicpage/cell-signaling-14047077/.

[2] The top 10 causes of death, . URL https://www.who.int/news-room/fact-sheets/detail/the-top-10-causes-of-death.

[3] Translation: DNA to mRNA to Protein | Learn Science at Scitable, . URL https://www.nature.com/scitable/topicpage/translation-dna-to-mrna-to-protein-393/.

[4] Neil R. Aggarwal, Landon S. King, and Franco R. D'Alessio. Diverse Macrophage Populations Mediate Acute Lung Inflammation and Resolution. American Journal of Physiology-Lung Cellular and Molecular Physiology, 306(8):L709–L725, February 2014. ISSN 1040-0605. doi: 10.1152/ajplung.00341.2013. URL http://www.physiology.org/doi/abs/10.1152/ajplung.00341.2013.

[5] Parya Aghasafari, Israr Bin M. Ibrahim, and Ramana Pidaparti. Strain-induced inflammation in pulmonary alveolar tissue due to mechanical ventilation. Biomechanics and Modeling in Mechanobiology, 16(4):1103–1118, August 2017. ISSN 1617-7940. doi: 10.1007/s10237-017-0879-5. URL https://doi.org/10.1007/s10237-017-0879-5.

[6] Kellie J. Archer and Ryan V. Kimes. Empirical characterization of random forest variable importance measures. Computational Statistics & Data Analysis, 52(4):2249–

2260, January 2008. ISSN 0167-9473. doi: 10.1016/j.csda.2007.08.015. URL http://www.sciencedirect.com/science/article/pii/S0167947307003076.

[7] Jason H. T. Bates and Charles G. Irvin. Time dependence of recruitment and derecruitment in the lung: a theoretical model. Journal of Applied Physiology, 93(2):705–713, August 2002. ISSN 8750-7587. doi: 10.1152/japplphysiol.01274.2001. URL https://journals.physiology.org/doi/full/10.1152/japplphysiol.01274.2001. Publisher: American Physiological Society.

[8] Yoav Benjamini and Yosef Hochberg. Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing. Journal of the Royal Statistical Society. Series B (Methodological), 57(1):289–300, 1995. ISSN 0035-9246. URL https://www.jstor.org/stable/2346101. Publisher: [Royal Statistical Society, Wiley].

[9] Pascal Bezel, Alan Valaperti, Urs Steiner, Dieter Scholtze, Stephan Wieser, Maya Vonow-Eisenring, Andrea Widmer, Benedikt Kowalski, Malcolm Kohler, and Daniel P. Franzen. Evaluation of cytokines in the tumor microenvironment of lung cancer using bronchoalveolar lavage fluid analysis. Cancer immunology, immunotherapy: CII, January 2021. ISSN 1432-0851. doi: 10.1007/s00262-020-02798-z.

[10] Alessandro Boianelli, Van Nguyen, Thomas Ebensen, Kai Schulze, Esther Wilk, Niharika Sharma, Sabine Stegemann-Koniszewski, Dunja Bruder, Franklin Toapanta, Carlos Guzmán, Michael Meyer-Hermann, Esteban Hernandez-Vargas, Alessandro Boianelli, Van Kinh Nguyen, Thomas Ebensen, Kai Schulze, Esther Wilk, Niharika Sharma, Sabine Stegemann-Koniszewski, Dunja Bruder, Franklin R. Toapanta, Carlos A. Guzmán, Michael Meyer-Hermann, and Esteban A. Hernandez-Vargas. Modeling Influenza Virus Infection: A Roadmap for Influenza Research. Viruses, 7(10): 5274–5304, October 2015. doi: 10.3390/v7102875. URL https://www.mdpi.com/1999-4915/7/10/2875.

[11] Maria Carla Bosco. Macrophage polarization: Reaching across the aisle? Journal of Allergy and Clinical Immunology, 143(4):1348–1350, April 2019. ISSN 0091-6749, 1097-6825. doi: 10.1016/j.jaci.2018.12.995. URL https://www.jacionline.org/article/S0091-6749(19)30005-3/abstract. Publisher: Elsevier.

[12] Christina Brandenberger and Christian Mühlfeld. Mechanisms of lung aging. Cell and Tissue Research, 367(3):469–480, March 2017. ISSN 1432-0878. doi: 10.1007/s00441-016-2511-x. URL https://doi.org/10.1007/s00441-016-2511-x.

[13] Bear F. Braumoeller. Hypothesis Testing and Multiplicative Interaction Terms. International Organization, 58(4):807–820, 2004. ISSN 0020-8183. URL http://www.jstor.org/stable/3877804. Publisher: [MIT Press, University of Wisconsin Press, Cambridge University Press, International Organization Foundation].

[14] Bryan N. Brown, Ian M. Price, Franklin R. Toapanta, Dilhari R. DeAlmeida, Clayton A. Wiley, Ted M. Ross, Tim D. Oury, and Yoram Vodovotz. An Agent-Based Model of Inflammation and Fibrosis Following Particulate Exposure in the Lung. Mathematical Biosciences, 231(2):186–196, June 2011. ISSN 0025-5564. doi: 10.1016/j.mbs.2011.03.005. URL http://www.sciencedirect.com/science/article/pii/S0025556411000356.

[15] Giuseppe Bruno, Serena Perelli, Claudia Fabrizio, and Giovanni Battista Buccoliero. Short-term outcomes in individuals aged 75 or older with severe coronavirus disease (COVID-19): First observations from an Infectious Diseases Unit in Southern Italy. The Journal of Infection, May 2020. ISSN 0163-4453. doi: 10.1016/j.jinf.2020.05.024. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7224683/.

[16] Cynthia H. Canan, Nandan S. Gokhale, Bridget Carruthers, William P. Lafuse, Larry S. Schlesinger, Jordi B. Torrelles, and Joanne Turner. Characterization of Lung Inflammation and its Impact on Macrophage Function in Aging. Journal of

Leukocyte Biology, 96(3):473–480, September 2014. ISSN 0741-5400, 1938-3673. doi: 10.1189/jlb.4A0214-093RR. URL http://www.jleukbio.org.proxy.library.vcu.edu/content/96/3/473.

[17] Martina Cantone, Guido Santos, Pia Wentker, Xin Lai, and Julio Vera. Multiplicity of Mathematical Modeling Strategies to Search for Molecular and Cellular Insights into Bacteria Lung Infection. Frontiers in Physiology, 8, 2017. ISSN 1664-042X. doi: 10.3389/fphys.2017.00645. URL http://journal.frontiersin.org/article/10.3389/fphys.2017.00645/full.

[18] Pere-Joan Cardona, Martí Català, Marta Arch, Lilibeth Arias, Sergio Alonso, Paula Cardona, Daniel López, Cristina Vilaplana, and Clara Prats. Can systems immunology lead tuberculosis eradication? Current Opinion in Systems Biology, 12: 53–60, December 2018. ISSN 2452-3100. doi: 10.1016/j.coisb.2018.10.004. URL http://www.sciencedirect.com/science/article/pii/S2452310018300763.

[19] Alison J. Carey, Chee K. Tan, and Glen C. Ulett. Infection-induced IL-10 and JAK-STAT. JAK-STAT, 1(3):159–167, July 2012. ISSN 2162-3988. doi: 10.4161/jkst.19918. URL http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3670239/.

[20] Chris Carter and Joy Notter. COVID-19 disease: a critical care perspective. Clinics in Integrated Care, 1:100003, July 2020. ISSN 2666-8696. doi: 10.1016/j.intcar.2020.100003. URL http://www.sciencedirect.com/science/article/pii/S2666869620300038.

[21] J. Geoffrey Chase, Jean-Charles Preiser, Jennifer L. Dickson, Antoine Pironet, Yeong Shiong Chiew, Christopher G. Pretty, Geoffrey M. Shaw, Balazs Benyo, Knut Moeller, Soroush Safaei, Merryn Tawhai, Peter Hunter, and Thomas Desaive. Next-generation, personalised, model-based critical care medicine: a state-of-the art review of in silico virtual patient models, methods, and cohorts, and how to validation

them. BioMedical Engineering OnLine, 17(1):1–29, December 2018. ISSN 1475-925X. doi: 10.1186/s12938-018-0455-y. URL https://biomedical-engineering-online.biomedcentral.com/articles/10.1186/s12938-018-0455-y.

[22] Igor L. Chernyavsky, Huguette Croisier, Lloyd A. C. Chapman, Laura S. Kimpton, Jonathan E. Hiorns, Bindi S. Brook, Oliver E. Jensen, Charlotte K. Billington, Ian P. Hall, and Simon R. Johnson. The Role of Inflammation Resolution Speed in Airway Smooth Muscle Mass Accumulation in Asthma: Insight from a Theoretical Model. PLOS ONE, 9(3):e90162, March 2014. ISSN 1932-6203. doi: 10.1371/journal.pone.0090162. URL http://journals.plos.org/plosone/article?id=10.1371/journal.pone.0090162.

[23] Ben A. Croker, Hiu Kiu, and Sandra E. Nicholson. SOCS regulation of the JAK/-STAT signalling pathway. Seminars in Cell & Developmental Biology, 19(4):414–422, August 2008. ISSN 1084-9521. doi: 10.1016/j.semcdb.2008.07.010. URL http://www.sciencedirect.com/science/article/pii/S1084952108000499.

[24] P. S. Crooke, J. J. Marini, and J. R. Hotchkiss. A new look at the stress index for lung injury. Journal of Biological Systems, 13(03):261–272, September 2005. ISSN 0218-3390. doi: 10.1142/S0218339005001549. URL https://www.worldscientific.com/doi/abs/10.1142/S0218339005001549. Publisher: World Scientific Publishing Co.

[25] Lynn M. Crosby and Christopher M. Waters. Epithelial Repair Mechanisms in the Lung. American Journal of Physiology-Lung Cellular and Molecular Physiology, 298 (6):L715–L731, April 2010. ISSN 1040-0605. doi: 10.1152/ajplung.00361.2009. URL http://www.physiology.org/doi/abs/10.1152/ajplung.00361.2009.

[26] Roberto Santa Cruz, Fernando Villarejo, Alvaro Figueroa, Marcela Cortés-Jofré, Juan Gagliardi, and Marcelo Navarrete. Mortality in Critically Ill Elderly Individuals Re-

ceiving Mechanical Ventilation. Respiratory Care, 64(4):473–483, April 2019. ISSN 0020-1324, 1943-3654. doi: 10.4187/respcare.06586. URL http://rc.rcjournal.com/content/64/4/473. Publisher: Respiratory Care Section: Systematic Review.

[27] Judy Day, Avner Friedman, and Larry S. Schlesinger. Modeling the Immune Rheostat of Macrophages in the Lung in Response to Infection. Proceedings of the National Academy of Sciences, 106(27):11246–11251, July 2009. ISSN 0027-8424, 1091-6490. doi: 10.1073/pnas.0904846106. URL http://www.pnas.org.proxy.library.vcu.edu/content/106/27/11246.

[28] Ioannis D. Dimitriou, Liliana Clemenza, Andrew J. Scotter, Grace Chen, Fiona M. Guerra, and Robert Rottapel. Putting out the fire: coordinated suppression of the innate and adaptive immune systems by SOCS1 and SOCS3 proteins. Immunological Reviews, 224(1):265–283, 2008. ISSN 1600-065X. doi: 10.1111/j.1600-065X.2008.00659.x. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1600-065X.2008.00659.x.

[29] Vu Dinh, Ann E. Rundell, and Gregery T. Buzzard. Experimental Design for Dynamics Identification of Cellular Processes. Bulletin of Mathematical Biology, 76(3):597–626, March 2014. ISSN 0092-8240, 1522-9602. doi: 10.1007/s11538-014-9935-9. URL http://link.springer.com/10.1007/s11538-014-9935-9.

[30] Tamas Dolinay, Blanca E. Himes, Maya Shumyatcher, Gladys Gray Lawrence, and Susan S. Margulies. Integrated Stress Response Mediates Epithelial Injury in Mechanical Ventilation. American Journal of Respiratory Cell and Molecular Biology, 57(2):193–203, March 2017. ISSN 1044-1549. doi: 10.1165/rcmb.2016-0404OC. URL http://www.atsjournals.org/doi/full/10.1165/rcmb.2016-0404OC. Publisher: American Thoracic Society - AJRCMB.

[31] Elisa Domínguez-Hüttinger, Neville J. Boon, Thomas B. Clarke, and Reiko J. Tanaka.

Mathematical Modeling of Streptococcus pneumoniae Colonization, Invasive Infection and Treatment. Frontiers in Physiology, 8, March 2017. ISSN 1664-042X. doi: 10.3389/fphys.2017.00115. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5332394/.

[32] M. M. Donahue, G. T. Buzzard, and A. E. Rundell. Experiment design through dynamical characterisation of non-linear systems biology models utilising sparse grids. IET Systems Biology, 4(4):249–262, July 2010. ISSN 1751-8857. doi: 10.1049/iet-syb.2009.0031. URL https://digital-library.theiet.org/content/journals/10.1049/iet-syb.2009.0031. Publisher: IET Digital Library.

[33] Wubei Dong, Xiaojia Tang, Yihai Yu, Roger Nilsen, Rosemary Kim, James Griffith, Jonathan Arnold, and H.-Bernd Schüttler. Systems Biology of the Clock in Neurospora crassa. PLOS ONE, 3(8):e3105, August 2008. ISSN 1932-6203. doi: 10.1371/journal.pone.0003105. URL http://journals.plos.org/plosone/article?id=10.1371/journal.pone.0003105. Publisher: Public Library of Science.

[34] F. Driessler, K. Venstrom, R. Sabat, K. Asadullah, and A. J. Schottelius. Molecular mechanisms of interleukin-10-mediated inhibition of NF-κB activity: a role for p50. Clinical & Experimental Immunology, 135(1):64–73, 2004. ISSN 1365-2249. doi: https://doi.org/10.1111/j.1365-2249.2004.02342.x. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1365-2249.2004.02342.x. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1365-2249.2004.02342.x.

[35] Sean Quan Du and Weiming Yuan. Mathematical modeling of interaction between innate and adaptive immune responses in COVID-19 and implications for viral pathogenesis. Journal of Medical Virology, 92(9):1615–1628, 2020. ISSN 1096-9071. doi: https://doi.org/10.1002/jmv.25866.

URL https://onlinelibrary.wiley.com/doi/abs/10.1002/jmv.25866. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/jmv.25866.

[36] Joanne L. Dunster. The macrophage and its role in inflammation and tissue repair: mathematical and systems biology approaches. WIREs Systems Biology and Medicine, 8(1):87–99, 2016. ISSN 1939-005X. doi: https://doi.org/10.1002/wsbm.1320. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/wsbm.1320. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/wsbm.1320.

[37] Martin Eberhardt, Xin Lai, Namrata Tomar, Shailendra Gupta, Bernd Schmeck, Alexander Steinkasserer, Gerold Schuler, and Julio Vera. Third-Kind Encounters in Biomedicine: Immunology Meets Mathematics and Informatics to Become Quantitative and Predictive. Methods in Molecular Biology (Clifton, N.J.), 1386:135–179, 2016. ISSN 1940-6029. doi: 10.1007/978-1-4939-3283-2_9.

[38] Bard Ermentrout. Simulating, Analyzing, and Animating Dynamical Systems. Software, Environments and Tools. Society for Industrial and Applied Mathematics, January 2002. ISBN 978-0-89871-506-4. doi: 10.1137/1.9780898718195. URL http://epubs.siam.org/doi/book/10.1137/1.9780898718195.

[39] Yan Feng, Yaw Amoateng-Adjepong, David Kaufman, Cristina Gheorghe, and Constantine A Manthous. Age, duration of mechanical ventilation, and outcomes of patients who are critically ill. Chest, 136(3):759–764, 2009.

[40] Anna S Frank, Kamila Larripa, Hwayeon Ryu, Ryan G. Snodgrass, and Susanna Röblitz. Bifurcation and sensitivity analysis reveal key drivers of multistability in a model of macrophage polarization. Journal of Theoretical Biology, 509:110511, January 2021. ISSN 0022-5193. doi: 10.1016/j.jtbi.2020.110511. URL http://www.sciencedirect.com/science/article/pii/S0022519320303660.

[41] Aaron Gardner, Lee A. Borthwick, and Andrew J. Fisher. Lung Epithelial Wound

Healing in Health and Disease. Expert Review of Respiratory Medicine, 4(5):647–660, October 2010. ISSN 1747-6348. doi: 10.1586/ers.10.62. URL https://doi.org/10.1586/ers.10.62.

[42] Andrey Golov, Sergey Simakov, Yan Naing Soe, Roman Pryamonosov, Ospan Mynbaev, and Alexander Kholodov. Multiscale CT-Based Computational Modeling of Alveolar Gas Exchange during Artificial Lung Ventilation, Cluster (Biot) and Periodic (Cheyne-Stokes) Breathings and Bronchial Asthma Attack. Computation, 5(1): 11, February 2017. doi: 10.3390/computation5010011. URL http://www.mdpi.com/2079-3197/5/1/11.

[43] Siamon Gordon. Alternative Activation of Macrophages. Nature Reviews Immunology, 3(1):23–35, January 2003. ISSN 1474-1741. doi: 10.1038/nri978. URL http://www.nature.com/articles/nri978.

[44] Jochen Grommes and Oliver Soehnlein. Contribution of Neutrophils to Acute Lung Injury. Molecular Medicine, 17(3-4):293–307, 2011. ISSN 1076-1551. doi: 10.2119/molmed.2010.00138. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3060975/.

[45] F. J. J. Halbertsma, M. Vaneker, G. J. Scheffer, and J. G. van der Hoeven. Cytokines and biotrauma in ventilator-induced lung injury: a critical review of the literature. The Netherlands Journal of Medicine, 63(10):382–392, November 2005. ISSN 0300-2977.

[46] Katharine L. Hamlington, Bradford J. Smith, Gilman B. Allen, and Jason H. T. Bates. Predicting ventilator-induced lung injury using a lung injury cost function. Journal of Applied Physiology, 121(1):106–114, May 2016. ISSN 8750-7587. doi: 10.1152/japplphysiol.00096.2016. URL https://journals.physiology.org/doi/full/10.1152/japplphysiol.00096.2016. Publisher: American Physiological Society.

[47] Esteban A. Hernandez-Vargas and Jorge X. Velasco-Hernandez. In-host Mathematical Modelling of COVID-19 in Humans. Annual Reviews in Control, September 2020. ISSN 1367-5788. doi: 10.1016/j.arcontrol.2020.09.006. URL http://www.sciencedirect.com/science/article/pii/S1367578820300638.

[48] Susanne Herold, Konstantin Mayer, and Juergen Lohmeyer. Acute Lung Injury: How Macrophages Orchestrate Resolution of Inflammation and Tissue Repair. Frontiers in Immunology, 2, November 2011. ISSN 1664-3224. doi: 10.3389/fimmu.2011.00065. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3342347/.

[49] Moniek Heusinkveld, Peggy J. de Vos van Steenwijk, Renske Goedemans, Tamara H. Ramwadhdoebe, Arko Gorter, Marij J. P. Welters, Thorbald van Hall, and Sjoerd H. van der Burg. M2 Macrophages Induced by Prostaglandin E2 and IL-6 from Cervical Carcinoma Are Switched to Activated M1 Macrophages by CD4+ Th1 Cells. The Journal of Immunology, 187(3):1157–1165, August 2011. ISSN 0022-1767, 1550-6606. doi: 10.4049/jimmunol.1100889. URL http://www.jimmunol.org/content/187/3/1157.

[50] John Hiscott, Hakju Kwon, and Pierre Génin. Hostile takeovers: viral appropriation of the NF-kB pathway. The Journal of Clinical Investigation, 107(2):143–151, January 2001. ISSN 0021-9738. doi: 10.1172/JCI11918. URL https://www.jci.org/articles/view/11918. Publisher: American Society for Clinical Investigation.

[51] Andrew P. Hutchins, Diego Diez, and Diego Miranda-Saavedra. The IL-10/STAT3-mediated anti-inflammatory response: recent developments and future challenges. Briefings in Functional Genomics, 12(6):489–498, November 2013. ISSN 2041-2649. doi: 10.1093/bfgp/elt028. URL https://academic-oup-com.proxy.library.vcu.edu/bfg/article/12/6/489/238388/The-IL-10-STAT3-mediated-anti-inflammatory.

[52] I. Bin M. Ibrahim, R. M. Pidaparti, and K. R. Ward. Evaluation of Ventilation-Induced

Lung Inflammation Through Multi-Scale Simulations. IEEE Journal of Translational Engineering in Health and Medicine, 6:1–7, 2018. ISSN 2168-2372. doi: 10.1109/JTEHM.2018.2795031. Conference Name: IEEE Journal of Translational Engineering in Health and Medicine.

[53] Joanneke E. Jansen, Eamonn A. Gaffney, Jonathan Wagg, and Mark C. Coles. Combining Mathematical Models With Experimentation to Drive Novel Mechanistic Insights Into Macrophage Function. Frontiers in Immunology, 10, 2019. ISSN 1664-3224. doi: 10.3389/fimmu.2019.01283. URL https://www.frontiersin.org/articles/10.3389/fimmu.2019.01283/full#F1. Publisher: Frontiers.

[54] Laura K. Johnston, Cliff R. Rims, Sean E. Gill, John K. McGuire, and Anne M. Manicone. Pulmonary Macrophage Subpopulations in the Induction and Resolution of Acute Lung Injury. American Journal of Respiratory Cell and Molecular Biology, November 2012. doi: 10.1165/rcmb.2012-0090OC. URL http://www.atsjournals.org/doi/abs/10.1165/rcmb.2012-0090OC.

[55] V. Roshan Joseph. Space-filling designs for computer experiments: A review. Quality Engineering, 28(1), January 2016. ISSN 0898-2112. doi: 10.1080/08982112.2015.1100447. URL https://www.osti.gov/pages/biblio/1405184-space-filling-designs-computer-experiments-review. Institution: Georgia Tech Research Corp., Atlanta, GA (United States) Number: DOE-GT-0010548-9 Publisher: American Society for Quality Control.

[56] JongWon Kim, Rebecca L. Heise, Angela M. Reynolds, and Ramana M. Pidaparti. Aging effects on airflow dynamics and lung function in human bronchioles. PloS One, 12(8):e0183654, 2017. ISSN 1932-6203. doi: 10.1371/journal.pone.0183654.

[57] JongWon Kim, Rebecca L. Heise, Angela M. Reynolds, and Ramana M. Pidaparti. Quantification of Age-Related Lung Tissue Mechanics under Mechanical Ventilation.

Medical Sciences, 5(4):21, December 2017. doi: 10.3390/medsci5040021. URL https://www.mdpi.com/2076-3271/5/4/21. Number: 4 Publisher: Multidisciplinary Digital Publishing Institute.

[58] Denise Kirschner, Elsje Pienaar, Simeone Marino, and Jennifer J. Linderman. A review of computational and mathematical modeling contributions to our understanding of Mycobacterium tuberculosis within-host infection and treatment. Current Opinion in Systems Biology, 3:170–185, June 2017. ISSN 2452-3100. doi: 10.1016/j.coisb.2017.05.014. URL http://www.sciencedirect.com/science/article/pii/S2452310016300117.

[59] Denise E. Kirschner. Matlab Functions for PRCC and eFAST, 2008. URL http://malthus.micro.med.umich.edu/lab/usadata/.

[60] Takeshi Koga, Ippei Kuwahara, Erik P. Lillehoj, Wenju Lu, Takeshi Miyata, Yoichiro Isohama, and K. Chul Kim. TNF-α induces MUC1 gene transcription in lung epithelial cells: its signaling pathway and biological implication. American Journal of Physiology-Lung Cellular and Molecular Physiology, 293(3):L693–L701, September 2007. ISSN 1040-0605. doi: 10.1152/ajplung.00491.2006. URL http://journals.physiology.org/doi/full/10.1152/ajplung.00491.2006. Publisher: American Physiological Society.

[61] Elzbieta Kolaczkowska and Paul Kubes. Neutrophil Recruitment and Function in Health and Inflammation. Nature Reviews Immunology, 13(3):159–175, March 2013. ISSN 1474-1741. doi: 10.1038/nri3399. URL https://www.nature.com/articles/nri3399.

[62] J. Kretschmer, C. Schranz, C. Knöbel, J. Wingender, E. Koch, and K. Möller. Efficient computation of interacting model systems. Journal of Biomedical Informatics, 46(3):

401–409, June 2013. ISSN 1532-0464. doi: 10.1016/j.jbi.2013.01.004. URL http://www.sciencedirect.com/science/article/pii/S1532046413000178.

[63] Vijay Kumar and A. Sharma. Neutrophils: Cinderella of Innate Immune System. International Immunopharmacology, 10(11):1325–1334, November 2010. ISSN 1567-5769. doi: 10.1016/j.intimp.2010.08.012. URL http://www.sciencedirect.com/science/article/pii/S1567576910002663.

[64] Chih-Cheng Lai, Tzu-Ping Shih, Wen-Chien Ko, Hung-Jen Tang, and Po-Ren Hsueh. Severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2) and coronavirus disease-2019 (COVID-19): The epidemic and the challenges. International Journal of Antimicrobial Agents, 55(3):105924, March 2020. ISSN 0924-8579. doi: 10.1016/j.ijantimicag.2020.105924. URL https://www.sciencedirect.com/science/article/pii/S0924857920300674.

[65] James Le. Decision Trees in R, June 2018. URL https://www.datacamp.com/community/tutorials/decision-trees-R.

[66] Andy Liaw and Matthew Wiener. Classification and Regression by randomForest, 2002.

[67] Eimear Linehan and Denise Fitzgerald. Ageing and the Immune System: Focus on Macrophages. European Journal of Microbiology and Immunology, 5(1):14–24, March 2015. doi: 10.1556/EuJMI-D-14-00035. URL http://akademiai.com/doi/abs/10.1556/EuJMI-D-14-00035.

[68] Charles X. Ling and Victor S. Sheng. Class Imbalance Problem. In Claude Sammut and Geoffrey I. Webb, editors, Encyclopedia of Machine Learning and Data Mining, pages 204–205. Springer US, Boston, MA, 2017. ISBN 978-1-4899-7687-1. doi: 10.1007/978-1-4899-7687-1_110. URL https://doi.org/10.1007/978-1-4899-7687-1_110.

[69] Tomasz Lipniacki, Pawel Paszek, Allan R. Brasier, Bruce Luxon, and Marek Kimmel. Mathematical model of NF-$\varkappa$B regulatory module. Journal of Theoretical Biology, 228(2):195–215, May 2004. ISSN 0022-5193. doi: 10.1016/j.jtbi.2004.01.001. URL http://www.sciencedirect.com/science/article/pii/S0022519304000037.

[70] Yoram Louzoun, Chuan Xue, Gregory B. Lesinski, and Avner Friedman. A mathematical model for pancreatic cancer growth and treatments. Journal of Theoretical Biology, 351:74–82, June 2014. ISSN 0022-5193. doi: 10.1016/j.jtbi.2014.02.028. URL http://www.sciencedirect.com/science/article/pii/S0022519314001039.

[71] Elisabeth Mahase. Covid-19: most patients require mechanical ventilation in first 24 hours of critical care. BMJ, 368, March 2020. ISSN 1756-1833. doi: 10.1136/bmj.m1201. URL https://www.bmj.com/content/368/bmj.m1201. Publisher: British Medical Journal Publishing Group Section: News.

[72] Shegufta Mahbub, Cory R. Deburghgraeve, and Elizabeth J. Kovacs. Advanced Age Impairs Macrophage Polarization. Journal of Interferon & Cytokine Research, 32(1):18–26, December 2011. ISSN 1079-9907. doi: 10.1089/jir.2011.0058. URL http://online.liebertpub.com/doi/abs/10.1089/jir.2011.0058.

[73] Shreya Maiti, Wei Dai, Robert C. Alaniz, Juergen Hahn, and Arul Jayaraman. Mathematical Modeling of Pro- and Anti-Inflammatory Signaling in Macrophages. Processes, 3(1):1–18, December 2014. doi: 10.3390/pr3010001. URL http://www.mdpi.com/2227-9717/3/1/1.

[74] S. Manikandan. Measures of dispersion. Journal of Pharmacology and Pharmacotherapeutics, 2(4):315–316, October 2011. ISSN 0976500X. doi: http://dx.doi.org/10.4103/0976-500X.85931. URL https://search.proquest.com/docview/898546209/abstract/C1F56519EFA444CEPQ/1. Num Pages: 1 Place: Mumbai, Mumbai Publisher: Medknow Publications & Media Pvt. Ltd.

[75] John J. Marini, Philip S. Crooke, and J. D. Truwit. Determinants and Limits of Pressure-Preset Ventilation: a Mathematical Model of Pressure Control. Journal of Applied Physiology, 67(3):1081–1092, September 1989. ISSN 8750-7587, 1522-1601. URL http://jap.physiology.org.proxy.library.vcu.edu/content/67/3/1081.

[76] Simeone Marino, Ian B. Hogue, Christian J. Ray, and Denise E. Kirschner. A methodology for performing global uncertainty and sensitivity analysis in systems biology. Journal of Theoretical Biology, 254(1):178–196, September 2008. ISSN 0022-5193. doi: 10.1016/j.jtbi.2008.04.011. URL http://www.sciencedirect.com/science/article/pii/S0022519308001896.

[77] Simeone Marino, Nicholas A. Cilfone, Joshua T. Mattila, Jennifer J. Linderman, JoAnne L. Flynn, and Denise E. Kirschner. Macrophage Polarization Drives Granuloma Outcome during Mycobacterium tuberculosis Infection. Infection and Immunity, 83(1):324–338, January 2015. ISSN 0019-9567, 1098-5522. doi: 10.1128/IAI.02494-14. URL https://iai.asm.org/content/83/1/324.

[78] Malvina Marku, Nina Verstraete, Flavien Raynal, Miguel Madrid-Mencía, Marcin Domagala, Jean-Jacques Fournié, Loïc Ysebaert, Mary Poupot, and Vera Pancaldi. Insights on TAM Formation from a Boolean Model of Macrophage Polarization Based on In Vitro Studies. Cancers, 12(12):3664, December 2020. doi: 10.3390/cancers12123664. URL https://www.mdpi.com/2072-6694/12/12/3664. Number: 12 Publisher: Multidisciplinary Digital Publishing Institute.

[79] Robert J. Mason. Biology of alveolar type II cells. Respirology, 11(s1):S12–S15, 2006. ISSN 1440-1843. doi: 10.1111/j.1440-1843.2006.00800.x. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1440-1843.2006.00800.x.

[80] Shibin Mathew, John Bartels, Ipsita Banerjee, and Yoram Vodovotz. Global sensitivity analysis of a mathematical model of acute inflammation identifies nonlinear

dependence of cumulative tissue damage on host interleukin-6 responses. Journal of Theoretical Biology, 358:132–148, October 2014. ISSN 0022-5193. doi: 10.1016/j.jtbi.2014.05.036. URL https://www.sciencedirect.com/science/article/pii/S0022519314003191.

[81] Michael A. Matthay, Laurent Robriquet, and Xiaohui Fang. Alveolar Epithelium. Proceedings of the American Thoracic Society, 2(3):206–213, October 2005. ISSN 1546-3222. doi: 10.1513/pats.200501-009AC. URL https://www.atsjournals.org/doi/full/10.1513/pats.200501-009AC.

[82] Reginald L. McGee and Gregery T. Buzzard. Maximally informative next experiments for nonlinear models. Mathematical Biosciences, 302:1–8, August 2018. ISSN 0025-5564. doi: 10.1016/j.mbs.2018.04.007. URL http://www.sciencedirect.com/science/article/pii/S0025556417305801.

[83] Michael D. McKay, Richard J. Beckman, and William J. Conover. Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code. Technometrics, 21(2):239–245, May 1979. ISSN 0040-1706. doi: 10.1080/00401706.1979.10489755.

[84] Patrick E. McKight and Julius Najab. Kruskal-Wallis Test. In The Corsini Encyclopedia of Psychology, pages 1–1. American Cancer Society, 2010. ISBN 978-0-470-47921-6. doi: 10.1002/9780470479216.corpsy0491. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/9780470479216.corpsy0491.

[85] Sarah Minucci, Rebecca L. Heise, Michael S. Valentine, Franck J. Kamga Gninzeko, and Angela M. Reynolds. Mathematical modeling of ventilator-induced lung inflammation. Journal of Theoretical Biology, page 110738, April 2021. ISSN 0022-5193. doi: 10.1016/j.jtbi.2021.110738. URL https://www.sciencedirect.com/science/article/pii/S0022519321001600.

[86] Sarah B. Minucci, Rebecca L. Heise, and Angela M. Reynolds. Review of Mathematical Modeling of the Inflammatory Response in Lung Infections and Injuries. Frontiers in Applied Mathematics and Statistics, 6, 2020. ISSN 2297-4687. doi: 10.3389/fams.2020.00036. URL https://www.frontiersin.org/articles/10.3389/fams.2020.00036/full?&utm_source=Email_to_authors_&utm_medium=Email&utm_content=T1_11.5e1_author&utm_campaign=Email_publication&field=&journalName=Frontiers_in_Applied_Mathematics_and_Statistics&id=556294. Publisher: Frontiers.

[87] Hugh Mitchell, Drew Levin, Stephanie Forrest, Catherine A. A. Beauchemin, Jennifer Tipper, Jennifer Knight, Nathaniel Donart, R. Colby Layton, John Pyles, Peng Gao, Kevin S. Harrod, Alan S. Perelson, and Frederick Koster. Higher Level of Replication Efficiency of 2009 (H1N1) Pandemic Influenza Virus than Those of Seasonal and Avian Strains: Kinetics from Epithelial Cell Culture and Computational Modeling. Journal of Virology, 85(2):1125–1135, January 2011. ISSN 0022-538X. doi: 10.1128/JVI.01722-10. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3019989/.

[88] David M. Mosser and Justin P. Edwards. Exploring the Full Spectrum of Macrophage Activation. Nature Reviews Immunology, 8(12):958–969, December 2008. ISSN 1474-1733. doi: 10.1038/nri2448. URL http://www.nature.com.proxy.library.vcu.edu/nri/journal/v8/n12/full/nri2448.html.

[89] Janina Mothes, Inbal Ipenberg, Seda Çöl Arslan, Uwe Benary, Claus Scheidereit, and Jana Wolf. A Quantitative Modular Modeling Approach Reveals the Effects of Different A20 Feedback Implementations for the NF-kB Signaling Dynamics. Frontiers in Physiology, 11, 2020. ISSN 1664-042X. doi: 10.3389/fphys.2020.00896. URL https://www.frontiersin.org/articles/10.3389/fphys.2020.00896/full?report=reader. Publisher: Frontiers.

[90] C. Moya, Z. Huang, P. Cheng, A. Jayaraman, and J. Hahn. Investigation of IL-6

and IL-10 signalling via mathematical modelling. IET Systems Biology, 5(1):15–26, January 2011. ISSN 1751-8857. doi: 10.1049/iet-syb.2009.0060.

[91] Lisa N. Murillo, Michael S. Murillo, and Alan S. Perelson. Towards multiscale modeling of influenza infection. Journal of Theoretical Biology, 332:267–290, September 2013. ISSN 0022-5193. doi: 10.1016/j.jtbi.2013.03.024. URL http://www.sciencedirect.com/science/article/pii/S0022519313001501.

[92] Carl Nathan. Neutrophils and Immunity: Challenges and Opportunities. Nature Reviews Immunology, 6(3):173–182, March 2006. ISSN 1474-1741. doi: 10.1038/nri1785. URL https://www.nature.com/articles/nri1785.

[93] Emma J. Naylor, Denise Bakstad, Mark Biffen, Bob Thong, Peter Calverley, Stephen Scott, C. Anthony Hart, Robert J. Moots, and Steven W. Edwards. Haemophilus Influenzae Induces Neutrophil Necrosis. American Journal of Respiratory Cell and Molecular Biology, 37(2):135–143, August 2007. ISSN 1044-1549. doi: 10.1165/rcmb.2006-0375OC. URL https://www.atsjournals.org/doi/full/10.1165/rcmb.2006-0375OC.

[94] Niloofar Nickaeen, Jafar Ghaisari, Monika Heiner, Shiva Moein, and Yousof Gheisari. Agent-based modeling and bifurcation analysis reveal mechanisms of macrophage polarization and phenotype pattern distribution. Scientific Reports, 9(1):12764, September 2019. ISSN 2045-2322. doi: 10.1038/s41598-019-48865-z. URL https://www.nature.com/articles/s41598-019-48865-z. Number: 1 Publisher: Nature Publishing Group.

[95] Steven M. Opal and Vera A. DePalo. Anti-Inflammatory Cytokines. Chest, 117(4):1162–1172, April 2000. ISSN 0012-3692. doi: 10.1378/chest.117.4.1162. URL http://www.sciencedirect.com/science/article/pii/S0012369215328208.

[96] Fred C. Pampel. Logistic Regression: A Primer. SAGE Publications, September 2020. ISBN 978-1-07-181617-2. Google-Books-ID: TRD8DwAAQBAJ.

[97] Beom Seok Park and Jie-Oh Lee. Recognition of lipopolysaccharide pattern by TLR4 complexes. Experimental & Molecular Medicine, 45(12):e66–e66, December 2013. ISSN 2092-6413. doi: 10.1038/emm.2013.97. URL https://www.nature.com/articles/emm201397. Number: 12 Publisher: Nature Publishing Group.

[98] Stephan Peter, Peter Dittrich, and Bashar Ibrahim. Structure and Hierarchy of SARS-CoV-2 Infection Dynamics Models Revealed by Reaction Network Analysis. Viruses, 13(1):14, January 2021. doi: 10.3390/v13010014. URL https://www.mdpi.com/1999-4915/13/1/14. Number: 1 Publisher: Multidisciplinary Digital Publishing Institute.

[99] Tài Pham, Laurent J. Brochard, and Arthur S. Slutsky. Mechanical Ventilation: State of the Art. Mayo Clinic Proceedings, 92(9):1382–1400, September 2017. ISSN 0025-6196. doi: 10.1016/j.mayocp.2017.05.004. URL http://www.sciencedirect.com/science/article/pii/S0025619617303245.

[100] Ramana M. Pidaparti, Matthew Burnette, Rebecca L. Heise, and Angela Reynolds. Analysis for Stress Environment in the Alveolar Sac Model. Journal of biomedical science and engineering, 6(9):901–907, September 2013. ISSN 1937-6871. doi: 10.4236/jbise.2013.69110. URL http://www.ncbi.nlm.nih.gov/pmc/articles/PMC4057278/.

[101] Filippo Posta and Tom Chou. A mathematical model of intercellular signaling during epithelial wound healing. Journal of Theoretical Biology, 266(1):70–78, September 2010. ISSN 0022-5193. doi: 10.1016/j.jtbi.2010.05.029. URL http://www.sciencedirect.com/science/article/pii/S0022519310002729.

[102] Joshua J. Pothen, Matthew E. Poynter, and Jason H. T. Bates. A Computational

Model of Unresolved Allergic Inflammation in Chronic Asthma. American Journal of Physiology - Lung Cellular and Molecular Physiology, 308(4):L384–L390, February 2015. ISSN 1040-0605, 1522-1504. doi: 10.1152/ajplung.00268.2014. URL http://ajplung.physiology.org.proxy.library.vcu.edu/content/308/4/L384.

[103] Mauro Provinciali, Maurizio Cardelli, and Francesca Marchegiani. Inflammation, chronic obstructive pulmonary disease and aging. Current Opinion in Pulmonary Medicine, 17:S3, December 2011. ISSN 1070-5287. doi: 10.1097/01.mcp.0000410742.90463.1f. URL https://journals.lww.com/co-pulmonarymedicine/Abstract/2011/12001/Inflammation,_chronic_obstructive_pulmonary.2.aspx.

[104] Pooran Qasimi, Andrew Ming-Lum, Ali Ghanipour, Christopher J. Ong, Michael E. Cox, James Ihle, Nicolas Cacalano, Akihiko Yoshimura, and Alice L.-F. Mui. Divergent Mechanisms Utilized by SOCS3 to Mediate Interleukin-10 Inhibition of Tumor Necrosis Factor α and Nitric Oxide Production by Macrophages. Journal of Biological Chemistry, 281(10):6316–6324, March 2006. ISSN 0021-9258, 1083-351X. doi: 10.1074/jbc.M508608200. URL http://www.jbc.org/content/281/10/6316.

[105] C. Quesnel, L. Nardelli, P. Piednoir, V. Leçon, J. Marchal-Somme, S. Lasocki, L. Bouadma, I. Philip, P. Soler, B. Crestani, and M. Dehoux. Alveolar fibroblasts in acute lung injury: biological behaviour and clinical relevance. European Respiratory Journal, 35(6):1312–1321, June 2010. ISSN 0903-1936, 1399-3003. doi: 10.1183/09031936.00074709. URL http://erj.ersjournals.com/content/35/6/1312. Publisher: European Respiratory Society Section: Original Articles: Critical care and lung injury.

[106] Ubaidur Rahaman. Mathematics of Ventilator-induced Lung Injury. Indian Journal of Critical Care Medicine : Peer-reviewed, Official Publication of Indian Society of Critical Care Medicine, 21(8):521–524, August 2017. ISSN 0972-5229, 1998-359X.

doi: 10.4103/ijccm.ijccm_411_16. URL https://europepmc.org/article/pmc/pmc5588487.

[107] Jason S. Rawlings, Kristin M. Rosler, and Douglas A. Harrison. The JAK/STAT signaling pathway. Journal of Cell Science, 117(8):1281–1283, March 2004. ISSN 0021-9533, 1477-9137. doi: 10.1242/jcs.00963. URL https://jcs.biologists.org/content/117/8/1281.

[108] Nathalie de Rekeneire, Rita Peila, Jingzhong Ding, Lisa H. Colbert, Marjolein Visser, Ronald I. Shorr, Stephen B. Kritchevsky, Lewis H. Kuller, Elsa S. Strotmeyer, Ann V. Schwartz, Bruno Vellas, and Tamara B. Harris. Diabetes, Hyperglycemia, and Inflammation in Older Individuals: The Health, Aging and Body Composition study. Diabetes Care, 29(8):1902–1908, August 2006. ISSN 0149-5992, 1935-5548. doi: 10.2337/dc05-2327. URL https://care.diabetesjournals.org/content/29/8/1902.

[109] Ines Rentzsch, Cíntia L. Santos, Robert Huhle, Jorge M. C. Ferreira, Thea Koch, Christian Schnabel, Edmund Koch, Paolo Pelosi, Patricia R. M. Rocco, and Marcelo Gama de Abreu. Variable stretch reduces the pro-inflammatory response of alveolar epithelial cells. PLOS ONE, 12(8):e0182369, August 2017. ISSN 1932-6203. doi: 10.1371/journal.pone.0182369. URL http://journals.plos.org/plosone/article?id=10.1371/journal.pone.0182369. Publisher: Public Library of Science.

[110] Julia Rex, Ute Albrecht, Christian Ehlting, Maria Thomas, Ulrich M. Zanger, Oliver Sawodny, Dieter Häussinger, Michael Ederer, Ronny Feuer, and Johannes G. Bode. Model-Based Characterization of Inflammatory Gene Expression Patterns of Activated Macrophages. PLOS Computational Biology, 12(7):e1005018, July 2016. ISSN 1553-7358. doi: 10.1371/journal.pcbi.1005018. URL http://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1005018.

[111] Angela Reynolds, G. Bard Ermentrout, and Gilles Clermont. A Mathematical Model of

Pulmonary Gas Exchange Under Inflammatory Stress. Journal of Theoretical Biology, 264(2):161–173, May 2010. ISSN 0022-5193. doi: 10.1016/j.jtbi.2010.01.011. URL http://www.sciencedirect.com/science/article/pii/S0022519310000159.

[112] Joan K. Riley, Kiyoshi Takeda, Shizuo Akira, and Robert D. Schreiber. Interleukin-10 Receptor Signaling through the JAK-STAT Pathway: Requirement for Two Distinct Receptor-Derived Signals for Anti-Inflammatory Action. Journal of Biological Chemistry, 274(23):16513–16521, June 1999. ISSN 0021-9258, 1083-351X. doi: 10.1074/jbc.274.23.16513. URL http://www.jbc.org.proxy.library.vcu.edu/content/274/23/16513.

[113] C. T. Robb, K. H. Regan, D. A. Dorward, and A. G. Rossi. Key Mechanisms Governing Resolution of Lung Inflammation. Seminars in Immunopathology, 38(4):425–448, July 2016. ISSN 1863-2297, 1863-2300. doi: 10.1007/s00281-016-0560-6. URL https://link-springer-com.proxy.library.vcu.edu/article/10.1007/s00281-016-0560-6.

[114] Rafael Rosolem, Hoshin V. Gupta, W. James Shuttleworth, Xubin Zeng, and Luis Gustavo Gonçalves de Gonçalves. A fully multiple-criteria implementation of the Sobol method for parameter sensitivity analysis. Journal of Geophysical Research: Atmospheres, 117(D7), 2012. ISSN 2156-2202. doi: https://doi.org/10.1029/2011JD016355. URL https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2011JD016355. _eprint: https://agupubs.onlinelibrary.wiley.com/doi/pdf/10.1029/2011JD016355.

[115] Robert Sabat, Gerald Grütz, Katarzyna Warszawska, Stefan Kirsch, Ellen Witte, Kerstin Wolk, and Jens Geginat. Biology of interleukin-10. Cytokine & Growth Factor Reviews, 21(5):331–344, October 2010. ISSN 1359-6101. doi: 10.1016/j.cytogfr.2010.09.002. URL https://www.sciencedirect.com/science/article/pii/S1359610110000651.

[116] Mamatha Serasanambati and Shanmuga Reddy Chilakapati. Function of Nuclear Factor Kappa B (NF-kB) in human diseases- A Review. South Indian Journal of Biological Sciences, 2:368, October 2016. doi: 10.22205/sijbs/2016/v2/i4/103443.

[117] Stephanie M. Sison, Gayathri K. Sivakumar, Christine Caufield-Noll, William B. Greenough, Esther S. Oh, and Panagis Galiatsatos. Mortality outcomes of patients on chronic mechanical ventilation in different care settings: A systematic review. Heliyon, 7(2):e06230, February 2021. ISSN 2405-8440. doi: 10.1016/j.heliyon.2021.e06230. URL https://www.sciencedirect.com/science/article/pii/S2405844021003352.

[118] Arthur S. Slutsky and V. Marco Ranieri. Ventilator-Induced Lung Injury. New England Journal of Medicine, 369(22):2126–2136, November 2013. ISSN 0028-4793. doi: 10.1056/NEJMra1208707. URL http://dx.doi.org/10.1056/NEJMra1208707.

[119] Rod Smallwood. Computational modeling of epithelial tissues. WIREs Systems Biology and Medicine, 1(2):191–201, 2009. ISSN 1939-005X. doi: https://doi.org/10.1002/wsbm.18. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/wsbm.18.

[120] Amber M. Smith. Host-pathogen kinetics during influenza infection and coinfection: insights from predictive modeling. Immunological Reviews, 285(1):97–112, September 2018. ISSN 1600-065X. doi: 10.1111/imr.12692. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/imr.12692.

[121] Lincoln S. Smith, Sina A. Gharib, Charles W. Frevert, and Thomas R. Martin. Effects of Age on the Synergistic Interactions between Lipopolysaccharide and Mechanical Ventilation in Mice. American Journal of Respiratory Cell and Molecular Biology, 43(4):475–486, October 2010. ISSN 1044-1549. doi: 10.1165/rcmb.2009-0039OC. URL https://www.atsjournals.org/doi/full/10.1165/rcmb.2009-0039OC. Publisher: American Thoracic Society - AJRCMB.

[122] Tim D. Smith, Margaret J. Tse, Elizabeth L. Read, and Wendy F. Liu. Regulation

of macrophage polarization and plasticity by complex activation signals. Integrative Biology, 8(9):946–955, September 2016. ISSN 1757-9708. doi: 10.1039/c6ib00105j. URL https://doi.org/10.1039/c6ib00105j.

[123] Oliver Soehnlein and Lennart Lindbom. Phagocyte partnership during the onset and resolution of inflammation. Nature Reviews Immunology, 10(6):427–439, June 2010. ISSN 1474-1741. doi: 10.1038/nri2779. URL https://www.nature.com/articles/nri2779.

[124] Peter S. Stewart and Oliver E. Jensen. Patterns of recruitment and injury in a heterogeneous airway network model. Journal of The Royal Society Interface, 12(111):20150523, October 2015. doi: 10.1098/rsif.2015.0523. URL https://royalsocietypublishing.org/doi/full/10.1098/rsif.2015.0523. Publisher: Royal Society.

[125] Charlotte Summers, Sara M. Rankin, Alison M. Condliffe, Nanak Singh, A. Michael Peters, and Edwin R. Chilvers. Neutrophil Kinetics in Health and Disease. Trends in Immunology, 31(8):318–324, August 2010. ISSN 1471-4906. doi: 10.1016/j.it.2010.05.006. URL http://www.sciencedirect.com/science/article/pii/S147149061000075X.

[126] Taiga Tamiya, Ikko Kashiwagi, Reiko Takahashi, Hideo Yasukawa, and Akihiko Yoshimura. Suppressors of Cytokine Signaling (SOCS) Proteins and JAK/STAT Pathways. Arteriosclerosis, Thrombosis, and Vascular Biology, 31(5):980–985, May 2011. doi: 10.1161/ATVBAHA.110.207464. URL https://www.ahajournals.org/doi/full/10.1161/ATVBAHA.110.207464.

[127] Reiko J. Tanaka, Masahiro Ono, and Heather A. Harrington. Skin Barrier Homeostasis in Atopic Dermatitis: Feedback Regulation of Kallikrein Activity. PLOS ONE, 6(5):e19895, May 2011. ISSN 1932-6203. doi: 10.1371/journal.pone.0019895. URL http://

journals.plos.org/plosone/article?id=10.1371/journal.pone.0019895. Publisher: Public Library of Science.

[128] Marcella Torres, Jing Wang, Paul J. Yannie, Shobha Ghosh, Rebecca A. Segal, and Angela M. Reynolds. Identifying important parameters in the inflammatory process with a mathematical model of immune cell influx and macrophage polarization. PLOS Computational Biology, 15(7):e1007172, July 2019. ISSN 1553-7358. doi: 10.1371/journal.pcbi.1007172. URL https://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1007172.

[129] Lorraine N. Tremblay, Debra Miatto, Qutayba Hamid, Anand Govindarajan, and Arthur S. Slutsky. Injurious ventilation induces widespread pulmonary epithelial expression of tumor necrosis factor-α and interleukin-6 messenger RNA. Critical Care Medicine, 30(8):1693–1700, August 2002. ISSN 0090-3493. URL http://journals.lww.com/ccmjournal/Fulltext/2002/08000/Injurious_ventilation_induces_widespread_pulmonary.3.aspx.

[130] Alan Valaperti, Pascal Bezel, Maya Vonow-Eisenring, Daniel Franzen, and Urs C. Steiner. Variability of cytokine concentration in whole blood serum and bronchoalveolar lavage over time. Cytokine, 123:154768, November 2019. ISSN 1043-4666. doi: 10.1016/j.cyto.2019.154768. URL https://www.sciencedirect.com/science/article/pii/S1043466619301978.

[131] M. S. Valentine, P. A. Link, J. A. Herbert, F. J. Kamga Gninzeko, M. B. Schneck, K. Shankar, J. Nkwocha, A. M. Reynolds, and R. L. Heise. Inflammation and Monocyte Recruitment Due to Aging and Mechanical Stretch in Alveolar Epithelium are Inhibited by the Molecular Chaperone 4-Phenylbutyrate. Cellular and Molecular Bioengineering, 11(6):495–508, December 2018. ISSN 1865-5033. doi: 10.1007/s12195-018-0537-8. URL https://doi.org/10.1007/s12195-018-0537-8.

[132] P. Van Liedekerke, M. M. Palm, N. Jagiella, and D. Drasdo. Simulating tissue mechanics with agent-based models: concepts, perspectives and some novel results. Computational Particle Mechanics, 2:401–444, December 2015. doi: 10.1007/s40571-015-0082-3. URL http://adsabs.harvard.edu/abs/2015CPM.....2..401V.

[133] Renu Verma, Lavanya Balakrishnan, Kusum Sharma, Aafaque Ahmad Khan, Jayshree Advani, Harsha Gowda, Srikanth Prasad Tripathy, Mrutyunjay Suar, Akhilesh Pandey, Sheetal Gandotra, T. S. Keshava Prasad, and Subramanian Shankar. A network map of Interleukin-10 signaling pathway. Journal of Cell Communication and Signaling, 10 (1):61–67, March 2016. ISSN 1873-9601, 1873-961X. doi: 10.1007/s12079-015-0302-x. URL http://link.springer.com/10.1007/s12079-015-0302-x.

[134] Nicholas E. Vlahakis, Mark A. Schroeder, Andrew H. Limper, and Rolf D. Hubmayr. Stretch Induces Cytokine Release by Alveolar Epithelial Cells in Vitro. American Journal of Physiology-Lung Cellular and Molecular Physiology, 277(1):L167–L173, July 1999. ISSN 1040-0605. doi: 10.1152/ajplung.1999.277.1.L167. URL http://www.physiology.org/doi/abs/10.1152/ajplung.1999.277.1.L167.

[135] Nan Wang, Hongwei Liang, and Ke Zen. Molecular Mechanisms That Influence the Macrophage M1–M2 Polarization Balance. Frontiers in Immunology, 5, November 2014. ISSN 1664-3224. doi: 10.3389/fimmu.2014.00614. URL http://www.ncbi.nlm.nih.gov/pmc/articles/PMC4246889/.

[136] Y. Wang, Y. P. Wang, G. Zheng, V. W. S. Lee, L. Ouyang, D. H. H. Chang, D. Mahajan, J. Coombs, Y. M. Wang, S. I. Alexander, and D. C. H. Harris. Ex vivo programmed macrophages ameliorate experimental chronic inflammatory renal disease. Kidney International, 72(3):290–299, August 2007. ISSN 0085-2538. doi: 10.1038/sj.ki.5002275. URL https://www.sciencedirect.com/science/article/pii/S008525381552632X.

[137] Yunji Wang, Tianyi Yang, Yonggang Ma, Ganesh V. Halade, Jianqiu Zhang, Merry L. Lindsey, and Yu-Fang Jin. Mathematical modeling and stability analysis of macrophage activation in left ventricular remodeling post-myocardial infarction. BMC Genomics, 13(6):S21, October 2012. ISSN 1471-2164. doi: 10.1186/1471-2164-13-S6-S21. URL https://doi.org/10.1186/1471-2164-13-S6-S21.

[138] Lorraine B. Ware and Michael A. Matthay. The Acute Respiratory Distress Syndrome. New England Journal of Medicine, 342(18):1334–1349, May 2000. ISSN 0028-4793. doi: 10.1056/NEJM200005043421806. URL https://doi.org/10.1056/NEJM200005043421806.

[139] George W. Williams, Nathaniel K. Berg, Alexander Reskallah, Xiaoyi Yuan, and Holger K. Eltzschig. Acute Respiratory Distress Syndrome: Contemporary Management and Novel Approaches during COVID-19. Anesthesiology, 134(2):270–282, February 2021. ISSN 0003-3022. doi: 10.1097/ALN.0000000000003571. URL https://doi.org/10.1097/ALN.0000000000003571.

[140] Michael R. Wilson, Kieran P. O'Dea, Da Zhang, Alexander D. Shearman, Nico van Rooijen, and Masao Takata. Role of Lung-marginated Monocytes in an In Vivo Mouse Model of Ventilator-induced Lung Injury. American Journal of Respiratory and Critical Care Medicine, 179(10):914–922, May 2009. ISSN 1073-449X. doi: 10.1164/rccm.200806-877OC. URL http://www.atsjournals.org/doi/full/10.1164/rccm.200806-877OC. Publisher: American Thoracic Society - AJRCCM.

[141] Esther K. Wolthuis, Alexander PJ Vlaar, Goda Choi, Joris JTH Roelofs, Nicole P. Juffermans, and Marcus J. Schultz. Mechanical ventilation using non-injurious ventilation settings causes lung injury in the absence of pre-existing lung injury in healthy mice. Critical Care, 13(1):R1, January 2009. ISSN 1364-8535. doi: 10.1186/cc7688. URL https://doi.org/10.1186/cc7688.

[142] Tien-Tsin Wong, Wai-Shing Luk, and Pheng-Ann Heng. Sampling with Hammersley and Halton Points. Journal of Graphics Tools, 2(2):9–24, January 1997. ISSN 1086-7651. doi: 10.1080/10867651.1997.10487471. URL https://doi.org/10.1080/10867651.1997.10487471. Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/10867651.1997.10487471.

[143] Peng Wu and Hongyong Zhao. Dynamics of an HIV Infection Model with Two Infection Routes and Evolutionary Competition between Two Viral Strains. Applied Mathematical Modelling, 84:240–264, August 2020. ISSN 0307-904X. doi: 10.1016/j.apm.2020.03.040. URL https://www.sciencedirect.com/science/article/pii/S0307904X20301748.

[144] Zunyou Wu and Jennifer M. McGoogan. Characteristics of and Important Lessons From the Coronavirus Disease 2019 (COVID-19) Outbreak in China: Summary of a Report of 72 314 Cases From the Chinese Center for Disease Control and Prevention. JAMA, 323(13):1239–1242, April 2020. ISSN 0098-7484. doi: 10.1001/jama.2020.2648. URL https://jamanetwork.com/journals/jama/fullarticle/2762130. Publisher: American Medical Association.

[145] Hannah Wunsch. Mechanical Ventilation in COVID-19: Interpreting the Current Epidemiology. American Journal of Respiratory and Critical Care Medicine, 202(1):1–4, July 2020. ISSN 1073-449X, 1535-4970. doi: 10.1164/rccm.202004-1385ED. URL https://www.atsjournals.org/doi/10.1164/rccm.202004-1385ED.

[146] Zohar Yagil, Hovav Nechushtan, Gillian Kay, Christopher M. Yang, David M. Kemeny, and Ehud Razin. The enigma of the role of Protein inhibitor of Activated STAT3 (PIAS3) in the immune response. Trends in Immunology, 31(5):199–204, May 2010. ISSN 1471-4906. doi: 10.1016/j.it.2010.01.005. URL http://www.sciencedirect.com/science/article/pii/S1471490610000062.

[147] Hideo Yasukawa, Hiroyuki Misawa, Hiroshi Sakamoto, Masaaki Masuhara, Atsuo Sasaki, Toru Wakioka, Satoshi Ohtsuka, Tsutomu Imaizumi, Tadashi Matsuda, James N. Ihle, and Akihiko Yoshimura. The JAK-binding protein JAB inhibits Janus tyrosine kinase activity through binding in the activation loop. The EMBO Journal, 18(5):1309–1320, March 1999. ISSN 1460-2075. doi: 10.1093/emboj/18.5.1309. URL https://www.embopress.org/doi/abs/10.1093/emboj/18.5.1309.

[148] Douglas Young, Jaroslav Stark, and Denise Kirschner. Systems biology of persistent infection: tuberculosis as a case study. Nature Reviews Microbiology, 6(7):520–528, July 2008. ISSN 1740-1534. doi: 10.1038/nrmicro1919. URL https://www.nature.com/articles/nrmicro1919. Number: 7 Publisher: Nature Publishing Group.

[149] Chen Zhao, Adam C. Mirando, Richard J. Sové, Thalyta X. Medeiros, Brian H. Annex, and Aleksander S. Popel. A mechanistic integrative computational model of macrophage polarization: Implications in human pathophysiology. PLOS Computational Biology, 15(11):e1007468, November 2019. ISSN 1553-7358. doi: 10.1371/journal.pcbi.1007468. URL https://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1007468. Publisher: Public Library of Science.

[150] Dexi Zhou, Cheng Huang, Zhen Lin, Shuxiang Zhan, Lingna Kong, Chengbo Fang, and Jun Li. Macrophage polarization and function with emphasis on the evolving roles of coordinated regulation of cellular signaling pathways. Cellular Signalling, 26 (2):192–197, February 2014. ISSN 0898-6568. doi: 10.1016/j.cellsig.2013.11.004. URL https://www.sciencedirect.com/science/article/pii/S0898656813003252.

Vita

Sarah Bartlett Minucci, originally from Franklin, Tennessee, attended Lee University in Cleveland, Tennessee and graduated summa cum laude with a Bachelor of Science in Mathematics, Bachelor of Arts in Spanish and minor in Biblical Studies. She discovered her passion for mathematical biology at an REU at Marquette University in Milwaukee, Wisconsin. After graduating from Lee, she moved to Richmond, Virginia to pursue a Master of Science in Applied Mathematics and a Ph.D. in Systems Modeling & Analysis at Virginia Commonwealth University. She was also selected for a number of extracurricular opportunities, including internships at VoluMetrix, Inc. in Nashville, Tennessee and UnitedHealth Group R&D in Minnetonka, Minnesota, a SIAM/AMS Tondeur Fellow (2019-2021) and president of the VCU chapter of SIAM (2020-2021). After graduating, Sarah will continue using mathematical modeling to understand complex biological processes through her job as a Senior Scientist at Applied BioMath in Concord, Massachusetts.