Available online at www.sciencedirect.com
ScienceDirect
Comput. Methods Appl. Mech. Engrg. 401 (2022) 115315
Computer methods in applied mechanics and engineering
www.elsevier.com/locate/cma

Machine-learning and digital-twins for rapid evaluation and design of injected vaccine immune-system responses

T.I. Zohdi

Department of Mechanical Engineering, 6117 Etcheverry Hall, University of California, Berkeley, CA, 94720-1740, USA

Available online 13 July 2022

Abstract

A computational framework is developed that researchers in the field can easily implement and subsequently use as an efficient tool to study the immune-response to a vaccine injection. There are three main components to this work: **Part I-Digital-twin construction:** An approach is developed that efficiently simulates the time-transient proliferation of cells/antibodies (proteins) and regulator/antigens (deactivated toxin) to an injected vaccine within tissue possessing complex heterogeneous microstructure. Here, we use the terms "cells" and "antibodies", as well as "regulator" and "antigen" interchangeably. The approach utilizes two strongly-coupled conservation laws: (a) **Conservation Law 1:** comprises (a) rate of change of cells/antibodies, (b) cellular/antibody migration, (c) cellular/antibody proliferation controlled by a cell/antibody mitosis regulating chemical (antigen), (d) cell/antibody apoptosis and (b) **Conservation Law 2:** comprises (a) rate of change of the cell/antibody mitosis chemical regulator/antigen, (b) regulator/antigen diffusion, (c) regulator production by cells/antibody and (d) regulator/antigen decay. **Part II-Efficient computation:** A technique based on a voxel (3D "volume pixels") representation of tissue microstructures and corresponding digital solution methods is developed for the calculations, which avoids computationally expensive steps involved in usual Finite Element procedures such as topologically conforming meshing, mapping, volume integration, stiffness matrix generation and matrix-based solution methods. The process proceeds by converting the tissue microstructure into voxels. The problem then becomes "digital" on a regular "voxel-grid", directly manipulating voxel values, allowing extremely fast methods to be used to construct derivatives and to iteratively solve the system with minimal memory requirements. **Part III-Machine-learning:** The rapid and efficient computation allows for many vaccines to be tested quickly and uses a genomic-based Machine-Learning Algorithm to optimize the system. *This is particularly useful for rapid design of next-generation vaccines and boosters for disease strain mutations.* Numerical examples are provided to illustrate the results, with the overall goal being to provide a computational framework to rapidly design and deploy a vaccine for a targeted response.

Keywords: Vaccine design; Digital-twins; Machine-learning

1. Introduction

A vaccine is a treatment that supplies acquired immunity to a particular disease. Vaccines greatly reduce the risk of infection by working with the body's natural defense to safely develop immunity to the disease. They typically contain a substance that resembles the actual disease micro-organism, that is not harmful, for example, weakened

E-mail address: zohdi@berkeley.edu.

https://doi.org/10.1016/j.cma.2022.115315
0045-7825/

forms of the infectious disease, such as disabled forms of the microbes and weakened forms of the associated toxin or surface proteins. The substance ("vaccine") stimulates and trains the immune system to attack and destroy the infectious disease. The World Health Organization (WHO) indicates that there are 25 preventable infectious diseases that have licensed vaccines, such as polio, measles, tetanus, chickenpox, etc.

1.1. Brief history

In 1798, Edward Jenner used cowpox ("smallpox of cows", variolae vaccinae) to make humans immune to smallpox by opening a small wound and placing cowpox (into the wound), making the person slightly ill, but eventually immune to smallpox. In the mid-1700s, John Williams also used weaken smallpox (weakened by subjecting samples to smoke, camphor and underground burial) to similarly inoculate people. It is notable to remark that these types of approaches had been used in the eastern population centers (such as Constantinople) for centuries before. In 1881, Pasteur suggested that these types of processes be called "vaccinations", in honor of Jenner. As an example of vaccine efficacy, consider the measles in the United States. In 1958, before the measles vaccine, there were 768,094 cases of the measles (552 deaths). The next year (1959) after mass inoculation, there were 150 cases. It is estimated that 1,000,000 lives are saved from measles per year worldwide by vaccination [1].

1.2. Types of vaccines

There are a variety of vaccine types:

- **Attenuated-type:** live attenuated (cultivated) micro-organisms with disabled virulence,
- **Inactivated-type:** previously virulent micro-organisms that have been previously irradiated or chemically destroyed — they are an empty bacterial cell envelope,
- **Toxoid-type:** inactivated toxins that come from the microorganisms,
- **Subunit-type:** fragments of micro-organisms, for example only the surface proteins,
- **Conjugate-type:** combines a weak antigen with a strong antigen as a carrier, so that the immune system has a stronger response to the weak antigen,
- **Outer membrane vesicle-type:** OMV's are naturally immunogenic and can be manipulated to produce vaccines,
- **Heterotypic-type**: "Jennerian Vaccines" use pathogens of other animals that do not cause the full disease in humans, but give immunity,
- **Viral vector-type:** use a safe virus to insert pathogen genes into the body to produce antigens to stimulate an immune response and
- **mRNA-type:** composed of the nucleic acid pf RNA packaged within a vector such as lipid nanoparticles-the synthetic RNA stimulates the immune system.

Vaccine licensing follows multiple phases in order to demonstrate safety as a given dose, effectiveness in preventing infections for target populations and an enduring preventive effects. Prominent regulatory bodies include the WHO, FDA (Food and Drug Association) and EMA (European Medicine Association). For Covid-19, vaccines include:

- Pfizer-Biotech: mRNA-type,
- Moderna: mRNA-type,
- AstraZeneca: heterotypic-type genetically-modified chimpanzee adenovirus (such as colds),
- Sputnik: vector-type genetically-modified common cold viruses,
- Sinopharm: inactivated-type Covid-19 viruses and
- Johnson and Johnson: vector-type genetically modified human adenovirus.

1.3. Vaccine efficacy

The key to any vaccine efficacy (such as the above) is that the immune system, when properly trained on a weakened version of the infectious disease (or surrogate), "remembers" its strategy and is able to fight off the real version by recognizing that protein coat on the virus and preparing a response by (1) neutralizing the target "agent" before it enters the cells and (2) recognizing and destroying infected cells before the disease can multiply in vast numbers. Two key terms are:

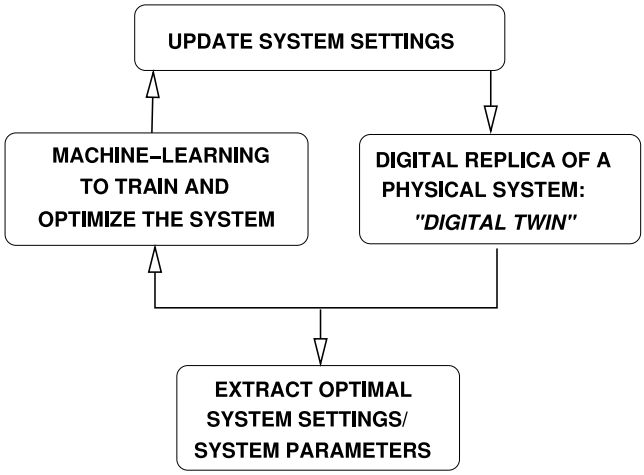

Fig. 1. A combined Machine-Learning and Digital-Twin model workflow.

- *Antigen:* a toxin or foreign substance that induces an immune response in the body-for example the production of "antibodies".
- *Antibody:* a blood protein produced in response to and countering an "antigen". Antibodies chemically combine with entities that the body identifies as alien-bacteria viruses and other foreign bodies in the blood, etc.

The following are common efficacy problems:

- vaccine attenuation over time (thus warranting "boosters"),
- host immune deficiency due to age, ethnicity and individual genetic variations, etc.,
- lack of "B-cells", which generate antibodies to react and bind with pathogen antigens,
- slow development of immunity and
- antibodies which cannot completely disable the pathogen (although the vaccine still has a positive effect since it reduces the mortality rate).

Further issues controlling the effectiveness are (1) the type of disease, (2) strain of the disease and the (3) vaccination schedule. There are of course potential adverse effects from a vaccine such as fever, pain and muscle aches.

1.4. Objectives of this work

Advances in data science and machine-learning are transforming society and engineering. An objective of this work is to harness these advances for vaccine design. Specifically, there have been dramatic advances in technologies associated with autonomous operations across many industries. These have the potential to drastically improve industrial efficiency, quality and safety. Some approaches are methodical and systematic, while others are ad-hoc and haphazard. In the world of systems engineering, increasingly sophisticated and integrated approaches for digital systems are appearing at a rapid rate. Key tools include digital-twins, which are digital replicas of complex systems which can be safely manipulated and optimized in a virtual world and then deployed in the physical world afterwards, reducing costs of experiments and accelerating development of new technologies. This process involves the application of machine-learning to digital-twins, whereby they learn from their mistakes/errors and constantly evolve to improve in a virtual environment. Digital-twins can also run in tandem with real systems and thereby serve as controllers. These concepts, which leverage the data-centric world, and the corresponding modeling paradigms, are increasingly used in fields outside of traditional Computational Mechanics (Fig. 1).

Accordingly, in this work, a computational approach is developed that researchers in the field can easily implement and subsequently use as an efficient tool to study the immune-response to a vaccine injection. There are three main components to this work:

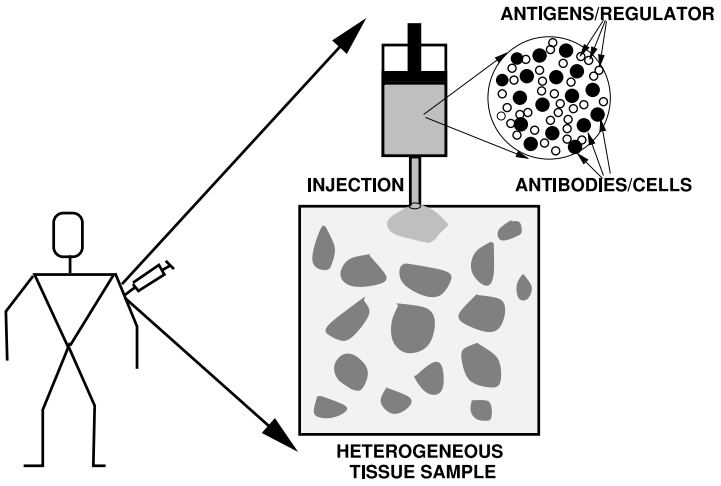

Fig. 2. Model Problem: an injection into a representative volume element of heterogeneous "marbled" tissue. The injection site is given high concentration of antibodies/cells and antigens/regulator.

1. **Digital-twin construction:** The work develops a computational framework that efficiently simulates the time-transient proliferation of cells/antibodies (proteins) and regulator/antigens (deactivated toxin) to an injected vaccine within tissue possessing complex heterogeneous microstructure. We use the terms "cells" and "antibodies" interchangeably and "regulator" and "antigen" interchangeably. The approach utilizes two strongly-coupled conservation laws: (1) one for the antibodies and (2) one for the antigens.
2. **Efficient computation:** A technique based on voxel (3D "volume pixels') representation of tissue microstructures and corresponding digital solution methods is developed for the calculations, which avoids computationally expensive steps involved in usual Finite Element procedures such as topologically conforming meshing, mapping, volume integration, stiffness matrix generation and matrix-based solution methods. The process proceeds by converting the material tissue microstructure into voxels. The problem then becomes "digital" on a regular "voxel-grid", directly manipulating voxel values, allowing extremely fast methods to be used to construct derivatives and to iteratively solve the system with minimal memory requirements.
3. **Machine-Learning:** The rapid and efficient computation allows for many vaccines to be tested quickly and uses a genomic-based Machine-Learning Algorithm to optimize the system. *This is particularly useful for rapid design of next-generation vaccines and boosters for disease strain mutations.*

Numerical examples are provided to illustrate the results, with the overall goal being to provide a computational framework to rapidly design and deploy a vaccine for a targeted response.

2. A flexible immune-response digital-twin

We start by developing a flexible model for the time-transient proliferation of cells/antibodies (proteins) and regulator/antigens (deactivated toxin) to an injected vaccine within tissue possessing heterogeneous "marbled" microstructure (Fig. 2). We develop two strongly-coupled conservation laws:

- **Conservation Law 1:** comprises (a) rate of change of cells/antibodies, (b) cellular/antibody migration, (c) cellular/antibody proliferation controlled by a cell/antibody mitosis regulating chemical (antigen), (d) cell/antibody apoptosis and
- **Conservation Law 2:** comprises (a) rate of change of the cell/antibody mitosis chemical regulator/antigen, (b) regulator/antigen diffusion, (c) regulator production by cells/antibody and (d) regulator/antigen decay.

Throughout the construction of the model, we consider infinitesimal deformations, $\dot{()} = \frac{\partial ()}{\partial t}|$. *In other words, the domain does not change its shape or geometry with changes in concentration.* The "cell/antibody" balance (c) per unit volume and a cell/antigen' mitosis regulating chemical (s) denoted by the normalized concentration of c

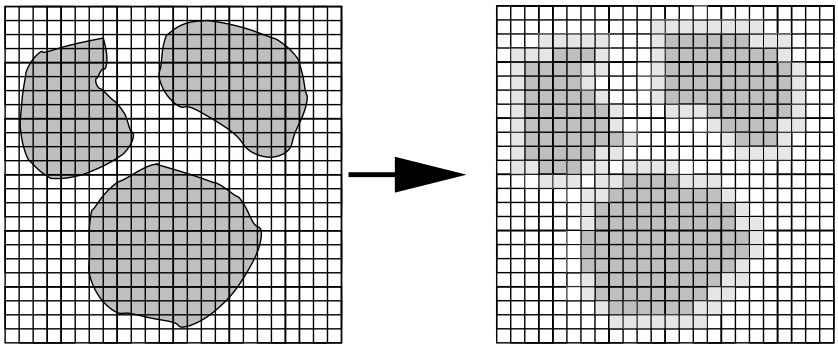

Fig. 3. Left: a material microstructure and Right: a voxel representation of the microstructure.

(cells/antibodies), in an arbitrary subvolume of material contained within Ω, denoted ω, consists of a concentration (storage) term c, an inward normal migration flux term, $-\boldsymbol{m}\cdot\boldsymbol{n}$, a proliferation term, $r(s)$, and a cell/antibody apoptosis term, $\tau(c) < 0$, leading to

$$\underbrace{\frac{\partial}{\partial t}\int_{\omega} c\, d\omega}_{\text{antibody storage}} = \underbrace{-\int_{\partial\omega} \boldsymbol{m}(c)\cdot\boldsymbol{n}\, da}_{\text{antibody migration}} + \underbrace{\int_{\omega} r(s)\, d\omega}_{\text{antibody proliferation}} + \underbrace{\int_{\omega} \tau(c)\, d\omega}_{\text{antibody apoptosis}} . \tag{2.1}$$

and simultaneously the balance of a cell/antibody mitosis regulating chemical/antigen (s)

$$\underbrace{\frac{\partial}{\partial t}\int_{\omega} s\, d\omega}_{\text{antigen storage}} = \underbrace{-\int_{\partial\omega} \boldsymbol{f}(s)\cdot\boldsymbol{n}\, da}_{\text{antigen diffusion}} + \underbrace{\int_{\omega} p(c)\, d\omega}_{\text{antigen production}} + \underbrace{\int_{\omega} \gamma(s)\, d\omega}_{\text{antigen loss}}, \tag{2.2}$$

where s is the cell/antibody mitosis regulator/antigen concentration, $-\boldsymbol{f}\cdot\boldsymbol{n}$ is an inward normal migration flux term, $p(c)$ is a production term and $\gamma(s) < 0$ is a regulator loss term. After using the divergence theorem on the flux terms, since the volume ω is arbitrary, one obtains a diffusion–reaction model in strong form (assuming a Fickian-type law, $\boldsymbol{m} = -\mathbb{D}\cdot\nabla c$ and $\boldsymbol{f} = -\mathbb{K}\cdot\nabla s$)

$$\frac{\partial c}{\partial t} = \nabla\cdot\mathbb{D}\cdot\nabla c + r(s) + \tau(c) \tag{2.3}$$

and simultaneously the balance of a mitosis regulating chemical (s)

$$\frac{\partial s}{\partial t} = \nabla\cdot\mathbb{K}\cdot\nabla s + p(c) + \gamma(s). \tag{2.4}$$

There is extensive literature on the construction of the functions $r(s)$, $\tau(c)$, $p(c)$ and $\gamma(s)$ for specific types of problems, such as wound healing and infection response. See Murray [2] for an extensive review, with early experimental studies dating back at least to Lindquist [3], Van den Brenk [4], Crosson et al. [5], Zieske et al. [6], Franz et al. [7] and Sherratt and Murray [8]. Such a coupled system can represent a variety of biological systems, such as growth in biological scaffolding, proliferation of damaged cellular tissue, etc. The modeling of this process has a close similarity to multicomponent diffusion–reaction industrial processes, and we refer the reader to Zohdi [9–12].

3. Rapid voxel based computation

In many methods that analyze micro-heterogeneous materials, the computation of the response of multiple representative volume elements (RVE) of heterogeneous materials are required. The RVE domain is usually taken to be cubical, but contains complex microstructure, for example randomly distributed particulates representing functionalizing dopants in a binding matrix material. Two frequent applications where this occurs are:

- Material design optimization, where the microstructures are constantly being changed during the search process. Usually, the Finite Element Method (FEM) is the default method used for such an analysis.

- Multiscale methods, where the FEM is used on the macroscale and then it is also used on microscale RVE-like problems which are solved at select locations throughout the domain. This is sometimes referred to as FE^2.

The fundamental problem is that the FEM is computationally expensive for microscale problems with a cubical RVE domain, due to the "ingredients" involved in the usual Finite Element process: meshing, mapping, volume integration, stiffness matrix generation and matrix-based solution methods. This computational expense is even more unwarranted when there is an interest in time-dependent regimes. An alternative family of techniques that is naturally suited for these types of problems are so-called "Digital/Voxel-Image" methods, which convert the material microstructure into voxels (3D "volume pixels"; see Fig. 3 and Fig. 4). The problem then becomes "digital" on a regular "voxel-grid", directly manipulating voxel values. Extremely fast methods can then be used to construct derivatives and solve the system.

The use of volume pixels, so-called "voxels" (Foley et al. [13]), is widespread in the visualization and analysis of medical and scientific data (Chmielewski et al. [14]) and in the video-gaming industry. The well-known video-game "Outcast", and others in the 1990s employed this graphics technique for effects such as reflection and bump-mapping and usually for terrain rendering, although other techniques have overtaken it as the method of choice. However, the most widely used application of a voxel is to represent material properties. For example, in CT scans, so-called Hounsfield units are used which measure the opacity of material to X-rays Novelline [15]. There are approximately 30 different types of values acquired from MRI or ultrasound. Thus, in many cases, the voxels are already supplied, and it makes little sense to employ the usual Finite Element machinery: topologically conforming meshing, mapping, volume integration, stiffness matrix generation, etc. In this work, we illustrate the process of voxelization, derivative construction and solution methods. Additionally, we also supply an analysis of the operation counts. Numerical examples are provided to illustrate the approach.

4. Numerical simulation of the coupled system

The present section develops a flexible and robust solution strategy to resolve the coupled system. There are two main components to the computational approach:

- Spatio-temporal discretization of the diffusive continuum model,
- Iterative staggering to solve the coupled system, whereby the time-steps are adaptively adjusted to control the error associated with the incomplete resolution of the concentration fields.

4.1. Discretization of the c- and s-fields

The concentration field will require spatial discretization with some type of mesh, for example using a finite difference, finite volume or finite element method.

4.1.1. Temporal approximation

For the c-concentration field, we write

$$\frac{\partial c}{\partial t} = \nabla \cdot \mathbb{D} \cdot \nabla c + r(s) + \tau(c) \stackrel{\text{def}}{=} L. \tag{4.1}$$

We discretize for time=$t + \phi \Delta t$, and using a trapezoidal "$\phi - scheme$" ($0 \leq \phi \leq 1$)

$$c(t + \Delta t) \approx c(t) + \Delta t \left(\phi L(t + \Delta t) + (1 - \phi) L(t)\right). \tag{4.2}$$

Similarly for s-field,

$$\frac{\partial s}{\partial t} = \nabla \cdot \mathbb{K} \cdot \nabla s + p(c) + \gamma(s) \stackrel{\text{def}}{=} M. \tag{4.3}$$

and

$$s(t + \Delta t) \approx s(t) + \Delta t \left(\phi M(t + \Delta t) + (1 - \phi) M(t)\right). \tag{4.4}$$

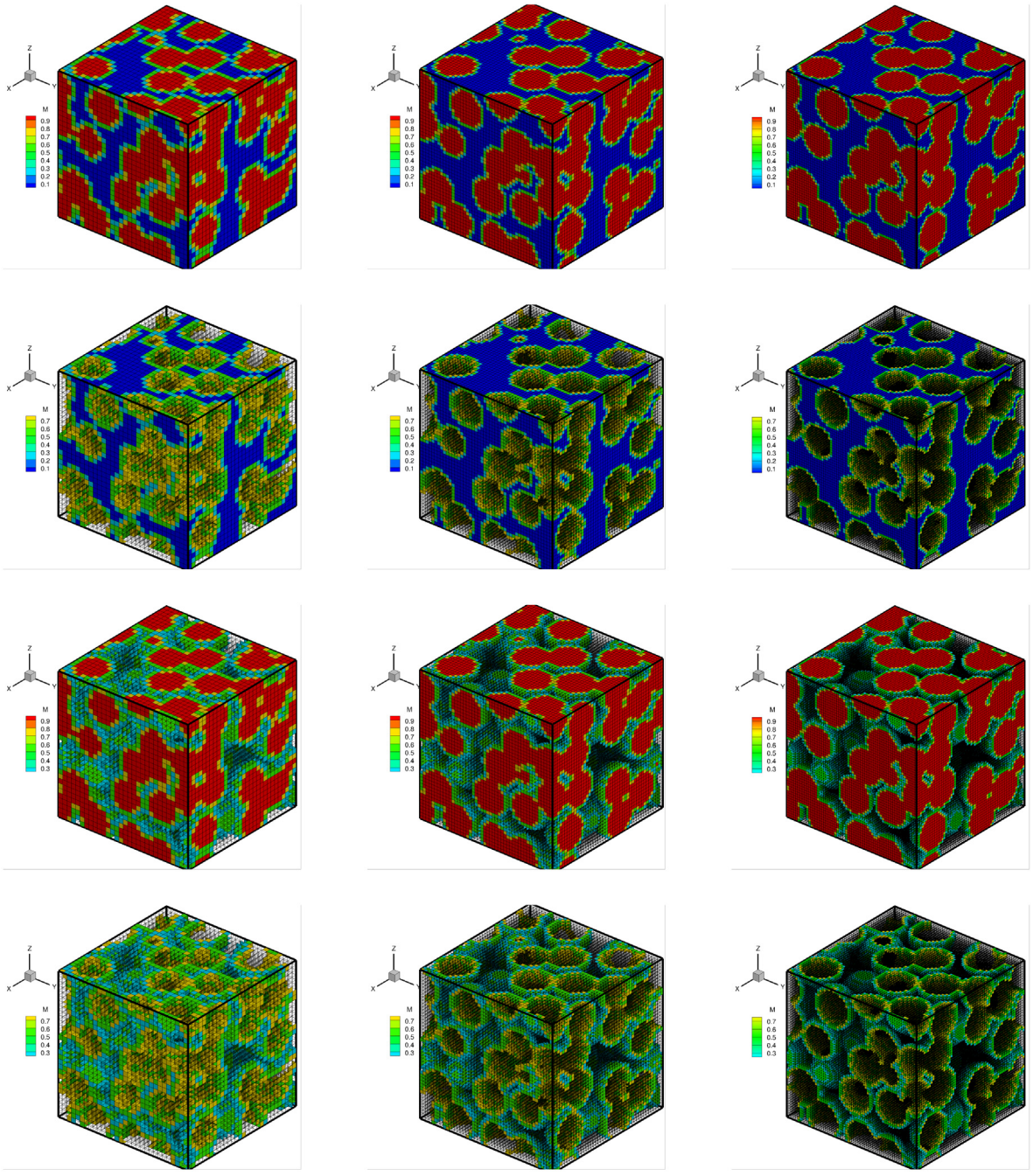

Fig. 4. Sequentially finer voxel representations of slightly overlapping particles in a matrix: for $41 \times 41 \times 41$ ($206763\, DOF$) voxel-mesh, for $61 \times 61 \times 61$ ($680943\, DOF$) voxel-mesh and for $81 \times 81 \times 81$ ($1594323\, DOF$) voxel-mesh. Top row: Microstructure/marbled tissue (particles and matrix). Second row: Just tissue (matrix). Third row: Just marbling (particles). Bottom row: Just interfaces (between marbling and matrix).

4.1.2. Spatial discretization of the fields

Numerically, the components of the gradient of c are approximated by central finite difference stencils of the basic form, for example for c, for example $\frac{\partial}{\partial x_1}$

$$\frac{\partial c}{\partial x_1}|_{(x_1,x_2,x_3)} \approx \frac{c(x_1+\Delta x_1,x_2,x_3)-c(x_1-\Delta x_1,x_2,x_3)}{2\Delta x_1} \tag{4.5}$$

for each of the (x_1, x_2, x_3)-directions, in order to form the terms needed. This is a second-order accurate stencil. For a generic second order scheme spatial derivative for an arbitrary flux component in the x_1 direction (components of $\nabla \cdot \mathcal{F}$)

$$\frac{\partial \mathcal{F}_1}{\partial x_1}|_{(x_1,x_2,x_3)} \approx \frac{\mathcal{F}_1(x_1+\frac{\Delta x_1}{2},x_2,x_3)-\mathcal{F}_1(x_1-\frac{\Delta x_1}{2},x_2,x_3)}{\Delta x_1}, \tag{4.6}$$

where generically, for example with an arbitrary material coefficient for example $\mathbb{D}(\boldsymbol{x}) = \mathbb{D}(\boldsymbol{x})\mathbf{1}$

$$\left(\underbrace{\mathbb{D}\frac{\partial c}{\partial x_1}}_{\mathcal{F}_1}\right)|_{(x_1+\frac{\Delta x_1}{2},x_2,x_3)} \approx \mathbb{D}(x_1+\frac{\Delta x_1}{2},x_2,x_3)\underbrace{\frac{c(x_1+\Delta x_1,x_2,x_3)-c(x_1,x_2,x_3)}{\Delta x_1}}_{\frac{\partial c}{\partial x_1}|_{(x_1+\frac{\Delta x_1}{2},x_2,x_3)}} \tag{4.7}$$

and

$$\left(\underbrace{\mathbb{D}\frac{\partial c}{\partial x_1}}_{\mathcal{F}_1}\right)|_{(x_1-\frac{\Delta x_1}{2},x_2,x_3)} \approx \mathbb{D}(x_1-\frac{\Delta x_1}{2},x_2,x_3)\underbrace{\frac{c(x_1,x_2,x_3)-c(x_1-\Delta x_1,x_2,x_3)}{\Delta x_1}}_{\frac{\partial c}{\partial x_1}|_{(x_1-\frac{\Delta x_1}{2},x_2,x_3)}} \tag{4.8}$$

where

$$\mathbb{D}(x_1+\frac{\Delta x_1}{2},x_2,x_3) \approx \frac{1}{2}(\mathbb{D}(x_1+\Delta x_1,x_2,x_3)+\mathbb{D}(x_1,x_2,x_3)), \tag{4.9}$$

and

$$\mathbb{D}(x_1-\frac{\Delta x_1}{2},x_2,x_3) \approx \frac{1}{2}(\mathbb{D}(x_1,x_2,x_3)+\mathbb{D}(x_1-\Delta x_1,x_2,x_3)). \tag{4.10}$$

These approximations are made for all components and combinations in $\mathbb{D}\frac{\partial c}{\partial x_j}$ appearing in the field equations. The mathematical representation of the derivatives can be summarized in the following manner ($j = 1, 2, 3$), for example for $j = 1$:

1. **VOXEL-GRADIENT:** $\mathbb{D}\frac{\partial c}{\partial x_1} \approx \mathbb{D}(x_1,x_2,x_3)\frac{c(x_1+\Delta x_1,x_2,x_3)-c(x_1-\Delta x_1,x_2,x_3)}{2\Delta x_1}$
2. **VOXEL-LAPLACIAN:**

$$\begin{aligned}\frac{\partial}{\partial x_1}\left(\mathbb{D}\frac{\partial c}{\partial x_1}\right) &\approx \frac{\left(\mathbb{D}\frac{\partial c}{\partial x_1}\right)|_{(x_1+\frac{\Delta x_1}{2},x_2,x_3)}-\left(\mathbb{D}\frac{\partial c}{\partial x_1}\right)|_{(x_1-\frac{\Delta x_1}{2},x_2,x_3)}}{\Delta x_1}\\ &= \frac{1}{\Delta x_1}\left[\mathbb{D}(x_1+\frac{\Delta x_1}{2},x_2,x_3)\left(\frac{c(x_1+\Delta x_1,x_2,x_3)-c(x_1,x_2,x_3)}{\Delta x_1}\right)\right]\\ &\quad -\frac{1}{\Delta x_1}\left[\mathbb{D}(x_1-\frac{\Delta x_1}{2},x_2,x_3)\left(\frac{c(x_1,x_2,x_3)-c(x_1-\Delta x_1,x_2,x_3)}{\Delta x_1}\right)\right]\end{aligned}$$

3. **VOXEL-INTERFACE:** $\mathbb{D}(x_1 \pm \frac{\Delta x_1}{2},x_2,x_3) \approx \frac{1}{2}\left(\mathbb{D}(x_1 \pm \Delta x_1,x_2,x_3)+\mathbb{D}(x_1,x_2,x_3)\right)$

Remark. To illustrate second-order accuracy, consider a Taylor series expansion for an arbitrary function u

$$\begin{aligned}c(x_1+\Delta x_1,x_2,x_3) &= c(x_1,x_2,x_3)+\frac{\partial c}{\partial x_1}|_{(x_1,x_2,x_3)}\Delta x_1+\frac{1}{2}\frac{\partial^2 c}{\partial x_1^2}|_{(x_1,x_2,x_3)}(\Delta x_1)^2 x_1\\ &\quad +\frac{1}{6}\frac{\partial^3 c}{\partial x_1^3}|_{(x_1,x_2,x_3)}(\Delta x_1)^3+\mathcal{O}((\Delta x_1)^4)\end{aligned} \tag{4.11}$$

and

$$c(x_1 - \Delta x_1, x_2, x_3) = c(x_1, x_2, x_3) - \frac{\partial c}{\partial x_1}|_{(x_1,x_2,x_3)}\Delta x_1 + \frac{1}{2}\frac{\partial^2 c}{\partial x_1^2}|_{(x_1,x_2,x_3)}(\Delta x_1)^2 x_1 - \frac{1}{6}\frac{\partial^3 c}{\partial x_1^3}|_{(x_1,x_2,x_3)}(\Delta x_1)^3 + \mathcal{O}((\Delta x_1)^4). \tag{4.12}$$

Subtracting the two expressions yields

$$\frac{\partial c}{\partial x_1}|_{(x_1,x_2,x_3)} = \frac{c(x_1 + \Delta x_1, x_2, x_3) - c(x_1 - \Delta x_1, x_2, x_3)}{2\Delta x_1} + \mathcal{O}((\Delta x_1)^2). \tag{4.13}$$

All other derivatives follow from this basic process, which is relatively standard in the all stencil-based discretizations. For the s-field, the discretization is the same.

Remark. At the length-scales of interest, it is questionable whether the ideas of a sharp material interface are justified. Accordingly, later, we simulated the system with and without Laplacian smoothing, whereby one smooths the material data by post-processing the original material data, voxel by voxel, to produce a smoother material representation, for example, for $\mathbb{D}$ (using the previous voxel approximations and nodal subscript notation):

$$\begin{aligned}\nabla^2 \mathbb{D} = &\frac{1}{(\Delta x_i)^2}\left(\mathbb{D}_{i+1,j,k} - 2\mathbb{D}_{i,j,k} + \mathbb{D}_{i-1,j,k}\right)\\ &+ \frac{1}{(\Delta x_j)^2}\left(\mathbb{D}_{i,j+1,k} - 2\mathbb{D}_{i,j,k} + \mathbb{D}_{i,j-1,k}\right)\\ &+ \frac{1}{(\Delta x_k)^2}\left(\mathbb{D}_{i,j,k+1} - 2\mathbb{D}_{i,j,k} + \mathbb{D}_{i,j,k-1}\right) = 0\end{aligned} \tag{4.14}$$

which yields a smoother value of $\mathbb{D}_{i,j,k}$, denoted $\hat{\mathbb{D}}_{i,j,k}$, given by

$$\nabla^2 \mathbb{D} = 0 \Rightarrow \hat{\mathbb{D}}_{i,j,k} = \frac{1}{6}\left(\mathbb{D}_{i+1,j,k} + \mathbb{D}_{i-1,j,k} + \mathbb{D}_{i,j+1,k} + \mathbb{D}_{i,j-1,k} + \mathbb{D}_{i,j,k+1} + \mathbb{D}_{i,j,k-1}\right). \tag{4.15}$$

The same process was applied to the other parameters, generically denoted, $A(\boldsymbol{x})$, by enforcing $\nabla_x^2 A = 0$, as well as for any other material data. The simulations were run with and without data smoothing, with the results being negligibly different for sufficiently fine voxel-meshes.

4.2. Iterative (implicit) solution method

Implicit time-stepping methods, with time-step size adaptivity, built on approaches found in Zohdi [9–12,16], will be used throughout the upcoming analysis. In order to introduce basic concepts, we consider a first order vector-valued differential equation

$$\dot{\boldsymbol{U}} = \boldsymbol{F}(\boldsymbol{U}), \tag{4.16}$$

which, after being discretized using a trapezoidal "ϕ-method" ($0 \leq \phi \leq 1$)

$$\boldsymbol{U}^{L+1} = \boldsymbol{U}^L + \Delta t\left(\phi \boldsymbol{F}(\boldsymbol{U}^{L+1}) + (1-\phi)\boldsymbol{F}(\boldsymbol{U}^L)\right), \tag{4.17}$$

yields the following abstract form

$$\mathcal{A}(\boldsymbol{U}^{L+1}) = \mathcal{B}. \tag{4.18}$$

Explicitly, $\boldsymbol{U}$ is the vector of all voxel values in the system

$$\boldsymbol{U} \stackrel{\text{def}}{=} \left\{ \begin{array}{c} c_1 \\ c_2 \\ c_3 \\ \cdots \\ c_N \\ s_1 \\ s_2 \\ s_3 \\ \cdots \\ s_N \end{array} \right\} \tag{4.19}$$

It is convenient to write

$$\mathcal{A}(\boldsymbol{U}^{L+1}) - \mathcal{B} = \mathcal{G}(\boldsymbol{U}^{L+1}) - \boldsymbol{U}^{L+1} + \mathcal{R} = \boldsymbol{0}, \tag{4.20}$$

where $\mathcal{R}$ is a remainder term that does not depend on the solution, i.e. $\mathcal{R} \neq \mathcal{R}(\boldsymbol{U}^{L+1})$. A straightforward iterative scheme can be written as

$$\boldsymbol{U}^{L+1,K} = \mathcal{G}(\boldsymbol{U}^{L+1,K-1}) + \mathcal{R}, \tag{4.21}$$

where $K = 1, 2, 3, \ldots$ is the index of iteration within time-step $L+1$. The convergence of such a scheme is dependent on the behavior of $\mathcal{G}$. Namely, a sufficient condition for convergence is that $\mathcal{G}$ is a contraction mapping for all $\boldsymbol{U}^{L+1,K}$, $K = 1, 2, 3...$ In order to investigate this further, we define the iteration error as

$$\varpi^{L+1,K} \stackrel{\text{def}}{=} \boldsymbol{U}^{L+1,K} - \boldsymbol{U}^{L+1}. \tag{4.22}$$

A necessary restriction for convergence is iterative self consistency, i.e. the "exact" (discretized) solution must be represented by the scheme

$$\mathcal{G}(\boldsymbol{U}^{L+1}) + \mathcal{R} = \boldsymbol{U}^{L+1}. \tag{4.23}$$

Enforcing this restriction, a sufficient condition for convergence is the existence of a contraction mapping

$$\begin{aligned} \varpi^{L+1,K} = \|\boldsymbol{U}^{L+1,K} - \boldsymbol{U}^{L+1}\| &= \|\mathcal{G}(\boldsymbol{U}^{L+1,K-1}) - \mathcal{G}(\boldsymbol{U}^{L+1})\| \\ &\leq \eta^{L+1,K} \|\boldsymbol{U}^{L+1,K-1} - \boldsymbol{U}^{L+1}\|, \end{aligned} \tag{4.24}$$

where, if $0 \leq \eta^{L+1,K} < 1$ for each iteration K, then $\varpi^{L+1,K} \rightarrow \boldsymbol{0}$ for any arbitrary starting value $\boldsymbol{U}^{L+1,K=0}$, as $K \rightarrow \infty$. This type of contraction condition is sufficient, but not necessary, for convergence. Inserting these approximations into $\dot{\boldsymbol{U}} = \boldsymbol{F}(\boldsymbol{U})$ leads to

$$\boldsymbol{U}^{L+1,K} \approx \underbrace{\Delta t \left(\phi \boldsymbol{F}(\boldsymbol{U}^{L+1,K-1})\right)}_{\mathcal{G}(\boldsymbol{U}^{L+1,K-1})} + \underbrace{\Delta t(1-\phi)\boldsymbol{F}(\boldsymbol{U}^{L}) + \boldsymbol{U}^{L}}_{\mathcal{R}}, \tag{4.25}$$

whose contraction constant is scaled by $\eta \propto \phi \Delta t$. Therefore, if convergence is slow within a time-step, the time-step size, which is adjustable, can be reduced by an appropriate amount to increase the rate of convergence. Decreasing the time-step size improves the convergence, however, we want to simultaneously maximize the time-step sizes to decrease overall computing time, while still meeting an error tolerance on the numerical solution's accuracy. In order to achieve this goal, we follow an approach found in Zohdi [9] originally developed for continuum thermo-chemical multifield problems in which one first approximates

$$\eta^{L+1,K} \approx S(\Delta t)^p \tag{4.26}$$

(S is a constant) and secondly one assumes the error within an iteration to behave according to

$$(S(\Delta t)^p)^K \varpi^{L+1,0} = \varpi^{L+1,K}, \tag{4.27}$$

$K = 1, 2, \ldots$, where $\varpi^{L+1,0}$ is the initial norm of the iterative error and S is intrinsic to the system.[1] Our goal is to meet an error tolerance in exactly a preset number of iterations. To this end, one writes

$$(S(\Delta t_{\text{tol}})^p)^{K_d} \varpi^{L+1,0} = C_{tol}, \tag{4.28}$$

[1] For the class of problems under consideration, due to the quadratic dependency on Δt, $p \approx 1$.

where C_{tol} is a (coupling) tolerance and where K_d is the number of desired iterations.[2] If the error tolerance is not met in the desired number of iterations, the contraction constant $\eta^{L+1,K}$ is too large. Accordingly, one can solve for a new smaller step size, under the assumption that S is constant,

$$\Delta t_{\text{tol}} = \Delta t \left(\frac{(\frac{C_{tol}}{\varpi^{L+1,0}})^{\frac{1}{pK_d}}}{(\frac{\varpi^{L+1,K}}{\varpi^{L+1,0}})^{\frac{1}{pK}}} \right). \tag{4.29}$$

The assumption that S is constant is not critical, since the time-steps are to be recursively refined and unrefined throughout the simulation. Clearly, the expression in Eq. (4.29) can also be used for time-step enlargement, if convergence is met in less than K_d iterations.[3] Specifically, the solution steps are, within a time-step (Fig. 5):

- (1): Start a global fixed iteration (set $i = 1, \ldots, N_n$ (voxel counter) and $K = 0$ (iteration counter))
- (2): If $i > N_n$ then go to (4)
- (3): If $i \leq N_n$ then:
  - (a) Compute the concentration $c_i^{L+1,K}$
  - (b) Go to (2) for the next voxel ($i = i + 1$)
- (4): Repeat steps 1–3 for the voxels, $i = 1, \ldots, N_n$.
- (5): Measure error (normalized) quantities (where w_c is a weight on the cell contribution and w_s is a weight on the regulator contribution)
  - (a) $\varpi^{L+1,K} \stackrel{\text{def}}{=} w_c \frac{\sum_{i=1}^{N_n} \|c_i^{L+1,K} - c_i^{L+1,K-1}\|}{\sum_{i=1}^{N_n} \|c_i^{L+1,K}\|} + w_s \frac{\sum_{i=1}^{N_p} \|s_i^{L+1,K} - s_i^{L+1,K-1}\|}{\sum_{i=1}^{N_p} \|s_i^{L+1,K}\|}$
  - (b) $E_K \stackrel{\text{def}}{=} \frac{\varpi^{L+1,K}}{TOL}$ where TOL is an error tolerance.
  - (c) $\Lambda_K \stackrel{\text{def}}{=} \left(\frac{(\frac{TOL}{\varpi^{L+1,0}})^{\frac{1}{pK_d}}}{(\frac{\varpi^{L+1,K}}{\varpi^{L+1,0}})^{\frac{1}{pK}}} \right)$.
- (6): If the tolerance is met: $E_K \leq 1$ and $K < K_d$ then
  - (a) Increment time: $t = t + \Delta t$
  - (b) Construct the next time-step: $(\Delta t)^{new} = \Lambda_K (\Delta t)^{old}$,
  - (c) Select the minimum size: $\Delta t = MIN((\Delta t)^{lim}, (\Delta t)^{new})$ and go to (1)
- (7): If the tolerance is not met: $E_K > 1$ and $K < K_d$ then
  - (a) Update the iteration counter: $K = K + 1$
  - (b) Reset the voxel counter: $i = 1$
  - (c) Go to (2)
- (8): If the tolerance is not met ($E_K > 1$) and $K = K_d$ then
  - (a) Construct a new time-step: $(\Delta t)^{new} = \Lambda_K (\Delta t)^{old}$
  - (b) Restart at time t and go to (1)

Time-step size adaptivity is critical, since the system's dynamics can dramatically change over the course of time, possibly requiring quite different time-step sizes to control the iterative error. However, to maintain the accuracy of the time-stepping scheme, one must respect an upper bound dictated by the discretization error, i.e., $\Delta t \leq \Delta t^{lim}$. Note that in step (5), Λ_K may enlarge the time-step if the error is lower than the preset tolerance. At a given time, once the process is complete, the time is incremented forward and the process is repeated. The overall goal is to deliver solutions where the iterative error is controlled and the temporal discretization accuracy dictates the upper limit on the time-step size (Δt^{lim}). Clearly, there are various combinations of solution methods that one can choose from. For example, for the overall field coupling, one may choose implicit or explicit staggering and within the staggering process, either implicit ($0 < \phi \leq 1$) or explicit time-stepping ($\phi = 0$), and, in the case of implicit time-stepping, iterative or direct solvers. Furthermore, one could employ internal iterations for each field equation, then update, more sophisticated metrics for certain components of the error, etc. For example, we utilized an error

[2] Typically, K_d is chosen to be between five to ten iterations, although this is problem and analyst dependent.

[3] At the implementation level, since the exact solution is unknown, the following relative error term is used, $\varpi^{L+1,K} \stackrel{\text{def}}{=} \boldsymbol{U}^{L+1,K} - \boldsymbol{U}^{L+1,K-1}$.

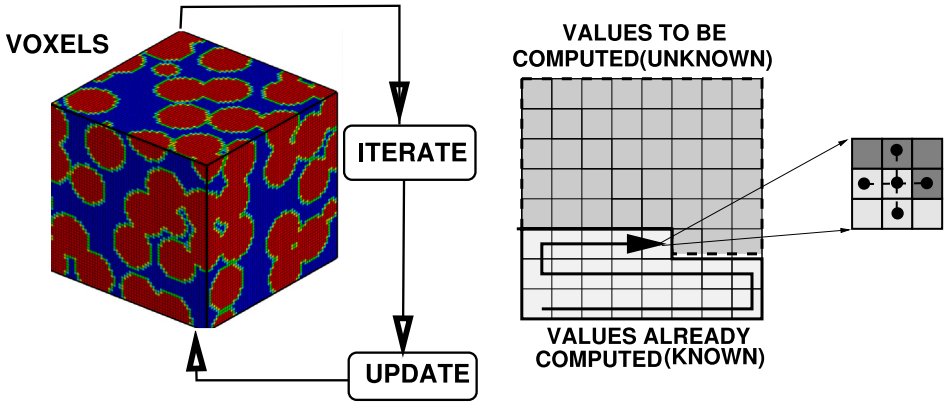

Fig. 5. The overall iterative (left) solution and the matrix-free approach using a moving front through the voxels (right) During the iterative solution process, the most current value of a voxel is used in any calculation, for example a construction of the Laplacian, or any other term in the governing differential equations.

measure that used the concentrations at the voxels of the Finite Difference grid, but other metrics are certainly possible. For details see Zohdi [9–12,16].

Remark 1. Because the internal system solvers within the staggering scheme are also iterative and use the previously converged solution as their starting value to solve the system of equations, a field that is relatively insensitive at given stage of the simulation will converge in very few internal iterations (perhaps even one). Staggering schemes are widely used in the computational mechanics literature, dating back, at least, to Zienkiewicz [17] and Zienkiewicz et al. [18]. For in depth overviews, see the works of Lewis and Schrefler (Lewis et al. [19] and Lewis and Schrefler [20]) and a series of works by Schrefler and collaborators: Schrefler [21], Turska and Schrefler [22], Bianco et al. [23] and Wang and Schrefler [24].

Remark 2. During the iterative solution process, the most current value of a voxel is used in any calculation, for example a construction of the Laplacian, or any other term in the governing differential equations.

Remark 3. At the length-scales of interest, it is questionable whether the ideas of a sharp material interface are justified. Accordingly, we simulated the system with and without Laplacian smoothing, whereby one smooths the material data by post-processing the material data, voxel by voxel, to produce a smoother material representation. The simulations were run with and without data smoothing, with the results being negligibly different for sufficiently fine meshes.

5. Operation counts in a voxel-based method

The cost of constructing an array for a temporal update using a voxel calculation is:

- $P \times V$, where V is the number of voxels and $S = \mathcal{O}(10)$ is, associated with summing up the terms needed to construct L and M. Specifically, there are six Laplacian terms needed in Eqs. (2.3) and (2.4). To construct each Laplacian term one must perform four operations. Thus, S = 24 operations in total.
- Thus, $K \times S \times V \approx 24$ KV, is the total count per time-step, where K is the number of iterations in a time-step.

The cost of constructing an array for temporal update using an FEM calculation is associated with (1) meshing the microstructure (mappings, etc.) (2) numerically integrating the weak form (3) generating a stiffness matrix and (4) solving a system of equations. Specifically (Zohdi [25]):

- Construction of the stiffness matrix: $P \times E$, where E is the number of elements and $P = \mathcal{O}(700)$ stems from mapping and integrating the terms needed to construct the stiffness matrix associated with the weak form $\mathcal{O}(10)$, of which there are 35 entries $[8 \times 8 = 64]$ in a (symmetric) linear diffusion element stiffness matrix (linear hexahedra), which must be computed for both equations, yielding $P = 70 \times 10 = 700$.

- Iterative solution: $I \times Q$, where I is the number of iterations associated with, for example, a Conjugate-Gradient solver and Q is the cost of a matrix–vector multiplication. Q is on the order of N^q, where N^q is the number of voxels in the system and $1 < q \leq 2$, and depends on the sparsity of the stiffness matrix. For example, for linearized elasticity, using an element-by-element multiplier (not counting preconditioning), for linearized elasticity and using linear hexahedra, $Q = 70N$. Thus, $I \times Q \approx 70IN$.
- Thus, a comparison of the total operation counts between the Voxel method and FEM is roughly

$$\frac{Voxel}{FEM} \approx \frac{24KV}{700E + 70IN}. \tag{5.1}$$

Eq. (5.1) clearly shows the favorable ratio of operation counts of the voxel approach relative to FEM. In more complex (nonlinear) problems, where the stiffness matrix would have to be reformed after each iteration, the term $700E$ would need to be multiplied by I.

6. Numerical examples

As an example, we consider a cubical domain with an initial interior concentration of zero cells and zero regulator. We inject both cells and regulator at a given location at the top (Fig. 2). The injection site has a controlled concentration of both cells and regulator over time. The boundary conditions for the cells and regulator were held to be zero, other than at the injection site. Key qualitative ratios can be identified by considering the special case of steady state and spatially uniform fields:

$$\frac{\partial c}{\partial t} = \nabla \cdot \mathbb{D} \cdot \nabla c + r(s) + \tau(c) = 0 \Rightarrow r(s) + \tau(c) = 0 \tag{6.1}$$

and

$$\frac{\partial s}{\partial t} = \nabla \cdot \mathbb{K} \cdot \nabla s + p(c) + \gamma(s) = 0 \Rightarrow p(c) + \gamma(s) = 0. \tag{6.2}$$

Assuming linear relationships yields ($\tau(c) = -\tau c < 0$)

$$r(s) + \tau(c) = rs - \tau c = 0 \Rightarrow c = \frac{rs}{\tau} \tag{6.3}$$

and ($\gamma(s) = -\gamma s < 0$)

$$p(c) + \gamma(s) = pc - \gamma s = 0 \Rightarrow c = \frac{\gamma s}{p}. \tag{6.4}$$

This, while it is unrealistic that the field would ever be uniform, these ratios are key to understanding what controls the growth of c. We considered a heterogeneous domain where the medium has a microstructure comprised of randomly distributed spheres (occupying approximately 25% volume fraction) in a homogeneous matrix-an idealization of "marbled" tissue. The following parameters were used (with standard metric units used throughout):

- size of the domain was $0.01 \times 0.01 \times 0.01$ m,
- injection site, a controlled cell concentration $c(t) = c_o e^{at}$, $c_o = 1$, $a = 0.01$,
- injection site, a controlled regulator concentration $s(t) = s_o e^{at}$, $s_o = 1$, $a = 0.01$,
- injection site was 0.005×0.00125 m (elliptical cross-section) and 0.0025 m deep,
- total simulation time was $T = 20$ seconds,
- cell proliferation term, $r(s) = +\hat{r}s$, with a different $\hat{r}$ for each material phase,
- cell apoptosis term, $\tau(c) = -\hat{\tau}c$, with a different $\hat{\tau}$ for each material phase,
- regulator production term, $p(c) = +\hat{p}c$, with a different $\hat{p}$ for each material phase,
- regulator loss term, $\gamma(s) = -\hat{\gamma}s$, with a different $\hat{\gamma}$ for each material phase,
- homogeneous base, $\hat{r}_o = 20$, $\hat{r}_{1R} = \frac{\hat{r}_1}{\hat{r}_o} = 10$,
- homogeneous base, $\hat{\tau}_o = 0.1$, $\hat{\tau}_{1R} = \frac{\hat{\tau}_1}{\hat{\tau}_o} = 10$,
- homogeneous base, $\hat{p}_o = 0.001$, $\hat{p}_{1R} = \frac{\hat{p}_1}{\hat{p}_o} = 100$,
- homogeneous base, $\hat{\gamma}_o = 0.1$, $\hat{\gamma}_{1R} = \frac{\hat{\gamma}_1}{\hat{\gamma}_o} = 100$,
- homogeneous base, $\mathbb{D} = D\mathbf{1}$, $D = 10^{-6}$,
- homogeneous base, $\mathbb{K} = K\mathbf{1}$, $K = 10^{-7}$,

- heterogeneous/marbled case, $\hat{r}_o = 20$, $\hat{r}_{1R} = \frac{\hat{r}_1}{\hat{r}_o} = 10$, $\hat{r}_{2R} = \frac{\hat{r}_2}{\hat{r}_o} = 1$,
- heterogeneous/marbled case, $\hat{\tau}_o = 0.1$, $\hat{\tau}_{1R} = \frac{\hat{\tau}_1}{\hat{\tau}_o} = 10$, $\hat{\tau}_{2R} = \frac{\hat{\tau}_2}{\hat{\tau}_o} = 1$,
- heterogeneous/marbled case, $\hat{p}_o = 0.001$, $\hat{p}_{1R} = \frac{\hat{p}_1}{\hat{p}_o} = 100$, $\hat{p}_{2R} = \frac{\hat{p}_2}{\hat{p}_o} = 1$,
- heterogeneous/marbled case, $\hat{\gamma}_o = 0.1$, $\hat{\gamma}_{1R} = \frac{\hat{\gamma}_1}{\hat{\gamma}_o} = 100$, $\hat{\gamma}_{2R} = \frac{\hat{\gamma}_2}{\hat{\gamma}_o} = 1$,
- heterogeneous/marbled case, $\mathbb{D}_1 = D_1\mathbf{1}$, $D_1 = 10^{-6}$ and $\mathbb{D}_2 = D_2\mathbf{1}$, $D_2 = 10^{-7}$,
- heterogeneous/marbled case, $\mathbb{K}_1 = K_1\mathbf{1}$, $K_1 = 10^{-7}$ and $\mathbb{K}_2 = K_2\mathbf{1}$, $K_2 = 10^{-8}$.

The time-steps were initially started to be quite small in order to allow the system to evolve the time-step size during the beginning of the simulation. A trapezoidal time-stepping parameter of $\phi = 0.5$ was chosen. In this case, we started the time-step size at 0.001 s and allowed it to be enlarged up to 20 times that size, if the algorithm and error estimates warranted it (which was the case in the examples given). During the computations with the heterogeneous "marbled" media, the spatial discretization meshes were repeatedly refined until the solutions did not exhibit any more sensitivity to further refinement of the grid-spacing. We started with meshes such as a $21 \times 21 \times 21$ mesh, arising from having a cubical mesh with 10 voxels from the centerline plane of symmetry and one voxel in the middle, and then repeatedly refined in the following sequential manner:

1. **Mesh # 1:** a $21 \times 21 \times 21$ mesh, which has 9261 *degrees of freedom* per field, for a total of 18,522 *degrees of freedom*,
2. **Mesh # 2:** a $41 \times 41 \times 41$ mesh, which has 68,921 *degrees of freedom* per field, for a total of 137,842 *degrees of freedom*,
3. **Mesh # 3:** a $61 \times 61 \times 61$ mesh which has 226,981 *degrees of freedom* per field, for a total of 453,962 *degrees of freedom*, etc.

Approximately between a 41-level and a 61-level mesh, the results stabilized, indicating that the results are essentially free of any appreciable numerical error. Fig. 6 illustrates the morphology of the tissue microstructure as resolved by the grid. Fig. 7 shows cross-sections of the concentration of cells domain over time. Fig. 8 depicts the evolution of the time-step size over time. The computations are designed so that they take a few minutes on a standard laptop. The selected parameter choices were provided to illustrate the overall working on the model, and a wide variety of parameter choices are possible, depending on the application. This is discussed further next.

7. Genomic machine-learning framework

The rapid rate at which these simulations can be completed enables the ability to explore inverse problems seeking to determine what parameter combinations can deliver a desired result. Following Zohdi [26–31], we formulate the objective as a cost function minimization problem that seeks system parameters that match a desired response by minimizing a cost/error function $\Pi(\boldsymbol{\Lambda})$. Specifically, we use

$$\Pi = w_1 \frac{\|ANTIBODIES - TARGET_1\|}{\|TARGET_1\|} + w_2 \frac{\|ANTIGENS - TARGET_2\|}{\|TARGET_2\|}. \tag{7.1}$$

The system parameter search is conducted within the constrained ranges of $\Lambda_1^{(-)} \leq \Lambda_1 \leq \Lambda_1^{(+)}$, $\Lambda_2^{(-)} \leq \Lambda_2 \leq \Lambda_2^{(+)}$ and $\Lambda_3^{(-)} \leq \Lambda_3 \leq \Lambda_3^{(+)}$, etc. These upper and lower limits would, in general, be dictated by what is physically feasible. The system parameters to vary and optimize are the 20 parameters associated with injection concentration of antibodies, injection concentration of antigens, antibody reaction rate constants, antigen reaction rate constants, antibody diffusion constants, antigen diffusion constants, antibody regulator constants and antigen regulator constants.

7.1. System parameter search/machine-learning algorithm

Cost functions such as Π are nonconvex in design parameter space and often nonsmooth. Their minimization is usually difficult with direct application of gradient methods. This motivates nonderivative search methods, for example those found in machine-learning algorithms. One of the most basic subsets of machine-learning algorithms are so-called genetic algorithms. For a review of genetic algorithms, see the pioneering work of John Holland [32,33], as well as Goldberg [34], Davis [35], Onwubiko [36] and Goldberg and Deb [37]. A description of the algorithm will be described next (Zohdi [26–31]).

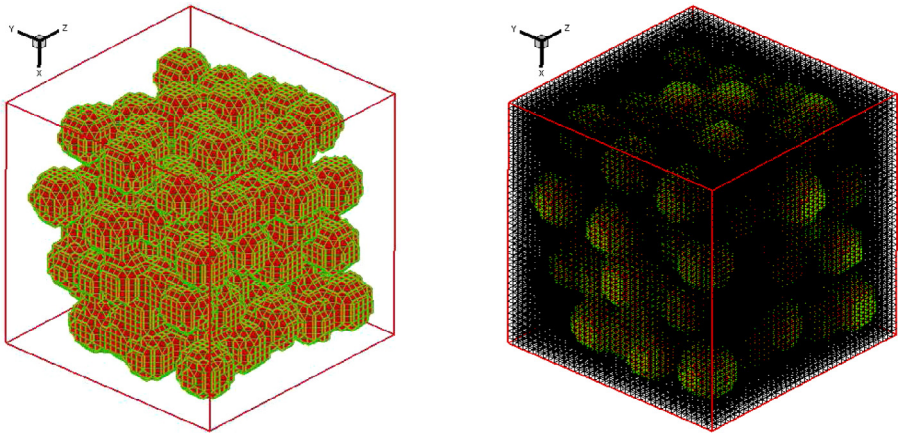

Fig. 6. Left, the morphology of the tissue microstructure and, right, the morphology of the tissue microstructure and mesh.

7.2. Algorithmic specifics

Following Zohdi [26–31] the algorithm is as follows (see Figs. 9 and 10)

- **STEP 1:** Randomly generate a population of S starting genetic strings, $\boldsymbol{\Lambda}^i$, $(i = 1, 2, 3, \ldots, S)$:

$$\boldsymbol{\Lambda}^i \stackrel{\text{def}}{=} \left\{ \begin{array}{c} \Lambda_1^i \\ \Lambda_2^i \\ \Lambda_3^i \\ \ldots \\ \Lambda_N^i \end{array} \right\} \tag{7.2}$$

  The system parameter search is conducted within the constrained ranges of $\Lambda_1^{(-)} \leq \Lambda_1 \leq \Lambda_1^{(+)}$, $\Lambda_2^{(-)} \leq \Lambda_2 \leq \Lambda_2^{(+)}$ and $\Lambda_3^{(-)} \leq \Lambda_3 \leq \Lambda_3^{(+)}$, etc.
- **STEP 2:** Compute fitness of each string $\Pi(\boldsymbol{\Lambda}^i)$, (i = 1, …, S)
- **STEP 3:** Rank genetic strings: $\boldsymbol{\Lambda}^i$, (i = 1, …, S)
- **STEP 4:** Mate nearest pairs and produce two offspring, (i = 1, …, S):

$$\boldsymbol{\lambda}^i \stackrel{\text{def}}{=} \boldsymbol{\Phi} \circ \boldsymbol{\Lambda}^i + (\mathbf{1} - \boldsymbol{\Phi}) \circ \boldsymbol{\Lambda}^{i+1} \stackrel{\text{def}}{=} \left\{ \begin{array}{l} \phi_1 \Lambda_1^i \\ \phi_2 \Lambda_2^i \\ \phi_3 \Lambda_3^i \\ \ldots \\ \phi_N \Lambda_N^i \end{array} \right\} + \left\{ \begin{array}{l} (1-\phi_1)\Lambda_1^{i+1} \\ (1-\phi_2)\Lambda_2^{i+1} \\ (1-\phi_3)\Lambda_3^{i+1} \\ \ldots \\ (1-\phi_N)\Lambda_N^{i+1} \end{array} \right\} \tag{7.3}$$

  and

$$\boldsymbol{\lambda}^{i+1} \stackrel{\text{def}}{=} \boldsymbol{\Psi} \circ \boldsymbol{\Lambda}^i + (\mathbf{1} - \boldsymbol{\Psi}) \circ \boldsymbol{\Lambda}^{i+1} \stackrel{\text{def}}{=} \left\{ \begin{array}{l} \psi_1 \Lambda_1^i \\ \psi_2 \Lambda_2^i \\ \psi_3 \Lambda_3^i \\ \ldots \\ \psi_N \Lambda_N^i \end{array} \right\} + \left\{ \begin{array}{l} (1-\psi_1)\Lambda_1^{i+1} \\ (1-\psi_2)\Lambda_2^{i+1} \\ (1-\psi_3)\Lambda_3^{i+1} \\ \ldots \\ (1-\psi_N)\Lambda_N^{i+1} \end{array} \right\} \tag{7.4}$$

  where for this operation, the ϕ_i and ψ_i are random numbers, such that $0 \leq \phi_i \leq 1$, $0 \leq \psi_i \leq 1$, which are different for each component of each genetic string
- **STEP 5:** Eliminate the bottom M strings and keep top K parents and their K offspring (K offspring+K parents+$M = S$)
- **STEP 6:** Repeat STEPS 1–5 with top gene pool (K offspring and K parents), plus M new, randomly generated, strings

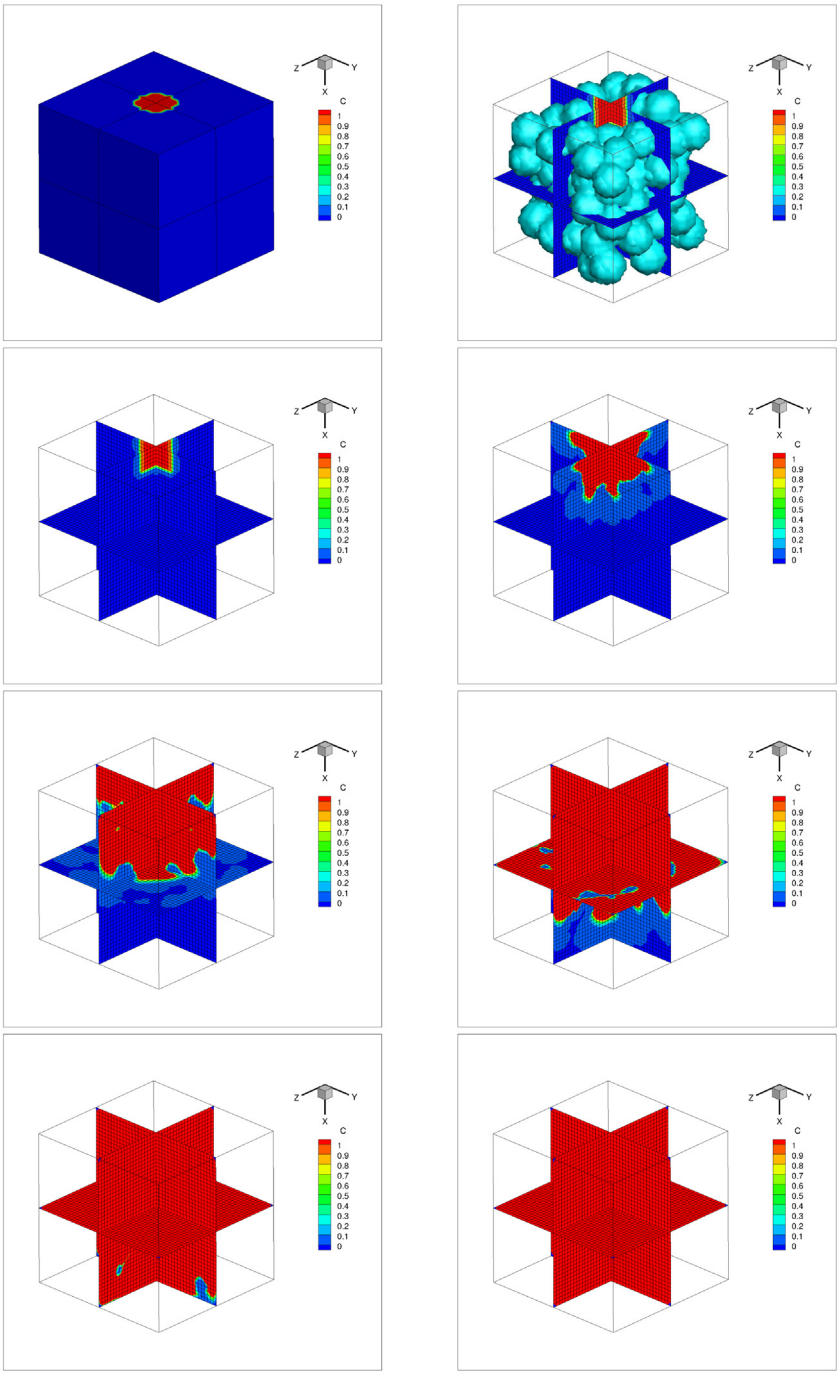

Fig. 7. The injection viewed from the exterior and the morphology of the microstructural-marbling. From left to right and top to bottom: Cell concentration (c) and growth from an injection at the surface.

- **IMPORTANT OPTIONS:** One can rescale and restart search around best performing parameter set every few generations, thus refocussing computational effort around optimal locations in design parameter space. Thus, one could use a genetic algorithm first in order to isolate multiple local minima, and then use a gradient-based

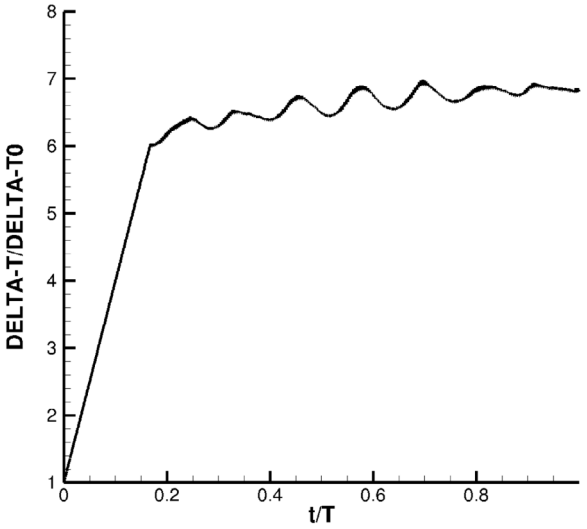

Fig. 8. The evolution of the time-step size over time.

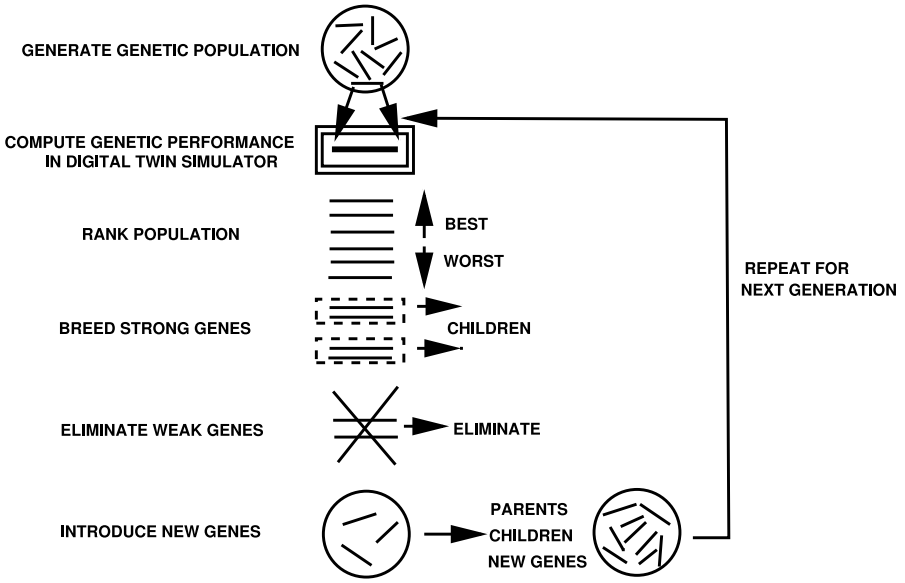

Fig. 9. The overall process for a genetic-based machine-learning algorithm.

algorithm in locally convex regions or reset the genetic algorithm to concentrate its search over these more constrained regions.

Remark 1. If one selects the mating parameters ϕ's and ψ's to be greater than one and/or less than zero, one can induce "mutations", i.e. characteristics that neither parent possesses. However, this is somewhat redundant with introduction of new random members of the population in the current algorithm.

Remark 2. If one does not retain the parents in the algorithm above, it is possible that inferior performing offspring may replace superior parents. Thus, top parents should be kept for the next generation. Additionally, retained parents do not have to be reevaluated in the next generation, making the algorithm less computationally

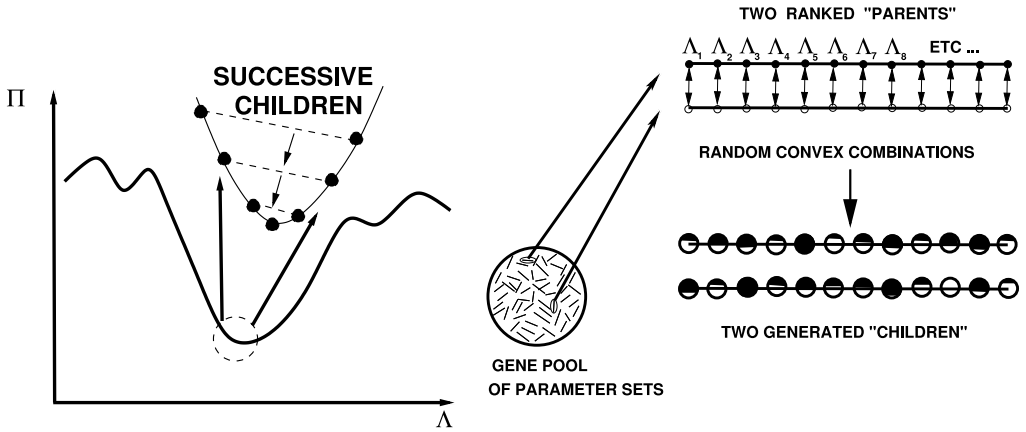

Fig. 10. The basic action of a genetic-based machine-learning algorithm.

expensive. Numerous studies of the author have shown that advantages parent retention outweighs inbreeding, for sufficiently large population sizes. We also remark that this algorithm is easily parallelizable.

7.3. Algorithmic settings

In the upcoming example, the design parameters $\boldsymbol{\Lambda} = \{\Lambda_1, \Lambda_2 ... \Lambda_N\}$ are optimized over the search intervals (20 variables): $\boldsymbol{\Lambda}_i^- \leq \boldsymbol{\Lambda}_i \leq \boldsymbol{\Lambda}_i^+$, $i = 1, 2, \ldots, 20$. Specifically, we varied the injection concentration of antibodies, injection concentration of antigens, antibody reaction rate constants, antigen reaction rate constants, antibody diffusion constants, antigen diffusion constants, antibody regulator constants and antigen regulator constants. Fig. 11 and Fig. 12 show the reduction of the cost function for the 20 parameter set. Shown are the best performing gene (design parameter set, in *red*) as a function of successive generations, as well as the average performance of the entire population of the genes (designs, in *green*). We used the following settings:

- Number of design variables: 20,
- Population size per generation: 24,
- Number of parents to keep in each generation: 6,
- Number of children created in each generation: 6,
- Number of completely new genes created in each generation: 12,
- Number of generations for re-adaptation around a new search interval: 10,
- Number of generations: 200.

7.4. Parameter search ranges and results

We considered a 20 parameter vaccine design. The following search parameter ranges were used (with $w_1 = 1$ and $w_2 = 1$):

- $\Lambda_1 =$ Injection concentration of Antibodies: $\Lambda_1^- = 0.01 \leq \Lambda_1 \leq \Lambda_1^+ = 10$,
- $\Lambda_2 =$ Injection concentration of Antigens: $\Lambda_2^- = 0.1 \leq \Lambda_2 \leq \Lambda_2^+ = 10$,
- $\Lambda_3 =$ Antibody reaction rate constant of media 1: $\Lambda_3^- = 0.1 \leq \Lambda_3 \leq \Lambda_3^+ = 10$,
- $\Lambda_4 =$ Antibody reaction rate constant of media 2: $\Lambda_4^- = 0.1 \leq \Lambda_4 \leq \Lambda_4^+ = 10$,
- $\Lambda_5 =$ Antibody reaction rate constant of media 3 : $\Lambda_5^- = 0.1 \leq \Lambda_5 \leq \Lambda_5^+ = 10$,
- $\Lambda_6 =$ Antigen reaction rate constant of media 1: $\Lambda_6^- = 0.1 \leq \Lambda_6 \leq \Lambda_6^+ = 10$,
- $\Lambda_7 =$ Antigen reaction rate constant of media 2: $\Lambda_7^- = 0.1 \leq \Lambda_7 \leq \Lambda_7^+ = 10$,
- $\Lambda_8 =$ Antigen reaction rate constant of media 3 : $\Lambda_8^- = 0.1 \leq \Lambda_8 \leq \Lambda_8^+ = 10$,
- $\Lambda_9 =$ Antibody diffusion constant of media 1: $\Lambda_9^- = 0.1 \leq \Lambda_9 \leq \Lambda_9^+ = 10$,

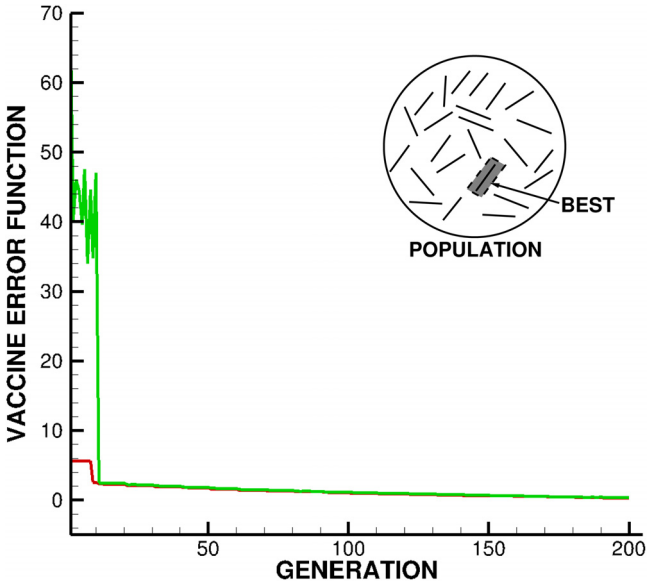

Fig. 11. Shown are the cost function for the best performing gene (*red*) as a function of successive generations, as well as the average cost function of the entire population of genes (*green*). We allowed the genetic-base machine-learning algorithm to readapt every 10 generations, leading to the (slight) nonmonotone reduction of the cost function. Often, this action is more efficient than allowing the algorithm not to readapt, since it probes around the current optimum for better local alternatives. In this case, the algorithm makes slow progress until generation 10, when a readaptation/recentering occurred, and then slowed reduced the cost function over 200 generations from approximately $\Pi \approx 2$ to $\Pi = 0.4097$. The algorithm produces a massive reduction of error from $\Pi \approx 50$ to $\Pi \approx 0.4$-a factor of 125. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

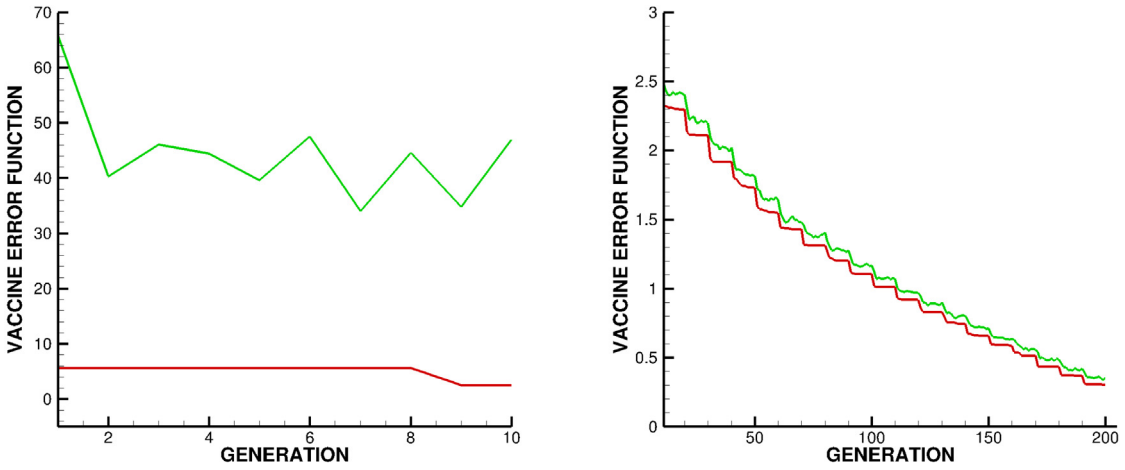

Fig. 12. ZOOM: Shown are the cost function for the best performing gene (*red*) as a function of successive generations, as well as the average cost function of the entire population of genes (*green*). We allowed the genetic-based machine-learning algorithm to readapt every 10 generations, leading to the (slight) nonmonotone reduction of the cost function. Often, this action is more efficient than allowing the algorithm not to readapt, since it probes around the current optimum for better local alternatives. In this case, the algorithm makes slow progress until generation 10, when a readaptation/recentering occurred, and then slowed reduced the cost function over 200 generations from approximately $\Pi \approx 2$ to $\Pi = 0.4097$. The algorithm produces a massive reduction of error from $\Pi \approx 50$ to $\Pi \approx 0.4$-a factor of 125. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

- Λ_{10} = Antibody diffusion constant of media 2: $\Lambda_{10}^{-} = 0.1 \leq \Lambda_{10} \leq \Lambda_{10}^{+} = 10$,
- Λ_{11} = Antibody diffusion constant of media 3 : $\Lambda_{11}^{-} = 0.1 \leq \Lambda_{11} \leq \Lambda_{11}^{+} = 10$,
- Λ_{12} = Antigen diffusion constant of media 1: $\Lambda_{12}^{-} = 0.1 \leq \Lambda_{12} \leq \Lambda_{12}^{+} = 10$,

Table 1
The system parameters ($\Lambda_1 - \Lambda_{20}$) for the best performing design (gene) with $w_1 = w_2 = 1$.

Λ_1	Λ_2	Λ_3	Λ_4	Λ_5	Λ_6	Λ_7	Λ_8	Λ_9	Λ_{10}
0.119	0.168	7.185	9.043	8.114	2.562	6.988	1.873	0.543	8.641
Λ_{11}	Λ_{12}	Λ_{13}	Λ_{14}	Λ_{15}	Λ_{16}	Λ_{17}	Λ_{18}	Λ_{19}	Λ_{20}	Π
6.042	4.673	9.197	9.408	3.730	0.078	0.072	9.720	0.0016	0.061	0.4097

- Λ_{13} = Antigen diffusion constant of media 2: $\Lambda_{13}^- = 0.1 \leq \Lambda_{13} \leq \Lambda_{13}^+ = 10$,
- Λ_{14} = Antigen diffusion constant of media 3 : $\Lambda_{14}^- = 0.1 \leq \Lambda_{14} \leq \Lambda_{14}^+ = 10$,
- Λ_{15} = Antibody regulator constant of media 1: $\Lambda_{15}^- = 0.1 \leq \Lambda_{15} \leq \Lambda_{15}^+ = 10$,
- Λ_{16} = Antibody regulator constant of media 2: $\Lambda_{16}^- = 0.001 \leq \Lambda_{16} \leq \Lambda_{16}^+ = 0.1$,
- Λ_{17} = Antibody regulator constant of media 3 : $\Lambda_{17}^- = 0.001 \leq \Lambda_{17} \leq \Lambda_{17}^+ = 0.1$,
- Λ_{18} = Antigen regulator constant of media 1: $\Lambda_{18}^- = 0.1 \leq \Lambda_{18} \leq \Lambda_{18}^+ = 10$,
- Λ_{19} = Antigen regulator constant of media 2: $\Lambda_{19}^- = 0.001 \leq \Lambda_{19} \leq \Lambda_{19}^+ = 0.1$,
- Λ_{20} = Antigen regulator constant of media 3 : $\Lambda_{20}^- = 0.001 \leq \Lambda_{20} \leq \Lambda_{20}^+ = 0.1$,

Fig. 11 and Fig. 12 illustrate the results for the cost function for the best performing gene (*red*) as a function of successive generations, as well as the average performance cost function of the entire population of genes (designs, in *green*), using design weights of $w_1 = 1$ and $w_2 = 0.1$. We allowed the genetic-based machine-learning algorithm to readapt every 10 generations, leading to the (slightly) nonmonotone reduction of the cost function. Often, this action is more efficient than allowing the algorithm not to readapt, since it probes around the current optimum for better local alternatives. Table 1 shows the final design parameters. The entire 200 generation simulation, with 24 genes per evaluation (4800 total vaccine designs) took a few minutes on a laptop, *making it ideal as a design tool.* We note that, for a given set of parameters, a complete simulation takes a fraction of a second, thus thousands of parameter sets can be evaluated in an hour, *without even exploiting the inherent parallelism of the genetic-based machine-learning algorithm. The algorithm produces a massive reduction of error from* $\Pi \approx 50$ *to* $\Pi \approx 0.4$*-a factor of 125.*

Remark-Design of a booster: For diseases that are constantly evolving, such as the current strains of COVID19, the utility of the presented method becomes even clearer, since one can start the process from the current vaccine design to develop a modified booster.

8. Discussion and summary

In summary, the purpose of this work was to present a flexible computational modeling framework that researchers in the field can easily implement and subsequently use as an efficient tool to study the immune-response to a vaccine injection. The framework is flexible enough to allow researchers to input virtually any type vaccine and immune-system interaction. However, in the present formulation, notably absent are the effects of deformation and stress in the system. At a minimum, this would require a third field equation governing the balance of linear momentum, $\nabla_x \cdot \boldsymbol{\sigma} + \boldsymbol{f} = \rho \dot{\boldsymbol{v}}$, where $\boldsymbol{\sigma}$ is the Cauchy stress, $\boldsymbol{f}$ are the body forces, ρ is the density and $\boldsymbol{v}$ is the velocity, in addition to constitutive laws for soft tissue (see the extensive works of Fung [38–40] Holzapfel [41,42] or Humphrey [43,44]). At finite deformations, the previous conservation laws can be generated in the following manner:

$$\begin{aligned} \frac{d}{dt}\int_{\omega} c\, d\omega = \frac{d}{dt}\int_{\omega_o} cJ\, d\omega_o = \int_{\omega_o} (\frac{dc}{dt}J + c\frac{dJ}{dt})\, d\omega_o = \int_{\omega_o} (\frac{dc}{dt}J + cJ\nabla_x \cdot \boldsymbol{v})\, d\omega_o \\ = \int_{\omega} (\frac{\partial c}{\partial t} + \boldsymbol{v}\cdot\nabla_x c + c\nabla_x \cdot \boldsymbol{v})\, d\omega = \int_{\omega} (\frac{\partial c}{\partial t} + \nabla_x \cdot (c\boldsymbol{v}))\, d\omega \end{aligned} \tag{8.1}$$

thus

$$\frac{\partial c}{\partial t} + \nabla_x \cdot (c\boldsymbol{v}) = \nabla \cdot \mathbb{D} \cdot \nabla c + r(s) - \tau(c). \tag{8.2}$$

and

$$\frac{\partial s}{\partial t} + \nabla_x \cdot (s\boldsymbol{v}) = \nabla \cdot \mathbb{K} \cdot \nabla s + p(c) - \gamma(s). \tag{8.3}$$

Clearly, specific material data is needed for tissue. In this regard, we again refer the reader to Murray [2] for an extensive review, with early experimental studies dating back at least to Lindquist [3] Van den Brenk [4], Crosson et al. [5], Zieske et al. [6], Franz et al. [7] and Sherratt and Murray [8]. Generally, because the distribution of water, biological fluids and chemical species within such tissue are dependent on the deformation of the solid, coupled multifield computations are necessary to realistically simulate such systems. *For example*, in many models of muscle tissue, it is usually assumed that the response depends on the concentration of a mobile chemical species present, for example, intracellular calcium Ca^{2+}, and U is the stretch along the muscle fiber, relative to a reference sarcomere length. A basic form suggested is $\boldsymbol{\sigma} = \boldsymbol{\sigma}(Ca^{2+}, U)$, where $\boldsymbol{\sigma}$ is the total Cauchy stress (active and passive), which combines the mechanical (passive) contribution and the actively generated muscle tension. We refer the reader to Rachev and Hayashi [45], Humphrey [43,44], Klepach et al. [46] and Ambrosi et al. [47] for reviews. Incorporation of such effects is under investigation by the author.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

Acknowledgments

This work has been partially supported by the UC Berkeley College of Engineering, USA and the USDA AI Institute for Next Generation Food Systems (AIFS), USDA award number 2020-67021-32855.

References

[1] Wikipedia-Vaccines: https://en.wikipedia.org/wiki/Vaccine.
[2] J.D. Murray, Mathematical Biology, third ed., Springer Verlag, 2004.
[3] G. Lindquist, The healing of skin defects: an experimental study of the white rat, Acta Chirurgica Scandinnavica 94 (Supplement 107) (1946) 1–163.
[4] H.A.S. Van den Brenk, Studies in restorative growth processes in mammalian wound healing, Br. J. Surg. 43 (1956) 525–550.
[5] C.E. Crosson, S.D. Klyce, R.W. Beuerman, Epithelial wound closure in rabbit cornea wounds invest, Ophthalmol Vis. Sci. 27 (1986) 464–473.
[6] J.D. Zieske, S.C. Higashij, S.J. Spurmic, I.K. Gipson, Biosynthetic response of the rabbit cornea to a keratectomy wound, Invest. Ophthalmol. Vis. Sci. 28 (1987) 1668–1677.
[7] J.M. Franz, B.M. Dupuy, H.E. Kaufman, R.W. Beuerman, The effects of collagen shields on epithelial wound healing in rabbits, Am. J. Ophthalmol. 108 (1989) 524–528.
[8] J.A. Sherratt, J.D. Murray, Models of epiderma wound healing, Proc. R. Soc. Lond. B 241 (1990) 29–36.
[9] T.I. Zohdi, An adaptive-recursive staggering strategy for simulating multifield coupled processes in microheterogeneous solids, Internat. J. Numer. Methods Engrg. 53 (2002) 1511–1532.
[10] T.I. Zohdi, Modeling and simulation of a class of coupled thermo-chemo-mechanical processes in multiphase solids, Comput. Methods Appl. Mech. Eng. 193 (6/8) (2004) 679–699.
[11] T.I. Zohdi, Computation of strongly coupled multifield interaction in particle-fluid systems, Comput. Methods Appl. Mech. Eng. 196 (2007) 3927–3950.
[12] T.I. Zohdi, Simulation of coupled microscale multiphysical-fields in particulate-doped dielectrics with staggered adaptive FDTD, Comput. Methods Appl. Mech. Eng. 199 (2010) 79–101.
[13] J.D. Foley, A. van Dam, J.F. Hughes, S.K. Feiner, Spatial-partitioning representations; surface detail, in: Computer Graphics: Principles and Practice, in: The Systems Programming Series, Addison-Wesley, 1990.
[14] S. Chmielewski, P. Tompalski, Estimating outdoor advertising media visibility with voxel-based approach, Appl. Geogr. 87 (2017) 1–13, http://dx.doi.org/10.1016/j.apgeog.2017.07.007.
[15] R. Novelline, Squire's Fundamentals of Radiology, fifth ed., Harvard University Press, ISBN: 0-674-83339-2, 1997.
[16] T.I. Zohdi, Embedded electromagnetically sensitive particle motion in functionalized fluids, Comput. Part. Mech. 1 (2014) 27–45.
[17] O.C. Zienkiewicz, Coupled problems & their numerical solution, in: R.W. Lewis, P. Bettes, E. Hinton (Eds.), Numerical Methods in Coupled Systems, Wiley, Chichester, 1984, pp. 35–58.

[18] O.C. Zienkiewicz, D.K. Paul, A.H.C. Chan, Unconditionally stable staggered solution procedure for soil-pore fluid interaction problems, Internat. J. Numer. Methods Engrg. 26 (1988) 1039–1055.
[19] R.W. Lewis, B.A. Schrefler, L. Simoni, Coupling versus uncoupling in soil consolidation, Int. J. Num. Anal. Metho. Geomech. 15 (1992) 533–548.
[20] R.W. Lewis, B.A. Schrefler, The Finite Element Method in the Static and Dynamic Deformation and Consolidation of Porous Media, second ed., Wiley press, 1998.
[21] B.A. Schrefler, A partitioned solution procedure for geothermal reservoir analysis, Comm. Appl. Num. Meth. 1 (1985) 53–56.
[22] E. Turska, B.A. Schrefler, On consistency stability and convergence of staggered solution procedures, Rend. Mat. Acc. Lincei, Rome 5 (9) (1994) 265–271.
[23] M. Bianco, G. Bilardi, F. Pesavento, G. Pucci, B.A. Schrefler, A frontal solver tuned for fully coupled non-linear hygro-thermo-mechanical problems, Internat. J. Numer. Methods Engrg. 57 (2003) 1801–1818.
[24] X. Wang, B.A. Schrefler, A multifrontal parallel algorithm for coupled thermo-hydro-mechanical analysis of deforming porous media, Internat. J. Numer. Methods Engrg. 43 (1998) 1069–1083.
[25] T.I. Zohdi, A finite element primer for beginners, in: The Basics, Springer International Publishing, 2018.
[26] T.I. Zohdi, The game of drones: rapid agent-based machine-learning models for multi-UAV path planning, Comput. Mech. (2019) http://dx.doi.org/10.1007/s00466-019-01761-9.
[27] T.I. Zohdi, Machine-learning framework for rapid adaptive digital-twin based fire-propagation simulation in complex environments, Comput. Methods Appl. Mech. Eng. (2020) http://dx.doi.org/10.1016/j.cma.2020.112907.
[28] T.I. Zohdi, A digital twin framework for machine learning optimization of aerial fire fighting and pilot safety, Comput. Methods Appl. Mech. Eng. 373 (2021) 113446.
[29] T.I. Zohdi, A digital-twin and machine-learning framework for ventilation system optimization for capturing infectious disease respiratory emissions, Arch. Comput. Methods Eng. 28 (2021) 4317–4329.
[30] T.I. Zohdi, A digital-twin and machine-learning framework for the design of multiobjective agrophotovoltaic solar farms, Comput. Mech. 68 (2021) 357–370.
[31] T.I. Zohdi, A digital-twin and machine-learning framework for precise heat and energy management of data-centers, Comput. Mech. 69 (2022) 1501–1516.
[32] J.H. Holland, Adaptation in Natural & Artificial Systems, Ann Arbor, Mich. University of Michigan Press., 1975.
[33] J.H. Holland, J.H. Miller, Artificial adaptive agents in economic theory (PDF), Amer. Econ. Rev. 81 (2) (1991) 365–371, Archived from the original (PDF) on October 27, 2005.
[34] D.E. Goldberg, Genetic algorithms in search, in: Optimization & Machine Learning, Addison-Wesley, 1989.
[35] L. Davis, Handbook of Genetic Algorithms, Thompson Computer Press, 1991.
[36] C. Onwubiko, Introduction to Engineering Design Optimization, Prentice Hall, 2000.
[37] D.E. Goldberg, K. Deb, Special issue on genetic algorithms, Comput. Methods Appl. Mech. Eng. 186 (2–4) (2000) 121–124.
[38] Y.C. Fung, Elasticity of soft tissues in simple elongation, Am. J. Physiol. 28 (1967) 1532–1544.
[39] Y.C. Fung, Biorheology of soft tissues, Biorheology 10 (1973) 139–155.
[40] Y.C. Fung, On the foundations of biomechanics, ASME J. Appl. Mech. 50 (1983) 1003–1009.
[41] G.A. Holzapfel, Biomechanics of soft tissue, in: J. Lemaitre (Ed.), The Handbook of Materials Behavior Models. Volume III, Multiphysics Behaviors, Chapter 10, Composite Media, Biomaterials, Academic Press, Boston, 2001, pp. 1049–1063.
[42] G.A. Holzapfel, R.W. Ogden, Biomechanical modeling at the molecular, in: Cellular and Tissue Levels, Springer-Verlag, 2009.
[43] J.D. Humphrey, Cardiovascular solid mechanics, in: Cells, Tissues, and Organs, Springer-Verlag, New York, 2002.
[44] J.D. Humphrey, Continuum biomechanics of soft biological tissues, Proc. R. Soc. 459 (2003) 3–46.
[45] A. Rachev, K. Hayashi, Theoretical study of the effects of vascular smooth muscle contraction on strain and stress distributions in arteries, Ann. Biomed. Engng. 27 (1999) 459–468.
[46] D. Klepach, L.C. Lee, J. Wenk, M. Ratcliffe, T.I. Zohdi, J. Navia, G. Kassab, E. Kuhl, J.M. Guccione, Growth and remodeling of the left ventricle: a case study of myocardial infarction and surgical ventricular restoration, Mech. Res. Commun. 42 (2012) 134–141.
[47] D. Ambrosi, G.A. Ateshian, E.M. Arruda, S.C. Cowin, J. Dumais, A. Goriely, G.A. Holzapfel, J.D. Humphrey, R. Kemkemer, E. Kuhl, Olberding. J.E., L.A. Taber, K. Garikipati, Perspectives on biological growth and remodeling, J. Mech. Phys. Solids 59 (2011) 863–883.