International Journal of HIGH PERFORMANCE COMPUTING APPLICATIONS

Research Paper

The International Journal of High Performance Computing Applications
2022, Vol. 36(2) 198–213

DOI: 10.1177/10943420211045939
journals.sagepub.com/home/hpc


Highly scalable numerical simulation of coupled reaction–Diffusion systems with moving interfaces

Mojtaba Barzegari[1] and **Liesbet Geris**[1,2]

Abstract

A combination of reaction–diffusion models with moving-boundary problems yields a system in which the diffusion (spreading and penetration) and reaction (transformation) evolve the system's state and geometry over time. These systems can be used in a wide range of engineering applications. In this study, as an example of such a system, the degradation of metallic materials is investigated. A mathematical model is constructed of the diffusion-reaction processes and the movement of corrosion front of a magnesium block floating in a chemical solution. The corresponding parallelized computational model is implemented using the finite element method, and the weak and strong-scaling behaviors of the model are evaluated to analyze the performance and efficiency of the employed high-performance computing techniques.

Keywords

High-performance computing, reaction-diffusion systems, finite element method, performance analysis, partial differential equations

Introduction

Moving-boundary problems (Crank, 1987) are a subset of the general concept of boundary-value problems which not only require the solution of the underlying partial differential equation (PDE), but also the determination of the boundary of the domain (or sub-domains) as part of the solution. Moving-boundary problems are usually referred to as Stefan problems (Crank, 1987) and can be used to model a plethora of phenomena ranging from phase separation and multiphase flows in materials engineering to bone development and tumor growth in biology. Reaction-diffusion systems are the mathematical models in which the change of state variables occurs via transformation and spreading. These systems are described by a set of parabolic PDEs and can model a large number of different systems in science and engineering, for instance, predator-prey models in biology and chemical components reactions in chemistry (Grindrod, 1996). Combining the reaction–diffusion systems with moving-boundary problems provides a way to study the systems in which the diffusion and reaction lead to the change of domain geometry. Such systems have great importance in various real-world scenarios in chemistry and chemical engineering as well as environmental and life sciences.

In this study, the material degradation phenomenon has been investigated as an example of a reaction–diffusion system with moving boundaries, in which the loss of material due to corrosion leads to movement of the interface of the bulk material and surrounding corrosion environment. More specifically, the degradation of magnesium (Mg) in simulated body fluid has been chosen as a case study. Magnesium has been chosen due to its growing usability as a degradable material in biomedicine, where it is usually used in biodegradable implants for bone tissue engineering and cardiovascular applications (Chen et al., 2014; Zhao et al., 2017). The ultimate application of such a model can be then to study the degradation behavior of resorbable Mg-based biomaterials.

A wide range of different techniques has already been developed to study the moving interfaces in reaction–diffusion problems, which can be grouped into 3 main

[1]Biomechanics Section, Department of Mechanical Engineering, KU Leuven, Leuven, Belgium
[2]Biomechanics Research Unit, GIGA in Silico Medicine, University of Liège, Liège, Belgium

Corresponding author:
Mojtaba Barzegari, Biomechanics Section, Department of Mechanical Engineering, KU Leuven, Celestijnenlaan 300, B-3001 Heverlee, Leuven 3000, Belgium.
Email: mojtaba.barzegari@kuleuven.be

categories: (1) mesh elimination techniques, in which some elements are eliminated to simulate the interface movement (or loss of material in corrosion problems), (2) explicit surface representation, such as the arbitrary Lagrangian-Eulerian (ALE) method, which tracks the interface by moving a Lagrangian mesh inside an Eulerian grid, and (3) implicit surface tracking, in which an implicit criterion is responsible to define the moving interface during the reaction–diffusion process. Related to the aforementioned case study, studies performed by Gao et al. (2018) and Gastaldi et al. (2011) are examples of the first group. Gao et al. (2018) have constructed a simulation of degradation using the mesh elimination technique. Gastaldi et al. (2011) have developed a continuous damage (CD) model by using an explicit solver to study the degradation. The work of Grogan et al. (2014) is an example of the second group as they have developed one of the first models to correlate the mass flux of the metallic ions in the biodegradation interface to the velocity of said interface. This was used to build an ALE model to explicitly track the boundary of the material during degradation. Studies of the third category are based more on mathematical modeling rather than available models in simulation software packages. This approach results in more flexibility and control over the implementation of the computational model. For instance, Wilder et al. (2014) have derived a system of mathematical equations to study galvanic corrosion of metals, taking advantage of the level set method (LSM) to track the corrosion front. Bajger et al. (2016) have used the definition of velocity of the biodegradation interface as the speed of the moving-boundary in LSM, enabling them to track the geometrical changes of the material during degradation. Similarly, Vagbharathi and Gopalakrishnan (2014) have used a combination of LSM and extended finite element method (XFEM), a method to model regions with spatial discontinuities, to study the moving corrosion front in the pitting corrosion process. A very similar approach and formulation has been taken by Duddu (2014) to model localized pitting corrosion. An alternative method for tracking the moving interface is the phase-field method, which has been used in a wide range of relevant studies. A comparison between the behavior of phase-field and LSM formulations for an evolving solid-liquid interface has been performed by Xu et al. (2012), showing that both methods lead to the same results for diffusion-reaction systems. The approach taken in this study was similar to the one from Bajger et al., where LSM was employed to correlate the diffusion and reaction processes to the movement of the solid-solution interface using continuous variables.

Tracking the moving front at the diffusion interface requires high numerical accuracy of the diffusive state variables, which can be achieved using a refined computational grid. This makes the model computationally intensive, and as a consequence, implementing parallelization is an inevitable aspect of simulating such a model. Such an approach enables the model to simulate large-scale systems with a large number of degrees of freedom (DOF) in 3D with higher performance and efficiency in high-performance computing (HPC) environments. In recent years, parallelization of diffusion-reaction systems simulation has been investigated, but the studies are mainly conducted for stochastic (statistical) models. For instance, Chen and Schutter (2017) have developed a parallel stochastic model for large-scale spatial reaction–diffusion simulation, and similarly, Arjunan et al. (2020) have developed a stochastic high-performance simulator for specific biological applications. Also as an example for massively parallel systems, Hallock et al. (2014) have conducted a simulation of reaction–diffusion processes in biology using graphics processing units (GPUs). Although stochastic models have more parallel-friendly algorithms, explaining the underlying process, especially when it involves reaction–diffusion processes of chemistry and biology, is less complex and more universal using mechanistic (deterministic) models, which are based on well-developed mathematical models of continuous systems (Kendall et al., 1999). To the best of authors' knowledge, none of the previous contributions to the topic of reaction–diffusion systems with moving interfaces has employed parallelization techniques to increase the performance and speed of execution of the model without compromising the accuracy of the interface tracking.

In the current study, we developed a mechanistic model of a reaction–diffusion system coupled with a moving interface problem. Improving the accuracy of the interface capturing requires a refined computational mesh, leading to a more computation-intensive simulation. To overcome this challenge and yield more interactable simulations, scalable parallelization techniques were implemented making the model capable of being run on massively parallel systems to reduce the simulation time. The investigated case-study is the material degradation process. The developed model captures the release of metallic ions to the medium, formation of a protective film on the surface of the material, the effect of presented ions in the medium on the thickness of this protection layer, and tracking of the movement of the corrosion front (Figure 1). The interface tracking was performed using an implicit distance function that defined the position of the interface during degradation. This implicit function was obtained by constructing and solving a level set model. It is also worth noting that in a real-world application, such systems require a calibration (also called parameter estimation or inverse problem), in which the model should be simulated hundreds of times. This makes the parallelization even more crucial for these models.

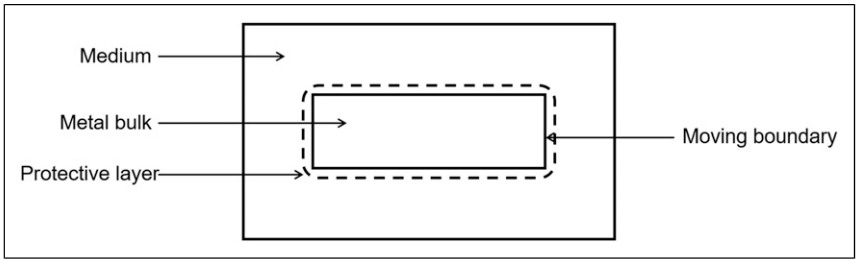

Figure 1: A schematic representation of different components of the developed model for simulation of the degradation process with a moving front.

Background theory and model description

Before elaborating the parallel implementation strategy, the mathematical model is briefly described in this section. The model is constructed based on the chemistry of degradation, starting from the previous work by Bajger et al. (2016), in which the ions can diffuse to the medium and react with each other.

Chemistry of degradation

In metals, degradation occurs through the corrosion process, which usually consists of electrochemical reactions, including anodic and cathodic reactions as well as the formation of side products (Zheng et al., 2014).

For Mg, the corrosion reactions comprise the following steps (Zheng et al., 2014): first, the material is released as metallic ions and free electrons, which causes the volume of the bulk material to be reduced

$$\mathrm{Mg} \rightarrow \mathrm{Mg}^{2+} + 2\mathrm{e}^{-} \tag{1}$$

The free electron reduces water to hydrogen gas and hydroxide ions

$$2\mathrm{H_2O} + 2\mathrm{e}^{-} \rightarrow \mathrm{H_2} + 2\mathrm{OH}^{-} \tag{2}$$

Then, with the combination of the metallic and hydroxide ions, a porous film is formed on the surface, slowing down the degradation rate by protecting the material underneath

$$\mathrm{Mg}^{2+} + 2\mathrm{OH}^{-} \rightarrow \mathrm{Mg(OH)_2} \tag{3}$$

With the presence of some specific ions in the surrounding medium, such as chloride ions in a saline solution, the protective film might be broken partially, which contributes to an increase of the rate of degradation

$$\mathrm{Mg(OH)_2} + 2\mathrm{Cl}^{-} \rightarrow \mathrm{Mg}^{2+} + 2\mathrm{Cl}^{-} + 2\mathrm{OH}^{-} \tag{4}$$

The degradation process of metals is a continuous repetition of the above reactions.

Reaction–diffusion equation

A reaction–diffusion partial differential equation can describe the state of a reaction–diffusion system by tracking the change of the concentration of the different components of the system over time (Grindrod, 1996). The equation is a parabolic PDE and can be expressed as

$$\frac{\partial u}{\partial t} - \nabla \cdot [D \nabla u] = f(u) \tag{5}$$

In which the change of the state variable $u = u(\mathbf{x},t), \mathbf{x} \in \Omega \subset \mathbb{R}^3$ is described as a combination of how it diffuses and how it is produced or eliminated via reactions. The term $f(u)$ is a smooth function that describes the reaction processes. In the example used in this study, the state variable in equation (5) is the concentration of effective chemical components involved in the degradation process, namely magnesium ions and the protective layer, denoted by C_{Mg} and C_{Film}, respectively

$$C_{\mathrm{Mg}} = C_{\mathrm{Mg}}(\mathbf{x},t), \quad C_{\mathrm{Film}} = C_{\mathrm{Film}}(\mathbf{x},t) \quad \mathbf{x} \in \Omega \subset \mathbb{R}^3 \tag{6}$$

Ω is the whole domain of interest, including the bulk material and its surrounding medium. So, by assuming that the reaction rates of equations (3) and (4) are k_1 and k_2, respectively, one can write the change of those state variables according to equations (3) and (4) as

$$\frac{\partial C_{\mathrm{Mg}}}{\partial t} = \nabla \cdot \left(D^{e}_{\mathrm{Mg}} \nabla C_{\mathrm{Mg}} \right) - k_1 C_{\mathrm{Mg}} + k_2 C_{\mathrm{Film}} [\mathrm{Cl}]^2 \tag{7}$$

$$\frac{\partial C_{\mathrm{Film}}}{\partial t} = k_1 C_{\mathrm{Mg}} - k_2 C_{\mathrm{Film}} [\mathrm{Cl}]^2. \tag{8}$$

We assumed that the concentration of the chloride ions is constant (denoted by [Cl] in the equation) and does not diffuse into the protective film. The missing part of the model described by equations (7) and (8) is the effect of the protective film on the reduction of the degradation rate. To this end, we defined a saturation term, $\left(1 - \frac{C_{\mathrm{Film}}}{[\mathrm{Film}]_{\max}}\right)$ for the concentration of Mg ions in the equations. By considering the film's porosity (ϵ), the

maximum concentration of the protective layer can be calculated based on its density ($\rho_{Mg(OH)_2}$)

$$[\text{Film}]_{max} = \rho_{Mg(OH)_2} \cdot (1 - \epsilon) \tag{9}$$

The defined saturation term acts as a function of space that varies between 0 and 1 in each point. By adding this term to the concentration of Mg ions, we can write

$$\frac{\partial C_{Mg}}{\partial t} = \nabla \cdot \left(D^e_{Mg} \nabla C_{Mg}\right) - k_1 C_{Mg}\left(1 - \frac{C_{Film}}{[\text{Film}]_{max}}\right) + k_2 C_{Film}[Cl]^2 \tag{10}$$

$$\frac{\partial C_{Film}}{\partial t} = k_1 C_{Mg}\left(1 - \frac{C_{Film}}{[\text{Film}]_{max}}\right) - k_2 C_{Film}[Cl]^2 \tag{11}$$

Since the film is a porous layer and allows the ions to diffuse through it, the diffusion coefficient in equation (10) is a function of space and not a constant value (which is the reason for being denoted as D^e_{Mg}). We can calculate this effective diffusion function by interpolating two values at any point: (1) $D^e_{Mg} = D_{Mg}$ when $C_{Film} = 0$ and (2) $D^e_{Mg} = \frac{\epsilon}{\tau} D_{Mg}$ when $C_{Film} = [\text{Film}]_{max}$, in which ϵ and τ are the porosity and tortuosity of the protective film, respectively. The interpolation leads to the effective diffusion function

$$D^e_{Mg} = D_{Mg}\left(\left(1 - \frac{C_{Film}}{[\text{Film}]_{max}}\right) + \frac{C_{Film}}{[\text{Film}]_{max}} \frac{\epsilon}{\tau}\right). \tag{12}$$

Level set method

The level set method is a methodology that allows moving interfaces to be described by an implicit function. In other words, the boundaries of domains can be tracked as a function instead of being explicitly defined. In the level set method, a signed distance function, $\phi = \phi(x, y, z, t)$, describes the distance of each point in space to the interface, and the zero iso-contour of this function implies the interface (Ronald Fedkiw, 2002). In the current study, this function was defined in a way that divides the domain into two sub-domains: (1) the bulk material, in which the implicit function is positive ($\phi > 0$), and (2) the medium, in which the function is negative ($\phi < 0$). The interface is defined as the points in space where $\phi = 0$. Figure 2 shows a schematic representation of the solid-medium interface in the current study, in which the interface moves as the material degrades over time.

The level set equation defines this implicit function. The full level set equation can be written as (Ronald Fedkiw, 2002)

$$\frac{\partial \phi}{\partial t} + \overrightarrow{V^E} \cdot \nabla\phi + V^N\left|\nabla\phi\right| = b\kappa\left|\nabla\phi\right| \tag{13}$$

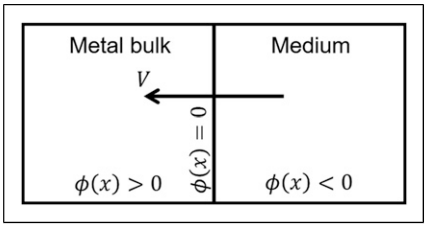

Figure 2: A schematic representation of the implicit function definition in the current study. *V* denotes the shrinkage speed of the interface due to degradation.

In which the terms correspond to temporal changes, external velocity field effect, normal direction motion, and curvature-dependent interface movement, respectively. $\overrightarrow{V^E}$ is the external velocity field, and V^N is the magnitude of the interface velocity along the normal axis. In practical usage, some of the terms are neglected. In this study, perfusion (rotation of the liquid around the bulk sample) is not considered, and the degradation rate does not depend on the curvature of the interface. As a result, by assuming that the interface moves in normal direction only, equation (13) can be simplified to

$$\frac{\partial \phi}{\partial t} + V^N\left|\nabla\phi\right| = 0 \tag{14}$$

where V^N is depicted in Figure 2. The Rankine–Hugoniot equation can be used to calculate the interface velocity in mass transfer problems (Scheiner and Hellmich, 2007)

$$\{\mathbf{J}(x,t) - (c_{sol} - c_{sat})V(x,t)\} \cdot n = 0 \tag{15}$$

in which $\mathbf{J}$ is the mass flux, c_{sol} is the concentration of the material in the bulk part (i.e., its density), and c_{sat} is the concentration at which the material (here, the ions) saturates through the medium. So, for the investigated Mg degradation problem, equation (15) will be

$$D^e_{Mg} \nabla_n C_{Mg} - ([Mg]_{sol} - [Mg]_{sat})V^N = 0. \tag{16}$$

Inserting the obtained velocity of equation (16) into equation (14) and considering the direction of the shrinkage velocity, which is in the opposite direction of the surface normal vector, yields

$$\frac{\partial \phi}{\partial t} - \frac{D^e_{Mg} \nabla_n C_{Mg}}{[Mg]_{sol} - [Mg]_{sat}}\left|\nabla\phi\right| = 0. \tag{17}$$

Equation (17) is the final formulation of the level set equation in the current study, which alongside equations (10) and (11), forms the mathematical model of degradation of Mg with a moving interface. Equation (17) contributes indirectly to the evolution of equations (10) and (11) as it defines the boundary, the zero iso-contour of the ϕ function,

on which the boundary conditions of the equations are applied.

Methodology of model implementation

The developed mathematical model comprises equations (10), (11), and (17), and cannot be solved using analytical techniques. The alternative approach in these scenarios is solving the derived PDEs numerically. In this study, we used a combination of finite element and finite difference methods to solve the aforementioned equations. In the developed numerical model, the PDEs are solved one by one, each of which is a linear equation, so the model implementation follows the principles of solving linear systems. In the following section, only the process to obtain the solution of equation (10) is described in detail, but the other PDEs were solved using the same principle. Although the adopted finite element method is standard, we elaborate on its derivation to clarify the bottlenecks of the later-discussed implementation.

Finite element discretization (bottleneck of the algorithm)

In order to solve equation (10) numerically, we used a finite difference scheme for the temporal term and a finite element formulation for the spatial terms. For simplicity of writing, notations of variables are changed, so C_{Mg} is represented as u (the main unknown state variable to find), C_{Film} is denoted by p, [Cl] is denoted by q, and the saturation term $\left(1-\frac{F}{F_{\max}}\right)$ is denoted by s. By doing this, equation (10) can be written as

$$\frac{\partial u}{\partial t}=\nabla\cdot(D\nabla u)-k_1su+k_2pq^2. \tag{18}$$

To obtain the finite element formulation, the weak form of derived PDE is required. In order to get this, we define a space of test functions and then, multiply each term of the PDE by any arbitrary function as a member of this space. The test function space is

$$\mathcal{V}=\left\{v(\mathbf{x})\middle|\mathbf{x}\in\Omega,v(\mathbf{x})\in\mathcal{H}^1(\Omega),\text{ and }v(\mathbf{x})=0\text{ on }\Gamma\right\} \tag{19}$$

in which the Ω is the domain of interest, Γ is the boundary of Ω, and $\mathcal{H}^1$ denotes the Sobolev space of the domain Ω, which is a space of functions whose derivatives are square-integrable functions in Ω. The solution of the PDE belongs to a trial function space, which is similarly defined as

$$\mathcal{S}_t=\left\{u(\mathbf{x},t)\middle|\mathbf{x}\in\Omega,t>0,u(\mathbf{x},t)\in\mathcal{H}^1(\Omega),\text{ and }\frac{\partial u}{\partial n}=0\text{ on }\Gamma\right\}. \tag{20}$$

Then, we multiply equation (18) to an arbitrary function $v\in\mathcal{V}$

$$\frac{\partial u}{\partial t}v=\nabla\cdot(D\nabla u)v-k_1suv+k_2pq^2v. \tag{21}$$

Integrating over the whole domain yields

$$\int_\Omega\frac{\partial u}{\partial t}vd\omega=\int_\Omega\nabla\cdot(D\nabla u)vd\omega-\int_\Omega k_1suvd\omega+\int_\Omega k_2pq^2vd\omega. \tag{22}$$

The diffusion term can be split using the integration by parts technique

$$\int_\Omega\nabla\cdot(D\nabla u)vd\omega=\int_\Omega\nabla\cdot[v(D\nabla u)]d\omega-\int_\Omega(\nabla v)\cdot(D\nabla u)d\omega \tag{23}$$

in which the second term can be converted to a surface integral on the domain boundary by applying the Green's divergence theory

$$\int_\Omega\nabla\cdot[v(D\nabla u)]d\omega=\int_\Gamma Dv\frac{\partial u}{\partial n}d\gamma. \tag{24}$$

For the temporal term, we use the finite difference method and apply a first-order backward Euler scheme for discretization, which makes it possible to solve the PDE implicitly

$$\frac{\partial u}{\partial t}=\frac{u-u^n}{\Delta t} \tag{25}$$

where u^n denotes the value of the state variable in the previous time step (or initial condition for the first time step). Inserting equations (23)–(25) into equation (22) yields:

$$\begin{aligned}\int_\Omega\frac{u-u^n}{\Delta t}vd\omega=&\int_\Gamma Dv\frac{\partial u}{\partial n}d\gamma-\int_\Omega D\nabla u\cdot\nabla vd\omega\\&-\int_\Omega k_1suvd\omega+\int_\Omega k_2pq^2vd\omega\end{aligned} \tag{26}$$

The surface integral is zero because there is a no-flux boundary condition on the boundary of the computational domain (defined in the trial function space according to equation (20)). By reordering the equation, we get the weak form of equation (18)

$$\begin{aligned}\int_\Omega uvd\omega+\int_\Omega\Delta tD\nabla u\cdot\nabla vd\omega+\int_\Omega\Delta tk_1suvd\omega\\=\int_\Omega u^nvd\omega+\int_\Omega\Delta tk_2pq^2vd\omega\end{aligned} \tag{27}$$

So, the problem is finding a function $u(t)\in\mathcal{S}_t$ such that for all $v\in\mathcal{V}$ equation (27) would be satisfied. By defining a linear functional $(f,\ v)=\int_\Omega fvd\omega$ and encapsulating the

independent concentration terms into $f^n = pq^2$, equation (27) can be simplified as

$$(u,v)[1 + \Delta t k_1 s] + \Delta t(D\nabla u,\nabla v) = (u^n,v) + \Delta t(f^n,v) \quad (28)$$

which can be further converted to the common form of the weak formulation of time-dependent reaction–diffusion PDEs by multiplying to a new coefficient $\alpha = \frac{1}{1+\Delta t k_1 s}$

$$(u,v) + \alpha\Delta t(D\nabla u,\nabla v) = \alpha(u^n,v) + \alpha\Delta t(f^n,v). \quad (29)$$

One can approximate the unknown function u in equation (29) by $u(x) \approx \sum_{i=0}^{N} c_i\psi_i(x)$, where the ψ_i are the basis functions used to discretize the function space, and c_0, …, c_N are the unknown coefficients. The finite element method uses Lagrange polynomials as the basis function and discretizes the computational domain using a new function space υ_h spanned by the basis function $\{\psi_i\}_{i\in\mathcal{I}_s}$, in which $\mathcal{I}_s$ is defined as $\mathcal{I}_s = \{0,\ldots,N\}$, where N denotes the degrees of freedom in the computational mesh. The computational mesh discretizes the space into a finite number of elements, in each of which the ψ_i is non-zero inside the ith element and zero everywhere else. In this study, first-order Lagrange polynomials were used as the basis functions to define the finite element space.

In order to derive a linear system of equations for obtaining the unknown coefficients c_j, we define

$$u = \sum_{j=0}^{N} c_j\psi_j(\mathbf{x}), \quad u^n = \sum_{j=0}^{N} c_j^n\psi_j(\mathbf{x}) \quad (30)$$

as the definition of the unknown function u and its value in the previous time step u^n. We then insert it into equation (29), which yields the following equation for each degree of freedom $i = 0$, …, N, where the test functions are selected as $v = \psi_i$

$$\begin{aligned}\sum_{j=0}^{N}(\psi_i,\psi_j)c_j + \alpha\Delta t\sum_{j=0}^{N}(\nabla\psi_i,D\nabla\psi_j)c_j \\ = \sum_{j=0}^{N}\alpha(\psi_i,\psi_j)c_j^n + \alpha\Delta t(f^n,\psi_i).\end{aligned} \quad (31)$$

Equation (31) is a linear system

$$\sum_j A_{i,j}c_j = b_i \quad (32)$$

with

$$A_{i,j} = (\psi_i,\psi_j) + \alpha\Delta t(\nabla\psi_i,D\nabla\psi_j) \quad (33)$$

$$b_i = \sum_{j=0}^{N}\alpha(\psi_i,\psi_j)c_j^n + \alpha\Delta t(f^n,\psi_i) \quad (34)$$

By solving equation (32) and substituting the obtained c in equation (30), u (C_{Mg} in the example in this study) can be calculated in the current time step. As stated before, the same approach can be applied to equations (11) and (17) to get C_{Film} and ϕ. This procedure is repeated in each time step to compute the values of C_{Mg}, C_{Film}, and ϕ over time.

A common practice to save time for solving equation (32) for a constant time step size is to compute the left-hand side matrix [A in equation (33)] once and compute only the right-hand side vector of the equation at each time iteration. But in this case, although the time step size is fixed, due to the presence of the α coefficient, the matrix changes along the time. The α coefficient is not constant and should be updated in each time step because it depends on the penalization term s [which is a function of the concentration of the film as can be seen by comparing equations (10) and (18)]. In addition to this, the diffusion coefficient is not constant [Equation (12)], making the second term in equation (33) non-constant even in the absence of α coefficient. Consequently, the left-hand side matrix of the equation (32) cannot be computed before the start of the main time loop, and computing it in each time step is an extra but inevitable computational task in comparison to similar efficient and high-performance finite element implementations. This contributes to a slower algorithm for solving the aforementioned PDEs.

Implementation and parallelization

The model was implemented in FreeFEM (Hecht, 2012), which is an open-source PDE solver to facilitate converting the weak formulation [Equation (27)] to a linear system $Ax = b$ [with A from equation (33) and b from equation (34)]. The computational mesh was generated using Netgen (Schöberl, 1997) in the SALOME platform (Ribes and Caremoli, 2007) by a set of linear tetrahedral elements, and all the other preprocessing steps were performed in FreeFEM. The mesh was adaptively refined on the material-medium interface in order to increase the accuracy of the level set model. Postprocessing of the results was carried out using Paraview (Ahrens et al., 2005).

Computing the diffusion solely in the medium domain causes oscillations close to the interface, and to prevent this, the mass lumping feature of FreeFEM was employed. In this technique, the desired mass matrix is handled node-wise and not element-wise. Technically speaking, this means that the state variable is stored in the mesh nodes, and although this is the natural formulation in the finite difference method, it requires artificial modification in the standard finite element formulation (Wendland and Schulz, 2005). The mass lumping feature of FreeFEM applies a quadratic formula at the vertices of elements to make the mass matrix diagonal, which contributes positively to the convergence of the solution.

The main parallelization approach for the current study was domain decomposition, in which the mesh is split into

smaller domains (can be overlapping or non-overlapping), and the global solution of the linear system is achieved by solving the problem on each smaller local mesh. What really matters in this approach is providing virtual boundary conditions to the smaller sub-domains by ghost elements, transferring neighboring sub-domain solutions (Badri et al., 2018). As a result, a high-performance parallelism is feasible by assigning each sub-domain to one processing unit.

In computational science, preconditioning is widely used to enhance the convergence, which means instead of directly working with a linear system $Ax = b$, one can consider the preconditioned system (Daas et al., 2019)

$$M^{-1}Ax = M^{-1}b \tag{35}$$

in which the M^{-1} is the preconditioner. In the current study, we considered this approach for both the domain composition and the solution of the linear system. We opted to use an overlapping Schwarz method for domain decomposition, in which the mesh is first divided into a graph of N non-overlapping meshes using METIS (or ParMETIS) (Karypis and Kumar, 1998). Then, by defining a positive number δ, the overlapping decomposition $\{\mathcal{T}_i^{\delta}\}_{1\leqslant i\leqslant N}$ can be created recursively for each sub-mesh $\{\mathcal{T}_i\}_{1\leqslant i\leqslant N}$ by adding all adjacent elements of $\mathcal{T}_i^{\delta-1}$ to it. Then, the finite element space v_h [Equation (19)] can be mapped to the local space $\{\mathrm{v}_i^{\delta}\}_{1\leqslant i\leqslant N}$ by considering the restrictions $\{R_i\}_{1\leqslant i\leqslant N}$ and a local partition of unity $\{D_i\}_{1\leqslant i\leqslant N}$ such that

$$\sum_{j=1}^{N} R_j^{\top} D_j R_j = I_{n\times n} \tag{36}$$

where I and n denote identity matrix and the global number of unknowns, respectively (Dolean et al., 2015).

In this study, we decomposed the mesh by using the one-level preconditioner Restricted Additive Schwarz (RAS)

$$M_{\mathrm{RAS}}^{-1} = \sum_{i=1}^{N} R_i^{\top} D_i A_i^{-1} R_i \tag{37}$$

in which $\{A_i\}_{1\leqslant i\leqslant N}$ is the local operator of the sub-matrices (Dolean et al., 2015). For this purpose, we took advantage of the HPDDM (high-performance domain decomposition methods) package interface in FreeFEM (Jolivet et al., 2013). The partitioned mesh is shown in Figure 3. The effect of the construction of these local sub-domains on the sparsity pattern of the global matrix is also depicted in Figure 4. The global matrix is a sparse matrix according to equation (33) and the definition of the basis function ψ.

Generally, two categories of methods have been used to solve a large linear system of equations on parallel machines: direct solvers [e.g., Multifrontal Massively Parallel Sparse, MUMPS (Amestoy et al., 2001)] and iterative solvers [e.g., Generalized Minimal Residual Method, GMRES (Saad and Schultz, 1986)]. While direct solvers are quite robust, they suffer from the memory requirement problem on large systems. Inversely, iterative solvers are quite efficient on memory consumption, but similar to other iterative approaches, they are not very reliable in some cases (Saad, 2003). Direct solvers modify the matrix by factorization (e.g., Cholesky decomposition), but an iterative solver does not manipulate the matrix and works solely using basic algebraic operations. However, for an efficient usage of iterative solvers, a proper preconditioner is crucial (Saad, 2003). By evaluating and comparing the performance of the aforementioned methods for the current model, we decided to use an iterative approach using the Krylov subspaces (KSP) method, in which we preconditioned the equation using a proper preconditioner [Equation (35)] and then solved it with an iterative solver.

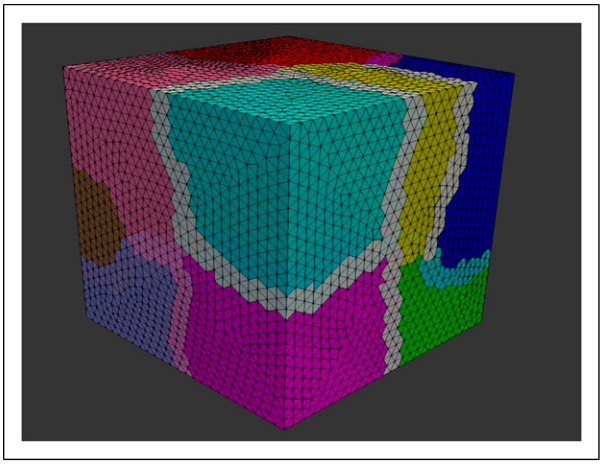

Figure 3: Overlapping domain decomposition in the current study. Each color shows a separate sub-domain, and the narrow lighter bands are the overlapped regions.

Krylov methods have been frequently used by researchers as robust iterative approaches to parallelism (Ipsen and Meyer, 1998). What matters in this regard is ensuring proper scaling of the parallelized algorithm for both the assembling of the matrices and the solution of the linear system of equations. One good solution to this challenge is taking advantage of HPC-ready mathematical libraries to achieve efficient distributed-memory parallelism through the Message Passing Interface (MPI). In the current study, we used the PETSc (Portable, Extensible Toolkit for Scientific Computation) library (Balay et al., 2019), which provides a collection of high-performance preconditioners and solvers for this purpose.

In order to yield the highest performance, a variety of different combinations of KSP types and preconditioners were evaluated, such as Conjugate Gradients (CG) (Hestenes and Stiefel, 1952), Successive Over-Relaxation (SOR) (Habetler and Wachspress, 1961), block Jacobi, and

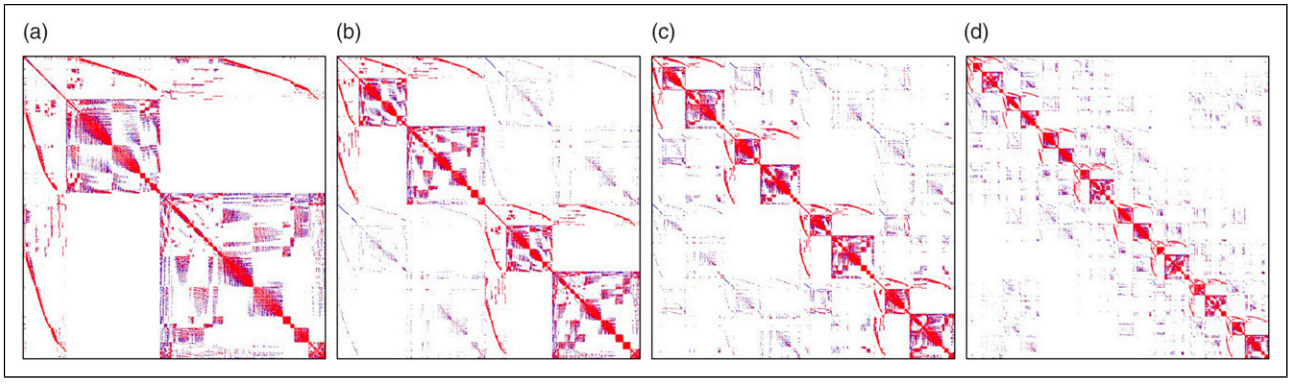

Figure 4: Comparison of the sparsity patterns (highlighting non-zero elements) of the global matrix A for a different number of decomposed domains: (a) 1 domain, (b) 2 sub-domains, (c) 4 sub-domains, and (d) 8 sub-domains.

Algebraic Multigrid (AMG) (McCormick, 1987), to name a few. The performance tests results are presented in the supplementary materials of this paper. The best performance for the reaction–diffusion system model was achieved using the HYPRE BoomerAMG preconditioner (Falgout and Yang, 2002) and the GMRES solver (Saad and Schultz, 1986). This was the combination used for all the performance analysis tests.

Level set issues

As mentioned before, in order to track the interface of the bulk material and the surrounding fluid, an implicit signed distance function is defined as the solution of equation (17). This equation can be solved using the aforementioned finite element discretization, but in a practical implementation, there are usually a couple of problems associated with this PDE.

The first issue is defining D^e_{Mg} and $\nabla_n C_{\mathrm{Mg}}$ on the moving interface. To ensure correct boundary conditions for equation (16), the value of C_{Mg} is set constant on the whole bulk material by using the penalty method. As a result, the implicit interface is not necessarily aligned on the computational mesh. Although this is a beneficial fact for the interface tracking, it inserts the problem of overestimation of C_{Mg} on the nodes close to the interface, which makes it difficult to calculate $\nabla_n C_{\mathrm{Mg}}$ on these nodes correctly. The same problem exists for calculating D^e_{Mg}. To overcome this issue, the values of C_{Mg} and D^e_{Mg} are calculated at the distance h from the interface in the normal direction (towards the medium), where h is the edge size of the smallest element of the computational mesh.

The next issue is a well-known problem of the level set method: If the velocity of the interface is not constant [as in Equation (13)], the level set function ϕ may become distorted by having too flat or too steep gradients close to the moving front. This could cause unwanted movements of the interface. The problem becomes even worse when the distance function is advected. A solution to this issue is re-initializing the distance function in each time step (re-distancing), but this operation requires solving a new PDE. From numerical investigations, it has been observed that this operation inserts new errors in the numerical computation of the level set equation (Russo and Smereka, 2000). This can be resolved by improving the method of reconstruction of the distance function (Russo and Smereka, 2000).

However, re-initialization results in another issue on a massively parallel implementation: as the mesh is partitioned into smaller sub-meshes, it is not feasible anymore to evaluate the distance to the interface globally on each sub-domain. As a result, the inverse process of domain decomposition should be taken to assemble the mesh again. This can be done by the restriction matrix and the partition of unity [defined in Equations (36) and (37)], but it is rather a very inefficient procedure regarding the parallelization of the simulation and results in a long execution time in each time step.

In the current study, the distance function ϕ was initialized only once at the beginning of the simulation. The re-initialization process was unnecessary in this case because according to equation (17), the distance function is advected only in the regions where there is a gradient of the concentration of Mg ions, which means that advection is applied only on the regions close to the interface in the medium. This prevented the whole distance function of being distorted, and as a result, it was not required to re-initialize it in each time step. This also removed the need for inverting the decomposition process.

Simulation setup

In order to verify the performance of the developed model, a degradation experiment was reconstructed in-silico, in which the degradation of a block of Mg (with the size of 13 mm × 13 mm × 4 mm) was investigated in a simulated body fluid solution. All the experimental parameter data

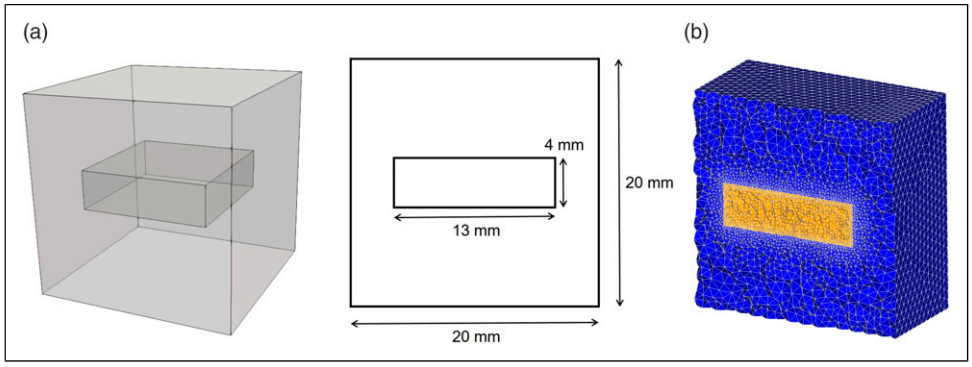

Figure 5: Representation of the experimental setup simulated to perform numerical validation of the developed model and evaluate parallel performance. (a) A cuboid of Mg (with the size of 13 *mm* × 13 *mm* × 4 *mm*) is floating inside a simulated body fluid solution to investigate the degradation process, and (b) a cross-section of the computational mesh, refined on the metal-medium interface to increase the interface capturing accuracy.

(used to setup the simulation), as well as the degradation rates (used to calibrate and validate the numerical model) were extracted from Mei et al. (2019).

As can be seen in equations (1) and (2), each mole removed from the Mg block corresponds to one mole of the produced hydrogen. As a result, instead of a direct measurement of mass loss, one can collect and measure the amount of produced hydrogen to monitor the degradation rate. This is a common way of reporting degradation in this type of studies (Abidin et al., 2013). In order to get this quantity out of the developed model, we used the level set model output. The total mass loss of Mg at each desired time can be calculated based on the movement of the corrosion front

$$\mathrm{Mg}_{\mathrm{lost}} = \int_{\Omega_+(t)} \mathrm{Mg}_{\mathrm{solid}} \mathrm{d}V - \int_{\Omega_+(0)} \mathrm{Mg}_{\mathrm{solid}} \mathrm{d}V \tag{38}$$

where $\Omega_+(t) = \{\mathbf{x}: \phi(\mathbf{x}, t) \geq 0\}$. It is worth noting that this integration should be performed by ignoring the ghost elements generated in the mesh partitioning process, otherwise the calculated material loss will be higher than the real value. Then, the amount of formed hydrogen gas can be calculated based on the ideal gas law

$$H_f = \frac{\mathrm{Mg}_{\mathrm{lost}}}{\mathrm{Mg}_{\mathrm{mol}}} \frac{RT}{PA} \tag{39}$$

in which R is the universal gas constant, P is the pressure, T is the solution temperature, A is the exposed corrosion surface area (which can be computed using the level set function), and $\mathrm{Mg}_{\mathrm{mol}}$ is the molar mass of Mg. Plotting a comparison of the predicted and experimentally obtained values of hydrogen can show the overall validity of the mathematical model because both the diffusion-reaction equations and the level set equation contribute to the prediction made by the computational model.

The geometry of the simulation experiment is depicted in Figure 5. Based on this geometry, an Eulerian computational mesh was constructed by generating tetrahedral elements on the whole domain, including the Mg block and the medium. This resulted in 830,808 elements with a total of 143,719 DOFs for each PDE [equations (10), (11), and (17)], which indicates the size of matrix A in equation (33). Model parameters and material properties were obtained from Bajger et al. (2016). The diffusion coefficient of Mg was calculated using an inverse problem setup in which a Bayesian optimization process (Mockus, 1989) was used to run the simulation code multiple times and minimize the difference of the model output and the experimental data reported by Mei et al. (2019). A time step convergence study was performed to measure the sensitivity of the model to the time stepping parameter, and based on the results, the time step value was set to 0.025 h.

Performance analysis

To investigate the performance and scaling behavior of the implemented parallel code, we conducted a set of weak-scaling and strong-scaling tests on the computational model. To do this, the time required to solve each PDE in each time step was measured in a simulation. This acted as a rough estimation of the time required in each time step because it ignores all the other factors contributing to speedup results such as communication costs, load imbalance, limited memory bandwidth, and parallelization-caused overhead.

Weak-scaling was evaluated by dividing the computational domain into smaller sub-domains (each of which was $\frac{1}{16}$ of the whole domain, Figure 6) and conducting simulation experiments with 1, 2, 4, and 8 computational cores in a way that the number of processors corresponded to the number of employed sub-domains. In Figure 6 the upper row shows different domains as an accumulation of the smaller

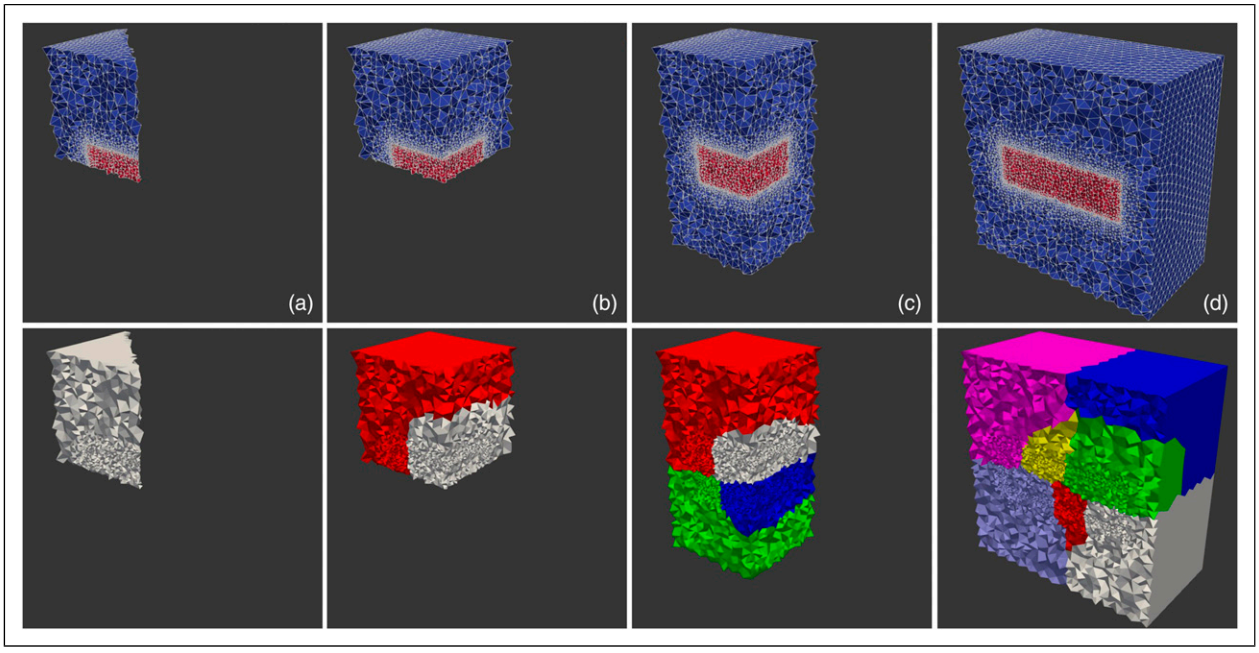

Figure 6: Models used for weak-scaling, in which the number of elements was doubled each time while doubling the number of computational cores. Upper row: actual computational domain in which colors show the medium (blue) and the material block (red). Lower row: domain decomposition for parallelization, colors show different decomposed mesh parts (distributed to different MPI processing units). Each column corresponds to a different simulation with (a) 1 MPI unit, (b) 2 MPI units, (c) 4 MPI units, and (d) 8 MPI units.

divisions, and the lower row shows the corresponding domain decomposition for parallel computing by depicting each processing unit in a different color. In fact, it demonstrates the concept of increasing the number of MPI processing units as we increase the size of the problem.

After calculating the speedup of each test (by comparing the differences in execution time), we can use Gustafson's law (Gustafson, 1988) to calculate the sequential and parallelizable portion of computation in the current implementation in weak-scaling evaluation

$$\text{Speedup} = f + (1 - f) \times N \tag{40}$$

where N is the total number of computational cores, f is the fraction of operations in the computation that are sequential, and as a result, $1 - f$ is the fraction of the execution time spent on the parallelizable part.

The strong-scaling evaluation was performed using the entire domain. The evaluation was done using 1, 8, 16, 40, 60, 90, 200, and 300 MPI cores. In strong-scaling, Amdahl's law (Amdahl, 1967) is used to calculate the portion of the algorithm that runs in parallel

$$\text{Speedup} = \frac{1}{f + \frac{1-f}{N}} \tag{41}$$

in which the parameters are the same as equation (40).

Compute environment

Simulations were conducted on the VSC (Flemish Supercomputer Center) supercomputer with the availability of Intel CPUs in three different micro-architectures: Ivy Bridge, Haswell, and Skylake. Due to a better performance, the strong and weak-scaling measurements were solely performed on the Skylake nodes. On this supercomputer, we made use of 9 nodes, 36 cores each, with 1.7 TB of the total memory, each node holding 2 Intel Xeon Gold 6132 CPUs with a base clock speed of 2.6 GHz. The supercomputer uses CentOS 7.6.1810 with kernel version 3.10.0. For interprocess communication, Intel's MPI implementation 2018 was used.

Results

Numerical simulation results

The performed numerical simulation produces the output of three main quantities: the concentration of the Mg ions in the medium [as the solution of equation (10)], the concentration of the protective film [as the solution of equation (11)], and the level set function values at each element [as the solution of equation (17)]. In addition to this, a quantitative prediction of the mass loss is also generated according to equations (38) and (39).

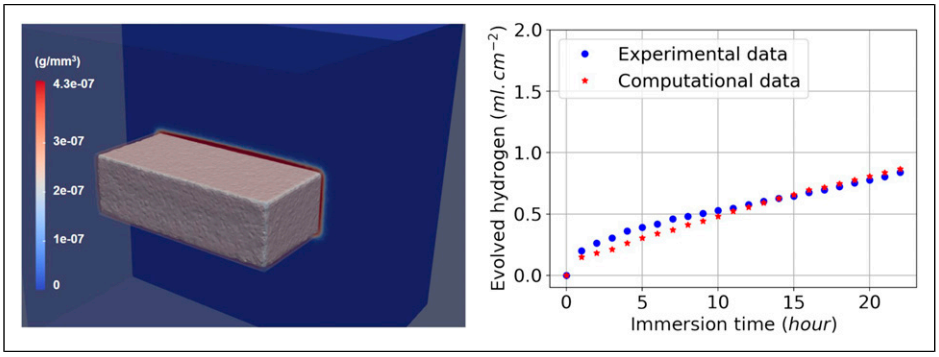

Figure 7: Numerical simulation result. Left: formation of a protective layer on the surface of the Mg block (red region). Right: comparison of the produced hydrogen (a surrogate for the material loss) in the computational model and the experimental data, which is a validation of the full model as both the reaction–diffusion equations and the level set equation are involved in the computation of this quantity.

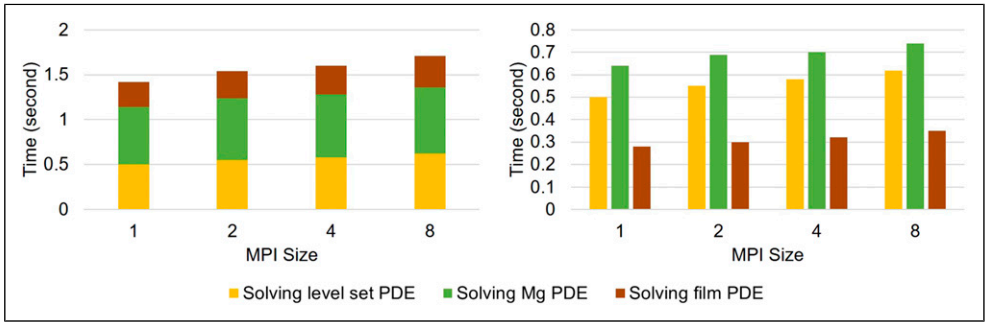

Figure 8: Weak-scaling test result. Results are broken down into contributions for each PDE, which are plotted cumulatively and separately in the left and right plot, respectively.

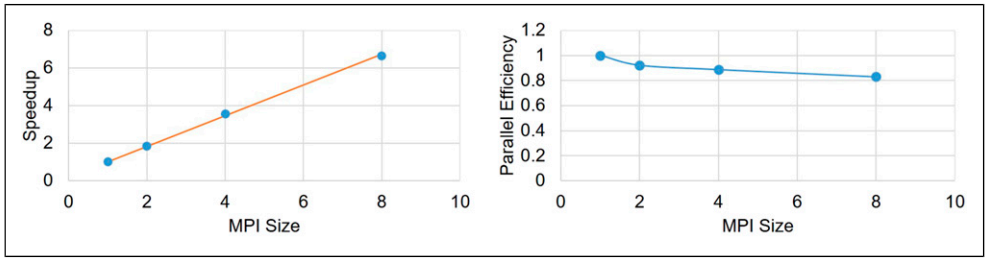

Figure 9: Speedup and parallel efficiency of the weak-scaling experiment. The orange line in the left plot shows the fitted curve based on the Gustafson equation.

In order to have quantitative predictions, the coefficients of equations (10) and (11) (diffusion rates and reaction rates) should be calibrated using an inverse problem. Figure 7 shows the results produced by the computational model after this parameter estimation stage. A narrow layer of the protective film is formed on the surface of the Mg block, and the volume of produced hydrogen gas is compared with values obtained from experiments. Additionally, by plotting the zero iso-contour of the level set function, we can obtain the shape of the material block as it degrades during the degradation process (i.e., tracking the moving corrosion front). This is depicted by the gray surface in Figure 7.

Weak and strong-scaling results

Weak-scaling results are plotted in Figure 8, in which the execution time of each time step is broken down into the time spent on each PDE. The results show good scalability of the parallel implementation.

Speedup and parallel efficiency of the weak-scaling experiment is plotted in Figure 9. By fitting a curve

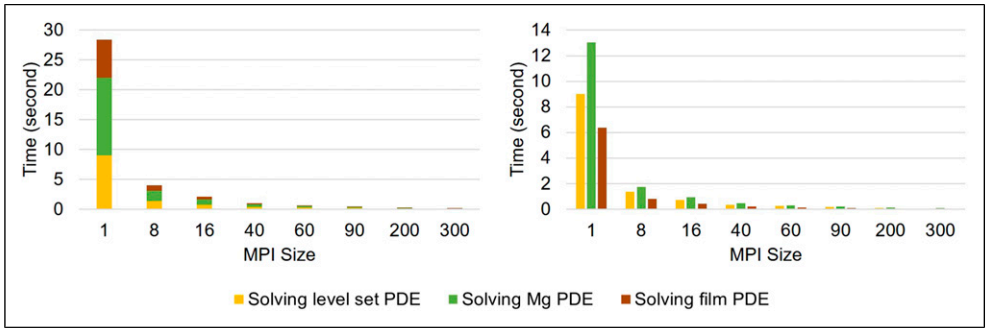

Figure 10: Strong-scaling test result. Results are broken down into contributions for each PDE, which are plotted cumulatively and separately in the left and right plot, respectively.

Table 1: Strong-scaling test result, presented by the execution time of each PDE in simulations with a different number of employed MPI cores.

MPI size		1	8	16	40	60	90	200	300
Solution time of each time step (s)	LS PDE	9	1.39	0.75	0.36	0.26	0.19	0.11	0.07
	Mg PDE	13.04	1.76	0.94	0.46	0.31	0.22	0.12	0.09
	Film PDE	6.38	0.84	0.45	0.21	0.14	0.09	0.05	0.04
Total time (s)		28.42	3.99	2.14	1.03	0.71	0.5	0.28	0.2

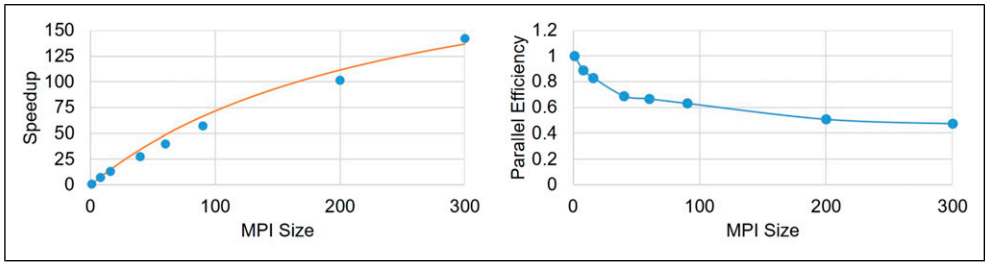

Figure 11: Speedup and parallel efficiency of the strong-scaling experiment. The orange line in the left plot is the fitted equation based on the Amdahl rule.

based on the Gustafson equation [equation (40)] on the obtained results (Figure 9), the sequential proportion of the current implementation was calculated to be 18%, which means that 82% of the code can be parallelized, which is a proper but not an ideal scalability.

The strong-scaling results are plotted in Figure 10, which shows a better scalability in comparison to the weak-scaling test. For a better representation, exact measured values are presented in Table 1.

Similar to weak-scaling results, Figure 11 demonstrates the speedup and parallel efficiency of the developed code for strong-scaling evaluation. From the results, it is obvious that increasing the number of cores leads to a better performance but a lower efficiency. By fitting Amdahl's equation [equation (41)] on the obtained speedup results (Figure 11), f was obtained as 0.01, which means in strong-scaling terms that 99% of the code can run in parallel.

Discussion

In this investigation, the derivation and implementation of a reaction–diffusion model with moving boundaries were presented. Such an approach finds application in many scientific and engineering problems. The target application in the current work was the degradation of a bulk metal cuboid in a liquid environment, specifically Mg in an aqueous ion solution as a representative for temporary medical devices. The simulations were based on the corrosion of Mg metal to Mg ions to form a film of Mg hydroxide that partially protects the metal block from further degradation except where this film is impacted by reaction with other ions in the environment (such as chloride ion). The reactive moving-boundary problem was cast in the form of equations in which the change of the concentrations of the different chemical components is represented by parabolic PDEs. The coupled equations depend on several

kinetic constants that have been calibrated from experiments. The moving interface between the metal bulk and the liquid phase was described by an implicit function using the level set method. The derivation led to equations that require the use of numerical techniques for which a combination of finite difference and finite element methods was implemented. As the required high accuracy on the moving interface results in an increase in computation time, parallelization was crucial for the computational model to decrease the execution time of the simulations. The results of the total execution time in each time step (Table 1) clearly indicate that without the parallelization, the simulation of the model is slow and as a result, less interactable for real-world simulation analyses. Considering the properly employed parallelization, the computational time has been decreased noticeably for the investigated case-study.

The output of the conducted numerical simulation demonstrates that the developed mathematical model is capable of capturing the degradation interface movement and of modeling of the underlying chemical phenomena. The predicted mass loss is in line with the experimental results, and the simulated corrosion behavior is as expected for such a system. It is worth noting that the chosen system is highly idealized as a model for medical devices. A more realistic chemical environment would contain many more species that play a role in the formation of either soluble ions or the protective film. Moreover, in real-world scenarios, corrosion occurs in a more complex way than the simplified one described in this paper, which will have a significant influence on the local concentration of ions in the regions close to the solid surface. Nevertheless, the developed framework is capable of capturing these physical and chemical phenomena in the future by simply adding the appropriate terms to the base PDEs without any major changes in the computational model. Furthermore, although it requires some changes to the parallelization approach, the addition of the fluid flow around the block is feasible by adding convective terms to form a reaction–diffusion-convection system. Such a system can be used to model relevant systems such as experimental bioreactor setups in biology and medical sciences.

The parallel algorithm was implemented using a domain decomposition method. Standard domain decomposition preconditioners, such as restricted additive Schwarz, are widely used for parallel implementation of computational models. In a parallel implementation, such preconditioners bring the benefit of relatively low communication costs (Daas et al., 2019). Beside this, the formed linear system of equations in each partition of the mesh was solved using Krylov methods by taking advantage of the highly-efficient preconditioners and iterative solvers of the PETSc library. According to the obtained results, the employed parallelization approach of the current study yields reasonable scaling with respect to the available computational resources (or the number of sub-domains). Out of multiple evaluations, the best performance was achieved using the preconditioner/solver combination of HYPRE/GMRES, which is in agreement with findings in more specific studies in this regard (Ghai et al., 2018).

To evaluate the scaling performance of the implemented parallelism, a set of weak and strong-scaling tests was conducted. In weak-scaling, the main approach is changing the problem size proportional to the change in the available computing resources. In an ideal parallelization, we expect that the speedup remains the same for all the setups because we provide double resources as we double the size of the problem. In strong-scaling, the size of the problem remains constant, but the number of computing units increases. So, in an ideal case, we should observe a double speedup as the number of computing units doubles. By fitting Gustafson's and Amdahl's laws on the scaling test results (Figures 9 and 11), the maximum parallelizable portion of the code was calculated to be 82% and 99% for the weak-scaling and strong-scaling tests, respectively. This is a reasonable theoretical scaling for both cases.

The obtained scaling behavior is similar to other conducted studies for diffusion or diffusion-convection systems (Hassan and El-Shenawee, 2011; Rettinger et al., 2017), in which the efficiency of the parallelization decreases with increasing the number of available computational resources. The reason behind this behavior in the current model lies in the mesh partitioning process. Indeed, the mesh is partitioned into semi-equal partitions, each of which has the same number of elements, but the main computation is only carried out on the nodes located outside the degrading material block (i.e., in the medium). In other words, the computational resources assigned to the nodes inside the material bulk do not contribute significantly to the simulation. This limitation can be prevented by modifying the mesh generation process in a way that a lower number of elements be generated inside the material block, but doing this requires remeshing of the interior region as the moving interface approaches it, which imposes even more complexity to the algorithm due to the partitioned mesh. Another bottleneck of the current model, as discussed before, routed in the non-constant right-hand matrix of the linear system [equation (32)], which requires computing the A matrix [equation (33)] in each time step and leads to a slower execution time.

One important point in this regard is that the way that the results are interpreted does not necessarily imply the true scaling behavior of the system. Indeed, it is more like a surrogate model of the system performance. The correct methodology for obtaining true scaling factors is rather starting from an analysis of the code and time used in each routine for a non-parallel run. Then, based on the fraction of routines that are possible to execute in parallel, one can get a theoretical limit for the speedup. This will be reduced by

practical limitations such as load balancing and communication costs of the network. Since it is a theoretical limit, it is not fully correct to ignore those extra parts and use the execution time to invert the relation to predict the fraction of the code that is parallel. However, for a complex computational model like the one that was developed in the current study, doing such a measurement of each routine is very difficult due to the complexity of the orchestrated libraries and tools. As a result, we were limited to use the roughly approximated speedup limit to evaluate the scaling of the constructed model. Regarding the scalability results, it is worth mentioning that although having studies with thousands of MPI ranks is more common in this field, due to the limitation we faced in accessing computational resources, the maximum number of employed cores were limited to 300. The goal of the current study was to demonstrate the scalability of the developed model on massively parallel systems, and the behavior of the model in moving from 90 cores to 300 shows the consistency in the performed performance analysis. As a result, we expect to see the same scalability behavior for problems in a larger scale with a higher number of employed computing nodes.

Conclusion

In this work, a mathematical model of a reaction–diffusion system with a moving front was constructed, and the corresponding computational model was implemented using the finite element method. In order to correlate the diffusion phenomenon to the moving-boundary position, high numerical accuracy is necessary at the diffusion interface, which requires a finer discretization of space near the moving front. This leads to an expensive computational model, which makes employing HPC techniques crucial in order to improve the simulation execution time. To this end, a high-performance domain decomposition approach was employed to partition the mesh and distribute the workload to available computing resources. Additionally, an efficient preconditioner/solver combination for reaction–diffusion PDEs was used to optimize the model to be used for the high-performance simulation of large-scale systems in which the movement of system boundaries is controlled by reaction–diffusion phenomena.

The investigated problem was the degradation of a magnesium block inside a solution, in which the surface of the block moves due to the reaction–diffusion phenomena in the metal-medium interface. The implemented model showed a good agreement with the experimental data in terms of the degradation rate and chemical reactions, and the parallel efficiency and linear scalability were appropriate in performance evaluation tests. For the next stage of the study, it could be interesting to evaluate the model and its performance on a much larger system and tune the resources and memory usage by testing different preconditioners and solvers.

Data accessibility statement

The developed source codes and data that support the findings of this study are openly available in BioDeg repository at https://github.com/mbarzegary/BioDeg.

Acknowledgments

The computational resources and services used in this work were provided by the VSC (Flemish Supercomputer Center), funded by the Research Foundation – Flanders (FWO) and the Flemish Government – department EWI.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This research is financially supported by the Prosperos project, funded by the Interreg VA Flanders – The Netherlands program, CCI grant no. 2014TC16RFCB046 and by the Fund for Scientific Research Flanders (FWO), grant G085018N. LG acknowledges support from the European Research Council under the European Union's Horizon 2020 research and innovation program, ERC CoG 772418.

ORCID iD

Mojtaba Barzegari https://orcid.org/0000-0002-1456-0610

Supplementary Material

Supplementary material for this article is available online.

References

Abidin NIZ, Rolfe B, Owen H, et al. (2013) The in vivo and in vitro corrosion of high-purity magnesium and magnesium alloys WZ21 and AZ91. *Corrosion Science* 75: 354–366. DOI: 10.1016/j.corsci.2013.06.019.

Ahrens J, Geveci B and Law C (2005) Paraview: An end-user tool for large data visualization. In: *The Visualization Handbook* 717–731. DOI: 10.1016/B978-012387582-2/50038-1.

Amdahl GM (1967) Validity of the single processor approach to achieving large scale computing capabilities (Spring). In: Proceedings of the April 18-20, 1967, Spring Joint Computer Conference on—AFIPS' 67, Atlantic City, New Jersey, 18–20 April 1967, ACM Press. DOI: 10.1145/1465482.1465560.

Amestoy PR, Duff IS, Koster J, et al. (2001) A fully asynchronous multifrontal solver using distributed dynamic scheduling. *SIAM Journal on Matrix Analysis and Applications* 23(1): 15–41. DOI: 10.1137/S0895479899358194.

Arjunan SN, Miyauchi A, Iwamoto K, et al. (2020) pSpatiocyte: a high-performance simulator for intracellular reaction-diffusion

systems. *BMC Bioinformatics* 21(1): 33. DOI: 10.1186/s12859-019-3338-8.

Badri M, Jolivet P, Rousseau B, et al. (2018) High performance computation of radiative transfer equation using the finite element method. *Journal of Computational Physics* 360: 74–92. DOI: 10.1016/j.jcp.2018.01.027.

Bajger P, Ashbourn JMA, Manhas V, et al. (2016) Mathematical modelling of the degradation behaviour of biodegradable metals. *Biomechanics and Modeling in Mechanobiology* 16(1): 227–238. DOI: 10.1007/s10237-016-0812-3.

Balay S, Abhyankar S, Adams MF, et al. (2019) PETSc Web Page. Available at: https://www.mcs.anl.gov/petsc (Accessed November 2020).

Chen W and Schutter ED (2017) Parallel steps: Large scale stochastic spatial reaction-diffusion simulation with high performance computers. *Frontiers in Neuroinformatics* 11: 13. DOI: 10.3389/fninf.2017.00013.

Chen Y, Xu Z, Smith C, et al. (2014) Recent advances on the development of magnesium alloys for biodegradable implants. *Acta Biomaterialia* 10(11): 4561–4573. DOI: 10.1016/j.actbio.2014.07.005.

Crank J (1987) *Free and Moving Boundary Problems*. Oxford, UK: OUP Oxford.

Daas HA, Grigori L, Jolivet P, et al. (2019) A multilevel schwarz preconditioner based on a hierarchy of robust coarse spaces. *SIAM Journal on Scientific Computing* 43(3): A1907–A1928.

Dolean V, Jolivet P and Nataf F (2015) *An Introduction to Domain Decomposition Methods*. Philadelphia, PA: Society for Industrial and Applied Mathematics. DOI: 10.1137/1.9781611974065.

Duddu R (2014) Numerical modeling of corrosion pit propagation using the combined extended finite element and level set method. *Computational Mechanics* 54(3): 613–627. DOI: 10.1007/s00466-014-1010-8.

Falgout RD and Yang UM (2002) hypre: A library of high performance preconditioners. In: *Lecture Notes in Computer Science*. Berlin, Heidelberg: Springer, pp. 632–641. DOI: 10.1007/3-540-47789-666.

Gao Y, Wang L, Gu X, et al. (2018) A quantitative study on magnesium alloy stent biodegradation. *Journal of Biomechanics* 74: 98–105. DOI: 10.1016/j.jbiomech.2018.04.027.

Gastaldi D, Sassi V, Petrini L, et al. (2011) Continuum damage model for bioresorbable magnesium alloy devices—application to coronary stents. *Journal of the Mechanical Behavior of Biomedical Materials* 4(3): 352–365. DOI: 10.1016/j.jmbbm.2010.11.003.

Ghai A, Lu C and Jiao X (2018) A comparison of preconditioned krylov subspace methods for large-scale nonsymmetric linear systems. *Numerical Linear Algebra with Applications* 26(1): e2215. DOI: 10.1002/nla.2215.

Grindrod P (1996) *The Theory and Applications of Reaction-Diffusion Equations : Patterns and Waves*. Oxford, UK: Oxford University Press.

Grogan J, Leen S and McHugh P (2014) A physical corrosion model for bioabsorbable metal stents. *Acta Biomaterialia* 10(5): 2313–2322. DOI: 10.1016/j.actbio.2013.12.059.

Gustafson JL (1988) Reevaluating amdahl's law. *Communications of the ACM* 31(5): 532–533. DOI: 10.1145/42411.42415.

Habetler GJ and Wachspress EL (1961) Symmetric successive overrelaxation in solving diffusion difference equations. *Mathematics of Computation* 15(76): 356–362.

Hallock MJ, Stone JE, Roberts E, et al. (2014) Simulation of reaction diffusion processes over biologically relevant size and time scales using multi-GPU workstations. *Parallel Computing* 40(5–6): 86–99. DOI: 10.1016/j.parco.2014.03.009.

Hassan AM and El-Shenawee M (2011) Parallel implementation of the diffusion–drift algorithm for modeling the electrophysiological activity of breast tumors. *Journal of Parallel and Distributed Computing* 71(7): 1011–1023. DOI: 10.1016/j.jpdc.2011.04.004.

Hecht F (2012) New development in freefem++. *Journal of Numerical Mathematics* 20(3–4): 251–265. Available at: https://freefem.org/ (Accessed November 2020). DOI: 10.1515/jnum-2012-0013.

Hestenes MR and Stiefel E (1952) Methods of conjugate gradients for solving linear systems. *Journal of Research of the National Bureau of Standards* 49(6): 409–436.

Ipsen ICF and Meyer CD (1998) The idea behind krylov methods. *The American Mathematical Monthly* 105(10): 889–899. DOI: 10.1080/00029890.1998.12004985.

Jolivet P, Hecht F, Nataf F, et al. (2013) Scalable domain decomposition preconditioners for heterogeneous elliptic problems. In: Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis, SC '13, Denver, Colorado, 17–21 November 2013, New York, USA: Association for Computing Machinery. DOI: 10.1145/2503210.2503212.

Karypis G and Kumar V (1998) A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on Scientific Computing* 20(1): 359–392.

Kendall BE, Briggs CJ, Murdoch WW, et al. (1999) Why do populations cycle? a synthesis of statistical and mechanistic modeling approaches. *Ecology* 80(6): 1789–1805. DOI: 10.1890/0012-9658(1999)080.

McCormick SF (1987) *Multigrid Methods*. Philadelphia, USA: SIAM.

Mei D, Lamaka SV, Gonzalez J, et al. (2019) The role of individual components of simulated body fluid on the corrosion behavior of commercially pure mg. *Corrosion Science* 147: 81–93. DOI: 10.1016/j.corsci.2018.11.011.

Mockus J (1989) *Bayesian Approach to Global Optimization*. Netherlands: Springer. DOI: 10.1007/978-94-009-0909-0.

Rettinger C, Godenschwager C, Eibl S, et al. (2017) Fully resolved simulations of dune formation in riverbeds. In: Kunkel JM, Yokota R, Balaji P, et al. (eds.), *High*

Performance Computing. Cham: Springer International Publishing, pp. 3–21.

Ribes A and Caremoli C (2007) Salome platform component model for numerical simulation. In: 31st Annual International Computer Software and Applications Conference—COMPSAC 2007, Beijing, China, 24–27 July 2007, Vol. 2. DOI: 10.1109/compsac.2007.185.

Ronald Fedkiw SO (2002) *Level Set Methods and Dynamic Implicit Surfaces*. New York: Springer.

Russo G and Smereka P (2000) A remark on computing distance functions. *Journal of Computational Physics* 163(1): 51–67. DOI: 10.1006/jcph.2000.6553.

Saad Y (2003) *Iterative methods for sparse linear systems*. Philadelphia, USA: Society for Industrial and Applied Mathematics. DOI: 10.1137/1.9780898718003.

Saad Y and Schultz MH (1986) Gmres: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM Journal on Scientific and Statistical Computing* 7(3): 856–869. DOI: 10.1137/0907058.

Schöberl J (1997) NETGEN an advancing front 2d/3d-mesh generator based on abstract rules. *Computing and Visualization in Science* 1(1): 41–52. DOI: 10.1007/s007910050004.

Scheiner S and Hellmich C (2007) Stable pitting corrosion of stainless steel as diffusion-controlled dissolution process with a sharp moving electrode boundary. *Corrosion Science* 49(2): 319–346. DOI: 10.1016/j.corsci.2006.03.019.

Vagbharathi AS and Gopalakrishnan S (2014) An extended finite-element model coupled with level set method for analysis of growth of corrosion pits in metallic structures. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* 470(2168): 20140001. DOI: 10.1098/rspa.2014.0001.

Wendland E and Schulz H (2005) Numerical experiments on mass lumping for the advection-diffusion equation. *Pesquisa e Tecnologia Minerva* 2: 227–233.

Wilder JW, Clemons C, Golovaty D, et al. (2014) An adaptive level set approach for modeling damage due to galvanic corrosion. *Journal of Engineering Mathematics* 91(1): 121–142. DOI: 10.1007/s10665-014-9732-3.

Xu Z, Huang H, Li X, et al. (2012) Phase field and level set methods for modeling solute precipitation and/or dissolution. *Computer Physics Communications* 183(1): 15–19. DOI: 10.1016/j.cpc.2011.08.005.

Zhao D, Witte F, Lu F, et al. (2017) Current status on clinical applications of magnesium-based orthopaedic implants: A review from clinical translational perspective. *Biomaterials* 112: 287–302. DOI: 10.1016/j.biomaterials.2016.10.017.

Zheng Y, Gu X and Witte F (2014) Biodegradable metals. *Materials Science and Engineering: R: Reports* 77: 1–34. DOI: 10.1016/j.mser.2014.01.001.

Author biographies

Mojtaba Barzegari is a Ph.D. researcher at KU Leuven in Belgium, in the field of computational biomedical engineering. His work is mainly focused on developing mathematical models and high-performance numerical simulations of tissue engineering systems. His research interests include scientific computing, computational tissue engineering, machine learning, and high-performance computing. He has published several peer-reviewed journal papers, conference papers, and book chapters.

Liesbet Geris is Collen-Francqui Research Professor in biomechanics and computational tissue engineering at the University of Liège and KU Leuven in Belgium. Her research focuses on the multi-scale and multi-physics modeling of biological processes. Together with her team and their clinical and industrial collaborators, she uses these models to investigate the etiology of non-healing fractures, to design in silico potential cell-based treatment strategies, and to optimize manufacturing processes of these tissue engineering constructs She has received 2 prestigious ERC grants to finance her research and has received a number of young investigator and research awards.