AIMS Mathematics

https://www.aimspress.com/journal/Math

AIMS Mathematics, 9(12): 35401–35431.
DOI:10.3934/math.20241682
Received: 15 March 2023
Revised: 20 November 2023
Accepted: 17 January 2024
Published: 19 December 2024

*Research article*

# Moving mesh simulations of pitting corrosion

**Abu Naser Sarker**[1], **Ronald D. Haynes**[2,*] **and Michael D. Robertson**[3]

[1] Scientific Computing Program, Memorial University, St. John's, NL, A1C 5S7, Canada
[2] Department of Mathematics and Statistics, Memorial University, St. John's, NL, A1C 5S7, Canada
[3] Department of Physics, Acadia University, Wolfville, NS, B4P 2R6, Canada.

* **Correspondence:** Email: rhaynes@mun.ca.

**Abstract:** Damages due to pitting corrosion of metals cost industry billions of dollars per year and can put human lives at risk. The design and implementation of an adaptive moving mesh method is provided for a moving boundary problem related to pitting corrosion. The adaptive mesh is generated automatically by solving a mesh PDE coupled to the nonlinear potential problem. The moving mesh approach is shown to enable initial mesh generation, provide automatic mesh adjustment (or recovery) and is able to smoothly tackle changing pit geometry. Materials with varying crystallography are considered. Changing mesh topology due to the merging of pits is also handled. The evolution of the pit shape, the pit depth, and the pit width are computed and compared to existing results in the literature. Mesh quality results are also included.

**Keywords:** pitting corrosion; adaptive moving mesh; MMPDE; FEM; crystallography
**Mathematics Subject Classification:** 65M50

## 1. Introduction

Corrosion is a deterioration or breakdown of a material due to chemical or electrochemical reactions. In particular, pitting corrosion is one of the most disastrous and devastating localized forms of corrosion; generating a small pit, cavity or hole in the metal. Pitting corrosion is difficult to identify, and can have a big impact on the structural integrity of metal [1,2]. The pit geometry depends on many factors such as the components of the metal, the surface orientation, and the physical and chemical environment at the time of attack [3]. Corrosion pits can have different shapes [4] and with the ability to grow over time, failure of engineering structures such as bridges, pipelines, and nuclear power plants may result [4–6].

Computational modeling and simulations have been a tremendous asset in the study of pitting corrosion over a wide range of conditions and materials. Determining the pitting behavior experimentally is time consuming, expensive, and physically difficult or impossible in many situations. Numerical simulations allow us to study pitting under a wide range of conditions within a reasonable time.

In last few decades, several papers have focused on partial differential equation (PDE) based models for pitting corrosion based on finite element or finite volume methods [7–14]. A recent and extensive overview of the mathematical models for pitting corrosion based on the anodic reaction at the corrosion front, the transportation of ions in the pits of the electrolyte domain, and the pit growth over time is provided in [15]. In many of these studies COMSOL® is used to solve the PDE in the electrolyte domain and the corrosion front movement and meshing is computed by the arbitrary Lagrangian-Eulerian (ALE) approach and the level set method [7, 16]. In other studies, a 2D PDE model is solved with the finite element method [17, 18] and the finite volume method [9, 19]. Pit growth is determined by finite element methods and a level set approach in [18], and using an extended finite element method (XFEM) and level set method in [17]. In 2020, an ALE method was implemented to move the mesh at the pit boundary and analyze the relationship between the corrosion behavior and the local corrosive environment within a single pit [20].

The previously mentioned FEM approaches relied on a complete remeshing of the domain at every time step. An alternative technique, presented here, uses an adaptive moving mesh method where the mesh size, shape and orientation of the mesh elements are automatically and continuously varied for each time step, while keeping the number of nodes and mesh topology fixed throughout the computation.

Continuous mesh movement approaches are divided into two main categories: velocity-based approaches and location-based approaches. Most velocity-based approaches are motivated by the Lagrangian algorithm, where the mesh movement is tightly associated with the fluid or material particle flow. The Eulerian approach has a fixed computational mesh and the continuum moves with respect to the mesh nodes. The Eulerian and Lagrangian algorithms are commonly used in fluid dynamics and structural material problems, respectively [21]. In general, Eulerian meshes avoid mesh tangling and diffusive solutions, but the method can have difficulty adjusting to sharp material interfaces. One of the advantages of the Lagrangian approach is that the advective terms do not appear in the governing equations. Thus, the Lagrangian methods are less diffusive compared to the Eulerian approach, while also maintaining sharp material interfaces [22]. The ALE methods are velocity-based methods, which provide a combination of Lagrangian and Eulerian approaches [23–28].

The main goal of the location-based mesh movement approach is to directly control the location of mesh points in particular regions of the computational domain. A typical location-based method is the variational approach, which relocates the mesh points by movements that are based on minimizing a functional formulated to measure the difficulty or the error in the numerical solution [22]. Other location-based algorithms are based on elliptic PDE descriptions which can be used to generate boundary-fitted meshes [29,30], sometimes known as Winslow's approach [31]. Winslow's idea can be generalized using a functional [32], which provides a combination of the mesh adaptivity, smoothness, and orthogonality conditions.

A number of articles consider mesh adaptation functionals including mechanical models [33–35], vector fields [36], a weighted Jacobian matrix [27, 37], a matrix-valued diffusion coefficient [38, 39], and the equidistribution and isotropy (or alignment) conditions presented in [40]. The moving mesh PDE (MMPDE) method that we use has been developed by several authors [38,41,42,42–44]. Therein, the mesh movement is determined by a gradient flow equation, and the functional plays the vital role.

To our knowledge, an adaptive moving mesh method, our method of choice, has not been implemented for PDE-based modelling of pitting corrosion. In our moving mesh method, the FEM provides the spatial discretization and our computational framework is an extensively expanded version of MMPDElab developed by Huang [45]. MMPDElab is a general adaptive moving mesh finite element solver for time dependent PDEs based on integration of the MMPDE. The solver uses an alternating mesh and physical solution approach. Here the solver is used to achieve an adaptive moving mesh which provides sufficient mesh elements in and around the pit. Here a mesh is sufficient if it able to resolve the changing features of the pit boundary. We focus on the development of an appropriate mesh density function which implicitly and automatically determines an appropriate distribution of nodes as the pit evolves.

Our paper provides: 1) a proof of concept implementation of the moving mesh approach for pitting corrosion, 2) an adaptive solver for both single and multiple crystal directions, 3) the ability to handle single and multiple pits, and 4) the ability to provide (provably) nonsingular quality evolving meshes in an automatic way. The test material for demonstrating these techniques is 316 stainless steel using the parameters provided in [7, 16].

This paper is organized as follows. We discuss the preliminaries of the pitting corrosion mechanism, the crystal orientation, the PDE-model, and an overview of the moving mesh methodology in Section 2. The finite element method approach for the physical PDE, the choice of mesh density function and moving mesh parameters, initial mesh generation, the boundary movement strategy, and the overall alternating solution approach are discussed in Section 3. Section 4 is devoted to our numerical results and validation.

## 2. Model problem

In this section we detail our prototype model problem and adaptive solution strategy including the description of the domain, model PDEs and boundary conditions, the necessary crystallography, and an overview of the moving mesh strategy.

### *2.1. PDE model equation*

The transport of species (atoms, molecules or ions) in a solution can be modelled by the conservation of mass [46, 47], which leads to the mathematical form

$$\underbrace{\frac{\partial c_j}{\partial t}}_{\text{Storage}} = \underbrace{-\nabla\cdot\mathbf{N}_j}_{\text{Flux in - Flux out}} + \underbrace{R_j}_{\text{Generation}}, \tag{2.1}$$

where $t$ is time, $c_j$ is the concentration of the $j$th species, $\mathbf{N}_j$ is the flux of the $j$th species, and $R_j$ is the rate of species generation due to chemical reactions. The measure of the rate at which ions pass through a specific area (or the ionic flux) depends on the gradient of ion concentration, electro-migration, and convection (or flow in a liquid medium). An electrolyte is a charged substance containing ions. For each individual species $i$, the transport of a species in the electrolyte is described by the Nernst-Planck equation as

$$\mathbf{N}_j = -D_j\nabla c_j - z_jFu_jc_j\nabla\varphi + c_j\mathbf{v}, \tag{2.2}$$

where $D_j$ is the diffusion coefficient of the $j$th species, $u_j$ is the mobility of the species, $z_j$ is the charge of the species, $\varphi$ is the electric potential, $F$ is Faraday's constant, and $\mathbf{v}$ is the solvent velocity [48]. The solvent is usually water. Equation (2.2) gives the flux of the species as a combination of three contributing fluxes. The term $-D_j\nabla c_j$ describes the diffusive flux, the term $-z_jFu_jc_j\nabla\varphi$ gives the electro-migration flux and the term $c_j\mathbf{v}$ is the convection flux. Generally, the diffusion coefficient varies with the position of the species but we assume the diffusion coefficient is constant in our model.

We now substitute the assumed form of the transport as described by Nernst-Planck equation (2.2) into (2.1) to obtain the complete coupled behavior observed in the electrolyte domain during pitting corrosion. To obtain our model problem we make certain physically motivated assumptions: a) the absence of gradients in the species concentration due to the rapid mixing of the electrolyte ($D_j\nabla^2 c_j = 0$); b) the incompressibility of the solvent ($\frac{\partial c_j}{\partial t} = 0$); c) the zero net production of the reactants ($R_j = 0$); and d) the electro-neutrality of the electrolyte solution ($\nabla(c_j\mathbf{v}) = 0$). These assumptions simplify (2.1) to the well-known Laplace equation. In this case, the electrolyte potential can be found by solving (on the electrolyte domain, $\Omega$, shown in Figure 1)

$$\nabla^2\varphi = 0 \text{ in } \Omega, \tag{2.3}$$

with the following boundary conditions

$$\begin{aligned} \varphi &= 0 \text{ on } \Gamma_1, \\ \nabla\varphi\cdot\mathbf{n} &= 0 \text{ on } \Gamma_2, \Gamma_3, \Gamma_4, \\ \nabla\varphi\cdot\mathbf{n} &= \frac{i_a(\varphi)}{\sigma_c} \text{ on } \Gamma_p, \end{aligned} \tag{2.4}$$

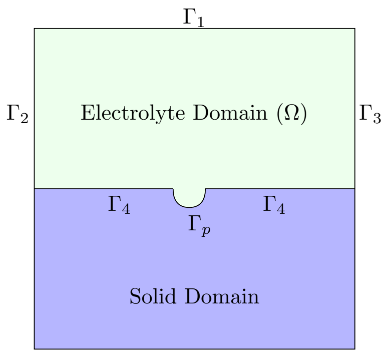


**Figure 1.** The 2D computational domain.

where $\nabla\varphi\cdot\mathbf{n} = \frac{\partial\varphi}{\partial\mathbf{n}}$, $\mathbf{n}$ is the (outward) unit normal vector with respect to $\Omega$, $i_a(\varphi)$ is the anodic current density, $\sigma_c$ is the electrical conductivity of the electrolyte, $\Gamma_p$ is the pit boundary, and $\Gamma_1$, $\Gamma_2$, $\Gamma_3$, and $\Gamma_4$ are the top, left, right, and bottom of the domain (excluding the pit boundaries), respectively. The boundary condition $\frac{\partial\varphi}{\partial\mathbf{n}} = 0$ enforced on $\Gamma_2$, $\Gamma_3$ and $\Gamma_4$ ensures there is no flow of ions across these boundaries. We denote the horizontal and vertical co-ordinates of the electrolyte region in Figure 1 by $x$ and $y$, respectively.

The current density is modelled by the Butler-Volmer relation

$$i_a(\varphi) = zFA_s \cdot e^{\left(\frac{zF(V_{corr}+\alpha\eta_a)}{RT}\right)}, \tag{2.5}$$

where $\alpha$ is the transfer coefficient, $z$ is the average charge number for the dissolving metal, $A_S$ is the material dissolution affinity, $T$ is the temperature, and $R$ is the universal gas constant [49].

The Butler-Volmer relation is used to describe the experimental data as a function of the applied over-potential

$$\eta_a = V_{app} - V_{corr} - \varphi,$$

where $V_{app}$ and $V_{corr}$ are the applied and the corrosion potentials, respectively [9].

As the metal corrodes, the pit boundary moves as the pit becomes larger. In our model, the new position of corrosion front at $t = t_{n+1}$, $X_{n+1}$, is computed from the old position, $X_n$, by a simple time stepping procedure

$$X_{n+1} = X_n + \Delta t V_n \mathbf{n}, \tag{2.6}$$

where $V_n$ is the magnitude of normal velocity at $t = t_n$. The magnitude of normal velocity at the corrosion interface (or the movement of the corrosion front) is described using Faraday's law

$$V_n = \frac{i_a(\varphi)}{zFc_{solid}}, \tag{2.7}$$

where $c_{solid}$ is the atomic mass concentration of the metal and $z$ is the average charge number for the metal. Table 1 gives a list of parameters.

**Table 1.** List of parameters used in the corrosion model.

| Parameter | : | Description | Value |
|---|---|---|---|
| $z$ | : | Average charge number for the metal | 2.19 |
| $F$ | : | Faraday's constant | 96485 C/mol |
| $R$ | : | Universal gas constant | 8.315 J/(mol K) |
| $T$ | : | Temperature | 298.15 K |
| $V_{\text{corr}}$ | : | Mean corrosion potential (homogeneous) | -0.24 V |
| $V_{\text{app}}$ | : | Applied potential | -0.14 V |
| $A_{\text{s}}$ | : | Dissolution affinity | 4 mol/cm$^2$s |
| $c_{\text{solid}}$ | : | Solid concentration | 143 mol/l |
| $\Delta t$ | : | Time step size | 1 |

## 2.2. *Crystallography for corrosion*

### 2.2.1. Crystal structure and unit cell

The crystal structure of the solid part of the computational domain refers to the arrangement of atoms, ions, or molecules in a crystalline material. Crystals are solids with a regular and repeating three-dimensional pattern known as a lattice as demonstrated in Figure 2. This lattice structure is responsible for the characteristic geometric shapes and symmetries observed in crystals. The importance of crystal structure to various scientific disciplines and industries is due to its significant impact on material properties and behavior. For example, the arrangement of atoms in a crystal lattice directly influences material properties such as mechanical strength, thermal conductivity, electrical conductivity, and optical behavior, etc, [50]. Understanding crystal structure is essential for designing and tailoring materials with specific properties for various applications.

A unit cell, as shown in Figure 2, is the basic building block of a crystal lattice, representing the smallest repeating unit of a crystal structure. It is a three-dimensional parallelepiped (a six-faced figure with each face being a parallelogram) with edges defined by lattice parameters $a$, $b$, and $c$, and interaxial angles $\alpha$, $\beta$, and $\gamma$. There are several types of unit cells, with the most common for metals being primitive cubic, body-centered cubic, and face-centered cubic, which represent different crystal structures based on the arrangement of atoms within the unit cell. The unit cell allows scientists and researchers to predict and analyze the physical, mechanical, and thermal properties of crystals, as well as predict their diffraction patterns using techniques like X-ray crystallography.

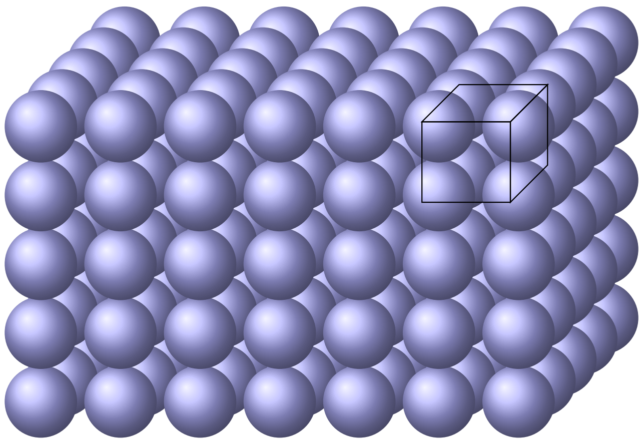


**Figure 2.** Crystal lattice of a simple cubic material with some atomic layers and a unit cell displayed.

### 2.2.2. Miller indices, directions and planes

Miller indices represent the three-dimensional coordinate system for crystals, based on their unit cell. The indices are used in crystallography to describe the orientation of crystal planes and directions within a crystal lattice. The indices are represented by three integers denoted by [$h$ $k$ $l$]. Note that the values of Miller indices cannot be fractions, and there is no need for commas to separate the values of Miller indices.

In crystallography, square brackets [ ] represent a particular direction and angle brackets <> denote a family of directions. Parentheses () denote a specific plane, and curly brackets {} indicate a family of symmetrically equivalent planes. Miller indices use a unique notation for negatives, represented by a "bar" symbol (e.g., $\bar{1}$ for -1). For example, [0$\bar{1}$0] is pronounced as "the zero, bar one, zero direction."

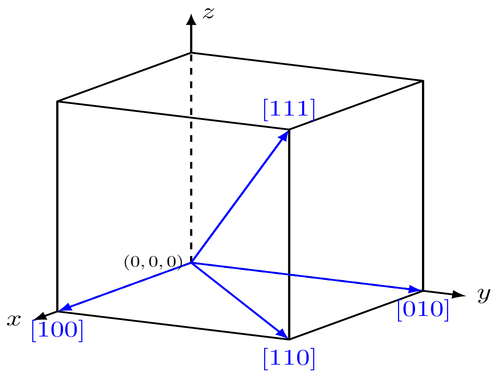
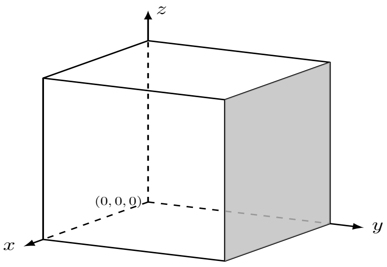


**Figure 3.** Miller indices of some directions within a cubic crystal.

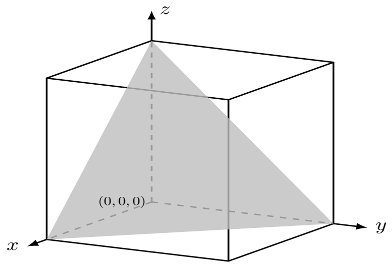
In a cubic system, [100] and [010] are perpendicular directions and $< 100 >$ includes the directions [100], [$\bar{1}$00], [010], [0$\bar{1}$0], [001], and [00$\bar{1}$]. The example of a direction for a face diagonal is [110], and [111] represents a body diagonal for a cubic unit cell as displayed in Figure 3.

Miller indices are defined with respect to a coordinate system, so the first definition we need is an origin. The origin is a point (0,0,0) and we can define it anywhere in the crystal since the crystal is assumed to be infinite in all directions. For a given crystal direction, we can draw the direction vector in the unit cell using Miller indices. It is important to note that Miller indices of directions always translate the direction so that it starts at the origin. The drawing procedure is as follows: first, determine the origin as the tail of the direction vector (if there is a negative number in the direction vector, move the origin in the positive direction for the axis). Next, determine the vector endpoint (remove a common factor if any of the indices are larger than 1). Finally, draw the vector.

Consider a unit cubic cell with atoms at each corner. Now, let us assume an imaginary plane passing through the unit cell (or lattice points of the unit cell). Miller indices for the plane are calculated by



**(a)** Lattice plane (010).



**(b)** Lattice plane (111).

**Figure 4.** Lattice planes.

determining the intercepts made by the plane on the $x$, $y$, and $z$ axes, respectively, and then take the reciprocals of these intercepts. If the plane passes through the origin, we need to shift the origin to the nearest lattice point on the parallel face. For example, planes (010) and (111) have been highlighted in grey in Figure 4. A zone axis is a lattice vector, usually aligned along a highly symmetrical direction, where two or more sets of lattice planes intersect.

### 2.2.3. Crystal orientation and corrosion potential

The corrosion potential is the mathematical link between the etching effects of the electrolyte and the material undergoing pitting. This potential term is present in the Butler-Volmer equation (2.5) and is an important parameter governing the velocity of the sides of the pit during corrosion. If the electrolyte etches the material homogeneously in all directions, then the crystal structure is not a variable in the modeling and only one number is required for $V_{corr}$. However, if the material is crystalline, then the corrosion potential may vary dependent on the particular crystallographic surface exposed to the electrolyte. Hence, a connection between the Cartesian $(x, y, 0)$ geometry used for defining the computational domains as presented in Figure 1, and the directions in the crystal, is needed. In general, these two geometries will not align since crystals can be rotated to lie along an infinite number of directions and a transformation will be required to relate the two coordinate systems. Letting $\mathbf{n}_{CD}$ represent the outward unit normal in the crystal coordinate system, and using the notation given in [51, 52], its relationship to $\mathbf{n}$ is

$$\mathbf{n}_{CD} = \mathbf{A}^{-1}\mathbf{n}. \tag{2.8}$$

The matrix $\mathbf{A}^{-1}$ is defined by

$$\mathbf{A}^{-1} = \left[ \begin{array}{c|c|c} \mathbf{i} & \mathbf{j} & \mathbf{k} \end{array} \right], \tag{2.9}$$

where **i**, **j**, and **k** are orthogonal unit column vectors in Cartesian space. **k** is chosen as the desired zone axis of the crystal and **i** is chosen as the direction perpendicular to **k**, which will be oriented along the $x$ axis in the computational domain. Thus, the crystal can be rotated in any direction about the zone axis offering maximum flexibility in the problems that can be studied. The third unit vector, **j**, is found using the perpendicularity property of vector cross products

$$\mathbf{j} = \mathbf{k} \times \mathbf{i}.$$

In order to better see how to form the required rotation matrix, two examples will be presented. First, select a crystal orientation where the zone axis is aligned along $\mathbf{k} = [001]$. Next, choose the $\mathbf{i} = [100]$ direction to be along the $x$ computational domain direction. Performing the $\mathbf{j} = \mathbf{k} \times \mathbf{i}$ cross product, it is found that $\mathbf{j} = [010]$. Hence, the transformation matrix is

$$\mathbf{A}_{001}^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

In this case, the transformation matrix is the identity matrix and the crystal coordinate system is the same as the computational domain coordinate system, i.e., $\mathbf{n}_{CD} = \mathbf{n}$. For the second example, choose the zone axis to be along the [101] direction and the $[\bar{1}01]$ crystal direction to be along the $x$ computational domain direction. This leads to $\mathbf{j} = [010]$. After normalizing the vectors, the transformation matrix in this case is

$$\mathbf{A}_{110}^{-1} = \begin{bmatrix} \frac{-1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} \\ 0 & 1 & 0 \\ \frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} \end{bmatrix}.$$

The next step is to define a corrosion potential for each crystallographic direction. Unfortunately, experimental data giving the corrosion potential as a function of crystallographic surface is usually not available, hence we will adopt a similar form of semi-empirical potential for 316 stainless steel

$$V_{corr} = k - s\left[1 - (\langle 001 \rangle \cdot \mathbf{n}_{CD})_{\max}\right], \tag{2.10}$$

where $k = -0.2297$ and $s = 0.054$ gives a 10% difference between the maximum and minimum $V_{corr}$ values, that is between the [001] and [111] crystal planes, as used by DeGiorgi et al. in [16]. We write $(\langle 001 \rangle \cdot \mathbf{n}_{CD})_{\max}$, rather than $[001] \cdot \mathbf{n}_{CD}$ as used in [16]. In our case, $\langle 001 \rangle$ represents any one of the six cryptographically equivalent [001], $[\bar{1}00]$, [010], $[0\bar{1}0]$, [001], $[00\bar{1}]$ directions that maximizes the dot product with the crystal direction normal vector. Maximizing this dot product minimizes the angle between the normal vector and the particular $\langle 001 \rangle$ vector so that an equivalent result to the standard stereographic triangle is obtained. Figure 5 is an example of this procedure where a single crystal has been oriented along the [001] zone axis. The semicircle represents the pit boundary, the black lines and arrows the outward pit normal vectors, and the heavy blue line is the value of $V_{corr}$ as a function of position along the edge of the pit. Highlighted by the three colours are sections along the pit boundary that have a different $\langle 001 \rangle$ vector for use in Eq (2.10). These vectors are presented in blue text and are [001], $[0\bar{1}0]$ and $[\bar{1}00]$ for the right, centre and left sections of the pit, respectively.

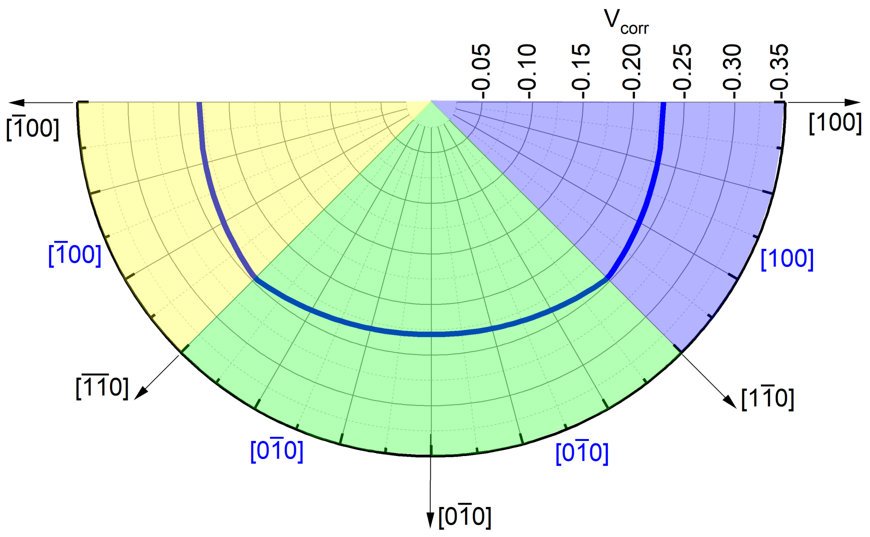


**Figure 5.** A plot of $V_{corr}$ as a function of location around the pit boundary. The blue, green and yellow sections of the pit edge require different <001> vectors for use in Eq (2.10).

## 2.3. *Overview of the moving mesh strategy*

The basic idea of the moving mesh method is to automatically redistribute a fixed number of nodes where additional accuracy is required. The mesh moves or evolves automatically as the solution or domain evolves, and is obtained by solving a MMPDE. This MMPDE depends on a mesh density or monitor function, which is large where the mesh density is needed to be large. The mesh density function is often chosen to depend on variations or errors in the solution of the physical PDE or is chosen by geometrical considerations (as in this paper).

In 1D, the equidistribution principle (EP) is used to derive the moving mesh system. The EP uses a mesh density function $\boldsymbol{\rho} = \boldsymbol{\rho}(x) > 0$, which is to be distributed evenly among the mesh elements in the domain. Given an integer $N > 1$, the continuous and bounded function $\boldsymbol{\rho}$ on $[a, b]$ is evenly distributed on the mesh $\mathcal{T}_h = a = x_0 < x_1 < \cdots < x_N = b$, if

$$\int_{x_0}^{x_1} \boldsymbol{\rho}(x)dx = \int_{x_1}^{x_2} \boldsymbol{\rho}(x)dx = \cdots = \int_{x_{N-1}}^{x_N} \boldsymbol{\rho}(x)dx. \tag{2.11}$$

A mesh $\mathcal{T}_h$ is called an equidistributing mesh if the mesh satisfies the equidistribution principle.

The physical problem is assumed to require a non-uniform $x$-coordinate, $x \in \Omega$. This physical coordinate, $x$, is a mapping of the computational $\xi$-coordinate where $\xi \in \Omega_c = [0, 1]$, and $x(0) = a$ and $x(1) = b$, if $\Omega = [a, b]$. We attempt to generate a physical mesh $\mathcal{T}_h$ using a mesh transformation $x = x(\xi) : \Omega_c \to \Omega$ and a uniform mesh in the $\xi$-coordinate

$$\xi_i = \frac{i}{N}, \quad i = 0, 1, \ldots, N.$$

The equidistribution principle (2.11) can then be written as

$$\begin{aligned}\int_a^{x_i} \boldsymbol{\rho}(\tilde{x})d\tilde{x} &= \frac{i}{N}\int_a^b \boldsymbol{\rho}(\tilde{x})d\tilde{x} \\ &= \frac{i}{N}\boldsymbol{\sigma}, \quad i = 0, 1, \ldots, N,\end{aligned}$$

where $\boldsymbol{\sigma} = \int_a^b \boldsymbol{\rho}(\tilde{x})d\tilde{x}$. The function $\int_a^x \boldsymbol{\rho}(\tilde{x})d\tilde{x}$ is strictly monotonically increasing if $\boldsymbol{\rho} > 0$, therefore each $x_i$ is unique. Here $\boldsymbol{\sigma}$ and $\frac{i}{N}\boldsymbol{\sigma}$ are estimates (or surrogates) for the total error and average error in the approximating solution, respectively.

Using the mesh transformation, we have

$$\int_a^{x(\xi_i)} \boldsymbol{\rho}(\tilde{x})d\tilde{x} = \xi_i\boldsymbol{\sigma}, \quad i = 0, 1, \ldots, N,$$

and the continuous version is given by

$$\int_a^{x(\xi)} \boldsymbol{\rho}(\tilde{x})d\tilde{x} = \xi\boldsymbol{\sigma}, \quad \forall\xi \in \Omega_c. \tag{2.12}$$

The continuous mapping $x = x(\xi)$ is called a *equidistributing coordinate transformation* for $\boldsymbol{\rho}$ if it satisfies relation (2.12). Differentiating (2.12) with respect to $\xi$ gives

$$\boldsymbol{\rho}(x)\frac{dx}{d\xi} = \boldsymbol{\sigma}. \tag{2.13}$$

Equation (2.13) indicates that $\frac{dx}{d\xi}$ is small when $\boldsymbol{\rho}$ is large. Again, differentiating with respect to $\xi$ gives

$$\frac{d}{d\xi}\Big(\boldsymbol{\rho}(x)\frac{dx}{d\xi}\Big) = 0, \tag{2.14}$$

with the boundary conditions

$$x(0) = a, \quad x(1) = b. \tag{2.15}$$

This is a nonlinear boundary value problem (BVP) for the required mesh transformation and physical mesh. The mesh and physical solution on that mesh is determined by solving this BVP and the physical PDE as a coupled system.

In higher dimensions, in order to describe the equidistribution and alignment conditions at the discrete level, we consider a mesh $\mathcal{T}_h$ of $N$ triangular elements with $N_v$ vertices in the physical domain $\Omega \in \mathbf{R}^d$ ($d \geq 1$). Furthermore, we consider an invertible affine mapping $F_K : \hat{K} \to K$ and its Jacobian matrix, $F_K^{'}$, where $\hat{K}$ is the reference or master element for a physical element $K$ in $\mathcal{T}_h$. In higher dimensions we will assume that a metric (or a monitor function) $\mathbb{M} = \mathbb{M}(x)$ is given on $\Omega$ which determines the shape, size, and orientation of mesh elements of the domain $\Omega$. Generally, a mesh is uniform if all of its elements have the same size and is similar to a reference element $\hat{K}$. So, the main idea of the MMPDE method is to view any adaptive mesh $\mathcal{T}_h$ as a uniform mesh in the metric $\mathbb{M}$.

The requirements of the equidistribution and alignment in higher dimensions can be expressed mathematically at the discrete level [22] as

$$|K|\sqrt{\mathtt{det}(\mathbb{M}_K)} = \frac{\sigma_h}{N}, \forall K \in \mathcal{T}_h,$$

and

$$\frac{1}{N}tr((F'_K)^T\mathbb{M}_K F'_K) = \mathtt{det}((F'_K)^T\mathbb{M}_K F'_K)^{\frac{1}{d}}, \forall k \in \mathcal{T}_h,$$

where $|K|$ is the volume of $K$ and $\sigma_h = \sum_{K\in\mathcal{T}_h}|K|\sqrt{\mathtt{det}(\mathbb{M}_k)}$.

A standard choice of the metric $\mathbb{M}$ is based on the approximate Hessian of the solution. This choice of $\mathbb{M}$ is known to be optimal with respect to the $L_2$ norm of the linear interpolation error [53]. Here, we focus on the evolution of the pit geometry and choose $\mathbb{M}$ through geometrical considerations (see Section 3.2).

A discrete functional associated with the equidistribution and alignment conditions is given by

$$I[\mathcal{T}_h] = \sum_{K\in\mathcal{T}_h}|K|\sqrt{\det(\mathbb{M}_K)}\left[\theta\left(\mathrm{tr}(\mathbb{J}\mathbb{M}_K^{-1}\mathbb{J}^T)\right)^{\frac{d\gamma}{2}} + (1-2\theta)d^{\frac{d\gamma}{2}}\Big(\frac{\det(\mathbb{J})}{\sqrt{\det(\mathbb{M}_K)}}\Big)^{\gamma}\right], \tag{2.16}$$

where $\mathbb{J} = (F'_K)^{-1}$. Minimizing this functional $I[\mathcal{T}_h]$ approximately satisfies the equidistribution and alignment conditions [40]. The value of the parameters, $\theta = \frac{1}{3}$ and $\gamma = \frac{3}{2}$, are used for our numerical experiments.

The MMPDE moving mesh equation can then be defined as the (modified) gradient system (or gradient flow equation) for the energy functional, i.e.,

$$\frac{d\mathbf{x}_i}{dt} = -\frac{P_i}{\tau}\frac{\partial I[\mathcal{T}_h]}{\partial \mathbf{x}_i}, \quad i = 1,2,\ldots,N_v, \quad t\in(t_n,t_{n+1}], \tag{2.17}$$

where $P_i = \det(\mathbb{M}_i)^{\frac{1}{d+2}}$ is a scalar function used to ensure the mesh equation has invariance properties and $\tau$ is a positive parameter used to adjust the response time of mesh movement to the change in $\mathbb{M}$. A smaller value of $\tau$ provides a faster response.

## 3. The numerical implementation

This section describes the details of the adaptive MMPDE strategy used to solve the PDE pitting corrosion model using a customized version of MMPDELab [45].

### *3.1. Discretization of the physical PDE*

MMPDElab requires the user to specify the physical PDE in weak form, where the strong form of our model problem is given in Eqs (2.3) and (2.4). Let $V$ be the trial space, chosen in this case as

$$V = \{v\in H^1(\Omega(t)) : v = 0 \text{ on } \Gamma_1\} \subset H^1(\Omega(t)),$$

where $H^1(\Omega(t))$ is, roughly speaking, the function space whose members, and their first derivatives, are square integrable (see, for example, [54] for details). At any time $t$ the weak form is constructed as follows: find $\varphi\in V$ such that

$$\int_{\Omega(t)}\nabla\varphi\cdot\nabla v d\Omega = \int_{\Gamma_p(t)} v\frac{i(\varphi)}{\sigma_c}ds, \quad \forall v\in V, \tag{3.1}$$

where $\Gamma_p(t)$ is the boundary of the pit at time $t$. Here $V_h$ denotes a finite dimensional subspace of $V$ spanned by a collection of finitely many basis functions (often associated with a mesh). We discretize the weak form (3.1) to find a solution in the discrete trial space. The discrete FEM solution is then found by finding $\varphi_h \in V_h \subset V$, such that

$$\int_{\Omega(t)} \nabla\varphi_h \cdot \nabla v_h d\Omega = \int_{\Gamma_{p(t)}} \frac{i(\varphi_h)}{\sigma_c} v_h ds, \quad \forall v_h \in V_h. \tag{3.2}$$

We can solve the discrete variational problem (3.2) in the following way. First, introduce $\{\phi_j\}_{j=1}^{N}$ as a basis for $V_h$. Let $\varphi_h \in V_h$ be a linear combination of the basis functions $\phi_j$, $j = 1, 2, \ldots, N$, with coefficients $\tilde{\varphi}_j$, given by

$$\varphi_h = \sum_{j=1}^{N} \tilde{\varphi}_j \phi_j. \tag{3.3}$$

Considering $v = \phi_k$, for $k = 1, 2, \ldots, N$, and using relation (3.3), gives

$$\sum_{j=1}^{N} \tilde{\varphi}_j \int_{\Omega} \nabla\phi_j . \nabla\phi_k d\Omega - \frac{1}{\sigma_c} \int_{\Gamma_p} i\Big(\sum_{j=1}^{N} \tilde{\varphi}_j \phi_j\Big)\phi_k ds = 0, \quad k = 1, 2, \ldots, N.$$

For each time, $t$, this is a system of non-linear equations which is solved using Newton's method.

## *3.2. The choice of the mesh density function*

The appropriate specification of the mesh density is crucial — it controls how the mesh automatically adapts to the changing solution features.

As mentioned previously, for problems with fixed domain boundaries, the Hessian based monitor function is an often used, general purpose, driver of the adaptive mesh. Here, however, we wish to ensure sufficient resolution of the pit geometry. To ensure a sufficient number of elements in the evolving pit and near the pit boundary, we use a modified version of MacKenzie's distance-based monitor function

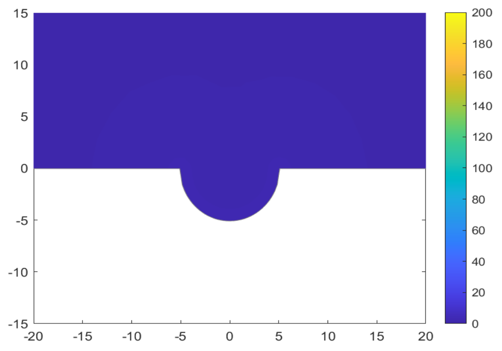
$$\mathbb{M}_K(x, y) = \left(1 + \frac{\mu_1}{\sqrt{\mu_2^2 \tilde{d}^2 + 1}}\right) I, \tag{3.4}$$

where

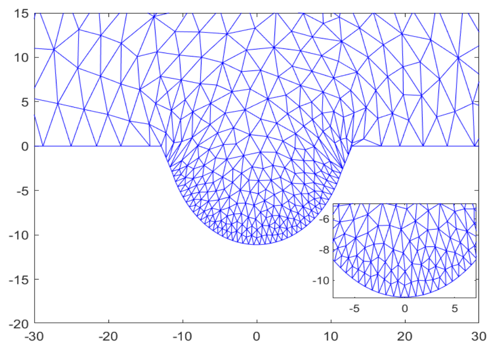
$$\tilde{d}(x, y) = \min_p ||(x, y) - (x_p, y_p)||,$$

and $(x_p, y_p)$ denotes any point on the boundary of the pit, $\Gamma_p$, cf. [55]. At any point $(x, y) \in \Omega$, the value of the monitor function involves the minimum distance, measured in the two-norm $|| \cdot ||$, from $(x, y)$ to any point on the pit boundary. The reciprocal of $\mathbb{M}_K$ indicates that $\mathbb{M}_K$ will be largest in $(x, y)$ regions where the distance to the pit boundary is the smallest, and hence the mesh spacing will be automatically smaller in these regions. The parameter $\mu_1$ controls the minimum mesh spacing, whereas $\mu_2$ (and $\tau$) will control the rate at which mesh clustering occurs during the integration of the MMPDE [56]. To understand the effect of the $\mu_1$ and $\mu_2$ parameters, we consider a simple experiment, simulation of the evolution of a single pit in a homogeneous material. A quality initial mesh, generated using the process outlined in Section 3.3 is used for this experiment.

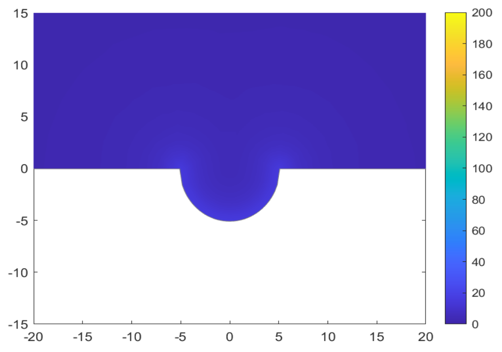
We begin by fixing $\mu_2$, at $\mu_2 = 1$, and consider the effect of increasing $\mu_1$. The plots on the left of each row in Figure 6 show a representative mesh density function (computed at $t = 2s$), while



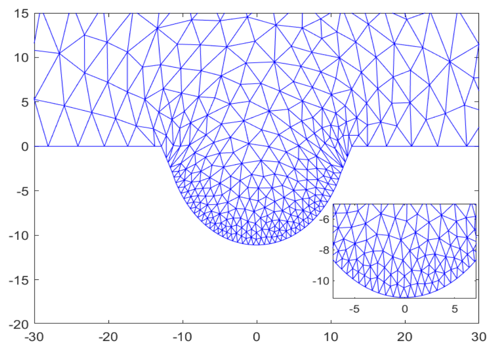
**(a)** Monitor function, $\mu_1 = 1$.



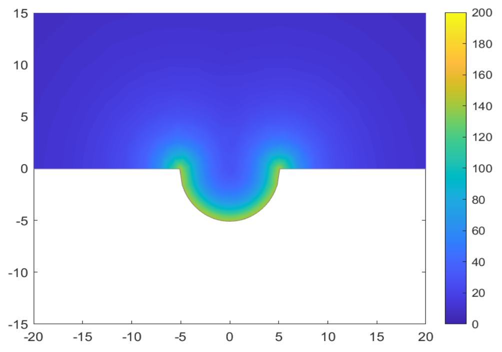
**(b)** The mesh at $t = 120$ s with $\mu_1 = 1$.



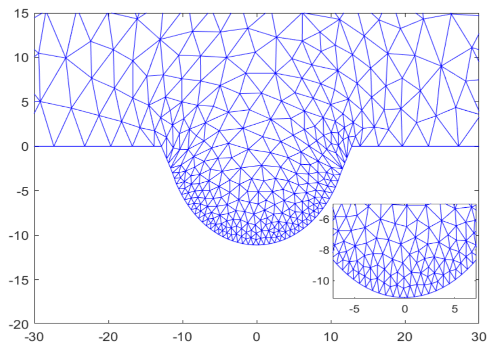
**(c)** Monitor function, $\mu_1 = 10$.



**(d)** The mesh at $t = 120$ s with $\mu_1 = 10$.



**(e)** Monitor function, $\mu_1 = 100$.

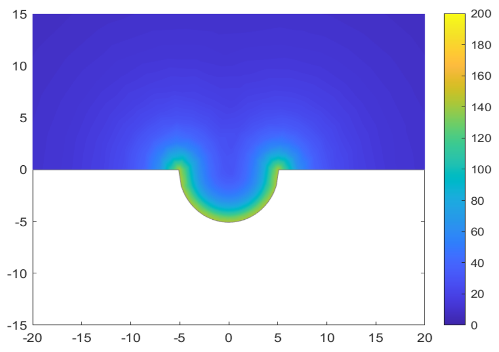


**(f)** The mesh at $t = 120$ s with $\mu_1 = 100$.

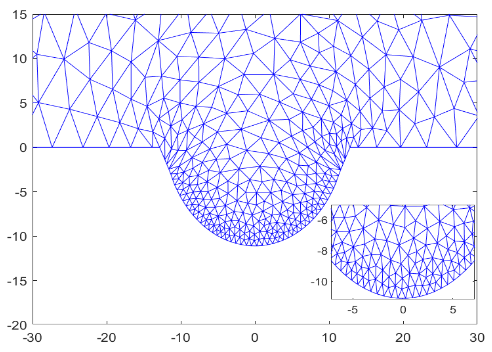
**Figure 6.** Effect of $\mu_1$ on the mesh at $t = 120$ s for the simulation of a pit in a homogeneous material with the monitor function (3.4) and $\mu_2 = 1$.

the computed mesh after $t = 120s$ is shown on the right. The results show that increasing $\mu_1$ leads to a monitor function which is (relatively) larger near the pit boundary and hence maintains smaller grid spacings near the pit boundary. Counting the number of elements in the pit shows an increase of approximately 6% as $\mu_1$ is increased from 1 to 20. Further increasing $\mu_1$ to 100 only adds an additional 1.5% to the number of elements in the pit. Increasing $\mu_1$ larger than 100, however, provides no additional improvements. Choosing a sub-optimal value of $\mu_1$ causes the mesh to slightly lag behind the movement of the pit boundary, an effect that compounds as we integrate further in time.

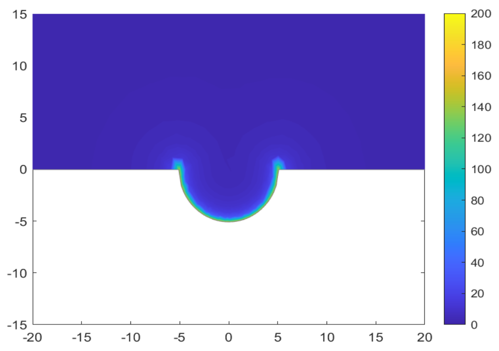
Mackenzie [56] reports that increasing $\mu_2$ reduces the spatial extent of node clustering near the pit boundary. To explore this effect, we fix $\mu_1 = 100$ and vary $\mu_2$. Representative mesh density function values, and the final mesh obtained for the propogation of a homogeneous pit, are shown in Figure 7. The figure shows that the effect of increasing $\mu_2$ is more subtle. Increasing $\mu_2$ to 20 produces a mesh with only a 1% increase in the number of elements in the pit.



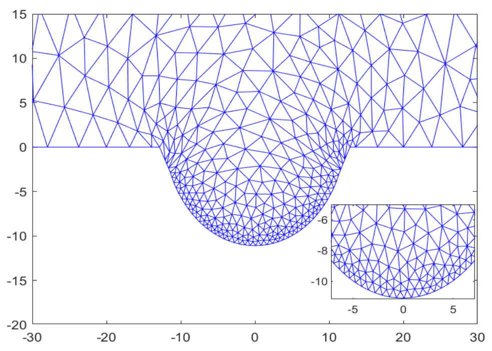
**(a)** Monitor function, $\mu_2 = 1$.



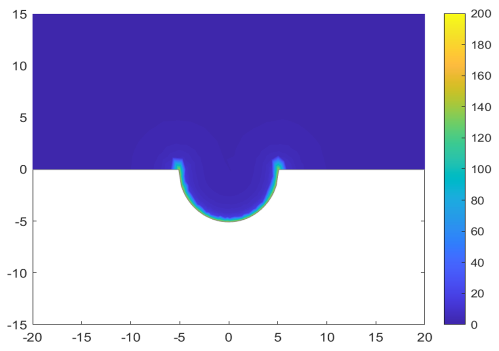
**(b)** The mesh at $t = 120$ s using $\mu_2 = 1$.



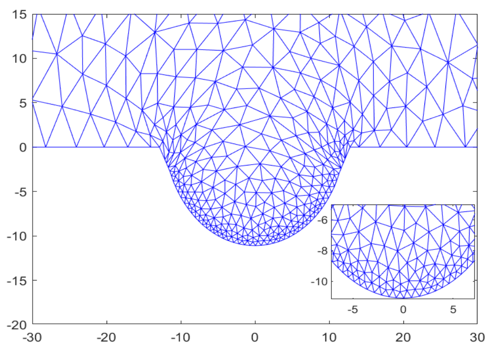
**(c)** Monitor function, $\mu_2 = 10$.



**(d)** The mesh at $t = 120$ s using $\mu_2 = 10$.



**(e)** Monitor function, $\mu_2 = 20$.



**(f)** The mesh at $t = 120$ s using $\mu_2 = 20$.

**Figure 7.** Effect of $\mu_2$ on the mesh at $t = 120$ s during the simulation of a pit in a homogeneous material with the monitor function (3.4) and using $\mu_1 = 100$.

### *3.3. Initial mesh generation*

The numerical simulations in this paper require an initial pit geometry and an initial spatial grid. With a prescription of the spatial domain, many software platforms provide tools known as mesh generators for this purpose, for example **initmesh** in Matlab or **mesh node** in COMSOL. These tools require a description of the domain boundary and then generate a mesh subject to constraints on the mesh size (or number of nodes) and aspect ratios of the mesh elements.

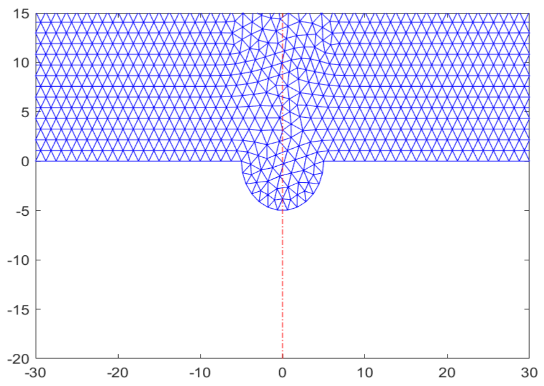
In most cases the solutions found on the initial meshes generated by these approaches will not be optimal. For example, there is no guarantee that the error in the numerical solution will be minimized. There are alternatives for initial mesh generation that involve the MMPDE approach considered in this paper, and hence are consistent with the technique used for all subsequent time steps.

One technique for initial mesh tuning begins by using the simple mesh generators mentioned above to find a (nearly) uniform mesh. The physical problem defined by Eqs (2.3) and (2.4) is then solved on this mesh to give an initial potential. Using this initial potential and its associated mesh density function $\mathbb{M}_K$, the gradient flow equation (2.17) can be solved to a steady state by alternating its solution with physical solves. The result is a mesh which minimizes the discrete functional (2.16), thereby equidistributing the initial potential over the initial computational domain. Should a more sophisticated non-uniform mesh generator be available, then an initial non–uniform mesh can be smoothed in the same manner. In practice, Equation (2.17) may not be solved to a steady state. Instead, Equation (2.17) can be integrated for a specified number of time steps, or can be integrated until a specified difference between two meshes is found. We will call this mesh smoothing. The number of steps required to reach a suitable approximation of the steady state is a function of the physical solution, the number of mesh nodes, and the mesh density parameters. This idea of using the MMPDE to tune the initial mesh has the added benefit of giving a mesh which has the same properties as all subsequent meshes, while using the same code base as the rest of the simulation. We note that this process can also be used to smooth abrupt changes in element sizes or shapes that appear during the solution of the moving boundary problem. This is particularly useful should the pit boundary movement be large or if the pit movement induces a discontinuous change in the geometry (during pit merging for example).

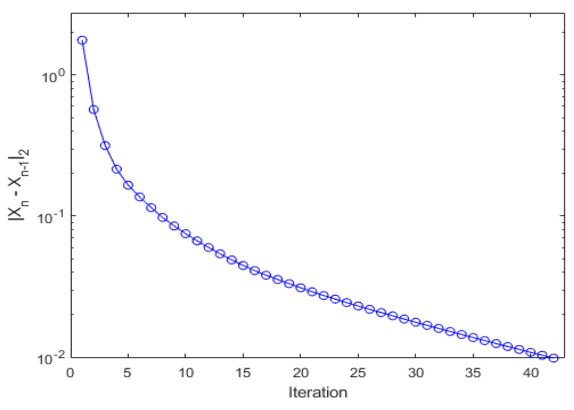
In Figure 8 we show the evolution of the pit geometry in a homogeneous material. The moving mesh method is implemented with the monitor function (3.4) starting from the initial uniform mesh shown in Figure 8a. The non-uniform mesh resulting from the solution of the gradient flow equation, shown in Figure 8c, has successfully concentrated the mesh elements in the initial pit and near the initial pit boundary. The convergence to a steady state solution of the gradient flow equation is shown in Figure 8b. This resulting non-uniform mesh is now an appropriate initial mesh to use to evolve the pitting corrosion problem forward in time. It is important to stress that no hand-tuning of the initial mesh is necessary, it is generated automatically during the smoothing process based on the characteristics of the chosen mesh density function. The final mesh after $t = 60$ s is given in Figure 8d.

Presented in Figure 9a is the non-uniform initial mesh generated with the Matlab function **initmesh**, where 45 nodes are located on the pit boundary. In order to optimize this mesh, smoothing steps were performed, and the sum of the absolute differences in position of the nodes between subsequent smoothing iterations is displayed in Figure 9b. The motion of the nodes decreases with iteration number, and after 17 iterations the absolute difference is down to $10^{-2}$. The optimized mesh is shown in Figure 9c and significant differences in the locations of the nodes both inside and outside the pit are observed. The smoothed mesh has a more uniform element size and shape near the pit boundary. This impact of smoothing is more pronounced if the elements become more stretched at some point in the calculation due to large changes in the pit boundary location. Note that this smoothing operation only needs to be performed once since the results can be saved and used as the starting mesh in subsequent experiments. The computational mesh after the pit has evolved for 60 s is presented in Figure 9d, and it is observed that node spacing within the pit remains very good.

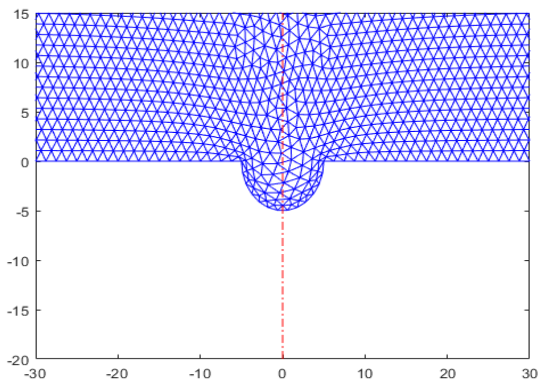
The results in Figures 8 and 9 show that the MMPDE approach is robust with respect to the initial grid, continuously evolving the mesh according to changing domain and solution features. Even with a uniform initial mesh, the MMPDE approach does quite well, automatically recovering the requested increased mesh density near the pit boundary. We do notice, however, some additional stretching of the nodes in this case as compared to the simulation which starts from an improved non-uniform initial mesh. The stretching of the mesh can be reduced through the use of monitor functions designed to control the shape of the elements.



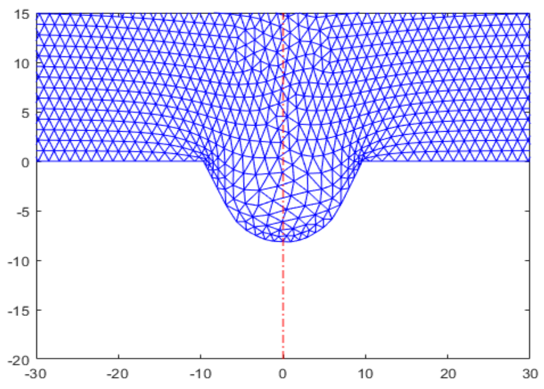
**(a)** An uniform initial mesh.



**(b)** The convergence of the initial mesh smoothing.



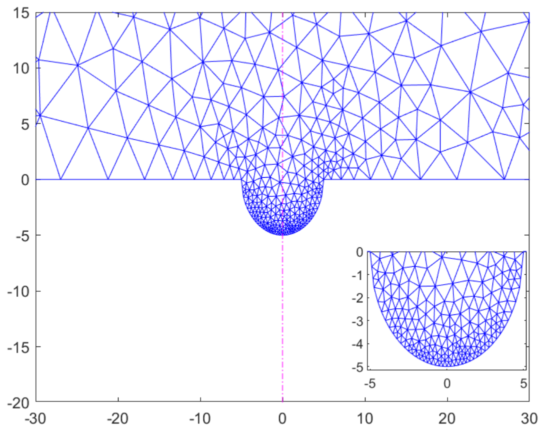
**(c)** The smoothed initial mesh.



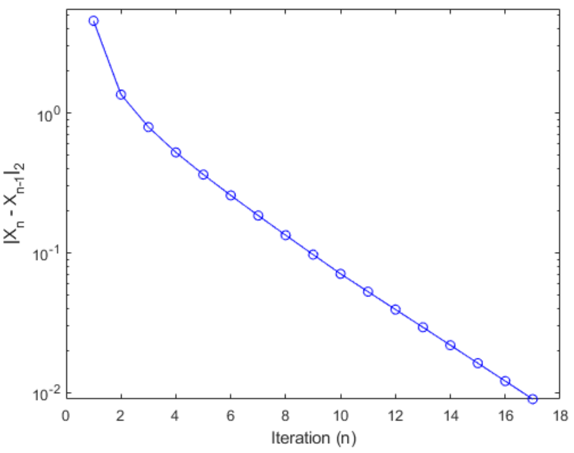
**(d)** The mesh after 60 s.

**Figure 8.** (a) Uniform initial mesh, (b) convergence of the mesh smoothing process, (c) initial mesh after mesh smoothing, and (d) the mesh after 60 s using the monitor function (3.4).

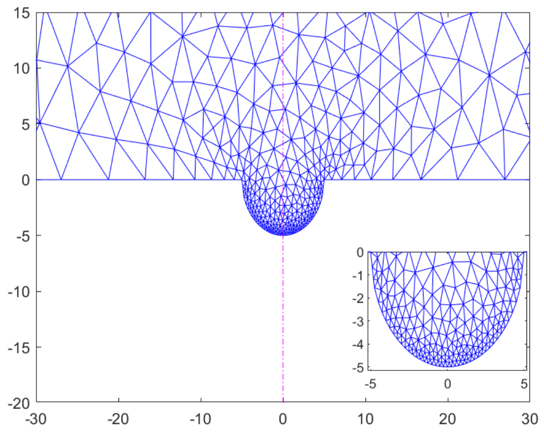
The relatively small scale mesh smoothing is particularly useful should the pit boundary movement be large or if the pit boundary movement induces a discontinuous change in the geometry. We will see this effect in Section 4.



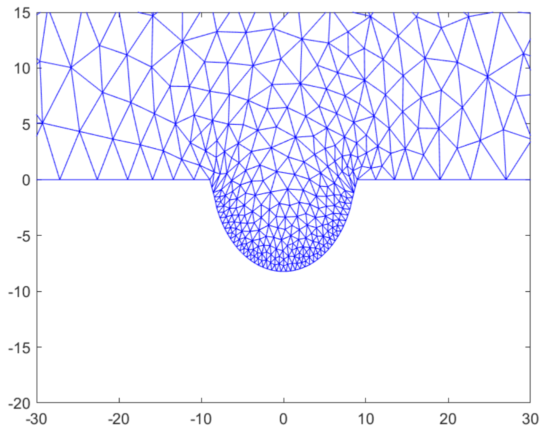
**(a)** A nonuniform initial mesh.



**(b)** The convergence of the initial mesh smoothing.



**(c)** The smoothed initial mesh.

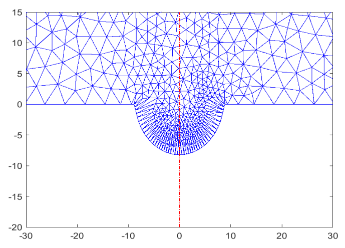


**(d)** The mesh after $t = 60$ s.

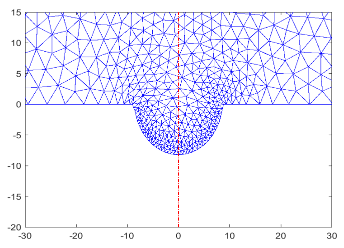
**Figure 9.** (a) A nonuniform initial mesh, (b) the effect of mesh smoothing on the positions of the nodes, (c) the mesh after smoothing, and (d) the mesh a $t = 60$ s using the monitor function (3.4).

## *3.4. Effect of $\tau$ on the moving mesh*

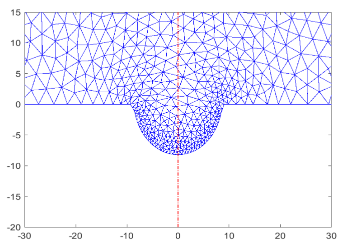
At the end of Section 2.3, we mentioned the MMPDE relaxation parameter $\tau$, and here we demonstrate the effect of $\tau$ on the moving mesh. Instead of forcing exact equidistribution for each time $t$, we relax the condition and require equidistribution at time $t + \tau$. Hence, the smaller the size of $\tau$, the quicker the mesh will react to changing features in the solution. To demonstrate this effect, we start from a uniform initial grid and show the resulting meshes after $t = 60$ s using three values of $\tau$, as shown in Figure 10. We observe a greater concentration of nodes near the pit boundary for smaller values of $\tau$. Larger values of $\tau$ lead to stretched elements; the mesh is not able to keep up with the changing computational domain. This relaxation does come at a cost, however, as smaller values of $\tau$ require more time steps for the integration of the MMPDE. In practice, one should select $\tau$ in tandem with mesh density parameters, choosing the largest value of $\tau$ which allows a balance of computational cost and mesh quality.



**(a)** Using $\tau = 10^{-2}$.



**(b)** Using $\tau = 10^{-4}$.

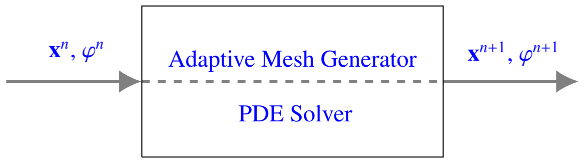


**(c)** Using $\tau = 10^{-6}$.

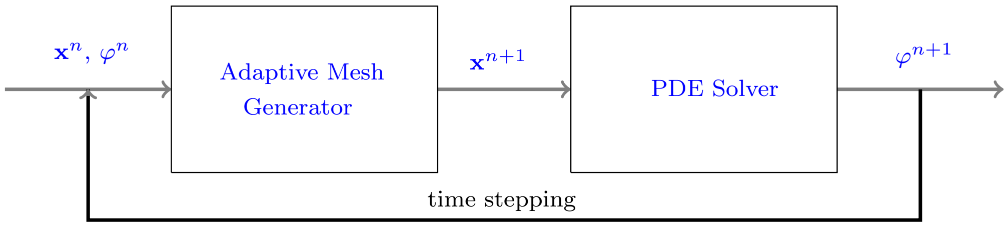
**Figure 10.** Effect of $\tau$ on the mesh after 120 s with the monitor function (3.4) using $\mu_1 = 100$ and $\mu_2 = 1$.

### 3.5. *Alternating mesh and physical PDE iteration*

There are two approaches that can be used to solve the coupled physical PDE and mesh equation: a simultaneous or an alternating approach. In a simultaneous approach, the discrete physical PDE and the discrete mesh equation provide a fully coupled system for both the mesh and solution unknowns, as shown in Figure 11a. The disadvantage of this approach is the highly nonlinear coupling between the physical solution and the mesh, resulting in a potentially difficult, large discrete system.



**(a)** The simultaneous solution approach.



**(b)** The alternating solution approach.

**Figure 11.** The (a) simultaneous and (b) alternating approaches to solve the coupled corrosion model and mesh PDE.

On the other hand, the alternating solution approach generates the mesh $X_{n+1}$ at time $t_{n+1}$ using the physical solution $\varphi^n$ and the mesh $X_n$ at time $t_n$. Then the solution $\varphi^{n+1}$ at the new time level is computed, as shown in Figure 11b. In this approach, there may be a lag between the solution and the mesh. Generally, this does not create any difficulties if the time step is reasonably small. The main advantages of the alternating approach are: (i) the mesh generation code is not directly coupled to the physical PDE solve, thereby increasing flexibility and reusability of code, (ii) the mesh PDE and physical PDE solvers can be developed and optimized in a modular way, and hence, (iii) the individual mesh and physical PDE solvers are more efficient. MMPDElab uses this alternating approach.

The defaults in MMPDElab, which we maintain, use implicit time steppers for both the physical

PDE and the MMPDE. The physical PDE is stepped forward in time using RADAU IIA, an implicit Runge-Kütta methods with great stability problems for stiff problems. The MMPDE is integrated using ODE15s, a variable order backward differentiation appproach.

Given a mesh a time $t = t_n$, $X_n$, and the solution on that mesh, the boundary of the pit is updated using (2.6) to give $X_{n+1}$ using a chosen time step $\Delta_t$. The time steppers for the physical and mesh PDEs use local error control to update the physical solution and mesh over the interval $[t_n, t_n + \Delta t]$. Hence these solvers may use time steps smaller than $\Delta t$ to achieve the required error tolerances.

### *3.6. Solution of the moving boundary value problem*

The flowchart in Figure 12 outlines the implementation of our computational pitting corrosion model using the MMPDElab framework. The first two steps are the same as the alternating step approach given in Figure 11b. The pit boundary is then moved based on the new positions of the adaptive mesh, followed by the movement of the corners of the pit. These last steps are detailed in the following section.

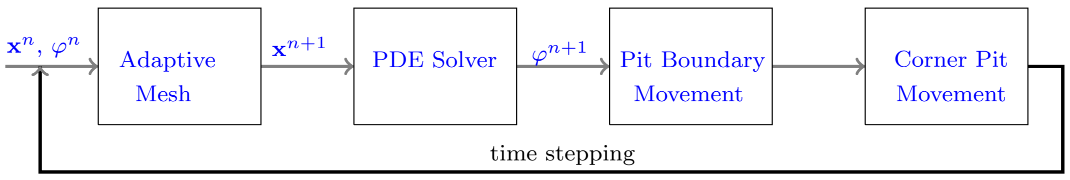


**Figure 12.** Flow chart for the physical PDE solve, the mesh PDE solve, and the pit boundary movement.

### *3.7. Details of the pit boundary movement*

To get the new position of the pit, we have to specify a direction and magnitude of movement for each node on the boundary of the pit, as well as the appropriate movement for each corner of the pit. A pit corner is a vertex, which is part of the pit boundary, and has a $y$–coordinate of zero.

As shown in Figure 13, a face normal is the outward pointing vector perpendicular to an edge or segment joining vertices. Taking the average of two face normals on adjacent edges gives us the vertex normal for the vertex between those edges. The vertex normals give the direction of movement for the pit boundary.

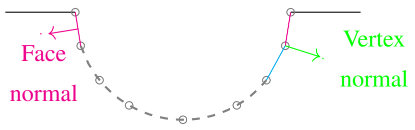


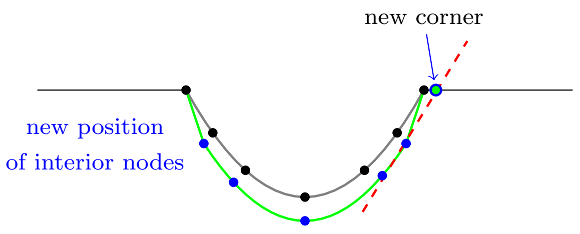
**Figure 13.** Definitions of face and vertex normals.

As mentioned in Section 2.1, the magnitude of the normal velocity of each vertex on the boundary of the pit is given by (2.7), which we may write as

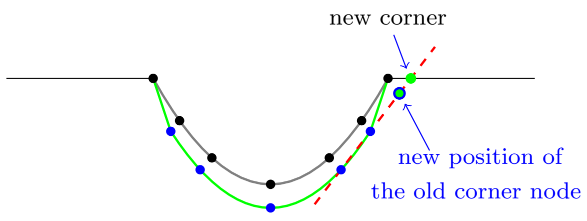
$$V_n = \frac{1}{c_{\text{solid}}} \cdot A_{\text{diss}} \cdot e^{\left(\frac{zF(V_{\text{corr}}+\alpha(V_{\text{app}}-V_{\text{corr}}-\varphi))}{RT}\right)}. \tag{3.5}$$

Once the vertices on the pit boundary are moved, the location of the corner nodes for the pit are updated using the following procedure. A linear extrapolation of the edge joining the two vertices that are closest to the corner and lie on the pit boundary is computed. The new corner location is given by the intersection of this line and $y = 0$, as necessary.

There are several situations for corner movement which may arise as shown in Figure 14. If the new corner is close to the old corner (Figure 14a), then no further changes are required. If the new corner is not close to the old corner (Figure 14b), then the old corner is moved into the pit by interpolating this same extrapolation line. The old corner and the new corner are considered to be not close if the distance between them is greater than 20% of the minimum altitude of all the elements in the mesh. The idea here is to support large movements of the pit boundary by moving grid points along $y = 0$ into the pit boundary. Large movements of the pit boundary could lead to other scenarios. Depending on material structure, the magnitude of this movement is controlled by the time step used in the update of the pit boundary. For example, if the intersection point is to the left of the original right corner of the pit, then we do not move the original corner. We note that, depending on material structure, the magnitude of this movement can be controlled by the time step used in the update of the pit boundary.



**(a)** New corner remains on $y = 0$.



**(b)** Corner node movement onto the pit boundary.

**Figure 14.** Updating the corner position: (a) the corner is moved to its new location along $y = 0$ or (b) the (old) corner is moved onto the boundary of the pit.

### 3.8. *Merging pits*

As multiple pits evolve, two pits may merge and form a single, larger pit. We visualize the pit merging procedure in Figure 16. We choose a tolerance to detect if a pit merge is necessary; this tolerance is calculated as $2 \times \max(V_n)\Delta t$. We then compute the length, $L$, of the element edges situated along $y = 0$ and in between the two pits. If $L$ is smaller than this tolerance, as in Figure 16a, then we go ahead and merge the pits. The left and right endpoints of that edge are tagged with red and black, respectively. In Figure 16b, these two points merge to a single point. In order to avoid changing the number of mesh nodes and mesh topology, either the black or the red node in Figure 16a has to move into the pit boundary. For example, the black and red nodes can merge, creating a new red node (see Figure 16b), and the black node will move half-way between the red and green nodes. Alternately, the black and red node merge, creating a new black node and the red node moves half-way between the red node and green node (not shown). To determine which vertex (red or black) moves into the pit boundary we choose the vertex corresponding to the larger angle in the element whose bottom edge is the single edge between the pits, see Figure 15. Once the merge has occurred, a mesh smoothing procedure is used to obtain the mesh shown in Figure 16c. We can see that the mesh smoothing has evened out the size and shape of the elements to the right of the red apex node.

For each time step after the merge, the location of the apex is obtained as the intersection point of the linear extrapolations of the second last edges to the left and right of the apex.

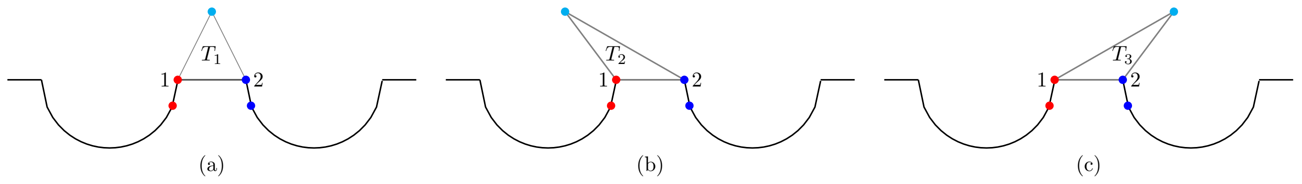
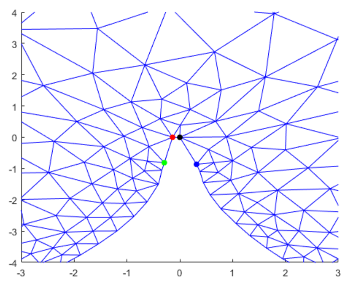
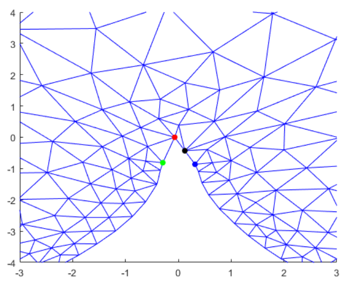


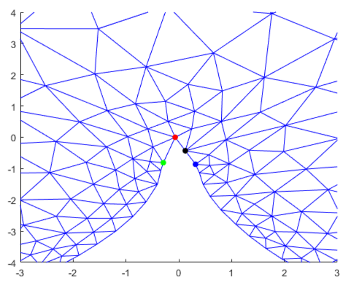
**Figure 15.** Three possible element orientations between the pits at the time of a merge.



**(a)** A pit merge is initiated.



**(b)** A merge with no mesh topology change.



**(c)** Post merge mesh smoothing.

**Figure 16.** The pit merging process.

## 4. Numerical results

Based on the simple experiments in Section 3, throughout this numerical results section a constant value of $\tau = 10^{-5}$ is used in the MMPDE, and the constants $\mu_1 = 100$ and $\mu_2 = 1$ are used as default values for the mesh density function (3.4). The initial number of mesh points inside the pit is set to 61.

### *4.1. Single pit simulations*

We begin by using our computational framework to compare the evolution of a single pit in three cases: a homogeneous solid material (without a crystal direction), a solid material with a specified crystal direction, and a solid material with a discontinuity in the crystal direction. All simulations use an initial mesh constructed by smoothing the non-uniform mesh generated using **initmesh** as discussed in Section 3.3.

Figures 17(a–d) show the final pit geometry and final meshes for a homogeneous material, a material with crystal direction [001], a material with crystal direction [101], and a material with a discontinuity in crystal directions, [001] to the left of $x = 0$ and [101] to the right of $x = 0$.

It is observed in Figure 17a that the final shape of the homogeneous pit is the same as the initial pit since the chosen $V_{corr}$ has the same value, $-0.24V$, in every direction within the pit. Hence, from equation (3.5), the normal velocity at all locations within the pit will be equal. The situation is not the same for pits with a crystalline structure since $V_{corr}$ will vary with crystal direction. For example, for crystal directions of the forms $\langle 001\rangle$, $\langle 011\rangle$ and $\langle 111\rangle$, $V_{corr}$ will have values of -0.2297 V, -0.2455 V, and -0.2525 V, respectively. Again from equation (3.5), we see that the normal velocity is greater for lower magnitude $V_{corr}$ values; that is, $V_n\,(111) < V_n\,(011) < V_n\,(001)$. The effect of this $V_{corr}$ dependent velocity is displayed in Figure 17b, where we observe that the sides of the pit have become straight and angled 90 degrees with one another (when axes are equally scaled). This behaviour is expected.

As shown in Figure 5, for a crystal oriented with a zone axis along [001], $\langle 001\rangle$ directions are located along the horizontal and vertical axes and $\langle 011\rangle$ directions midway in between. Thus, we expect that the [100] and $[0\bar{1}0]$ directions will move faster than the $\left[1\bar{1}0\right]$ direction. As the faster locations on the pit boundary move, their orientation will change and eventually become the same direction as the slowest moving axis, in this case $\left[1\bar{1}0\right]$. For all future times, the sides of the pit will move outward perpendicular to these two lines while maintaining the same angular relationship. We observe the same effect in Figure 17c where the crystal has been oriented along a zone axis of [101]. In this case, the slowest moving directions are along $\langle 111\rangle$, and the angle between the $\left[11\bar{1}\right]$ and $\left[\bar{1}11\right]$ planes agrees with the expected value of 70.5 degrees. Figure 17d displays the final pit shape where there is a discontinuity in the crystal directions, and the left and right sides of the crystal are oriented along zone axes of [001] and [101], respectively. The left and right sides of the pit are straight lines moving along $\left[\bar{1}\bar{1}0\right]$ and $\left[\bar{1}11\right]$ directions, respectively.

### *4.2. Multiple pit simulations*

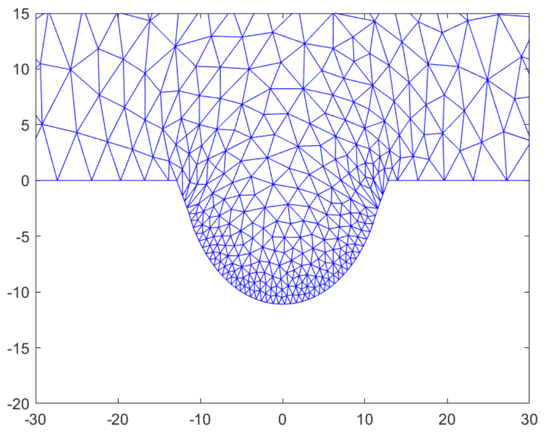
As mentioned previously, if multiple pits exist in a material and the pits grow large enough, there is the potential that the pits will merge during the simulation. The imminent merge needs to be detected, the boundaries of each of the previously isolated pits need to be updated in a way which avoids boundary crossing, and the mesh around the merge location needs to be adjusted smoothly and without topology changes. Once complete, the merged pit is then treated as one larger pit, and the evolution continues. See Section 3.8 for details.

To demonstrate the robustness of our adaptive simulation framework for the evolution of multiple pits, we start with two pits relatively close together in a homogeneous material, as shown in plot Figure 18a. A quality initial mesh which concentrates nodes near the boundary of both pits is generated as discussed in Section 3.3. As corrosion continues, the pits grow, and hence the pit boundaries grow closer together. The pits then merge and continue to evolve as shown in Figure 18b. Figure 18c provides the result of a similar simulation with a material oriented in the [101] crystal direction. The crystal direction clearly affects the geometry of the merged pit. Figure 18d shows the resulting pit geometry and associated mesh for merged pits in a material with two crystal directions, where the discontinuity in crystal direction is located at $x = 0$.

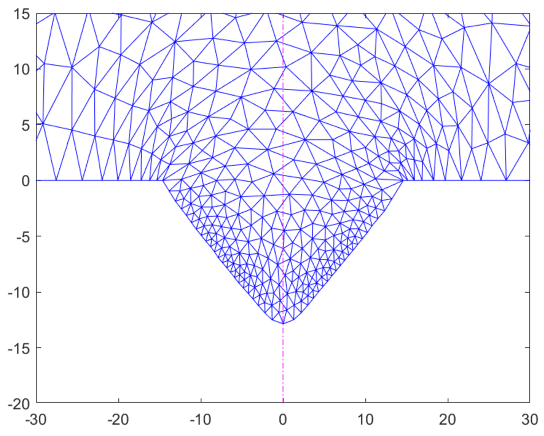
We now more closely study the pit depth and width as a function of time for a single homogeneous pit, a single crystal oriented with a zone axis along [001], and two crystals with an interface at $x = 0$, where the left and right crystals are oriented along [001] and [101] zone axes, respectively. Recall, the initial and final pit configurations for these three situations are displayed in Figures 17a, 17b, and 17d, respectively. Plotting pit depth and width as a function of time leads to nonlinear curves as shown in Figures 19a and 19b. It has been common practice to fit corrosion loss curves using a power-law equation, and for the initial stages of corrosion, it seems to work well, see [57]. Since the loss of material due to corrosion is a function of the dimensions of the pit, it is expected that the same power-law behaviour should hold for our pit depth and width data. The model we use is

$$\text{depth(t)} \ \ \text{(or)} \ \ \text{width(t)} = at^b + c,$$

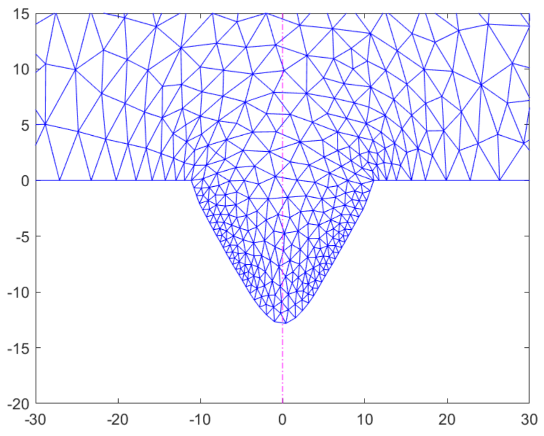
where $a$, $b$ and $c$ are fitting parameters and the initial pit is defined when $t = 1$. The curve fits were excellent and the fitting parameters found are presented in Table 2. The initial pit width and depth



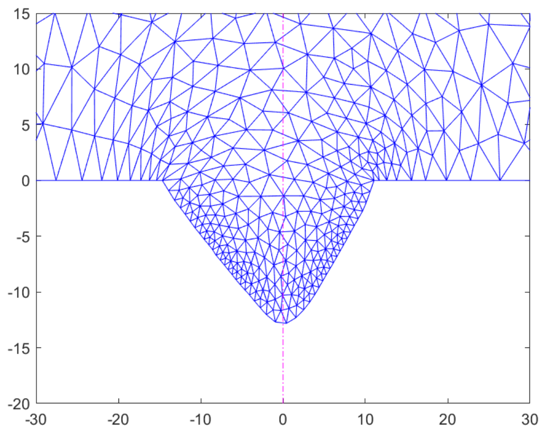
**(a)** Mesh for a homogeneous crystal.



**(b)** Mesh for a crystal with direction [001].



**(c)** Mesh for a crystal with direction [101].

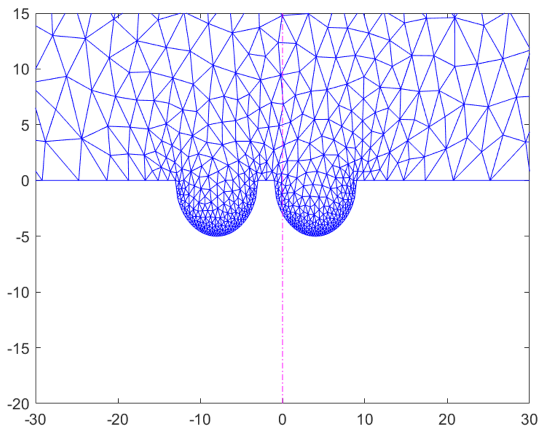


**(d)** Mesh for a crystal with two directions [001] and [101].

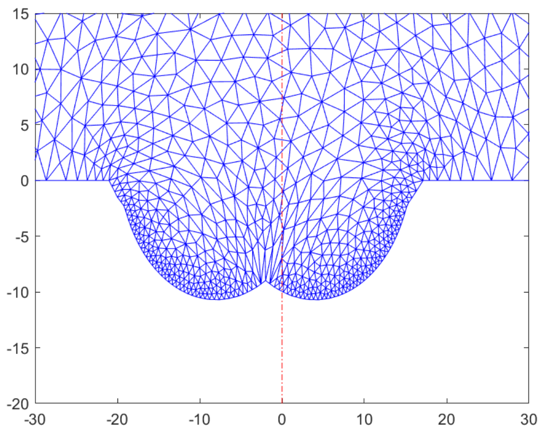
**Figure 17.** Pit configurations and meshes at $t = 120$ s for a) a homogeneous material, b) a single crystal oriented with a zone axis along [001], c) a single crystal oriented with a zone axis along [101], and d) a crystal with an interface at $x = 0$ where the crystal directions to the left and right of $x = 0$ are [001] and [101], respectively.

**Table 2.** Power-law model fitting parameters for the 6 curves presented in Figure 19. The numbers in brackets represent uncertainty in the last significant digit.

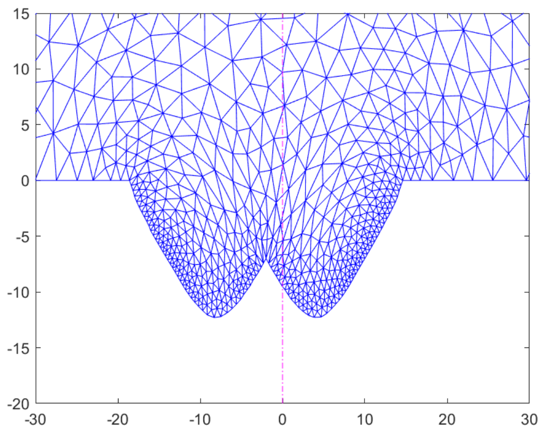
| **Case** | **Width** | | | **Depth** | | |
|---|---|---|---|---|---|---|
| | a | b | c | a | b | c |
| Homogeneous | 0.142(2) | 0.980(3) | 9.83(2) | 0.076(1) | 0.917(3) | 4.95(1) |
| [001] | 0.243(8) | 0.907(6) | 9.62(5) | 0.116(3) | 0.886(4) | 4.89(2) |
| [001]/[101] | 0.181(5) | 0.927(6) | 9.74(4) | 0.121(3) | 0.877(6) | 4.89(2) |



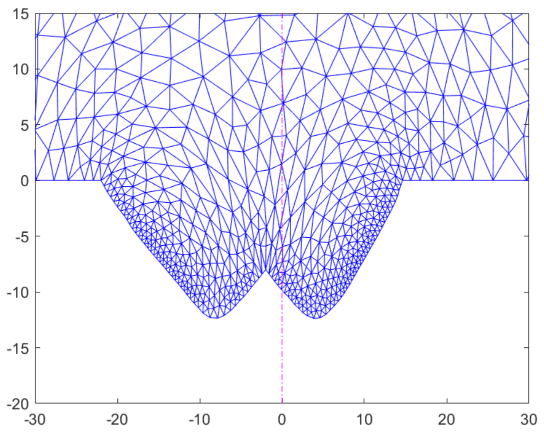
**(a)** Initial mesh for two pits.



**(b)** Mesh for merged pits at $t = 120$ s.



**(c)** Mesh for merged pits for a material with a single crystal direction [101] at $t = 120$ s.



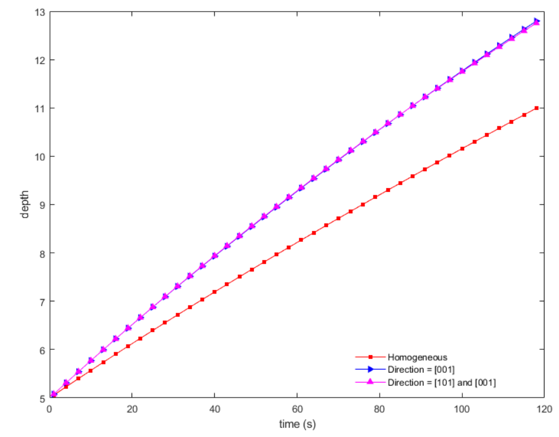
**(d)** Mesh for merged pits for a material with two crystal directions, [001] if $x < 0$ and [101] if $x > 0$, at $t = 120$ s.

**Figure 18.** Pit evolution and adaptive mesh generation for merging multiple pits for three material configurations.

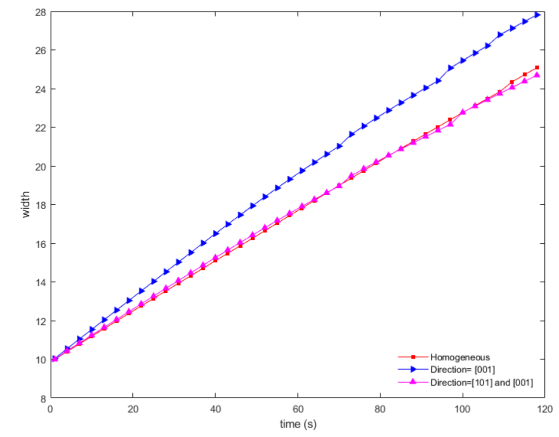
were 10 microns and 5 microns, respectively, and these values are close to the value of $a + c$; the initial dimension of the pit predicted from the fitting procedure. It is reassuring to note that the modelled pitting corrosion behaviour follows an expression used to fit experimental corrosion losses.

Figure 20 displays the pit depths and pit widths for various initial numbers of mesh points using both the moving mesh approach and the re-meshing technique at each time step. For a lower number of initial mesh points (i.e., INPPit = 31) on the pit, the results do not match the pit depth and width obtained by using the remeshing technique at every time step as time increases. However, with more than 50 initial mesh points on the pit, the pit depths and pit widths for both approaches converge to the same solution. Thus, we can conclude that the adaptive moving mesh approach provides reasonable solution accuracy by delivering precise pit depth and width measurements.

Figure 21 shows the numerical error in pit depths and widths when comparing the remeshing

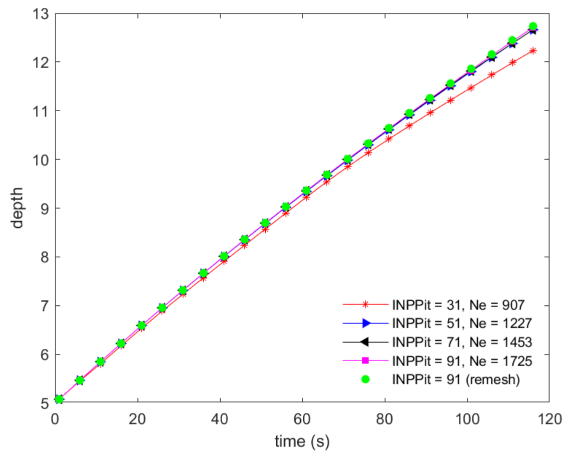


**(a)** Pit-depth over time for homogeneous and non-homogeneous crystals.

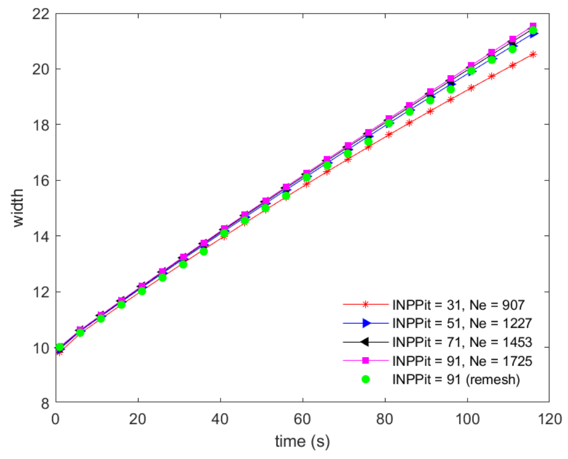


**(b)** Pit-width over time for homogeneous and non-homogeneous crystals.

**Figure 19.** Pit-depths and widths for homogeneous and non-homogeneous crystals.



**(a)** Pit-depth over time.



**(b)** Pit-width over time.

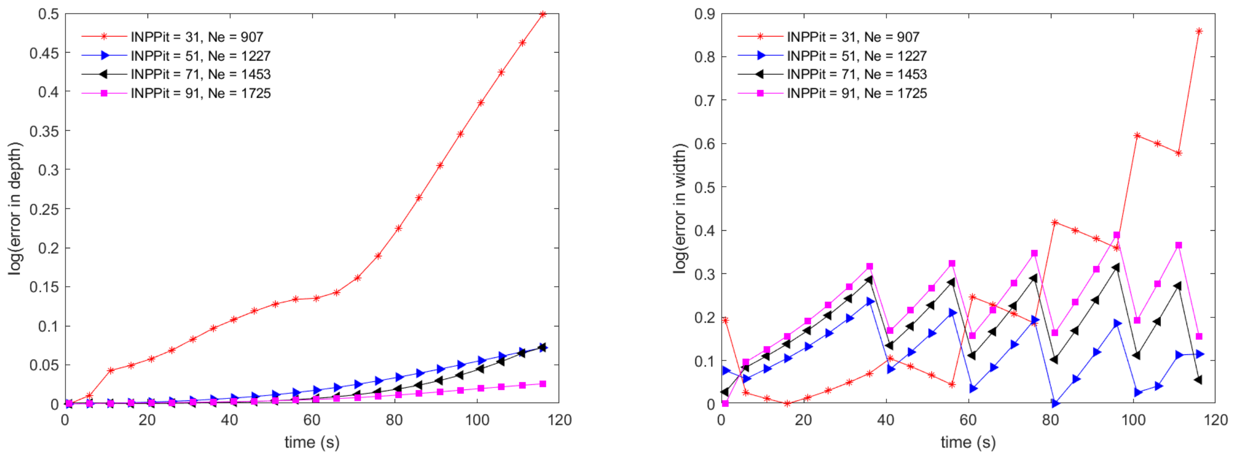
**Figure 20.** Pit-depths and widths for the homogeneous case for various numbers of initial mesh point in the pit.

technique with the adaptive moving mesh approach for various initial numbers of mesh points on the pit. The absolute error is calculated by comparing the remeshing solution with the moving mesh solution at every step. For a lower number of initial mesh points (i.e., INPPit = 31) on the pit, the error increases over time. However, with more than 50 initial mesh points on the pit, the error is controlled.

Assessing mesh quality is relatively complicated. We follow [22] and assess the equidistribution quality measure

$$Q_{\mathrm{eq}}(K) = \frac{N\rho_K|K|}{\sigma_h}, \qquad \text{for all} \quad K \in \mathcal{T}_h.$$

Note, $\max_{K\in\mathcal{T}_h} Q_{\mathrm{eq}}(K) = 1$ would indicate a grid providing perfect equidistribution of the mesh density function. Of course, the ability to achieve perfect equidistribution is affected by the space and time



**(a)** Error in pit depth. **(b)** Error in pit width.

**Figure 21.** The numerical error for pit depths and widths for various initial numbers of mesh points on the pit.

discretization, and the relaxation of equidistribution and choice of parameters in the MMPDE. For the simulations depicted in Figure 17, we obtain $\max_{K\in\mathcal{T}_h} Q_{\text{eq}}$ values of approximately $2.89, 2.86, 2.92$ and $2.84$ at the end of the simulations.

These numerical results suggest that the adaptive moving mesh is a suitable choice for solving the moving boundary problem related to pitting corrosion.

## 5. Conclusions

We have presented a robust, fully automatic, moving mesh solution framework for pitting corrosion. The moving mesh approach continuously and smoothly evolves a fixed mesh topology according to changing pit geometry. Single and multiple pits are considered, as are materials with different crystal direction(s). A procedure is presented which allows pits to merge without a change in mesh topology, allowing computation to proceed without restarting the computation.

The simulation of large pit growth or the initiation of many pits, would likely benefit from an *hr*–refinement strategy (which not only redistributes nodes as we have presented here, but also allows periodic changes to the number of mesh nodes) coupled with a domain decomposition approach to allow the problem to be spatially partitioned and the computation distributed to harness additional processors. This will be the subject of future work. Current work includes extending the computational framework to allow for more heterogenous materials, with corrosive resistant “pockets” or holes or voids.

## Data availability

The raw or processed data required to reproduce these figures and findings cannot be shared at this time due to technical or time limitations, but are available from the authors upon request.

## Author contributions

Abu Naser Sarker: Writing – original draft; Ronald D. Haynes and Michael D. Robertson: Conceptualization, Supervision, Writing – review & editing. All authors have read and agreed to the published version of the manuscript.

## Use of Generative-AI tools declaration

We have not used Artificial Intelligence (AI) tools in the creation of this article.

## Acknowledgments)

The work of Ronald D. Haynes and Abu N. Sarker has been supported by a NSERC discovery grant (Canada).

## Conflict of interest

We have no competing interests to declare in this paper.

## References

1. X. Yu, F. Wan, Y. Guo, Micromechanics modeling of skin panel with pitting corrosion for aircraft structural health monitoring, *2016 IEEE International Conference on Prognostics and Health Management (ICPHM)*, (2016), 1–8. IEEE. https://doi.org/10.1109/ICPHM.2016.7542831
2. L. B. Simon, M. Khobaib, T. E. Matikas, C. Jeffcoate, M. Donley, Influence of pitting corrosion on structural integrity of aluminum alloys, *Nondestructive Evaluation of Aging Materials and Composites III*, **3585** (1999), 40–47. SPIE. https://doi.org/10.1117/12.339861
3. S. Sharland, A review of the theoretical modelling of crevice and pitting corrosion, *Corros. Sci.*, **27** (1987), 289–323. https://doi.org/10.1016/0010-938X(87)90024-2
4. A. S. H. Makhlouf, V. Herrera, E. Muñoz, Corrosion and protection of the metallic structures in the petroleum industry due to corrosion and the techniques for protection, *Handbook of Materials Failure Analysis*, (2018), 107–122. Elsevier. https://doi.org/10.1016/B978-0-08-101928-3.00006-9
5. P. R. Roberge, *Corrosion Engineering*, McGraw-Hill Education, 2008.
6. F. Cattant, D. Crusset, D. Féron, Corrosion issues in nuclear industry today, *Mater. Today*, **11** (2008), 32–37. https://doi.org/10.1016/S1369-7021(08)70205-0
7. V. G. DeGiorgi, N. Kota, A. C. Lewis, S. M. Qidwai, Numerical modeling of pit growth in microstructure, *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, **55850** (2013), 1–7. American Society of Mechanical Engineers. https://doi.org/10.1115/DETC2013-12074
8. S. Sharland, A mathematical model of crevice and pitting corrosion—II. the mathematical solution, *Corros. Sci.*, **28** (1988), 621–630. https://doi.org/10.1016/0010-938X(88)90028-5

9. S. Scheiner, C. Hellmich, Finite volume model for diffusion-and activation-controlled pitting corrosion of stainless steel, *Comput. Method. Appl. Mech. Eng.*, **198** (2009), 2898–2910. https://doi.org/10.1016/j.cma.2009.04.012
10. N. J. Laycock, D. P. Krouse, S. C. Hendy, D. E. Williams, Computer simulation of pitting corrosion of stainless steels, *The Electrochemical Society Interface*, **23** (2014), 65–71. https://doi.org/10.1149/2.F05144IF
11. D. Krouse, N. Laycock, C. Padovani, Modelling pitting corrosion of stainless steel in atmospheric exposures to chloride containing environments, *Corros. Eng. Sci. Techn.*, **49** (2014), 521–528. https://doi.org/10.1179/1743278214Y.0000000221
12. D. De Meo, E. Oterkus, Finite element implementation of a peridynamic pitting corrosion damage model, *Ocean Eng.*, **135** (2017), 76–83. https://doi.org/10.1016/j.oceaneng.2017.03.002
13. J. C. Walton, Mathematical modeling of mass transport and chemical reaction in crevice and pitting corrosion, *Corros. Sci.*, **30** (1990), 915–928. https://doi.org/10.1016/0010-938X(90)90013-U
14. S. Sharland, P. Tasker, A mathematical model of crevice and pitting corrosion-I. The physical model, *Corros. Sci.*, **28** (1988), 603–620. https://doi.org/10.1016/0010-938X(88)90027-3
15. S. Jafarzadeh, Z. Chen, F. Bobaru, Computational modeling of pitting corrosion, *Corros. Rev.*, **37** (2019), 419–439. https://doi.org/10.1515/corrrev-2019-0049
16. N. Kota, S. M. Qidwai, A. C. Lewis, V. G. DeGiorgi, Microstructure-based numerical modeling of pitting corrosion in 316 stainless steel, *ECS Transactions*, **50** (2013), 155–164. https://doi.org/10.1149/05031.0155ecst
17. R. Duddu, Numerical modeling of corrosion pit propagation using the combined extended finite element and level set method, *Comput. Mech.*, **54** (2014), 613–627. https://doi.org/10.1007/s00466-014-1010-8
18. A. Turnbull, D. Horner, B. Connolly, Challenges in modelling the evolution of stress corrosion cracks from pits, *Eng. Fract. Mech.*, **76** (2009), 633–640. https://doi.org/10.1016/j.engfracmech.2008.09.004
19. S. Scheiner, C. Hellmich, Stable pitting corrosion of stainless steel as diffusion-controlled dissolution process with a sharp moving electrode boundary, *Corros. Sci.*, **49** (2007), 319–346. https://doi.org/10.1016/j.corsci.2006.03.019
20. K. Wang, C. Li, Y. Li, J. Lu, Y. Wang, X. Luo, Multi-physics coupling analysis on the time-dependent localized corrosion behavior of carbon steel in $CO_2$-$H_2O$ environment, *J. Electrochem. Soc.*, **167** (2019), 013505–013505. https://doi.org/10.1149/2.0052001JES
21. J. Donea, A. Huerta, J.-P. Ponthot, A. Rodríguez-Ferran, Arbitrary Lagrangian-Eulerian methods, *Encyclopedia of Computational Mechanics*, (2004).
22. W. Huang, R. D. Russell, *Adaptive Moving Mesh Methods*, vol. 174. Springer Science & Business Media, 2010.
23. C. W. Hirt, A. A. Amsden, J. Cook, An arbitrary Lagrangian-Eulerian computing method for all flow speeds, *J. Comput. Phys.*, **14** (1974), 227–253. https://doi.org/10.1016/0021-9991(74)90051-5

24. C. Hirt, A. Amsden, J. Cook, An arbitrary Lagrangian-Eulerian computing method for all flow speeds, *J. Comput. Phys.*, **135** (1997), 203–216. https://doi.org/10.1006/jcph.1997.5702

25. L. G. Margolin, Introduction to "an arbitrary Lagrangian-Eulerian computing method for all flow speeds", *J. Comput. Phys.*, **135** (197), 198–202. https://doi.org/10.1006/jcph.1997.5727

26. F. Nobile, L. Formaggia, A stability analysis for the arbitrary Lagrangian Eulerian formulation with finite elements, *East-West Journal of Numerical Mathematics*, **7** (1999), 105–132.

27. P. M. Knupp, L. G. Margolin, M. Shashkov, Reference Jacobian optimization-based rezone strategies for arbitrary Lagrangian Eulerian methods, *J. Comput. Phys.*, **176** (2002), 93–128. https://doi.org/10.1006/jcph.2001.6969

28. B. Wells, M. J. Baines, P. Glaister, Generation of arbitrary Lagrangian–Eulerian (ALE) velocities, based on monitor functions, for the solution of compressible fluid equations, *Int. J. Numer. Meth. Fl.*, **47** (2005), 1375–1381. https://doi.org/10.1002/fld.915

29. J. F. Thompson, F. C. Thames, C. W. Mastin, Automatic numerical generation of body-fitted curvilinear coordinate system for field containing any number of arbitrary two-dimensional bodies, *J. Comput. Phys.*, **15** (1974), 299–319. https://doi.org/10.1016/0021-9991(74)90114-4

30. A. M. Winslow, Numerical solution of the quasilinear Poisson equation in a nonuniform triangle mesh, *J. Comput. Phys.*, **1** (1966), 149–172. https://doi.org/10.1016/0021-9991(66)90001-5

31. A. M. Winslow, *Adaptive-mesh zoning by the equipotential method*, tech. rep., Lawrence Livermore National Lab., CA (USA), 1981. https://doi.org/10.2172/6227449

32. J. U. Brackbill, J. S. Saltzman, Adaptive zoning for singular problems in two dimensions, *J. Comput. Phys.*, **46** (1982), 342–368. https://doi.org/10.1016/0021-9991(82)90020-1

33. O.-P. Jacquotte, A mechanical model for a new grid generation method in computational fluid dynamics, *Comput. Method. Appl. Mech. Eng.*, **66** (1988), 323–338. https://doi.org/10.1016/0045-7825(88)90005-9

34. O.-P. Jacquotte, Generation, optimization and adaptation of multiblock grids around complex configurations in computational fluid dynamics, *Int. J. Numer. Meth. Eng.*, **34** (1992), 443–454. https://doi.org/10.1002/nme.1620340204

35. O.-P. Jacquotte, G. Coussement, Structured mesh adaption: space accuracy and interpolation methods, *Comput. Method. Appl. Mech. Eng.*, **101** (1992), 397–432. https://doi.org/10.1016/0045-7825(92)90031-E

36. P. M. Knupp, Mesh generation using vector fields, *J. Comput. Phys.*, **119** (1995), 142–148. https://doi.org/10.1006/jcph.1995.1122

37. P. M. Knupp, Jacobian-weighted elliptic grid generation, *SIAM J. Sci. Comput.*, **17** (1996), 1475–1490. https://doi.org/10.1137/S1064827594278563

38. W. Huang, R. D. Russell, A high dimensional moving mesh strategy, *Appl. Numer. Math.*, **26** (1998), 63–76. https://doi.org/10.1016/S0168-9274(97)00082-2

39. W. Cao, W. Huang, R. D. Russell, A study of monitor functions for two-dimensional adaptive mesh generation, *SIAM J. Sci. Comput.*, **20** (1999), 1978–1994. https://doi.org/10.1137/S1064827597327656

40. W. Huang, Variational mesh adaptation: isotropy and equidistribution, *J. Comput. Phys.*, **174** (2001), 903–924. https://doi.org/10.1006/jcph.2001.6945

41. Y. Ren, R. D. Russell, Moving mesh techniques based upon equidistribution, and their stability, *SIAM Journal on Scientific and Statistical Computing*, **13** (1992), 1265–1286. https://doi.org/10.1137/0913072

42. W. Huang, Y. Ren, R. D. Russell, Moving mesh partial differential equations (MMPDES) based on the equidistribution principle, *SIAM J. Numer. Anal.*, **31** (1994), 709–730. https://doi.org/10.1137/0731038

43. W. Huang, R. D. Russell, Moving mesh strategy based on a gradient flow equation for two-dimensional problems, *SIAM J. Sci. Comput.*, **20** (1998), 998–1015. https://doi.org/10.1137/S1064827596315242

44. W. Cao, On the error of linear interpolation and the orientation, aspect ratio, and internal angles of a triangle, *SIAM J. Numer. Anal.*, **43** (2005), 19–40. https://doi.org/10.1137/S0036142903433492

45. W. Huang, An introduction to MMPDElab, *ArXiv Preprint ArXiv:1904.05535*, 2019.

46. C. Xie, F. Wei, P. Wang, H. Huang, Modeling of corrosion pit growth for prognostics and health management, *Prognostics and Health Management (PHM), 2015 IEEE Conference on*, (2015), 1–7. IEEE. https://doi.org/10.1109/ICPHM.2015.7245024

47. S. Sharland, C. Jackson, A. Diver, A finite-element model of the propagation of corrosion crevices and pits, *Corros. Sci.*, **29** (1989), 1149–1166. https://doi.org/10.1016/0010-938X(89)90051-6

48. A. J. Bard, L. R. Faulkner, J. Leddy, C. G. Zoski, *Electrochemical Methods: Fundamentals and Applications*, vol. 2, Wiley New York, 1980.

49. A. J. Bard, L. R. Faulkner, H. S. White, *Electrochemical Methods: Fundamentals and Applications*, John Wiley & Sons, 2022.

50. S. Chattoraj, L. Shi, C. C. Sun, Understanding the relationship between crystal structure, plasticity and compaction behaviour of theophylline, methyl gallate, and their 1: 1 co-crystal, *CrystEngComm*, **12** (2010), 2466–2472. https://doi.org/10.1039/c000614a

51. M. B. Boisen, G. V. Gibbs, *Mathematical Crystallography*, vol. 15, Walter de Gruyter GmbH & Co KG, 2018.

52. M. Robertson, K. Raffel, Imaging crystals, *Microscopical Society of Canada Bulletin*, **36** (2008), 13–18.

53. W. Huang, Metric tensors for anisotropic mesh generation, *J. Comput. Phys.*, **204** (2005), 633–665. https://doi.org/10.1016/j.jcp.2004.10.024

54. D. R. Adams, L. I. Hedberg, *Function Spaces and Potential Theory*, vol. 314, Springer Science & Business Media, 1999.

55. G. Beckett, J. A. Mackenzie, M. Robertson, A moving mesh finite element method for the solution of two-dimensional Stefan problems, *J. Comput. Phys.*, **168** (2001), 500–518. https://doi.org/10.1006/jcph.2001.6721

56. J. Mackenzie, M. Robertson, The numerical solution of one-dimensional phase change problems using an adaptive moving mesh method, *J. Comput. Phys.*, **161** (2000), 537–557. https://doi.org/10.1006/jcph.2000.6511

57. R. E. Melchers, Predicting long-term corrosion of metal alloys in physical infrastructure, *npj Mat. Degrad.*, **3** (2019), 1–7. https://doi.org/10.1038/s41529-018-0066-x

AIMS Press