Chapter 1

Prologue

In this thesis we discuss and develop *adaptive* multigrid solvers for space–time discretisations of parabolic reaction diffusion equations with a potentially nonlinear forcing term. The latter present a broad class of partial differential equations that can be written in the form

$$\partial_t u - \operatorname{div}(D(x)\nabla u) = f(u) \tag{1.1}$$

for some $u = u(x,t)$ in a space domain over a time interval. We will postpone more rigorous definitions to the following chapters and for now simply assume the problem to be well posed which includes appropriately defined boundary conditions. This type of partial differential equation is used to describe a variety of physical phenomena. In its simplest form, we have a zero source term, i.e. $f = 0$. The resulting heat equation describes the variation of temperature in the space domain over time starting from a set of initial conditions. Other important applications with non-zero source terms, that can be described by (1.1), are chemical reactions over time, the development of populations in [1] or the propagation of wavefronts [2].

A particular instance of a traveling wave which we will particularly focus on and which originally motivated the topic of this thesis, is the propagation of electric signals in heart tissue. It can be modeled using the so-called monodomain equations which are also a reaction-diffusion system [3]. The contraction of our heart is governed by an electric impulse whose charge distribution is a wavefront front through our cell tissue. When trying to numerically approximate such a process one faces a number of challenges. One major challenge that arises is the multiscale range in space and time [4]. The overall domain in space and time is very large compared to the characteristic length-scale on which the rapid local changes of the current potential occur. Thus one requires a high accuracy in time and space. Accurate discretisations result in large systems of equations involving large numbers of degrees of freedom. Solving them in an robust and efficient way is an active area of interest and research [5].

In general when trying to numerically approximate the solution of a partial differential equation there is no unique way to do so and hence many design choices have to be made. A frequently used possibility is the method of lines approach, where one first discretises in space e.g. using finite elements and where the time variable remains continuous, which results in a system of ordinary differential equations that is then to be solved by an appropriate method [source]. A very common approach is to use a time stepping method [source]. That is one computes an approximation for all space elements or nodes at a certain time t_n and then uses those results or even preceding ones to compute the approximate solution at the next time t_{n+1}. This is the natural way to perform operations, because this is also how we move through time, sequentially, causality implies that the solution at a given time depends on the previous one but not the other way around. However in current technological development where there is no further significant increase in CPU clockspeed, the only way to really achieve a gain in computational power is through an increase in the number of processors. Therefore for this to actually translate to a computational speed up one requires algorithms to be more and more parallelisable, that is to allow for more operations to be performed at the same time. The time stepping approach outlined above contains inherently sequential processes, since only the space dimensions allow for parallelisation. As this saturates [source] there is no possibility for a further speed up. Thus it makes sense to look for methods that utilise a parallelisation in space and time simultaneously. This in turn naturally leads to a space–time discretisation of the equation as a whole [6], which is also what we will be considering in this thesis, a large space–time system which we want to be able to mainly solve in parallel. A short discussion of the research done on this field so far, advantages and difficulties as well as some further references can be found in Section 3.1., while the particular discretisation we chose will be introduced in chapter 4.

It has been shown that large linear systems of equations are often most efficiently solved using iterative schemes [source]. Among them, multigrid methods represent an important and powerful class to approximate such solutions. In the case of sparse, symmetric, positive definite systems they even provide optimality in the sense that their complexity can be bounded by $O(N)$, where N is the number of degrees of freedom [7]. Unfortunately the behavior of multigrid algorithms in an indefinite or not symmetric setting is often not yet very well understood or not suitable [source], and convergence is generally not guaranteed [source]. Therefore we would like to aim for the construction of a system that can claim as many of these preferrable properties as possible. However most space-time solution methods do not give rise to symmetric positive definite systems [6] which is why we recast problem (1.1) as an optimisation probelm, an ansatz known as least squares finite element methods [8] and which will be first introduced in Section 3.4. It entails the construction of a minimisation problem whose solution coincides with the solution of the differential equation. Instead of solving the original problem we now apply a finite element approach in space-time to solve the auxiliary problem whose value for a given input u denotes an energy that we can minimise. In the linear case we are, due to the symmetry and positive definiteness of the system, guaranteed the existence of a global minimiser. In the nonlinear case we consider linearisations of the system which are generally not positive definite, but the symmtetry is maintained because of the commutativity of derivatives. The problem is non-convex but by successively reducing energy we can still find local minima. Hence for a nonlinear source or reaction term f, we additionally require an outer nonlinear iteration scheme which succuessively solves linearisations of the least squares functional. The non-linear solvers that were employed in the implementation section here are a damped Newton method [9] and a trust region method [10], and will be introduced in Section 3.2. We have convergence to a global minimiser in a neighbourhood of the solution, that is for a sufficiently good initial starting iterate the solution of the original problem is recovered.

Below we can see a schematic overview of how these beforementioned core concepts are tied together in order to give rise to a comprehensive solution method.

Overview of the different Steps towards an Approximate Solution

1. Reformulate (1.1) as a mimimisation problem J whose solution coincides with the one of the original equation.
2. Discretise the problem using a space-time finite element approach
3. Derive a nonlinear iteration scheme or energy minimisation method (e.g. Newton or Trust Region method) where we solve a simplified problem using the current iterative solution in each step
4. Solve the arising linear system of equations using a multigrid method
5. Repeat Step 4 with the updated solution until a stopping criterion is met

These are the main ingredients that we will tie together in this thesis in order to develop an efficient, robust and accurate solver to tackle problems of type (1.1). It is a rather novel construction that has, to our knowledge, not been studied in this context and will therefore require further investigations before drawing any final conclusions on its utility. The mathematical methodologies will be introduced more thoroughly in Chapter 3, where we will also explain the particular choice for each of them in more detail, attempting to make use of their favourable properties while trying to avoid the pitfalls. In Chapter 4 we derive a proper problem formulation, which we will then discretise in order to derive linear systems of equations to be solved iteratively. Afterwards we introduce multrigrid methods in Chapter 5, especially discussing the particularities that arise due to the construction presented in Chapter 4. Chapter 6 then contains the numerical results we obtained for various test cases and discusses certain behaviors we observed during our work which will then be followed by a conclusion and an outlook in Chapter 7.

In order to really obtain a meaningful solution u we need a number of properties to be fulfilled. In each nonlinear iteration step the multigrid solver has to converge to the solution of the linearised least squares minimisation problem. In the outer iteration we need the nonlinear iteration scheme to converge to the minimiser of our non-linear functional whose solution as mentioned above is supposed to correspond to the solution of the original problem. However we are not ensured global convergence since the problem is in general non-convex.

Overall we are aiming for a better understanding of the versatility of space-time least squares finite element approaches in general and in combination with multigrid methods. *A focus will be given to the construction of a particular algebraic multigrid method that takes intrinsic properties related to the monodomain equation into account, developing an equally accurate but more efficient way through an adapted coarse grid construction.* To allow for a better understanding of the processes involved in this particular application the following chapter will give a brief insight into the functioning of the human heart, the transmission of electric potential through tissue, the different charge distribution within or between cells or cellular structures and how this can be turned into a mathematical model.

Chapter 4

Derivation of a Least Squares Finite Element Space-Time Discretisation

4.1 Construction of an Equivalent Minimisation Problem

In this chapter we will tie the beforementioned concepts together to derive a discretised problem formulation and subsequently develop a methodology capable of approximating reaction diffusion equations. In order to do so let us first set the ground for the overall framework we are looking at. We consider a space-time domain

$$\Omega = \mathcal{S} \times \mathcal{T}, \quad \mathcal{T} = (0,T),\ T > 0 \text{ and } \mathcal{S} \subset \mathbb{R}^N,\ N = 1,2,3 \tag{4.1}$$

where $\mathcal{T}$ represents the time domain and $\mathcal{S}$ the domain in space, which we require to be Lipschitz regular. We allow a mixture of Dirichlet and Neumann boundary conditions on $\partial\Omega$ which we denote as $\Gamma_D, \Gamma_N \subset \partial\Omega$ respectively. We further assume them to be such that the problem is well posed. The class of partial differential equations introduced in the prologue that we would like to solve for then reads as the following:

$$\begin{aligned} \partial_t u - \operatorname{div}(D(x)\nabla u) &= f(u) && \text{in } \Omega \\ u &= g_D && \text{on } \Gamma_D \\ \nabla u \cdot n &= g_N && \text{on } \Gamma_N \end{aligned} \tag{4.2}$$

It describes a parabolic partial differential equation with a potentially nonlinear right-hand side. We further assume that $D(x)$ is a bounded, symmetric uniformly positive definite matrix of size $N \times N$ with functions in $L^2(\mathcal{S})$ for almost all $x \in \bar{\mathcal{S}}$. Typically we will have that $u(x,0) = u_0 = g_D$ for all $x \in \mathcal{S}$ and Neumann boundary conditions on the boundary of $\mathcal{S}$ for $t \in (0,T)$.

The next step will be to derive an equivalent optimisation problem whose solution therefore then coincides with the solution of (4.2) at least in a weak sense. Since we are entirely working with finding solutions in Sobolev spaces in the least squares setting we can generally only require equivalence to a primal weak formulation of (4.2) or equivalence with respect to the solution space of the variational formulation, for a further discussion we refer to [8]. Generally when working with least squares finite element formulations choosing a suitable solution space U and data space Y is often non trivial as there are a number of difficulties that can arise. One usually faces a trade off between constructing a mathematically well-defined problem and allowing for a relatively simple, efficient, robust while still accurate implementation. Therefore further considerations need to be taken into account to make the methodology of LSFEMs competitive compared to other approaches like Galerkin approximations. We saw in the previous chapter that it is possible to derive least squares formulations that recover the properties of the Rayleigh–Ritz setting, however one hindrance one encounters in this setting as well as in many others is the higher order operator arising in (3.25), that would require a solution space of higher regularity. When considering a simple Poisson equation with Dirichlet boundary conditions this would for example imply that we would require the solution u to be from H_0^2, instead of H_0^1 [8] *more here or in previous LSFEM section ...?*. This does not only heavily limit the set of admissible solutions but it is additionally much harder to construct appropriate finite dimensional subspaces for and is therefore impractical to use. In order to succomb this obstacle we will recast (4.2) as a system of mixed equations only containing first order derivatives to apply the methodologies introduced in section (3.4) at the price of introducing an additional variable. Hence let

$$\begin{aligned} \partial_t u - \operatorname{div}(\sigma) &= f(u) && \text{in } \Omega \\ \sigma &= D(x)\nabla u && \text{in } \Omega \\ u &= g_D && \text{on } \Gamma_D \\ \nabla u \cdot n &= g_N && \text{on } \Gamma_N. \end{aligned} \tag{4.3}$$

Rearranging this equation into a vector form we obtain in Ω

$$\begin{pmatrix} I & -D(x)\nabla \\ -\operatorname{div} & \frac{\partial}{\partial t} \end{pmatrix} \begin{pmatrix} \sigma \\ u \end{pmatrix} = \begin{pmatrix} 0 \\ f(u) \end{pmatrix} \tag{4.4}$$

which we will refer to as the mixed strong form of our problem and can be shortened to $\mathcal{A}([\sigma, u]) = \tilde{f}(u)$, where $\mathcal{A}$ denotes the differential operator and $\tilde{f}(u)$ the right hand side.

Remark. It is not clear yet in which space we will be looking for a solution. In general when working with strong and weak formulations there is a a number of function spaces involved, problem formulations that are similar but not the same and it is difficult to represent them in clear but short notation. Therefore we will first derive a least squares functional J with a corresponding variational formulation, postponing the definition of appropriate spaces for now, but simply assuming them to already be well-defined, and later on show that they indeed exist.

So instead of looking for a solution of the strong formulation which is far more restrictive, let us now turn to the derivation of an optimisation problem that we would like to satisfy in a weak sense. The properties that we would like to be fulfilled are the following; we want a solution of (4.3) to be a global minimum of the optimisation problem, independently of the choice of the spaces that we are using and its associated norm, hence we additionally want the least squares functional J to be zero for a solution of (4.3). On the other hand if $J([\sigma, u], f) = 0$ then the orginial problem (4.3) also has to be satisfied if only in a weak sense with respect to the spaces U and Y. We can check that all three properties hold for the functional

$$\tilde{J}(\sigma, u) = \frac{1}{2}\|u_t - \operatorname{div}(\sigma) - f(u)\|_Y^2 + \frac{1}{2}\|\sigma - D(x)\nabla u\|_Y^2 \tag{4.5}$$

Furthermore it has shown to be practical to be able to weight the two terms with coefficients [source] which does not affect the solution of the continuous problem but grants us the possibility to numerically give more importance to one term than the other, a further exploration of this will be in section 6.3.2. Therefore the problem then reads

$$\min_{(\sigma,u)\in U} J([\sigma, u], f) = \frac{1}{2}c_1\|u_t - \operatorname{div}(\sigma) - f(u)\|_Y^2 + \frac{1}{2}c_2\|\sigma - D(x)\nabla u\|_Y^2. \tag{4.6}$$

J now defines an energy we can minimise. If we consider the functional for $f = 0$ we obtain

$$J([\sigma, u], 0) = \frac{1}{2}c_1\langle u_t - \operatorname{div}(\sigma), u_t - \operatorname{div}(\sigma)\rangle_Y + \frac{1}{2}c_2\langle \sigma - D(x)\nabla u, \sigma - D(x)\nabla u\rangle_Y. \tag{4.7}$$

which more specifically gives rise to the following bilinear form if we differentiate J with respect to directional derivatives τ and v, and subsequently sum over them, see section 4.5, and Appendix A for further derivations.

$$\mathcal{B}([\sigma, u], [\tau, v]) = \langle \begin{pmatrix} I & -D\nabla \\ -\operatorname{div} & \frac{\partial}{\partial t} \end{pmatrix} \begin{pmatrix} \sigma \\ u \end{pmatrix}, \begin{pmatrix} I & -D\nabla \\ -\operatorname{div} & \frac{\partial}{\partial t} \end{pmatrix} \begin{pmatrix} \tau \\ v \end{pmatrix} \rangle_Y \tag{4.8}$$

The resulting candidate for a variational formulation can be derived as usual [source] that is as the *sum* of the directional derivatives of J, where the *directions* become the testfunctions from the space W.

$$\begin{aligned} &\text{Find } (\sigma, u) \in U \text{ such that } \mathcal{B}([\sigma,u],[\tau,v]) = \mathcal{L}_{\tilde{f}}(u)([\tau,v]) \quad \forall [\tau,v] \in W \\ &\text{with } \mathcal{B}([\sigma,u],[\tau,v]) = (\mathcal{A}([\sigma,u]),\mathcal{A}([\tau,v])) \text{ and } \mathcal{L}_{\tilde{f}}(u)([\tau,v]) = (\mathcal{A}[\tau,v],\tilde{f}(u))_Y \end{aligned} \tag{4.9}$$

Remark: We can see that $\mathcal{B}$ is a symmetric bilinear form. In addition we have that $\mathcal{B}([\sigma,u],[\sigma,u]) \geq 0$ for all $[\sigma,u] \in U$, since $\mathcal{B}([\sigma,u],[\sigma,u]) = J([\sigma,u],0)$ is the sum of two squared L^2–norms. $\mathcal{L}_{\tilde{f}}(u)([\tau,v])$ is a linear operator in v but nonlinear in u. What remains to be determined are the function spaces U, W, and Y.

4.2 Function Spaces

In order to define U and Y let us consider the optimisation problem (4.6) and its variational formulation (4.9) again. We would like to allow for the broadest class of solutions possible while still ensuring that all terms are well-defined in U, and staying away from Sobolev spaces of negative or fractional powers due to the beforementioned practicality reasons. Additionally to guarantee the existence of all terms involved we have to make sure that the present weak derivatives of σ and u exist in the induced inner product of Y. Let us therefore consider the following spaces

$$H_{\text{div}}(\mathcal{S}) = \{\sigma \in (L^2(\mathcal{S}))^N : \text{div}(\sigma) \in L^2(\mathcal{S})\} \tag{4.10}$$

$$H_{\text{div}}(\Omega) = H_{\text{div}}(\mathcal{S}) \times L^2(\mathcal{T}) \tag{4.11}$$

$$U = W = H_{\text{div}}(\Omega) \times H^1(\Omega) \tag{4.12}$$

$$Y = L^2(\Omega) \tag{4.13}$$

And thus we have that

$$\sigma \in H_{\text{div}}(\Omega) \text{ and } u \in H^1(\Omega) \text{ with} \tag{4.14}$$

$$\|\sigma\|^2_{H_{\text{div}}(\Omega)} = \|\sigma\|^2_{L^2(\Omega)} + \|\operatorname{div}(\sigma)\|^2_{L^2(\Omega)}, \tag{4.15}$$

$$\|u\|^2_{H^1(\Omega)} = \|u\|^2_{L^2(\Omega)} + \|\nabla u\|^2_{L^2(\Omega)} + \|u_t\|^2_{L^2(\Omega)} \text{ and hence} \tag{4.16}$$

$$\|(\sigma,u)\|^2_U = \|\sigma\|^2_{H_{\text{div}}(\Omega)} + \|u\|^2_{H^1(\Omega)} \tag{4.17}$$

$$\tag{4.18}$$

The direct sum of two Hilbert spaces is again a Hilbert space using the inner product induced by the sum of the respective inner products [28]. We have that $L^2(\mathcal{T})$, $H^1(\Omega)$ and $H_{\text{div}}(\mathcal{S})$ are Hilbert spaces [29], and hence we obtain that U, W, and Y are as well. We can then check that all above terms in $J([\sigma,u],0)$ are well defined. If we additionally assume $f(u) \in L^2(\Omega)$ for all u, the problem remains to be well-posed.

In order to fully set the theoretical framework that guarantees us all the favourable attributes of the beforementioned Rayleigh–Ritz setting we would like to fulfill the assumptions of *Theorem 1* from section 3.3, which require the usage of conforming discrete subspaces $U^h \subset U$ and the following norm equivalence for some $\alpha, \beta > 0$

$$\alpha\|[\sigma,u]\|^2_U \leq J([\sigma,u],0) \leq \beta\|[\sigma,u]\|^2_U \quad \forall\ [\sigma,u] \in U. \tag{4.19}$$

In order to show the upper bound on J we use that $Y = L^2(\Omega)$, $D(x)$ is bounded, i.e. there exists $d_{\max} \in \mathbb{R}$ such that $\|D(x)\|_Y \leq d_{\max}$ for all $x \in \mathcal{S}$ and the parallelogram identity which holds in Hilbert spaces.

$$\begin{aligned} J([\sigma, u], 0) &= \|u_t - \operatorname{div}(\sigma)\|^2_{L^2(\Omega)} + \|\sigma - D(x)\nabla u\|^2_{L^2(\Omega)} \\ &\leq \|u_t - \operatorname{div}(\sigma)\|^2_{L^2(\Omega)} + \|u_t + \operatorname{div}(\sigma)\|^2_{L^2(\Omega)} + \|\sigma - D(x)\nabla u\|^2_{L^2(\Omega)} + \|\sigma + D(x)\nabla u\|^2_{L^2(\Omega)} \\ &\leq 2\|u_t\|^2_{L^2(\Omega)} + 2\|\operatorname{div}(\sigma)\|^2_{L^2(\Omega)} + 2\|\sigma\|^2_{L^2(\Omega)} + 2d^2_{\max}\|\nabla u\|^2_{L^2(\Omega)} \\ &\leq 2\|u\|^2_{L^2(\Omega)} + 2\|u_t\|^2_{L^2(\Omega)} + 2d^2_{\max}\|\nabla u\|^2_{L^2(\Omega)} + 2\|\operatorname{div}(\sigma)\|^2_{L^2(\Omega)} + 2\|\sigma\|^2_{L^2(\Omega)} \\ &\leq \max(2, 2d^2_{\max})(\|u\|^2_{H^1(\Omega)} + \|\sigma\|^2_{H_{\operatorname{div}}(\Omega)}) \\ &= \beta\|[\sigma, u]\|^2_U \qquad \text{with } \beta := \max(2, 2d^2_{\max}). \end{aligned} \tag{4.20}$$

The proof of the coercivity is not straight forward and potentially not even true. In the paper of Z. Cai et al. on "First-order system least squares for second-order partial differential equations: Part I", [30], they show norm equivalence for a similar class of problems that are however elliptic and therefore the terms and spaces involved differ. Nevertheless it might be possible to proceed similarly to their work to show that such an α exists, this is however beyond the scope of this thesis but could be an interesting extension to the topic.

A point that has not really been discussed so far but will have to be taken into account is the way of how to treat the boundary conditions in least–squares formulations. One possibility is to also include them in the functional as an additional term while another one would be to directly include them in the discretised system of the space. The former one entails the additional definition of an appropriate norm on the boundary while also requiring the treatment of the additional term. Since we have assumed it to be at least L^2-regular the appropriate conditions can directly be imposed as part of the discretised system which will be discussed in more detail in the implementation section.

4.3 A Finite Element Space-Time Formulation

After having derived a continuous least squares formulation, let us turn towards deriving a finite element discretisation of the problem. We want to consider conforming finite-dimensional subspaces U^h and W^h of U and W respectively, that are defined on the entire space-time domain, where U^h contains the solution space for σ^h and u^h, that is $s^h = [\sigma^h, u^h] \in U^h$. However the subspaces for the two do not have to be the same, and can be chosen independently of each other which can potentially be advantageous as the continuous spaces differ as well, due to their different properties. Hence let us assume that $\sigma^h \in \tilde{U}^h$ and $u^h \in \hat{U}^h$, where $U^h = \tilde{U}^h \times \hat{U}^h$.

So suppose we have $\tilde{n} = \dim(\tilde{U}^h)$ and let $\{\tilde{\phi}_1, ..., \tilde{\phi}_{\tilde{n}}\}$ be a basis of $\tilde{U}^h$ and similarly $n = \dim(\hat{U}^h)$ with $\operatorname{span}\{\phi_1, ..., \phi_n\} = \hat{U}^h$. We furthermore assume U^h to be constructed such that $\inf_{s^h \in U^h} \|s - s^h\|_U \to 0$ as $h \to 0$ for all $s \in U$. *Is this a reasonable assumption, Raviart - Thomas only in space what for time?! Some reference? Direct sums so it works?.* It is also worth noting that since we are in a space-time setting we have $\tilde{\phi} = \tilde{\phi}(x, t)$ and $\phi = \phi(x, t)$.

We can then represent σ^h and u^h as a linear combination of basis functions in $\tilde{U}^h$ or equivalently $\hat{U}^h$ that is

$$\sigma_h(x, t) = \sum_{i=1}^{\tilde{n}} \sigma_i \tilde{\phi}_i(x, t) \qquad u_h(x, t) = \sum_{i=1}^{n} u_i \phi_i(x, t) \tag{4.21}$$

The functional J then looks as follows

$$J(\sigma_h, u_h) = \| \sum_{i=1}^{n} u_i \ (\phi_i)_t - \sum_{i=1}^{\tilde{n}} \sigma_i \ \operatorname{div}(\tilde{\phi}_i) - f(\sum_{i=1}^{n} u_i \phi_i) \|_Y^2 + \| \sum_{i=1}^{\tilde{n}} \sigma_i \tilde{\phi}_i - D(x) \nabla (\sum_{i=1}^{n} u_i \phi_i) \|_Y^2. \tag{4.22}$$

In the arising variational formulation we will also have to consider finite dimensional subspaces of W_h of W. In the scope of this thesis we restrict ourselves to the assumption that $W_h = U_h$. That is we introduce a set of test functions consisting of the basis vectors of $\tilde{U}^h$ and U^h. The discretised weak form then reads

$$\text{Find } [\sigma_h, u_h] \in U_h \text{ such that } B \cdot [\sigma_h, u_h]^T = L_{\tilde{f}}(u_h) \tag{4.23}$$

where $B \in \mathbb{R}^{m \times m}$, with $m = \tilde{n} + n$, be the matrix arising from the discretised bilinear operator, and $L_{\tilde{f}}(u_h) \in \mathbb{R}^m$ being the discretised right-hand side which we for now assume to contain all nonlinear terms, that is those related to f. Since we assume that the solution $s_h = [\sigma_h, u_h]$ first contains all values corresponding to σ_h and then for u_h we obtain a block structure for B and $L_{\tilde{f}}$ of the following form.

$$B = \begin{bmatrix} B_{\sigma\sigma} & B_{\sigma u} \\ B_{u\sigma} & B_{uu} \end{bmatrix} \qquad L_{\tilde{f}}(u_h) = \begin{bmatrix} (L_{\tilde{f}}(u_h))_\sigma \\ (L_{\tilde{f}}(u_h))_u \end{bmatrix} \tag{4.24}$$

Each entry of each of the blocks of B can be computed explicitly according to the subsequent schemes. For the detailed computation we refer to the next section and appendix A.

For $B_{\sigma\sigma}$: $$B_{ij} = \langle \tilde{\phi}_j, \tilde{\phi}_i \rangle_Y + \langle \operatorname{div}(\tilde{\phi}_j), \operatorname{div}(\tilde{\phi}_i) \rangle_Y \qquad \forall i,j \in \{1, ..., \tilde{n}\} \tag{4.25}$$

For $B_{\sigma u}$: $$B_{ij} = -\langle D(x) \nabla \phi_j, \tilde{\phi}_i \rangle_Y - \langle (\phi_j)_t, \operatorname{div}(\tilde{\phi}_i) \rangle_Y \qquad \forall i \in \{1, ..., \tilde{n}\}, j \in \{\tilde{n}+1, ..., m\} \tag{4.26}$$

For $B_{u\sigma}$: $$B_{ij} = -\langle D(x) \nabla \phi_i, \tilde{\phi}_j \rangle_Y - \langle (\phi_i)_t, \operatorname{div}(\tilde{\phi}_j) \rangle_Y \qquad i \in \{\tilde{n}+1, ..., m\}, \forall j \in \{1, ..., \tilde{n}\} \tag{4.27}$$

For B_{uu} : $$B_{ij} = \langle D(x) \nabla \phi_j, D(x) \nabla \phi_i \rangle_Y + \langle (\phi_j)_t, (\phi_i)_t \rangle_Y \qquad \forall i,j \in \{\tilde{n}+1, ..., m\} \tag{4.28}$$

In the case that f is independent of u, that is $f = 0$ or $f = f(x,t)$, the derivative of f with respect to u, is zero and therefore the right-hand side, and which we will denote by $L_{\tilde{f}} = L_{\tilde{f}}(u_h)$ to underline its independence of u. Thus it only contains the following terms

$$(L_{\tilde{f}})_i = 0 \qquad i \in \{1, ..., \tilde{n}\} \tag{4.29}$$

$$(L_{\tilde{f}})_i = \sum_{j=1}^{\tilde{n}} \langle \operatorname{div}(\tilde{\phi}_j), f \rangle_Y - \langle (\phi_i)_t, f \rangle_Y \qquad i \in \{\tilde{n}+1, ..., m\} \tag{4.30}$$

Thus we now know how compute each term and can assemble one large linear system of equations

$$B \begin{pmatrix} \sigma \\ u \end{pmatrix} = L_{\tilde{f}} \tag{4.31}$$

which can then be solved immediately for $[\sigma, u]$ using for example a multigrid method. In the case of $f = f(u)$ the situation is more complicated, since we also have to take the derivatives of f with respect to u into account. If $f(u)$ is a linear function we can directly include the additional terms in B. Otherwise we require a nonlinear iteration scheme which considers linearisations of the problem and whose construction will be the topic of the remaining part of the chapter.

4.4 Gradient and Hessian of the Objective

Let us compute the gradient and the Hessian of the functional J, which are needed for the construction of B and the nonlinear iteration scheme. We therefore determine its first and second order Gateaux derivatives. We formulate them here now in their continuous form and will discuss the discretisation in the subsequent section. Furthermore for the remainder of this chapter we will assume that $\langle\cdot,\cdot\rangle = \langle\cdot,\cdot\rangle_Y$ in order to simplify the notation. Let us first assume $f = 0$ to derive the matrix B. That is we have

$$\begin{aligned} J([\sigma,u],0) &= J_1([\sigma,u]) + J_2([\sigma,u]), \qquad \text{where} \\ J_1(\sigma,u) &:= \frac{1}{2}c_1\langle u_t - \operatorname{div}(\sigma), u_t - \operatorname{div}(\sigma)\rangle \\ J_2(\sigma,u) &:= \frac{1}{2}c_2\langle \sigma - D(x)\nabla u, \sigma - D(x)\nabla u\rangle \end{aligned} \tag{4.32}$$

Since $u \in H^1(\Omega)$ and $\sigma \in H_{\text{div}}(\Omega)$ they are not defined pointwise but we can nevertheless take the subsequent limits [*something more here, reference?!*], but then also only hold almost everywhere. We then obtain the following directional partial derivative for J_1 using the linearity and the symmetry of the inner product.

$$\begin{aligned} \frac{\partial J_1}{\partial\sigma} &= \lim_{\epsilon\to 0}\frac{J_1(\sigma+\epsilon\tau,u) - J_1(\sigma,u)}{\epsilon} \\ &= \lim_{\epsilon\to 0}\frac{c_1}{2\epsilon}(\langle u_t - \operatorname{div}(\sigma+\epsilon\tau), u_t - \operatorname{div}(\sigma+\epsilon\tau)\rangle - \langle u_t - \operatorname{div}(\sigma), u_t - \operatorname{div}(\sigma)\rangle) \\ &= \lim_{\epsilon\to 0}\frac{c_1}{2\epsilon}(\langle u_t,u_t\rangle - \langle u_t,\operatorname{div}(\sigma)\rangle - \epsilon\langle u_t,\operatorname{div}(\tau)\rangle - \langle\operatorname{div}(\sigma),u_t\rangle + \langle\operatorname{div}(\sigma),\operatorname{div}(\sigma)\rangle \\ &+ \epsilon\langle\operatorname{div}(\sigma),\operatorname{div}(\tau)\rangle - \epsilon\langle\operatorname{div}(\tau),u_t\rangle + \epsilon\langle\operatorname{div}(\tau),\operatorname{div}(\sigma)\rangle + \epsilon^2\langle\operatorname{div}(\tau),\operatorname{div}(\tau)\rangle \\ &- \langle u_t,u_t\rangle + \langle u_t,\operatorname{div}(\sigma)\rangle + \langle\operatorname{div}(\sigma),u_t\rangle - \langle\operatorname{div}(\sigma),\operatorname{div}(\sigma)\rangle) \\ &= \lim_{\epsilon\to 0}\frac{c_1}{2\epsilon}(-2\epsilon\langle u_t,\operatorname{div}(\tau)\rangle + 2\epsilon\langle\operatorname{div}(\sigma),\operatorname{div}(\tau)\rangle + \epsilon^2\langle\operatorname{div}(\tau),\operatorname{div}(\tau)\rangle) \\ \\ &= -c_1\langle u_t,\operatorname{div}(\tau)\rangle + c_1\langle\operatorname{div}(\sigma),\operatorname{div}(\tau)\rangle \end{aligned} \tag{4.33}$$

By proceeding analogously for equation J_2 and the partial directional derivatives for u we end up with

$$\frac{\partial J_2}{\partial\sigma} = c_2\langle\sigma,\tau\rangle - c_2\beta\langle\tau,\nabla u\rangle \tag{4.34}$$

$$\frac{\partial J_1}{\partial u} = c_1\langle u_t,v_t\rangle - c_1\langle v_t,\operatorname{div}(\sigma)\rangle \tag{4.35}$$

$$\frac{\partial J_2}{\partial u} = -c_2\langle\sigma,D(x)\nabla v\rangle + c_2\langle D(x)\nabla u, D(x)\nabla v\rangle \tag{4.36}$$

Hence we obtain the following partial first order directional derivatives.

$$D_\sigma J[\tau] = \frac{\partial}{\partial\sigma}J([\sigma,u])[\tau] = c_2\langle\sigma,\tau\rangle + c_1\langle\operatorname{div}(\sigma),\operatorname{div}(\tau)\rangle - c_2\langle D(x)\nabla u,\tau\rangle - c_1\langle u_t,\operatorname{div}(\tau)\rangle \tag{4.37}$$

$$D_u J[v] = \frac{\partial}{\partial u}J([\sigma,u])[v] = c_1\langle u_t,v_t\rangle - c_1\langle v_t,\operatorname{div}(\sigma)\rangle - c_2\langle\sigma,D(x)\nabla v\rangle + c_2\langle D(x)\nabla u,D(x)\nabla v\rangle$$

(4.38)

Following the same principles one can determine the second order partial derivatives.

$$\begin{aligned} \frac{\partial^2}{\partial\sigma^2}J[\tau][\rho] &= c_2\langle\rho,\tau\rangle + c_1\langle\operatorname{div}(\rho),\operatorname{div}(\tau)\rangle \\ \frac{\partial^2}{\partial\sigma\partial u}[v][\tau] = \frac{\partial^2}{\partial u\partial\sigma}[\tau][v] &= -c_2\langle\tau, D(x)\nabla v\rangle - c_1\langle v_t,\operatorname{div}(\tau)\rangle \\ \frac{\partial^2 J}{\partial u^2}[v][w] &= c_1\langle w_t, v_t\rangle + c_2\langle D(x)\nabla w, D(x)\nabla v\rangle \end{aligned} \tag{4.39}$$

If we now recast this in the finite dimensional setting using the beforementioned basis functions instead of the τ, ρ, v and w, and we can compute the matrix B *proper name, just call it H for Hessian?* which corresponds exactly to the description given by (4.25)–(4.28).

For the terms involving f we can proceed in a similar way, given that f is sufficiently smooth. The exact assumptions and the corresponding computations can be found in appendix A. This entails first the construction of an entire continuous iteration scheme and then its discretisation. One finally ends up with the following first and second order partial directional derivatives for a $f = f(u)$.

$$D_\sigma J[\tau] = c_2\langle\sigma,\tau\rangle + c_1\langle div(\sigma), div(\tau)\rangle - c_2\langle D(x)\nabla u,\tau\rangle - c_1\langle u_t, div(\tau)\rangle - c_1\langle f(u), div(\tau)\rangle \tag{4.40}$$

$$D_u J[v] = c_1\langle u_t, v_t\rangle - c_1\langle v_t, div(\sigma)\rangle - c_2\langle\sigma, D(x)\nabla v\rangle + c_2\langle D(x)\nabla u, D(x)\nabla v\rangle \tag{4.41}$$

$$- c_1\langle u_t, f'(u)\cdot v\rangle - c_1\langle v_t, f(u)\rangle - c_1\langle div(\sigma), f'(u)\cdot v\rangle + c_1\langle f(u), f'(u)\cdot v\rangle, \tag{4.42}$$

(4.43)

$$D_{\sigma\sigma}J[\tau][\rho] = c_2\langle\rho,\tau\rangle + c_1\langle\operatorname{div}(\rho),\operatorname{div}(\tau)\rangle \tag{4.44}$$

$$D_{\sigma u}J[v][\tau] = D_{u\sigma}J[\tau][v] = -\langle\tau, D(x)\nabla v\rangle - \langle v_t, div(\tau)\rangle - \langle div(\tau), f'(u)v\rangle \tag{4.45}$$

$$D_{uu}J[v][w] = c_1\langle w_t, v_t\rangle + c_2\langle D(x)\nabla w, D(x)\nabla v\rangle + c_1\langle u_t, w^T f''(u)v\rangle - c_1\langle w_t, f'(u)\cdot v\rangle \tag{4.46}$$

$$- c_1\langle v_t, f'(u)\cdot w\rangle - c_1\langle div(\sigma), w^T f''(u)v\rangle + c_1\langle f(u), w^T f''(u)v\rangle + \langle f'(u)\cdot w, f'(u)\cdot v\rangle \tag{4.47}$$

Another way to formulate the problem is to first restrict the functional to finite dimensional subspaces and then differentiate which is the way we implemented it. For a further discussion of the pros and cons of either approach we can refer to [*find good source?!*]. Hence the terms involvoing f are constructed slightly differently, and will be introduced in the next section, as we derive the iterative schemes.

4.5 Nonlinear Iteration Scheme

As discussed in section (3.4) the general approach to finding a minimiser of the functional J is to search for a tuple $[\sigma, u]$ for which $\nabla J([\sigma, u]) = 0$, while $\nabla^2 J([\sigma, u])$ is positive definite. We start with an initial guess $s_0 = s_{\text{init}} = [\sigma_{\text{init}}, u_{\text{init}}]$ and then successively try to decrease energy. In case of a Newton step by finding a quadratic approximation of J at the value of the current iterate s_k, where the updated solution s_{k+1} is the minimiser of the quadratic approximation. However if J is not convex, and hence $\nabla^2 J([\sigma, u])$ not positive definite, the extremum of the quadratic approximation can actually lead us to a maximum of J. Therefore if we want to use a Newton iteration and maintain a decrease or at least no increase of energy in every step, that is $J(s_0) \geq J(s_1) \geq ... \geq J(s_k) \geq ...$ we need to be checking for convexity. In order to perform a

Newton step we have to compute the Hessian of J, as well as its gradient in each iterate. The other option that was discussed previously to obtain a reduction in energy is to use a gradient descent method where we simply take an iteration step in the steepest descent direction. Hence we only need to evaluate the gradient in this case, which is computationally much cheaper but usually leads to a very slow convergence rate.

There are different ways to linearise and discretise in f, that is we have to decide how to represent f in our finite dimensional subspace formulation. For simplicity we assume that $f \in C^2$, and denote the first and second derivative with respect to u by f' and f''. This is in line with the forcing term stemming from the FitzHugh-Nagumo model as well as many other physical applications [source]. Hence for each degree of freedom in u we can determine coefficients for f, f', f'' and represent them in the basis of $\hat{U}^h$, that is

$$f_h = \sum_{i=1}^{n} f_i \phi_i, \qquad f_h' = \sum_{i=1}^{n} f_i' \phi_i, \qquad f_h'' = \sum_{i=1}^{n} f_i'' \phi_i \tag{4.48}$$

Therefore f_h, f_h' and f_h'' now represent a finite dimensional approximation of f for one particular approximation u_h. With this representation we can now proceed to formulate a discretisation of the gradient of J, linearised in $s_k = [\sigma_k, u_k]$ where we know that inner products are still well-defined since we are working in conforming subspaces.
We want $\nabla J = 0$, where

$$\nabla J_k = \nabla J(s_k) = B \begin{pmatrix} \sigma_k \\ u_k \end{pmatrix} - L_{\tilde{f}} \quad \text{with} \tag{4.49}$$

$$((L_{\tilde{f}})_\sigma)_i = \sum_{j=1}^{\tilde{n}} \langle \sigma_j \operatorname{div}(\tilde{\phi}_j), f_i' \phi_i \rangle_Y \tag{4.50}$$

$$((L_{\tilde{f}})_u)_i = -\sum_{j=1}^{n} \langle u_j (\phi_j)_t, f_i' \phi_i \rangle_Y - \sum_{j=1}^{n} \langle (\phi_j)_t, f_i \phi_i \rangle_Y + \sum_{j=1}^{n} \langle f_j \phi_j, f_i' \phi_i \rangle_Y \tag{4.51}$$

$$\tag{4.52}$$

For one gradient descent step we therefore then compute the update by

$$s_{k+1} = s_k + \alpha(-\nabla J_k) \tag{4.53}$$

where $\alpha > 0$ is a scaling parameter that can be chosen in different ways, for example a line search algorithm and which will be discussed in more detail in the implementation chapter.

From the gradient and previous section we can determine the discretised Hessian which is needed for a Newton step. We obtain

$$H_k = \nabla^2 J(s_k) = B_{\text{lin}} + Q \tag{4.54}$$

where $B_{\text{lin}} = B$ from before and Q contains the nonlinear part, that is

$$Q = \begin{bmatrix} 0 & Q_{\sigma u} \\ Q_{u\sigma} & Q_{uu} \end{bmatrix}, \text{ where} \tag{4.55}$$

$$\tag{4.56}$$

where each of the submatrices can be determined in the same manner as the gradient in (4.50) and (4.51) that is using the discretisation of f given by (4.48) and the second order derivatives from (4.44) - (4.46). This then leads to the iteration

$$s_{k+1} = s_k - H_k^{-1}(\nabla J_k) \tag{4.57}$$

of which we would like to solve the linear system of equations

$$e_{k+1} = -H_k^{-1}(\nabla J_k) \quad \text{with} \quad s_{k+1} = s_k + e_{k+1} \tag{4.58}$$

But since it is generally expensive to compute the inverse of a large matrix, even if H_k is sparse and symmetric, because this property does in general not translate to the inverse we apply a multigrid method to solve (4.58) to solve this linear system of equations in each iteration. The following chapter will provide an insight into the underlying principles of the multigrid methodology before we turn to its actual implementation.