UNIVERSITY OF CALIFORNIA
SANTA CRUZ

AN IMPACT OF DIVERGENCE-FREE MAGNETIC FIELD INTERPOLATION USING A SOLENOIDAL GAUSSIAN PROCESS KERNEL

A thesis submitted in partial satisfaction of the
requirements for the degree of

MASTER OF SCIENCE

in

SCIENTIFIC COMPUTING AND APPLIED MATHEMATICS

by

Liliya Milenska

March 2018

The Thesis of Liliya Milenska
is approved:

Professor Dongwook Lee, Chair

Professor Nicholas Brummell

Professor Daniele Venturi

Tyrus Miller
Vice Provost and Dean of Graduate Studies

Table of Contents

List of Figures **iv**

Abstract **vi**

1 Introduction **1**
1.1 Magnetohydrodynamics 2
1.1.1 The Governing Equations 3
1.1.2 The Importance Of The Solenoidal Condition 5
1.2 The Unsplit Staggered Mesh Scheme 8
1.3 Interpolation Using Gaussian Processes 11

2 Results **17**
2.1 Implementation 17
2.2 Numerical Tests 21
2.2.1 Field Loop Problem 21
2.2.2 Orszag-Tang Problem 26

3 Conclusion **30**

4 Appendix **32**
4.1 Analytically Divergence-Free Interpolant 32
4.2 Numerical Divergence of the Matrix-Valued Kernel 34

Bibliography **36**

List of Figures

1.1 E_z calculated at the cell-corners, used to evolve $b^n_{x,i+\frac{1}{2},j}$ and $b^n_{y,i,j+\frac{1}{2}}$ to the next time step. 7

1.2 The staggered geometry. The variables $\rho, \mathbf{u}$ and p are collocated at the cell face centers. The magnetic fields are collocated both at the cell face centers ($\mathbf{B}$) and the cell interface centers (b_x and b_y). $\mathbf{V}_{i,j}$ contains the cell-centered variables. 9

1.3 Configuration of the Riemann states at a given cell (i, j). 10

2.1 After each update, USM provides b_x along the x-faces and b_y along the y-faces as shown in 2.1a. To perform divergence-free GP interpolation, we need both b_x and b_y at each sample point as shown in 2.1b. The missing variables are shown in red. 18

2.2 Four-point stencils used for single-valued GP interpolation. The unknown variable is shown in red in each case and the red arrows indicate GP interpolation from the sample points to the point where that quantity is unknown. 19

2.3 Stencil of radius $R_{stencil}$ used for divergence-free GP interpolation. The sample points are shown in black (cell interface centers) and the interpolation point is in red (the cell center at (i, j)). For this choice of $R_{stencil}$ we have a total of 16 sample points. 20

2.4 Comparison of the magnetic pressure at $t = 2.0$ for the Field Loop problem, without and with GP interpolation and two values of ℓ. The same color scheme is used for values between 5.5×10^{-26} and 5.8×10^{-7} and the resolution is 256×256. 23

2.5 Comparison of the z-component of the vector potential, A_z at $t = 2.0$ for the Field Loop problem, without GP interpolation and with GP interpolation and a value of the parameter $\ell = 8.0\Delta y$. Plotted are 20 contour lines between -2.16×10^{-6} and 2.7×10^{-4} and the resolution is 256×256. 25

2.6 Comparison of L1 errors on B_x and B_y at $t = 2.0$ using the USM scheme with arithmetic averaging (blue) and with GP interpolation (green) for resolutions $N = 8, 16, 32, 64$ and 128. 27

2.7 Comparison of 30 density contour lines for the Orszag-Tang problem plotted at $t = 0.5$ (with density values between 0.38 and 2.21) and $t = 1.0$ (with density values between 0.18 and 1.8) with and without GP interpolation. The resolution is 512×512 and we have used the Roe Riemann solver and $\ell = 1.5\Delta x$ for the GP interpolation. 29

Abstract

An Impact of Divergence-Free Magnetic Field Interpolation Using a Solenoidal Gaussian Process Kernel

by

Liliya Milenska

A new interpolation method using an analytically solenoidal Gaussian Process (GP) kernel has been introduced within the Unsplit Staggered Mesh (USM) magnetohydrodynamics (MHD) solver, implemented in the University of Chicago ASC FLASH Center's FLASH4 code. USM computes numerically divergence-free (up to machine accuracy) magnetic fields at the cell interface-centers at each time step, and then uses those values to predict the cell-centered magnetic fields through simple linear interpolation. The GP interpolation method presented here (which replaces the simple linear interpolation) uses the computed cell interface-centered magnetic fields as sample points to train a GP model with an analytically divergence-free kernel function, which is used to predict these fields at the cell centers. The performance of the USM scheme using GP interpolation is studied and compared to the original scheme using arithmetic averaging. Some improvements can be seen, particularly when calculating the numerical divergence using the cell-centered fields. However, the interpolation is highly sensitive to the hyperparameter ℓ defining the kernel function and its optimization is subject to further study.

Chapter 1

Introduction

Magnetohydrodynamics (MHD) is the study of electrically-conducting fluid flows which interact with magnetic fields. MHD has numerous applications in modern science, ranging from astrophysical and geophysical problems, through nuclear energy, to metallurgy. Being highly non-linear, the equations of MHD require reliable, accurate and robust numerical methods for their solution. An important constraint in these equations is the solenoidal condition for the magnetic field: $\nabla \cdot \mathbf{B} = 0$. In analytical solutions, this requirement is automatically satisfied, but this is not necessarily true in a numerical setting, and could potentially lead to unwanted numerical oscillations and inaccuracies [1].

Several different numerical methods have been introduced and are routinely used to take care of the numerical divergence-free condition. For this study we use the Unsplit Staggered Mesh (USM) [9] MHD solver implemented in the University of Chicago ASC FLASH Center's FLASH4 [5] code release, which uses a Constrained Transport (CT) [6] type of scheme to ensure zero divergence for magnetic fields collocated at the cell interfaces. In typical second order CT schemes, the magnetic fields at the cell centers are not guaranteed to be divergence-free, as they are calculated by taking arithmetic averages of the ones collocated at the

cell interfaces.

The aim of this study is to improve this simple arithmetic averaging by replacing it with an interpolation method using Gaussian Processes (GP) within the USM scheme. This method uses a specific kernel function definition, which will ensure that the resulting predicted values will be analytically divergence-free.

In the following introductory sections, we present a brief general overview of the governing equations of MHD, the importance of the numerical solenoidal condition, as well as a description of the USM scheme. We then present and develop the GP interpolation methods we will use in our implementation. In Chapter 2 we summarize the main points of the implementation and present a series of numerical tests, comparing the results to those obtained through the USM solver with the default arithmetic averaging.

1.1 Magnetohydrodynamics

As already mentioned, magnetohydrodynamics (MHD) [3] is the study of electrically-conducting fluid flows and their interaction with magnetic fields. This mutual interaction can be viewed as a three-part process. Let $\mathbf{B}$ denote the magnetic field and $\mathbf{u}$, the velocity field of the flow. First, the relative movement of the fluid and the magnetic field will create an electromotive force proportional to $|\mathbf{u} \times \mathbf{B}|$ which will in turn generate electrical currents. Next, by Ampère's law, these currents will produce a new induced magnetic field, interfering with the original $\mathbf{B}$. The result of this interference will be a "drag" of the $\mathbf{B}$ field lines along with the fluid flow. Finally, the two magnetic fields together will interact with the induced current density (which we denote $\mathbf{J}$). This will produce a Lorentz force proportional to $\mathbf{J} \times \mathbf{B}$, and this force will affect the relative movement of the fields $\mathbf{u}$ and $\mathbf{B}$. Naturally, the amplitude of all these effects will depend on the impor-

tance of the fluid's velocity, its conductivity, as well as the characteristic length scale of the motion, which is linked to the strentgth of the induced magnetic field.

1.1.1 The Governing Equations

The governing equations of MHD are a combination of a simplified version of Maxwell's equations (1.1) of electrodynamics and the Navier-Stokes equation.

$$\begin{aligned} \nabla \cdot \mathbf{E} &= \frac{\rho_e}{\varepsilon_0}, \\ \nabla \cdot \mathbf{B} &= 0, \\ \nabla \times \mathbf{E} &= -\frac{\partial \mathbf{B}}{\partial t}, \\ \nabla \times \mathbf{B} &= \mu \left(\mathbf{J} + \varepsilon_0 \frac{\partial \mathbf{E}}{\partial t} \right). \end{aligned} \tag{1.1}$$

In these equations $\mathbf{E}$ denotes the electric field, ρ_e is the charge density, ε_0 is the permittivity of free space, $\mathbf{J}$ is the current density and μ is the permeability of free space. In addition to these, we also have the charge conservation equation and the force equation (with q being the total charge):

$$\begin{aligned} \nabla \cdot \mathbf{J} &= -\frac{\partial \rho_e}{\partial t}, \\ \mathbf{F} &= q(\mathbf{E} + \mathbf{u} \times \mathbf{B}). \end{aligned} \tag{1.2}$$

Several simplifications may be made of Maxwell's equations for the purposes of MHD. Indeed, in a conducting fluid the free charge density ρ_e can be shown to be insignificantly small, which will lead to some terms disappearing in the equations above. The new, revised equations become:

$$\begin{aligned} \nabla \cdot \mathbf{J} &= 0, \\ \nabla \cdot \mathbf{B} &= 0, \\ \nabla \times \mathbf{E} &= -\frac{\partial \mathbf{B}}{\partial t}, \\ \nabla \times \mathbf{B} &= \mu \mathbf{J}. \end{aligned} \tag{1.3}$$

We also have (with σ, the electrical conductivity) Ohm's law and the Lorentz force (per unit volume) given by:

$$\begin{aligned} \mathbf{J} &= \sigma(\mathbf{E} + \mathbf{u} \times \mathbf{B}), \\ \mathbf{F} &= \mathbf{J} \times \mathbf{B}. \end{aligned} \tag{1.4}$$

Combining Ohm's law with Faraday's equation and Ampère's law, we write the induction equation, which gives the relationship between $\mathbf{B}$ and $\mathbf{u}$:

$$\frac{\partial \mathbf{B}}{\partial t} = \nabla \times (\mathbf{u} \times \mathbf{B}) + \lambda \nabla^2 \mathbf{B}, \quad \lambda = (\mu\sigma)^{-1}. \tag{1.5}$$

In this expression, λ is the magnetic diffusivity, and for ideal conductors this quantity is zero. Furthermore, we have the Navier-Stokes equation combined with the Loretz force, governing the evolution of our fluid flow:

$$\rho \frac{D\mathbf{u}}{Dt} = -\nabla p + \rho\nu \nabla^2 \mathbf{u} + \mathbf{J} \times \mathbf{B}. \tag{1.6}$$

In this expression, ρ is the mass density of the fluid, p denotes the fluid pressure, ν is the viscosity and $\frac{D}{Dt}$ denotes the material derivative. For our purposes, we will limit ourselves to the equations of ideal MHD written in conservation form:

$$
\begin{aligned}
&\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0, && \text{Continuity} \\
&\frac{\partial \rho \mathbf{u}}{\partial t} + \nabla \cdot (\rho \mathbf{u}\mathbf{u} - \mathbf{B}\mathbf{B}) + \nabla p_{tot} = 0, && \text{Momentum} \\
&\frac{\partial \mathbf{B}}{\partial t} + \nabla \cdot (\mathbf{u}\mathbf{B} - \mathbf{B}\mathbf{u}) = 0, && \text{Induction} \\
&\frac{\partial E}{\partial t} + \nabla \cdot (\mathbf{u}E + \mathbf{u}p_{tot} - \mathbf{B}\mathbf{B} \cdot \mathbf{u}) = 0. && \text{Energy}
\end{aligned}
\tag{1.7}
$$

Here the momentum of the fluid is $\rho\mathbf{u}$, the total energy density is E and with γ being the ratio of specific heats we also have the current density and total pressure:

$$
\begin{aligned}
&\mathbf{J} = \nabla \times \mathbf{B}, \quad \text{Current Density} \\
&p_{tot} = p + B_p = (\gamma - 1)\left(E - \frac{1}{2}\rho||\mathbf{u}||^2 - \frac{1}{2}||\mathbf{B}||^2\right) + \frac{||\mathbf{B}||^2}{2}.
\end{aligned}
\tag{1.8}
$$

The norm used in the expression for the total pressure is the Euclidean norm: $||\mathbf{u}||^2 = u^2 + v^2 + w^2$ where $\mathbf{u} = (u, v, w)$.

1.1.2 The Importance Of The Solenoidal Condition

From an analytical point of view, the solenoidal condition on the magnetic field, $\nabla \cdot \mathbf{B} = 0$ must be maintained exactly, however this will not be automatically satisfied in numerical solutions. Furthermore, it has been shown in [1] that even a very small non-zero term for the magnetic field divergence may lead to numerical instability and unphysical components in the solution to the equations, appearing as an additional force parallel to the field itself.

When solving the equations in 1D, the divergence-free condition is easily enforced by setting the only component of the magnetic field B_x as a constant. However, in higher dimension when using conservative shock-capturing schemes, this is not straightforward. Different algorithms maintaining this divergence-free condition throughout the simulation have been proposed and extensively studied.

In this section we shall briefly mention some of them, along with their advantages and related issues. A complete study of the methods listed here is presented in [16].

A first approach is the so-called 8-wave scheme [8], which considers the addition of non-zero source terms proportional to $\nabla \cdot \mathbf{B}$ explicitly, on the right hand side of the MHD equations. A new discretization was developed in order to capture the evolution of the non-zero $\nabla \cdot \mathbf{B}$ term, which appears as an extra wave structure and propagates throughout the computational domain. The algorithm is simple to implement, but a significant downside is that the non-conservative nature of this scheme may lead to incorrect jump conditions and propagation speeds when the solution presents some discontinuities.

Another approach is the constrained transport (or CT) scheme - this is the method implemented into the USM solver [9] we will be using. It implements a particular discretization of the magnetic fields on a staggered grid, which maintains the divergence-free condition to machine round-off error. In this staggered grid formulation, we denote by $\mathbf{b}$ the magnetic field at the cell interface centers. Next, placing the z-component of the electric field E_z at the cell corners, one can solve the resulting set of discretized induction equations along the cell edges in order to evolve the components of $\mathbf{b}$ through Equation (1.9). The variables appearing on the right hand side are shown in Figure (1.1).

$$\begin{aligned} b^{n+1}_{x,i+\frac{1}{2},j} &= b^{n}_{x,i+\frac{1}{2},j} - \frac{\Delta t}{\Delta y}\left\{ E^{n+\frac{1}{2}}_{z,i+\frac{1}{2},j+\frac{1}{2}} - E^{n+\frac{1}{2}}_{z,i+\frac{1}{2},j-\frac{1}{2}} \right\}, \\ b^{n+1}_{y,i,j+\frac{1}{2}} &= b^{n}_{y,i,j+\frac{1}{2}} - \frac{\Delta t}{\Delta x}\left\{ -E^{n+\frac{1}{2}}_{z,i+\frac{1}{2},j+\frac{1}{2}} - E^{n+\frac{1}{2}}_{z,i-\frac{1}{2},j+\frac{1}{2}} \right\}. \end{aligned} \tag{1.9}$$

From these expressions it is then straightforward to show that the value of the numerical divergence of $\mathbf{b}$ given in Equation (1.10) will remain small (of the order of machine accuracy) throughout the simulation.

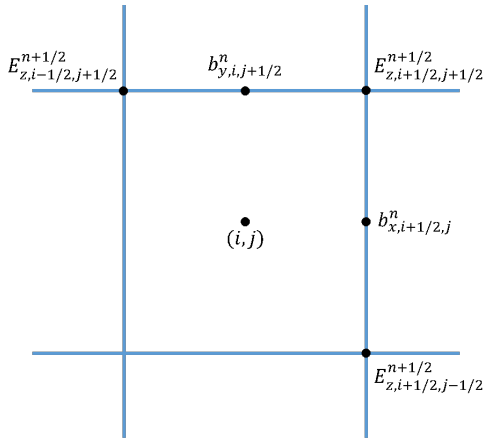


Figure 1.1: E_z calculated at the cell-corners, used to evolve $b^n_{x,i+\frac{1}{2},j}$ and $b^n_{y,i,j+\frac{1}{2}}$ to the next time step.

$$(\nabla \cdot \mathbf{b})_{i,j} = \frac{b^{n+1}_{x,i+\frac{1}{2},j} - b^{n+1}_{x,i-\frac{1}{2},j}}{\Delta x} + \frac{b^{n+1}_{x,i,j+\frac{1}{2}} - b^{n+1}_{x,i,j-\frac{1}{2}}}{\Delta y}. \tag{1.10}$$

Further details of this approach, specifically implemented into the USM scheme will be discussed further on.

A third approach is the so-called projection scheme [1], which will, as the name suggests, project the numerical solution for the magnetic field onto a subspace formed by zero-divergence solutions. The projection step is applied through a linear operator, and its output is then used as the magnetic field value for the following time step calculation.

Another possible way of maintaining $\nabla \cdot \mathbf{B}$ to be exactly zero is to consider the equations in terms of the vector potential $\mathbf{A}$ instead ($\mathbf{B} = \nabla \times \mathbf{A}$) [4]. This will however increase the order of the spatial derivatives, leading to a decreased order of accuracy.

1.2 The Unsplit Staggered Mesh Scheme

For our applications we use the FLASH code (version 4.4), developed at the Flash Center for Computational Science at the University of Chicago. It is a publicly available high-performance astrophysics simulation code, containing a wide array of conveniently separated modules and solvers. Its capabilities range from hydrodynamics and magnetohydrodynamics through nuclear physics to gravitational and cosmological applications. Regarding MHD, FLASH contains two units: the 8-wave model [8] (which evolves the MHD equations in a directionally-split manner) and the Unsplit Staggered Mesh algorithm (which is directionally unsplit) [9]. The main issue with directionally split methods for MHD is that they do not take into account the normal magnetic field (with respect to the sweep direction) when performing each directional sweep. The advantage of directionally-unsplit methods is that they naturally contain the necessary multidimensional terms, which take into account all components of the fields. The USM scheme is based on a finite-volume high-order Godunov method and uses a CT-type implementation to enforce the solenoidal condition on the magnetic field.

In the staggered geometry, the mass density ρ, the momentum density $\rho\mathbf{u}$ and the energy density E are collocated at the cell centers, whereas the magnetic fields are collocated both at the cell centers and cell interface-centers. We denote the cell-centered magnetic fields by $\mathbf{B}$ and the cell interface-centered fields by $\mathbf{b}$. The geometry is shown in Figure (1.2) for a given cell (i, j). In the following paragraphs we shall briefly explain how the USM scheme is constructed and how it treats the $\nabla \cdot \mathbf{B} = 0$ condition. The scheme can be summarized in the following steps.

(a) The first step is to transform the nonlinear system of MHD equations into a quasi-linearized version (using the primitive variables $\mathbf{V} = (\rho, \mathbf{u}, \mathbf{B}, p)^T$ rather

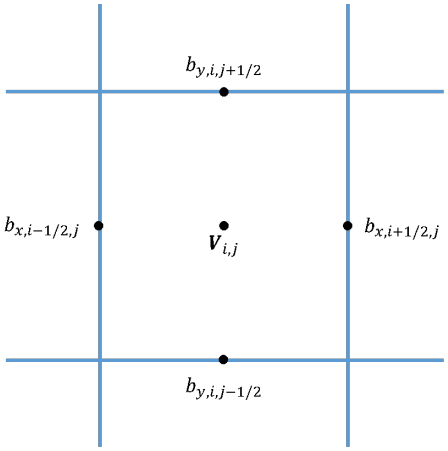


Figure 1.2: The staggered geometry. The variables $\rho, \mathbf{u}$ and p are collocated at the cell face centers. The magnetic fields are collocated both at the cell face centers ($\mathbf{B}$) and the cell interface centers (b_x and b_y). $\mathbf{V}_{i,j}$ contains the cell-centered variables.

than the conservative ones) and then to proceed to the reconstruction-evolution step using a second-order MUSCL-Hancock type Total Variation Diminishing (TVD) algorithm. Throughout this process the necessary multidimensional MHD terms will be included by treating the evolution of the normal magnetic field separately from the other variables.

(b) The first step results in four Riemann states $\mathbf{V}^{n+\frac{1}{2}}_{i,j,\hat{N},S,E,W}$ (as shown in Figure (1.3)) at each cell's boundaries, evolved by half a time-step, with the exception of the normal components of the magnetic field, which remain to be evolved. With the already evolved Riemann states at the interfaces we can obtain the corresponding high-order Godunov fluxes by solving the appropriate (intermediate) Riemann problem. In order to evolve the normal components of the magnetic field, the CT-method is applied using these fluxes, by solving a set of discrete induction equations given in Equation (1.11)

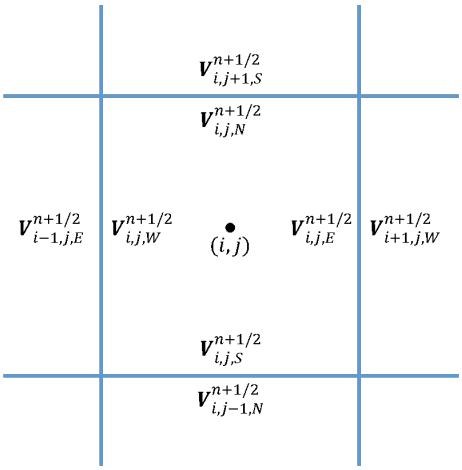


Figure 1.3: Configuration of the Riemann states at a given cell (i, j).

$$
\begin{aligned}
b^{n+\frac{1}{2}}_{x,i+\frac{1}{2},j} &= b^{n}_{x,i+\frac{1}{2},j} - \frac{\Delta t}{2\Delta y}\left[\tilde{E}^{n+\frac{1}{2}}_{z,i+\frac{1}{2},j+\frac{1}{2}} - \tilde{E}^{n+\frac{1}{2}}_{z,i+\frac{1}{2},j-\frac{1}{2}}\right] \\
b^{n+\frac{1}{2}}_{y,i,j+\frac{1}{2}} &= b^{n}_{y,i,j+\frac{1}{2}} - \frac{\Delta t}{2\Delta x}\left[-\tilde{E}^{n+\frac{1}{2}}_{z,i+\frac{1}{2},j+\frac{1}{2}} - \tilde{E}^{n+\frac{1}{2}}_{z,i-\frac{1}{2},j+\frac{1}{2}}\right]
\end{aligned}
\tag{1.11}
$$

The newly evolved values $b^{n+\frac{1}{2}}_{x,i\pm\frac{1}{2},j}$ and $b^{n+\frac{1}{2}}_{y,i,j\pm\frac{1}{2}}$ are then added to the four Riemann states in cell (i, j), $\mathbf{V}^{n+\frac{1}{2}}_{i,j,N,S,E,W}$, which finalizes their half-time updating. These states are then used to calculate the full second-order Godunov fluxes at the cell faces by solving the Riemann problems (RP):

$$
\begin{aligned}
\mathbf{F}^{*,n+\frac{1}{2}}_{i-\frac{1}{2},j} &= \mathrm{RP}(\mathbf{V}^{n+\frac{1}{2}}_{i-1,j,E}, \mathbf{V}^{n+\frac{1}{2}}_{i,j,W}) \qquad & \mathbf{F}^{*,n+\frac{1}{2}}_{i+\frac{1}{2},j} &= \mathrm{RP}(\mathbf{V}^{n+\frac{1}{2}}_{i,j,E}, \mathbf{V}^{n+\frac{1}{2}}_{i+1,j,W}) \\
\mathbf{G}^{*,n+\frac{1}{2}}_{i,j-\frac{1}{2}} &= \mathrm{RP}(\mathbf{V}^{n+\frac{1}{2}}_{i,j-1,N}, \mathbf{V}^{n+\frac{1}{2}}_{i,j,S}) \qquad & \mathbf{G}^{*,n+\frac{1}{2}}_{i,j+\frac{1}{2}} &= \mathrm{RP}(\mathbf{V}^{n+\frac{1}{2}}_{i,j,N}, \mathbf{V}^{n+\frac{1}{2}}_{i,j+1,S})
\end{aligned}
\tag{1.12}
$$

(c) With these fluxes, the cell-centered conservative variables are evolved to the next time step:

$$
\mathbf{U}^{n+1}_{i,j} = \mathbf{U}^{n}_{i,j} - \frac{\Delta t}{\Delta x}\left[\mathbf{F}^{*,n+\frac{1}{2}}_{i+\frac{1}{2},j} - \mathbf{F}^{*,n+\frac{1}{2}}_{i-\frac{1}{2},j}\right] - \frac{\Delta t}{\Delta y}\left[\mathbf{G}^{*,n+\frac{1}{2}}_{i,j+\frac{1}{2}} - \mathbf{G}^{*,n+\frac{1}{2}}_{i,j-\frac{1}{2}}\right] \tag{1.13}
$$

(d) Some additional precautions are required to ensure that the interface-centered

magnetic fields satisfy the solenoidal condition. Using the second-order accurate Godunov fluxes, the electric fields collocated at the cell corners may be constructed, and they are then used to solve the discrete induction equations to evolve the magnetic fields. In the final step, the cell-centered magnetic fields $\mathbf{B}$ are reconstructed by simply taking arithmetic averages of the cell interface values as shown in Equation (1.14).

$$\begin{aligned} B^{n+1}_{x,i,j} &= \frac{b^{n+1}_{x,i+\frac{1}{2},j} + b^{n+1}_{x,i-\frac{1}{2},j}}{2} \\ B^{n+1}_{y,i,j} &= \frac{b^{n+1}_{y,i,j+\frac{1}{2}} + b^{n+1}_{y,i,j-\frac{1}{2}}}{2} \end{aligned} \tag{1.14}$$

This is the step that we would like to modify, in order to improve the interpolation method, in an attempt to obtain divergence-free values for the cell-centered $\mathbf{B}$ as well.

1.3 Interpolation Using Gaussian Processes

As mentioned in the previous section, our main goal here is to improve the last step in the USM algorithm, i.e., the linear interpolation of the components of the magnetic field at the cell centers (see Equation (1.14)). We will replace the current arithmetic averaging method by a more accurate interpolation using Gaussian Processes (GP). Furthermore, we will make use of a particular form of interpolant, which will ensure that the resulting magnetic field remains divergence-free (at least analytically). The interpolation itself will depend on two elements: a hyperparameter ℓ in the definition of the kernel function, and the size of the stencil we sample from. In this section we take a closer look at how GP interpolation works and how these elements may influence our results. What follows has been thoroughly developed in [2, 10, 12, 15] and we only present briefly the elements

necessary for our implementation.

When it comes to data interpolation, the usual methods used are polynomial-based ones. Indeed, these methods are quite intuitive due to their close relation to Taylor expansion types of approximations, and a higher inteprolation accuracy can be simply achieved by considering higher degree polynomials. Unfortunately, there are also several considerable downsides known to accompany polynomial interpolation: it is known to suffer greatly from overfitting issues and oscillations when treating discontinuous data; it can only be applied on stencils of fixed size (thus making the use of unstructured meshes more complicated); and the algorithmic complexity is known to grow with higher orders of accuracy and with dimensionality in FVM. As we shall see, GP interpolation, which belongs to a class of non-polynomial methods, may allow us to overcome these difficulties without increasing the algorithmic complexity.

By definition, GP forms a class of stochastic processes, which sample functions from a space of infinite dimensionality. A Gaussian process is a set of random variables, any finite subset of which presents a joint Gaussian distribution. Two elements are necessary to define a Gaussian process: a mean function $\bar{f}(\mathbf{x}) = \mathbb{E}[f(\mathbf{x})])$ in $\mathbb{R}^N$, as well as a convariance function $K(\mathbf{x}, \mathbf{y})$ over $\mathbb{R}^N \times \mathbb{R}^N$ (symmetric and positive-definite). Within the Gaussian modeling framework, we begin by specifying a prior probability distribution over the function space. In order to predict the quantities of interest at unknown points, the process of GP modeling first trains the observed and known data at its respective coordinates and is then capable of producing a data-informed prediction for the posterior probability distribution at the interpolation points by using Bayes' theorem. This posterior probability distribution comes with two pieces of information: a mean value and a corresponding uncertainty. Furthermore it is possible and quite straightforward

to include a given uncertainty on the prior data and take it into account when calculating the posterior. In our case however, we will only consider the mean value of the posterior in our interpolation method.

In order to fully define the mean function $\bar{f}$ and the covariance $K(\mathbf{x}, \mathbf{y})$ needed to describe our GP, we need some fitting procedure to determine the parameters controlling these functions. If we consider the sample points $\mathbf{x}_i$ with $i = 1, \ldots, N$ (where $\mathbf{f} \equiv [f(\mathbf{x}_1), \ldots, f(\mathbf{x}_N)]^T$ is known), we can calculate and maximize the likelihood $\mathcal{L}$ of observing the values $\mathbf{f}$, given the GP model with its parameters:

$$\mathcal{L} \equiv P(\mathbf{f}) = (2\pi)^{-\frac{N}{2}} \det |\mathbf{K}|^{-\frac{1}{2}} \exp\left[-\frac{1}{2}(\mathbf{f} - \bar{\mathbf{f}})^T \mathbf{K}^{-1} (\mathbf{f} - \bar{\mathbf{f}})\right]. \tag{1.15}$$

In this expression, matrix $\mathbf{K}$ is composed of the kernel function evaluated for all pairs of points: $K_{ij} \equiv K(\mathbf{x}_i, \mathbf{x}_j)$.

In order to make a prediction for the function f at a new unknown point $\mathbf{x}_*$, we use the conditioning property of GP. We consider the joint distribution of the known data $\mathbf{f}$ and the new value f_* as well as the corresponding augmented likelihood function $\mathcal{L}_*$ (of $\mathbf{f}$ and f_* given the GP model):

$$\mathcal{L}_* \equiv P(\mathbf{f}, f_*) = (2\pi)^{-\frac{N+1}{2}} \det |\mathbf{M}|^{-\frac{1}{2}} \exp\left[-\frac{1}{2}(\mathbf{g} - \bar{\mathbf{g}})^T \mathbf{M}^{-1} (\mathbf{g} - \bar{\mathbf{g}})\right]. \tag{1.16}$$

In this expression we have the augmented vectors:

$$\mathbf{g} \equiv [f_*, \mathbf{f}]^T, \quad \bar{\mathbf{g}} \equiv \left[\bar{f}(\mathbf{x}_*), \bar{\mathbf{f}}\right]^T, \tag{1.17}$$

and the $(N+1) \times (N+1)$ augmented covariance matrix:

$$\mathbf{M} = \begin{pmatrix} k_{**} & \mathbf{k}_*^T \\ \mathbf{k}_* & \mathbf{K} \end{pmatrix}. \tag{1.18}$$

Here we use the following notation for the scalar k_{**} and the N-dimensional vector $\mathbf{k}_* = [\mathbf{k}_{*,i}]_{i=1,\ldots,N}$:

$$k_{**} \equiv K(\mathbf{x}_*, \mathbf{x}_*), \quad \mathbf{k}_{*,i} \equiv K(\mathbf{x}_*, \mathbf{x}_i). \tag{1.19}$$

We now apply Bayes' Theorem and the conditioning property to the joint Gaussian prior distribution, which will give us the posterior distribution of the desired function f_*, given the values $\mathbf{f}$:

$$P(f_*|\mathbf{f}) = \left(2\pi U^2\right)^{-\frac{1}{2}} \exp\left[-\frac{(f_* - \bar{f}_*)^2}{2U^2}\right]. \tag{1.20}$$

In this expression, the updated posterior mean function and covariance are respectively:

$$\tilde{f}_* \equiv \bar{f}(\mathbf{x}_*) + \mathbf{k}_*^T \mathbf{K}^{-1} \cdot (\mathbf{f} - \bar{\mathbf{f}}), \tag{1.21}$$

$$U^2 \equiv k_{**} - \mathbf{k}_*^T \mathbf{K}^{-1} \cdot \mathbf{k}_*. \tag{1.22}$$

It is precisely $\tilde{f}_*$ that we are interested in to perform the interpolation. Furthermore, if we choose a constant, zero mean function $\bar{f}(\mathbf{x}) = 0$ (the most general choice) in Equation (1.21), it will be simplified to:

$$\tilde{f}_* \equiv \mathbf{k}_*^T \mathbf{K}^{-1} \cdot \mathbf{f}. \tag{1.23}$$

In what we have described to this point, we only consider interpolating single-

valued functions. We aim however to interpolate a vector-valued quantity - the magnetic field - whose components are not mutually independent. We could consider applying the method as described to the components of the magnetic field separately, however we cannot guarantee that the interpolant will preserve the relation between the components, i.e., the fact that $\mathbf{B}$ is divergence-free. Before further developing the interpolation method for multivariate functions, we note that this first, single-valued method, will still be used within the USM scheme at an intermediate step.

We thus need a vector-valued equivalent of this interpolation method, which also preserves the divergence-free quality of the function. It has been shown [11] that many statistical learning methods can be transformed in such a way simply by constructing an appropriate matrix-valued kernel function (instead of a scalar one). Such a kernel function, which also satisfies the divergence-free condition has been presented in [12] and [10]. Indeed, it is constructed so that its columns define themselves divergence-free vector fields. This is a sufficient condition to guarantee that the result of the interpolation will also be divergence-free, since the latter will be a linear combination of these columns.

In our implementation we use a squared exponential kernel of the form:

$$\phi(\mathbf{x}) = \Sigma^2 \exp(-\frac{||\mathbf{x}||^2}{2\ell^2})) \tag{1.24}$$

The parameters Σ and ℓ in this expression are "hyperparameters" defining the model. The first hyperparameter, Σ, will not affect the value of the posterior mean function, so we can set it to $\Sigma^2 = 1$. On the other hand, the second hyperparameter, ℓ, will play an important role. Indeed, it defines the correlation length scale of the GP model and should reflect the physical length scale of the problem at hand. Typical values of ℓ are usually larger than the minimum grid separation

and even larger than the radius of the interpolation stencil. The optimal value of ℓ will thus vary depending on the grid size and on the problem we consider. It may even be preferable to use different values of the parameter at different locations on the grid.

One may also argue that this choice of smooth kernel function may be inappropriate to treat discontinuous data. Other types of kernel functions with less strict smoothness requirements may be considered, but it has been shown [15] that since they are of finite order of differentiability, their use may lead to noisy, less accurate and unstable solutions.

The construction of the divergence-free kernel itself is done by applying a specific linear differential operator onto our scalar kernel function ϕ of choice. Such an operator is the Hessian H: $(H\phi)_{ij} = \frac{\partial^2 \phi}{\partial x_i \partial x_j}$. For the matrix-valued kernel to be divergence-free, the applied linear operator can be: $\mathcal{L} = H - \mathrm{tr}(H) \cdot I$ [12]. Applying this operator onto the scalar-valued kernel specified above, we obtain the matrix-valued kernel (with n being the dimension of the physical space):

$$\mathbf{K}(\mathbf{x}, \mathbf{y}) = \frac{1}{\ell^2} e^{-\frac{||\mathbf{x}-\mathbf{y}||^2}{2\ell^2}} \left[\left(\frac{\mathbf{x}-\mathbf{y}}{\ell} \right) \left(\frac{\mathbf{x}-\mathbf{y}}{\ell} \right)^T + \left((n-1) - \frac{||\mathbf{x}-\mathbf{y}||^2}{\ell^2} \right) \cdot \mathbf{I} \right] \quad (1.25)$$

We can indeed verify (see Appendix (4.1)) that this kernel function is divergence-free, which will in turn guarantee that the interpolated magnetic field at the cell centers will be divergence-free as well. At this point we make a very important note. The above matrix valued kernel function is only divergence-free analytically. There is, however, no guarantee that it will still be divergence-free numerically. In fact, we can show (see Appendix (4.2)) that the numerical divergence is non-zero. With the GP interpolation method now defined, we can put together the necessary algorithm for our specific application and perform several numerical tests.

Chapter 2

Results

2.1 Implementation

Some precautions are necessary in order to implement divergence-free GP interpolation into the USM scheme. We recall that our goal is to use interface averaged training data (the evolved magnetic fields b_x and b_y, collocated at the cell interface centers) in order to predict (at least analytically) divergence-free values of the magnetic fields at the cell centers, B_x and B_y by using the kernel in Equation (1.25). We notice however, that this requires the simultaneous knowledge of both b_x and b_y at each cell interface-center, and the USM scheme does not supply that information. Indeed, only b_x is evolved along the cell interfaces in the x-direction, and only b_y along the cell interfaces in the y-direction. We thus need to construct an intermediate step in order to find b_x at the interface centers along the y-direction and similarly - b_y along the x-direction. The situation is shown in Figure (2.1).

Our intermediate step consists in a single-valued GP interpolation along each spatial direction separately. Figure (2.2) describes the situation. We choose, as our stencil of sample points, a simple set of four points surrounding each interpolation

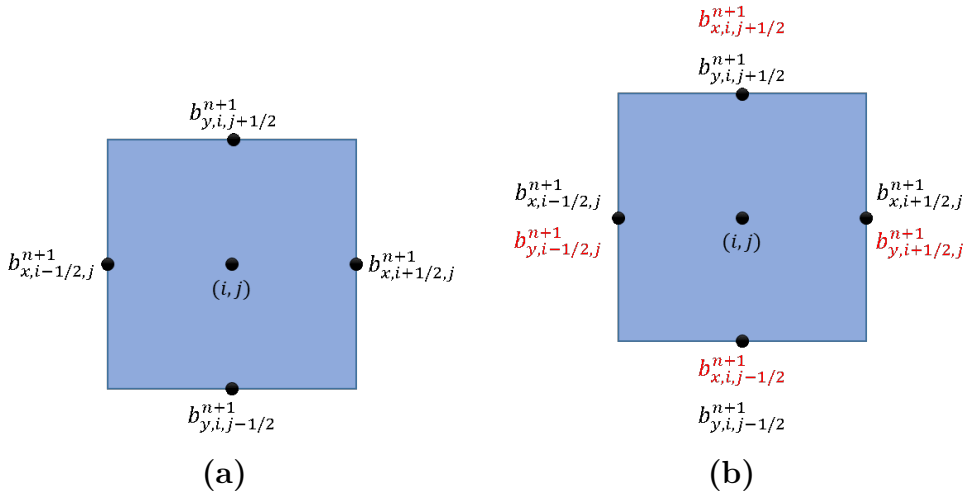


Figure 2.1: After each update, USM provides b_x along the x-faces and b_y along the y-faces as shown in 2.1a. To perform divergence-free GP interpolation, we need both b_x and b_y at each sample point as shown in 2.1b. The missing variables are shown in red.

point (where we wish to calculate the missing field value). At each time step, sweeping over all interface-centered points, first along the x-direction and then along y, we perform single-valued GP interpolation, using Equation (1.23) with zero mean.

Once we have both components of the magnetic field at the cell interface centers, we can proceed to the divergence-free GP interpolation. This time we will use a stencil of sample points as described in Figure (2.3), whose size can be controlled by the parameter $R_{stencil}$, called the GP radius. We then loop over all the cell-centered interpolation points (i, j), and we use Equation (1.23) with the matrix-valued kernel function defined in Equation (1.25) in order to predict $B_x(i, j)$ and $B_y(i, j)$ at those points.

With the newly interpolated cell-centered field $\mathbf{B}$, we also define its corresponding numerical divergence in Equation (2.1).

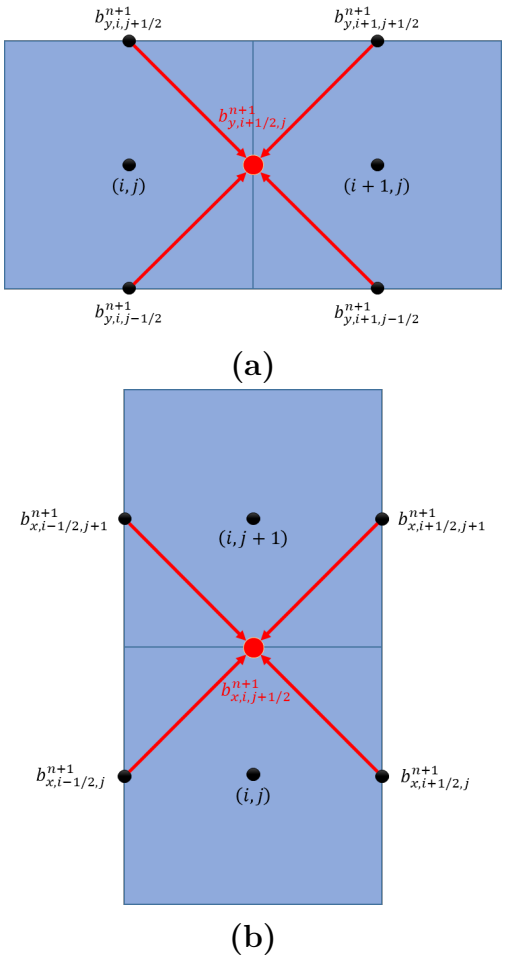


Figure 2.2: Four-point stencils used for single-valued GP interpolation. The unknown variable is shown in red in each case and the red arrows indicate GP interpolation from the sample points to the point where that quantity is unknown.

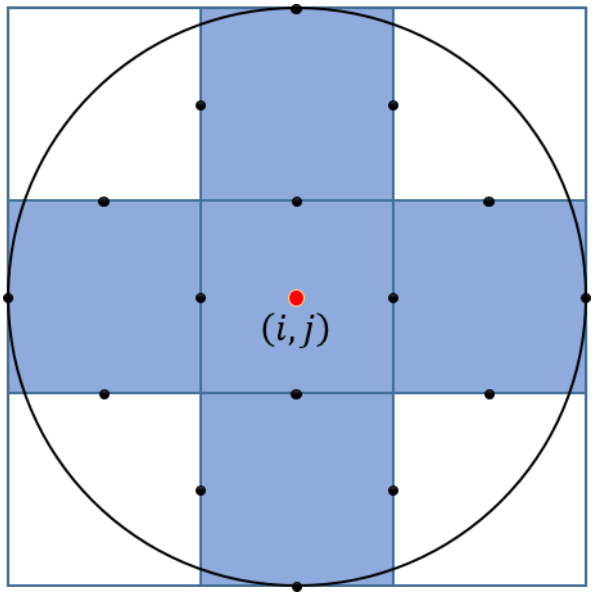


Figure 2.3: Stencil of radius $R_{stencil}$ used for divergence-free GP interpolation. The sample points are shown in black (cell interface centers) and the interpolation point is in red (the cell center at (i, j)). For this choice of $R_{stencil}$ we have a total of 16 sample points.

$$\nabla \cdot \mathbf{B}_{(i,j)} = \frac{B_{x,(i+1,j)} - B_{x,(i-1,j)}}{2\Delta x} + \frac{B_{x,(i,j+1)} - B_{x,(i,j-1)}}{2\Delta y} \tag{2.1}$$

Although we do not expect this quantity to be zero (as explained in Appendix (4.2), we can use it to compare the performance of the original USM scheme with arithmetic averaging (see Equation (1.14)) and the new scheme with GP interpolation.

At this stage we must make an important remark. The process may seem computationally expensive, since the interpolation involves the inversion of a $N_{ker} \times N_{ker}$ matrix, with $N_{ker} = N \times D$ where N is the number of sample points and D is the dimension ($D = 1, 2, 3$). However, looking back at Equation (1.23), we note that both $\mathbf{k}_*$ and $\mathbf{K}$ are independent of the values at the sample points $\mathbf{f}$, but only depend on the distances between pairs of these sample points. This

means that if the locations of these sample points are known in advance (which is the case in the simulations of interest to us), we can calculate $\mathbf{k}_*^T\mathbf{K}^{-1}$ (also referred to as a vector of weights $\mathbf{w}^T = \mathbf{k}_*^T\mathbf{K}^{-1}$) a single time at the beginning of the simulation. If we wish to use AMR (Adaptive Mesh Refinement) during our simulation, we usually specify the maximum level of refinement we wish to use. We can then calculate the weight vectors $\mathbf{w}^T$ for each refinement level and store them, once again, at the beginning of the simulation. This way we only need to compute the matrix inversion once (or as many times as there are levels of refinement), significantly reducing the computation time.

2.2 Numerical Tests

2.2.1 Field Loop Problem

The first test we consider is the problem of the advection of a weakly magnetized field loop, traveling diagonally across the computational domain [7]. In our setup, we set this domain to be $[0, 2] \times [0, 1]$ with periodic boundary conditions, and the corresponding resolution to be 256×256. The initial conditions for the mass density is set to $\rho = 1$ and the pressure is $p = 1$, with a ratio of specific heats $\gamma = 5/3$. With the advection angle defined as $\theta = \tan^{-1}(0.5) \approx 26.57°$, we can write the initial velocity field as:

$$\mathbf{U} = u_0(\cos(\theta), \sin(\theta), 1), \quad u_0 = \sqrt{5} \tag{2.2}$$

The initial condition for the magnetic field here is defined in an indirect way, using the numerical curl of the third component of a specifically defined vector potential A_z:

$$b_x = \frac{\partial A_z}{\partial y} \quad \text{and} \quad b_y = -\frac{\partial A_z}{\partial x}. \tag{2.3}$$

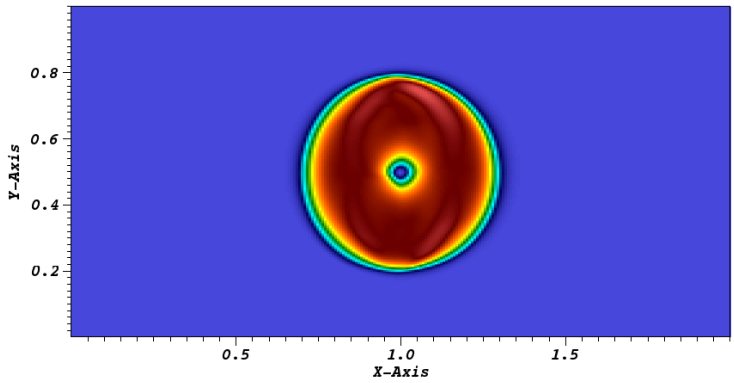
Using a centered numerical difference allows us to obtain initially numerically divergence-free b_x and b_y components at the cell faces. We take this component of the vector potential as:

$$A_z = \begin{cases} A_0(R - r) & \text{if} \quad r \leq R, \\ 0 & \text{otherwise.} \end{cases} \tag{2.4}$$

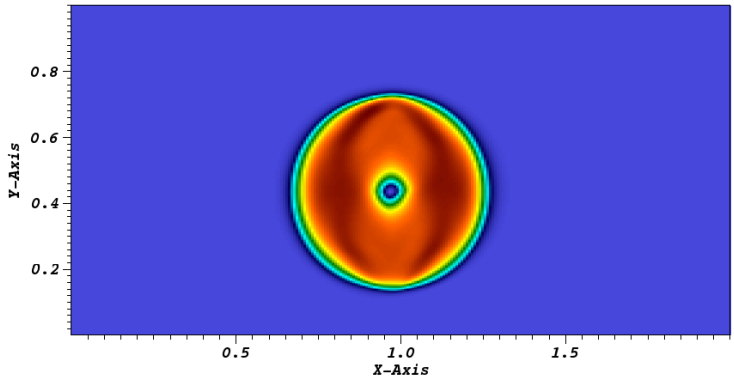
In this expression we have $A_0 = 10^{-3}$ and $R = 0.3$ (which is the loop radius). This choice of parameters results in a divergence-free initial magnetic field (with a numerical divergence of up to 10^{-16}). Furthermore, the strength of the defined magnetic field on the inside of the loop is very weak, which means that the dynamics of the flow will be governed mainly by the gas pressure. The choice of initial parameters is such that one complete advection cycle is performed in time $t = 1$.

Knowing that the loop is supposed to reach its initial position at the end of the simulation, we may test the effect of the parameters used in our GP interpolation. Indeed, as mentioned previously, the choice of the ℓ parameter defining the kernel function will influence the resulting interpolation, and we may see this quite clearly here. Figure (2.4) shows a comparison of the different solutions using the USM scheme without GP interpolation (2.4a); with GP interpolation using $\ell = 1.5\Delta y$ (2.4b); and using $\ell = 8.0\Delta y$ (2.4c) at time $t = 2.0$.

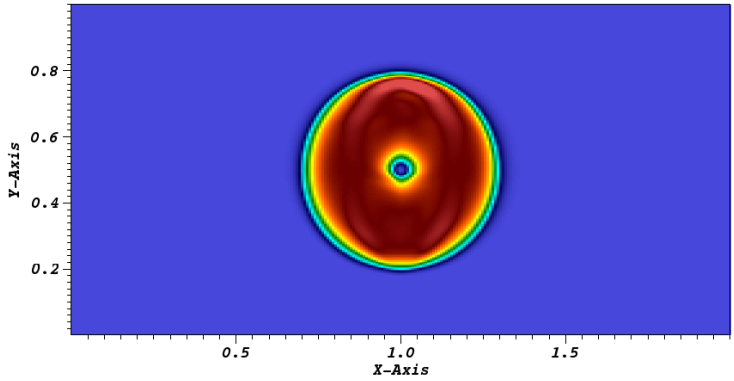
As we can see, the $\ell = 8.0\Delta y$ case leads to a more accurate solution, as the one with $\ell = 1.5\Delta y$ does not seem to reach the original position precisely (instead of returning to $(1.0, 0.5)$, the center of the field loop is approximately situated at $(0.969, 0.438)$). If we further increase the value of ℓ we begin to see



(a) USM with arithmetic averaging at $t = 2.0$



(b) USM with GP interpolation using $\ell = 1.5\Delta y$ at $t = 2.0$



(c) USM with GP interpolation using $\ell = 8.0\Delta y$ at $t = 2.0$

Figure 2.4: Comparison of the magnetic pressure at $t = 2.0$ for the Field Loop problem, without and with GP interpolation and two values of ℓ. The same color scheme is used for values between 5.5×10^{-26} and 5.8×10^{-7} and the resolution is 256×256.

unwanted diffusive effects throughout the computational domain. The chosen value $\ell = 8.0\Delta y$ has not been shown to be optimal, but this suggests that such a value should exist. For further discussion on the optimal value of ℓ, see [15].

A good way to test the performance of our scheme is to also check whether the circular symmetry of the solution is maintained throughout the simulation. We can do this by plotting the contour lines of the z-component of the vector potential we used to define the magnetic field. Figure (2.5) shows a plot of 20 contour lines of A_z at $t = 2.0$, both for the original USM scheme with arithmetic averaging and with GP interpolation. We have used the parameter $\ell = 8.0\Delta y$ in this case. As we can see, the circular symmetry of the solution is well maintained in the latter case.

To verify this quantitaively, we measure the horizontal (d_h) and vertical (d_v) diameters of the outermost contour lines, both for the original case (with arithmetic averaging) and the one using GP interpolation, and calculate the ratio of the vertical diameter to the horizontal one. With the original USM scheme we have:

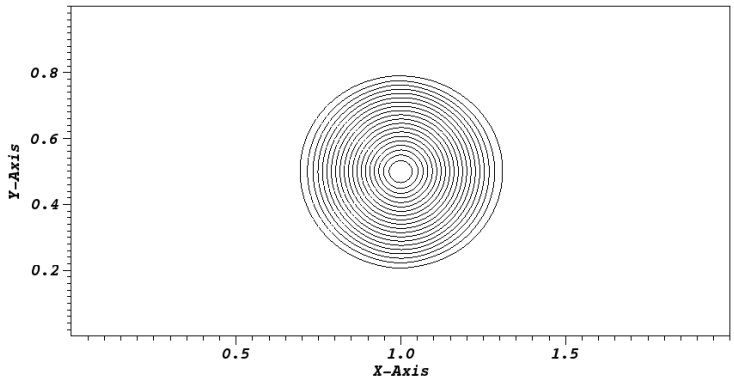
$$\frac{d_v}{d_h} = \frac{0.583}{0.616} \approx 0.946 \tag{2.5}$$

Now with GP interpolation we obtain:

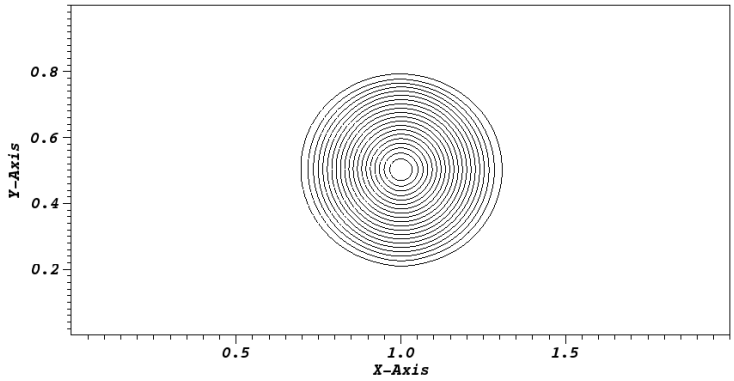
$$\frac{d_v}{d_h} = \frac{0.584}{0.614} \approx 0.951 \tag{2.6}$$

As we can see the two values are very close, with the latter suggesting a slightly better-preserved circular shape. This could potentially be further improved with a better choice of GP parameters.

We can also consider the L1 error on the two components of the magnetic field B_x and B_y calculated at $t = 2.0$. For a given variable q_k, the associated L1 error



(a) USM with arithmetic averaging at $t = 2.0$



(b) USM with GP interpolation using $\ell = 8.0\Delta y$ at $t = 2.0$

Figure 2.5: Comparison of the z-component of the vector potential, A_z at $t = 2.0$ for the Field Loop problem, without GP interpolation and with GP interpolation and a value of the parameter $\ell = 8.0\Delta y$. Plotted are 20 contour lines between -2.16×10^{-6} and 2.7×10^{-4} and the resolution is 256×256.

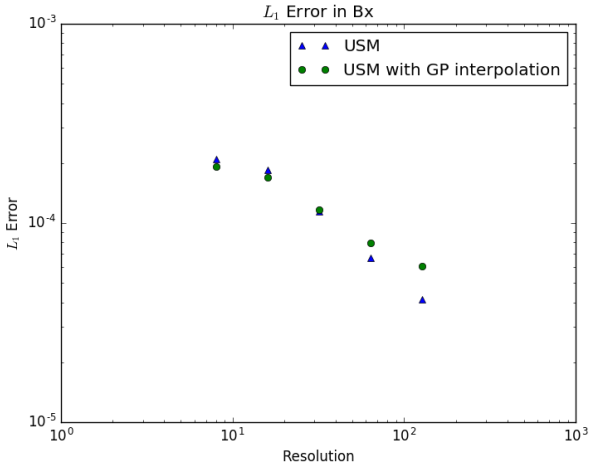
at time $t = n$ is given by:

$$\delta q_k = \Delta x \Delta y \sum_{i,j} |q^n_{k,(i,j)} - q^0_{k,(i,j)}|. \tag{2.7}$$

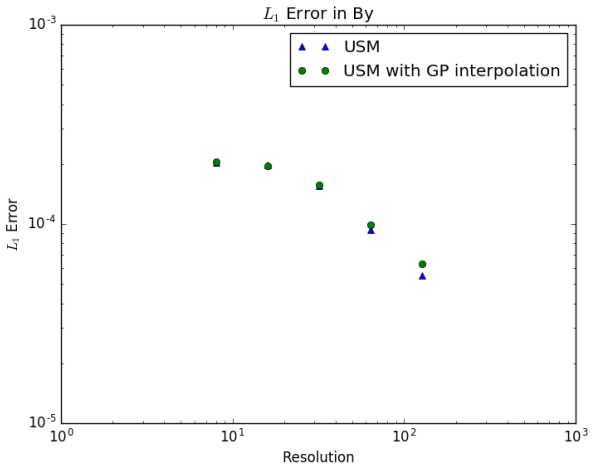
In this expression the summation is over all grid cells (i, j). Figure (2.6) shows these errors on a logarithmic scale. The different resolutions we have used are $N = 8, 16, 32, 64$ and 128. We can see from these plots that there does not seem to be any particular improvement when using GP interpolation, however this can be explained once again by the dependence of GP on its hyperparameter. Since for a given problem the choice for this hyperparameter depends on the grid resolution, ideally we would compare the performance of the scheme with GP interpolation to the original one, by choosing the optimal value of ℓ for each resolution. The values used to plot Figure (2.6) have been chosen on a case by case basis and a more accurate method is a subject of further study. Another possible explanation is that matrix $\mathbf{K}$ becomes ill-conditioned as we take higher resolutions. In [15], quadruple precision was used to overcome this issue, whereas we only use double precision in the current study.

2.2.2 Orszag-Tang Problem

The Orszag-Tang vortex is a typical problem studied to verify the performance of 2D MHD solvers. Although the initial conditions are smooth, the evolution of this highly nonlinear system involves numerous shock wave interactions, traveling at different speeds. The resulting features are thus characterized by complex strong discontinuities throughout the computational domain. We define the latter as $[0, 1] \times [0, 1]$ with a $N \times N$ resolution. In our case we use $N = 512$ with periodic boundary conditions (both in the x- and y-directions). The initial velocity and magnetic fields are given respectively in Equations (2.8) and (2.9)



(a) L1 error on B_x $t = 2.0$



(b) L1 error on B_y at $t = 2.0$

Figure 2.6: Comparison of L1 errors on B_x and B_y at $t = 2.0$ using the USM scheme with arithmetic averaging (blue) and with GP interpolation (green) for resolutions $N = 8, 16, 32, 64$ and 128.

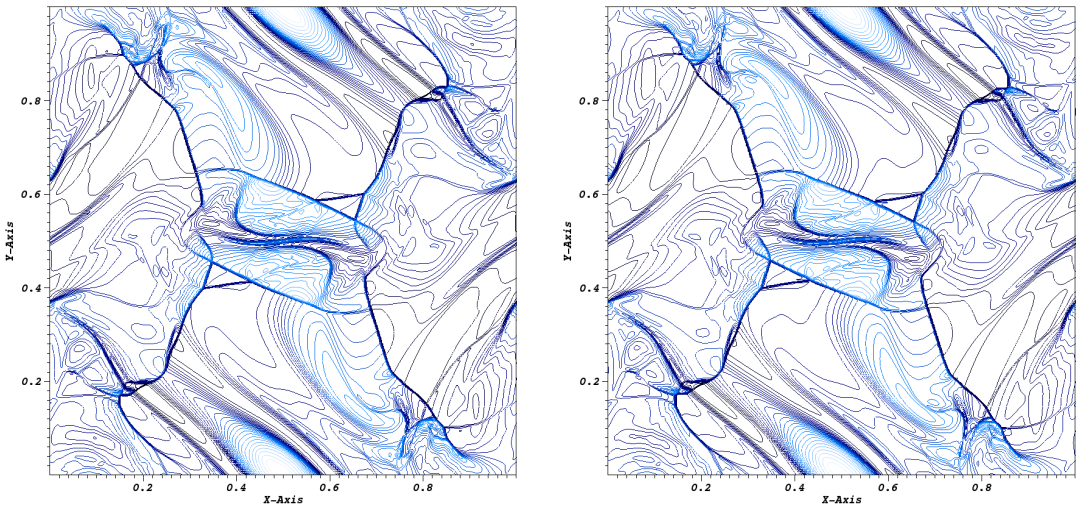
$$\mathbf{U} = u_0 \left(-\sin(\pi y), \sin(2\pi x), 0\right) \tag{2.8}$$

$$\mathbf{B} = B_0 \left(-\sin(\pi y), \sin(4\pi x), 0\right) \tag{2.9}$$

The initial conditions on the density and velocity field are $\rho = 1$ and $u_0 = 1$. We also take the sound speed to be equal to one. Our choice for B_0 is then $B_0 = 1/\gamma$, where $\gamma = 5/3$.

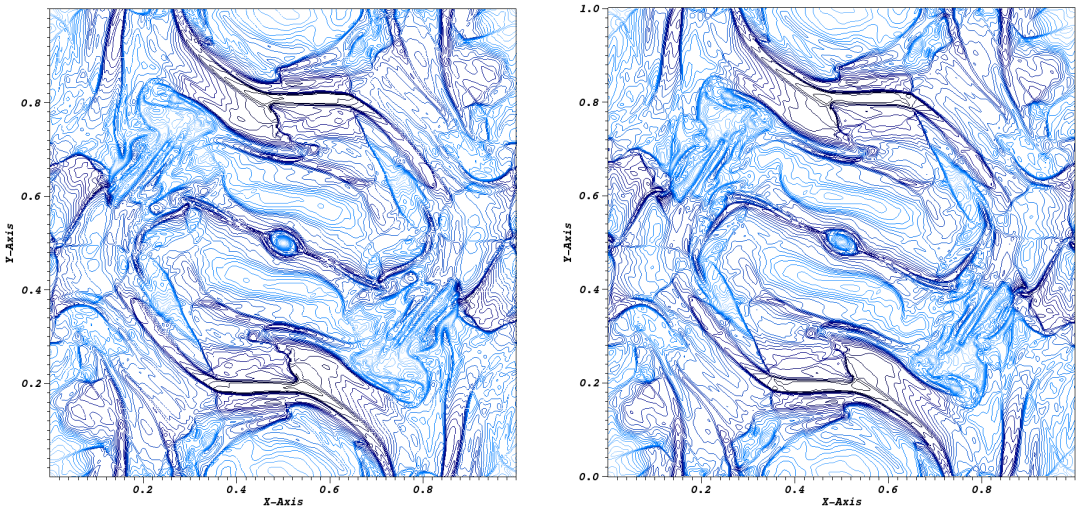
In Figure (2.7) we can see the contour plots of the mass density at $t = 0.5$ and $t = 1.0$ for both the original USM scheme and the one with GP interpolation. We can see that our new addition to the scheme maintains the complex features of the flow, as well as its characteristic symmetry and this is also verified for the other primitive variables.

We have verified that the numerical divergence (both its original definition using the interface-centered values in Equation (1.10) and the new definition using the cell-centered values defined in Equation (2.1)) remains well under control. The GP-interpolation scheme presents some improvement, sometimes as much as one order of magnitude less, but this result is again strongly dependent on the resolution used and the ℓ parameter chosen for the simulation. In the case of $\ell = 1.5\Delta x$ (which is the value we used for the plots here) the order of magnitude remains the same for both definitions of the numerical divergence, with the one using the cell-interface values being as low as 10^{-13}, as expected. The plots of the divergence have been omitted as they are noisy and simply show that this quantity is well under control throughout the computational domain. Our choice of $\ell = 1.5\Delta x$ (since $\Delta x = \Delta y$) has not been verified to be optimal, but produces satisfactory results for our current purposes.



(a) USM with arithmetic averaging at time $t = 0.5$

(b) USM with GP interpolation at time $t = 0.5$



(c) USM with arithmetic averaging at time $t = 1.0$

(d) USM with GP interpolation at time $t = 1.0$

Figure 2.7: Comparison of 30 density contour lines for the Orszag-Tang problem plotted at $t = 0.5$ (with density values between 0.38 and 2.21) and $t = 1.0$ (with density values between 0.18 and 1.8) with and without GP interpolation. The resolution is 512×512 and we have used the Roe Riemann solver and $\ell = 1.5\Delta x$ for the GP interpolation.

Chapter 3

Conclusion

A new implementation using divergence-free Gaussian Process interpolation has been presented and tested within the Unsplit Staggered Mesh MHD solver. Although USM is capable of computing numerically divergence-free magnetic fields at the cell interface-centers, it only predicts these values at the cell centers by performing simple arithmetic averaging. Our additional implementation uses the divergence-free interface-centered values as sample data to train a GP model, to predict B_x and B_y at the cell centers.

We have tested our additional scheme and compared its performance with that of the original USM method with arithmetic averaging. We can see some improvements with GP interpolation, particularly when calculating the numerical divergence using the cell-centered fields. We have seen however that the interpolation is highly sensitive to the hyperparameter ℓ appearing in our squared exponential kernel function. Another important parameter is the sample stencil radius that we use to train the model. Since our method involves two separate GP interpolations done successively, a non-optimal choice of these parameters in the first will lead to inaccuracies in the second.

The optimal hyperparameter may be calculated by maximizing the likelihood

function, and this is a subject of further study. In addition, since the current matrix-valued kernel is only divergence-free analytically, our goal will also be to construct a new such kernel, satisfying $\nabla \cdot \mathbf{B}$ numerically as well.

Chapter 4

Appendix

4.1 Analytically Divergence-Free Interpolant

Here we will show that the interpolant we used is indeed analytically of zero divergence. We recall that the interpolant calculated at a given point $\mathbf{x}$ is given by:

$$\mathbf{t}(\mathbf{x}) = \sum_{k=1}^{N} \psi(\mathbf{x}, \mathbf{x}_k)\mathbf{s}_k. \tag{4.1}$$

In this expression, the index goes over all the sample points $\mathbf{x}_k$, and in order to find the coefficient vector $\mathbf{s}$, we evaluate the interpolant at the sample points themselves:

$$\mathbf{t}(\mathbf{x}_j) = \sum_{k=1}^{N} \psi(\mathbf{x}_j, \mathbf{x}_k)\mathbf{s}_k. \tag{4.2}$$

Let $\mathbf{t}_j = (t_{1j}, t_{2j})$ and $\mathbf{s}_k = (s_{1k}, s_{2k})$. We write vector $\tilde{\mathbf{t}} = (t_{11}, t_{21}, \ldots, t_{1N}, t_{2,N})^T$ and $\tilde{\mathbf{s}} = (s_{11}, s_{21}, \ldots, s_{1N}, s_{2,N})^T$ and we call Ψ the matrix whose row j is the vector $\psi(\mathbf{x}_j, \mathbf{x}_k)$. With this we can write:

$$\tilde{\mathbf{t}} = \Psi\tilde{\mathbf{s}}. \tag{4.3}$$

We can find the coefficients $\tilde{\mathbf{s}}$ simply by evaluating: $\tilde{\mathbf{s}} = \Psi^{-1}\tilde{\mathbf{t}}$, and this will depend on the invertibility of matrix Ψ. By definition, the latter is symmetric and semi-positive-definite, which means that it is invertible and the solution thus exists. Since the sample points and the data therein are constant, we only have to show that every $\psi(\mathbf{x}, \mathbf{x}_k)$ in the expression for the interpolant will be divergence-free, to ensure that the inteprolant will itself satisfy that property.

We recall also that constructing the matrix-valued kernel function could be done by applying a certain linear operator onto a scalar-valued kernel. For a scalar kernel $\phi(r)$ (where $r = ||\mathbf{x} - \mathbf{x}_k||$), and with $\mathbf{x} = (x, y)$ and $\mathbf{x}_k = (x_k, y_k)$, the matrix-valued kernel becomes:

$$\psi(\mathbf{x}, \mathbf{x}_k) = -F(r)\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} - G(r)\begin{bmatrix} (y-y_k)^2 & -(x-x_k)(y-y_k) \\ -(x-x_k)(y-y_k) & (x-x_k)^2 \end{bmatrix}. \tag{4.4}$$

In this expression, we have:

$$F(r) = \frac{1}{r}\phi'(r), \quad G(r) = \frac{1}{r}F'(r). \tag{4.5}$$

In our case we used a squared exponential scalar kernel (depending only on the distance r between points) of the form $\phi(\mathbf{x}, \mathbf{x}_k) = \exp(-||\mathbf{x} - \mathbf{x}_k||^2)$. Plugging this into the matrix-valued kernel in Equation (4.4), we get its two columns ψ_1 and ψ_2:

$$\psi_1 = \begin{bmatrix} 2\exp(-||\mathbf{x} - \mathbf{x}_k||^2) - 4\exp(-||\mathbf{x} - \mathbf{x}_k||^2)(y - y_k)^2 \\ 4\exp(-||\mathbf{x} - \mathbf{x}_k||^2)(x - x_k)(y - y_k) \end{bmatrix} \tag{4.6}$$

$$\psi_2 = \begin{bmatrix} 4\exp(-||\mathbf{x}-\mathbf{x}_k||^2)(x-x_k)(y-y_k) \\ 2\exp(-||\mathbf{x}-\mathbf{x}_k||^2) - 4\exp(-||\mathbf{x}-\mathbf{x}_k||^2)(x-x_k)^2 \end{bmatrix}. \tag{4.7}$$

All that remains to be done now is to show that each of these columns is divergence-free. We shall show this for the first one, as the procedure for the second one is equivalent.

$$\begin{aligned} \nabla \cdot \psi_1 =& \frac{\partial}{\partial x}\left(2\exp(-||\mathbf{x}-\mathbf{x}_k||^2) - 4\exp(-||\mathbf{x}-\mathbf{x}_k||^2)(y-y_k)^2\right) + \\ & \frac{\partial}{\partial y}\left(4\exp(-||\mathbf{x}-\mathbf{x}_k||^2)(x-x_k)(y-y_k)\right) \\ =& 4\exp(-||\mathbf{x}-\mathbf{x}_k||^2)(x-x_k)(2(y-y_k)^2-1) + \\ & 4\exp(-||\mathbf{x}-\mathbf{x}_k||^2)(x-x_k)(1-2(y-y_k)^2) \\ =& 0. \end{aligned} \tag{4.8}$$

We note that for the exact form of the kernel we use in our interpolation, $\phi(\mathbf{x},\mathbf{x}_k) = \exp(-\frac{||\mathbf{x}-\mathbf{x}_k||^2}{2\ell^2})$ this condition will of course still be verified.

4.2 Numerical Divergence of the Matrix-Valued Kernel

In the previous section we showed that analytically the matrix-valued kernel function we are using is indeed divergence-free. However, this is not guaranteed to be true numerically. The numerical partial derivative with respect to x (and equivalently with respect to y) of a given function $f(x,y)$ is given by:

$$\frac{\partial f}{\partial x} = \frac{f(x_{i+1}) - f(x_{i-1})}{2\Delta x}. \tag{4.9}$$

Here the index i denotes the i-th cell (using cell-centered coordinates). We can once again calculate the divergence of the first column ψ_1 to illustrate:

$$\begin{aligned}\nabla \cdot \psi_1 =& \frac{1}{\Delta x}[2\exp(-||\mathbf{x}_{(i+1,j)} - \mathbf{x}_k||^2)(1 - 2(y_j - y_k)^2) \\ & - 2\exp(-||\mathbf{x}_{(i-1,j)} - \mathbf{x}_k||^2)(1 - 2(y_j - y_k)^2)] \\ & + \frac{1}{\Delta y}[4\exp(-||\mathbf{x}_{(i,j+1)} - \mathbf{x}_k||^2)(x_i - x_k)(y_{j+1} - y_k) \\ & - 4\exp(-||\mathbf{x}_{(i,j-1)} - \mathbf{x}_k||^2)(x_i - x_k)(y_{j-1} - y_k)]\end{aligned} \tag{4.10}$$

In this expression, the notation $\mathbf{x}_{(i,j)}$ denotes the coordinates (x_i, y_j). At this point it is clear that we have no guarantee that these terms will cancel out exactly. Numerical divergence will thus remain non-zero. Based on the numerical expression for the divergence of the interpolant (4.10) we could think of ways to ensure this divergence goes to zero. Since none of the terms will disappear explicitly, we could attempt to force this expression to be zero by selecting, for each $\mathbf{x}_{(i,j)}$, only $\mathbf{x}_k$ points such that the coefficient of each exponential term in the expression disappears. This would lead to the following restrictions:

$$x_i - x_k = 0, \quad y_j - y_k = \frac{1}{\sqrt{2}} \tag{4.11}$$

This implies that the chosen point $\mathbf{x}_k$ should have the same x-coordinate as $\mathbf{x}_{(i,j)}$ and that the distance between the two along the y-direction is $\frac{1}{\sqrt{2}}$. These conditions are however too restrictive, limiting our choice of grid size drastically, and allowing us only two sample points, which cannot be expected to improve the interpolation accuracy. A better idea would be to construct the matrix-valued kernel from a numerical point of view altogether, ensuring that its columns are numerically divergence-free. This will be a subject of future study.

Bibliography

[1] Jeremiah U Brackbill and Daniel C Barnes. The effect of nonzero $\nabla \cdot \mathbf{B}$ on the numerical solution of the magnetohydrodynamic equations. *Journal of Computational Physics*, 35(3):426–430, 1980.

[2] M Bishop Christopher. *Pattern Recognition And Machine Learning*. Springer-Verlag New York, 2016.

[3] P Davidson. An introduction to magnetohydrodynamics, cambridge texts in applied mathematics,(2001). doi: 10.1017. *CBO9780511626333*.

[4] Charles R Evans and John F Hawley. Simulation of magnetohydrodynamic flows-a constrained transport method. *The Astrophysical Journal*, 332:659–677, 1988.

[5] Bruce Fryxell, Kevin Olson, Paul Ricker, FX Timmes, Michael Zingale, DQ Lamb, Peter MacNeice, Robert Rosner, JW Truran, and H Tufo. Flash: An adaptive mesh hydrodynamics code for modeling astrophysical thermonuclear flashes. *The Astrophysical Journal Supplement Series*, 131(1):273, 2000.

[6] Thomas A Gardiner and James M Stone. An unsplit godunov method for ideal mhd via constrained transport. *Journal of Computational Physics*, 205(2):509–539, 2005.

[7] Thomas A Gardiner and James M Stone. An unsplit godunov method for ideal mhd via constrained transport. *Journal of Computational Physics*, 205(2):509–539, 2005.

[8] Tamas I Gombosi, Kenneth G Powell, and Darren L De Zeeuw. Axisymmetric modeling of cometary mass loading on an adaptively refined grid: Mhd results. *Journal of Geophysical Research: Space Physics*, 99(A11):21525–21539, 1994.

[9] Dongwook Lee and Anil E Deane. An unsplit staggered mesh scheme for multidimensional magnetohydrodynamics. *Journal of Computational Physics*, 228(4):952–975, 2009.

[10] I Macêdo and R Castro. Learning divergence-free and curl-free vector fields with matrix-valued kernels. *Instituto Nacional de Matematica Pura e Aplicada, Brasil, Tech. Rep*, 2008.

[11] Charles A Micchelli and Massimiliano Pontil. On learning vector-valued functions. *Neural computation*, 17(1):177–204, 2005.

[12] Francis J Narcowich and Joseph D Ward. Generalized hermite interpolation via matrix-valued conditionally positive definite functions. *Mathematics of Computation*, 63(208):661–687, 1994.

[13] Francis J Narcowich and Joseph D Ward. Generalized hermite interpolation via matrix-valued conditionally positive definite functions. *Mathematics of Computation*, 63(208):661–687, 1994.

[14] S. A. Orszag and C.-M. Tang. Small-scale structure of two-dimensional magnetohydrodynamic turbulence. *Journal of Fluid Mechanics*, 90:129–143, January 1979.

[15] Adam Reyes, Dongwook Lee, Carlo Graziani, and Petros Tzeferacos. A new class of high-order methods for fluid dynamics simulations using gaussian process modeling: One-dimensional case. *Journal of Scientific Computing*, pages 1–38, 2017.

[16] Gábor Tóth. The $\nabla \cdot \mathbf{B} = 0$ constraint in shock-capturing magnetohydrodynamics codes. *Journal of Computational Physics*, 161(2):605–652, 2000.