Properties of Divergence-Free Kernel Methods for Approximation and Solution of Partial Differential Equations

by

Arthur Araujo Mitrano

A Dissertation Presented in Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

Approved June 2016 by the
Graduate Supervisory Committee:

Rodrigo Platte, Chair
Grady Wright
Bruno Welfert
Anne Gelb
Rosemary Renaut

ARIZONA STATE UNIVERSITY

August 2016

ABSTRACT

Divergence-free vector field interpolants properties are explored on uniform and scattered nodes, and also their application to fluid flow problems. These interpolants may be applied to physical problems that require the approximant to have zero divergence, such as the velocity field in the incompressible Navier-Stokes equations and the magnetic and electric fields in the Maxwell's equations. In addition, the methods studied here are meshfree, and are suitable for problems defined on complex domains, where mesh generation is computationally expensive or inaccurate, or for problems where the data is only available at scattered locations.

The contributions of this work include a detailed comparison between standard and divergence-free radial basis approximations, a study of the Lebesgue constants for divergence-free approximations and their dependence on node placement, and an investigation of the flat limit of divergence-free interpolants. Finally, numerical solvers for the incompressible Navier-Stokes equations in primitive variables are implemented using discretizations based on traditional and divergence-free kernels. The numerical results are compared to reference solutions obtained with a spectral method.

DEDICATION

To my parents and Lin.

ACKNOWLEDGMENTS

First, I would like to thank my committee members for all the help, time and valuable additions towards this dissertation.

I also thank Grady Wright for suggesting the use of polynomial stream functions as a way to derive divergence-free finite difference schemes and many others fruitful discussions that improved this document. I'm grateful to Varun Shankar for explaining the importance of projections schemes for incompressible fluid flows and suggesting many references in the literature.

A special thanks has to go to my advisor, Rodrigo Platte, for his continuous effort and dedication, not only concerning this dissertation, but also during my whole PhD. I appreciate his endless patience and detailed explanations in the middle of our meetings.

Finally, I show my gratitude to my parents and my girlfriend Lin for all the eternal emotional support provided in and out of my PhD.

TABLE OF CONTENTS

CHAPTER Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

Chapter 1

INTRODUCTION

In this dissertation, we explore properties of solenoidal vector field interpolants on uniform and scattered nodes and their application to fluid flow problems. These interpolants can be applied to physical problems that require the approximant to have zero divergence, such as the velocity field in the incompressible Navier-Stokes equations and the magnetic and electric fields in the Maxwell's equations. In addition, the methods we explore here are meshfree, and are suitable for problems defined on complex domains, where mesh generation is computationally expensive or inaccurate, or for problems where the data is only available at scattered locations.

Scattered data interpolation and meshfree methods have been in fast development in the past years. This growth is due to important applications such as terrain modeling in geology; surface reconstruction in computer graphics; fluid-structure interaction in engineering; numerical solution of PDEs in applied mathematics; option pricing in mathematical finance; and many others, where rectangular grids or triangular meshes are difficult to implement or not cost effective. Radial basis functions (RBFs) approximation, in particular, have been applied successfully to a wide range of problems.

The origin of RBF interpolation is usually credited to Hardy, due to his 1971 original paper [35]. However, existence, uniqueness, and stability results date back to 1930s (Bochner [3] and Schoenberg [60]). Other notable papers on RBF methods are due to Kansa [39, 40], where treatment of PDEs was introduced. Many of the theoretical and practical results for RBF interpolation has been well documented in the literature, in particular in the monographs by Buhmann [4], Iske [38], Wendland

[69], Fasshauer [14] and the more recent [17]. The last mentioned reference presents RBF methods as a generalization of pseudospectral methods, which is the point of view we consider in our work as well.

As previously mentioned, in physical applications such as fluid mechanics, meteorology, and electromagnetics, vectorial data needs to be interpolated on a given set of nodes. This can be done by interpolating each component of the vector field independently. However, it reasonable to think that when the field is divergence-free or curl-free, more accurate approximations might be obtained if the connection between the components of the vector field is taken into consideration.

The present work is motivated by incompressible fluid flow simulations with constant density, where the velocity field is divergence-free because of mass conservation. It is well known that instability can arise in incompressible fluid flow simulations if the divergence-free condition is not met [28]. When approximations are not divergence-free, projection methods applied at each time step can enforce numerical mass conservation [6, 37]. An alternative to those projection methods is to use a divergence-free basis to approximate such vector fields, as proposed in [26].

Divergence-free radial kernels were explored in [1, 9, 10, 33, 34] using a variational spline setting. In 1994, matrix-valued radial basis functions (RBFs) were introduced to approximate generalized interpolation problems on scattered data [48]. In that framework, the divergence-free kernels are generated using linear side conditions. Stability estimates for those interpolants were given in [45, 48] and improved in [25], while error estimates for functions on native spaces were derived in [44] and later extended for rougher functions in [23], where Sobolev-type error estimates were derived. Similar divergence-free interpolants were developed in [49], where the interpolant fits a vector field of zero divergence that is tangent to an orientable surface. Error and stability estimates for those interpolants on the sphere were presented in [24].

1.1 The Contribution of This Work

This dissertation presents several results that have not yet been reported by others in the literature. Part of this work has been published by the author in [47].

This is the first work to compare *accuracy* between approximations by traditional RBF methods, in which the components of a vector field are treated independently, and by divergence-free kernel methods. Not surprisingly, the latter is more accurate, in particular in the finite difference mode. The main improvement, however, is the directions of the derivatives that are constrained by the zero divergence condition. Approximations in the other directions decay at the same rate as for traditional RBF methods. This is only true when the field being interpolated is solenoidal, otherwise divergence-free approximations will fail to converge. An additional comparison is made with divergence-free polynomial approximation and similar convergence patterns are observed. For small stencil sizes, exact finite difference weights are derived and convergence rates are proved using symbolic computations.

Another contribution is a numerical study of the *Lebesgue constants* for divergence-free approximations and their dependence on node placement. Lebesgue constants are a measure of the sensitivity of the approximation process to perturbations on the data. It is well known that standard global RBF approximations are very sensitive to node placement, as reported in [51]. Here we show a similar trend for divergence-free approximations. It is also shown that clustering the nodes more densely near the boundaries is an effective strategy to stabilize the interpolation process for global approximations.

The *flat limit* ($\varepsilon \to 0$) of divergence-free interpolants is also explored. As for standard RBF approximations, we provide strong numerical evidence, and proof for

small number of points using symbolic computations, that the limit of these interpolants is also a polynomial. What makes this study challenging, even though, such limits are expected to exist and be unique, is their difficult computation because the basis functions used in the expansions (without special treatments) become extremely ill-conditioned as $\varepsilon \to 0$. To circumvent this difficulty, the contour Padé algorithm introduced in [20], was adapted to the divergence-free case.

Finally, numerical solvers for the *incompressible Navier-Stokes equations* in primitive variables are implemented using discretizations based on traditional RBFs and divergence-free kernels. Two standard problems are considered: the lid driven cavity flow and the buoyancy driven cavity flow. For comparison we use spectral collocation to generate reference solutions. The difficulty in using divergence-free finite differences to approximate derivatives is that the velocity field at each time step must be projected into a divergence-free space. Truncation errors in the projection step may lead to inaccurate results. Although our implementation is able to simulate certain flows, improvements are needed to make the code more broadly applicable.

1.2 Organization

The remaining of this monograph is organized as follows. In Chapter 2 we review the relevant and basic concepts of standard RBF and divergence-free approximations in two and three dimensions, including the derivation of finite difference weights. Chapter 3 presents most of the contributions listed above (first three items). In this chapter we present the numerical study of several properties of divergence-free based approximations, including accuracy of finite difference formulas, Lebesgue constants and the flat limit. In Chapter 4 we use localized interpolants based on traditional radial basis function and divergence-free kernels introduced in Chapter 3 to simulate incompressible fluid flows. Particular attention is given to the projection step. Final

remarks and future directions are presented in Chapter 5.

Chapter 2

BACKGROUND

This chapter presents fundamental results related to RBF interpolation. Of particular interest is the theory that establishes existence and uniqueness of scattered data interpolants and their dependence on the shape parameter. For convenience, the most relevant theorems are provided but not proved. Additional details on their derivation and proofs may be found in the monographs of Wendland [69] and Fasshauer [14], for instance. In addition, we review main results from [48, 49], which provide the main foundation for divergence-free interpolation of vector fields using positive-definite kernels. The concept of divergence-free polynomial interpolation for two-dimensional vector fields is also introduced. Besides this, differentiation matrices and finite difference operators are derived in this chapter.

2.1 Radial Basis Function Interpolation

In many scientific applications, it is necessary to infer processes only by measured data at specific locations. One way to approximate those processes is to use interpolation. Specifically, given the information $f_1, \ldots, f_N \in \mathbb{R}$ at the locations $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N \in \mathbb{R}^d$, we look for function s that satisfies $s(\boldsymbol{x}_i) = f_i$ for $i = 1, \ldots, N$. Even more desirable is the possibility to apply calculus operations to the interpolant s, requiring certain smoothness.

For the univariate case, polynomials of degree up to $N-1$ form a good approximant space, $\mathcal{P}^{\mathbb{R}}_{N-1}$. The functions are smooth and can be uniquely determined by any given data $(\boldsymbol{x_i}, f_i)$, $i = 1, \ldots, N$. To generalize this concept to scattered data interpolation, we consider Haar spaces.

Definition 1. Suppose that $\Omega \subseteq \mathbb{R}^d$ contains at least N points. Let $\mathcal{V} \subseteq \mathcal{C}(\Omega)$ be an N-dimensional vector space. Then $\mathcal{V}$ is called an N-dimensional *Haar space* on Ω if for any distinct points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N \in \Omega$ and any $f_1, \ldots, f_N \in \mathbb{R}$ there exists exactly one function $s \in \mathcal{V}$ with $s(\boldsymbol{x}_i) = f_i$, $i = 1, \ldots, N$.

Haar spaces guarantee the existence and uniqueness of an interpolant depending only on the number of points and space dimension, independently of the data or where it is sampled. Surprisingly, multivariate polynomials do not form a Haar space on $\mathbb{R}^d$ for $d > 1$. This is due to the Mairhuber-Curtis theorem.

Theorem 1 (Mairhuber-Curtis)**.** *Suppose that $\Omega \subseteq \mathbb{R}^d$, $d \geq 2$, contains an interior point. Then there exists no Haar space on Ω of dimension $N \geq 2$.*

Theorem 1 not only shows the impossibility to interpolate any data with multivariate polynomials, but also with other bases that do not depend on the given data. Next we will introduce an interpolation process that explicitly depends on the given data location (nodes) and is uniquely determined.

2.1.1 Existence of Radial Basis Function Interpolants

The Mairhuber-Curtis theorem states that it is not possible to find a Haar space in the multivariate setting. Thus, to solve the scattered interpolation problem, the approximant space must depend at least on the data location $\{\boldsymbol{x}_k\}_{k=1}^N$. A common way to address this issue is to consider a fixed *kernel* $\Phi \colon \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ and construct the interpolant as

$$s(\boldsymbol{x}) = \sum_{k=1}^{N} \alpha_k \Phi(\boldsymbol{x}, \boldsymbol{x}_k), \tag{2.1}$$

where the interpolation conditions

$$s(\boldsymbol{x}_k) = f_k, \quad k = 1, \ldots, N, \tag{2.2}$$

lead to the following linear system for the coefficients α_k:

$$\underbrace{\begin{bmatrix} \Phi(\boldsymbol{x}_1, \boldsymbol{x}_1) & \cdots & \Phi(\boldsymbol{x}_1, \boldsymbol{x}_N) \\ \vdots & \ddots & \vdots \\ \Phi(\boldsymbol{x}_N, \boldsymbol{x}_1) & \cdots & \Phi(\boldsymbol{x}_N, \boldsymbol{x}_N) \end{bmatrix}}_{A_{\Phi,X}} \underbrace{\begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_N \end{bmatrix}}_{\boldsymbol{\alpha}} = \underbrace{\begin{bmatrix} f_1 \\ \vdots \\ f_N \end{bmatrix}}_{\boldsymbol{f}}. \tag{2.3}$$

The *interpolation matrix* $A_{\Phi,X}$ depends on the *centers* $X = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$ and the kernel Φ. The challenge is to choose Φ that guarantees a nonsingular interpolation matrix, and therefore a unique interpolant. Although invertibility is sufficient for the well-posedness of the scattered data interpolation problem, it is easier to characterize positive definite matrices of the form $A_{\Phi,X}$ using the concept of *positive definite kernels.*

Definition 2. A continuous kernel $\Phi\colon \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{C}$ is *positive semi-definite* if, for all $N \in \mathbb{N}$, all pairwise distinct centers $X = \{\boldsymbol{x}_1 \ldots, \boldsymbol{x}_N\} \subset \mathbb{R}^d$, and all $\boldsymbol{\alpha} \in \mathbb{C}^N$, the quadratic form

$$\sum_{j=1}^{N} \sum_{k=1}^{N} \alpha_j \overline{\alpha_k} \Phi(\boldsymbol{x}_j, \boldsymbol{x}_k)$$

is non-negative. The kernel Φ is *positive definite* if the quadratic form is positive for all $\boldsymbol{\alpha} \in \mathbb{C}^N \setminus \{\boldsymbol{0}\}$.

In other words, provided a positive definite kernel is used, a unique interpolant can be found. Fortunately, much has been done to characterize special cases of those kernels: *positive semi-definite functions* by Bochner [3] and *positive semi-definite radial functions* by Schoenberg [60]. Bellow we list the necessary definitions and the characterization results for convenience of the reader. Details and proofs of those characterizations, can be found in the monograph by Wendland [69].

Definition 3. The multivariate function $\widetilde{\Phi}\colon \mathbb{R}^d \to \mathbb{C}$ is a *positive (semi-)definite function* if the associated kernel $\Phi(\boldsymbol{x}, \boldsymbol{y}) := \widetilde{\Phi}(\boldsymbol{x} - \boldsymbol{y})$, $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^d$, is positive (semi-)definite.

Definition 4. The univariate function $\varphi\colon [0,\infty) \to \mathbb{R}$ is a *positive (semi-)definite radial function* if the corresponding multivariate function $\widetilde{\Phi}(\boldsymbol{x}) \coloneqq \varphi(\|\boldsymbol{x}\|)$, $\boldsymbol{x} \in \mathbb{R}^d$, is positive (semi-)definite.

Theorem 2 (Bochner)**.** *A continuous function $\widetilde{\Phi}\colon \mathbb{R}^d \to \mathbb{C}$ is positive semi-definite if and only if it is the Fourier transform of a finite non-negative Borel measure μ on $\mathbb{R}^d$, i.e.,*

$$\widetilde{\Phi}(\boldsymbol{x}) = \widehat{\mu}(\boldsymbol{x}) = \frac{1}{\sqrt{(2\pi)^d}} \int_{\mathbb{R}^d} e^{-i\boldsymbol{x}^\intercal \boldsymbol{\omega}} d\mu(\boldsymbol{\omega}), \qquad \boldsymbol{x} \in \mathbb{R}^d.$$

Bochner's theorem characterizes positive semi-definite functions in terms of Fourier transforms. Schoenberg's characterization uses *completely monotone functions* to classify all positive semi-define radial functions on all $\mathbb{R}^d$, avoiding computation of Fourier transforms of complicated functions.

Definition 5. A continuous function $\varphi\colon [0,\infty) \to \mathbb{R}$ is *completely monotone* if $\varphi \in \mathcal{C}^\infty((0,\infty))$ and

$$(-1)^\ell \varphi^{(\ell)}(r) \geq 0$$

for all $\ell \in \mathbb{N} \cup \{0\}$ and all $r > 0$.

Theorem 3 (Schoenberg)**.** *A function φ is completely monotone on $[0,\infty)$ if and only if $\widetilde{\Phi} \coloneqq \varphi(\|\cdot\|^2)$ is positive semi-definite on every $\mathbb{R}^d$.*

The characterization of positive semi-definite kernels is not enough to guarantee existence and uniqueness of the interpolation problem. However, for most commonly used kernels, it is possible to determine whether a positive semi-definite kernel is also positive definite. Nevertheless, developments towards a complete integral characterization of positive definite kernels are available in [5].

Although there are several kernels that are positive definite, there are still interesting and commonly used kernels that are not. The interpolation form (2.1) can be augmented with multivariate polynomials such that the scattered data interpolation problem is still well-posed for a larger set of kernels. Consider the interpolant of the data $\{(\boldsymbol{x}_k, f_k)\}_{k=1}^N$ of the form

$$s(\boldsymbol{x}) = \sum_{k=1}^{N} \alpha_k \Phi(\boldsymbol{x}, \boldsymbol{x}_k) + \sum_{k=1}^{Q} \beta_k p_k(\boldsymbol{x}). \tag{2.4}$$

where $p_1, \dots p_Q$ is a basis of $\mathcal{P}_{m-1}^{\mathbb{R}^d}$. Due to the additional degrees of freedom introduced by the extra polynomial bases, the interpolation conditions (2.2) are complemented by Q vanishing moment conditions

$$\sum_{k=1}^{N} \alpha_k p_j(\boldsymbol{x}_k) = 0, \quad j = 1, \dots, Q. \tag{2.5}$$

The conditions (2.2) and (2.5) lead to the linear system

$$\underbrace{\begin{bmatrix} A_{\Phi,X} & P_X \\ P_X{}^\intercal & 0 \end{bmatrix}}_{\widetilde{A}_{\Phi,X}} \begin{bmatrix} \boldsymbol{\alpha} \\ \boldsymbol{\beta} \end{bmatrix} = \begin{bmatrix} \boldsymbol{f} \\ \boldsymbol{0} \end{bmatrix}, \tag{2.6}$$

where $A_{\Phi,X}$ is as before, and $(P_X)_{kj} = p_j(\boldsymbol{x}_k)$ with $j = 1, \dots, Q$ and $k = 1, \dots, N$. The interpolant (2.4) will exist and be unique if the matrix $\widetilde{A}_{\Phi,X}$ is invertible. To find a condition on Φ and the centers X, we need the concept of *conditionally positive definite* and *m-unisolvent set*.

Definition 6. A continuous kernel $\Phi\colon \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{C}$ is *conditionally positive semi-definite of order m* if, for all $N \in \mathbb{N}$, all pairwise distinct centers $\boldsymbol{x}_1, \dots, \boldsymbol{x}_N \in \mathbb{R}^d$, and all $\boldsymbol{\alpha} \in \mathbb{C}^N$ satisfying

$$\sum_{k=1}^{N} \alpha_k p(\boldsymbol{x}_k) = 0 \quad \text{for any } p \in \mathcal{P}_{m-1}^{\mathbb{C}^d},$$

the quadratic form

$$\sum_{j=1}^{N}\sum_{k=1}^{N}\alpha_j\overline{\alpha_k}\Phi(\boldsymbol{x}_j,\boldsymbol{x}_k)$$

is non-negative. The kernel Φ is said to be *conditionally positive definite of order* m if the quadratic form is positive, unless $\boldsymbol{\alpha}$ is zero.

Similar to Definition 3 and Definition 4, one can define *conditionally positive (semi-)definite functions* and *conditionally positive (semi-)definite radial functions*.

Definition 7. A set of points $X = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\} \subset \mathbb{R}^d$ is *m-unisolvent* if the only polynomial in $\mathcal{P}_m^{\mathbb{C}^d}$ interpolating the zero data on X is the zero polynomial.

Theorem 4. *Suppose that Φ is conditionally positive definite of order m and X is $(m-1)$-unisolvent set of centers. Then the system (2.6) is uniquely solvable.*

Proof. Assume that $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ satisfy

$$A_{\Phi,X}\boldsymbol{\alpha} + P_X\boldsymbol{\beta} = 0,$$
$$P_X{}^{\intercal}\boldsymbol{\alpha} = 0.$$

Hence,

$$\boldsymbol{\alpha}^{\intercal}A_{\Phi,X}\boldsymbol{\alpha} + \boldsymbol{\alpha}^{\intercal}P_X\boldsymbol{\beta} = \boldsymbol{\alpha}^{\intercal}A_{\Phi,X}\boldsymbol{\alpha} + (P_X{}^{\intercal}\boldsymbol{\alpha})^{\intercal}\boldsymbol{\beta} = \boldsymbol{\alpha}^{\intercal}A_{\Phi,X}\boldsymbol{\alpha} = 0.$$

Therefore, since Φ is conditionally positive definite, $\boldsymbol{\alpha} = \mathbf{0}$. Moreover, due to $(m-1)$-unisolvency,

$$P_X\boldsymbol{\beta} = \mathbf{0} \implies \boldsymbol{\beta} = \mathbf{0}.$$

This shows that $\mathbf{0}$ is the only element of the null space of $\widetilde{A}_{\Phi,X}$, i.e., $\widetilde{A}_{\Phi,X}$ is non-singular. □

A generalization of Bochner's theorem for conditionally positive semi-definite functions is shown in [63] and a characterization similar to Schoenberg's theorem can

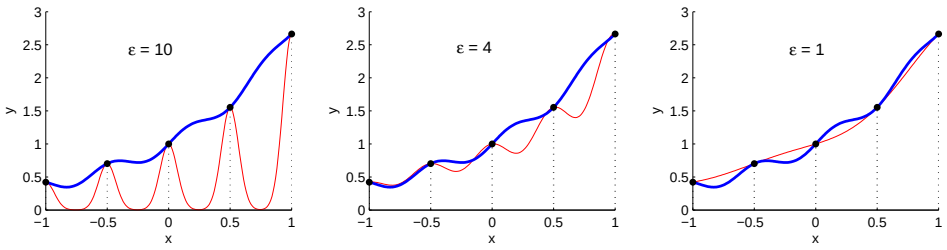

Figure 2.1: Gaussian RBF interpolation of $f(x) = \exp(x) + 0.1\sin(10x)$ using three values of ε.

be found in [32]. For more details and conditions under Φ the reader is refered to the monographs [38, 69].

2.1.2 Example and Properties of Radial Basis Function Interpolation

Due to its simple structure and straightforward implementation, radial kernels are often used in practical computations. In this case, $\Phi(\boldsymbol{x}, \boldsymbol{y}) = \varphi(\varepsilon\|\boldsymbol{x} - \boldsymbol{y}\|)$. The *shape parameter* ε controls how wide or narrow the kernel is. Examples of radial functions that result in unique interpolants are

$$\begin{aligned}
\varphi(r) &= \exp(-r^2) && \text{Gaussians,}\\
\varphi(r) &= \sqrt{1+r^2} && \text{multiquadrics,}\\
\varphi(r) &= 1/\sqrt{1+r^2} && \text{inverse multiquadrics,}\\
\varphi(r) &= 1/(1+r^2) && \text{inverse quadratics.}
\end{aligned}$$

The convergence properties of a RBF expansion depend on the smoothness of φ and the node distribution. The functions in the list above are all analytic and lead to approximations that converge exponentially when the target function is sufficiently smooth [51].

The parameter ε plays a key role in the accuracy of the approximation and conditioning of the interpolation matrix. Figure 2.1 shows the role of ε and how it

changes the interpolant. Larger values of ε lead to more localized interpolants, and smaller values result in global approximations. In the limit $\varepsilon \to 0$, it was shown in [11], for the univariate case, and in [42, 43, 59], for multivariate interpolation, that smooth RBF interpolants converge to polynomial interpolants on the same nodes. This is known as the *flat limit* and one of the goals in this dissertation is to explore the flat limit of divergence-free interpolants. A detailed study in this direction is presented in the next chapter.

Basis functions of finite smoothness are also popular. They include linear splines, $\varphi(r) = r$, cubic splines $\varphi(r) = r^3$ and compactly supported radial functions, also known as Wendland's functions [69]. Due to singularities in the RBF expansion (jump in derivatives), these functions lead to approximations that converge algebraically, with the order depending on the number of smooth derivatives. For simplicity, we focus only on Gaussians and inverse multiquadrics.

2.1.3 Application to Partial Differential Equations

One can use the collocation approach to compute solutions of partial differential equations using RBF interpolants. Similar to pseudospectral methods, discretized differential operators are used to approximate differential operators present on PDEs. To construct a matrix $D_{\mathcal{L}}$ that discretizes a linear operator $\mathcal{L}$, we apply the continuous operator $\mathcal{L}$ to the interpolant (2.1),

$$\mathcal{L}s(\boldsymbol{x}) = \sum_{k=1}^{N} \alpha_k \mathcal{L}\Phi(\boldsymbol{x}, \boldsymbol{x}_k).$$

Evaluating this last expression at the collocation points $X = \{\boldsymbol{x}_i\}_{i=1}^{N}$ leads to

$$\boldsymbol{f}_{\mathcal{L}} = A_{\mathcal{L}}\boldsymbol{\alpha},$$

where $(A_{\mathcal{L}})_{ik} = \mathcal{L}\Phi(\boldsymbol{x}, \boldsymbol{x}_k)\Big|_{\boldsymbol{x}=\boldsymbol{x_i}}$ and $(f_{\mathcal{L}})_i = \mathcal{L}f(\boldsymbol{x})\Big|_{\boldsymbol{x}=\boldsymbol{x}_i}$ for $i = 1, \ldots, N$. Using (2.3), we also have that $\boldsymbol{\alpha} = A^{-1}_{\Phi,X}\boldsymbol{f}$. Thus,

$$\boldsymbol{f}_{\mathcal{L}} = A_{\mathcal{L}}A^{-1}_{\Phi,X}\boldsymbol{f},$$

such that the discretized differential operator is given by

$$D_{\mathcal{L}} = A_{\mathcal{L}}A^{-1}_{\Phi,X}.$$

If the kernel Φ is conditionally positive definite we apply $\mathcal{L}$ to the interpolant (2.4) to obtain a similar discretized operator.

As an example, consider the Poisson equation

$$\begin{aligned} \Delta u(x,y) = f(x,y) &= -\frac{5}{4}\pi^2 \sin(\pi x)\cos\left(\frac{\pi y}{2}\right), \quad (x,y) \in \Omega = [0,1]^2, \\ u(x,y) = g(x,y) &= \begin{cases} \sin(\pi x), & (x,y) \in \Gamma_1, \\ 0, & (x,y) \in \Gamma_2, \end{cases} \end{aligned} \tag{2.7}$$

where $\Gamma_1 = \{(x,y) \in \Omega \mid y = 0\}$ and $\Gamma_2 = \partial\Omega \setminus \Gamma_1$. Given a set of N nodes, $X = X_I \bigcup X_B$, with $X_I = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{N_I}\}$ the interior points and $X_B = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{N_B}\}$ the boundary points. Then, we can approximate the solution of (2.7) by solving the linear system

$$\begin{bmatrix} D_\Delta \\ B \end{bmatrix} \boldsymbol{u} = \begin{bmatrix} \boldsymbol{f} \\ \boldsymbol{g} \end{bmatrix},$$

where D_Δ is the $N_I \times N$ finite dimension approximation of Δ on X_I, B the $N_B \times N$ discrete boundary operator, $\boldsymbol{f} = f\Big|_{X_I}$ and $\boldsymbol{g} = g\Big|_{X_B}$. In Figure 2.2 we display the solution of PDE (2.7) and the collocation points. For this problem a simple geometry is used, however, since our interpolant is based on radial basis function, a similar implementation can be used for complicated geometries and nonuniform node distributions.

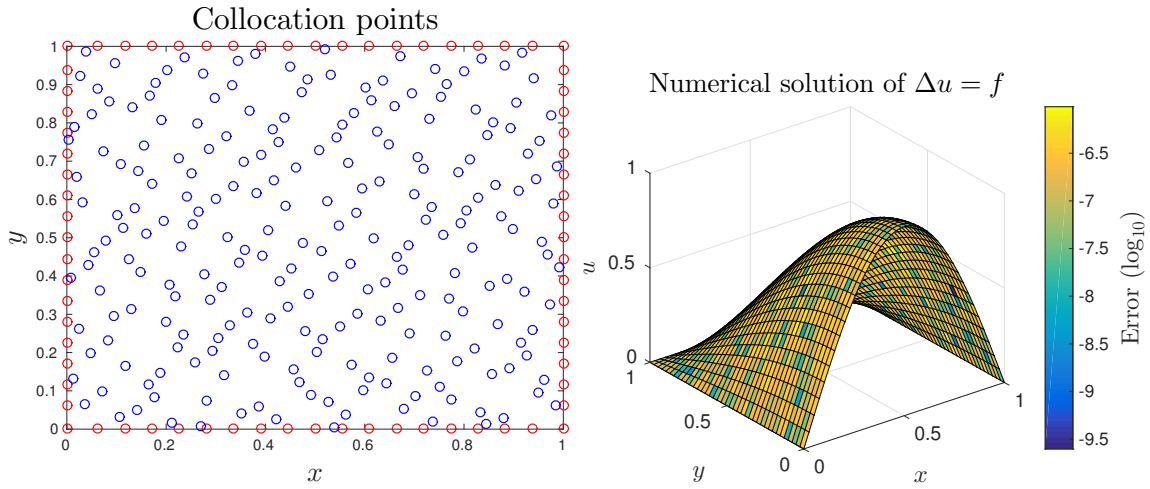

Figure 2.2: Solution of Poisson equation using a uniform distribution using 40 boundary points and 225 interior points.

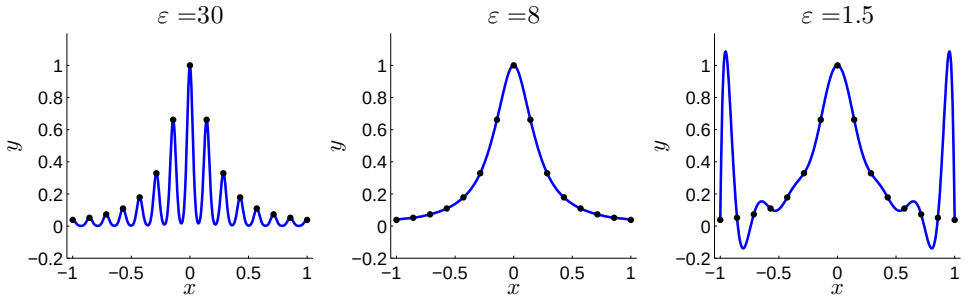

Figure 2.3: Gaussian RBF interpolant of $f(x) = 1/(1 + 25x^2)$ on 21 equally spaced points. The Runge phenomenon is clearly noticeable for $\varepsilon = 1.5$.

2.1.4 Radial Basis Function Generated Finite Differences – RBF-FD

Global RBF interpolants have a few drawbacks. Among them are the conditioning of the interpolation matrix, the presence of Runge phenomenon (see Figure 2.3) and high computational cost due to dense matrices. To avoid those issues, there are a few methods available: compactly support RBFs [52, 67, 71], partition of unity methods [68] and RBF generated finite differences (RBF-FD) [15, 17, 62, 70]. Those localized approaches allow sparse and better conditioned interpolation matrices, reducing the computational cost. In this dissertation we will focus on RBF-FD methods as the alternative to global approximations.

Similar to standard finite differences, an approximation for the differential operator

$\mathcal{L}$ at $\boldsymbol{x}_c$ is approximated by applying the operator to a RBF interpolant over a local stencil $S_{\boldsymbol{x}_c}$ that includes $\boldsymbol{x}_c$. Choosing a positive definite kernel we write

$$\mathcal{L}\Phi(\boldsymbol{x}, \boldsymbol{x}_k)\Big|_{\boldsymbol{x}=\boldsymbol{x}_c} = \sum_{\ell \in [S_{\boldsymbol{x}_c}]} w_\ell^{(c)} \Phi(\boldsymbol{x}_\ell, \boldsymbol{x}_k) \quad \text{for } k = 1, \ldots, N, \tag{2.8}$$

where $w_\ell^{(c)}$ are weights to be calculated and $[S_{\boldsymbol{x}_c}]$ is the set of indices of the nodes $\{\boldsymbol{x}_k\}_{k=1}^N$ in the local stencil $S_{\boldsymbol{x}_c}$. Hence, applying the differential operator $\mathcal{L}$ to the local interpolant $s^{(c)}$ we have

$$\begin{aligned}
\mathcal{L}s^{(c)}(\boldsymbol{x})\Big|_{\boldsymbol{x}=\boldsymbol{x}_c} &= \sum_{k \in [S_{\boldsymbol{x}_c}]} \alpha_k^{(c)} \mathcal{L}\Phi(\boldsymbol{x}, \boldsymbol{x}_k)\Big|_{\boldsymbol{x}=\boldsymbol{x}_c} \\
&= \sum_{k \in [S_{\boldsymbol{x}_c}]} \alpha_k^{(c)} \sum_{\ell \in [S_{\boldsymbol{x}_c}]} w_\ell^{(c)} \Phi(\boldsymbol{x}_\ell, \boldsymbol{x}_k) \\
&= \sum_{\ell \in [S_{\boldsymbol{x}_c}]} w_\ell^{(c)} \sum_{k \in [S_{\boldsymbol{x}_c}]} \alpha_k^{(c)} \Phi(\boldsymbol{x}_\ell, \boldsymbol{x}_k) \\
&= \sum_{\ell \in [S_{\boldsymbol{x}_c}]} w_\ell^{(c)} s^{(c)}(\boldsymbol{x}_\ell) \\
&= \sum_{\ell \in [S_{\boldsymbol{x}_c}]} w_\ell^{(c)} f_\ell.
\end{aligned}$$

That is, the approximation to the differential operator at the point $\boldsymbol{x}_c$ can be written as linear combination of the values of the function on the stencil. Therefore, the discrete operator $D_{\mathcal{L}}$ is given by

$$D_{\mathcal{L}} = \begin{bmatrix} w_1^{(1)} & \cdots & w_N^{(1)} \\ \vdots & \ddots & \vdots \\ w_1^{(N)} & \cdots & w_N^{(N)} \end{bmatrix},$$

where $w_\ell^{(c)} = 0$ for $\boldsymbol{x}_\ell \notin S_{\boldsymbol{x}_c}$, i.e., $D_{\mathcal{L}}$ will be sparse if the amount of points used in each stencil is small. To calculate the nonzero weights we must solve the linear system (2.8) for each $\boldsymbol{x}_c$, which can be precomputed and stored for a choice of data points $\{\boldsymbol{x}_k\}_{k=1}^N$ and kernel Φ.

2.2 Divergence-Free Interpolation

We now arrive to the main topic of this dissertation, the approximation of divergence-free vector fields. In the preceding sections, the methods presented can be used to approximate each component of a vector field individually. The kernels presented in this section, on the other hand, take advantage of the fact that the components to solenoidal vector fields are related by the divergence-free condition. It is reasonable to think that leveraging this constraint increases accuracy.

2.2.1 Two-Dimensional Vector Fields

In [49], a method for fitting divergence-free vector fields tangent to a two-dimensional surface was presented. Here, we use the kernels presented in [49] restricted to a planar region in $\mathbb{R}^2$.

Let $\boldsymbol{t}_1, \ldots, \boldsymbol{t}_N$ be samples of a vector field at the points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$, then, according to [49], a divergence-free interpolant is similar to (2.1),

$$\boldsymbol{t}(\boldsymbol{x}) = \sum_{k=1}^{N} \Psi(\boldsymbol{x}, \boldsymbol{x}_k)\boldsymbol{\alpha}_k, \tag{2.9}$$

with the *coefficient vectors* $\{\boldsymbol{\alpha}_k\}_{k=1}^N$ calculated to ensure data interpolation and the *matrix-valued kernel* defined by

$$\Psi(\boldsymbol{x}, \boldsymbol{y}) = F(r)(\boldsymbol{n_y}\boldsymbol{n_x}^T - \boldsymbol{n_y}^T\boldsymbol{n_x}I) - G(r)(\boldsymbol{n_x} \times (\boldsymbol{x} - \boldsymbol{y}))(\boldsymbol{n_y} \times (\boldsymbol{x} - \boldsymbol{y}))^T,$$

where

$$F(r) = \frac{1}{r}\varphi'(r), \quad G(r) = \frac{1}{r}\left(\frac{1}{r}\varphi'(r)\right)' = \frac{1}{r}F'(r),$$

$\boldsymbol{n_v}$ denotes the surface normal at the point $\boldsymbol{v}$, $r = \|\boldsymbol{x} - \boldsymbol{y}\|$ is the Euclidean distance from $\boldsymbol{x}$ to $\boldsymbol{y}$, and φ is a positive definite radial function.

If the surface is a plane perpendicular to the z-axis and the vector field has no

component in the z direction, then the kernel is simplified to

$$\Psi(\boldsymbol{x},\boldsymbol{y}) = -F(r)\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix} - G(r)\begin{bmatrix}(x_2-y_2)^2 & -(x_1-y_1)(x_2-y_2)\\ -(x_1-y_1)(x_2-y_2) & (x_1-y_1)^2\end{bmatrix}.$$

To find the coefficient vectors $\{\boldsymbol{\alpha}_k\}_{k=1}^N$, we evaluate the interpolant at the nodes $\boldsymbol{x}_j$

$$\boldsymbol{t}_j = \boldsymbol{t}(\boldsymbol{x}_j) = \sum_{k=1}^{N} \Psi(\boldsymbol{x}_j,\boldsymbol{x}_k)\boldsymbol{\alpha}_k \quad j = 1,\dots,N,$$

leading to the linear system

$$\underbrace{\begin{bmatrix}\boldsymbol{t}_1\\ \vdots\\ \boldsymbol{t}_N\end{bmatrix}}_{\boldsymbol{t}} = \underbrace{\begin{bmatrix}\Psi(\boldsymbol{x}_1,\boldsymbol{x}_1) & \cdots & \Psi(\boldsymbol{x}_1,\boldsymbol{x}_N)\\ \vdots & \ddots & \vdots\\ \Psi(\boldsymbol{x}_N,\boldsymbol{x}_1) & \cdots & \Psi(\boldsymbol{x}_N,\boldsymbol{x}_N)\end{bmatrix}}_{A_{\Psi,X}} \underbrace{\begin{bmatrix}\boldsymbol{\alpha}_1\\ \vdots\\ \boldsymbol{\alpha}_N\end{bmatrix}}_{\boldsymbol{\alpha}}, \tag{2.10}$$

where $\boldsymbol{t}$ and $\boldsymbol{\alpha}$ are stacks of $\boldsymbol{t}_j$s and $\boldsymbol{\alpha}_k$s, respectively, and the *divergence-free interpolation matrix* $A_{\Psi,X}$ is a $2N \times 2N$ matrix compose by the blocks $\{\Psi(\boldsymbol{x}_i,\boldsymbol{x}_j)\}_{i,j=1}^N$.

If Ψ_1 and Ψ_2 denote the first and second columns of $\Psi(\boldsymbol{x},\boldsymbol{x}_k)$, respectively, and $\boldsymbol{\alpha}_k = (\alpha_k^u, \alpha_k^v)$ then

$$\begin{aligned}\nabla\cdot\boldsymbol{t}(\boldsymbol{x}) &= \sum_{k=1}^{N} \nabla\cdot\Psi(\boldsymbol{x},\boldsymbol{x}_k)\boldsymbol{\alpha}_k\\ &= \sum_{k=1}^{N} [\alpha_k^u \nabla\cdot\Psi_1 + \alpha_k^v \nabla\cdot\Psi_2] = 0,\end{aligned}$$

since a direct calculation shows that $\nabla\cdot\Psi_1 = 0 = \nabla\cdot\Psi_2$.

The existence of a solution to the linear system (2.10) depends on the invertibility of the matrix $A_{\Psi,X}$. For positive definite radial functions φ, it was shown in [49] that this matrix is positive definite, and therefore invertible.

Differentiation matrices based on divergence-free kernels can be computed by repeating the steps presented in Subsection 2.1.3 and we omit the details.

2.2.2 Three-Dimensional Vector Fields

To deal with three-dimensional vector fields, we use the divergence-free kernel introduced by Narcowich and Ward in [48]. The interpolant has the same form of (2.9), except that now the vectors lie in $\mathbb{R}^3$ and the matrix-valued kernel is

$$\begin{aligned}\Psi(\boldsymbol{x},\boldsymbol{y}) &= \left(-\Delta I + \nabla\nabla^T\right)\varphi(\|\boldsymbol{x}-\boldsymbol{y}\|)\\ &= \left(-\Delta I + \nabla\nabla^T\right)e^{-\varepsilon\|\boldsymbol{x}-\boldsymbol{y}\|^2} \qquad (2.11)\\ &= \left[\left(2\varepsilon - 4\varepsilon^2\|\boldsymbol{x}-\boldsymbol{y}\|^2\right)I + 4\varepsilon^2(\boldsymbol{x}-\boldsymbol{y})(\boldsymbol{x}-\boldsymbol{y})^T\right]e^{-\varepsilon\|\boldsymbol{x}-\boldsymbol{y}\|^2}, \quad \varepsilon > 0,\end{aligned}$$

where ∇ is the gradient operator, Δ is the Laplacian and I the 3×3 identity matrix. For our numerical experiments, we use the same kernel in [48], $\varphi(r) = e^{\varepsilon r^2}$. In $\mathbb{R}^2$, this kernel is exactly the same as the one introduced in the previous section. It is not hard to show that the columns of the matrix $\Psi(\boldsymbol{x},\boldsymbol{y})$ also have zero divergence with respect to $\boldsymbol{x}$, which guarantees a divergence-free interpolant.

Using the given data, $t_j^{\bar{\ell}} = t_{\bar{\ell}}(\boldsymbol{x}_j)$, we have

$$t_j^{\bar{\ell}} = \sum_{k=1}^{N}\sum_{\ell=1}^{3}\alpha_k^{\ell}\Psi_{\bar{\ell}\ell}(\boldsymbol{x}_j,\boldsymbol{x}_k) = \sum_{\ell=1}^{3}\left(A^{\bar{\ell}\times\ell}\boldsymbol{\alpha}^{\ell}\right)_j,$$

where $A_{jk}^{\bar{\ell}\times\ell} = \Psi_{\bar{\ell}\ell}(\boldsymbol{x}_j,\boldsymbol{x}_k)$, $\boldsymbol{t}^{\bar{\ell}} = [t_1^{\bar{\ell}},\ldots,t_N^{\bar{\ell}}]^\intercal$ and $\boldsymbol{\alpha}^{\ell} = [\alpha_1^{\ell},\ldots,\alpha_N^{\ell}]^\intercal$ for $\bar{\ell},\ell = 1,2,3$ and $j,k = 1,\ldots,N$. Defining

$$A = \begin{bmatrix} A^{1\times1} & A^{1\times2} & A^{1\times3}\\ A^{2\times1} & A^{2\times2} & A^{2\times3}\\ A^{3\times1} & A^{3\times2} & A^{3\times3}\end{bmatrix}, \quad \boldsymbol{\alpha} = \begin{bmatrix}\boldsymbol{\alpha}^1\\ \boldsymbol{\alpha}^2\\ \boldsymbol{\alpha}^3\end{bmatrix} \quad \text{and} \quad \boldsymbol{t} = \begin{bmatrix}\boldsymbol{t}^1\\ \boldsymbol{t}^2\\ \boldsymbol{t}^3\end{bmatrix},$$

we can summarize the interpolation conditions in the linear system $\boldsymbol{t} = A\boldsymbol{\alpha}$, which was proved to have a unique solution in [48]. Other choices for φ in (2.11) are also possible, still maintaining the invertibility of the interpolation matrix as proven in [48]. Note that we use a different ordering than in the two dimensional case because it simplifies

the implementation. Again, we follow the same procedure used in Subsection 2.1.3 to acquire differentiation matrices.

2.2.3 Divergence-Free Polynomial Approximation

In addition to the kernels presented in the preceding section, we shall also explore approximations by polynomial vector fields. To this end, consider a polynomial stream function of degree n given by

$$\psi(x,y) = \sum_{i=0}^{n}\sum_{j=0}^{n} a_{ij}x^i y^j, \quad n \in \mathbb{N}. \tag{2.12}$$

If the components of the polynomial vector field are given by $\boldsymbol{p} = (p_u, p_v, 0) = \nabla \times (0,0,\psi) = (\psi_y, -\psi_x, 0)$, then $\nabla \cdot \boldsymbol{p} = 0$. To find the coefficients a_{ij}, we use the interpolation condition

$$\boldsymbol{p}(\boldsymbol{x}_k) = (p_u, p_v)\Big|_{(x_k,y_k)} = (u,v)\Big|_{(x_k,y_k)} = \boldsymbol{f}(\boldsymbol{x}_k), \quad k = 1,\ldots,N, \tag{2.13}$$

for each interpolation point $\boldsymbol{x}_k = (x_k, y_k)$ and the target vector field $\boldsymbol{f}$. Depending on the degree n of the stream function, the linear system for the coefficients a_{ij} may or may not have a solution. In the latter case, we use the solution in the least square sense, which is unique if the interpolation matrix is full rank. If the system does not have a solution and is rank deficient there will be multiple least square solutions. If multiple solutions or least square solutions are available, we use the basic solution [30, Chapter 5] computed by the QR decomposition with column pivoting (the algorithm used by Matlab). Notice that we can take $a_{00} = 0$ since the vector field does not depend on the constant term. We discuss the relationship between the polynomial degree n and the number of data points in Subsection 3.1.2.

Chapter 3

PROPERTIES OF DIVERGENCE-FREE APPROXIMATIONS

In this chapter we study divergence-free methods using several procedures. One objective is to explore the methods presented in the previous chapter through careful numerical experiments. Of particular interest are the convergence rates of divergence-free finite difference formulas based on polynomial and RBF expansions, Lebesgue constants and their dependence on node distributions, and the flat limit ($\varepsilon \to 0$) of such approximations.

It is important to point out that although many of our observations stem from numerical experiments, we are able to prove convergence rates of divergence-free finite difference formulas for stencils of small size, as well as provide exact finite difference weights for partial derivatives. Analytic expressions for the flat limit are also provided for certain stencils. Most results presented in these chapter are novel contributions of the author and have appeared in [47]. The convergence analysis presented in Section 3.2 was developed more recently and was not included in [47].

3.1 Numerical Results

In order to study the accuracy of our divergence free approximations, we consider the test function defined by

$$\boldsymbol{f}(x,y) = \left(\frac{\sin(k_1(x-a))\cos(k_2(y-b))}{k_1}, -\frac{\cos(k_1(x-a))\sin(k_2(y-b))}{k_2} \right), \quad (x,y) \in \mathbb{R}^2, \tag{3.1}$$

which satisfies $\nabla \cdot \boldsymbol{f} = 0$ for all $k_1, k_2 \in \mathbb{R} \setminus \{0\}$ and $a, b \in \mathbb{R}$. We fix $a = 0.1$ and $b = 0.2$ to avoid symmetries in the vector field which could lead to biased observations.

In our numerical experiments we set $k_1 = 7$ and $k_2 = 7$. We first consider global interpolants and later local approximations.

3.1.1 Global Divergence-Free Numerical Results

Figure 3.1 illustrates the vector field and its divergence-free interpolant computed using (2.9). We use a Gaussian kernel with shape parameter $\varepsilon = 2$. There are $N = 256$ data sites distributed equally on both dimensions of the unit square $[-1, 1]^2$.

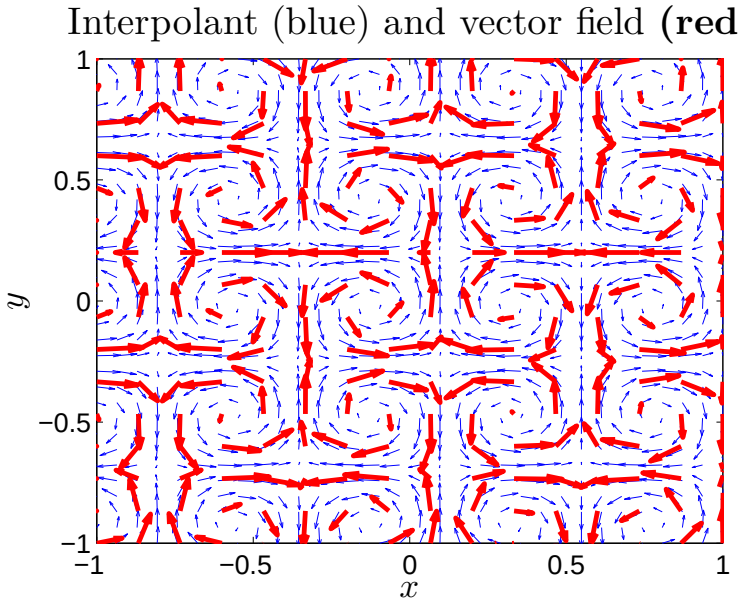

Figure 3.1: Divergence-free interpolant (thin ⟶) and the vector field at the data sites (thick ⟶) for $N = 256$.

Figure 3.2 shows how the error decays for the u-component (horizontal direction) of the vector field using the regular RBF interpolation and the divergence-free RBF method. We see that both methods achieve a precision of about 10^{-6} when $\sqrt{N} = 30$. For larger values of N, the error ceases to decrease. This behavior occurs because the interpolation matrices for both methods, although invertible, became ill-conditioned. Table 3.1 shows how fast the condition numbers grow for regular and divergence-free RBF interpolation.

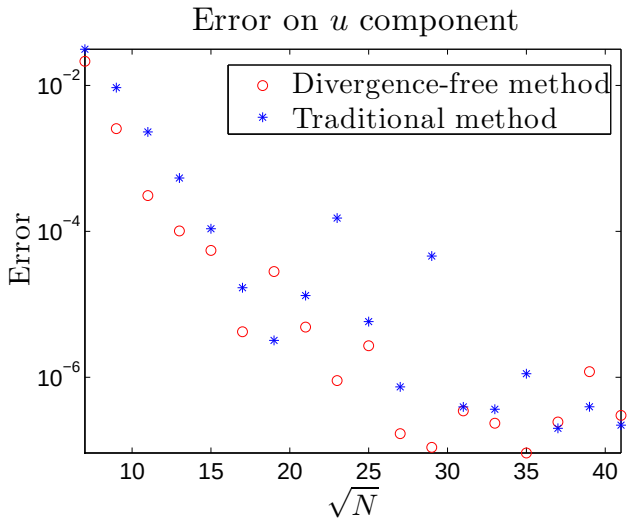

Figure 3.2: Error decay for the first component of the vector field for both methods of interpolation.

We point out that the interpolation matrix is $N \times N$ for the traditional method and $2N \times 2N$ for the divergence-free method, which makes the generation of divergence-free interpolants more computationally expensive. Moreover, Table 3.1 exhibits a larger condition number for the divergence-free method when the same shape parameter is used.

In Table 3.2, we present the error decay for the partial derivative of the u-component of the vector field as N increases. Note that we get a slightly more accurate result using the divergence-free interpolant when compared with the traditional RBF interpolant. The same feature is observed for local divergence-free interpolants in Subsection 3.1.2.

Notice that we only comment on the u-component of the field. Due to symmetry, the same observations hold for the vertical component of the vector field (v), with v_y behaving analogously to u_x and v_x to u_y.

	cond(A)	
$\sqrt{N}$	divergence-free	regular
3	1.52×10^{0}	1.11×10^{0}
5	1.12×10^{2}	1.87×10^{1}
7	2.48×10^{5}	2.82×10^{3}
9	4.36×10^{9}	2.03×10^{6}
11	2.62×10^{14}	5.06×10^{9}
13	7.21×10^{18}	3.36×10^{13}
15	3.05×10^{18}	8.69×10^{16}

Table 3.1: Condition number of the interpolation matrices of the divergence-free and regular RBF method.

3.1.2 Local Divergence-Free Numerical Results

We now focus on the accuracy of local approximations. Our first example uses a rectangular 3×3 grid. The derivatives are evaluated at the center point and compared to the exact derivatives of $\boldsymbol{f}$ defined in (3.1).

Figure 3.3 shows two experiments, one using degree $n = 2$ and the other one using degree $n = 3$ for the polynomial expansion of the stream function (2.12). In the case $n = 2$, there are 8 unknown coefficients in (2.12) (discarding the constant term) and 18 data values to fit in the 3×3 grid (9 for u and 9 for v), resulting in an 18×8 least squares system that is full rank. For $n = 3$ the size of the linear system is 18×15 and is rank deficient, with 14 linearly independent columns.

These experiments illustrate how the degree n can change the rate of convergence of the method. For $n = 3$, u_x is accurate to fourth order and u_y to second. On the other hand, both directions are second order accurate for $n = 2$. Note that the results are similar for the partial derivatives of the vertical component of the vector field.

	u_x		u_y	
$\sqrt{N}$	divergence-free	traditional	divergence-free	traditional
3	1.23×10^{-1}	1.26×10^{-1}	7.08×10^{-1}	6.17×10^{-1}
5	1.91×10^{-1}	1.63×10^{-1}	8.20×10^{-1}	7.97×10^{-1}
7	4.52×10^{-3}	7.09×10^{-3}	1.26×10^{-1}	3.46×10^{-2}
9	7.91×10^{-5}	1.89×10^{-3}	4.94×10^{-3}	9.23×10^{-3}
11	1.46×10^{-6}	4.06×10^{-4}	4.08×10^{-4}	1.98×10^{-3}
13	3.06×10^{-8}	5.13×10^{-5}	9.27×10^{-5}	2.50×10^{-4}
15	1.03×10^{-7}	4.16×10^{-6}	7.31×10^{-6}	2.03×10^{-5}

Table 3.2: Error decay for the derivatives of the first component of the vector field (u). Note that the divergence-free interpolant provides more accurate results for u_x and slightly better for u_y for most values of $\sqrt{N}$.

One possible explanation for this improvement in the convergence rates is that u_x and v_y are the derivative components that appear in the divergence operator. Writing

$$u(x, y) = u_1(x, y) + u_2(y) \quad \text{and} \quad v(x, y) = v_1(x, y) + v_2(x),$$

we have that the divergence-free condition is satisfied whenever $(u_1)_x + (v_1)_y = 0$, independently of the functions u_2 and v_2. Because our stencil is rectangular and we are using three points in each direction, the derivatives of u_2 and v_2 can only be accurate to second order, while u_1 and v_1 are accurate to fourth order in the x and y directions respectively. In Section 3.2, we show that error of the generated finite difference formulas originated from those polynomial approximations of $\boldsymbol{f}$, have indeed the order of decay as $h \to 0$ displayed in Figure 3.3.

Figure 3.4 shows the error decay for the same numerical experiment, but using the RBF-FD method. The divergence-free RBF-FD method has the same order of decay as the divergence-free FD method. One difference is that RBF-FD does not require

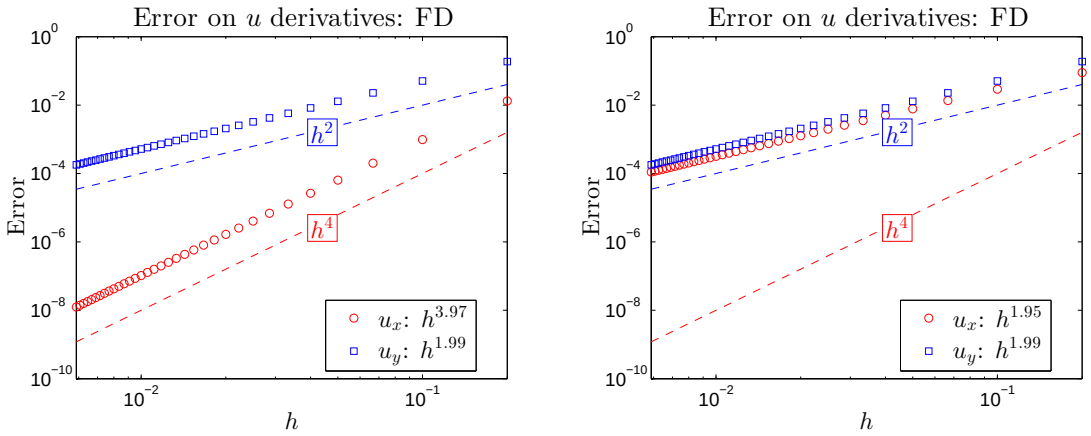

Figure 3.3: u_x and u_y denote the approximation of the partial derivatives of the first component of $\boldsymbol{f}$ using 9 point stencils. The right picture uses a polynomial stream function of degree $n = 2$ and the left picture degree $n = 3$.

the solution of a least squares problem to find a divergence-free approximation, that is, an interpolant is guaranteed to exist.

It is important to point out that for the polynomial finite differences method we need to choose an adequate degree in order to obtain faster convergence rates for the derivatives. For example, for a 3×3 stencil, $n = 3$ is the smallest degree for fourth order convergence in u_x and v_y (and larger degrees will not improve accuracy, see Figure 3.5). In the scattered node case, the choice of degree is less obvious. Figure 3.5 displays the error decay for a fixed number of points as the degree n of the stream function is increased. For the scattered nodes case, the error does not decrease for degrees larger than 5 when using a 9 point approximation. Notice that the linear system to be solved in this case is 18×35 and the coefficients of (2.12) are chosen as the basic solution computed by Matlab's backslash (`mldivide`) command (see Subsection 2.2.3).

We point out that the accuracy of the results are significantly different for scattered nodes. Our next experiment shows that neither direction is necessarily favored (in contrast to rectangular grids). We generate N random points uniformly distributed

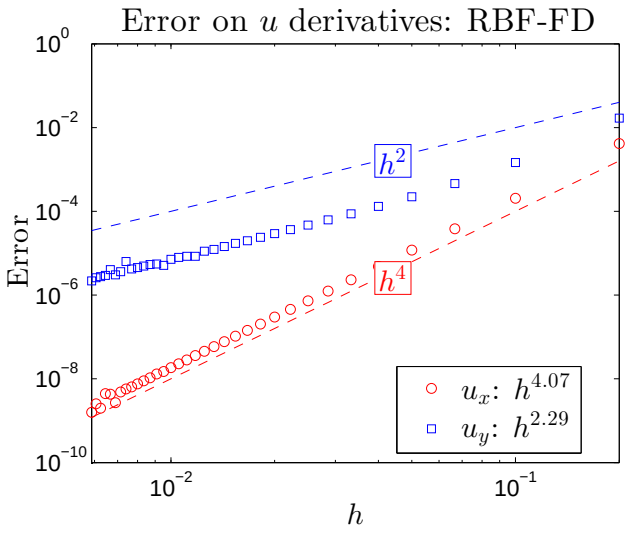

Figure 3.4: Error decay of the derivatives of u using divergence-free RBF-FD method on 9 point stencils. The dashed lines represent the decay of a 2nd and a 4th order methods.

in $[-1,1]^2$ and then select the closest 8 points to where we want to approximate the derivatives. Those 9 points are used to create our local divergence-free interpolant for the RBF and polynomial cases.

Figure 3.6 shows the error decay for the derivative of the first component of the vector field for both methods. We note the methods seem to converge with a rate between second and fourth order. The plot in Figure 3.6 shows the error as a function of the *fill distance*

$$h = \sup_{\boldsymbol{x}\in\Omega} \min_{\boldsymbol{x}_j\in X} \|\boldsymbol{x} - \boldsymbol{x}_j\|_2,$$

where $\Omega = [-1,1]^2$ and X is the set of interpolation points.

Unfortunately, the higher convergence rates for u_x and v_y are not maintained for scattered nodes. It is not completely clear from our numerical experiments, but in Section 3.2 we study the error decay rates analytically for the polynomial method using a 3×3 Cartesian stencil rotated by $\pi/4$ angle. The decay rates in this scenario are only second order.

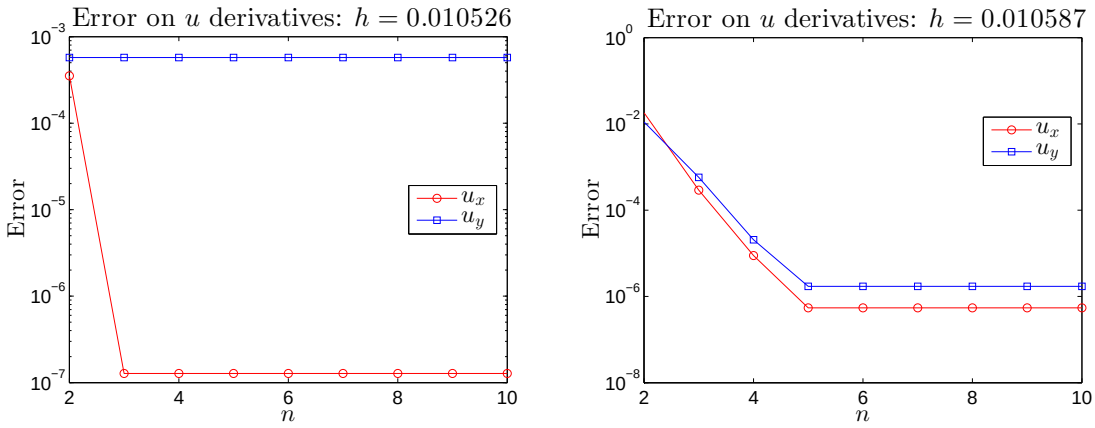

Figure 3.5: Error decay as a function of the polynomial degree n for fixed number of points. *Left:* Using a 3×3 rectangular stencil. *Right:* Using 9 scattered points. For the rectangular grid case, polynomial degrees larger than 3 do not improve the approximation, while for the unstructured case, degrees larger than 5 saturate the approximation.

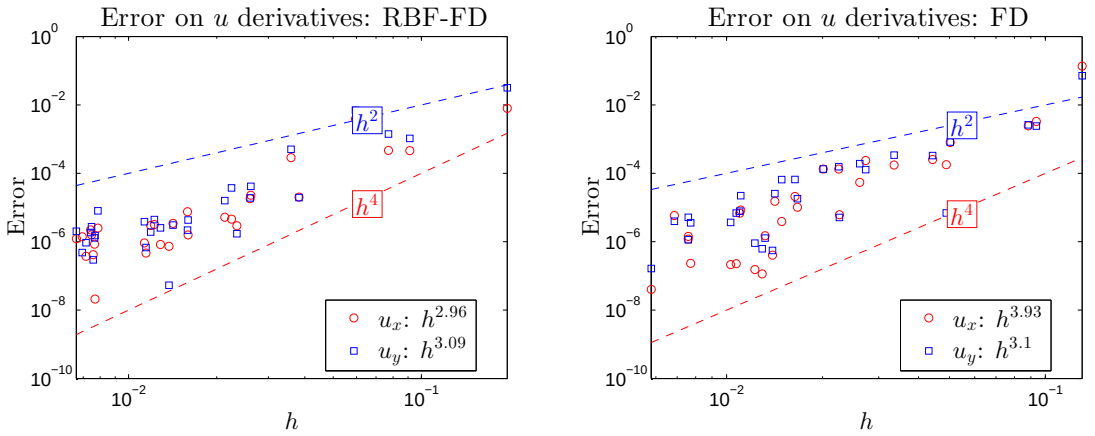

Figure 3.6: Error decay for divergence-free RBF-FD method on scattered nodes. *Left:* Divergence-free RBF-FD; *Right:* Divergence-free polynomial FD using $n = 5$ for the degree of the stream function (2.12).

3.1.3 Three Dimensional Case

To test the local accuracy of the three-dimensional kernel described in Subsection 2.2.2 we use the function $\boldsymbol{f}(x, y, z) = \nabla \times \boldsymbol{\psi}(x, y, z)$, where $\boldsymbol{\psi} = (\psi_1, \psi_2, \psi_3)$

and

$$\psi_1(x,y,z) = \sin\left((x-1)^2 + (y-1)^2 + (z-1)^2\right),$$
$$\psi_2(x,y,z) = \cos\left((x-0.5)^2 + (y-0.5)^2 + (z-0.5)^2\right),$$
$$\psi_3(x,y,z) = \sin\left((x+0.5)^2 + (y-1)^2 + (z+1)^2\right).$$

Figure 3.7 shows the error decay for evenly spaced $3 \times 3 \times 3$ Cartesian stencils using divergence-free kernels based on Gaussians. Just as in the two-dimensional case, we have an increase in the convergence rate of the derivatives present in the divergence operator (u_x, v_y and w_z), however this precision is due to the disposition of our grid points. In Figure 3.8, we repeat the experiment using a set of scattered nodes formed by the origin and the 26 nearest points. As in the 2D case, a convergence rate between second and fourth order is observed. For both experiments we use a fixed shape parameter $\varepsilon = 8$.

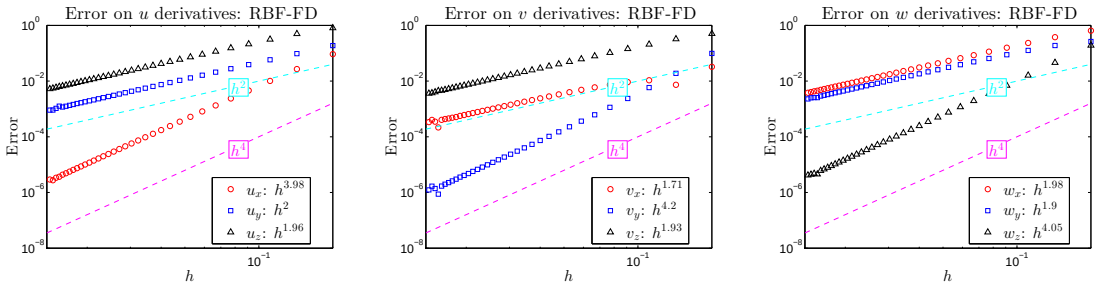

Figure 3.7: Error in the approximation of the derivatives of a divergence-free vector field using the 3D divergence-free RBF method with Gaussians and $\varepsilon = 8$.

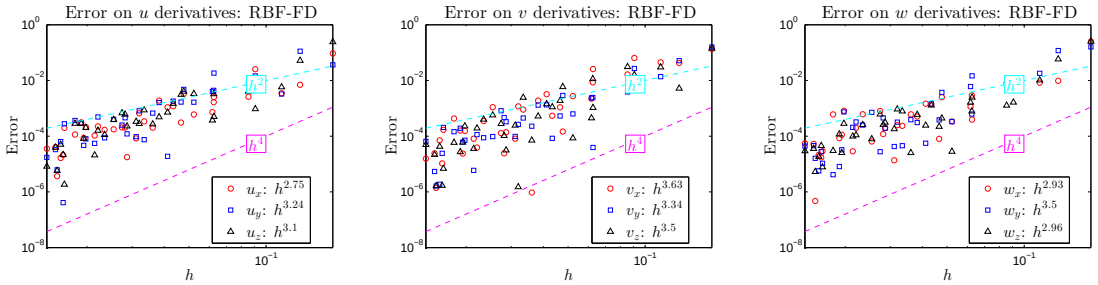

Figure 3.8: Error on the derivatives of the 3D divergence-free RBF interpolant using scattered nodes and $\varepsilon = 8$ (Gaussians).

3.2 Convergence Analysis of Local Divergence-Free Polynomials

In this section, we present analytical results that support the achieved decay rates of the finite difference schemes presented in Subsection 3.1.2. For simplicity, we focus on the two dimensional case, but the results can be extended to three dimension as well.

The idea is to compute the finite difference weights analytically and study the local truncation error in approximating the partial derivatives of the vector field. Specifically, we Taylor expand $u\Big|_{\boldsymbol{x}_k}$ and $v\Big|_{\boldsymbol{x}_k}$ around $\mathbf{0}$ (the center point), for $k = 1, \ldots, 9$, where $\boldsymbol{x}_k$ are the stencil points. Two stencils will be considered: a 3×3 squared Cartesian stencil; and its rotation by $\pi/4$ angle.

The finite difference schemes based on the polynomial stream function (2.12) has a simpler structure than the ones based on divergence-free kernels. For this reason we were able to compute the expression for the weights analytically only for the polynomial case. First we study the error for $n = 2$ and $n = 3$ to verify the decay rates shown Figure 3.3. Later, we change the polynomial basis such that a full rank system with a unique solution is acquired. This will simplify the expression of the finite difference weights and still maintain the faster order of decay for u_x and v_y.

To calculate the finite difference weights, we rewrite the polynomial stream function (2.12) as

$$\psi(x, y) = a_0 + \sum_{k=1}^{m} a_k q_k(x, y),$$

where the q_ks are the bivariate monomial terms in (2.12) and $m = (n+1)^2 - 1$. Consequentially, the components of the divergence-free polynomial are expressed as

$$p_u(x, y) = +\sum_{k=1}^{m} a_k q_{k_y}(x, y),$$
$$p_v(x, y) = -\sum_{k=1}^{m} a_k q_{k_x}(x, y),$$

and the linear system originated from the interpolation condition (2.13) is

$$\underbrace{\begin{bmatrix} q_{1_y}(x_1,y_1) & \cdots & q_{m_y}(x_1,y_1) \\ \vdots & \ddots & \vdots \\ q_{1_y}(x_N,y_N) & \cdots & q_{m_y}(x_N,y_N) \\ -q_{1_x}(x_1,y_1) & \cdots & -q_{m_x}(x_1,y_1) \\ \vdots & \ddots & \vdots \\ -q_{1_x}(x_N,y_N) & \cdots & -q_{m_x}(x_N,y_N) \end{bmatrix}}_{M} \underbrace{\begin{bmatrix} a_1 \\ \vdots \\ a_m \end{bmatrix}}_{\boldsymbol{a}} = \underbrace{\begin{bmatrix} u(x_1,y_1) \\ \vdots \\ u(x_N,y_N) \\ v(x_1,y_1) \\ \vdots \\ v(x_N,y_N) \end{bmatrix}}_{\mathfrak{f}}. \tag{3.2}$$

As mention in Subsection 2.2.3, the linear system above might have from infinitely many solutions to none. For this reason, we use the least squares solution given by the Moore-Penrose pseudoinverse, that is, we look for a divergence-free polynomial vector field that best fit the data. Hence,

$$\boldsymbol{a} = M^{\dagger}\mathfrak{f}$$

where † denotes the Moore-Penrose pseudoinverse. Defining the vectors

$$\boldsymbol{p_u} \coloneqq +\begin{bmatrix} q_{1_y} \cdots q_{m_y} \end{bmatrix} \quad \text{and} \quad \boldsymbol{p_v} \coloneqq -\begin{bmatrix} q_{1_x} \cdots q_{m_x} \end{bmatrix},$$

allows approximating the partial derivative of the vector field at the origin as

$$\begin{aligned} u_x(0,0) &\approx (\boldsymbol{p_u})_x\Big|_{(0,0)}\boldsymbol{a} = \underbrace{(\boldsymbol{p_u})_x\Big|_{(0,0)}M^{\dagger}}_{\boldsymbol{w}_{u_x}}\mathfrak{f} = \boldsymbol{w}_{u_x}\mathfrak{f}, \\ u_y(0,0) &\approx (\boldsymbol{p_u})_y\Big|_{(0,0)}\boldsymbol{a} = \underbrace{(\boldsymbol{p_u})_y\Big|_{(0,0)}M^{\dagger}}_{\boldsymbol{w}_{u_y}}\mathfrak{f} = \boldsymbol{w}_{u_y}\mathfrak{f}, \\ v_x(0,0) &\approx (\boldsymbol{p_v})_x\Big|_{(0,0)}\boldsymbol{a} = \underbrace{(\boldsymbol{p_v})_x\Big|_{(0,0)}M^{\dagger}}_{\boldsymbol{w}_{v_x}}\mathfrak{f} = \boldsymbol{w}_{v_x}\mathfrak{f}, \\ v_y(0,0) &\approx (\boldsymbol{p_v})_y\Big|_{(0,0)}\boldsymbol{a} = \underbrace{(\boldsymbol{p_v})_y\Big|_{(0,0)}M^{\dagger}}_{\boldsymbol{w}_{v_y}}\mathfrak{f} = \boldsymbol{w}_{v_y}\mathfrak{f}, \end{aligned}$$

where $\boldsymbol{w}_{u_x}$, $\boldsymbol{w}_{u_y}$, $\boldsymbol{w}_{v_x}$ and $\boldsymbol{w}_{v_y}$ are the weights to approximate the partial derivatives of the vector field.

In Table 3.3 we show the finite difference weights for the squared Cartesian 3×3 stencil using $n = 2$. Moreover, those weights allow us to calculate the truncation error of the finite difference formulas using Taylor series expansions around $\mathbf{0}$. Expanding each element of $\mathfrak{f}$ and assuming (u, v) is a solenoidal vector field leads to

$$
\begin{aligned}
(\boldsymbol{p_u})_x \boldsymbol{a} &= \frac{u_x - v_y}{2} + \left(\frac{u_{xxx} - v_{yyy}}{12} + \frac{u_{xyy} - v_{xxy}}{6} \right) h^2 + \mathcal{O}\left(h^4\right) && = u_x + \mathcal{O}\left(h^2\right), \\
(\boldsymbol{p_u})_y \boldsymbol{a} &= u_y + \left(\frac{u_{xxy} + u_{yyy} + v_{xyy}}{6} \right) h^2 + \mathcal{O}\left(h^4\right) && = u_y + \mathcal{O}\left(h^2\right), \\
(\boldsymbol{p_v})_x \boldsymbol{a} &= v_x + \left(\frac{u_{xxy} + v_{xxx} + v_{xyy}}{6} \right) h^2 + \mathcal{O}\left(h^4\right) && = v_x + \mathcal{O}\left(h^2\right), \\
(\boldsymbol{p_v})_y \boldsymbol{a} &= -\frac{u_x - v_y}{2} - \left(\frac{u_{xxx} - v_{yyy}}{12} + \frac{u_{xyy} - v_{xxy}}{6} \right) h^2 + \mathcal{O}\left(h^4\right) && = v_y + \mathcal{O}\left(h^2\right).
\end{aligned}
$$

Note that we have a second order finite difference scheme for this case. Table 3.4 have the weights for the same type of stencil, but using $n = 3$. Similarly, we obtain

$$
\begin{aligned}
(\boldsymbol{p_u})_x \boldsymbol{a} &= \frac{360\left(u_x - v_y\right) + 60\left(u_{xxx} + v_{xxy} - u_{xyy} - v_{yyy}\right) h^2 + \mathcal{O}\left(h^4\right)}{720h^8 + 1440h^4 + 720} = \\
&\qquad\qquad u_x + \mathcal{O}\left(h^4\right), \\
(\boldsymbol{p_u})_y \boldsymbol{a} &= u_y + \frac{1}{6}\left(u_{xxy} + u_{yyy} + v_{xyy}\right) h^2 + \mathcal{O}\left(h^4\right) = u_y + \mathcal{O}\left(h^2\right), \\
(\boldsymbol{p_v})_x \boldsymbol{a} &= v_x + \frac{1}{6}\left(u_{xxy} + v_{xxx} + v_{xyy}\right) h^2 + \mathcal{O}\left(h^4\right) = v_x + \mathcal{O}\left(h^2\right), \\
(\boldsymbol{p_v})_y \boldsymbol{a} &= \frac{-360\left(u_x - v_y\right) - 60\left(u_{xxx} + v_{xxy} - u_{xyy} - v_{yyy}\right) h^2 + \mathcal{O}\left(h^4\right)}{720h^8 + 1440h^4 + 720} = \\
&\qquad\qquad v_y + \mathcal{O}\left(h^4\right),
\end{aligned}
$$

Here, higher order of accuracy for u_x and v_y are achieved due to the divergence-free condition ($u_x + v_y = 0$). Those analytical results explain the observed convergence rates in Figure 3.3.

Rotating the 3×3 stencil from before by an $\pi/4$ angle and fixing the polynomial degree of the stream function to $n = 3$, leads to the weights in Table 3.5. Unfortunately,

	$\boldsymbol{w}_{u_x} \cdot 12h$		$\boldsymbol{w}_{u_y} \cdot 12h$		$\boldsymbol{w}_{v_x} \cdot 12h$		$\boldsymbol{w}_{v_y} \cdot 12h$	
(x, y)	$\boldsymbol{u}$	$\boldsymbol{v}$	$\boldsymbol{u}$	$\boldsymbol{v}$	$\boldsymbol{u}$	$\boldsymbol{v}$	$\boldsymbol{u}$	$\boldsymbol{v}$
$(-1,-1)h$	-1	1	-1	-1	-1	-1	1	-1
$(-1,0)h$	-1	0	0	2	0	-4	1	0
$(-1,1)h$	-1	-1	1	-1	1	-1	1	1
$(0,-1)h$	0	1	-4	0	2	0	0	-1
$(0,0)h$	0	0	0	0	0	0	0	0
$(0,1)h$	0	-1	4	0	-2	0	0	1
$(1,-1)h$	1	1	-1	1	-1	1	-1	-1
$(1,0)h$	1	0	0	-2	0	4	-1	0
$(1,1)h$	1	-1	1	1	1	1	-1	1

Table 3.3: Finite difference weights for partial derivatives using $n = 2$ for the polynomial stream function.

only second order rates of decay for the partial derivatives of the vector field are achieved, since the terms of order h^2 will not vanish even if the vector field is solenoidal, as it can be seen below

$$
\begin{aligned}
(\boldsymbol{p}_{\boldsymbol{u}})_x \boldsymbol{a} &= \frac{30\left(u_x - v_y\right) + 10\left(u_{xxx} - v_{yyy}\right) h^2 + \mathcal{O}\left(h^4\right)}{120h^4 + 60} \\
&= u_x + \frac{1}{6}\left(u_{xxx} - v_{yyy}\right) h^2 + \mathcal{O}\left(h^4\right), \\
(\boldsymbol{p}_{\boldsymbol{u}})_y \boldsymbol{a} &= u_y + \frac{1}{24}\left(u_{xxy} + 7u_{yyy} - v_{xxx} + v_{xyy}\right) h^2 + \mathcal{O}\left(h^4\right) \\
&= u_y + \frac{1}{24}\left(7u_{yyy} - v_{xxx}\right) h^2 + \mathcal{O}\left(h^4\right), \\
(\boldsymbol{p}_{\boldsymbol{v}})_x \boldsymbol{a} &= v_x + \frac{1}{24}\left(u_{xxy} + 7v_{xxx} - u_{yyy} + v_{xyy}\right) h^2 + \mathcal{O}\left(h^4\right) \\
&= v_x + \frac{1}{24}\left(7v_{xxx} - u_{yyy}\right) h^2 + \mathcal{O}\left(h^4\right), \\
(\boldsymbol{p}_{\boldsymbol{v}})_y \boldsymbol{a} &= -\frac{30\left(u_x - v_y\right) - 10\left(u_{xxx} - v_{yyy}\right) h^2 + \mathcal{O}\left(h^4\right)}{120h^4 + 60} \\
&= v_y - \frac{1}{6}\left(u_{xxx} - v_{yyy}\right) h^2 + \mathcal{O}\left(h^4\right).
\end{aligned}
$$

	$\boldsymbol{w}_{u_x} \cdot \beta$		$\boldsymbol{w}_{u_y} \cdot \beta$		$\boldsymbol{w}_{v_x} \cdot \beta$		$\boldsymbol{w}_{v_y} \cdot \beta$	
(x, y)	$\boldsymbol{u}$	$\boldsymbol{v}$	$\boldsymbol{u}$	$\boldsymbol{v}$	$\boldsymbol{u}$	$\boldsymbol{v}$	$\boldsymbol{u}$	$\boldsymbol{v}$
$(-1,-1)h$	1	-1	-1	-1	-1	-1	-1	1
$(-1,0)h$	$-\alpha$	0	0	2	0	-4	α	0
$(-1,1)h$	1	1	1	-1	1	-1	-1	-1
$(0,-1)h$	0	α	-4	0	2	0	0	$-\alpha$
$(0,0)h$	0	0	0	0	0	0	0	0
$(0,1)h$	0	$-\alpha$	4	0	-2	0	0	α
$(1,-1)h$	-1	-1	-1	1	-1	1	1	1
$(1,0)h$	α	0	0	-2	0	4	$-\alpha$	0
$(1,1)h$	-1	1	1	1	1	1	1	-1

Table 3.4: Finite difference weights for partial derivatives using a polynomial stream function of degree $n = 3$. $\alpha = 4(3h^2 + 2)$ and $\beta = (h^4 + 1)^2 24h$.

Hence, the results in Subsection 3.1.2 for unstructured grids will not necessarily have decay rates faster than 2, indicating that a careful selection of stencil points is needed to obtain faster convergence rates.

In the three previous cases, the linear system for the coefficients of the polynomial stream function were all rank deficient. For $n = 2$, the system had rank 8, while for $n = 3$, it had rank 14 for the Cartesian and rotated stencils. In this way, depending on the right hand side $\mathfrak{f}$, the system might have multiple or no solutions.

One alternative to get a full rank matrix is to increase the degrees of freedom of system by adding more basis terms to the stream function. This strategy might not work for all point distributions, but it works for the Cartesian and rotated stencils seen above. Once we have a full rank system, we can eliminate the redundancy of multiple solutions by removing the basis terms that only add linear dependent columns to M in (3.2). This can be easily accomplished using the reduced row echelon form of M.

	$\boldsymbol{w}_{u_x} \cdot \frac{16h^5+8h}{\sqrt{2}}$		$\boldsymbol{w}_{u_y} \cdot \frac{24h}{\sqrt{2}}$		$\boldsymbol{w}_{v_x} \cdot \frac{24h}{\sqrt{2}}$		$\boldsymbol{w}_{v_y} \cdot \frac{16h^5+8h}{\sqrt{2}}$	
(x, y)	$\boldsymbol{u}$	$\boldsymbol{v}$	$\boldsymbol{u}$	$\boldsymbol{v}$	$\boldsymbol{u}$	$\boldsymbol{v}$	$\boldsymbol{u}$	$\boldsymbol{v}$
$\sqrt{2}(-1,0)h$	-1	0	0	1	0	-5	1	0
$\frac{\sqrt{2}}{2}(-1,1)h$	0	0	1	-1	1	-1	0	0
$\sqrt{2}(0,1)h$	0	-1	5	0	-1	0	0	1
$\frac{\sqrt{2}}{2}(-1,-1)h$	0	0	-1	-1	-1	-1	0	0
$(0,0)h$	0	0	0	0	0	0	0	0
$\frac{\sqrt{2}}{2}(1,1)h$	0	0	1	1	1	1	0	0
$\sqrt{2}(0,-1)h$	0	1	-5	0	1	0	0	-1
$\frac{\sqrt{2}}{2}(1,-1)h$	0	0	-1	1	-1	1	0	0
$\sqrt{2}(1,0)h$	1	0	0	-1	0	5	-1	0

Table 3.5: Finite difference weights for the 3×3 stencil rotated by a $\pi/4$ angle using of polynomial stream function with $n = 3$.

Setting $n = 5$ for the Cartesian grid case results in a full rank system, that is, $\text{rank}\,(M) = 18$. A linear system with unique solution is found by removing the 17 redundant basis terms: y^4, y^5, x^3y^3, x^3y^4, x^3y^5, x^4, x^4y, x^4y^2, x^4y^3, x^4y^4, x^4y^5, x^5, x^5y, x^5y^2, x^5y^3, x^5y^4 and x^5y^5. Using the weights calculated by solving this system we get the following approximations to the partial derivatives

$$\begin{aligned}
(\boldsymbol{p_u})_x \boldsymbol{a} &= u_x + (u_{xxx} + v_{xxy})\,\frac{h^2}{6} + \mathcal{O}\left(h^4\right) &&= u_x + \mathcal{O}\left(h^4\right), \\
(\boldsymbol{p_u})_y \boldsymbol{a} &= u_y + u_{yyy}\frac{h^2}{6} + \mathcal{O}\left(h^4\right) &&= u_y + \mathcal{O}\left(h^2\right), \\
(\boldsymbol{p_v})_x \boldsymbol{a} &= v_x + v_{xxx}\frac{h^2}{6} + \mathcal{O}\left(h^4\right) &&= v_y + \mathcal{O}\left(h^2\right), \\
(\boldsymbol{p_v})_y \boldsymbol{a} &= -u_x - (u_{xxx} + v_{xxy})\,\frac{h^2}{6} + \mathcal{O}\left(h^4\right) &&= v_y + \mathcal{O}\left(h^4\right).
\end{aligned} \tag{3.3}$$

Note that the expression for the weights (see Table 3.6) and the local truncation error is much simpler, and we maintained the 4th order error decay for u_x and v_y.

Similarly, for the rotated grid case, using $n = 5$ the linear system will also have

	$\boldsymbol{w}_{u_x} \cdot 12h$		$\boldsymbol{w}_{u_y} \cdot 12h$		$\boldsymbol{w}_{v_x} \cdot 12h$		$\boldsymbol{w}_{v_y} \cdot 12h$	
(x, y)	$\boldsymbol{u}$	$\boldsymbol{v}$	$\boldsymbol{u}$	$\boldsymbol{v}$	$\boldsymbol{u}$	$\boldsymbol{v}$	$\boldsymbol{u}$	$\boldsymbol{v}$
$(-1,-1)h$	0	-1	0	0	0	0	0	1
$(-1,0)h$	-6	0	0	0	0	-6	6	0
$(-1,1)h$	0	1	0	0	0	0	0	-1
$(0,-1)h$	0	2	-6	0	0	0	0	-2
$(0,0)h$	0	0	0	0	0	0	0	0
$(0,1)h$	0	-2	6	0	0	0	0	2
$(1,-1)h$	0	-1	0	0	0	0	0	1
$(1,0)h$	6	0	0	0	0	6	-6	0
$(1,1)h$	0	1	0	0	0	0	0	-1

Table 3.6: Finite difference weights for partial derivatives using the stream function of polynomial degree $n = 5$ and removing the redundant terms in a 3×3 squared Cartesian stencil.

rank 18. Moreover, the linear system will have a unique solution if we remove the 17 basis terms: x^2y^4, x^2y^5, x^3y, x^3y^4, x^3y^5, x^4, x^4y, x^4y^2, x^4y^3, x^4y^4, x^4y^5, x^5, x^5y, x^5y^2, x^5y^3, x^5y^4 and x^5y^5. Thus, using the calculated weights the partial derivatives approximations are

$$
\begin{aligned}
(\boldsymbol{p_u})_x \boldsymbol{a} &= u_x + u_{xxx}\frac{h^2}{3} + \mathcal{O}\left(h^4\right) && = u_x + \mathcal{O}\left(h^2\right), \\
(\boldsymbol{p_u})_y \boldsymbol{a} &= u_y + \left(u_{xxy} - v_{xxx} + v_{xyy}\right)\frac{h^2}{3} + \mathcal{O}\left(h^4\right) && = u_y + \mathcal{O}\left(h^2\right), \\
(\boldsymbol{p_v})_x \boldsymbol{a} &= v_x + v_{xxx}\frac{h^2}{3} + \mathcal{O}\left(h^4\right) && = v_y + \mathcal{O}\left(h^2\right), \\
(\boldsymbol{p_v})_y \boldsymbol{a} &= -u_x - u_{xxx}\frac{h^2}{3} + \mathcal{O}\left(h^4\right) && = v_y + \mathcal{O}\left(h^2\right).
\end{aligned}
$$

We lose the higher order error decay rates, but the expression for the weights in Table 3.7 is simpler when comparing with the rank deficient case.

	$\boldsymbol{w}_{u_x} \cdot \frac{12h}{\sqrt{2}}$		$\boldsymbol{w}_{u_y} \cdot \frac{12h}{\sqrt{2}}$		$\boldsymbol{w}_{v_x} \cdot \frac{12h}{\sqrt{2}}$		$\boldsymbol{w}_{v_y} \cdot \frac{12h}{\sqrt{2}}$	
(x, y)	$\boldsymbol{u}$	$\boldsymbol{v}$	$\boldsymbol{u}$	$\boldsymbol{v}$	$\boldsymbol{u}$	$\boldsymbol{v}$	$\boldsymbol{u}$	$\boldsymbol{v}$
$\sqrt{2}(-1,0)h$	-3	0	0	4	0	-3	3	0
$\frac{\sqrt{2}}{2}(-1,1)h$	0	0	4	-4	0	0	0	0
$\sqrt{2}(0,1)h$	0	0	-1	0	0	0	0	0
$\frac{\sqrt{2}}{2}(-1,-1)h$	0	0	-4	-4	0	0	0	0
$(0,0)h$	0	0	0	0	0	0	0	0
$\frac{\sqrt{2}}{2}(1,1)h$	0	0	4	4	0	0	0	0
$\sqrt{2}(0,-1)h$	0	0	1	0	0	0	0	0
$\frac{\sqrt{2}}{2}(1,-1)h$	0	0	-4	4	0	0	0	0
$\sqrt{2}(1,0)h$	3	0	0	-4	0	3	-3	0

Table 3.7: Finite difference weights for partial derivatives using a stream function of polynomial degree $n = 5$ and removing the redundant basis terms in a rotated stencil.

3.3 Lebesgue Constants and the Kosloff & Tal-Ezer Map

It is well known that polynomial interpolation on equally spaced nodes suffers from *Runge phenomenon* – wild oscillations near the boundaries of the domain when certain analytic functions are interpolated. Classical RBF interpolation with smooth global kernels are also susceptible to this phenomenon [22, 51, 53]. This is illustrated in Figure 2.3, where the RBF interpolant of $f(x) = 1/(1 + 25x^2)$ is shown for decreasing values of shape parameters. As mention before, the RBF interpolant approaches the Lagrange interpolant on a given set of nodes as $\varepsilon \to 0$, therefore the Runge phenomenon is expected on equispaced points. Associated to this phenomenon is the sensitivity of the interpolation process. That is, even for functions where convergence should take place in theory, the exponential ill-conditioning of the interpolation operator causes divergence in floating point arithmetic.

In this section we explore the sensitivity of the interpolation operator for the divergence-free method to perturbations on the data. To do so, we analyze the values of the Lebesgue constants. To be precise, we defined the Lebesgue constants as

$$\Lambda_N = \sup_{\substack{\boldsymbol{f}\in L^\infty(\Omega) \\ \|\boldsymbol{f}\|_{N,\infty}\neq 0}} \left\{ \frac{\|\boldsymbol{t}\|_\infty}{\|\boldsymbol{f}\|_{N,\infty}} \right\},$$

where $\boldsymbol{t}$ is the divergence-free interpolant at N grid points, Ω is the approximation domain, $\|\boldsymbol{t}\|_\infty$ is the sup norm over Ω, and $\|\boldsymbol{f}\|_{N,\infty}$ is the sup norm of the values of $\boldsymbol{f}$ on the grid. Because $\boldsymbol{f}$ is a vector field, we defined these two norms as the maximum of the norms over each component.

We compute the Lebesgue constant numerically using an equivalent expression (for the 2D case),

$$\Lambda_N = \max_{\boldsymbol{x}\in\Omega} \left(\sum_{i=1}^{N} \|\boldsymbol{\varphi}_i^u(\boldsymbol{x})\| + \sum_{i=1}^{N} \|\boldsymbol{\varphi}_i^v(\boldsymbol{x})\| \right),$$

where the functions $\boldsymbol{\varphi}_i^{u,v}$ are the cardinal functions on the nodes $\boldsymbol{x}_j$, i.e., $\boldsymbol{\varphi}_i^{u,v}(\boldsymbol{x}_j) = [0\ 0]^\intercal$ if $i \neq j$, $\boldsymbol{\varphi}_i^u(\boldsymbol{x}_i) = [1\ 0]^\intercal$, and $\boldsymbol{\varphi}_i^v(\boldsymbol{x}_i) = [0\ 1]^\intercal$. Figure 3.9 shows two cardinal functions in a 3×3 stencil using Gaussian divergence-free kernels with shape parameter $\varepsilon = 2$.

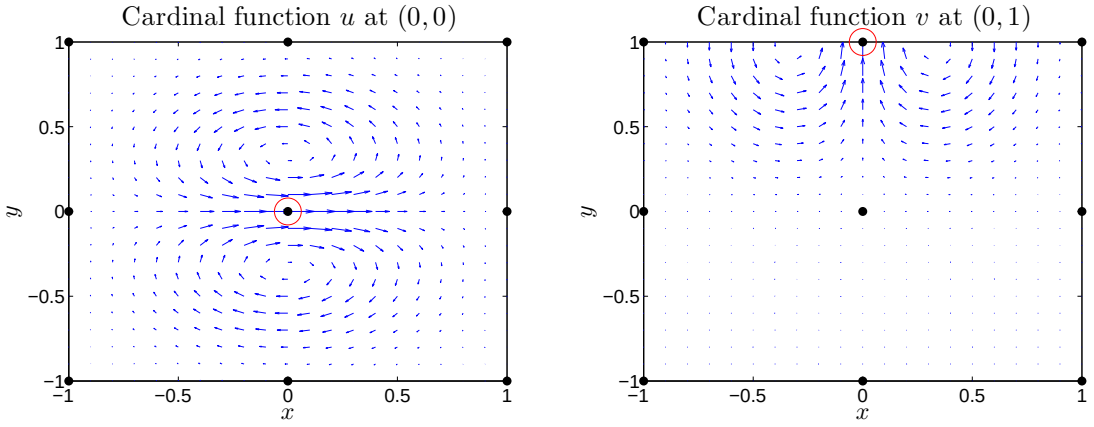

Figure 3.9: Cardinal functions for divergence-free RBF interpolation with $\varepsilon = 2$ in a 3×3 stencil. *Left:* $u = 1$ at $(0,0)$. *Right:* $v = 1$ at $(0,1)$.

Figure 3.10 displays the growth of the Lebesgue constant for different values of shape parameter of the divergence-free RBF interpolant described in Subsection 2.2.1 and the polynomial divergence-free interpolant described in Subsection 2.2.3 when using global stencils. Unsurprisingly, the polynomial method has the largest Lebesgue constant values. Polynomial approximations are notoriously ill-conditioned for interpolation on equispaced nodes [55] and similar behavior is observed in the divergence-free case.

Our experiments in Section 3.4 indicate that the flat limit ($\varepsilon \to 0$) of RBF divergence-free interpolants is also a polynomial. Consequently, we see that for small ε, divergence-free RBF approximations also have large Lebesgue constants, while larger values of ε lead to better conditioned approximations.

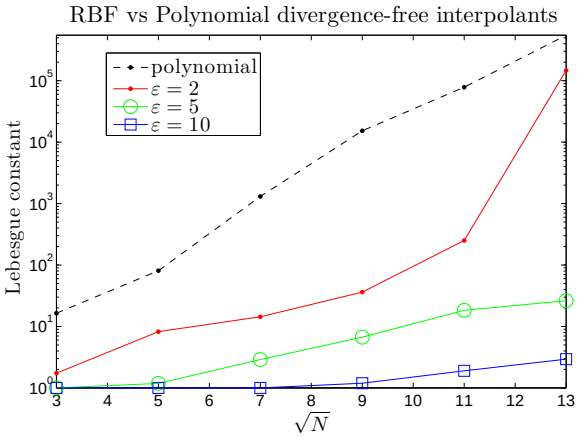

Figure 3.10: Lebesgue constant growth for divergence-free methods.

One can improve the values of the Lebesgue constant using a mapping technique. In [51], the Kosloff & Tal-Ezer mapping

$$x^{\mathrm{kte}(\alpha)} := \frac{\arcsin(\alpha x_j^{\mathrm{cheb}})}{\arcsin(\alpha)}, \quad j = 1, \ldots, N,$$

where $x_j^{\mathrm{cheb}} = \cos(\pi(j-1)/(N-1))$. Here we use the same parameter α in each spatial direction. The K–T-E mapping, introduced in [41], maps Chebyshev points into more evenly spaced ones as $\alpha \to 1$, and as $\alpha \to 0$ the map becomes the identity.

By choosing $\alpha \in (0, 1)$ one can tune the clustering of nodes near the boundaries of the domain.

In Table 3.8 we show the values of the optimized Lebesgue constants together with the values of the corresponding mapping parameters α. For the RBF case, it can be observed that mapping nodes greatly improves the Lebesgue constant. For instance, in Figure 3.11 we display the cardinal function on a 11×11 grid using the shape parameter $\varepsilon = 2$ with $u = 1$ at $(0, 0)$ (u and v are zero at the other nodes). This figure shows that the cardinal function for equally spaced nodes becomes large near the boundaries, contributing to a large value of the Lebesgue constant for the interpolation process. Using the K–T-E mapping with $\alpha = 0.967$, on the other hand, leads to smaller values near the boundaries resulting in lower values for the Lebesgue constant. For the polynomial case, using Chebyshev points for both spatial directions leads to small Lebesgue constants (around 40 in our experiments), indicating that this node distribution is also a good choice for divergence-free polynomial bases.

	$\varepsilon = 2$			$\varepsilon = 5$			$\varepsilon = 10$		
$\sqrt{N}$	$\Lambda_{\alpha=1}$	$\Lambda_{\alpha_{\min}}$	$\alpha_{\min}$	$\Lambda_{\alpha=1}$	$\Lambda_{\alpha_{\min}}$	$\alpha_{\min}$	$\Lambda_{\alpha=1}$	$\Lambda_{\alpha_{\min}}$	$\alpha_{\min}$
3	1.7	1.7	0.788	1.0	1.0	1.000	1.0	1.0	1.000
5	8.2	5.8	0.817	1.2	1.6	0.624	1.0	1.0	0.797
7	14.4	13.6	0.980	2.9	2.9	0.973	1.0	1.6	0.690
9	36.2	24.0	0.992	6.7	5.4	0.964	1.2	1.6	0.974
11	250.4	44.4	0.967	18.3	15.7	0.949	1.9	1.9	0.998
13	1.2×10^4	166.9	0.950	26.3	24.7	0.998	2.9	2.9	1.000

Table 3.8: Value of the Lebesgue constant Λ for equally spaced points ($\alpha = 1$) and optimized nodes ($\Lambda_{\alpha_{\min}}$). Lebesgue constants calculated for divergence-free RBF interpolants using $\varepsilon = 2, 5, 10$.

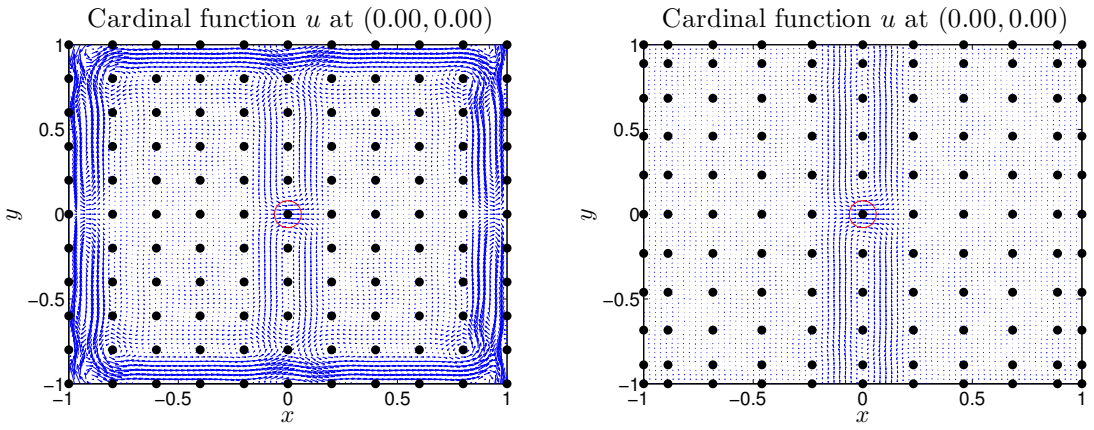

Figure 3.11: Cardinal functions for divergence-free RBF interpolation with $\varepsilon = 2$ in a 11×11 stencil for $u = 1$ at $(0, 0)$. *Left:* Using equispaced points. *Right:* using K–T-E mapping with $\alpha = 0.967$.

3.4 Limit of the Divergence-Free Interpolant as $\varepsilon \to 0$

It has been shown in [11], for the unidimensional case, and in [42, 43, 59] for the multivariate case, that the limit of increasingly flat RBF interpolants converge to polynomials. In the multivariate case, the limit is only guaranteed to exist if the node set is polynomial unisolvent, except for Gaussians (see also [21] for details). In this section we present numerical results that indicate that the flat limit of divergence-free RBF interpolants is also a polynomial. We only show results for two divergence-free kernels: Gaussians (G) and generalized inverse multiquadrics. Nevertheless, similar behavior is expected for other kernel choices.

For a few nodes, the flat limit can be found using symbolic computations. For more than a few nodes, exact computation grows too complex to provide useful information about the limit. Computations in this case were carried out in double precision. As the shape parameter decreases to zero, the condition number of the interpolation matrix becomes too large for practical computations (even with only 9 points). Fortunately, a technique presented in [20] allows the computation of the interpolant at the limit $\varepsilon \to 0$. The main idea is to consider ε (the shape parameter) a complex variable. The

RBF interpolant is an analytic function of ε near $\varepsilon = 0$. We can then compute the interpolant at the flat limit by evaluating a contour integral around the origin. The computational cost of this technique is too high for large scale problems, but works well for the problem at hand. This idea is illustrated in Figure 3.12.

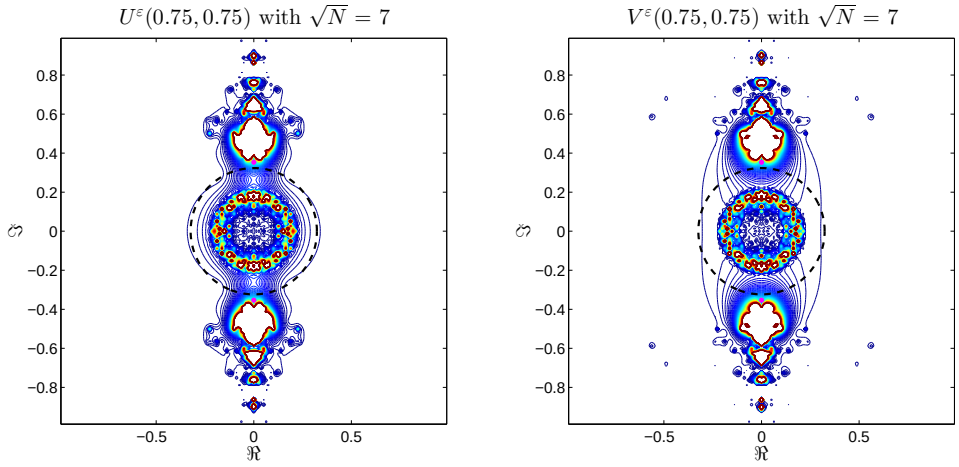

Figure 3.12: Absolute value of u (*left*) and v (*right*) at $(0.75, 0.75)$ as the shape parameter varies in the complex plane. The main idea to calculate the limit via the contour-Padé algorithm is to evaluate the interpolant for complex values of ε. The limit is calculated via a contour integral, where the contour is chosen (dashed line) to avoid the ill-conditioned region of the interpolation matrix and the poles of the interpolant.

Using the contour integral method, we evaluate the flat limit interpolant at several points. To verify whether the limit is a polynomial or not, we interpolate the values using a tensor product of Chebyshev polynomials. In all our test cases, only a few nonzero coefficients are needed for this representation. The non-zero coefficients are $\mathcal{O}(1)$ while the zero coefficients are about machine precision. We use the two-dimensional capability of *Chebfun* [12, 54, 65] to obtain the polynomial representation.

Figure 3.13 shows the divergence-free interpolant for a five point stencil with data given by $\boldsymbol{f}(0,0) = [1\ 0]^\mathsf{T}$ and $\boldsymbol{f}(1,0) = \boldsymbol{f}(0,1) = \boldsymbol{f}(-1,0) = \boldsymbol{f}(0,-1) = [0\ 0]^\mathsf{T}$. In this case we were able to find the interpolant analytically, which allowed us to compare

with the *chebfun2* approximation and see that they agree up to machine precision. The expression for the limit is:

$$\boldsymbol{p}_G(x,y) = \begin{bmatrix} -x^2 - y^2 + 1 \\ 2xy \end{bmatrix} = \boldsymbol{p}_{IM}(x).$$

Note that the limit is the same for both basic functions (G and IM), however this is not always the case.

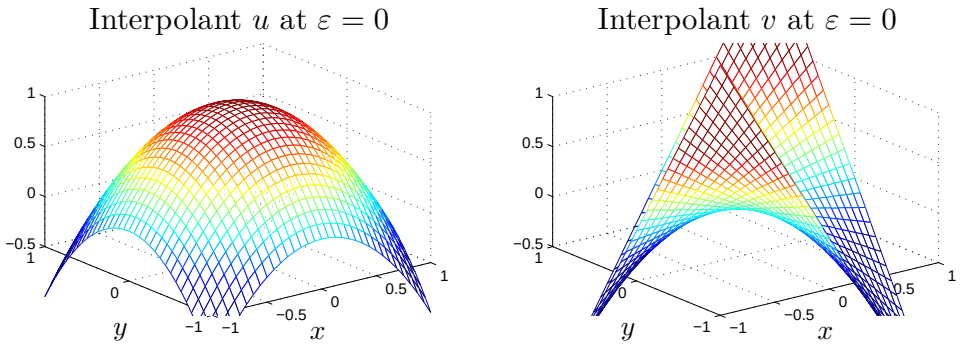

Figure 3.13: Divergence-free RBF interpolant for $\boldsymbol{f}(0,0) = [1\ 0]^\intercal$ and $\boldsymbol{f}(0,1) = \boldsymbol{f}(1,0) = \boldsymbol{f}(0,-1) = \boldsymbol{f}(-1,0) = [0\ 0]^\intercal$, using the contour-Padé algorithm. In this case, the difference between analytically and numerically computed limits is $\mathcal{O}(10^{-15})$.

For an interpolant passing through $[1\ 0]^\intercal$ at the origin and $[0\ 0]^\intercal$ for all the other points on a 3×3 rectangular stencil we obtain

$$\boldsymbol{p}^*_G(x,y) = \begin{bmatrix} \frac{2}{3}x^4 + x^2y^2 - \frac{5}{3}x^2 - y^2 + 1 \\ -\frac{8}{3}x^3y - \frac{2}{3}xy^3 + \frac{10}{3}xy \end{bmatrix},$$

$$\boldsymbol{p}^*_{IM}(x,y) = \begin{bmatrix} \frac{2}{3}x^4 + x^2y^2 - \frac{5}{3}x^2 + \frac{47}{429}y^4 - \frac{476}{429}y^2 + 1 \\ -\frac{8}{3}x^3y - \frac{2}{3}xy^3 + \frac{10}{3}xy \end{bmatrix}.$$

Here the interpolation points are $\{(i,j) \in \mathbb{R}^2 \mid i,j = -1,0,1\}$. These interpolants are displayed in Figure 3.14. For this case we were not able to acquire the interpolant analytically, however the approximation via *chebfun2* of the numerical limit given by the contour-Padé algorithm is a polynomial (we truncate the terms of the polynomial expansion that have coefficients numerically zero). Moreover, here we see that the

limits can be different for different kernels. In this case, the difference between $\boldsymbol{p}_G$ and $\boldsymbol{p}_{IM}$ is of the order 10^{-2} in the square $[-1,1]^2$ but grows larger outside this region.

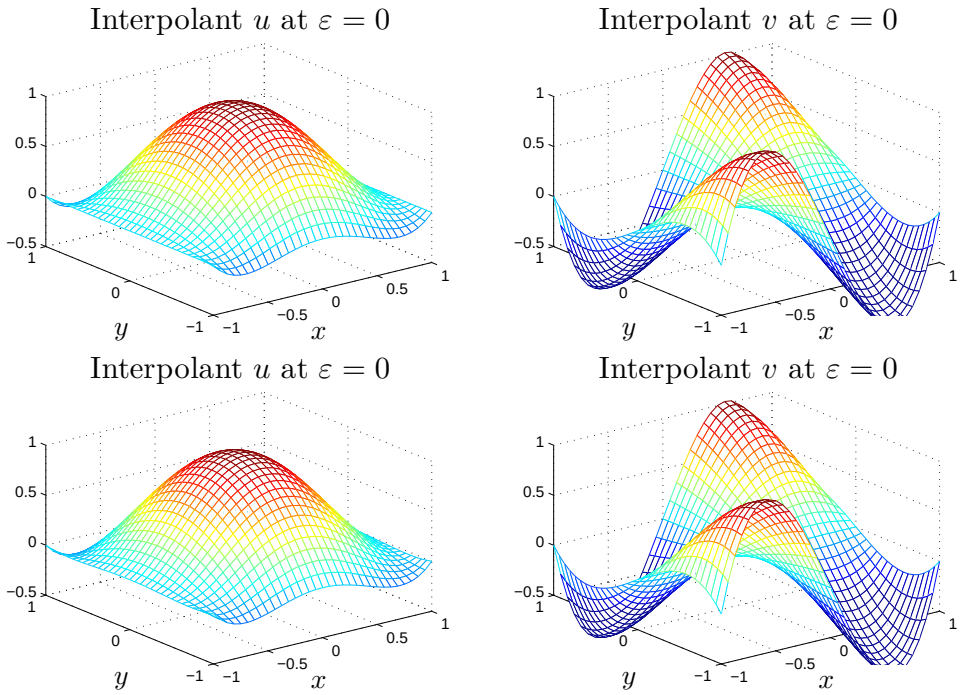

Figure 3.14: Divergence-free RBF interpolant for $\boldsymbol{f}(0,0) = [1\ 0]^\intercal$ and $\boldsymbol{f}(0,1) = \boldsymbol{f}(1,0) = \boldsymbol{f}(0,-1) = \boldsymbol{f}(-1,0) = \boldsymbol{f}(1,1) = \boldsymbol{f}(-1,1) = \boldsymbol{f}(1,-1) = \boldsymbol{f}(-1,-1) = [0\ 0]^\intercal$. *Top:* Gaussians. *Bottom:* inverse multiquadrics. The difference between the top and bottom interpolants is approximately 2.7×10^{-2} on $[-1,1]^2$.

3.5 Remarks

Divergence-free vector fields can be more accurately represented by solenoidal bases than with independent componentwise approximations. On a rectangular grid, the main gain in accuracy is in the directions of the derivatives present in the divergence operator. Two additional orders of accuracy were observed using RBF kernels in two and three dimensions, as well as in polynomial based approximations in 2D Cartesian stencils. When random or scattered nodes are used, the gain in accuracy is more evenly distributed in all directions. In this case, using a 9-point approximation in 2D resulted in an effective convergence rate between $\mathcal{O}(h^2)$ and $\mathcal{O}(h^4)$.

We point out that the computational cost for computing a divergence-free interpolant is $\mathcal{O}((dN)^3)$, in the global case and using a direct solver, where d is the vector dimension and N the number of nodes. Using traditional approximations on each vector component, on the other hand, requires $\mathcal{O}(N^3)$, since factorizations need only to be done once and can be reused for the approximation of each entry of the vector. Therefore, for 2-dimensional problems the cost would be increase by a factor of 8 and for 3-dimensional problems by a factor of 27 which is rather expensive and seems to offset gains in accuracy.

The computational cost for solving time dependent PDEs with explicit time stepping, on the other hand, is dictated by matrix-vector multiplications (if the problem is formulated using differentiation matrices, for instance). At each time step, the cost of calculating all the derivatives of a 2D (3D) vector field is $4 \times \mathcal{O}(N^2)$ $(9 \times \mathcal{O}(N^2))$ for the traditional method and $2 \times \mathcal{O}((2N)^2)$ $(3 \times \mathcal{O}((3N)^2))$ for the divergence-free approach. This leads to an increase in computational cost by a factor of 2 (3) due to the larger differentiation matrices of the divergence-free method. Thus, looking back at Table 3.2, the error between the two approaches seems comparable for a fixed flops budget, however the divergence-free approach guarantees a numerically zero divergence, which is a desirable feature in certain applications.

As for regular RBF approximations, our results indicate that the flat limit ($\varepsilon \to 0$) of divergence-free kernels is also a polynomial (in each component). For small problems (less than ten points), the limit can be obtained analytically using symbolic computations depending on the kernel and node distribution. For the more general cases, and larger number of points, the limit can be computed using the contour integral technique introduced in [21]. We have not been able to draw a connection between the limiting polynomial and the polynomial approximation computed using the method presented in Subsection 2.2.3. In some instances, the same limit was

obtained with Gaussians and inverse multiquadrics, but like in the standard RBF case [42], the limit can be different polynomials for these two kernels depending on the node distribution.

Our results also show that the condition number of the approximation procedure can be very large on an equispaced Cartesian grid. This is true for both polynomial and smooth divergence-free RBF kernels, although Lebesgue constants do not grow as fast in the latter case if the shape parameter is not close to zero. Clustering nodes more densely near boundaries was shown to lead to significantly smaller Lebesgue constants. The clustering was performed by optimizing the Kosloff–Tal-Ezer mapping (one parameter optimization). Unfortunately, even if Lebesgue constants are of moderate size, condition number of interpolation matrices can still grow exponentially for smooth kernels, a problem that can be addressed by a change of basis using a procedure similar to the RBF-QR method presented in [18, 19].

Chapter 4

APPLICATION TO FLUID PROBLEMS

In this chapter, we use localized interpolants based on traditional radial basis function and divergence-free kernels introduced in Chapter 2 to simulate incompressible fluid flows. As proof of concept, we simulate flows inside a square cavity where spectral methods generate a reference solution for our simulations. Two types of problems are considered: one driven by the motion of the cavity's lid (the lid driven cavity flow) [29], and the other induced by a heat transfer at the bottom boundary (the buoyancy driven flow) [58].

The flows in both problems are assumed to be incompressible. In this case it is well known that after time stepping the solution with a time integrator, we must ensure that the velocity field is divergence-free to avoid instabilities and maintain accuracy. Two well known ways to numerically stablish incompressibility are:

- the use of vorticity and stream function formulation and
- the use of projection schemes that recover the divergence-free part of the velocity field at each time step.

In two dimensions, the vorticity-stream function formulation is an easy way to assure incompressibility of the velocity field. In higher dimensions, however, it has an elevated computational cost besides the difficulty to implement vorticity boundary conditions. Projection methods, introduced by Chorin [6], allow us to use the more efficient primitive variable formulation in three dimensional problems. Our goal in this chapter is to combine these projection methods with the RBF finite difference methods studied in the preceding chapters. For simplicity, we restrict our studies to

two dimensions.

To measure the accuracy of numerical computations, one could use the method of manufactured solutions in which a solution is assumed and an external force function is added to the PDE to guarantee the model conforms to the desired solution. Unfortunately, except in rare cases, manufactured solutions do not resemble flow dynamics of practical interest. Thus, when testing our algorithms we solve the original fluid flow problem (without artificial forcing terms) and use spectral methods to find a reference solution.

Chapter 4 is organized as follows. First we describe the fluid flow equations in primitive variables and in vorticity-stream function formulation. Second, we present the spectral method used to compute our target solutions. Third, finite difference schemes based on traditional and divergence-free RBFs are described for each flow problem. Moreover, we explain the projection method used to guarantee incompressibility in the primitive variables formulation. Last, we compare the numerical results obtained by both schemes.

4.1 Fluid Flow Equations

We are interested in solving viscous incompressible homogeneous fluid flows, which are modeled by the incompressible Navier-Stokes equations

$$\begin{cases} \boldsymbol{u}_t + (\boldsymbol{u}\cdot\nabla)\boldsymbol{u} = -\nabla p + \nu\Delta\boldsymbol{u} + \boldsymbol{f} \\ \nabla\cdot\boldsymbol{u} = 0 \end{cases} \quad \text{in} \quad \Omega, \tag{4.1}$$

subject to the no-slip boundary conditions

$$\boldsymbol{u} = \boldsymbol{g} \quad \text{on} \quad \partial\Omega.$$

Here, $\boldsymbol{u}$, p, $\boldsymbol{f}$ and ν are the velocity, the kinematic pressure, the mass density of body forces, and kinematic viscosity, respectively.

The system (4.1) presents the velocity-pressure formulation of the incompressible Navier-Stokes equations. In what follows, we will also make use of the vorticity-stream function formulation. The vorticity is defined as $\boldsymbol{\omega} = \nabla \times \boldsymbol{u}$. Taking the curl of the momentum equation leads to

$$\nabla \times \boldsymbol{u}_t + \nabla \times [(\boldsymbol{u} \cdot \nabla)\boldsymbol{u}] = -\nabla \times (\nabla p) + \nu \nabla \times (\Delta \boldsymbol{u}) + \nabla \times \boldsymbol{f}. \tag{4.2}$$

To simplify this expression, we will use the following identities,

$$\begin{aligned}(\boldsymbol{u} \cdot \nabla) &= \frac{1}{2}\nabla(\boldsymbol{u} \cdot \boldsymbol{u}) - \boldsymbol{u} \times \underbrace{(\nabla \times \boldsymbol{u})}_{=\boldsymbol{\omega}} \\ &= \frac{1}{2}\nabla(\boldsymbol{u} \cdot \boldsymbol{u}) - \boldsymbol{u} \times \boldsymbol{\omega},\end{aligned}$$

and

$$\begin{aligned}\nabla \times (\boldsymbol{u} \times \boldsymbol{\omega}) &= \underbrace{(\nabla \cdot \boldsymbol{\omega})}_{=0}\boldsymbol{u} - \underbrace{(\nabla \cdot \boldsymbol{u})}_{=0}\boldsymbol{\omega} + (\omega \cdot \nabla)\boldsymbol{u} - (\boldsymbol{u} \cdot \nabla)\boldsymbol{\omega} \\ &= (\boldsymbol{\omega} \cdot \nabla)\boldsymbol{u} - (\boldsymbol{u} \cdot \nabla)\boldsymbol{\omega},\end{aligned}$$

We can now rewrite (4.2) as

$$\begin{aligned}(\nabla \times \boldsymbol{u})_t + \nabla \times \left[\frac{1}{2}\nabla(\boldsymbol{u} \cdot \boldsymbol{u}) - \boldsymbol{u} \times \boldsymbol{\omega}\right] &= \nu\Delta(\nabla \times \boldsymbol{u}) + \nabla \times \boldsymbol{f} \\ \boldsymbol{\omega}_t + \frac{1}{2}\underbrace{\nabla \times \nabla(\boldsymbol{u} \cdot \boldsymbol{u})}_{=0} - \nabla \times (\boldsymbol{u} \times \boldsymbol{\omega}) &= \nu\Delta\boldsymbol{\omega} + \nabla \times \boldsymbol{f} \\ \boldsymbol{\omega}_t - (\boldsymbol{\omega} \cdot \nabla)\boldsymbol{u} + (\boldsymbol{u} \cdot \nabla) &= \nu\Delta\boldsymbol{u} + \nabla \times \boldsymbol{f}.\end{aligned}$$

Furthermore, because $\nabla \cdot \boldsymbol{u} = 0$, there exist a $\boldsymbol{\psi}$ such that $\boldsymbol{u} = \nabla \times \boldsymbol{\psi}$. It is also possible to take $\boldsymbol{\psi}$ such that $\nabla \cdot \boldsymbol{\psi} = 0$. To demonstrate this, consider the Helmholtz decomposition of a general vector field $\boldsymbol{\psi}$

$$\boldsymbol{\psi} = \nabla \times \boldsymbol{A} + \nabla b,$$

Hence,

$$\boldsymbol{u} = \nabla \times \boldsymbol{\psi} = \nabla \times (\nabla \times \boldsymbol{A} + \nabla b) = \nabla \times (\nabla \times \boldsymbol{A}),$$

that is, the vector field generated by the curl of $\boldsymbol{\psi}$ only depends on the divergence-free part of $\boldsymbol{\psi}$. Finally, we point out that $\boldsymbol{\psi}$ must satisfy the following Poisson equation,

$$\boldsymbol{\omega} = \nabla \times \boldsymbol{u} = \nabla \times (\nabla \times \boldsymbol{\psi}) = \nabla \underbrace{(\nabla \cdot \boldsymbol{\psi})}_{=0} - \Delta \boldsymbol{\psi} = -\Delta \boldsymbol{\psi}.$$

Summarizing, the vorticity-stream function formulation of the incompressible Navier-Stokes equation is expressed as

$$\begin{cases} \boldsymbol{\omega}_t - (\boldsymbol{\omega} \cdot \nabla)\boldsymbol{u} + (\boldsymbol{u} \cdot \nabla)\boldsymbol{\omega} = \nu \Delta \boldsymbol{\omega} + \nabla \times \boldsymbol{f} \\ \Delta \boldsymbol{\psi} = -\boldsymbol{\omega} \end{cases} \quad \text{in} \quad \Omega, \tag{4.3}$$

with appropriate boundary conditions.

In the two dimensional case, the velocity and the mass density of body forces will only vary in the x and y directions. Hence, the stream function must be given by $\boldsymbol{\psi} = (0, 0, \psi)$ and

$$\begin{aligned} \boldsymbol{\omega} = \nabla \times \boldsymbol{u} &= (-v_z, u_z, v_x - u_y) = (0, 0, \omega) \\ \nabla \times \boldsymbol{f} &= (-f_z^v, f_z^u, \underbrace{f_x^v - f_y^u}_{f_\omega}) = (0, 0, f_\omega). \end{aligned}$$

Additionally, the terms in (4.3) simplify to

$$\begin{aligned} (\boldsymbol{\omega} \cdot \nabla)\boldsymbol{u} &= \omega \frac{\partial \boldsymbol{u}}{\partial z} = \boldsymbol{0}, \\ (\boldsymbol{u} \cdot \nabla)\boldsymbol{\omega} &= \left(u \frac{\partial}{\partial x} + v \frac{\partial}{\partial y} \right) \boldsymbol{\omega} (0, 0, u\omega_x + v\omega_y), \\ \Delta \boldsymbol{\omega} &= (0, 0, \Delta \omega), \\ \Delta \boldsymbol{\psi} &= (0, 0, \Delta \psi). \end{aligned}$$

Then, (4.3) is reduced to 2 scalar equations in ω and ψ

$$\begin{cases} \omega_t + u\omega_x + v\omega_y = \nu\Delta\omega + f_\omega \\ \Delta\psi = -\omega \end{cases} \quad \text{in} \quad \Omega$$
$$\begin{cases} w = -\Delta\psi\Big|_{\partial\Omega} \\ \psi_y = g^u, \quad \psi_x = -g^v \end{cases} \quad \text{at} \quad \partial\Omega. \tag{4.4}$$

This formulation is more efficient than the velocity-pressure formulation that has 3 unknowns (u,v,p) in the 2D case. However, in three dimensions, the $\boldsymbol{u}$-p formulation is more efficient because there are only 4 unknowns to be calculated (u,v,w,p) versus 6 unknowns in (4.3) (3 components for the vorticity and stream function).

4.1.1 The Lid Driven Cavity Flow

The lid driven cavity flow has long been used as a benchmark for fluid flow algorithms [29]. Despite its simple geometry, complex flow dynamics can be observed, including chaotic solutions for large Reynolds numbers. One of the difficulties in the numerical simulation of the conventional lid driven cavity flows is the singularity at the top corners of the domain, where the horizontal component of the velocity is discontinuous. We shall address this problem later by regularizing the velocity of the lid. At this moment, we describe the problem with a constant velocity at the top of the cavity.

Let $\boldsymbol{u} = (u, v)$ and p denote the dimensionless velocity and pressure, respectively. Then, the flow inside a cavity (Figure 4.1) is given by the incompressible Navier-Stokes

equation

$$\begin{cases} \boldsymbol{u}_t + (\boldsymbol{u}\cdot\nabla)\boldsymbol{u} = -\nabla p + \frac{1}{\mathrm{Re}}\Delta\boldsymbol{u}, \\ \nabla\cdot\boldsymbol{u} = 0, \end{cases} \qquad \text{in} \quad \Omega = [0,1]^2,$$

$$\begin{cases} \boldsymbol{u} = (1,0), & \text{on} \quad \Gamma = [0,1]\times\{1\}, \\ \boldsymbol{u} = \boldsymbol{0}, & \text{on} \quad \partial\Omega\setminus\Gamma, \end{cases}$$

in the primitive variables formulation. Here $\mathrm{Re} = LU_{\mathrm{ref}}/\nu$ is the Reynolds number, where L is a characteristic width (the length of one of the sides of the cavity), U_{ref} is a reference speed and ν is the kinematic viscosity.

The equivalent vorticity-stream function formulation in this case is

$$\begin{cases} \omega_t + u\omega_x + v\omega_y = \frac{1}{\mathrm{Re}}\Delta\omega, \\ \Delta\psi = -\omega, \end{cases} \qquad \text{in} \quad \Omega = [0,1]^2,$$

$$\begin{cases} \omega = -\Delta\psi\Big|_{\partial\Omega}, \\ \psi = \text{constant}, \end{cases} \qquad \text{on} \quad \partial\Omega,$$

where $u = \psi_y$ and $v = -\psi_x$.

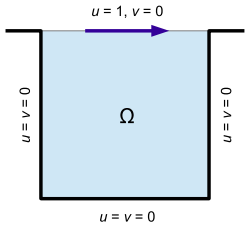

Figure 4.1: Lid driven cavity problem using nondimensional variables.

4.1.2 Buoyancy Driven Flow

The second fluid flow problem of our study is the natural convection in a cavity. The motion is modeled by the incompressible Navier-Stokes equation with a forcing

term that depends on the temperature and the velocity field. In primitive variables formulation

$$\begin{cases} \boldsymbol{u}_t + (\boldsymbol{u}\cdot\nabla)\boldsymbol{u} = -\nabla p + \mathrm{Pr}\Delta\boldsymbol{u} + \underbrace{\mathrm{PrRa}\begin{bmatrix}0\\ T\end{bmatrix}}_{\boldsymbol{f}}, \\ \nabla\cdot\boldsymbol{u} = 0, \\ T_t + (\boldsymbol{u}\cdot\nabla)T = \Delta T, \end{cases} \quad \text{in} \quad \Omega = [-1,1]^2, \tag{4.5}$$

subject to the no-slip boundary conditions $\boldsymbol{u} = \boldsymbol{0}$ and

$$T(t,x,y) = \begin{cases} \tanh^4(100t)e^{-30x^2}, & \text{on} \quad \Gamma = [-1,1]\times\{0\} \\ 0, & \text{on} \quad \partial\Omega\setminus\Gamma. \end{cases} \tag{4.6}$$

Here, $\boldsymbol{u} = (u,v)$, p and T are the dimension less velocity, pressure and the temperature. The Prandtl number Pr measures the ratio of the kinematic viscosity and the thermal diffusivity, while the Rayleigh number Ra measures how much the buoyancy forces contribute to the fluid motion in respect to the viscosity forces.

To obtain the ω-ψ formulation of (4.5), we rewrite the momentum equation of (4.4) as

$$\begin{cases} \omega_t + u\omega_x + v\omega_y = \mathrm{Pr}\Delta\omega + \underbrace{\nabla\times\boldsymbol{f}}_{\mathrm{PrRa}T_x}, \\ \Delta\psi = -\omega, \\ T_t + (\boldsymbol{u}\cdot\nabla)T = \Delta T, \end{cases} \quad \text{in} \quad \partial\Omega,$$

with $u = \psi_y$ and $v = -\psi_x$ and boundary conditions

$$\begin{cases} \omega = -\Delta\psi\Big|_{\partial\Omega} \\ \psi_y = 0 = -\psi_x \implies \psi = \text{constant} \end{cases} \quad \text{on} \quad \partial\Omega.$$

The temperature on the boundary is still described by (4.6).

4.2 The Spectral Method Discretization

The spectral method described in this section will be used to compare the accuracy of the numerical solution of the methods based on RBF generated finite differences. We specifically use the vorticity-stream function formulation to guarantee incompressibility of the flow of our reference solutions. The spatial discretization is obtained using Chebyshev collocation as described in [66].

We point out that although the buoyancy driven flow, as stated in (4.5) with temperature described by (4.6) and zero velocity on the boundaries, does not present discontinuities, large gradients will develop in the solution of flows with high Rayleigh numbers requiring very fine discretizations to maintain accuracy. For this reason in our numerical experiments we consider low to moderate Rayleigh numbers.

Similarly, the lid driven cavity flow will contain high gradients for large Reynolds numbers, which can be addressed with enough discretization points. Nevertheless, the problem (as proposed in Subsection 4.1.1) presents a discontinuous boundary condition, which affects the accuracy of the whole numerical solution due to the global character of spectral methods. To circumvent this issue, we consider a regularized version of the lid driven cavity flow problem [61]. More specifically, we use the boundary condition $u_s(t,x) = 16x^2(x-1)^2\tanh^2(100t)$ for the velocity at the top wall.

In the buoyancy driven flow we use a 32×32 tensor product grid of Chebyshev points, while for the lid driven cavity low a 64×64 grid. Moreover, we use Euler's method to time step the temporal part. The spectral solution together with the finite difference schemes based on RBFs will be displayed in Section 4.4. The error is calculated by evaluating the spectral solution on the same nodes used by the traditional and divergence-free schemes. This interpolation can be done easily using the *chebfun2* [65], ensuring that the error in the interpolation process is only due to rounding errors

(which are negligible in this case).

4.3 Finite Differences Discretizations

The goal of this section is to solve the fluid flow problems stated in Subsection 4.1.1 and Subsection 4.1.2 using the velocity-pressure formulation and approximating the partial derivatives with finite differences schemes based on traditional and divergence-free RBFs. First, we present the projection method used to ensure the incompressibility of the velocity field. Next we describe the spatial RBF discretization and later compare the solutions with the ones computed with the spectral method.

4.3.1 Projection Method and Time Stepping Scheme

When using primitive variables, it is important to ensure that $\nabla \cdot \boldsymbol{u} = 0$ at each time step. Failure to do so often results in unstable simulations. An efficient way to impose this condition was proposed by [6, 64] – the so called projection methods. This method is based on an evolution equation for the velocity field

$$\boldsymbol{u}_t = \mathcal{P}(-(\boldsymbol{u} \cdot \nabla)\boldsymbol{u} + \nu \Delta \boldsymbol{u}). \tag{4.7}$$

Here, $\mathcal{P}$ denotes the *Leray projector*. That is, given a vector field $\boldsymbol{w} \in \Omega$, $\mathcal{P}\boldsymbol{w}$ is divergence-free and tangent to $\partial\Omega$. The existence of such operator comes from the Helmholtz-Hodge decomposition [2, 7, 16, 57].

Theorem 5 (Helmholtz-Hodge decomposition)**.** *A vector field $\boldsymbol{w} \in \Omega$ can be uniquely decomposed in the form*

$$\boldsymbol{w} = \boldsymbol{u} + \nabla q,$$

where $\nabla \cdot \boldsymbol{u} = 0$ and $\boldsymbol{u} \cdot \boldsymbol{n} = 0$ on $\partial\Omega$.

Proof. Using the identity $\nabla \cdot (q\boldsymbol{u}) = (\nabla \cdot \boldsymbol{u})q + \boldsymbol{u} \cdot \nabla q$,

$$\int_\Omega \boldsymbol{u} \cdot \nabla q \mathrm{d}V = \int_\Omega \nabla \cdot (q\boldsymbol{u}) - \underbrace{(\nabla \cdot \boldsymbol{u})}_{=0} q \mathrm{d}V = \int_{\partial\Omega} q \underbrace{\boldsymbol{u} \cdot \boldsymbol{n}}_{=0} \mathrm{d}S = 0 \tag{4.8}$$

[*Uniqueness*]: Let $\boldsymbol{w} = \boldsymbol{u}_1 + \nabla q_1 = \boldsymbol{u}_2 + \nabla q_2$, then

$$0 = \int_\Omega \|\boldsymbol{u}_1 - \boldsymbol{u}_2 + \nabla(q_1 - q_2)\|^2 \mathrm{d}V \overset{(4.8)}{=} \int_\Omega \|\boldsymbol{u}_1 - \boldsymbol{u}_2\|^2 \mathrm{d}V + \int_\Omega \|\nabla(q_1 - q_2)\|^2 \mathrm{d}V,$$

which implies $\boldsymbol{u}_1 = \boldsymbol{u}_2$ and $\nabla q_1 = \nabla q_2$.

[*Existence*]: Let $\boldsymbol{w} = \boldsymbol{u} + \nabla q$. Then, the problem

$$\begin{cases} \Delta q = \nabla \cdot \boldsymbol{w} & \text{in } \Omega, \\ \boldsymbol{n} \cdot \nabla q = \boldsymbol{n} \cdot \boldsymbol{w} & \text{on } \partial\Omega, \end{cases} \tag{4.9}$$

has a unique solution up to a constant, since the compatibility condition

$$\int_\Omega \nabla \cdot \boldsymbol{w} \mathrm{d}V = \int_{\partial\Omega} \boldsymbol{w} \cdot \boldsymbol{n} \mathrm{d}S$$

is satisfied by the divergence theorem [8]. Defining $\boldsymbol{u} = \boldsymbol{w} - \nabla q$ we note that

$$\nabla \cdot \boldsymbol{u} = \nabla \cdot \boldsymbol{w} - \nabla \cdot \nabla q \overset{(4.9)}{=} \nabla \cdot \boldsymbol{w} - \nabla \cdot \boldsymbol{w} = 0 \quad \text{in } \Omega,$$

and

$$\boldsymbol{u} \cdot \boldsymbol{n} = \boldsymbol{w} \cdot \boldsymbol{n} - \nabla q \cdot \boldsymbol{n} \overset{(4.9)}{=} \boldsymbol{w} \cdot \boldsymbol{n} - \boldsymbol{w} \cdot \boldsymbol{n} = 0 \quad \text{on } \partial\Omega.$$

Therefore, q defined by (4.9) and $\boldsymbol{u} = \boldsymbol{w} - \nabla q$ is a unique decomposition of $\boldsymbol{w}$ such that $\nabla \cdot \boldsymbol{u} = 0$ in Ω and $\boldsymbol{u} \cdot \boldsymbol{n} = 0$ on $\partial\Omega$. □

Thus, $\mathcal{P}$ is an orthogonal projection operator defined by

$$\mathcal{P}(\boldsymbol{w}) = \boldsymbol{u}, \quad \boldsymbol{w} \in \Omega,$$

where $\boldsymbol{w} = \boldsymbol{u} + \nabla q$ is the Helmholtz-Hodge decomposition of $\boldsymbol{w}$. Hence, applying $\mathcal{P}$ to (4.1) leads to (4.7).

Based on the formulation (4.7) of the incompressible Navier-Stokes equation, Chorin [6] and Tèmam [64] proposed a fractional step method to effectively decouple the pressure term from the velocity term. This method has been the first numerical scheme enabling a cost-effective solution of the 3D time dependent problems [57]. The scheme consists of two parts:

1. Calculate an intermediate velocity field $\boldsymbol{u}^*$ from the time discretized momentum equation omitting the pressure term. This approximation to the velocity field does not need to satisfy the incompressibility condition due to mass conservation. For instance, one could use Euler's method in time to obtain

$$\frac{\boldsymbol{u}^* - \boldsymbol{u}^n}{\Delta t} = -(\boldsymbol{u}^n \cdot \nabla)\boldsymbol{u}^n + \nu \Delta \boldsymbol{u}^n$$
$$\boldsymbol{u}^* = \boldsymbol{u}^n + \Delta t[-(\boldsymbol{u}^n \cdot \nabla)\boldsymbol{u}^n + \nu \Delta \boldsymbol{u}^n + \boldsymbol{f}^n],$$

   and impose the no-slip boundary conditions $\boldsymbol{u}^* = g^{n+1}$.

2. Enforce the incompressibility using the projection operator $\mathcal{P}$, i.e., decompose $\boldsymbol{u}^*$ in a solenoidal field $\boldsymbol{u}^{n+1}$ tangent to the boundary plus a gradient field

$$\boldsymbol{u}^* = \boldsymbol{u}^{n+1} + \nabla q,$$
$$\nabla \cdot \boldsymbol{u}^{n+1} = 0 \quad \text{and} \quad \boldsymbol{n} \cdot \boldsymbol{u}^{n+1}\Big|_{\partial\Omega} = 0.$$

   Here, q is the unique solution (up to a constant) of (4.9) with $\boldsymbol{w}$ replaced by $\boldsymbol{u}^*$.

A similar way to arrive to the same scheme, is to consider the time discretization of the momentum equation including the pressure term

$$\begin{aligned} \boldsymbol{u}^{n+1} &= \boldsymbol{u}^n + \Delta t[-\nabla \phi^{n+1} - (\boldsymbol{u}^n \cdot \nabla)\boldsymbol{u}^n + \nu \Delta \boldsymbol{u}^n + \boldsymbol{f}^n], \\ \boldsymbol{u}^{n+1} &= \boldsymbol{u}^* - \Delta t \nabla \phi^{n+1}, \end{aligned} \tag{4.10}$$

where ϕ^{n+1} is an approximation to the pressure at time t^{n+1}. The idea is to calculate the pressure term such that $\nabla \cdot \boldsymbol{u}^{n+1} = 0$ and $\boldsymbol{u}^{n+1} \cdot \boldsymbol{n} = 0$. Taking the divergence of

(4.10)

$$\nabla \cdot \boldsymbol{u}^{n+1} = \nabla \cdot \boldsymbol{u}^* - \Delta\phi^{n+1}\Delta t \rightarrow \Delta\phi^{n+1} = -\frac{1}{\Delta t}\nabla \cdot \boldsymbol{u}^*$$
$$\boldsymbol{n} \cdot \boldsymbol{u}^{n+1} = \boldsymbol{n} \cdot \boldsymbol{u}^* - \Delta t \boldsymbol{n} \cdot \nabla\phi^{n+1} \rightarrow \boldsymbol{n} \cdot \nabla\phi^{n+1} = \frac{1}{\Delta t}(\boldsymbol{u}^{n+1} - \boldsymbol{u}^*) \cdot \boldsymbol{n} = -\frac{1}{\Delta t}\boldsymbol{n} \cdot \boldsymbol{u}^*,$$

assuming $\boldsymbol{n} \cdot \boldsymbol{u}^{n+1}\Big|_{\partial\Omega} = 0$. In other words, we end up with the same system as before (4.9), with $q = \Delta t \nabla\phi^{n+1}$. Basically we are calculating the pressure such that $\boldsymbol{u}^{n+1}$ is divergence-free and tangent to the boundary.

4.3.2 Traditional and Divergence-Free RBF Spatial Discretizations

Using the fractional time stepping scheme of the previous subsection, we must approximate the Laplacian operator and the derivatives with respect to x and y in order to generate a spatial discretization for the intermediate step $\boldsymbol{u}^*$. For this task, we use the traditional or divergence-free RBF interpolants in a local stencil and calculate the differentiation weights that will approximate the differential operators of the local interpolants, as described in Chapter 2. Using Euler's method as a time integrator, the first fractional step of the fully discrete scheme is

$$\vec{\boldsymbol{u}}^* = \vec{\boldsymbol{u}}^n + \Delta t[-\vec{\boldsymbol{F}}^n - \nu L\vec{\boldsymbol{u}}^n + \vec{\boldsymbol{f}}^n], \tag{4.11}$$

where

$$\vec{\boldsymbol{F}}^n = \begin{bmatrix} \vec{u}^n \\ \vec{u}^n \end{bmatrix} \circ D_x\vec{\boldsymbol{u}}^n + \begin{bmatrix} \vec{v}^n \\ \vec{v}^n \end{bmatrix} \circ D_y\vec{\boldsymbol{u}}^n$$

with L, D_x and D_y being discrete operators approximating the vector Laplacian and the derivatives in respect to x and y, respectively. If f is a scalar function, $\vec{f}$ denotes its discretization over the interpolation nodes, while the vector function $\vec{\boldsymbol{f}}$ is the stacked discretization of each component of the vector $\boldsymbol{f} = \left[\begin{smallmatrix} f_u \\ f_v \end{smallmatrix}\right]$, i.e., $\vec{\boldsymbol{f}} = \begin{bmatrix} \vec{f}_u \\ \vec{f}_v \end{bmatrix}$. The $\circ$ represent the Hadamard product, i.e., the entrywise product of matrices.

When using divergence-free interpolation the discrete operators above will be $2N \times 2N$ matrices, where N is the total number of nodes in the domain. However, the traditional RBFs, we interpolate in each component of the vector field separately, giving rise to $N \times N$ matrices. Thus, for (4.11), when using traditional RBF interpolants

$$D_x = \begin{bmatrix} D_x^{trad} & \\ & D_x^{trad} \end{bmatrix}, \quad D_y = \begin{bmatrix} D_y^{trad} & \\ & D_y^{trad} \end{bmatrix} \quad \text{and} \quad L = \begin{bmatrix} L^{trad} & \\ & L^{trad} \end{bmatrix}.$$

with the superscript *trad* denoting the discrete differential operators coming from traditional RBF interpolation.

In Section 2.2 we have seen that the divergence-free interpolants can always be found in a given set of nodes, even when the discrete vector field is not solenoidal. This of course will lead to a bad approximation of the derivatives using such an interpolant. For this reason, in the second step of the fractional method, we will use only traditional RBFs. Consequentially, approximating the differential operators in (4.9)

$$\begin{cases} I_i DG\vec{q} = I_i D\vec{\boldsymbol{u}}^* & \text{for the interior nodes,} \\ NI_b G\vec{q} = NI_b\vec{\boldsymbol{u}}^* & \text{for the boundary nodes,} \end{cases} \tag{4.12}$$

where $D = [D_x^{trad}\ D_y^{trad}]$, $G = \begin{bmatrix} D_x^{trad} \\ D_y^{trad} \end{bmatrix}$, $N = [\text{diag}\,(\vec{n}_x)\ \text{diag}\,(\vec{n}_y)]$ with $\boldsymbol{n} = (n_x, n_y)$ being the normal vector at the boundary. Additionally, the matrices I_i and I_b selects the interior and boundary elements of the discretized solution, respectively.

The Neumann problem (4.9) has a unique solution up to a constant, so the linear system (4.12) is expected to be singular. There are a couple ways to avoid the singularity of the matrix A originated from (4.12). One is to add an extra Dirichlet boundary condition. Another approach is to enforce the mean of the solution to be zero. Although both ways are straightforward to implement, the resulting system is not square. Moreover, it is known that assigning an arbitrary value to a node in

place of the Neumann condition (at that node) causes spurious values depending on the spatial discretization [13]. Instead of using those techniques, we follow the same approach used in [36], which is based on bordered matrices [31].

Consider the augmented system

$$\begin{bmatrix} A & \boldsymbol{c} \\ \boldsymbol{d}^\intercal & 0 \end{bmatrix} \begin{bmatrix} \vec{q} \\ \alpha \end{bmatrix} = \begin{bmatrix} \boldsymbol{b} \\ \beta \end{bmatrix} \quad \longrightarrow \quad \begin{cases} A\vec{q} + \alpha\boldsymbol{c} = \boldsymbol{b}, \\ \boldsymbol{d}^\intercal\vec{q} = \beta, \end{cases} \tag{4.13}$$

where $\boldsymbol{c} \notin \mathcal{R}(A)$ and $\boldsymbol{d} \notin \mathcal{R}(A^\intercal)$. This system is nonsingular and can be used to compute $\vec{q}$. Let $\boldsymbol{l} \in \mathcal{N}(A^\intercal)$ be the left singular vector of A, this way

$$\boldsymbol{l}^\intercal A\vec{q} + \alpha\boldsymbol{l}^\intercal\boldsymbol{c} = \boldsymbol{l}^\intercal\boldsymbol{b} \quad \Longrightarrow \quad \alpha = \frac{\boldsymbol{l}^\intercal\boldsymbol{b}}{\boldsymbol{l}^\intercal\boldsymbol{c}}.$$

The first constrain $A\vec{q} + \alpha\boldsymbol{c} = \boldsymbol{b}$ is satisfied, since

$$\boldsymbol{l}^\intercal(\boldsymbol{b} - \alpha\boldsymbol{c}) = \boldsymbol{l}^\intercal\boldsymbol{b} - \boldsymbol{l}^\intercal\boldsymbol{c}\frac{\boldsymbol{l}^\intercal\boldsymbol{b}}{\boldsymbol{l}^\intercal\boldsymbol{c}} = 0,$$

that is, $\boldsymbol{b} - \alpha\boldsymbol{c} \in \mathcal{N}(A^\intercal)^\perp = \mathcal{R}(A)$. Denoting this particular solution by $\vec{q}_p$ and observing that

$$A(\vec{q}_p + \gamma\boldsymbol{r}) + \alpha\boldsymbol{c} = A\vec{q}_p + \alpha\boldsymbol{c} = \boldsymbol{b},$$

for $\boldsymbol{r} \in \mathcal{N}(A)$, we conclude that $\left[\begin{smallmatrix} \vec{q}_p + \gamma\boldsymbol{r} \\ \alpha \end{smallmatrix}\right]$ is the solution of (4.13) where γ is determined such that $\boldsymbol{d}^\intercal\vec{q} = \beta$ is satisfied. The advantage of solving the augmented system is that we will end up with a nonsingular square and sparse linear system where iterative methods can be applied more efficiently than direct solvers. The caveat with this concept is how to choose the vectors $\boldsymbol{c}$ and $\boldsymbol{d}$.

When dealing with usual finite difference schemes, a good choice for $\boldsymbol{d}$ is a normalized constant vector, because A approximates a differential operator. Nevertheless, when using only RBFs to approximate differential operators, we will recover constant functions only in the flat limit of the shape parameter ($\varepsilon \to 0$). This can be a daunting

task without using stable computations for the finite difference weights to avoid the severe ill-conditioning in the interpolation process. We remediate this by adding a constant term to the RBF expansion using formulation (2.4) for presented in Chapter 2 to reproduce the constant function exactly.

Choosing $\boldsymbol{c}$ is more complicated because we lack information about the $\mathcal{N}(A^\intercal)$, except when A is symmetric, which only happens in structured grids and with a careful implementation of boundary conditions. The approach used here is to calculate the left singular vector associated with the zero singular value of A. Although we only need $\boldsymbol{d} \notin \mathcal{R}(A)$, if $\boldsymbol{c} = \boldsymbol{l}$

$$A\vec{q} = \boldsymbol{b} - \alpha\boldsymbol{c} = \boldsymbol{b} - \boldsymbol{l}\boldsymbol{l}^\intercal\boldsymbol{b} = (I - \boldsymbol{l}\boldsymbol{l}^\intercal)\boldsymbol{b},$$

that is, by solving the augmented linear system with this choice of $\boldsymbol{c}$, we are solving the least squares problem $A\vec{q} = \boldsymbol{b}$ in case the compatibility condition is not satisfied. This is a preferable way to solve the discrete Neumann problem when $\boldsymbol{l}$ is available [56].

4.4 Fluid Flow Numerical Experiments

In this section we present numerical simulations of the lid driven cavity flow and buoyancy driven flow. In our simulations we use a uniform Cartesian grid with 101 points in x- and y-directions for the spatial discretization of the generated finite differences schemes. We calculate the weights using interpolation on a local 3×3 stencil and approximate the derivatives at the center node. For boundary points, we use the closest 8 grid points generating a sided finite difference approximation. We choose the shape parameter such that the condition number of the interpolation matrix was kept constant at 10^8 and we add polynomial basis terms that guarantees constant reproduction to standard RBF and divergence-free kernel expansions.

We compare the solutions of the approximations based on RBFs with the spectral solution of each problem using the vorticity-stream function formulation. For the buoyancy driven flow, we use a 32×32 tensor product of Chebyshev points for the cases of Rayleigh numbers of 10, 500 and 1000. For the lid driven cavity flow, a 32×32 resolution was used for Reynolds number 10, and a 64×64 was necessary for Reynolds number 100 and 1000.

4.4.1 Lid Driven Cavity Flow

Re $= 10$

Figure 4.2 displays the contour lines of the stream function and vorticity for Re $= 10$. Figure 4.4 shows the relative error for the vorticity and the components of the velocity at time $t_f = 50$ where a time step size $\Delta t = 1 \times 10^{-5}$ was used for the time steeping scheme. We see that the divergence-free scheme introduce more error than the traditional RBF-FD scheme. This result was unexpected at first since in Chapter 3 we observe that the divergence free schemes have a higher order of decay for u_x and v_y. A possible reason for the inaccuracy in the divergence-free kernel based code, and possible corrections, are discussed in Chapter 5. In Figure 4.3 we display the velocity field color by speed (in log scale) with normalized arrows indicating flow direction.

Re $= 100$

For the case Re $= 100$, we use a 64×64 grid for the spectral method, which is sufficient to resolve the flow well – see Figure 4.5. The velocity field and the pressure are displayed in Figure 4.6. We used a time step of $\Delta t = 3.9 \times 10^{-5}$ and all the plots are for $t_f = 50$. Figure 4.7 shows one more time that the error on the vorticity and velocity components are smaller for the scheme based on traditional RBF interpolation.

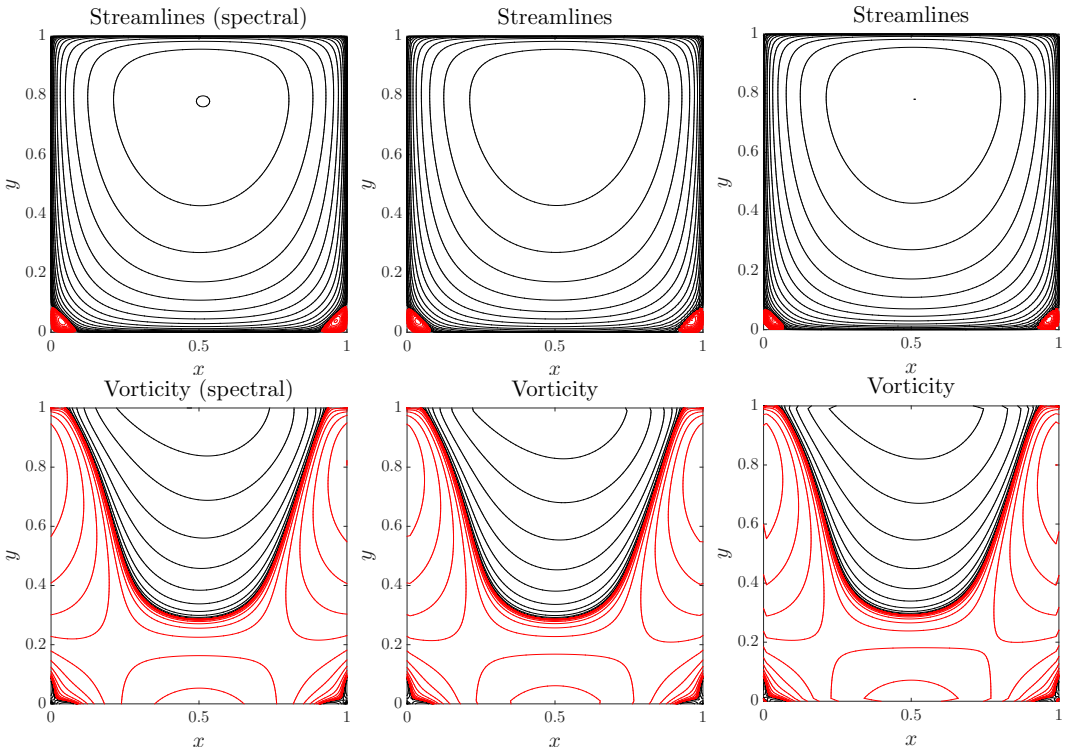

Figure 4.2: Contour levels of vorticity and stream function Re = 10 at $t_f = 50$ using $\Delta t = 1 \times 10^{-5}$. *(1st column):* spectral method. *(2nd column):* traditional RBFs. *(3rd column):* divergence-free RBFs.

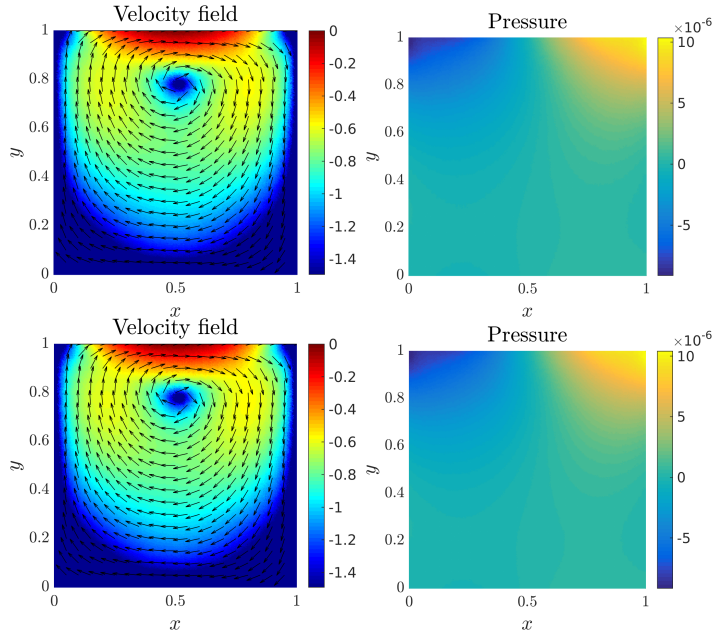

Figure 4.3: Velocity field and pressure for Re = 10 at $t_f = 50$ and using $\Delta t = 1 \times 10^{-5}$. *(Top):* traditional RBFs. *(Bottom):* divergence-free RBFs

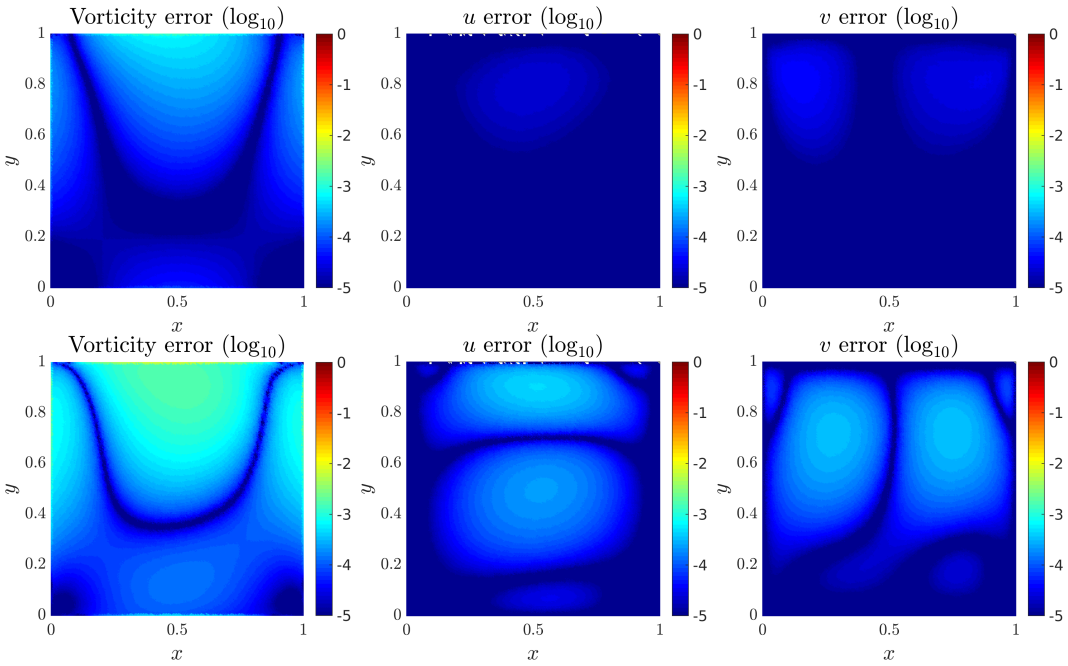

Figure 4.4: Vorticity and velocity field components error at $t_f = 50$ and using $\Delta t = 1 \times 10^{-5}$. *(Top):* traditional RBFs. *(Bottom):* divergence-free RBFs.

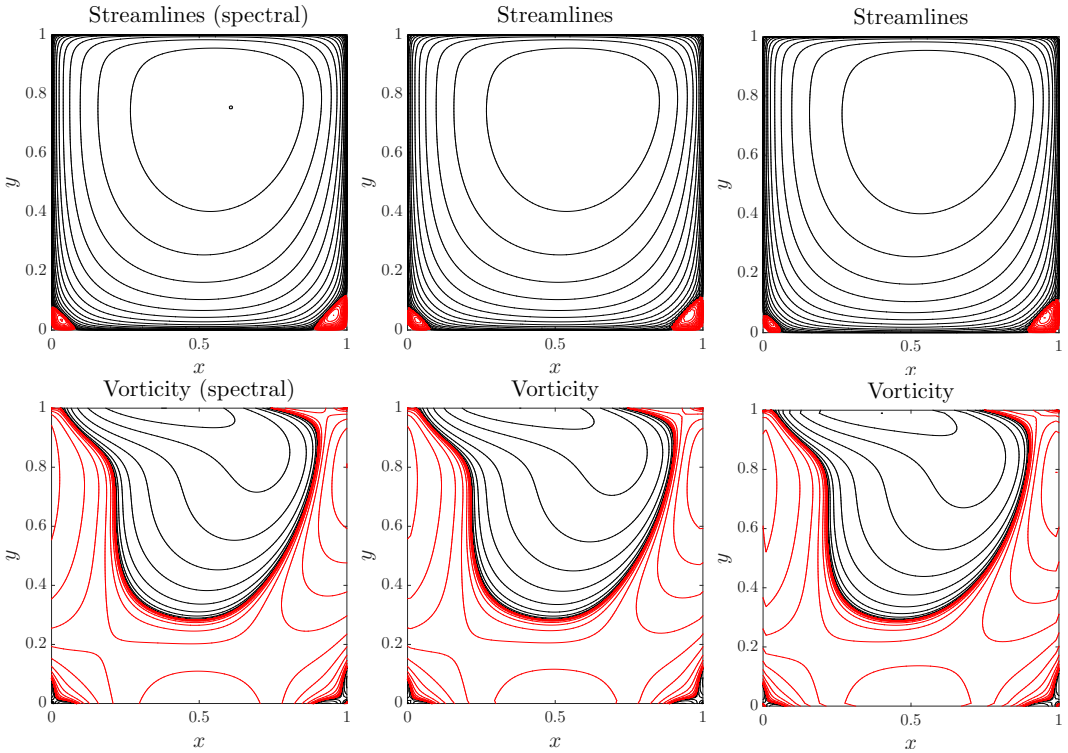

Figure 4.5: Contour levels of vorticity and stream function for $\mathrm{Re} = 100$ at $t_f = 50$ using $\Delta t = 3.9 \times 10^{-5}$. *(1st column):* spectral method. *(2nd column):* traditional RBFs. *(3rd column):* divergence-free RBFs.

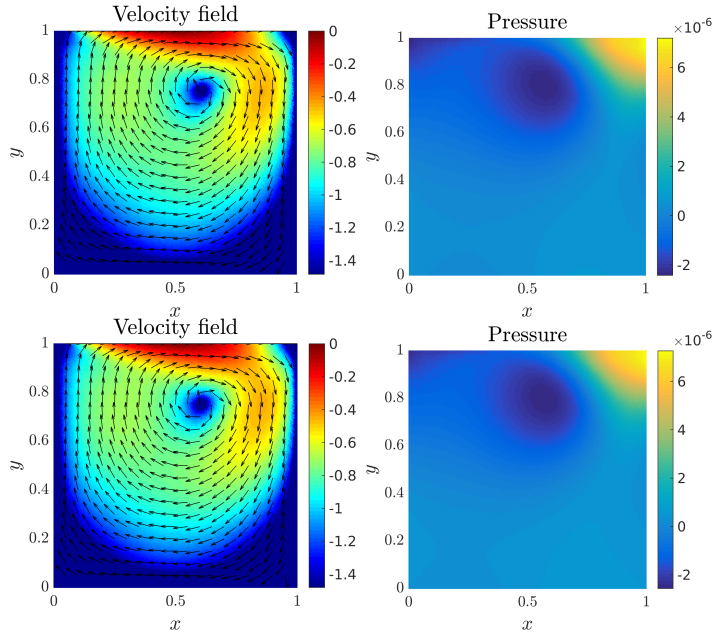

Figure 4.6: Velocity and pressure for Re = 100 at $t_f = 50$ using $\Delta t = 3.9 \times 10^{-5}$. *(Top):* traditional RBFs. *(Bottom):* divergence-free RBFs.

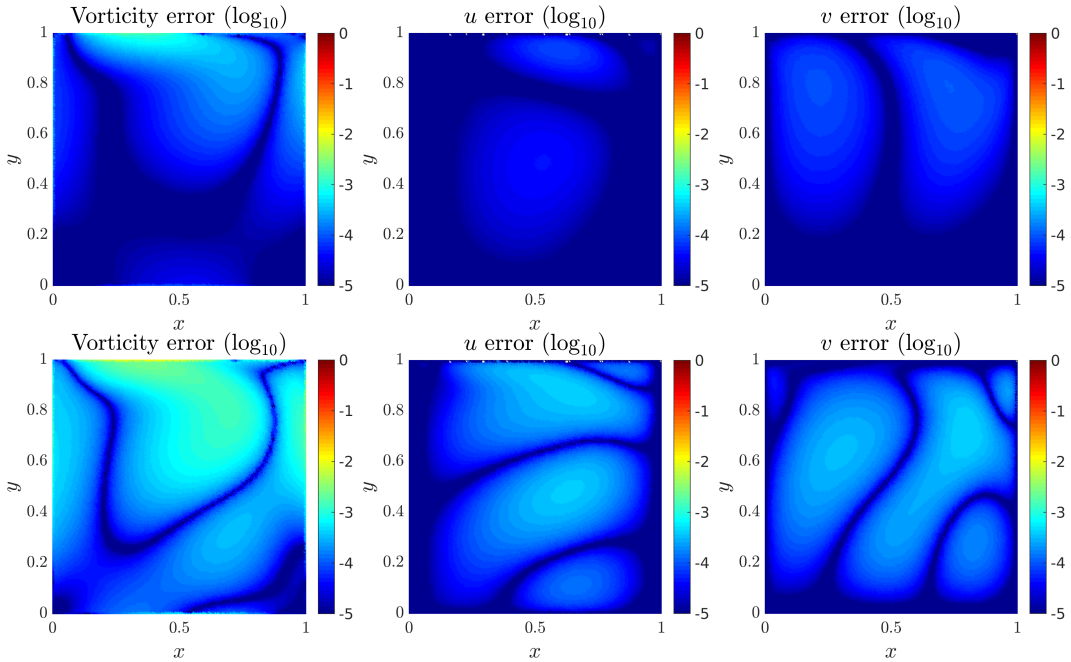

Figure 4.7: Vorticity and velocity error for Re = 100 at $t_f = 50$ using $\Delta t = 3.9 \times 10^{-5}$. *(Top):* traditional RBFs. *(Bottom):* divergence-free RBFs.

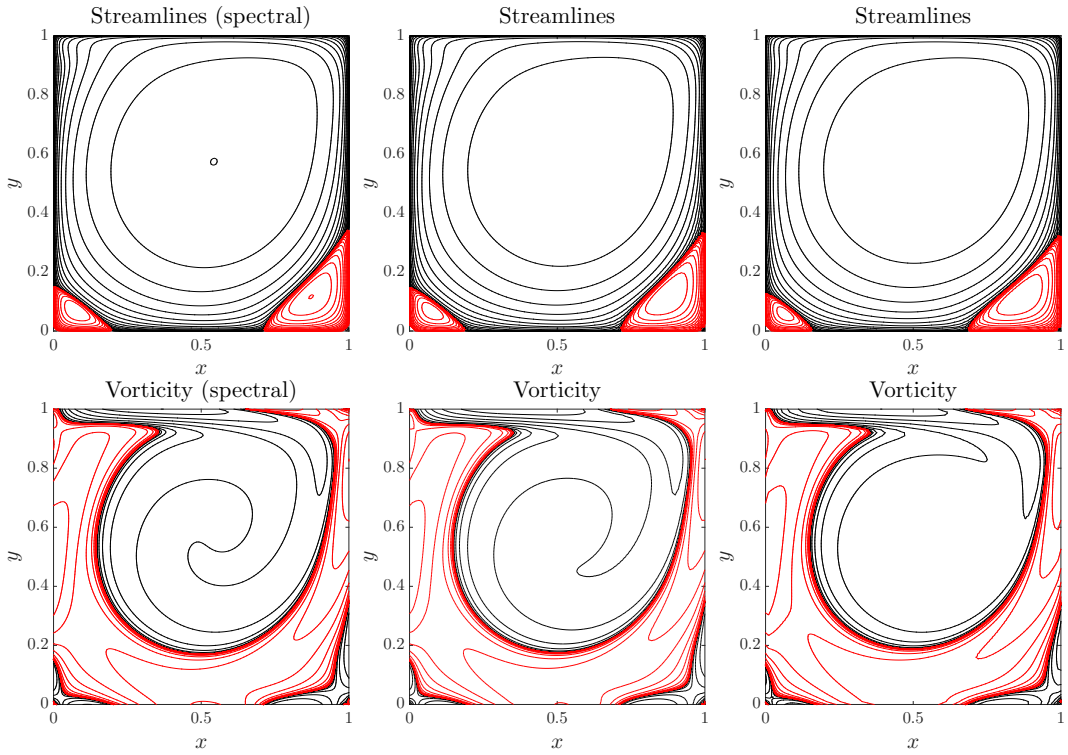

Figure 4.8: Contour levels of vorticity and stream function for Re = 1000 at $t_f = 50$ using $\Delta t = 3.9 \times 10^{-5}$. *(1st column):* spectral method. *(2nd column):* traditional RBFs. *(3rd column):* divergence-free RBFs.

Re = 1000

Figure 4.8 shows the stream lines and the vorticity for all three schemes. For this case we kept a 64×64 grid of Chebyshev points for the reference solution and time step with $\Delta t = 3.9 \times 10^{-5}$. The velocity field and the pressure are displayed on Figure 4.9. The error in the vorticity and velocity components are shown on Figure 4.10.

4.4.2 Buoyancy Driven Flow

The simulations of the buoyancy driven flow present similar results to the lid driven cavity flow simulations. Figures 4.13, 4.16, 4.19 all show better approximation for the traditional RBF scheme over the divergence-free scheme. In Figures 4.11, 4.14, and 4.17 we present streamlines and the vorticity for Ra = $100, 500$, and 1000, respectively.

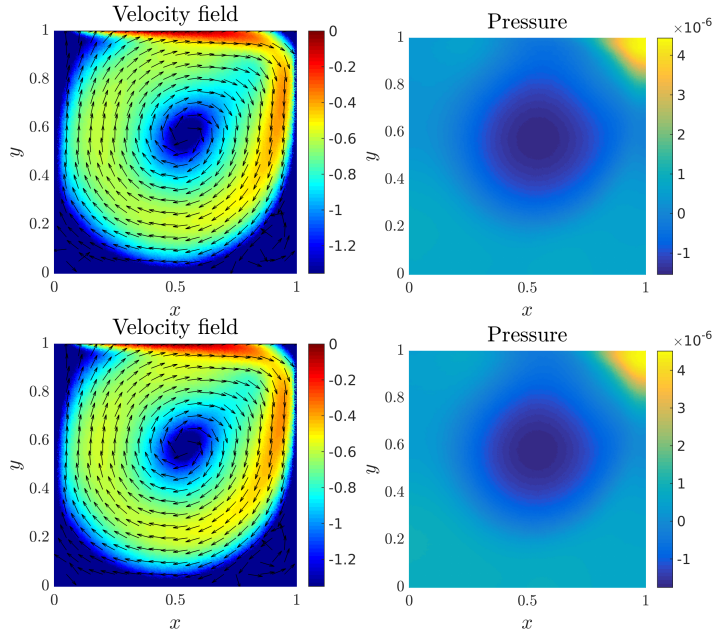

Figure 4.9: Velocity field and pressure for Re = 1000 at $t_f = 50$ using $\Delta t = 3.9 \times 10^{-5}$. *(Top):* traditional RBFs. *(Bottom):* divergence-free RBFs.

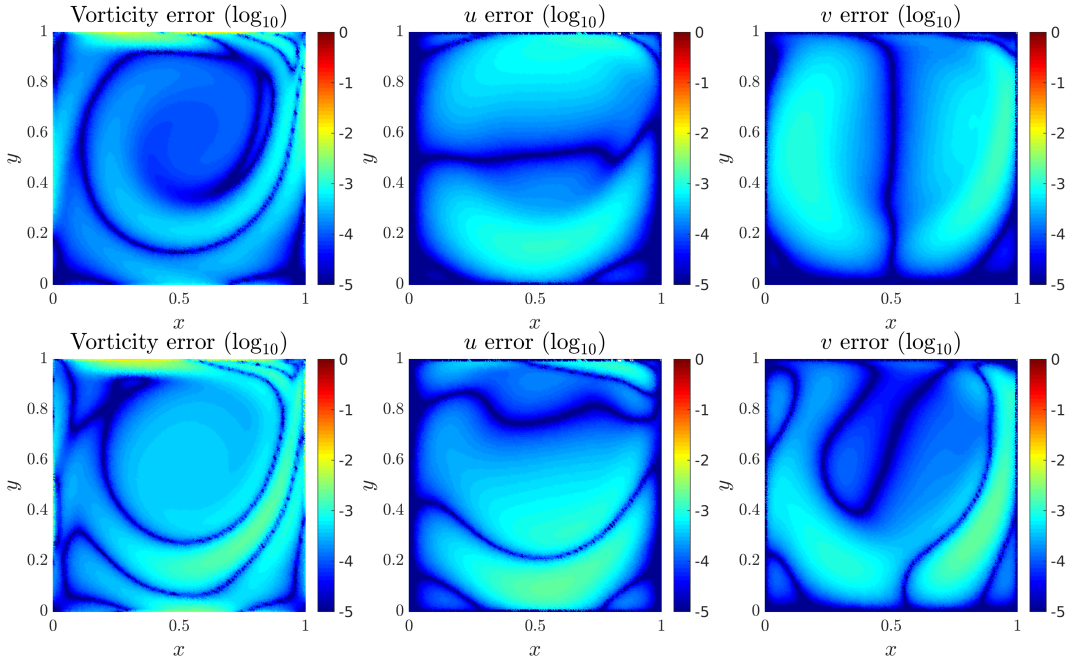

Figure 4.10: Vorticity and velocity field components error for Re = 1000 at time $t_f = 50$ using $\Delta t = 3.9 \times 10^{-5}$. *(Top):* traditional RBFs. *(Bottom):* divergence-free RBFs.

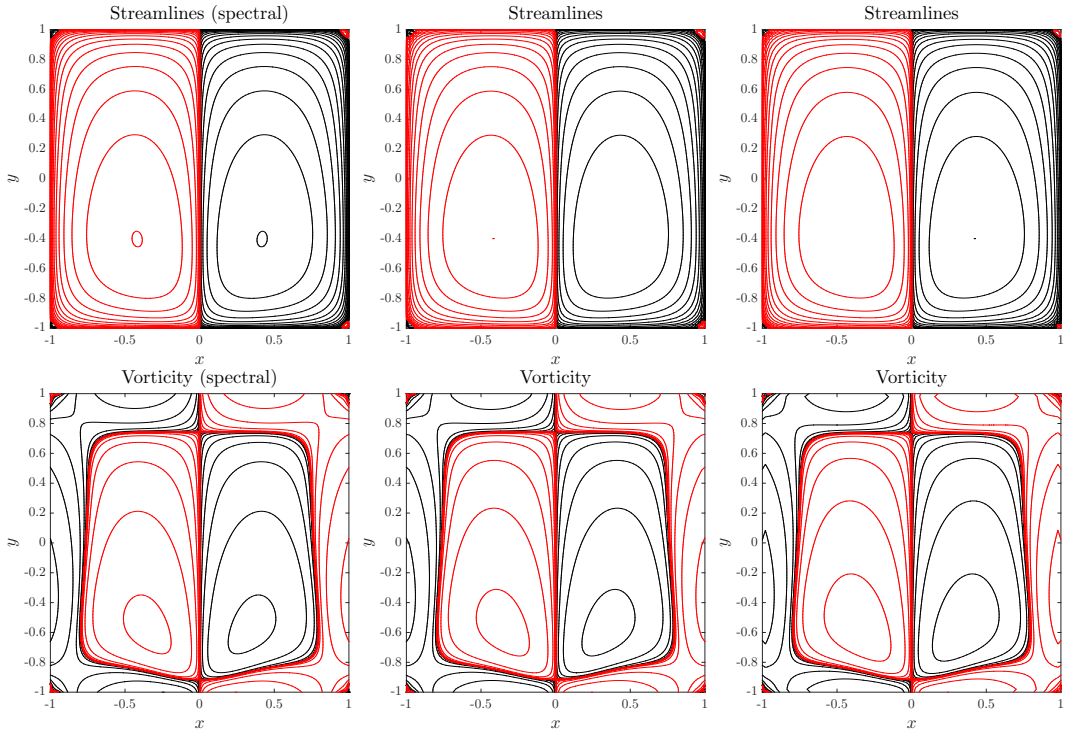

Figure 4.11: Contour levels of vorticity and stream function Ra = 100 at $t_f = 1$ and using $\Delta t = 2.22 \times 10^{-5}$. *(1st column):* spectral method. *(2nd column):* traditional RBFs. *(3rd column):* divergence-free RBFs.

The temperature inside the cavity, the temperature error, and the velocity field at $t_f = 1$ for each value of Ra are shown in Figures 4.12, 4.15, and 4.18.

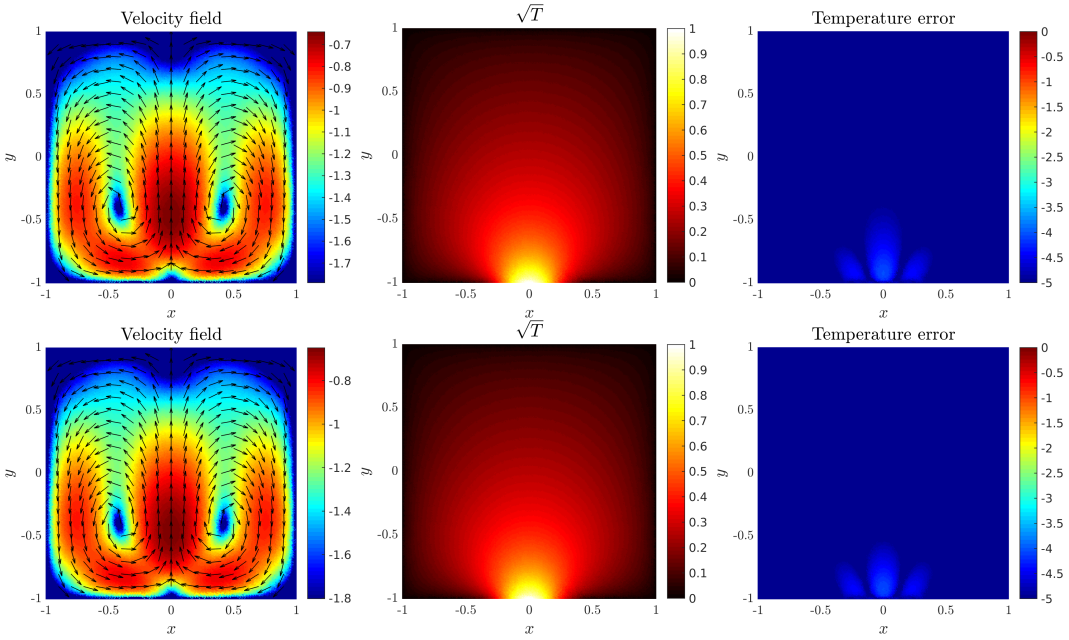

Figure 4.12: Velocity field, temperature and temperature error for Ra = 100 at $t_f = 1$ and using $\Delta t = 2.22 \times 10^{-5}$. *(Top):* traditional RBFs. *(Bottom):* divergence-free RBFs.

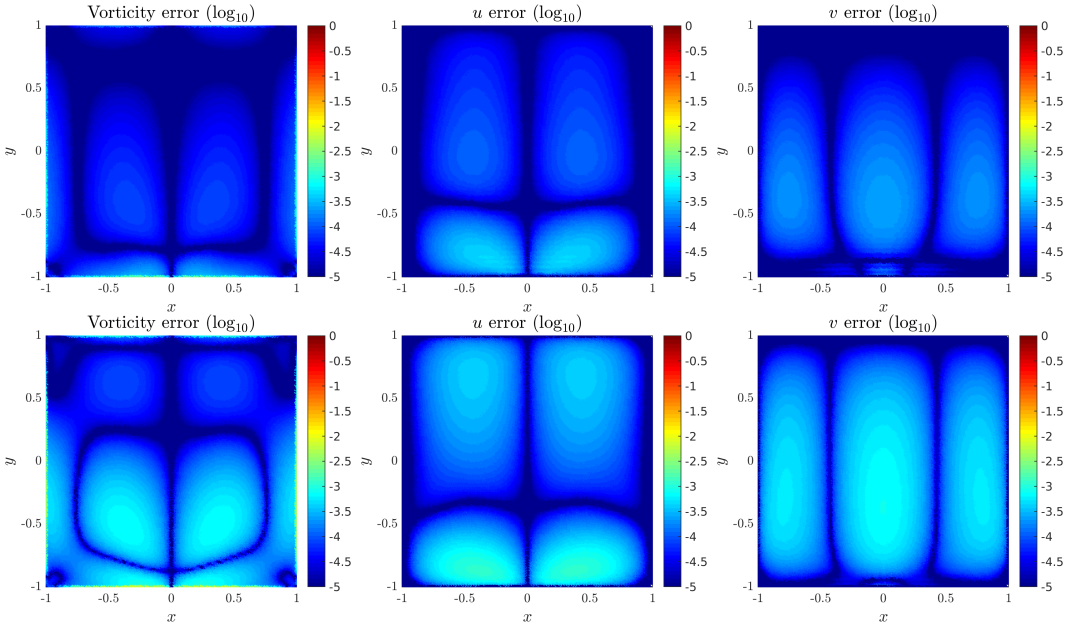

Figure 4.13: Vorticity and velocity field error for Ra = 100 at $t_f = 1$ using $\Delta t = 2.22 \times 10^{-5}$. *(Top):* traditional RBFs. *(Bottom):* divergence-free RBFs.

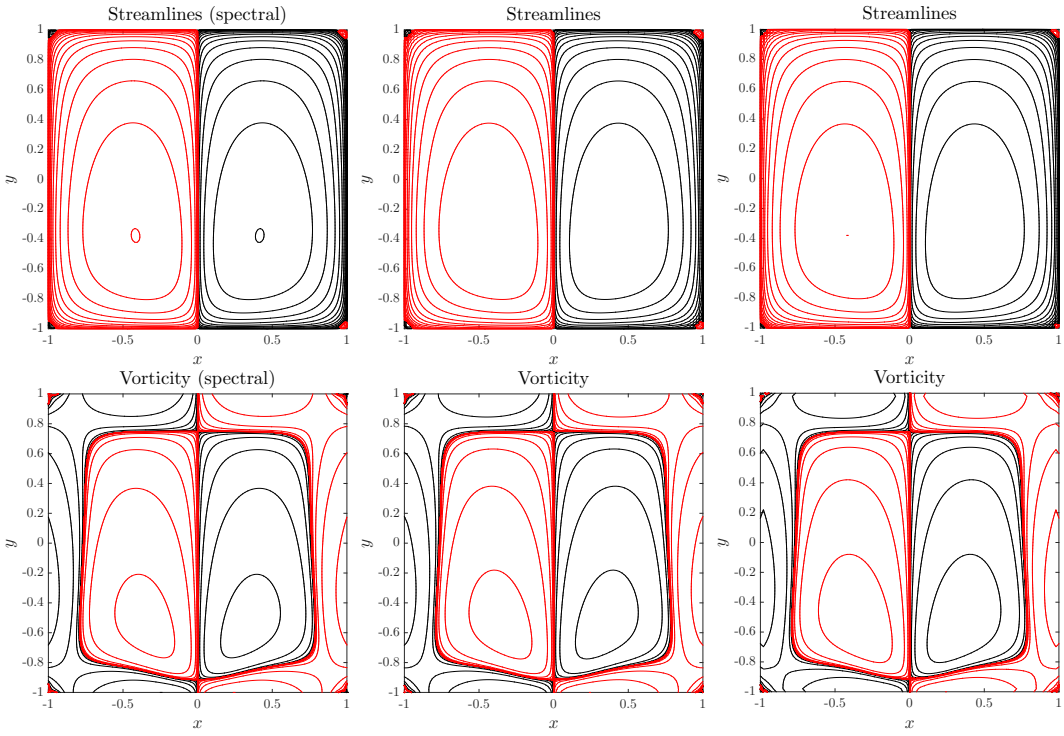

Figure 4.14: Contour levels of vorticity and stream function Ra $= 500$ at $t_f = 1$ and using $\Delta t = 4.44 \times 10^{-6}$. *(1st column):* spectral method. *(2nd column):* traditional RBFs. *(3rd column):* divergence-free RBFs.

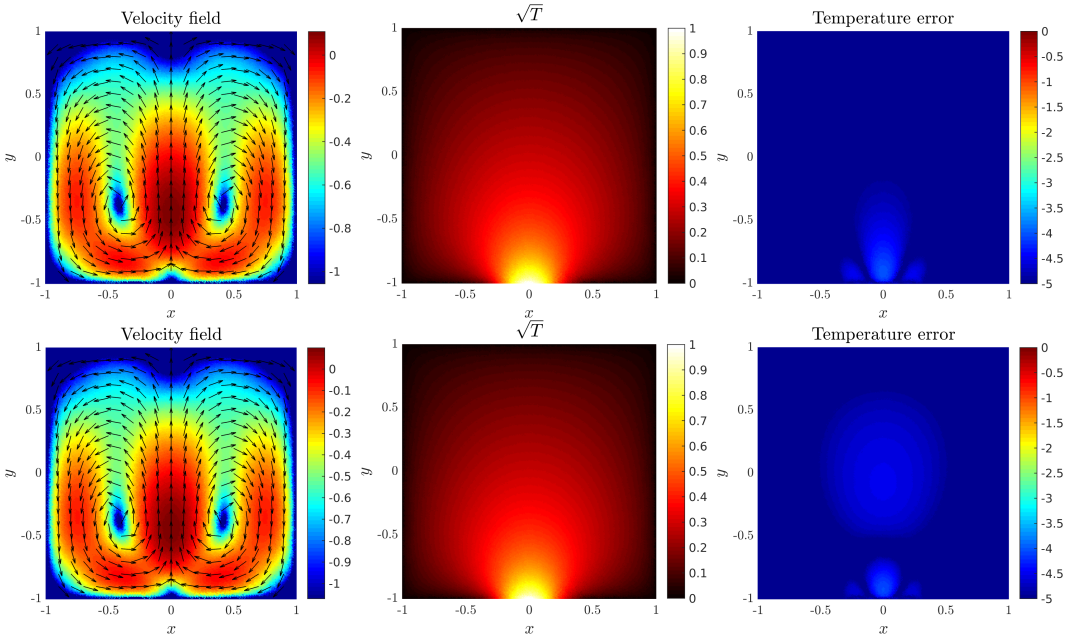

Figure 4.15: Velocity field, temperature and temperature error for Ra $= 500$ at $t_f = 1$ and using $\Delta t = 4.44 \times 10^{-6}$. *(Top):* traditional RBFs. *(Bottom):* divergence-free RBFs.

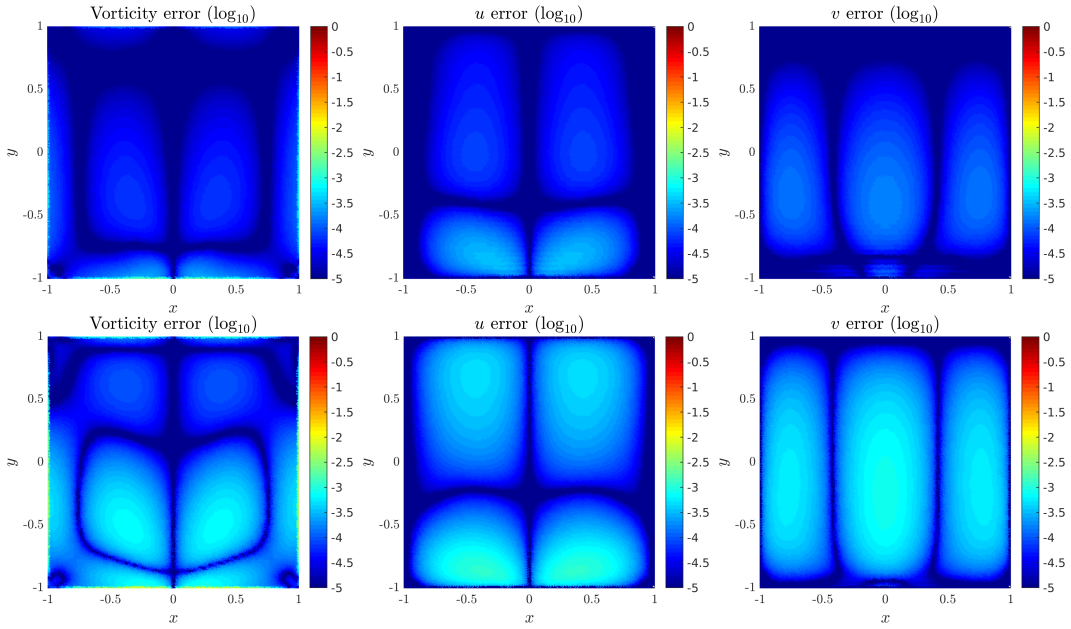

Figure 4.16: Vorticity and velocity field error for Ra = 500 at $t_f = 1$ using $\Delta t = 4.44 \times 10^{-6}$. *(Top):* traditional RBFs. *(Bottom):* divergence-free RBFs

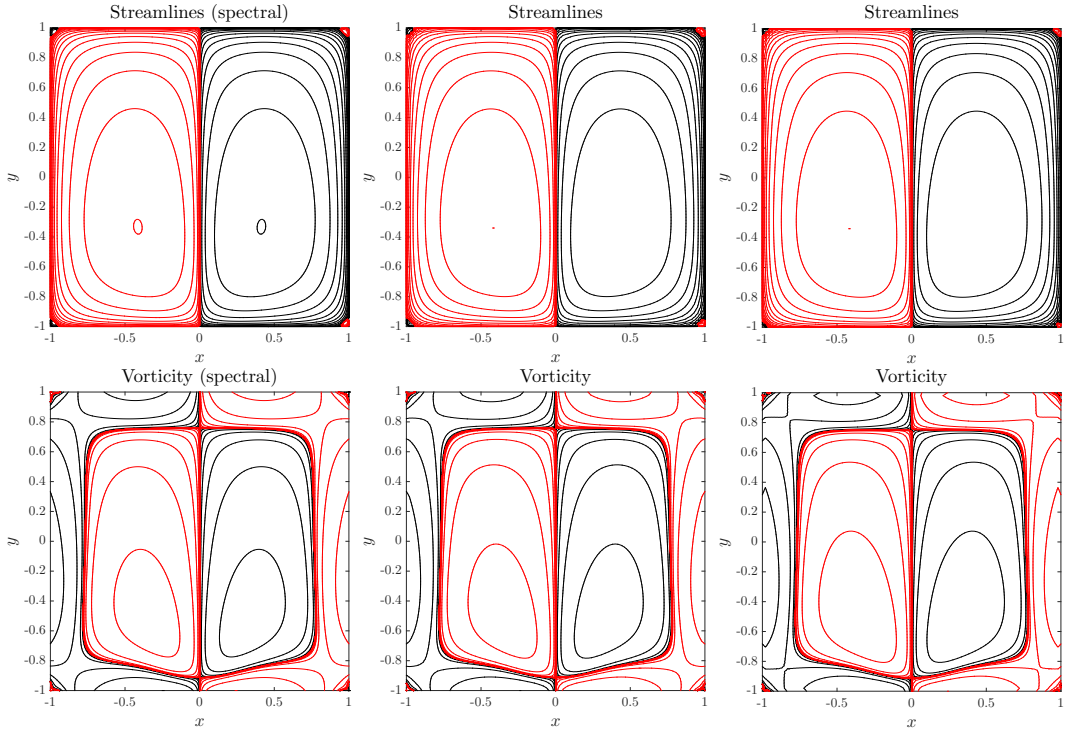

Figure 4.17: Contour levels of vorticity and stream function Ra = 1000 at $t_f = 1$ and using $\Delta t = 2.22 \times 10^{-6}$. *(1st column):* spectral method. *(2nd column):* traditional RBFs. *(3rd column):* divergence-free RBFs.

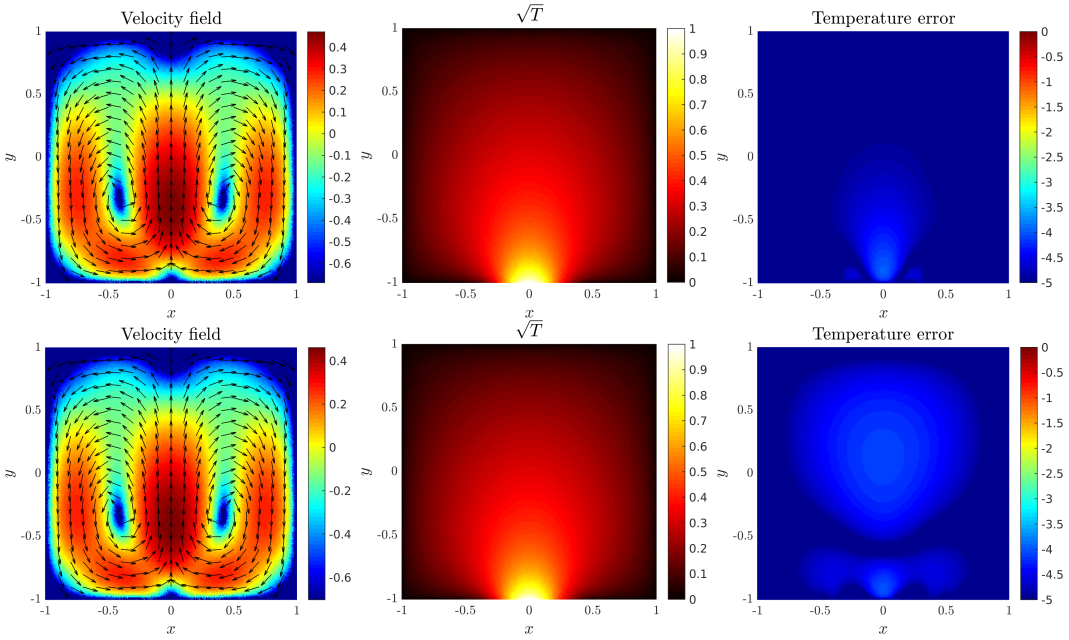

Figure 4.18: Velocity field, temperature and temperature error for Ra = 1000 at $t_f = 1$ and using $\Delta t = 2.22 \times 10^{-6}$. *(Top):* traditional RBFs. *(Bottom):* divergence-free RBFs.

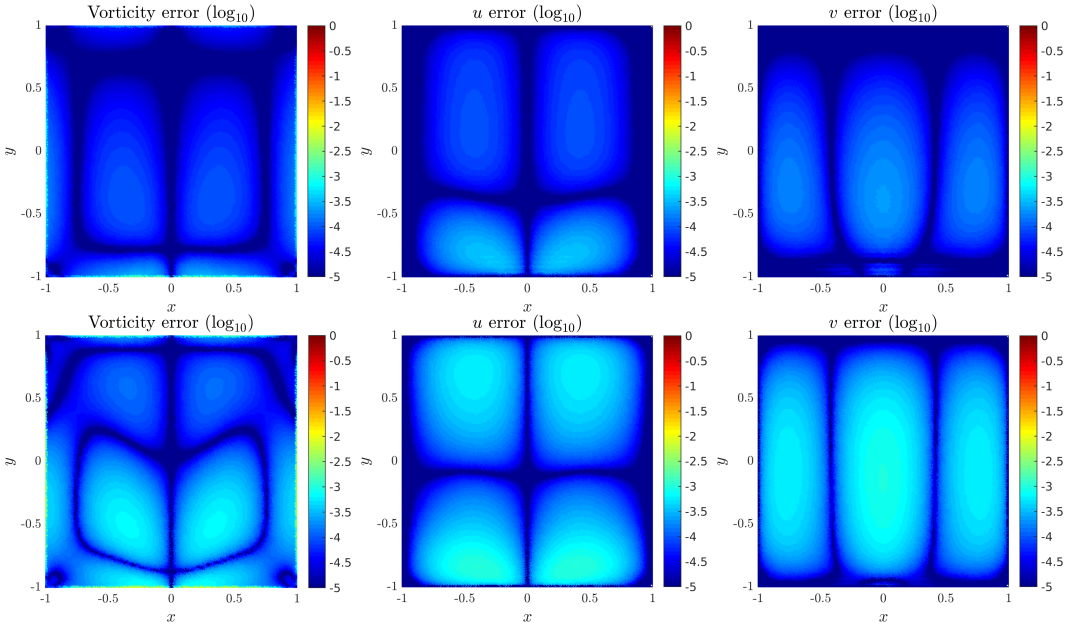

Figure 4.19: Vorticity and velocity field error for Ra = 1000 at $t_f = 1$ using $\Delta t = 2.22 \times 10^{-6}$. *(Top):* traditional RBFs. *(Bottom):* divergence-free RBFs

Chapter 5

FINAL REMARKS AND FUTURE DIRECTIONS

In this dissertation we were primarily concerned with properties of divergence-free based approximations. In particular, in Chapter 3 we were able to demonstrate that finite differences schemes based on divergence-free kernels improve the rate of convergence of partial derivatives when Cartesian stencils are used, assuming the vector field being approximated is divergence-free. Besides this, we derived weights for a divergence-free polynomial method on rectangular stencils. In the polynomial setting, the restriction to have the vector field solenoidal may be relaxed to $\nabla \cdot \boldsymbol{u} = \mathcal{O}(h^4)$ and still obtain convergence rates of $\mathcal{O}(h^4)$ for the derivatives u_x and u_y (see (3.3)). Strong evidence that the flat limit of divergence-free kernel approximations is polynomial was provided in Section 3.4. As for standard RBF approximations, the Lebesgue constants for global approximations by divergence-free kernel methods are strongly dependent on node distributions. As demonstrated in Section 3.3, clustering of nodes near the boundary of the domain is effective for controlling the growth of such constants (and hence stabilizing the approximation process).

Although the numerical experiments of Chapter 4 showed that approximations by divergence-free kernel methods were less accurate when compared to traditional RBFs methods, we must point out that we are using different shape parameters for each method. In our experiments, instead of fixing the shape parameter, we fixed the condition number of the interpolation matrix. Table 3.1 shows the conditioning of divergence-free kernels increasing faster than traditional RBFs when the amount points in the stencil increases and fixing the shape parameter. In other words, when using a flatter basis the condition number of the divergence-free scheme will be higher

than the one for traditional RBFs, and consequentially produce better approximations. Hence, we should be more careful with our comparison and the value we choose for the conditioning of each interpolation matrix.

Another reason the divergence-free generated finite differences did not perform well in our simulations is the presence of u_y and v_x in the incompressible Navier-Stokes equations, which did not present higher convergence rates in our experiments in Chapter 2. Better accuracy is only expected for the u_x and v_y derivatives. When u_y and v_x are combined in the momentum equation to march the velocity field forward in time, we lose the higher order accuracy.

In the simulation of fluid flow problems, we use the projection scheme due to Chorin [6] and Tèmam [64]. Although straightforward to implement for traditional RBF schemes, when only using divergence-free kernels, it fails to work. We point out that the equation (4.12) would not make sense if the discrete divergence operator D comes from a divergence-free kernel, since applying D returns zero by construction even for vector fields that are not solenoidal. Nevertheless, there has been a successful approach to approximate the Leray projector using divergence-free and curl-free kernels [26] based on the Helmhotz-Hodge decomposition introduce in [27]. Using this approximation to the Leray projector, instead of solving (4.12), has the advantage of not only imposing normal boundary conditions but also tangential boundary conditions.

There are many areas that remain to be investigated in using divergence-free approximations. One of them is to study the eigenvalue stability of these methods, and how they compare to traditional RBF approximations. In [50] it was shown that differential operators originated from Gaussian RBF approximations might lead to eigenvalues with positive real part, hindering the possibility to use explicit time stepping schemes. A similar study for divergence-free differential operators should be a valuable addition, and a detailed study is necessary, in particular when working

with scattered nodes.

Part of the source code used to generate the numerical results presented in this dissertation is available at [46].

REFERENCES

[1] Amodei, L. and M. Benbourhim, "A vector spline approximation", Journal of Approximation Theory **67**, 1, 51–79, URL http://dx.doi.org/10.1016/0021-9045(91)90025-6 (1991).

[2] Bhatia, H., G. Norgard, V. Pascucci and P.-T. Bremer, "The Helmholtz-Hodge decomposition – A survey", Visualization and Computer Graphics, IEEE Transactions on **19**, 8, 1386–1404, URL http://dx.doi.org/10.1109/TVCG.2012.316 (2013).

[3] Bochner, S., "Monotone funktionen, Stieltjessche integrale und harmonische analyse", Mathematische Annalen **108**, 1, 378–410, URL http://dx.doi.org/10.1007/BF01452844 (1933).

[4] Buhmann, M. D., *Radial basis functions: theory and implementations*, vol. 12 (Cambridge university press, 2003).

[5] Chang, K.-F., "Strictly positive definite functions", Journal of Approximation Theory **87**, 2, 148–158, URL http://dx.doi.org/10.1006/jath.1996.0097 (1996).

[6] Chorin, A. J., "Numerical solution of the Navier-Stokes equations", Mathematics of Computation **22**, 104, 745–762, URL http://dx.doi.org/10.1090/S0025-5718-1968-0242392-2 (1968).

[7] Chorin, A. J. and J. E. Marsden, *A mathematical introduction to fluid mechanics*, vol. 3 (Springer-Verlag, 1990).

[8] Courant, R. and D. Hilbert, *Methods of mathematical physics*, vol. 1 (Wiley, 1953).

[9] Dodu, F. and C. Rabut, "Vectorial interpolation using radial-basis-like functions", Computers & Mathematics with Applications **43**, 3–5, 393–411, URL http://dx.doi.org/10.1016/S0898-1221(01)00294-2 (2002).

[10] Dodu, F. and C. Rabut, "Irrotational or divergence-free interpolation", Numerische Mathematik **98**, 3, 477–498, URL http://dx.doi.org/10.1007/s00211-004-0541-x (2004).

[11] Driscoll, T. A. and B. Fornberg, "Interpolation in the limit of increasingly flat radial basis functions", Computers & Mathematics with Applications **43**, 3, 413–422, URL http://dx.doi.org/10.1016/S0898-1221(01)00295-4 (2002).

[12] Driscoll, T. A., N. Hale and L. N. Trefethen, "Chebfun guide", URL http://www.chebfun.org/docs/guide/chebfun_guide.pdf (2014).

[13] Escobar-Vargas, J., P. Diamessis and C. Van Loan, "The numerical solution of the pressure poisson equation for the incompressible navier–stokes equations using a quadrilateral spectral multidomain penalty method", J. Comp. Phys.(submitted) (2011).

[14] Fasshauer, G. F., *Meshfree Approximation Methods with MATLAB* (World Scientific Publishing Co., Inc., River Edge, NJ, USA, 2007).

[15] Flyer, N., E. Lehto, S. B. Blaise, G. B. Wright and A. St-Cyr, "A guide to RBF-generated finite differences for nonlinear transport: Shallow water simulations on a sphere", Journal of Computational Physics **231**, 11, 4078–4095, URL http://www.sciencedirect.com/science/article/pii/S0021999112000587 (2012).

[16] Foias, C., O. Manley, R. Rosa and R. Témam, *Navier-Stokes equations and turbulence*, vol. 83 (Cambridge University Press, 2001).

[17] Fornberg, B. and N. Flyer, *A primer on radial basis functions with applications to the geosciences*, vol. 87 (SIAM, 2015).

[18] Fornberg, B., E. Larsson and N. Flyer, "Stable computations with Gaussian radial basis functions", SIAM Journal on Scientific Computing **33**, 2, 869–892, URL http://dx.doi.org/10.1137/09076756X (2011).

[19] Fornberg, B. and C. Piret, "A stable algorithm for flat radial basis functions on a sphere", SIAM Journal on Scientific Computing **30**, 1, 60–80, URL http://dx.doi.org/10.1137/060671991 (2007).

[20] Fornberg, B. and G. B. Wright, "Stable computation of multiquadric interpolants for all values of the shape parameter", Computers & Mathematics with Applications **48**, 5, 853–867, URL http://dx.doi.org/10.1016/j.camwa.2003.08.010 (2004).

[21] Fornberg, B., G. B. Wright and E. Larsson, "Some observations regarding interpolants in the limit of flat radial basis fuctions", Computers & Mathematics with Applications **47**, 1, 37–55, URL http://dx.doi.org/10.1016/S0898-1221(04)90004-1 (2004).

[22] Fornberg, B. and J. Zuev, "The Runge phenomenon and spatially variable shape parameters in RBF interpolation", Computers & Mathematics with Applications **54**, 3, 379–398, URL http://www.sciencedirect.com/science/article/pii/S0898122107002210 (2007).

[23] Fuselier, E., "Sobolev-type approximation rates for divergence-free and curl-free RBF interpolants", Mathematics of Computation **77**, 263, 1407–1423, URL http://dx.doi.org/10.1090/S0025-5718-07-02096-0 (2008).

[24] Fuselier, E., F. Narcowich, J. Ward and G. Wright, "Error and stability estimates for surface-divergence free RBF interpolants on the sphere", Mathematics of Computation **78**, 268, 2157–2186, URL http://dx.doi.org/10.1090/S0025-5718-09-02214-5 (2009).

[25] Fuselier, E. J., "Improved stability estimates and a characterization of the native space for matrix-valued RBFs", Advances in Computational Mathematics **29**, 3, 269–290, URL http://dx.doi.org/10.1007/s10444-007-9046-3 (2008).

[26] Fuselier, E. J., S. Varun and G. B. Wright, "A radial basis function (RBF)-based Leray projection method for the incompressible unsteady Stokes equations", Computers & Fluids **128**, 41–52, URL http://dx.doi.org/10.1016/j.compfluid.2016.01.009 (2016).

[27] Fuselier, E. J. and G. B. Wright, "A radial basis function method for computing Helmholtz-Hodge decompositions", IMA Journal of Numerical Analysis URL http://arxiv.org/abs/1502.01575v2, accepted. preprint arXiv:1502.01575 (2016).

[28] Gerbeau, J.-F., C. Le Bris and M. Bercovier, "Spurious velocities in the steady flow of an incompressible fluid subjected to external forces", International Journal for Numerical Methods in Fluids **25**, 6, 679–695, URL http://dx.doi.org/10.1002/(SICI)1097-0363(19970930)25:6<679::AID-FLD582>3.0.CO;2-Q (1997).

[29] Ghia, U., K. N. Ghia and C. T. Shin, "High-re solutions for incompressible flow using the Navier-Stokes equations and a multigrid method", Journal of Computational Physics **48**, 3, 387–411, URL http://dx.doi.org/10.1016/0021-9991(82)90058-4 (1982).

[30] Golub, G. H. and C. Van Loan, "Matrix computations, 4th edition", (2012).

[31] Govaerts, W. J., *Numerical methods for bifurcations of dynamical equilibria*, vol. 66 (SIAM, 2000), URL http://dx.doi.org/10.1137/1.9780898719543.

[32] Guo, K., S. Hu and X. Sun, "Conditionally positive definite functions and Laplace-Stieltjes integrals", Journal of Approximation Theory **74**, 3, 249–265, URL http://www.sciencedirect.com/science/article/pii/S0021904583710658 (1993).

[33] Handscomb, D., "Interpolation and differentiation of multivariate functions and interpolation of divergence-free vector fields using surface splines", Tech. Rep. 91/5, Numerical Analysis Group, Oxford University Computing Laboratory, URL http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.50.533&rep=rep1&type=pdf (1991).

[34] Handscomb, D., "Local recovery of a solenoidal vector field by an extension of the thin-plate spline technique", Numerical Algorithms **5**, 2, 121–129, URL http://dx.doi.org/10.1007/BF02212043 (1993).

[35] Hardy, R. L., "Multiquadric equations of topography and other irregular surfaces", Journal of Geophysical Research **76**, 8, 1905–1915, URL http://dx.doi.org/10.1029/JB076i008p01905 (1971).

[36] Henshaw, W. D., "A fourth-order accurate method for the incompressible navier-stokes equations on overlapping grids", Journal of Computational Physics **113**, 1, 13–25, URL http://dx.doi.org/10.1006/jcph.1994.1114 (1994).

[37] Hirsch, C., *Numerical computation of internal and external flows: the fundamentals of computational fluid dynamics*, vol. 2 (Butterworth-Heinemann, 2007).

[38] Iske, A., *Multiresolution methods in scattered data modelling*, vol. 37 (Springer, 2004).

[39] Kansa, E. J., "Multiquadrics—A scattered data approximation scheme with applications to computational fluid-dynamics—I surface approximations and partial derivative estimates", Computers & Mathematics with Applications **19**, 8–9, 127–145, URL http://dx.doi.org/10.1016/0898-1221(90)90270-T (1990).

[40] Kansa, E. J., "Multiquadrics—A scattered data approximation scheme with applications to computational fluid-dynamics—II solutions to parabolic, hyperbolic and elliptic partial differential equations", Computers & Mathematics with Applications **19**, 8–9, 147–161, URL http://dx.doi.org/10.1016/0898-1221(90)90271-K (1990).

[41] Kosloff, D. and H. Tal-Ezer, "A modified Chebyshev pseudospectral method with an $\mathcal{O}(N^{-1})$ time step restriction", Journal of Computational Physics **104**, 2, 457–469, URL http://dx.doi.org/10.1006/jcph.1993.1044 (1993).

[42] Larsson, E. and B. Fornberg, "Theoretical and computational aspects of multivariate interpolation with increasingly flat radial basis functions", Computers & Mathematics with Applications **49**, 1, 103–130, URL http://dx.doi.org/10.1016/j.camwa.2005.01.010 (2005).

[43] Lee, Y. J., G. J. Yoon and J. Yoon, "Convergence of increasingly flat radial basis interpolants to polynomial interpolants", SIAM Journal on Mathematical Analysis **39**, 2, 537–553, URL http://dx.doi.org/10.1137/050642113 (2007).

[44] Lowitzsch, S., "Error estimates for matrix-valued radial basis function interpolation", Journal of Approximation Theory **137**, 2, 238–249, URL http://dx.doi.org/10.1016/j.jat.2005.09.008 (2005).

[45] Lowitzsch, S., "Matrix-valued radial basis functions: stability estimates and applications", Advances in Computational Mathematics **23**, 3, 299–315, URL http://dx.doi.org/10.1007/s10444-004-1786-8 (2005).

[46] Mitrano, A. A., "divfree-src", URL http://dx.doi.org/10.5281/zenodo.15647 (2015).

[47] Mitrano, A. A. and R. B. Platte, "A numerical study of divergence-free kernel approximations", Applied Numerical Mathematics **96**, 94–107, URL http://dx.doi.org/10.1016/j.apnum.2015.05.001 (2015).

[48] Narcowich, F. J. and J. D. Ward, "Generalized Hermite interpolation via matrix-valued conditionally positive definite functions", Mathematics of Computation **63**, 208, 661–687, URL http://dx.doi.org/10.1090/s0025-5718-1994-1254147-6 (1994).

[49] Narcowich, F. J., J. D. Ward and G. B. Wright, "Divergence-free RBFs on surfaces", Journal of Fourier Analysis and Applications **13**, 6, 643–663, URL http://dx.doi.org/10.1007/s00041-006-6903-2 (2007).

[50] Platte, R. and T. Driscoll, "Radial basis functions and related multivariate meshfree approximation methods: Theory and applications eigenvalue stability of radial basis function discretizations for time-dependent problems", Computers & Mathematics with Applications **51**, 8, 1251–1268, URL http://dx.doi.org/10.1016/j.camwa.2006.04.007 (2006).

[51] Platte, R. B., "How fast do radial basis function interpolants of analytic functions converge?", IMA Journal of Numerical Analysis **31**, 4, 1578–1597, URL http://imajna.oxfordjournals.org/content/31/4/1578.abstract (2011).

[52] Platte, R. B., "C^∞ compactly supported and positive definite radial kernels", SIAM Journal on Scientific Computing **37**, 4, A1934–A1956, URL http://dx.doi.org/10.1137/14M1000683 (2015).

[53] Platte, R. B. and T. A. Driscoll, "Polynomials and potential theory for Gaussian radial basis function interpolation", SIAM Journal on Numerical Analysis **43**, 2, 750–766, URL http://dx.doi.org/10.1137/040610143 (2005).

[54] Platte, R. B. and L. N. Trefethen, "Chebfun: A new kind of numerical computing", in "Progress in Industrial Mathematics at ECMI 2008", vol. 15 of *Mathematics in Industry*, pp. 69–87 (Springer, 2010), URL http://dx.doi.org/10.1007/978-3-642-12110-4_5.

[55] Platte, R. B., L. N. Trefethen and A. B. Kuijlaars, "Impossibility of fast stable approximation of analytic functions from equispaced samples", SIAM Review **53**, 2, 308–318, URL http://dx.doi.org/10.1137/090774707 (2011).

[56] Pozrikidis, C., "A note on the regularization of the discrete poisson–neumann problem", Journal of Computational Physics **172**, 2, 917–923, URL http://dx.doi.org/10.1006/jcph.2001.6857 (2001).

[57] Quartapelle, L., *Numerical solution of the incompressible Navier-Stokes equations*, vol. 113 (Birkhäuser-Verlag, 1993).

[58] Sarris, I., I. Lekakis and N. Vlachos, "Natural convection in rectangular tanks heated locally from below", International Journal of Heat and Mass Transfer **47**, 14–16, 3549–3563, URL http://dx.doi.org/10.1016/j.ijheatmasstransfer.2003.12.022 (2004).

[59] Schaback, R., "Multivariate interpolation by polynomials and radial basis functions", Constructive Approximation **21**, 3, 293–317, URL http://dx.doi.org/10.1007/s00365-004-0585-2 (2005).

[60] Schoenberg, I. J., "Metric spaces and completely monotone functions", Annals Mathematics **39**, 4, 811–841, URL http://dx.doi.org/10.2307/1968466 (1938).

[61] Shen, J., "Hopf bifurcation of the unsteady regularized driven cavity flow", Journal of Computational Physics **95**, 1, 228–245, URL http://dx.doi.org/10.1016/0021-9991(91)90261-I (1991).

[62] Shu, C., H. Ding and K. Yeo, "Local radial basis function-based differential quadrature method and its application to solve two-dimensional incompressible Navier-Stokes equations", Computer Methods in Applied Mechanics and Engineering **192**, 7, 941–954, URL http://dx.doi.org/10.1016/S0045-7825(02)00618-7 (2003).

[63] Sun, X., "Conditionally positive definite functions and their application to multivariate interpolations", Journal of Approximation Theory **74**, 2, 159–180, URL http://dx.doi.org/10.1006/jath.1993.1059 (1993).

[64] Témam, R., "Sur l'approximation de la solution des équations de navier-stokes par la méthode des pas fractionnaires (ii)", Archive for Rational Mechanics and Analysis **33**, 5, 377–385, URL http://dx.doi.org/10.1007/BF00247696 (1969).

[65] Townsend, A. and L. N. Trefethen, "An extension of Chebfun to two dimensions", SIAM Journal on Scientific Computing **35**, 6, C495–C518, URL http://dx.doi.org/10.1137/130908002 (2013).

[66] Trefethen, L. N., *Spectral methods in MATLAB*, vol. 10 (SIAM, 2000), URL http://dx.doi.org/10.1137/1.9780898719598.

[67] Wendland, H., "Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree", Advances in Computational Mathematics **4**, 1, 389–396, URL http://dx.doi.org/10.1007/BF02123482 (1995).

[68] Wendland, H., "Fast evaluation of radial basis functions: Methods based on partition of unity", in "Approximation Theory X: Wavelets, Splines, and Applications", pp. 473–483 (Vanderbilt University Press, 2002).

[69] Wendland, H., *Scattered data approximation*, vol. 17 (Cambridge University Press, 2005).

[70] Wright, G. B. and B. Fornberg, "Scattered node compact finite difference-type formulas generated from radial basis functions", Journal of Computational Physics **212**, 1, 99–123, URL http://dx.doi.org/10.1016/j.jcp.2005.05.030 (2006).

[71] Wu, Z., "Compactly supported positive definite radial functions", Advances in Computational Mathematics **4**, 1, 283–292, URL http://dx.doi.org/10.1007/BF03177517 (1995).