Variational Reconstruction of Vector and Scalar Images from Non-Uniform Samples

Muthuvel Arigovindan

Thèse N° 3329 (2005)

Thèse présentée à la faculté des sciences et techniques de l'ingénieur
pour l'obtention du grade de docteur ès sciences
et acceptée sur proposition du jury

Prof. Martin Vetterli, *président*
Prof. Michael Unser, *directeur de thèse*
Prof. Christophe Rabut, *rapporteur*
Dr. méd. Patrick Hunziker, *rapporteur*
Dr. MER. Michel Bierlaire, *rapporteur*

École polytechnique fédérale de Lausanne—2005

Cover design by Annette Unser
Printing and binding by Repro-EPFL
Typeset with LaTeX

Available at `http://bigwww.epfl.ch/`

Abstract

We address the problem of reconstructing scalar and vector functions from non-uniform data. The reconstruction problem is formulated as a minimization problem where the cost is a weighted sum of two terms. The first data term is the quadratic measure of goodness of fit, whereas the second regularization term is a smoothness functional. We concentrate on the case where the later is a semi-norm involving differential operators. We are interested in a solution that is invariant with respect to scaling and rotation of the input data. We first show that this is achieved whenever the smoothness functional is both scale- and rotation-invariant.

In the first part of the thesis, we address the scalar problem. An elegant solution having the above mentioned invariant properties is provided by Duchon's method of thin-plate splines. Unfortunately, the solution involves radial basis functions that are poorly conditioned and becomes impractical when the number of samples is large. We propose a computationally efficient alternative where the minimization is carried out within the space of uniform B-splines. We show how the B-spline coefficients of the solution can be obtained by solving a well-conditioned, sparse linear system of equations. By taking advantage of the refinable nature of B-splines, we devise a fast multiresolution-multigrid algorithm. We demonstrate the effectiveness of this method in the context of image processing.

Next, we consider the reconstruction of vector functions from projected samples, meaning that the input data do not contain the full vector values, but only some directional components. We first define the rotational invariance and the scale invariance of a vector smoothness functional, and then characterize the complete family of such functionals. We show that such a functional is composed of a weighted sum of two sub-functionals: (i) Duchon's

scalar semi-norm applied on the divergence field; (ii) and the same applied to each component of the rotational field. This forms a three-parameter family, where the first two are the Duchon's order of the above sub-functionals, and the third is their relative weight. Our family is general enough to include all vector spline formulations that have been proposed so far.

We provide the analytical solution for this minimization problem and show that the solution can be expressed as a weighted sum of vector basis functions, which we call the generalized vector splines. We construct the linear system of equations that yields the required weights. As in the scalar case, we also provide an alternative B-spline solution for this problem, and propose a fast multigrid algorithm.

Finally, we apply our vector field reconstruction method to cardiac motion field recovery from ultrasound pulsed wave Doppler data, and demonstrate its clinical potential.

Résumé

Nous abordons le problème de la reconstruction de fonctions scalaires et vectorielles à partir de données non uniformes. Le problème de reconstruction est formulé comme un problème de minimisation, où le coût est une somme pondérée de deux termes. Le premier terme est la mesure quadratique d'adéquation aux données, tandis que le deuxième terme est une fonctionnelle régularisante. Nous nous concentrons sur le cas où ce dernier est une seminorme impliquant des opérateurs différentiels. Nous sommes intéressés par une solution invariante par rapport au changement d'échelle et à la rotation des données d'entrée. Nous démontrons en premier que tel est le cas quand la fonctionnelle régularisante est invariante d'échelle et invariante par rotation.

Dans la première partie de la thèse, nous abordons le problème scalaire. Une solution élégante, possédant les propriétés mentionnées plus haut, est fournie par la méthode des splines plaque mince de Duchon. Malheureusement, la solution implique des fonctions de base radiales qui sont mal conditionnées, et perd son applicabilité quand le nombre d'échantillons est grand. Nous proposons une alternative efficace au point de vue calculatoire, où la minimisation est effectuée dans l'espace des B-Splines uniformes. Nous montrons comment les coefficients B-spline de la solution peuvent être obtenus en résolvant un système lineaire d'équations creux et bien conditionné. En profitant de la propriété d'échelle naturelle des B-splines, nous élaborons un algorithme rapide de multirésolution-multigrille. Nous démontrons l'efficacité de cette méthode dans le contexte du traitement d'images.

Puis, nous considérons la reconstruction de fonctions vectorielles à partir d'echantillons projetés, ce qui signifie que les données d'entrée ne contiennent pas les valeurs vectorielles complètes, mais seulement certaines composantes directionnelles. Nous définissons d'abord l'invariance par rotation

et l'invariance d'échelle d'une fonctionnelle vectorielle régularisante; puis nous caractérisons la famille entière de telles fonctionnelles. Nous montrons que celles-ci sont composées d'une somme pondérée de deux sous-fonctionnelles: (i) La semi-norme scalaire de Duchon appliquée au champ de divergence; (ii) et la même semi-norme appliquée à chaque composante du champ rotationnel. Cela forme une famille à trois paramètres, où les deux premiers sont l'ordre de régularité des sous-fonctionnelles mentionnées plus haut, et le troisième est leur poids relatif. Notre famille est suffisamment générale pour inclure toutes les formulations de splines vectorielles proposées jusqu'à présent.

Nous fournissons la solution analytique à ce problème de minimisation et démontrons qu'elle peut être exprimée comme une somme pondérée de fonction de base vectorielles, que nous appelons les splines vectorielles généralisées. Nous construisons le système linéaire d'équations qui fournit les poids exigés. Comme dans le cas scalaire, nous obtenons aussi une solution B-spline alternatif pour ce problème, et proposons un algorithme multigrille rapide.

Enfin, nous appliquons notre méthode de reconstruction de champs vectoriels à la reconstruction de champs de mouvement cardiaque à partir de données ultrason Doppler, et démontrons son potentiel pour une application clinique.

Contents

Chapter 1

Background and the Problem Addressed

1.1 Introduction

The problem of reconstructing images from nonuniform samples is frequently encountered in medical imaging and other areas of applied sciences. The non-uniformity of the distribution of sample locations is primarily due to the nature of the acquisition devices. Examples include spiral scanning [1], sector scanning [2], and free-hand scanning [3]. The computational problem is to recover a continuously defined image from these samples. The required image can be either a scalar image or a vectorial image. A typical example for the scalar problem is to recover the spatial distribution of tissue density of an organ [4]. An example for the vector problem is to recover velocity (both magnitude and direction) of each tissue region of a beating heart [5]. The focus of this thesis is to develop computational methods to recover scalar and vector images from such non-uniform samples.

We are interested in a situation where there is no restriction on the distribution of sample points. Since this obviously makes the problem ill-posed, we adopt a variational strategy, where a plausibility criterion is introduced in order to make the problem well-posed.

In this chapter, we first give some example applications where scalar and vector non-uniform sampling is encountered (Section 1.2), and then provide

a general overview of some reconstruction methods present in the literature (Section 1.3). We finally give an overview of our main contributions and an outline of the thesis.

1.2 Examples of Non-Uniform Sampling

1.2.1 Non-uniform Scalar Sampling

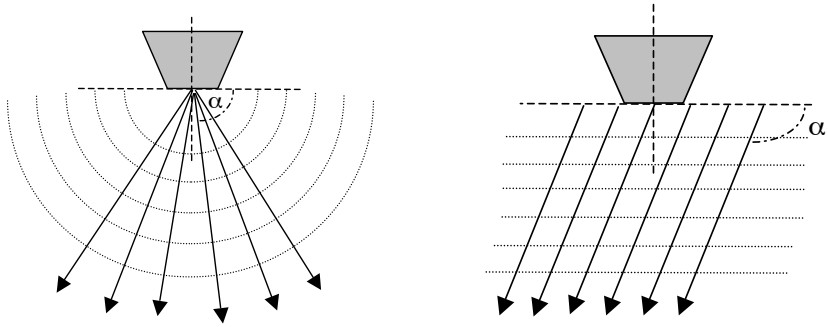
A well-known problem in image processing is the reconstruction of an image after a geometric transformation. The transformation, which typically deforms a regular grid of image pixels into a non-uniform grid, can arise due to variety of reasons. For example, in super-resolution imaging [6], it is required to reconstruct a high-resolution image from a set of low-resolution images of the same scene acquired from different views. The reconstruction problem amounts to transforming all the low-resolution images to a common reference system—which results in a non-uniform distribution of samples—and then to reconstructing a high-resolution image out of them. Other applications that involve a transformation and a subsequent reconstruction from the resulting non-uniform samples include motion compensation for video coding [7] and disparity compensation in stereo processing [8].

In medical imaging, a typical example is pulsed-wave ultrasound imaging. The imaging is performed along a series of scan lines. Each scan-line acquisition amounts to transmitting a periodic train of sinusoidal pulses at a chosen position and orientation. The backscattered wave, which is periodic as well, is measured by the transducer. Let $s_{\mathrm{rf}}(t)$ be the measured waveform, and let T be the period. If T is greater than the time taken to travel the penetration depth, then there is a one-to-one mapping between the time instant of the waveform $s_{\mathrm{rf}}(t)$ and the distance from the transducer along the scan line. This mapping is linear, since the speed of ultrasound is nearly constant across the tissue layers. For example, the scatterers in any interval $[d_1, d_2]$ contribute to the part of the signal in the intervals $\{[d_1/c + k\,T, d_2/c + k\,T],$ $k \in \mathbb{N}\}$, where c is the speed of sound in the probing medium. Hence, the strength of the signal in the intervals $\{[d_1/c + k\,T, d_2/c + k\,T],\ k \in \mathbb{N}\}$, provides information about the acoustic impedence of the tissue in the interval $[d_1, d_2]$. The system extracts the strength of the backscattered wave (intensity) for a set of regularly spaced intervals (known as sample volumes)

$\{[n\,d_{\rm s} - \delta, n\,d_{\rm s} + \delta],\ n \in \mathbb{N}\}$, where $d_{\rm s}$ is the sampling distance. To get the density for the sample volume $[n\,d_{\rm s} - \delta, n\,d_{\rm s} + \delta]$, the system computes the strength of the signal, $s_n(t) = s_{\rm rf}(t)\,G_n(t)$, where

$$\begin{aligned} G_n(t + k\,T) &= G_n(t),\ k \in \mathbb{N}. \\ G_n(t) &= 1,\ \forall t \in (1/c)\,[n\,d_{\rm s} - \delta, n\,d_{\rm s} + \delta], \\ &= 0,\ \text{otherwise.} \end{aligned}$$

The measurement from the sample volume $[n\,d_{\rm s} - \delta, n\,d_{\rm s} + \delta]$ is considered to be a point measurement, and $n\,d_{\rm s}$ is the sampling location. The distribution of such sampling locations depends on the way the scan lines are launched. It varies from system to system. We give two examples in Figure 1.1. The solid lines with arrows represent the scan lines, whereas the dotted lines represent the sampling grid along the scan lines. In the cone-beam scheme, the beams are launched from a single point along a series of regularly spaced angles. In the parallel-beam acquisition, beams are launched with a fixed angle from a set of points spaced regularly along a line. Note that the samples are non-uniformly distributed. The non-uniformity is more striking if one combines multiple acquisitions obtained from different views in a manner similar to super-resolution imaging.



(a) Cone-beam acquisition.

(b) Parallel-beam acquisition.

Figure 1.1: Sampling grids in ultrasound imaging.

1.2.2 Non-Uniform Vector Sampling

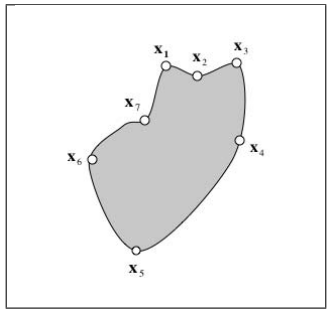
In medical imaging, it is often required to compute a correspondence function between a pair of images, which is typically known as the deformation field [9]. For example, the images can originate from a specific anatomical cross section of different patients. This task is often accomplished by identifying a set of corresponding landmarks. Let $(\mathbf{x}_i)_{i\in[1\ldots N]}$ and $(\mathbf{x}'_i)_{i\in[1\ldots N]}$ be such landmark points. These two landmark sets induce a measurement set $(\mathbf{x}_i, \mathbf{f}_i)_{i\in[1\ldots N]}$ on the deformation field $\mathbf{f}(\mathbf{x})$; we have N samples of the deformation field at locations $(\mathbf{x}_i)_{i\in[1\ldots N]}$ with the values $(\mathbf{f}_i = \mathbf{x}'_i - \mathbf{x}_i)_{i\in[1\ldots N]}$. The problem is to find a function $\mathbf{f}(\mathbf{x})$ such that $\mathbf{f}(\mathbf{x}_i) = \mathbf{f}_i$, $\forall i \in [1 \ldots N]$. With this setting, the image containing $(\mathbf{x}_i)_{i\in[1\ldots N]}$ is known as the source image and the one containing $(\mathbf{x}'_i)_{i\in[1\ldots N]}$ is known as the target image. This vector-sampling problem is demonstrated in Figure 1.2 using a toy example. We give the source and target images with the landmarks in Figures 1.2(a) and 1.2(b), and the samples of the required deformation field in Figures 1.2(c) and 1.2(d). These samples are visualized component-wise in an equivalent form in Figures 1.2(d) and 1.2(e).

The second interesting example is pulsed-wave Doppler imaging (PWD), which is an extension of the pulsed imaging described in Section 1.2.1. Here, in addition to the waveform, the frequency shift of the backscattered signal is also measured. This frequency shift is proportional to the axial velocity of the point under consideration, where the term "axial velocity" refers to the component of the true velocity along the ultrasound beam direction. We give in Figure 1.3 the schematic representation of a typical PWD imaging setup. If $\mathbf{d}_i$ denotes the direction of the beam probing the point $\mathbf{x}_i$, then its axial velocity s_i is related to the true velocity $\mathbf{v}(\mathbf{x})$ as follows:

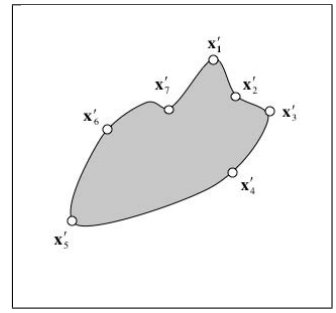
$$s_i = \mathbf{d}_i^\top \, \mathbf{v}(\mathbf{x}_i).$$

The measurement set for this imaging can be represented as $(\mathbf{d}_j, \mathbf{x}_j, s_j)_{j\in[1\ldots N]}$, and the reconstruction problem is the task of finding a function $\mathbf{f}(\mathbf{x})$ such that $\mathbf{d}_j^\top \, \mathbf{f}(\mathbf{x}_j) = s_j$, $\forall j \in [1 \ldots N]$.

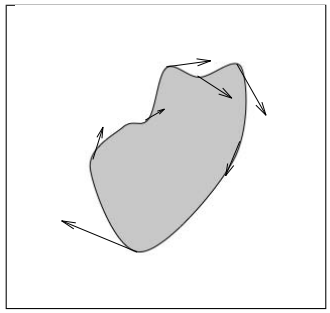
We illustrate in Figure 1.4 the sampling scheme of PWD imaging. Figure 1.4(a) contains a rotating object and Figure 1.4(b) depicts the axial velocity components measured by a cone-beam probe. The arrows represent the vectors $s_i \, \mathbf{d}_i$. The missing velocity components can be found in Figure 1.4(c);



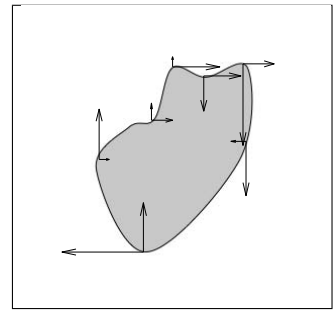
(a) Source image.



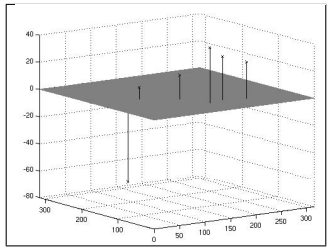
(b) Target image.



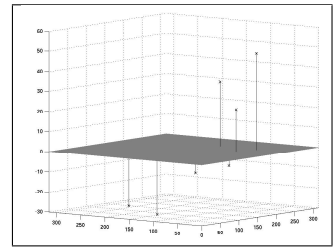
(c) Vector samples of the required deformation field.



(d) x and y components of the vector samples.



(e) Samples of the x component.



(f) Samples of the y component.

Figure 1.2: A toy example for the problem of computing a deformation field from landmarks.

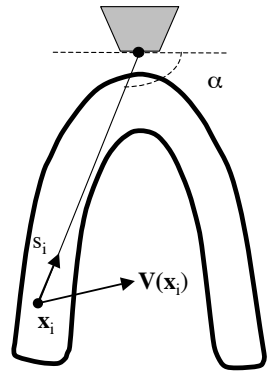


Figure 1.3: The axial velocity s_i corresponds to the projection of the true velocity $\mathbf{v}$ along the beam direction α.

they are equal to $\mathbf{v}(\mathbf{x}_i) - s_i\,\mathbf{d}_i$, and we give in Figure 1.4(d) the true velocities at the sampling locations.

Note that the sample set in the landmark interpolation problem $(\mathbf{x}_i, \mathbf{f}_i)_{i\in[1\ldots N]}$ can be also represented in a form similar to that of PWD imaging. The representation is given by $(\mathbf{d}_j, \mathbf{z}_j,\, s_j)_{j\in[1\ldots n\,N]}$, where the entries in the list are now defined by

$$\mathbf{d}_{n\,i+k} = \mathbf{e}_k,\ k \in [1\ldots n],\ i \in [0\ldots N-1],$$

$$s_{n\,i+k} = \{\mathbf{f}_{i+1}\}_k,\ k \in [1\ldots n],\ i \in [0\ldots N-1],$$

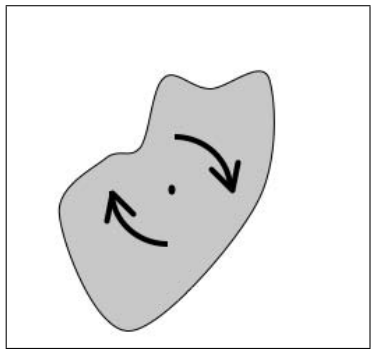
$$\mathbf{z}_j = \mathbf{x}_{\lceil j/n\rceil}.$$

The reconstruction problem now becomes one of finding a function $\mathbf{f}(\mathbf{x})$ such that $\mathbf{d}_j^\top\,\mathbf{f}(\mathbf{z}_j) = s_j,\ \forall j \in [1\ldots n\,N]$.

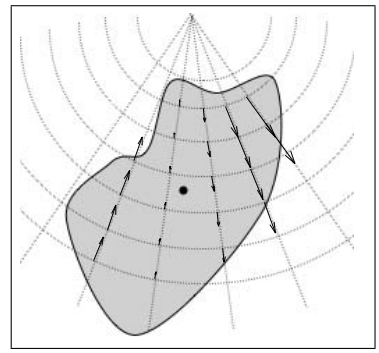
It is important to note that, even though the two examples given above can be represented by the same sampling model, the first one can be decomposed into a set of n scalar problems, whereas the second, in general, is a coupled problem.

1.3 Review of Reconstruction Methods

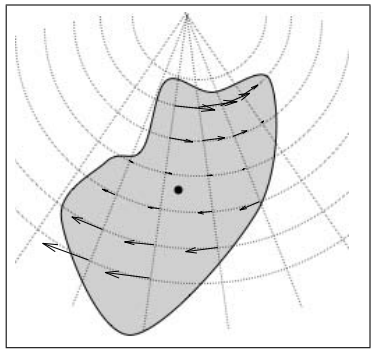
We provide here an overview of some available methods for the reconstruction of images from non-uniform samples [10]. This review is restricted to the



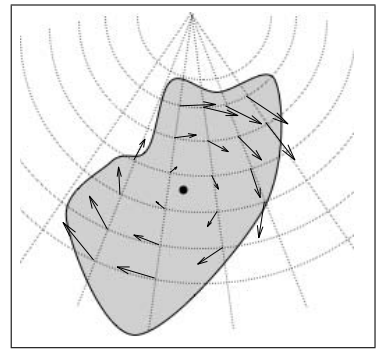
(a) A rotating object.



(b) Axial velocity components measured by the ultrasound probe. Arrows represent the vectors $s_i \, \mathbf{d}_i$.



(c) The missing velocity component at the sampling locations.



(d) True velocities at the sampling locations (sum of (b) and (c)).

Figure 1.4: Demonstration of the sampling model in PWD imaging.

scalar reconstruction problem. Obviously, these techniques can be applied to the vector problem in a component-wise fashion. Nevertheless, there exist also some variational techniques that are specifically tailored to vector fields; these will be discussed in Chapter 2.

1.3.1 Global Fitting Techniques

One of the oldest approach to the interpolation/approximation problem is based on global polynomial fitting. The required reconstruction is a polynomial of the form

$$p(x, y) = \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} a(i, j)\, x^i\, y^j,$$

where the coefficients are computed such that $p(x_i, y_i) = f_i$, $\forall i \in [1 \dots N]$, which amounts to solving a dense system of linear equations. In [11], the authors provide the conditions on the sample locations that guarantee a unique solution. The downside of this type of approach is that the solution is highly oscillatory. This is a notorious problem with high-order global polynomial models.

A similar method for band-limited images with an application to the reconstruction of potential fields has been proposed in [12, 13], where the author uses trignometric polynomials (*i.e.*, harmonic basis functions). The condition there for the uniqueness of the solution is that each Nyquist interval should contain one sample point. The author proposes a fast DCT-based algorithm for computing the required coefficients, which is the main strength of the proposed method.

1.3.2 Distance-Weighted Methods

One of the earliest methods that was proposed for 2D interpolation is Shepard's method [14]. Given some samples $(f_i)_{i\in[1\dots N]}$ and their corresponding locations $(\mathbf{x}_i)_{i\in[1\dots N]}$, the reconstructing function is expressed as

$$f(\mathbf{x}) = \frac{\sum_{i=1}^{N} \|\mathbf{x} - \mathbf{x}_i\|^{-2} f_i}{\sum_{i=1}^{N} \|\mathbf{x} - \mathbf{x}_i\|^{-2}}. \tag{1.1}$$

This method is known as the inverse-distance weighted method, for obvious reasons. The main problem with this reconstruction is that the function is not continuous at the data locations. Further, it is computationally expensive

to resample on a regular grid, especially when the input data size is large. In order to overcome these difficulties, a modified method was proposed in [15]. The modified reconstructing function is given by

$$f(\mathbf{x}) = \frac{\sum_{i=1}^{N} W_i(\mathbf{x})\, Q_i(\mathbf{x})}{\sum_{i=1}^{N} W_i(\mathbf{x})}, \tag{1.2}$$

where $(Q_i(\mathbf{x}))_{i\in[1\ldots N]}$ are some quadratic polynomials, and where $(W_i(\mathbf{x}))_{i\in[1\ldots N]}$ are the modified weight functions. The quadratic polynomials are expressed as

$$\begin{aligned} Q_i(x,y) &= c_{i1}(x-x_i)^2 + c_{i2}(y-y_i)^2 + c_{i3}(x-x_i)(y-y_i) \\ &\quad + c_{i4}(x-x_i) + c_{i5}(y-y_i) + f_i. \end{aligned}$$

Note that, by construction, these polynomials satisfy the interpolation condition

$$Q_i(\mathbf{x}_i) = f_i,\ \forall i \in [1\ldots N].$$

Further, for each $Q_i(\mathbf{x})$, the coefficients are chosen such that the following cost is minimized:

$$D_i = \sum_{j=1}^{N} W_j(\mathbf{x}_i)\ \left(Q_i(\mathbf{x}_j) - f_j\right)^2.$$

The modified weight function $W_i(\mathbf{x})$ of Frank *et al.* is

$$W_i(\mathbf{x}) = \left(\frac{T\ (R_i - \|\mathbf{x}-\mathbf{x}_i\|)}{R_i\ \|\mathbf{x}-\mathbf{x}_i\|}\right]^2,$$

where

$$T(x) = \begin{cases} x, & \text{if } x > 0, \\ 0, & \text{otherwise,} \end{cases}$$

and where R_i is the radius of influence for the sample point $\mathbf{x}_i$. Note that the introduction of the quadratic polynomials improves the smoothness at the sample locations, while the modified weight function results in a more local computational scheme thereby reducing complexity.

1.3.3 Moving Least-Squares (MLS) Methods

Moving least-squares methods reconstruct the required image by fitting polynomials locally [16]. Let $(p_j)_{j\in[1\ldots Q]}$ be a set of polynomials of degree

$\leq s$. The reconstruction problem is now to find a coefficient vector function $\mathbf{c}(\mathbf{x}) = [c_1(\mathbf{x}) \cdots c_Q(\mathbf{x})]^\top$ such that

$$f(\mathbf{x}) = \mathbf{c}(\mathbf{x})^\top \, \mathbf{p}(\mathbf{x}) \tag{1.3}$$

yields the required function, where $\mathbf{p}(\mathbf{x}) = [p_1(\mathbf{x}) \cdots p_Q(\mathbf{x})]^\top$. For each $\mathbf{x}$, the value of the coefficient function $\mathbf{c}(\mathbf{x})$ is defined as the minimizer of a quadratic cost functional J that is a function of $\mathbf{x}$, which is given by

$$J(\mathbf{x}) = \sum_{k=1}^{N} \left(f_k - \mathbf{c}(\mathbf{x})^\top \, \mathbf{p}(\mathbf{x}_i - \mathbf{x}) \right)^2 w(\mathbf{x}_k - \mathbf{x}), \tag{1.4}$$

where w is a radial-window function. In other words, for each $\mathbf{x}$, $f(\mathbf{x})$ is obtained first by computing a polynomial that fits the data within a neighborhood defined by $w(\cdot - \mathbf{x})$ in the least-squares sense, and then by evaluating it at $\mathbf{x}$.

The function $\mathbf{c}(\mathbf{x})$ that minimizes $J(\mathbf{x})$ for each $\mathbf{x}$ is given by

$$\mathbf{c}(\mathbf{x}) = \left(\mathbf{Q}^\top \, \mathbf{W}(\mathbf{x}) \, \mathbf{Q} \right)^{-1} \, \mathbf{Q}^\top \, \mathbf{W}(\mathbf{x}) \, \mathbf{f}, \tag{1.5}$$

where

$$\mathbf{Q} = [\mathbf{p}(\mathbf{x}_1) \cdots \mathbf{p}(\mathbf{x}_N)]^\top,$$

$$\mathbf{f} = [f_1 \cdots f_N]^\top,$$

and where $\mathbf{W}(\mathbf{x})$ is the diagonal matrix with entries $\{\mathbf{W}(\mathbf{x})\}_{ii} = w(\mathbf{x}_i - \mathbf{x})$.

Note that $Q = 1$ when $s = 0$. Hence, $\mathbf{p}(\mathbf{x})$ and $\mathbf{c}(\mathbf{x})$ are actually scalars for this restriction. In this case, the solution takes the simple form given by

$$f(\mathbf{x}) = c_1(\mathbf{x}) = \frac{\sum_{i=1}^{N} \, w(\mathbf{x}_i - \mathbf{x}) \, f_i}{\sum_{i=1}^{N} \, w(\mathbf{x}_i - \mathbf{x})}.$$

It is interesting to note that, if $w(\mathbf{x}) = \|\mathbf{x}\|^{-2}$, then the above solution is the same as Shepard's solution given by (1.1).

1.3.4 Mesh-Based Methods

Another class of empirical methods that are widely used in the context of surface reconstruction by the computer-graphics community is a mesh-based method [17, 18, 19, 20, 21]. First, a mesh is generated by triangulating the data points. Then, for each triangle, a polynomial surface is fit for its vertices. Finally, the sum of all the polynomials yields the required surface.

1.3.5 Reconstruction in Shift-Invariant Spaces

The problem of reconstructing functions in shift-invariant spaces has been addressed by many researchers [22, 23, 24, 25, 26, 27, 28]. See [29] for a recent survey. We present here the method described in [29] with an appropriate simplification. We start by defining a shift-invariant space: given a so-called generator ϕ, a shift-invariant space $\mathcal{V}(\phi)$ is defined as the space of functions of the form

$$f(\mathbf{x}) = \sum_{\mathbf{k}\in\mathbb{Z}^n} c_{\mathbf{k}}\, \phi(\mathbf{x}-\mathbf{k}) \text{ such that } \sum_{\mathbf{k}\in\mathbb{Z}^n} c_{\mathbf{k}}^2 < \infty. \tag{1.6}$$

In order to make the space $\mathcal{V}(\phi)$ well-defined, the following condition is imposed on the Fourier transform of ϕ:

$$0 <\leq \hat{a}_\phi(\boldsymbol{\omega}) = \sum_{\mathbf{j}\in\mathbb{Z}^n} \left|\hat{\phi}(\boldsymbol{\omega}+\mathbf{j}\,2\,\pi)\right|^2 \leq B < \infty, \text{ almost everywhere in } \boldsymbol{\omega}. \tag{1.7}$$

If this condition is satisfied, then the set $\{\phi(\cdot-\mathbf{k})\}$ forms a Riesz basis for $\mathcal{V}(\phi)$. This condition ensures that (1.6) is a stable representation. It also guarantees the existence of a function $\tilde{\phi}$ satisfying $\left\langle \tilde{\phi}(\cdot), \phi(\cdot-k)\right\rangle = \delta(k)$. It can be shown that $\left\{\tilde{\phi}(\cdot-\mathbf{k})\right\}$ also forms a Riesz basis. The function $\tilde{\phi}$ is called the dual generator. Since the dual generator belongs to $\mathcal{V}(\phi)$, it can be expressed as

$$\tilde{\phi}(\mathbf{x}) = \sum_{\mathbf{k}\in\mathbb{Z}^n} b_{\mathbf{k}}\, \phi(\mathbf{x}-\mathbf{k}).$$

The coefficients $b_{\mathbf{k}}$ are determined by the relation

$$\sum_{\mathbf{k}\in\mathbb{Z}^n} b_{\mathbf{k}}\, e^{-2\,\pi\,\mathbf{k}^\top\,\boldsymbol{\omega}} = \left(\sum_{\mathbf{k}\in\mathbb{Z}^n} \left|\hat{\phi}(\boldsymbol{\omega}+2\,\pi\,\mathbf{k})\right|^2\right)^{-1}.$$

In other words, $(b_{\mathbf{k}})$ is the discrete inverse Fourier transform of $(\sum_{\mathbf{k}} |\hat{\phi}(\boldsymbol{\omega}+2\,\pi\,\mathbf{k})|^2)^{-1}$.

The algorithm to reconstruct a function in $\mathcal{V}(\phi)$ from a set of its non-uniform samples relies on the existence of the so-called reproducing kernel. The reproducing kernel $K_{\mathbf{x}}$ associated with the location $\mathbf{x}$ is defined by the identity

$$f(\mathbf{x}) = \langle f, K_{\mathbf{x}}\rangle\,,\ \forall f \in \mathcal{V}(\phi).$$

A sufficient set of conditions for the existence of a reproducing kernel for every $\mathbf{x}$ is (i) that the generator ϕ is continuous, (ii) that it is absolutely

integrable, and (iii) that the sum $\sum_{\mathbf{k}} s_{\phi,\mathbf{k}}^2$ is finite, where

$$s_{\phi,\mathbf{k}} = \text{esssup } \phi(\mathbf{x}+\mathbf{k}),\ \mathbf{x} \in [0,1]^n.$$

If these conditions are satisfied, then the reproducing kernel can be expressed explicitly as

$$K_{\mathbf{x}}(\mathbf{y}) = \sum_{\mathbf{k}\in\mathbb{Z}^n} \phi(\mathbf{x}-\mathbf{k})\,\tilde{\phi}(\mathbf{y}-\mathbf{k}).$$

Now, we suppose that, for each $\mathbf{x}$, a unique reproducing kernel exists. Let $(\mathbf{x}_i)_{i\in[1\ldots N]}$ be the set of sampling locations. Then, any function $f \in \mathcal{V}(\phi)$ can be perfectly reconstructed from its samples $(f_i)_{i\in[1\ldots N]}$ if the associated reproducing kernels $(K_{\mathbf{x}_i})_{i\in[1\ldots N]}$ form a valid frame, or, in other words, when the functions $(K_{\mathbf{x}_i})_{i\in[1\ldots N]}$ are such that

$$A\ \|f\|^2 \leq \sum_{i=1}^{N} |\langle f, K_{\mathbf{x}_i}\rangle|^2 = \sum_{i=1}^{N} |f_i|^2 \leq B\ \|f\|^2\,,\ \forall f \in \mathcal{V}(\phi),$$

where A and B are some positive constants independent of f. If this condition is satisfied, the function can be recovered as

$$f(\mathbf{x}) = \sum_{i=1}^{N} \langle f, K_{\mathbf{x}_i}\rangle\ \tilde{K}_{\mathbf{x}_i}(\mathbf{x}) = \sum_{i=1}^{N} f_i\,\tilde{K}_{\mathbf{x}_i}(\mathbf{x}), \tag{1.8}$$

where $(\tilde{K}_{\mathbf{x}_i})_{i\in[1\ldots N]}$ is the dual frame. The dual frame, however, is difficult to find in general; hence, this way of recovering the function is often not practical.

An indirect way to recover the function using this frame formalism is to invert the frame operator defined by

$$Tf(\mathbf{x}) = f_1(\mathbf{x}) = \sum_{i=1}^{N} \langle f, K_{\mathbf{x}_i}\rangle\ K_{\mathbf{x}_i}(\mathbf{x}).$$

The inverse of the operator T can be expressed as

$$T^{-1} = \frac{2}{A+B} \sum_{i\in\mathbb{N}} \left(I - \frac{2\,T}{A+B}\right)^i.$$

Applying this inverse on $f_1(\mathbf{x})$ yields

$$\begin{aligned} f(\mathbf{x}) = T^{-1} f_1(\mathbf{x}) &= \frac{2}{A+B} \sum_{i\in\mathbb{N}} \left(I - \frac{2\,T}{A+B}\right)^i f_1(\mathbf{x}) \\ &= \lim_{m\to\infty} f_m(\mathbf{x}), \end{aligned}$$

where

$$f_m(\mathbf{x}) = \frac{2}{A+B} \sum_{i=0}^{m} \left(I - \frac{2\,T}{A+B} \right)^i f_1(\mathbf{x}).$$

This gives an iterative algorithm where the mth iterate satisfies the recursive relation

$$f_m = \frac{2}{A+B} f_1 + \left(I - \frac{2\,T}{A+B} \right) f_{m-1}.$$

The author [29] shows that the reconstruction can also be obtained as the least-squares fit to the data instead of interpolation, which is attractive in noisy situations.

1.3.6 Variational Methods

Variational methods provide more robust reconstructions than most other techniques, especially when the samples are sparsely distributed. This robustness is achieved by incorporating a so-called regularization. We provide here a brief overview. A more detailed discussion is provided in Chapter 2.

Given measurements $(f_i)_{i\in[1\ldots N]}$ at locations $(\mathbf{x}_i)_{i\in[1\ldots N]}$, the reconstruction problem is defined as the task of finding $f^{(\mathrm{opt})}$, which is a solution to the following constrained minimization problem:

$$\left\{ \begin{array}{rcl} f^{(\mathrm{opt})}(\mathbf{x}_i) & = & f_i,\ \forall i \in [1 \ldots N], \\ f^{(\mathrm{opt})}(\mathbf{x}) & = & \mathrm{argmin}_f\ J(f), \end{array} \right. \tag{1.9}$$

where $J(f)$ is the regularization functional that has the properties of a semi-norm. The regularization term takes care of the ill-posedness of the problem and gives a meaningful reconstruction. It typically penalizes the lack of smoothness. The presence of the regularization is crucial for dealing with large sampling gaps; it allows these to be filled-in in a smooth way using the information from the surrounding samples.

The most-widely used regularization functional belongs to Duchon's family of semi-norms [30]. A Duchon's semi-norm is parameterized by its order m and has a simple expression in the Fourier domain given by

$$J(f) = \int_{-\infty}^{\infty} \|\boldsymbol{\omega}\|^{2\,m}\ |\hat{f}(\boldsymbol{\omega})|^2\,\mathrm{d}\boldsymbol{\omega}. \tag{1.10}$$

These semi-norms are invariant to translation, scaling, and rotation of the function that is measured by them. As a consequence, the reconstruction is invariant with respect to such transformations of the data. In other words,

if we have two data sets that are related by a transformation composed of translation, rotation, and scaling, then the function reconstructed out of them will also be related by the same transformation.

The general solution to the above variational problem can be written as

$$S(\mathbf{x}) = \sum_{i=1}^{N} w_i \, \psi(\|\mathbf{x} - \mathbf{x}_i\|) + \sum_{i=k}^{Q} a_k \, p_k(\mathbf{x}). \tag{1.11}$$

It is made up of two terms. The first is a linear combination of so-called thin-plate splines which are radially symmetric and positioned at the sampling locations. The second term is a linear combination of bases for the polynomial space of maximum order $(m-1)$. Explicit formulæ for the thin-plate splines can found in Chapter 2 (Equation 2.8).

The optimal weights in (1.11) are determined by solving a system of linear equations. Specifically, the weights $\mathbf{w} = [\cdots w_i \cdots]^\top$ and $\mathbf{a} = [a_0 \cdots a_{p-1}]^\top$ are the solution of

$$\begin{bmatrix} \mathbf{A} & \mathbf{Q} \\ \mathbf{Q}^\top & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{w} \\ \mathbf{a} \end{bmatrix} = \begin{bmatrix} \mathbf{f} \\ \mathbf{0} \end{bmatrix}, \tag{1.12}$$

where $\{\mathbf{A}\}_{i,j} = \psi(\|\mathbf{x}_i - \mathbf{x}_j\|)$, $\mathbf{f} = [\cdots f_i \cdots]^T$, and $\{\mathbf{Q}\}_{i,j} = p_j(\mathbf{x}_i)$.

This variational method, which is also referred to as the thin-plate spline method, has become one of the preferred techniques for approximating images from their non-uniform samples [10]. Recent developments include methods adopted for space-time interpolations [31] and methods with mixed semi-norms [32].

Inspired from the above theory, various kinds of basis functions that are radially symmetric were introduced for the non-uniform sampling problem, which are collectively known as radial basis functions (RBFs) [33]. Methods that use such functions for non-uniform interpolation often approach the problem directly, and independently of any variational view point. The main ingredients of these methods are discussed in the next section.

1.3.7 Radial Basis Function (RBF) Methods

In the RBF method, one attempts to compute a function of the form

$$f(\mathbf{x}) = \sum_{i=1}^{N} w_i \, \phi(\mathbf{x} - \mathbf{x}_i) \tag{1.13}$$

that satisfies the interpolation conditions

$$f(\mathbf{x}_i) = f_i,\ \forall i \in [1 \dots N],$$

where $\phi(\mathbf{x})$ is some radial basis function. This problem is equivalent to solving the linear system

$$\mathbf{A}\,\mathbf{w} = \mathbf{f},$$

where $\{\mathbf{A}\}_{i,j} = \phi(\mathbf{x}_i - \mathbf{x}_j)$. Obviously, this problem has a unique solution iff the so-called *distance matrix* $\mathbf{A}$ is non-singular. Since $\mathbf{A}$ is symmetric, the non-singularity is equivalent to positive-definiteness, which is the condition that

$$\mathbf{w}^\top\,\mathbf{A}\,\mathbf{w} = \sum_{i=1}^{N}\sum_{j=1}^{N} w_i\,w_j\,\phi(\mathbf{x}_i - \mathbf{x}_j) > 0,\ \forall \mathbf{w} \in \mathbb{R}^N.$$

This leads to the definition of the so-called positive-definite function [33].

Definition 1 *A function ϕ is called a positive-definite function if, for every set of distinct points $(\mathbf{x}_i)_{i\in[1\dots N]}$, and for every set of real numbers $(w_i)_{i\in[1\dots N]}$, the quadratic form*

$$\sum_{i=1}^{N}\sum_{j=1}^{N} w_i\,w_j\,\phi(\mathbf{x}_i - \mathbf{x}_j)$$

is positive.

Hence, one chooses a positive-definite function to compute an interpolating function of the form (1.13). Examples of positive-definite functions include the Gaussian RBF $\phi(r) = e^{-c^2\,r^2}$, and the second-order inverse multiquadric function $\phi(r) = 1/\sqrt{r^2 + c^2}$, where $r = \|\mathbf{x}\|$.

This positive definiteness is often overly restrictive. It is not satisfied for many interesting functions, including thin-plate splines. Micchelli arrived at a more-general condition [34], which is the so-called conditionally positive definiteness.

Definition 2 *A function ϕ is called a conditionally positive-definite function of order m if, for every set of distinct points $(\mathbf{x}_i)_{i\in[1\dots N]}$, and for every set of real numbers $(w_i)_{i\in[1\dots N]}$ satisfying $\sum_{i=1}^{N} w_i\,p_j(\mathbf{x}_i) = 0$, $\forall j \in [1 \dots Q]$, the quadratic form*

$$\sum_{i=1}^{N}\sum_{j=1}^{N} w_i\,w_j\,\phi(\mathbf{x}_i - \mathbf{x}_j)$$

is positive, where $(p_j)_{j\in[1\dots Q]}$ *are the bases for the polynomial space of maximum order* m.

If a RBF ϕ is conditionally positive-definite of order m, then the function of the form

$$f(\mathbf{x}) = \sum_{i=1}^{N} w_i\, \phi(\mathbf{x} - \mathbf{x}_i) + \sum_{i=k}^{Q} a_k\, p_k(\mathbf{x}), \tag{1.14}$$

satisfying the interpolation conditions

$$f(\mathbf{x}_i) = f_i,\ \forall i \in [1 \dots N],$$

together with the orthogonality conditions

$$\sum_{i=1}^{N} w_i\, p_j(\mathbf{x}_i) = 0,\ \forall j \in [1 \dots Q],$$

is unique, provided that the sample points satisfy some mild conditions [34]. The weights that yield this unique solution are given by (1.12), where the sub-matrices involved are appropriately redefined.

In order to provide the condition on the sample locations $(\mathbf{x}_i)_{i\in[1\dots N]}$ that ensure the uniqueness of the solution, we first define a unisolvent set of points. A set of points $(\mathbf{x}_i)_{i\in[1\dots Q]}$ is called a unisolvent set in $(p_i)_{i\in[1\dots Q]}$ if, for every set of real numbers $(t_i)_{i\in[1\dots Q]}$, there exists one and only one function $p(\mathbf{x}) \in \text{span}(p_i)_{i\in[1\dots Q]}$ such that, $\forall i \in [1 \dots Q]$, $p(\mathbf{x}_i) = t_i$. Then, the condition that ensures the uniqueness of the solution is that the sample locations $(\mathbf{x}_i)_{i\in[1\dots N]}$ should contain a unisolvent set [34].

Functions that are not positive-definite, but conditionally positive-definite, include the thin-plate splines, multiquadrics, and inverse multiquadrics [35, 36, 37]. In [38, 39], the authors provide a variational interpretation for some functions other than the thin-plate spline such as multiquadrics or Gaussians. Of course, if one uses thin-plate splines, the resulting function corresponds to the variational problem introduced in the previous section. Further developments include some results on approximation properties [35, 36], preconditioning [40], fast evaluation methods [41, 42], approximate methods to handle large data sets [43, 44], compactly supported RBFs [45], and numerically well-conditioned RBFs [46].

1.3.8 Comparison of Methods

We compare now some important features of the methods described above. The features we consider are the following: (i) existence of an explicit expression for the interpolant; (ii) invariance to rotations; (iii) invariance to scaling; (iv) ability to handle large sampling gaps; (v) reconstruction quality; (vi) computational complexity; (vii) numerical conditioning, which refers to the sensitivity of computations; (viii) robustness to noise; (ix) cost of resampling; and (x) convergence rate. We provide an overview of these comparisons in Table 1.1.

All methods have an explicit expression of the solution, except the MLS and the mesh-based ones. The former specifies the solution implicitly in the form of a system of linear equations for each spatial location. The latter uses a triangular mesh to specify the solution as a sum of piecewise polynomials that are defined for each triangular region independently.

Next, we recall that a reconstruction method is regarded as invariant to some transformation group if the following holds: if we have two data sets that are related by a transformation, then the function reconstructed out of them will also be related by the same transformation. The specific invariances that we are interested in—scale- and rotation-invariances—are satisfied by all the methods, except the shift-variant space and RBF ones. The latter is not scale-invariant, whereas the former is neither scale-invariant nor rotationally invariant.

We examine now their ability to handle sampling gaps. Variational and RBF methods are the best ones in handling large sampling gap. The least-efficient ones are the global-fitting and shift-invariant space methods, since they require a minimum density to ensure the well-posedness of the problem. Mesh-based and MLS methods handle the gaps moderately well.

With respect to reconstruction quality, variational methods are usually believed to be the best. Global fitting and Shepard's method yield the worst quality.

All global methods, namely (i) global fitting, (ii) variational, and (iii) RBF methods, have the highest computational complexity, since they require solving a dense system of linear equations. Among the local ones, mesh-based methods also have a complexity that is comparable to the global ones, especially in the presence of regularization. Among the other methods, shift-

invariant space and MLS methods have the next-highest complexity. The former employs an iterative procedure, while the later specifies the interpolant for each point implicitly in terms of a system of linear equations. Modified Shepard's method has the least complexity.

Considering the numerical conditioning, we do not observe any specific problems with the local methods. On the other hand, the global methods, especially the variational and global-fitting ones, are badly conditioned. The RBF method is slightly better-conditioned than the other two, provided that an appropriate basis function is used.

Concerning the robustness, variational method is the most robust one, and the RBF method can be considered to be in the next level. Among the local methods, the ones which allow a least-square fit, namely the shift-invariant space and the MLS methods, can be robust. The other two—Shepard's and the mesh-based methods—do not place any particular emphasis on robustness.

The global methods obviously have the highest resampling complexity. The next highest level of complexity is attained by modified Shepard's and by mesh-based methods since they require evaluating multiple polynomials. The MLS method has an intermediate level of complexity, whereas the shift-invariant space methods have the lowest complexity, especially if one uses a compactly supported generating function.

1.4 Main Contributions of the Thesis

The focus of the thesis is on the design of variational methods for reconstructing scalar and vector functions. The main contributions fall under the following three headings:

Theory. Starting from general invariance principles, we propose to derive optimal methods for approximating non-uniform data. In particular, we provide a complete family of vector-regularization functionals that result in a reconstruction that is invariant to scaling and rotation of the input data. The proposed family is general enough to include all known functionals used for the vector problem so far. We provide the analytical solution for the projected sampling model using this family of functionals.

	Modified Shepard's	MLS	Mesh-based	Shift-inv.	Global fit	Varia-tional	RBF	
Explicit formula	Yes	No	No	Yes	Yes	Yes	Yes	
Rotational invariance	Yes	Yes	Yes	No	Yes	Yes	Yes	
Scale invariance	Yes	Yes	Yes	No	Yes	Yes	No	
Handling of gaps	–	+	+	−−	−−	++	+	
Quality	−−	+	–	+	−−	++	+	
Complexity	+	–	−−	–	−−	−−	−−	
Conditioning	++	++	++	++	−−	−−	–	
Robustness	−−	+	−−	+	−−	++	+	
Cost of resampling	–	+	–	++	−−	−−	−−	
Convergence	−−	−−	–	+	−−	++	+	

Table 1.1: Comparison of Non-uniform interpolation methods

Computationally Efficient Algorithms. The main drawback of the analytical methods in the variational formulation is their computational complexity. A practical contribution of this thesis is the development of some computationally efficient alternatives for the analytical methods. The minimization therein is carried out within a uniform B-spline space. We propose fast multiresolution algorithms for this formulation, which cut down the computational cost by several orders of magnitude.

Applications. First, we apply our scalar algorithm to image processing tasks such as reconstruction of images from sparse feature points, texture mapping, and more. Second, we demonstrate the potential of our vector algorithm in clinical echocardiography. Specifically, we show how the algorithm can be used to recover the full cardiac velocity field from pulsed-wave Doppler data, and we provide experimental results for real clinical data. As far as we know, this type of velocity-field reconstruction has never been attempted before. The results that we obtain are very promising.

1.5 Organization and Summary of the Thesis

Since the focus of the thesis is on variational methods, we provide an introductory chapter where we establish our notations and give an in-depth description of state-of-the-art methods in the field (Chapter 2). We then proceed with the innovative material of our research, which can be organized under two headings: scalar and vectorials.

1.5.1 Scalar Problem

In Chapter 3, we provide a computationally efficient alternative for Duchon's thin-plate spline method. We consider the approximation problem and we search for the minimizer of the approximation functional within the space of uniform B-splines. The step size for the B-spline space can be freely specified by the user; it allows a tradeoff between the computational complexity and the accuracy (closeness to the analytical solution). The minimization problem now reduces to finding an appropriate set of B-spline expansion coefficients. We show how the coefficients can be computed from a system of linear equations. The key point is that the linear system is well-conditioned, in contrast to the analytical method where the linear system involved is ill-conditioned. Further, the present system is sparse. We derive an algebraic relation that links together the systems of linear equations specifying reconstructions at different levels of resolution. We use this relation to develop a fast multigrid algorithm. We demonstrate the effectiveness of our approach on several image-reconstruction examples.

1.5.2 Vector Problem

In chapter 4, we consider the problem of reconstructing a vector field from non-uniform projected samples, where the term "projected sample" refers to the directional (scalar) component of the vector value at the sample location, as in the case of pulsed-wave Doppler imaging (axial velocity). Even though this model includes conventional sampling as well (full vector sampling), we are especially interested in situations where the data set contains only one scalar value at each sample location. We argue that, in order to recover the full vector field from such kind of data, the regularization functional should be rich enough to incorporate *a priori* physical constraints. It should also provide a realistic smoothing if necessary. To achieve this, it is necessary to design a family of functionals satisfying generally desirable properties.

To this end, we extend Duchon's notion of rotational invariance and scale invariance of semi-norms for vector fields, and find a complete family of functionals having these invariance properties. Notably, we show that the family is composed of a weighted sum of two distinct parts (sub-functionals): (i) Duchon's scalar semi-norm applied on the *divergence* field; (ii) the same applied to each component of the *rotational* field. If we further include the relative weight of these two parts, we obtain a three-parameter family of vector semi-norms. Our family is general enough to include all vector formulations that have been proposed so far.

We provide an analytical solution to the reconstruction problem above. We show that the solution can be expressed as a weighted sum of vector basis functions which, in turn, are derived from scalar thin-plate splines. We call these functions the generalized vector splines. The weights are obtained as a solution of a system of linear equations. Using numerical examples, we demonstrate the importance of tuning the parameters of the vector semi-norm according to the *a priori* knowledge of the underlying vector field. In particular, we show how irrotational and solenoidal solutions can be obtained as limit cases—with respect to relative weights of the sub-functionals— by choosing unequal orders for rotational and divergence sub-functionals.

In chapter 5, similar to the scalar case, we provide a numerically efficient alternative solution to the problem above, where the minimization is carried out within a uniform B-spline space. We explicitly construct the system of linear equations that yields the B-spline expansion coefficients. We

also provide a fast multiresolution algorithm. We demonstrate the numerical advantages of this alternative method.

Finally, we apply our method to recover the cardiac motion field from ultrasound pulsed-wave Doppler data. We first provide quantitative validations on synthetic and real phantom data, and then give reconstruction examples for clinical data.

Chapter 2

Review of Variational Reconstruction Methods

Among the various reconstruction methods available, we have singled out the variational ones because they have a wider range of applicabilty (e.g. sparse and noisy samples), and also because they have attractive invariance properties. These methods were briefly reviewed for the scalar problem in Section 1.3.6. In this Chapter, we give a more detailed description of the techniques currently available including some specific ones devoted to the vector problem.

Notations and Conventions

- We represent vectors by bold faced lower case letters and matrices by bold faced upper case letters. For example, $\mathbf{x}$ is a column vector with its ith element given by $\{\mathbf{x}\}_i = x_i$ and $\mathbf{X}$ is a matrix with the elements $\{\mathbf{X}\}_{ij} = x_{ij}$. $\mathbf{e}_i$ with $\{\mathbf{e}_i\}_j = \delta(i-j)$ is the ith standard basis vector; in other words, we have $\mathbf{x} = \sum_{i=1}^{n} x_i \mathbf{e}_i$.

- $\mathbf{x} = [x_1 \cdots x_n]^T$ usually denotes a spatial location. The notations $[x\, y]^T$ and $[x\, y\, z]^T$ are sometimes used to represent 2D and 3D spatial locations, respectively.

- $|\boldsymbol{\mu}|$ represent the sum of its elements; i.e., $\sum_{i=1}^{n} \mu_i$. $\boldsymbol{\mu}!$ is the product of the factorials of its elements; i.e., $\prod_{i=1}^{n} \mu_i!$.

- We work with functions $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^n, n = 2, 3$ that represent vector fields. The scalar product of two vector functions is defined as $\langle \mathbf{f}, \mathbf{g} \rangle = \int_{\mathbb{R}^n} \mathbf{f}^T(\mathbf{x})\mathbf{g}(\mathbf{x})d\mathbf{x}$. By extension, the notation $\langle \mathbf{X}, \mathbf{f} \rangle$ applied on the matrix $\mathbf{X}$ and a vector $\mathbf{f}$ is a vector of scalar products between the columns of $\mathbf{X}$ and $\mathbf{f}$. In other words, $\mathbf{y} = \langle \mathbf{X}, \mathbf{f} \rangle \iff y_i = \sum_j \langle x_{ij}, f_j \rangle$.

- We deal with convolution of matrices and vectors. We define it by following the same rules of matrix multiplication. Examples: (a) $\mathbf{f}^T * \mathbf{g} = \sum_j f_j * g_j$; (b) $\mathbf{y}^T * \mathbf{x} = \mathbf{Z}$ means $y_i * x_j = z_{ij}$; (c) $\mathbf{Z} * \mathbf{x} = \mathbf{y}$ means $\sum_j z_{ij} * f_j = y_i$

- The Dirac distribution δ is defined as $\langle \delta, f \rangle = f(0)$, for every continuous function f. Convolving a function with a Dirac yields the same function, i.e., $\delta * f = f$. The derivative of the Dirac δ' is defined as $\langle \delta', f \rangle = -f'(0)$. Consequently, $\delta' * f = f'$. More generally, any partial derivative, for example $\frac{\partial^2 f}{\partial x_i \partial x_j}$, can be expressed as $\left(\frac{\partial^2 \delta}{\partial x_i \partial x_j} \right) * f$.

- $\triangle f$ denotes the Laplacian of f, which is given by $\sum_{i=1}^n \frac{\partial^2 f}{\partial x_i^2}$. It can be represented in the form of a convolution as $L(\mathbf{x}) * f(\mathbf{x})$, where $L(\mathbf{x}) = \sum_{i=1}^n \frac{\partial^2 \delta}{\partial x_i^2}$.

- $\partial^{\boldsymbol{\mu}} f$ denotes $\frac{\partial^{|\boldsymbol{\mu}|}}{\partial x_1^{\mu_1} \cdots \partial x_n^{\mu_n}} f$. $\mathbf{h}_m f$ is the vector of all partial derivative of the form $\sqrt{\frac{m!}{\boldsymbol{\mu}!}} \partial^{\boldsymbol{\mu}} f$ such that $|\boldsymbol{\mu}| = m$. $\mathbf{h}_m$ is Duchon's operator.

- $\mathbb{P}_m(\mathbb{R}^n)$ is the space of scalar polynomials of order m. $\mathbb{P}_m(\mathbb{R}^n; \mathbb{R}^n)$ is the space of vector functions whose components are in $\mathbb{P}_m(\mathbb{R}^n)$.

- We will denote $\hat{\mathbf{f}}$, $\mathcal{F}\mathbf{f}$, as the Fourier transform of $\mathbf{f}$, $\hat{\mathbf{f}}(\boldsymbol{w}) = \int_{\mathbb{R}^n} \mathbf{f}(\mathbf{x}) e^{-i\boldsymbol{\omega}^T \mathbf{x}} d\mathbf{x}$. $\hat{\mathbf{f}}^\dagger(\boldsymbol{\omega})$ denotes the Hermitian conjugate transpose of $\hat{\mathbf{f}}(\boldsymbol{\omega})$; i.e., $\hat{\mathbf{f}}^\dagger(\boldsymbol{\omega}) = \left(f^T(\boldsymbol{\omega}) \right)^*$.

- The operator $\nabla = [\frac{\partial}{\partial x_1} \cdots \frac{\partial}{\partial x_n}]^T$ plays an important role in our analysis. When operated on a scalar function f, it yields its gradient given by ∇f. Its scalar product and the vector product with a vector function yields its divergence and the curl respectively. In other words, $\operatorname{div} \mathbf{f} = \nabla \cdot \mathbf{f} = \sum_{i=1}^n \frac{\partial f_i}{\partial x_i}$.

$$\operatorname{rot} \mathbf{f} = \nabla \times \mathbf{f} = -\frac{\partial f_1}{\partial x_2} + \frac{\partial f_2}{\partial x_1},$$

in 2D, and is defined as

$$\operatorname{rot}\mathbf{f} = \nabla \times \mathbf{f} = \begin{bmatrix} -\frac{\partial f_2}{\partial x_3} + \frac{\partial f_3}{\partial x_2} \\ -\frac{\partial f_1}{\partial x_3} + \frac{\partial f_3}{\partial x_1} \\ -\frac{\partial f_1}{\partial x_2} + \frac{\partial f_2}{\partial x_1} \end{bmatrix},$$

in 3D.

- $\partial_{ij} f$ is the short hand notation for $\frac{\partial^2}{\partial x_1 \partial x_2} f$.
- $D^{-m}L^2(\mathbb{R}^n)$ denotes the space of scalar functions whose partial derivatives of order m are in $L^2(\mathbb{R}^n)$. $D^{-m}L^2(\mathbb{R}^n;\mathbb{R}^n)$ is the space of vector functions whose components are in $D^{-m}L^2(\mathbb{R}^n)$. $D^{-m}L^2(\mathbb{R}^n)$ is known as the Beppo-Levi space. $\mathbb{W}^{m,2}(\mathbb{R}^n)$ is the Sobolev space of order m, which is the intersection of $D^{-m}L^2(\mathbb{R}^n)$ and $L^2(\mathbb{R}^n)$. $\mathbb{W}^{m,2}(\mathbb{R}^n;\mathbb{R}^n)$ is the corresponding space of vector functions.

2.1 Variational Reconstruction of Scalar Functions

2.1.1 The Scalar Minimization Problem

We recall the basic variational interpolation problem, which is to determine the continuously-defined function $f(\mathbf{x})$ that minimizes a suitable regularization functional $J(f)$ subject to the interpolation constraints, $f(\mathbf{x}_i) = f_i, \ \forall i \in [1:N]$.

In practice, the measurements are often noisy or lacking precision, and it is not necessarily desirable to reconstruct a function $f(\mathbf{x})$ that interpolates the data f_i exactly. One therefore relaxes the interpolation constraint, trading some closeness of fit against more smoothness on the solution [30, 47]. In that case, the reconstruction task is formulated as the following minimization problem:

$$f^{(\mathrm{opt})}(\mathbf{x}) = \underset{f}{\operatorname{argmin}} \left(\sum_{i=1}^{N} (f(\mathbf{x}_i) - f_i)^2 + \lambda J(f) \right) \tag{2.1}$$

where λ is the so-called regularization parameter that works as a trade-off factor. It is set up such as to find a compromise between fitting the data well and penalizing reconstructions that are not smooth enough. Note that, in this second version, we have an approximation problem rather than an

interpolation. Interestingly, we can get back to the first case by selecting λ to be arbitrarily small (but non-zero).

2.1.2 The Regularization Functional

The most widely used regularization functionals are the Duchon semi-norms. The Fourier domain representation of Duchon semi-norm of order m (integer) is

$$\mathcal{D}_m(f) = \int \|\boldsymbol{\omega}\|^{2m} |\hat{f}(\boldsymbol{\omega})|^2 d\boldsymbol{\omega} \tag{2.2}$$

It should be noted that this expression is valid only for functions that belongs to the Sobolev space of order m, $\mathbb{W}^{m,2}(\mathbb{R}^n)$, since, only for these functions, the Fourier transform of their partial derivatives of order m exist.

In the spatial domain, this semi-norm reads

$$J(f) = \mathcal{D}_m(f) = \int \|\mathbf{h}_m f(\mathbf{x})\|^2 d\mathbf{x}, \tag{2.3}$$

where $\mathbf{h}_m$ is the vector of all partial derivative operators of the form $\sqrt{\frac{m!}{\boldsymbol{\mu}!}}\partial^{\boldsymbol{\mu}}$ such that $|\boldsymbol{\mu}| = m$.

Table 2.1 gives the expression for the first lower order instances of $\mathbf{h}_m$. In order to make the reconstruction problem well-defined, one needs to impose the constraint $m > n/2$. Note that when $m = 1$, the functional is equal to the L_2 norm of the gradient. In other words,

$$\begin{aligned} \mathcal{D}_1(f) &= \int \|\mathbf{h}_1 f(\mathbf{x})\|^2 d\mathbf{x} \\ &= \int \|\nabla f(\mathbf{x})\|^2 d\mathbf{x} \end{aligned}$$

Further, it can be shown that, when f is in $\mathbb{W}^{m,2}(\mathbb{R}^n)$, $\mathcal{D}_2(f)$ is equal to the L_2 norm of the Laplacian, i.e.,

$$\mathcal{D}_2(f) = \int (\triangle f(\mathbf{x}))^2 d\mathbf{x}.$$

This can be easily verified from the Fourier domain expression.

As mentioned before, an attractive property of the above class of semi-norms is that they are invariant to translation, scaling, and rotations. In order to state these properties formally, we first define the following transformations:

$$f_{\boldsymbol{\Omega}}(\mathbf{x}) = f(\boldsymbol{\Omega}^T \mathbf{x}),$$

Parameters	$\mathbf{h}_m$
$n=2, m=1$	$\left[\frac{\partial}{\partial x_1} \ \frac{\partial}{\partial x_2}\right]^T$
$n=2, m=2$	$\left[\frac{\partial^2}{\partial x_1^2} \ \frac{\partial^2}{\partial x_2^2} \ \sqrt{2}\frac{\partial^2}{\partial x_1 \partial x_2}\right]^T$
$n=3, m=1$	$\left[\frac{\partial}{\partial x_1} \ \frac{\partial}{\partial x_2} \ \frac{\partial}{\partial x_3}\right]^T$
$n=3, m=2$	$\left[\frac{\partial^2}{\partial x_1^2} \ \frac{\partial^2}{\partial x_2^2} \ \frac{\partial^2}{\partial x_3^2} \ \sqrt{2}\frac{\partial^2}{\partial x_1 \partial x_2} \ \sqrt{2}\frac{\partial^2}{\partial x_2 \partial x_3} \sqrt{2}\frac{\partial^2}{\partial x_1 \partial x_3}\right]^T$

Table 2.1: Operators for Duchon's semi-norms

$$f_a(\mathbf{x}) = f(a\mathbf{x}),$$

where $\mathbf{\Omega}$ is an orthogonal matrix and a is real positive number. Then the invariances are expressed as

$$\text{Translation invariance:} \qquad \mathcal{D}_m(f(\cdot - \mathbf{t})) = \mathcal{D}_m(f), \tag{2.4}$$

$$\text{Rotational invariance:} \qquad \mathcal{D}_m(f_{\mathbf{\Omega}}) = \mathcal{D}_m(f), \tag{2.5}$$

$$\text{Scale invariance:} \qquad \mathcal{D}_m(f_a) = \frac{1}{a^{2m}}\mathcal{D}_m(f). \tag{2.6}$$

These properties can be readily verified from (2.2).

The implication of the rotational invariance of J is that the reconstruction is rotationally invariant: if f is the function reconstructed from the data $\{\mathbf{x}_i, f_i\}$, then the function reconstructed from $\{\mathbf{\Omega}\mathbf{x}_i, f_i\}$ is given by $f_{\mathbf{\Omega}}$, where $\mathbf{\Omega}$ is an orthogonal matrix. The implication of scale invariance is similar with a slight modification: if f is the function reconstructed from the data $\{\mathbf{x}_i, f_i\}$, then the function reconstructed from $\{a\mathbf{x}_i, f_i\}$ is given by f_a, provided that, in the case of approximation, the regularization parameter is multiplied by a^{2m}. In summary, it can be stated that the reconstruction operation commutes with geometric transformations if the regularization functional J is invariant to them.

The kernel $\mathcal{K}$ of the semi-norm J is the space of functions such that $\forall f \in \mathcal{K}, J(f) = 0$. For Duchon's semi-norm $\mathcal{D}_m$, the kernel is the space of functions such that $\mathbf{h}_m f$ is zero, which is equal to the space of polynomials of degree $m-1$, $\mathbb{P}_{m-1}(\mathbb{R}^n)$.

2.1.3 The Solution

The solution for the interpolation problem can be written as (cf. [30])

$$S(\mathbf{x}) = \sum_{i=1}^{N} w_i \psi(||\mathbf{x} - \mathbf{x}_i||) + \sum_{i=k}^{Q} a_k p_k(\mathbf{x}), \tag{2.7}$$

where w_i's and a_i's are a suitable set of coefficients. Here, $(p_k(\mathbf{x}))_{k\in[1:Q]}$ are the bases for the kernel of $\mathcal{D}_m$, which is $\mathbb{P}_{m-1}(\mathbb{R}^n)$, and the thin-plate splines basis functions are given by

$$\psi(r) = \begin{cases} r^{2m-n} \log r, & \text{if } 2m-n \text{ is even,} \\ r^{2m-n}, & \text{if } 2m-n \text{ is odd.} \end{cases} \tag{2.8}$$

The optimal weights for the interpolation problem satisfy

$$\begin{bmatrix} \mathbf{A} & \mathbf{Q} \\ \mathbf{Q}^T & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{w} \\ \mathbf{a} \end{bmatrix} = \begin{bmatrix} \mathbf{f} \\ \mathbf{0} \end{bmatrix},$$

where $\{\mathbf{A}\}_{i,j} = \psi(||\mathbf{x}_i - \mathbf{x}_j||)$, and $\{\mathbf{Q}\}_{i,j} = p_j(\mathbf{x}_i)$.

The solution to the approximation problem is of the same form given by (2.7), but now the optimal weights should satisfy

$$\begin{bmatrix} \mathbf{B} + \lambda\mathbf{I} & \mathbf{Q} \\ \mathbf{Q}^T & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{w} \\ \mathbf{a} \end{bmatrix} = \begin{bmatrix} \mathbf{f} \\ \mathbf{0} \end{bmatrix}.$$

It is interesting to note that choosing $\lambda \to 0$ in the approximation solution yields the interpolating spline.

Thin-plate splines, which are radially symmetric, are the earliest examples of radial basis functions (RBFs). Other RBFs were introduced later and the study of such functions is still an active area of research in computational mathematics [33].

Since thin-plate splines do not have a finite support, the matrices in the above system of equation are dense. Also, since the magnitude of the functions grow with distance from the center, the matrices are poorly conditioned. This makes thin-plate splines impractical and numerically unstable when the number of samples is large. Despite the solutions that have been proposed for reducing this complexity and improving the numerical behaviour [41], their numerical efficiency is still insufficient. Note that one can use alternative RBFs that have better numerical behavior [46, 38, 39], but their approximation power is inferior to that of the thin-plate splines.

2.1.4 The Extended Semi-norm

The semi-norm given in (2.3) is actually a simplified form of the semi-norm that was actually used by Duchon. The later is given by

$$\mathcal{D}_{m,\mathfrak{s}}(f) = \int \|\boldsymbol{\omega}\|^{2\mathfrak{s}} \|\mathcal{F}\,\mathbf{h}_m f\|^2 \, d\boldsymbol{\omega} \tag{2.9}$$

where $\mathfrak{s}$ is a real number. In order to make the reconstruction problem well-defined, the following restrictions are imposed on m and $\mathfrak{s}$.

$$\begin{aligned} m + \mathfrak{s} &> n/2 \\ -m - n/2 &< \mathfrak{s} < n/2 \end{aligned}$$

To obtain the reconstruction with the above semi-norm, one needs to replace (2.8) by the following equation for the definition of the thin-plate spline:

$$\psi(r) = \begin{cases} r^{2(m+\mathfrak{s})-n} \log r, & \text{if } 2(m+\mathfrak{s}) - n \text{ is an even integer,} \\ r^{2(m+\mathfrak{s})-n}, & \text{otherwise.} \end{cases} \tag{2.10}$$

2.2 Variational Reconstruction of Vector Functions

Several authors have proposed using variational methods for the reconstruction of vector functions. Most of the work published in this area is concerned with interpolation to the complete samples, i.e., samples with full vector values. Note that the reconstruction problem can be considered as a set of n scalar reconstruction tasks, even though it may not be optimal to do so. The landmark interpolation problem is indeed usually addressed in this way [48]. The interpolation is carried out for each component independently using Duchon's thin-plate splines. Surprisingly, even though this approach does not take into account the vectorial nature of the problem, it has been widely used by many researchers.

In the context of reconstructing wind velocity field from full vector samples, it has been pointed out by researchers that treating the components separately leads to unrealistic fluctuations in the derived divergence and rotational fields. To get more realistic solutions, it has therefore been suggested that the plausibility criterion should be based on some smoothing operators on divergence and rotational fields [49].

In this section, we provide a brief overview of methods that use semi-norms based on divergence and rotational fields. For all such methods, the general statement of the problem is as follows: given vector samples $(\mathbf{s}_i)_{i\in[1:N]}$ at locations $(\mathbf{x}_i)_{i\in[1:N]}$, the reconstruction problem is the task of finding a function $\mathbf{f}^{(\text{opt})}$ such that

$$\begin{aligned} \mathbf{f}^{(\text{opt})}(\mathbf{x}_i) &= \mathbf{s}_i, \forall\, i \in [1:N], \quad \text{and } \mathbf{f}^{(\text{opt})} \in \mathcal{V}, \\ \mathbf{f}^{(\text{opt})}(\mathbf{x}) &= \underset{\mathbf{f}}{\operatorname{argmin}}\, J_{\text{div,rot}}(\mathbf{f}), \end{aligned} \tag{2.11}$$

where $\mathcal{V}$ is an appropriate space of vector functions, and $J_{\text{div,rot}}(\mathbf{f})$ is the vectorial semi-norm.

Let $\mathcal{K}$ be the kernel of $J_{\text{div,rot}}(\mathbf{f})$. Then, the part of $\mathcal{K}$ that lies inside $\mathcal{V}$ is called the reconstructing kernel $\mathcal{K}_f$, which is given by $\mathcal{K}_f = \mathcal{K} \cap \mathcal{V}$. The elements from $\mathcal{K}_f$ are useful because they can be included into the reconstructing function as much as needed without contributing to the cost functional. On the other hand, $\mathcal{K}_f$ should not be too large. Otherwise, the problem may not have a unique solution.

To ensure the uniqueness of the solution, the sample points $(\mathbf{x}_i)_{i\in[1:N]}$ are assumed to contain a unisolvent set in $\mathcal{K}_f$ [49, 50, 51].

2.2.1 The 2D Problem

Amodei and Benbourhim in [49] address the problem of reconstructing 2D wind velocity fields. They use the following semi-norm:

$$J_{\text{div,rot}}(\mathbf{f}) = \alpha\mathcal{D}_1(\operatorname{div}\mathbf{f}) + \beta\mathcal{D}_1(\operatorname{rot}\mathbf{f}) \tag{2.12}$$

for some non-negative real numbers α and β. The minimization space is the second order Beppo-Levi space; i.e., $\mathcal{V} = D^{-2}L^2(\mathbb{R}^2;\mathbb{R}^2)$. The reconstructing kernel $\mathcal{K}_f$ in this case is equal to $\mathbb{P}_1(\mathbb{R}^2;\mathbb{R}^2)$. Note that the basis for $\mathbb{P}_1(\mathbb{R}^2)$ is $\{1, x, y\}$ and hence the basis for $\mathbb{P}_1(\mathbb{R}^2;\mathbb{R}^2)$ is $\{\binom{1}{0}, \binom{0}{1}, \binom{x}{0}, \binom{0}{x}, \binom{y}{0}, \binom{0}{y}\}$.

Let $(\mathbf{x}_i = [x_i\ y_i]^T)_{i\in[1:N]}$ be sample locations and let $(\mathbf{s}_i = [u_i\ v_i]^T)_{i\in[1:N]}$ be the sample values. Under the assumption that the sample locations contain a unisolvent set in $\mathbb{P}_1(\mathbb{R}^2;\mathbb{R}^2)$, Amodei and Benbourhim have shown that

the problem has a unique solution, which is given by

$$\begin{aligned}
\{\mathbf{f}^{(\mathrm{opt})}(\mathbf{x})\}_1 &= \sum_{i=1}^{N} a_i \left(\frac{1}{\alpha}\psi_{11}(\mathbf{x}-\mathbf{x}_i) + \frac{1}{\beta}\psi_{22}(\mathbf{x}-\mathbf{x}_i) \right) \\
&\quad + \sum_{i=1}^{N} b_i \left(\frac{1}{\alpha} - \frac{1}{\beta} \right) \psi_{12}(\mathbf{x}-\mathbf{x}_i) + c_1 + c_2 x + c_3 y, \\
\{\mathbf{f}^{(\mathrm{opt})}(\mathbf{x})\}_2 &= \sum_{i=1}^{N} b_i \left(\frac{1}{\alpha}\psi_{22}(\mathbf{x}-\mathbf{x}_i) + \frac{1}{\beta}\psi_{11}(\mathbf{x}-\mathbf{x}_i) \right) \\
&\quad + \sum_{i=1}^{N} a_i \left(\frac{1}{\alpha} - \frac{1}{\beta} \right) \psi_{12}(\mathbf{x}-\mathbf{x}_i) + d_1 + d_2 x + d_3 y,
\end{aligned}$$

where

$$\psi_{ij}(\mathbf{x}) = \frac{\partial^2}{\partial x_i \partial x_j} \|\mathbf{x}\|^4 \log \|\mathbf{x}\| .$$

The coefficients are obtained by solving a $(2N+6)\times(2N+6)$ linear system. To specify the solution explicitly, we first define the following:

$$\begin{aligned}
\{\mathbf{A}_d\}_{2i-2:2i,2j-2:2j} &= \begin{bmatrix} \psi_{11}(\mathbf{x}_i-\mathbf{x}_j) & \psi_{12}(\mathbf{x}_i-\mathbf{x}_j) \\ \psi_{12}(\mathbf{x}_i-\mathbf{x}_j) & \psi_{22}(\mathbf{x}_i-\mathbf{x}_j) \end{bmatrix} \\
\{\mathbf{A}_c\}_{2i-2:2i,2j-2:2j} &= \begin{bmatrix} \psi_{22}(\mathbf{x}_i-\mathbf{x}_j) & -\psi_{12}(\mathbf{x}_i-\mathbf{x}_j) \\ -\psi_{12}(\mathbf{x}_i-\mathbf{x}_j) & \psi_{11}(\mathbf{x}_i-\mathbf{x}_j) \end{bmatrix} \\
\mathbf{Q} &= \begin{bmatrix} 1 & x_1 & y_1 \\ \vdots & \vdots & \vdots \\ 1 & x_2 & y_2 \end{bmatrix} \\
\mathbf{P} &= \begin{bmatrix} \mathbf{Q} & \mathbf{0} \\ \mathbf{0} & \mathbf{Q} \end{bmatrix} \\
\mathbf{a} &= [a_1 \cdots a_N]^T \\
\mathbf{b} &= [b_1 \cdots b_N]^T \\
\mathbf{u} &= [u_1 \cdots u_N]^T \\
\mathbf{v} &= [v_1 \cdots v_N]^T \\
\mathbf{c} &= [c_1\, c_2\, c_3]^T \\
\mathbf{d} &= [d_1\, d_2\, d_3]^T
\end{aligned}$$

The linear system that yields the required coefficients is then given by

$$\left(\frac{1}{\lambda_d}\mathbf{A}_d + \frac{1}{\lambda_c}\mathbf{A}_c\right)\begin{bmatrix}\mathbf{a}\\ \mathbf{b}\end{bmatrix} + \mathbf{P}\begin{bmatrix}\mathbf{c}\\ \mathbf{d}\end{bmatrix} = \begin{bmatrix}\mathbf{u}\\ \mathbf{v}\end{bmatrix},$$
$$\mathbf{P}^T\begin{bmatrix}\mathbf{a}\\ \mathbf{b}\end{bmatrix} = \mathbf{0}.$$

Let $\mathbf{f}_{ei}^{(\mathrm{opt})}(\mathbf{x})$ be the solution obtained by solving the above problem with $\beta \to \infty$. Then, the authors show that $\mathbf{f}_{ei}^{(\mathrm{opt})}(\mathbf{x})$ is the same as the solution to the modified problem, where the semi-norm is $J_{\mathrm{div}}(\mathbf{f}) = \mathcal{D}_1(\mathrm{div}\,\mathbf{f})$ and the reconstruction space is $\mathcal{V} := \{\mathbf{f} \in D^{-2}L^2(\mathbb{R}^2;\mathbb{R}^2); \mathcal{D}_1(\mathrm{rot}\,\mathbf{f}) = 0\}$. The solution $\mathbf{f}_{ei}^{(\mathrm{opt})}(\mathbf{x})$ is essentially irrotational. It is not exactly irrotational; it may contain a rigid rotations of the form $[-y\ x]^T$, since $\mathcal{D}_1(\mathrm{rot}\,\mathbf{f}) = 0$ is satisfied for rigid forms of rotation. Analogously, let $\mathbf{f}_{es}^{(\mathrm{opt})}(\mathbf{x})$ be the reconstruction obtained by solving the original problem with $\alpha \to \infty$. Then the authors show that it is the same as the solution to the modified problem where the semi-norm is $J_{\mathrm{rot}}(\mathbf{f}) = \mathcal{D}_1(\mathrm{rot}\,\mathbf{f})$ and reconstruction space is $\mathcal{V} := \{\mathbf{f} \in D^{-2}L^2(\mathbb{R}^2;\mathbb{R}^2); \mathcal{D}_1(\mathrm{div}\,\mathbf{f}) = 0\}$. $\mathbf{f}_{es}^{(\mathrm{opt})}(\mathbf{x})$ is essentially solenoidal. It is not exactly solenoidal for it may contain a velocity component of the form $[x\,y]^T$, which satisfies $\mathcal{D}_1(\mathrm{div}\,\mathbf{f}) = 0$.

2.2.2 The 3D Problem

The 3D reconstruction problem has been solved by Dodu and Rabut in [50]. These authors also use a more general semi-norm, which is given by

$$J_{\mathrm{div,rot,m}}(\mathbf{f}) = \rho\mathcal{D}_{m-1}(\mathrm{div}\,\mathbf{f}) + \sum_{i=1}^{3}\mathcal{D}_{m-1}\left(\{\mathrm{rot}\,\mathbf{f}\}_i\right), \tag{2.13}$$

and the reconstruction space is $\mathcal{V} = D^{-m}L^2(\mathbb{R}^3;\mathbb{R}^3)$. The reconstructing kernel is $\mathbb{P}_{m-1}(\mathbb{R}^3;\mathbb{R}^3)$.

Using the theory of abstract splines [52], the authors prove that a unique solution exists provided that the sample locations contains a unisolvent set. Similar to the previous case, the solution is expressed as weighted sum of basis functions located at the sample points. In oder to provide the expression for the solution, we first define the following $n \times n$ matrix basis function:

$$\mathbf{\Psi} = \frac{1}{\alpha}\mathbf{\Psi}_d + \frac{1}{\beta}\mathbf{\Psi}_c \tag{2.14}$$

where

$$\begin{aligned}
\mathbf{\Psi}_d &= (-1)^m \begin{bmatrix} \partial_{11}^2\psi & \partial_{12}^2\psi & \partial_{13}^2\psi \\ \partial_{12}^2\psi & \partial_{22}^2\psi & \partial_{23}^2\psi \\ \partial_{13}^2\psi & \partial_{23}^2\psi & \partial_{33}^2\psi \end{bmatrix} \\
\mathbf{\Psi}_c &= (-1)^m \begin{bmatrix} (\partial_{22}^2+\partial_{33}^2)\psi & -\partial_{12}^2\psi & -\partial_{13}^2\psi \\ -\partial_{12}^2\psi & (\partial_{11}^2+\partial_{33}^2)\psi & -\partial_{23}^2\psi \\ -\partial_{13}^2\psi & -\partial_{23}^2\psi & (\partial_{11}^2+\partial_{22}^2)\psi \end{bmatrix}
\end{aligned}$$

with ψ being defined by

$$\psi(\mathbf{x}) = \frac{-1}{4\pi(2m)!}\|\mathbf{x}\|^{2m-1}.$$

The solution is given by

$$\mathbf{f}^{(\mathrm{opt})}(\mathbf{x}) = \sum_{i=1}^{N} \Phi(\mathbf{x}-\mathbf{x}_i)\mathbf{a}_i + \sum_{k=1}^{Q} b_i \mathbf{p}_k(\mathbf{x}) \tag{2.15}$$

where $(\mathbf{p}_k)_{k\in[1:Q]}$ is basis for $\mathbb{P}_{m-1}(\mathbb{R}^3;\mathbb{R}^3)$, and $(\mathbf{a}_i \in \mathbb{R}^3)_{i\in[1:N]}$ and $(b_i)_{i\in[1:Q]}$ are some vector and scalar coefficients respectively. The coefficients are obtained by solving a $(3N+Q)\times(3N+Q)$ linear system of equations. To specify the system, we define the following:

$$\begin{aligned}
\{\mathbf{A}\}_{3(i-1)+k,3(j-1)+l} &= \{\mathbf{\Psi}(\mathbf{x}_i-\mathbf{x}_j)\}_{k,l}, \\
&\quad \forall\, i,j\in[1:N], \quad \forall\, k,l\in[1:3]. \\
\{\mathbf{Q}\}_{3(i-1)+k,j} &= \{\mathbf{p}_j(\mathbf{x}_i)\}_k, \\
&\quad \forall i\in[1:N], \quad \forall j\in[1:Q], \quad \forall k\in[1:3]. \\
\{\mathbf{a}\}_{3(i-1)+k} &= \{\mathbf{a}_i\}_k, \\
&\quad \forall i\in[1:N], \quad \forall k\in[1:3]. \\
\{\mathbf{s}\}_{3(i-1)+k} &= \{\mathbf{s}_i\}_k, \\
&\quad \forall i\in[1:N], \quad \forall k\in[1:3]. \\
\{\mathbf{b}\}_i &= b_i, \\
&\quad \forall i\in[1:Q].
\end{aligned}$$

The linear system is now given by

$$\begin{bmatrix} \mathbf{A} & \mathbf{Q} \\ \mathbf{Q}^T & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix} = \begin{bmatrix} \mathbf{s} \\ \mathbf{0} \end{bmatrix} \tag{2.16}$$

Here too, as in the case of 2D, the limit cases $\rho \to \infty$ and $\rho \to 0$ yield essentially divergence-free and rotation-free solutions respectively—especially when m is small. If it is required to get a true divergence-free or rotation-free solution, one needs to choose $m = 1$, which poses a technical problem: the elements of $D^{-1}L^2(\mathbb{R}^3;\mathbb{R}^3)$ are not continuous.

This divergence-free (solenoidal) and rotation-free (irrotational) interpolation problem is treated separately by the same authors in [51]. To this end, the authors consider the irrotational and solenoidal subspaces of $D^{-m}L^2(\mathbb{R}^3;\mathbb{R}^3)$, which are defined by

$$\begin{aligned} \mathcal{V}_{m,irr} &= \{\mathbf{f} \,|\, \mathbf{f} \in D^{-m}L^2(\mathbb{R}^3;\mathbb{R}^3); \operatorname{rot}\mathbf{f} = \mathbf{0}\} && (2.17)\\ \mathcal{V}_{m,sol} &= \{\mathbf{f} \,|\, \mathbf{f} \in D^{-m}L^2(\mathbb{R}^3;\mathbb{R}^3); \operatorname{div}\mathbf{f} = 0\}, && (2.18) \end{aligned}$$

Now, the solenoidal and irrotational interpolation solutions are defined by

$$\begin{aligned} \mathbf{f}_{\text{irr}}^{(\text{opt})}(\mathbf{x}_i) = \mathbf{s}_i, \forall\, i \in [1:N], \quad \text{and } \mathbf{f}_{\text{irr}}^{(\text{opt})} \in \mathcal{V}_{m,irr},\\ \mathbf{f}_{\text{irr}}^{(\text{opt})} = \operatorname*{argmin}_{\mathbf{f}} \mathcal{D}_{m-1}(\operatorname{div}\mathbf{f}). \end{aligned} \tag{2.19}$$

$$\begin{aligned} \mathbf{f}_{\text{sol}}^{(\text{opt})}(\mathbf{x}_i) = \mathbf{s}_i, \forall\, i \in [1:N], \quad \text{and } \mathbf{f}_{\text{sol}}^{(\text{opt})} \in \mathcal{V}_{m,sol},\\ \mathbf{f}_{\text{sol}}^{(\text{opt})} = \operatorname*{argmin}_{\mathbf{f}} \sum_{i=1}^{3} \mathcal{D}_{m-1}(\{\operatorname{rot}\mathbf{f}\}_i). \end{aligned} \tag{2.20}$$

Let $\mathcal{K}_{\text{irr}}$ and $\mathcal{K}_{\text{sol}}$ be the reconstructing kernel for irrotational and solenoidal problems respectively, which are defined by

$$\begin{aligned} \mathcal{K}_{\text{irr}} &= \{\mathbf{f} \,|\, \mathcal{D}_{m-1}(\operatorname{div}\mathbf{f}) = 0; \mathbf{f} \in \mathcal{V}_{m,irr}\}\\ \mathcal{K}_{\text{sol}} &= \left\{\mathbf{f} \,|\, \sum_{i=1}^{3} \mathcal{D}_{m-1}\left(\{\operatorname{rot}\mathbf{f}\}_i\right) = 0; \mathbf{f} \in \mathcal{V}_{m,sol}\right\} \end{aligned}$$

It can be shown that these spaces are given by

$$\begin{aligned} \mathcal{K}_{\text{irr}} &= \left\{\mathbf{f} \,|\, \exists g \in \mathbb{P}_{m-1}(\mathbb{R}^3) : \mathbf{f} = \nabla g\right\}\\ \mathcal{K}_{\text{sol}} &= \{\mathbf{f} \,|\, \exists \mathbf{g} \in \mathbb{P}_{m-1}(\mathbb{R}^3;\mathbb{R}^3) : \mathbf{f} = \operatorname{rot}\mathbf{g}\} \end{aligned}$$

Let $(\mathbf{p}_k)_{k\in[1:Q_1]}$ and $(\mathbf{q}_k)_{k\in[1:Q_2]}$ be the bases for $\mathcal{K}_{\text{irr}}$ and $\mathcal{K}_{\text{sol}}$ respectively.

The authors show that the irrotational problem has a unique solution if the sample set contains a unisolvent set in $\mathcal{K}_{\text{irr}}$. The solution is given by

$$\mathbf{f}_{\text{irr}}^{(\text{opt})}(\mathbf{x}) = \sum_{i=1}^{N} \Phi_d(\mathbf{x} - \mathbf{x}_i)\mathbf{a}_i + \sum_{k=1}^{Q_1} b_i \mathbf{p}_k(\mathbf{x}).$$

where the coefficients are given by the same equation (2.16), but the matrices $\mathbf{A}$ and $\mathbf{Q}$ are now defined by

$$\begin{aligned}
\{\mathbf{A}\}_{3(i-1)+k,3(j-1)+l} &= \{\mathbf{\Psi}_d(\mathbf{x}_i - \mathbf{x}_j)\}_{k,l}, \\
&\quad \forall\, i,j \in [1:N], \quad \forall\, k,l \in [1:3], \\
\{\mathbf{Q}\}_{3(i-1)+k,j} &= \{\mathbf{p}_j(\mathbf{x}_i)\}_k, \\
&\quad \forall i \in [1:N], \quad \forall j \in [1:Q_1], \quad \forall k \in [1:3].
\end{aligned}$$

Similarly, it is proved that the solenoidal solution has a unique solution if the sample set contains a unisolvent set in $\mathcal{K}_{\text{sol}}$, and the solution is given by

$$\mathbf{f}_{\text{sol}}^{(\text{opt})}(\mathbf{x}) = \sum_{i=1}^{N} \Phi_c(\mathbf{x} - \mathbf{x}_i)\mathbf{a}_i + \sum_{k=1}^{Q_2} b_i \mathbf{q}_k(\mathbf{x}).$$

The matrices $\mathbf{A}$ and $\mathbf{Q}$ in this case become

$$\begin{aligned}
\{\mathbf{A}\}_{3(i-1)+k,3(j-1)+l} &= \{\mathbf{\Psi}_c(\mathbf{x}_i - \mathbf{x}_j)\}_{k,l}, \\
&\quad \forall\, i,j \in [1:N], \quad \forall\, k,l \in [1:3] \\
\{\mathbf{Q}\}_{3(i-1)+k,j} &= \{\mathbf{q}_j(\mathbf{x}_i)\}_k, \\
&\quad \forall i \in [1:N], \quad \forall j \in [1:Q_2], \quad \forall k \in [1:3]
\end{aligned}$$

In sequel, we will first propose a more efficient computational solution for the scalar thin-plate interpolation/approximation problem (Chapter 3). We will then consider the vector problem and propose some practically relevant generalizations (Chapter 4). We will also consider several applications in image processing (Chapter 3) and ultrasound imaging (Chapter 5).

Chapter 3

Scalar Image Reconstruction from Non-Uniform Samples: A Fast Multiresolution B-spline Solution

Summary

We propose a novel method for image reconstruction from non-uniform samples with no constraints on their locations. We adopt a variational approach where the reconstruction is formulated as the minimizer of a cost that is a weighted sum of two terms: (i) the sum of squared errors at the specified points; (ii) a quadratic functional that penalizes the lack of smoothness. We search for a solution that is a uniform spline and show how it can be determined by solving a large, sparse system of linear equations. We interpret the solution of our approach as an approximation of the analytical solution that involves thin-plate splines and demonstrate the computational advantages of our approach. Using the two-scale relation for B-splines, we derive an algebraic relation that links together the linear systems of equations specifying

reconstructions at different levels of resolution. We use this relation to develop a fast multigrid algorithm. We demonstrate the effectiveness of our approach on some image reconstruction examples.

This chapter is largely based on our article [53].

3.1 Motivation and Main Contributions

Our goal in this chapter is to develop a numerically efficient algorithm to reconstruct images from non-uniform samples. We adopt the variational approach, since we do not intend to impose any restriction on the distribution of sample locations, and also, since we are interested in handling noisy samples, we choose the approximation method.

As seen Chapter 2, the optimal solution for this problem is given by Duchon's method, where the regularization functional belongs to a class of rotationally invariant semi-norms; the general solution is expressed as a weighted sum of thin-plate splines placed at the sample locations (cf. Equation (2.7)). Unfortunately, there are some practical limitations with this type of formulation. First, the weights have to be computed through a poorly conditioned linear system of equation, which is the main draw back of this method. Second, the matrix is not sparse. Hence, solving the system soon becomes overly expensive; the complexity is in $\mathcal{O}(M^3)$, where M is the number of sample points. Various solutions have been proposed for reducing this complexity and improving the numerical behaviour [41], but there is still a long way to go for making them practical. Another fundamental limitation is that computing the weights is only a part of the effort. Indeed, the solution has to be resampled numerically if it is to be displayed on a regular grid; this will cost an additional $\mathcal{O}(MN)$ operations, where N is the number of grid points.

In 1D, the situation is not as dramatic because the optimal solution can be expressed in terms of non-uniform B-splines which are compactly supported as opposed to the thin-plate splines which are not; this is the key for obtaining an efficient solution [54]. Unfortunately, this is not possible in higher dimensions; i.e., there are no compactly supported functions that reproduce thin-plate splines.

In this chapter, we provide a computationally efficient alternative to the

thin-plate spline method, where we use the same class of semi-norms for regularization, but where we choose to carry out the minimization of the approximation functional within the space of uniform B-splines. The key difference is that the basis functions are now attached to the reconstruction grid, as opposed to the data points. In other words, we are proposing to discretize thin plate splines using uniform B-splines with a degree that is matched to the underlying semi-norm. In this way, we have at least the guarantee that the solution in 1D coincides with the theoretical one, provided that the sample points are on the reconstruction grid. This helps to say qualitatively that the solution in general will not be very different from the exact analytical one.

The proposed approach has many advantages over the optimal method. First, the linear system for getting the B-spline coefficients is well-conditioned. Second, the system matrix is sparse resulting in much faster computation. Third, the formulation lends itself quite naturally to an efficient multiresolution and multigrid implementation, thanks to an interscale relation that is derived in this chapter. This reduces the complexity of solving the linear system to $\mathcal{O}(N)$, where N is the number of grid points. Fourth, after solving the linear system, there is no expensive resampling step as in thin-plate spline reconstruction. The samples at the grid locations are obtained by a simple digital filtering [55].

The present algorithm is flexible. One can reconstruct the image at any desired resolution (step size a); the solution will converge to the analytical one as a gets sufficiently small—the rate is given by the order of the spline. Since the computational complexity (number of unknown B-spline coefficients) is inversely proportional to a, it is a trade-off parameter that allows a compromise between the complexity and the accuracy. Another interesting points is that the complexity is essentially independent of number and location of the data points, and it primarily depends on the required resolution a.

This chapter is organized as follows. In Section 2.2, we discuss about the link between thin-plate splines and B-splines and provide some qualitative arguments for the suitability of B-splines to approximate thin-plate splines. In Section 2.3, we present our B-spline formulation and derive the corresponding linear system of equations. We also derive the interscale relation that relates the system matrices at various levels of resolution. We present the algorithm

in Section 2.4 and give the experimental results in Section 2.5.

3.2 Thin-plate Splines and B-splines

For the unidimensional problem, the thin-plate spline is given by $\phi(x) = |x|^{2p-1}$, which is a polynomial of degree $2p-1$, with a discontinuity of order $2p-2$ at the origin. This implies that the solution is a polynomial spline of degree $2p-1$ with knots at the x_i's. It turns out that these splines have basis functions, the so-called non-uniform B-splines, which are compactly supported and therefore much better conditioned than the radial basis functions [54, 56]. The corresponding numerical technique is called "smoothing splines" and is widely used in practical applications [57]. In the special case where the samples are uniformly spaced, i.e. $x_i = i \cdot a$, there is still another simplification since all the B-splines become shifted replicates of each other. The corresponding solution can be represented as $S(x) = \sum_k c_k \beta^{2p-1}(x/a - k)$, where $\beta^{2p-1}(x)$ is the central B-spline of degree $2p-1$. This means that the approximation problem can be discretized exactly if we work with B-splines of size a. Moreover, the expansion coefficients c_k of the smoothing spline can be evaluated very efficiently by recursive digital filtering [55]. Now, if there are gaps in the samples but the available ones are positioned on the grid (integer multiples of a), then the solution can still be expressed as a linear combination of B-splines with step size a, but the algorithm does not have a simple filtering interpretation anymore. The two cases of interest to us are $p = 1$ and $p = 2$, which lead to piecewise linear and cubic spline solutions, respectively.

Unfortunately, for dimensions higher than one, there are no compactly supported functions that span the same space as the thin-plate splines. Thus, a uniform B-spline discretization of the problem is not rigorously exact anymore. However, a B-spline basis with a degree that is matched to p remains the best possible choice among all tensor product shift-invariant bases, because it has a high enough order of differentiation and it is compatible with the optimal 1D solution. The slight discrepancy with the optimal analytical solution that this may generate is largely compensated by the computational advantages (sparse matrices, multi-resolution) provided by this type of representation. Additionally, the error can be made arbitrarily small by decreasing

the sampling step a. In this last respect, B-splines offer another advantage: for a given support size, they are the refinable functions that result in the smallest discretization error [58]. Specifically, for the B-spline of degree n which is of support $L = n + 1$, the approximation error decays like $\mathcal{O}(a^L)$, which is the maximum rate possible [59]. In other words, they provide the best quality for a given computational cost.

3.3 B-spline Discretization and the Solution

In this section, we present the proposed B-spline discretization of the problem and specify its solution. We discretize the variational reconstruction problem by searching for the optimal solution within the space of uniform splines.

3.3.1 B-spline Formulation

Given a set of noisy measurements $\{f_i\}$ of the image at sampling locations $\{x_i, y_i\}$, the approximation problem is now to find a uniform spline $S(x, y)$ of the form

$$S(x,y) = \sum_{l=0}^{(N-1)} \sum_{k=0}^{(N-1)} c_{k,l} \beta^n(x/a - k) \beta^n(y/a - l) \tag{3.1}$$

such that

$$\mathcal{C}(S) = \sum_i |S(x_i, y_i) - f_i|^2 + \lambda \int\int ||D^p S||^2 dxdy \tag{3.2}$$

is minimized. Obviously, the degree of the B-spline should be chosen such that the regularization term does not blow up; i.e., $n \geq p$.

The analytical part of this discretization process is to express the second part of the cost function in term of the expansion coefficients $c_{k,l}$. The regularization term is

$$\int\int ||D^p S||^2 dxdy = \sum_{q_1+q_2=p} \binom{p}{q_1} D_{q_1,q_2}$$

where

$$D_{q_1,q_2} = \int\int \left[\frac{\partial^p S(x,y)}{\partial x^{q_1} \partial y^{q_2}} \right]^2 dxdy$$

Define $\alpha_{q,a}(x) = \frac{d^q}{dx^q}\beta^n(x/a)$. Then

$$\begin{aligned} D_{q_1,q_2} &= \sum_{l,n}\sum_{k,m} c_{k,l}c_{m,n} \\ &\quad \left[\int \alpha_{q_1,a}(x-k)\alpha_{q_1,a}(x-m)dx\right] \\ &\quad \left[\int \alpha_{q_2,a}(y-l)\alpha_{q_2,a}(y-n)dy\right], \end{aligned}$$

which yields

$$D_{q_1,q_2} = \sum_{l,n}\sum_{k,m} c_{k,l}c_{m,n}\eta_{q_1}((m-k)a)\eta_{q_2}((n-l)a)$$

where

$$\eta_q(x) = \frac{d^q}{dx^q}\beta^n(x/a) * \frac{d^q}{dx^q}\beta^n(-x/a) \tag{3.3}$$

Define,

$$\gamma_q(k) = \eta_q(ka)$$

$$r_{q_1,q_2}(k,l) = \gamma_{q_1}(k)\gamma_{q_2}(l) \tag{3.4}$$

Then

$$\begin{aligned} D_{q_1,q_2} &= \sum_{l,n}\sum_{k,m} c_{k,l}c_{m,n}r_{q_1,q_2}(m-k,n-l) \\ &= \sum_{k,l} c_{k,l}\sum_{m,n} c_{m,n}r_{q_1,q_2}(m-k,n-l) \\ &= \langle r_{q_1,q_2}(k,l) * c_{k,l},\ \ c_{k,l}\rangle \end{aligned}$$

Hence,

$$\int\int ||D^p S||^2 dxdy = \sum_{q_1+q_2=p} \binom{p}{q_1} \langle r_{q_1,q_2}(k,l) * c_{k,l},\ \ c_{k,l}\rangle$$

Finally,

$$\int\int ||D^p S||^2 dxdy = \langle r(k,l) * c_{k,l},\ \ c_{k,l}\rangle$$

with $r(k,l) = \sum_{q_1+q_2=p}\binom{p}{q_1} r_{q_1,q_2}(k,l)$; in other words, the regularization term is a quadratic form of the $c_{k,l}$'s with a special convolutional structure.

In the filter component (3.4), $\gamma_q(k)$ is the discrete B-spline kernel of order $2n-2q+1$, convolved with the finite difference operator of order $2p$. This can be verified by using the following properties of splines [55]: (i) the derivative of a B-spline is a B-spline of reduced degree convolved with a finite difference

operator; (ii) the convolution of two B-splines results in a B-spline of increased order. Table 3.1 gives the regularization filters for the first order and the second order semi-norms.

p	$R(z_1, z_2)$
1	$\triangle_*^2(z_1)B^{2n-1}(z_1)B^{2n+1}(z_2)+$ $\triangle_*^2(z_2)B^{2n-1}(z_2)B^{2n+1}(z_1)$
2	$\frac{1}{(a)^2}\left[\triangle_*^4(z_1)B^{2n-3}(z_1)B^{2n+1}(z_2)+\right.$ $2\triangle_*^2(z_1)B^{2n-1}(z_1)\triangle_*^2(z_2)B^{2n-1}(z_2)+$ $\left.\triangle_*^4(z_2)B^{2n-3}(z_2)B^{2n+1}(z_1)\right]$

Table 3.1: Regularization filters. $\triangle_*^{2n}(z) = (z - 2 + z^{-1})^n$ (central finite difference operator of order $2n$); $B^n(z) = \sum_{k=-n/2}^{n/2} \beta^n(k) z^{-k}$ (discrete B-spline kernel of degree n).

To introduce the corresponding matrix formulation, we define the coefficient and data vectors

$$\mathbf{c} = [c_{0,0} \cdots c_{N-1,0} \cdots c_{N-1,N-1}]^T.$$

$$\mathbf{f} = [\cdots \ f_i \ \cdots]^T.$$

Then the cost is given by

$$\mathcal{C}(S) \quad = \quad ||\mathbf{f} - \mathbf{S}\mathbf{c}||^2 + \lambda \mathbf{c}^T \mathbf{R} \mathbf{c}$$

where

$$\{\mathbf{S}\}_{i,Nl+k} = \beta^n(x_i/a - k)\beta^n(y_i/a - l),$$

and $\mathbf{R}$ is a block-circulant matrix that correspond to the discrete filter $r(k, l)$. Through vector differentiation, we get the minimizer of the above cost as a solution of the following equation:

$$[\mathbf{D} + \lambda \mathbf{R}]\, \mathbf{c} = \mathbf{b} \tag{3.5}$$

where $\mathbf{D} = \mathbf{S}^T\mathbf{S}$ and $\mathbf{b} = \mathbf{S}^T\mathbf{f}$.

3.3.2 Interscale Relation

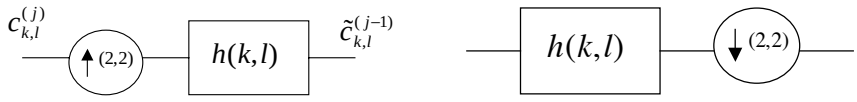
Let us now consider signal reconstructions at different scales. Specifically, let 2^j be the reconstruction grid size (scale j) and

$$S_j(x,y) = \sum_{l=0}^{(N-1)/2^j} \sum_{k=0}^{(N-1)/2^j} c_{k,l}^{(j)} \beta^n(x/2^j - k)\beta^n(y/2^j - l) \tag{3.6}$$

be the reconstructing function. A key property of the central B-spline of odd degree is the two-scale relation:

$$\beta^n(x/2^j) = \sum h(k)\beta^n(x/2^{j-1} - k) \tag{3.7}$$

where $h(k) = 2^{-n}\binom{n+1}{k}$ is the binomial filter. Consider a set of coefficients $\{c_{k,l}^{(j)}\}$ representing a 2D signal at scale j. It is well known from wavelet theory [60, 61] that the same signal can be represented at scale $j-1$ by the coefficients $\{\tilde{c}_{k,l}^{(j-1)}\}$ that are obtained by upsampling $\{c_{k,l}^{(j)}\}$ and filtering with h. The schematic is given in the figure 3.1(a) where $h(k,l) = h(k)h(l)$. This operation can be represented by a matrix, $\mathbf{U}_j$, that is obtained from the circulant matrix corresponding to the filter $h(k,l)$ after suppression of its odd index columns. The adjoint operation is the downsamping operation represented in the figure 3.1(b). The equivalent matrix is $\mathbf{U}_j^T$.



(a) Upsampling operation

(b) Downsampling operation

Figure 3.1:

The coefficient vector $\mathbf{c}_j = [\cdots c_{k,l}^{(j)} \cdots]$ of the image reconstruction at scale j must satisfy the equation

$$\mathbf{A}_j\mathbf{c}_j = \mathbf{b}_j, \tag{3.8}$$

with $\mathbf{A}_j = \mathbf{D}_j + \lambda\mathbf{R}_j$ where the matrices $\mathbf{D}_j$ and $\mathbf{R}_j$ are defined as in the previous section with a being replaced by 2^j. The block circulant matrix $\mathbf{R}_j$

is generated from the filter

$$r_j(k,l) = \sum_{q_1+q_2=p} r_{q_1,q_2,j}(k,l)$$

where

$$r_{q_1,q_2,j}(k,l) = \gamma_{q_1,j}(k)\gamma_{q_2,j}(l), \tag{3.9}$$

$$\gamma_{q,j}(k) = \eta_{q,j}(k2^j), \tag{3.10}$$

$$\eta_{q,j}(x) = \frac{d^q}{dx^q}\beta^n(-x/2^j) * \frac{d^q}{dx^q}\beta^n(x/2^j) \tag{3.11}$$

The following theorem gives an important property of the filters $\{r_{q_1,q_2,j}(k,l)\}$.

Theorem 1 *The filters $\{r_{q_1,q_2,j}(k,l)\}$ satisfy*

$$r_{q_1,q_2,j+1}(k,l) = [h_a(k,l) * r_{q_1,q_2,j}(k,l)]_{\downarrow(2,2)}$$

where $h_a(k,l) = h(k,l) * h(k,l)$ with $h(k,l)$ being the 2D version of the binomial filter in the two-scale relation (3.7), and where $\downarrow (2,2)$ denotes down sampling by a factor of 2 in each dimension.

Using the above theorem, we prove the following:

Corollary 1 *The system matrices at scales j and $j+1$ are related by*

$$\mathbf{A}_{j+1} = \mathbf{U}_j^T \mathbf{A}_j \mathbf{U}_j \tag{3.12}$$

$$\mathbf{b}_{j+1} = \mathbf{U}_j^T \mathbf{b}_j \tag{3.13}$$

where $\mathbf{U}_j$ is the subsampling matrix described in this section.

Proof: Applying the two-scale relation, we get $\mathbf{S}_{j+1} = \mathbf{S}_j \mathbf{U}_j$. Hence

$$\mathbf{b}_{j+1} = \mathbf{S}_{j+1}^T \mathbf{f} = \mathbf{U}_j^T \mathbf{S}_j^T \mathbf{f} = \mathbf{U}_j^T \mathbf{b}_j$$

and

$$\mathbf{D}_{j+1} = \mathbf{S}_{j+1}^T \mathbf{S}_{j+1} = \mathbf{U}_j^T \mathbf{S}_j^T \mathbf{S}_j \mathbf{U}_j = \mathbf{U}_j^T \mathbf{D}_j \mathbf{U}_j \tag{3.14}$$

Now, consider $\mathbf{U}_j^T \mathbf{R}_j \mathbf{U}_j$. The equivalent filter is $[h_a(k,l) * r_j(k,l)]_{\downarrow 2}$. By Theorem 1, it is equal to $r_{j+1}(k,l)$, which is equivalent to writing

$$\mathbf{R}_{j+1} = \mathbf{U}_j^T \mathbf{R}_j \mathbf{U}_j \tag{3.15}$$

Combining (3.14) and (3.15), we get

$$\mathbf{A}_{j+1} = \mathbf{U}_j^T \mathbf{A}_j \mathbf{U}_j.$$

■

3.4 Reconstruction Algorithm

We have shown that the reconstruction problem is equivalent to solving a system of linear equations (cf. (3.5)). A key property is that the present system is sparse and well-conditioned in contrast with the thin-plate spline method where the matrix is dense and ill-conditioned. We also have the flexibility to choose the step size a with the guarentee that the solution converges to the analytical one (thin-plate spline) when a is sufficiently small. The main difficulty here is that the number of unknowns $\{c_{k,l}\}$ for a given step size a is typically very large. Therefore the choice of the method for solving the linear system becomes crucial in order to do the reconstruction in affordable time. Enabled by the right choice of the basis functions and our theorem on interscale relation, we develop an efficient multiresolution algorithm to achieve fast reconstruction.

3.4.1 Multiresolution Strategy

Since the dimension of the linear system is very large and depends upon the resolution of reconstruction, one naturally thinks of multiresolution.

Let a_0 be the required resolution and $N = 2^J + 1$. Then the coarsest resolution will be $2^{J-1}a_0$. The idea is to solve $\mathbf{c}_{J-1}$ exactly and to interpolate the solution to the next finer scale using the two-scale relation for B-spline [55]. This serves as initialization for the computation of $\mathbf{c}_{J-2}$ and so on. At the end of the process, $\mathbf{c}_0$ gives the reconstruction at the required resolution.

Note that the regularization matrix $\mathbf{R}_j$ has a generic form independent of the resolution, while the matrix $\mathbf{D}_j$ is clearly scale-dependent since it is defined by the sample points. Because of the compact support of B-splines, the cost for the direct evaluation of $\mathbf{D}_j$ is the same at each scale and is proportional to the number of data points, which is typically quite large. To cut down on this cost, we evaluate the matrix at the fine scale and use our inter-scale relation to derive the matrices at coarser resolutions. Here the computation is by sparse matrix multiplication which makes the complexity now depend on the resolution level. This enables us to compute all the matrices very efficiently with complexity $\mathcal{O}(N \times N)$; that is, a complexity that is proportional to the number of grid points (i.e., the number of unknown B-spline coefficients).

We handle the computation of the coefficient vectors at each scale from their initializations using the multigrid iteration described below.

3.4.2 Multigrid Iteration

A multigrid iteration is obtained by using classical iterators as the building blocks. We first describe the classical iterative scheme and then the multigrid iteration. A classical iterative scheme gives a way to get a refined estimate of the solution from a given estimate. Let $\mathbf{c}_j^{(k)}$ be the current estimate of the solution for level j. A refinement step is represented as follows:

$$\mathbf{c}_j^{(k+1)} = \mathbf{c}_j^{(k)} + \omega \hat{\mathbf{A}}_j^{-1}(\mathbf{b}_j - \mathbf{A}_j \mathbf{c}_j^{(k)}) \tag{3.16}$$

where $\hat{\mathbf{A}}_j^{-1}$ denotes an approximate inverse and ω is a damping factor. For the estimate $\mathbf{c}_j^{(k)}$, the error $\mathbf{e}_j$ and the residue $\mathbf{r}_j$ are given by

$$\mathbf{e}_j = \mathbf{c}_j - \mathbf{c}_j^{(k)}$$

$$\mathbf{r}_j = \mathbf{b}_j - \mathbf{A}_j \mathbf{c}_j^{(k)}$$

When $\hat{\mathbf{A}}_j$ in (3.16) is the diagonal of $\mathbf{A}_j$, then the iterator is called damped-Jacobi; if instead it is the lower triangular part of $\mathbf{A}_j$, the algorithm yields the so-called Gauss-Seidel iterator. See [62] for a comprehensive treatment of such numerical schemes.

It is important here to note that such iterators have a smoothing effect on the error. The larger k, the smoother the error, and for sufficiently large k, there will not be significant improvement in the error anymore. Since the error is smooth after a few relaxations (iterations), it can be well represented at lower resolution. In other words, one can try to compute the error at a coarser scale.

To do this we first consider the residual equation:

$$\mathbf{A}_j \mathbf{e}_j = \mathbf{r}_j$$

Then the computation of error is formulated as

$$\mathbf{A}_{j+1} \tilde{\mathbf{c}}_{j+1} = \tilde{\mathbf{b}}_{j+1}$$

where $\tilde{\mathbf{b}}_{j+1}$ is obtained by filtering and downsampling $\mathbf{r}_j$; that is,

$$\tilde{\mathbf{b}}_{j+1} = \mathbf{U}_j^T \mathbf{r}_j$$

Here $\tilde{\mathbf{c}}_{j+1}$ is the lower dimensional version of the error to be computed. $\tilde{\mathbf{c}}_{j+1}$ is computed by the same iterative scheme with zero initialization. Afterwards one gets back the error $\mathbf{e}_j = \mathbf{U}_j\tilde{\mathbf{c}}_{j+1}$, which is used to correct $\mathbf{c}_j^{(k)}$. This is called coarse-grid correction.

The advantage is twofold. First, it is computationally efficient to iterate in lower dimension. Second, a smooth error becomes somewhat bumpier at a coarser resolution. Hence, relaxation on the coarser grid further reduces the error. However, the coarse grid correction is not perfect and also introduce some amount of non-smooth error. Hence it is customary to do a few relaxations after coarse-grid correction. The overall scheme is computationally more effective than relaxation at the finest grid. However, when the dimension of the system at the coarser grid is large, relaxation will also stall there. Since the problem here is exactly the same as the original one, one can think of applying the same three-step procedure recursively. This is called a multigrid V-cycle [63, 62]. Figure 3.2 gives the schematic of multigrid V-cycle. Here $\hat{\mathbf{c}}_j$ is the initialization for V-cycle that is obtained by interpolating $\mathbf{c}_{j+1}$ (solution for the level $j+1$) using the B-spline two-scale filter.

Whenever applicable, multigrid algorithms are extremely efficient; in fact, they are among the best numerical methods available. There are also some general convergence proofs [62] that are directly applicable to our case because of the existence of the interscale relation (3.12) that relates the systems of equations at successive scales.

The parameters that affect the computational complexity of a multigrid V-cycle are the number of iterations before and after the coarse-grid correction, n_1 and n_2 respectively.

3.4.3 Implementation Issues

The forgoing discussions cover the main philosophy of our method. However, to get the full benefit of the multiresolution/multigrid algorithm described above and to make the algorithm almost real-time, efficient implementation of the building blocks in the algorithm is crucial. The implementation should meet stringent constraints on computational complexity and storage requirements.

To get a flavor of what is involved, let us consider the structure of the matrix $\mathbf{A}_0$ or $\mathbf{D}_0$ for cubic reconstruction. The matrix is block-band diagonal

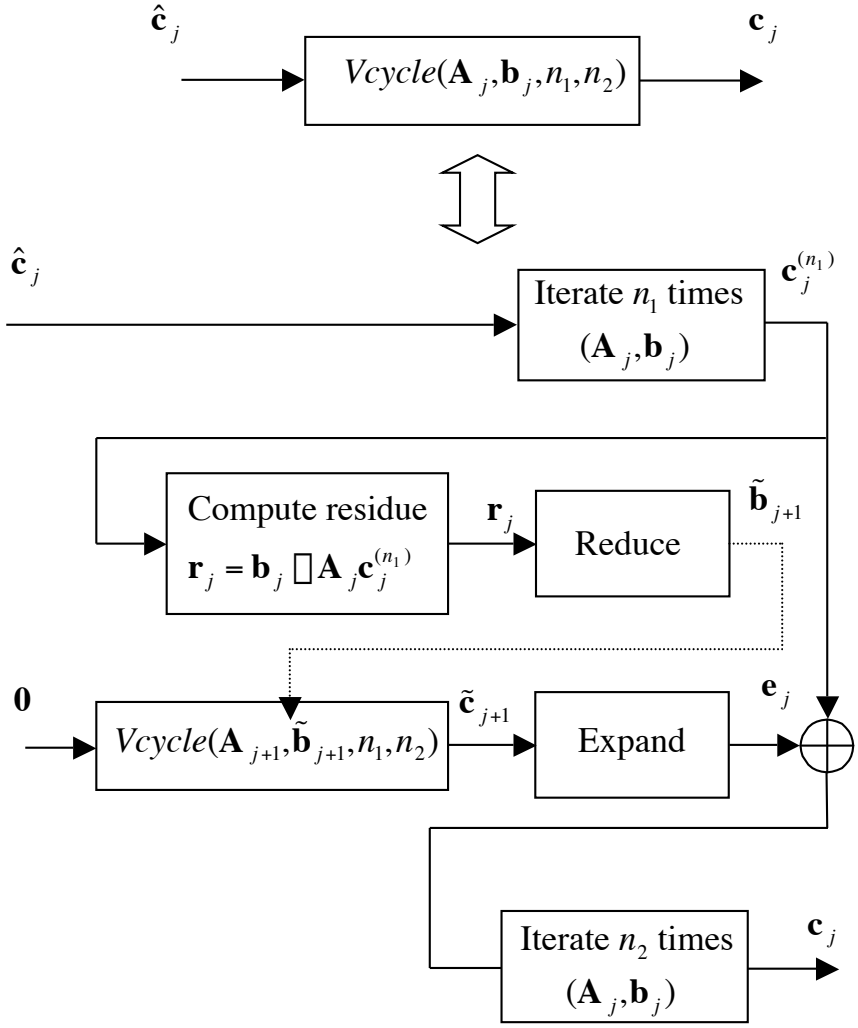


Figure 3.2: Schematic of multigrid V-cycle

of width seven, and each block is band-diagonal of width seven. The whole matrix is symmetric and each block in the matrix is symmetric as well. In our implementation, we exploit this structure to reduce storage requirements. The non-redundant elements are stored in a sparse format without explicit indexing. The basic matrix iterators and modules (especially Gauss-Seidel iteration) are coded specifically for this sparse format.

The sample matrix $\mathbf{D}_0$ is computed from the sample points directly into the above mentioned sparse format. The matrices $\mathbf{A}_1, \ldots, \mathbf{A}_{J-1}$ are also precomputed using relation (3.12) directly into the sparse format exploiting the special structure of $\mathbf{U}_j$. In this way, we are able to compute the matrices $\mathbf{A}_1, \ldots, \mathbf{A}_{J-1}$ with around $20(N \times N)$ multiplications in the case of reconstruction with linear splines.

3.5 Experimental Results

For our experiments, we consider three kinds of scenarios: (i) reconstruction from a subset of samples of a given digital image; (ii) reconstruction after a geometric transformation of a given image; and (iii) reconstruction from sparse samples of a synthetic phantom. In the first case, the inputs sample points are obtained by randomly choosing the pixels or choosing the pixels along some selected contours. In the second case, the input sample points are obtained by a geometric transformation of a uniform grid points. In the last case, we choose samples from a synthetic phantom along some lines. We provide the samples to our algorithm in a list format $\{x_i, y_i, f_i\}$.

In all our experiments we consider two settings: (i) linear spline reconstruction with Duchon's first semi-norm as the regularization; (ii) cubic spline reconstruction with Duchon's second semi-norm as the regularization. Unless stated otherwise, we adjusted λ empirically for the best visual results. We observed that the most favorable value of λ is image specific and typically proportional to the noise variance when the data is corrupted by noise.

3.5.1 Reconstruction from Incomplete Data

For the experiments in this category, we define the reconstruction error as $e_r = \frac{||\mathbf{I}_o - \mathbf{I}_r||}{||\mathbf{I}_o||}$, where $\mathbf{I}_o$ is the original image and $\mathbf{I}_r$ is the reconstructed image. Note that this will be an underestimate of the performance of the

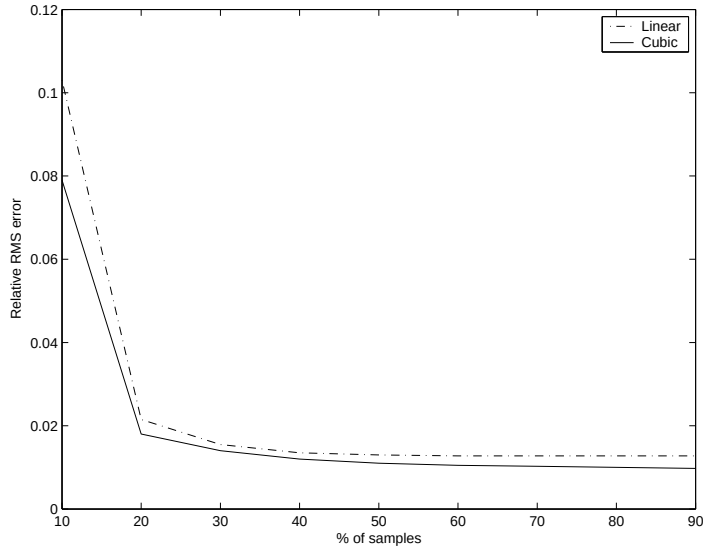


Figure 3.3: Reconstruction error for various degrees of subsampling

algorithm—especially when the initial data is sampled arbitrarily—since it does not correspond to the objective function that we minimize in our formulation of the problem.

We first applied the algorithm on a face image. We sub-sampled the image by applying a binary mask obtained by thresholding the Laplacian of the image. Figure 3.3 gives the reconstruction error for both linear and cubic reconstructions for various numbers of input samples. Cubic spline reconstruction gives lower reconstruction error as one would expect. Figure 3.4 compares the reconstructed images from 20% of the samples. This example demonstrates that our algorithm is able to handle both large and small gaps simultaneously in an efficient way.

In figure 3.5, we give lower resolution reconstructions from the same set of samples. One can clearly see the artifacts; they are somewhat reduced in the case of cubic spline reconstruction.

Figure 3.6 displays reconstruction results for an MRI brain image. The reconstruction was from 30% of samples that were retained in the same way as in the previous experiment. In this case, the improvement of the cubic spline reconstruction over the linear one is more visible even though both images

(a) Original image

(b) 20% of samples

(c) Reconstructed image using linear splines

(d) Reconstructed image using cubic splines

Figure 3.4: Reconstruction from high Laplacian points. Sampling density: 20%

(a) Linear spline reconstruction with $a = 4$

(b) Cubic spline reconstruction with $a = 4$

(c) Linear spline reconstruction with $a = 8$

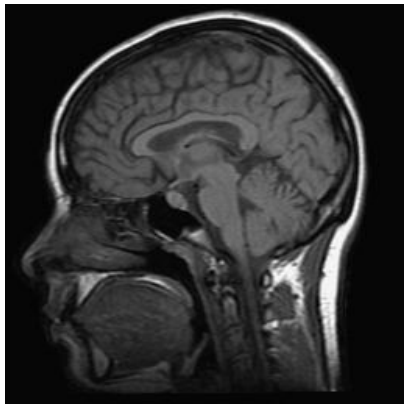
(d) Cubic spline reconstruction with $a = 8$

Figure 3.5: Lower resolution reconstruction

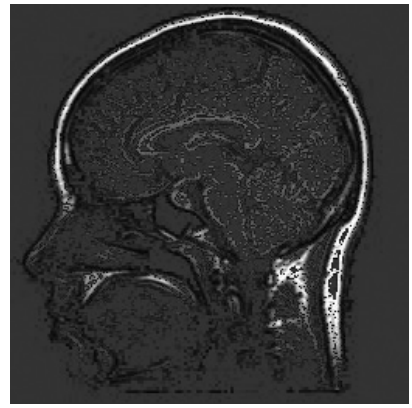
have same reconstruction error. The image from linear spline reconstruction shows more bright spots than that of cubic spline reconstruction. This is due to the fact that the contours are less prominent than those of the previous image and that regularization with second order semi-norm is more suitable when there are discontinuities in the contours. Figure 3.7 shows the images for a larger smoothing factor. In this case as well, both cubic spline and linear spline reconstructions have the same reconstruction error. However, cubic spline reconstruction is smoother than the linear one with less artifacts (bright spots). Figure 3.8 gives lower resolution reconstructions of the MRI image from the same set of samples as in the figure 3.6. One can clearly see the artifacts in this case too.

The above examples demonstrate the ability of the algorithm to reconstruct data when there are large sampling gaps, something that is typically not possible with the reconstruction algorithms for bandlimitted functions mentioned in the introduction [12, 13]. However, we must admit that our algorithm will fill the sampling gaps smoothly by extrapolating the available information (samples). For this reason, it cannot correctly recover image parts for which the contour or texture information has been lost. Note that this effect may also be used to our advantage for suppressing unwanted objects or features in images, as demonstrated in the next example. Here, we started with the contour map of 3.4(b) and applied a coarse binary mask to suppress all samples in the regions of the round spot on the face and the rose in the lower left corner. The corresponding reconstruction is given in Figure 3.9. It is still looking realistic, even though the selected objects have entirely disappeared. This is due to the regularization term that smoothly extrapolates the missing pixel values from the surroundings.

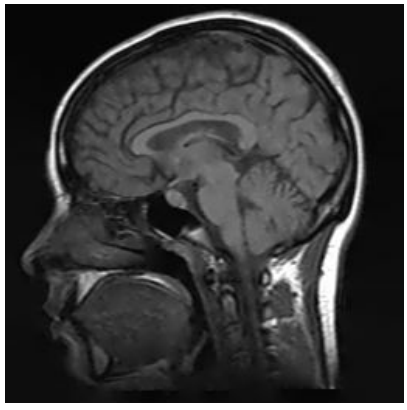
The proposed algorithm is obviously also applicable to the case of random sub-sampling, which is the context in which reconstruction algorithm for band-limited functions are usually tested [12, 13]. An example of such a reconstruction is given in Figure 3.10. As one would expect, the quality is inferior to that obtained with the reconstruction from high-Laplacian points using the same percentage of samples ($e_r = 0.0204$ vs. 0.0145). We have verified experimentally that the reconstruction errors obtained under these conditions are essentially equivalent to those of alternative techniques for bandlimited functions, provided that the reconstruction parameters are



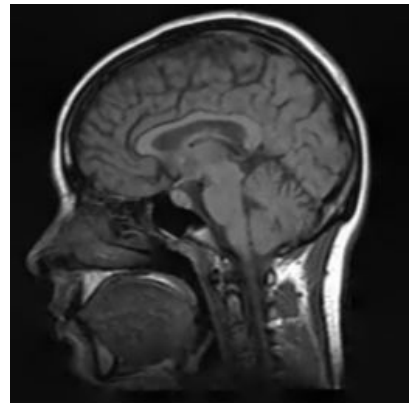
(a) Original image



(b) 30% of samples

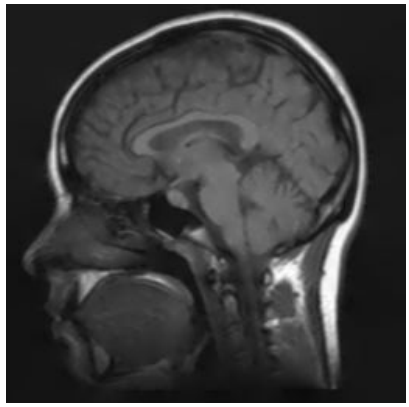


(c) Reconstructed image using linear spline. $\lambda = 1.5848 \times 10^{-3}$

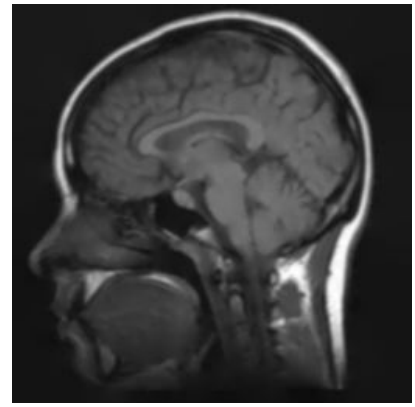


(d) Reconstructed image using cubic spline. $\lambda = 1.5848 \times 10^{-3}$

Figure 3.6: Reconstruction from high Laplacian points. Sampling density: 30%



(a) Reconstructed image using linear spline. $\lambda = 7.9430 \times 10^{-1}$



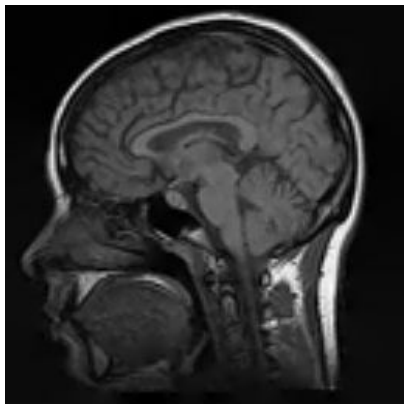
(b) Reconstructed image using cubic spline. $\lambda = 7.9430 \times 10^{-1}$

Figure 3.7: Effect of λ

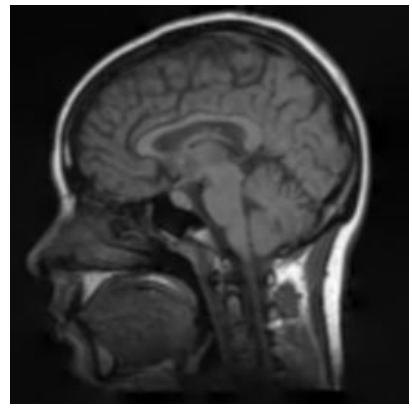
matched (same reconstruction density and λ small). This behavior is consistent with the property that a cubic spline is a very good approximation of a bandlimited function [64], and that our algorithms provides the least squares solution as λ tends to zero (data term only). A very significant advantage of our method is its computational speed: this is due to two important ingredients that are specific to our formulation: (1) the use of compactly supported basis functions, and (2) a most efficient multi-grid algorithm that can solve the linear system of equations with a complexity that is essentially proportional to the number of reconstructed samples.

3.5.2 Reconstruction with geometric transformation (texture mapping)

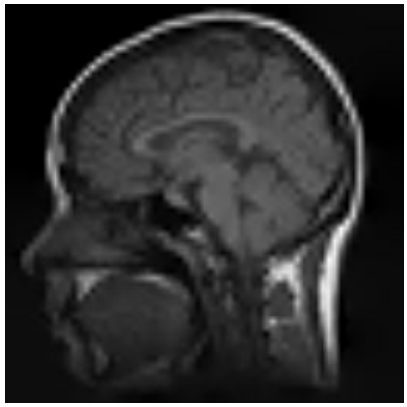
Texture mapping typically refers to the process of geometrically transforming a given source image or pattern in order to simulate its mapping onto a 3D surface. There are potentially two ways to do this: (i) applying the inverse transformation for each pixel position in the target image to get the interpolated value from the source image; (ii) applying the transformation of each source pixel and using our non-uniform reconstruction method to get



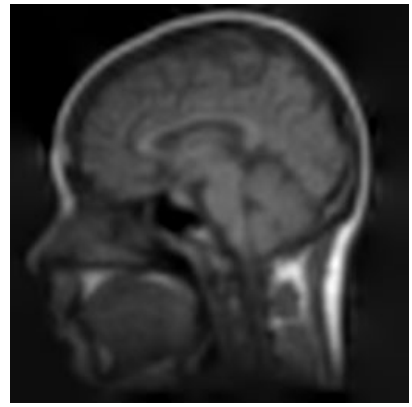
(a) Linear spline reconstruction with $a = 2$



(b) Cubic spline reconstruction with $a = 2$



(c) Linear spline reconstruction with $a = 4$

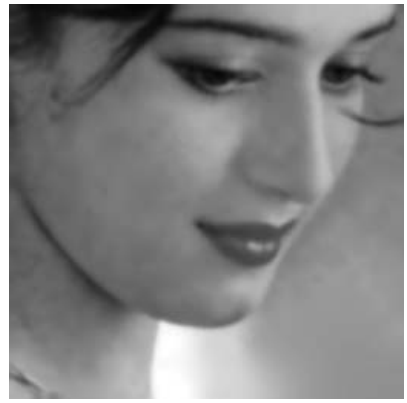


(d) Cubic spline reconstruction with $a = 4$

Figure 3.8: Lower resolution reconstruction

(a) Original image

(b) Input samples



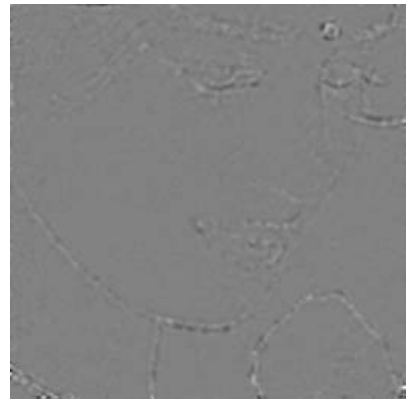
(c) Reconstructed image

Figure 3.9: Reconstruction from partial contours

(a) Original image

(b) 30% of samples

(c) Reconstructed image. $\lambda = 3.086 \times 10^{-1}$. $e_r = 0.0204$



(d) Rescaled error image

Figure 3.10: Reconstruction from random samples

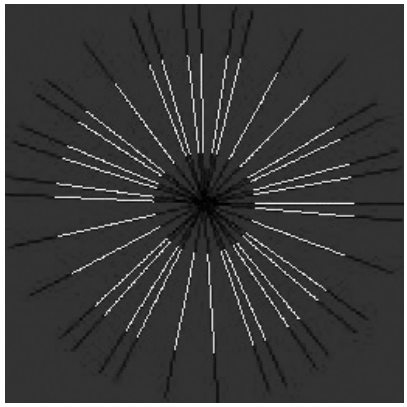
Figure 3.11: Example of texture mapping: reconstruction with geometric transformation

the target image. The second method has the clear advantage that it uses the information present in the source image completely, whereas, there might be some loss of information with the first approach (unused pixels in the source image). Our method will give the least squares fit in the regions where the input samples (transformed source pixels) outnumber the reconstruction grid points (target pixels). This reduces reconstruction artifacts. Figure 3.11 gives an example of texture mapping generated using our algorithm, where the lena image is mapped onto a cylinder. The key feature of our technique is that there are no aliasing artifacts and that the sharpness of the pictorial information is essentially preserved when λ is small.

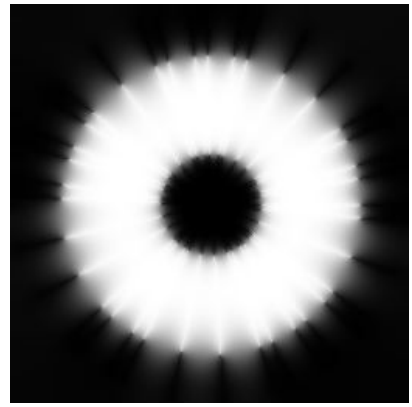
3.5.3 Phantom reconstruction

Our next test image takes the value 255 inside a circular ring and zero outside. We sample this phantom along some radial lines. The data is sampled non-uniformly along the angular dimension and uniformly in the radial direction. The reconstruction is challenging because the samples are sparse and the boundaries are lost. The results are given in the figure 3.12. One can

clearly see that the cubic spline reconstruction gives better reconstruction of boundaries. This is due to the fact that the underlying radial basis function for the first order smoothness semi-norm is less suitable for recovering lost boundaries. In fact, the analytical RBF method with $p = 1$ is not numerically stable in 2D since the radial basis function $\log r$ is unbounded at the sample locations.



(a) Input samples



(b) Linear spline reconstruction

(c) Cubic spline reconstruction

Figure 3.12: Reconstruction of boundary.

In all the cases above, in order to make the comparison meaningful, we

set up the iterations such that the residue of the linear system goes to the machine precision. To achieve this, it was required to have $n_1 = n_2 = 32$ for a reconstruction in a 256×256 grid; it took 2 seconds and 0.8 seconds on a 1.8 GHz Macintosh G5 system for cubic and linear splines, respectively. However, we were able get a visually acceptable reconstruction with $n_1 = n_2 = 2$ that took 0.8 seconds and 0.2 seconds using cubic splines and linear splines, respectively. Also, we should mention that the number and the distribution of input samples have negligible effect on the speed of the algorithm. The speed is primarily determined by the size of the reconstruction grid. This factor makes our method quite attractive in the noisy situations where one needs to have more input samples than the number of reconstruction grid points.

The present algorithm is very general; it includes previously published spline algorithms as particular cases. For instance, consider the case when the input data is a standard digital image. When $a = 1$, the algorithm gives the smoothing spline approximation of the image and is functionally equivalent to filtering algorithm described in [65]. In fact, because of the multigrid implementation, the computational complexity of our algorithm is in the same order as that of the FFT algorithm given in [65] with the advantage that the present scheme works for arbitrary samples. When $\lambda \rightarrow 0$, it gives the standard B-spline interpolation [55]. One can also generate least squares pyramid, by choosing $\lambda \rightarrow 0$ and $a = 2^i$. By choosing an appropriate reconstruction grid size and a diagonal affine transformation, one can achieve least squares rescaling [66], as well as more general types of geometric transformations as illustrated in the figure 3.11.

Conclusion

We developed a new method for regularized image reconstruction from arbitrarily spaced samples. We chose to reconstruct a continuously-defined function that is a uniform spline and selected a regularization term within Duchon's class of smoothness semi-norms. We interpreted our scheme as a way to discretize the thin-plate spline method which gives the optimal analytical solution of the approximation problem in the continuous domain. The key point of our scheme is that it uses basis functions (B-splines) that are

well conditioned; this makes our approach much more stable numerically and computationally advantageous than the classical thin-plate spline method. We provided a multiresolution formulation that allowed us to accelerate the reconstruction by way of a fast multigrid algorithm; the key ingredient here is an algebraic relation that links the reconstruction equations at different resolutions. Our algorithm has a number of advantageous features that should make it attractive for practical applications:

- The user can select the resolution a of the reconstruction grid. This parameter controles the trade-off between computational complexity and reconstruction accuracy. For a sufficiently small, the reconstruction converges to the solution of the thin-plate spline problem.
- It has a complexity that depends primarily on the number of reconstruction grid points. There is essentially no dependence on the location and the number of samples.

Appendix

Proof of theorem

Define

$$\beta_j^n(x) = \beta^n(x/2^j)$$
$$\alpha_{q,j}(x) = \frac{d^q}{dx^q}\beta_j^n(x)$$

Let $h(k)$ be the two-scale filter. Since

$$\beta_{j+1}^n(x) = \sum_k h(k)\beta_j^n(x - k2^j)$$

We get

$$\alpha_{q,j+1}(x) = \sum_k h(k)\alpha_{q,j}(x - k2^j) \tag{3.17}$$

From (3.11), $\eta_{q,j}(x) = \alpha_{q,j}(x) * \alpha_{q,j}(-x)$. Hence from (3.17) we get

$$\eta_{q,j+1}(x) = \sum_k h_a(k)\eta_{q,j}(x - k2^j) \tag{3.18}$$

where $h_a(k) = h(k) * h(k)$. Substituting $x = m2^{j+1}$ yields

$$\eta_{q,j+1}(m2^{j+1}) = \sum_k h_a(k)\eta_{q,j}(m2^{j+1} - k2^j) \tag{3.19}$$

From (3.10), the above equation gives

$$\gamma_{q,j+1}(m) = \sum_k h_a(k)\gamma_{q,j}(2m-k) \tag{3.20}$$

Hence $\gamma_{q,j+1}(k) = [\gamma_{q,j}(k) * h_a(k)]_{\downarrow 2}$. Now, from (3.9), we get

$$r_{q_1,q_2,j}(k,l) = \gamma_{q_1,j}(k)\gamma_{q_2,j}(l)$$

Due to separability

$$r_{q_1,q_2,j+1}(k,l) = [h_a(k,l) * r_{q_1,q_2,j}(k,l)]_{\downarrow(2,2)} \tag{3.21}$$

where $h_a(k,l) = h(k,l) * h(k,l)$. Note that the proof also carries over to any refinable function other than splines.

Chapter 4

Vector Field Reconstruction from Non-uniform Projected Samples: Optimal Method

Summary

We address the problem of reconstructing a vector field, $\mathbf{f} : \mathbb{R}^n \rightarrow \mathbb{R}^n, n = 2, 3$, from a set of projected samples of the form $s_i = \mathbf{d}_i^T \mathbf{f}(\mathbf{x}_i)$, where $\{\mathbf{x}_i\}$ are sampling locations, and $\{\mathbf{d}_i\}$ are projection directions. We formulate the reconstruction task as finding the minimizer of the functional $J_a(\mathbf{f}) = \sum_i (\mathbf{d}_i^T \mathbf{f}(\mathbf{x}_i) - s_i)^2 + \lambda J(\mathbf{f})$, where J is a suitable quadratic plausibility criterion given by $J(\mathbf{f}) = \int_{\mathbb{R}^n} \mathbf{f}^T(\mathbf{y}) \mathbf{U}(\mathbf{x} - \mathbf{y}) \mathbf{f}(\mathbf{x}) \mathrm{d}\mathbf{x}\mathrm{d}\mathbf{y}$, with $\mathbf{U}$ being a vector smoothness operator. We provide the solution to this minimization problem and show how it can be expressed in terms of the Green's function of the operator $\mathbf{U}$.

To identify the most appropriate class of regularization operators $\mathbf{U}$, we impose that the solution has two desirable properties: (i) rotational invariance, and (ii) scale invariance. These properties induce some restrictions on $\mathbf{U}$, and yield a complete parametric family of operators. We provide the

expressions for the corresponding Green's functions, which we call the *generalized vector splines*.

Using synthetic phantom data, we demonstrate the effectiveness of the method, and also the importance of tuning $\mathbf{U}$ appropriately for the problem at hand.

4.1 Motivation and Main Contributions

Our goal is to develop a method to reconstruct a vector field $\mathbf{f} : \mathbb{R}^n \rightarrow \mathbb{R}^n, n = 2, 3$, from non-uniformly spaced samples of the form $s_i = \mathbf{d}_i^T \mathbf{f}(\mathbf{x}_i)$, where $(\mathbf{x}_i)_{i\in[1:N]}$ are sampling locations with corresponding project directions $(\mathbf{d}_i,)_{i\in[1:N]}$. This sampling scheme is called the "projected sampling" and the set of triplets $(\mathbf{x}_i, \mathbf{d}_i, s_i)_{i\in[1:N]}$ forms our data set. There is no restriction on the sampling locations $(\mathbf{x}_i)_{i\in[1:N]}$ other than the fact that $(\mathbf{x}_i, \mathbf{d}_i, s_i)_{i\in[1:N]}$ should include a unisolvent set; in particular there may be several distinct measurements originating from the same spatial location $\mathbf{x}$. Thus the level of generality of the method is such that it includes conventional vector sampling as a particular case (cf. Section 1.2.2).

It should be noted that the incompleteness of the data can be two-fold: (i) the sample locations are non-uniform and there can be arbitrarily large sampling gaps; (ii) the samples themselves are incomplete. This type of problem is notoriously ill-posed, and hence, is best handled by a variational approach. To remove any kind of ambiguity, we formulate the problem as the task of finding the minimizer of the following cost functional:

$$J_a(\mathbf{f}) = \sum_{i=1}^{N} (\mathbf{d}_i^T \mathbf{f}(\mathbf{x}_i) - s_i)^2 + \lambda J(\mathbf{f}), \tag{4.1}$$

where $J(\mathbf{f})$ is a quadratic functional. The role of J is to make the problem well-posed by specifying a "plausible" or a "smooth" solution. This also gives a mechanism for trading closeness of the fit to the data against smoothness of the solution. Since a smoothness measure of a vector quantity that is physically meaningful should be vectorial in nature, one should allow cross-terms among the components of the vector field in J. Hence, we write the general form of J as

$$J(\mathbf{f}) = \int_{\mathbb{R}^n} \mathbf{f}^T(\mathbf{y}) \mathbf{U}(\mathbf{x} - \mathbf{y}) \mathbf{f}(\mathbf{x}) d\mathbf{x} d\mathbf{y}$$

Here $\mathbf{U}$ is the $n \times n$ matrix smoothness operator.

We are especially interested in handling the situation where the data set is highly "indeterminate". In such cases, it is advantageous to find a complete parametric family of $\mathbf{U}$ that yield some generally desirable properties on the solution. The completeness of the parameterization will ensure the adaptability of $\mathbf{U}$ according to the problem at hand. In this regard, we are not aware of any such characterization of vector smoothness operators in literature. As mentioned in Chapter 1, there exists such a characterization in the scalar case; it is provided by Duchon's theory which gives the family of semi-norms that satisfies rotational invariance and scale invariance (Section 1.3.6).

Since we believe that rotational- and scale-invariances are highly desirable properties, we have taken up the task of extending these notions for vector field reconstruction. We find a corresponding family of smoothness operators $\mathbf{U}$. We then give the solution to our reconstruction problem using this smoothness model.

The main contributions of this chapter are as follows:

1. *The extension of the notion of rotational invariance for vector fields and the study of its connection with the properties of the vector smoothness operator.* In the process, we also generalize the notion of scale invariance to subspace scale invariance that is less restrictive, and show its implication on the smoothness operator (Section 4.2).

2. *The identification of the complete family of smoothness operator that satisfy the rotational- and scale-invariance* (Section 4.3). Interestingly, we show that a rotation- and scale-invariant regularization operator consists of a weighted sum of two parts: (i) a divergence operator followed by a fractional Duchon's operator of order γ_d ; (iii) a curl operator followed by a fractional Duchon's operator of order γ_c on each component. If one further Includes the relative weighting between these two parts, this yields a three-parameter family of operators. The operators proposed in [49, 50] are special cases of this family, with $\gamma_d = \gamma_c = m$ (integer) (Section 2.2). We provide some physical arguments to demonstrate the use of having unequal smoothness order for divergence and curl fields. In particular, we explain how divergence-free or rotation-free reconstruction can be achieved by appropriately choosing the parame-

ters.

3. *The specification of the solution for the optimal reconstruction from the projected samples using this proposed family smoothness operators* $\mathbf{U}$. In particular, we show how the required solution can be constructed in terms of the Green's functions of $\mathbf{U}$. We give the expression for the Greens functions, which we call the *generalized vector splines* (Section 4.5).

4. The implementation and demonstration of the method with some test examples examples (Section 4.6).

4.2 Problem Formulation

4.2.1 The Minimization Problem

Given the measurement set $\{\mathbf{x}_j, \mathbf{d}_j, s_j\}$, the reconstruction problem is defined as the following minimization task:

$$\mathbf{f}^{(\text{opt})}(\mathbf{x}) = \underset{\mathbf{f}\in\mathcal{V}}{\operatorname{argmin}}\, J_a(\mathbf{f}), \tag{4.2}$$

where $J_a(\mathbf{f})$ is as given in (4.1). We postpone the specification of $\mathcal{V}$ until we find a characterization of J.

$\mathcal{K}$ is defined as the space all functions such that $J(\mathbf{f}) = 0$, which we call the full kernel. The space of function within the reconstruction space $\mathcal{V}$ such that $J(\mathbf{f}) = 0$ is called the reconstructing kernel, which is given by $\mathcal{K}_f = \mathcal{K} \bigcap \mathcal{V}$. It is important to make sure that the reconstructing kernel is sufficiently small (at least finite dimensional). Otherwise, (4.2) may not have a unique solution.

4.2.2 General Form of J

We write the general form of the semi-norm as follows:

$$J(\mathbf{f}) = \sum_{i=1}^{N} \left\langle \mathbf{l}_i^T * \mathbf{f}, \mathbf{l}_i^T * \mathbf{f} \right\rangle \tag{4.3}$$

Here, $\mathbf{l}_i$ is a vector with its elements formed of derivatives of dirac distribution. Note that, by construction, $J(\mathbf{f})$ is translation-invariant. Any quadratic

semi-norm that is based on derivatives can be expressed in this form. For example, applying second order Duchon's semi-norm (cf. Table 2.1) on each component independently on a 2D vector field can be expressed as

$$J(\mathbf{f}) = \sum_{j=1}^{2} \sum_{i=1}^{3} \left\langle \mathbf{l}_{ij}^T * \mathbf{f}, \mathbf{l}_{ij}^T * \mathbf{f} \right\rangle \tag{4.4}$$

where

$$\mathbf{l}_{1j} = \frac{\partial^2 \delta}{\partial x^2} \mathbf{e}_j, \;\; \mathbf{l}_{2j} = \frac{\partial^2 \delta}{\partial y^2} \mathbf{e}_j, \;\; \mathbf{l}_{3j} = \sqrt{2} \frac{\partial^2 \delta}{\partial x \partial y} \mathbf{e}_j.$$

Equation (4.3) can also be written as

$$J(\mathbf{f}) \quad = \quad \sum_i \left\langle \mathbf{f}, \mathbf{U}_i * \mathbf{f} \right\rangle$$

where $\mathbf{U}_i = \mathbf{l}_i * \mathbf{l}_i^T$. Substituting $\mathbf{U} = \sum_i \mathbf{U}_i$ yields

$$J(\mathbf{f}) = \left\langle \mathbf{f}, \mathbf{U} * \mathbf{f} \right\rangle \tag{4.5}$$

We call $\mathbf{U}$, the *convolutional operator* of J.

$J(\mathbf{f})$ is a semi-norm, since its kernel $\mathcal{K}$ is non-empty. One can easily see that, if $\mathcal{K}_i$ is the space of functions such that $\forall\, \mathbf{f} \in \mathcal{K}_i$, $\mathbf{L}_i^T * \mathbf{f}$ is zero almost everywhere, then $\mathcal{K} = \bigcap_i \mathcal{K}_i$.

The Fourier domain equivalent of (4.5) is

$$J(\mathbf{f}) = \int_{\mathbb{R}^2} \hat{\mathbf{f}}^\dagger(\boldsymbol{\omega}) \mathbf{U}(\boldsymbol{\omega}) \hat{\mathbf{f}}(\boldsymbol{\omega}) d\boldsymbol{\omega} \tag{4.6}$$

Note that writing J in the Fourier domain is not a complete definition, i.e., (4.6) is applicable only for functions whose Fourier transform is a true function. In particular, one cannot find $\mathcal{K}$ from (4.6) alone.

4.2.3 Rotationally Invariant Reconstruction

We start by defining the transformation of a vector field by an orthogonal matrix.

Definition 3 *Let $\mathbf{f}(\mathbf{x})$ be a vector field. Then the function $\mathbf{f}_{\boldsymbol{\Omega}}(\mathbf{x})$, associated with an orthogonal matrix $\boldsymbol{\Omega}$ is called the transformed vector field if it satisfies*

$$\mathbf{f}_{\boldsymbol{\Omega}}(\mathbf{x}) = \boldsymbol{\Omega} \mathbf{f}(\boldsymbol{\Omega}^T \mathbf{x}). \tag{4.7}$$

The function $\mathbf{f}_{\mathbf{\Omega}}(\mathbf{x})$ denotes the same vector field $\mathbf{f}(\mathbf{x})$ represented in a new coordinate system that is obtained by transforming the original one by an orthogonal matrix $\mathbf{\Omega}$.

Let $\mathbf{f}^{(\min)}$ be the function reconstructed from the data set $\{\mathbf{x}_j, \mathbf{d}_j, s_j\}$. Now, consider a transformed data set $\{\mathbf{\Omega}\mathbf{x}_j, \mathbf{\Omega}\mathbf{d}_j, s_j\}$. One would normally expect that the function reconstructed from this new data set should be $\mathbf{f}_{\mathbf{\Omega}}^{(\min)}$. The reconstruction method is called invariant to orthogonal transformation if this is true. The following proposition gives a sufficient condition for such an invariant reconstruction.

Proposition 1 *Let $\mathbf{f}^{(\min)}$ and $\mathbf{f}'^{(\min)}$ be the functions reconstructed from data sets $\{\mathbf{x}_j, \mathbf{d}_j, s_j\}$ and $\{\mathbf{\Omega}\mathbf{x}_j, \mathbf{\Omega}\mathbf{d}_j, s_j\}$ respectively, by minimizing (4.1). Then $\mathbf{f}'^{(\min)} = \mathbf{f}_{\mathbf{\Omega}}^{(\min)}$, if $J(\mathbf{f}) = J(\mathbf{f}_{\mathbf{\Omega}})$ for every $\mathbf{f}$.*

Proof: See appendix.

We say that the functional is invariant to orthogonal transformation if it satisfies the above relation. For brevity, we call such a functional "rotationally invariant", even though orthogonal transformation may represent not only a rotation but also a symmetry transformation. Rotational invariance implies that $\mathbf{f}$ and $\mathbf{f}_{\mathbf{\Omega}}$ should have the same cost, since they are physically the same.

Using (4.6), we deduce the corresponding condition on $\mathbf{U}(\boldsymbol{\omega})$:

$$\mathbf{U}(\boldsymbol{\omega}) = \mathbf{\Omega}^T \mathbf{U}(\mathbf{\Omega}\boldsymbol{\omega})\mathbf{\Omega} \tag{4.8}$$

4.2.4 Scale Invariant Reconstruction

Let us consider the situation where the samples locations undergo a scaling. Let $\{a\mathbf{x}_j, \mathbf{d}_j, s_j\}$ be the new scaled measurement set, where a is the scale factor. Let $\mathbf{f}^{(a)}$ be the function reconstructed from the measurement set scaled by a, which is defined as

$$\mathbf{f}^{(a)} = \underset{\mathbf{f}}{\operatorname{argmin}} \sum_{j}^{N} \left(\mathbf{d}_j^T \mathbf{f}(a\mathbf{x}_j) - s_j\right)^2 + \lambda(a) J(\mathbf{f}) \tag{4.9}$$

We say that the reconstruction is scale invariant if, there exists a continuous function $\lambda(a)$ such that $\mathbf{f}^{(a_1)}(\mathbf{x}) = \mathbf{f}^{(a_2)}((a_1/a_2)\mathbf{x})$, for every a_1 and a_2.

The meaning of scale invariance is that when we multiply the sample locations by a scale factor, then it should be possible to find a value for λ

such that the new solution is a scaled version of the previous solution. The new value of λ should be obtained from the previous value by multiplying with a function that depend only on the scale factor. The purpose of the scale invariance is to guarantee a predictable numerical behavior when there is an ambiguity in the unit of distance. The following proposition gives the condition for scale invariant reconstruction:

Proposition 2 *The reconstruction is scale invariant if J is scale invariant, i.e., $J(\mathbf{f}_a) = c(a)J(\mathbf{f})$, where $\mathbf{f}_a(\mathbf{x}) = \mathbf{f}(a\mathbf{x})$ and $c(a)$ is a continuous function on a.*

Proof: See appendix.
Using (4.6), above theorem implies

$$\mathbf{U}(\boldsymbol{\omega}) = \hat{c}(a)\mathbf{U}(a\boldsymbol{\omega}), \tag{4.10}$$

for some continuous function $\hat{c}(.)$.

In practice, scale invariance is often overly restrictive. We propose to replace the scale invariance condition with the so-called subspace scale invariance. To this end, we first suppose that the functional can be written as a sum of elementary functionals:

$$J(\mathbf{f}) = \sum_{i=1}^{N_J} \lambda_i J_i(\mathbf{f}) \tag{4.11}$$

Let the corresponding decomposition for $\mathbf{U}$ be

$$\mathbf{U} = \sum_{i=1}^{N_J} \lambda_i \mathbf{U}_i \tag{4.12}$$

The corresponding problem is

$$\mathbf{f}^{(a)} = \underset{\mathbf{f}}{\operatorname{argmin}} \sum_{j}^{N} \left(\mathbf{d}_j^T \mathbf{f}(a\mathbf{x}_j) - s_j\right)^2 + \lambda \left(\sum_{i=1}^{N_J} \lambda_i(a) J_i(\mathbf{f}) \right) \tag{4.13}$$

We say that the reconstruction is subspace scale invariant if there exists N_J continuous functions $(\lambda_i(a))_{i\in[1:N_J]}$ such that $\mathbf{f}^{(a_1)}(\mathbf{x}) = \mathbf{f}^{(a_2)}((a_1/a_2)\mathbf{x})$, for every a_1 and a_2. This implies that each of the elementary functionals is scale invariant; i.e., $J_i(\mathbf{f}_a) = c_i(a)J_i(\mathbf{f})$, or, equivalently, $\mathbf{U}_i(\boldsymbol{\omega}) = \hat{c}_i(a)\mathbf{U}_i(a\boldsymbol{\omega})$.

4.3 Imposing Invariances on U

In this section, we find the complete family of smoothness operators $\mathbf{U}$ that satisfy the rotational and subspace scale invariance properties. We first characterize the family that satisfies rotational invariance. We then identify a natural decomposition on the derived structure. Finally, we impose the subspace scale invariance on this decomposition.

We find it convenient to impose the invariances on the Fourier expression for $\mathbf{U}$. Hence we initially consider the action of $\mathbf{U}$ only on test functions (the functions that are indefinitely differentiable and of rapid descent). This restriction facilitates writing every partial derivative in the Fourier domain. Later in Section 4.3.3, we extend the semi-norm for a larger class of functions.

4.3.1 Rotational Invariance

The following theorem gives the structure for $\mathbf{U}$ that satisfies rotational invariance.

Theorem 2 *The Fourier kernel matrix* $\mathbf{U}$ *satisfies rotational invariance iff it can be expressed as*

$$\mathbf{U}(\boldsymbol{\omega}) = \alpha_d(\|\boldsymbol{\omega}\|)\boldsymbol{\omega}\boldsymbol{\omega}^T + \alpha_c(\|\boldsymbol{\omega}\|)(\|\boldsymbol{\omega}\|^2 \mathbf{I} - \boldsymbol{\omega}\boldsymbol{\omega}^T), \tag{4.14}$$

where $\alpha_d(.)$ *and* $\alpha_c(.)$ *are some real functions.*

Proof: See appendix.

The decomposition (4.14) has some interesting implications. Substituting (4.14) in (4.6) gives

$$J(\mathbf{f}) = J_d(\mathbf{f}) + J_c(\mathbf{f}) \tag{4.15}$$

where

$$J_d(\mathbf{f}) = \int_{\mathbb{R}^n} \hat{\mathbf{f}}^\dagger(\boldsymbol{\omega})\mathbf{U}_d(\boldsymbol{\omega})\hat{\mathbf{f}}(\boldsymbol{\omega})d\boldsymbol{\omega}, \tag{4.16}$$

$$J_c(\mathbf{f}) = \int_{\mathbb{R}^n} \hat{\mathbf{f}}^\dagger(\boldsymbol{\omega})\mathbf{U}_c(\boldsymbol{\omega})\hat{\mathbf{f}}(\boldsymbol{\omega})d\boldsymbol{\omega} \tag{4.17}$$

with $\mathbf{U}_d(\boldsymbol{\omega})$ and $\mathbf{U}_c(\boldsymbol{\omega})$ being given by

$$\mathbf{U}_d(\boldsymbol{\omega}) = \alpha_d(\|\boldsymbol{\omega}\|)\left(\boldsymbol{\omega}\boldsymbol{\omega}^T\right) \tag{4.18}$$

$$\mathbf{U}_c(\boldsymbol{\omega}) = \alpha_c(\|\boldsymbol{\omega}\|)\left(\|\boldsymbol{\omega}\|^2 \mathbf{I} - \boldsymbol{\omega}\boldsymbol{\omega}^T\right) \tag{4.19}$$

The following theorem gives an interpretation of the sub-functionals $J_d(\mathbf{f})$ and $J_c(\mathbf{f})$.

Theorem 3 *The functionals $J_d(\mathbf{f})$ and $J_c(\mathbf{f})$ act on the divergence and the curl of $\mathbf{f}$, i.e.,*

$$J_d(\mathbf{f}) = \int_{\mathbb{R}^n} \alpha_d(\|\boldsymbol{\omega}\|)\hat{f}^\dagger d(\boldsymbol{\omega})\hat{f}_d(\boldsymbol{\omega})d\boldsymbol{\omega}, \tag{4.20}$$

$$J_c(\mathbf{f}) = \int_{\mathbb{R}^n} \alpha_c(\|\boldsymbol{\omega}\|)\hat{\mathbf{f}}^\dagger d(\boldsymbol{\omega})\hat{\mathbf{f}}_c(\boldsymbol{\omega})c\boldsymbol{\omega}, \tag{4.21}$$

where $f_d = \operatorname{div}\mathbf{f}$ and $\mathbf{f}_c = \operatorname{rot}\mathbf{f}$.

Note that $\mathbf{f}_c$ is an $n_c \times 1$ vector, where $n_c = 3$ if $n = 3$, and $n_c = 1$ if $n = 2$. It is well known from vector calculus that any vector field $\mathbf{f}$ can be expressed as $\mathbf{f} = \mathbf{f}_{\text{irr}} + \mathbf{f}_{\text{sol}}$, where $\mathbf{f}_{\text{irr}}$ and $\mathbf{f}_{\text{sol}}$ are some vector fields such that $\operatorname{div}\mathbf{f}_{\text{sol}} = 0$ and $\operatorname{rot}\mathbf{f}_{\text{irr}} = 0$. Consequently,

$$J(\mathbf{f}) = J_d(\mathbf{f}_{\text{irr}}) + J_c(\mathbf{f}_{\text{sol}}).$$

$\mathbf{f}_{\text{irr}}$ is known as the irrotational component and $\mathbf{f}_{\text{sol}}$ is known as the solenoidal component.

4.3.2 Subspace Scale Invariance

For imposing subspace scale invariance, we need to have a decomposition of J. We choose the decomposition (4.15), since it has a physical relevance. We intend to impose the following relations:

$$\mathbf{U}_d(a\boldsymbol{\omega}) = c_d(a)\mathbf{U}_d(\boldsymbol{\omega}), \tag{4.22}$$

$$\mathbf{U}_c(a\boldsymbol{\omega}) = c_c(a)\mathbf{U}_c(\boldsymbol{\omega}), \tag{4.23}$$

where $c_d(a)$ and $c_c(a)$ are some continuous functions. Substituting (4.18) and (4.19) in (4.22) and (4.23) yields

$$\alpha_d(a\,\|\boldsymbol{\omega}\|)a^2 = c_d(a)\alpha_d(\|\boldsymbol{\omega}\|),$$

$$\alpha_c(a\,\|\boldsymbol{\omega}\|)a^2 = c_c(a)\alpha_c(\|\boldsymbol{\omega}\|).$$

This implies that the functions $\alpha_d(\|\boldsymbol{\omega}\|)$ and $\alpha_c(\|\boldsymbol{\omega}\|)$ are necessarily of the following form,

$$\alpha_d(\|\boldsymbol{\omega}\|) = \lambda_d \|\boldsymbol{\omega}\|^{2\gamma_d},$$

$$\alpha_c(\|\boldsymbol{\omega}\|) = \lambda_c \|\boldsymbol{\omega}\|^{2\gamma_c},$$

where $\gamma_d, \gamma_c, \lambda_d$, and λ_c are some real numbers.

The Fourier kernel matrix is now given by

$$\mathbf{U}(\boldsymbol{\omega}) = \lambda_d \mathbf{U}_d(\boldsymbol{\omega}) + \lambda_c \mathbf{U}_c(\boldsymbol{\omega}) \tag{4.24}$$

where

$$\mathbf{U}_d(\boldsymbol{\omega}) = \|\boldsymbol{\omega}\|^{2\gamma_d} \boldsymbol{\omega}\boldsymbol{\omega}^T \tag{4.25}$$

$$\mathbf{U}_c(\boldsymbol{\omega}) = \|\boldsymbol{\omega}\|^{2\gamma_c} (\|\boldsymbol{\omega}\|^2 \mathbf{I} - \boldsymbol{\omega}\boldsymbol{\omega}^T) \tag{4.26}$$

4.3.3 Specifying $J_d(\mathbf{f})$ and $J_c(\mathbf{f})$ in Spatial Domain

The functionals J_d and J_c defined by the equations (4.16) and (4.17) are now given by

$$J_d(\mathbf{f}, \gamma_d) = \int_{\mathbb{R}^n} \|\boldsymbol{\omega}\|^{2\gamma_d} \hat{\mathbf{f}}^\dagger(\boldsymbol{\omega}) \left(\boldsymbol{\omega}\boldsymbol{\omega}^T\right) \hat{\mathbf{f}}(\boldsymbol{\omega}) d\boldsymbol{\omega}, \tag{4.27}$$

$$J_c(\mathbf{f}, \gamma_c) = \int_{\mathbb{R}^n} \|\boldsymbol{\omega}\|^{2\gamma_c} \hat{\mathbf{f}}^\dagger(\boldsymbol{\omega}) (\|\boldsymbol{\omega}\|^2 \mathbf{I} - \boldsymbol{\omega}\boldsymbol{\omega}^T) \hat{\mathbf{f}}(\boldsymbol{\omega}) d\boldsymbol{\omega}, \tag{4.28}$$

By Theorem 3, Equations (4.27) and (4.28) can also be written as

$$J_d(\mathbf{f}, \gamma_d) = \int_{\mathbb{R}^n} \|\boldsymbol{\omega}\|^{2\gamma_d} \hat{f}_d^\dagger(\boldsymbol{\omega}) \hat{f}_d(\boldsymbol{\omega}) d\boldsymbol{\omega}, \tag{4.29}$$

$$J_c(\mathbf{f}, \gamma_c) = \int_{\mathbb{R}^n} \|\boldsymbol{\omega}\|^{2\gamma_c} \hat{\mathbf{f}}_c^\dagger(\boldsymbol{\omega}) \hat{\mathbf{f}}_c(\boldsymbol{\omega}) d\boldsymbol{\omega} \tag{4.30}$$

Now, we intend to write (4.27) and (4.28) partly is space domain. To this end, we first define the following:

$$\begin{aligned} m_d &= \lfloor \gamma_d \rfloor \\ m_c &= \lfloor \gamma_c \rfloor \\ \gamma_d' &= \gamma_d - m_d \\ \gamma_c' &= \gamma_c - m_c \end{aligned}$$

We now consider J_d:

$$\begin{aligned} J_d(\mathbf{f}, \gamma_d) &= \int_{\mathbb{R}^n} \|\boldsymbol{\omega}\|^{2\gamma_d'} \left\|\mathcal{F}\left(\mathbf{h}_{m_d} f_d\right)\right\|^2 d\boldsymbol{\omega}, \\ &= \int_{\mathbb{R}^n} \|\boldsymbol{\omega}\|^{2\gamma_d'} \left\|\mathcal{F}\left(\mathbf{h}_{m_d} (\nabla \cdot \mathbf{f})\right)\right\|^2 d\boldsymbol{\omega}, \end{aligned} \tag{4.31}$$

where $\mathbf{h}_{m_d}$ is the Duchon's operator of order m_d.

Note that this new version is applicable for a more general class of functions; the expression is valid for the function for which $\mathcal{F}(\mathbf{h}_{m_d}(\nabla \cdot \mathbf{f}))$ is a true function. This is clearly less restrictive than (4.27). Note that if f_d has a well-defined Fourier transform, then

$$\|\mathcal{F}(\mathbf{h}_{m_d}(\nabla \cdot \mathbf{f}))\|^2 = \|\boldsymbol{\omega}\|^{2m_d} \hat{f}_d^{\dagger}(\boldsymbol{\omega}) \hat{f}_d(\boldsymbol{\omega}),$$

and hence (4.31) is equivalent to (4.29).

Now, we extend J_c in a similar way:

$$\begin{aligned} J_c(\mathbf{f}, \gamma_c) &= \int_{\mathbb{R}^n} \|\boldsymbol{\omega}\|^{2\gamma_c} \hat{\mathbf{f}}_d^{\dagger}(\boldsymbol{\omega}) \hat{\mathbf{f}}_c(\boldsymbol{\omega}) d\boldsymbol{\omega} \\ &= \sum_{i=1}^{n_c} \int_{\mathbb{R}^n} \|\boldsymbol{\omega}\|^{2\gamma_c} \left| \{\hat{\mathbf{f}}_c(\boldsymbol{\omega})\}_i \right|^2 d\boldsymbol{\omega} \\ &= \sum_{i=1}^{n_c} \int_{\mathbb{R}^n} \|\boldsymbol{\omega}\|^{2\gamma_c'} \|\mathcal{F}(\mathbf{h}_{m_c}\{\mathbf{f}_c\}_i)\|^2 d\boldsymbol{\omega}, \\ &= \sum_{i=1}^{n_c} \int_{\mathbb{R}^n} \|\boldsymbol{\omega}\|^{2\gamma_c'} \|\mathcal{F}(\mathbf{h}_{m_c}\{\nabla \times \mathbf{f}\}_i)\|^2 d\boldsymbol{\omega}. \end{aligned} \tag{4.32}$$

Note that when γ_d and γ_c are integers; i.e., when γ_d' and γ_c' are zero, then (4.31) and (4.32) have the equivalent space domain expressions

$$J_d(\mathbf{f}, m_d) = \int_{\mathbb{R}^n} \|(\mathbf{h}_{m_d}(\nabla \cdot \mathbf{f})\|^2 d\mathbf{x}, \tag{4.33}$$

$$J_c(\mathbf{f}, m_c) = \sum_{i=1}^{n_c} \int_{\mathbb{R}^n} \|\mathbf{h}_{m_c}\{\nabla \times \mathbf{f}\}_i\|^2 d\mathbf{x}. \tag{4.34}$$

Now, note that (4.27), (4.29), and (4.31) are equivalent for test functions. Similarly, (4.28), (4.30), and (4.32) are equivalent. To summarize, our final semi-norm is given by

$$J(\mathbf{f}) = \lambda_d J_d(\mathbf{f}, \gamma_d) + \lambda_c J_c(\mathbf{f}, \gamma_c), \tag{4.35}$$

where J_d and J_c are of the form specified by (4.31) and (4.32).

4.3.4 Physical Interpretation of $J_d(\mathbf{f})$ and $J_c(\mathbf{f})$

In continuum mechanics, the divergence of the velocity field quantifies the rate change of the density of the medium at a given point; the curl of a velocity field at a point is equal to the twice the angular velocity within an

infinitesimal neighborhood. In electromagnetics, the divergence of an electric field is equal to the electric charge density, whereas the curl is equal to the rate of change of the associated magnetic field, if present. When applied to the magnetic field, the divergence is always zero, whereas its curl is equal to the current density.

First, we will consider the effect of $J_d(\mathbf{f})$, which imposes a γ_d-order smoothness on the divergence of $\mathbf{f}$. One can incorporate an a priori knowledge about the vector field by choosing $\gamma_d \to 0$ and $\lambda_d \to \infty$, which is equivalent to specifying a divergence-free solution, which is functionally equivalent to the method provided in [51]. In the case of velocity field approximation, the corresponding a priori knowledge is the incompressibility of the medium. This setting is also an appropriate one for computing a magnetic field or current density distribution, since such fields are divergence-free.

When the required vector is not divergence-free, choosing a finite value of λ_d is equivalent to specifying some family of preferential solutions. For example, in the case of velocity field reconstruction, choosing $\gamma_d \to 0$ is equivalent to minimizing the overall compression of the medium, whereas a finite value of γ_d will specify a γ_d-order smoothness on the density (mass) distribution of the medium. The later, for an electric field, will specify a γ_d-order smoothness on the electric charge density.

Next, let us consider the effect of $J_c(\mathbf{f})$, which imposes a γ_c-order smoothness on the each component of the curl of $\mathbf{f}$. Unlike $J_d(\mathbf{f})$, the choice $\gamma_c \to 0$ and $\lambda_c \to \infty$ is not of practical interest for vectors fields such as velocity field, magnetic field, and current field, since it will specify a trivial solution. However, this setting should be chosen for an electric field, since the latter is rotation-free when there is no time varying magnetic field (Faraday's law). Choosing finite values for γ_c and λ_c is again equivalent to specifying some convex set of preferential solutions. For example, in the case of a velocity field, this choice will minimize the overall deformation, since spatially-varying rotation (curl) contributes to deformation.

The above arguments only give the physical relevance in choosing the parameters. One also needs to ensure that the minimization problem is well-defined for a particular choice.

4.4 The Minimizations Space $\mathcal{V}$

4.4.1 Imposing Constraints on $\mathcal{V}$

Our goal now is to specify the minimization space in such a way that the reconstructing kernel $\mathcal{K}_f$ is fairly small. This will ensure that only simple functions, such as low order polynomials, should contribute a zero cost to the functional J. Also, this will guarantee the uniqueness of the solution especially when the data size is small.

We already have a constraint on $\mathcal{V}$: $J(f) < \infty$. However, this condition is not sufficiently restrictive. Indeed, we can construct a large class of functions for which $\nabla \cdot \mathbf{f} = 0$ and $\nabla \times \mathbf{f} = \mathbf{0}$. For example, consider the following 2D function:

$$\mathbf{f}(x, y) = \nabla\phi(x, y), \quad \phi(x, y) = Re\{(x + i.y)^k\}, \tag{4.36}$$

where Re stands for real part of a complex function. First, by construction, $\nabla \times \mathbf{f} = \mathbf{0}$. Second, $\phi(x, y)$ belongs to the class of analytic functions, which satisfy $\triangle\phi = 0$. Hence $\nabla \cdot \mathbf{f} = \triangle\phi = 0$. Hence (4.36) gives an example for the kernel of J, which is valid for any integer k. In other words, there is an infinite number of such functions. This shows that the kernel is too large, which calls for some further restriction on $\mathcal{V}$.

To this end, we will identify an interesting relationship for the sum of J_d and J_c that holds for test functions. Then, we will designate the minimization space as the space of functions that satisfy this property.

Substituting $\gamma_d = \gamma_c = m$(integer) in (4.27) and (4.28) and adding them yields

$$J_d(\mathbf{f}, m) + J_c(\mathbf{f}, m) = \int_{\mathbb{R}^n} \|\boldsymbol{\omega}\|^{2m+2} \hat{\mathbf{f}}^\dagger(\boldsymbol{\omega})\hat{\mathbf{f}}(\boldsymbol{\omega})d\boldsymbol{\omega} \tag{4.37}$$

In other words, for the class of test functions, the above sum is equal to the Duchon's norm applied to each component independently. Next, using (4.33) and (4.34) we write (4.37) in space domain as follows:

$$\int \|(\mathbf{h}_m(\nabla \cdot \mathbf{f})\|^2 \, d\mathbf{x} + \sum_{i=1}^{n_c} \int \|\mathbf{h}_m\{\nabla \times \mathbf{f}\}_i\|^2 \, d\mathbf{x} = \sum_{i=1}^{n} \int \|\mathbf{h}_{m+1}\{\mathbf{f}\}_i\|^2 \, d\mathbf{x} \tag{4.38}$$

Now, we are ready to define the minimization space:

Definition 4 *The minimization space $\mathcal{V}$ is the space of functions such that $J(f)$ is finite, and (4.38) is satisfied for $m = \max(m_d, m_c)$.*

Note that the first condition ensures that $J_d(\mathbf{f}, \gamma_d)$ and $J_c(\mathbf{f}, \gamma_c)$ given in the equations (4.31) and (4.32) are well defined, and the second ensures that the reconstructing kernel $\mathcal{K}_f$ is fairly small. In particular, it ensures that the dimension of the kernel is finite.

4.4.2 The Kernel $\mathcal{K}_f$

The kernel $\mathcal{K}_f$ is the subspace of functions that satisfy $J(\mathbf{f}) = 0$ as well as equation (4.38). First, note that $J(\mathbf{f})$ is zero only when $J_d(\mathbf{f})$ and $J_c(\mathbf{f})$ are simultaneously zero. Further, note that the solution of the equations $J_d(\mathbf{f}) = 0$ and $J_c(\mathbf{f}) = 0$ is determined only by the integers m_d and m_c. Hence $\mathcal{K}_f$ is included in the simultaneous solution of the equations

$$J_d(\mathbf{f}, m_d) = 0 \tag{4.39}$$

$$J_c(\mathbf{f}, m_c) = 0 \tag{4.40}$$

Substituting (4.33) and (4.34) yields

$$\int \left\| (\mathbf{h}_{m_d}(\nabla \cdot \mathbf{f}) \right\|^2 d\mathbf{x} = 0 \tag{4.41}$$

$$\sum_{i=1}^{n_c} \int \left\| \mathbf{h}_{m_c}\{\nabla \times \mathbf{f}\}_i \right\|^2 d\mathbf{x} = 0 \tag{4.42}$$

The kernel determination problem now boils down to solving (4.41), (4.42), and (4.38) simultaneously. Now, we consider two cases: (i) $m_c \geq m_d$; (ii) and $m_c < m_d$. We solve for each case individually.

Case (i): $m_c \geq m_d$

We first substitute (4.38) in (4.42) with $m = m_c$:

$$\sum_{i=1}^{n} \int \left\| \mathbf{h}_{m_c+1}\{\mathbf{f}\}_i \right\|^2 d\mathbf{x} - \int \left\| (\mathbf{h}_{m_c}(\nabla \cdot \mathbf{f}) \right\|^2 d\mathbf{x} = 0 \tag{4.43}$$

Now, we need to solve (4.43) and (4.41). To this end, we first observe that if $\int \left\| (\mathbf{h}_{m_d}(\nabla \cdot \mathbf{f}) \right\|^2 d\mathbf{x}$ is zero, then $\int \left\| (\mathbf{h}_{m_c}(\nabla \cdot \mathbf{f}) \right\|^2 d\mathbf{x}$ is automatically zero, since $m_c \geq m_d$. Consequently, it only remains to solve,

$$\sum_{i=1}^{n} \int \left\| \mathbf{h}_{m_c+1}\{\mathbf{f}\}_i \right\|^2 d\mathbf{x} = 0, \tag{4.44}$$

together with (4.41). We know that the solution space of (4.44) is $\mathbb{P}_{m_c}(\mathbb{R}^n;\mathbb{R}^n)$. Hence the required kernel is a subspace of $\mathbb{P}_{m_c}(\mathbb{R}^n;\mathbb{R}^n)$ that satisfies (4.41). This subspace can be determined as follows:

Let B be the number of basis functions for $\mathbb{P}_{m_c}(\mathbb{R}^n)$ and let $\boldsymbol{\zeta}(\mathbf{x})$ be the vector of polynomials that form a basis for $\mathbb{P}_{m_c}(\mathbb{R}^n)$. Then any function in $\mathbb{P}_{m_c}(\mathbb{R}^n)$ can be expressed as $\mathbf{v}^T\boldsymbol{\zeta}(\mathbf{x})$ for some $\mathbf{v} \in \mathbb{R}^B$. Now, consider a function $\mathbf{f} \in \mathbb{P}_{m_c}(\mathbb{R}^n;\mathbb{R}^n)$. There exist vectors $\{\mathbf{v}_i, i = 1, \cdots, n\} \in \mathbb{R}^B$ such that

$$\{\mathbf{f}(\mathbf{x})\}_i = \mathbf{v}_i^T\boldsymbol{\zeta}(\mathbf{x}).$$

Since, $\nabla \cdot \mathbf{f}$ is also in $\mathbb{P}_{m_c}(\mathbb{R}^n)$, there exist some $\mathbf{v}_d \in \mathbb{R}^B$ such that $\nabla \cdot \mathbf{f} = \mathbf{v}_d^T\boldsymbol{\zeta}(\mathbf{x})$. Since $\nabla\cdot$ is a linear operator, the vector $\mathbf{v}_d$ can be expressed as

$$\mathbf{v}_d = \mathbf{M}_{\text{div}}\mathbf{u},$$

where $\mathbf{u} = [\cdots \ \mathbf{v}_i^T \ \cdots]^T$ and $\mathbf{M}_{\text{div}}$ is some $B \times nB$ matrix. Now, (4.41) implies that each partial derivative of $\nabla \cdot \mathbf{f}$ with order m_d is zero. Let $\mathbf{t}_i$ be a vector such that $\{\mathbf{h}_{m_d}\text{div}\,\mathbf{f}\}_i = \mathbf{t}_i^T\boldsymbol{\zeta}(\mathbf{x})$. This vector is then expressed as

$$\mathbf{t}_i = \mathbf{D}_i\mathbf{M}_{\text{div}}\mathbf{u},$$

where $\mathbf{D}_i$ is matrix representing the corresponding partial derivative. Now let

$$\mathbf{M} = \begin{bmatrix} \vdots \\ \mathbf{D}_i \\ \vdots \end{bmatrix} \mathbf{M}_{\text{div}}$$

This means that the kernel $\mathcal{K}_f$ is determined by the null vectors of $\mathbf{M}$. In other words, there is a one-to-one correspondence between the kernel of $\mathbf{M}$ and the kernel $\mathcal{K}_f$ as follows: for any vector $\mathbf{n}$ such that $\mathbf{Mn} = \mathbf{0}$, the function $\mathbf{f}(\mathbf{x})$ defined by

$$\{\mathbf{f}(\mathbf{x})\}_i = \mathbf{n}_i^T\boldsymbol{\zeta}(\mathbf{x})$$

is an element of $\mathcal{K}_f$, where $\mathbf{n}_i$ is the ith block of $\mathbf{n}$. Hence the problem of determining $\mathcal{K}_f$ boils down to the task of computing the kernel of $\mathbf{M}$, which can by done by standard numerical methods.

Note that when $m_d = 0$, $\mathcal{K}_f$ is the divergence-free subspace of $\mathbb{P}_{m_c}(\mathbb{R}^n;\mathbb{R}^n)$, which is the same as the kernel of the divergence-free reconstruction problem considered in [51]. According to these authors, it is equal

to $\operatorname{Rot}\mathbb{P}_{m_c+1}(\mathbb{R}^n;\mathbb{R}^n)$ where $\operatorname{Rot}\mathbb{P}_{m_c+1}(\mathbb{R}^n;\mathbb{R}^n)$ is defined as the space of all $\mathbf{f}$ such that there exists some $\mathbf{g}$ in $\mathbb{P}_{m_c+1}(\mathbb{R}^n;\mathbb{R}^n)$ meeting $\mathbf{f} = \nabla \times \mathbf{g}$.

Case (ii): $m_c < m_d$

In this case, we substitute (4.38) in (4.41) with $m = m_d$:

$$\sum_{i=1}^{n} \int \|\mathbf{h}_{m_d+1}\{\mathbf{f}\}_i\|^2 \, d\mathbf{x} - \sum_{i=1}^{n_c} \int \|\mathbf{h}_{m_c}\{\nabla \times \mathbf{f}\}_i\|^2 \, d\mathbf{x} = 0 \tag{4.45}$$

The kernel determination problem is now to solve (4.45) and (4.42). Using similar arguments as in the previous case, the problem is equivalent to solve the following equation together with (4.42):

$$\sum_{i=1}^{n} \int \|\mathbf{h}_{m_d+1}\{\mathbf{f}\}_i\|^2 \, d\mathbf{x} = 0 \tag{4.46}$$

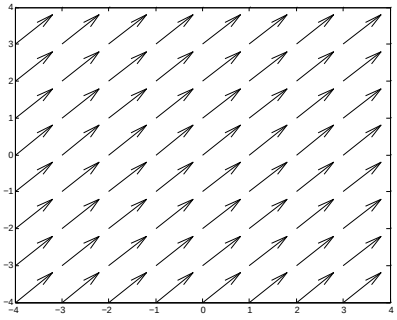
Consequently, the kernel is a subspace of $\mathbb{P}_{m_d}(\mathbb{R}^n;\mathbb{R}^n)$ that satisfies (4.42). This subspace can be computed numerically in a way similar to the previous case.

As in the previous case, when $m_c = 0$, $\mathcal{K}_f$ is the rotation-free subspace of $\mathbb{P}_{m_d}(\mathbb{R}^n;\mathbb{R}^n)$, which is the same as the kernel of the rotation-free reconstruction problem addressed in [51]. It is equal to $\nabla\mathbb{P}_{m_c+1}(\mathbb{R}^n)$ where $\nabla\mathbb{P}_{m_c+1}(\mathbb{R}^n)$ is defined as the space of all $\mathbf{f}$ such that there exists some g in $\mathbb{P}_{m_c+1}(\mathbb{R}^n)$ meeting $\mathbf{f} = \nabla g$.

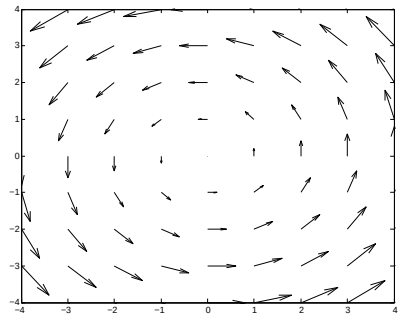
To illustrate these various concepts, we consider a concrete 2D example for the semi-norm and its corresponding kernel $\mathbf{K}_f$. For the choice $m_d = 0$ and $m_c = 1$, the kernel is spanned by the following vector polynomials:

$$\begin{aligned}
\mathbf{n}_1(x,y) &= [1\ 0]^T \\
\mathbf{n}_2(x,y) &= [0\ 1]^T \\
\mathbf{n}_3(x,y) &= [-y\ x]^T \\
\mathbf{n}_4(x,y) &= [-x\ y]^T \\
\mathbf{n}_5(x,y) &= [y\ x]^T
\end{aligned}$$

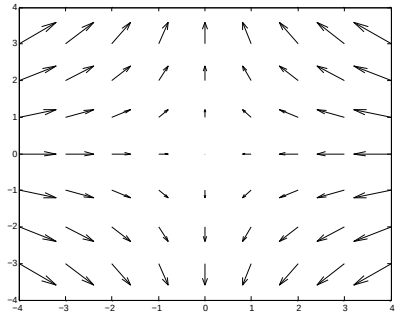
Figure (4.1) provide a visualization of these vector fields in terms of arrows. Let us now provide the physical interpretation assuming that the vector field that is under consideration is a velocity field. The vectors $\mathbf{n}_1$, $\mathbf{n}_2$, and $\mathbf{n}_3$ represent rigid forms of motion: $\mathbf{n}_1$, $\mathbf{n}_2$ represent translational motion, whereas



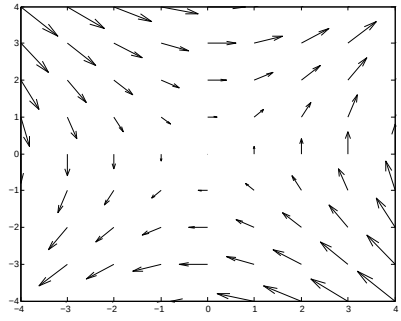
(a) $\mathbf{n}_1(x, y) + \mathbf{n}_2(x, y)$



(b) $\mathbf{n}_3(x, y)$



(c) $\mathbf{n}_4(x, y)$



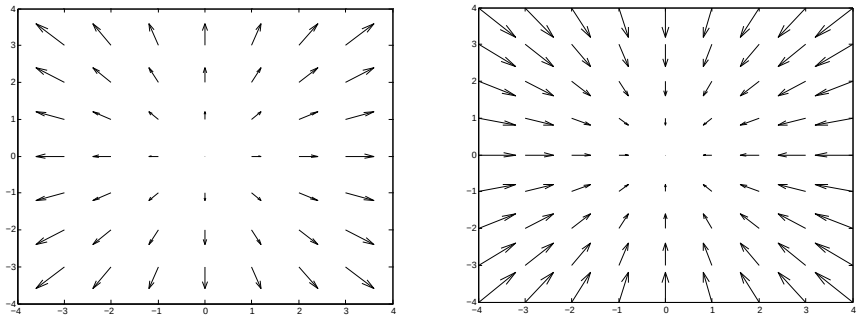
(d) $\mathbf{n}_5(x, y)$

Figure 4.1: 2D kernel vector fields for $m_d = 0$ and $m_c = 1$

$\mathbf{n}_3$ represents rotational motion. $\mathbf{n}_1$ and $\mathbf{n}_2$ fall under the category of laminar flow, but they are trivial forms of laminar flow. $\mathbf{n}_4$ and $\mathbf{n}_5$ are non-trivial forms of laminar flow; each of them represent two in-flows and two out-flows around the origin. There is only one vector that is in $\mathbb{P}_1(\mathbb{R}^2;\mathbb{R}^2)$, but not in $\mathcal{K}_f$:

$$\mathbf{n}_6(x, y) = [x\ y]^T.$$

Figure (4.2) gives its visualization. This flow field can represent, for example,



(a) $\mathbf{n}_6(x, y)$ (b) $-\mathbf{n}_6(x, y)$

Figure 4.2: 2D first order polynomial vector field representing source or sink

the existence of a source or sink. Alternatively, if present in a small neighborhood, it may indicate some compression or rarefaction of the medium that undergoes motion. This will normally correspond to some internal energy (other than kinetic energy) of the system. By choosing $m_d = 0$ and $m_c = 1$, it becomes possible to penalize such motion. Note that if m_d and m_c are constrained to be equal (which corresponds to [49, 50]), there is no possibility to penalize such a motion alone. However, such a motion can be eliminated completely by the divergence-free method proposed in [51].

4.5 The Solution: Vector Splines

4.5.1 The Reconstruction Formula

We now give the solution for the reconstruction problem. The solution is constructed using the Green's function of the operator $\mathbf{U}$. It is defined as the $n \times n$ matrix function such that $\mathbf{\Psi} * \mathbf{U} = \delta\mathbf{I}$. The following theorem gives the solution based on $\mathbf{\Psi}$.

Theorem 4 *The minimum of the functional (4.1) is given by*

$$\mathbf{f}_{opt}(\mathbf{x}) = \sum_{i=1}^{N} w_i \boldsymbol{\varphi}_{\mathbf{d_i}}(\mathbf{x} - \mathbf{x}_i) + \sum_{k=1}^{q} a_k \mathbf{p}_k(\mathbf{x}) \tag{4.47}$$

where $\boldsymbol{\varphi}_{\mathbf{d_i}}(\mathbf{x}) = \mathbf{\Psi}(\mathbf{x})\mathbf{d}_i$, *and* $(\mathbf{p}_k)_{k\in[1:q]}$ *is a basis for the kernel of* J. *The weight vectors* $\mathbf{w} = [\ldots\ w_i\ \ldots]^T$ *and* $\mathbf{a} = [\ldots\ a_i\ \ldots]^T$ *are given by*

$$\begin{bmatrix} \mathbf{A} + \lambda\mathbf{I} & \mathbf{Q} \\ \mathbf{Q}^T & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{w} \\ \mathbf{a} \end{bmatrix} = \begin{bmatrix} \mathbf{s} \\ \mathbf{0} \end{bmatrix} \tag{4.48}$$

where

$$\begin{aligned} \{\mathbf{A}\}_{ij} &= \mathbf{d}_j^T \boldsymbol{\varphi}_{\mathbf{d_i}}(\mathbf{x}_j - \mathbf{x}_i),\ \ \forall\, i, j \in [1:N] \\ \{\mathbf{Q}\}_{ij} &= \mathbf{d}_j^T \mathbf{p}_i(\mathbf{x}_j),\ \ \forall\, i \in [1:N|;\ \ \forall\, j \in [1:q]. \\ \mathbf{s} &= [\ldots\ s_i\ \ldots]^T. \end{aligned}$$

Proof: See appendix.

Note that the solution is the weighted sum of basis functions placed at each sampling locations. Moreover, each basis function is derived from the Green's function by making a weighted sum of its columns; the weights are simply specified by the projection vector $\mathbf{d}_i$, and are dependent upon the sampling direction.

This theorem is compatible with the vector spline methods of Amodei [49], and more recently, Rabut et al [50]. However, it provides two important generalizations: (i) the reconstruction from projected data ($\mathbf{d}_i \neq \mathbf{e}_i$); (ii) and a more general class of regularization including fractional derivatives.

4.5.2 The Vector Splines

We have shown so far that the solution to the variational problem is a weighted sum of basis functions of the form

$$\boldsymbol{\varphi}_{\mathbf{d}_i}(\mathbf{x}) = \boldsymbol{\Psi}(\mathbf{x})\mathbf{d}_i, \tag{4.49}$$

located at the sample points, where is the $\boldsymbol{\Psi}(\mathbf{x})$ is the Green's function, and $\mathbf{d}_i$ is the measurement operator. The Green's function is obtained by solving the equation $\boldsymbol{\Psi} * \mathbf{U} = \delta\mathbf{I}$. Because of (4.8), $\boldsymbol{\Psi}(\mathbf{x})$ satisfies

$$\boldsymbol{\Psi}(\mathbf{x}) = \boldsymbol{\Omega}^T\boldsymbol{\Psi}(\boldsymbol{\Omega}\mathbf{x})\boldsymbol{\Omega} \tag{4.50}$$

To find the implication of (4.50) on $\boldsymbol{\varphi}_{\mathbf{d}_i}(\mathbf{x})$, we substitute in (4.49):

$$\begin{aligned}\boldsymbol{\varphi}_{\mathbf{d}_i}(\mathbf{x}) &= \boldsymbol{\Omega}^T\boldsymbol{\Psi}(\boldsymbol{\Omega}\mathbf{x})\boldsymbol{\Omega}\mathbf{d}_i \\ &= \boldsymbol{\Omega}^T\boldsymbol{\varphi}_{\boldsymbol{\Omega}\mathbf{d}_i}(\boldsymbol{\Omega}\mathbf{x})\end{aligned}$$

Now, we provide the expression for the Green's function $\boldsymbol{\Psi}$ in the following theorem:

Theorem 5 *The Green's function $\boldsymbol{\Psi}$ for the operator $\mathbf{U}$ expressed in (4.24) is given by*

$$\begin{aligned}\boldsymbol{\Psi}(\mathbf{x}) = &\frac{1}{\lambda_d}\psi_{l_1}(\|\mathbf{x}\|) * \mathbf{D}(\mathbf{x}) \\ &+ \frac{1}{\lambda_c}\psi_{l_2}(\|\mathbf{x}\|) * (L(\mathbf{x})\mathbf{I} - \mathbf{D}(\mathbf{x}))\end{aligned} \tag{4.51}$$

where

$$L(\mathbf{x}) = \sum_{i=1}^{n}\frac{\partial^2}{\partial x_i^2}\delta,$$

$$\{\mathbf{D}(\mathbf{x})\}_{ij} = \frac{\partial^2}{\partial x_i \partial x_j}\delta,$$

$$\psi_l(r) = \begin{cases} r^l \log r, & \textit{if } l \textit{ is an even integer,} \\ r^l, & \textit{otherwise,} \end{cases} \tag{4.52}$$

and

$$l_1 = 2\gamma_d + 4 - n, \quad l_2 = 2\gamma_c + 4 - n.$$

Proof: See appendix.

Note that the Green's function is the sum of two distinct parts, i.e.,

$$\mathbf{\Psi}(\mathbf{x}) = \frac{1}{\lambda_d}\mathbf{\Psi}^{(d)}(\mathbf{x}) + \frac{1}{\lambda_c}\mathbf{\Psi}^{(c)}(\mathbf{x}), \tag{4.53}$$

where

$$\mathbf{\Psi}^{(d)}(\mathbf{x}) = \psi_{l_1}(\|\mathbf{x}\|) * \mathbf{D}(\mathbf{x}) \tag{4.54}$$

$$\mathbf{\Psi}^{(c)}(\mathbf{x}) = \psi_{l_2}(\|\mathbf{x}\|) * (L(\mathbf{x})\mathbf{I} - \mathbf{D}(\mathbf{x})) \tag{4.55}$$

Let $\boldsymbol{\psi}_i^{(d)}(\mathbf{x})$ and $\boldsymbol{\psi}_i^{(c)}(\mathbf{x})$ denote the ith columns of $\mathbf{\Psi}^{(d)}(\mathbf{x})$ and $\mathbf{\Psi}^{(c)}(\mathbf{x})$ respectively. The following theorem gives an interesting property of these functions.

Theorem 6 *The functions $\boldsymbol{\psi}_i^{(d)}(\mathbf{x})$ and $\boldsymbol{\psi}_i^{(c)}(\mathbf{x})$ satisfy the following property:*

$$\begin{aligned} \operatorname{rot} \boldsymbol{\psi}_j^{(d)}(\mathbf{x}) &= 0, \quad \forall\, j \in [1:n] \\ \operatorname{div} \boldsymbol{\psi}_j^{(c)}(\mathbf{x}) &= 0, \quad \forall\, j \in [1:n] \end{aligned}$$

Proof: See appendix.

In other words, $\boldsymbol{\psi}_i^{(d)}(\mathbf{x})$s are irrotational and $\boldsymbol{\psi}_i^{(c)}(\mathbf{x})$s are solenoidal. Now, note that the basis functions $\boldsymbol{\varphi}_{\mathbf{d}_i}$ can also be expressed as

$$\boldsymbol{\varphi}_{\mathbf{d}_i}(\mathbf{x}) = \frac{1}{\lambda_d}\boldsymbol{\varphi}_{\mathbf{d}_i}^{(d)}(\mathbf{x}) + \frac{1}{\lambda_c}\boldsymbol{\varphi}_{\mathbf{d}_i}^{(c)}(\mathbf{x}) \tag{4.56}$$

where

$$\boldsymbol{\varphi}_{\mathbf{d}_i}^{(d)}(\mathbf{x}) = \mathbf{\Psi}^{(d)}(\mathbf{x})\mathbf{d}_i \tag{4.57}$$

$$\boldsymbol{\varphi}_{\mathbf{d}_i}^{(c)}(\mathbf{x}) = \mathbf{\Psi}^{(c)}(\mathbf{x})\mathbf{d}_i \tag{4.58}$$

As a consequence of the theorem 6,

$$\begin{aligned} \operatorname{rot} \boldsymbol{\varphi}_{\mathbf{d}_i}^{(d)}(\mathbf{x}) &= 0 \\ \operatorname{div} \boldsymbol{\varphi}_{\mathbf{d}_i}^{(c)}(\mathbf{x}) &= 0. \end{aligned}$$

In other words, each of the basis functions can be expressed as a sum of its irrotational and solenoidal components.

Because of the special structure of the functions, $\mathbf{\Psi}^{(d)}$ and $\mathbf{\Psi}^{(c)}$, their columns $\boldsymbol{\psi}_i^{(d)}$ and $\boldsymbol{\psi}_i^{(c)}$ are related, as stated below.

Theorem 7 *The vector functions* $\boldsymbol{\psi}_i^{(d)}$ *and* $\boldsymbol{\psi}_i^{(c)}$ *satisfy the relation*

$$\boldsymbol{\psi}_{k+1}^{(d)}(\mathbf{x}) = \mathbf{P}\boldsymbol{\psi}_k^{(d)}(\mathbf{P}^T\mathbf{x}), \quad \boldsymbol{\psi}_{k+1}^{(c)}(\mathbf{x}) = \mathbf{P}\boldsymbol{\psi}_k^{(c)}(\mathbf{P}^T\mathbf{x}), \tag{4.59}$$

where the increment from k *and* $k+1$ *is considered in the cyclic sense, and* $\mathbf{P}$ *is the first order cyclic permutation matrix given by*

$$\mathbf{P} = \begin{cases} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, & \text{for } n = 2, \\ \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}, & \text{for } n = 3. \end{cases}$$

Proof: See appendix.

Consequently, the irrotational an the solenoidal basis function $\boldsymbol{\varphi}_{\mathbf{d}_i}^{(d)}$ and $\boldsymbol{\varphi}_{\mathbf{d}_i}^{(c)}$ given in (4.57) and (4.58) are now expressed as

$$\boldsymbol{\varphi}_{\mathbf{d}_i}^{(d)}(\mathbf{x}) = \sum_{j=0}^{n-1} d_{ij}\mathbf{P}^j\boldsymbol{\psi}_1^{(d)}((\mathbf{P}^T)^j\mathbf{x}), \tag{4.60}$$

$$\boldsymbol{\varphi}_{\mathbf{d}_i}^{(c)}(\mathbf{x}) = \sum_{j=0}^{n-1} d_{ij}\mathbf{P}^j\boldsymbol{\psi}_1^{(c)}((\mathbf{P}^T)^j\mathbf{x}), \tag{4.61}$$

where d_{ij} is the jth component of $\mathbf{d}_i$. This implies that the overall basis function $\boldsymbol{\varphi}_{\mathbf{d}_i}$ can be expressed as

$$\boldsymbol{\varphi}_{\mathbf{d}_i}(\mathbf{x}) = \sum_{j=0}^{n-1} d_{ij}\mathbf{P}^j\boldsymbol{\psi}((\mathbf{P}^T)^j\mathbf{x}), \tag{4.62}$$

where $\boldsymbol{\psi}$ is first column of $\boldsymbol{\Psi}$ given by

$$\boldsymbol{\psi}(\mathbf{x}) = \frac{1}{\lambda_d}\boldsymbol{\psi}_1^{(d)}(\mathbf{x}) + \frac{1}{\lambda_c}\boldsymbol{\psi}_1^{(c)}(\mathbf{x}) \tag{4.63}$$

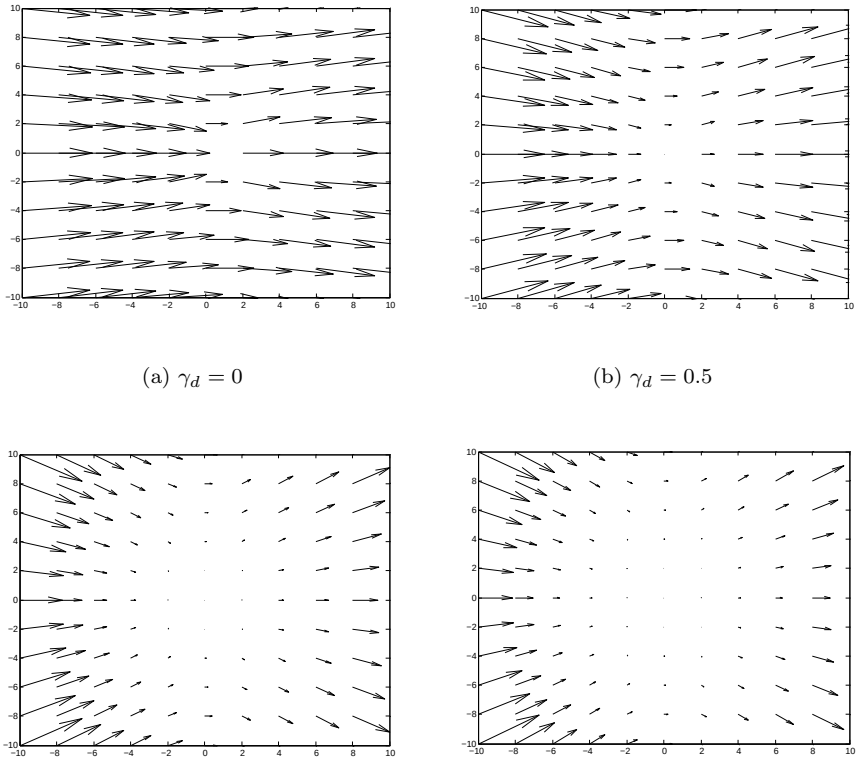
We call $\boldsymbol{\psi}(\mathbf{x})$ the *generalized vector spline*. It is characterized by the parameter set $\{\gamma_d, \gamma_c, \lambda_d, \lambda_c\}$. Note that the shape of $\boldsymbol{\psi}(\mathbf{x})$ is determined by the orders γ_d and γ_c, and the ratio $\frac{\lambda_c}{\lambda_d}$; and hence $\boldsymbol{\psi}(\mathbf{x})$ essentially represents a three-parameter family of functions. However, we retain the four-parameter structure in (4.63), since it facilitates retrieving the following limit cases:

$$\lim_{\lambda_c \to \infty} \boldsymbol{\psi}(\mathbf{x}) = \frac{1}{\lambda_d}\boldsymbol{\psi}_1^{(d)}(\mathbf{x}) \tag{4.64}$$

$$\lim_{\lambda_d \to \infty} \boldsymbol{\psi}(\mathbf{x}) = \frac{1}{\lambda_c}\boldsymbol{\psi}_1^{(c)}(\mathbf{x}) \tag{4.65}$$

This allows the user to specify a irrotational or a solenoidal solution.

Figures 4.3 and 4.4 gives arrow plots of the functions $\boldsymbol{\psi}_1^{(d)}(\mathbf{x})$ and $\boldsymbol{\psi}_1^{(c)}(\mathbf{x})$ for the 2D case. Their magnitude distribution is visualized in Figures 4.5 and 4.6. Note that these functions are not radial unlike the scalar splines.



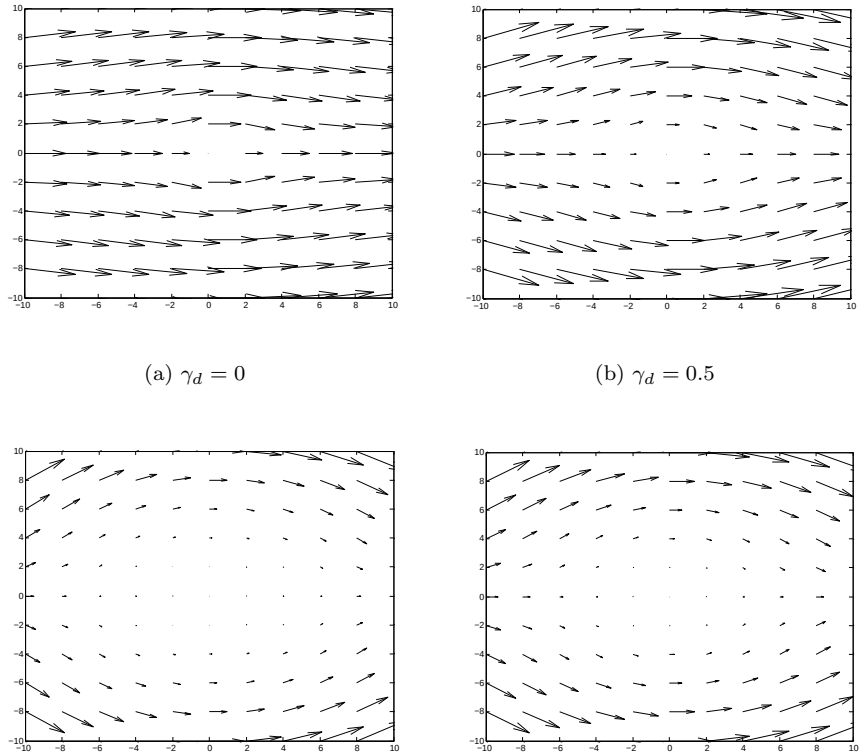
(a) $\gamma_d = 0$ (b) $\gamma_d = 0.5$

(c) $\gamma_d = 1$ (d) $\gamma_d = 1.5$

Figure 4.3: Arrow plots of 2D irrotational vector spline $\boldsymbol{\psi}_1^{(d)}(\mathbf{x})$

4.5.3 Admissible Choice of Parameters

Even thought $\gamma_d, \gamma_c, \lambda_d$, and λ_c are essentially free parameters, choosing the orders γ_d and γ_c less than unity leads to pathological condition when $n = 3$. The corresponding scalar functions ψ_{l_1} and ψ_{l_2} are not twice differentiable in that case, and hence the Green's functions $\boldsymbol{\Psi}^{(d)}$ and $\boldsymbol{\Psi}^{(c)}$ defined in Equations



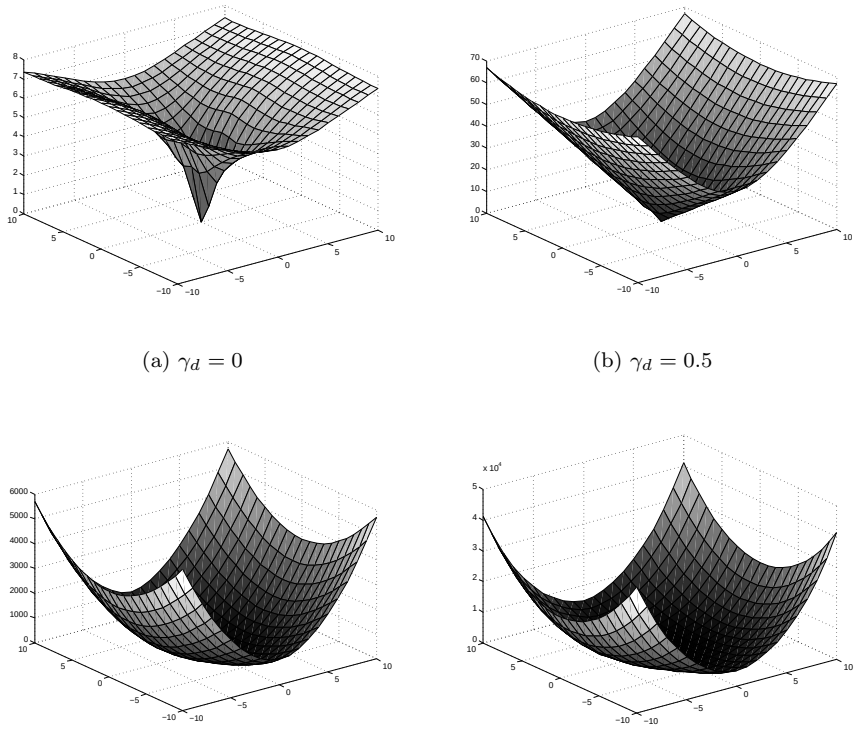
(a) $\gamma_d = 0$

(b) $\gamma_d = 0.5$

(c) $\gamma_d = 1$

(d) $\gamma_d = 1.5$

Figure 4.4: Arrow plots of 2D solenoidal vector spline $\boldsymbol{\psi}_1^{(c)}(\mathbf{x})$



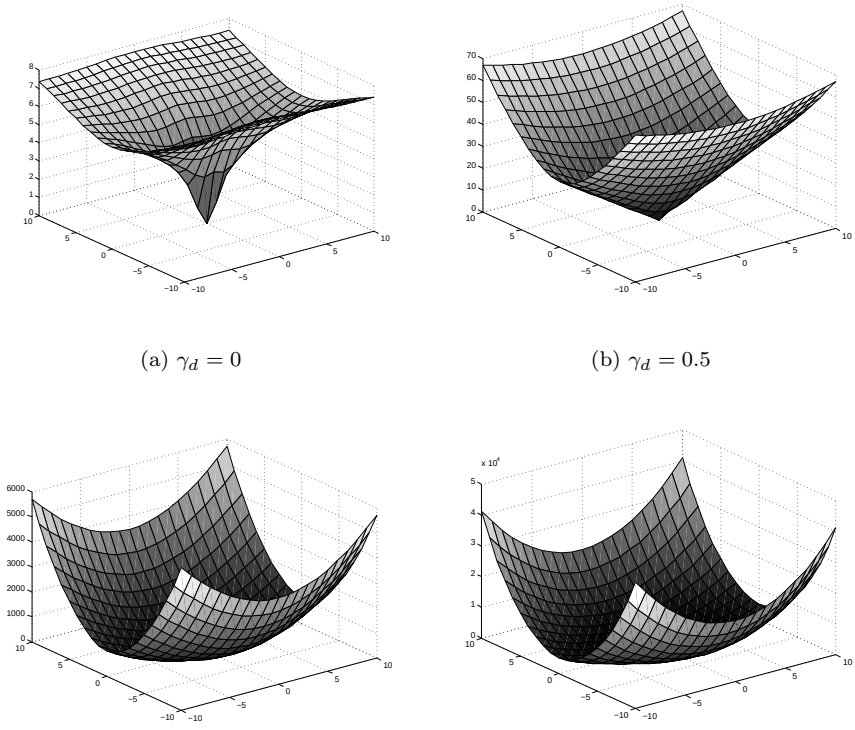
(a) $\gamma_d = 0$

(b) $\gamma_d = 0.5$

(c) $\gamma_d = 1$

(d) $\gamma_d = 1.5$

Figure 4.5: Magnitude plots of 2D irrotational vector spline $\boldsymbol{\psi}_1^{(d)}(\mathbf{x})$



(a) $\gamma_d = 0$

(b) $\gamma_d = 0.5$

(c) $\gamma_d = 1$

(d) $\gamma_d = 1.5$

Figure 4.6: Magnitude plots of 2D solenoidal vector spline $\boldsymbol{\psi}_1^{(c)}(\mathbf{x})$

(4.54) and (4.55) are not continuous.

Nevertheless, choosing γ_d and γ_c less than unity is still useful if the corresponding weight is set to infinity. For example, setting $\gamma_d = 0$ and $\lambda_d \to \infty$ is well-defined in our formulation and it yields a divergence-free solution. Similarly, the setting $\gamma_c = 0$ and $\lambda_c \to \infty$ gives a irrotational solution.

4.6 Numerical Examples

We present here some numerical results for experiments that we conducted on synthetic data. The phantom model is given by

$$\mathbf{f}(\mathbf{x}) = w\mathbf{f}_{irr}(\mathbf{x}) + (1-w)\mathbf{f}_{sol}(\mathbf{x}), \;\; 0 \leq w \leq 1,$$

where $\mathbf{f}_{irr}(\mathbf{x})$ and $\mathbf{f}_{sol}(\mathbf{x})$ are some solenoidal and irrotational vector fields respectively. They are defined in terms of a potential function given below:

$$\phi(\mathbf{x}) = e^{-\|\mathbf{x}\|^2}$$

The irrotational part $\mathbf{f}_{irr}(\mathbf{x})$ is given by

$$\mathbf{f}_{irr}(\mathbf{x}) = \nabla\phi(\mathbf{x}),$$

and the solenoidal part $\mathbf{f}_{sol}(\mathbf{x})$ is given by

$$\mathbf{f}_{sol}(\mathbf{x}) = \begin{cases} \begin{bmatrix} -\partial_y\phi(\mathbf{x}) \\ \partial_x\phi(\mathbf{x}) \end{bmatrix}, & \text{for n = 2,} \\ \nabla \times \begin{bmatrix} -\partial_y\phi(\mathbf{x}) \\ \partial_x\phi(\mathbf{x}) \\ 0 \end{bmatrix}, & \text{for n = 3.} \end{cases}$$

The input data is some list of triplets of the form $\{(\mathbf{x}_i, \mathbf{d}_i, s_i)\}$, where $\mathbf{x}_i$ and $\mathbf{d}_i$ are randomly chosen, and $s_i = \mathbf{d}_i^T\mathbf{f}(\mathbf{x}_i) + n_i$ with n_i being some i.i.d Gaussian noise with variance σ^2. Before we present the numerical results, we define the following quantities:

$$\begin{aligned} \text{Input SNR} &= 10 \times \log\left(\frac{E((\mathbf{d}_i^T\mathbf{f}(\mathbf{x}_i))^2)}{\sigma^2}\right) \\ \text{Reconstruction SNR} &= 10 \times \log\left(\frac{E(\|\mathbf{f}(\mathbf{k})\|^2)}{E(\|\mathbf{f}(\mathbf{k}) - \mathbf{f}_r(\mathbf{k})\|^2)}\right) \end{aligned}$$

Figure 4.7 gives the 2D reconstruction results for $w = 0.25$ with input SNR of 20 dB. It compares the SNR obtained by vector spline reconstruction with $\gamma_d = \gamma_c = 1$ to that of second order thin-plate spline reconstruction, which does not involve any coupling between the x and y components. The regularization parameters were chosen as

$$\lambda_d = k\sigma^2 \frac{(1-w)^2}{w^2 + (1-w)^2},$$

$$\lambda_c = k\sigma^2 \frac{w^2}{w^2 + (1-w)^2},$$

The factor k was chosen empirically such that

$$E(s_i^2) = E((\mathbf{d}_i^T \mathbf{f}_r(\mathbf{x}_i))^2) + \sigma^2,$$

where $\mathbf{f}_r$ is the reconstructed output. Note that choosing $\lambda_d = \lambda_c = k\sigma^2$ yields the second order thin-plate spline solution. From the figure, one can clearly see the advantage of using vector splines.

Figure 4.7 gives the 3D reconstruction result for $w = 1$ with input SNR of 15 dB. Since, we need irrotational reconstruction, the optimal choice of parameters is given by

$$\lambda_d = k\sigma^2,$$

$$\lambda_c \to \infty.$$

With this choice, the figure compares the reconstruction SNR for the choices of order $(\gamma_d = 1, \gamma_c = 1)$ and $(\gamma_d = 1, \gamma_c = 0)$. Note that the choice $(\gamma_d = 1, \gamma_c = 0)$ gives better performance since it specifies the true rotation-free constraint.

Conclusion

We developed a variational method to reconstruct a vector field from projected non-uniform samples. We first characterized a complete family of regularization functionals that satisfy rotational- and scale-invariance properties. We showed that such a functional is composed of a weighted sum of two sub-functionals: (i) Duchon's scalar semi-norm applied on the divergence field; (ii) and the same applied to each component of the rotational field. We provided the analytical solution for the reconstruction using this family of

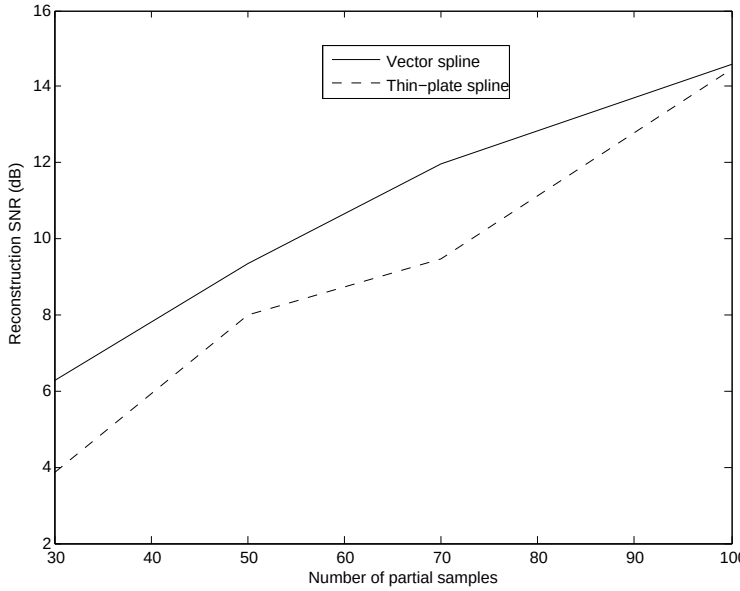


Figure 4.7: Reconstruction error for 2D phantom $\mathbf{f}(\mathbf{x}) = w\mathbf{f}_{irr}(\mathbf{x}) + (1-w)\mathbf{f}_{sol}(\mathbf{x})$. Comparison between second order thin-plate spline and vector spline with $\lambda_d \propto \sigma^2 \frac{(1-w)^2}{w^2+(1-w)^2}, \lambda_c \propto \sigma^2 \frac{w^2}{w^2+(1-w)^2}, \gamma_d = 1, \gamma_c = 1$.

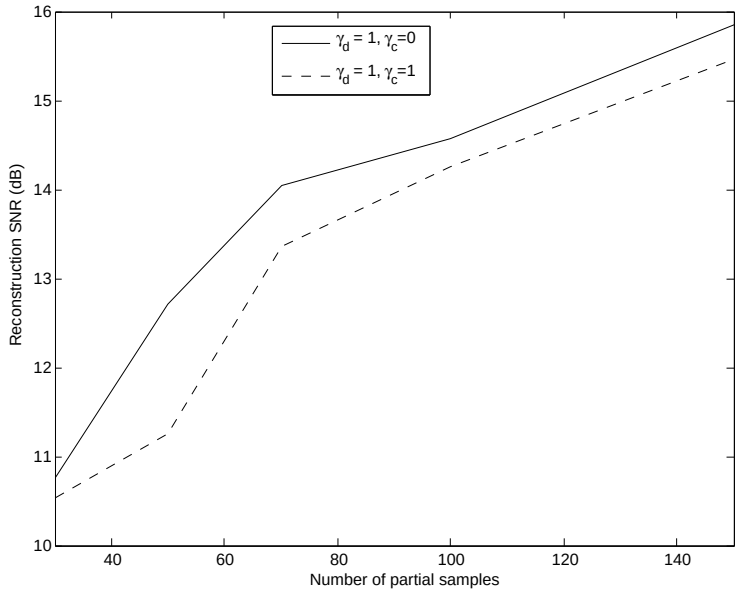


Figure 4.8: Reconstruction error for 3D phantom $\mathbf{f}(\mathbf{x}) = \mathbf{f}_{irr}(\mathbf{x})$ with $\lambda_d \propto \sigma^2, \lambda_c \to \infty$.

regularization functionals. In particular, we showed that the optimal solution consists of two parts: (i) a weighted sum of polynomial basis functions that span the kernel of the regularization functional; (ii) and a weighted sum of some basis functions that are derived from the so called generalized vector spline. We showed how the form of this vector spline is determined by the form of the chosen regularization functional. We derived the linear system of equations for computing the weights to yield the final solution. We demonstrated the method by synthetic phantom reconstruction.

Proof of Theorems

Proposition 1

The functions $\mathbf{f}^{(\min)}$ and $\mathbf{f}'^{(\min)}$ are given by

$$\mathbf{f}^{(\min)} = \underset{\mathbf{f}}{\operatorname{argmin}} \sum_{j}^{N} \left(\mathbf{d}_j^T \mathbf{f}(\mathbf{x}_j) - s_j\right)^2 + \lambda J(\mathbf{f}) \tag{4.66}$$

$$\mathbf{f}'^{(\min)} = \underset{\mathbf{f}}{\operatorname{argmin}} \sum_{j}^{N} \left((\mathbf{\Omega}\mathbf{d}_j)^T \mathbf{f}(\mathbf{\Omega}\mathbf{x}_j) - s_j\right)^2 + \lambda J(\mathbf{f}) \tag{4.67}$$

Applying a transformation on the coordinate system of (4.66) yields

$$\mathbf{f}_{\mathbf{\Omega}}^{(\min)} = \underset{\mathbf{f}}{\operatorname{argmin}} \sum_{j}^{N} \left((\mathbf{\Omega}\mathbf{d}_j)^T \mathbf{f}(\mathbf{\Omega}\mathbf{x}_j) - s_j\right)^2 + \lambda J(\mathbf{f}_{\mathbf{\Omega}}) \tag{4.68}$$

From (4.67) and (4.68) it is clear that $\mathbf{f}'^{(\min)} = \mathbf{f}_{\mathbf{\Omega}}^{(\min)}$ if $J(\mathbf{f}) = J(\mathbf{f}_{\mathbf{\Omega}})$ for every $\mathbf{f}$.

Proposition 2

Define

$$\mathbf{f}^{(a_1)} = \underset{\mathbf{f}}{\operatorname{argmin}} \sum_{j}^{N} \left(\mathbf{d}_j^T \mathbf{f}(a_1\mathbf{x}_j) - s_j\right)^2 + \lambda(a_1) J(\mathbf{f}) \tag{4.69}$$

$$\mathbf{f}^{(a_2)} = \underset{\mathbf{f}}{\operatorname{argmin}} \sum_{j}^{N} \left(\mathbf{d}_j^T \mathbf{f}(a_2\mathbf{x}_j) - s_j\right)^2 + \lambda(a_2) J(\mathbf{f}) \tag{4.70}$$

Let $\mathbf{f}_a(\mathbf{x}) = \mathbf{f}(a\mathbf{x})$. Now, in (4.70), applying a scaling on the coordinate system by a factor a_1/a_2, we get

$$\mathbf{f}_{a_1/a_2}^{(a_2)} = \underset{\mathbf{f}}{\operatorname{argmin}} \sum_{j}^{N} \left(\mathbf{d}_j^T \mathbf{f}(a_1\mathbf{x}_j) - s_j\right)^2 + \lambda(a_2) J(\mathbf{f}_{a_1/a_2}) \tag{4.71}$$

From (4.69) and (4.71) it is clear that $\mathbf{f}^{(a_1)} = \mathbf{f}_{a_1/a_2}^{(a_2)}$ if $\lambda(a_1)J(\mathbf{f}) = \lambda(a_2)J(\mathbf{f}_{a_1/a_2})$. This implies that $J(\mathbf{f}_a) = c(a)J(\mathbf{f})$ for some continuous function c.

Theorem 2

We intend to show (4.8) for every orthogonal matrix $\mathbf{\Omega}$. We consider two cases of n.

Case (i): $n = 2$

We first choose two orthogonal matrices that are functions of the frequency variable $\boldsymbol{\omega} = [\omega_1 \;\; \omega_2]^T$. To this end, we define the following matrices:

$$\boldsymbol{\Theta} = \begin{bmatrix} \omega_1 & \omega_2 \\ -\omega_2 & \omega_1 \end{bmatrix}$$

$$\mathbf{D} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

Now we investigate the implication of (4.8) for the following matrices:

$$\boldsymbol{\Omega}_1 = \frac{1}{\|\boldsymbol{\omega}\|}\boldsymbol{\Theta} \tag{4.72}$$

$$\boldsymbol{\Omega}_2 = \mathbf{D}\boldsymbol{\Omega}_1 \tag{4.73}$$

Substituting $\boldsymbol{\Omega}_1$ and $\boldsymbol{\Omega}_2$ in (4.8) yields

$$\mathbf{U}(\boldsymbol{\omega}) = \boldsymbol{\Omega}_1^T \mathbf{U}(\mathbf{e}_1\|\boldsymbol{\omega}\|)\boldsymbol{\Omega}_1 \tag{4.74}$$

$$\mathbf{U}(\boldsymbol{\omega}) = \boldsymbol{\Omega}_1^T \mathbf{D}\mathbf{U}(\mathbf{e}_1\|\boldsymbol{\omega}\|)\mathbf{D}\boldsymbol{\Omega}_1 \tag{4.75}$$

Equations (4.74) and (4.75) imply that $\mathbf{U}(\mathbf{e}_1\|\boldsymbol{\omega}\|) = \mathbf{D}\mathbf{U}(\mathbf{e}_1\|\boldsymbol{\omega}\|)\mathbf{D}$, which in turn implies that $\mathbf{U}(\mathbf{e}_1\|\boldsymbol{\omega}\|)$ is diagonal. Hence, (4.74) is indeed the eigenvalue decomposition of $\mathbf{U}(\boldsymbol{\omega})$. Hence $\mathbf{U}(\boldsymbol{\omega})$ is necessarily of the form

$$\mathbf{U}(\boldsymbol{\omega}) = \boldsymbol{\Omega}_1^T \mathbf{A}(\|\boldsymbol{\omega}\|)\boldsymbol{\Omega}_1, \tag{4.76}$$

where $\mathbf{A}(\|\boldsymbol{\omega}\|)$ is is diagonal matrix that is a function of $\|\boldsymbol{\omega}\|$ only. Equation (4.76) can be written as

$$\mathbf{U}(\boldsymbol{\omega}) = \boldsymbol{\Theta}^T \begin{bmatrix} \alpha_d(\|\boldsymbol{\omega}\|) & 0 \\ 0 & \alpha_c(\|\boldsymbol{\omega}\|) \end{bmatrix} \boldsymbol{\Theta}, \tag{4.77}$$

where we have used the relation (4.72) and made the substitution

$$\begin{bmatrix} \alpha_d(\|\boldsymbol{\omega}\|) & 0 \\ 0 & \alpha_c(\|\boldsymbol{\omega}\|) \end{bmatrix} = \frac{1}{\|\boldsymbol{\omega}\|^2}\mathbf{A}(\|\boldsymbol{\omega}\|).$$

Next we rewrite (4.77) as

$$\mathbf{U}(\boldsymbol{\omega}) = \alpha_d(\|\boldsymbol{\omega}\|) \begin{bmatrix} \omega_1 \\ \omega_2 \end{bmatrix} [\omega_1 \;\; \omega_2] + \alpha_d(\|\boldsymbol{\omega}\|) \begin{bmatrix} -\omega_2 \\ \omega_1 \end{bmatrix} [-\omega_2 \;\; \omega_1] \tag{4.78}$$

Now, it is straightforward to verify that

$$\begin{bmatrix} -\omega_2 \\ \omega_1 \end{bmatrix} [-\omega_2 \;\; \omega_1] = \|\boldsymbol{\omega}\|^2 \mathbf{I} - \boldsymbol{\omega}\boldsymbol{\omega}^T.$$

Hence (4.78) becomes

$$\mathbf{U}(\boldsymbol{\omega}) = \alpha_d(\|\boldsymbol{\omega}\|)\boldsymbol{\omega}\boldsymbol{\omega}^T + \alpha_c(\|\boldsymbol{\omega}\|)(\|\boldsymbol{\omega}\|^2 \mathbf{I} - \boldsymbol{\omega}\boldsymbol{\omega}^T).$$

Case (ii): $n = 3$

The first step is to analyze (4.8) for a rotation matrix with $\boldsymbol{\omega}$ as its axis of rotation; i.e., for a rotation matrix $\boldsymbol{\Omega}_1$ satisfying

$$\boldsymbol{\Omega}_1\boldsymbol{\omega} = \boldsymbol{\omega}. \tag{4.79}$$

Substituting $\boldsymbol{\Omega} = \boldsymbol{\Omega}_1$ in (4.8) and multiplying by $\boldsymbol{\omega}$ on both sides yields

$$\begin{aligned} \boldsymbol{\Omega}_1^T\mathbf{U}(\boldsymbol{\Omega}_1\boldsymbol{\omega})\boldsymbol{\Omega}_1\boldsymbol{\omega} &= \mathbf{U}(\boldsymbol{\omega})\boldsymbol{\omega}, \\ \Leftrightarrow \quad \boldsymbol{\Omega}_1^T\mathbf{U}(\boldsymbol{\omega})\boldsymbol{\omega} &= \mathbf{U}(\boldsymbol{\omega})\boldsymbol{\omega}, \\ \Leftrightarrow \quad \mathbf{U}(\boldsymbol{\omega})\boldsymbol{\omega} &= \boldsymbol{\Omega}_1(\mathbf{U}(\boldsymbol{\omega})\boldsymbol{\omega}). \end{aligned}$$

The above equation implies that whenever $\boldsymbol{\omega}$ is an eigen vector of $\boldsymbol{\Omega}_1$, the same is also true for $\mathbf{U}(\boldsymbol{\omega})\boldsymbol{\omega}$. Hence $\boldsymbol{\omega}$ is also an eigen vector of $\mathbf{U}(\boldsymbol{\omega})$.

Now, we look at the other eigen vectors of $\boldsymbol{\Omega}_1$. They are complex and they can be expressed in the following form:

$$\mathbf{u}(\boldsymbol{\omega}) = \mathbf{u}_1(\boldsymbol{\omega}) + j\mathbf{u}_2(\boldsymbol{\omega}), \tag{4.80}$$

$$\mathbf{u}^*(\boldsymbol{\omega}) = \mathbf{u}_1(\boldsymbol{\omega}) - j\mathbf{u}_2(\boldsymbol{\omega}), \tag{4.81}$$

where $\mathbf{u}_1(\boldsymbol{\omega})$ and $\mathbf{u}_2(\boldsymbol{\omega})$ are some real vectors such that

$$\begin{aligned} \|\mathbf{u}_1\| &= 1/2 \\ \|\mathbf{u}_2\| &= 1/2 \\ \mathbf{u}_i^T(\boldsymbol{\omega})\boldsymbol{\omega} &= 0, \;\; i = 1, 2 \\ \mathbf{u}_1^T(\boldsymbol{\omega})\mathbf{u}_2(\boldsymbol{\omega}) &= 0 \end{aligned}$$

By construction, the following hold:

$$\begin{aligned}
\mathbf{u}^T(\boldsymbol{\omega})\boldsymbol{\omega} &= 0 && (4.82)\\
\mathbf{u}^{*T}(\boldsymbol{\omega})\boldsymbol{\omega} &= 0 && (4.83)\\
\mathbf{u}^T(\boldsymbol{\omega})\mathbf{u}(\boldsymbol{\omega}) &= 0 && (4.84)\\
\mathbf{u}^{*T}(\boldsymbol{\omega})\mathbf{u}^*(\boldsymbol{\omega}) &= 0 && (4.85)\\
\mathbf{u}^{*T}(\boldsymbol{\omega})\mathbf{u}(\boldsymbol{\omega}) &= 1 && (4.86)
\end{aligned}$$

Since $\mathbf{u}(\boldsymbol{\omega})$ and $\mathbf{u}^*(\boldsymbol{\omega})$ are the eigen vectors of $\boldsymbol{\Omega}_1$, they satisfy the following:

$$\begin{aligned}
\boldsymbol{\Omega}_1\mathbf{u}(\boldsymbol{\omega}) &= e^{j\theta}\mathbf{u}(\boldsymbol{\omega}) && (4.87)\\
\boldsymbol{\Omega}_1\mathbf{u}^*(\boldsymbol{\omega}) &= e^{-j\theta}\mathbf{u}^*(\boldsymbol{\omega}) && (4.88)
\end{aligned}$$

We now apply (4.87) on (4.8):

$$\begin{aligned}
\boldsymbol{\Omega}_1^T\mathbf{U}(\boldsymbol{\Omega}_1\boldsymbol{\omega})\boldsymbol{\Omega}_1\mathbf{u}(\boldsymbol{\omega}) &= \mathbf{U}(\boldsymbol{\omega})\mathbf{u}(\boldsymbol{\omega})\\
\Leftrightarrow\ \boldsymbol{\Omega}_1^T\mathbf{U}(\boldsymbol{\omega})e^{j\theta}\mathbf{u}(\boldsymbol{\omega}) &= \mathbf{U}(\boldsymbol{\omega})\mathbf{u}(\boldsymbol{\omega})\\
\Leftrightarrow\ \boldsymbol{\Omega}_1(\mathbf{U}(\boldsymbol{\omega})\mathbf{u}(\boldsymbol{\omega})) &= e^{j\theta}(\mathbf{U}(\boldsymbol{\omega})\mathbf{u}(\boldsymbol{\omega}))
\end{aligned}$$

The above equation implies that $\mathbf{u}(\boldsymbol{\omega})$ is an eigen vector of $\mathbf{U}(\boldsymbol{\omega})$. In a similar way, it can be shown that $\mathbf{u}^*(\boldsymbol{\omega})$ is also an eigen vector of $\mathbf{U}(\boldsymbol{\omega})$. Hence $\mathbf{U}(\boldsymbol{\omega})$ can be expressed as follows:

$$\mathbf{U}(\boldsymbol{\omega}) = \frac{\mu_1(\boldsymbol{\omega})}{\|\boldsymbol{\omega}\|^2}\boldsymbol{\omega}\boldsymbol{\omega}^T + \mu_2(\boldsymbol{\omega})\mathbf{u}(\boldsymbol{\omega})\mathbf{u}^{*T}(\boldsymbol{\omega}) + \mu_2^*(\boldsymbol{\omega})\mathbf{u}^*(\boldsymbol{\omega})\mathbf{u}^T(\boldsymbol{\omega}), \qquad (4.89)$$

where $\mu_1(\boldsymbol{\omega})$ and $\mu_2(\boldsymbol{\omega})$ are some real and complex functions, respectively.

For a general orthogonal matrix, the RHS of the equation (4.8) now is given by

$$\boldsymbol{\Omega}^T\mathbf{U}(\boldsymbol{\Omega}\boldsymbol{\omega})\boldsymbol{\Omega} = \frac{\mu_1(\boldsymbol{\Omega}\boldsymbol{\omega})}{\|\boldsymbol{\omega}\|^2}\boldsymbol{\omega}\boldsymbol{\omega}^T + \mu_2(\boldsymbol{\Omega}\boldsymbol{\omega})\mathbf{v}(\boldsymbol{\omega})\mathbf{v}^{*T}(\boldsymbol{\omega}) + \mu_2^*(\boldsymbol{\Omega}\boldsymbol{\omega})\mathbf{v}^*(\boldsymbol{\omega})\mathbf{v}^T(\boldsymbol{\omega}), \qquad (4.90)$$

where

$$\begin{aligned}
\mathbf{v}(\boldsymbol{\omega}) &= \boldsymbol{\Omega}^T\mathbf{u}(\boldsymbol{\Omega}\boldsymbol{\omega}) && (4.91)\\
\mathbf{v}^*(\boldsymbol{\omega}) &= \boldsymbol{\Omega}^T\mathbf{u}^*(\boldsymbol{\Omega}\boldsymbol{\omega}) && (4.92)
\end{aligned}$$

We now observe that (4.82) implies

$$\mathbf{v}^T(\boldsymbol{\omega})\boldsymbol{\omega} = 0 \qquad (4.93)$$

This can be verified as follows:

$$\begin{aligned}\boldsymbol{\omega}^T\mathbf{v}(\boldsymbol{\omega}) &= \boldsymbol{\omega}^T\boldsymbol{\Omega}^T\mathbf{u}(\boldsymbol{\Omega}\boldsymbol{\omega}) \\ &= (\boldsymbol{\Omega}\boldsymbol{\omega})^T\mathbf{u}(\boldsymbol{\Omega}\boldsymbol{\omega}) \\ &= 0\end{aligned}$$

In a similar way, it can be verified that (4.83)—(4.86) imply the following:

$$\mathbf{v}^{*T}(\boldsymbol{\omega})\boldsymbol{\omega} = 0 \tag{4.94}$$

$$\mathbf{v}^T(\boldsymbol{\omega})\mathbf{v}(\boldsymbol{\omega}) = 0 \tag{4.95}$$

$$\mathbf{v}^{*T}(\boldsymbol{\omega})\mathbf{v}^*(\boldsymbol{\omega}) = 0 \tag{4.96}$$

$$\mathbf{v}^{*T}(\boldsymbol{\omega})\mathbf{v}(\boldsymbol{\omega}) = 1 \tag{4.97}$$

Now (4.93) implies that $\mathbf{v}(\boldsymbol{\omega})$ is in the space spanned by $\mathbf{u}(\boldsymbol{\omega})$ and $\mathbf{u}^*(\boldsymbol{\omega})$. Hence, it can expressed as

$$\mathbf{v}(\boldsymbol{\omega}) = \vartheta_1\mathbf{u}(\boldsymbol{\omega}) + \vartheta_2\mathbf{u}^*(\boldsymbol{\omega}), \tag{4.98}$$

for some complex numbers ϑ_1 and ϑ_2. Substituting (4.98) in (4.95) and simplifying using the relations (4.84), (4.85), and (4.86) yields

$$\vartheta_1\vartheta_2 = 0$$

This implies that $\mathbf{v}(\boldsymbol{\omega})$ is either of the form $\vartheta_1\mathbf{u}(\boldsymbol{\omega})$ or $\vartheta_2\mathbf{u}^*(\boldsymbol{\omega})$, and ϑ_1 and ϑ_2 are unit modulus numbers. Now let $\boldsymbol{\Omega}_2$ and $\boldsymbol{\Omega}_3$ be some orthogonal matrices such that

$$\mathbf{v}_2(\boldsymbol{\omega}) = \boldsymbol{\Omega}_2^T\mathbf{u}(\boldsymbol{\Omega}_2\boldsymbol{\omega}) = \vartheta_1\mathbf{u}(\boldsymbol{\omega}) \tag{4.99}$$

$$\mathbf{v}_3(\boldsymbol{\omega}) = \boldsymbol{\Omega}_3^T\mathbf{u}(\boldsymbol{\Omega}_3\boldsymbol{\omega}) = \vartheta_2\mathbf{u}^*(\boldsymbol{\omega}) \tag{4.100}$$

Substituting $\boldsymbol{\Omega} = \boldsymbol{\Omega}_2$ in (4.90), and applying (4.8) and (4.99) yields

$$\mathbf{U}(\boldsymbol{\omega}) = \frac{\mu_1(\boldsymbol{\Omega}_2\boldsymbol{\omega})}{\|\boldsymbol{\omega}\|^2}\boldsymbol{\omega}\boldsymbol{\omega}^T + \mu_2(\boldsymbol{\Omega}_2\boldsymbol{\omega})\mathbf{u}(\boldsymbol{\omega})\mathbf{u}^{*T}(\boldsymbol{\omega}) + \mu_2^*(\boldsymbol{\Omega}_2\boldsymbol{\omega})\mathbf{u}^*(\boldsymbol{\omega})\mathbf{u}^T(\boldsymbol{\omega}). \tag{4.101}$$

Similarly, substituting $\boldsymbol{\Omega} = \boldsymbol{\Omega}_3$ in (4.90), and applying (4.8) and (4.99) yields

$$\mathbf{U}(\boldsymbol{\omega}) = \frac{\mu_1(\boldsymbol{\Omega}_3\boldsymbol{\omega})}{\|\boldsymbol{\omega}\|^2}\boldsymbol{\omega}\boldsymbol{\omega}^T + \mu_2(\boldsymbol{\Omega}_3\boldsymbol{\omega})\mathbf{u}^*(\boldsymbol{\omega})\mathbf{u}^T(\boldsymbol{\omega}) + \mu_2^*(\boldsymbol{\Omega}_3\boldsymbol{\omega})\mathbf{u}(\boldsymbol{\omega})\mathbf{u}^{*T}(\boldsymbol{\omega}). \tag{4.102}$$

In the equations (4.89) , (4.101), and (4.102), equating the scalar coefficients of the corresponding terms yields

$$\mu_1(\boldsymbol{\omega}) = \mu_1(\boldsymbol{\Omega}_2\boldsymbol{\omega}) = \mu_1(\boldsymbol{\Omega}_3\boldsymbol{\omega})$$

$$\mu_2(\boldsymbol{\omega}) = \mu_2(\boldsymbol{\Omega}_2\boldsymbol{\omega}) = \mu_2^*(\boldsymbol{\Omega}_3\boldsymbol{\omega})$$

$$\mu_2(\boldsymbol{\omega}) = \mu_2^*(\boldsymbol{\Omega}_2\boldsymbol{\omega}) = \mu_2(\boldsymbol{\Omega}_3\boldsymbol{\omega})$$

Since, $\boldsymbol{\Omega}_2$ and $\boldsymbol{\Omega}_3$ are arbitrary matices, this implies that $\mu_1(\boldsymbol{\omega})$ and $\mu_2(\boldsymbol{\omega})$ are radial functions, and that $\mu_2(\boldsymbol{\omega})$ is indeed a real function. Hence (4.102) becomes

$$\mathbf{U}(\boldsymbol{\omega}) = \mu_1(\|\boldsymbol{\omega}\|)\boldsymbol{\omega}\boldsymbol{\omega}^T + \mu_2(\|\boldsymbol{\omega}\|)\left(\mathbf{u}(\boldsymbol{\omega})\mathbf{u}^{*T}(\boldsymbol{\omega}) + \mathbf{u}^*(\boldsymbol{\omega})\mathbf{u}^T(\boldsymbol{\omega})\right), \qquad (4.103)$$

Now, Equations (4.82)— (4.86) imply

$$\frac{1}{\|\boldsymbol{\omega}\|^2}\boldsymbol{\omega}\boldsymbol{\omega}^T + \mathbf{u}(\boldsymbol{\omega})\mathbf{u}^{*T}(\boldsymbol{\omega}) + \mathbf{u}^*(\boldsymbol{\omega})\mathbf{u}^T(\boldsymbol{\omega}) = \mathbf{I}$$

Consequently (4.103) becomes

$$\mathbf{U}(\boldsymbol{\omega}) = \mu_1(\|\boldsymbol{\omega}\|)\boldsymbol{\omega}\boldsymbol{\omega}^T + \mu_2(\|\boldsymbol{\omega}\|)\left(\|\boldsymbol{\omega}\|^2\mathbf{I} - \boldsymbol{\omega}\boldsymbol{\omega}^T\right)$$

Substituting $\alpha_d(\|\boldsymbol{\omega}\|) = \mu_1(\|\boldsymbol{\omega}\|)$ and $\alpha_c(\|\boldsymbol{\omega}\|) = \mu_2(\|\boldsymbol{\omega}\|)$ yields the required relation:

$$\mathbf{U}(\boldsymbol{\omega}) = \alpha_d(\|\boldsymbol{\omega}\|)\boldsymbol{\omega}\boldsymbol{\omega}^T + \alpha_c(\|\boldsymbol{\omega}\|)(\|\boldsymbol{\omega}\|^2\mathbf{I} - \boldsymbol{\omega}\boldsymbol{\omega}^T).$$

Theorem 3: Div–Curl sub-functionals

We first rewrite (4.16) by substituting (4.18)

$$\begin{aligned} J_d(\mathbf{f}) &= \int \alpha_d(\boldsymbol{\omega})\hat{\mathbf{f}}^\dagger(\boldsymbol{\omega})\left(\boldsymbol{\omega}\boldsymbol{\omega}^T\right)\hat{\mathbf{f}}(\boldsymbol{\omega})d\boldsymbol{\omega} \\ &= \int \alpha_d(\|\boldsymbol{\omega}\|)\left(j\boldsymbol{\omega}^T\hat{\mathbf{f}}(\boldsymbol{\omega})\right)^\dagger\left(j\boldsymbol{\omega}^T\hat{\mathbf{f}}(\boldsymbol{\omega})\right)d\boldsymbol{\omega} \end{aligned}$$

Since $\mathcal{F}(\nabla\cdot\mathbf{f}) = j\boldsymbol{\omega}^T\hat{\mathbf{f}}(\boldsymbol{\omega})$, the above equation becomes

$$J_d(\mathbf{f}) = \int \alpha_d(\|\boldsymbol{\omega}\|)\hat{f}_d^\dagger(\boldsymbol{\omega})\hat{f}_d(\boldsymbol{\omega})d\boldsymbol{\omega}.$$

To prove the second part, we rewrite (4.17) by substituting (4.19)

$$J_c(\mathbf{f}) = \int \alpha_c(\|\boldsymbol{\omega}\|)\hat{\mathbf{f}}^\dagger(\boldsymbol{\omega})(\|\boldsymbol{\omega}\|^2\mathbf{I} - \boldsymbol{\omega}\boldsymbol{\omega}^T)\hat{\mathbf{f}}(\boldsymbol{\omega})d\boldsymbol{\omega} \qquad (4.104)$$

Now, by a simple algebra, it is straight forward to verify that

$$\|\boldsymbol{\omega}\|^2 \mathbf{I} - \boldsymbol{\omega}\boldsymbol{\omega}^T = \mathbf{C}(\boldsymbol{\omega})^T \mathbf{C}(\boldsymbol{\omega})$$

where

$$\mathbf{C}(\boldsymbol{\omega}) = \begin{cases} [-\omega_2 \;\; \omega_1], & \text{for } n = 2, \\ \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ -\omega_3 & 0 & \omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix}, & \text{for } n = 3. \end{cases}$$

Hence (4.104) becomes

$$J_c(\mathbf{f}) = \int \alpha_c(\|\boldsymbol{\omega}\|) \left(j\mathbf{C}(\boldsymbol{\omega})\hat{\mathbf{f}}(\boldsymbol{\omega})\right)^\dagger \left(j\mathbf{C}(\boldsymbol{\omega})\hat{\mathbf{f}}(\boldsymbol{\omega})\right) d\boldsymbol{\omega} \tag{4.105}$$

Now, we recognize that $j\mathbf{C}(\boldsymbol{\omega})\hat{\mathbf{f}}(\boldsymbol{\omega})$ is the Fourier transform of the curl, i.e.,

$$\hat{\mathbf{f}}_c(\boldsymbol{\omega}) = \mathcal{F}(\nabla \times \mathbf{f}) = j\mathbf{C}(\boldsymbol{\omega})\hat{\mathbf{f}}(\boldsymbol{\omega}).$$

Hence (4.105) becomes

$$J_c(\mathbf{f}) = \int \alpha_c(\|\boldsymbol{\omega}\|) \hat{f}_c^\dagger(\boldsymbol{\omega}) \hat{f}_c(\boldsymbol{\omega}) d\boldsymbol{\omega}.$$

Theorem 4: Solution to the Variational Problem

Minimality Conditions

The goal is to find the minimum of (4.1) within the space $\mathcal{V}$. We first note that $\mathbf{d}_i\mathbf{f}(\mathbf{x}_i)$ can be written as

$$\mathbf{d}_i\mathbf{f}(\mathbf{x}_i) = \langle \mathbf{r}_i, \mathbf{f} \rangle, \tag{4.106}$$

where $\mathbf{r}_i(\mathbf{x}) = \mathbf{d}_i\delta(\mathbf{x} - \mathbf{x}_i)$. Then (4.1) becomes

$$J_a(\mathbf{f}) = \sum_{i=1}^{N} (\langle \mathbf{r}_i, \mathbf{f} \rangle - s_i)^2 + \lambda J(\mathbf{f}). \tag{4.107}$$

A necessary condition for a function $\mathbf{f}_{\text{opt}}$ to be a local minimum is

$$\lim_{\alpha \to 0} \frac{J_a(\mathbf{f} + \alpha\mathbf{g}) - J_a(\mathbf{f})}{\alpha} = 0, \forall \mathbf{g} \in \mathcal{V} \tag{4.108}$$

It can be shown that the above equation implies

$$\lambda B(\mathbf{f}_{\text{opt}}, \mathbf{g}) = -\sum_{i=1}^{N} \langle \mathbf{r}_i, \mathbf{g} \rangle \left(\langle \mathbf{r}_i, \mathbf{f}_{\text{opt}} \rangle - s_i\right), \forall \mathbf{g} \in \mathcal{V} \tag{4.109}$$

Since the $J_a(\mathbf{f})$ is a quadratic functional, the above condition is also sufficient. Let $\mathcal{D}$ be the space test functions. We assume that $\mathcal{D} + \mathcal{K}_f$ is dense in $\mathcal{V}$, i.e., for every $\mathbf{f} \in \mathcal{V}$, we assume that there exist a polynomial $\mathbf{p} \in \mathcal{K}_f$, and a sequence of test functions $\{\mathbf{u}_n\}$ such that

$$\lim_{n\to\infty} J_a(\mathbf{f} - \mathbf{u}_n - \mathbf{p}) = 0 \tag{4.110}$$

The consequence is that (4.109) can be decomposed into the following two conditions:

$$\sum_{i=1}^{N} \langle \mathbf{r}_i, \mathbf{g} \rangle \left(\langle \mathbf{r}_i, \mathbf{f}_{\text{opt}} \rangle - s_i \right) = -\lambda B(\mathbf{f}_{\text{opt}}, \mathbf{g}), \forall \mathbf{g} \in \mathcal{D} \tag{4.111}$$

$$\sum_{i=1}^{N} \langle \mathbf{r}_i, \mathbf{g} \rangle \left(\langle \mathbf{r}_i, \mathbf{f}_{\text{opt}} \rangle - s_i \right) = 0, \forall \mathbf{g} \in \mathcal{K}_f \tag{4.112}$$

Constructing the Solution

The goal is now to construct a function that satisfies (4.111) and (4.112). We suppose that we have found a set of functions $\{\boldsymbol{\varphi}_i\}$ such that

$$B(\boldsymbol{\varphi}_i, \mathbf{g}) = \langle \mathbf{r}_i, \mathbf{g} \rangle \tag{4.113}$$

We consider the following candidate solution:

$$\mathbf{f}_{\text{opt}}(\mathbf{x}) = \sum_{k=1}^{N} w_k \boldsymbol{\varphi}(\mathbf{x}) + \sum_{k=1}^{Q} a_k \mathbf{p}_k(\mathbf{x}), \tag{4.114}$$

where $\{\mathbf{p}_k(\mathbf{x}), k = 1, \ldots, Q\}$ is the basis for the kernel $\mathcal{K}_f$. We intend to show that it possible to choose the weights $\{w_k, k = 1, \ldots, N\}$ and $\{a_k, k = 1, \ldots, Q\}$ such that (4.111) and (4.112) are satisfied. To this end, we substitute (4.114) into (4.111) and (4.112). We first consider (4.111):

$$\begin{aligned} &\sum_{j=1}^{N} \sum_{i=1}^{N} w_j \langle \mathbf{r}_i, \mathbf{g} \rangle \langle \mathbf{r}_i, \boldsymbol{\varphi}_j \rangle + \sum_{j=1}^{Q} \sum_{i=1}^{N} a_j \langle \mathbf{r}_i, \mathbf{g} \rangle \langle \mathbf{r}_i, \mathbf{p}_j \rangle - \sum_{i=1}^{N} \langle \mathbf{r}_i, \mathbf{g} \rangle s_i \\ &\quad = \lambda \sum_{j=1}^{N} w_j B(\boldsymbol{\varphi}_j, \mathbf{g}) + \underbrace{\lambda \sum_{j=1}^{Q} a_j B(\mathbf{p}_j, \mathbf{g})}_{0}, \quad \forall \mathbf{g} \in \mathcal{D} \end{aligned} \tag{4.115}$$

Plugging-in (4.113) gives

$$\begin{aligned}\sum_{j=1}^{N}\sum_{i=1}^{N} w_j \left\langle \mathbf{r}_i, \mathbf{g} \right\rangle \left\langle \mathbf{r}_i, \boldsymbol{\varphi}_j \right\rangle + \sum_{j=1}^{Q}\sum_{i=1}^{N} a_j \left\langle \mathbf{r}_i, \mathbf{g} \right\rangle \left\langle \mathbf{r}_i, \mathbf{p}_j \right\rangle - \sum_{i=1}^{N} \left\langle \mathbf{r}_i, \mathbf{g} \right\rangle s_i \\ = \lambda \sum_{j=1}^{N} w_j \left\langle \mathbf{r}_i, \mathbf{g} \right\rangle \quad \forall \mathbf{g} \in \mathcal{D}\end{aligned} \tag{4.116}$$

We make the following substitutions:

$$\begin{aligned}\mathbf{a} &= [\cdots a_j \cdots]^T \\ \mathbf{w} &= [\cdots w_j \cdots]^T \\ \mathbf{s} &= [\cdots s_j \cdots]^T \\ \mathbf{b} &= [\cdots \left\langle \mathbf{r}_i, \mathbf{g} \right\rangle \cdots]^T \\ \{\mathbf{A}\}_{ij} &= \left\langle \mathbf{r}_i, \boldsymbol{\varphi}_j \right\rangle \\ \{\mathbf{Q}\}_{ij} &= \left\langle \mathbf{r}_i, \mathbf{p}_j \right\rangle\end{aligned}$$

This yields

$$\mathbf{b}^T\mathbf{A}\mathbf{w} + \mathbf{b}^T\mathbf{Q}\mathbf{a} - \mathbf{b}^T\mathbf{s} = -\lambda \mathbf{b}^T\mathbf{w}, \forall \mathbf{b} \in \mathbb{R}^N, \tag{4.117}$$

where we have replaced the condition $\forall \mathbf{g} \in \mathcal{D}$ by the condition $\forall \mathbf{b} \in \mathbb{R}^N$. This is valid if the mapping from $\mathcal{D}$ to $\mathbb{R}^N$ induced by the measurement functionals $\{\mathbf{r}_i, i = 1, \ldots, N\}$ is surjective. Equation (4.117) implies

$$(\mathbf{A} + \lambda \mathbf{I})\,\mathbf{w} + \mathbf{Q}\mathbf{a} = \mathbf{s} \tag{4.118}$$

Replacing (4.114) in (4.112) yields

$$\begin{aligned}\sum_{j=1}^{N}\sum_{i=1}^{N} w_j \left\langle \mathbf{r}_i, \mathbf{g} \right\rangle \left\langle \mathbf{r}_i, \boldsymbol{\varphi}_j \right\rangle + \sum_{j=1}^{Q}\sum_{i=1}^{N} a_j \left\langle \mathbf{r}_i, \mathbf{g} \right\rangle \left\langle \mathbf{r}_i, \mathbf{p}_j \right\rangle \\ - \sum_{i=1}^{N} \left\langle \mathbf{r}_i, \mathbf{g} \right\rangle s_i = 0 \quad \forall \mathbf{g} \in \mathcal{K}_f\end{aligned} \tag{4.119}$$

This can be equivalently written in terms of the basis functions of $\mathcal{K}_f$ as follows:

$$\begin{aligned}\sum_{j=1}^{N}\sum_{i=1}^{N} w_j \left\langle \mathbf{r}_i, \mathbf{p}_k \right\rangle \left\langle \mathbf{r}_i, \boldsymbol{\varphi}_j \right\rangle + \sum_{j=1}^{Q}\sum_{i=1}^{N} a_j \left\langle \mathbf{r}_i, \mathbf{p}_k \right\rangle \left\langle \mathbf{r}_i, \mathbf{p}_j \right\rangle \\ - \sum_{i=1}^{N} \left\langle \mathbf{r}_i, \mathbf{g} \right\rangle s_i = 0 \quad \text{for } k = 1, \ldots, Q\end{aligned} \tag{4.120}$$

The above equation is then written in matrix form,

$$\mathbf{q}_k^T \mathbf{A}\mathbf{w} + \mathbf{q}_k^T \mathbf{Q}\mathbf{a} = \mathbf{q}_k^T \mathbf{s}, \text{for } k = 1, \ldots, Q. \tag{4.121}$$

where $\mathbf{q}_k = [\cdots \langle \mathbf{r}_i, \mathbf{q}_k \rangle \cdots]^T$. Let $\mathbf{Q} = [\mathbf{q}_1 \cdots \mathbf{q}_Q]$. Then the above set of Q equations can be written as

$$\mathbf{Q}^T \mathbf{A}\mathbf{w} + \mathbf{Q}^T \mathbf{Q}\mathbf{a} = \mathbf{Q}^T \mathbf{s} \tag{4.122}$$

Multiplying (4.118) with $\mathbf{Q}^T$ and subtracting from (4.122) gives

$$\mathbf{Q}^T \mathbf{w} = \mathbf{0} \tag{4.123}$$

Equations (4.118) and (4.123) together yield

$$\begin{bmatrix} \mathbf{A} + \lambda \mathbf{I} & \mathbf{Q} \\ \mathbf{Q}^T & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{w} \\ \mathbf{a} \end{bmatrix} = \begin{bmatrix} \mathbf{s} \\ \mathbf{0} \end{bmatrix}$$

Hence we have shown that the function given in (4.114) with the weights satisfying (4.48) solves the minimization problem.

Now, it remains to show how to construct functions $\{\boldsymbol{\varphi}_i\}$ satisfying (4.113). They can be obtained as

$$\boldsymbol{\varphi}_i = \boldsymbol{\Psi} * \mathbf{r}_i, \tag{4.124}$$

where $\boldsymbol{\Psi}$ is the $n \times n$ matrix function satisfying $\mathbf{U} * \boldsymbol{\Psi} = \delta \mathbf{I}$. It can be verified that the above construction satisfies (4.113) as follows:

$$\begin{aligned} B(\boldsymbol{\varphi}_i, \mathbf{g}) &= \langle \mathbf{U} * \boldsymbol{\varphi}_i, \mathbf{g} \rangle \\ &= \langle \mathbf{U} * \boldsymbol{\Psi} * \mathbf{r}_i, \mathbf{g} \rangle \\ &= \langle \mathbf{r}_i, \mathbf{g} \rangle, \text{ since } \mathbf{U} * \boldsymbol{\Psi} = \delta \mathbf{I} \end{aligned}$$

$\boldsymbol{\Psi}$ is the Green's function of the operator $\mathbf{U}$.

Theorem 5: Vector Splines

The goal is to find $\boldsymbol{\Psi}$ such that

$$\mathbf{U} * \boldsymbol{\Psi} = \delta \mathbf{I}$$

The Fourier domain equivalent of this equation reads

$$\mathbf{U}(\boldsymbol{\omega}) \boldsymbol{\Psi}(\boldsymbol{\omega}) = \mathbf{I},$$

which implies

$$\boldsymbol{\Psi}(\boldsymbol{\omega}) = \mathbf{U}^{-1}(\boldsymbol{\omega}).$$

From (4.24), (4.25), and (4.26), $\mathbf{U}(\boldsymbol{\omega})$ is equal to

$$\mathbf{U}(\boldsymbol{\omega}) = \lambda_d \left\|\boldsymbol{\omega}\right\|^{2\gamma_d} \mathbf{U}_1(\boldsymbol{\omega}) + \lambda_c \left\|\boldsymbol{\omega}\right\|^{2\gamma_c} \mathbf{U}_2(\boldsymbol{\omega}),$$

where

$$\begin{aligned} \mathbf{U}_1(\boldsymbol{\omega}) &= \boldsymbol{\omega}\boldsymbol{\omega}^T \\ \mathbf{U}_2(\boldsymbol{\omega}) &= \left\|\boldsymbol{\omega}\right\|^2 \mathbf{I} - \boldsymbol{\omega}\boldsymbol{\omega}^T \end{aligned}$$

Now, it is straightforward to verify that the following hold for $\mathbf{U}_1(\boldsymbol{\omega})$ and $\mathbf{U}_2(\boldsymbol{\omega})$.

$$\begin{aligned} \mathbf{U}_i(\boldsymbol{\omega})\mathbf{U}_j(\boldsymbol{\omega}) &= \mathbf{0}, \quad \text{for } i \neq j \\ \mathbf{U}_i^2(\boldsymbol{\omega}) &= \left\|\boldsymbol{\omega}\right\|^2 \mathbf{U}_i(\boldsymbol{\omega}) \\ \mathbf{U}_1(\boldsymbol{\omega}) + \mathbf{U}_2(\boldsymbol{\omega}) &= \left\|\boldsymbol{\omega}\right\|^2 \mathbf{I} \end{aligned} \tag{4.125}$$

The above set of relations allows to write the inverse of $\mathbf{U}(\boldsymbol{\omega})$ in a simple form:

$$\mathbf{U}^{-1}(\boldsymbol{\omega}) = \frac{1}{\lambda_d} \frac{1}{\left\|\boldsymbol{\omega}\right\|^{2\gamma_d+4}} \mathbf{U}_1(\boldsymbol{\omega}) + \frac{1}{\lambda_c} \frac{1}{\left\|\boldsymbol{\omega}\right\|^{2\gamma_c+4}} \mathbf{U}_2(\boldsymbol{\omega}) \tag{4.126}$$

Hence, the Green's function $\boldsymbol{\Psi}$ is given by

$$\boldsymbol{\Psi}(\mathbf{x}) = \mathcal{F}^{-1}\hat{\boldsymbol{\Psi}}(\boldsymbol{\omega}) = \frac{1}{\lambda_d}\mathbf{U}_1(\mathbf{x}) * \mathcal{F}^{-1}\left[\frac{1}{\left\|\boldsymbol{\omega}\right\|^{2\gamma_d+4}}\right] + \frac{1}{\lambda_c}\mathbf{U}_2(\mathbf{x}) * \mathcal{F}^{-1}\left[\frac{1}{\left\|\boldsymbol{\omega}\right\|^{2\gamma_c+4}}\right].$$

The functions $\mathbf{U}_1(\mathbf{x})$ and $\mathbf{U}_2(\mathbf{x})$ are obtained as

$$\begin{aligned} \mathbf{U}_1(\mathbf{x}) &= \mathcal{F}^{-1}\mathbf{U}_1(\boldsymbol{\omega}), \\ &= -\mathbf{D}(\mathbf{x}), \\ \mathbf{U}_2(\mathbf{x}) &= \mathcal{F}^{-1}\mathbf{U}_2(\boldsymbol{\omega}), \\ &= -L(\mathbf{x})\mathbf{I} + \mathbf{D}(\mathbf{x}), \end{aligned}$$

where

$$\begin{aligned} L(\mathbf{x}) &= \sum_{i=1}^{n} \frac{\partial^2}{\partial x_i^2}\delta \\ \{\mathbf{D}(\mathbf{x})\}_{ij} &= \frac{\partial^2}{\partial x_i \partial x_j}\delta \end{aligned}$$

This yields

$$\mathbf{\Psi}(\mathbf{x}) = -\frac{1}{\lambda_d}\mathbf{D}(\mathbf{x}) * \mathcal{F}^{-1}\left[\frac{1}{\|\boldsymbol{\omega}\|^{2\gamma_d+4}}\right] - \frac{1}{\lambda_c}\left(L(\mathbf{x})\mathbf{I} - \mathbf{D}(\mathbf{x})\right) * \mathcal{F}^{-1}\left[\frac{1}{\|\boldsymbol{\omega}\|^{2\gamma_c+4}}\right].$$

Now, from [67],

$$\mathcal{F}^{-1}\left[\frac{1}{\|\boldsymbol{\omega}\|^{2m}}\right] = \psi_{2m-n}(\mathbf{x})$$

where

$$\psi_l(\mathbf{x}) = \begin{cases} c_1\|\mathbf{x}\|^l \log\|\mathbf{x}\|, & \text{if } l \text{ is even,} \\ c_2\|\mathbf{x}\|^l, & \text{otherwise,} \end{cases}$$

for some constants c_1 and c_2. For practical purposes, one can ignore these constants, since they can be absorbed into the parameters λ_d and λ_c.

Finally, $\mathbf{\Psi}(\mathbf{x})$ becomes

$$\mathbf{\Psi}(\mathbf{x}) = \frac{1}{\lambda_d}\mathbf{D}(\mathbf{x}) * \psi_{2\gamma_d+4-n}(\mathbf{x}) + \frac{1}{\lambda_c}\left(L(\mathbf{x})\mathbf{I} - \mathbf{D}(\mathbf{x})\right) * \psi_{2\gamma_c+4-n}(\mathbf{x}).$$

Theorem 6: Irrotational and solenoidal Green's functions

Recall that the functions $\boldsymbol{\psi}_j^{(d)}$ and $\boldsymbol{\psi}_j^{(c)}$ are of the following form:

$$\boldsymbol{\psi}_j^{(d)}(\mathbf{x}) = \left[\frac{\partial^2\psi_{l_1}(\mathbf{x})}{\partial x_j \partial x_1} \cdots \frac{\partial^2\psi_{l_1}(\mathbf{x})}{\partial x_j \partial x_n}\right]^T \tag{4.127}$$

$$\boldsymbol{\psi}_j^{(c)}(\mathbf{x}) = \mathbf{e}_j \triangle\psi_{l_2}(\mathbf{x}) - \left[\frac{\partial^2\psi_{l_2}(\mathbf{x})}{\partial x_j \partial x_1} \cdots \frac{\partial^2\psi_{l_2}(\mathbf{x})}{\partial x_j \partial x_n}\right]^T \tag{4.128}$$

Note that $\boldsymbol{\psi}_j^{(d)}(\mathbf{x})$ in (4.127) is the gradient of the scalar functions $\frac{\partial\psi_{l_1}(\mathbf{x})}{\partial x_j}$; i.e.,

$$\boldsymbol{\psi}_j^{(d)}(\mathbf{x}) = \nabla\left(\frac{\partial\psi_{l_1}(\mathbf{x})}{\partial x_j}\right).$$

Since $Curl$ of gradient is zero,

$$Curl\,\boldsymbol{\psi}_j^{(d)}(\mathbf{x}) = \mathbf{0}.$$

Now, we evaluate $Div\,\boldsymbol{\psi}_j^{(c)}(\mathbf{x})$.

$$\begin{aligned} Div\,\boldsymbol{\psi}_j^{(c)}(\mathbf{x}) &= \nabla\cdot\mathbf{e}_j\triangle\psi_{l_2}(\mathbf{x}) - \nabla\cdot\left[\frac{\partial^2\psi_{l_2}(\mathbf{x})}{\partial x_j\partial x_1}\ \cdots\ \frac{\partial^2\psi_{l_2}(\mathbf{x})}{\partial x_j\partial x_n}\right]^T \\ &= \nabla\cdot\mathbf{e}_j\triangle\psi_{l_2}(\mathbf{x}) - \nabla\cdot\nabla\left(\frac{\partial\psi_{l_2}(\mathbf{x})}{\partial x_j}\right) \\ &= \nabla\cdot\mathbf{e}_j\triangle\psi_{l_2}(\mathbf{x}) - \triangle\left(\frac{\partial\psi_{l_2}(\mathbf{x})}{\partial x_j}\right) \\ &= \frac{\partial}{\partial x_j}\triangle\psi_{l_2}(\mathbf{x}) - \frac{\partial}{\partial x_j}\triangle\psi_{l_2}(\mathbf{x}) \\ &= 0 \end{aligned}$$

Theorem 7: The columns of Green's matrix function are related

The functions $\boldsymbol{\psi}_j^{(d)}$ and $\boldsymbol{\psi}_j^{(c)}$ are expressed in Fourier as follows:

$$\begin{aligned} \hat{\boldsymbol{\psi}}_j^{(d)}(\boldsymbol{\omega}) &= \hat{\mathbf{g}}_j(\boldsymbol{\omega})\hat{\psi}_{l_1}(\boldsymbol{\omega}) \\ \hat{\boldsymbol{\psi}}_j^{(c)}(\boldsymbol{\omega}) &= \hat{\mathbf{l}}_j(\boldsymbol{\omega})\hat{\psi}_{l_2}(\boldsymbol{\omega}) - \hat{\mathbf{g}}_j(\boldsymbol{\omega})\hat{\psi}_{l_2}(\boldsymbol{\omega}) \end{aligned}$$

where

$$\begin{aligned} \hat{\mathbf{l}}_j(\boldsymbol{\omega}) &= \mathbf{e}_j\left\|\boldsymbol{\omega}\right\|^2 \\ \hat{\mathbf{g}}_j(\boldsymbol{\omega}) &= \omega_j\boldsymbol{\omega} \end{aligned}$$

Note that the functions $\hat{\mathbf{l}}_j(\boldsymbol{\omega})$ and $\hat{\mathbf{g}}_j(\boldsymbol{\omega})$ satisfy the following relations:

$$\begin{aligned} \hat{\mathbf{l}}_{j+1}(\boldsymbol{\omega}) &= \mathbf{P}\hat{\mathbf{l}}_j(\mathbf{P}^T\boldsymbol{\omega}) \\ \hat{\mathbf{g}}_{j+1}(\boldsymbol{\omega}) &= \mathbf{P}\hat{\mathbf{g}}_j(\mathbf{P}^T\boldsymbol{\omega}) \end{aligned}$$

where

$$\mathbf{P} = \begin{cases} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, & \text{for } n = 2, \\ \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}, & \text{for } n = 3. \end{cases}$$

Since $\hat{\psi}_l(\boldsymbol{\omega})$ is a radial function, the above relations imply

$$\begin{aligned} \hat{\boldsymbol{\psi}}_{j+1}^{(d)}(\boldsymbol{\omega}) &= \mathbf{P}\hat{\boldsymbol{\psi}}_j^{(d)}(\mathbf{P}^T\boldsymbol{\omega}) \\ \hat{\boldsymbol{\psi}}_{j+1}^{(c)}(\boldsymbol{\omega}) &= \mathbf{P}\hat{\boldsymbol{\psi}}_j^{(c)}(\mathbf{P}^T\boldsymbol{\omega}) \end{aligned}$$

Chapter 5

Vector Field Reconstruction from Non-uniform Projected Samples: B-spline Solution

Summary

We address the problem of reconstructing a 2D vector field, $\mathbf{v} : \mathbb{R}^2 \to \mathbb{R}^2$, from a set of projected samples of the form $s_i = d_i^T \mathbf{v}(\mathbf{x}_i)$, where $\{\mathbf{x}_i\}$ are sampling locations, and $\{\mathbf{d}_i\}$ are projection directions. We formulate the reconstruction task as finding the minimizer of the functional $J_a(\mathbf{f}) = \sum_i (\mathbf{d}_i^T \mathbf{v}(\mathbf{x}_i) - s_i)^2 + \lambda J(\mathbf{v})$, where J is a quadratic plausibility criterion given by $J(\mathbf{f}) = \lambda_d \mathcal{D}_{m_d}(\operatorname{div} \mathbf{v}) + \lambda_d \mathcal{D}_{m_c}(\operatorname{rot} \mathbf{v})$, and where $\mathcal{D}_m$ is Duchon's semi-norm of order m. We consider the case $m_d, m_c \in [0, 1]$. We search for the solution within the space of uniform B-splines and interpret our method as the computationally efficient alternative for vector splines method. We show that the required solution can be obtained by solving a sparse linear system of equations. Exploiting the refinable nature of B-splines, we develop a fast multiresolution-multigrid algorithm. We demonstrate the computational advantages of this new method over the analytical vector-spline method. We

apply our method for recovering full velocity field from pulsed wave ultrasound Doppler data and demonstrate its utility for clinical echocardiography.

5.1 Motivation and Main Contributions

Our development of the previous vector-spline method was motivated by our desire to reconstruct a full motion field from partial projection data provided by clinical echocardiography. Echocardiography relies on Pulsed wave ultrasound Doppler (PWD) imaging (Chapter 1) as a tool for cardiac flow visualization. We recall here the imaging scheme briefly. The imaging system sends a periodic pulse train along a set of scan lines and measures the backscattered signal. By analyzing the Doppler frequency shift in the received signal, the system retrieves a set of velocity estimates; these estimates are the axial velocities of some points in the myocardium, where the term "axial velocity" refers to the projected velocity along the beam direction. These axial components are resampled on a regular grid and presented in a color coded form that is known as the color flow image. See [5] for a comprehensive treatment of the various apects of such systems. In the context of clinical echocardiography, PWD imaging is also known as color Doppler imaging when used to asses blood flow, and tissue Doppler imaging, when applied to tissue motion.

The color flow image sequence contains the instantaneous velocity information, but it is incomplete; in particular, the system is blind to the motion that is orthogonal to the beam. Nevertheless, such a partial velocity field has been used for the determination of quantitative parameters such as flow volume [68, 69, 70]. Further, qualitative motion analysis obtained from this kind of data was found to be clinically useful in several instances [71, 72, 73, 74, 75, 76]. However, clinical echocardiography poses more complex problems such as discriminating the regions of wall thickening and thinning. Such tasks call for the recovery of the true velocity field (vector field) of the myocardium.

We propose to recover the full velocity field from PWD data by fusing multiple view acquisitions of a particular region of interest. This problem fits well with the theoretical framework developed in Chapter 4. However, since fusing multiple view acquisitions results in a large number of samples,

we are faced with a considerable computational challenge. Unfortunately, the vector spline method proposed in Chapter 4 becomes computationally impractical when the number of samples is large. This numerical inefficiency is analogous to that of the thin-plate spline method for the scalar problem. We recall here the factors that degrade the numerical efficiency: (i) the basis functions are poorly conditioned since their magnitude grows from the center; (ii) it requires to solve a dense, ill-conditioned linear system of equations; (iii) the complexity grows rapidly with the number of input samples; (iii) after computing the solution, displaying the result in a regular grid calls for an expensive resampling step.

We propose to overcome this computational bottleneck by approximating the theoretical vector-spline within a space of uniform vector B-splines. The resulting computational advantages of this alternative method are analogous to that of the method proposed for the scalar problem in Chapter 3. The advantages are the following: (i) the reconstruction is simplified considerably and only amounts to solving a well-conditioned, sparse linear system of equations; (ii) the complexity now essentially depends on the reconstruction grid size (step size for B-splines), which acts as a trade-off parameters that allows to find a compromise between the computational cost and the reconstruction accuracy; (iii) the complexity is essentially independent of the input samples; (iv) refinable nature of the present basis functions allows to devise a fast multiresolution algorithm.

In Section 5.2, we set up the reconstruction problem and the derive the linear system of equations yielding the required solution. In order to demonstrate the importance of using the appropriate regularizer, we derive the linear systems for three cases: (i) least squares approximation of the data without regularization; (ii) approximation with scalar regularization based on Duchon's semi-norm; (iii) approximation using the vector regularization proposed in Chapter 4. We also derive the interscale relation that links together the linear systems of equations specifying solutions in different levels of resolution. In Section 5.3 we provide the numerical algorithm, which is a modified version of the scalar one proposed in Chapter 3. We apply our method for PWD imaging in Section 5.4, where we demonstrate how the partial nature of the data demands the use of appropriately tuned vector regularizer. We provide some reconstruction results on clinical data.

5.2 The Proposed Method

Let $\mathbf{v}(\mathbf{x}) = [u(x,y)\ v(x,y)]$ be the velocity field. Our aim is to recover $\mathbf{v}(\mathbf{x})$ from the given Doppler measurement set $\{\mathbf{x}_i, \mathbf{d}_i, m_i\}$. We search for the solution in the space of uniform B-splines; in other words, we restrict the velocity field to be of the following form:

$$u(x,y) = \sum_{k=0}^{N_g-1} \sum_{l=0}^{N_g-1} c^u_{k,l} \beta^n(x/a-k)\beta^n(y/a-l) \tag{5.1}$$

$$v(x,y) = \sum_{k=0}^{N_g-1} \sum_{l=0}^{N_g-1} c^v_{k,l} \beta^n(x/a-k)\beta^n(y/a-l) \tag{5.2}$$

Here, β^n is the B-spline of degree n and a is the step size that controls the accuracy. The idea is to formulate the reconstruction as the minimizer of a quadratic functional. Consequently, the expansion coefficients $c_{k,l}$ are expressed as a solution of a linear system of equation. We study three forms of the quadratic functional.

5.2.1 Least Squares Method (LS)

The least squares B-spline solution is the minimizer of the following quadratic cost functional:

$$J_{LS}(\mathbf{v}) = \sum_{i=1}^{N} \left(\mathbf{d}_i^T \mathbf{v}(\mathbf{x}_i) - m_i\right)^2 \tag{5.3}$$

Let $\mathbf{c}_u = [\cdots c^u_{k,l} \cdots]^T$, $\mathbf{c}_v = [\cdots c^v_{k,l} \cdots]^T$, and $\mathbf{c} = [\mathbf{c}_u^T\ \mathbf{c}_v^T]^T$. Now, the task is express the above functional in terms of $\mathbf{c}$. Let $\mathbf{t}_u = [\cdots\ u(\mathbf{x}_i)\ \cdots]^T$ and $\mathbf{t}_v = [\cdots\ v(\mathbf{x}_i)\ \cdots]^T$. The vectors $\mathbf{t}_u$ and $\mathbf{t}_v$ are computed as

$$\mathbf{t}_u = \mathbf{S}_s \mathbf{c}_u,$$

$$\mathbf{t}_v = \mathbf{S}_s \mathbf{c}_v,$$

where the sample matrix $\mathbf{S}_s$ is defined as

$$\{\mathbf{S}_s\}_{i,N_g l+k} = \beta^n(x_i/a-k)\beta^n(y_i/a-l). \tag{5.4}$$

Let $\mathbf{W}_x$ and $\mathbf{W}_y$ be the diagonal matrices defined by

$$\{\mathbf{W}_x\}_{ii} = d_{xi},$$

$$\{\mathbf{W}_y\}_{ii} = d_{yi},$$

where d_{xi} and d_{yi} are the components of $\mathbf{d}_i$. Further, let

$$\mathbf{t}_{vd} = [\cdots\ \mathbf{d}_i^T\mathbf{v}(\mathbf{x}_i)\ \cdots]^T.$$

It can be verified that

$$\mathbf{t}_{vd} = \mathbf{W}_x\mathbf{t}_u + \mathbf{W}_y\mathbf{t}_v = \mathbf{W}_x\mathbf{S}\mathbf{c}_u + \mathbf{W}_y\mathbf{S}\mathbf{c}_v$$

We now define,

$$\mathbf{W} = [\mathbf{W}_x\ \mathbf{W}_y], \tag{5.5}$$

$$\mathbf{S} = \begin{bmatrix} \mathbf{S}_s & \mathbf{0} \\ \mathbf{0} & \mathbf{S}_s \end{bmatrix}. \tag{5.6}$$

Then, $\mathbf{t}_{vd}$ becomes

$$\mathbf{t}_{vd} = \mathbf{W}\mathbf{S}\mathbf{c}.$$

Let $\mathbf{m} = [\cdots\ m_i\ \cdots]^T$. Putting all these elements together, we express (5.3) in the following standard matrix form:

$$\begin{aligned} J_{LS}(\mathbf{v}) &= \|\mathbf{t}_{vd} - \mathbf{m}\|^2 = \|\mathbf{W}\mathbf{S}\mathbf{c} - \mathbf{m}\|^2 \\ &= \mathbf{c}^T\mathbf{A}_{LS}\mathbf{c} - 2\mathbf{c}^T\mathbf{B}^T\mathbf{m} + \mathbf{m}^T\mathbf{m} \end{aligned} \tag{5.7}$$

where

$$\mathbf{A}_{LS} = \mathbf{S}^T\mathbf{W}^T\mathbf{W}\mathbf{S}, \tag{5.8}$$

$$\mathbf{B} = \mathbf{W}\mathbf{S}. \tag{5.9}$$

Note that $\mathbf{A}_{LS}$ is a square matrix of size $2N_g^2$. Finally, the least squares reconstruction—i.e., the minimizer of (5.7)—is given by the solution of the following equation:

$$\mathbf{c}_{LS} = \mathbf{A}_{LS}^{-1}\mathbf{b}, \tag{5.10}$$

where $\mathbf{b} = \mathbf{B}^T\mathbf{m}$. In the case where $\mathbf{A}_{LS}$ is not of full rank, we consider the minimum norm solution which is given by $\mathbf{c}_{LS} = \mathbf{A}_{LS}^{\dagger}\mathbf{b}$, where $\mathbf{A}_{LS}^{\dagger}$ is the generalized (Moore-Penrose) inverse of $\mathbf{A}_{LS}$.

5.2.2 Regularized Least Squares Method (RLS)

The reconstruction in the regularized least squares methods is the minimizer of an extended cost functional; it is obtained by adding a smoothness functional to the original least squares functional, i.e.,

$$J_{RLS}(\mathbf{v}) = J_{LS}(\mathbf{v}) + \lambda R(\mathbf{v}) \tag{5.11}$$

A typical choice of smoothness functional, $R(\mathbf{v})$, that is frequently used for the estimation of deformation fields is a weighted sum of membrane spline and thin-plate spline regularizers [30, 48] applied to each component. Specifically, we have

$$R(\mathbf{v}) = \lambda_1(\mathcal{D}_1(u) + \mathcal{D}_1(v)) + \lambda_2(\mathcal{D}_2(u) + \mathcal{D}_2(v)) \tag{5.12}$$

where $\mathcal{D}_m$ is Duchon's semi-norm of order m, which is given by

$$\mathcal{D}_1(u) = \int\int \left(\frac{\partial u(x,y)}{\partial x}\right)^2 + \left(\frac{\partial u(x,y)}{\partial y}\right)^2 dxdy, \tag{5.13}$$

$$\mathcal{D}_2(u) = \int\int \left(\frac{\partial^2 u(x,y)}{\partial x^2}\right)^2 + 2\left(\frac{\partial^2 u(x,y)}{\partial x \partial y}\right)^2 + \left(\frac{\partial^2 u(x,y)}{\partial y^2}\right)^2 dxdy, \tag{5.14}$$

with $\lambda_1 \geq 0$ and $\lambda_2 \geq 0$. We set one these two parameters to zero, so that the regularization functional corresponds to the original scale invariant semi-norm. It is important to note that this fairly standard regularizer does not have any coupled terms, meaning that it does not enforce any special relationship between velocity components.

The main task now is to express $\mathcal{D}_1(.)$ and $\mathcal{D}_2(.)$ in terms of expansion coefficients. In the Appendix, we show that these can be written down relatively simply in terms of digital filtering operations and l_2-inner products as given below:

$$\mathcal{D}_1(u) = \left\langle g(k,l) * c^u_{k,l},\ c^u_{k,l} \right\rangle_{l_2(\mathbb{Z}^2)} \tag{5.15}$$

$$\mathcal{D}_2(u) = \left\langle l(k,l) * c^u_{k,l},\ c^u_{k,l} \right\rangle_{l_2(\mathbb{Z}^2)} \tag{5.16}$$

The digital filters g and l are conveniently characterized by their z-transforms

$G(z_1, z_2)$ and $L(z_1, z_2)$. For B-splines they take the following form:

$$\begin{aligned} G(z_1, z_2) &= (z_1 + 2 + z_1^{-1})B^{2n-1}(z_1)B^{2n+1}(z_2) \\ &\quad + B^{2n+1}(z_1)(z_2 + 2 + z_2^{-1})B^{2n-1}(z_2) \qquad (5.17) \\ L(z_1, z_2) &= \frac{1}{a^2}\left[(z_1 + 2 + z_1^{-1})^2 B^{2n-3}(z_1)B^{2n+1}(z_2)\right. \\ &\quad + 2(z_1 + 2 + z_1^{-1})B^{2n-1}(z_1)(z_2 + 2 + z_2^{-1})B^{2n-1}(z_2) \\ &\quad \left. + B^{2n+1}(z_1)(z_2 + 2 + z_2^{-1})^2 B^{2n-3}(z_2)\right] \qquad (5.18) \end{aligned}$$

where $B^n(z) = \sum_{k\in\mathbb{Z}} \beta^n(k) z^{-1}$ denotes the z-transform of the discrete B-spline of degree n. Note that these filters can also be obtained from the expression derived in Chapter 3. Equations (5.15) and (5.16) can also be written as

$$\mathcal{D}_1(u) = \mathbf{c}_u^T \mathbf{R}_g \mathbf{c}_u \qquad (5.19)$$

$$\mathcal{D}_2(u) = \mathbf{c}_u^T \mathbf{R}_l \mathbf{c}_u \qquad (5.20)$$

where $\mathbf{R}_g$ and $\mathbf{R}_l$ are the circulant matrices corresponding to $G(z_1, z_2)$ and $L(z_1, z_2)$, respectively. This together with (5.7) yields the following expression for $J_{RLS}(\mathbf{v})$:

$$J_{RLS}(\mathbf{v}) = \mathbf{c}^T \mathbf{A}_{LS} \mathbf{c} - 2\mathbf{c}^T \mathbf{B}^T \mathbf{m} + \mathbf{m}^T \mathbf{m} + \lambda_1 \mathbf{c}^T \mathbf{R}_{ge} \mathbf{c} + \lambda_1 \mathbf{c}^T \mathbf{R}_{le} \mathbf{c}, \quad (5.21)$$

where

$$\mathbf{R}_{ge} = \begin{bmatrix} \mathbf{R}_g & \mathbf{0} \\ \mathbf{0} & \mathbf{R}_g \end{bmatrix}$$

$$\mathbf{R}_{le} = \begin{bmatrix} \mathbf{R}_l & \mathbf{0} \\ \mathbf{0} & \mathbf{R}_l \end{bmatrix}$$

Thus the minimizer of (5.21) can be formally expressed as

$$\mathbf{c}_{RLS} = \mathbf{A}_{RLS}^{-1} \mathbf{b},$$

where $\mathbf{A}_{RLS} = \mathbf{A}_{LS} + \lambda_1 \mathbf{R}_{ge} + \lambda_2 \mathbf{R}_{le}$. In practice, we do not compute the inverse but we solve the system using direct or iterative methods that are efficient for sparse systems.

5.2.3 Vector Regularized Least Squares (VRLS)

We now introduce the vector regularization proposed in 4. The criterion to minimize is

$$J_{VRLS}(\mathbf{v}) = J_{LS}(\mathbf{v}) + \lambda R_E(\mathbf{v}), \qquad (5.22)$$

where $R_E(\mathbf{v})$ is the vector regularizer. The functional, $R_E(\mathbf{v})$, is based on the *divergence* and *curl* of the velocity field, which are expressed as follows for the present set up:

$$\begin{aligned} \operatorname{div} \mathbf{v}(\mathbf{x}) &= \partial_x u(x,y) + \partial_y v(x,y), && (5.23) \\ \operatorname{rot} \mathbf{v}(\mathbf{x}) &= -\partial_y u(x,y) + \partial_x v(x,y). && (5.24) \end{aligned}$$

The divergence of the velocity field quantifies the rate change of the density of the medium at a given point. The curl of a velocity field, on the other hand, is equal to the twice the angular velocity within an infinitesimal neighborhood at the point of interest. Thus, we construct $R_E(\mathbf{v})$ as given below:

$$\begin{aligned} R_E(\mathbf{v}) &= \lambda_{d_0} \int \left(\operatorname{div} \mathbf{v}(\mathbf{x})\right)^2 d\mathbf{x} + \lambda_{d_1} \int \left\| \nabla \left(\operatorname{div} \mathbf{v}(\mathbf{x})\right) \right\|^2 d\mathbf{x} \\ &+ \lambda_{c_0} \int \left(\operatorname{rot} \mathbf{v}(\mathbf{x})\right)^2 d\mathbf{x} + \lambda_{c_1} \int \left\| \nabla \left(\operatorname{rot} \mathbf{v}(\mathbf{x})\right) \right\|^2 d\mathbf{x}. \quad (5.25) \end{aligned}$$

Here we choose only one of the parameters in $\{\lambda_{d_0}, \lambda_{d_1}\}$ to be non-zero, and only one of the parameters in $\{\lambda_{c_0}, \lambda_{c_1}\}$ to be non-zero. Consequently, $R_E(\mathbf{v})$ is a special case of the functional proposed in Chapter 4 which is given by Equations (4.35), (4.31), and (4.32) with the restriction $\gamma_d, \gamma_c \in (0, 1)$.

This restriction is imposed in order to simplify the computations. Nevertheless, this choice is sufficient to include every possible case that has direct physical interpretation in terms of the deformation of the medium undergoing motion. Specifically, since the divergence gives the density change, the first term in the equation (5.25) quantifies the overall compression rate, whereas the second term gives the spatial roughness of this compression rate. Both terms are related to the deformation of the medium. The third term sums up the squared angular velocity. It does not directly quantify the deformation, but will tend to penalize rotations including rigid ones. The last term, on the other hand, is indeed a measure of deformation, as it captures the spatial variation of the angular velocity. Hence, $R_E(\mathbf{v})$ is sufficient to include any a priori knowledge of the type of velocity field and also to specify physically plausible solutions.

In the Appendix, we show that the various terms in (5.25) can be written

as

$$\begin{aligned}
\int \left(Div\,\mathbf{v}(\mathbf{x})\right)^2 d\mathbf{x} &= \left\langle r_{11}(k,l) * c^u_{k,l},\ c^u_{k,l}\right\rangle + \left\langle r_{12}(k,l) * c^v_{k,l},\ c^v_{k,l}\right\rangle \\
&\quad +2\left\langle r_{13}(k,l) * c^u_{k,l},\ c^v_{k,l}\right\rangle, \qquad (5.26)\\
\int \left(Curl\,\mathbf{v}(\mathbf{x})\right)^2 d\mathbf{x} &= \left\langle r_{11}(k,l) * c^u_{k,l},\ c^u_{k,l}\right\rangle + \left\langle r_{12}(k,l) * c^v_{k,l},\ c^v_{k,l}\right\rangle \\
&\quad -2\left\langle r_{13}(k,l) * c^u_{k,l},\ c^v_{k,l}\right\rangle, \qquad (5.27)\\
\int \left\|\nabla\left(Div\,\mathbf{v}(\mathbf{x})\right)\right\|^2 d\mathbf{x} &= \left\langle r_{21}(k,l) * c^u_{k,l},\ c^u_{k,l}\right\rangle + \left\langle r_{22}(k,l) * c^v_{k,l},\ c^v_{k,l}\right\rangle \\
&\quad +2\left\langle r_{23}(k,l) * c^u_{k,l},\ c^v_{k,l}\right\rangle, \qquad (5.28)\\
\int \left\|\nabla\left(Curl\,\mathbf{v}(\mathbf{x})\right)\right\|^2 d\mathbf{x} &= \left\langle r_{21}(k,l) * c^u_{k,l},\ c^u_{k,l}\right\rangle + \left\langle r_{22}(k,l) * c^v_{k,l},\ c^v_{k,l}\right\rangle \\
&\quad -2\left\langle r_{23}(k,l) * c^u_{k,l},\ c^v_{k,l}\right\rangle. \qquad (5.29)
\end{aligned}$$

where the underlying digital filters are defined as follows:

$$\begin{aligned}
R_{11}(z_1,z_2) &= (z_1+2+z_1^{-1})B^{2n-1}(z_1)B^{2n+1}(z_2) \qquad (5.30)\\
R_{12}(z_1,z_2) &= B^{2n+1}(z_1)(z_2+2+z_2^{-1})B^{2n-1}(z_2) \qquad (5.31)\\
R_{13}(z_1,z_2) &= B'^{2n+1}(z_1)B'^{2n+1}(z_2^{-1}) \qquad (5.32)\\
R_{21}(z_1,z_2) &= \frac{1}{a^2}\left[(z_1+2+z_1^{-1})^2 B^{2n-3}(z_1)B^{2n+1}(z_2)\right.\\
&+ \left.(z_1+2+z_1^{-1})B^{2n-1}(z_1)(z_2+2+z_2^{-1})B^{2n-1}(z_2)\right] \qquad (5.33)\\
R_{22}(z_1,z_2) &= \frac{1}{a^2}\left[B^{2n+1}(z_1)(z_2+2+z_2^{-1})^2 B^{2n-3}(z_2)\right.\\
&+ \left.(z_1+2+z_1^{-1})B^{2n-1}(z_1)(z_2+2+z_2^{-1})B^{2n-1}(z_2)\right] \qquad (5.34)\\
R_{23}(z_1,z_2) &= \frac{1}{a^2}\left[(z_1+2+z_1^{-1})D^{2n-2}(z_1)B'^{2n+1}(z_2^{-1})\right.\\
&+ \left.D^{2n+1}(z_1)(z_2+2+z_2^{-1})B'^{2n-2}(z_2^{-1})\right] \qquad (5.35)
\end{aligned}$$

Here $D^n(z) = \sum_{k\in\mathbb{Z}}\left(\beta^n(x+1/2)-\beta^n(x-1/2)\right)z^{-1}$. In matrix form, Equations (5.26), (5.27), (5.28), and (5.29) read

$$\begin{aligned}
\int \left(Div\,\mathbf{v}(\mathbf{x})\right)^2 d\mathbf{x} &= \mathbf{c}_u^T\mathbf{R}_{11}\mathbf{c}_u + \mathbf{c}_v^T\mathbf{R}_{12}\mathbf{c}_v + 2\mathbf{c}_u^T\mathbf{R}_{13}\mathbf{c}_v \qquad (5.36)\\
\int \left(Curl\,\mathbf{v}(\mathbf{x})\right)^2 d\mathbf{x} &= \mathbf{c}_u^T\mathbf{R}_{12}\mathbf{c}_u + \mathbf{c}_v^T\mathbf{R}_{11}\mathbf{c}_v - 2\mathbf{c}_u^T\mathbf{R}_{13}\mathbf{c}_v \qquad (5.37)\\
\int \left\|\nabla\left(Div\,\mathbf{v}(\mathbf{x})\right)\right\|^2 d\mathbf{x} &= \mathbf{c}_u^T\mathbf{R}_{21}\mathbf{c}_u + \mathbf{c}_v^T\mathbf{R}_{22}\mathbf{c}_v + 2\mathbf{c}_u^T\mathbf{R}_{23}\mathbf{c}_v \qquad (5.38)\\
\int \left\|\nabla\left(Div\,\mathbf{v}(\mathbf{x})\right)\right\|^2 d\mathbf{x} &= \mathbf{c}_u^T\mathbf{R}_{22}\mathbf{c}_u + \mathbf{c}_v^T\mathbf{R}_{21}\mathbf{c}_v - 2\mathbf{c}_u^T\mathbf{R}_{23}\mathbf{c}_v \qquad (5.39)
\end{aligned}$$

The cost functional, $J_{VRLS}(\mathbf{v})$, now becomes

$$\begin{aligned} J_{VRLS}(\mathbf{v}) &= \mathbf{c}^T\mathbf{A}_{LS}\mathbf{c} - 2\mathbf{c}^T\mathbf{B}^T\mathbf{m} + \mathbf{m}^T\mathbf{m} + \lambda_{d_0}\mathbf{c}^T\mathbf{R}_{d_0}\mathbf{c} + \lambda_{c_0}\mathbf{c}^T\mathbf{R}_{c_0}\mathbf{c} \\ &\quad + \lambda_{d_1}\mathbf{c}^T\mathbf{R}_{d_1}\mathbf{c} + \lambda_{c_1}\mathbf{c}^T\mathbf{R}_{d_1}\mathbf{c}, \end{aligned} \tag{5.40}$$

where

$$\mathbf{R}_{d_0} = \begin{bmatrix} \mathbf{R}_{11} & \mathbf{R}_{13} \\ \mathbf{R}_{13} & \mathbf{R}_{12} \end{bmatrix} \tag{5.41}$$

$$\mathbf{R}_{c_0} = \begin{bmatrix} \mathbf{R}_{12} & -\mathbf{R}_{13} \\ -\mathbf{R}_{13} & \mathbf{R}_{11} \end{bmatrix} \tag{5.42}$$

$$\mathbf{R}_{d_1} = \begin{bmatrix} \mathbf{R}_{21} & \mathbf{R}_{23} \\ \mathbf{R}_{23} & \mathbf{R}_{22} \end{bmatrix} \tag{5.43}$$

$$\mathbf{R}_{c_1} = \begin{bmatrix} \mathbf{R}_{22} & -\mathbf{R}_{23} \\ -\mathbf{R}_{23} & \mathbf{R}_{21} \end{bmatrix} \tag{5.44}$$

Note that the functional, $R_E(\mathbf{v})$, includes $R(\mathbf{v})$ as a special case. To show this, we first observe from (5.30), (5.31) and (5.17) that

$$G(z_1, z_2) = R_{11}(z_1, z_2) + R_{12}(z_1, z_2).$$

Similarly, using (5.33), (5.34), and (5.18), we get

$$L(z_1, z_2) = R_{21}(z_1, z_2) + R_{22}(z_1, z_2).$$

These two relations in turn imply

$$\mathbf{R}_{d_0} + \mathbf{R}_{c_0} = \mathbf{R}_{ge}$$

$$\mathbf{R}_{d_1} + \mathbf{R}_{c_1} = \mathbf{R}_{le}$$

Thus we conclude that $R_E(\mathbf{v}) = R(\mathbf{v})$ if $\lambda_{d_0} = \lambda_{c_0} = \lambda_1$ and $\lambda_{d_1} = \lambda_{c_1} = \lambda_2$. This agrees with the relation (4.37).

Finally, the solution for the VRLS method—i.e., the minimizer of (5.40)—is given by

$$\mathbf{c}_{VRLS} = \mathbf{A}_{VRLS}^{-1}\mathbf{b},$$

where $\mathbf{A}_{VRLS} = \mathbf{A}_{LS} + \lambda_{d_0}\mathbf{R}_{d_0} + \lambda_{c_0}\mathbf{R}_{c_0} + \lambda_{d_1}\mathbf{R}_{d_1} + \lambda_{c_1}\mathbf{R}_{c_1}$.

Note that in all the forms of the solution, the size of the linear system is inversely proportional to step size a. Hence a is a trade-off parameter (complexity vs. accuracy).

5.2.4 Interscale Relation

Let us now consider signal reconstructions at different scales. Specifically, let 2^j be the reconstruction grid size (scale j) and

$$u_j(x,y) = \sum_{k}^{(N_g-1)/2^j} \sum_{l}^{(N_g-1)/2^j} c_{k,l}^{u,(j)} \beta(x/2^j - k)\beta(y/2^j - l), \tag{5.45}$$

$$v_j(x,y) = \sum_{k}^{(N_g-1)/2^j} \sum_{l}^{(N_g-1)/2^j} c_{k,l}^{v,(j)} \beta(x/2^j - k)\beta(y/2^j - l) \tag{5.46}$$

be the reconstructing functions, and let

$$\mathbf{c}_{u,(j)} = [\cdots c_{k,l}^{u,(j)} \cdots]^T$$

$$\mathbf{c}_{v,(j)} = [\cdots c_{k,l}^{v,(j)} \cdots]^T$$

$$\mathbf{c}_{(j)} = [\mathbf{c}_{u,(j)}^T \;\; \mathbf{c}_{u,(j)}^T]^T$$

From Equations (5.7), (5.21), and (5.40), the general form of the cost functional for all the three methods can be written as

$$J_{\text{spline}}\left(\mathbf{c}_{(j)}\right) = \mathbf{c}_{(j)}^T \mathbf{A}_{(j)} \mathbf{c}_{(j)} - 2\mathbf{c}_{(j)}^T \mathbf{b}_{(j)} + \mathbf{m}^T\mathbf{m}, \tag{5.47}$$

where $\mathbf{A}_{(j)}$ now denotes the matrix corresponding to any of the above three methods (LS, RLS, or VRLS) with the subscript j signifying the dependence of the matrices on the scale. The solution minimizing the above cost is given by

$$\mathbf{A}_{(j)}\mathbf{c}_{(j)} = \mathbf{b}_{(j)}, \tag{5.48}$$

Let $h(k,l)$ be the two-scale filter, and let $\mathbf{E}_{(j)}$ be the matrix obtained from the circulant matrix corresponding to the filter $h(k,l)$ after suppression of its odd index columns. Define

$$\mathbf{U}_{(j)} = \begin{bmatrix} \mathbf{E}_{(j)} & \mathbf{0} \\ \mathbf{0} & \mathbf{E}_{(j)} \end{bmatrix}$$

The following theorem relates the reconstruction matrices across at two successive scales.

Theorem 8 *The system matrices at scales j and $j+1$ are related by*

$$\mathbf{A}_{(j+1)} = \mathbf{U}_{(j)}^T \mathbf{A}_{(j)} \mathbf{U}_{(j)} \tag{5.49}$$

$$\mathbf{b}_{(j+1)} = \mathbf{U}_{(j)}^T \mathbf{b}_{(j)} \tag{5.50}$$

We will use this relation to construct the multi-resolution algorithm as done in the scalar case (Chapter 3). The proof is given in the Appendix. Even though this theorem can be obtained by a suitable extension of the proof for the scalar case (Chapter 3), we provide here a more general proof without looking into the structure of the regularization filters.

5.3 Numerical Algorithm

We have shown that the reconstruction problem is equivalent to solving a system of linear equations. A key property is that the present system is sparse and well-conditioned in contrast with the theoretical vector spline method where the matrix is dense and ill-conditioned. We also have the flexibility to choose the step size a with the guarentee that the solution converges to the analytical one when a is sufficiently small. However, for most of the practical cases, sufficient accuracy is achieved with a fairly large value of a. Eventually, solving the system of equations by direct methods (example: Gaussian elimination) will be sufficiently fast (few seconds) on personal computers with a moderate computational power, provided the system of equations is set-up in an efficient sparse format. Nevertheless, iterative methods are more efficient under favorable conditions, especially is the context of a multi-grid solver.

As in the scalar case, the iterative method can be constructed based on the multiresolution strategy. Specifically, one solves the linear system exactly at coarsest resolution and propagates the solution across the scales up to the required resolution; at each stage of propagation, the solution at the current resolution is obtained by an iterative refinement from its initialization, where the later is obtained by expanding the solution from the coarser resolution. As in the scalar case, one can compute the matrices for all the coarse resolutions by using the interscale relation.

In the scalar case, we used multigrid V-cycle for the iterative refinement. In the present problem, one can either use the multigrid V-cycle if the system is diagonally dominant. Otherwise conjugate gradient method can be used.

5.4 Performance Evaluation

We study the performance of the proposed method in the context of velocity field recovery from pulsed wave ultrasound doppler data. We first discuss about the nature of input data and the role of appropriately tuned regularizer in recovering the true velocity field. We then study the reconstruction on a synthetic phantom. Next, we move to real phantom data, where the motion is controlled and known a priori. Finally, we present reconstruction results obtained from echocardiographic patient data, and demonstrate its clinical potential.

5.4.1 Data Indeterminacy and the Role of Regularization

Let us consider a plane rotating with an angular velocity ω about the center (x_c, y_c); its velocity field is given by

$$u(x,y) = -\omega(y - y_c) \tag{5.51}$$

$$v(x,y) = \omega(x - x_c) \tag{5.52}$$

Let the measurement device be a cone beam probe located at the origin. The probe measures the samples of the following function:

$$v_d(x,y) = [d_x(x,y)\ d_y(x,y)] \begin{bmatrix} u(x,y) \\ v(x,y) \end{bmatrix}$$

Here $[d_x(x,y)\ d_y(x,y)]^T$ is the direction of the beam from the probe, whose components, in the present geometry, are given by

$$d_x(x,y) = \frac{x}{\sqrt{x^2+y^2}},\ d_y(x,y) = \frac{y}{\sqrt{x^2+y^2}}.$$

Then $v_d(x,y)$ reads

$$v_d(x,y) = \frac{x\omega y_c - y\omega x_c}{\sqrt{x^2+y^2}}.$$

It is clear from the above equation that we cannot recover the motion parameters $\{\omega, x_c, y_c\}$ individually from the samples of $v_d(x,y)$, no matter how many. We can only recover the products ωx_c and ωy_c. In other words, any rotating plane with an angular velocity, ω_1, and a center of rotation, (x_1, y_1), is a candidate solution, provided $\omega_1 x_1 = \omega x_c$ and $\omega_1 x_1 = \omega y_c$. Hence, there

is no reconstruction method that can recover the motion uniquely. We can resolve this ambiguity if we have the value of one of the components (u or v) at any point. In fact, the minimal sample set (possibly artificial) required to determine all the three parameters is the following: (i) samples of the function $v_d(x, y)$ at two points that do not lie on the same scan line; (ii) a sample of one of the components.

An ideal non-parametric regularized method should recover the true motion from a measurement set, whenever it is possible to recover parametrically. For example, the method should recover the rotational motion from the minimal measurement set described above, since they are sufficient to compute the motion field parametrically. To achieve this, the regularization functional should be rich enough to incorporate the main physical properties of the motion.

5.4.2 Rotating Synthetic Phantom Reconstruction

The synthetic phantom we consider is the vector field described above, which corresponds to a plane rotating with a constant angular velocity. We demonstrated earlier that the single probe measurement set is ambiguous and does not permit the recovery of the motion parameters. We also described a minimal artificial sample set required to recover the center of rotation and the angular velocity of the plane (two samples from the probe and one sample of one of the components). Here, we first demonstrate how the velocity field can be recovered by VRLS from such a minimal set. Then, in the second part, we consider measurements from two probe locations, since it is the only practical way to resolve the ambiguity. We study the performance of all three method on such a measurement set.

Reconstruction from a minimal measurement set

For a rigid rotation, the divergence is zero, and so is the gradient of the curl. Hence the appropriate regularization set up for VRLS is $\{\lambda_{d_0} \to \infty, \lambda_{d_1} = 0, \lambda_{c_0} = 0, \lambda_{c_1} \to \infty\}$. However, since there will be a non-zero gradient of the curl at the boundary, a non-zero value of λ_{c_1} will force the rotation at the boundary to be zero, and hence will give a rather large reconstruction error away from the center. We therefore choose $\{\lambda_{d_0} \to \infty, \lambda_{d_1} = 0, \lambda_{c_0} =$

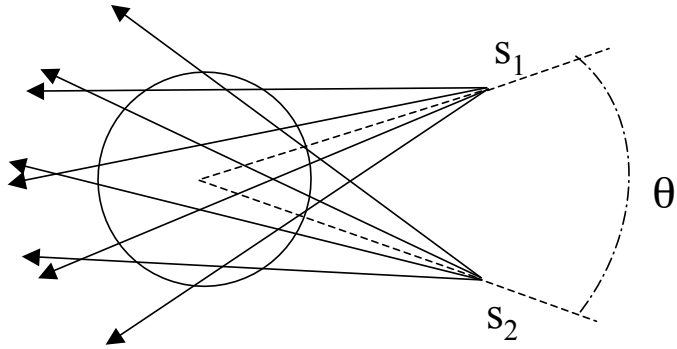


Figure 5.1: Schematic for synthetic phantom experiment

$0, \lambda_{c_1} = 0\}$. With this setting, the VRLS algorithm reconstructs the velocity field perfectly from the minimal measurement set. Also, as we expected, the extra freedom that results from choosing $\lambda_{c_1} = 0$ does not have any effect in the interior. Indeed, we verified that the reconstruction at the interior is essentially insensitive to the value of λ_{c_1}. By contrast, choosing a non-zero value of λ_{c_1} tends to result in a sizable reconstruction error at the boundaries. Finally, note that neither the RLS nor the LS can recover the velocity field from the minimal measurement set.

Reconstruction from simulated measurements for two probe positions

We consider now the Doppler data simulated for two probe locations $\mathbf{s}_1 = [x_1 \ y_1]^T$, and $\mathbf{s}_2 = [x_2 \ y_2]^T$, that subtend an angle, θ, with respect to the center of rotation (figure 5.1). The idealized phantom data is now obtained by sampling two functions $v_1(\mathbf{x})$ and $v_2(\mathbf{x})$, where

$$\begin{aligned} v_j(\mathbf{x}) &= v_j(x,y), \\ &= \mathbf{d}_j^T(\mathbf{x})\mathbf{v}(\mathbf{x}), \\ &= [d_{x_j}(x,y) \ d_{y_j}(x,y)] \begin{bmatrix} u_(x,y) \\ v_(x,y) \end{bmatrix}. \end{aligned}$$

Here, $u(x,y) = -\omega(y - y_c)$, $v(x,y) = \omega(x - x_c)$, and

$$d_{x_j}(x,y) = \frac{x - x_j}{\sqrt{(x-x_j)^2 + (y-y_j)^2}},\ d_{y_j}(x,y) = \frac{y - y_j}{\sqrt{(x-x_j)^2 + (y-y_j)^2}}.$$

We are also adding noise to make the problem more realistic. Each of the functions $v_1(\mathbf{x})$ and $v_2(\mathbf{x})$ are uniformly sampled with respect the polar coordinate systems with origins $\mathbf{s}_1$, and $\mathbf{s}_2$ respectively. However, note that the samples are non-uniform with respect to our cartesian reconstruction system. Let $\{\mathbf{x}_{ji},\ i = 1, \ldots, N\}$ be the set of sample locations for the probe $\mathbf{s}_j$. The input for the algorithm is given by the following list of triplets:

$$\{\mathbf{x}_{ji}, \mathbf{d}_{ji}, m_{ji}, j = 1, 2; i = 1, \ldots, N\},$$

where $\mathbf{d}_{ji} = \mathbf{d}_j(\mathbf{x}_{ji})$, and $m_{ji} = \mathbf{d}_j^T(\mathbf{x}_{ji})\mathbf{v}(\mathbf{x}_{ji}) + n_{ji}$ with n_{ji} being some i.i.d Gaussian measurement noise with variance σ^2. Note that the richness of the data set is controlled by the magnitude of θ, and that the reconstruction becomes easier as θ tends to 90^o. Because of the presence of noise, we need to modify the previous regularization setting for VRLS method; we use a non-zero value for λ_{c_1} to favor the reconstruction of a smooth field. The best value for λ_{c_1} is proportional to the input noise variance. In other words, $\{\lambda_{d_0} \to \infty, \lambda_{d_1} = 0, \lambda_{c_0} = 0, \lambda_{c_1} = k\sigma^2\}$ is the appropriate setting, where k is a adequately chosen positive real number. We choose k somewhat empirically such that $E[m_{ji}^2] = E[(\mathbf{d}_{ji}^T\mathbf{v}_r(\mathbf{x}_{ji}))^2] + \sigma^2$, where $\mathbf{v}_r(\mathbf{x})$ is the reconstructed velocity field, and $E[\bullet]$ denotes the expectation operator. For the RLS method, we need to set $\lambda_1 = 0$, since a rigid rotation contains the first order spatial variation. Hence the appropriate setting is $\{\lambda_1 = 0, \lambda_2 = k\sigma^2\}$. Before presenting our results, we define the following quantities:

$$\begin{aligned}
\text{Input SNR} &= 10 \times \log\left(\frac{E\left[\mathbf{d}_{ji}^T\mathbf{v}(\mathbf{x}_{ji})^2\right]}{\sigma^2}\right) \\
\text{Reconstruction SNR} &= 10 \times \log\left(E\left[\frac{\|\mathbf{v}(\mathbf{k})\|^2}{\|\mathbf{v}_r(\mathbf{k}) - \mathbf{v}(\mathbf{k})\|^2}\right]\right)
\end{aligned}$$

where $\mathbf{v}(\mathbf{x})$ is the true velocity field. Note that the input SNR is computed over the available sample locations, whereas the reconstruction SNR is computed over the reconstruction grid. Figure 5.2 gives reconstruction SNR for VRLS and RLS as a function of θ, for different values of the input SNR. It can be observed that the reconstruction error decreases with increasing θ,

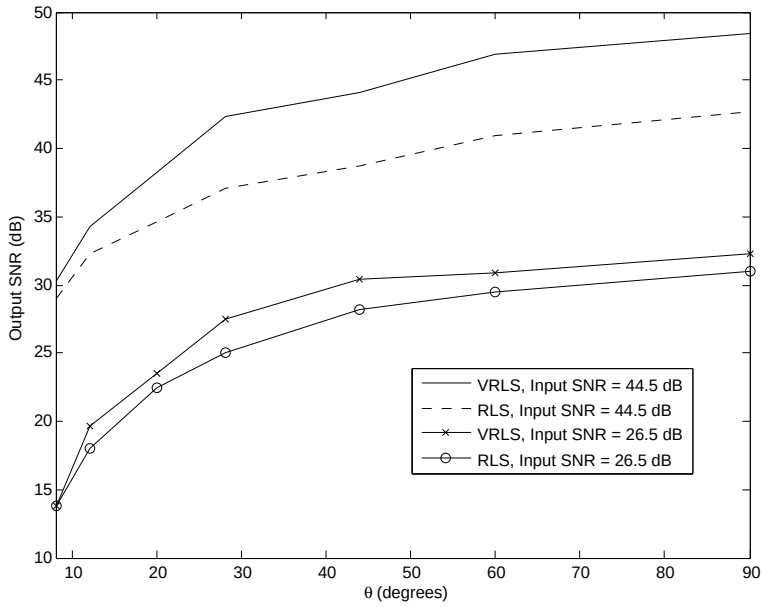


Figure 5.2: Reconstruction error for synthetic phantom experiment.

as one expects. Also, one can clearly see the superiority of VRLS. Figure 5.3 gives the reconstruction SNR for all three methods as a function of the input SNR with $\theta = 30^o$. Note that VRLS performs the best, and that the reconstruction error for the conventional LS approach is far worse (off by more than 30 dB). All the reconstructions were done on a 16×16 grid with $a = 4$ and with the number of samples equal to the number of grid points.

5.4.3 Synthetic Phantom with Non-Rigid Motion

In this experiment, we consider a more complex model for the synthetic phantom. Specifically, we adopt a non-rigid motion model given by

$$\mathbf{v}(\mathbf{x}) = \alpha \begin{bmatrix} \frac{\partial}{\partial x}\varphi(\|\mathbf{x}\|) \\ \frac{\partial}{\partial y}\varphi(\|\mathbf{x}\|) \end{bmatrix} + \beta \begin{bmatrix} \frac{-\partial}{\partial y}\varphi(\|\mathbf{x}\|) \\ \frac{\partial}{\partial x}\varphi(\|\mathbf{x}\|) \end{bmatrix} \tag{5.53}$$

where $\varphi(r) = \frac{1}{2\pi}e^{-r^2/2}$, $\alpha \geq 0$, and $\beta \geq 0$. The first term is the curl-free component and the second is the divergence-free component. In this experiment too, the input data are the measurements simulated for two probes (Figure 5.1) with some i.i.d Gaussian noise. Since the velocity field has a non-zero divergence and a non-zero curl, we have to choose $\lambda_{d_0} = \lambda_{c_0} = 0$. The overall

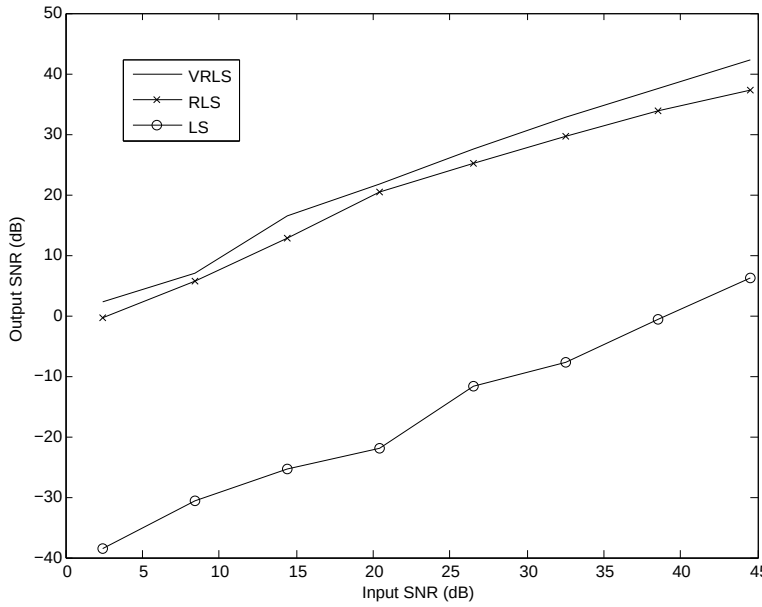


Figure 5.3: Reconstruction error for synthetic phantom experiment.

preferred setting is $\{\lambda_{d_0} = 0, \lambda_{d_1} = k\frac{\beta^2}{\alpha^2+\beta^2}\sigma^2, \lambda_{c_0} = 0, \lambda_{c_1} = k\frac{\alpha^2}{\alpha^2+\beta^2}\sigma^2\}$, where k is chosen as before. The chosen range for the reconstruction grid is $[-3, 3]$ (essential support of the Gaussian) and the average sampling density is 28 samples per unit area. Figure 5.4 compares the reconstruction result for VRLS obtained for different values of the step size a with $\theta = 30^o$. The input SNR is 10 dB. The reconstruction SNR is nearly constant when a is sufficiently small and falls off drastically when it becomes too low. For practical purpose, the choice $a = 2$ offers a good compromise in terms of quality and computational cost.

5.4.4 Real Phantom Experiment

The real phantom that we constructed for this experiment is a cylindrical tissue-mimicking object (sponge) immersed in a water container. The object rotates with a constant angular velocity. Doppler data were acquired, and reconstruction was performed from views differing by 10 degrees. Figure 5.5 shows one frame of the B-mode intensity image with the super-imposed reconstructed motion field. The input SNR is 14.27 dB. The reconstruction

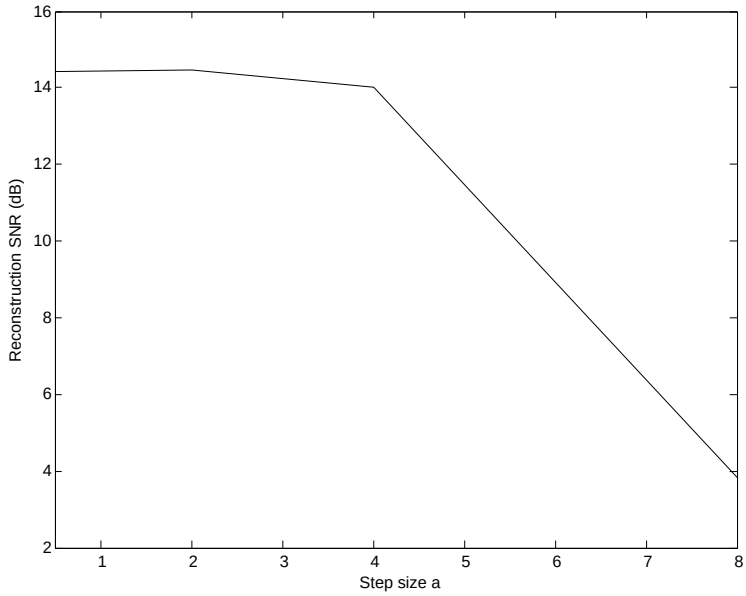


Figure 5.4: Reconstruction error for VRLS with synthetic phantom with non-rigid motion. Input SNR = 10 dB.

SNR for the VRLS method is 13.67 dB. The regularization set-up was the same as that of the previous experiment.

5.4.5 Echocardiographic Data

The echocardiographic data are from a healthy individual and were acquired using Siemens Sequoia Ultrasound System from two echocardiographic standard views: (i) apical long axis view, and (ii) parasternal long axis view, with careful observation of anatomical landmarks to make sure that the images belong to the same cross sectional plane. Along with the Doppler images, the data set contains B-mode intensity images that reveal the anatomical structure of the cross sectional plane. The acquisition was done by an experienced echocardiographer who adjusted the probe such that both acquisitions are from the same cross-sectional plane. They are also synchronized with respect to cardiac cycle by ECG gating. Hence, the premise of this data set is that the image planes are related by a rigid transformation (translation+rotation). To reconstruct the motion field in one image plane, one therefore has to register

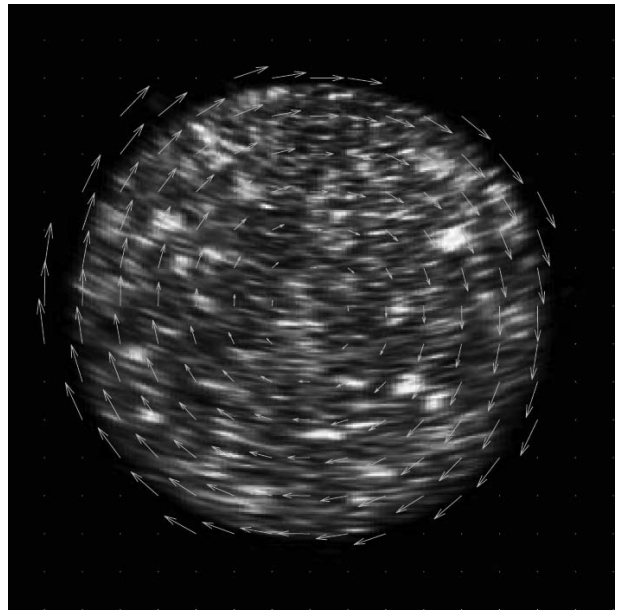
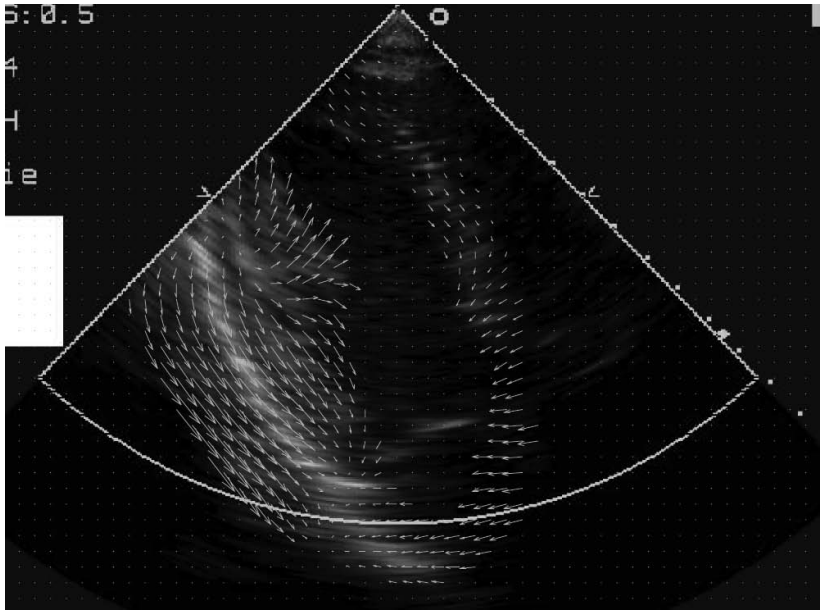


Figure 5.5: Reconstructed motion field for the rotating real phantom.

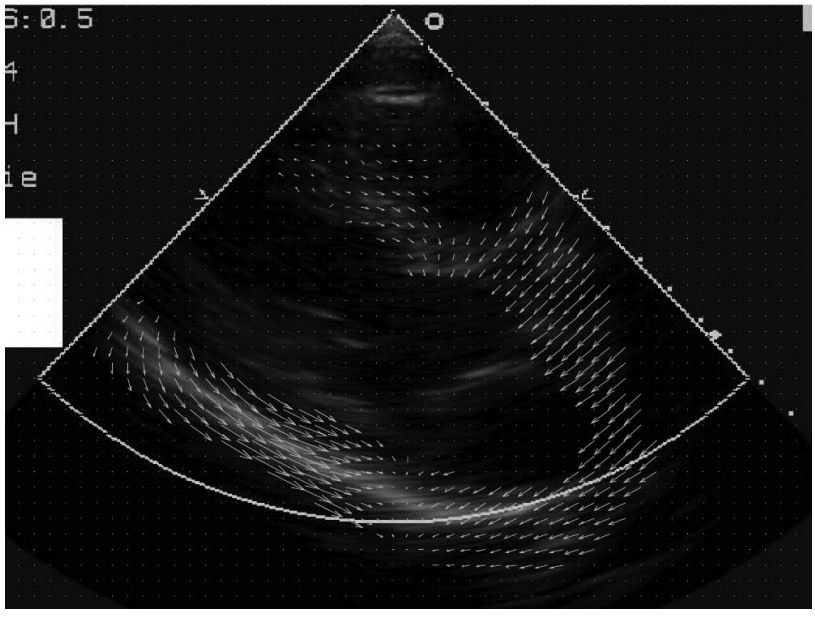
the two data sets and superimpose them onto a common reference system. To this end, we computed the required transformation by identifying a pair of landmarks (base of the mitral valves) in both the sequences. Figure 5.6 shows the reconstructed motion field for one frame in both views. The reconstructed velocity field was validated in a qualitative manner by comparison of the motion directions determined from grayscale images by several echocardiographic experts, who could confirm the agreement of expert readings with the results found by our algorithm.

5.4.6 Some Notes on the Performance

The above experimental examples demonstrate the effectiveness of the proposed method. We observe a striking difference between the performance of the vector regularizer and the more traditional thin-plate spline regularizer. This is due to the fact that the vector regularizer introduces a coupling between the x and y components whereas the thin-plate spline regularizer— which works well when the sample are complete (both components)— treats the components independently. The incompleteness of the Doppler data



(a) Apical long axis view.



(b) Parasternal long axis view.

Figure 5.6: Reconstructed motion field from two view cardiac Doppler data.

makes the coupling crucial, and hence the improvement of coupled regularizer over the uncoupled one is significant. Another point that makes the proposed vector regularizer attractive is that it allows one to incorporate some a priori knowledge on the motion. Note that it can be specified to penalize the pure deformation only; in other words, the null space of the regularization operator can be chosen to include all the rigid forms of motions such as translation and rotation.

The computational task essentially consists of two parts: (i) setting up the system of equations, and (ii) solving the system. The size of the data set influences the first part only, which is typically negligible when compared to the second. The later depends only on the required resolution a or, equivalently, the number of grid points $N_g \times N_g$. Thanks to the right choice of basis functions, this complexity can be brought down to $\mathcal{O}(N_g \times N_g)$, if one uses efficient techniques that are specialized for sparse systems [62]. To sum up, the overall complexity is essentially proportional to $\frac{1}{a^2}$. Thus we can control the complexity by adjusting the grid size, an option that is especially useful when dealing with large data set that originate from multiple acquisitions. Of course, having more measurements is advantageous because it will contribute to improving the reconstruction SNR.

Even though our results are still preliminary and require some careful imaging set-up, they open up some interesting future perspectives. The key point is that modern systems have the capability of acquiring Doppler data in multiple views without moving the probe, for example, by steering the beam in the parallel beam system (figure 1.1(b)). However, in such systems, the users can only visualize the axial velocity in a color coded form, and the raw data is not accessible normally. The proposed algorithm, if integrated with such systems, opens up a possibility of visualizing the true velocity field without any manual registration procedure.

All the reconstructions were done with the grid size in the range $32 - 50$. The corresponding size of the linear system is in the range $2048 - 5000$. Thanks to the right choice of the basis functions, the matrices are sparse and hence it takes only $3-9$ seconds to solve by direct methods using matlab on a Apple G5 system. We also implemented a multi-grid version of the algorithm for the case $\lambda_{d_0} = \lambda_{c_0}$, and $\lambda_{d_1} = \lambda_{c_1}$. We verified that the multigrid solver will speed up by at least one order of magnitude. The same strategy should

also work for the more general case but will require a significant programming effort as well as a re-engineering of the structure of the current program, which is an adaptation of the scalar algorithm developed in Chapter 3.

Conclusion

We developed a numerically efficient method to reconstruct vector field from projected samples that is a preferable alternative for vector spline method when the data size is large. Using this method, we demonstrated the feasibility of recovering true velocity field from pulsed wave Doppler data. Using the reconstruction examples, we also demonstrated the importance of tuning the smoothness functional, and showed how the data indeterminacy can be overcome by a suitable choice of vector regularization.

Appendix

Regularization Filters

Our goal is to find expressions for $R(\mathbf{v})$ and $R_E(\mathbf{v})$ in terms of the B-spline expansion coefficients. Each constitutive term is a bilinear functional involving the velocity components. They are of the following form:

$$B(f, g, p_1, q_1, p_2, q_2) = \int \left(\partial_x^{p_1} \partial_y^{q_1} f(x, y)\right) \left(\partial_x^{p_2} \partial_y^{q_2} g(x, y)\right) dxdy. \tag{5.54}$$

where f and g are expressed as

$$f(x, y) = \sum_{k,l} c_{k,l}^f \beta^n(x/a - k)\beta^n(y/a - l) \tag{5.55}$$

$$g(x, y) = \sum_{m,n} c_{k,l}^g \beta^n(x/a - m)\beta^n(y/a - n) \tag{5.56}$$

Here, f and g are the symbolic replacements for the velocity components, u and v; they either refer to the same component (square terms) or different components (cross terms). This representation facilitates deriving a generic computational formula that will be useful for both square terms and cross terms. To derive the formula, we first define

$$\alpha_{p,a}(x) = \frac{d^p}{dx^p} \left\{\beta^n(x/a)\right\}. \tag{5.57}$$

Substituting (5.55) and (5.56) in (5.54), we get

$$\begin{aligned} B(f,g,p_1,q_1,p_2,q_2) &= \sum_{l,n}\sum_{k,m} c^f_{k,l} c^g_{m,n} \\ &\quad \left[\int \alpha_{p_1,a}(x-ka)\alpha_{p_2,a}(x-ma)dx\right] \\ &\quad \left[\int \alpha_{q_1,a}(y-la)\alpha_{q_2,a}(y-na)dy\right]. \end{aligned}$$

This in turn yields

$$B(f,g,p_1,q_1,p_2,q_2) = \sum_{l,n}\sum_{k,m} c^f_{k,l} c^g_{m,n} \eta_{p_1,p_2}((m-k)a)\eta_{q_1,q_2}((n-l)a), \quad (5.58)$$

where $\eta_{p_1,p_2}(x)$ is defined by the following convolution:

$$\eta_{p_1,p_2}(x) = \alpha_{p_1,a}(x) * \alpha_{p_2,a}(-x). \quad (5.59)$$

Now define

$$\gamma_{p_1,p_2}(k) = \eta_{p_1,p_2}(ka), \quad (5.60)$$

$$\mu_{p_1,q_1,p_2,q_2}(k,l) = \gamma_{p_1,p_2}(k)\gamma_{q_1,q_2}(l). \quad (5.61)$$

Substituting we get,

$$\begin{aligned} B(f,g,p_1,q_1,p_2,q_2) &= \sum_{l,n}\sum_{k,m} c^g_{k,l} c^f_{m,n} \mu_{p_1,q_1,p_2,q_2}(m-k,n-l), \\ &= \sum_{k,l} c^g_{k,l} \sum_{m,n} c^f_{m,n} \mu_{p_1,q_1,p_2,q_2}(m-k,n-l), \\ &= \left\langle \mu_{p_1,q_1,p_2,q_2}(k,l) * c^f_{k,l},\ \ c^g_{k,l} \right\rangle_{l_2(\mathbb{Z}^2)} \quad (5.62) \end{aligned}$$

In other words, the bilinear form $B(f,g,p_1,q_1,p_2,q_2)$ that involves an integral over $\mathbb{R}^2$ is re-expressed as a bilinear form that involves a discrete inner product over l_2. Hence (5.62) gives a basic building block to represent any vector regularization functional for B-splines by discrete convolutions. The discrete filter $\mu_{p_1,q_1,p_2,q_2}(k,l)$ is completely determined by the quadruple of derivative orders $\{p_1,q_1,p_2,q_2\}$, the degree of the B-spline, and the scale a (equations (5.57), (5.59), (5.60), (5.61)). Let $\Gamma_{p_1,p_2}(z)$ and $M_{p_1,q_1,p_2,q_2}(z_1,z_2)$ be z-transforms of $\gamma_{p_1,p_2}(k)$ and $\mu_{p_1,q_1,p_2,q_2}(k,l)$ respectively. Then

$$M_{p_1,q_1,p_2,q_2}(z_1,z_2) = \Gamma_{p_1,p_2}(z_1)\Gamma_{q_1,q_2}(z_1) \quad (5.63)$$

The next step now is to find explicit expression for $\Gamma_{p_1,p_2}(z)$. To this end, we first write the Fourier expressions for B-spline [55]:

$$\beta^n(x) \quad \longleftrightarrow \quad \hat{\beta}^n(\omega) = \frac{(e^{j\omega/2} - e^{-j\omega/2})^{n+1}}{(j\omega)^{n+1}}$$

$$\beta_a^n(x) = \beta^n(x/a) \quad \longleftrightarrow \quad \hat{\beta}_a^n(\omega) = a\hat{\beta}^n(a\omega) = a\frac{(e^{ja\omega/2} - e^{-ja\omega/2})^{n+1}}{(ja\omega)^{n+1}}$$

Then Fourier expression for $\alpha_{p,a}(x)$ defined in the equation (5.57) is given by

$$\begin{aligned}
\hat{\alpha}_{p,a}(\omega) &= (j\omega)^p\, \hat{\beta}_a^n(\omega) \\
&= (j\omega)^p\, a\hat{\beta}^n(a\omega) \\
&= (j\omega)^p\, a\frac{(e^{ja\omega/2} - e^{-ja\omega/2})^{n+1}}{(ja\omega)^{n+1}} \\
&= \frac{1}{a^p}(e^{ja\omega/2} - e^{-ja\omega/2})^p \left[a\frac{(e^{ja\omega/2} - e^{-ja\omega/2})^{n+1-p}}{(ja\omega)^{n+1-p}}\right] \quad (5.64)
\end{aligned}$$

Now, $\eta_{p_1,p_2}(x)$ defined in the equation (5.59) is given by

$$\hat{\eta}_{p_1,p_2}(\omega) \quad = \quad \hat{\alpha}_{p_1,a}(\omega)\tilde{\hat{\alpha}}_{p_2,a}(\omega)$$

Substituting (5.64) yields

$$\begin{aligned}
\hat{\eta}_{p_1,p_2}(\omega) &= \frac{1}{a^{p_1+p_2}}(e^{ja\omega/2} - e^{-ja\omega/2})^{p_1}(e^{-ja\omega/2} - e^{ja\omega/2})^{p_2} \\
&\quad \times a^2\left[\frac{(e^{ja\omega/2} - e^{-ja\omega/2})^{n+1-p_1}}{(ja\omega)^{n+1-p_1}}\right]\left[\frac{(e^{-ja\omega/2} - e^{ja\omega/2})^{n+1-p_2}}{(-ja\omega)^{n+1-p_2}}\right]
\end{aligned}$$

Now let us consider the case when $p_1 + p_2$ is even. We rewrite the above equation as

$$\begin{aligned}
\hat{\eta}_{p_1,p_2}(\omega) &= \frac{1}{a^{p_1+p_2-1}}(e^{ja\omega/2} - e^{-ja\omega/2})^{p_1}(e^{-ja\omega/2} - e^{ja\omega/2})^{p_2} \\
&\quad \times a\left[\frac{(e^{ja\omega/2} - e^{-ja\omega/2})^{2n+2-p_1-p_2}}{(ja\omega)^{2n+2-p_1-p_2}}\right] \\
&= \frac{1}{a^{p_1+p_2-1}}(e^{ja\omega/2} - e^{-ja\omega/2})^{p_1}(e^{-ja\omega/2} - e^{ja\omega/2})^{p_2} a\hat{\beta}^{2n+1-p_1-p_2}(a\omega) \\
&= \frac{1}{a^{p_1+p_2-1}}(e^{ja\omega/2} - e^{-ja\omega/2})^{p_1}(e^{-ja\omega/2} - e^{ja\omega/2})^{p_2} \hat{\beta}_a^{2n+1-p_1-p_2}(\omega) \\
&= \frac{1}{a^{p_1+p_2-1}}\hat{\zeta}^{(p_1,p_2)}(e^{ja\omega})\hat{\beta}_a^{2n+1-p_1-p_2}(\omega) \quad (5.65)
\end{aligned}$$

where

$$\hat{\zeta}^{(p_1,p_2)}(e^{ja\omega}) = (e^{ja\omega/2} - e^{-ja\omega/2})^{p_1}(e^{-ja\omega/2} - e^{ja\omega/2})^{p_2}.$$

Since p_1+p_2 is even $\hat{\zeta}^{(p_1,p_2)}(e^{ja\omega})$ is a polynomial in $e^{-ja\omega}$ (or even polynomial in $e^{-ja\omega/2}$). Let

$$\hat{\zeta}^{(p_1,p_2)}(e^{ja\omega}) = \sum_l \zeta_l^{(p_1,p_2)} e^{-jla\omega}. \tag{5.66}$$

Substituting in (5.65) and applying an inverse Fourier transformation, we get

$$\begin{aligned}
\eta_{p_1,p_2}(x) &= \frac{1}{a^{p_1+p_2-1}} \sum_l \zeta_l^{(p_1,p_2)} \beta_a^{2n+1-p_1-p_2}(x-la) \\
&= \frac{1}{a^{p_1+p_2-1}} \sum_l \zeta_l^{(p_1,p_2)} \beta^{2n+1-p_1-p_2}(x/a-l)
\end{aligned}$$

This yields

$$\begin{aligned}
\gamma_{p_1,p_2}(k) &= \eta_{p_1,p_2}(ka) \\
&= \frac{1}{a^{p_1+p_2-1}} \sum_l \zeta_l^{(p_1,p_2)} \beta^{2n+1-p_1-p_2}(k-l) \\
&= \frac{1}{a^{p_1+p_2-1}} \zeta_k^{(p_1,p_2)} * \beta^{2n+1-p_1-p_2}(k)
\end{aligned}$$

Hence in z-domain we get,

$$\Gamma_{p_1,p_2}(z) = \hat{\zeta}^{(p_1,p_2)}(z) B^{2n+1-p_1-p_2}(z)$$

Now, let $p_1 + p_2$ be odd. We rewrite (5.65) as follows:

$$\hat{\eta}_{p_1,p_2}(\omega) = \frac{1}{a^{p_1+p_2-1}} \hat{\zeta}^{(p_1-1,p_2)}(e^{ja\omega})(e^{ja\omega/2} - e^{-ja\omega/2}) \hat{\beta}_a^{2n+1-p_1-p_2}(\omega) \tag{5.67}$$

Note that $\hat{\zeta}^{(p_1-1,p_2)}(e^{ja\omega})$ is now a polynomial in $e^{-ja\omega}$, and let

$$\hat{\zeta}^{(p_1-1,p_2)}(e^{ja\omega}) = \sum_l \zeta_l^{(p_1-1,p_2)} e^{-jla\omega}.$$

Substituting the above equation in (5.67) and applying inverse Fourier transformation, we get

$$\begin{aligned}
\eta_{p_1,p_2}(x) &= \frac{1}{a^{p_1+p_2-1}} \sum_l \zeta_l^{(p_1-1,p_2)} \\
&\quad \times \left(\beta_a^{2n+1-p_1-p_2}(x-la+a/2) - \beta_a^{2n+1-p_1-p_2}(x-la-a/2)\right) \\
&= \frac{1}{a^{p_1+p_2-1}} \sum_l \zeta_l^{(p_1-1,p_2)} \\
&\quad \times \left(\beta^{2n+1-p_1-p_2}(x/a-l+1/2) - \beta^{2n+1-p_1-p_2}(x-l-1/2)\right)
\end{aligned}$$

This yields

$$\begin{aligned}\gamma_{p_1,p_2}(k) &= \eta_{p_1,p_2}(ka) \\ &= \frac{1}{a^{p_1+p_2-1}}\sum_l \zeta_l^{(p_1-1,p_2)} \\ &\quad \times \left(\beta^{2n+1-p_1-p_2}(k-l+1/2) - \beta^{2n+1-p_1-p_2}(k-l-1/2)\right) \\ &= \frac{1}{a^{p_1+p_2-1}}\zeta_k^{(p_1-1,p_2)} * d^{2n+1-p_1-p_2}(k),\end{aligned}$$

where $d^n(x) = \beta^n(x+1/2) - \beta^n(x-1/2)$. In z-domain it reads

$$\Gamma_{p_1,p_2}(z) = \hat{\zeta}^{(p_1-1,p_2)}(z) D^{2n+1-p_1-p_2}(z),$$

where

$$D^n(z) = \sum_k d^n(k) z^{-k}.$$

In summary, the filter $\Gamma_{p_1,p_2}(z)$ is given by

$$\Gamma_{p_1,p_2}(z) = \begin{cases} \left(\frac{1}{a}\right)^{p_1+p_2-1} \hat{\zeta}^{(p_1,p_2)}(z) B^{2n+1-p_1-p_2}(z), \text{ if } p_1+p_2 \text{ is even, and} \\ \left(\frac{1}{a}\right)^{p_1+p_2-1} \hat{\zeta}^{(p_1-1,p_2)}(z) D^{2n+1-p_1-p_2}(z), \text{otherwise,} \end{cases} \tag{5.68}$$

where

$$\begin{aligned}\hat{\zeta}^{(p_1,p_2)}(z) &= \left(z^{1/2} - z^{-1/2}\right)^{p_1} \left(z^{-1/2} - z^{1/2}\right)^{p_2}, \\ B^n(z) &= \sum_k \beta^n(k) z^{-k}, \\ D^n(z) &= \sum_k (\beta^n(k+1/2) - \beta^n(k-1/2)) z^{-k}.\end{aligned}$$

The filter, $\Gamma_{p_1,p_2}(z)$, have some symmetry properties. In (5.68), all the terms in the right hand side are symmetric except $\beta'^{2n+1-p_1-p_2}(z)$, which is anti-symmetric. Hence we get

$$\Gamma_{p_1,p_2}(z^{-1}) = (-1)^{p_1+p_2}\Gamma_{p_1,p_2}(z) \tag{5.69}$$

On the other hand, (5.59) and (5.60) imply

$$\Gamma_{p_2,p_1}(z) = \Gamma_{p_1,p_2}(z^{-1}) \tag{5.70}$$

This these two relations together yield

$$\Gamma_{p_2,p_1}(z) = (-1)^{p_1+p_2}\Gamma_{p_1,p_2}(z) \tag{5.71}$$

The next step is to use (5.68) to get explicit expressions for each of the terms in $R(\mathbf{v})$ and $R_E(\mathbf{v})$. We first consider $\mathcal{D}_1(u)$. Using (5.54), (5.13) can be written as

$$\mathcal{D}_1(u) = B(u,u,1,0,1,0) + B(u,u,0,1,0,1).$$

Substituting (5.62) in the above equation yields

$$\begin{aligned} \mathcal{D}_1(u) &= \left\langle \mu_{1,0,1,0}(k,l) * c^u_{k,l},\ \ c^u_{k,l} \right\rangle + \left\langle \mu_{0,1,0,1}(k,l) * c^u_{k,l},\ \ c^u_{k,l} \right\rangle \\ &= \left\langle (\mu_{1,0,1,0}(k,l) + \mu_{0,1,0,1}(k,l)) * c^u_{k,l},\ \ c^u_{k,l} \right\rangle \end{aligned} \tag{5.72}$$

The constituent filters in z-domain read

$$M_{1,0,1,0}(z_1,z_2) = \Gamma_{1,1}(z_1)\Gamma_{0,0}(z_2)$$

$$M_{0,1,0,1}(z_1,z_2) = \Gamma_{0,0}(z_1)\Gamma_{1,1}(z_2)$$

Using (5.68), the filters $\Gamma_{1,1}(z)$ and $\Gamma_{0,0}(z)$ are expressed as

$$\Gamma_{0,0}(z) = B^{2n+1}(z)$$

$$\Gamma_{1,1}(z) = (z+2+z^{-1})B^{2n-1}(z)$$

Substituting these expressions in (5.72) yields (5.15). In a similar way, we can establish (5.16).

Next, we intend prove (5.26). We first expand $\int (Div\,\mathbf{v}(\mathbf{x}))^2\, d\mathbf{x}$ using (5.23) and then substitute (5.54) for each resulting term. We get

$$\int (Div\,\mathbf{v}(\mathbf{x}))^2\, d\mathbf{x} = B(u,u,1,0,1,0) + B(v,v,0,1,0,1) + 2B(u,v,1,0,0,1)$$

Using (5.62), this yields

$$\begin{aligned} \int (Div\,\mathbf{v}(\mathbf{x}))^2\, d\mathbf{x} &= \left\langle r_{11}(k,l) * c^u_{k,l},\ \ c^u_{k,l} \right\rangle + \left\langle r_{12}(k,l) * c^v_{k,l},\ \ c^v_{k,l} \right\rangle \\ &\quad + \left\langle r_{13}(k,l) * c^u_{k,l},\ \ c^v_{k,l} \right\rangle \end{aligned}$$

where

$$r_{11}(k,l) = \mu_{1,0,1,0}(k,l)$$

$$r_{12}(k,l) = \mu_{0,1,0,1}(k,l)$$

$$r_{13}(k,l) = \mu_{1,0,0,1}(k,l)$$

Using (5.63) and (5.68) results

$$\begin{aligned}
R_{11}(z_1,z_2) &= \Gamma_{11}(z_1)\Gamma_{00}(z_2) \\
&= (z_1+2+z_1^{-1})B^{2n-1}(z_1)B^{2n+1}(z_2) \\
R_{12}(z_1,z_2) &= \Gamma_{00}(z_1)\Gamma_{11}(z_2) \\
&= B^{2n+1}(z_1)(z_2+2+z_2^{-1})B^{2n-1}(z_2) \\
R_{13}(z_1,z_2) &= \Gamma_{10}(z_1)\Gamma_{01}(z_2) \\
&= \Gamma_{10}(z_1)\Gamma_{10}(z_2^{-1}) \quad \text{(using (5.70))} \\
&= B'^{2n+1}(z_1)B'^{2n+1}(z_2^{-1})
\end{aligned}$$

which proves (5.26). We obtain (5.27) in a similar way.

Next, to establish (5.28), we first expand it as follow:

$$\begin{aligned}
\int \|\nabla\left(Div\,\mathbf{v}(\mathbf{x})\right)\|^2 d\mathbf{x} &= B(u,u,2,0,2,0)+B(v,v,0,2,0,2)+B(u,u,1,1,1,1) \\
&+ B(v,v,1,1,1,1)+B(u,v,2,0,1,1)+B(u,v,1,1,0,2)
\end{aligned}$$

Using (5.62) yields

$$\begin{aligned}
\int \|\nabla\left(Div\,\mathbf{v}(\mathbf{x})\right)\|^2 d\mathbf{x} &= \left\langle r_{21}(k,l)*c^u_{k,l},\ \ c^u_{k,l}\right\rangle + \left\langle r_{22}(k,l)*c^v_{k,l},\ \ c^v_{k,l}\right\rangle \\
&\quad + \left\langle r_{23}(k,l)*c^u_{k,l},\ \ c^v_{k,l}\right\rangle
\end{aligned}$$

where

$$\begin{aligned}
R_{21}(z_1,z_2) &= M_{2,0,2,0}(z_1,z_2)+M_{1,1,1,1}(z_1,z_2) \\
&= \Gamma_{22}(z_1)\Gamma_{00}(z_2)+\Gamma_{11}(z_1)\Gamma_{11}(z_2) \qquad (5.73) \\
R_{22}(z_1,z_2) &= M_{0,2,0,2}(z_1,z_2)+M_{1,1,1,1}(z_1,z_2) \\
&= \Gamma_{00}(z_1)\Gamma_{22}(z_2)+\Gamma_{11}(z_1)\Gamma_{11}(z_2) \qquad (5.74) \\
R_{23}(z_1,z_2) &= M_{2,0,1,1}(z_1,z_2)+M_{1,1,0,2}(z_1,z_2) \\
&= \Gamma_{21}(z_1)\Gamma_{01}(z_2)+\Gamma_{10}(z_1)\Gamma_{12}(z_2)
\end{aligned}$$

Now, using (5.71), we get

$$\Gamma_{21}(z_1)\Gamma_{01}(z_2) = (-\Gamma_{12}(z_1))(-\Gamma_{10}(z_2)) = \Gamma_{12}(z_1)\Gamma_{10}(z_2).$$

Hence, $R_{23}(z_1,z_2)$ becomes

$$R_{23}(z_1,z_2) = \Gamma_{12}(z_1)\Gamma_{10}(z_2)+\Gamma_{10}(z_1)\Gamma_{12}(z_2) \qquad (5.75)$$

Applying the formula (5.68) in each of the terms in equations (5.73), (5.74), and (5.75), we finally get

$$\begin{aligned}
R_{21}(z_1,z_2) &= \frac{1}{a^2}\left[(z_1+2+z_1^{-1})^2 B^{2n-3}(z_1)B^{2n+1}(z_2)\right.\\
&\quad \left.+(z_1+2+z_1^{-1})B^{2n-1}(z_1)(z_2+2+z_2^{-1})B^{2n-1}(z_2)\right]\\
R_{22}(z_1,z_2) &= \frac{1}{a^2}\left[B^{2n+1}(z_1)(z_2+2+z_2^{-1})^2 B^{2n-3}(z_2)\right.\\
&\quad \left.+(z_1+2+z_1^{-1})B^{2n-1}(z_1)(z_2+2+z_2^{-1})B^{2n-1}(z_2)\right]\\
R_{23}(z_1,z_2) &= \frac{1}{a^2}\left[(z_1+2+z_1^{-1})D^{2n-2}(z_1)D^{2n+1}(z_2^{-1})\right.\\
&\quad \left.+D^{2n+1}(z_1)(z_2+2+z_2^{-1})D^{2n-2}(z_2^{-1})\right]
\end{aligned}$$

This proves (5.28). (5.29) is established in a similar way.

Interscale Relation

Let $\mathbf{v}_{(j)}(\mathbf{x}) = [u_{(j)}(\mathbf{x}) \;\; v_{(j)}(\mathbf{x})]^T$ and $\mathbf{v}_{(j+1)}(\mathbf{x}) = [u_{(j+1)}(\mathbf{x}) \;\; v_{(j+1)}(\mathbf{x})]^T$ be the velocity fields at scales j and $j+1$ that are expressed as

$$\begin{aligned}
u_{(j)}(x,y) &= \sum_{k}^{(N_g-1)/2^j}\sum_{l}^{(N_g-1)/2^j} c_{k,l}^{u,(j)}\beta(x/2^j-k)\beta(y/2^j-l),\\
v_{(j)}(x,y) &= \sum_{k}^{(N_g-1)/2^j}\sum_{l}^{(N_g-1)/2^j} c_{k,l}^{v,(j)}\beta(x/2^j-k)\beta(y/2^j-l),\\
u_{(j+1)}(x,y) &= \sum_{k}^{(N_g-1)/2^{j+1}}\sum_{l}^{(N_g-1)/2^{j+1}} c_{k,l}^{u,(j+1)}\beta(x/2^{j+1}-k)\beta(y/2^{j+1}-l),\\
v_{(j+1)}(x,y) &= \sum_{k}^{(N_g-1)/2^{j+1}}\sum_{l}^{(N_g-1)/2^{j+1}} c_{k,l}^{v,(j+1)}\beta(x/2^{j+1}-k)\beta(y/2^{j+1}-l).
\end{aligned}$$

Let

$$\begin{aligned}
\mathbf{c}_{(j)} &= [\cdots \;\; c_{k,l}^{u,(j)} \;\; \cdots \;\; c_{k,l}^{v,(j)} \;\; \cdots]^T, \text{ and}\\
\mathbf{c}_{(j+1)} &= [\cdots \;\; c_{k,l}^{u,(j+1)} \;\; \cdots \;\; c_{k,l}^{v,(j+1)} \;\; \cdots]^T.
\end{aligned}$$

The cost functionals for the scales j and $j+1$ are given by (equation (5.47))

$$J_{\text{spline}}\left(\mathbf{c}_{(j)}\right) = \mathbf{c}_{(j)}^T\mathbf{A}_{(j)}\mathbf{c}_{(j)} - 2\mathbf{c}_{(j)}^T\mathbf{b}_{(j)} + \mathbf{m}^T\mathbf{m}, \text{ and} \tag{5.76}$$

$$J_{\text{spline}}\left(\mathbf{c}_{(j+1)}\right) = \mathbf{c}_{(j+1)}^T\mathbf{A}_{(j+1)}\mathbf{c}_{(j+1)} - 2\mathbf{c}_{(j+1)}^T\mathbf{b}_{(j+1)} + \mathbf{m}^T\mathbf{m} \tag{5.77}$$

Let $\{\tilde{c}_{k,l}^{u,(j)}\}$ and $\{\tilde{c}_{k,l}^{v,(j)}\}$ be the sequences obtained by up-sampling and filtering the sequences $\{c_{k,l}^{u,(j+1)}\}$ and $\{c_{k,l}^{v,(j+1)}\}$ respectively by the B-spline

two-scale filter; i.e., $\tilde{\mathbf{c}}_{(j)} = \mathbf{U}_{(j)}\mathbf{c}_{(j+1)}$ where

$$\tilde{\mathbf{c}}_{(j)} = [\cdots \;\; \tilde{c}^{u,(j)}_{k,l} \;\; \cdots \;\; \tilde{c}^{v,(j)}_{k,l} \;\; \cdots]^T, \text{ and}$$

$\mathbf{U}_{(j)}$ is as defined before. Then, the velocity field $\tilde{\mathbf{v}}_{(j)}(\mathbf{x}) = [\tilde{u}_{(j)}(\mathbf{x}) \;\; \tilde{v}_{(j)}(\mathbf{x})]^T$ should be equal to $\mathbf{v}_{(j+1)}(\mathbf{x})$, where

$$\tilde{u}_{(j)}(x,y) = \sum_{k}^{(N_g-1)/2^j} \sum_{l}^{(N_g-1)/2^j} \tilde{c}^{u,(j)}_{k,l}\beta(x/2^j-k)\beta(y/2^j-l)$$

$$\tilde{v}_{(j)}(x,y) = \sum_{k}^{(N_g-1)/2^j} \sum_{l}^{(N_g-1)/2^j} \tilde{c}^{v,(j)}_{k,l}\beta(x/2^j-k)\beta(y/2^j-l)$$

Its cost functional is given by

$$J_{\text{spline}}\left(\tilde{\mathbf{c}}_{(j)}\right) = \tilde{\mathbf{c}}^T_{(j)}\mathbf{A}_{(j)}\tilde{\mathbf{c}}_{(j)} - 2\tilde{\mathbf{c}}^T_{(j)}\mathbf{b}_{(j)} + \mathbf{m}^T\mathbf{m} \tag{5.78}$$

Substituting $\tilde{\mathbf{c}}_{(j)} = \mathbf{U}_{(j)}\mathbf{c}_{(j+1)}$ we get

$$J_{\text{spline}}\left(\tilde{\mathbf{c}}_{(j)}\right) = \mathbf{c}^T_{(j+1)}\mathbf{U}^T_{(j)}\mathbf{A}_{(j)}\mathbf{U}_{(j)}\mathbf{c}_{(j+1)} - 2\mathbf{c}^T_{(j+1)}\mathbf{U}^T_{(j)}\mathbf{b}_{(j)} + \mathbf{m}^T\mathbf{m} \tag{5.79}$$

Since $\tilde{\mathbf{v}}_{(j)}(\mathbf{x}) = \mathbf{v}_{(j+1)}(\mathbf{x})$, $J_{\text{spline}}\left(\tilde{\mathbf{c}}_{(j)}\right) = J_{\text{spline}}\left(\mathbf{c}_{(j+1)}\right)$. Hence the minima of the equations (5.77) and (5.79) should be the same. In other words, the solutions of the following equations should be equal:

$$\begin{aligned} \mathbf{A}_{(j+1)}\mathbf{c}_{(j+1)} &= \mathbf{b}_{(j+1)} \\ \mathbf{U}^T_{(j)}\mathbf{A}_{(j)}\mathbf{U}_{(j)}\mathbf{c}_{(j+1)} &= \mathbf{U}^T_{(j)}\mathbf{b}_{(j)} \end{aligned}$$

This in turn implies that

$$\mathbf{A}_{(j+1)} = \mathbf{U}^T_{(j)}\mathbf{A}_{(j)}\mathbf{U}_{(j)}$$

$$\mathbf{b}_{(j+1)} = \mathbf{U}^T_{(j)}\mathbf{b}_{(j)}$$

We must measure what is measurable and make measurable what cannot be measured.

— Galileo Galilei

Chapter 6

Some Future Extensions

There are few possible extensions that broaden the scope of the proposed framework; these are the focus of this chapter.

6.1 Generalizing Invariances

There are generalizations of the imposed invariance principles that could possibly yield a larger family of regularization functionals. We present them in the vector field context only, since, whenever applicable, our arguments can be carried over to the scalar problem in a straightforward manner.

6.1.1 Rotational Invariance

First we consider a generalization of the rotational invariance for vector functions. Let $\mathbf{f}$ and $\mathbf{f}_{\mathbf{\Omega}}$ be original and rotated functions, respectively. Recall that they are related by

$$\mathbf{f}_{\mathbf{\Omega}}(\mathbf{x}) = \mathbf{\Omega}\mathbf{f}(\mathbf{\Omega}^T\mathbf{x}).$$

The original rotational invariance for the regularization functional is

$$J(\mathbf{f}) = J(\mathbf{f}_{\mathbf{\Omega}}).$$

The basic idea behind this equality is that, since both functions represent the same physical vector field, their energy, whatsoever, should be the same, and

hence J should measure them equally. This also ensures an invariant reconstruction for both interpolation and approximation problems. However, this is not a necessary condition for invariant reconstruction. It can be replaced by the following more general condition:

$$J(\mathbf{f}) = c(\mathbf{\Omega})J(\mathbf{f}_{\mathbf{\Omega}}),$$

where c is a non-negative functional. This new invariance ensures the invariance reconstruction, provided that, in the case of approximation, the regularization factor is adjusted appropriately. The correction factor can be obtained from the function c. One needs to verify whether this generalization leads to a more general family of regularization functionals or not, if indeed it does, this will provide a more general family of vector splines.

6.1.2 Affine Invariance

In our original definition of the scale-invariance, a scaled version of the function $\mathbf{f}(\mathbf{x})$ was $\mathbf{f}(a\mathbf{x})$, which is also denoted by $\mathbf{f}_a$. The invariance was defined by

$$J(\mathbf{f}) = c(a)J(\mathbf{f}_a).$$

Now, we can extend the notion of scaling of a function to the multiplication of its argument by a symmetric matrix (affine transformation). In other words, we can define a transformed version of a functions as follows:

$$\mathbf{f}_{\mathbf{A}}(\mathbf{x}) = \mathbf{f}(\mathbf{A}\mathbf{x}).$$

Let $\mathbf{A} = \mathbf{\Theta}\mathbf{D}\mathbf{\Theta}^T$ be the eigen decomposition of $\mathbf{A}$. $\mathbf{f}_{\mathbf{A}}$ is obtained by scaling $\mathbf{f}$ along a set of n orthogonal directions with different scale factors, where the former is given by the columns of $\mathbf{\Theta}$, and the later is given by the diagonal elements of $\mathbf{D}$; i.e., the eigen values of $\mathbf{A}$. Now the generalized scale invariance (or affine invariance) can be defined as

$$J(\mathbf{f}) = c(\mathbf{A})J(\mathbf{f}_{\mathbf{A}}).$$

This generalization should result in a larger family of regularization functionals.

6.2 Non-Local Measurement Models

In our work, we considered only point measurements. However, the theory is equally applicable to any linear measurement model. In this section, we present some possible extensions to the models that involve integrals. We use the approximation formulation to explain the extended variational problems; these arguments are obviously applicable for variational interpolation as well.

6.2.1 Scalar Problem

A general scalar problem can be expressed as the following minimization:

$$f^{(\mathrm{opt})} = \operatorname*{argmin}_f \sum_{i=1}^{N} \left(\langle r_i, f\rangle - m_i\right)^2 + \lambda J(f),$$

where $(r_i)_{i\in[1:N]}$ are the general measurement operators representing weighted integrals (or scalar products) and $(m_i)_{i\in[1:N]}$ are the measurements.

Tomographic imaging is a well-known example that involves integral measures. The system measures the so-called Radon transform of an image, which is defined as

$$\check{f}(t,\boldsymbol{\tau}) \quad = \quad \int f(\mathbf{x})\delta(\boldsymbol{\tau}^T\mathbf{x} - t)d\mathbf{x}, \tag{6.1}$$

where $\boldsymbol{\tau}$ is an unit vector. For a given $(t,\boldsymbol{\tau})$, $\check{f}(t,\boldsymbol{\tau})$ is equal to the integral of $f(\mathbf{x})$ over the hyperplane that is orthogonal to $\boldsymbol{\tau}$, and that is at position t from the origin. Note that $\boldsymbol{\tau}$ has $n-1$ parameters; and hence, if one includes the variable t, $\check{f}$ is a function in $\mathbb{R}^n$. This transform is invertible. The so-called filtered back projection method is used to invert this transform numerically [77]. Note that in 2D, the samples of the Radon transform correspond to line integrals, whereas in 3D, they correspond to planar integrals.

A typical tomographic data set contains a discrete set of sample of $\check{f}$, which can be represented as $(t_i,\boldsymbol{\tau}_i,m_i)_{i\in[1:N]}$, where $m_i = \check{f}(t_i,\boldsymbol{\tau}_i) + n_i = \left\langle \delta(\boldsymbol{\tau}_i^T\mathbf{x} - t_i), f(\mathbf{x})\right\rangle + n_i$ with n_i being some noise. The corresponding approximation problem reads

$$f^{(\mathrm{opt})} = \operatorname*{argmin}_f \sum_{i=1}^{N} \left(\left\langle \delta(\boldsymbol{\tau}_i^T\mathbf{x} - t_i), f(\mathbf{x})\right\rangle - m_i\right)^2 + \lambda J(f).$$

This problem has been solved in [78] for the 2D case, where the authors demonstrate that the variational reconstruction yields much better quality

when compared to the back projection method. This problem has not yet been solved for 3D.

6.2.2 Vector Problem

The vector field approximation problem with non-local measurements can be expressed as

$$\mathbf{f}^{(\mathrm{opt})} = \underset{\mathbf{f}}{\operatorname{argmin}} \sum_{i=1}^{N} \left(\langle \mathbf{r}_i, \mathbf{f} \rangle - m_i\right)^2 + \lambda J(\mathbf{f}),$$

where $(\mathbf{r}_i)_{i\in[1:N]}$ are now vector measurement operators.

An example that involves integral measures of vector functions is the extension of the conventional tomography for vector fields, which is known as the vector tomography [79, 80, 81], and which has been developed for flow field recovery. A vector tomographic system measures the so-called probe transform of the vector field, which is an extension of the conventional Radon transform. It is defined by

$$f^{\mathbf{P}}(t, \boldsymbol{\tau}) = \int \mathbf{p}^T(t, \boldsymbol{\tau}) \mathbf{f}(\mathbf{x}) \delta(\boldsymbol{\tau}^T \mathbf{x} - t) d\mathbf{x}, \tag{6.2}$$

where $\mathbf{p}$ is the so-called probe field. Note that, similar to the scalar case, $\check{f}(t, \boldsymbol{\tau})$ is equal to the integral of $f(\mathbf{x})$ over the hyperplane represented by $(t, \boldsymbol{\tau})$, but now the quantity that is integrated is the projection of the vector field along the direction specified by $\mathbf{p}(t, \boldsymbol{\tau})$.

Some numerical methods similar to back projection have been developed to recover the flow field from discrete samples of one or more probe transforms [79, 80, 81]. They recover the flow field either partly or fully depending on the transform used.

Similar to the scalar case, one can expect that variational reconstruction should yield much better results in this vector problem as well. The data set originating from the discrete samples of one or more probe transforms can be represented as $(t_i, \boldsymbol{\tau}_i, \mathbf{p}_i, m_i)_{i\in[1:N]}$, where

$$m_i = \int \mathbf{p}_i^T \mathbf{f}(\mathbf{x}) \delta(\boldsymbol{\tau}_i^T \mathbf{x} - t_i) d\mathbf{x} + n_i.$$

The approximation problem now reads

$$f^{(\mathrm{opt})} = \underset{f}{\operatorname{argmin}} \sum_{i=1}^{N} \left(\left\langle \mathbf{p}_i \delta(\boldsymbol{\tau}_i^T \mathbf{x} - t_i), \mathbf{f}(\mathbf{x}) \right\rangle - m_i\right)^2 + \lambda J(\mathbf{f}).$$

In the remainder of this section, we will have a closer look at some specific measurement schemes available for 2D and 3D flow fields.

2D Vector Fields

Recall that, in 2D, these measurements are line integrals. In most of the methods known so far, the quantity that is integrated is the component of the vector along the line of integration. In other words, the probe field is of the following form:

$$\mathbf{p}(t, \boldsymbol{\tau}) = \hat{\boldsymbol{\tau}}, \tag{6.3}$$

where $\hat{\boldsymbol{\tau}}$ is the unit vector that is orthogonal to $\boldsymbol{\tau}$. Such integrals can be measured in a variety of ways. Examples include measurement schemes that are based on acoustic time-of-flight [82, 83], optical phase shift [84], and the Doppler effect [85, 86]. The first two are transmission measurements, whereas the third is performed on the back scattered signal. It is primarily performed using an ultrasound beam. It is known as continuous wave Doppler measurement, which is an extension of pulsed wave Doppler scheme discussed in Chapter 1. Here, each scan line acquisition is performed by a pair of transducers, where one transmits a continuous wave while the other one receives the backscattered wave. With the assumption that the time taken for the ultrasound beam to travel the penetration depth is negligible with respect to the rate of change of motion, the total spectral spread in the received wave measures the required integral.

The 2D probe transform with the above specific choice of probe field given in (6.3) is also known as Doppler Radon transform. The main drawback of the Doppler Radon transform is that, its null space is made-up of all irrotational vector fields [85, 79]. Hence only the solenoidal component of a vector field can be recovered from the Doppler Radon (DR) data, even if one combines multiple acquisitions. However, in the context of recovering a full vector field, they can be still usefull in conjunction with pulsed wave Doppler data (PWD) for improving the reconstruction accuracy. This possibility of improvement originates from the fact that DR data may contain some information that is complementary to the PWD data in the following sense: the PWD data may be inconsistent with the underlying imaging model in the regions where the axial velocity is greater than certain maximum value [5], whereas the DR data do not suffer from any such limitations.

3D Vector Fields

The generalized vector measurement model (6.2) represent a planar integral in 3D. Interestingly, probe transforms for 3D can be measured with almost any arbitrary probe fields using MRI [81]. The author in this paper presents some recommended choices of probe fields, and gives formulas for recovering different component of flow field (solenoidal or irrotational) from specific probe transforms.

Note that the line integrals with probe field given in (6.3) can be obviously measured on 3D flow fields as well. However, since they do not have any equivalence with 3D Radon transforms, these type of measurements have not yet been considered for 3D flow field recovery. They may be used efficiently in the proposed variational setting. For example, they can be combined with PWD data as mentioned above for the 2D problem. To this end, one needs to write down the measurement equation for line integrals in 3D. Let $\hat{\boldsymbol{\tau}}$ be the unit vector representing the line direction, and let $\mathbf{s}$ be a point on the line. Then, the line integral with the probe field given by (6.3) can be written as

$$f_L(\hat{\boldsymbol{\tau}}, \mathbf{s}) = \int \hat{\boldsymbol{\tau}}^T \mathbf{f}(\mathbf{x}) \delta(\boldsymbol{\Gamma}^T(\mathbf{x} - \mathbf{s})) d\mathbf{x}, \tag{6.4}$$

where $\boldsymbol{\Gamma}$ is a matrix such that $[\hat{\boldsymbol{\tau}}\ \boldsymbol{\Gamma}]$ forms an orthogonal matrix.

Epilogue

The focus of this thesis has been on the design and application of variational methods for reconstructing scalar and vector images from non-uniform measurements. The motivation for adopting a variational strategy was to make the reconstruction problem well-posed when the sample locations are arbitrarily distributed.

The main aspects of the variational reconstruction problem are: (i) the measurement model; (ii) choice of the regularization functional; (iii) and the minimization space. The first is specified by the physical problem at hand, whereas the other two are under the control of the user; obviously, the latter two affect the performance of the method. In particular, the choice of regularization determines the quality of the reconstruction.

As far as the measurements are concerned, we considered the conventional local sampling model (dirac delta) in the scalar case, and the directional local model (dirac delta multiplied by a unit vector) in the vector case. A scalar problem with a more general measurement model that corresponds to tomographic projection has been addressed in [78]. We also proposed some future extensions of the measurement model for vector fields that correspond to some Doppler Radon transform or more general Probe transforms [79, 80, 81]. A variational reconstruction method that is tailored to these models may be a promising alternative to conventional filtered back projection algorithms that are currently used in tomography.

To address the second aspect, we adopted an axiomatic approach. We defined two generally desirable properties; namely, rotational-invariance and scale-invariance. Such invariant regularization functionals are desirable for they make the reconstruction invariant with respect to scaling and rotations of the input data, provided of course that the reconstruction space is in-

variant as well. While there exist a family of functionals satisfying these invariance properties for scalar functions [30], such a characterization for vector functions was not known until now. We extended these notions of invariance and characterized the corresponding complete family of functionals. Interestingly, this family specifically involves the divergence and rotational field components. It also includes all previously known functionals for vector splines. We demonstrated experimentally how the completeness of the family allows us to tune the functional according to the problem at hand, which results in a better reconstruction performance.

Concerning the final ingredient, which is the minimization space, one choses a specific space instead of considering "all functions" for which the cost is finite, even though the later is more logical. The reason is rather technical. Such an a priori constraint is imposed in order to make the minimization problem well-defined for the given regularization functional. In particular, the restriction facilitates proving the existence and uniqueness of the solution. An appropriate Sobolev space is typically used for the scalar problem [30], while Beppo-Levi product spaces are used for the vector problem [49, 50, 51]. Nevertheless, in all of these methods, these theoretical restrictions are very mild; in other words the minimization space can be supposed to be large enough to include any physical function. In particular, they are rotational- and scale invariant by construction. In our work on the vector problem, we defined the minimization space as the space of functions satisfying a specific property that relates divergence and rotational sub-functionals. We were less specific about the minimization space, since we did not attempt to prove the existence and uniqueness of the solution. In stead, the existence and uniqueness has to be verified a posteriori.

For computational purposes, we proposed to do the minimization within a given shift-invariant space (e.g. cubic splines) that can approximate any function as close as desired by appropriately choosing the step size. We provided the explicit construction of the linear system of equation that specifies the required solution for both scalar and vector problem. The notable advantages of this new approach over the previous one are the following: (i) it offers a simple way to trade complexity against accuracy and vice versa. (ii) it yields a system that is numerically well-conditioned; (iii) it allows multiresolution; (iv) resampling is inexpensive, which amounts to a simple digital filtering

[55]. We chose B-spline shift-invariant space since it gives the best trade-off between the complexity and the reconstruction quality [59]. This approach is preferable to the conventional approach when the number of measurements is large.

It should be noted that shift-invariant spaces are not rigorously rotational- and scale-invariant. This means that they cannot provide a solution that is invariant exactly; i.e., a scaled and rotated data set does not yield an exact scaled and rotated reconstructed output, even if we use an invariant regularization functional. However, if the later is used, the discrepancy with the theoretical solution can be made negligible by choosing a sufficiently small step size. This means that the use of an invariant functional is still justified and that it should have beneficial effect on the solution.

Bibliography

[1] Y. Ye and J. Zhu, "Geometric studies on variable radius sprial cone-beam scanning," *Medical Physics*, vol. 31, pp. 1473–1480, 2004.

[2] M. Sonka and J.M. Fitzpatrick, Eds., *Handbook of Medical Imaging*, vol. 2, chapter 11: Echocardiography, SPIE PRESS, 2000.

[3] Y. Zhang, R. Rohling, and D. Pai, "Direct surface extraction from 3D freehand ultrasound images," in *Proceedings of the IEEE conference on Visualization*, 2002, pp. 45–52.

[4] A. E. Weyman, *Principles and Practice of Echocardiography*, Lea & Febiger, 2nd edition, 1994.

[5] D. Evans and W. McDicken, *Doppler Ultrasound: Physics, Instrumentation and Signal Processing*, Wiley, 2000.

[6] S. Park, M. Park, and M. Kang, "Super-resolution image reconstruction: A technical overview," *IEEE Signal Processing Magazine*, pp. 21–36, 2003.

[7] A. Secker and B. Taubman, "Highly scalable video compression using a lifting based 2D wavelet transform with deformable mesh motion compensation," in *Proceedings of International Conference on Image Processing and Applications*, 2002, pp. 749–752.

[8] J. Konrad, "View reconstruction for 3D video entertainment: Issues, algorithms and applications," in *Proceedings of International Conference on Image Processing and Applications*, 1999, pp. 8–12.

[9] L. Brown, "A survey of image registration techniques," *ACM Computing Surveys*, vol. 24, pp. 326–376, 1992.

[10] H. Wendland, *Scattered Data Approximation*, Cambridge University Press, 2005.

[11] A. Zakhor and G. Alvstad, "Two-dimensional polynomial interpolation from nonuniform samples," *IEEE Transactions on Signal Processing*, vol. 40, pp. 169–180, 1992.

[12] T. Strohmer, "Computationally attractive reconstruction of bandlimited images from irregular samples," *IEEE Transactions on Image Processing*, vol. 6, pp. 540–548, Apr. 1997.

[13] T. Strohmer, "Fast scattered data approximation with neumann and other boundary conditions," *Linear Algebra and Application*, To appear.

[14] D. Shepard, "A two dimensional interpolation function for irregularly spaced data," in *Proceedings of the 23rd National Conference, ACM, New York*, 1968, pp. 517–523.

[15] R. Franke and G. Nielson, "Smooth interpolation large sets of scattered data," *International Journal for Numerical Methods in Engineering*, vol. 15, pp. 1691–1704, 1980.

[16] M. Fenn and G. Steidl, *Geometric Properties from Incomplete Data*, chapter Robust local approximation of scattered data, Kluwer Academic Press, to appear edition, 2005.

[17] M. Margaliot and C. Gotsman, "Piecewise-linear surface approximation from noisy scattered samples," in *Proceedings of the IEEE conference on Visualization*, 1994, pp. 61–68.

[18] M. Lai, "Scattered data interpolation and approximation using bivariate c^1 piecewise cubic polynomials," *Computer Aided Geometric Design*, vol. 13, pp. 81–88, 1996.

[19] M. Bertram et al, "Piecewise optimal triangulation for the approximation of scattered data in the plane," *Computer Aided Geometric Design*, vol. 17, pp. 767–787, 2000.

[20] J. Haber et al, "Smooth approximation and rendering of large scattered data sets," in *Proceedings of the IEEE conference on Visualization*, 2001, pp. 341–348.

[21] E. Chan and B. Ong, "Range restricted scattered data interpolation using convex combination of cubic bezier triangles," *Journal of Computational and Applied Mathematics*, vol. 136, pp. 135–147, 2001.

[22] Y. Liu and G. Walter, "Irregular sampling in wavelet subspaces," *Journal of Fourier Analysis and Applications*, vol. 2, pp. 181–189, 1996.

[23] Y. Liu, "Irregular sampling for spline wavelet subspaces," *IEEE Transactions on Image Processing*, vol. 42, pp. 623–627, 1996.

[24] O. Christensen, "Moment problems for frames and applications to irregular sampling and Gabor frames," *Applied Computational Harmonic Analysis*, vol. 3, pp. 82–86, 1996.

[25] W. Chen, S. Itoh, and J. Shiki, "Irregular sampling theorem for wavelet subspaces," *IEEE Transactions on Information Theory*, vol. 44, pp. 1131–1142, 1998.

[26] A. Aldroubi and H. Feichtinger, "Exact iterative reconstruction algorithm for multivariate irregularly sampled functions in spline-like spaces: The l_p theory," *Proceedings of American Mathematical Society*, vol. 126, pp. 2677–2686, 1998.

[27] X. Zhou and W. Sun, "On the sampling theorem for wavelet subspaces," *Journal of Fourier Analysis and Applications*, vol. 5, pp. 347–354, 1999.

[28] A. Aldroubi and K. Gröchenig, "Beurling-Landau-type theorems for nonuniform sampling in shift-invariant spaces," *Journal of Fourier Analysis and Application*, vol. 6, pp. 91–101, 2000.

[29] A. Aldroubi and K. Grochenig, "Nonuniform sampling and reconstruction in shift invariant spaces," *SIAM Review*, vol. 43, pp. 585–620, 2001.

[30] J. Duchon, "Splines minimizing rotation-invariant semi-norms in sobolev spaces," *Constructive Theory of Functions of Several Variables*, pp. 85–100, 1977.

[31] D. Myers, S. Iaco, D. Posa, and L. Cesare, "Space-time radial basis functions," *Computers and Mathematics with Applications*, vol. 43, pp. 539–549, 2002.

[32] A. Bouhamidi and A. Mehaute, "Radial basis functions under tension," *Journal of Approximation Theory*, vol. 127, pp. 135–154, 2004.

[33] M. Buhmann, *Radial Basis Functions: Theory and Implementations*, Cambridge University Press, 2003.

[34] C. A. Micchelli, "Interpolation of scattered data: distance matrices and conditionally positive definite functions," *Constructive Approximation*, vol. 2, pp. 11–22, 1986.

[35] J. Yoon, "Spectral approximation orders of radial basis function interpolation on the sobolev space," *SIAM Journal of Mathematical Analysis*, vol. 33, pp. 946–958, 2001.

[36] J.. Yoon, "Interpolation by radial basis functions on sobolev space," *Journal of Approximation Theory*, vol. 112, pp. 1–15, 2001.

[37] J. Yoon, "Approximation by conditionally positive definite functions with finitely many centers," *Trends in Approximation Theory*, pp. 437–446, 2001.

[38] H. Gutmann, "On the semi-norm of radial basis function interpolants," *Journal of Approximation Theory*, vol. 111, pp. 315–328, 2001.

[39] F. Hickernell and Y. Hon, "Radial basis function approximations as smoothing splines," *Applied Mathematics and Computation*, vol. 102, pp. 1–24, 1999.

[40] L. Ling and E. Kansa, "Preconditioning for radial basis functions with domain decomposition methods," *Mathematics and Computer Modelling*, vol. 40, pp. 1413–1427, 2004.

[41] R.K. Beatson, W. A Light, and S. Billings, "Fast solutions of radial basis function interpolation equations: Domain decomposistion methods," *SIAM Journal of Scientific Computing*, vol. 22, No. 5, pp. 1717–1740, 2000.

[42] G. Roussos and B. Baxter, "Rapid evaluation of radial basis functions," *Journal of Computational and Applied Mathematics*, vol. 180, pp. 51–70, 2005.

[43] F. Narcowich, R. Schaback, and J. Ward, "Multilevel interpolation and approximation," *Applied and Computational Harmonic Analysis*, vol. 7, pp. 243–261, 1999.

[44] D. Lazzaro and B. Montefusco, "Radial basis functions for the multivariate interpolation of large scattered data sets," *Journal of Computational and Applied Mathematics*, vol. 140, pp. 521–536, 2002.

[45] Y. Ohtake, A. Belyaev, and H. Seidel, "3d scattered data interpolation and approximation with multilevel compactly supported RBFs," *Graphical Models*, vol. 67, pp. 150–165, 2005.

[46] E. Larsson and B. Fornberg, "Theoretical and computational aspects of multivariate interpolation with increasingly flat radial basis functions," *Computers and Mathematics with Applications*, vol. 49, pp. 103–130, 2005.

[47] G. Wabha, *Spline Models for Observational Data*, SIAM, 1990.

[48] F. L. Bookstein, *Morphometric Tools for Landmark Data*, Cambridge University Press, 1991.

[49] L. Amodei and M. Benbourhim, "A vector spline approximation," *Journal of Approximation Theory*, vol. 67, pp. 51–79, 1991.

[50] F. Dodu and C. Rabut, "Vector interpolation using radial-basis-like functions," *Computers and Mathematics with Applications*, vol. 43, pp. 393–411, 2002.

[51] F. Dodu and C. Rabut, "Irrotational or divergence-free interpolation," *Numerische Mathematik*, vol. 98, pp. 477–498, 2004.

[52] M. Atteia, *Hilbertian Kernels and Spline Functions*, North Holland, 1992.

[53] M. Arigovindan, M. Sühling, P. Hunziker, and M. Unser, "Variational image reconstruction from arbitrarily spaced samples: A fast multiresolution spline solution," *IEEE Transactions on Image Processing*, vol. 14, no. 4, pp. 450–460, April 2005.

[54] C. de Boor, *A Practical Guide to Splines*, Springer-Verlag, 1978.

[55] M. Unser, "Splines: A perfect fit for signal and image processing," *IEEE Signal Processing Magazine*, vol. 16, No. 2, pp. 22–38, Nov. 1999.

[56] I. J. Schoenberg, "Spline functions and the problem of graduation," *Proc. Nat. Acad. Sci.*, vol. 52, pp. 652–654, 1964.

[57] R. L. Eubank, *Nonparametric regression and spline smoothing*, Marcel Dekker, Inc, 1999.

[58] M. Unser, "Approximation power of biorthogonal wavelet expansions," *IEEE Transactions on Signal Processing*, vol. 44, pp. 519–527, 1996.

[59] T. Blu, P. Thevenaz, and M. Unser, "MOMS: Maximal-order interpolation of minimal support," *IEEE Transactions on Image Processing*, vol. 10, pp. 1069–1080, 2001.

[60] G. Strang and T. Nguyen, *Wavelets and filter banks*, Wellesley-Cambridge Press, 1997.

[61] M. Vetterli and J. Kovacevic, *Wavelets and subband coding*, Prentice Hall, 1995.

[62] W. Hakbusch, *Iterative Solutions of Large Sparse Systems of Equations*, Springer-Verlag, 1994.

[63] W. L. Briggs, V. E. Henson, and S. F. McCormick, *A Multigrid Tutorial*, SIAM, 2000.

[64] A Aldroubi, M. Unser, and M. Eden, "Cardinal spline filters: Stability and convergence to the ideal sinc interpolator," *Signal Processing*, vol. 28, pp. 127–138, 1992.

[65] M. Berman, "Automated smoothing of image and other regularly spaced data," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, pp. 460–468, 1994.

[66] A. Munoz, T. Blu, and M. Unser, "Least-squares image resizing using finite differences," *IEEE Transactions on Image Processing*, vol. 10, pp. 1365–1378, 2001.

[67] J. Kybic, T. Blu, and M. Unser, "Generalized sampling: A variational approach—Part I: Theory," *IEEE Transactions on Signal Processing*, vol. 50, pp. 1965–1976, 2002.

[68] H. Tsujino et al., "Combination of pulsed-wave doppler and real-time three-dimensional color doppler for quantifying the stroke volume in the left ventricular outflow tract," *Ultrasound in Medicine and biology*, vol. 30, no. 11, pp. 1441–1446, 2004.

[69] C. Holland et al., "Volumetric flow estimation in vivo and in vitro using pulsed-doppler ultrasound," *Ultrasound in Medicine and biology*, vol. 22, no. 5, pp. 591–603, 1996.

[70] C. DeGroot et al., "Evaluation of 3-D color doppler ultrasound for the measurement of proximal isovelocity surface area," *Ultrasound in Medicine and biology*, vol. 26, no. 6, pp. 989–999, 2000.

[71] W.N. McDicken, G.R. Sutherland, C.M. Moran, and L.N. Gordon, "Colour doppler velocity imaging of the myocardium," *Ultrasound Med. Biol.*, vol. 18, no. 6-7, pp. 651–654, 1992.

[72] G.R. Sutherland, M.J. Steward, K.W.E. Grounstroem, C.M. Moran, A. Fleming, F.J. Guell-Peris, R.A. Riemersma, L.N. Fenn, K.A. Fox, and W.N. Mc Dicken, "Color doppler myocardial imaging: a new technique for assessment of myocardial function," *J. Am. Soc. Echocardiogr.*, vol. 7, no. 5, pp. 441–458, Sept.–Oct. 1994.

[73] K. Miyatake, M. Yamagishi, N. Tanaka, M. Uematsu, N. Yamazaki, Y. Mine, A. Sano, and M. Hirama, "New method of evaluating left ventricular wall motion by color-coded tissue Doppler imaging: in vitro and in vivo studies," *J. Am. Coll. Cardiol.*, vol. 25, no. 3, pp. 717–724, Mar. 1995.

[74] J. Gorcsan, V.K. Gulati, W.A. Mandarino, and W.E. Katz, "Color-coded measures of myocardial velocity throughout the cardiac cycle by tissue Doppler imaging to quantify regional left ventricular function," *Am. Heart J.*, vol. 131, no. 6, pp. 1203–1213, June 1996.

[75] L. Galiuto, G. Ignone, and A.N. DeMaria, "Contraction and relaxation velocities of the normal left ventricle using pulsed-wave tissue Doppler echocardiography," *Am. J. Cardiology*, vol. 81, no. 5, pp. 609–614, Mar. 1998.

[76] P.R. Hunziker, M.H. Picard, N. Jander, M. Scherrer-Crosbie, M. Pfisterer, and P.T. Buser, "Regional wall motion assessment in stress echocardiography by tissue Doppler bull's-eyes.," *J. Am. Soc. Echocardiography*, vol. 12, no. 3, pp. 196–202, Mar. 1999.

[77] S. Deans, *The Radon Transform and Some of Its Applications*, Wiley, 1983.

[78] J. Kybic, T. Blu, and M. Unser, "Generalized sampling: A variational approach—Part II: Theory," *IEEE Transactions on Signal Processing*, vol. 50, no. 8, pp. 1965–1976, August 2002.

[79] S. Norton, "Tomographic reconstruction of 2D vector fields: Application for flow imaging," *Geophysical Journal*, vol. 97, pp. 161–168, 1988.

[80] J. Prince, "Tomographic reconstruction of 3-D vector fields using inner product probes," *IEEE Transactions on Image Processing*, vol. 3, pp. 216–219, 1994.

[81] J. Prince, "Convolution backprojection fromulas for 3-D vector tomography with application to MRI," *IEEE Transactions on Image Processing*, vol. 5, pp. 1462–1472, 1996.

[82] W. Munk and C. Wunsch, "Ocean acoustic tomography: A scheme for large scale monitoring," *Deep-Sea Research*, vol. 26A, pp. 123–161, 1979.

[83] B. Howe, P. Worchester, and W. Munk, "Ocean acoustic tomography: Mesoscale velocity," *Journal of Geophysical Research*, vol. 92, pp. 3785–3805, 1987.

[84] S. Norton, "Unique tomographic reconstruction of vector fields using boundary data," *IEEE Transactions on Image Processing*, vol. 1, pp. 406–412, 1992.

[85] G. Sparr, K. Strahlen, K. Lindstrom, and H. Persson, "Doppler tomography for vector fields," *Inverse Problems*, vol. 11, pp. 1051–1061, 1995.

[86] T. Jansson et al, "Ultrasound doppler vector tomography measurements of directional blood flow," *Ultrasound in Medicine and Biology*, vol. 23, pp. 47–57, 1997.

Author's Acknowledgements

I am grateful to my thesis supervisor Prof. Michael Unser for his support and guidance throughout the development of this work. His enthusiasm has been largely responsible for maintaining a consistent motivation on the topic of my research. He always encourages new ideas as well as refines them. Those refinements have been crucial for the progress towards my research goals.

I thank Dr. Thierry Blu for his guidance on some theoretical aspects of this thesis. Special thanks to Dr. Daniel Sage, who has always been providing any necessary help during my stay in the Biomedical Imaging Group. I admire his patience in helping the students.

I thank the jury members, Prof. Martin Vetterli, Prof. Christophe Rabut, Dr. MER. Michel Bierlaire, and Dr. med. Patrick Hunziker for their valuable time in evaluating this thesis. I appreciate their constructive and encouraging feedback about the thesis.

The collaboration with echocardiography specialists Dr. med. Patrick Hunziker, Dr. med. Christian Jansen, and Dr. Michael Sühling forms one of the key ingredients of this thesis. Michael Sühling has also been helping me in solving my computer-related problems. On the similar grounds, the help of Dr. Dimitri Van De Ville should also be appreciated.

Thanks to Dr. Philippe Thévenaz for his help in managing my administrative duties in the Biomedical Imaging Group. The administrative help provided by Mlle. Manuelle Borruat and Mme. Yvette Bernhard is greatly appreciated.

I thank Bubhaneswar Mandal, Rajesh Langoju, Sathish Ramani, Ramachandra Rao for their timely help and the company during my stay in

Lausanne. I also thank Mathews Jacob for hosting me when I arrived here to start my PhD work.

Finally, I thank the past and present members and visitors of Biomedical Imaging Group, Manuela Feilner, Brigitte Forster, Stefan Horbelt, Michael Liebling, Eric Maijering, Nadja Subotic, Shai Tirosh, François Aguet, Ildar Khalidov, Florian Luisier, Cédric Vonesch, Bart Janssen, Akira Hirabayashi, Koichi Ichige, Laurent Condat, Vijay Divi, and Cristina Manfredotti for their cooperation.

Muthuvel Arigovindan
Lausanne, September 2005

Author's CV

Muthuvel Arigovindan was born in Pondicherry, India in 1974. He received his B. Tech. degree in electronics and communication engineering from Pondicherry Engineering College in 1995. He received the M. Sc. degree from Indian Institute of Science in 1999, where he worked on multirate signal processing. He then carried out his PhD research in the Biomedical Imaging Group at the Swiss Federal Institute of Technology Lausanne, where he worked on variational methods for reconstruction of scalar and vector images from non-uniform samples.

He is mainly interested in numerical methods for inverse problems. His specific interests include non-uniform sampling, flow field imaging, magnetic source imaging, and vector tomography.