Wavelets and Fluid Motion Estimation

Pierre Dérian

ANNÉE 2012

THÈSE / UNIVERSITÉ DE RENNES 1
sous le sceau de l'Université Européenne de Bretagne

pour le grade de

DOCTEUR DE L'UNIVERSITÉ DE RENNES 1

Mention : Mathématiques Appliquées

Ecole doctorale Matisse

présentée par

Pierre Dérian

préparée à l'unité de recherche INRIA
Centre INRIA Rennes - Bretagne Atlantique
Université Rennes 1

Wavelets and Fluid Motion Estimation

Thèse soutenue à l'INRIA Rennes le 7 novembre 2012

devant le jury composé de :

Didier AUROUX
Professeur, Laboratoire J.-A. Dieudonné / rapporteur

Valérie PERRIER
Professeur, Laboratoire J. Kuntzmann / rapporteur

Patrick BOUTHEMY
Directeur de Recherche, INRIA / examinateur

Jalal FADILI
Professeur, GREYC-ENSICAEN / examinateur

Benjamin LECLAIRE
Ingénieur de Recherche, ONERA / examinateur

Étienne MÉMIN
Directeur de Recherche, INRIA / directeur de thèse

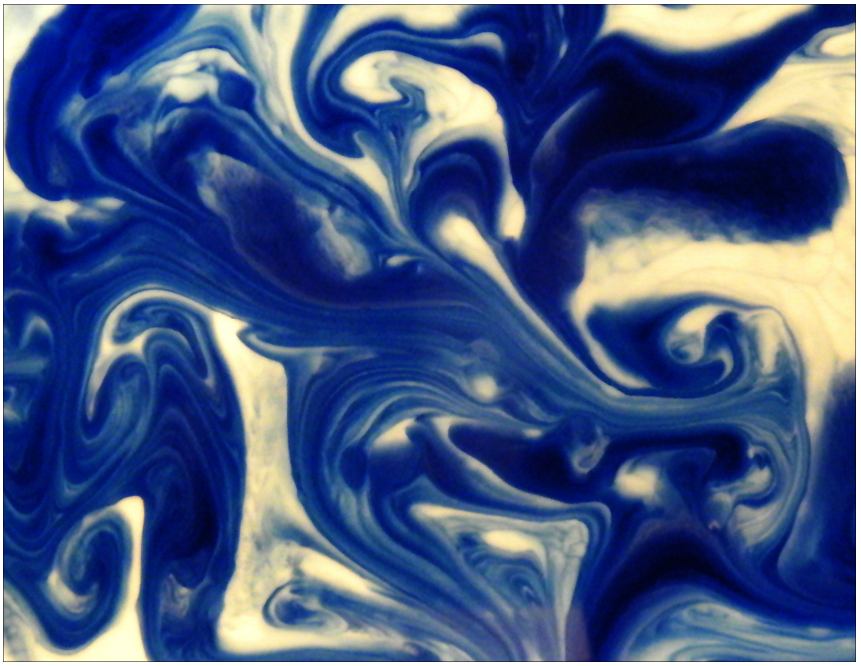
Patrick HÉAS
Chargé de Recherche, INRIA / co-directeur de thèse

THESIS / UNIVERSITÉ DE RENNES 1
part of the Université Européenne de Bretagne

Mention: Applied Mathematics
Matisse doctoral school

Wavelets & Fluid Motion Estimation

by Pierre DÉRIAN



Conducted from 2009 to 2012 at
Inria Rennes - Bretagne Atlantique
Fluminance Team
under the supervision of Étienne MÉMIN and Patrick HÉAS

UNIVERSITÉ DE RENNES 1

ueb

Contents

List of Figures

Notations and Abbreviations

Maths Notations and Conventions

Velocity/Motion Related

- $\boldsymbol{x} = (x_1, x_2)\,\boldsymbol{T} \in \mathbb{R}^2$;
- $\boldsymbol{v}(\boldsymbol{x})$ 2D velocity vector, $\boldsymbol{v}(\boldsymbol{x}) = (v_1(\boldsymbol{x}), v_2(\boldsymbol{x}))^{\boldsymbol{T}}$;

Image Related

- $\Omega \subset \mathbb{R}^2$ the image domain;
- $I_0(\boldsymbol{x}), \boldsymbol{x} \in \Omega$ an image from a sequence at a given time t;
- $I_1(\boldsymbol{x}), \boldsymbol{x} \in \Omega$ an image from a sequence at a given time $t+1$;
- n_p the total number of pixels in the image.

Wavelet Related

- 2^j a scale – see Remark 1 below;
- φ a scaling function;
- ψ a wavelet function;
- V_j an approximation space at scale 2^j;
- W_j a detail space at scale 2^j;
- $\mathbf{V}_j$ a 2D approximation space at scale 2^j: $\mathbf{V}_j = V_j \otimes V_j$;
- Θ_i set of coefficients representing velocity component v_i in the chosen basis;
- $\Theta_{i|j}$ set of coefficients Θ_i, restricted to scales coarser and up to j;
- Φ the projection operator from a given wavelet basis to the canonical basis (also referred to as the *reconstruction* operator);
- Φ^{-1} the projection operator from the canonical basis to a given wavelet basis (also referred to as the *decomposition* operator);

Miscellaneous

- $H^1(\mathbb{R}^2)$: the Sobolev space where functions of $\mathbb{R}^2$ and their 1st order derivatives $\in \mathbf{L}^2(\mathbb{R}^2)$.
- $\langle \cdot\,;\cdot \rangle$ the dot-product;
- $\otimes$ the tensor-product;
- δ_{ij} the Kronecker symbol;
- $\circ$ the Hadamard product, element-wise matrix multiplication: $\big(A \circ B\big)_{i,j} = A_{i,j} B_{i,j}$;
- : the Frobenius inner product: $A : B = \sum_{i,j} A_{i,j} B_{i,j} = \sum_{i,j} \big(A \circ B\big)_{i,j}$;
- $\check{\cdot}$ the expansion operator: $\check{x}[n] = x[p]$ if $n = 2p$, 0 elsewhere;
- $\downarrow$ the decimation operator: $\downarrow x[n] = x[2n]$;
- $\bar{\cdot}$ the time-reverse operator: $\bar{x}[n] = x[-n]$;
- div the divergence operator. For $\boldsymbol{v} \in \mathbb{R}^3$,

$$\mathrm{div}(\boldsymbol{v}) = \nabla \cdot \boldsymbol{v} = \partial_{x_1} v_1 + \partial_{x_2} v_2 + \partial_{x_3} v_3 \ \in \mathbb{R}$$

For $\boldsymbol{v} \in \mathbb{R}^2$, $\mathrm{div}(\boldsymbol{v}) = \partial_{x_1} v_1 + \partial_{x_2} v_2$;

– **curl** the rotational operator. For $\boldsymbol{v} \in \mathbb{R}^3$,

$$\mathbf{curl}(\boldsymbol{v}) = \nabla \times \boldsymbol{v} = (\partial_{x_2} v_3 - \partial_{x_3} v_2,\, -\partial_{x_1} v_3 + \partial_{x_3} v_1,\, \partial_{x_1} v_2 - \partial_{x_2} v_1)^{\boldsymbol{T}} \in \mathbb{R}^3 .$$

To compute the **curl** of $\boldsymbol{v} \in \mathbb{R}^2$, we consider

$$\nabla \times (v_1(x_1, x_2),\, v_2(x_1, x_2),\, 0)^{\boldsymbol{T}} = (0,\, 0,\, \partial_{x_1} v_2 - \partial_{x_2} v_1)^{\boldsymbol{T}} ,$$

and only keep the third component $\partial_{x_1} v_2 - \partial_{x_2} v_1$. This scalar field will be written $\mathrm{curl}(\boldsymbol{v})$ for short. Similarly, the **curl** of a scalar field $z(x_1, x_2) \in \mathbb{R}$ can be defined by

$$\mathbf{curl}(z) = \nabla \times (0\,, 0,\, z(x_1, x_2))^{\boldsymbol{T}} = (\partial_{x_2} z,\, -\partial_{x_1} z,\, 0)^{\boldsymbol{T}} ,$$

from which only the two first component are kept, giving a 2D vector field.

Remark 1. *By abuse of notation, the actual scale value 2^j will be often described by its exponent j only, i.e. "scale j" actually means "scale 2^j". With finite signals, $j \leq 0$ and $0 < 2^j \leq 1$. This* scale *has not to be confused with the corresponding* resolution 2^{-j}*, which corresponds to the number of samples for a discrete signal.*

Remark 2. *Dilations and translations of a given function f are written in a general form $f_{j,k}(x) \triangleq 2^{-j/2} f(2^{-j}x - k)$, where j stands for the scale of the dilation (see previous remark) and k is the translation index.*

Abbreviations

DFD: displaced frame difference;

PSD: power spectrum density;

IHT: isotropic homogeneous turbulence;

(B)MRA: (biorthogonal) multiresolution approximation;

OFC: optical flow constraint;

PIV: particle image velocimetry;

px: pixel (unit);

RMSE: root mean square end-point error;

SST: sea surface temperature;

VM: vanishing moments;

Résumé en Français

Contexte des Travaux

L'analyse d'écoulements de fluide est une pierre angulaire dans nombre de disciplines scientifiques et techniques. Les études d'aérodynamique préalables à la conception d'avions ou à l'implémentation d'un parc éolien, la prévision météorologique à court terme tout comme la simulation du changement climatique, et bien entendu la physique fondamentale des fluides ; ces exemples s'appuient tous, à des degrés et des échelles variés, sur des analyses d'écoulements. Mais avant toute analyse, il est nécessaire de s'assurer de la connaissance de l'état de l'écoulement en question – c'est à dire, de le décrire à partir de certaines, sinon toutes, des grandeurs caractéristiques que sont la vitesse, la pression, la température... S'il est souvent possible de recourir à des simulations numériques, ces dernières peuvent néanmoins s'avérer complexes ou coûteuses à mettre en oeuvre. Par ailleurs, certaines de ces approches numériques se "nourrissent" de mesures réelles, c'est donc à ces dernières que nous nous intéresserons ici, et plus particulièrement à la mesure de la vitesse.

Les mouvements de fluides ont une nature complexe, qui est modélisée par les célèbres équations de Navier-Stokes. Lorsque le fluide est dans un état *turbulent*, ses mouvements font intervenir un grand nombre de tourbillons et vortex qui interagissent sur une vaste gamme d'échelles spatiales et temporelles. La description d'un tel écoulement demande donc de tenir compte de l'évolution spatiale et temporelle de ces structures, aussi finement que possible ; c'est ici que les choses se corsent. L'expérimentateur se heurte en effet à un dilemme : d'un coté, l'utilisation de sondes, par exemple l'anémométrie par fil chaud, permet d'obtenir une mesure très fine de l'évolution temporelle, mais qui reste limitée à un point unique de l'espace. Il est bien entendu possible d'utiliser un réseau de capteurs pour obtenir plusieurs mesures simultanées, mais cela augmente la complexité du dispositif et peut s'avérer intrusif pour l'écoulement, tout en ne fournissant qu'une résolution spatiale très grossière. D'un autre coté, les structures et le comportement d'un écoulement sont très simplement mises en évidence aux yeux de l'expérimentateur par l'utilisation de *traceurs* : colorant, fumée, petites particules (Figure 1)... Ou, plus naïvement, nuages, écume ou feuilles mortes, dont l'observation du mouvement renseigne immédiatement tout un chacun sur la direction du vent ou du courant sous-jacent . Il est ainsi possible de mettre en évidence puis d'analyser qualitativement des phénomènes complexes de la mécanique des fluides, comme le montre le célèbre album de photographie de M. van Dyke *An Album of Fluid Motion* [43]. Mais alors, comment tirer partie, d'un point de vue quantitatif, de ces observations ?

C'est ici que la mécanique des fluides rencontre la *vision par ordinateur*. Ce domaine s'intéresse à l'étude des mécanismes de la vision humaine, et au développement d'approches artificielles similaires. Acquisition d'image, reconnaissance de formes, mais aussi estimation de mouvement sont quelque-uns des nombreux aspects du problème. Ces techniques ont grandement bénéficié du développement exponentiel, à la fin du XXe siècle, de l'informatique et de l'imagerie numérique, au point d'être aujourd'hui utilisées dans des applications aussi diverses que le suivi des inondations par satellite, la compression vidéo, la navigation autonome (voitures, drones, ...), l'archéologie ou la chirurgie (e.g. reconstruction 3D). La porte est alors ouverte à l'utilisation de ces techniques, en particulier l'estimation de mouvements à partir de d'images, dans le cadre de l'observation de fluides présentée plus tôt. Il s'agit donc de mettre en place une visualisation de l'écoulement considéré, d'en réaliser un enregistrement numérique à l'aide d'une caméra adéquate, et enfin de traiter les images obtenues pour en extraire le mouvement supposé.

FIGURE 1 – Visualisation, à l'aide de particules, de l'écoulement autour d'une dune en forme de croissant (*barkhane*, forme sombre en haut). La recirculation créée à l'abri de la dune montre des trajectoires complexes à faible vitesse, tandis que l'écoulement extérieur présente des trajectoires quasi rectilignes et des vitesses beaucoup plus élevées.

Remarque. Afin d'éviter d'allonger inutilement cette synthèse, certains symboles utilisés ci-après ne sont pas explicités dans cette partie. Le lecteur est invité à se reporter au besoin à la liste des Notations & Abréviations page ix.

Approches Usuelles

Si le concept est simple, sa mise en pratique l'est cependant bien moins. Il est important de garder à l'esprit que les méthodes décrites ci-après se proposent d'estimer le *mouvement apparent*, ou *flux (flot) optique*, qui est la projection 2D, dans le plan de l'image, du mouvement réel a priori 3D. On utilise alors les variations temporelles et spatiales d'une quantité observable de l'image, telle que la luminance, pour déterminer le mouvement apparent entre deux images consécutives.

Estimation de mouvement – premiers concepts

Il s'agit, dans un premier temps, de relier le champ de vitesse inconnu aux données image. L'équation qui s'en charge est appelée *modèle de données*, elle découle souvent d'une hypothèse de conservation de la luminance (intensité lumineuse) le long de la trajectoire d'une particule. En notant $I(\boldsymbol{x}, t)$ la luminance en un point de l'image $\boldsymbol{x} \in \Omega$ et à un instant t et $\boldsymbol{v}(\boldsymbol{x}, t)$ le champ de vitesse apparente que l'on cherchera à estimer,

$$\frac{dI(\boldsymbol{x}, t)}{dt} = \frac{\partial I}{\partial t}(\boldsymbol{x}, t) + \boldsymbol{v}(\boldsymbol{x}, t) \cdot \nabla I(\boldsymbol{x}, t) = 0\,. \tag{1}$$

A ce stade, une première difficulté surgit. En effet, de par l'équation ci-dessus, seule la composante de mouvement perpendiculaire aux courbes d'iso-intensité de l'image peut être déduite ; cette situation est connue comme le *problème de l'ouverture*. Puisqu'aucune information sur la composante tangentielle n'est disponible, une infinité de solutions est envisageables – le problème d'estimation est donc sous-déterminé. En outre, la composante perpendiculaire ne peut être estimée qu'en présence de gradients spatio-temporels de luminance non nuls, ce qui pose un nouveau problème d'indétermination à l'intérieur d'éventuelles zones uniformes. Afin de lever cette indétermination, il est ainsi nécessaire d'ajouter un *mécanisme de régularisation*, qui, avec le modèle de données, constituent les deux aspects fondamentaux du problème d'estimation de mouvement.

Méthodes d'estimation

La technique la plus répandue se nomme *Digital Image Correlations* (DIC), elle réalise un ensemble de mesures *indépendantes* et *locales* sur des sous-régions de l'image. Il s'agit ici de corréler une région (fenêtre) de la première image avec une autre région de la seconde image ; le vecteur qui engendre un pic maximal de corrélation est gardé comme une mesure du déplacement au centre de la fenêtre en question. Le modèle de données est ici la fonction de corrélation croisée, et la régularisation est imposée par un vecteur constant sur une fenêtre donnée, fenêtre dont la taille doit être choisie de façon à contenir suffisamment d'information pour s'affranchir du problème d'indétermination. En considérant un sous-ensemble de points de l'image $\Omega_C \in \Omega$, le problème d'estimation s'écrit

$$\forall\, \boldsymbol{x} \in \Omega_C\,,\ \hat{\boldsymbol{v}}(\boldsymbol{x}) = \arg\min_{\boldsymbol{v}} \sum_{\boldsymbol{y}\in W(\boldsymbol{x})} -\frac{\left(I_1(\boldsymbol{y}+\boldsymbol{v}) - \mu_1(\boldsymbol{x}+\boldsymbol{v})\right)\left(I_0(\boldsymbol{y}) - \mu_1(\boldsymbol{x})\right)}{\sigma_1^2(\boldsymbol{x}+\boldsymbol{v})\sigma_0^2(\boldsymbol{x})}\,, \tag{2}$$

où $\mu_j(\boldsymbol{x})$ et $\sigma_j(\boldsymbol{x})$ sont des estimations de la moyenne et la variance de la luminance de l'image I_j, sur une fenêtre $W(\boldsymbol{x})$ centrée en $\boldsymbol{x} \in \Omega_C$. Les principaux inconvénients de cette méthode sont, d'une part que le modèle de données et la régularisation sont fixes, d'autre part que la résolution du champ obtenu est plus grossière que les images d'origine (typiquement, un vecteur tous les 4 pixels). En revanche, elle est robuste, s'implémente efficacement grâce à la transformée de Fourier rapide, et comme les mesures sont indépendantes, elle est en outre fortement parallélisable. Les corrélations fonctionnent particulièrement bien avec les images de particules (*Particle Image Velocimetry*, PIV), elles sont ainsi couramment utilisées en laboratoire.

Une seconde famille de méthodes, dites *paramétriques*, régularise le problème en adoptant une formulation paramétrique locale. Tout comme la méthode des corrélations, ces approches reposent sur un ensemble de sous-régions de l'image. A l'intérieur de chaque région, le champ de vitesse recherché est exprimé en fonction d'un petit nombre de paramètres. Les champs obtenus sont typiquement constant, affines ou quadratiques par morceaux. Le problème d'estimation s'écrit par exemple :

$$\begin{cases} \boldsymbol{v}(x) = \Phi(\boldsymbol{x})\theta\,, \\ \hat{\theta}_{\boldsymbol{x}} = \arg\min_{\theta} \dfrac{1}{2}\displaystyle\int_{W(\boldsymbol{x})} g(\boldsymbol{x}-\boldsymbol{y})\Big[\partial_t I(\boldsymbol{y}) + \nabla I(\boldsymbol{y})\cdot\Phi(\boldsymbol{y})\theta_{\boldsymbol{x}}\Big]^2 d\boldsymbol{y}\,, \end{cases} \tag{3}$$

où θ_x est le jeu de paramètres pour la fenêtre $W(\boldsymbol{x})$, et g une fonction de pondération donnant plus d'importance aux pixels $\boldsymbol{y}$ proches de $\boldsymbol{x}$ (typiquement, une gaussienne). L'estimateur de Lucas & Kanade [27], couramment employé, s'appuie sur une formulation constante par morceaux.

Une troisième famille de méthodes, enfin, s'attache à fournir une solution *globale* via la minimisation d'une fonctionnelle définie sur tout le domaine de l'image. Par abus de langage, ces méthodes sont généralement appelées *optical flow* (flux, flot optique), bien que formellement les approches paramétriques ou par corrélations croisées mesurent également le flux optique. Le problème peut s'exprimer comme :

$$\begin{aligned} \hat{\boldsymbol{v}} &= \arg\min_{\boldsymbol{v}} J_{\text{data}}(I,\boldsymbol{v}) + \alpha J_{\text{reg}}(\boldsymbol{v}) \\ &= \arg\min_{\boldsymbol{v}} \frac{1}{2}\int_\Omega \Big[f_{\text{data}}(I(\boldsymbol{x},t),\boldsymbol{v}(\boldsymbol{x},t))\Big]^2 d\boldsymbol{x} + \frac{\alpha}{2}\int_\Omega \Big[f_{\text{reg}}(\boldsymbol{v}(\boldsymbol{x},t))\Big]^2 d\boldsymbol{x} \end{aligned} \tag{4}$$

où f_{data} et f_{reg} sont le modèle de données et la régularisation, respectivement, et α un paramètre scalaire qui équilibre les deux termes de la fonctionnelle. Les modèles de données les plus fréquemment employés découlent de (1) ; selon que l'on garde la dérivée particulaire ou que l'on l'intègre entre deux instants, on obtient les modèles connus sous les noms d'*Optic Flow Constraint* (OFC) et de *Displaced Frame Difference* (DFD). En notant $I_0(\boldsymbol{x}) = I(\boldsymbol{x},t)$ et $I_1(\boldsymbol{x}) = I(\boldsymbol{x},t+1)$ la luminance de deux images successives dans la séquence considérée, ces modèles s'écrivent :

$$\text{(OFC)} \quad I_1(\boldsymbol{x}) - I_0(\boldsymbol{x}) + \boldsymbol{v}(\boldsymbol{x})\cdot\nabla I_1(\boldsymbol{x}) = 0\,; \tag{5}$$

$$\text{(DFD)} \quad I_0(\boldsymbol{x}) - I_1(\boldsymbol{x}+\boldsymbol{v}(\boldsymbol{x})) = 0\,. \tag{6}$$

L'OFC (5) est linéaire en $\boldsymbol{v}$, ce qui facilite grandement la minimisation de la fonctionnelle. Ce modèle n'est en revanche valide que pour de petits déplacements (dans la zone de linéarité de la

luminance), de sorte qu'en pratique il est nécessaire de l'imbriquer dans un schéma de minimisation incrémental. Ces schémas réalisent une succession d'estimations, à partir d'approximations d'abord grossières, puis de plus en plus fines, des images. La solution du problème est donnée par la somme des mouvements estimés aux différents étages de la "pyramide" d'images ainsi constituée. Ces approches ont l'inconvénient de geler, à un étage donné, les estimés précédemment obtenus. La DFD (6) est non-linéaire en $\boldsymbol{v}$, ce qui complexifie le problème de minimisation. En revanche, elle n'est pas restreinte aux petits déplacements. Ses limites découlent plutôt de l'existence de minima locaux, qui peuvent notamment surgir dans le cadre de grands déplacements. Enfin, les termes de régularisation tendent classiquement à privilégier une solution "lisse", en pénalisant les gradients trop importants :

$$J_{\text{reg}}(\boldsymbol{v}) = \frac{1}{2} \int_{\Omega} |\nabla v_1(\boldsymbol{x})|^2 + |\nabla v_2(\boldsymbol{x})|^2 \, d\boldsymbol{x} \,. \tag{7}$$

Ce lissage au premier ordre a été introduit par Horn & Schunck [20]. Un des grands avantages des méthodes de flux optique est leur modularité : il est en effet possible d'adapter le modèle de données et le terme de régularisation aux spécificités des images et du type de mouvement considérés. En outre, ces approches retournent un champ dense, soit un vecteur par pixel ; elles permettent ainsi l'estimation d'échelles plus fines que les méthodes précédentes. En revanche, elles impliquent un bien plus grand nombre d'inconnues : pour des images de 1024×1024 px, il y a 2 097 152 variables ! Enfin, comme la fonctionnelle est globale, la parallélisation est moins évidente.

Approches dédiées aux fluides

Les méthodes de corrélations croisées sont largement employées, et toujours développées [4]. Des systèmes complets, comprenant lasers, caméra, cartes d'acquisition et logiciel de traitement sont ainsi disponibles sur le marché, tout comme des solutions libres.

Les approches paramétriques et denses ont été explorée plus récemment, en mécanique des fluides expérimentale comme en géophysique. On trouve ainsi un modèle de données basé sur l'équation de continuité [11, 16], une modélisation paramétrique par "particules" de vortex et source [12], ou linéaire par morceaux [2]. Du coté des termes de régularisation, les lissages au premier ordre (7) sont peu adaptés : ils pénalisent du même coup la *vorticité* et la *divergence* du mouvement, qui sont des quantités essentielles à la bonne représentation d'un écoulement. Des schémas d'ordre supérieurs, adaptés à ces quantités, ont ainsi été proposés [2, 11, 38, 45]. D'autres approches intègrent directement des contraintes issues de la physique des fluides et de la modélisation de turbulence. Il est ainsi possible d'imposer une divergence nulle [2, 45], ou de reproduire des caractéristiques spectrales de la turbulence [18].

Problématique

Au vu des développements précédents, les challenges posés dans le cadre de ces travaux sont les suivants :

- Proposer une méthode dense, accédant aux fines échelles du mouvement.
- Intégrer un formalisme multiéchelle, pour éviter dans la mesure du possible d'utiliser des approches pyramidales (comme avec l'OFC), et pour tenir compte de la nature fortement multiéchelle des mouvements de fluides turbulents.
- Offrir des schémas de régularisations généraux ou adaptés aux fluides, et simples à implémenter.
- Maintenir une complexité de calcul satisfaisante.

L'approche proposée se base sur une représentation en ondelettes du champ de vitesse estimé. Les ondelettes constituent en effet des bases naturellement multiéchelle, qui rappellent favorablement la structure multiéchelle des écoulements. Ces outils ont ainsi été utilisés à des fins d'analyse [29], de simulation [13, 22] ou de modélisation [15] d'écoulements turbulents. Dans le cadre d'estimation de mouvement, les ondelettes ont été également remarquées pour leurs propriétés multiéchelle [5, 44], tout comme pour l'élaboration de schémas de régularisation [10]. Enfin, l'existence de transformées rapides permet une implémentation efficace de l'algorithme envisagé.

Ondelettes – quelques concepts

Les ondelettes permettent de mettre en place un cadre d'*analyses multirésolutions* (MRA) de $\mathbf{L}^2(\mathbb{R})$. Conceptuellement, il s'agit d'*espaces d'approximation imbriqués* notés V_j, d'échelle j décroissante [1], vérifiant notamment :

$$\forall j \in \mathbb{Z}, \quad V_{j+1} \subset V_j \,; \tag{8}$$

$$\forall (j,k) \in \mathbb{Z}^2, \quad f(t) \in V_j \Leftrightarrow f(t - 2^j k) \in V_j \,; \tag{9}$$

$$\forall j \in \mathbb{Z}, \quad f(2t) \in V_j \Leftrightarrow f(t) \in V_{j+1} \,; \tag{10}$$

$$\lim_{j \to +\infty} V_j = \bigcap_{j=-\infty}^{+\infty} V_j = \{0\} \,; \tag{11}$$

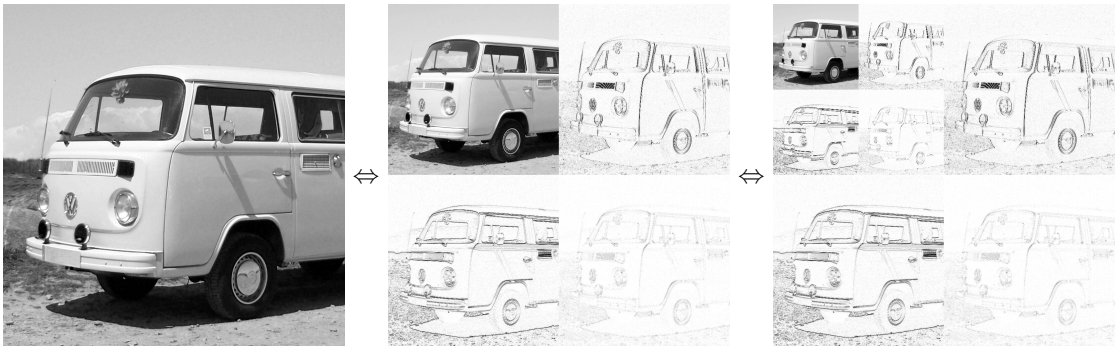
$$\lim_{j \to -\infty} V_j = \text{Clôture}\left(\bigcup_{j=-\infty}^{+\infty} V_j \right) = \mathbf{L}^2(\mathbb{R}) \tag{12}$$

La propriété (8) traduit l'imbrication des espaces V_j, tandis que (10) permet de calculer une approximation plus grossière dans V_{j+1} à partir de celle de V_j. La MRA de $\mathbf{L}^2(\mathbb{R})$ est formée par l'ensemble $\{V_j\}_{j \in \mathbb{Z}}$ [28].

Les espaces d'approximation V_j étant imbriqués, il peuvent être décomposés selon :

$$V_j = V_{j+1} \oplus W_{j+1} \,. \tag{13}$$

Ces W_j sont les compléments des espaces d'approximation, ils sont appelés *espaces de détails*. Le concept de décomposition en ondelettes réside dans cette association entre espaces d'approximation et de détails, combinée à la propriété d'imbrication : un signal donné est récursivement séparé en une approximation et les détails qui lui sont associés, de sorte qu'au terme du procédé ne restent qu'une approximation grossière et un ensemble de détails à plusieurs échelles. Le signal peut être reconstruit par l'opération inverse, qui réassocie à une approximation ses détails pour récupérer une approximation plus fine, etc. Ces transformations sont illustrées Figure 2.



(a) (b) (c)

FIGURE 2 – Exemple d'une transformée en ondelettes 2D (ici dyadique et isotrope). L'image en entrée (a) est transformée en (b) : une approximation grossière (*coin supérieur gauche*) et des détails (*tout le reste*), et le processus peut être répété (c) et/ou inversé pour récupérer (a). Les détails nuls apparaissent en blanc.

Bases, coefficients et transformées rapides

Selon la manière dont sont définis les W_j, on obtient des bases *orthogonales* ou *biorthogonales* pour les différents espaces d'approximation et de détails. Les fonctions de ces bases sont construites

1. Voir Remarque 1 page x sur la notation des échelles.

par des dilatations et des translations [2] de la *fonction d'échelle* et de l'*ondelette* "mères", respectivement notées φ et ψ. La décomposition d'un signal $f \in \mathbf{L}^2(\mathbb{R})$ jusqu'à une approximation grossière d'échelle $C \in \mathbb{Z}$ s'écrit par exemple :

$$f(x) = \sum_{k\in\mathbb{Z}} a_{C,k}\varphi_{C,k}(x) + \sum_{j\leq C}\sum_{k\in\mathbb{Z}} d_{j,k}\psi_{j,k}(x)\,, \tag{14}$$

où les $\{a_{C,k}\}_k$ et $\{d_{j,k}\}_{j,k}$ sont les *coefficients d'approximation et de détails*, respectivement, et sont donnés dans le cas orthogonal par les produits scalaires :

$$a_{C,k} = \langle f\,;\,\varphi_{C,k}\rangle_{\mathbf{L}^2} \quad \text{and} \quad d_{j,k} = \langle f\,;\,\psi_{j,k}\rangle_{\mathbf{L}^2}\,.$$

On retrouve bien dans (14) une approximation grossière à l'échelle C et une somme de détails à des échelles plus fines $j \leq C$. En pratique, les signaux considérés sont discrets et finis. Dans ce cadre, on construit des bases de $\mathbf{L}^2([0,1])$ au lieu de $\mathbf{L}^2(\mathbb{R})$; l'approche la plus simple consiste à périodiser les diverses fonctions de la base originale en les enroulant sur $[0,1]$. L'échelle accessible la plus grossière est toujours $C = 0$, tandis que la plus fine correspond à la discrétisation initiale du signal ($F < 0$ pour un signal de 2^{-F} échantillons).

Enfin, grâce à la propriété d'imbrication (8), les fonctions d'échelles et les ondelettes vérifient les relations *d'affinage*, ici données dans le cas orthogonal :

$$\begin{aligned}
\varphi_{1,0}(x) = \frac{1}{\sqrt{2}}\varphi(\frac{x}{2}) &= \sum_{n=-\infty}^{+\infty} h[n]\varphi_{0,n}(x)\,, \qquad \text{avec } h[n] = \langle\varphi_{1,0}\,;\,\varphi_{0,n}\rangle\ ;\\
\psi_{1,0}(x) = \frac{1}{\sqrt{2}}\psi(\frac{x}{2}) &= \sum_{n=-\infty}^{+\infty} g[n]\varphi_{0,n}(x)\,, \qquad \text{avec } g[n] = \langle\psi_{1,0}\,;\,\varphi_{0,n}\rangle\ ;
\end{aligned} \tag{15}$$

Les séquences h et g sont appelées *filtres miroir conjugués* ; ces filtres permettent la mise en oeuvre de la transformé en ondelettes rapide (FWT). Les coefficients d'approximation et de détails sont ainsi calculés récursivement, à travers des *bancs de filtres*, par des opérations de décimation/expansion et de convolution avec h et g :

$$\begin{aligned}
a_{j+1,p} &= \sum_{n=-\infty}^{+\infty} h[n-2p]a_{j,n}\,;\\
d_{j+1,p} &= \sum_{n=-\infty}^{+\infty} g[n-2p]a_{j,n}\,;\\
a_{j,p} &= \sum_{k=-\infty}^{+\infty} h[p-2n]a_{j+1,n} + \sum_{k=-\infty}^{+\infty} g[p-2n]d_{j+1,n}
\end{aligned} \tag{16}$$

Les algorithmes de décomposition/reconstruction ainsi constitués ont une complexité linéaire du nombre d'échantillons signal considéré. Tous ces concepts sont ensuite étendus sans difficulté au cas 2D, pour former des bases *séparables* de $\mathbf{L}^2(\mathbb{R}^2)$ et $\mathbf{L}^2([0,1]^2)$. Selon la manière dont on réalise la décomposition, on aboutit en outre à une transformée *isotrope* (cas présenté Figure 2) ou *anisotrope*.

Applications particulières

Bases à divergence nulles

La décomposition de Helmholtz sépare un champ de vitesse suffisamment régulier en ses composantes à *vorticité nulle* (irrotationelle) et à *divergence nulle* (solénoïdale) :

$$\begin{aligned}
\boldsymbol{v} = \begin{pmatrix} v_1 \\ v_2 \end{pmatrix} \in \mathbf{L}^2(\mathbb{R}^2)\times\mathbf{L}^2(\mathbb{R}^2) \Rightarrow\ \boldsymbol{v} &= \boldsymbol{v}^{\mathrm{curl}} + \boldsymbol{v}^{\mathrm{div}}\,;\\
\text{avec } \boldsymbol{v}^{\mathrm{curl}} \in \mathcal{H}^{\mathrm{curl}}(\mathbb{R}^2) &= \left\{\boldsymbol{v}\in(\mathbf{L}^2(\mathbb{R}^2))^2\,,\ \mathrm{curl}(\boldsymbol{v}) = \partial_{x_1}v_2 - \partial_{x_2}v_1 = 0\right\}\,,\\
\text{et } \boldsymbol{v}^{\mathrm{div}} \in \mathcal{H}^{\mathrm{div}}(\mathbb{R}^2) &= \left\{\boldsymbol{v}\in(\mathbf{L}^2(\mathbb{R}^2))^2\,,\ \mathrm{div}(\boldsymbol{v}) = \partial_{x_1}v_1 + \partial_{x_2}v_2 = 0\right\}\,.
\end{aligned} \tag{17}$$

2. Voir Remarque 2 page x.

La composante irrotationelle $\boldsymbol{v}^{\text{curl}}$ correspond à la présence de puits ou de sources dans l'écoulement, tandis que la partie solénoïdale $\boldsymbol{v}^{\text{div}}$ est liée aux structures tourbillonnaires. Dans les équations de Navier-Stokes pour un fluide *incompressible*, la densité constante implique un champ à divergence nulle. Dès lors, il peut être intéressant de s'assurer que le champ de vitesse estimé par le flot optique vérifie cette contrainte de divergence nulle. Une possibilité consiste à construire des bases de $\mathcal{H}^{\text{div}}(\mathbb{R}^2)$ pour le problème d'estimation, afin de reconstruire explicitement des mouvements à divergence nulle.

A cet effet, on considère la *fonction courant*, notée ici $z \in H^1(\mathbb{R}^2)$, tel que $\boldsymbol{v} = \mathbf{curl}(z) \in \mathcal{H}^{\text{div}}(\mathbb{R}^2)$. On construit ensuite une MRA et une base d'ondelettes pour la fonction courant (dans $H^1(\mathbb{R}^2)$), de telle sorte que le **curl** [3] de ces ondelettes produise une nouvelle MRA et une base d'ondelettes biorthogonales à divergence nulle (dans $\mathcal{H}^{\text{div}}(\mathbb{R}^2)$). Cette construction, non triviale, fait intervenir des résultats sur les dérivées et primitives d'ondelettes et fonctions d'échelles [22, 25]. A terme, on obtient des relations directes entre les coefficients représentant la fonction courant z, notés $d^{\text{div}}_{\boldsymbol{j},\boldsymbol{k}}$, et ceux notés $d^{i}_{\boldsymbol{j},\boldsymbol{k}}$ représentant chaque composante scalaire v_i, $i = 1, 2$ du champ à divergence nulle $\boldsymbol{v} = \mathbf{curl}(z)$:

$$d^{1}_{\boldsymbol{j},\boldsymbol{k}} = 2^{j_2+2} d^{\text{div}}_{\boldsymbol{j},\boldsymbol{k}}\,, \tag{18}$$

$$d^{2}_{\boldsymbol{j},\boldsymbol{k}} = -2^{j_1+2} d^{\text{div}}_{\boldsymbol{j},\boldsymbol{k}}\,, \tag{19}$$

$$d^{\text{div}}_{\boldsymbol{j},\boldsymbol{k}} = \frac{1}{2^{j_1+2} + 2^{j_2+2}} (d^{1}_{\boldsymbol{j},\boldsymbol{k}} - d^{2}_{\boldsymbol{j},\boldsymbol{k}})\,. \tag{20}$$

Il est à noter que l'implémentation pratique (signaux discrets et finis) de telles bases comporte quelques difficultés supplémentaires qui doivent être abordées convenablement.

Equivalence de norme et coefficients de connexion

Les propriétés de dérivation d'une ondelette permettent de construire une équivalence entre, d'une part la norme $\mathbf{L}^2$ de la n-ième dérivée d'une fonction donnée, d'autre part une pondération de la norme $\mathbf{l}^2$ des coefficients représentant cette fonction dans une base d'ondelettes appropriée [24]. Cette équivalence peut s'écrire, dans le cas 1D, comme :

$$\left\| \frac{d^n f}{dx^n} \right\|_{\mathbf{L}^2} \sim \left\| (-4)^n 2^{nj} d_{j,k} \right\|_{\mathbf{l}^2}\,, \tag{21}$$

où les $d_{j,k}$ sont les coefficients d'ondelettes représentant f. Cette relation permettra plus loin d'élaborer de simples schémas de régularisation.

D'autres schémas de régularisation plus généraux, moins simplistes, peuvent être construits en utilisant les *coefficients de connexion* de la base d'ondelettes considérée. Ces coefficients correspondent à des termes de forme

$$\left\langle \varphi_{i,p}\,;\, \varphi^{(n)}_{j,k} \right\rangle_{\mathbf{L}^2} = \int_{\mathbb{R}} \varphi_{i,p}(x) \frac{d^n \varphi_{j,k}}{dx^n}(x) dx\,. \tag{22}$$

Les relations d'affinage (15) permettent de calculer simplement les coefficients de connexion d'une base d'ondelettes donnée [7], par le biais d'un problème de vecteurs propres et d'une transformée en ondelettes.

Méthodes Proposées

Les ondelettes ont déjà été utilisées dans un contexte d'estimation de mouvement. Bernard [5] projette l'OFC (5) sur une base multiéchelle, pour construire un schéma d'estimation incrémental. Il n'introduit pas de terme explicite de régularisation, mais utilise à la place l'hypothèse d'un déplacement constant sur le support des fonctions de la base, et obtient finalement un ensemble de petits problèmes linéaires à inverser. L'approche de Wu & Kanade [44] est similaire à celle retenue ici : seul le champ de vitesse est projeté sur une base multiéchelle. Les coefficients sont estimés en

3. Le curl d'un champ scalaire $z(\boldsymbol{x})$ est donné par $(\partial_{x_2} z, -\partial_{x_1} z)^{\boldsymbol{T}}$ – voir Notations page ix.

minimisant l'erreur incrémentale quadratique de la DFD (6), en procédant séquentiellement des échelles les plus grossières vers les plus fines. A chaque nouvelle échelle de détails, les coefficients précédemment estimés sont utilisés comme point de départ de la minimisation. L'ondelette de Cai-Wang [8] qu'ils utilisent leur permet de calculer directement les coefficients à une échelle donnée, là où les ondelettes plus classiques demandent de calculer d'abord tous les coefficients aux échelles plus fines par les bancs de filtres (16). Un inconvénient de leur approche est le temps de calcul : il est nécessaire de recalculer la matrice hessienne du problème à chaque nouvelle échelle, puis la méthode de minimisation (Levenberg-Marquard) demande d'inverser un système linéaire de très grande taille, ce qui devient rapidement prohibitif à mesure que l'on considère des échelles de plus en plus fines ou des images de grande taille. Enfin, l'absence de mécanisme de régularisation complique l'estimation aux plus fines échelles, ou lorsque les images sont peu texturées. Enfin, Chen *et al.* [10] implémentent un estimateur de Horn & Schunck [20]. Chacun des termes de l'OFC au carré (e.g. $[\partial_{x_1} I_1]^2$), tout comme le champ de vitesse, est projeté sur une base d'approximation (non multiéchelle), et le terme de régularisation au premier ordre (7) est évalué à partir des coefficients de connexion (22). Un système de très grande dimension est finalement assemblé, puis inversé. Bien que précise, cette approche est également couteuse. En outre, elle n'intègre pas d'aspect multiéchelle, ce qui peut compliquer l'estimation de grands déplacements.

Notre approche reprend certains de ces concepts : le champ uniquement est projeté sur une base multiéchelle, et ses coefficients sont estimés successivement en partant des échelles grossières jusqu'aux plus fines. Diverses régularisations sont proposées, en tronquant la base, ou s'appuyant sur l'équivalence de norme (21) ou les coefficients de connexion (22). La méthode de minimisation L-BFGS permet finalement de conserver une complexité satisfaisante.

Représentation du champ de vitesse

Chaque composante scalaire v_i du champ $\boldsymbol{v}$ recherché est projetée sur une base d'ondelettes multiéchelle. On note Θ_i l'ensemble des coefficients (approximation et détails) représentant v_i, et $\boldsymbol{\Theta}$ est l'ensemble de tous les coefficients représentant $\boldsymbol{v}$:

$$\boldsymbol{\Theta} = \begin{pmatrix} \Theta_1 \\ \Theta_2 \end{pmatrix} . \tag{23}$$

On écrit alors :

$$\begin{aligned} \forall \boldsymbol{x} \in \Omega,\, v_i(\boldsymbol{x}) = \Phi^{\boldsymbol{T}}(\boldsymbol{x})\Theta_i\,, \quad i = 1, 2\,, \\ \boldsymbol{v}(\boldsymbol{x}) = \boldsymbol{\Phi}^{\boldsymbol{T}}(\boldsymbol{x})\boldsymbol{\Theta}\,. \end{aligned} \tag{24}$$

où $\Phi(\boldsymbol{x})$est un vecteur contenant les valeurs des fonctions de base en $\boldsymbol{x}$, et $\boldsymbol{\Phi}$ s'écrit :

$$\boldsymbol{\Phi}^{\boldsymbol{T}}(\boldsymbol{x}) = \begin{pmatrix} \Phi^{\boldsymbol{T}}(\boldsymbol{x}) & 0 \cdots 0 \\ 0 \cdots 0 & \Phi^{\boldsymbol{T}}(\boldsymbol{x}) \end{pmatrix} . \tag{25}$$

En pratique, toutes les transformées en ondelettes directes (décomposition) ou inverses (reconstruction) sont réalisées par les bancs de filtres. Les inconnues du problème d'estimation sont à présent les coefficients $\boldsymbol{\Theta}$, et le problème original (4) devient :

$$\begin{cases} \hat{\boldsymbol{v}}(\boldsymbol{x}) = \boldsymbol{\Phi}^{\boldsymbol{T}}(\boldsymbol{x})\hat{\boldsymbol{\Theta}}\,,\ \forall \boldsymbol{x} \in \Omega \\ \hat{\boldsymbol{\Theta}} = \underset{\boldsymbol{\Theta}}{\arg\min}\, J_{\text{data}}(I, \boldsymbol{\Theta}) + \alpha J_{\text{reg}}(\boldsymbol{\Theta})\,. \end{cases} \tag{26}$$

Modèle de données

Les modèles de données sont simplement obtenus en replaçant $\boldsymbol{v}$ par $\boldsymbol{\Phi}^{\boldsymbol{T}}\boldsymbol{\Theta}$. Le terme de données pour la DFD devient par exemple :

$$J_{\text{data}}(I, \boldsymbol{\Theta}) = \frac{1}{2} \int_{\Omega} \left[I_0(\boldsymbol{x}) - I_1(\boldsymbol{x} + \boldsymbol{\Phi}^{\boldsymbol{T}}(\boldsymbol{x})\boldsymbol{\Theta}) \right]^2 d\boldsymbol{x}. \tag{27}$$

L'évaluation de son gradient par rapport à $\boldsymbol{\Theta}$ est nécessaire à la minimisation du problème. Il est très rapide de montrer que pour un coefficient $\theta_{i,p} \in \Theta_i$ quelconque, on a

$$\frac{\partial J_{\text{data}}(\boldsymbol{\Theta})}{\partial \theta_{i,p}} = \left\langle \left[I_0(\cdot) - I_1(\cdot + \boldsymbol{\Phi}^{\boldsymbol{T}}(\cdot)\boldsymbol{\Theta})\right] \frac{\partial I_1(\cdot + \boldsymbol{\Phi}(\cdot)^{\boldsymbol{T}}\boldsymbol{\Theta})}{\partial x_i} \,;\, \phi_p \right\rangle_{\mathbf{L}^2([0,1]^2)} , \tag{28}$$

où ϕ_p est l'atome de la base correspondant à $\theta_{i,p}$. Ainsi, les composantes du gradient sont simplement donnés par les coefficients résultant de la transformée en ondelettes, sur la base considérée, des deux termes :

$$\left[I_0(\cdot) - I_1(\cdot + \boldsymbol{\Phi}^{\boldsymbol{T}}(\cdot)\boldsymbol{\Theta})\right] \frac{\partial I_1(\cdot + \boldsymbol{\Phi}(\cdot)^{\boldsymbol{T}}\boldsymbol{\Theta})}{\partial x_i} \quad \text{pour } i = 1, 2\,. \tag{29}$$

L'utilisation de l'OFC (5) au lieu de la DFD, ou d'une base à divergence nulle (17) au lieu des bases usuelles, ne pose pas de difficulté particulière.

Tout comme la méthode de Wu & Kanade [44], notre approche estime séquentiellement les coefficients, échelle par échelle, de la plus grossière à la plus fine. A une échelle donnée, les coefficients précédemment estimés font toujours partie des inconnues, de sorte qu'ils continuent à être mis à jour cependant que les nouveaux coefficients de détails sont estimés. Le mouvement est ainsi recherché et mis à jour dans des espaces imbriqués de résolution de plus en plus fine, jusqu'à atteindre l'échelle fine F souhaitée. Pour limiter le coût en calcul malgré le grand nombre de variables, nous nous appuyons sur l'algorithme de minimisation l-BFGS [32] qui approxime la hessienne et ne nécessite pas de stocker la matrice complète.

Régularisations

Une première approche consiste à réduire le nombre d'inconnues en tronquant la base considérée aux petites échelles. Typiquement, les coefficients du mouvement correspondant aux deux ou trois espaces de détails les plus fins ne sont pas estimées. On retrouve alors une formulation de type paramétrique (3), et ici, la solution est une approximation polynomiale par morceaux, dont le degré dépend de la régularité (du nombre de *moments nuls*) des fonctions de la base considérée. Cette approche a le mérite d'être extrêmement simple, mais ne permet pas d'estimer les plus fines échelles du mouvement. En outre, le choix de l'échelle de coupure n'est pas forcément évident. Trop grossière, et le mouvement estimé risque de manquer d'énergie, Trop fine, et les incertitudes du problème d'ouverture engendrent une solution fortement irrégulière, en l'absence de terme explicite de régularisation.

Une seconde alternative, appelée "approximation discrète", repose sur l'équivalence de norme (21). Elle permet d'approcher des termes de la forme

$$J_{\text{reg}}(\boldsymbol{v}) = \frac{1}{2} \sum_{i,p=1,2} \left\| \frac{\partial^n v_i}{\partial x_p^n} \right\|_{\mathbf{L}^2}^2 \tag{30}$$

directement par des pondérations des coefficients $\boldsymbol{\Theta}$ du champ de vitesse $\boldsymbol{v}$. Dans le cadre d'une transformée isotrope, on obtient ainsi :

$$J_{\text{reg}}(\boldsymbol{v}) \sim J_{\text{reg}}(\boldsymbol{\Theta}) = \frac{1}{2} \sum_{i=1,2} \sum_{j,\boldsymbol{k}} (4^{nj}) \left| d^i_{j,\boldsymbol{k}} \right|^2 . \tag{31}$$

L'évaluation de ces termes est peu couteuse, puisqu'elle ne demande que des multiplications point à point de matrices et des sommes.

Une troisième option, enfin, utilise les coefficients de connexion (22). Cette approche, appelée "approximation continue", permet l'élaboration de schémas beaucoup plus évolués tout en assurant une évaluation plus précise. Les schémas obtenus s'écrivent comme une somme de termes de la forme :

$$\cdots \pm \Theta_i : \left(N^{(n_1)} \Theta_j N^{(n_2)\boldsymbol{T}} \right) \pm \cdots \tag{32}$$

où Θ_i est la matrice des coefficients représentant v_i, et $N^{(n_1)}$, $N^{(n_2)}$ sont deux matrices obtenues à partir des coefficients de connexion, matrices qui ne dépendent que de la base choisie et de l'ordre

de dérivation et peuvent donc être précalculées. A titre d'exemple, le schéma de régularisation du gradient de la vorticité s'écrit :

$$\begin{aligned} J_{\text{reg}}(\boldsymbol{\Theta}) =& \frac{1}{2}\int_{\Omega} |\nabla \operatorname{curl}(\boldsymbol{v})(\boldsymbol{x})|^2 \, d\boldsymbol{x} \\ =& \frac{1}{2}\Theta_1 : \left(N^{(0)}\Theta_1 N^{(4)} + N^{(2)}\Theta_1 N^{(2)}\right) \\ &+ \frac{1}{2}\Theta_2 : \left(N^{(4)}\Theta_2 N^{(0)} + N^{(2)}\Theta_2 N^{(2)}\right) \\ &- \Theta_1 : \left(N^{(3)}\Theta_2 N^{(1)\boldsymbol{T}}\right) - \Theta_2 : \left(N^{(1)}\Theta_1 N^{(3)\boldsymbol{T}}\right) . \end{aligned} \tag{33}$$

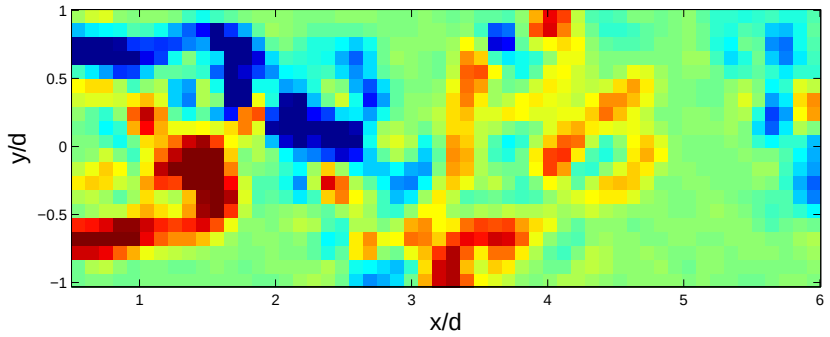
Cette approche est plus couteuse, puisque chaque terme de type (32) demande trois multiplications matricielles.

Résultats

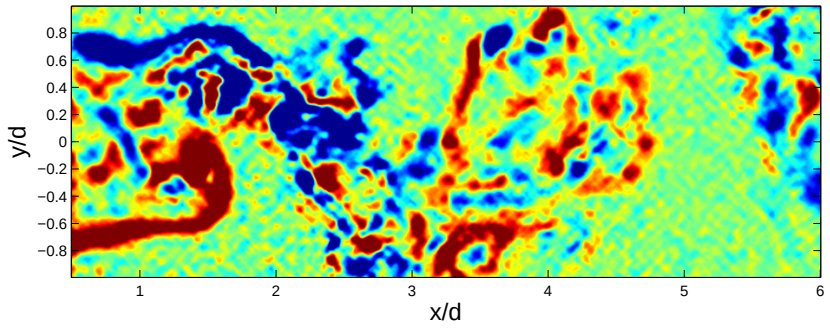
Les estimateurs proposés sont dans un premier temps testés et validés sur deux séquences *synthétiques*, pour lesquelles le déplacement effectif est connu, permettant ainsi des mesures d'erreur. Deux types d'images, habituellement employés dans le cadre de visualisation d'écoulements, sont ici considérés : particules et advection/diffusion d'un scalaire. Les influences des divers paramètres (nombre d'échelles, type et nombre de moments nuls de l'ondelette, régularisation...) et le domaine de fonctionnement optimal sont mis en avant à travers plusieurs séries d'expériences. Il est ainsi possible de conclure favorablement à l'intérêt d'une base multiéchelle, ainsi qu'à l'avantage net de la DFD sur l'OFC, même lorsque cette dernière est imbriquée dans un cadre incrémental. Si le mouvement sous-jacent est effectivement à divergence nulle, les bases tronquées à divergence nulle s'avèrent particulièrement efficaces, tout comme les bases usuelles avec pénalisation explicite de la divergence, en particulier si les images sont faiblement texturées. Sur les images de particules, les méthodes présentées dépassent l'état de l'art ; elles sont en revanche légèrement moins performantes sur les images de scalaire.

Dans un deuxième temps, les méthodes sont appliqués à des images réelles de PIV, correspondant à deux configurations classiques de mécanique des fluides expérimentale : le *sillage de cylindre* à Reynolds 3900, et la *couche de mélange* turbulente. Les champs estimés sont comparés à ceux fournis par les corrélations croisées (2) et des sources externes (littérature ou anémométrie), sur la base de quantités statistiques utilisées pour la description d'écoulements turbulents. Sur le cas du cylindre, les résultats donnés par les deux approches vision (corrélations croisées et flot optique) sont souvent très proches, avec toutefois une résolution supérieure pour les estimés de la méthode proposée. A titre d'exemple, une comparaison de la vorticité calculé à partir de deux estimés est présentée Figure 3 ; le gain apporté aux fines échelles y est clairement visible. L'expérience de couche de mélange permet de mettre en avant les limites posées par l'utilisation de bases périodiques, qui peuvent entrainer une "pollution" du champ estimé près des bords de du domaine. Elle semble également suggérer que l'utilisation d'images comportant de grands déplacements ($\sim$10 pixels), nécessaires au bon fonctionnement des corrélations croisées (contrairement au flot optique), peuvent détériorer la qualité des estimés, notamment pour les structures les plus rapides.

Les solutions proposées remplissent globalement les objectifs fixés. Il serait néanmoins intéressant d'améliorer la gestion des conditions de bords. Si les temps de calculs sont acceptables (d'une dizaine de seconde, pour des images de 256×256 px à quelques minutes pour des 1024×1024 px), il reste des efforts à fournir pour atteindre des objectifs de temps réel. Un gain important se trouve probablement du coté des convolutions des transformées en ondelettes, à travers une implémentation GPGPU. Les bases à divergence nulle peuvent sembler anecdotiques ici, dans la mesure où les mouvements apparents 2D d'un écoulement 3D ne sont pas à divergence nulle. En revanche, dans le cadre d'estimation de mouvements 3D, actuellement en développement, elles pourraient s'avérer un outil puissant si le coût en calcul reste acceptable. Enfin, un autre aspect important concerne les critères de qualité des estimés, en l'absence de vérité terrain. Le choix de ces critères et le développement des algorithmes en conséquence ne peut se faire sans l'appui des utilisateurs, spécialistes en fluides, expérimentateurs et techniciens.



(a) corrélations croisées



(b) méthode proposée

FIGURE 3 – Comparaison de la vorticité obtenue pour deux estimés différents, par corrélations croisées (*haut*) ou flot optique avec régularisation du second ordre (*bas*). Cas du sillage de cylindre.

Organisation du Document

Outre cette synthèse en français et une courte introduction, ce document est organisé en quatre grandes parties. La première, qui recouvre le contexte des travaux, développe tout d'abord les aspect liés à la mécanique des fluides et la visualisation d'écoulements (Chapitre 1), puis à la vision par ordinateur et l'estimation de mouvement (Chapitre 2). La seconde partie, plus mathématique, introduit les bases ondelettes, les transformées et leur propriétés (Chapitre 3), puis certaines applications parmi lesquelles les bases à divergence nulle (Chapitre 4). La troisième partie détaille les méthodes proposées, en s'attachant tout d'abord au terme de données (Chapitre 5), puis aux régularisations (Chapitre 6) et enfin à certains aspects de l'implémentation (Chapitre 7). La quatrième partie procède à l'évaluation des algorithmes, sur les cas synthétiques dans un premier temps (Chapitre 8) puis sur les images expérimentales réelles (Chapitre 9). Enfin, en annexe se trouvent les détails de l'implémentation pratique des bases à divergence nulle (Annexe A), et sur le coût en calcul des bancs de filtres (Annexe B). Une liste des figures est par ailleurs disponible page vii, ainsi qu'un récapitulatif des notations et abréviations employées page ix.

Introduction

General Problematic

From fluid-structure interactions in aircraft design to wind farm conception, from very short-term rain forecast to climate change simulation, the comprehension of fluid dynamics phenomenons influences our direct environment on a very wide range of scales. If some aspects of fluid dynamics are now well-described, many others, such as *turbulence*, are still only partially understood despite extensive studies.

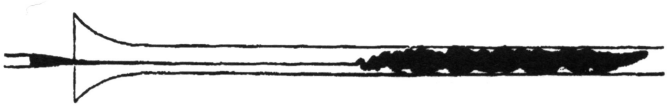


Figure 4: Observation of laminar-to-turbulence transition in a pipe flow, as drawn by Osborne Reynolds in his *An experimental investigation of the circumstances which determine whether the motion of water in parallel channels shall be direct or sinuous and of the law of resistance in parallel channels* paper, 1883 [37].

Tackling such a demanding subject requires adequate tools. Aside from direct observation, the only available option remained for a long time "probe-like" measurements, in the sense that it gives a temporal evolution, on a single spacial point, of a physical quantity. As such, it does not enable to recover an instantaneous, spatially dense measure – unless, of course, an array of probes is employed to record simultaneous measurements in different locations, which is not without adding issues. Yet the second half of the XXth century has seen a flourishing breakthrough in microelectronics and informatics that made available the ingredients for another category of measurement approaches. The development of CCD sensors gave rise to the digital photography, which now extends from our mobile phones to earth observation satellites. With the help of the ongoing increase of computing power, *computer vision* algorithms were soon designed to extract information from images, e.g. objects recognition, 3D reconstruction, or, more interesting to us, motion estimation. A wide family of methods, known as *optical flow*, indeed enables to recover the apparent motion in an image sequence. These methods rely on a so-called *data-model*, which links image data to the unknown apparent motion.

Adapting these computer vision approaches to the context of fluid motion estimation was another issue, due to the specificities of fluid flows. One has to cope with particular structures like eddies, complex 3D displacements over a wide range of spatial and temporal scales, ... Today, the most operational and widely used method is probably the Particle Image Velocimetry, where photos of a fluid flow seeded with particles are processed with a *cross-correlations* software. This method however has the drawback of returning estimated velocity vectors on a grid coarser than input image data. Dense methods (i.e. returning a dense motion field, contrary to the cross-correlations) are available, but suffer from other issues, e.g. their poor ability to deal with large displacements and their computational efficiency due to the very high number of unknowns involved. Moreover, adding extern priors is mandatory, as image data itself does not contain all the necessary information to recover the motion with certainty. This is known as the *aperture problem*, and is shared by other motion estimation approaches as well. The choice of this prior – the *regularizer* – is of high importance, as it largely influences the characteristics of the returned motion estimate.

We believe that dense estimation methods deserve to be developed and used. By giving access to

smaller structures otherwise invisible in cross-correlations estimates, they enable a more complete comprehension of flow dynamics. Moreover, their flexible framework allows the use of various data-models and regularizers that can be chosen accordingly to the specificities and physics of the investigated images and flow.

The Question

From previous remarks, our objective is to find a suitable representation for the motion, compatible with fluid flow specificities while being adapted to dense optical flow methods.

Regarding the fluid aspect, the critical points are the kinematic properties of the fluid – notably a wide range of motion scales and amplitudes – and the computational efficiency, from the very high number of variables involved. The mathematical tool known as *wavelets* may answer these two points. Wavelets are functions verifying a few specific properties, they can be used to design particular bases in which signals are represented. The (often) compact support of these functions leads to a fast implementation widely known as the Fast Wavelet Transform (FWT). Moreover, wavelet form intrinsically *multiscale* bases, which echoes to the multiscale nature of fluid dynamics. In this context, wavelets have been first used for turbulent flow analysis [29], then to build specific bases (e.g. divergence-free) for flow simulation purpose, while handling boundary conditions [13, 22]. Wavelet-based turbulence models have been proposed as well [15].

Within the motion estimation context, we need to design general regularizers to close the estimation problem. From the sensitivity of these methods to large displacements, a multiscale approach is also mandatory. Once again, wavelets seem to provide an appropriate answer. They have been noticed already for their multiscale properties [5, 44]. The computation of their connection coefficients enabled Chen *et al.* [10] to implement a first order (Horn & Schunk) regularizer.

The proposed work combines and extends these ideas: a wavelet representation of the motion field leading to a "natural" multiscale sequential estimation, fluid-dedicated regularizers built from wavelet properties, divergence-free bases to incorporate a physical constraint, while keeping in mind the computational efficiency of the overall algorithm.

Overlook on the Dissertation

This document is organized as follows. Part I explicits the context of this work, starting with fluid dynamics before addressing the computer vision aspects. Part II is more mathematical, as it introduces the wavelet framework and some of its applications later used. Next Part III presents the proposed wavelet-based optical flow approach and its implementation. Technical details, especially related to divergence-free bases implementation, are also available in appendices Part V. Resulting algorithms are finally evaluated on synthetic and real data in Part IV. Note that a table of figures is available after the table of content page vii, as well as a list of the main notations and abbreviations used in this document page ix.

Part I

Context

Chapter 1

Fluid Motion and Turbulence

1.1 Flow Visualization Methods

When I was taught surfing, a very important topic – even before any technical aspect – was the observation and analysis of our playground. Sat on the top of a dune, we looked at the tide, wave's amplitude and period of course; even more important for safety considerations were the wind and the surface currents. One option among many to determine wind direction is to grab a handful of sand, drop it slowly and watch it as it flies away. Regarding the currents, the easiest way is to look at the foam, or at other surfers as they are transported by the flow. These two situations are simple, yet legitimate examples of the use of *passive tracers* to infer the otherwise invisible flow motion – just like looking at fallen leaves, or clouds, passing by. Indeed, flow visualization techniques have been used for a long time to enlighten some characteristics, behaviors or structures of fluid flows. M. Van Dyke's *An Album of Fluid Motion* [43] is a remarkable compilation of black and white shots illustrating many different aspects of fluid dynamics and featuring several visualization techniques. Most of these visualization methods fall within the two families presented below.

Now, since we – as human beings – are able to make pertinent analyses from the observation of the aforementioned tracers, next stage consists in developing computer programs that mimic our eyes-brain faculties in order to process flow visualization pictures and extract the underlying motions, in a more systematic and efficient way than we could do. This step, obviously much more recent, is detailed in Chapter 2: "Motion Estimation and Inverse Problems". For now, let us present some visualization methods.

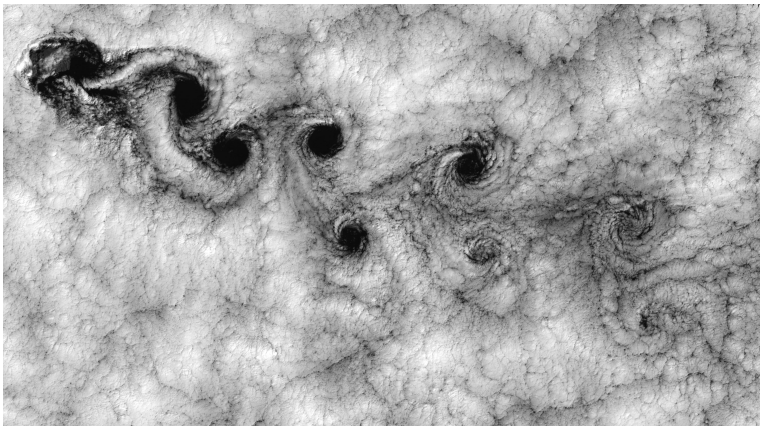


Figure 1.1: Clouds reveal the presence of a von Kármán vortex street, as the wind hits one of the Juan Fernandez Island (located *top-left*).

Passive Tracers

Methods using passive tracers might involve the addition of external objects – our tracers – to the flow: sand, foam and leaves in our introductory examples; more realistically smoke, oil, dye, small bubbles, ... are employed. Generically named *particles*, their *passive* nature means their characteristics (weight, shape, size, ...) do not perturb the flow to be studied in which they are added – flow is said to be "seeded". Flow behavior may then be deduced from the observation of tracer motions: Figure 1.2 shows an example of flow visualization using particles, and gives a quick interpretation of the visible structures. Figure 1.1 is another example exploiting clouds as tracers. The most common images, from laboratory experiments, rely on small particles of smoke, or oil or water droplets. Such images are usually processed using *particle image velocimetry* (PIV) algorithms. Other images feature the transport and diffusion of a passive scalar quantity, such as dye concentration, water vapor concentration or sea surface temperature... These images might be taken by satellite imagery devices, and do not necessarily correspond to the visible range of the electromagnetic spectrum. Pictures of sea surface temperature, for instance, are often taken at specific wavelengths in the infrared spectrum – an example is displayed Figure 1.3. Contrary to particle images, scalar-transport images often show almost uniform, low textured areas, which makes them more complicated to process. The reason is explained in Chapter 2.

Figure 1.2: This image results from the combination (average) of 500 successive PIV frames, revealing particle trajectories – this is somehow equivalent to taking a single long-exposure shot. The dark, crescent-shaped area on the top of the picture is a sand dune – a "barchan" – viewed from above. This dune is plunged into a water stream, going downward on the picture. Left and right, almost straight particle trajectories reveal this main stream. Below (more exactly, behind) the dune, particle paths are more complex: this is the recirculation, a lower velocity region due to the shield created by the dune.

Optical Methods

In a given fluid, density gradients caused by flow motion result in local variations in the fluid refractive index. Optical methods exploit light ray distortions, caused by these local variations, to visualize structures in the flow: lighter regions correspond to positive density gradients, darker regions to negative gradients. The two main methods are the *shadowgraph* and the *schlieren photography* (Figure 1.4).

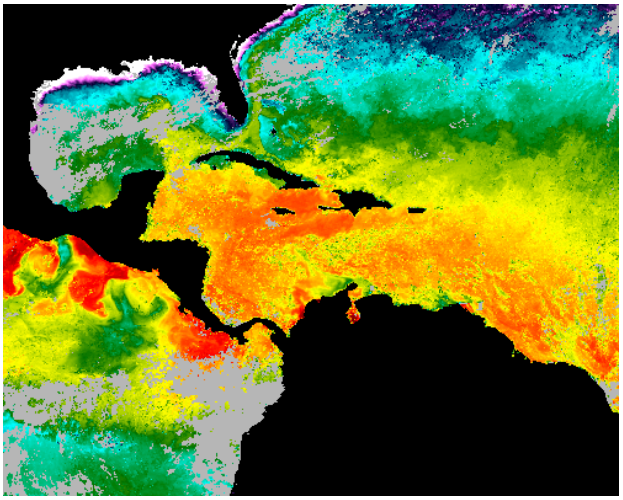


Figure 1.3: An example of sea surface temperature (SST) image, in the Gulf of Mexico region. Cold waters are in purple-blue, warm waters in orange-red. This image is computed thanks to AVHRR instrument of satellite Metop, mixing measurements from four different channels (one in the visible range, three in the infrared range). Clouds unfortunately block the wavelengths at which photographs are taken, resulting in grey areas with no data available. SST can be considered as a passive scalar; its temporal evolution reveals the underlying currents and surface motions.

1.2 Basic Fluid Dynamics

Fluid dynamics is modeled by the *Navier-Stokes equations*, presented here for an *incompressible* newtonian fluid. In this simplified form, the system has four unknowns – three components of the velocity field $\boldsymbol{v}$, and a scalar pressure field p:

$$\boldsymbol{v}(\boldsymbol{x},t) = \begin{pmatrix} v_1(\boldsymbol{x},t) \\ v_2(\boldsymbol{x},t) \\ v_3(\boldsymbol{x},t) \end{pmatrix} \in \mathbb{R}^3 \quad \text{and} \quad p(\boldsymbol{x},t) \in \mathbb{R}, \quad \boldsymbol{x} \in \mathbb{R}^3 \,. \tag{1.1}$$

The incompressible Navier-Stokes equations write:

$$\operatorname{div}(\boldsymbol{v}) = \nabla \cdot \boldsymbol{v} = 0 \,; \tag{1.2}$$

$$\rho \left(\frac{\partial \boldsymbol{v}}{\partial t} + \boldsymbol{v} \cdot \nabla \boldsymbol{v} \right) = -\nabla p + \mu \nabla^2 \boldsymbol{v} + \boldsymbol{f} \,. \tag{1.3}$$

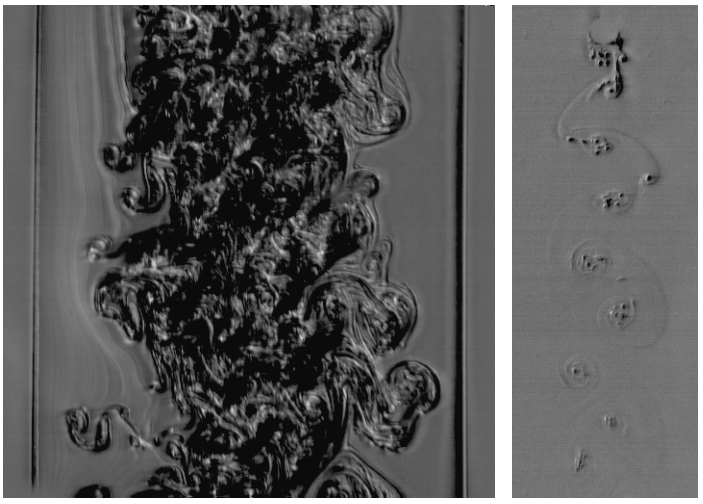
Equation (1.2) specifies a *volume conservation*; it is derived from the mass conservation principle along with the incompressibility hypothesis. Velocity field $\boldsymbol{v}$ is here divergence-free, which means there are neither sources nor sinks in the flow. Equation (1.3) is the *momentum conservation*, where ρ is the constant density (incompressible fluid), p is the pressure, μ the fluid cinematic viscosity and $\boldsymbol{f}$ the external forces.

Since (1.3) is a vector equation, the full incompressible Navier-Stokes system is made of four partial differential equations. From convection term $\boldsymbol{v} \cdot \nabla \boldsymbol{v}$, momentum conservation equations are non-linear in $\boldsymbol{v}$. Taking the divergence of 1.3 together with the divergence-free constraint, we have:

$$\Delta p = -\operatorname{div}(\boldsymbol{v} \cdot \nabla \boldsymbol{v} - f) \,.$$

Pressure is hence solution of a Poisson equation, which has a redistribution effect on the whole resolution domain. This non-local characteristic, associated to the non-linear advection term, makes the specificity of the Navier-Stokes equation. There is generally no analytical solution to the system, save for a very few specific cases; even the numerical resolution is known to be challenging.

When the convection term $\boldsymbol{v} \cdot \nabla \boldsymbol{v}$ becomes prominent, non-linearities have other consequences. First, the system is much more sensitive to its initial and boundary conditions: it impacts considerably the accuracy of predictions of the system evolution from a given state, unless this state is



(a) (b)

Figure 1.4: Two examples of visualization by schlieren photography of a bi-dimensional flow on the surface of a soap film. Figure 1.4a shows vortices of comb-generated turbulence. Figure 1.4b is another manifestation of the von Kármán instability previously shown in Figure 1.1.

perfectly known – a major issue in meteorology, for instance! Second, the flow may develop chaotic behaviors. This specific regime, known as *turbulence*, is more precisely studied hereafter.

1.3 Turbulence

Turbulence characterizes the state of a fluid flow. Arising from non-linearities in the momentum conservation equation (1.3), turbulence is described mathematically by chaotic fluctuations of flow variables $\boldsymbol{v}$, p, as well as a higher sensitivity to initial and boundary conditions. Physically, turbulent flows exhibit complex and highly tri-dimensional motions over a wide range of scales and amplitudes, by contrast with laminar flows which have slower and more regular behaviors.

Giving a robust description of turbulence phenomenology is challenging and out of the scope of this dissertation; this section rather aims at giving intuitions on the most important aspects of turbulence physics before introducing a few models. Readers looking for an extensive yet pedagogic presentation might refer to Chassaing [9] (in French) or Bernard [6] (in English). Classical books include e.g. Yaglom & Monin [31].

1.3.1 Easy Phenomenology

An immediate, visual criterion characterizing turbulent flows might be their numerous vortical structures, giving turbulent flow a rather "irregular" look – see Figure 1.4a. In other words, turbulent flows have a non-zero *vorticity*, where the vorticity is defined as the curl of the velocity field:

$$\boldsymbol{\omega} = \mathbf{curl}(\boldsymbol{v}) = \nabla \times \boldsymbol{v}\,. \tag{1.4}$$

These vortices and eddies, which appear over a wide range of scales, are responsible for the energy transfer mechanism inside the flow. Biggest structures are influenced by flow configuration (notably, the geometry), their size being about the same as the characteristic scale of the flow (e.g. the width of a canal, the diameter of a cylinder, ...). At this scale, viscous effects are negligible. Kinetic energy, initially given to these large-scale structures by the mean flow, is progressively transferred to smaller and smaller vortices through complex, non-linear interactions, unless these vortices reach a critical size known as the *Kolmogorov* scale. There, viscous dissipation finally balances inertial forces, so kinetic energy is transformed into heat by molecular dissipation. The process of energy transfer through scales is known as the *energy cascade* – Figure 1.5.

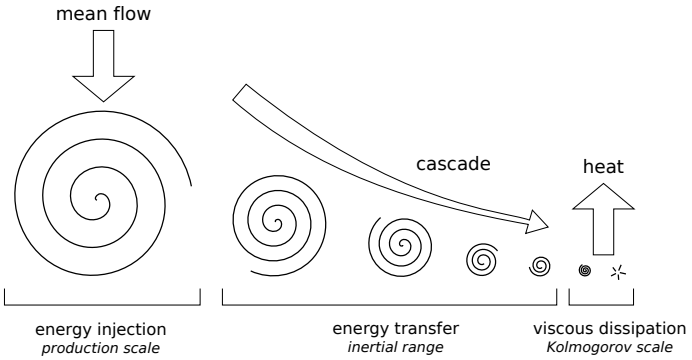


Figure 1.5: An illustration of the energy cascade occurring in turbulent flows.

Ratio between inertial and viscous forces in Navier-Stockes equations (1.3) is therefore an important point to describe turbulence mechanisms. This ratio may be quantified through dimensional analysis, for a given flow configuration, by the dimension-less *Reynolds number* noted Re:

$$Re = \frac{\rho \boldsymbol{v} L}{\mu}\,, \tag{1.5}$$

with ρ, μ the fluid density and the dinamic viscosity, and $\boldsymbol{v}$, L are characteristic velocity and scale of the flow, respectively. At "small" Re, viscous forces of the diffusion term dominates, flow is laminar. At "high" Reynolds, the convection term and its non-linearities control the behavior of the flow, which can be turbulent.

Turbulent flows are also characterized by the apparent loss of the symmetries present in the Navier-Stokes equation. These symmetries are however restored in the statistical point of view of isotropy and homogeneity. A statistic description is therefore preferred to the usual deterministic point of view. Studied quantities include for instance the *velocity increment structure functions*:

$$\text{(transversal)} \quad S_t(s,p) = E\left[|(\boldsymbol{v}(\boldsymbol{x}+s\boldsymbol{l}) - \boldsymbol{v}(\boldsymbol{x}))\cdot \boldsymbol{t}|^p\right] = E\left[\delta v_t(\boldsymbol{x},s)^p\right]\,, \tag{1.6}$$

$$\text{(longitudinal)} \quad S_l(s,p) = E\left[|(\boldsymbol{v}(\boldsymbol{x}+s\boldsymbol{l}) - \boldsymbol{v}(\boldsymbol{x}))\cdot \boldsymbol{l}|^p\right] = E\left[\delta v_l(\boldsymbol{x},s)^p\right] \tag{1.7}$$

$$= E\left[\delta v_l(s)^p\right]\ \forall \boldsymbol{x}, l \text{ if homogeneous and isotropic} \tag{1.8}$$

where $E[\cdot]$ is the expectation, $\delta v_l(\boldsymbol{x},s)$ and $\delta v_t(\boldsymbol{x},s)$ the longitudinal and transversal velocity increments at scale s and point $\boldsymbol{x}$, respectively; $\boldsymbol{l}$, $\boldsymbol{t}$ are the longitudinal and transversal directions, and p is the order. The *skewness* and *flatness*, the third and fourth normalized moments of a random variable, are also considered. They are given by:

$$\text{(skewness)} \quad \gamma_1 = E\left[\left(\frac{X-\mu}{\sigma}\right)^3\right]\,, \tag{1.9}$$

$$\text{(flatness)} \quad \beta_2 = E\left[\left(\frac{X-\mu}{\sigma}\right)^4\right]\,, \tag{1.10}$$

with X the random variable, μ and σ its mean and variance.

To make a long story short, the ideas on turbulent motions to be kept for the following are:
- they present complex, irregular motions featuring vortical structures;
- energy is redistributed by a cascade-like mechanism over a wide range of scale;
- they require a statistical description.

Chapter 2

Motion Estimation and Inverse Problems

In the previous chapter, we established how the vision-based observation and analysis of fluid flow dynamics are made possible. The first step consists in setting up a visualization method to enlighten some structures or behaviors of the flow – using particles, smoke, shadows, etc. The observer is then able to make suppositions on the dynamics of the flow, using the same tools as in his everyday life – his eyes to capture the visual information, his brain to process it. The processing step, although very natural and unconscious, is actually extremely complex: involving shape and pattern recognition as well as distance and motion estimation, it also relies on some "prior" knowledge progressively collected from birth.

Computer vision is the field dedicated to the study and development of the automated counterpart to human vision. Similarly, it involves tasks such as data acquisition, object recognition, motion estimation, machine learning and so on, in order for the system to observe and analyze its surrounding environment. Applications include for instance navigation (autonomous drones, driverless cars, ...) detection (video surveillance, satellite-based monitoring of forest fires, floods or crops, ...), 3D reconstruction (medical imagery, cinema, ...) [17].

Getting back to fluid dynamics, our concern is here restricted to the motion estimation task only. Resulting motion fields can be later used for turbulence dynamics study by researchers, or injected into a data assimilation scheme in the context of meteorological prediction, etc. Let us note already that more complex processes have already been set up, featuring for instance recognition and tracking of specific structures of the flow – cyclones [34], convective cells [3] – or even flow control [40]. This chapter starts by presenting the general concepts behind motion estimation methods, then focuses on the fluid-dedicated approaches.

2.1 Motion Estimation Context

Optic flow aims at recovering the *apparent 2D displacement* of a 3D scene depicted by a sequence of images, typically obtained from a camera. The time and space variations of an observable image quantity, e.g. its brightness, are used to infer the underlying motion occurring in the image plane between two consecutive frames.

2.1.1 From Images to Motion: a First Intuition

In the following, we will denote by $\Omega \subset \mathbb{R}^2$ the image domain. Our observable image quantity is noted $I(\boldsymbol{x}, t)$ at pixel $\boldsymbol{x} \in \Omega$ and discrete time index t – see Figure 2.1. The apparent motion, as a 2D vector field $\boldsymbol{v}(\boldsymbol{x}, t) : \Omega \times \mathbb{N} \mapsto \mathbb{R}^2$, is the observable projection on the image plane of the actual 3D motion. As such, it does enable to recover any actual motion component normal to the image plane, unless a stereoscopic imaging system is employed.

In order to recover the apparent displacement, the first required ingredient is an equation linking the observable image quantity I to the underlying motion $\boldsymbol{v}$: the *data term*. Such equations are

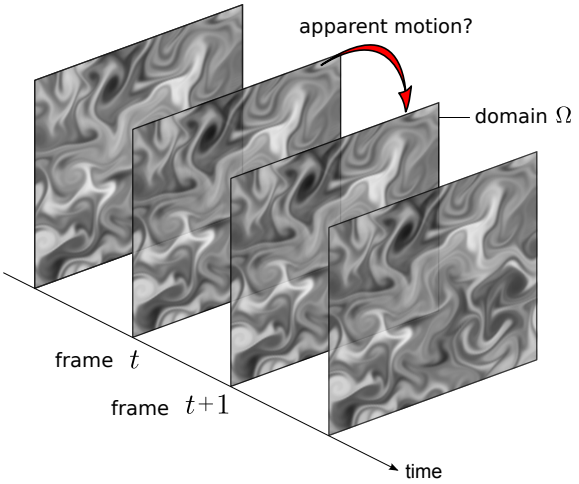


Figure 2.1: An example of image sequence.

usually built upon assumptions on the behavior of photometric invariants. The most simple and widely used models rely on the assumption of the *brightness consistency* along an image point trajectory, thus involving the transport equation:

$$\frac{dI}{dt}(\boldsymbol{x},t) = \frac{\partial I}{\partial t}(\boldsymbol{x},t) + \boldsymbol{v}(\boldsymbol{x},t)\cdot\nabla I(\boldsymbol{x},t) = 0. \tag{2.1}$$

However this assumption is not necessarily valid in the presence of illumination changes from one frame to the next, occlusions, or motions components normal to the image plane.

2.1.2 The Aperture Problem

Although being fairly intuitive, consistency assumption (2.1) has a major drawback: it does not provide any information on the motion component normal to the brightness gradient. Or equivalently, only motions perpendicular to image contours can be inferred. Writing $\boldsymbol{v}(\boldsymbol{x},t) = \boldsymbol{v}_N(\boldsymbol{x},t) + \boldsymbol{v}_T(\boldsymbol{x},t)$, with $\boldsymbol{v}_N$ and $\boldsymbol{v}_T$ the components respectively normal and tangential to the luminance level sets, equation (2.1) gives:

$$\begin{aligned} &\frac{\partial I}{\partial t}(\boldsymbol{x},t) + [\boldsymbol{v}_N(\boldsymbol{x},t) + \boldsymbol{v}_T(\boldsymbol{x},t)]\cdot\nabla I(\boldsymbol{x},t) = 0 \\ \Rightarrow\ &\boldsymbol{v}_N(\boldsymbol{x},t) = -\frac{\partial_t I(\boldsymbol{x},t)}{\|\nabla I(\boldsymbol{x},t)\|}\frac{\nabla I(\boldsymbol{x},t)}{\|\nabla I(\boldsymbol{x},t)\|}\,. \end{aligned} \tag{2.2}$$

The impossibility to determine the tangential component $\boldsymbol{v}_T$ leads to an ambiguous state, since different kinds of motion could possibly fit. This situation is named the *aperture problem*; it is illustrated by Figure 2.2. In order to close the estimation problem, it is necessary to resort to regularization schemes applied to the estimated motion – this will be our second ingredient. These regularizations compensate the lack of information from images, often by enforcing continuity properties or parametric expressions of the solution. Classically, spacial smoothing terms are employed.

From (2.2), it is clear that no motion can be estimated as soon as any of the temporal or spacial gradients ($\partial_t I(\boldsymbol{x},t)$ or $\nabla I(\boldsymbol{x},t)$, respectively) vanishes. This problematic situation occurs in particular with images featuring constant, uniform areas: inside such regions no information is available from our brightness consistency assumption – in other words, there exists an infinite number of solutions. From this last remark, we can directly conclude that optic flow methods in general will behave better on textured images.

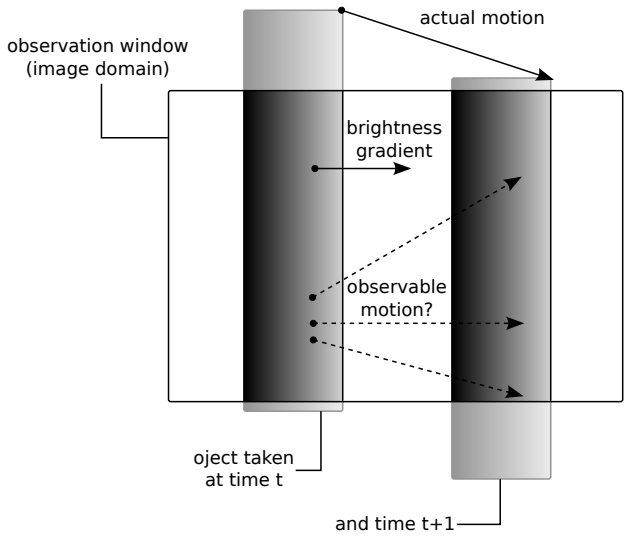


Figure 2.2: An illustration of the aperture problem. A gradient-filled rectangular object is diagonally shifted between time t and $t+1$. However, within the given observation window, luminance gradient of the rectangle remains always horizontal (level sets are vertical). Therefore the horizontal motion component (normal to level sets) is immediately identified, but the vertical component (tangential to level sets) cannot be inferred visually.

2.2 Motion Estimation Methods: a Quick Overview

We have established in Section 2.1.1 and 2.1.2 that a motion estimation method features two main aspects: the data term (or data model), to link the estimated motion to image data, and a regularization term to compensate for local lack of information. Let us now introduce the three most common types of estimation methods. Some more specific wavelet-based variants may be quickly mentioned hereafter; they will be more extensively presented in Chapter 5 after the preliminary introduction of wavelets.

2.2.1 Digital Image Correlations

This family of methods, named *digital image correlations*, performs *independent* local estimations on subregions of the images. The idea is to correlate a small region of the first image with a translated region of the second image; the translation vector inducing a correlation peak is considered to be the displacement at the center of the considered region.

A set of small windows $W(\boldsymbol{x})$ centered at various points $\boldsymbol{x}$ of a subset $\Omega_C \in \Omega$ of the image domain is considered. For each window, the algorithm looks for the constant displacement $\boldsymbol{v}$ that minimizes the (normalized and centered) cross-correlation function:

$$\forall\, \boldsymbol{x} \in \Omega_C\,,\ \hat{\boldsymbol{v}}(\boldsymbol{x}) = \arg\min_{\boldsymbol{v}} \sum_{\boldsymbol{y}\in W(\boldsymbol{x})} -\frac{\left(I_1(\boldsymbol{y}+\boldsymbol{v}) - \mu_1(\boldsymbol{x}+\boldsymbol{v})\right)\left(I_0(\boldsymbol{y}) - \mu_1(\boldsymbol{x})\right)}{\sigma_1^2(\boldsymbol{x}+\boldsymbol{v})\sigma_0^2(\boldsymbol{x})}\,, \tag{2.3}$$

where $\mu_j(\boldsymbol{x})$ and $\sigma_j(\boldsymbol{x})$ are estimations of the mean and the variance, respectively, of image j brightness over window $W(\boldsymbol{x})$ centered at $\boldsymbol{x}$. Here, the data term is the cross-correlation function. Regularization is implicitly defined by the size of windows $W(\boldsymbol{x})$, over which the sought displacement is assumed to be constant. The window should be large enough to contain enough information for the matching process to overcome local uncertainties due to the aperture problem. Such methods usually produce *sparse* motion fields, i.e. at a much lower resolution than input images. Evolved methods have been proposed, for instance to adapt locally the shape and orientation of the window [4]. Nevertheless, their efficient implementation in the Fourier domain and relative robustness has broaden their use, leading to widely used commercial solutions.

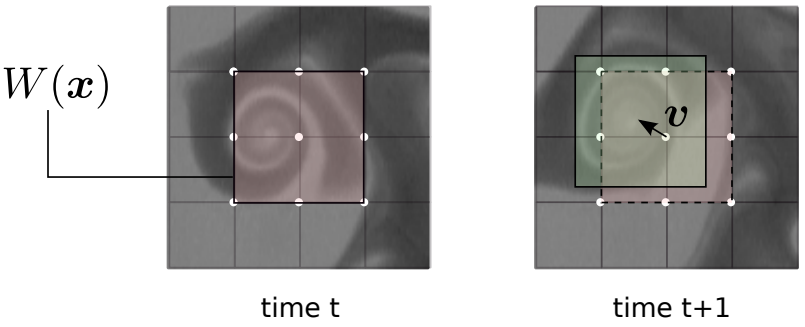


Figure 2.3: Illustration of the correlation-based method. At time t, window $W(\boldsymbol{x})$ (*red area*) is centered at $\boldsymbol{x} \in \Omega_C$ (*white dots*). At time $t+1$, the *green area* $W(\boldsymbol{x}+\boldsymbol{v})$ is the translated region that maximizes cross-correlations (2.3). The according translation vector $\boldsymbol{v}$ is the displacement at point $\boldsymbol{x}$. With the given grid, windows $W(\boldsymbol{x})$ have 50% overlay; here 9 vectors will be obtained (one per white dot).

2.2.2 Parametric Formulations

Another approach to tackle the aperture problem consists in adopting a local parametric formulation of the motion field $\boldsymbol{v}$:

$$\boldsymbol{v}(x) = \Phi(\boldsymbol{x})\theta\,, \tag{2.4}$$

where $\Phi(\boldsymbol{x})$ is a $2\times p$ matrix that depends on the chosen parametrization only and θ is the parameter vector of size p. Similarly to correlation methods, a set of subregions is considered. In each area $W(\boldsymbol{x})$, parameters $\theta_{\boldsymbol{x}}$ are estimated by the minimization of a functional based on the brightness consistency (2.1):

$$\hat{\theta}_{\boldsymbol{x}} = \arg\min_{\theta} \frac{1}{2} \int_{W(\boldsymbol{x})} g(\boldsymbol{x}-\boldsymbol{y}) \Big[\partial_t I(\boldsymbol{y}) + \nabla I(\boldsymbol{y}) \cdot \Phi(\boldsymbol{y})\theta_{\boldsymbol{x}}\Big]^2 d\boldsymbol{y}\,, \tag{2.5}$$

where g is a weighting function giving more importance to pixels $\boldsymbol{y}$ closer to the center $\boldsymbol{x}$ – typically, a gaussian function. Affine or quadratic parametrization can be considered, resulting in a piece-wise affine (or quadratic) motion. The well-known Lucas & Kanade estimator uses a constant parametrization [27]; the wavelet-based estimator proposed by Bernard [5] also relies on this assumption.

2.2.3 Differential Methods and Dense Estimation

A third family of methods aims at estimating *dense* motion fields, that is to say to get one velocity vector at every pixel of the considered images, vector which is dependent on other velocities in the neighborhood (by opposition to the correlations). These methods are often named *optic flow*, although from our first definition correlations-based and parametric approaches are also optic flow methods. Hence, in the following optic flow will refer explicitly to dense estimation methods.

Contrary to the two previous approaches which work locally, optic flow methods look for a *global solution* through the minimization of a functional, similar to an energy, defined over time and image domains:

$$\begin{aligned} \hat{\boldsymbol{v}} &= \arg\min_{\boldsymbol{v}} J_{\text{data}}(I,\boldsymbol{v}) + \alpha J_{\text{reg}}(\boldsymbol{v}) \\ &= \arg\min_{\boldsymbol{v}} \frac{1}{2}\int_{\Omega} \Big[f_{\text{data}}(I(\boldsymbol{x},t),\boldsymbol{v}(\boldsymbol{x},t))\Big]^2 d\boldsymbol{x} + \frac{\alpha}{2}\int_{\Omega}\Big[f_{\text{reg}}(\boldsymbol{v}(\boldsymbol{x},t))\Big]^2 d\boldsymbol{x} \end{aligned} \tag{2.6}$$

where function f_{data} is the data model (it depends on motion $\boldsymbol{v}$ and images I), f_{reg} is the regularization term (depends on $\boldsymbol{v}$ and its derivatives only, and possibly on some extern parameters), and α is a scalar parameter that balances the two terms. Apart from returning dense fields, a remarkable interest of optic flow methods lies in the wide choice of data and regularization terms,

which might be adapted to the nature of images, the behavior of the observed quantity and the specificities of the sought motion field. Horn and Schunck described in their famous 1981 article [20] the first design of such a dense motion estimator. They introduced a data model based on the brightness consistency assumption, which is now known as the optical flow constraint, and a first-order smoothing regularization term.

Before going into details, we have to keep in mind that estimation of a dense field involves a (very) high number of unknowns: with a square image of $N \times N = N^2$ px, the corresponding dense 2D motion field has $2N^2$ values. Even with images as small as 256×256 px, 131 072 different values have to be estimated. Considering instead 1024×1024 px images, the number of unknowns rises up to 2 097 152. Numerical resolution of problems involving such a high number of variables is to be addressed carefully. Furthermore, not only the aperture problem leads to uncertainties, but there are twice more unknowns than independent measurements (two unknowns per pixel), so that the estimation problem is even more underconstrained. Hence regularization is of very important matter.

Data Models

The two data terms presented below are obtained from the brightness consistency assumption (2.1). Nevertheless, a wide range of other models have been proposed.

From now on, we will denote by I_0 and I_1 the two successive frames of size $N \times N$ pixels at time t and $t+1$, respectively:

$$I_0(\boldsymbol{x}) \triangleq I(\boldsymbol{x}, t) \text{ and } I_1(\boldsymbol{x}) \triangleq I(\boldsymbol{x}, t+1)\,, \quad \forall\, \boldsymbol{x} \in \Omega\,. \tag{2.7}$$

The *optic flow constraint* (OFC) is directly obtained from the brightness consistency assumption (2.1). Time derivative $\partial_t I$ is replaced by a finite difference. With $\Delta t = 1$, one gets:

$$\text{(OFC)} \quad I_1(\boldsymbol{x}) - I_0(\boldsymbol{x}) + \boldsymbol{v}(\boldsymbol{x}) \cdot \nabla I_1(\boldsymbol{x}) = 0 \tag{2.8}$$

This model is linear in $\boldsymbol{v}$.

The *displaced frame difference* equation (DFD) is the integrated version of OFC, obtained by integrating (2.1) between t and $t+1$, under the assumption of a constant displacement during the image inter-frame latency ($\boldsymbol{v}(\boldsymbol{x}, t) = \boldsymbol{v}(\boldsymbol{x})$). Contrary to the later, it remains valid whatever the motion amplitude. It is, however, non linear in $\boldsymbol{v}$, and thus requires a specific optimization strategy.

$$\text{(DFD)} \quad I_0(\boldsymbol{x}) - I_1(\boldsymbol{x} + \boldsymbol{v}(\boldsymbol{x})) = 0 \tag{2.9}$$

Regularizations

A common strategy consists in adding a regularization term $J_{\text{reg}}(\boldsymbol{v})$ to the data-term functional, as in (2.6). The most simple choice is to drive the estimation toward a "smooth" solution, using a first-order regularization introduced by Horn & Schunck [20] that penalizes strong gradients:

$$J_{\text{reg}}(\boldsymbol{v}) = \frac{1}{2} \int_\Omega |\nabla v_1(\boldsymbol{x})|^2 + |\nabla v_2(\boldsymbol{x})|^2 \, d\boldsymbol{x}\,. \tag{2.10}$$

Many other, more complex schemes have been proposed in order to deal with the multiple natures of motions to be recovered – e.g. dealing with discontinuities. Some of these schemes, more adapted to fluid motion estimations, will be detailed later in Section 2.3. An issue, common to any penalization term, concerns the choice of parameter α that balances the relative importance given to the data model or to the regularization. Let us note that the estimation of the balance parameter α is a difficult problem in its own. This hyper-parameter estimation problem can be solved by tools such as bayesian selection, as well as with various criterions such as the L-curves.

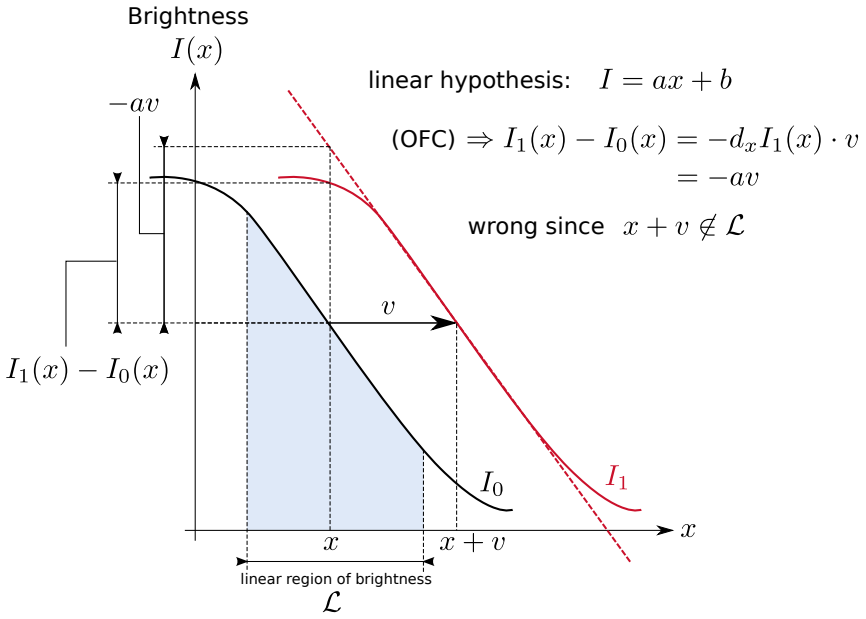


Figure 2.4: Illustration of the restriction of OFC (2.8) validity to displacements laying within the linear region $\mathcal{L}$ of brightness. Since $x + v$ does not belong to $\mathcal{L}$, the OFC is not verified.

2.2.4 Dealing with Large Displacements

Validity of some data models, such as the OFC model (2.8), is restricted to displacements laying within the linear region of image brightness – Fig. 2.4. When large displacements are involved, or in the presence of strong brightness gradients, the model is no longer valid. A common approach to overcome these limitations consists in adopting an incremental multiresolution estimation scheme. Images are sequentially subsampled and low-pass filtered, until apparent displacements are small enough for the linearity assumption to be valid. The resulting set of images is called a "pyramid". Motions are incrementally estimated at each level of the pyramid, following a coarse-to-fine scheme.

At scale [1] $j \leq 0$, approximations of images are computed by a gaussian filtering, followed by a decimation, of finer scale $j - 1$ images:

$$I_i^j = \downarrow \circ \left(G \star I_i^{j-1} \right) , \; i = 0, 1 , \tag{2.11}$$

where $\downarrow$ denotes the decimation operator and G s a gaussian kernel of variance proportional to 2^{-j}. Motions are estimated from the coarsest scale $C \leq 0$ down to the finest scale $F \leq C$. At scale j,

$$\begin{aligned} \boldsymbol{v}^j &= \tilde{\boldsymbol{v}}^j + \delta \boldsymbol{v}^j , \\ \text{with } \tilde{\boldsymbol{v}}^j &= \sum_{j < k \leq C} \mathcal{P}_j(\delta \boldsymbol{v}^k) . \end{aligned} \tag{2.12}$$

Let us clarify decomposition (2.12): current motion $\boldsymbol{v}_j$ is split into a known approximation $\tilde{\boldsymbol{v}}^j$, the sum of the projection $\mathcal{P}_j$ at current resolution 2^{-j} of all estimates obtained at coarser pyramid levels $k > j$, and an incremental part $\delta \boldsymbol{v}^j$ to be estimated. A first-order Taylor development of $I_1^j(\boldsymbol{x} + \boldsymbol{v}^j(\boldsymbol{x}))$ around known approximation $\tilde{\boldsymbol{v}}^j$ leads to:

$$\begin{gathered} \tilde{I}_1^j(\boldsymbol{x}) - I_0^j(\boldsymbol{x}) + \delta \boldsymbol{v}(\boldsymbol{x}) \cdot \nabla \tilde{I}_1^j(\boldsymbol{x}) = 0 \\ \text{where } \tilde{I}_1(\boldsymbol{x}) \triangleq \tilde{I}_1(\boldsymbol{x} + \tilde{\boldsymbol{v}}^j(\boldsymbol{x})) \text{ is the } \textit{motion-compensated} \text{ image.} \end{gathered} \tag{2.13}$$

The initial OFC estimation problem (2.8) is turned into a set of successive coarse-to-fine estimations, using (2.13) as the data model. Finally, the solution to the estimation problem is given by

1. See Remark 1 p. x regarding scale/resolution definitions and notations.

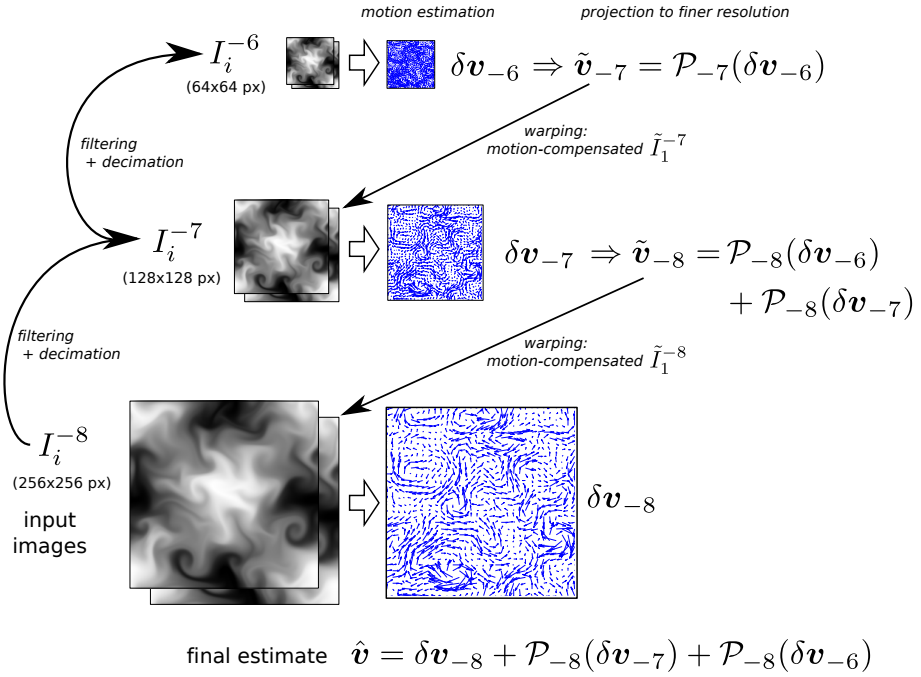


Figure 2.5: Illustration of the usual incremental approach. First the image pyramid is built (*left*), then apparent motions are estimated from coarser images (*top*) to the finest ones (*bottom*).

the sum of all coarse-to-fine estimates:

$$\boldsymbol{v} = \delta \boldsymbol{v}^F + \sum_{F<j\leq C} \mathcal{P}_F(\delta \boldsymbol{v}^j)\,. \tag{2.14}$$

A major drawback of this so-called incremental scheme is the freezing, at a given scale j, of all previous estimates $\delta \boldsymbol{v}^i$ with $j < i \leq C$. In addition, the low-pass filtering and decimation of the image essentially lie on heuristic arguments.

As previously pointed out, the DFD (2.2.3) validity is not limited to a certain range of displacements. However, being non-linear in $\boldsymbol{v}$, gradient-based minimization methods are likely to get caught in local minima, especially as large displacements are involved. We will see however that in practice the DFD is able to cope with much larger displacements that the OFC does, even associated to the incremental scheme described above.

2.3 Fluid-Dedicated Approaches

Correlation-based methods (Section 2.2.1) have been employed for long. Full systems including lasers, high-speed cameras, data acquisition interfaces and a digital image correlation software (e.g. Lavision's DaVis) are available and commonly used in laboratories. Free softwares are available as well, such as GPiv [42]. These approaches are particularly efficient with particle-seeded images (PIV) [4].

Optic-flow and parametric methods have been explored more recently in experimental fluid mechanics or in geophysics. Regarding the optic-flow data term, a model derived from the continuity equation was proposed by Fitzpatrick [16]. More consistent with the dynamics of fluid flows, its integrated version (ICE) was successfully used by Corpetti *et al.* [11]:

$$\text{(ICE)} \quad I_1(\boldsymbol{x} + \boldsymbol{v}(\boldsymbol{x})) - I_0(\boldsymbol{x}) e^{-\operatorname{div} \boldsymbol{v}} = 0\,. \tag{2.15}$$

Among the parametric approaches, Cuzol & Mémin [12] modeled the flow with a small number of vortex and source "particles", whereas Auroux & Fehrenbach [2] adopted a piecewise linear formulation.

When considering turbulent fluid motions, usual regularizations such as Horn & Schunck first-order scheme are not well-suited: the *curl* and *divergence* of the flow are important quantities expressed as linear combinations of motion gradients (Section 1.3.1). Hence, penalizing gradients without any consideration could result in inaccurate curl and divergence fields, and fail to describe accurately the flow. In order to preserve those quantities, fluid-dedicated regularization terms have been introduced by Suter [38] and later used by Corpetti *et al.* [11]. This regularization penalizes strong curl and divergence gradient, thereby moving the solution towards smooth blobs of vorticity and divergence:

$$J_{\text{reg}}(\boldsymbol{v}) = \frac{1}{2}\int_{\Omega} |\nabla \operatorname{curl}(\boldsymbol{v})(\boldsymbol{x})|^2 + |\nabla \operatorname{div}(\boldsymbol{v})(\boldsymbol{x})|^2 \, d\boldsymbol{x}\,. \tag{2.16}$$

Although leading to very good results, the implementation of this regularization term described by [11] is complex. More generally, high-order regularizations are more difficult to implement, since they require advanced discretization schemes. Yuan *et al.* [45] used finite mimetic difference method, while Auroux & Fehrenbach [2] chose instead finite elements. We will see how the wavelet formalism enables simple yet efficient implementations of such schemes.

Alternatively, constraints can be derived from some physical law governing the considered motion. When dealing with incompressible flows, an option consists in penalizing the divergence of the estimated motion, either by directly incorporating the constraint [45] or through an explicit regularization term [2]. Here, divergence-free vector wavelet bases will be built in Section 4.1. When dealing with turbulent fluid motion, Heas *et al.* [18] have used Kolmogorov's turbulence modeling. When the velocity increment function is strictly self-similar, isotropic and homogeneous, its p-th order longitudinal structure function behaves like a power law:

$$S_l(s,p) = E\left[\delta \boldsymbol{v}_l(s)^p\right] \sim \beta_p s^{p\zeta_p} \quad \forall s \in I \text{ the inertial range,} \tag{2.17}$$

where exponent ζ_p is though to be universal and depends on space dimension only, whereas β_p is a function of energy flux ϵ. From this self-similar prior, a set of constraint are derived at different scales $s \in I$ for the second order structure function. The estimation problem finally writes:

$$\begin{cases} \hat{\boldsymbol{v}} = \arg\min_{\boldsymbol{v}} J_{\text{data}}(I, \boldsymbol{v}) \\ \text{s.t. } h_s(\boldsymbol{v}) \triangleq \dfrac{1}{2}\left(E\left[\delta \boldsymbol{v}_l(s)^2\right] - \beta s^{\zeta}\right) = 0\,,\ \forall\, s \in I\,. \end{cases} \tag{2.18}$$

Power law parameters β and ζ are either given, or estimated using bayesian model selection recipes [18]. The later approach has the advantage of removing the need to supply external parameters β and ζ, however it requires much more computational time.

Part II

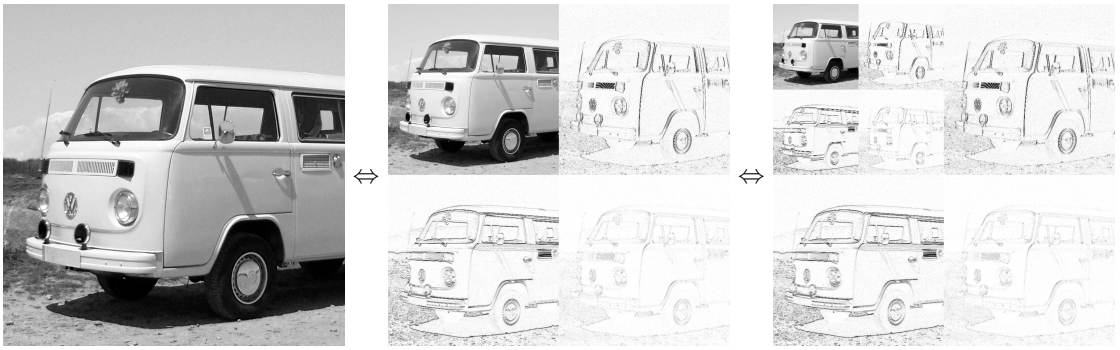
Wavelets

Chapter 3

Wavelets Framework

In order to analyze or exhibit some properties of a given signal, two interdependent spaces are often considered: the physical space, which describes temporal or spacial variations of the signal, and the spectral space, which instead gives information in terms of frequency – e.g. by the means of the Fourier transform. These two approaches are mutually exclusive: no frequency-related content can be accessed within the spacial space, and reciprocally. Wavelet transforms offer a sort of trade-off, giving a simultaneous access to both physical and spectral information, at the price however of a lower resolution in both spaces.

Roughly speaking, a step of the (dyadic) wavelet decomposition will split the input signal into a set of details, corresponding to information contained by a certain range of frequencies at spatial locations, and a coarser approximation of the signal. This step may then be applied again to this coarse approximation, giving again details and an even coarser approximation, and so on. Ultimately, a (very) coarse approximation remains, along with several sets of details containing spatially-localized information at various ranges of frequencies. Of course, by re-combining the remaining approximation with the various sets of details, the initial signal can be recovered. This forward and inverse transforms are illustrated in Fig 3.1.



(a) (b) (c)

Figure 3.1: Example of a 2D (dyadic, isotropic) wavelet transform. Input image (a) is transformed into (b): a coarser approximation (*top-left corner*) and details (*everything else*), and the process can be repeated again (c) and/or reversed to retrieve (a). Details were processed to enhance their visualization, vanishing values are in white.

3.1 General Principles

The following section introduces some of the basic principles of the wavelet formalism; most of this material is adapted from Stephane Mallat's *A Wavelet Tour of Signal Processing: The Sparse Way* [28], along with Kadri Harouna *et al.* [23]. The concept of nested approximation spaces that

form a multiresolution analysis is first derived for real 1D signal; construction of function bases for those spaces follows. Those principles are then extended to the case of real 2D signals, before considering the practical application to discrete signals.

3.1.1 Multiresolution Approximations and Detail Spaces

A multiresolution approximation of $\mathbf{L}^2(\mathbb{R})$ is built from a sequence $\{V_j\}_{j\in\mathbb{Z}}$ of closed subspaces, so-called *approximation spaces*, verifying:

$$\forall j \in \mathbb{Z}, \quad V_{j+1} \subset V_j \,; \tag{3.1}$$

$$\forall (j,k) \in \mathbb{Z}^2, \quad f(t) \in V_j \Leftrightarrow f(t-2^j k) \in V_j \,; \tag{3.2}$$

$$\forall j \in \mathbb{Z}, \quad f(2t) \in V_j \Leftrightarrow f(t) \in V_{j+1} \,; \tag{3.3}$$

$$\lim_{j\to+\infty} V_j = \bigcap_{j=-\infty}^{+\infty} V_j = \{0\} \,; \tag{3.4}$$

$$\lim_{j\to-\infty} V_j = \text{Closure}\left(\bigcup_{j=-\infty}^{+\infty} V_j \right) = \mathbf{L}^2(\mathbb{R}) \tag{3.5}$$

$$\exists \varphi \text{ such that } \{\varphi(t-n)\}_{n\in\mathbb{Z}} \text{ Riesz basis of } V_0. \tag{3.6}$$

Property (3.1) traduces the fact that approximation spaces are *nested*. From property (3.2), space V_j is invariant by any translation proportional to the considered scale 2^j. A representation in a given space V_j is sufficient to compute a coarser representation in space V_{j+1}, thanks to property (3.3). Finally, the set $\{V_j\}_{j\in\mathbb{Z}}$ is called a *multiresolution analysis* (*MRA*) of $\mathbf{L}^2(\mathbb{R})$.

Since approximation spaces are sequentially included within each other (3.1), they can be decomposed:

$$V_j = V_{j+1} \oplus W_{j+1} \,. \tag{3.7}$$

Those W_j are complements of approximation spaces, they are called *detail spaces*. The concept of wavelet decomposition resides in this association between approximation and detail spaces, along with the nesting property: a given signal is recursively split into details and approximation, in order to obtain a set of details at several scales and a remaining coarse approximation.

3.1.2 Wavelet Bases

Functions forming bases of V_j and W_j are called *scaling* and *wavelet* functions, respectively. From (3.7) only, the W_j are not unique. The completion of this definition will lead, either to the construction of *orthogonal* bases, or to the more general case of *biorthogonal* bases.

Orthogonal Wavelet Bases

We consider here the Riesz basis of V_0 (Prop. (3.6)) to be orthonormal – if necessary by orthonormalizing the original basis, see [28]. Using dilations and translations of this mother scaling function φ – see (3.2), (3.3) and remark 2, one can construct orthonormal bases of spaces V_j:

$$V_j = \text{span}\{\varphi_{j,k},\, k \in \mathbb{Z}\}, \quad \text{with } \langle \varphi_{j,p} \,;\, \varphi_{j,q} \rangle_{\mathbf{L}^2(\mathbb{R})} = \delta_{pq} \;\forall\, j,p,q \in \mathbb{Z}. \tag{3.8}$$

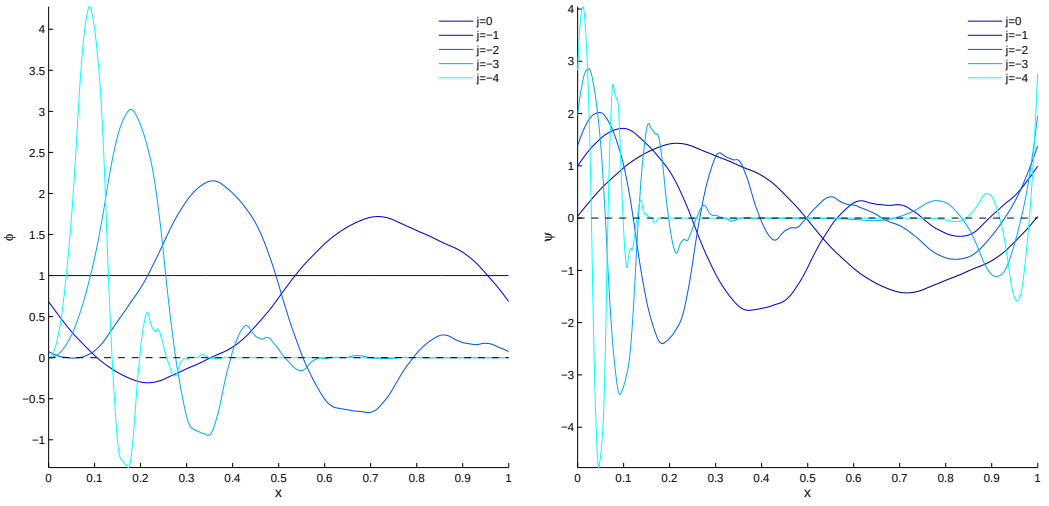
Then W_j is defined as the orthogonal complement of V_j in V_{j-1}:

$$W_j = V_{j-1} \cap (V_j)^{\perp} \,. \tag{3.9}$$

Similarly to (3.8), orthogonal bases of W_j are built from translations and dilations of the mother wavelet function denoted by ψ:

$$W_j = \text{span}\{\psi_{j,k},\, k \in \mathbb{Z}\}, \quad \text{with } \left\langle \psi_{i,p} \,;\, \psi_{j,q} \right\rangle_{\mathbf{L}^2(\mathbb{R})} = \delta_{ij}\delta_{pq} \;\forall\, i,j,p,q \in \mathbb{Z}. \tag{3.10}$$

Figure 3.2 gives an example of scale and wavelet functions along with their dilations.



(a) (b)

Figure 3.2: Example of (periodized) scale (*left*) and wavelet (*right*) functions. They correspond to the Daubechies wavelet family, with 5 vanishing moments.

Biorthogonal Wavelet Bases

An alternative option to (3.9) is to define W_j from another multiresolution analysis $\{\tilde{V}_j\}_{j\in\mathbb{Z}}$, of scaling and wavelet functions $\tilde{\varphi}$ and $\tilde{\psi}$, respectively:

$$W_j = V_{j-1} \cap (\tilde{V}_j)^{\perp} . \tag{3.11}$$

Then $\{V_j, \tilde{V}_j\}_j$ is called a *biorthogonal multiresolution analysis* (*BMRA*) of $\mathbf{L}^2(\mathbb{R})$. The biorthogonality property arises from following relations:

$$\begin{aligned}
\left\langle \varphi_{j,p} \,;\, \tilde{\varphi}_{j,q} \right\rangle_{\mathbf{L}^2(\mathbb{R})} &= \delta_{pq} \,; \\
\left\langle \psi_{i,p} \,;\, \tilde{\psi}_{j,q} \right\rangle_{\mathbf{L}^2(\mathbb{R})} &= \delta_{ij}\delta_{pq} \,; \\
\left\langle \varphi_{j,p} \,;\, \tilde{\psi}_{j,q} \right\rangle_{\mathbf{L}^2(\mathbb{R})} &= 0 \,; \\
\left\langle \tilde{\varphi}_{j,p} \,;\, \psi_{j,q} \right\rangle_{\mathbf{L}^2(\mathbb{R})} &= 0 \,; \quad \forall\, i,j,p,q \in \mathbb{Z} \,.
\end{aligned} \tag{3.12}$$

As a consequence, orthogonal multiresolution analyses are a particular case of biorthogonal ones, where $\tilde{\varphi} = \varphi$, $\tilde{\psi} = \psi$ and $\tilde{V}_j = V_j$. To remain general, the biorthogonal formulation will be employed in the following.

Refinement Relations

Property 3.1 rises up the following two-scale relations (a.k.a. *refinement relations*) between scaling and wavelet functions:

$$\varphi_{1,0}(x) = \frac{1}{\sqrt{2}}\varphi(\frac{x}{2}) = \sum_{n=-\infty}^{+\infty} h[n]\varphi_{0,n}(x)\,, \qquad \text{with } h[n] = \langle \varphi_{1,0}\,;\,\tilde{\varphi}_{0,n}\rangle\,; \tag{3.13}$$

$$\tilde{\varphi}_{1,0}(x) = \frac{1}{\sqrt{2}}\tilde{\varphi}(\frac{x}{2}) = \sum_{n=-\infty}^{+\infty} \tilde{h}[n]\tilde{\varphi}_{0,n}(x)\,, \qquad \text{with } \tilde{h}[n] = \langle \tilde{\varphi}_{1,0}\,;\,\varphi_{0,n}\rangle\,; \tag{3.14}$$

$$\psi_{1,0}(x) = \frac{1}{\sqrt{2}}\psi(\frac{x}{2}) = \sum_{n=-\infty}^{+\infty} g[n]\varphi_{0,n}(x)\,, \qquad \text{with } g[n] = \left\langle \psi_{1,0}\,;\,\tilde{\varphi}_{0,n}\right\rangle\,; \tag{3.15}$$

$$\tilde{\psi}_{1,0}(x) = \frac{1}{\sqrt{2}}\tilde{\psi}(\frac{x}{2}) = \sum_{n=-\infty}^{+\infty} \tilde{g}[n]\tilde{\varphi}_{0,n}(x)\,, \qquad \text{with } \tilde{g}[n] = \left\langle \tilde{\psi}_{1,0}\,;\,\varphi_{0,n}\right\rangle\,. \tag{3.16}$$

Sequences $\{h[n]\}_{n\in\mathbb{Z}}$, $\{\tilde{h}[n]\}_{n\in\mathbb{Z}}$, $\{g[n]\}_{n\in\mathbb{Z}}$ and $\{\tilde{g}[n]\}_{n\in\mathbb{Z}}$ are called the *conjugate mirror filters* associated to scaling and wavelet functions, respectively. They are linked together by:

$$\begin{aligned} g[n] &= (-1)^{1-n}\tilde{h}[1-n];\\ \tilde{g}[n] &= (-1)^{1-n}h[1-n]. \end{aligned} \tag{3.17}$$

Should an orthogonal basis be considered, then $\tilde{h} = h$ and $\tilde{g} = g$.

In practice, mother wavelet and scaling functions are often defined by their respective filter instead of an analytical formula. These filters also play an important role in the fast implementation of the forward and inverse wavelet transforms introduced hereafter.

3.1.3 Finite Signals and Fast Transforms

In practice, we work with finite and discrete signals. We will first see how to handle the discrete aspect and the fast transforms that arise, before considering a way to deal with the finiteness.

Approximation and Wavelet Coefficients

Let $\{z[n], n \in \mathbb{Z}\}$ be a discrete signal sampled every 2^F, $F \in \mathbb{Z}$. Formally, in order to achieve a wavelet decomposition of this signal, it is necessary:

(i) to associate to $\{z[n]\}_n$ a function $w(x) \in V_F$;

(ii) from which we may compute the coefficients of its representation in the chosen basis of V_F.

Assuming we already completed (i), a wavelet decomposition (ii) of this function w consists in its projection to a coarser space V_C, with $F \leq C$ and to the complementary detail spaces W_j, with $F+1 \leq j \leq C$:

$$\begin{aligned} w &= \mathcal{P}_C(w) + \sum_{F+1}^{C} \mathcal{Q}_j(w) \;\in V_F\,,\\ &\text{where } \mathcal{P}_C(w) \in V_C\\ &\text{and } \mathcal{Q}_j(w) = \mathcal{P}_j(w) - \mathcal{P}_{j+1}(w) \in W_j\,. \end{aligned} \tag{3.18}$$

Projections $\mathcal{P}_j(w)$ and $\mathcal{Q}_j(w)$ are defined by:

$$\mathcal{P}_j(w) = \sum_{k=-\infty}^{+\infty} a_{j,k}\varphi_{j,k}\,, \quad a_{j,k} = \langle w\,;\,\tilde{\varphi}_{j,k}\rangle\,, \tag{3.19}$$

and similarly:

$$\mathcal{Q}_j(w) = \sum_{k=-\infty}^{+\infty} d_{j,k}\psi_{j,k}\,, \quad d_{j,k} = \langle w\,;\,\tilde{\psi}_{j,k}\rangle\,. \tag{3.20}$$

Coefficients $a_{j,k}$ and $d_{j,k}$ in (3.19) and (3.20) are called *approximation and detail coefficients*, respectively. Altogether, the set $\{a_{C,k}\,,\, d_{j,k}\,,\, F+1 \leq j \leq C\,,\, -\infty \leq k \leq +\infty\}$ from projection (3.18) is the representation of function w in the considered wavelet basis of V_F. Now, before addressing the choice of function w (i), let us see how these coefficients are actually computed.

Fast Transforms

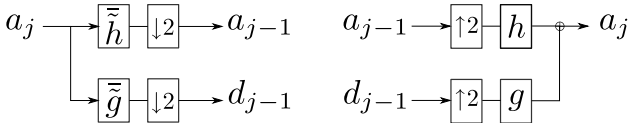
Thanks to refinement relations (3.13) and (3.15), wavelet coefficients in (3.19) and (3.20) can be computed recursively with a fast algorithm that cascades convolutions with conjugate mirror filters and decimation/expansion operations – a *filterbank*. The *forward* wavelet transform (decomposition) computes, from a given approximation $\mathcal{P}_j(w)$, coefficients of the coarser approximation $\mathcal{P}_{j+1}(w)$, along with its details $\mathcal{Q}_{j+1}(w)$:

$$\begin{aligned} a_{j+1,p} &= \sum_{n=-\infty}^{+\infty} \tilde{h}[n-2p]a_{j,n} = a_j \star \bar{\tilde{h}}[2p]\,; \\ d_{j+1,p} &= \sum_{n=-\infty}^{+\infty} \tilde{g}[n-2p]a_{j,n} = a_j \star \bar{\tilde{g}}[2p]\,; \end{aligned} \tag{3.21}$$

where $\bar{\cdot}$ denotes the time-reverse operator, i.e. $\bar{x}[n] = x[-n]$. Conversely, the *inverse* wavelet transform (reconstruction) recombines the coefficients of an approximation $\mathcal{P}_{j+1}(w)$ with its details $\mathcal{Q}_{j+1}(w)$ in order to recover the coefficients of finer approximation $\mathcal{P}_j(w)$:

$$\begin{aligned} a_{j,p} &= \sum_{k=-\infty}^{+\infty} h[p-2n]a_{j+1,n} + \sum_{k=-\infty}^{+\infty} g[p-2n]d_{j+1,n} \\ &= \check{a}_{j+1} \star h[p] + \check{d}_{j+1} \star g[p]\,; \end{aligned} \tag{3.22}$$

where $\check{\cdot}$ denotes the expansion operator, i.e. $\check{x}[n] = x[p]$ if $n = 2p$, 0 elsewhere. These so-called filterbanks are represented Figure 3.3.



(a) Forward transform (b) Inverse transform

Figure 3.3: Filterbanks for the 1D orhogonal transforms.

Initialization

From fast decomposition expressions 3.21, it suffices to know the fine approximation coefficients $\{a_{F,k}\}$ representing w in V_F to be able to compute iteratively every coefficients of coarser approximations and detail spaces. Now, how can this w be chosen (i) so that the $\{a_{F,k}\}$ can be deduced easily from known samples $\{z[n]\}_n$? Mallat [28] suggests

$$w(x) = \sum_{n=-\infty}^{+\infty} z[n]\varphi\left(2^{-F}t - n\right) \in V_F\,,$$

which, assuming w is regular, leads to

$$z[n] = \sqrt{2^{-F}}a_{F,n} \approx w\left(\frac{n}{2^{-F}}\right) \quad \Rightarrow \quad a_{F,k} = \frac{1}{\sqrt{2^{-F}}}z[k]\ \forall k \in \mathbb{Z}\,. \tag{3.23}$$

Hence the fine approximation coefficients are simply the input samples, up to a normalization factor. Other approaches exists, but were not considered in the presented work.

Finite signals

Finally, we must consider finite signals: $\{z[n]\,,\, 0 \leq n \leq 2^{-F}-1\}$ has 2^{-F} samples only. Therefore the set of fine approximation coefficients $\{a_{F,k}\}_k$ given by 3.23 is finite as well. Then, problems arise when computing convolutions in 3.21 and 3.22, as values are required outside boundaries of a_F. To get past these difficulties, input signals are considered to be defined on a discretization of $[0,1] \subset \mathbb{R}$, the resolution of this discretization increasing with the number of samples. Wavelet and scaling functions are then transformed to get bases of $\mathbf{L}^2([0,1])$ instead of $\mathbf{L}^2(\mathbb{R})$.

The simplest way to achieve this transformation, according to [28], is to periodize each basis function over $[0,1]$:

$$f \in \mathbf{L}^2(\mathbb{R}) \quad \Rightarrow f^{\mathrm{per}}(x) = \sum_{k=-\infty}^{+\infty} (x+k)\,,\, \forall x \in [0,1]\,. \tag{3.24}$$

Considering $[0,1]$ instead of $\mathbb{R}$ and applying periodization 3.24 has three important practical consequences:

- the coarsest scale available is always 0;
- at a given scale $j \leq 0$, there are only 2^{-j} different periodized functions $\varphi_{j,k}^{\mathrm{per}}$ and $\psi_{j,k}^{\mathrm{per}}$, and as many coefficients;
- the only practical modification is to replace convolutions in fast transforms 3.21 and 3.22 by circular convolutions.

Our discrete, finite signal $\{z[n]\}_n$ is now represented by

$$\left\{\{a_{C,k}\,,\, F \leq C \leq 0\,,\, 0 \leq k \leq 2^{-C}-1\}\,;\, \{d_{j,k}\,,\, F+1 \leq j \leq C\,,\, 0 \leq k \leq 2^{-j}-1\}\right\}\,.$$

The main drawback of this periodization is the creation of high-amplitude coefficients near boundaries. Alternative approaches are available (e.g. folding, padding ...) but are more complicated to implement. In the following, *we will always consider periodized bases* (see again Fig. 3.2), and drop notation $\cdot^{\mathrm{per}}$ accordingly.

For our 1D signal z of $N_F = 2^{-F}$ samples and wavelet filters h, g with K non-zero coefficients, a "full decomposition" (i.e.up to coarsest approximation $\mathcal{P}_0(w)$), or a "full reconstruction" (i.e. recovering w from $\mathcal{P}_0(w)$ and all details $\mathcal{Q}_j(w)$, $F-1 \leq j \leq 0$) is achieved in less than $2KN_F$ additions and multiplications. As a consequence, the computational cost of a fast wavelet transform is directly proportional to the length of the support of filters g, h. These supports are related to the supports of the associated scaling or wavelet functions. As we shall see in the forthcoming section 3.2.1, they are also directly related to φ and ψ regularity.

3.1.4 Extension to 2D Signals

Previous results are extended to the case of 2D signals, using tensor products of the one-dimensional periodized wavelet bases, in order to obtain *separable* multiscale bases of $\mathbf{L}^2([0,1]^2)$. The orthogonal case only is presented here, in order to keep notations simple.

From a given multiresolution analysis $\{V_j\}_j$ of $\mathbf{L}^2([0,1])$, we define space $V_j \otimes V_j \subset \mathbf{L}^2([0,1]^2)$ as:

$$V_j \otimes V_j = \mathrm{span}\left\{\varphi_{j,k_1} \otimes \varphi_{j,k_2}\,,\, (k_1,k_2) \in [0; 2^{-j}-1]^2\right\}\,,\, \forall\, j \in \mathbb{Z}\,, \tag{3.25}$$

where

$$\left(\varphi_{j,k_1} \otimes \varphi_{j,k_2}\right)(\boldsymbol{x}) \triangleq \varphi_{j,k_1}(x_1)\varphi_{j,k_2}(x_2)\,,\, \forall \boldsymbol{x} = (x_1,x_2)^{\boldsymbol{T}} \in [0,1]^2\,.$$

The set of spaces $\{V_j \otimes V_j\}_{j \in \mathbb{N}^-}$ forms a multiresolution analysis of $\mathbf{L}^2([0,1]^2)$. Then, there exists two different kinds of 2D dyadic wavelet transform, according to the way the detail spaces are introduced.

Isotropic Transform

The so-called *isotropic transform* is derived using (3.7):

$$\begin{aligned} V_j \otimes V_j &= (V_{j+1} \oplus W_{j+1}) \otimes (V_{j+1} \oplus W_{j+1}) \\ &= (V_{j+1} \otimes V_{j+1}) \oplus (V_{j+1} \otimes W_{j+1}) \oplus (W_{j+1} \otimes V_{j+1}) \oplus (W_{j+1} \otimes W_{j+1})\,. \end{aligned} \tag{3.26}$$

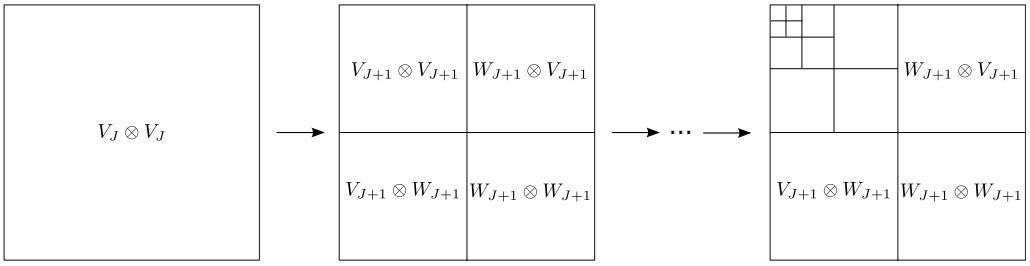


Figure 3.4: Illustration of the isotropic wavelet transform. Approximation spaces are recursively split into a coarser approximation space and three detail spaces.

Thus each approximation space $V_j \otimes V_j$ is recursively split into a coarser approximation space and three detail spaces; this is illustrated in Fig. 3.4. Basis functions corresponding to those detail spaces are given by translations and dilations of the three following mother wavelets:

$$\psi^1(\boldsymbol{x}) \triangleq (\varphi \otimes \psi)(\boldsymbol{x}) = \varphi(x_1)\psi(x_2)\,, \quad \text{for } V_0 \otimes W_0\,; \tag{3.27}$$

$$\psi^2(\boldsymbol{x}) \triangleq (\psi \otimes \varphi)(\boldsymbol{x}) = \psi(x_1)\varphi(x_2)\,, \quad \text{for } W_0 \otimes V_0\,; \tag{3.28}$$

$$\psi^3(\boldsymbol{x}) \triangleq (\psi \otimes \psi)(\boldsymbol{x}) = \psi(x_1)\psi(x_2)\,, \quad \text{for } W_0 \otimes W_0\,. \tag{3.29}$$

Supplemented by mother scaling function $\varphi \otimes \varphi$ associated to approximation spaces (Eq. (3.25)), it gives a total of four types of basis functions, each one having one dilation and two translation parameters: $j \in \mathbb{N}^-$ and $\boldsymbol{k} = (k_1, k_2) \in [0; 2^{-j} - 1]^2$.

Anisotropic Transform

In order to derive the *anisotropic transform*, one starts by recursively applying (3.7) to V_j, e.g.:

$$V_j = V_0 \oplus W_0 \oplus W_{-1} \oplus W_{-2} \oplus \cdots \oplus W_{j+1}\,, \tag{3.30}$$

Inserting the latter expression into $V_j \otimes V_j$ leads to:

$$\begin{aligned} V_j \otimes V_j &= (V_0 \oplus W_0 \oplus W_{-1} \oplus \cdots \oplus W_{j+1}) \otimes (V_0 \oplus W_0 \oplus W_{-1} \oplus \cdots \oplus W_{j+1}) \\ &= (V_0 \otimes V_0) \oplus \bigoplus_{j_2=0}^{j+1} (V_0 \otimes W_{j_2}) \oplus \bigoplus_{j_1=0}^{j+1} (W_{j_1} \otimes V_0) \oplus \bigoplus_{j_1,j_2=0}^{j+1} (W_{j_1} \otimes W_{j_2})\,. \end{aligned} \tag{3.31}$$

This approach is equivalent to performing a 1D wavelet decomposition along lines, then another one along columns – see Fig. 3.5. Basis functions corresponding to these spaces are given as well by translations and dilations of φ and ψ in following expressions:

$$(\varphi \otimes \varphi)(\boldsymbol{x}) = \varphi(x_1)\varphi(x_2)\,, \quad \text{for } V_0 \otimes V_0\,; \tag{3.32}$$

$$(\varphi \otimes \psi)(\boldsymbol{x}) = \varphi(x_1)\psi(x_2)\,, \quad \text{for } V_0 \otimes W_0\,; \tag{3.33}$$

$$(\psi \otimes \varphi)(\boldsymbol{x}) = \psi(x_1)\varphi(x_2)\,, \quad \text{for } W_0 \otimes V_0\,; \tag{3.34}$$

$$(\psi \otimes \psi)(\boldsymbol{x}) = \psi(x_1)\psi(x_2)\,, \quad \text{for } W_0 \otimes W_0\,. \tag{3.35}$$

Therefore basis functions for the anisotropic transform have four parameters: two dilations and two translations (one per direction): $\boldsymbol{j} = (j_1, j_2) \in \mathbb{N}^- \times \mathbb{N}^-$ and $\boldsymbol{k} \in [0; 2^{-j_1} - 1] \times [0; 2^{-j_2} - 1]$.

Fast Transforms

Just like the 1D transforms and thank to the separability of the 2D bases, both isotropic and anisotropic 2D transforms find fast implementations by filterbanks. Figure 3.6 gives the filterbanks structure for the forward and inverse isotropic transforms. Filterbanks for the anisotropic transform are exactly the same as the 1D case (Figure 3.3), applied first to rows, then to columns. Considering a 2D signal of $(N_F)^2 = 2^{-2F}$ samples, full decomposition or reconstruction are achieved in less than $\frac{8}{3}K(N_F)^2$ operations – see (B.1) and [28]. Finally, the biorthogonal case is very simple to handle, as it suffices to replace filters g, h at the decomposition by $\tilde{g}$, $\tilde{h}$, juste like the 1D case Figure 3.3.

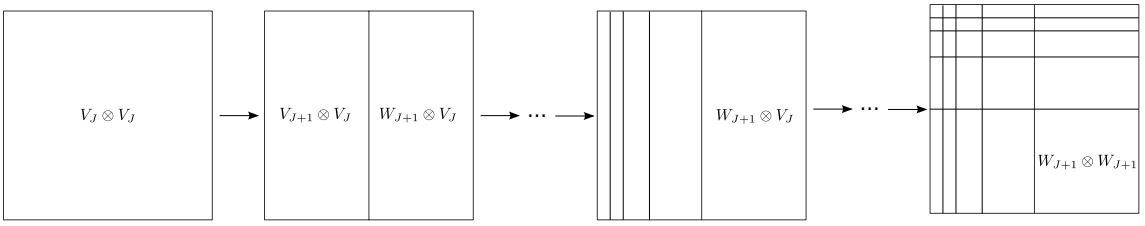
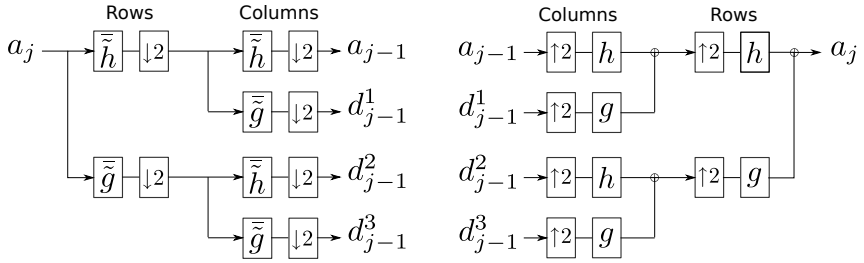


Figure 3.5: Illustration of the anisotropic wavelet transform. A 1D transform is applied along lines until the desired coarse approximation space is reached; then the operation is repeated along columns.



(a) Forward transform (b) Inverse transform

Figure 3.6: Filterbanks for the 2D isotropic orthogonal transforms.

3.2 Advanced Properties

This section introduces some more elaborate properties of the wavelet formalism that we shall use later.

3.2.1 Vanishing Moments and Regularity

Vanishing Moments

The notion of *vanishing moments* (VM) simply traduces the orthogonality of a function to polynomials up to a certain order: f in $\mathbf{L}^2(\mathbb{R})$ has m vanishing moments if and only if

$$\langle f\,;\, x^p \rangle_{\mathbf{L}^2} = \int_{\mathbb{R}} x^p f(x) dx = 0\,,\ \forall\, 0 \leq p < m\,. \tag{3.36}$$

Lipschitz regularity

The Lipschitz regularity gives a measure of the local regularity of a given signal. A function $w(t)$ is *pointwise-Lipschitz* $\alpha \geq 0$ at t_0 if there exists a local polynomial $P_{t_0}(t)$ of degree $n = \lfloor \alpha \rfloor$ and K constant such that

$$|w(t) - P_{t_0}(t)| \leq K|t - t_0|^\alpha \tag{3.37}$$

It is then *uniformly-Lipschitz* α over $[a, b]$ if it satisfies (3.37) for any $t_0 \in [a, b]$, with K independent of t_0.

Coefficients decay and vanishing moments

The uniform Lipschitz regularity of a signal can be related to the asymptotic decay, across scales, of its wavelet coefficients amplitude. Let us consider a signal $w \in \mathbf{L}^2([0, 1])$, uniformly Lipschitz α over $[0, 1]$, and its projection on a wavelet basis. The wavelet has n vanishing moments and is $\mathcal{C}^n$ with fast decay derivatives. At fine scales,

– if $n < \alpha$, the decay of coefficient amplitude depends on n:

$$\left|\left\langle w\,;\,\psi_p^j\right\rangle_{\mathbf{L}^2}\right| = |d_{j,p}| \sim 2^{-j(n+1/2)}\,;$$

– if $n \geq \alpha$, it depends on α [28]:

$$\exists A > 0 \text{ such that } |d_{j,p}| \leq A 2^{-j(\alpha+1/2)}\,.$$

A similar condition relates the uniform Lipschitz regularity and the asymptotic decay of Fourier coefficients. However, wavelets outperform the Fourier basis as they also allow to measure the *local* (pointwise) Lipschitz regularity [21].

Wavelet Approximations

Approximations of a given signal are obtained by keeping a small portion of basis atoms, for compression purposes for instance. Two different approaches are possible:

– The *linear* approximation projects the signal on the N first low-frequency atoms of the basis.
– With the *non-linear* approximation, the N "more interesting" atoms of the basis are chosen, according to the considered signal.

In both cases, excluding atoms of the basis leads to approximation errors. With uniformly regular signals, both approaches give comparable results. However, when the signal is not uniformly regular (e.g. in the presence of local, rapid variations or discontinuities), non-linear approximations enables to locally adjust the precision of the approximation, in order to minimize the error [28].

Thanks to their regularity properties, wavelets are able to encode the regular part of a signal with a small number of coefficients at coarse scales, then detail coefficients at finer scales appear only where the signal is locally irregular. This can be seen in Figure 3.1: non-vanishing coefficients are visible at the edges of the van (discontinuities), or within the grass and sand where the image is quite irregular. At the contrary, no small-scale coefficients are present within uniform (regular) areas, such as the sky or the shadow under the van. Wavelets therefore constitutes a powerful tool for non-linear approximation, as it suffices to keep the basis atoms corresponding to the N largest amplitude coefficients. Furthermore, there exists a bound on the reconstruction error as a function of N [28]. These last properties, combined to the local regularity measurement property, make wavelet bases a very powerful tool for signal compression and sparse representations, with numerous applications such as the JPEG2000 image compression standard.

Chapter 4

Wavelets Applications

After introducing some of the main concepts of the wavelet formalism in previous Chapter 3, we may now focus on particular applications that will be later involved into the design of our wavelet-based optical flow estimators. The first application is the construction of divergence-free wavelet bases – Section 4.1, which directly incorporate the *physical null-divergence constraint*. Then, we will examine some properties of wavelet derivatives in Section 4.2, as well as the computation of their *connection coefficients*, in order to set-up regularization schemes in Section 6.2.

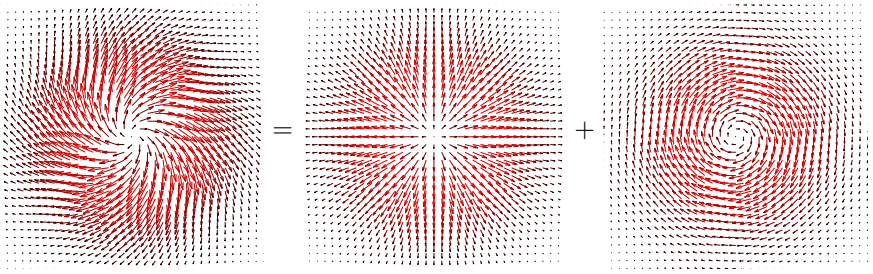
4.1 Divergence-Free Bases

4.1.1 Helmholtz's Decomposition and the Divergence-Free Constraint

The fundamental theorem of vector calculus, also known as *Helmholtz's decomposition* theorem, states that any smooth-enough, rapidly decaying vector field can be uniquely represented by the sum of a *curl-free* (irrotational) and a *divergence-free* (solenoidal) vector fields:

$$\begin{aligned} \boldsymbol{v} = \begin{pmatrix} v_1 \\ v_2 \end{pmatrix} \in \mathbf{L}^2(\mathbb{R}^2) \times \mathbf{L}^2(\mathbb{R}^2) \Rightarrow \ \boldsymbol{v} &= \boldsymbol{v}^{\text{curl}} + \boldsymbol{v}^{\text{div}} \, ; \\ \text{where } \boldsymbol{v}^{\text{curl}} \in \mathcal{H}^{\text{curl}}(\mathbb{R}^2) &= \left\{ \boldsymbol{v} \in (\mathbf{L}^2(\mathbb{R}^2))^2 \, , \ \text{curl}(\boldsymbol{v}) = \partial_{x_1} v_2 - \partial_{x_2} v_1 = 0 \right\} \, , \\ \text{and } \boldsymbol{v}^{\text{div}} \in \mathcal{H}^{\text{div}}(\mathbb{R}^2) &= \left\{ \boldsymbol{v} \in (\mathbf{L}^2(\mathbb{R}^2))^2 \, , \ \text{div}(\boldsymbol{v}) = \partial_{x_1} v_1 + \partial_{x_2} v_2 = 0 \right\} \, . \end{aligned} \tag{4.1}$$

Figure 4.1 shows the Helmholtz decomposition applied to a sample vector field. The irrotational component $\boldsymbol{v}^{\text{curl}}$ Fig. 4.1b corresponds to sources or sinks in the flow, whereas the solenoidal component $\boldsymbol{v}^{\text{div}}$ Fig. 4.1c is related to vortical structures. Here we considered null border condition at infinity for field $\boldsymbol{v}$; otherwise, the Helmholtz decomposition is defined up to an harmonic function, which encodes non-null boundary conditions.



(a) $\boldsymbol{v} \in (\mathbf{L}^2(\mathbb{R}^2))^2$ (b) $\boldsymbol{v}^{\text{curl}} \in \mathcal{H}^{\text{curl}}(\mathbb{R}^2)$ (c) $\boldsymbol{v}^{\text{div}} \in \mathcal{H}^{\text{div}}(\mathbb{R}^2)$

Figure 4.1: Vector field (a) is the sum of an *irrotational* vector field (b), i.e. curl-free, and a *solenoidal* vector field (c), i.e. divergence-free. Color intensity codes for vectors magnitude.

In the incompressible Navier-Stokes equations (1.2), the constant fluid density implies a divergence-free field. Then, back to our fluid motion estimation context, it might be particularly advantageous to ensure that solutions given by the optical flow problem respect this null-divergence constraint. Let us now see how, by building bases of $\mathcal{H}^{\mathrm{div}}(\mathbb{R}^2)$ or $\mathcal{H}^{\mathrm{curl}}(\mathbb{R}^2)$, it is possible to explicitly consider such divergence-free (or similarly, curl-free) vector fields.

2D divergence-free vector fields are generated by *stream functions*: if there exists a scalar field z such that $\boldsymbol{v} = \mathbf{curl}(z)$ [1], then $\mathrm{div}(\boldsymbol{v}) = \nabla \cdot (\partial_{x_2} z, -\partial_{x_1} z)^{\boldsymbol{T}} = 0$, so that $\boldsymbol{v} \in \mathcal{H}^{\mathrm{div}}(\mathbb{R}^2)$. From there, space $\mathcal{H}^{\mathrm{div}}(\mathbb{R}^2)$ may also be defined from the **curl** of stream functions:

$$\mathcal{H}^{\mathrm{div}}(\mathbb{R}^2) = \left\{ \boldsymbol{v} = \mathbf{curl}(z) \in (\mathbf{L}^2(\mathbb{R}^2))^2 \,,\; z \in H^1(\mathbb{R}^2) \right\}, \tag{4.2}$$

where $H^1(\mathbb{R}^2)$ is the Sobolev space of functions in $\mathbf{L}^2(\mathbb{R}^2)$ whose first-order derivatives also belong to $\mathbf{L}^2(\mathbb{R}^2)$. Since $\boldsymbol{v} = \mathbf{curl}(z)$ is a combination of theses first-order derivatives, this ensures $\boldsymbol{v} \in (\mathbf{L}^2(\mathbb{R}^2))^2$. A multiresolution analysis of $\mathcal{H}^{\mathrm{div}}(\mathbb{R}^2)$ can therefore be built by taking the **curl** of an usual multiresolution analysis of $H^1(\mathbb{R}^2)$. The construction of divergence-free bases proceeds as follows: first, by looking for MRAs of $H^1(\mathbb{R}^2)$ that are, in some sense, "compatible" with the derivation. Then by applying the **curl** operator and expressing the basis functions of $\mathcal{H}^{\mathrm{div}}(\mathbb{R}^2)$. Finally, by relating coefficients of the divergence-free basis to "regular" basis coefficients, in order to simplify implementation. For convenience reasons, note that the following construction is based on the anisotropic wavelet transform (Section 3.1.4); elements for the isotropic transform can be found in [22].

4.1.2 Preliminary Results – Wavelet derivatives

Applying the **curl** operator to a wavelet basis gives rise to derivatives of scaling and wavelet functions. In order to build multiresolution bases of $\mathcal{H}^{\mathrm{div}}(\mathbb{R}^2)$, we must ensure that those derivatives still respect some properties. The following results from Lemarié-Rieusset [25] enlighten the nature of such derivatives.

Let $(\varphi^1, \tilde{\varphi}^1)$ be a pair of biorthogonal scaling functions associated to biorthogonal wavelets $(\psi^1, \tilde{\psi}^1)$, with $\varphi^1 \in \mathcal{C}^{1+\epsilon}(\mathbb{R}), \epsilon > 0$. Then there exists another couple of biorthogonal scaling functions $(\varphi^0, \tilde{\varphi}^0)$ and wavelets $(\psi^0, \tilde{\psi}^0)$ such that:

$$\begin{aligned}
&\frac{d\varphi^1}{dx}(x) = \varphi^0(x) - \varphi^0(x-1)\,; \\
&\frac{d\tilde{\varphi}^0}{dx}(x) = \tilde{\varphi}^1(x) - \tilde{\varphi}^1(x-1)\,; \\
&\psi^1(x) = 4 \int_{-\infty}^{x} \psi^0(t)dt\,; \\
&\tilde{\psi}^0(x) = -4 \int_{-\infty}^{x} \tilde{\psi}^1(t)dt\,.
\end{aligned} \tag{4.3}$$

As a consequence, the derivative of a scaling function corresponds to a finite difference of another scaling function; the derivative of a wavelet is another wavelet. Moreover, BMRAs $\{V_j^1, \tilde{V}_j^1\}_j$ and $\{V_j^0, \tilde{V}_j^0\}_j$ of $\mathbf{L}^2(\mathbb{R})$ associated to the biorthogonal functions introduced in (4.3) also verify the following properties [24]:

$$\begin{aligned}
&\frac{d}{dx} V_j^1 = V_j^0\,, \\
&\tilde{V}_j^0 = \int_{-\infty}^{x} \tilde{V}_j^1\,,
\end{aligned} \tag{4.4}$$

$$\begin{aligned}
&\frac{d}{dx} \mathcal{P}_j^1(f) = \mathcal{P}_j^0(\frac{df}{dx})\,, \\
&\frac{d}{dx} \tilde{\mathcal{P}}_j^0(f) = \tilde{\mathcal{P}}_j^1(\frac{df}{dx})\,, \quad \forall f \in H^1(\mathbb{R})\,.
\end{aligned} \tag{4.5}$$

1. The curl of scalar field z is given by $(\partial_{x_2} z, -\partial_{x_1} z)^{\boldsymbol{T}}$ – see Notations page ix.

where $\mathcal{P}_j^i(f)$ is the projection of f onto V_j^i.

From scaling functions φ^1 and φ^0 previously related by (4.3), let us now consider their associated one-dimensional MRA of $\mathbf{L}^2(\mathbb{R})$: $\{V_j^1\}_{j\in\mathbb{Z}}$ and $\{V_j^0\}_{j\in\mathbb{Z}}$. Using tensor product like in Section 3.1.4, we build:

$$V_j^1 \otimes V_j^0 = \text{span}\left\{\varphi_{j,k_1}^1(x_1)\varphi_{j,k_2}^0(x_2)\,;\ k_1,k_2\in\mathbb{Z}\right\}\ ,\ j\in\mathbb{Z} \tag{4.6}$$

or

$$V_j^0 \otimes V_j^1 = \text{span}\left\{\varphi_{j,k_1}^0(x_1)\varphi_{j,k_2}^1(x_2)\,;\ k_1,k_2\in\mathbb{Z}\right\}\ ,\ j\in\mathbb{Z}\,. \tag{4.7}$$

The sequence of nested spaces made up from either (4.6) or (4.7) is a multiresolution analysis of $\mathbf{L}^2(\mathbb{R}^2)$. Note that having space V^1 associated to dimension x_1 different from V^0 associated to x_2, contrary to Section 3.1.4, does not restrain from building fast transform algorithms: taking $\{V_j^1 \otimes V_j^0\}_j$ as an example, a fast decomposition is set up using filters associated to $(\tilde{\varphi}^1, \tilde{\psi}^1)$ for x_1-direction and $(\tilde{\varphi}^0, \tilde{\psi}^0)$ for x_2. Similarly, the fast reconstruction will use filters from (φ^1, ψ^1) for x_1 and (φ^0, ψ^0) for x_2.

4.1.3 Multiresolution Analyses of $\mathcal{H}^{\mathbf{div}}(\mathbb{R}^2)$

From (4.2), null-divergence space $\mathcal{H}^{\text{div}}(\mathbb{R}^2)$ is obtained from the **curl** of $H^1(\mathbb{R}^2)$. Let us consider a multiresolution analysis of $H^1(\mathbb{R}^2)$ generated by nested spaces $V_j^a \otimes V_j^b$, with $V^a \neq V^b$. Taking its **curl** leads to:

$$\mathbf{curl}(V_j^a \otimes V_j^b) = \begin{pmatrix} V_j^a \otimes (V_j^b)' \\ -(V_j^a)' \otimes V_j^b \end{pmatrix}. \tag{4.8}$$

It follows from (4.8) that manipulating scaling and wavelet functions in $\mathbf{curl}(V_j^a \otimes V_j^b)$, with $a \neq b$, requires to use filters associated to the four one-dimensional MRA $\{V_j^a\}_j$, $\{(V_j^a)'\}_j$, $\{V_j^b\}_j$, $\{(V_j^b)'\}_j$, of which filters of $\{(V_j^a)'\}_j$, and $\{(V_j^b)'\}_j$ are a priori unknown. Fortunately, from (4.4), we have

$$\mathbf{curl}(V_j^1 \otimes V_j^1) \subset (V_j^1 \otimes V_j^0) \times (V_j^0 \otimes V_j^1) \triangleq \mathbf{V}_j^{\text{div}}\,, \tag{4.9}$$

where space $\mathbf{V}_j^{\text{div}}$ preserves the null-divergence property. Indeed, using (4.4),

$$\begin{aligned} \forall\, \boldsymbol{v} \in \mathcal{H}^{\text{div}}(\mathbb{R}^2),\ \text{div}(\mathbf{P}_j(\boldsymbol{v})) = \text{P}_j(\text{div}(\boldsymbol{v})) = 0\,, \\ \text{where } \mathbf{P}_j(\boldsymbol{v}) \triangleq (\mathcal{P}_j^1 \otimes \mathcal{P}_j^0) \times (\mathcal{P}_j^0 \otimes \mathcal{P}_j^1) \\ \text{and } \text{P}_j(\boldsymbol{v}) \triangleq \mathcal{P}_j^0 \otimes \mathcal{P}_j^0\,. \end{aligned} \tag{4.10}$$

Hence the divergence-free approximation spaces, obtained from $\mathbf{curl}(V_j^1 \otimes V_j^1)$ where V_1 and its associated functions φ^1, ψ^1 were chosen according to (4.3), are given by:

$$\mathbf{V}_j^{\text{div}} = \text{span}\left\{\boldsymbol{\varphi}_{j,\boldsymbol{k}}^{\text{div}}\,,\ \boldsymbol{k}\in\mathbb{Z}^2\right\}\ ,\ \forall\, j\in\mathbb{Z}\,, \tag{4.11}$$

with basis functions

$$\boldsymbol{\varphi}_{j,\boldsymbol{k}}^{\text{div}} = \mathbf{curl}\left(\varphi_{j,k_1}^1 \otimes \varphi_{j,k_2}^1\right) = \begin{pmatrix} \varphi_{j,k_1}^1 \otimes (\varphi_{j,k_2}^1)' \\ -(\varphi_{j,k_1}^1)' \otimes \varphi_{j,k_2}^1 \end{pmatrix}. \tag{4.12}$$

Similarly, the corresponding *anisotropic* detail spaces are given by:

$$\mathbf{W}_{\boldsymbol{j}}^{\text{div}} = \text{span}\left\{\boldsymbol{\psi}_{\boldsymbol{j},\boldsymbol{k}}^{\text{div}}\,,\ \boldsymbol{k}\in\mathbb{Z}^2\right\}\ ,\ \forall\, \boldsymbol{j}\in\mathbb{Z}^2\,, \tag{4.13}$$

with

$$\boldsymbol{\psi}_{\boldsymbol{j},\boldsymbol{k}}^{\text{div}} = \mathbf{curl}\left(\psi_{j_1,k_1}^1 \otimes \psi_{j_2,k_2}^1\right) = \begin{pmatrix} \psi_{j_1,k_1}^1 \otimes (\psi_{j_2,k_2}^1)' \\ -(\psi_{j_1,k_1}^1)' \otimes \psi_{j_2,k_2}^1 \end{pmatrix}. \tag{4.14}$$

Moreover, thanks to relations (4.3) between wavelet functions derivative, previous expression (4.14) further simplifies to give:

$$\boldsymbol{\psi}_{\boldsymbol{j},\boldsymbol{k}}^{\text{div}} = \begin{pmatrix} (2^{-j_2+2})\psi_{j_1,k_1}^1 \otimes \psi_{j_2,k_2}^0 \\ -(2^{-j_1+2})\psi_{j_1,k_1}^0 \otimes \psi_{j_2,k_2}^1 \end{pmatrix}. \tag{4.15}$$

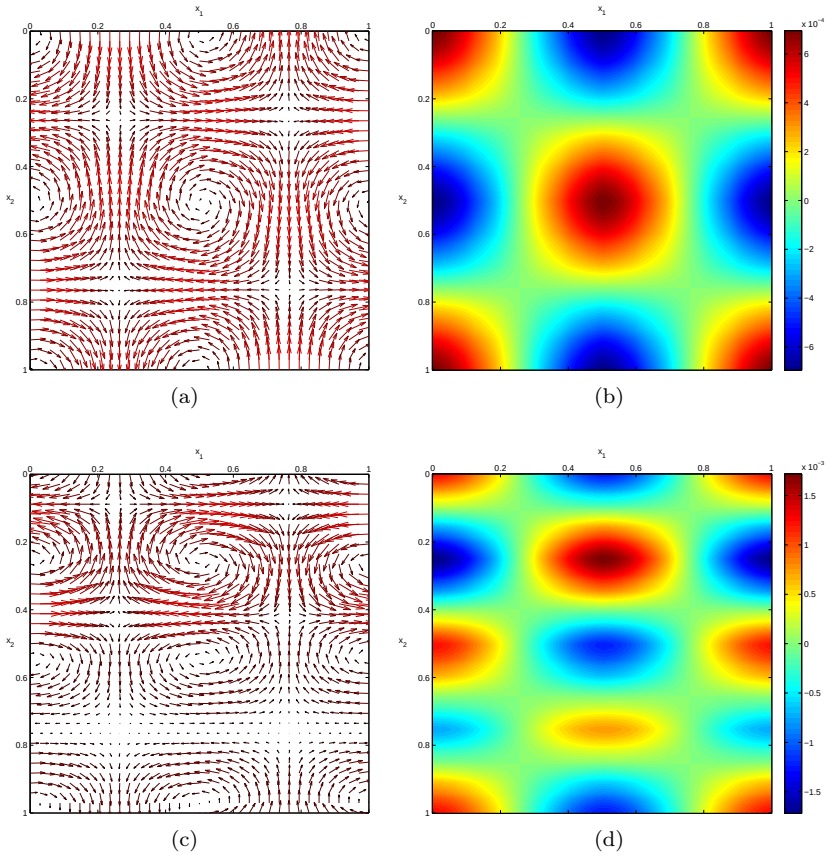


Figure 4.2: Examples of ($[0,1]$-periodized) divergence-free basis functions $\boldsymbol{\psi}^{\mathrm{div}}_{\boldsymbol{j},\boldsymbol{k}}$, obtained from Coiflets with $m = 10$ vanishing moments. *Left* is the vector field generated by the vector-wavelet, *right* is its corresponding vorticity. *Upper row* has $\boldsymbol{j} = (0,0)$ and $\boldsymbol{k} = (0,0)$; *bottom row* has $\boldsymbol{j} = (0,-1)$ and $\boldsymbol{k} = (0,0)$. These functions represent a pattern of several counter-rotative vortices.

Figure 4.2 shows examples of wavelets $\boldsymbol{\psi}^{\mathrm{div}}_{\boldsymbol{j},\boldsymbol{k}}$. We finally obtain bases for $\mathcal{H}^{\mathrm{div}}(\mathbb{R}^2)$:

$$\forall\, \boldsymbol{v} \in \mathcal{H}^{\mathrm{div}}(\mathbb{R}^2)\,, \quad \boldsymbol{v} = \sum_{\boldsymbol{j},\boldsymbol{k}\in\mathbb{Z}^2} d^{\mathrm{div}}_{\boldsymbol{j},\boldsymbol{k}} \boldsymbol{\psi}^{\mathrm{div}}_{\boldsymbol{j},\boldsymbol{k}}\,, \quad \text{with } d^{\mathrm{div}}_{\boldsymbol{j},\boldsymbol{k}} \triangleq \left\langle \boldsymbol{v}\,;\, \tilde{\boldsymbol{\psi}}^{\mathrm{div}}_{\boldsymbol{j},\boldsymbol{k}} \right\rangle_{\mathbf{L}^2(\mathbb{R}^2)\times\mathbf{L}^2(\mathbb{R}^2)}\,. \tag{4.16}$$

So far, we have derived wavelet bases for divergence-free space $\mathcal{H}^{\mathrm{div}}(\mathbb{R}^2)$, so that any divergence-free vector field $\boldsymbol{v} \in \mathcal{H}^{\mathrm{div}}(\mathbb{R}^2)$ has a unique decomposition following (4.16). Basis functions $\boldsymbol{\psi}^{\mathrm{div}}_{\boldsymbol{j},\boldsymbol{k}}$ are *vector wavelets* in $\mathbf{L}^2(\mathbb{R}^2) \times \mathbf{L}^2(\mathbb{R}^2)$; their associated coefficients $d^{\mathrm{div}}_{\boldsymbol{j},\boldsymbol{k}}$ are *scalar* values. It is important to remember that from our construction, these coefficients are in fact at the same time the representation of $\boldsymbol{v}$'s stream function z:

$$z = \sum_{\boldsymbol{j},\boldsymbol{k}} d^{\mathrm{div}}_{\boldsymbol{j},\boldsymbol{k}} \left(\psi^1_{j_1,k_1} \otimes \psi^1_{j_2,k_2}\right)\,, \quad \text{with } d^{\mathrm{div}}_{\boldsymbol{j},\boldsymbol{k}} \triangleq \left\langle z\,;\, \tilde{\psi}^1_{j_1,k_1} \otimes \tilde{\psi}^1_{j_2,k_2} \right\rangle_{\mathbf{L}^2(\mathbb{R}^2)}\,. \tag{4.17}$$

Only the choice of basis functions (scalar $\{\psi^1_{j_1,k_1} \otimes \psi^1_{j_2,k_2}\}_{\boldsymbol{j},\boldsymbol{k}}$ for z, or vector $\{\boldsymbol{\psi}^{\mathrm{div}}_{\boldsymbol{j},\boldsymbol{k}}\}_{\boldsymbol{j},\boldsymbol{k}}$ for $\boldsymbol{v}$) gives either the stream function z or the corresponding $\boldsymbol{v}$ such that $\boldsymbol{v} = \mathrm{curl}(z)$. The last step shows how those coefficients can be in practice computed from regular *scalar* wavelet transforms, resulting in a somewhat simple implementation of the divergence-free wavelet transform.

4.1.4 Fast Divergence-Free Transforms

We have seen in (4.16) and (4.17) that our divergence-free motion field $\boldsymbol{v}$ is represented by a set of scalar coefficients $\{d^{\mathrm{div}}_{\boldsymbol{j},\boldsymbol{k}}\}_{\boldsymbol{j},\boldsymbol{k}}$, associated to vector wavelet basis $\{\boldsymbol{\psi}^{\mathrm{div}}_{\boldsymbol{j},\boldsymbol{k}}\}_{\boldsymbol{j},\boldsymbol{k}}$ (4.15). In practice, it

is much more convenient to handle the two motion components v_i, $i = 1, 2$, separately. Therefore we seek a relation between divergence-free coefficients $\{d^{\text{div}}_{\boldsymbol{j},\boldsymbol{k}}\}_{\boldsymbol{j},\boldsymbol{k}}$ and "regular" coefficients $\{d^{i}_{\boldsymbol{j},\boldsymbol{k}}\}_{\boldsymbol{j},\boldsymbol{k}}$ associated to single component v_i.

The sequence of spaces $\mathbf{V}_j^{\text{div}} = (V_j^1 \otimes V_j^0) \times (V_j^0 \otimes V_j^1)$, $j \in \mathbb{Z}$, is a MRA of $\mathbf{L}^2(\mathbb{R}^2) \times \mathbf{L}^2(\mathbb{R}^2)$ that preserves the null-divergence property – (4.9). From definition (4.1), $\mathcal{H}^{\text{div}}(\mathbb{R}^2) \subset (\mathbf{L}^2(\mathbb{R}^2))^2$, so that any $\boldsymbol{v}$ in divergence-free space can be decomposed on a basis of $(\mathbf{L}^2(\mathbb{R}^2))^2$. The standard anisotropic vector wavelets associated to MRA $\{\mathbf{V}_j^{\text{div}}\}_j$ are:

$$\boldsymbol{\psi}^1_{\boldsymbol{j},\boldsymbol{k}} = \begin{pmatrix} \psi^1_{j_1,k_1} \otimes \psi^0_{j_2,k_2} \\ 0 \end{pmatrix}, \quad \boldsymbol{\psi}^2_{\boldsymbol{j},\boldsymbol{k}} = \begin{pmatrix} 0 \\ \psi^0_{j_1,k_1} \otimes \psi^1_{j_2,k_2} \end{pmatrix}. \tag{4.18}$$

Therefore any $\boldsymbol{v} \in \mathcal{H}^{\text{div}}(\mathbb{R}^2)$ satisfies:

$$\boldsymbol{v} = \begin{pmatrix} v_1 \\ v_2 \end{pmatrix} = \sum_{\boldsymbol{j},\boldsymbol{k} \in \mathbb{Z}^2} d^1_{\boldsymbol{j},\boldsymbol{k}} \boldsymbol{\psi}^1_{\boldsymbol{j},\boldsymbol{k}} + \sum_{\boldsymbol{j},\boldsymbol{k} \in \mathbb{Z}^2} d^2_{\boldsymbol{j},\boldsymbol{k}} \boldsymbol{\psi}^2_{\boldsymbol{j},\boldsymbol{k}}\,, \tag{4.19}$$

from which a simple identification gives

$$\begin{aligned} v_1 &= \sum_{\boldsymbol{j},\boldsymbol{k} \in \mathbb{Z}^2} d^1_{\boldsymbol{j},\boldsymbol{k}} \psi^1_{j_1,k_1} \otimes \psi^0_{j_2,k_2}\,, \\ v_2 &= \sum_{\boldsymbol{j},\boldsymbol{k} \in \mathbb{Z}^2} d^2_{\boldsymbol{j},\boldsymbol{k}} \psi^0_{j_1,k_1} \otimes \psi^1_{j_2,k_2}\,. \end{aligned} \tag{4.20}$$

Then, from $\boldsymbol{\psi}^{\text{div}}_{\boldsymbol{j},\boldsymbol{k}}$ expression (4.15) we obtain the relation:

$$\boldsymbol{\psi}^{\text{div}}_{\boldsymbol{j},\boldsymbol{k}} = (2^{-j_2+2})\boldsymbol{\psi}^1_{\boldsymbol{j},\boldsymbol{k}} - (2^{-j_1+2})\boldsymbol{\psi}^2_{\boldsymbol{j},\boldsymbol{k}}\,. \tag{4.21}$$

And finally:

$$d^1_{\boldsymbol{j},\boldsymbol{k}} = 2^{-j_2+2} d^{\text{div}}_{\boldsymbol{j},\boldsymbol{k}}\,, \tag{4.22}$$

$$d^2_{\boldsymbol{j},\boldsymbol{k}} = -2^{-j_1+2} d^{\text{div}}_{\boldsymbol{j},\boldsymbol{k}}\,, \tag{4.23}$$

$$d^{\text{div}}_{\boldsymbol{j},\boldsymbol{k}} = \frac{1}{2^{-j_1+2} + 2^{-j_2+2}} (d^1_{\boldsymbol{j},\boldsymbol{k}} - d^2_{\boldsymbol{j},\boldsymbol{k}})\,. \tag{4.24}$$

From (4.24), null-divergence coefficients $d^{\text{div}}_{\boldsymbol{j},\boldsymbol{k}}$ associated to vector wavelets $\boldsymbol{\psi}^{\text{div}}_{\boldsymbol{j},\boldsymbol{k}}$, are given by a linear combination of coefficients $d^1_{\boldsymbol{j},\boldsymbol{k}}$ and $d^2_{\boldsymbol{j},\boldsymbol{k}}$ resulting from the regular scalar wavelet decomposition of fields v_1, v_2; and reciprocally. As a consequence, decomposition and reconstruction for the divergence-free bases are simply performed using the corresponding scalar transforms with appropriate filters.

The theoretical construction of divergence-free bases is finally relatively simple to handle: first, two sets of biorthogonal wavelets are required, $(\psi^1, \tilde{\psi}^1)$ and $(\psi^0, \tilde{\psi}^0)$, linked by derivation/integration relations (4.3). Then divergence-free coefficients are computed from the two sets of regular coefficients corresponding to the two scalar components of the motion, and reciprocally, using relations (4.22)–(4.24). Only the wavelet bases associated to these scalar components v_i are somewhat unusual, for they use a different function for each direction x_i – Eq. (4.20) – contrary to the classical tensorial construction of Section 3.1.4. The practical implementation of these bases is however slightly more complex. The computation of conjugate mirror filters h, g associated to the two sets of biorthogonal wavelets is to be clarified, as well as the way to deal properly with finite signals. These elements are extensively detailed in Appendix A.

4.2 Norm Equivalence & Connection Coefficients

4.2.1 Norm Equivalence

Wavelet function derivative properties, introduced above in Section 4.1.2 with divergence-free bases, lead to an equivalence relation between the $\mathbf{L}^2$-norm of the n-th derivative of a given function

and a weighted $\mathbf{l}^2$-norm of coefficients representing this function in a given wavelet basis. Detailed proofs of following results can be found in [24].

Let $(\psi^1, \tilde{\psi}^1)$ be a pair of at least n times differentiable biorthogonal wavelet functions. Applying n times the wavelet derivation property (4.3) leads to another couple of biorthogonal wavelets $(\psi^0, \tilde{\psi}^0)$ such that:

$$\frac{d^n \psi^1(x)}{dx^n} = 4^n \psi^0 \quad \text{and} \quad \frac{d^n \tilde{\psi}^0(x)}{dx^n} = (-4)^n \tilde{\psi}^1 \,. \tag{4.25}$$

For any sufficiently smooth function $f \in \mathbf{L}^2(\mathbb{R})$, we consider the projection of its n-th derivative on the wavelet basis having ψ^0 as mother wavelet:

$$\frac{d^n f(x)}{dx^n} = \sum_{j,k} \bar{d}_{j,k} \psi^0_{j,k}(x) \,, \quad \text{where } \bar{d}_{j,k} = \left\langle \frac{d^n f}{dx^n} \,;\, \tilde{\psi}^0 \right\rangle_{\mathbf{L}^2} \,. \tag{4.26}$$

Then, $\{\psi^0_{j,k}\}_{j,k}$ being a Riesz basis, the norm equivalence holds. There exists two constants $0 < C_1 \leq C_2 < +\infty$ such that

$$C_1 \left\| \bar{d}_{j,k} \right\|_{\mathbf{l}^2} \leq \left\| \frac{d^n f}{dx^n} \right\|_{\mathbf{L}^2} \leq C_2 \left\| \bar{d}_{j,k} \right\|_{\mathbf{l}^2} \,. \tag{4.27}$$

Using integration by parts and (4.25) leads to:

$$\begin{aligned} \bar{d}_{j,k} &= \int_{\mathbb{R}} \frac{d^n f(x)}{dx^n} \tilde{\psi}^0_{j,k}(x) dx \\ &= (-4)^n 2^{-nj} \int_{\mathbb{R}} f(x) \tilde{\psi}^1_{j,k} dx \\ &= (-4)^n 2^{-nj} d_{j,k} \,, \quad \text{with } d_{j,k} = \left\langle f \,;\, \tilde{\psi}^1_{j,k} \right\rangle_{\mathbf{L}^2} \,. \end{aligned} \tag{4.28}$$

The $(-4)^n$ factor arises from (4.25), whereas the 2^{-nj} comes from the scaling factor in $\tilde{\psi}^0_{j,k}$. Finally, following equivalence holds:

$$\left\| \frac{d^n f}{dx^n} \right\|_{\mathbf{L}^2} \sim \left\| (-4)^n 2^{-nj} d_{j,k} \right\|_{\mathbf{l}^2} \,, \tag{4.29}$$

which involves coefficients of the function f instead of those of its n-th derivative. This relation can be easily extended to 2D signals; it will be used later in Section 6.2.1 to design high-order regularizers.

4.2.2 Wavelet Connection Coefficients

More precise and complex regularization schemes can be designed by transferring derivative computations from the function to the wavelet basis atoms [7].

Using refinement relations (3.13), the autocorrelation of a scaling function $\varphi \in \mathbf{L}^2(\mathbb{R})$ verifies a two-scale relation:

$$\begin{aligned} J^{(0)}_{\varphi}(x) &= \int_{\mathbb{R}} \varphi(y)\varphi(y-x)dy \\ &= \int_{\mathbb{R}} \left(\sqrt{2} \sum_{m \in \mathbb{Z}} h[m] \varphi(2y-m) \right) \left(\sqrt{2} \sum_{m \in \mathbb{Z}} h[n] \varphi(2(y-x)-n) \right) dy \\ &= \sum_{m \in \mathbb{Z}} \sum_{n \in \mathbb{Z}} 2h[m]h[n] \int_{\mathbb{R}} \varphi(2y-m)\varphi(2(y-x)-n)dy \\ &= \sum_{k \in \mathbb{Z}} \sum_{m \in \mathbb{Z}} h[m]h[m-k] \int_{\mathbb{R}} \varphi(y)\varphi(y-2x+k)dy \\ &= \sum_{k \in \mathbb{Z}} i[k] J^{(0)}_{\varphi}(2x-k) \,, \quad \text{with } i[k] = \sum_{m \in \mathbb{Z}} h[m]h[m-k] \,. \end{aligned} \tag{4.30}$$

Similarly, the correlation $J_\varphi^{(n)}$ of a scaling function φ and its n-th order derivative $\varphi^{(n)}$ verifies:

$$\begin{aligned} J_\varphi^{(n)}(x) &= \int_{\mathbb{R}} \varphi(y)\varphi^{(n)}(y-x)dy \\ &= 2^n \sum_{k\in\mathbb{Z}} i[k] J_\varphi^{(n)}(2x-k)\,. \end{aligned} \tag{4.31}$$

If φ has a compact support, so does filter h (Section 3.1.3) and therefore autocorrelation filter i. In order to recover connection coefficients of the form $\langle \varphi_{j,p}\,;\,\varphi_{j,q}^{(n)}\rangle_{\mathbf{L}^2} = J_\varphi^{(n)}(q-p)$, the two relations above may be rewritten as a linear system:

$$H J_\varphi^{(n)} = \frac{1}{2^n} J_\varphi^{(n)}\,, \quad n \geq 0\,, \tag{4.32}$$

where H is a matrix formed with elements $i[k]$ and $J_\varphi^{(n)}$ is the vector of connection coefficients. As a consequence, our connection coefficients $\langle \varphi_{j,p}\,;\,\varphi_{j,q}^{(n)}\rangle_{\mathbf{L}^2}$ are components of the eigenvector of matrix H associated to eigenvalue $1/2^n$, with $n \geq 0$ the derivation order. Finally, similar connection coefficients involving wavelet functions are obtained using the second refinement relation (3.15), e.g. :

$$\begin{aligned} \langle \psi_{0,p}\,;\,\psi_{0,q}\rangle_{\mathbf{L}^2} &= \int_{\mathbb{R}} \psi(y-p)\psi(y-q)dy \\ &= \int_{\mathbb{R}} \psi(y)\psi(y+p-q)dy \\ &= \int_{\mathbb{R}} \left(\sqrt{2}\sum_{m\in\mathbb{Z}} g[m]\varphi(2y-m)\right)\left(\sqrt{2}\sum_{m\in\mathbb{Z}} g[n]\varphi(2(y+p-q)-n)\right)dy \\ &= \sum_{k\in\mathbb{Z}}\sum_{m\in\mathbb{Z}} 2g[m]g[m-k]\int_{\mathbb{R}} \varphi(y)\varphi(y+2(p-q)+k)dy \\ &= \sum_{k\in\mathbb{Z}}\sum_{m\in\mathbb{Z}} 2g[m]g[m-k]J_\varphi^{(0)}(2(q-p)-k)\,. \end{aligned} \tag{4.33}$$

In practice, it can be shown that in order to recover every connection coefficients of a given derivation order n for a given *anisotropic* multiscale basis, it suffices:

(i) to compute these connection coefficients for the fine scale F approximation functions: $\langle \varphi_{F,p}\,;\,\varphi_{F,q}^{(n)}\rangle_{\mathbf{L}^2}$ for $0 \leq p,q < 2^{-F}$ with the eigenvalue problem (4.32);

(ii) to apply the anisotropic forward wavelet transform, up to the chosen coarse scale C, to the matrix $P^{(n)}$ where

$$(P^{(n)})_{p,q} = \langle \varphi_{F,p}\,;\,\varphi_{F,q}^{(n)}\rangle_{\mathbf{L}^2}\,.$$

Should a *biorthogonal basis* be considered (Section 3.1.2), this forward transform has to be done cautiously. For instance, if the chosen biorthogonal transform uses filters $(\tilde{h},\tilde{g})$ for the decomposition and (h,g) for the reconstruction (e.g. in (3.21) and (3.22)), then the decomposition applied to matrix $P^{(n)}$ must be carried out using (h,g) instead of $(\tilde{h},\tilde{g})$, in order to get the appropriate connection coefficients.

Part III

Wavelet-Based Motion Estimator

Chapter 5

Data-Term

Considering turbulence characteristics presented in Section 1.3, optic flow specificities of Section 2.2.3 and wavelet properties of Chapters 3 and 4, adopting a wavelet representation in the context of optic flow estimation may yield the following advantages:

- it provides a multiscale representation which not only echoes to the multiscale turbulent structure, but also constitutes an interesting framework for optic flow *coarse-to-fine* estimation;
- it enables the implementation of various regularization schemes for optic flow, based on motion derivatives.

Indeed some of these ideas have already been explored. Bernard [5] projects the full OFC equation (2.8) onto a multiscale wavelet basis in order to build a coarse-to-fine estimation scheme. No explicit smoothing term is introduced; instead, under the hypothesis of motion consistency on the support of basis functions, he obtains a set of small linear systems to be inverted. An interesting innovation is the use of analytic wavelets: such functions induce less vanishing coefficients than with real wavelets, which largely stabilizes the flow estimation. Wu & Kanade [44] used a different framework: motion components only are projected onto a multiscale wavelet basis; their coefficients are estimated sequentially, from coarse to finer scales, by minimizing the incremental quadratic error of the DFD (2.9), leading to a quadratic functional. At a given scale, previously estimated coefficients are still considered as unknowns, so as to enable their update or correction as the solver progresses towards finer scales. They used the specific Cai-Wang wavelet [8], which is built from a 4-th order B-spline. Its major feature is that coefficients corresponding to a given scale can be computed directly, whereas the conventional transform requires to go through all finer scales coefficients first. As we shall see later, a serious drawback in their approach is the computational efficiency: the computation of the Hessian matrix of their functional, and even more a matrix inversion in their minimization algorithm rapidly become prohibitive as fine scales and/or large images are considered. Furthermore, no regularization mechanism is involved, which may complicate the estimation when applied to low-textured images (such as scalar advection-diffusion, see e.g. Fig. 8.4b). Chen *et al.* [10] consider Horn & Schunck formulation [20]. They project every image-related function of the squared OFC (e.g. $[\partial_{x_1} I_1]^2$), as well as motion components, onto an approximation basis (i.e. non-multiscale, contrary to previous approaches). First-order smoothing terms are computed accurately thanks to the connection coefficients of the chosen basis (Section 4.2.2). A large linear system is assembled, then inverted, in order to minimize the functional. Although being accurate, this approach is computationally expensive, just as Wu & Kanade's. Moreover, and contrary to the two previous methods, the lack of multiresolution scheme may cause issues when dealing with large displacements.

Our proposal shares some concepts with Wu & Kanade and Chen *et al.*: we consider the projection of motion components only, on a multiscale basis. A sequential coarse-to-fine estimation process is designed to handle large displacements without freezing coarse-scale estimates. Regularization schemes based on motion coefficients are proposed, including high-order schemes relying on connection coefficients. This chapter introduces our wavelet representation of the motion field and its integration to optic flow data terms. The derivation of the sequential estimation process follows (Section 2.2.3), and is finally extended to divergence-free bases. Regularization terms are

later tackled in the next chapter.

5.1 Wavelet Representation of the Motion Field

We now consider square images of $2^{-F} \times 2^{-F}$ samples, i.e. at pixel scale F. The observable displacement, unknown of the optic flow problem, is

$$\boldsymbol{v} \in \mathbf{L}^2([0,1]^2) \times \mathbf{L}^2([0,1]^2) \tag{5.1}$$

In Chapter 3, we introduced multiresolution analyses (MRAs) of $\mathbf{L}^2([0,1]^2)$. Let $\mathbf{V}_F = V_F \otimes V_F$ be such a MRA. We consider each component v_i of the motion field to belong to $\mathbf{V}_F$:

$$v_i \in \mathbf{V}_F = V_F \otimes V_F\,,\ i = 1,2 \quad \Rightarrow \quad \boldsymbol{v} \in \mathbf{V}_F \times \mathbf{V}_F \subset \mathbf{L}^2([0,1]^2) \times \mathbf{L}^2([0,1]^2)\,. \tag{5.2}$$

Then, taking a coarse scale $C > F$, we consider a wavelet decomposition of $\mathbf{V}_F$. In the following, we will use the isotropic orthogonal decomposition (3.26), however the approach is very similar with an anisotropic and/or biorthogonal decomposition:

$$\mathbf{V}_F = V_F \otimes V_F = (V_C \otimes V_C) \oplus \bigoplus_{F+1 \le j \le C} (V_j \otimes W_j) \oplus (W_j \otimes V_j) \oplus (W_j \otimes W_j)\,. \tag{5.3}$$

For each motion component v_i, Θ_i is the vector containing *all coefficients* (i.e. approximations and details) resulting from the projection of v_i on the set of spaces in (5.3) – see Section 3.1.3 and Figure 5.1 for a graphical illustration. Then we denote by $\boldsymbol{\Theta}$ the superset of all coefficients related to $\boldsymbol{v}$:

$$\boldsymbol{\Theta} = \begin{pmatrix} \Theta_1 \\ \Theta_2 \end{pmatrix}\,. \tag{5.4}$$

Each component v_i is obtained from Θ_i by the appropriate inverse wavelet transform:

$$\forall \boldsymbol{x} \in \Omega,\ v_i(\boldsymbol{x}) = \Phi^{\boldsymbol{T}}(\boldsymbol{x})\Theta_i\,, \quad i = 1,2\,, \tag{5.5}$$

where $\Phi(\boldsymbol{x})$ is a vector containing values of all wavelet basis functions at current point $\boldsymbol{x}$. In order to simplify notations, we also define the "vector" reconstruction:

$$\boldsymbol{v}(\boldsymbol{x}) = \boldsymbol{\Phi}^{\boldsymbol{T}}(\boldsymbol{x})\boldsymbol{\Theta}\,, \tag{5.6}$$

where operator $\boldsymbol{\Phi}$ writes:

$$\boldsymbol{\Phi}^{\boldsymbol{T}}(\boldsymbol{x}) = \begin{pmatrix} \Phi^{\boldsymbol{T}}(\boldsymbol{x}) & 0 \cdots 0 \\ 0 \cdots 0 & \Phi^{\boldsymbol{T}}(\boldsymbol{x}) \end{pmatrix}\,. \tag{5.7}$$

In representations (5.5) and (5.6), operators Φ and $\boldsymbol{\Phi}$ are a vector of size $2^{-2F} \times 1$ and a matrix of size $2^{-2F+1} \times 2$, respectively. The reader should keep in mind that in practice, every decomposition/reconstruction operation involves the efficient filterbanks implementation presented in Section 3.1.3.

The unknown variables of the optic flow problem correspond to the set of coefficients $\boldsymbol{\Theta}$ representing the velocity field $\boldsymbol{v}$ in the chosen wavelet basis. Accordingly, the original problem (2.6) is replaced by:

$$\begin{cases} \hat{\boldsymbol{v}}(\boldsymbol{x}) = \boldsymbol{\Phi}^{\boldsymbol{T}}(\boldsymbol{x})\hat{\boldsymbol{\Theta}}\,,\ \forall \boldsymbol{x} \in \Omega \\ \hat{\boldsymbol{\Theta}} = \underset{\boldsymbol{\Theta}}{\arg\min}\, J_{\text{data}}(I, \boldsymbol{\Theta}) + \alpha J_{\text{reg}}(\boldsymbol{\Theta})\,. \end{cases} \tag{5.8}$$

5.2 Data Models

Wavelet-based data models are simply obtained by inserting our wavelet representation (5.6) into usual data models introduced in Section 2.2.3.

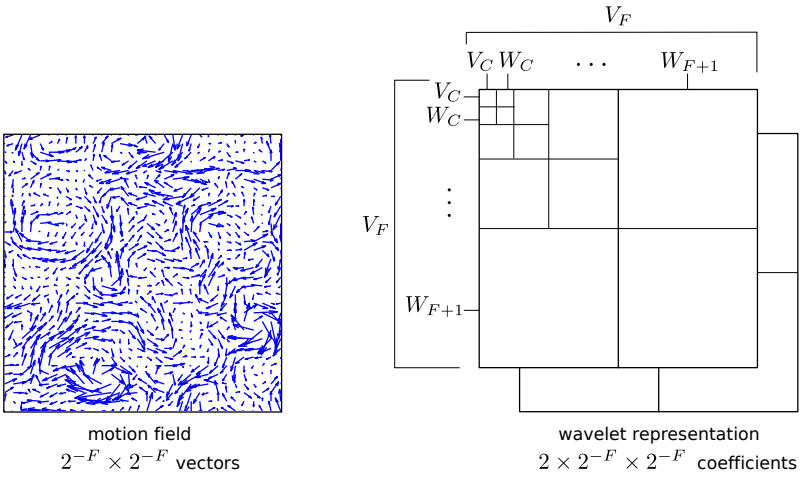


Figure 5.1: Illustration of the isotropic wavelet representation of the motion field $\boldsymbol{v}$.

5.2.1 DFD model

From DFD equation (2.9), the wavelet DFD becomes:

$$\text{(wav-DFD)} \quad I_0(\boldsymbol{x}) - I_1(\boldsymbol{x} + \boldsymbol{\Phi}^{\boldsymbol{T}}(\boldsymbol{x})\boldsymbol{\Theta}) = 0\,. \tag{5.9}$$

Using a quadratic penalty, the corresponding data functional writes:

$$J_{\text{data}}(I, \boldsymbol{\Theta}) = \frac{1}{2}\int_{\Omega} \left[I_0(\boldsymbol{x}) - I_1(\boldsymbol{x} + \boldsymbol{\Phi}^{\boldsymbol{T}}(\boldsymbol{x})\boldsymbol{\Theta})\right]^2 d\boldsymbol{x}. \tag{5.10}$$

Its gradient is required by the optimization process. Let $\theta_{i,p}$ be the p-th coefficient of vector Θ_i related to component v_i, and ϕ_p its corresponding basis function (either a scale function if $\theta_{i,p}$ is an approximation coefficient, or a wavelet function otherwise – see (3.19) and (3.20)):

$$\theta_{i,p} = \langle v_i\,;\, \phi_p \rangle_{\mathbf{L}^2([0,1]^2)}\,. \tag{5.11}$$

The partial derivative of functional (5.10) with respect to $\theta_{i,p}$ writes:

$$\begin{aligned}
\frac{\partial J_{\text{data}}(\boldsymbol{\Theta})}{\partial \theta_{i,p}} &= \int_{\Omega} \left[I_0(\boldsymbol{x}) - I_1(\boldsymbol{x} + \boldsymbol{\Phi}^{\boldsymbol{T}}(\boldsymbol{x})\boldsymbol{\Theta})\right] \frac{\partial I_1(\boldsymbol{x} + \boldsymbol{\Phi}^{\boldsymbol{T}}(\boldsymbol{x})\boldsymbol{\Theta})}{\partial x_i} \frac{\partial (x_i + \Phi^{\boldsymbol{T}}(\boldsymbol{x})\Theta_i)}{\partial \theta_{i,p}} d\boldsymbol{x} \\
&= \int_{\Omega} \left[I_0(\boldsymbol{x}) - I_1(\boldsymbol{x} + \boldsymbol{\Phi}^{\boldsymbol{T}}(\boldsymbol{x})\boldsymbol{\Theta})\right] \frac{\partial I_1(\boldsymbol{x} + \boldsymbol{\Phi}^{\boldsymbol{T}}(\boldsymbol{x})\boldsymbol{\Theta})}{\partial x_i} \phi_p(\boldsymbol{x}) d\boldsymbol{x} \\
&= \left\langle \left[I_0(\cdot) - I_1(\cdot + \boldsymbol{\Phi}^{\boldsymbol{T}}(\cdot)\boldsymbol{\Theta})\right] \frac{\partial I_1(\cdot + \boldsymbol{\Phi}(\cdot)^{\boldsymbol{T}}\boldsymbol{\Theta})}{\partial x_i}\,;\, \phi_p \right\rangle_{\mathbf{L}^2([0,1]^2)}\,.
\end{aligned} \tag{5.12}$$

From (5.12), one notices that the gradient component $\dfrac{\partial J_{\text{data}}(\boldsymbol{\Theta})}{\partial \theta_{i,p}}$ corresponds to the p-th coefficient resulting from the projection of

$$\left[I_0(\cdot) - I_1(\cdot + \boldsymbol{\Phi}^{\boldsymbol{T}}(\cdot)\boldsymbol{\Theta})\right] \frac{\partial I_1(\cdot + \boldsymbol{\Phi}(\cdot)^{\boldsymbol{T}}\boldsymbol{\Theta})}{\partial x_i} \tag{5.13}$$

onto the considered wavelet basis. Hence, gradient components of DFD functional (5.10) are simply given by the wavelet decomposition of the two terms given by (5.13) with $i = 1, 2$.

5.2.2 OFC model

Similarly, inserting wavelet representation (5.6) into OFC model (2.8) leads to:

$$\text{(wav-OFC)} \quad I_1(\boldsymbol{x}) - I_0(\boldsymbol{x}) + \left(\boldsymbol{\Phi}^{\boldsymbol{T}}(\boldsymbol{x})\boldsymbol{\Theta}\right) \cdot \nabla I_1(\boldsymbol{x}) = 0\,. \tag{5.14}$$

Then the corresponding data functional writes:

$$J_{\text{data}}(I,\boldsymbol{\Theta}) = \frac{1}{2}\int_{\Omega}\left[I_1(\boldsymbol{x}) - I_0(\boldsymbol{x}) + \left(\boldsymbol{\Phi}^{\boldsymbol{T}}(\boldsymbol{x})\boldsymbol{\Theta}\right)\cdot\nabla I_1(\boldsymbol{x})\right]^2 d\boldsymbol{x}. \tag{5.15}$$

The partial derivative of this functional (5.15) with respect to a given coefficient $\theta_{i,p}$ writes:

$$\begin{aligned}\frac{\partial J_{\text{data}}(\boldsymbol{\Theta})}{\partial\theta_{i,p}} &= \int_{\Omega}\left[I_1(\boldsymbol{x}) - I_0(\boldsymbol{x}) + \left(\boldsymbol{\Phi}^{\boldsymbol{T}}(\boldsymbol{x})\boldsymbol{\Theta}\right)\cdot\nabla I_1(\boldsymbol{x})\right]\frac{\partial I_1(\boldsymbol{x})}{\partial x_i}\phi_p(\boldsymbol{x})d\boldsymbol{x} \\ &= \left\langle\left[I_1(\boldsymbol{x}) - I_0(\boldsymbol{x}) + \left(\boldsymbol{\Phi}^{\boldsymbol{T}}(\boldsymbol{x})\boldsymbol{\Theta}\right)\cdot\nabla I_1(\boldsymbol{x})\right]\frac{\partial I_1(\boldsymbol{x})}{\partial x_i}\,;\,\phi_p\right\rangle_{\mathbf{L}^2([0,1]^2)}.\end{aligned} \tag{5.16}$$

Just like with DFD gradient (5.12), one identifies a wavelet coefficient. Again, gradient components of OFC functional (5.15) are given by the coefficients obtained by the projection, on the chosen wavelet basis, of the two terms:

$$\left[I_1(\boldsymbol{x}) - I_0(\boldsymbol{x}) + \left(\boldsymbol{\Phi}^{\boldsymbol{T}}(\boldsymbol{x})\boldsymbol{\Theta}\right)\cdot\nabla I_1(\boldsymbol{x})\right]\frac{\partial I_1(\boldsymbol{x})}{\partial x_i}, \quad \text{with } i = 1,2\,. \tag{5.17}$$

5.2.3 Sequential Estimation

Up to now, improvements brought by the wavelet formalism are somewhat minor: a simple change of basis, however leading to a nice formula for functional gradients. The first major idea, already introduced by Wu & Kanade [44], consists in taking advantage of the intrinsic multiscale representation given by wavelet analysis to design a sequential, coarse-to-fine motion estimation scheme that does not rely on the usual and heuristic image pyramidal representation described in Section 2.2.3.

Coefficients $\boldsymbol{\Theta}$ are estimated sequentially, scale after scale, starting with coarse approximation coefficients then gradually adding details. At a given scale j with $F \leq j \leq C$, let us denote by $\boldsymbol{\Theta}_{|j}$ the set of all coefficients corresponding to scales coarser than (and including) j, i.e. approximations and details at scales k, where $j \leq k \leq C$. Current unknown $\boldsymbol{\Theta}_{|j}$ is initialized to coefficients estimated at previous scale $j+1$, and to zeros for the new details to be estimated:

$$\boldsymbol{\Theta}_{|j} = \begin{pmatrix}\Theta_{1|j}\\ \Theta_{2|j}\end{pmatrix} = \begin{pmatrix}\hat{\Theta}_{1|j+1}\\ 0\\ \hat{\Theta}_{2|j+1}\\ 0\end{pmatrix}, \tag{5.18}$$

where we assume that

$$\hat{\boldsymbol{\Theta}}_{|j+1} = \begin{pmatrix}\hat{\Theta}_{1|j+1}\\ \hat{\Theta}_{2|j+1}\end{pmatrix} = \underset{\boldsymbol{\Theta}_{|j+1}}{\arg\min}\; J_{\text{data}}(I,\boldsymbol{\Theta}_{|j+1}) + \alpha J_{\text{reg}}(\boldsymbol{\Theta}_{|j+1}) \tag{5.19}$$

result from the estimation process at previous coarser scale $j+1$. Note that from the non-linearity of the DFD data-model, $\hat{\boldsymbol{\Theta}}_{|j+1}$ can be a local minimum instead of the global one. At the end of the step, coarser coefficients have been updated if necessary; new details at scale j (that were initialized to zero) have been estimated. It is then possible to move to finer scale $j-1$ and repeat the process. In other words, solution $\hat{\boldsymbol{v}}$ is sequentially sought within more and more detailed spaces, until finest scale F (pixel) is finally reached: $(\mathbf{V}_C \times \mathbf{V}_C) \subset (\mathbf{V}_{C-1} \times \mathbf{V}_{C-1}) \subset \cdots \subset (\mathbf{V}_F \times \mathbf{V}_F)$. This approach enables to update previous estimates while estimating new details, contrary to usual multiresolution schemes which freeze coarser estimates.

5.2.4 Computational Aspects

So far, some aspects still have to be clarified, starting with boundary conditions.

Boundary Conditions

The handling of boundary conditions has not been really mentioned yet. We consider images I_0, I_1 and motion $\boldsymbol{v}$ to be defined on $[0,1]^2$. Wavelet basis functions might have a support larger than $[0,1]^2$, especially with high numbers of vanishing moments; it is therefore necessary to modify these functions. Three options are available [28]:

- *periodization* of the signal over $[0,1]^2$ (giving an infinite 1-periodic signal);
- *folding* of the signal at $\boldsymbol{x} = (0,0)^{\boldsymbol{T}}$ (giving an infinite 2-periodic signal);
- using *boundary wavelets* whose support stays within $[0,1]^2$.

Building and implementing boundary wavelets is quite complex. The simplest option is the periodization, since it is only requires to change convolutions in fast transforms (Section 3.1.3) into *circular convolutions*. If the signal is non-periodic – as often, in practice – discontinuities at borders $x_i = 0, 1$ create high-amplitude coefficients. The same happens with the folding: signal is made continuous at borders, yet its first derivative is not.

From its experiments, Bernard [5] chose to pad the signal with either symmetric (i.e., folding) or constant values at the boundaries, as much as required by the support of the filters. In this work, we only implemented the periodization, being the simplest way. Our benchmarks being done on pre-existing periodic data (Section 8.1), it was not particularly inopportune. However on real, non-periodic data, consequences of the periodization on estimated motions are clearly visible and can be disadvantageous; it will be illustrated in Section 9.2.

Gradient and Warping Computation

We saw in (5.12) and (5.16) how data-term gradient components are obtained by wavelet transforms of image-like objects (5.13) and (5.17). At a given scale j, gradient components corresponding to $\boldsymbol{\Theta}_{|j}$ only are necessary. Since the usual wavelet decomposition computes coefficients from fine to coarse scales, it implies that all coefficients corresponding to scales k finer than j ($F \leq k < j$) need to be computed but are finally discarded, which clearly constitutes a waste of resources – especially at the beginning of the process. Wu & Kanade [44] used the specific Cai-Wang wavelet transform [8], which allows to compute wavelet coefficients from coarse to fine scales, to get around this issue. However, our estimator should be generic in terms of choice of wavelet basis, so we do not want to restrict ourselves to this specific wavelet.

A similar situation appears in particular with DFD model, where warped image $I_1(\boldsymbol{x} + \boldsymbol{v}(\boldsymbol{x}))$ has to be computed. At scale j, warping I_1 requires a fine-resolution motion, i.e. $\boldsymbol{v} \in \mathbf{V}_F \times \mathbf{V}_F$, although finest scale F has not been reached yet. Wu & Kanade used interpolation to get a dense motion. In our approach, the fine-resolution motion is obtained by replacing non-estimated coefficients with zeros (that is to say, at every scale k finer than current scale j: $F \leq k < j$), then performing the wavelet reconstruction. This strategy has the drawbacks of inducing many useless operations involving null fine-scale coefficients.

Both of these computational issues, although not critical, can be conveniently addressed by using "smart" filterbanks. Details on design and performance improvements brought by such filterbanks are given further in Chapter 7: "Implementation".

Optimization Routines

Wu & Kanade pointed out the main weakness of their algorithm as its overall computational efficiency. First, their approach requires the evaluation of the (not-so-sparse) Hessian matrix, of size $2^{-2j} \times 2^{-2j}$ with j the current scale, once at each scale level. Then, since they used Levenberg-Marquard optimization method, a linear system of the same size is later to be inverted in ordain to obtain the minimum. Computational cost and memory usage rapidly become prohibitive, hence motion estimation has to be restricted to coarsest scales only. This might be sufficient for most optic flow applications; when dealing with fluid flows however smaller scales often need not to be neglected. Our algorithm follows Wu & Kanade's suggestion to use a gradient-based optimization method, furthermore since gradient evaluation finds in our case a simple formulation. Since a high number of unknowns is involved, we suggest l-BFGS algorithm [32] which approximate the Hessian, thus avoiding its direct computation. Moreover, it does not require to store the full matrix (e.g. $2^{42} \simeq 4.4 \times 10^{12}$ elements with 1024×1024 px images!).

Pseudocode Synthesis

Figures 5.3 and 5.2 present a pseudocode synthesis of DFD gradient computation and sequential estimation process, respectively.

5.3 Null-Divergence Estimator

The null-divergence version of our wavelet-based motion estimator is set up analogously to the generic estimator presented before. Here, the unknown of the optical flow problem is the set of coefficients $\{d^{\text{div}}_{\boldsymbol{j},\boldsymbol{k}}\}_{\boldsymbol{j},\boldsymbol{k}}$ representing stream function $z \in H^1(\mathbb{R}^2)$ from which divergence-free $\boldsymbol{v} = \mathbf{curl}(z) \in \mathcal{H}^{\text{div}}(\mathbb{R}^2)$ is obtained – Section 4.1. We write Θ^{div} the set of all coefficients representing the stream function z, and the problem becomes:

$$\begin{cases} \hat{\boldsymbol{v}}^{\text{div}}(\boldsymbol{x}) = \boldsymbol{\Phi}^{\text{div}\boldsymbol{T}}(\boldsymbol{x})\hat{\Theta}^{\text{div}}\,, \ \forall \boldsymbol{x} \in \Omega \\ \hat{\Theta}^{\text{div}} = \underset{\Theta^{\text{div}}}{\arg\min}\, J_{\text{data}}(I, \Theta^{\text{div}}) + \alpha J_{\text{reg}}(\Theta^{\text{div}})\,. \end{cases} \tag{5.20}$$

Nevertheless the theoretical developments leading to vector wavelets $\boldsymbol{\psi}^{\text{div}}_{\boldsymbol{j},\boldsymbol{k}}$ in (4.16), such functions are in practice never manipulated to build $\boldsymbol{v}$ from $\{d^{\text{div}}_{\boldsymbol{j},\boldsymbol{k}}\}_{\boldsymbol{j},\boldsymbol{k}}$. Equivalence formulas of Section 4.1.4 are much more convenient, as they involve usual scalar basis functions. However, as already warned at the end of Section 4.1, the practical implementation of divergence-free bases for finite and discrete signals rises up several subtle aspects that are clarified in Appendix A. In particular, the coarse scale C – Eq. (5.3) – must be set to 0, and $a^{\text{div}}_{(0,0),(0,0)} = 0$.

Motion $\boldsymbol{v}$ corresponding to Θ^{div} is rebuilt when necessary (e.g. for warping I_1) using relations of Section A.3.1 (pseudocode also available Fig. A.6). Just like the previous estimator working with usual wavelet bases, data functional gradients are again given by the projection, on the considered divergence-free wavelet basis, of terms (5.13) and (5.17). Explicit formulas are given in Appendix A.3.3. The main difference with the previous estimator is the number of unknowns, two times smaller since coefficients corresponding to the irrotational field (yielding divergence) are not estimated.

It should be noticed that divergence-free bases allow to represent zero-mean velocity fields only. In practice, when a uniform translation is present, the estimation of a divergence-free motion can be carried out in two steps:

(i) a first estimation of the translational motion component, $\boldsymbol{v}^{\text{trans}}$, using the generic estimator. This step is very fast, since it suffices to estimate the two approximation coefficients at coarse scale $C = 0$. Image I_1 is then warped with $\hat{\boldsymbol{v}}^{\text{trans}}$, to "remove" the translation.

(ii) a second estimation of $\boldsymbol{v}^{\text{div}}$, the divergence-free component, using the divergence-free estimator.

The final estimate is the sum of both translation and diverge-free components:

$$\hat{\boldsymbol{v}} = \hat{\boldsymbol{v}}^{\text{trans}} + \hat{\boldsymbol{v}}^{\text{div}}\,. \tag{5.21}$$

```
        /**** Sequential Estimation ****/

Θ_|C = 0; //initialization

for (j=C; j>=F; j--) //loop over scales, coarse to fine
{
   //current scale estimate by l-BFGS
   Θ̂_|j = l-BFGS_min(I_0, I_1, Θ_|j, eval_Jdata_grad);

   Θ_|j+1 = (Θ̂^T_1|j, 0, Θ̂^T_2|j, 0)^T; //next initialization
}

return Φ^T Θ̂; //return motion
```

Figure 5.2: Sequential estimation pseudocode. L-BFGS implementations usually work as black boxes, it is only required to give the algorithm a procedure to evaluate the objective functional and its gradient, such as eval_Jdata_grad() given Figure 5.3 and a few parameters.

```
        /**** DFD Gradient & Functional Computation (scale j) ****/

[∂1 J_data, ∂2 J_data, J_data] = eval_Jdata_grad(I_0, I_1, Θ_1|j, Θ_2|j)
 {
  //-rebuild full-resolution motion from coefficients
  Θ = (Θ^T_1|j, 0···0, Θ^T_2|j, 0···0)^T; //fill remaining fine scales with 0
  v = Φ^T Θ; //actually, use smart filterbanks

  //-image gradients
  I_w = warp(I_1, v); //warp image
  I_w,x1 = grad(I_w, x_1); //spacial gradients
  I_w,x2 = grad(I_w, x_2);
  I_t = I_0 - I_w; //temporal difference

  //-terms to be decomposed
  G_x1 = I_t.mul(I_w,x1); //point-wise multiplication here
  G_x2 = I_t.mul(I_w,x2);

  Θ_x1 = Φ^-1 G_x1; //wavelet decomposition
  Θ_x2 = Φ^-1 G_x2; //actually use smart filterbanks again

  //-keep current scales only (C to j)
  ∂1 J_data = Θ_x1|j; //gradient w.r.t. first motion component
  ∂2 J_data = Θ_x2|j; //and w.r.t. second component

  //-finally compute functional value
  J_data = .5*sum(I_t.mul(I_t)); //point-wise square then sum
 }
```

Figure 5.3: Algorithm – DFD gradient & functional computation pseudocode.

Chapter 6

Regularization Schemes

As introduced in Section 2.1.2, regularization schemes are necessary in order to compensate for the lack of information that arises from the aperture problem, as well as to "correct" spurious motion estimates resulting, for instance, from noisy data.

6.1 A Simple Closure: Truncated Bases

A first, very simple approach lowers the number of unknowns by neglecting small scales coefficients: motion is estimated on a *truncated basis*. Let L be the finest estimated scale, taken strictlycoarser than pixel scale F: $F < L \leq C$. The motion therefore belongs to a lower-resolution space, $\boldsymbol{v} \in (\mathbf{V}_L \times \mathbf{V}_L) \subset (\mathbf{V}_F \times \mathbf{V}_F) \subset \mathbf{L}^2(\mathbb{R}^2) \times \mathbf{L}^2(\mathbb{R}^2)$. The problem writes:

$$\begin{cases} \hat{\boldsymbol{v}} = \boldsymbol{\Phi}^{\boldsymbol{T}} \hat{\boldsymbol{\Theta}}_{|L}\,, \; L > F \\ \hat{\boldsymbol{\Theta}}_{|L} = \underset{\boldsymbol{\Theta}_{|L}}{\arg\min} \; J_{\text{data}}(I, \boldsymbol{\Theta}_{|L})\,. \end{cases} \tag{6.1}$$

Solution is obtained following the sequential estimation presented previously in Chapter 5, stopping at scale L. In order to recover a full-resolution motion, coefficients corresponding to non-estimated small scales are replaced by zeros, similarly to what is done in order to compute warped image in Section 5.2.4.

The only parameter of this estimator, leaving aside the choice of the wavelet basis, is L. The full-scale motion has twice more unknowns as pixels in images. By neglecting the current finest scale, the number of coefficients is divided by four; therefore it is theoretically possible to pick $L = F + 1$, i.e. to neglect the pixel scale only. In practice, it often remains ill-posed due to the aperture problem (Section 2.1.2). Going towards fine scales, the size of support of wavelet functions decreases. If input images are not textured enough, so that image gradients vanish over the support of wavelet functions at a given scale L, then data model no longer provides information and any coefficient could possibly fit. As a consequence, local "homogeneity" is lost, leading to erroneous and noisy solutions. In practice, L must be taken "coarse enough" according to input images, and in any case greater than F.

6.1.1 Properties of the solution

The choice of the wavelet basis is actually of high importance, especially since smallest scales are neglected. Indeed, the regularity of the solution, as well as the quantity of energy "lost" from small scales cancellation, depends on the wavelet basis through the number of vanishing moments.

Polynomial approximations

From vanishing moments definition (3.2.1), a wavelet with n VM is orthogonal to any polynomial of degree $n-1$. Consequently, piece-wise [1] polynomials of degree $n-1$ belonging to V_F are exactly

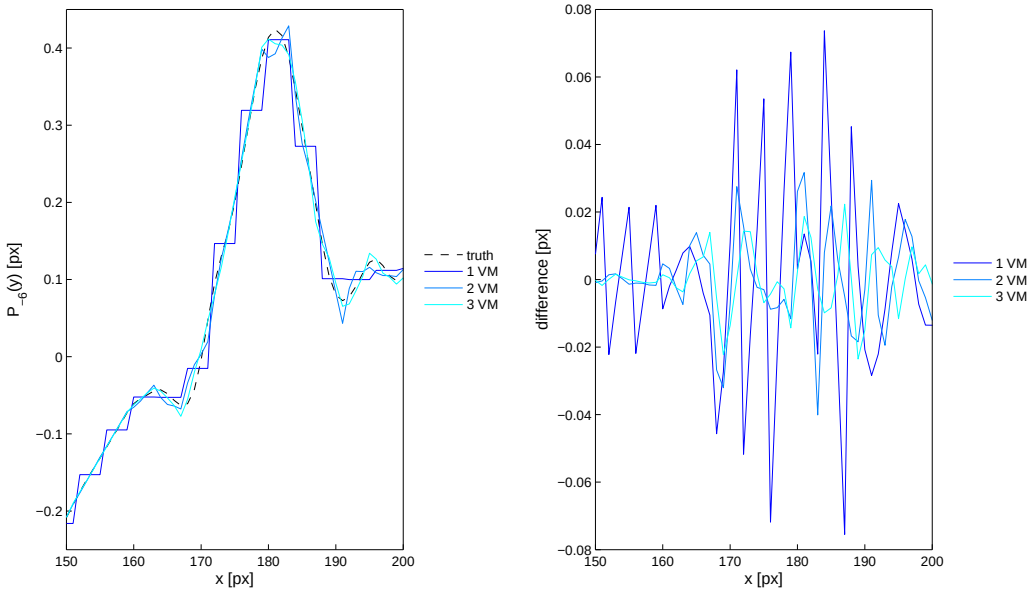
1. On the support of $\{\varphi_{F+1,k}\}$.

described in V_{F+1}, since wavelet basis atoms belonging to its orthogonal complement W_{F+1} have vanishing coefficients. Therefore $\hat{\boldsymbol{v}} \in \mathbf{V}_L \times \mathbf{V}_L$, solution of (6.1), is a piece-wise polynomial of order $n-1$ in $\mathbf{V}_{L-1} \times \mathbf{V}_{L-1}$ over the support of scaling functions.

Truncation error

Because of the energy conservation provided by the wavelet transform, truncating small scales coefficients certainly introduces an error – the smaller those coefficients are, and the fewer energy is lost. Using the Lipschitz regularity of the estimated motion, it is possible to obtain a bound for the number of VM, above which this truncation error no longer depends on the number of VM, but on the motion regularity only.

The amplitude of truncated small-scales coefficients depends on either α the signal regularity, or n the number of vanishing moments of the analyzing wavelet. Therefore the amount of energy lost by neglecting small scales will depend either on the wavelet basis, when $n < \alpha$, or on the signal regularity if $n \geq \alpha$. Consequently, using a "high enough" number of vanishing moments should ensure that the amount of neglected energy does not depend on the wavelet basis. The interest of this simple rule however has to be balanced, since raising the number of VM also results in a significant increase of the computational burden, due to the wider support of basis functions.



(a) (b)

Figure 6.1: Effects of fine scales canceling, applied to a 1D slice of a the first component v_1 of a turbulent flow $\boldsymbol{v} \in V_{-8}$. This slice y is projected to V_{-6} by canceling the two finest detail scales, using Daubechies wavelets with 1, 2 or 3 VM (a). The corresponding error $(y - \mathcal{P}_{-6}(y))$ is plotted in (b).

6.2 Dense Schemes

While previous regularization avoided the aperture problem by truncating the wavelet basis and neglecting the smallest motion scales, dense schemes allow to continue the estimation up to the finest (pixel) scale. Such regularizers compensate for the aperture problem by enforcing some spatial smoothness to the optical flow solution, through penalizations of combinations of motion derivatives. We present two families of schemes that benefit from the wavelet formalism and work on a similar fashion. The first family is based on a norm equivalence, it is limited to "basic" schemes only, yet is very simple to set up. The second family transfers derivatives calculations

to the wavelet basis; it enables to design much more general schemes such as the fluid-dedicated "curl gradient" introduced by Suter (Section 2.3). These high-order regularizers are added to the data-term functional, with a scalar factor α that gives a higher magnitude to either the data-term or to the regularizer. From (2.6), the estimation problem writes:

$$\begin{cases} \hat{\boldsymbol{v}} = \boldsymbol{\Phi}^{\boldsymbol{T}}\hat{\boldsymbol{\Theta}} \ \in \mathbf{V}_F \times \mathbf{V}_F \\ \hat{\boldsymbol{\Theta}} = \underset{\boldsymbol{\Theta}}{\arg\min}\, J_{\text{data}}(I, \boldsymbol{\Theta}) + \alpha J_{\text{reg}}(\boldsymbol{\Theta})\,. \end{cases} \tag{6.2}$$

J_{data} may be either DFD (5.10) or OFC (5.15) data models, or any other model designed the same way. Various regularizers J_{reg} and their gradients are given below. Estimation process follows the gradient-based sequential estimation introduced in Section 5.2.3, using full functional gradient:

$$\nabla J(\boldsymbol{\Theta}) = \nabla J_{\text{data}}(I, \boldsymbol{\Theta}) + \alpha \nabla J_{\text{reg}}(\boldsymbol{\Theta})\,. \tag{6.3}$$

In the following, concepts are derived for the *anisotropic* wavelet transform (Section 3.1.4) since they find a simpler derivation and implementation, especially regarding continuous approximations of Section 6.2.2. Reasonings however hold true for the isotropic transform as well.

6.2.1 Discrete Approximation

This approach gives basic yet high-order regularization schemes that write, with n the derivation order:

$$J_{\text{reg}}(\boldsymbol{v}) = \frac{1}{2}\int_\Omega \sum_{i,p=1,2} \left(\frac{\partial^n v_i(\boldsymbol{x})}{\partial x_p^n}\right)^2 d\boldsymbol{x} = \frac{1}{2}\sum_{i,p=1,2} \left\| \frac{\partial^n v_i}{\partial x_p^n} \right\|_{\mathbf{L}^2}^2 . \tag{6.4}$$

Contrary to continuous approximations of Section 6.2.2, more general derivatives combinations (e.g. $\|\Delta v_i\|_{\mathbf{L}^2}^2$) cannot be obtained. The derivation order is the only parameter; setting $n = 1$, one recognizes the classical first-order regularizer by Horn & Schunck.

In Section 4.2.1, we introduced an equivalence relation between the norm of a function derivative and the norm of this function's coefficients. Extended to 2D signals and applied to a motion component v_i, this relation writes:

$$\left\| \frac{\partial^n v_i}{\partial x_p^n} \right\|_{\mathbf{L}^2} \sim \left\| (-4)^{2n} 2^{n j_p} d^i_{\boldsymbol{j},\boldsymbol{k}} \right\|_{\mathbf{l}^2} , \quad \text{with } d^i_{\boldsymbol{j},\boldsymbol{k}} = \left\langle v_i \,;\, \psi_{j_1,k_1} \otimes \psi_{j_2,k_2} \right\rangle . \tag{6.5}$$

Consequently, amplitude of motion derivatives can be controlled by a weighted penalization of the motion coefficients, which are the variables of the estimation problem. Dropping factor $(-4)^{2n}$ which depends on derivation order only, regularizer (6.4) may be rewritten:

$$J_{\text{reg}}(\boldsymbol{v}) \sim J_{\text{reg}}(\boldsymbol{\Theta}) = \frac{1}{2}\sum_{i=1,2}\sum_{\boldsymbol{j},\boldsymbol{k}} (4^{-nj_1} + 4^{-nj_2}) \left| d^i_{\boldsymbol{j},\boldsymbol{k}} \right|^2 . \tag{6.6}$$

Its partial derivative with respect to coefficient $d^i_{\boldsymbol{j},\boldsymbol{k}}$ is

$$\frac{\partial J_{\text{reg}}(\boldsymbol{\Theta})}{\partial d^i_{\boldsymbol{j},\boldsymbol{k}}} = (4^{-nj_1} + 4^{-nj_2}) d^i_{\boldsymbol{j},\boldsymbol{k}}\,. \tag{6.7}$$

A similar development using instead the isotropic transform leads to:

$$J_{\text{reg}}(\boldsymbol{v}) \sim J_{\text{reg}}(\boldsymbol{\Theta}) = \frac{1}{2}\sum_{i=1,2}\sum_{j,\boldsymbol{k}} (4^{-nj}) \left| d^i_{j,\boldsymbol{k}} \right|^2 , \tag{6.8}$$

and corresponding gradient components are given by:

$$\frac{\partial J_{\text{reg}}(\boldsymbol{\Theta})}{\partial d^i_{j,\boldsymbol{k}}} = (4^{-nj}) d^i_{j,\boldsymbol{k}}\,. \tag{6.9}$$

The constants C_1, C_2 of norm equivalence (4.27) are not be computed. It is indeed simpler to embed the unknown upper constant within balance parameter α (Eq. 6.2).

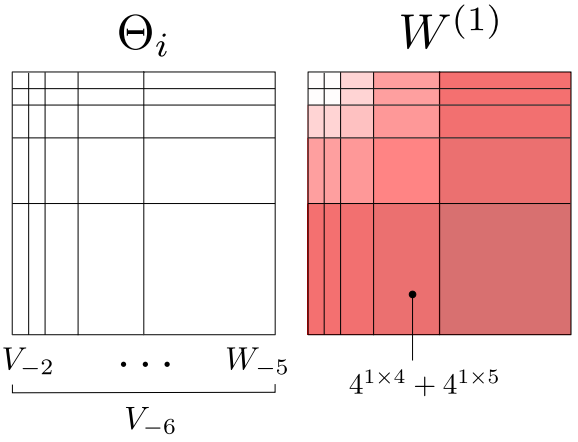


Figure 6.2: Illustration of weighting matrix $W^{(1)}$, from Horn & Schunk first-order scheme. Matrices are associated to space $\mathbf{V}_{-6}$, decomposed up to coarse scale $C = -2$. The darker the color, the higher the amplitude.

Computational aspects

These schemes are very simple to implement: at given scale $F \leq j \leq C$ of the multiscale estimation process, a matrix $W^{(n)}$ of size $2^{-j} \times 2^{-j}$ is built, containing coefficients $4^{-nk_1} + 4^{-nk_2}$ with $j_i \leq k_i \leq C$ corresponding to the tensorial structure of the multiscale basis – see Figure 6.2 for an illustration. For each component Θ_i, gradient is computed by

$$\nabla J_{\text{reg}}(\Theta_i) = W^{(n)} \circ \Theta_i \,, \tag{6.10}$$

where $\circ$ denotes the Hadamard product (element-wise multiplication). Functional value is then given by

$$J_{\text{reg}}(\boldsymbol{\Theta}) = \frac{1}{2} \sum_{i=1,2} \Theta_i : \nabla J_{\text{reg}}(\Theta_i) \,, \tag{6.11}$$

with : the Frobenius inner product (the sum of Hadamard product elements). Finally, the evaluation of this regularization scheme requires four point-wise matrix multiplications and two sums. Matrices $W^{(n)}$ and Θ_i are of the same size as input images ($n_p = 2^{-F} \times 2^{-F}$ pixels) at most (when scale $j = F$, last step of the process), so the cost of this evaluation is of order n_p operations.

6.2.2 Continuous Approximation

Continuous operator approximation allows to set up general regularization schemes by transferring the computation of motion derivatives and their integrals to the wavelet basis atoms. They rely on mass and stiffness matrices presented in Section 4.2.2, and take advantage of the tensorial structure of the wavelet basis (separability) to split computations. Concepts are first explained with a simple example, formulas for full operators follow.

Introductory Example

Let f be a scalar 2D field of $2^{-F} \times 2^{-F}$ samples and its representation a_F in the corresponding approximation space $\mathbf{V}_F = V_F \otimes V_F$:

$$\begin{aligned} f(\boldsymbol{x}) &= \sum_{\boldsymbol{k}} a_{F,\boldsymbol{k}} \varphi_{F,k_1} \otimes \varphi_{F,k_2}(\boldsymbol{x}) \\ &= \sum_{\boldsymbol{k}} a_{F,\boldsymbol{k}} \varphi_{F,k_1}(x_1) \varphi_{F,k_2}(x_2) \,, \quad \text{with } 0 \leq k_1, k_2 \leq 2^{-F} - 1 \,. \end{aligned} \tag{6.12}$$

We consider a simple regularization term involving only the first-order partial derivative with respect to the first variable:

$$J_{\text{reg}}(f) = \frac{1}{2}\left\|\frac{\partial f}{\partial x_1}\right\|^2_{\mathbf{L}^2} . \tag{6.13}$$

Inserting wavelet representation (6.12) and using the separability of the basis leads to [2]:

$$\begin{aligned}
\left\|\frac{\partial f}{\partial x_1}\right\|^2_{\mathbf{L}^2} &= \int_{\mathbb{R}^2}\left(\frac{\partial f(\boldsymbol{x})}{\partial x_1}\right)^2 d\boldsymbol{x} \\
&= \int_{\mathbb{R}^2}\left[\sum_{\boldsymbol{k}} a_{F,\boldsymbol{k}}\frac{\partial}{\partial x_1}\left(\varphi_{F,k_1}(x_1)\varphi_{F,k_2}(x_2)\right)\right]^2 d\boldsymbol{x} \\
&= \sum_{\boldsymbol{k},\boldsymbol{k}'} a_{F,\boldsymbol{k}}a_{F,\boldsymbol{k}'}\int_{\mathbb{R}}\frac{d\varphi_{F,k_1}(x)}{dx}\frac{d\varphi_{F,k_1'}(x)}{dx}dx\int_{\mathbb{R}}\varphi_{F,k_2}(x)\varphi_{F,k_2'}(x)dx \\
&= \sum_{\boldsymbol{k},\boldsymbol{k}'} a_{F,\boldsymbol{k}}a_{F,\boldsymbol{k}'}\left\langle\varphi^{(1)}_{F,k_1}\,;\,\varphi^{(1)}_{F,k_1'}\right\rangle_{\mathbf{L}^2}\left\langle\varphi_{F,k_2}\,;\,\varphi_{F,k_2'}\right\rangle_{\mathbf{L}^2}
\end{aligned} \tag{6.14}$$

Let $M^{(0)}$ and $M^{(2)}$ two matrices of elements indexed by $0 \le p,q \le 2^{-F}-1$:

$$\begin{aligned}
&\left(M^{(0)}\right)_{p,q} = \left\langle\varphi_{F,p}\,;\,\varphi_{F,q}\right\rangle_{\mathbf{L}^2} \\
\text{and}\quad &\left(M^{(2)}\right)_{p,q} = \left\langle\varphi^{(1)}_{F,p}\,;\,\varphi^{(1)}_{F,q}\right\rangle_{\mathbf{L}^2} = -\left\langle\varphi_{F,p}\,;\,\varphi^{(2)}_{F,q}\right\rangle_{\mathbf{L}^2} .
\end{aligned} \tag{6.15}$$

These connection coefficients are computed once for all by solving eigenvalue problems, as presented in Section 4.2.2. Hence regularizer (6.13) rewrites:

$$\begin{aligned}
J_{\text{reg}}(a_F) &= \frac{1}{2}\sum_{\boldsymbol{k},\boldsymbol{k}'} a_{F,\boldsymbol{k}}a_{F,\boldsymbol{k}'}(M^{(2)})_{k_1,k_1'}(M^{(0)})_{k_2,k_2'} \\
&= \frac{1}{2}\sum_{\boldsymbol{k}} a_{F,\boldsymbol{k}}\left(M^{(2)}a_F M^{(0)\boldsymbol{T}}\right)_{\boldsymbol{k}} \\
&= \frac{1}{2}a_F : \left(M^{(2)}a_F M^{(0)\boldsymbol{T}}\right) .
\end{aligned} \tag{6.16}$$

where a_F is the matrix of approximation coefficients, i.e. $\left(a_F\right)_{\boldsymbol{k}} = a_{F,\boldsymbol{k}}$, and : is still the Frobenius inner product. Shall a multiscale wavelet basis be considered (i.e. with approximation and details), the appropriate connection matrices $N^{(0)}$ and $N^{(2)}$ would be obtained by applying the wavelet decomposition [3] to matrices $M^{(0)}$ and $M^{(2)}$. Gradient components of (6.16) are finally given by:

$$\frac{\partial J_{\text{reg}}(a_F)}{\partial a_{F,\boldsymbol{k}}} = \left(M^{(2)}a_F M^{(0)\boldsymbol{T}}\right)_{\boldsymbol{k}} . \tag{6.17}$$

To sum-up, the regularization functional is simply set-up by:

(i) building connection matrices $M^{(0)}$ and $M^{(2)}$, according to chosen basis and current space $\mathbf{V}_F$, at the beginning of the estimation.

(ii) from current motion representation a_F, computing

$$\nabla J_{\text{reg}}(a_F) = M^{(2)}a_F M^{(0)\boldsymbol{T}} \quad \text{(three matrix products)}$$

from which functional value is given by

$$J_{\text{reg}}(a^F) = \frac{1}{2}a_F : \nabla J_{\text{reg}}(a_F) \quad \text{(pointwise product and sum).}$$

2. Notation $\cdot^{(n)}$ stands for the derivation order and has nothing to do with exponents that previously appeared, e.g. in definitions of isotropic or divergence-free bases.

3. See point (ii) of Section 4.2.2 if biorthogonal bases are involved.

Full Schemes for Regular Bases

High-order regularizers are obtained by following a reasoning similar to our introductory example. All schemes write as a sum of several terms of the form

$$\cdots \pm \Theta_i : \left(N^{(n_1)} \Theta_j N^{(n_2)\boldsymbol{T}} \right) \pm \cdots \tag{6.18}$$

with (Θ_i) is the set of multiscale coefficients corresponding to motion component v_i, taken as a matrix. Index n_j, $j = 1, 2$, is the *total* derivation order with respect to variable x_j, e.g. $n_1 = 2$ and $n_2 = 0$ in example (6.16). The convention adopted for schemes formulas given below is the following: matrices $N^{(n)}$ correspond to connection terms of the form

$$\begin{array}{rl} \left\langle \psi_{i,p}^{(n/2)} \,;\, \psi_{j,q}^{(n/2)} \right\rangle_{\mathbf{L}^2} & \text{if } n \text{ is even;} \\ \left\langle \psi_{i,p}^{(1)} \,;\, \psi_{j,q}^{(n-1)} \right\rangle_{\mathbf{L}^2} & \text{if } n \text{ is odd.} \end{array} \tag{6.19}$$

These matrices are symmetric when n is even, so that transpose notation $^{\boldsymbol{T}}$ will be dropped eventually. In order to compute their elements, it is necessary to iterate several integrations by parts until terms of the form $\left\langle \psi_{i,p} \,;\, \psi_{j,q}^{(n)} \right\rangle_{\mathbf{L}^2}$ arise (up to some factor, function of the scales and the derivation order). Following Section 4.2.2, these dot products are given by a wavelet transform of $\left\langle \varphi_{F,p} \,;\, \varphi_{F,q}^{(n)} \right\rangle_{\mathbf{L}^2}$, which are initially computed by solving eigenvalue problems.

– Horn & Schunk first order regularizer (2.10) writes

$$J_{\text{reg}}(\boldsymbol{\Theta}) = \frac{1}{2} \sum_{i=1,2} \Theta_i : \left(N^{(2)} \Theta_i N^{(0)} + N^{(0)} \Theta_i N^{(2)} \right) . \tag{6.20}$$

Figure 6.3 illustrates the structure of these multiscale connection coefficient matrices $N^{(0)}$ and $N^{(2)}$.

– The divergence regularizer is

$$\begin{aligned} J_{\text{reg}}(\boldsymbol{v}) &= \frac{1}{2} \left\| \operatorname{div}(\boldsymbol{v}) \right\|_{\mathbf{L}^2}^2 \\ \Rightarrow J_{\text{reg}}(\boldsymbol{\Theta}) &= \Theta_1 : \left(N^{(2)} \Theta_1 N^{(0)} \right) + \Theta_2 : \left(N^{(0)} \Theta_1 N^{(2)} \right) + \Theta_2 : \left(N^{(1)} \Theta_1 N^{(1)} \right) . \end{aligned} \tag{6.21}$$

– Laplacian regularizer is given by

$$\begin{aligned} J_{\text{reg}}(\boldsymbol{v}) &= \frac{1}{2} \sum_{i=1,2} \left\| \Delta v_i \right\|_{\mathbf{L}^2}^2 \\ \Rightarrow J_{\text{reg}}(\boldsymbol{\Theta}) &= \frac{1}{2} \sum_{i=1,2} \Theta_i : \left(N^{(4)} \Theta_i N^{(0)} + N^{(0)} \Theta_i N^{(4)} + 2 N^{(2)} \Theta_i N^{(2)} \right) . \end{aligned} \tag{6.22}$$

– The curl gradient regularizer (2.16) is

$$\begin{aligned} J_{\text{reg}}(\boldsymbol{\Theta}) = & \frac{1}{2} \Theta_1 : \left(N^{(0)} \Theta_1 N^{(4)} + N^{(2)} \Theta_1 N^{(2)} \right) \\ & + \frac{1}{2} \Theta_2 : \left(N^{(4)} \Theta_2 N^{(0)} + N^{(2)} \Theta_2 N^{(2)} \right) \\ & - \Theta_1 : \left(N^{(3)} \Theta_2 N^{(1)\boldsymbol{T}} \right) - \Theta_2 : \left(N^{(1)} \Theta_1 N^{(3)\boldsymbol{T}} \right) . \end{aligned} \tag{6.23}$$

– And divergence gradient regularizer, also in (2.16):

$$\begin{aligned} J_{\text{reg}}(\boldsymbol{\Theta}) = & \frac{1}{2} \Theta_1 : \left(N^{(4)} \Theta_1 N^{(0)} + N^{(2)} \Theta_1 N^{(2)} \right) \\ & + \frac{1}{2} \Theta_2 : \left(N^{(0)} \Theta_2 N^{(4)} + N^{(2)} \Theta_2 N^{(2)} \right) \\ & + \Theta_1 : \left(N^{(1)} \Theta_2 N^{(3)\boldsymbol{T}} \right) + \Theta_2 : \left(N^{(1)} \Theta_2 N^{(3)\boldsymbol{T}} \right) . \end{aligned} \tag{6.24}$$

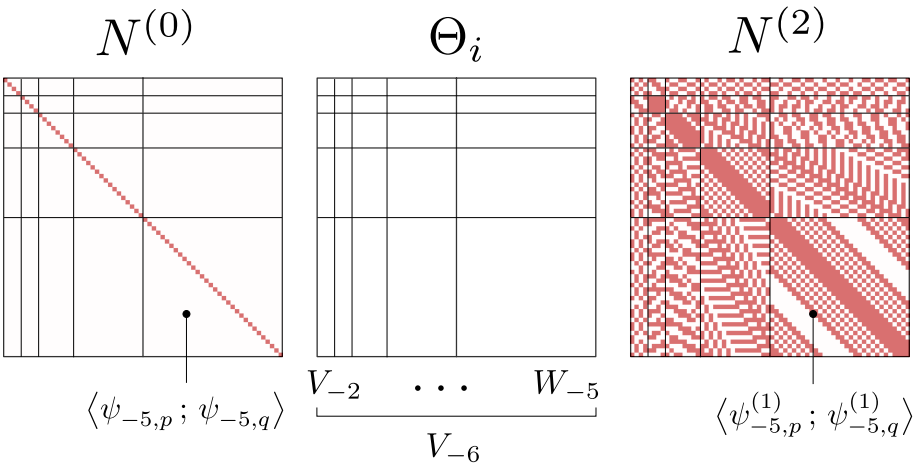


Figure 6.3: Illustration of connection coefficients matrices $N^{(0)}$ and $N^{(2)}$, from Horn & Schunk first-order scheme (6.20). Matrices are associated to space $\mathbf{V}_{-6}$, decomposed up to coarse scale $C = -2$. Connection coefficients were computed using Daubechies-10 wavelets; non-zero values are in *red*. This basis is orthogonal, hence $N^{(0)}$ (*left*) is the identity matrix. $N^{(2)}$ (*right*) reveals inter-scale connections, as well as the periodic boundary conditions.

Full Schemes for Divergence-free Bases

When divergence-free bases are employed, regularizers must be computed from stream function z coefficients Θ^{div}. Their expressions are obtained by inserting $\boldsymbol{v} = (v_1 \, , \, v_2)^{\boldsymbol{T}} = (\partial_{x_2} z \, , \, -\partial_{x_1} z)^{\boldsymbol{T}}$ in regularizers such as (2.10) or (2.16).

– Horn & Schunk regularizer (2.10) then writes

$$J_{\text{reg}}(\Theta^{\text{div}}) = \frac{1}{2}\Theta^{\text{div}} : \left(N^{(0)}\Theta^{\text{div}}N^{(4)} + N^{(4)}\Theta^{\text{div}}N^{(0)} + 2N^{(4)}\Theta^{\text{div}}N^{(4)} \right) . \tag{6.25}$$

– The curl gradient regularizer (2.16), now identical to the Laplacian regularizer from the divergence-free property, becomes:

$$\begin{aligned} J_{\text{reg}}(\Theta^{\text{div}}) = \frac{1}{2}\Theta^{\text{div}} : \Big(& N^{(0)}\Theta^{\text{div}}N^{(6)} + 3N^{(2)}\Theta^{\text{div}}N^{(4)} \\ & + N^{(6)}\Theta^{\text{div}}N^{(0)} + 3N^{(4)}\Theta^{\text{div}}N^{(2)} \Big) . \end{aligned} \tag{6.26}$$

Computational aspects

At each new step of the sequential estimation, the multiscale basis is completed with atoms corresponding to new detail coefficients to be estimated. Therefore matrices $N^{(n)}$ have to be re-computed according to the new basis. At the beginning of the estimation process at current scale j, this is done by:

(i) computing matrix $M^{(n)}$ corresponding to the current approximation space $\mathbf{V}_j = V_j \otimes V_j$;

(ii) then applying the 2D isotropic decomposition to this matrix (taken as an image) so as to recover connection coefficient matrix $N^{(n)}$ corresponding to the multiscale tensorial structure

$$\mathbf{V}_j = \big(V_C \oplus W_C \oplus \cdots \oplus W_{j+1}\big) \otimes \big(V_C \oplus W_C \oplus \cdots \oplus W_{j+1}\big).$$

Scaling functions have a compact support, therefore only a small number of connection coefficients is necessary to build $M^{(n)}$ at step (i). These values are pre-computed for every chosen scaling generator, for different orders, with the eigenvalue problem presented in Section 4.2.2. The only requirement is that the considered generator must be regular enough to stand its derivation up to order n. Identically to discrete approximation regularizers, matrix $M^{(n)}$ has at most the same size as input images ($n_p = 2^{-F} \times 2^{-F}$ pixels), so the cost of the wavelet transform to get $N^{(n)}$ at

step (ii) is a linear function of the total number of pixels n_p. Finally, the most expensive steps in functional (and gradient) evaluation are the triple matrix products of the form $(N^{(n_1)}\Theta_i N^{(n2)\boldsymbol{T}})$; each one require at most $(n_p)^{3/2}$ multiplications. Hence the overall complexity of this regularization is of order $(n_p)^{3/2}$.

Chapter 7

Implementation

7.1 Smart Filterbanks

As seen in Section 5.2.4, the proposed coarse-to-fine estimation process relies on inverse and forward wavelet transforms that involve useless operations: motion reconstruction to warp I_1 with null coefficients, and gradient evaluation where fine detail coefficients are computed to be later discarded. In order to minimize this waste of computations, we propose a simple modification of the decomposition and reconstruction filterbanks. Considering a given scale $F \leq j \leq C$ of the estimation process:

- at the decomposition, *useless* fine scale details (scales s, $F \leq s < j$) are not computed;
- at the reconstruction, *null* fine scale details (scales s, $F \leq s < j$) are not added.

Either at the reconstruction or at the decomposition, this approach saves 62.5% of computations at each fine scale having null or useless coefficients – see Appendix B. Remaining filterbank iterations are performed with the usual form.

At the beginning of the estimation process, almost all of the coefficients are null/useless, hence the gain is quite considerable. Figure 7.1 shows the improvements observed at the decomposition by the use of modified filterbanks, it can be up to 70, 80% as soon as several scales are neglected. Results at the reconstruction are very similar.

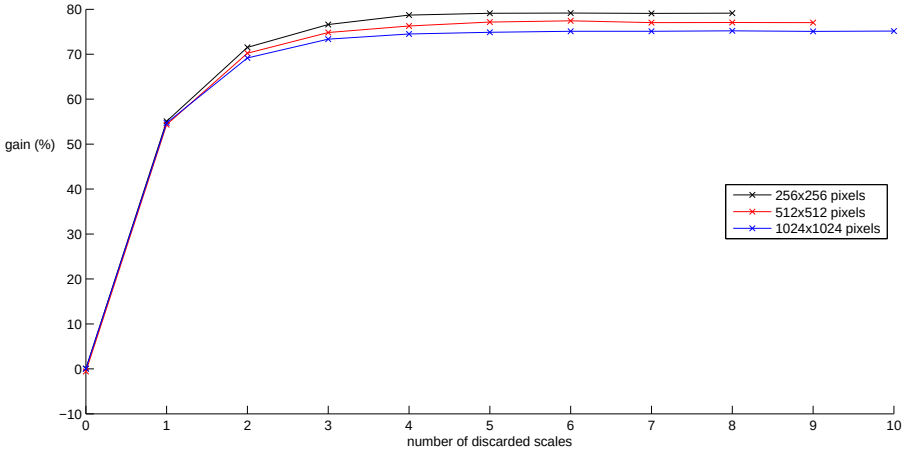


Figure 7.1: A wavelet decomposition up to coarsest scale $C = 0$ is applied to images of side size 256, 512 and 1024 px (Daubechies wavelet generator with 5 VM). The number of useless, neglected fine scales ranges from 0 (usual decomposition) to the maximum (only coarse approximation coefficients are kept). The time gain brought by the use of modified filterbanks instead of usual ones is given in percent. It is a mean value, computed over 100 successive decompositions.

7.2 Software

Current version of the software uses a custom C++ library for wavelet transform. It was written keeping it mind readability, hence it is far from being optimized. Therefore, computation times can be hardly compared to those of other algorithms. Other libraries involved are:

- `CImg`, a C++ image-processing library [26];
- `libLBFGS`, a C library implementing Nocedal's l-BFGS [32];
- `Spline`, a C library to handle spline-based interpolation and derivation [41], used for warping and spacial gradient computation;
- `cLapack` and `cBLAS` [1] for the matrix-related operations (eigenvalue problem of Section 4.2.2 and matrix products of high-order regularizations Section 6.2.2) ;
- `netcdf` (optional), a C library to support NetCDF data files [36].

Part IV

Evaluation

Chapter 8

Code Validation

The purpose of this chapter is to assess general performances and behaviors of the proposed algorithm. In particular, the following points must be answered:

(i) In which range of apparent displacements does the method perform best?
(ii) What is the influence of scale parameters C, L?
(iii) Which wavelet family should be used; how much vanishing moments?
(iv) Which data model, between OFC and DFD, gives the best results?
(v) What does the null-divergence estimator bring?
(vi) How do the various proposed regularization schemes behave?
(vii) How does the algorithm rank, compared to state-of-the-art?

The effects of those various parameters are not easily uncoupled: for instance, some regularizations or the null-divergence bases require "smooth enough" bases. Hence we propose to consider, in the first place, the simplest estimator with regularization by small scale truncation (Section 6.1), in order to study points (i) to (v). Then, having chosen the best data model, the optimal scale parameters and so on, more complex regularizations (vi) and comparisons to state-of-the-art (vii) will be examined.

Now, how can motion estimates be evaluated? Often the overall flow configuration is known, or can be determined visually, so that a qualitative evaluation is almost always possible. On the other hand, reference values are necessary to make quantitative comparisons and error measurements, yet no ground truth velocity value is generally known. One possibility is to refer to another measurement, obtained either by probes (hence punctual only) or by another image-based method such as the cross-correlations. This approach will be used in the next chapter, to investigate estimates from two laboratory experiments. The second option, chosen below, consists in using synthetic reference data for which the ground-truth velocity is known.

8.1 Synthetic Dataset

8.1.1 3D Turbulence & Cylinder Wake

These synthetic particle image velocimetry (PIV) datasets present two radically different flow configurations:

– a 3D isotropic homogeneous turbulent flow, denoted *IHT* hereafter;
– a cylinder wake at Reynolds 3900, referred to as *wake* in the following.

Both sequences were created in a similar fashion. From a 3D numerical simulation of the flow, a velocity field corresponding to a single time-step is kept. 100 000 particles are then advected within this stationary (time-constant) 3D flow using the Lagrangian equation for non-heavy particles, and synthetic visualizations are finally obtained by simulating the lightning of a single slice of the volume. Sample synthetic images as well as the vorticity field of the underlying flow $\boldsymbol{v}_{\text{ref}}$ are presented Fig. 8.1. The IHT flow was obtained by Direct Numerical Simulation using GHOST [1]

1. Geophysical High Order Suite for Turbulence

software with 256^3 spatial points [30]; resulting images size is 256×256 px. Wake flow was computed by Large Eddy Simulation [35]; images size is 338×192 px. Particles are of very small radius, below 2 px. Histograms of these synthetic images are similar to those observed on real experimental PIV images, such as the ones processed in Chapter 9.

A total of 100 images are available for each sequence, they were computed every 100 time-steps of the particle transport simulation. Resulting displacements, from one frame to the next one, are very small: mean displacement is of order 0.04 px/frame and 0.1 px/frame for IHT and wake sequence, respectively. Such low amplitudes are hardly detectable by the human eye. However, since for both sequences the velocity field is constant in time, this configuration enables interesting benchmarks. Indeed, the observable displacement between two given images is the time-integration of a particle trajectory during the time separating each frame acquisition – let us call it δt. This δt has to be chosen carefully, according to the flow configuration and the capacities of the analyzing method. Large apparent motions may lead to poor estimations of the actual instantaneous velocity – Fig. 8.2 – and similarly too small displacements result in erroneous estimations due to the numerical precision limit. Therefore the goal is to find the displacements range for which the algorithm gives optimal results, and to test whether the various parameters have any influence on this validity range.

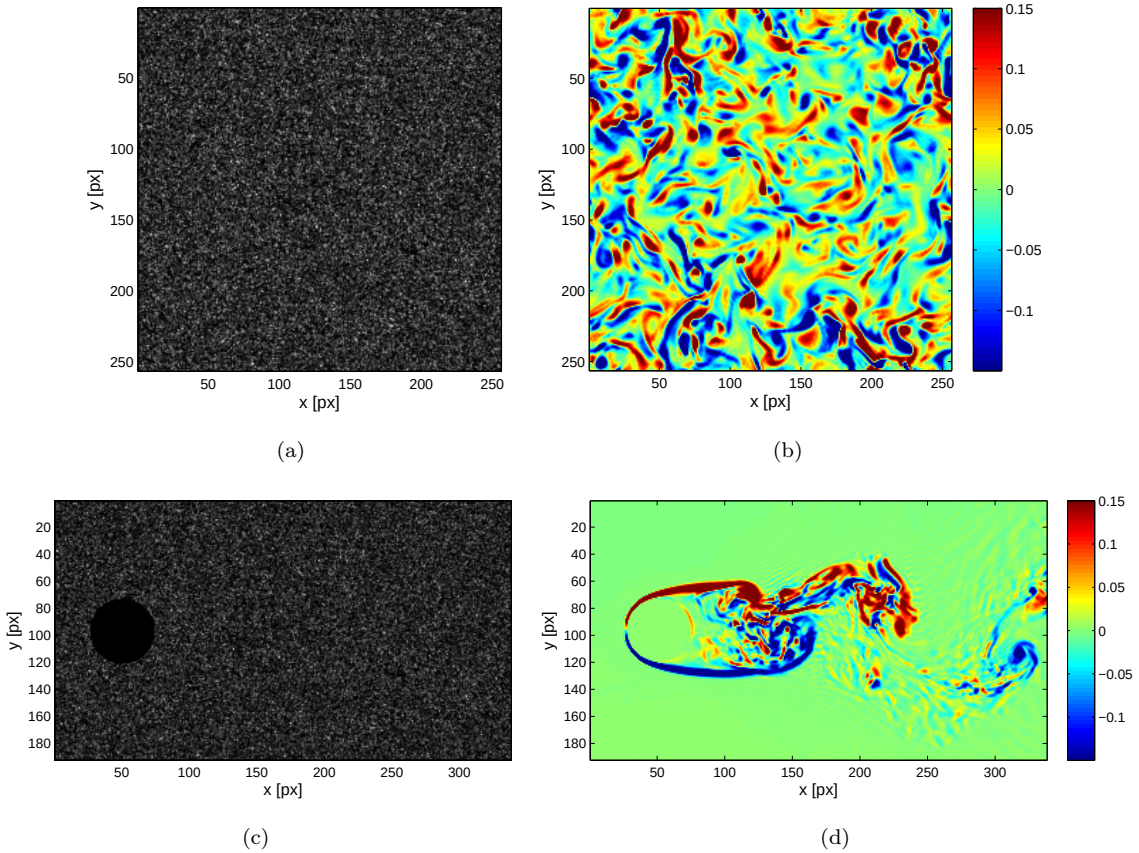


Figure 8.1: Synthetic 3D flow datasets: homogeneous isotropic turbulence (*top*) and cylinder wake (*bottom*). Sample synthetic PIV images for estimation (*left*), vorticity $\mathrm{curl}(\boldsymbol{v}_{\mathrm{ref}})$ of the underlying ground-truth motion (*right*).

Experiments with IHT and wake sequences will process as follows. Both sequences have 100 frames sampled every dt. We will perform estimations between frames at instants 0 and $n \cdot dt = \delta_t$ with $1 \leq n \leq 100$, that is to say with $I_0(\boldsymbol{x}) = I(\boldsymbol{x}, 0)$ and

- $I_1(\boldsymbol{x}) = I(\boldsymbol{x}, dt) \Rightarrow \delta_t = dt$;
- $I_1(\boldsymbol{x}) = I(\boldsymbol{x}, 2 \cdot dt) \Rightarrow \delta_t = 2 \cdot dt$;
- ...

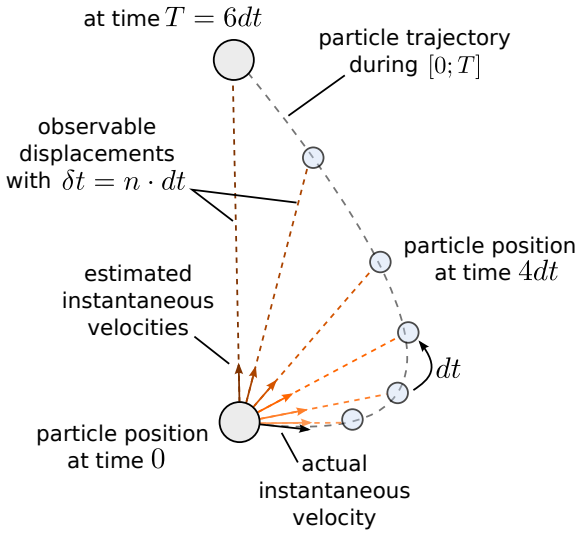


Figure 8.2: A particle is represented along its trajectory at arbitrary time intervals dt. Observable displacements (*dashed lines*) corresponding to different acquisition times $\delta_t = n \cdot dt$, with $n \in [1;6]$, and their corresponding instantaneous velocity (*arrows*) after conversion. Picking large δt leads to inaccurate estimations of the velocity.

– $I_1(\boldsymbol{x}) = I(\boldsymbol{x}, 100 \cdot dt) \Rightarrow \delta_t = 100 \cdot dt$.

Since the underlying flow is constant in time, we should be able to recover the same estimation of the instantaneous velocity for every n . In practice this will not occur, not only because with large n the apparent motion no longer reflects the actual velocity (see Fig. 8.2) so that large, fast structures are poorly estimated, but also since the estimator gets lost in local minima that arise with non-linearities and the amplitude of apparent displacements, as n increases. Therefore errors will be given, normalized in px/frame, as a function of n. The mean and max displacements corresponding to a given n will be displayed as well. Similar errors of estimates obtained with the open cross-correlations software `MatPIV` [39] are given for comparison. They were obtained using a multigrid schemes of 64×64 px, 32×32 px, 16×16 px and 8×8 px, with 50% overlay and two successive passes at each grid level.

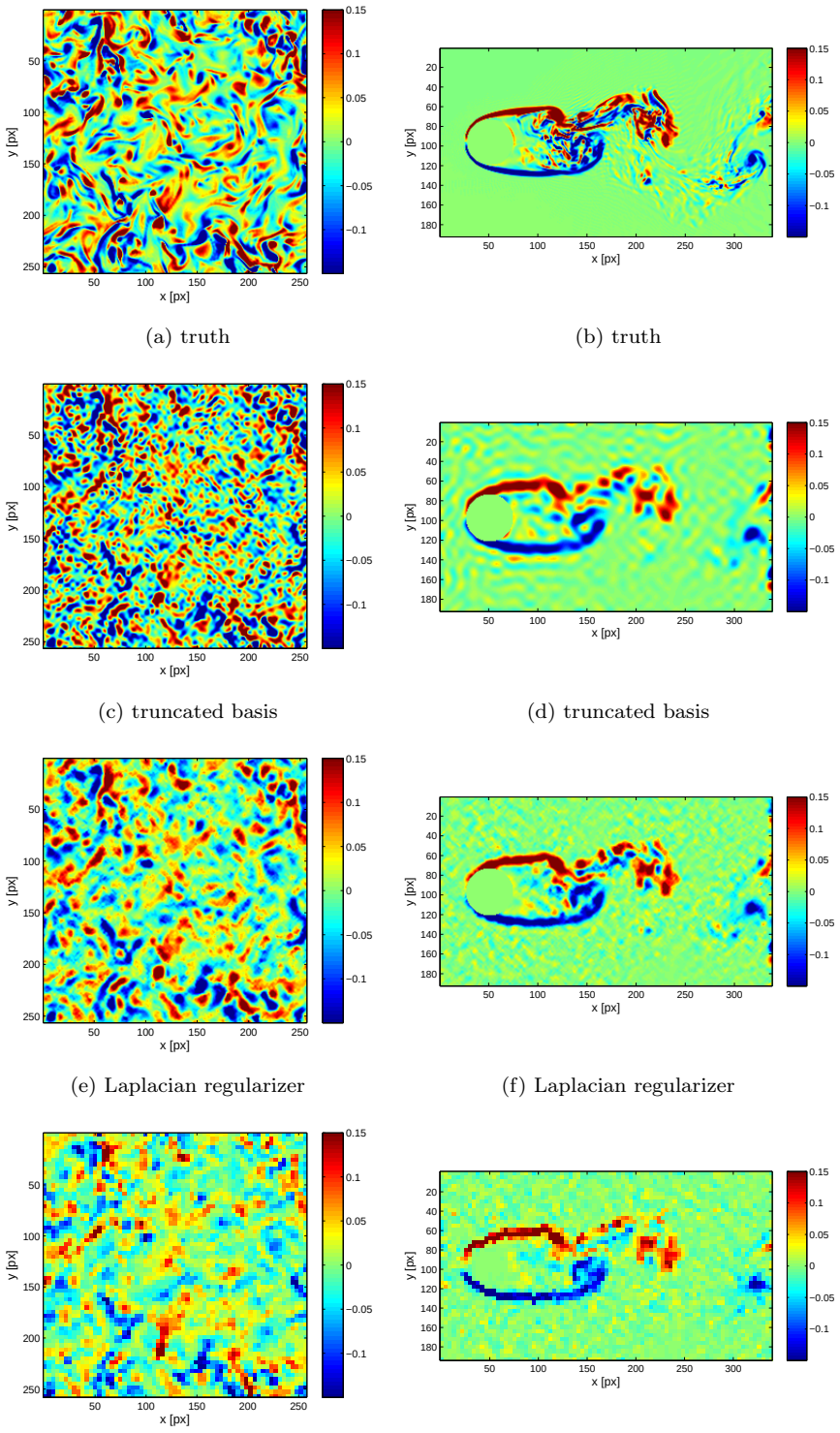
Following results are mostly error graphs. In order to illustrate the output of the methods, vorticity of "good" estimated fields are given in Figure 8.3 for the wavelet optical flow with truncated bases or smoothing regularizer, as well as for the cross-correlations.

8.1.2 2D Turbulence

This dataset was created in the context of the FLUID project [2], to serve as a reference for image-based fluid-motion estimation algorithms. It features two sequences, of 256×256 px synthetic images, representing the most common flow visualization methods: the first sequence is again PIV images (referred to as *particle* images in the following), the second one is scalar transport images (called *scalar* images hereafter for short) – see Section 1.1. The underlying fluid motion is first computed by numerical simulation of incompressible Navier-Stokes equations at Reynolds $Re = 3000$; it is a *2D* periodic turbulent flow. It features relatively small displacements, with a maximum magnitude of order 3.5 px/frame, which is adapted to optical flow methods. From there, particle and scalar images are generated using the Lagrangian equation for non-heavy particles transported by the flow. Simulation details can be found in [19]. Figure 8.4 presents a sample vorticity field of ground-truth motion $\boldsymbol{v}_{\text{ref}}$, along with sample images from both particle and scalar sequences.

This simulated incompressible flow has a null-divergence by construction – see Section 1.2. It is therefore adapted to test the null-divergence estimator. It should be noted that previously introduced 3D flows are incompressible and divergence-free as well; however, as we look at a 2D

2. `http://fluid.irisa.fr`



(a) truth (b) truth

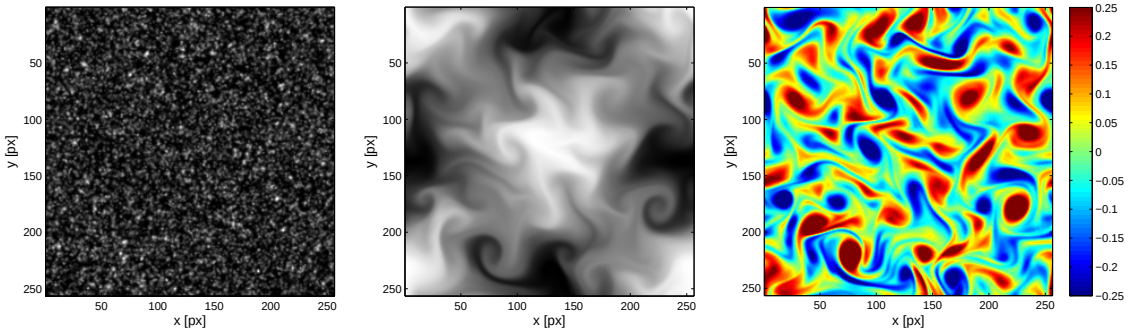
(c) truncated basis (d) truncated basis

(e) Laplacian regularizer (f) Laplacian regularizer

(g) cross-correlations (h) cross-correlations

Figure 8.3: IHT (*left*) and wake (*right*) sequences. Vorticity of ground truth (*upper row*) and good estimates obtained with truncated wavelet basis (*2nd row*, 20 VM and $L = -6$), full wavelet basis with Laplacian regularizer (*3rd row* , 5 VM and $\alpha = 0.01$), and cross-correlations (*bottom row*). Time step is $\delta_t = 20dt$.

slice of this 3D flow, *observable* motions are not divergence free – especially in the presence of a motion component normal to the observation plane. Furthermore, the two very different types of images (particle and scalar) associated to the same flow may help to enlighten how the estimation accuracy is influenced by image nature and characteristics. Finally, estimates from other state of the art methods are available for this dataset, so that we may benchmark our estimator.



(a) (b) (c)

Figure 8.4: Sample particle (*left*) and scalar (*middle*) images from the synthetic 2D turbulence dataset, ground-truth motion vorticity $\nabla \times \boldsymbol{v}_{\text{ref}}$ (*right*).

8.1.3 Notes on Evaluation Criteria

When ground-truth velocity is known, accuracy of estimates is evaluated in terms of root mean square end-point error (RMSE), in pixels:

$$RMSE = \left(\frac{1}{n_p} \sum_{\boldsymbol{x} \in \Omega} \left| \hat{\boldsymbol{v}}(\boldsymbol{x}) - \boldsymbol{v}_{\text{ref}}(\boldsymbol{x}) \right|^2 \right)^{\frac{1}{2}} . \tag{8.1}$$

8.2 Basic Estimator

The first algorithm to be evaluated is the basic estimator presented in Section 6.1, with regularization by small scale truncation. Its main parameters are the number of vanishing moments of the wavelet basis, the coarsest scale considered $C \leq 0$ (i.e. the depth of the wavelet decomposition) and the finest scale estimated L with $F < L \leq C$. The data model and the use of a divergence-free estimator are also to be discussed. Unless specified, the wavelet family is the Daubechies wavelet.

8.2.1 Influence of Scale Parameters

IHT, particle and scalar images Fig. 8.1a, 8.4a and 8.4b are of size 256×256 px, so their finest scale is $F = -8$. The wavelet transform requires square images of size a power of two, so the 338×192 px wake images Fig. 8.1c are inserted at the center of a 512×512 px uniform canvas. Hence their finest scale is $F = -9$.

Finest Estimated Scale (L)

The finest estimated scale parameter L fixes the approximation space $\mathbf{V}_L \times \mathbf{V}_L$ in which the solution is sought. Its choice is a tradeoff between the need to reduce the number of unknowns, in order to close the estimation problem, and the will to estimate the finest scales of the motion. With $L = 0$, only the two coarse approximation coefficients are estimated; it corresponds to a purely translational motion. Setting $L = F$, all detail scales are estimated, but from observations in Section 6.1 the problem is underconstrained (too many unknowns, not even mentioning the

aperture problem) and results would be noisy. Therefore the optimal value should lay somewhere between these two extrema.

In order to study the influence of this parameter L, we set for the moment the coarse scale parameter C to 0: this is a "full-depth" decomposition. Estimations are then achieved for both IHT and wake sequences, with various values $L \geq F$, a 5 vanishing moments (VM) wavelet basis and every time-steps $\delta_t = n \cdot dt$ with $1 \leq n \leq 100$. Particle and scalar sequences are processed as well.

Figure 8.5 presents results obtained for IHT and wake sequences. Optimum is $L = -6$ for both sequences. It is not surprising to find the same value for both cases, since images share approximatively the same particle size and density. As expected, these values are higher than F. The optimal displacements range is of order $[0.5; 5]$ px, it is not influenced by L. Results for particle and scalar sequences are given in Fig. 8.6. Optima are $L = -6$ and $L = -5$ for particle and scalar images, respectively. As previously mentioned, scalar diffusion images have much more low-gradient areas. Uncertainties due to these low gradients arise sooner than with particle images (as we proceed towards finer scales), therefore it is not surprising to find that the optimum solution space is coarser with scalar images than with particles.

These observations already raise up a limitation of our basic estimator – the main limitation, actually: parameter L must be chosen "small enough" in order to stay below the critical scale where aperture problem appears. Not only this critical scale highly depends on the nature of input images, but it may also vary within a given image (in the case of a non-homogeneously seeded flow, for instance). When no ground-truth is available, the only way to figure out a "reasonably good" value is to achieve several estimates and rely on a qualitative evaluation.

Coarsest Scale Considered (C)

Parameter C fixes the number of scales comprised within coarse approximation space V_C, and the number of remaining detail scales up to V_L: $V_F = V_C \oplus W_C \oplus \cdots \oplus W_{L+1}$. Setting $L = -6$ for IHT and wake sequences, we may consider :

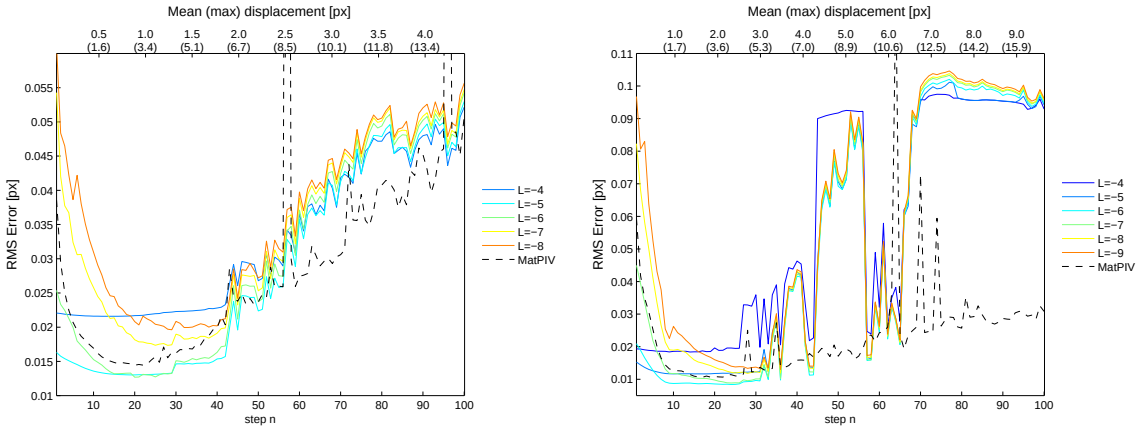
$$\begin{aligned} V_{-6} &= V_0 \oplus W_0 \oplus W_{-1} \oplus \cdots \oplus W_{-5} \\ &= \cdots \\ &= V_{-2} \oplus W_{-2} \oplus \cdots \oplus W_{-5} \\ &= V_{-5} \oplus W_{-5}\,. \end{aligned}$$

The closer is C to L, the lesser is the number of detail scales to be estimated. This choice has two main consequences:

(i) with C close to L, the sequential estimation process involves less steps since fewer detail scales are considered. Also, forward and inverse wavelet transforms require less operations, resulting in an overall much faster process;

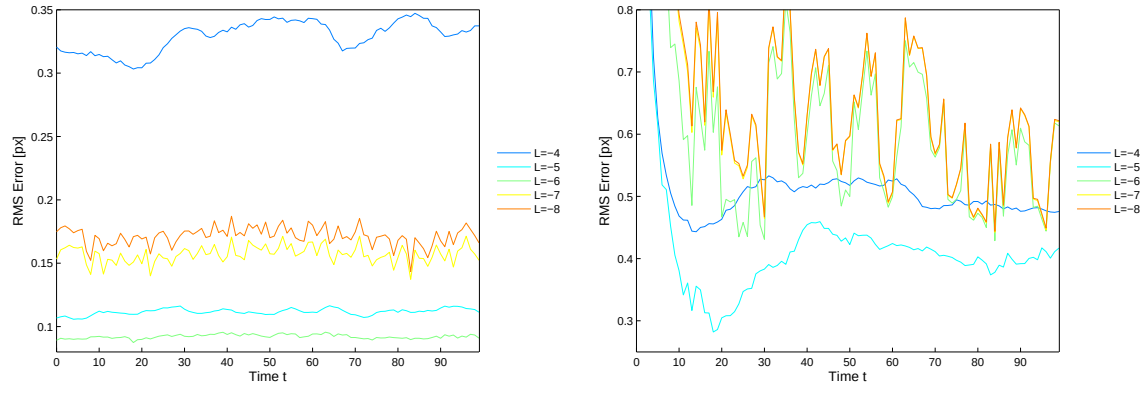
(ii) however, it has been observed that in the presence of strong non-linearities, in particular for large displacement, a high value of C will result in the impossibility to capture those high-amplitude motions.

Results obtained with various values of C are given in Fig. 8.7 for IHT & wake. Contrary to the previous experiment with L, this parameter C has a clear influence on the estimator ability to cope with larger displacements as n increases. With small displacements ($n < 20$), it suffices to take $C = -5$ to obtain satisfactory results; it corresponds to a decomposition with a coarse approximation and one detail scale only. With larger magnitude motions however, it is necessary to use at least $C = -3$, which corresponds to a 3 detail scale decomposition. When more detail scales are considered, the price to pay is a longer estimation time: with IHT images, it increases by 25% with $C = -3$ instead of $C = -4$, and by 100% with $C = 0$. With appropriate C values, the optimal displacements range is still around $[0.5; 5]$ px. Figure 8.8 gives results for particle & scalar sequences. Contrary to IHT & wake datasets which feature large amplitude motions at large n, here displacements stay within the "comfort zone" (below 5 px) all sequence long. Almost identical solutions are obtained for any $0 \leq C \leq L + 1$. In terms of computations, $C = L + 1$ is the best choice.



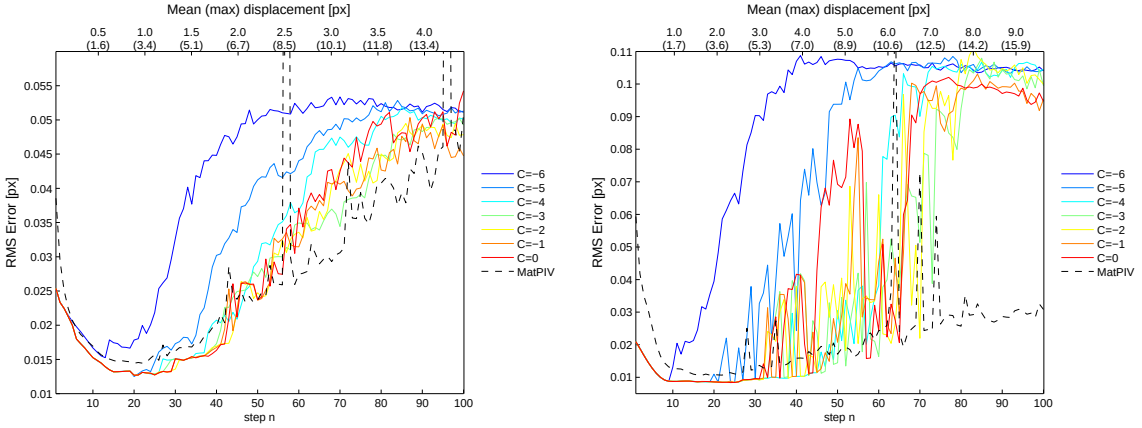
(a) IHT (b) wake

Figure 8.5: Influence of fine scale parameter L. Optimum value is $L = -6$ for both IHT & wake sequences. The displacements range for which the estimator gives the best results is of order $[0.5; 5]$ px, it does not depend on F.



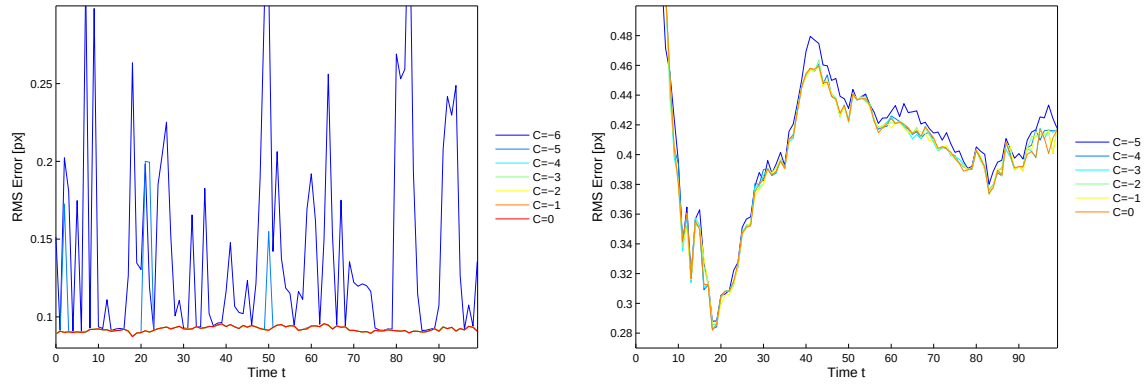
(a) particle (b) scalar

Figure 8.6: Influence of fine scale parameter L. Optimum value is $L = -6$ for particle and -5 for scalar.



(a) IHT (b) wake

Figure 8.7: Influence of coarse scale parameter C, having set $L = -6$. When displacements magnitude rises up ($n > 20$), it is necessary to consider "deeper" decompositions (i.e. $C \to 0$) to maintain the accuracy. The optimal displacement range is therefore linked to C.



(a) particle (b) scalar

Figure 8.8: Influence of coarse scale parameter C, having set $L = -6$ for particles and -5 for scalar. Almost identical solutions are obtained for any $0 \leq C \leq L + 1$. In terms of computations, $C = L + 1$ is the best choice.

8.2.2 Impact of Vanishing Moments

Basis functions with a larger support "analyze" the signal over a wider area, hence we may wonder if wide supports enable a better handling of large displacements. Since the size of basis functions support is linked to their number of VM, we display in Fig. 8.9 results obtained for IHT and wake sequences using various VM. Low VM values (1, 3) lead to a generally lower accuracy which rapidly deteriorates as the observable displacements magnitude increases. Also, increasing much the number of VM does not further widen the optimal displacement range, which is still around [0.5;5] px.

8.2.3 Choosing the Data Model

Experiment on fine scale parameter L (Fig. 8.5) is repeated with the IHT sequence, using this time the *linear* OFC data model – Eq. (5.14). This model does not require to warp image I_1 with $\boldsymbol{v}$ each time the functional is evaluated; it is therefore incomparably faster yet only remains valid within the linearity region of the image brightness (Fig. 2.4). For small n our image sequence shows very small displacements, likely to be valid with respect to the linear region, therefore it may be advantageous to use the OFC instead of DFD. A very simple pyramidal scheme is added on top of the estimator to try to extend the validity domain.

Figure 8.11 compares errors given by DFD and OFC data models. Although the OFC shows a much more regular behavior, its performance are far beyond those of the DFD, even at low amplitudes. The two-level pyramidal approach reduces the gap; the three-level pyramid shows a more unstable behavior. The OFC may however be used in as a first approach to obtain very quick results.

8.2.4 The Null-Divergence Constraint

For the particle and scalar sequences, the flow is divergence-less. We may now question the interest of incorporating the divergence-free constraint directly by the means of divergence-free wavelet bases introduced in Section 4.1. Coarse scale parameter C is necessary 0, as explained in Appendix A.3. Estimations with various L parameter values are achieved, results are given Fig. 8.12 to be compared with those of Fig. 8.6. Since the construction of divergence-free bases requires to derivate the wavelet, a more regular function was chosen: 10 VM instead of 5 VM in previous experiment. As illustrated in Fig. 8.10, using more VM should not drastically change results.

Optima L values with divergence-free bases are the same as with usual bases, i.e. -6 and -5 for particle and scalar, respectively. Divergence-free bases clearly enhance the estimate accuracy: mean RMS error, over the 100-frame sequences, drops down from 0.092 px to 0.072 px (-22%) with particles. It is much more spectacular with scalar images: from 0.45 px down to 0.26 px (-42%) thank to divergence-free bases.

Vorticity of estimates obtained from either usual or divergence-free bases are compared Fig. 8.13. Vorticity maps look rather alike at first glance, yet a closer look reveals spurious structures in estimates on usual bases that are corrected thanks to the divergence-free constraint, especially regarding scalar results (Fig 8.13d and 8.13e).

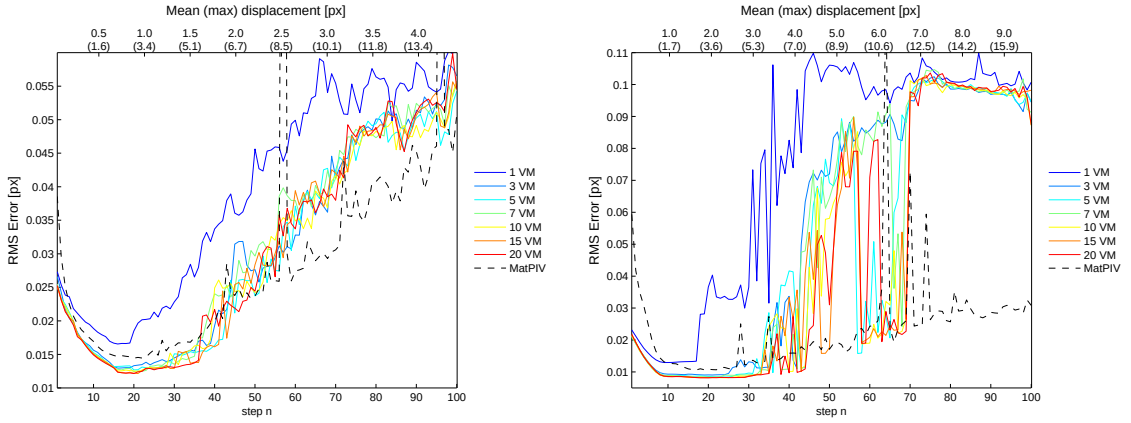
8.2.5 Synthesis

Before considering the addition of regularization schemes and the estimation up to the finest pixel scale, let us sum-up what previous experiments taught us.

(i) The wavelet-based approach gives its best results for apparent displacements laying roughly within $[0.5; 5]$ px.

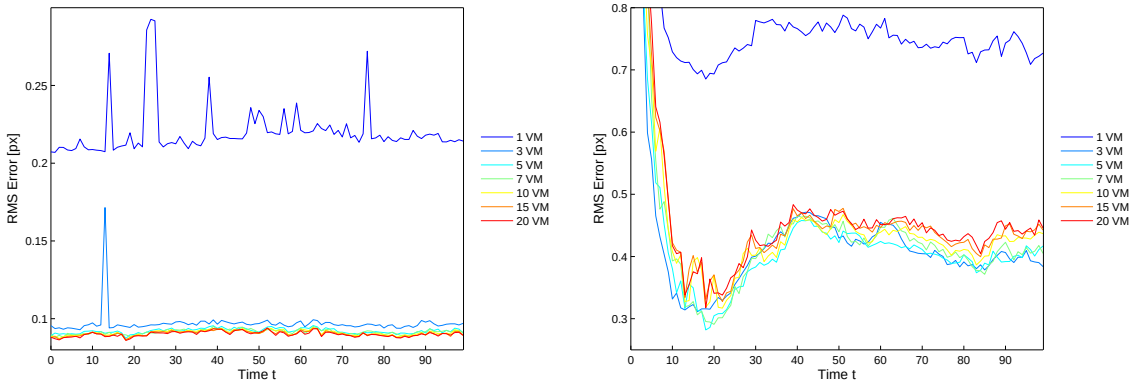
(ii) The fine scale cut-off parameter L must be chosen accordingly to input images characteristics, in order to exclude scales where the aperture problem becomes prominent (Fig. 8.6).

(iii) The choice of the coarsest scale parameter C influences the ability to cope with large displacements (Fig. 8.6). A basis with several detail scales (i.e. C close to 0) is to be preferred.



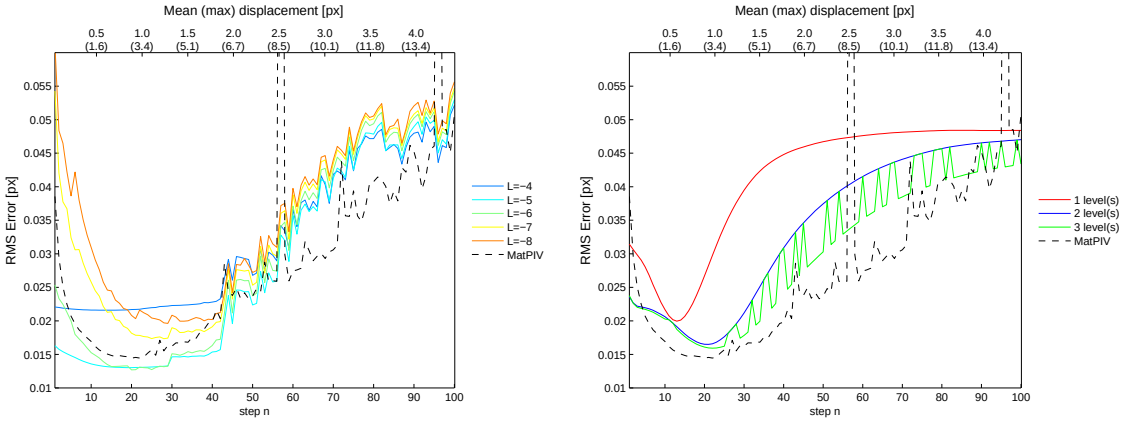
(a) IHT (b) wake

Figure 8.9: Influence of vanishing moments. Estimations are achieved with Daubechies wavelets having 1, 3, 5, 7, 10, 15 and 20 VM. Very low values not only lead to low accuracy but also show a worse behavior with high amplitude motions. Choosing more VM helps reducing the error but is more expensive in terms of computation time: +30% from 5 to 20 VM with IHT images.



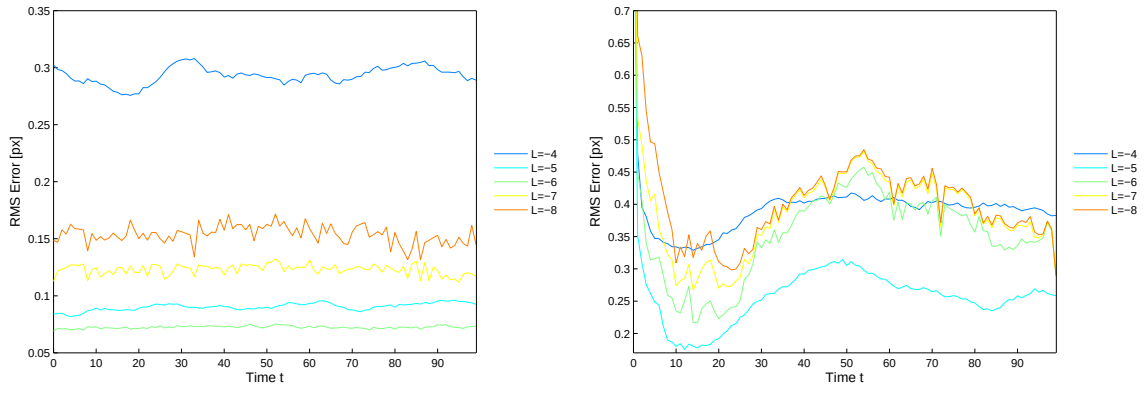
(a) particle (b) scalar

Figure 8.10: Influence of vanishing moments. Particle sequence (*left*) shows a behavior similar to IHT and wake PIV sequences (Fig. 8.9). Regarding scalar sequence however (*right*), raising too much the number of VM slightly deteriorate the estimation accuracy, which is best with 5 VM only.



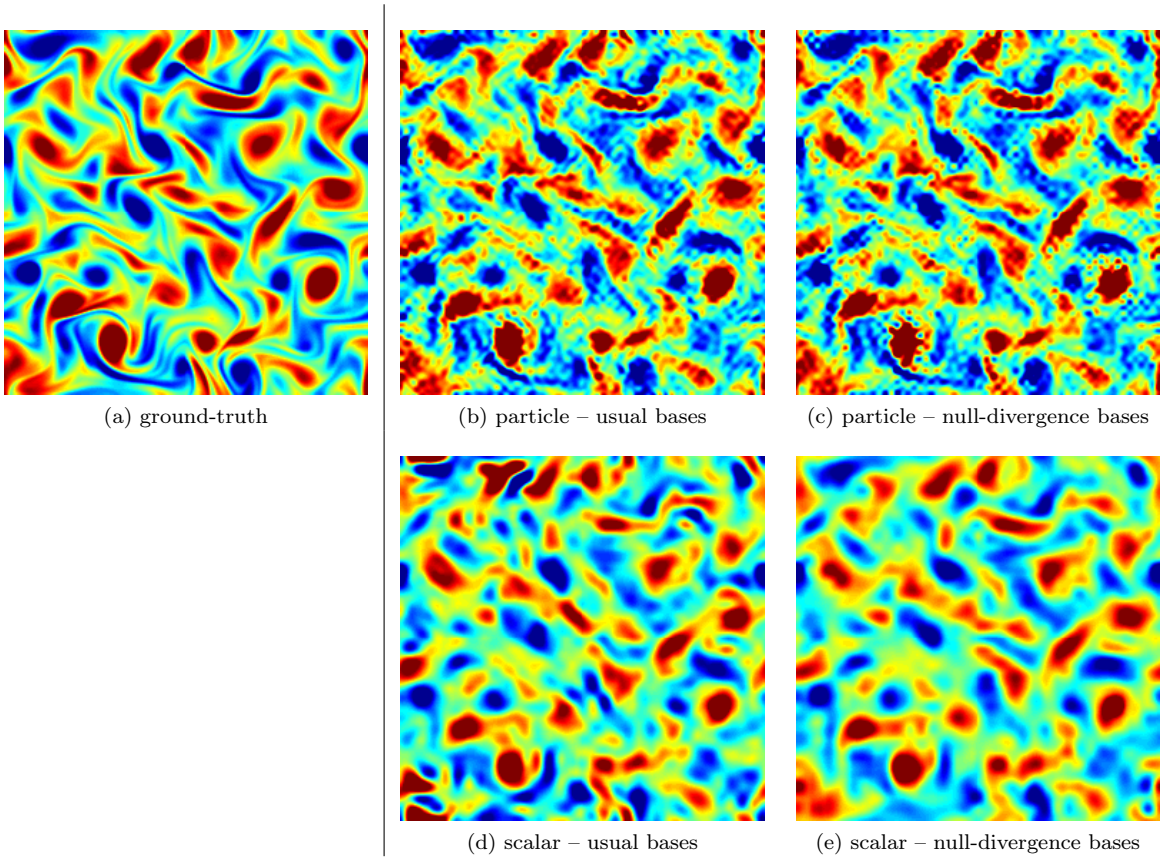
(a) IHT with DFD (b) IHT with OFC

Figure 8.11: Results previously obtained with DFD (*left*) are compared to OFC estimates (*right*), with an optional pyramidal scheme (levels > 1) and parameters $(L, C) = (-5, 0)$. If accuracy is much lower than with DFD, so is the computation time: for $n = 20$, it took 25.6 s with DFD versus 3.6 s only with OFC and 2 levels.



(a) particle (b) scalar

Figure 8.12: Interest of null-divergence bases. Results are to be compared with those of Fig. 8.6. Best L values are similar to those of the previous experiment: -6 and -5 for particle and scalar, respectively. With these parameters, divergence-free bases drops down the mean RMS error, over the 100-frame sequences, by 22% for particle and 42% for scalar.



(a) ground-truth (b) particle – usual bases (c) particle – null-divergence bases

(d) scalar – usual bases (e) scalar – null-divergence bases

Figure 8.13: Ground-truth vorticity Fig. (a). Comparison of vorticity fields of estimates obtained using either usual (*middle*) or divergence-free bases (*right*), for particle (*top*) and scalar (*bottom*) images.

(iv) Very low values of vanishing moments should be avoided, as they lead to poorly accurate solutions and negatively influence the ability to deal with largest displacements (Fig. 8.9). High values do not bring in much, in terms of RMS error, and increase the computation time.

(v) The DFD data model gives much better results than the OFC and is more adaptable to large displacements (Fig. 8.11). OFC model is incomparably faster.

(vi) Null-divergence bases enhance estimation of null divergence flows (Fig. 8.12), especially when images gives poor information – e.g. with scalar images.

8.3 Adding Regularizations

After playing with the various parameters of the basic motion estimator which relies on small scales truncation, we will now consider full-scale estimations (i.e. up to pixel scale) using regularizers introduced in Chapter 6. We will focus on the norm and the vorticity of the solution, as well as on the RMS error (8.1). As a differential quantity, vorticity enlightens the structures of the flow and also emphasizes its local variations.

Regarding explicit smoothing schemes (Sections 8.3.1 and 8.3.2), we will examine the behavior of the solution as parameter α, which balances between data term and regularization – Eq. (6.2), is increased. For each regularizer, two estimates obtained from particle images will be presented: one using the "best" α in terms of RMS error and another one over-regularized (i.e. with a too high α), in order to exhibit the effects of the various schemes. Optimal α values were tuned for each case by methodic searches, performing estimations for image pair at instant $t = 20$ with every $\alpha = a \times 10^b$, $a \in [1; 9]$ and $b \in \mathbb{Z}$.

8.3.1 Discrete Approximation of High-Order Schemes

We consider first the simple regularization schemes introduced in Section 6.2.1 that rely on norm equivalence (Section 4.2.1). Figures 8.14 and 8.15 display results obtained for orders 1 and 2, respectively. As expected, the order 2 regularization gives better results than the order 1: RMS error is lower and the vorticity map presents a better aspect, i.e. better-defined structures and less noise.

8.3.2 Continuous Approximation of High-Order Schemes

Continuous approximation of Section 6.2.2 enables to design more general schemes than the discrete approximation does. Following results were obtained using regularizers:

- $\|\nabla v_1\|^2 + \|\nabla v_2\|^2$, "Horn & Schunck", Eq. (6.20): Fig. 8.16.
- $\|\operatorname{div}(\boldsymbol{v})\|^2$, "div", Eq. (6.21): Fig. 8.17.
- $\|\Delta v_1\|^2 + \|\Delta v_2\|^2$, "laplacian", Eq. (6.22): Fig. 8.18.
- $\|\nabla(\operatorname{curl}(\boldsymbol{v}))\|^2$, "grad(curl)", Eq. (6.23): Fig. 8.19.
- $\|\operatorname{div}(\boldsymbol{v})\|^2 + \|\nabla(\operatorname{curl}(\boldsymbol{v}))\|^2$, "div + grad(curl)": Fig. 8.20.
- $\|\nabla(\operatorname{curl}(\boldsymbol{v}))\|^2 + \|\nabla(\operatorname{div}(\boldsymbol{v}))\|^2$, "grad(div) + grad(curl)": Fig. 8.21.
- divergence-free basis + $\|\nabla(\operatorname{curl}(\boldsymbol{v}))\|^2$, "div-0 + grad(curl)": Fig. 8.22.

These tests confirm the expected effects of the various regularizers – for instance, best "grad-curl" estimate (Fig. 8.19) indeed shows more regular blobs of vorticity than best "div" estimate (Fig. 8.17), whereas vorticity of "Horn & Schunck" estimate (Fig. 8.16) remains noisy. According to the chosen quality criterion (RMSE on the velocity fields), overall best results are obtained by divergence penalization, whether by an explicit regularization term (Fig. 8.17) or by the use of divergence-free bases (Fig. 8.22). This is not surprising: ground-truth motion being divergence-free, these regularizers have the most physical meaning. It is however interesting to notice that the simple divergence penalization gives even better results, in terms of velocity RMSE, than the more evolved divergence-free basis associated to curl gradient penalization. Should the RMSE be computed on the vorticity map, this ranking would have been probably different, judging from the visible artifacts in "div" penalization vorticity map (Fig. 8.17c). This point emphasizes the difficulty of choosing an appropriate quality criterion.

8.4 Comparisons to State-of-the-Art

Comparison between proposed approaches

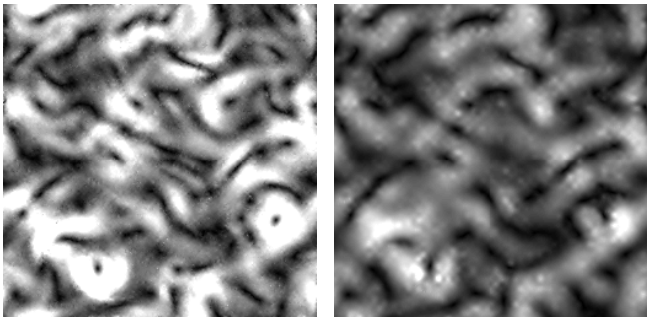
In the first place, the various proposed regularizations are compared between each others:

- small-scale truncation, both with usual and divergence-free bases;
- divergence penalization;
- divergence and curl gradient penalization;
- divergence-free bases with curl gradient penalization.

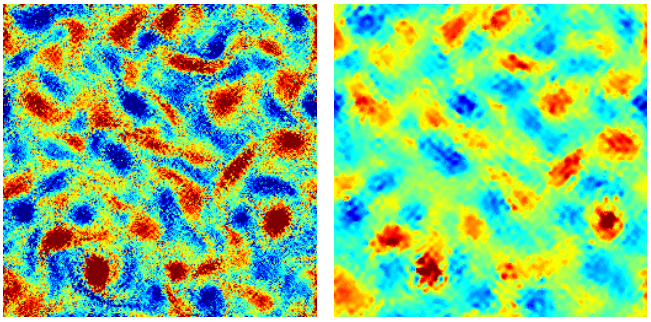
When needed, α parameter values are the "optimal" values found by methodic searches at instant $t = 20$, as explained in Section 8.3. Figure 8.23 displays RMS error values measured over the 100 first frames of both particle and scalar sequences.

Let us first focus on particle images. Regarding the basic estimator with small scales truncation, the use of divergence-free bases clearly enhances the estimates by incorporating the physical constraint. Then, as already stated in regularizer tests of Section 8.3.2, the simple divergence penalization clearly outperforms any other approach – although showing a slightly irregular behavior.

Scalar sequence results illustrate the difficulty of choosing the α parameter. As explained above, these "optimal" α values where determined from many experiments at $t = 20$. Nevertheless, due to the diffusion process occurring in the scalar sequence, the characteristics of these scalar images slightly evolve along the sequence (e.g. less uniform areas, but also less contrast). Therefore, and contrary to the particle sequence, there is absolutely no reason to assume that this optimal α obtained at $t = 20$ would still be optimal several frames later. This is particularly visible Figure 8.23b, where estimators with a penalization term show a good behavior around $t = 20$, but are later clearly outperformed by the truncated divergence-free basis.

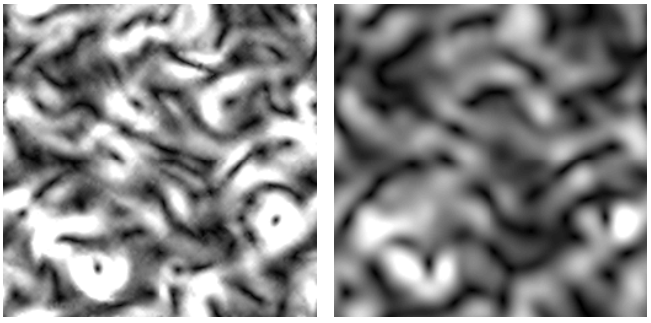


(a) 0.102 (2×10^{-2}) (b) 0.470 (2×10^{0})

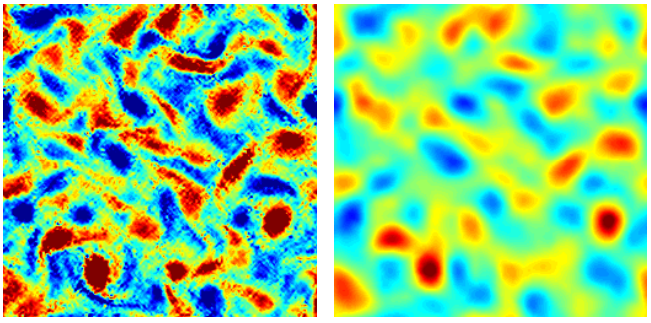


(c) (d)

Figure 8.14: Discrete approximation, order 1. Vector norm (*up*) and vorticity (*bottom*). RMS error (pixels) below norm images, with α value between brackets.

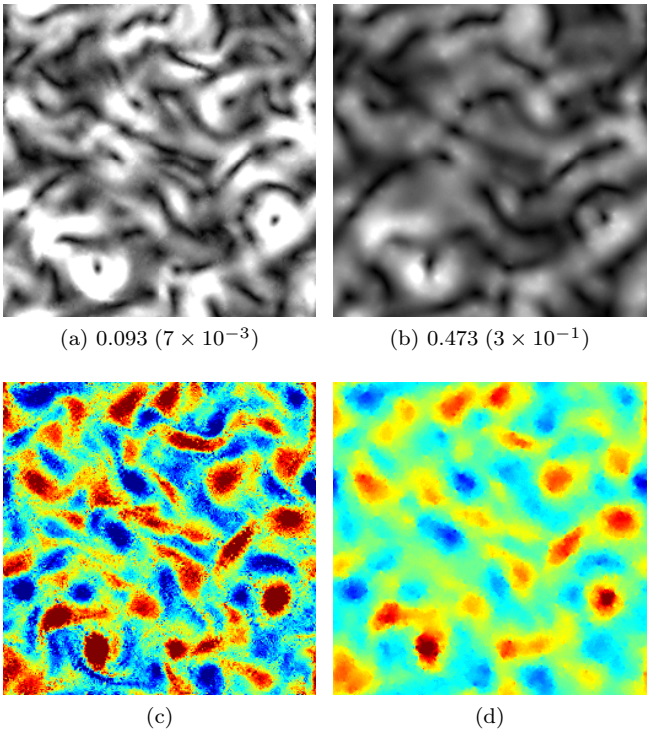


(a) 0.085 (3×10^{-1}) (b) 0.450 (5×10^{2})



(c) (d)

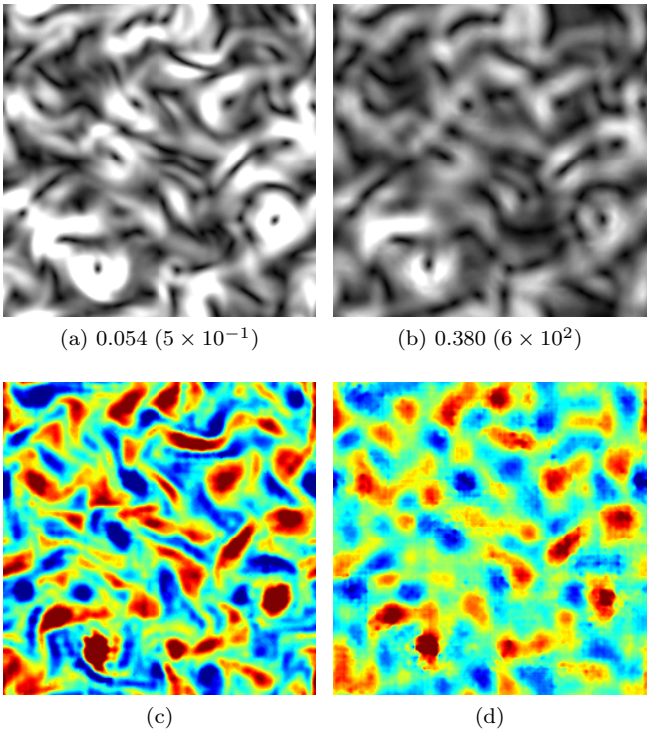
Figure 8.15: Discrete approximation, order 2.



(a) 0.093 (7×10^{-3}) (b) 0.473 (3×10^{-1})

(c) (d)

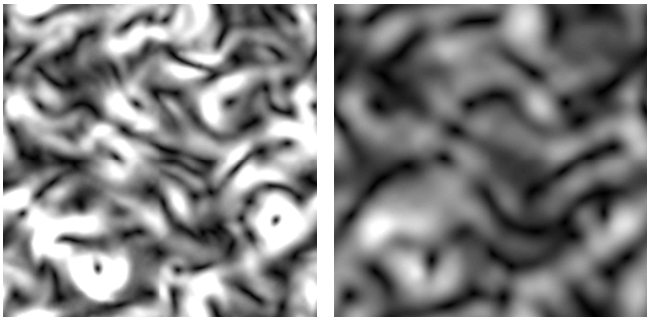
Figure 8.16: Continuous approximation, Horn & Schunk. Vector norm (*up*) and vorticity (*bottom*). RMS error (pixels) below norm images, with α value between brackets.



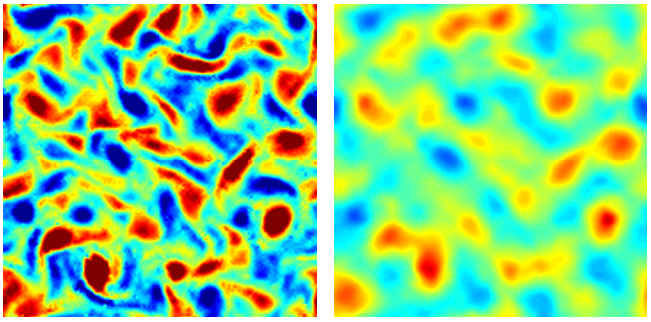
(a) 0.054 (5×10^{-1}) (b) 0.380 (6×10^{2})

(c) (d)

Figure 8.17: Continuous approximation, div.

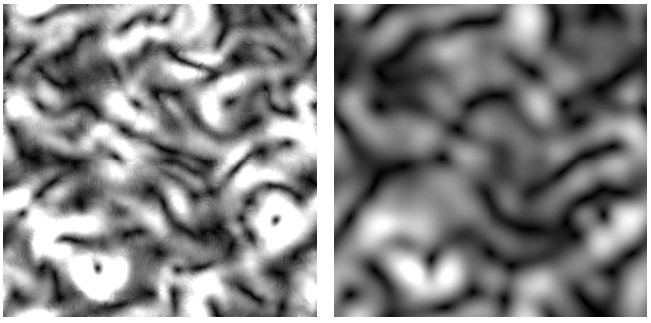


(a) 0.074 (2×10^{-2}) (b) 0.537 (2×10^{1})

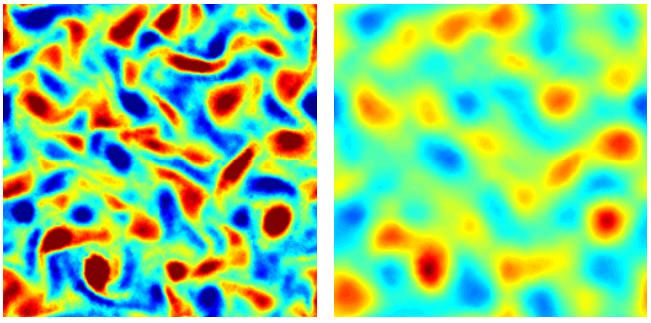


(c) (d)

Figure 8.18: Continuous approximation, laplacian. Vector norm (*up*) and vorticity (*bottom*). RMS error (pixels) below norm images, with α value between brackets.

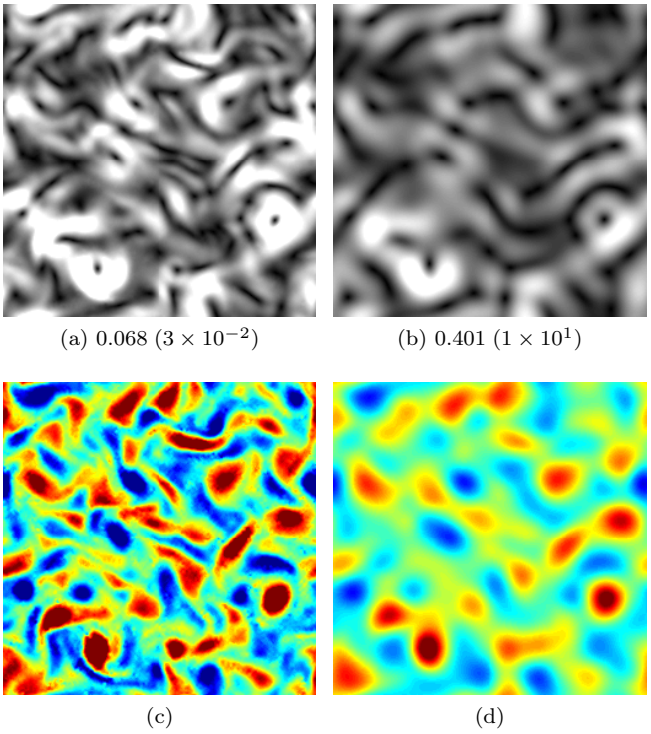


(a) 0.092 (4×10^{-2}) (b) 0.526 (2×10^{1})



(c) (d)

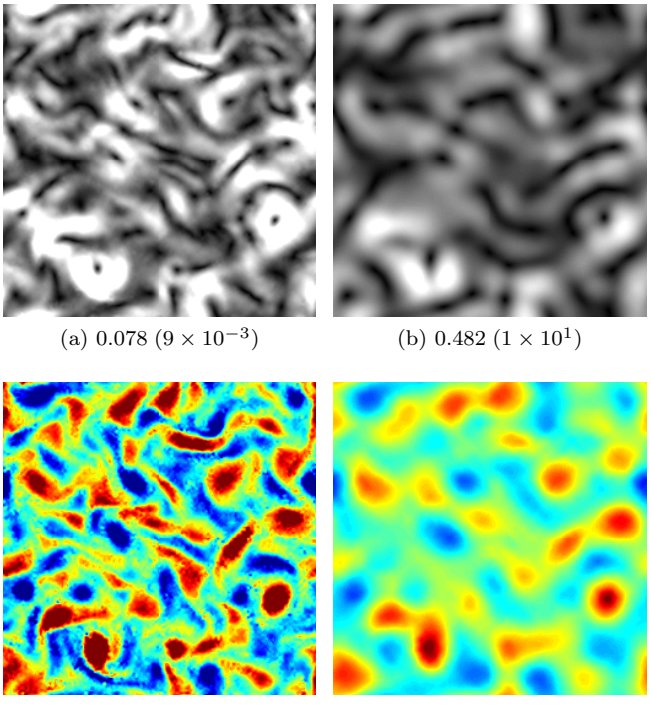
Figure 8.19: Continuous approximation, grad(curl).



(a) 0.068 (3×10^{-2}) (b) 0.401 (1×10^{1})

(c) (d)

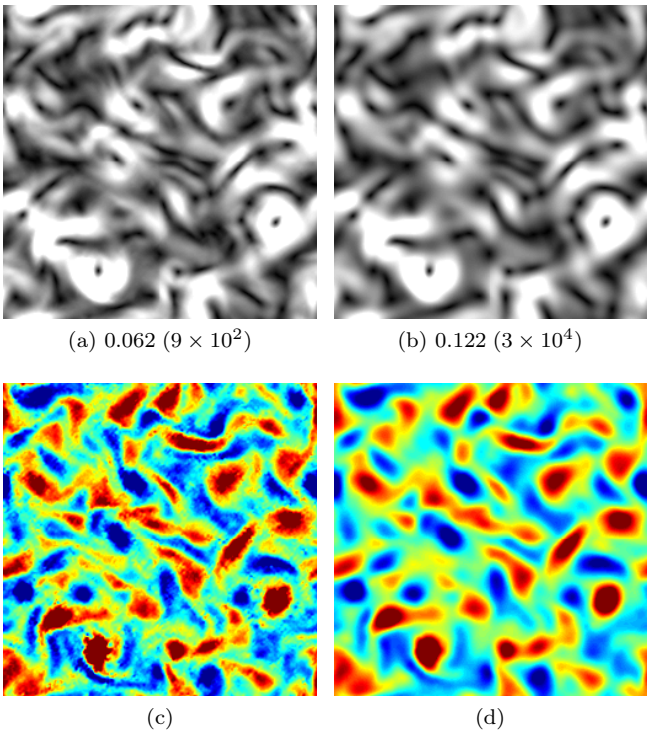
Figure 8.20: Continuous approximation, div & grad(curl). Vector norm (*up*) and vorticity (*bottom*). RMS error (pixels) below norm images, with α value between brackets.



(a) 0.078 (9×10^{-3}) (b) 0.482 (1×10^{1})

(c) (d)

Figure 8.21: Continuous approximation, grad(div) & grad(curl).



(a) 0.062 (9×10^2) (b) 0.122 (3×10^4)

(c) (d)

Figure 8.22: Divergence-free & continuous approximation, grad(curl). Vector norm (*up*) and vorticity (*bottom*). RMS error (pixels) below norm images, with α value between brackets.

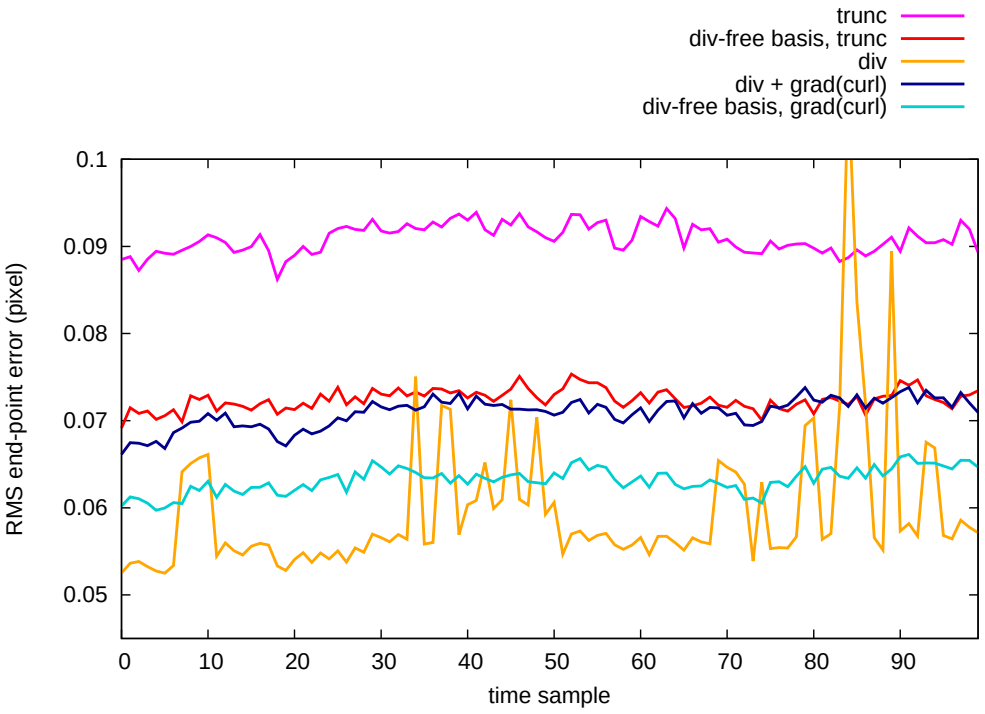
Comparison to state of the art

The best sequences for the small-scale truncation and high-order regularization are then compared to other state of the art methods:

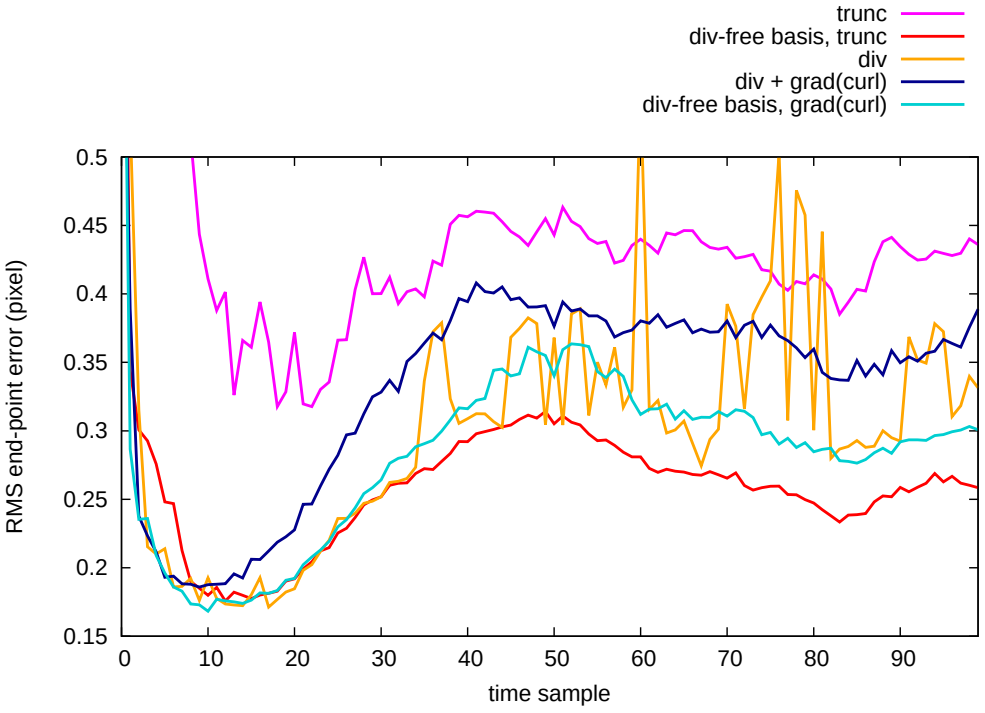
- Horn & Schunk [20] first-order regularization;
- Heas *et al.* autosimilarity regularization [18];
- Becker *et al.* adaptive correlations [4];
- Yuan *et al.* mimetic difference [45];
- LaVision's cross-correlations (DaVis software).

Results for the 100 first frames of the particle and scalar sequences are given Figure 8.24. Regarding the particle sequence, both proposed approaches (truncated divergence-free basis; usual basis with divergence penalization) outperform most of state of the art methods. Let us note that the only competitive approach, Heas *et al.* autosimilarity regularization, is very well suited to homogeneous isotropic turbulent flows such as in this test sequence. It is however not adapted to flows showing different regimes in the same domain (laminar, transition toward turbulence).

On the scalar sequence, performances of our methods (divergence-free basis, truncated or with curl gradient penalization) are slightly less satisfactory. Similarly to previous tests, estimators featuring an explicit smoothing term, and therefore an α parameter, see their results progressively deteriorate. It illustrates again one the difficulty of setting α, and from that point of view, our simple estimator with a truncated divergence-free basis seems very reasonable.

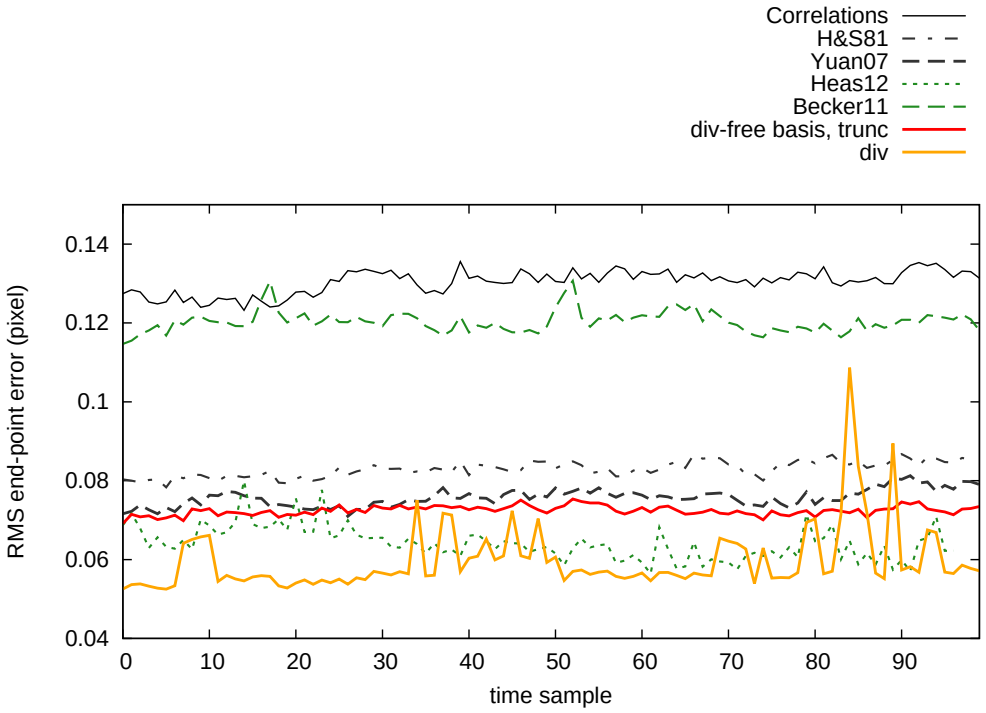


(a) particle

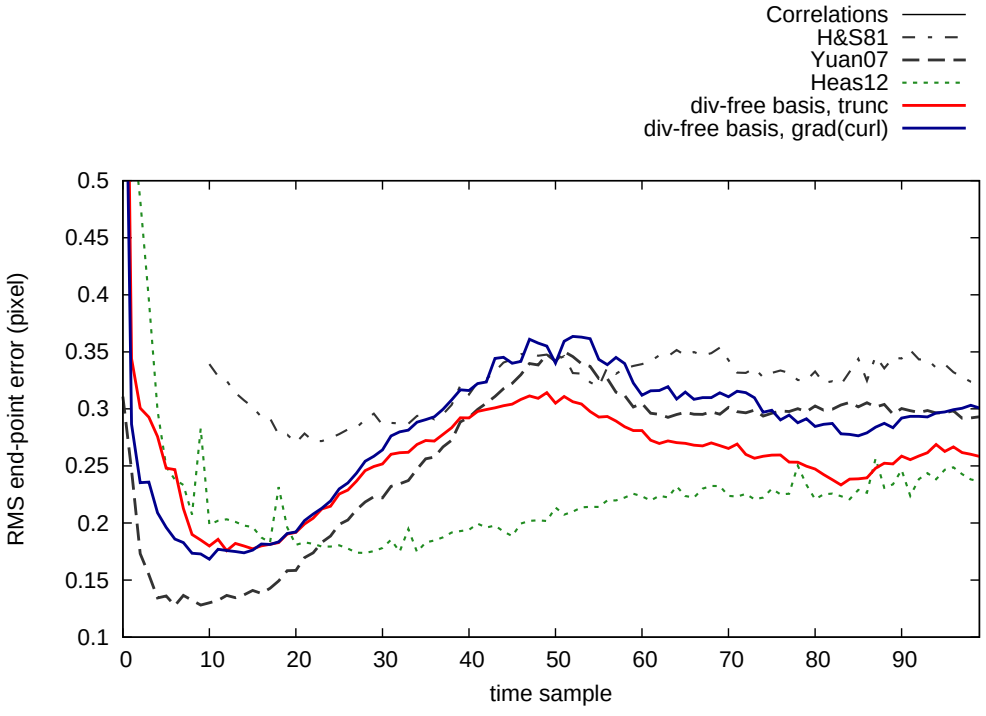


(b) scalar

Figure 8.23: Comparison between various proposed approaches, for the particle (*top*) and scalar (*bottom*) sequences.



(a) particle



(b) scalar

Figure 8.24: Comparison to state of the art, for the particle (*top*) and scalar (*bottom*) sequences.

Chapter 9

Experimental Results

After assessing the performances of proposed wavelet-based approaches on synthetic data, we will now focus on their application to actual images obtained during laboratory experiments. Without any ground-truth, estimates will be compared to measurements obtained using alternative approaches, such as vision-based cross-correlations and sensor-based hot-wire anemometry. Two classical configurations of turbulence physics will be investigated: a turbulent mixing layer and a cylinder wake.

9.1 Cylinder Wake

This first experiment was conducted in 2011 at IRSTEA Rennes [1] by A. Guibert & D. Heitz. It consists of several configurations of cylinder wake flows, using either one or three cylinders and different spacial and/or temporal resolutions. For this study, we chose a configuration which has already been extensively studied: the cylinder wake at Reynolds 3900. Set up of the experiment is detailed in Figure 9.1. Unless explicitly stated, all results presented hereafter have been nondimensionalized with respect to the cylinder diameter length d and far flow velocity v^∞ :

$$\begin{aligned} &x[\text{mm}] \Rightarrow x/d\,; && y[\text{mm}] \Rightarrow y/d\,; \\ &v_1[\text{m/s}] \Rightarrow v_1/v^\infty\,; && v_2[\text{m/s}] \Rightarrow v_2/v^\infty\,; \\ &\text{etc.} \end{aligned}$$

Cross-correlations estimates for this 3072 frame-long sequence have been obtained using Lavision's software `Davis`, using 32×32 px windows with 50% overlay. This gives a sparse field of 64×64 velocity vectors. Several estimations were then performed with the wavelet-based algorithms, using Daubechies wavelet with 10 VM. Before going into details, Figure 9.2 presents vorticity maps of the near-wake, obtained from three different estimates:

(i) using cross-correlations (Section 2.2.1): 64×64 estimated vectors;

(ii) or the simple estimator with small-scale truncation (Section 6.1): 5 smallest scales neglected, leaving as many unknowns as with (i), but representing 1024×1024 vectors;

(iii) and the discrete second-order regularization (Section 6.2.1), with parameter $\alpha = 10$: 1024×1024 estimated vectors.

All three fields are alike, both in terms of structures and magnitude. The estimate given by neglecting small scales – (ii) and Figure 9.2b – corresponds quite well to the one obtained from cross-correlations, (i) and Figure 9.2a, but shows much more regularity. The use of an explicit smoothing scheme further enables to recover finer scales and better defined structures – (iii) and Figure 9.2c. It should be noted that the unknowns of the two first estimates represent only 0.4% of the total unknowns, so that (i) and (ii) can be considered as rather efficient estimations. In the following we will restrict ourselves to the comparison between cross-correlations (i) and optic flow with explicit smoothing terms (iii).

1. `http://www.irstea.fr/linstitut/nos-centres/rennes`

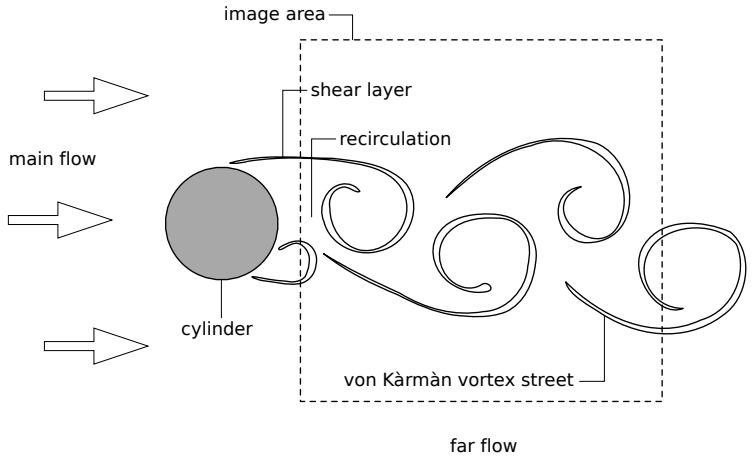


Figure 9.1: Configuration of the cylinder wake experiment (not on scale). Cylinder diameter is $d = 12\,\mathrm{mm}$; the $1024 \times 1024\,\mathrm{px}$ image area covers $78 \times 78\,\mathrm{mm}$. Image pairs are acquired at $1.5\,\mathrm{kHz}$ ($\simeq$ every $666\,\mu\mathrm{s}$), while the time step between the two frames of a pair is $70\,\mu\mathrm{s}$. To obtain a Reynolds of 3900 with a kinematic viscosity $\nu = 15.6 \times 10^{-6}\,\mathrm{m}^2/\mathrm{s}$, input flow velocity v^∞ is $5.07\,\mathrm{m/s}$, which corresponds to a displacement of $\simeq 4.7\,\mathrm{px}$ within a frame pair.

9.1.1 Mean Flow

From the complexity of turbulence dynamics, as quickly mentioned in Section 1.3.1, the analysis of turbulent flows requires a statistical description. A first step consists in studying the agreement between both methods for the time-averaged motion:

$$\bar{v}_i = \frac{1}{N} \sum_{n=1}^{N} v_i(n)\,, \quad i = 1, 2\,; \tag{9.1}$$

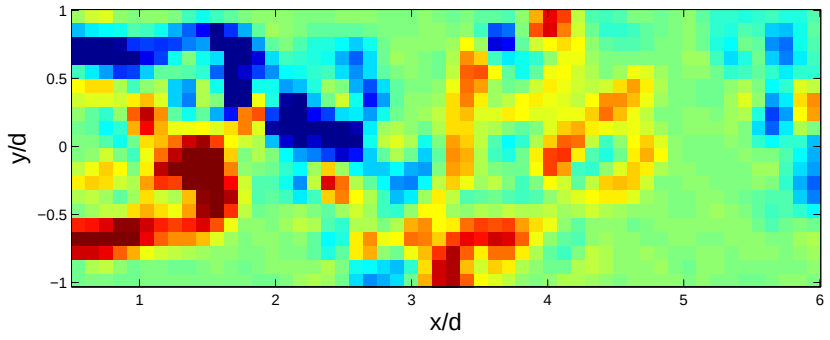
with $v_i(k)$ the i-th motion component estimated at discrete time k. These mean flows, computed over the $N = 3072$ estimates given by both methods, are displayed Figure 9.3. Cross-correlations and optical flow results are likely, again both in terms of structures and amplitudes. Some small deformations are visible in correlations estimate Figures 9.3a and 9.3a, this is caused by a halo-shaped lightning artifact in input images that could be corrected prior to applying optical-flow. Spurious structures located at the right border can be noticed for both estimates, they are also caused by lightning artifacts. Optic flow results seems however less sensitive to these artifacts, maybe thank to regularization schemes. In order to quantify differences between both results, cross-correlations estimate is interpolated to match with optic-flow grid, using `Matlab`'s cubic interpolation. Absolute relative differences with respect to optic flow values are then computed. The two mean first components V_1 show a remarkable agreement in Figure 9.3e: there is less than 2.5% difference almost everywhere, save for the recirculation region. Differences on the second mean component V_2 enlighten the consequences of lightning artifacts Figure 9.3f.

9.1.2 Cross Fluctuations

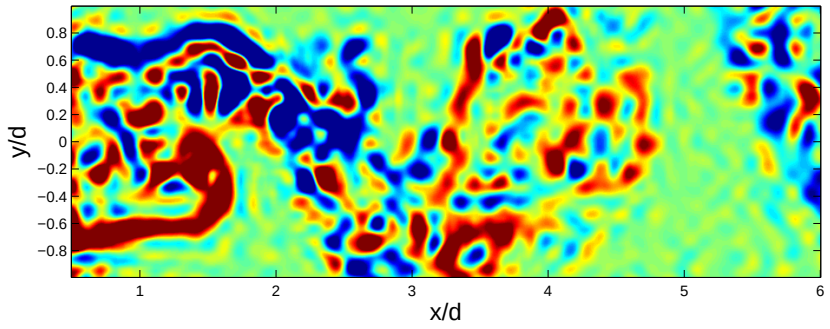
The second quantities to be examined are the mean Reynolds tensions:

$$\langle v_i' v_j' \rangle = \frac{1}{N} \sum_{n=1}^{N} \left(v_i(n) - \bar{v}_i\right)\left(v_j(n) - \bar{v}_j\right)\,, \quad i, j = 1, 2\,, \tag{9.2}$$

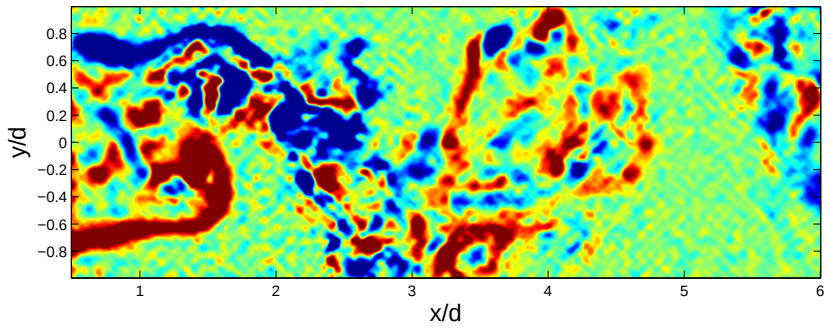
where $\bar{v}_i$ are components of the mean flow (9.1). Cross fluctuations $\langle v_1' v_2' \rangle$ are the most interesting here, they are displayed in Figure 9.4. Again, both cross-correlations and optical flow estimates lead to very similar results, with a better resolution for the optical flow.



(a) cross-correlations (i)



(b) truncated small-scales (ii)



(c) second-order regularization (iii)

Figure 9.2: Comparison of vorticity maps of the near-wake, for three different estimates.

9.1.3 Skewness

Skewness is the third moment of the velocity field considered as a random variable (Section 1.3.1); it quantifies the asymmetry of the signal. Here it is estimated by:

$$\gamma_i = \frac{\frac{1}{N}\sum_{n=1}^{N}\left[v_i(n) - \bar{v}_i\right]^3}{\frac{1}{N}\left[\sum_{n=1}^{N}\left[v_i(n) - \bar{v}_i\right]^2\right]^{3/2}}\,. \tag{9.3}$$

γ_2 (i.e. for second component v_2) is displayed Figure 9.5. Again, both results are in good agreement. The finer resolution brought by optical flow however reveals periodic patterns. Profiles are extracted

along black dotted lines of Figure 9.5a and 9.5b, and plotted below, Figure 9.5c. The spatial step between two cross-correlations vectors is of the same order as the wavelength of the signal, which makes it harder to detect. This wavelength λ/d is of order 0.23 to 0.25; it may be connected to the vortices of Kelvin-Helmholtz instability shown Figure 9.6. These vortices, much smaller than von Kármán's, develop in the shear layer. Their wavelength λ/d is indeed of order 0.25.

This example illustrates one of the advantages of the optical flow approach over the cross-correlations: with the dense estimation, more structures are resolved; it give access to a more complete description and therefore a better understanding of the dynamics of the studied configuration.

9.1.4 Power Spectral Density

Temporal signals are extracted from the 3072-frame sequence, at various x locations in the near wake, and at y points corresponding to different situations: the far flow, the shear layer, the recirculation region (see Figure 9.1). A power spectral density (PSD) estimate is obtained for each temporal signal using Welch's method with 512 point windows. These so-called periodograms are displayed Figure 9.7, for $x = -2.33$ (recirculation region) and $x = 0.42$ (near wake). For symmetry reasons, spectra corresponding to the upper half of the flow only are shown.

The frequency peak corresponding to von Kármán instability shows up at 77.6 Hz, in agreement with spectra computed from cross-correlations results (not shown). A nondimensional number is usually associated to this vortex shedding frequency: the *Strouhal* (St), defined as:

$$St = \frac{fL}{v}\,; \tag{9.4}$$

where f is the frequency of the phenomenon (in Hz), L and v are characteristic length and velocity (in m and m/s, respectively). Using $L = d$ (the cylinder diameter) and $v = v^\infty$, the von Kármán frequency corresponds to $St = 0.183$. This value is slightly below references from the literature which rather give a Strouhal between 0.2 and 0.21 at Reynolds 3900, e.g. in Ong & Wallace [33], Dong *et al.* [14] or Parnaudeau *et al.* [35]. However, this Strouhal value highly depends on the choice of the characteristic velocity v. Here, it can be noted that in practice, observed far flow velocity values are closer to $0.9 \times v^\infty$ (see Figure 9.3c). Using this value for the Strouhal gives this time $St = 0.204$. The Kelvin-Helmholtz instability peak, although less well-defined than the von Kármán, can be seen within the shear layer around 300 Hz to 400 Hz.

9.2 Turbulent Mixing Layer

This experiment was also conducted at IRSTEA Rennes, by J. Carlier, A. Guibert and K. Sodjavi in 2012. Two parallel flows at different velocities meet at the output of two wind tunnels. The different velocities create a shear at the interface of the flows, where Kelvin-Helmholtz instabilities develop. Flow visualization is also achieved by Particle Image Velocimetry. Several time steps were used in order to obtain different magnitudes of apparent displacements, more adapted either to optical flow or cross-correlations measurements. The main difficulty in processing these sequences comes from the fact that vertical displacements are extremely small, compared to the main horizontal motion. Setup of the experiment is given Figure 9.8.

Cross-correlations estimates for the 3072 frame-long sequence have been obtained using `GPiv` [42] free software, using 32×32 px windows with 50% overlay. Just like the cylinder wake experiment, it gives a 64×64 vector field. Only the sequence featuring the largest time-step, 300 µs (and therefore the largest displacements), was processed with this cross-correlation software. Estimates from our proposed approach were achieved for all three sequences (time-steps of 100 µs, 200 µs and 300 µs) using a 7 VM Daubechies wavelet, with first-order regularization and parameter $\alpha = 0.05$ and 0.1. In order to facilitate the convergence when large displacements are involved, a rough approximation of the mean horizontal displacement (a scalar value) was provided to the algorithm as a first guess. Optical flow and cross-correlations estimates from a sample image pair are presented Figure 9.9. Statistics obtained by hot-wire anemometry will be used as well, as a reference. Let us point out that these statistics were acquired over a longer time than the 3072 image pairs cover, thus are much better converged than what will be obtained from computer-vision estimates.

9.2.1 Profiles Comparisons

Mean velocity profiles are displayed in Figure 9.10. First component $\bar{v}_1$ profiles (Figure 9.10a) are in relatively good agreement, save for the borders. Indeed, as warned in Section 5.2.4, estimated velocity fields with our proposed approach are periodic due to the periodization of wavelet bases. It was already the case with the synthetic datasets and cylinder wake sequence, but was less noticeable and/or prejudicial. Here, despite the difference of horizontal velocities at opposite horizontal image borders, optical flow profiles show a strong inflection near borders and converge toward a mean value.

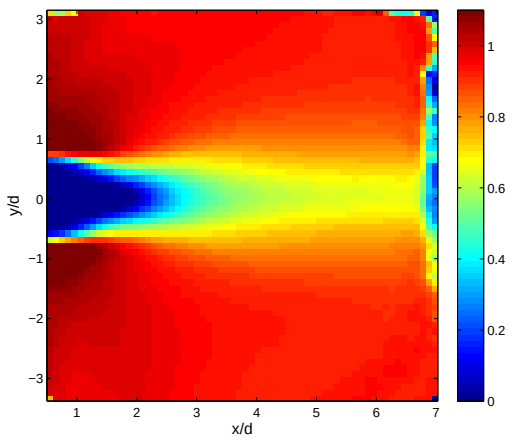
Regarding second component $\bar{v}_2$ (Figure 9.10b), all estimates are rather in complete disagreement. It can be noted however that optical flow profiles for the shortest time-step sequence ($dt = 100\,\mu s$) are closer to the anemometer profile than other profiles. And conversely, both optic flow and cross correlations profiles at $dt = 300\,\mu s$ look totally unrelated to the anemometer profile. Several explanations can be suggested. First of all, it seems that using larger time-steps deteriorates the larger structures, resulting in these curious profiles at $dt = 300\,\mu s$. Then, the obvious underestimation of the amplitude at smaller time steps can be explained by the poor quality of computer vision estimates for very small displacements. Also, averaging a longer sequence would probably help the convergence towards the anemometer profile. Finally, it may be noticed that increasing parameter α smoothes the profiles – especially with $dt = 100\,\mu s$, extremely small displacements – but does not affect their general shape.

Reynolds tensions (Eq. 9.2) profiles are given Figure 9.11. Consequences of the periodic bases can be seen again on $\langle v_1' v_1' \rangle$ profiles at the borders (Figure 9.11a), especially with large dt where the difference between the apparent velocities is at its maximum. The underestimation of v_2 component by both computer vision methods is confirmed again Figure 9.11b.

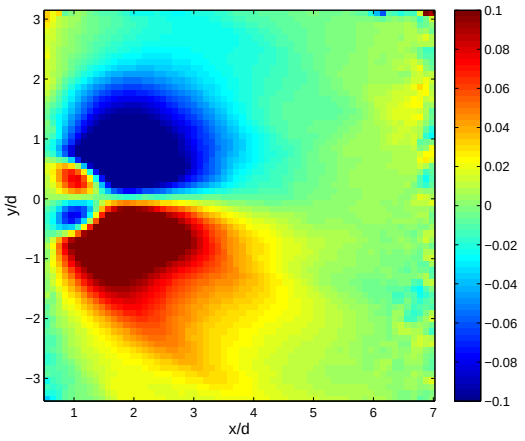
9.2.2 Time Signals and Spectra

Time signals extracted from 500 successive estimates, representing half a second, are displayed Figure 9.12. They were obtained from the third image sequence ($dt = 300\,\mu s$). The point where signals are extracted is located at the horizontal center of the image ($x = 800\,mm$) and as close as possible to the inflection point of the mean profiles ($y = -1\,mm$). Let us recall that this sequence is more adapted to the cross-correlations approach than to the optical flow, due to its large displacements. The two graphs recall the high difference, in terms of magnitude, between the longitudinal and transversal components – the later being 10 to 100 times smaller than the former. As a consequence, although cross-correlations and optical flow signals look rather similar for both components, the mean relative difference between the two estimates is much higher for the transversal component (154 %) than for the longitudinal one (2.6%).

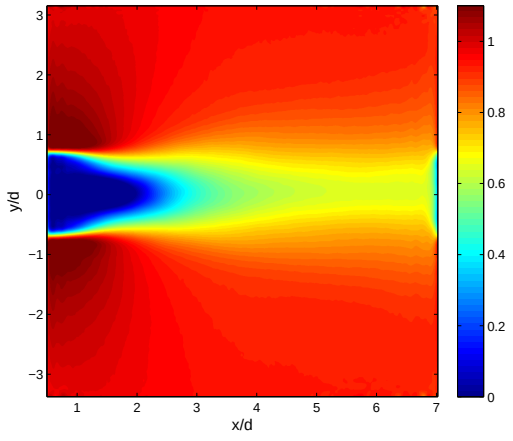
Figure 9.13 presents power spectral density computed from full time signals (3072 instants, around 3 seconds) extracted at $(x, y) = (800, -1)mm$, for the optical flow estimates (all 3 sequences) and the cross-correlations estimates (sequence with $dt = 300\,\mu s$). These PSDs were again obtained using Welch's periodograms, with 768 point windows. It can be noted that, despite the differences observed in time signals Figure 9.12, the PSDs of both optical flow and cross-correlations for sequence with $dt = 300\,\mu s$ coincide perfectly. The frequency peak around 13 Hz corresponds to the Kelvin-Helmholtz instability.



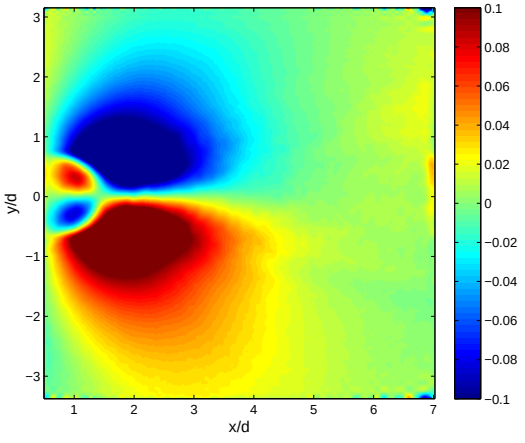
(a) cross-correlations – $\bar{v}_1/v^\infty$



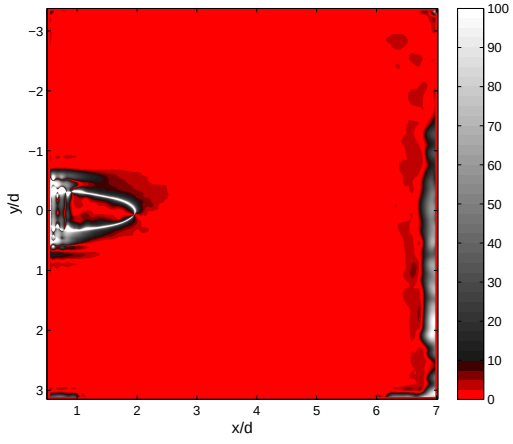
(b) cross-correlations – $\bar{v}_2/v^\infty$



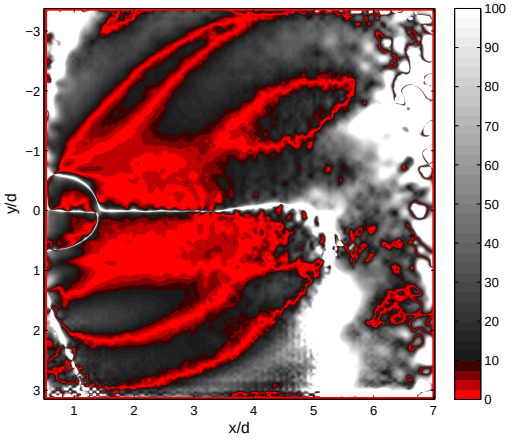
(c) optical flow – $\bar{v}_1/v^\infty$



(d) optical flow – $\bar{v}_2/v^\infty$

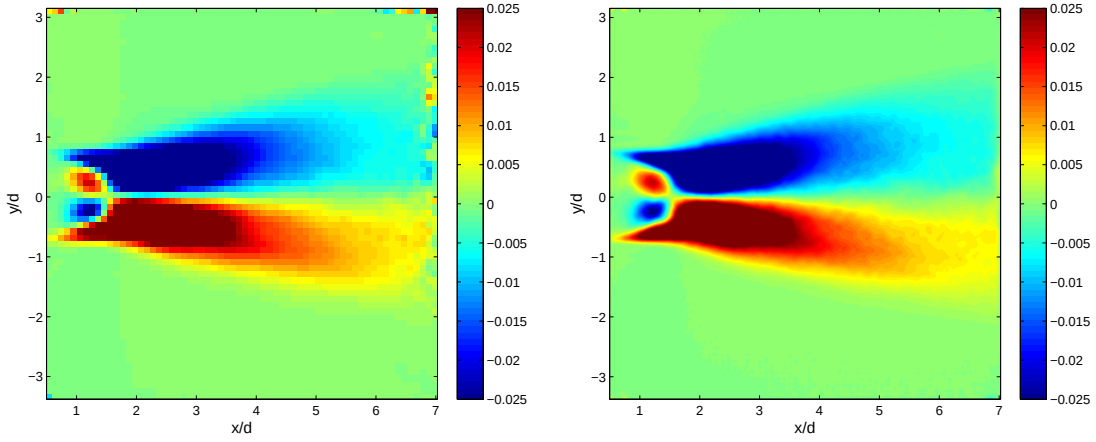


(e) differences – $\bar{v}_1/v^\infty$



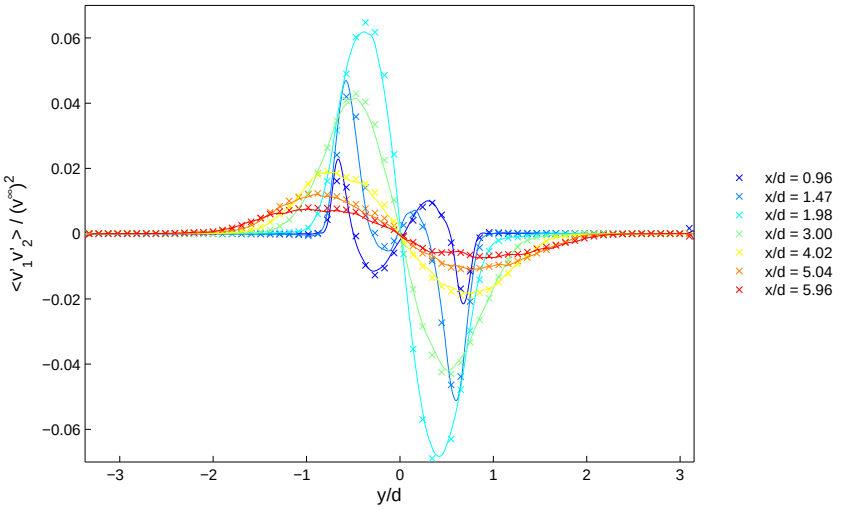
(f) differences – $\bar{v}_2/v^\infty$

Figure 9.3: Comparison of mean flow fields, (a) to (b). Absolute relative difference between both results with respect to optic flow values, in %, after interpolation of cross-correlations results. *Colored areas* are below 10% difference. Errors in cross-correlations estimates due to the halo-shaped artifacts are underlined in (f); it corresponds to the white crescent on the left and light gray areas for $x \in [1.5; 5]$.



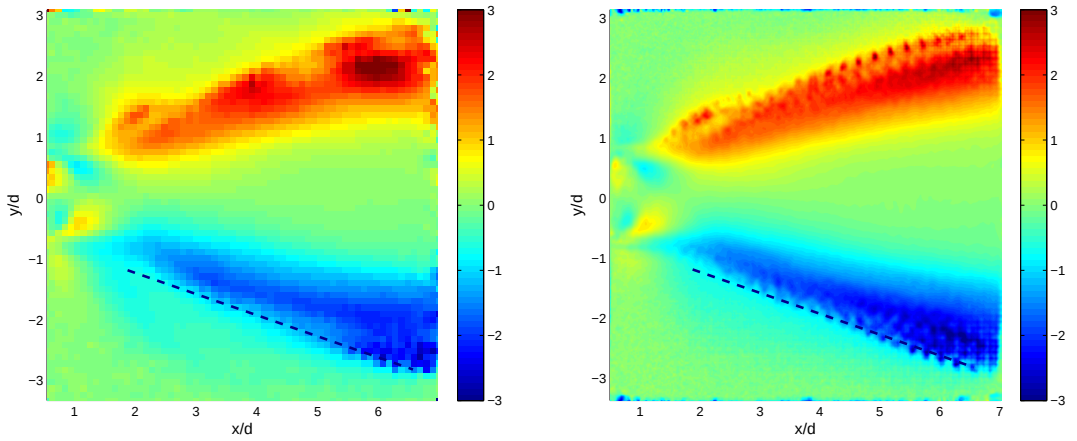
(a) cross-correlations – $\langle v_1' v_2' \rangle / (v^\infty)^2$

(b) optical flow – $\langle v_1' v_2' \rangle / (v^\infty)^2$



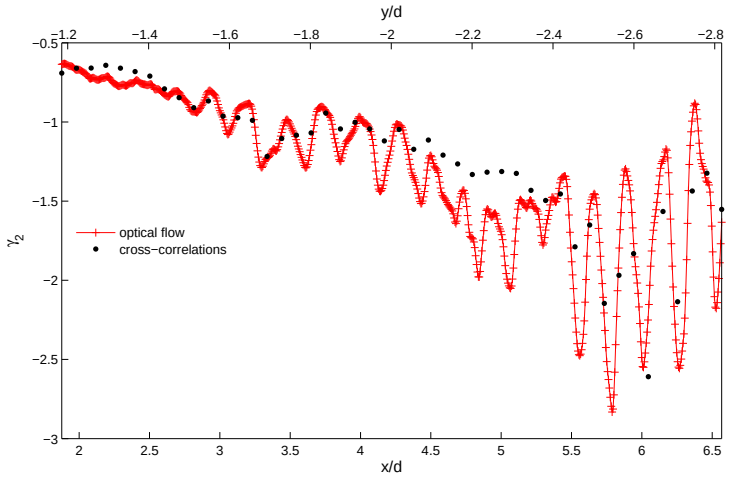
(c) $\langle v_1' v_2' \rangle / (v^\infty)^2$ profiles

Figure 9.4: Comparisons of mean cross-fluctuations $\langle v_1' v_2' \rangle / (v^\infty)^2$ computed from cross-correlations (a) and optical flow (b) estimates. Superposition of cross-correlations (*crosses*) and optical flow (*continuous lines*) profiles in (c) confirm the consistency of both estimates.



(a) cross-correlations – γ_2

(b) optical flow – γ_2



(c) γ_2 profiles

Figure 9.5: Comparisons of v_2 skewness, γ_2, computed from cross-correlations (a) and optical flow (b) estimates. Profiles in (c) were extracted along the black dashed lines.

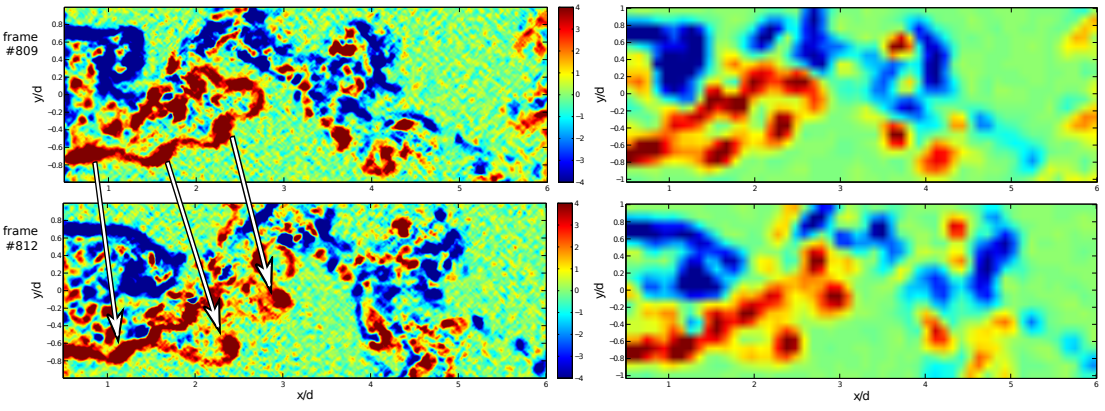
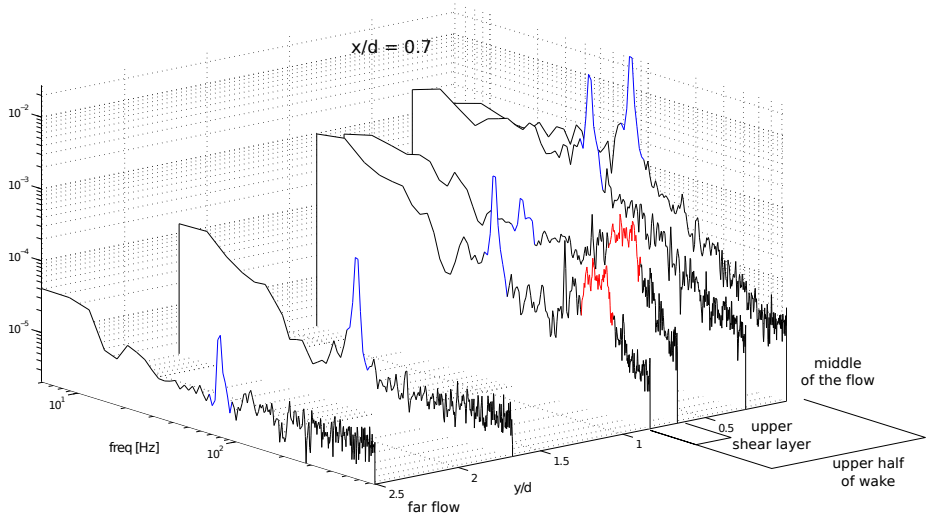
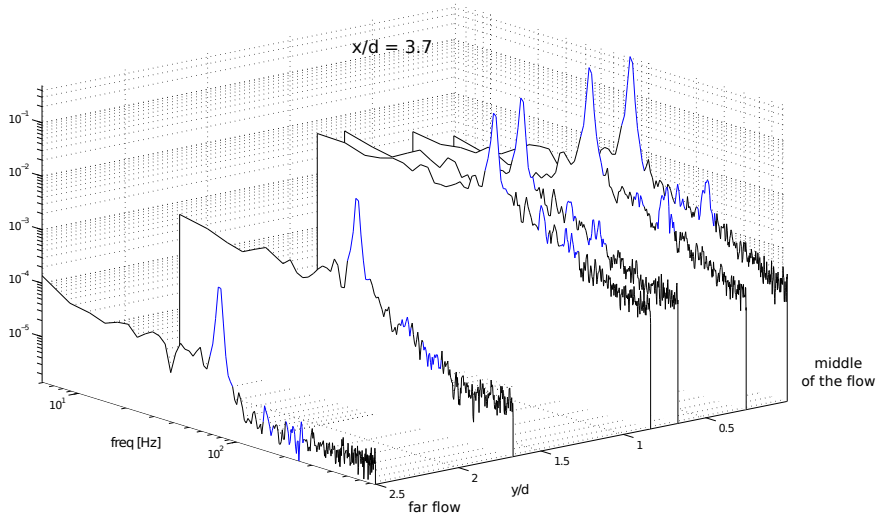


Figure 9.6: Development, in the shear layer, and advection of Kelvin-Helmholtz instabilities. *Left* vorticity maps were obtained from optical-flow estimates; *right* ones from cross-correlations.



(a) recirculation



(b) near wake

Figure 9.7: Power density spectra across the y direction for the crossflow velocity v_2, in the recirculation region (a) and near wake (b). Frequency peak of the von Kármán instability is in *blue* (as well as its first and second octaves), at 77.6 Hz. Higher frequency peaks corresponding to the Kelvin-Helmholtz instability can be seen in the recirculation, within the shear layer only (in *red*), around 300 Hz to 400 Hz.

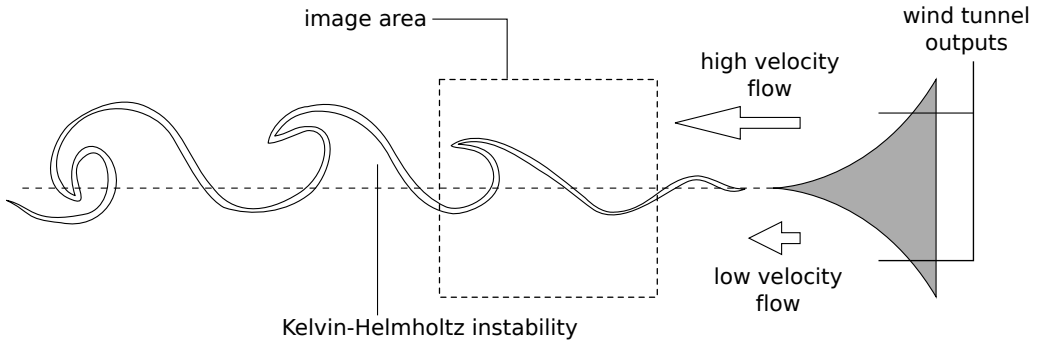
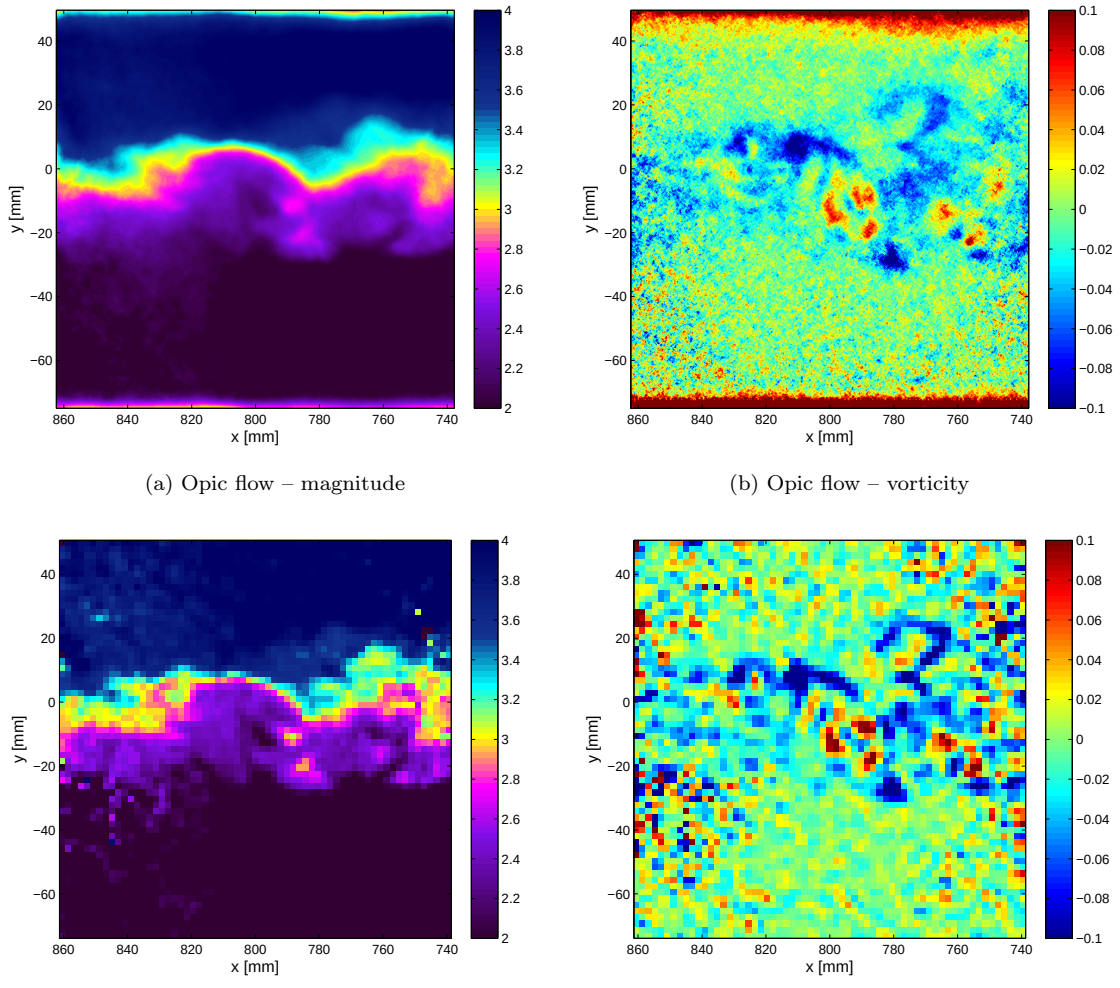


Figure 9.8: Configuration of the mixing layer experiment (not on scale). Flows speeds are 4 m/s and 2 m/s. Image center is located at 800 mm downstream of the trailing edge; each 1024×1024 px image covers 124×124 mm. Image pairs at acquired at 1000 Hz and time steps between two images of a given pair are 100 µs, 200 µs and 300 µs. Resulting maximum apparent horizontal displacements roughly cover 4 px to 10 px, while mean vertical displacements are almost always below 1 px.



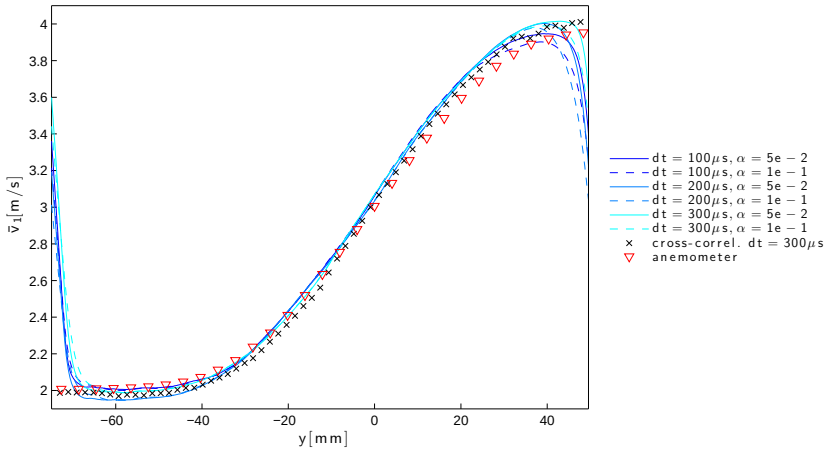
(a) Opic flow – magnitude

(b) Opic flow – vorticity

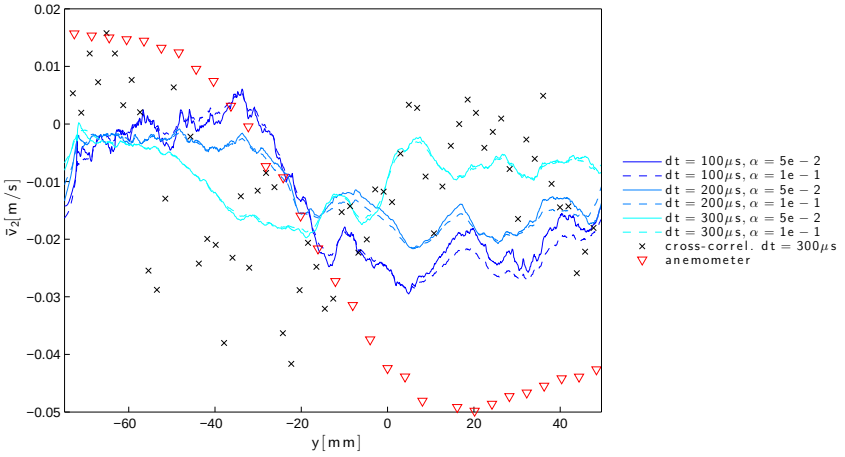
(c) Cross-correl. – magnitude

(d) Cross-correl. – vorticity

Figure 9.9: Norm and vorticity of motion estimates from image pair #6 of sequence with $dt = 200$ µs. Fields look rather similar, yet it can be noticed that optical flow estimate (*upper row*) shows an unexpected behavior near top and bottom borders of the image domain.

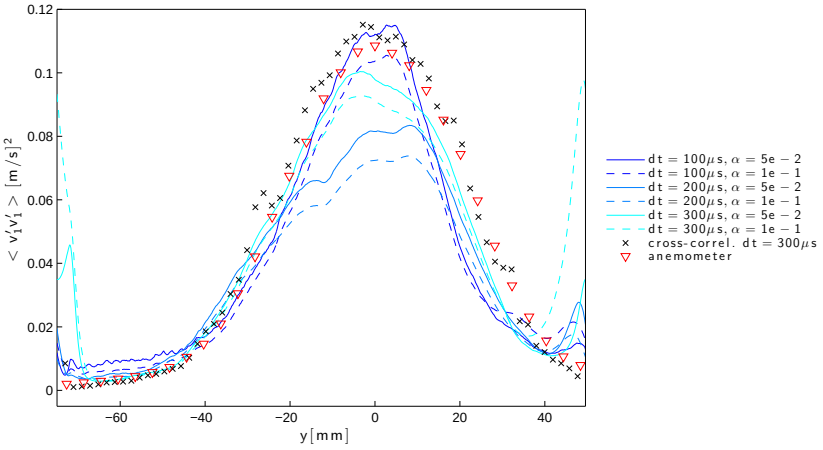


(a) $\bar{v}_1$ profiles

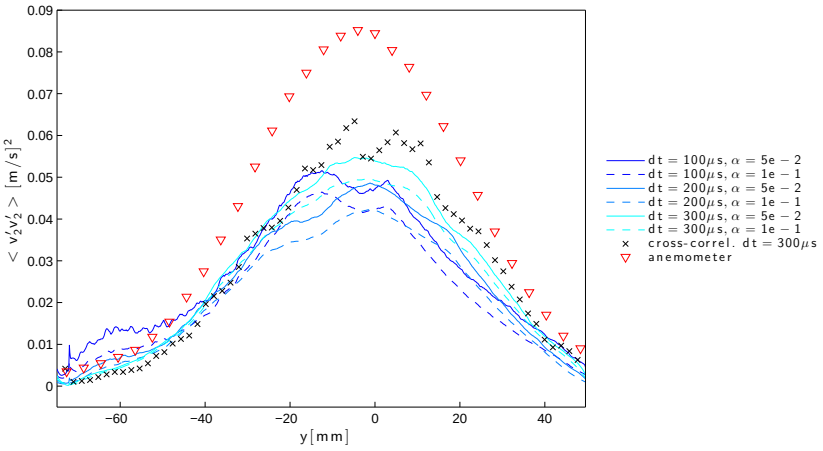


(b) $\bar{v}_2$ profiles

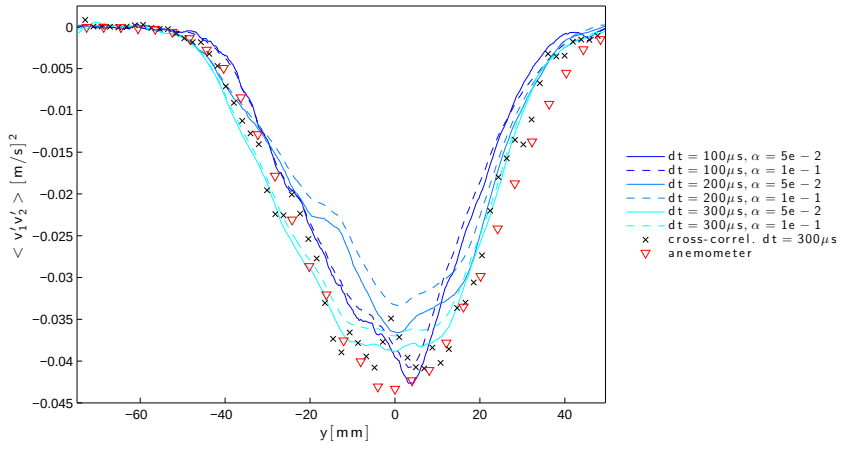
Figure 9.10: Comparison of mean flow profiles at $x = 840\,\text{mm}$ from the trailing edge, obtained from proposed wavelet-based optical flow (*colored lines*), cross-correlations (*black crosses*) or hot-wire anemometry (*red triangles*). *Dashed lines* profiles were obtained with a stronger α parameter. Optical flow profiles are periodic, due to the periodization of the wavelet basis.



(a) $\langle v_1' v_1' \rangle$ profiles

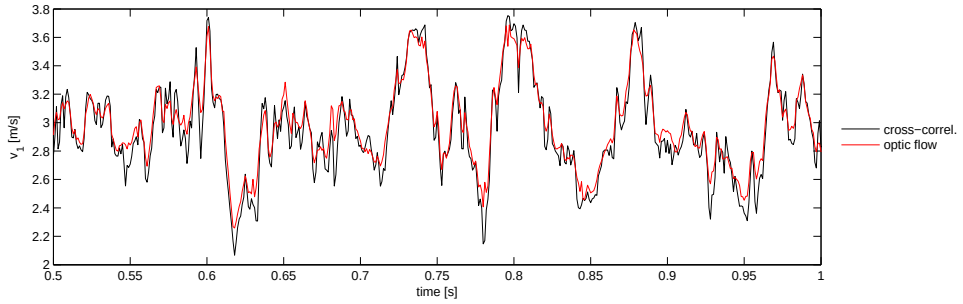


(b) $\langle v_2' v_2' \rangle$ profiles

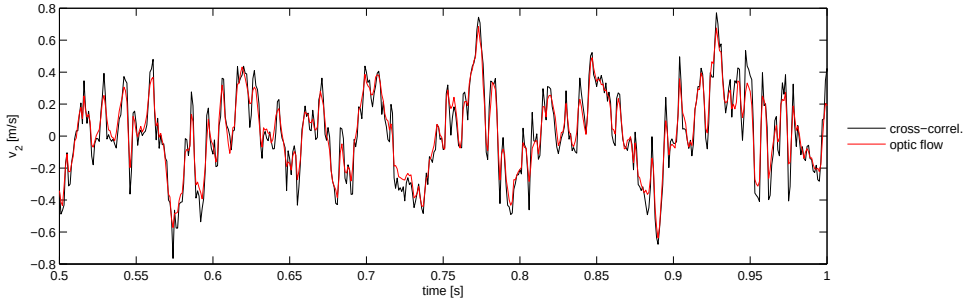


(c) $\langle v_1' v_2' \rangle$ profiles

Figure 9.11: Comparison of mean Reynolds tensions profiles at $x = 840\,\text{mm}$ from the trailing edge, obtained from proposed wavelet-based optical flow (*colored lines*), cross-correlations (*black crosses*) or hot-wire anemometry (*red triangles*).

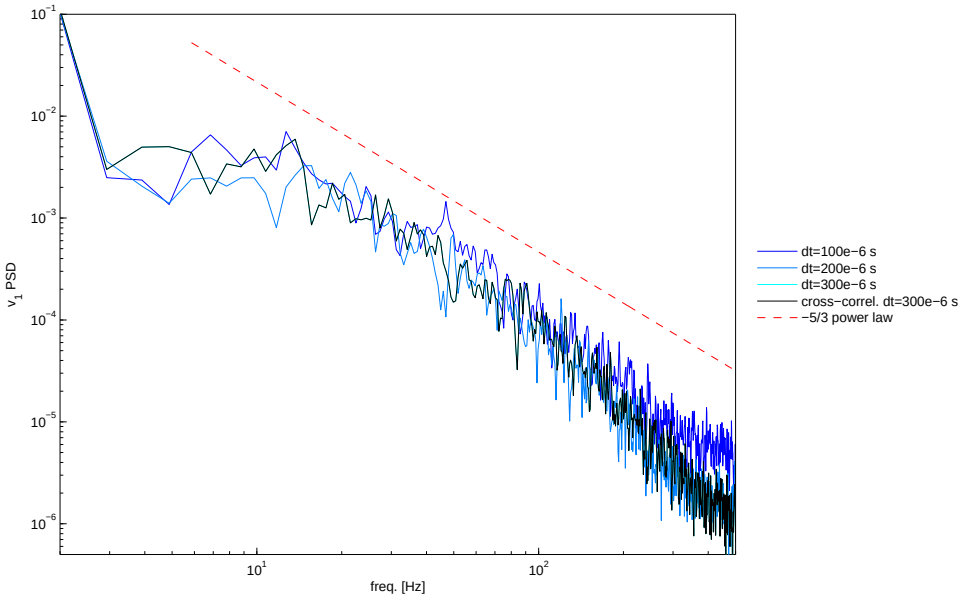


(a) v_1 temporal evolution

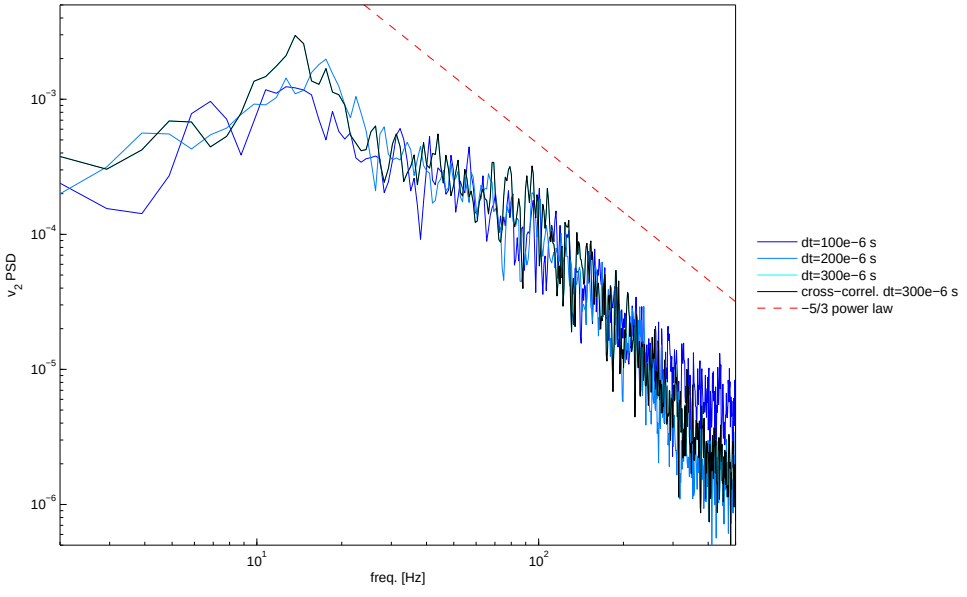


(b) v_2 temporal evolution

Figure 9.12: Comparison of time signals at $(x, y) = (800, -1)$mm given by cross-correlations (*black*) and optic flow (*red*). Displayed signal (half a second) represents values extracted from 500 image pairs. Mean relative difference between the two estimates, for the whole sequence (3072 frames), is 2.6% for v_1 and 154% for v_2, which illustrates again the difficulty to obtain reliable estimations for very small displacements.



(a) v_1 PSD



(b) v_2 PSD

Figure 9.13: Comparison of power spectral density of time signals extracted at $(x, y) = (800, -1)$mm. *Blue* spectra correspond to optical flow estimates, *black* spectra are for the cross-correlations. Spectra for both estimates of the third sequence (*lagoon blue* and *black*) coincide perfectly. Frequency peak at 13 Hz corresponds to the Kelvin-Helmholtz instability.

Conclusions

Across this document, I described the design and some elements of the practical implementation of a 2D wavelet-based optical flow estimator. While being explicitly directed at fluid flow processing, it is not absolutely restricted to this very specific family of motions. Presented works cover a basic estimator using regular wavelet bases, the addition of more specific divergence-free bases, as well as several regularization schemes that rely on wavelet properties.

The various versions of the estimator showed satisfying results on the synthetic datasets used for validation and evaluation, both in terms of behavior and with respect to other state-of-the-art methods – principally on particle images. The chosen quality criterion (root mean square norm of the difference between estimated and ground truth velocity vectors) however must not be considered as absolute, as many other criteria could have been employed (e.g. error on the reconstructed vorticity, power spectrum behavior, etc.). To me, these experiences therefore do not reflect much else than ensuring that the concept works and the regularizers act the way we expected, as well as giving some ideas on the influence of the various parameters and the operational domain – especially in terms of apparent displacements magnitude.

The second set of experiments, featuring the real datasets of the cylinder wake and mixing layer, is in my opinion much more meaningful. Because these experiments rely on actual, realistic, large images; because they illustrate well both the advantages and weaknesses of our method. And since the quality criteria – statistical quantities – seem more convincing to me than any "instantaneous" error measurement, in the first place. The wake experiment showed how finer structures, missed by cross-correlations, can be recovered using the proposed method. An animated sequence is sadly impossible to reproduce here, yet it would show beautiful dynamics with interactions between all scales, and a remarkable temporal continuity which tends to furthermore legitimate these optical flow results. On the contrary, the mixing layer experiment exhibited some limits of our wavelet-based approach. The first one being the periodic boundary conditions, employed for the sake of simplicity, which may corrupt the estimate near border regardless of local image data. The second one is shared by cross-correlations approach as well, it concerns the poor quality of second component estimates. This may be linked to the huge difference, in magnitude, between the first and the second apparent motion components, but more investigations are required.

On a more technical note, this work showed that a multiscale wavelet representation of the motion field, associated to the non-linear DFD, enables to recover apparent motions, up to a certain limit between 5 px to 10 px, without resorting to the traditional incremental scheme. Although the cross-correlations are able to handle higher magnitudes, it should be kept in mind that excessively large displacements may result in inaccurate estimations of larger, faster structures. Therefore our wavelet-based approach should probably give very satisfactory results, if the set-up of the experiment can be arranged to ensure the apparent displacements stay below this upper bound. The boundary condition issue however has to be fixed. There is also much to be done to lower the computation time for large images, notably by taking advantages of GPGPU to speed up the convolutions of the wavelet transforms. The divergence-free bases may seem somewhat anecdotal within this 2D context, as most of apparent 2D motions, actually 3D, are not divergence-free. It could however constitute a very powerful tool for 3D optical flow estimation, which is currently under development, if the computational cost of 3D wavelet transforms remain acceptable.

As implicitly suggested above, discussions and investigations on the quality criteria for reconstructed motion fields are to be considered, especially since these results are likely to be used, theoretically, for tasks such as fundamental turbulence studies, or embed into data assimilation processes in the context of meteorological or oceanographical simulations. This cannot be achieved

without a close cooperation with end users, fluid specialists, experimenters and technicians.

At the beginning of this thesis, I was a complete stranger to the field of computer vision, and had barely heard of wavelets and their applications. Although I was already interested in fluid dynamics, with its wide physics and beautiful instabilities, my main concern were images. I have always felt attracted by this medium, whether from a rather iconographic point-of-view, or simply as a raw material. My discovery of Milton van Dyke's collection of photography in its 1982 *An album of fluid motion* [43] made me realize how visualization techniques, and therefore images, are deeply connected to the study and comprehension of fluid dynamics. Not only such visualizations happen to give rise to actually beautiful images, but these images also constitute a powerful tool to describe and analyze the ongoing phenomenons. *Un dessin vaut mieux qu'un long discours*; a picture speaks a thousand words. I was then shown how computer vision methods are able to extract some motion information from such fluid flow image sequences, as well as the inherent difficulties in the process. And during these three years of developments and tests of my own wavelet-based motion estimator, I finally found myself creating pictures out of the extracted motions to help me analyze what had been recovered. From pictures, to motion, to pictures.

Bibliography

[1] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen. *LAPACK Users' Guide*. Society for Industrial and Applied Mathematics, Philadelphia, PA, third edition, 1999.

[2] D. Auroux and J. Fehrenbach. Identification of velocity fields for geophysical fluids from a sequence of images. *Experiments in Fluids*, 50(2):313–328, 2011.

[3] C. Avenel, E. Mémin, and P. Pérez. Tracking closed curves with non-linear stochastic filters. In *Conf. on Scale Space and Variational Methods (SSVM'09)*, Voss, Norway, June 2009.

[4] F. Becker, B. Wieneke, S. Petra, A. Schroder, and C. Schnorr. Variational adaptive correlation method for flow estimation. *Image Processing, IEEE Transactions on*, 21(6):3053 –3065, june 2012.

[5] C.P. Bernard. Discrete wavelet analysis for fast optic flow computation. *Applied and Computational Harmonic Analysis*, 11(1):32–63, 2001.

[6] D. Bernard. Turbulence for (and by) amateurs. *Arxiv preprint cond-mat/0007106*, 2000.

[7] G. Beylkin. On the representation of operators in bases of compactly supported wavelets. *SIAM Journal on Numerical Analysis*, 29(6):1716–1740, 1992.

[8] W. Cai and J. Wang. Adaptive multiresolution collocation methods for initial boundary value problems of nonlinear pdes. *SIAM Journal on Numerical Analysis*, pages 937–970, 1996.

[9] Patrick Chassaing. *Turbulence en Mécanique des Fluides: Analyse du Phénomène en Vue de sa Modélisation à l'Usage de l'Ingénieur (Politech)*. Cépaduès Éditions, INP Toulouse, France, 2000.

[10] L.F. Chen, H.Y.M. Liao, and J.C. Lin. Wavelet-based optical flow estimation. *Circuits and Systems for Video Technology, IEEE Transactions on*, 12(1):1–12, 2002.

[11] T. Corpetti, E. Mémin, and P. Pérez. Dense estimation of fluid flows. *Pattern Anal Mach Intel*, 24(3):365–380, 2002.

[12] A. Cuzol and E. Memin. Vortex and source particles for fluid motion estimation. In *Proc. 5th Int. Conf. on Scale-Space and PDE methods in Computer Vision (Scale-Space'05)*, Hofgeismar, Germany, 2005.

[13] E. Deriaz and V. Perrier. Direct numerical simulation of turbulence using divergence-free wavelet. *SIAM Multi. Mod. and Simul.*, 7(3):1101–1129, 2008.

[14] S. Dong, GE Karniadakis, A. Ekmekci, and D. Rockwell. A combined direct numerical simulation-particle image velocimetry study of the turbulent near wake. *Journal of Fluid Mechanics*, 569:185–208, 2006.

[15] M. Farge and K. Schneider. Coherent vortex simulation (cvs), a semi-deterministic turbulence model using wavelets. *Flow, Turbulence and Combustion*, 66(4):393–426, 2001.

[16] J.M. Fitzpatrick. A method for calculating velocity in time dependent images based on the continuity equation. In *Proc. Conf. Comp. Vision Pattern Rec.*, pages 78–81, San Francisco, USA, 1985.

[17] D.A. Forsyth and J. Ponce. *Computer vision: a modern approach*. Prentice Hall Professional Technical Reference, 2002.

[18] P. Héas, E. Mémin, D. Heitz, and P.D. Mininni. Power laws and inverse motion modelling: application to turbulence measurements from satellite images. *Tellus A*, 64(10962), 2012.

[19] D. Heitz, J. Carlier, and G. Arroyo. Final report on the evaluation of the tasks of the workpackage 2, FLUID project deliverable 5.4. Technical report, INRIA - Cemagref, 2007.

[20] B. Horn and B. Schunck. Determining optical flow. *Artificial Intelligence*, 17:185–203, 1981.

[21] S. Jaffard. Pointwise smoothness, two-microlocalization and wavelet coefficients. *Publ. Mat.*, pages 155–168, 1991.

[22] S. Kadri-Harouna. *Ondelettes pour la prise en compte de conditions aux limites en turbulence incompressible*. PhD thesis, Institut National Polytechnique de Grenoble, 2010. In French.

[23] Souleymane Kadri Harouna, Pierre Dérian, Patrick Héas, and Etienne Memin. Divergence-free Wavelets and High Order Regularization. Submited to IJCV, 2011.

[24] J.P. Kahane and P.G. Lemarié-Rieusset. *Fourier series and wavelets*, volume 3. Routledge, 1995.

[25] P.G. Lemarié-Rieusset. Analyses multirésolutions non orthogonales, commutation entre projecteurs et dérivation et ondelettes vecteurs à divergence nulle. *Revista Matematica Iberoamericana*, 8:221–237, 1992.

[26] CImg Library. `http://cimg.sourceforge.net/`.

[27] B. Lucas and T. Kanade. An iterative image registration technique with an application to stereovision. In *Int. Joint Conf. on Artificial Intel. (IJCAI)*, pages 674–679, 1981.

[28] S. Mallat. *A Wavelet Tour of Signal Processing: The Sparse Way*. Academic Press, 2008.

[29] C. Meneveau. Analysis of turbulence in the orthonormal wavelet representation. *Journal of Fluid Mechanics*, 232:469–520, 1991.

[30] P.D. Mininni, D.O. Gómez, and S.M. Mahajan. Direct simulations of helical hall-mhd turbulence and dynamo action. *The Astrophysical Journal*, 619:1019, 2005.

[31] A.S. Monin and A.M. Yaglom. *Statistical Fluid Mechanics: Mechanics of Turbulence*. Dover Pubns, 2007.

[32] J. Nocedal. Updating quasi-newton matrices with limited storage. *Mathematics of computation*, 35(151):773–782, 1980.

[33] L. Ong and J. Wallace. The velocity field of the turbulent very near wake of a circular cylinder. *Experiments in fluids*, 20(6):441–453, 1996.

[34] Nicolas Papadakis and Étienne Mémin. Variational optimal control technique for the tracking of deformable objects. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–7. IEEE, 2007.

[35] P. Parnaudeau, J. Carlier, D. Heitz, and E. Lamballais. Experimental and numerical studies of the flow over a circular cylinder at reynolds number 3900. *Physics of Fluids*, 20, 2008.

[36] R. K. Rew, G. P. Davis, S. Emmerson, and H. Davies. *NetCDF User's Guide for C, An Interface for Data Access, Version 3*, April 1997.

[37] O. Reynolds. An experimental investigation of the circumstances which determine whether the motion of water shall be direct or sinuous, and of the law of resistance in parallel channels. *Proceedings of the Royal Society of London*, 35(224-226):84–99, 1883.

[38] D. Suter. Motion estimation and vector splines. In *Proc. Conf. Comp. Vision Pattern Rec.*, pages 939–942, Seattle, USA, June 1994.

[39] J. K. Sveen. `http://folk.uio.no/jks/matpiv/`.

[40] R. Tatsambon Fomena and C. Collewet. Fluid flows control using visual servoing. In *18th IFAC World Congress*, pages 3142–3147, Milan, Italy, September 2011.

[41] P. Thévenaz, T. Blu, and M. Unser. Interpolation revisited. *IEEE TRANSACTIONS ON MEDICAL IMAGING*, 19(7):739, 2000.

[42] G. van der Graaf. `http://gpiv.sourceforge.net/`.

[43] Milton Van Dyke. *An album of fluid motion*. Parabolic Press Stanford, CA, 1982.

[44] Y. Wu, T. Kanade, C. Li, and J. Cohn. Image registration using wavelet-based motion model. *Int. J. Computer Vision*, 38(2):129–152, 2000.

[45] J. Yuan, C. Schnörr, and G. Steidl. Simultaneous higher-order optical flow estimation and decomposition. *SIAM Journal on Scientific Computing*, 29(6):2283–2304, 2007.

Figure Credits

Figure 4 is an extract from Reynolds' 1883 paper [37], now in the public domain. It was downloaded from wikipedia (en.wikipedia.org) through the "Osborne Reynolds" article.

Figure 1.1 was taken by NASA Landsat 7 satellite, 15 September 1999. This version (not copyrighted) was downloaded from wikipedia (en.wikidia.org) through the "von Kármán Vortex Street" article.

Figure 1.2 (same as Figure 1) has been computed from PIV pictures realized and kindly provided by Vincent Laval during its PhD thesis at the Institut de Mécanique des Fluides de Toulouse (IMFT), Toulouse, France, 2011.

Figure 1.3 is a SST product created and given by Cersat (IFREMER).

Figures 1.4a and 1.4b were taken at the Laboratorio de Mecánica de Fluidos, Facultad de Ingenieriá UBA, Buenos Aires, Argentina, in 2010 by Ines Auliel and Juan Martin Cabaleiro, respectively, under the supervision of Guillermo Artana.

Every other figure, scheme or graph was created by myself with the help of `Gimp`, `Gnuplot`, `Inkscape` and `Matlab`. Cover illustration is a close-up photo of ink floating on milk.

Part V

Appendices

Appendix A

Divergence-Free Bases: Practical Implementation

Consider a scalar field $z \in H^1(\mathbb{R}^2)$ and $\boldsymbol{v} = \mathbf{curl}(z) = (\partial_{x_1} z, -\partial_{x_2} z)^{\boldsymbol{T}} = (v_1, v_2)^{\boldsymbol{T}} \in \mathcal{H}^{\mathrm{div}}(\mathbb{R}^2)$ the divergence-free vector field uniquely defined by the **curl** of z. In Section 4.1, we derived (B)MRAs of these two function spaces so that motion $\boldsymbol{v}$ and its associated stream function z write:

$$\begin{aligned} z(\boldsymbol{x}) &= \sum_{\boldsymbol{j},\boldsymbol{k}} d^{\mathrm{div}}_{\boldsymbol{j},\boldsymbol{k}} \psi^1_{j_1,k_1}(x_1)\psi^1_{j_2,k_2}(x_2)\,; \\ v_1(\boldsymbol{x}) &= \sum_{\boldsymbol{j},\boldsymbol{k}} d^1_{\boldsymbol{j},\boldsymbol{k}} \psi^1_{j_1,k_1}(x_1)\psi^0_{j_2,k_2}(x_2)\,; \\ v_2(\boldsymbol{x}) &= \sum_{\boldsymbol{j},\boldsymbol{k}} d^2_{\boldsymbol{j},\boldsymbol{k}} \psi^0_{j_1,k_1}(x_1)\psi^1_{j_2,k_2}(x_2)\,; \end{aligned} \tag{A.1}$$

with $\boldsymbol{j}, \boldsymbol{k} \in \mathbb{Z}^2$. We derived formulas to compute $d^{\mathrm{div}}_{\boldsymbol{j},\boldsymbol{k}}$ coefficients from $d^1_{\boldsymbol{j},\boldsymbol{k}}$ and $d^2_{\boldsymbol{j},\boldsymbol{k}}$, and reciprocally – Eq. (4.22), (4.23), (4.24). However, the practical implementation of such bases reveals numerous subtle aspects which deserve to be clarified, such as the computation of the various filters involved in forward and inverse transforms. Above all, the use of periodic bases (in order to deal with finite signals) leads to non-trivial relations between the three sets of coefficients involved, that must be taken into account in order to design exact decomposition/reconstruction algorithms.

A.1 Filters

Two sets of scaling and wavelet functions are necessary to form divergence-free bases:
- $(\varphi^1, \tilde{\varphi}^1)$ and $(\psi^1, \tilde{\psi}^1)$ form a basis for the stream functions space $H^1(\mathbb{R}^2)$;
- $(\varphi^0, \tilde{\varphi}^0)$ and $(\psi^0, \tilde{\psi}^0)$ that, along with previous functions, form a basis for $\mathcal{H}^{\mathrm{div}}(\mathbb{R}^2)$.

These functions are related by derivation and integration relations (4.3). In practice, their corresponding filters are required for fast transforms. There are two ways to construct the set of all low-pass filters, either starting from the biorthogonal pair $(h_0, \tilde{h}_0)$ or from $(h_1, \tilde{h}_1)$, as illustrated in Figure A.1. Derivation and integration functions slightly differ between cases (a) and (b). Corresponding pseudocodes are given Section A.4, Figures A.4 and A.5. High-pass filters $(g_i, \tilde{g}_i)$ are then computed from $(h_i, \tilde{h}_i)$ for $i = 0, 1$ by conjugate mirror formulas (3.17). Finally, we have four pairs of high- and low-pass filters in total: (h_0, g_0), $(\tilde{h}_0, \tilde{g}_0)$, (h_1, g_1), $(\tilde{h}_1, \tilde{g}_1)$. Their various uses are:
- $\{d^{div}_{\boldsymbol{j},\boldsymbol{k}}\}_{\boldsymbol{j},\boldsymbol{k}} \mapsto z$: primal filters (h_1, g_1).
- $z \mapsto \{d^{div}_{\boldsymbol{j},\boldsymbol{k}}\}_{\boldsymbol{j},\boldsymbol{k}}$: dual filters $(\tilde{h}_1, \tilde{g}_1)$.
- $\{d^1_{\boldsymbol{j},\boldsymbol{k}}\}_{\boldsymbol{j},\boldsymbol{k}} \mapsto v_1$: primal filters (h_1, g_1) for dimension x_1, (h_0, g_0) for x_2.
- $v_1 \mapsto \{d^1_{\boldsymbol{j},\boldsymbol{k}}\}_{\boldsymbol{j},\boldsymbol{k}}$: dual filters $(\tilde{h}_1, \tilde{g}_1)$ for dimension x_1, $(\tilde{h}_0, \tilde{g}_0)$ for x_2.
- $\{d^2_{\boldsymbol{j},\boldsymbol{k}}\}_{\boldsymbol{j},\boldsymbol{k}} \mapsto v_2$: primal filters (h_0, g_0) for dimension x_1, (h_1, g_1) for x_2.
- $v_2 \mapsto \{d^2_{\boldsymbol{j},\boldsymbol{k}}\}_{\boldsymbol{j},\boldsymbol{k}}$: dual filters $(\tilde{h}_0, \tilde{g}_0)$ for dimension x_1, $(\tilde{h}_1, \tilde{g}_1)$ for x_2.

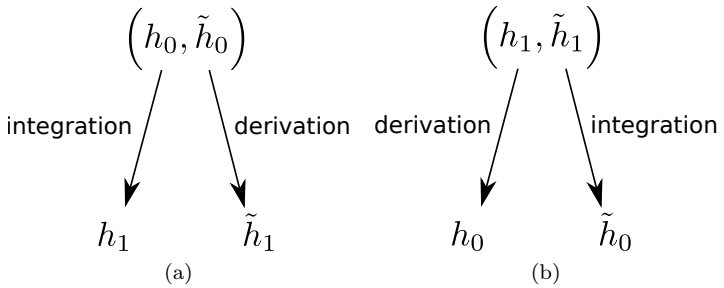


Figure A.1: Relations between filters for divergence-free bases.

– DFD gradient (see Section A.3.3): primal filters (h_0, g_0) and (h_1, g_1).
– connection coefficients for high-order regularization on $\{d^{div}_{\boldsymbol{j},\boldsymbol{k}}\}_{\boldsymbol{j},\boldsymbol{k}}$ (Section 6.2.2): primal filters (h_1, g_1).

Regarding the optical flow problem, only primal filters (h_0, g_0) and (h_1, g_1) are used (inverse transforms, gradient computation and connection coefficients). It should be noted that in (3.17), primal high-pass filter g_i is obtained from dual low-pass filter $\tilde{h}_i$.

A.2 Finite Signals & Scaling Functions Derivatives

Before going any further, we must remind that we work with finite signals with a fine (pixel) scale $F < 0$: $z \in H^1([0,1]^2)$ and $\boldsymbol{v} \in \mathcal{H}^{\mathrm{div}}([0,1]^2)$. Above expressions (A.1) are no longer correct, since there are a finite range of scales, form the finest $F \leq 0$ to a given coarsest $F \leq C \leq 0$. Hence the following expression for stream function $z \in V^1_F \otimes V^1_F$:

$$\begin{aligned} z(\boldsymbol{x}) = & \sum_{\boldsymbol{k}} a^{\mathrm{div}}_{(C,C),\boldsymbol{k}}\, \varphi^1_{C,k_1}(x_1)\varphi^1_{C,k_2}(x_2) \\ & + \sum_{j,\boldsymbol{k}} b^{\mathrm{div}}_{(C,j),\boldsymbol{k}}\, \varphi^1_{C,k_1}(x_1)\psi^1_{j,k}(x_2) + c^{\mathrm{div}}_{(j,C),\boldsymbol{k}}\, \psi^1_{j,k}(x_1)\varphi^1_{C,k_2}(x_2) \\ & + \sum_{\boldsymbol{j},\boldsymbol{k}} d^{\mathrm{div}}_{\boldsymbol{j},\boldsymbol{k}}\, \psi^1_{j_1,k_1}(x_1)\psi^1_{j_2,k_2}(x_2)\,; \end{aligned} \tag{A.2}$$

where $\boldsymbol{j} \in [F+1; C]^2$, $j \in [F+1; C]$, $\boldsymbol{k} \in [0; 2^{-j_1} - 1] \times [0; 2^{-j_2} - 1]$, and the coefficients are given by:

$$\begin{aligned} a^{\mathrm{div}}_{(C,C),\boldsymbol{k}} &= \left\langle z\,;\, \tilde{\varphi}^1_{C,k_1} \otimes \tilde{\varphi}^1_{C,k_2} \right\rangle_{\mathbf{L}^2}\,; \\ b^{\mathrm{div}}_{(C,j),\boldsymbol{k}} &= \left\langle z\,;\, \tilde{\varphi}^1_{C,k_1} \otimes \tilde{\psi}^1_{j_2,k_2} \right\rangle_{\mathbf{L}^2}\,; \\ c^{\mathrm{div}}_{(j,C),\boldsymbol{k}} &= \left\langle z\,;\, \tilde{\psi}^1_{j_1,k_1} \otimes \tilde{\varphi}^1_{C,k_2} \right\rangle_{\mathbf{L}^2}\,; \\ d^{\mathrm{div}}_{\boldsymbol{j},\boldsymbol{k}} &= \left\langle z\,;\, \tilde{\psi}^1_{j_1,k_1} \otimes \tilde{\psi}^1_{j_2,k_2} \right\rangle_{\mathbf{L}^2}\,. \end{aligned} \tag{A.3}$$

Similarly, first motion component v_1 in $V^1_F \otimes V^0_F$ writes

$$\begin{aligned} v_1(\boldsymbol{x}) = & \sum_{\boldsymbol{k}} a^1_{(C,C),\boldsymbol{k}}\, \varphi^1_{C,k_1}(x_1)\varphi^0_{C,k_2}(x_2) \\ & + \sum_{j,\boldsymbol{k}} b^1_{(C,j),\boldsymbol{k}}\, \varphi^1_{C,k_1}(x_1)\psi^0_{j,k}(x_2) + c^1_{(j,C),\boldsymbol{k}}\, \psi^1_{j,k}(x_1)\varphi^0_{C,k_2}(x_2) \\ & + \sum_{\boldsymbol{j},\boldsymbol{k}} d^1_{\boldsymbol{j},\boldsymbol{k}}\, \psi^1_{j_1,k_1}(x_1)\psi^0_{j_2,k_2}(x_2)\,; \end{aligned} \tag{A.4}$$

whith the coefficients:

$$\begin{aligned} a^1_{(C,C),\boldsymbol{k}} &= \left\langle v_1 \,;\, \tilde{\varphi}^1_{C,k_1} \otimes \tilde{\varphi}^0_{C,k_2} \right\rangle_{\mathbf{L}^2} ; \\ b^1_{(C,j),\boldsymbol{k}} &= \left\langle v_1 \,;\, \tilde{\varphi}^1_{C,k_1} \otimes \tilde{\psi}^0_{j_2,k_2} \right\rangle_{\mathbf{L}^2} ; \\ c^1_{(j,C),\boldsymbol{k}} &= \left\langle v_1 \,;\, \tilde{\psi}^1_{j_1,k_1} \otimes \tilde{\varphi}^0_{C,k_2} \right\rangle_{\mathbf{L}^2} ; \\ d^1_{\boldsymbol{j},\boldsymbol{k}} &= \left\langle v_1 \,;\, \tilde{\psi}^1_{j_1,k_1} \otimes \tilde{\psi}^0_{j_2,k_2} \right\rangle_{\mathbf{L}^2} . \end{aligned} \tag{A.5}$$

Finally, second component $v_2 \in V_F^0 \otimes V_F^1$ is:

$$\begin{aligned} v_2(\boldsymbol{x}) = &\sum_{\boldsymbol{k}} a^2_{(C,C),\boldsymbol{k}}\, \varphi^1_{C,k_0}(x_1)\varphi^1_{C,k_2}(x_2) \\ &+ \sum_{j,\boldsymbol{k}} b^2_{(C,j),\boldsymbol{k}}\, \varphi^0_{C,k_1}(x_1)\psi^1_{j,k}(x_2) + c^2_{(j,C),\boldsymbol{k}}\, \psi^0_{j,k}(x_1)\varphi^1_{C,k_2}(x_2) \\ &+ \sum_{\boldsymbol{j},\boldsymbol{k}} d^2_{\boldsymbol{j},\boldsymbol{k}}\, \psi^0_{j_1,k_1}(x_1)\psi^1_{j_2,k_2}(x_2) \,; \end{aligned} \tag{A.6}$$

and the coefficients:

$$\begin{aligned} a^2_{(C,C),\boldsymbol{k}} &= \left\langle v_2 \,;\, \tilde{\varphi}^0_{C,k_1} \otimes \tilde{\varphi}^1_{C,k_2} \right\rangle_{\mathbf{L}^2} ; \\ b^2_{(C,j),\boldsymbol{k}} &= \left\langle v_2 \,;\, \tilde{\varphi}^0_{C,k_1} \otimes \tilde{\psi}^1_{j_2,k_2} \right\rangle_{\mathbf{L}^2} ; \\ c^2_{(j,C),\boldsymbol{k}} &= \left\langle v_2 \,;\, \tilde{\psi}^0_{j_1,k_1} \otimes \tilde{\varphi}^1_{C,k_2} \right\rangle_{\mathbf{L}^2} ; \\ d^2_{\boldsymbol{j},\boldsymbol{k}} &= \left\langle v_2 \,;\, \tilde{\psi}^0_{j_1,k_1} \otimes \tilde{\psi}^1_{j_2,k_2} \right\rangle_{\mathbf{L}^2} . \end{aligned} \tag{A.7}$$

The goal is therefore to find the relations between stream function z coefficients (A.3) and motion components v_1, v_2 coefficients (A.5) & (A.7), using $\boldsymbol{v} = \mathbf{curl}(z)$.

Applying **curl** operator $(\partial_{x_2}, -\partial_{x_1})^{\boldsymbol{T}}$ to (A.2) will bring up first-order derivatives of the various basis functions involved: $\frac{d}{dx}\varphi^1_{C,k}(x)$ and $\frac{d}{dx}\psi^1_{j,k}(x)$. From derivation/integration relations (4.3), we obtain:

$$\begin{aligned} &\frac{d}{dx}\varphi^1_{C,k}(x) = 2^{-C}\left[\varphi^0_{C,k}(x) - \varphi^0_{C,k}(x-1)\right] = 2^{-C}\left[\varphi^0_{C,k}(x) - \varphi^0_{C,k+1}(x)\right] ; \\ &\frac{d}{dx}\psi^1_{j,k}(x) = 2^{-j+2}\psi^0_{j,k} \,; \end{aligned} \tag{A.8}$$

again with $j \in [F+1; C]$ and $k \in [0; 2^{-j}-1]$. Wavelet functions derivation simply rises a 2^{-j+2} factor. The crucial point is related to the $k+1$ translation that appears in scale function derivative. Since our basis is separable, we may first focus on the 1D case. Let us consider the fine scale approximation of signal f:

$$f(x) = \sum_{k=0}^{2^{-F}-1} a_{F,k}\, \varphi^1_{F,k}(x) .$$

Computing its derivative and inserting expression (A.8) leads to:

$$\begin{aligned} \frac{df}{dx}(x) &= \sum_{k=0}^{2^{-F}-1} a_{F,k}\, \frac{d\varphi^1_{F,k}}{dx}(x) \\ &= \sum_{k=0}^{2^{-F}-1} 2^{-F} a_{F,k}\left[\varphi^0_{F,k}(x) - \varphi^0_{F,k+1}(x)\right] \\ &= 2^{-F}\left[a_{F,0}\,\varphi^0_{F,0}(x) - a_{F,2^{-F}-1}\,\varphi^0_{F,2^{-F}}(x)\right] + \sum_{k=1}^{2^{-F}-1} 2^{-F}\left[a_{F,k} - a_{F,k-1}\right]\varphi^0_{F,k} . \end{aligned}$$

In order to deal with finite signals, basis functions are periodized – Section 3.1.3: at scale F, functions are 2^{-F} periodic. Therefore, $\varphi^0_{F,2^{-F}}(x) = \varphi^0_{F,0}(x)$, and we obtain:

$$\frac{df}{dx}(x) = 2^{-F}\left[a_{F,0} - a_{F,2^{-F}-1}\right]\varphi^0_{F,0}(x) + \sum_{k=1}^{2^{-F}-1} 2^{-F}\left[a_{F,k} - a_{F,k-1}\right]\varphi^0_{F,k} \,;$$

Finally, we may write the relation between coefficients $a_{F,k}$ of f and $\tilde{a}_{F,k}$ of f' by identification:

$$\frac{df}{dx}(x) = \sum_{k=0}^{2^{-F}-1} \tilde{a}_{F,k}\,\varphi^0_{F,k}(x) \quad \Rightarrow \begin{cases} \tilde{a}_{F,0} = 2^{-F}\big(a_{F,0} - a_{F,2^{-F}-1}\big) \\ \tilde{a}_{F,k} = 2^{-F}\big(a_{F,k} - a_{F,k-1}\big)\,, \text{ for } 1 \le k \le 2^{-F}-1\,. \end{cases}$$

Three important points are deduced from this last result:

(i) *Approximation coefficients* $\{\tilde{a}_{F,k}\}_k$ of a function derivative are given by a *finite difference* of that function approximation coefficients $\{a_{F,k}\}_k$, with *periodic boundary conditions*.

(ii) Due to periodization, inverting the system (i.e. deducing $\{\tilde{a}_{F,k}\}_k$ from $\{a_{F,k}\}_k$) is generally impossible.

(iii) A special case appears with $F = 0 \Rightarrow k = 0$: $\tilde{a}_{0,0} = 2^{-F}(a_{0,0} - a_{0,0}) = 0$. Indeed, with periodization, the coarsest scaling function $\varphi_{0,0}$ of any wavelet basis is always the *unit-constant function* [28]; it is not surprising to find that its derivative is null. A direct consequence is that underdetermination (ii) disappears so that the system can be inverted.

Keeping these observations in mind, me may now go back to our motion field.

A.3 Application to z and $\boldsymbol{v}$

We will now design algorithms to compute $\boldsymbol{v}$ coefficients from z coefficients (the "reconstruction"), and reciprocally (the "decomposition"). Most of the time, only the reconstruction is used within the motion estimation context. The decomposition is necessary, for instance, in order to initialize the algorithm (the unknowns being stream function coefficients) with a given motion field.

Regarding the reconstruction, two different algorithms may be derived, using either *multiscale* (i.e. with both scaling and wavelet functions, when $F < C \le 0$) or monoscale (scaling functions only, $C = F$) bases. From above remarks (ii)-(iii) however, the only available option at the decomposition, in order to recover approximation coefficients, is to consider $C = 0$: a "full" multiscale decomposition. Therefore both multiscale algorithms (decomposition and reconstruction) will be derived setting $C = 0$.

A.3.1 Reconstruction from Approximation Coefficients Only

Here, we have set $C = F$ in (A.2). Let us consider first what happens with component v_1.

$$\begin{aligned} v_1(\boldsymbol{x}) = \frac{\partial z(\boldsymbol{x})}{\partial x_2} &= \sum_{\boldsymbol{k}} a^{\text{div}}_{(F,F),\boldsymbol{k}}\,\varphi^1_{F,k_1}(x_1)\big[\varphi^1_{F,k_2}\big]'(x_2) \\ &= \sum_{\boldsymbol{k}} 2^{-F} a^{\text{div}}_{(F,F),\boldsymbol{k}}\,\varphi^1_{F,k_1}(x_1)\big[\varphi^0_{F,k_2}(x_2) - \varphi^0_{F,k_2+1}(x_2)\big] \\ &= \sum_{k_1} 2^{-F}\big[a^{\text{div}}_{(F,F),(k_1,0)} - a^{\text{div}}_{(F,F),(k_1,2^{-F}-1)}\big]\varphi^1_{F,k_1}(x_1)\varphi^0_{F,0}(x_2) \\ &\quad + \sum_{k_1}\sum_{k_2=1}^{2^{-F}-1} 2^{-F}\big[a^{\text{div}}_{(F,F),\boldsymbol{k}} - a^{\text{div}}_{(F,F),(k_1,k_2-1)}\big]\varphi^1_{F,k_1}(x_1)\varphi^0_{F,k_2}(x_2)\,; \end{aligned} \tag{A.9}$$

with $\boldsymbol{k} \in [0; 2^{-F}-1]^2$ and $k_1 \in [0; 2^{-F}-1]$. Since v_1 also writes

$$v_1(\boldsymbol{x}) = \sum_{\boldsymbol{k}} a^1_{(F,F),\boldsymbol{k}}\,\varphi^1_{F,k_1}(x_1)\varphi^0_{F,k_2}(x_2)\,, \tag{A.10}$$

we find by identification, for $k_1 \in [0; 2^{-F}-1]$:

$$a^1_{(F,F),(k_1,k_2)} = \begin{cases} 2^{-F}\big[a^{\text{div}}_{(F,F),(k_1,0)} - a^{\text{div}}_{(F,F),(k_1,2^{-F}-1)}\big]\,, & \text{for } k_2 = 0\,; \\ 2^{-F}\big[a^{\text{div}}_{(F,F),(k_1,k_2)} - a^{\text{div}}_{(F,F),(k_1,k_2-1)}\big]\,, & \text{for } k_2 \in [1; 2^{-F}-1]\,. \end{cases} \tag{A.11}$$

Approximation coefficients $\{a^1_{(F,F),\boldsymbol{k}}\}_{\boldsymbol{k}}$ of v_1 are given by a *finite difference along the second dimension* of z coefficients $\{a^{\text{div}}_{(F,F),\boldsymbol{k}}\}_{\boldsymbol{k}}$, with periodic boundary conditions, and a normalization by factor 2^{-F}. Similarly, we find for v_2, for $k_2 \in [0; 2^{-F}-1]$:

$$a^2_{(F,F),(k_1,k_2)} = \begin{cases} -2^{-F}\left[a^{\text{div}}_{(F,F),(0,k_2)} - a^{\text{div}}_{(F,F),(2^{-F}-1,k_2)}\right], & \text{for } k_1 = 0\,; \\ -2^{-F}\left[a^{\text{div}}_{(F,F),(k_1,k_2)} - a^{\text{div}}_{(F,F),(k_1-1,k_2)}\right], & \text{for } k_1 \in [1; 2^{-F}-1]\,. \end{cases} \tag{A.12}$$

Here, the finite difference with periodization proceeds along *the first dimension*. Should z be known on a multiscale basis (with F<C, Eq. (A.2)), it suffices to perform the appropriate 2D inverse wavelet transform until fine approximation coefficients only are left, prior to applying relations A.11 and A.12. This whole reconstruction algorithm is summed up in Fig. A.6. Finally and as already mentioned in remark (ii), it is impossible to invert the system in order to find z coefficients from those of $\boldsymbol{v}$.

A.3.2 Reconstruction & Decomposition from Multiscale Coefficients

These algorithms allow a full decomposition/reconstruction of motion $\boldsymbol{v}$ and its associated stream function z. Here, we set $C = 0$, i.e. consider a full multiscale decomposition. From above remark (iii), every term involving $(\varphi^1_{0,0})' = (\varphi^1)'$ vanishes. Starting again with the first component, we start by expressing v_1 as $\partial_{x_2} z$:

$$\begin{aligned} v_1(\boldsymbol{x}) = \partial_{x_2} z = {} & a^{\text{div}}_{(0,0),(0,0)}\, \varphi^1(x_1)\left[\varphi^1\right]'(x_2) \\ & + \sum_{j,k} b^{\text{div}}_{(0,j),(0,k)}\, \varphi^1(x_1)\left[\psi^1_{j,k}\right]'(x_2) + c^{\text{div}}_{(j,0),(k,0)}\, \psi^1_{j,k}(x_1)\left[\varphi^1\right]'(x_2) \\ & + \sum_{\boldsymbol{j},\boldsymbol{k}} d^{\text{div}}_{\boldsymbol{j},\boldsymbol{k}}\, \psi^1_{j_1,k_1}(x_1)\left[\psi^1_{j_2,k_2}\right]'(x_2)\,; \end{aligned} \tag{A.13}$$

Moreover, from (A.4) with $C = 0$, $v_1 \in V^1_F \otimes\ V^0_F$ also writes

$$\begin{aligned} v_1(\boldsymbol{x}) = {} & a^1_{(0,0),(0,0)}\, \varphi^1(x_1)\varphi^0(x_2) \\ & + \sum_{j,k} b^1_{(0,j),(0,k)}\, \varphi^1(x_1)\psi^0_{j,k}(x_2) + c^1_{(j,0),(k,0)}\, \psi^1_{j,k}(x_1)\varphi^0(x_2) \\ & + \sum_{\boldsymbol{j},\boldsymbol{k}} d^1_{\boldsymbol{j},\boldsymbol{k}}\, \psi^1_{j_1,k_1}(x_1)\psi^0_{j_2,k_2}(x_2)\,; \end{aligned} \tag{A.14}$$

Inserting wavelet derivative expression (A.8) and canceling the appropriate terms finally leads to:

$$\begin{aligned} & a^1_{(0,0),(0,0)} = 0\,; \\ & b^1_{(0,j),(0,k)} = (2^{-j+2})\, b^{\text{div}}_{(0,j),(0,k)} && \text{for } j \in [F+1; 0],\ k \in [0, 2^{-j}-1]\,; \\ & c^1_{(j,0),(k,0)} = 0 && \text{for } j \in [F+1; 0],\ k \in [0, 2^{-j}-1]\,; \\ & d^1_{\boldsymbol{j},\boldsymbol{k}} = (2^{-j_2+2})\, d^{\text{div}}_{\boldsymbol{j},\boldsymbol{k}} && \text{for } \boldsymbol{j} \in [F+1; 0]^2,\ \boldsymbol{k} \in [0; 2^{-j_1}-1] \times [0; 2^{-j_2}-1]\,. \end{aligned} \tag{A.15}$$

As for second component v_2, we have:

$$\begin{aligned} & a^2_{(0,0),(0,0)} = 0\,; \\ & b^2_{(0,j),(0,k)} = 0 && \text{for } j \in [F+1; 0],\ k \in [0, 2^{-j}-1]\,; \\ & c^2_{(j,0),(k,0)} = (-2^{-j+2})\, c^{\text{div}}_{(j,0),(k,0)} && \text{for } j \in [F+1; 0],\ k \in [0, 2^{-j}-1]\,; \\ & d^2_{\boldsymbol{j},\boldsymbol{k}} = (-2^{-j_1+2})\, d^{\text{div}}_{\boldsymbol{j},\boldsymbol{k}} && \text{for } \boldsymbol{j} \in [F+1; 0]^2,\ \boldsymbol{k} \in [0; 2^{-j_1}-1] \times [0; 2^{-j_2}-1]\,. \end{aligned} \tag{A.16}$$

Once v_1 and v_2 coefficients have been determined, the motion is obtained by applying the appropriate inverse wavelet transform to each set of coefficients. Regarding v_1, filters (h_1, g_1) are used for the first dimension and (h_0, g_0) for the second; conversely for v_2. Whereas previous reconstruction algorithm Section A.3.1 requires one inverse transform at most, here two transforms are necessary,

which makes it less interesting in terms of computations. However, and contrary to the previous algorithm, the system can be inverted in order to recover stream function z coefficients from those of $\boldsymbol{v}$. It gives:

$$
\begin{aligned}
&a^{\mathrm{div}}_{(0,0),(0,0)} = 0\,; \\
&b^{\mathrm{div}}_{(0,j),(0,k)} = \frac{1}{2^{-j+2}}\, b^1_{(0,j),(0,k)} && \text{for } j \in [F+1;0],\ k \in [0, 2^{-j}-1]\,; \\
&c^{\mathrm{div}}_{(j,0),(k,0)} = \frac{-1}{2^{-j+2}}\, c^2_{(j,0),(k,0)} && \text{for } j \in [F+1;0],\ k \in [0, 2^{-j}-1]\,; \\
&d^{\mathrm{div}}_{\boldsymbol{j},\boldsymbol{k}} = \frac{1}{2^{-j_2+2}+2^{-j_1+2}}\left[d^1_{\boldsymbol{j},\boldsymbol{k}} - d^2_{\boldsymbol{j},\boldsymbol{k}}\right] && \text{for } \boldsymbol{j} \in [F+1;0]^2,\ \boldsymbol{k} \in [0; 2^{-j_1}-1] \times [0; 2^{-j_2}-1]\,.
\end{aligned}
\tag{A.17}
$$

These relations are displayed in Fig. A.2, pseudocode for decomposition and reconstruction algorithms are given Fig. A.7 and Fig. A.8.

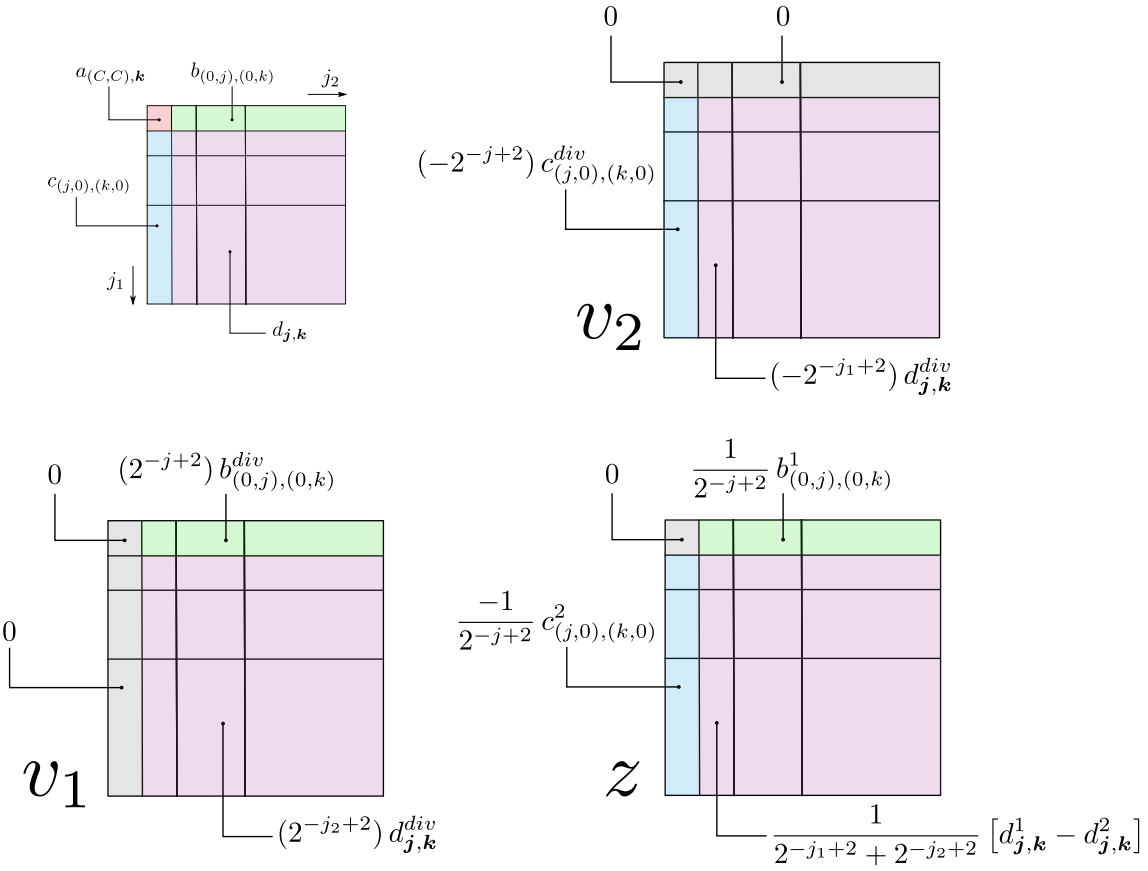


Figure A.2: Multiscale divergence-free decomposition/reconstruction schemes. *Top left*: disposition of coefficient sets a (*red*), b (*green*), c (*blue*), d (*pink*). v_1 coeff. from z coeff. (Eq. (A.15), *bottom left*); v_2 coeff. from z coeff. (Eq. (A.16), *top right*); z coeff. from v_1, v_2 coeff. (Eq. (A.17), *bottom right*).

A.3.3 DFD Gradient

The DFD gradient with respect to stream function z coefficients is obtained by applying the chain formula.

Gradient w.r.t. Fine Approximation Coefficients

Gradient of DFD functional with respect to a given fine approximation coefficient $a^{\text{div}}_{(F,F),(p_1,p_2)}$ is given by:

$$\begin{aligned}\frac{\partial J_{\text{DFD}}}{\partial a^{\text{div}}_{(F,F),(p_1,p_2)}}(\boldsymbol{v}) = 2^{-F}\left\langle \partial x_1 I_1(\cdot+\boldsymbol{v}(\cdot))\big[I_0 - I_1(\cdot+\boldsymbol{v}(\cdot))\big]\,;\, \varphi^1_{F,p_1}\otimes\varphi^0_{F,p_2} - \varphi^1_{F,p_1}\otimes\varphi^0_{F,p_2+1}\right\rangle_{\mathbf{L}^2}\\ - 2^{-F}\left\langle \partial x_2 I_1(\cdot+\boldsymbol{v}(\cdot))\big[I_0 - I_1(\cdot+\boldsymbol{v}(\cdot))\big]\,;\, \varphi^0_{F,p_1}\otimes\varphi^1_{F,p_2} - \varphi^0_{F,p_1+1}\otimes\varphi^1_{F,p_2}\right\rangle_{\mathbf{L}^2},\end{aligned} \tag{A.18}$$

for all $0 \le p_1, p_2 \le 2^{-F}-1$. Periodization of the basis has to be taken into account near borders:

$$p_i = 2^{-F}-1 \Rightarrow p_i + 1 = 0\,.$$

Moreover, using Mallat's approximation (3.1.3) for fine approximation coefficients simplifies previous expression and leads to:

$$\begin{aligned}\frac{\partial J_{\text{DFD}}}{\partial a^{\text{div}}_{(F,F),(p_1,p_2)}}(\boldsymbol{v}) = &\big(\partial x_1 I_1(\cdot+\boldsymbol{v}(\cdot))\big[I_0 - I_1(\cdot+\boldsymbol{v}(\cdot))\big]\big)_{p_1,p_2}\\ &- \big(\partial x_1 I_1(\cdot+\boldsymbol{v}(\cdot))\big[I_0 - I_1(\cdot+\boldsymbol{v}(\cdot))\big]\big)_{p_1,p_2+1}\\ &- \big(\partial x_2 I_1(\cdot+\boldsymbol{v}(\cdot))\big[I_0 - I_1(\cdot+\boldsymbol{v}(\cdot))\big]\big)_{p_1,p_2}\\ &+ \big(\partial x_2 I_1(\cdot+\boldsymbol{v}(\cdot))\big[I_0 - I_1(\cdot+\boldsymbol{v}(\cdot))\big]\big)_{p_1+1,p_2}\,.\end{aligned} \tag{A.19}$$

Gradient values with respect to multiscale coefficients can be obtained using direct formulas below, or by applying the appropriate forward transform – with primal filters (h_1, g_1) – to the values of (A.18).

Gradient w.r.t. Multiscale Coefficients

Let α be the set of all z coefficients in Eq. A.17 and β^1, β^2 the sets of all v_1 and v_2 coefficients in Eq. A.15 and A.16:

$$\frac{\partial J_{\text{DFD}}(\boldsymbol{v})}{\partial \alpha_p} = \int_\Omega \sum_{i=1,2}\left(\frac{\partial}{\partial v_i}\big[I_0(\boldsymbol{x}) - I1(\boldsymbol{x}+\boldsymbol{v}(\boldsymbol{x}))\big]^2 \sum_q \frac{\partial v_i}{\partial \beta^i_q}\frac{\partial \beta^i_q}{\partial \alpha_p}\right) d\boldsymbol{x}\,. \tag{A.20}$$

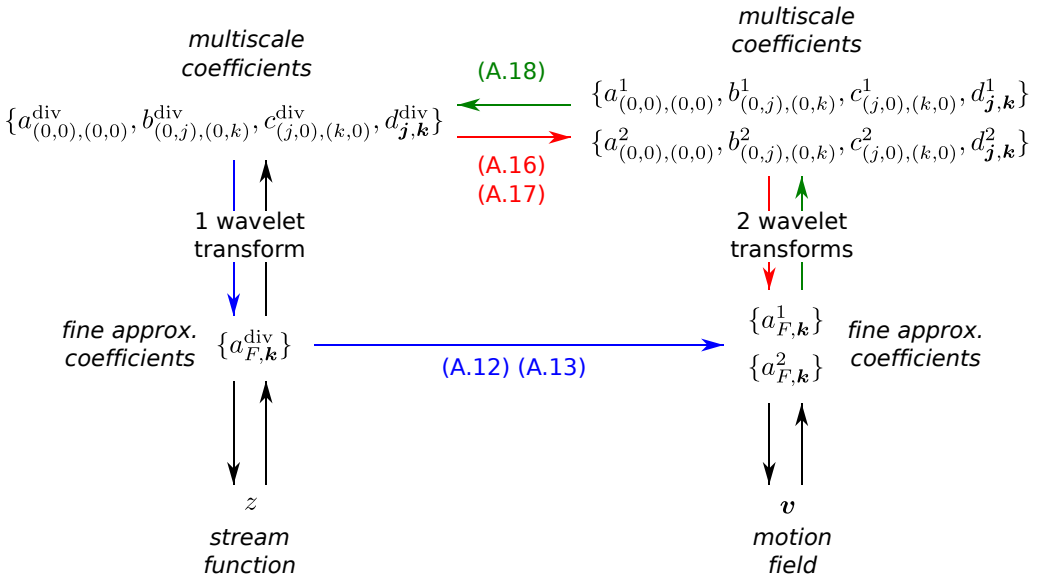
We finally obtain:

$$\begin{aligned}&\frac{\partial J_{\text{DFD}}}{\partial a^{\text{div}}_{(0,0),(0,0)}}(\boldsymbol{v}) = 0\,;\\ &\frac{\partial J_{\text{DFD}}}{\partial b^{\text{div}}_{(0,j),(0,k)}}(\boldsymbol{v}) = 2^{-j+2}\left\langle \partial x_1 I_1(\cdot+\boldsymbol{v}(\cdot))\big[I_0 - I_1(\cdot+\boldsymbol{v}(\cdot))\big]\,;\, \varphi^1\otimes\psi^0_{j,k}\right\rangle_{\mathbf{L}^2}\\ &\frac{\partial J_{\text{DFD}}}{\partial c^{\text{div}}_{(j,0),(k,0)}}(\boldsymbol{v}) = -2^{-j+2}\left\langle \partial x_2 I_1(\cdot+\boldsymbol{v}(\cdot))\big[I_0 - I_1(\cdot+\boldsymbol{v}(\cdot))\big]\,;\, \psi^0_{j,k}\otimes\varphi^1\right\rangle_{\mathbf{L}^2}\\ &\frac{\partial J_{\text{DFD}}}{\partial d^{\text{div}}_{\boldsymbol{j},\boldsymbol{k}}}(\boldsymbol{v}) = \begin{array}{l} 2^{-j_2+2}\left\langle \partial x_1 I_1(\cdot+\boldsymbol{v}(\cdot))\big[I_0 - I_1(\cdot+\boldsymbol{v}(\cdot))\big]\,;\, \psi^1_{j_1,k_1}\otimes\psi^0_{j_2,k_2}\right\rangle_{\mathbf{L}^2}\\ -2^{-j_1+2}\left\langle \partial x_2 I_1(\cdot+\boldsymbol{v}(\cdot))\big[I_0 - I_1(\cdot+\boldsymbol{v}(\cdot))\big]\,;\, \psi^0_{j_1,k_1}\otimes\psi^1_{j_2,k_2}\right\rangle_{\mathbf{L}^2},\end{array}\end{aligned} \tag{A.21}$$

with $\boldsymbol{j} \in [F+1;0]^2$, $\boldsymbol{k} \in [0;2^{-j_1}-1]\times[0;2^{-j_2}-1]$ and $j \in [F+1;0]$, $k \in [0,2^{-j}-1]$. It is important to notice that these gradient values are obtained by wavelet decomposition using *primal* wavelet filters (h_0, g_0) and (h_1, g_1).

Synthesis

Relations between stream function z, motion field $\boldsymbol{v} = \mathbf{curl}(z)$ and their respective fine approximation or multiscale coefficients are synthesized Figure A.3, as well as the steps required for DFD gradient computation.



(a) Motion reconstruction/decomposition

(b) DFD gradient

Figure A.3: Synthesis of relations for divergence-free transforms (a) and DFD gradient (b). Within the context of motion estimation, one usually works with multiscale coefficients of stream function z. Motion $\boldsymbol{v}$ can be rebuilt following either the *blue* path (one wavelet transform) or the *red* one (two transforms but less linear combinations). Stream function can be recovered from motion following the unique *green* path.

A.4 Pseudocodes

This section gathers pseudocodes for divergence-free filters construction (Figures A.4, A.4) and relations between stream function z and divergence-free motion $\boldsymbol{v}$ coefficients (Figures A.6, A.7, A.8). Regarding filters construction, filters supports are not necessarily in $\mathbb{N}$, and these supports are modified by integration/derivation operations. In order to simplify notations, the following convention is adopted:

– In algorithms below, an index (e.g. k in $h[k]$) refers to the memory index. Hence if the length of filter h is L, then $0 \leq k \leq L-1$.

– The actual "physical" index is obtained via the corresponding offset i_h (the lower bound of h support): for $0 \leq k \leq L-1$, $h[k]$ corresponds to the value at actual index $k + i_h$.

```
        /**** (h0, h̃0) from (h1, h̃1) ****/

[h0, h̃0, ih0, ih̃0] = filters_get0_from1(h1, h̃1, ih1, ih̃1)
 {
   //-primal h0 (derivation)
   //-------------------------------------
   L = length(h1)-1; //h0 is shorter than h1
   ih0 = ih1; //offsets are the same
   h0[0] = 2*h1[0]; //first element
   for (int k=1; k<L ; k++){
     h0[k] = 2*h1[k]-h0[k-1];
   }

   //-dual h̃0 (integration)
   //-------------------------------------
   L = length(h̃1)+1; //h̃0 is longer than h̃1
   ih̃0 = ih̃1; //offsets are the same
   h̃0[0] = .5*h̃1[0]; //first element
   for (int k=1; k<L-1; k++){
     h̃0[k] = .5*(h̃1[k-1]+h1[k]);
   }
   h̃0[L-1] = .5*h̃1[L-2]; //last element
 }
```

Figure A.4: Algorithm to compute $(h_0, \tilde{h}_0)$ from $(h_1, \tilde{h}_1)$.

```
        /**** (h1,h̃1) from (h0,h̃0) ****/

[h1, h̃1, ih1, ih̃1] = filters_get1_from0(h0, h̃0, ih0, ih̃0)
 {
   //-primal h1 (integration)
   //-------------------------------------
   L = length(h0)+1; //h1 is longer than h0
   ih1 = ih0; //offsets are the same
   h1[0] = .5*h0[0]; //first element
   for (int k=1; k<L-1; k++){
     h1[k] = .5*(h0[k-1]+h0[k]);
   }
   h1[L-1] = .5*h0[L-2]; //last element

   //-dual h̃1 (derivation)
   //-------------------------------------
   L = length(h̃0)-1; //h̃1 is shorter than h̃0
   ih̃1 = ih̃0+1; //offset is shifted by +1
   h̃1[L-1] = .5*h̃0[L]; //last element
   for (int k=L-2 ; k>=0 ; k--){
     h̃1[k] = 2*h̃0[k+1] - h1[k+1];
   }
 }
```

Figure A.5: Algorithm to compute $(h_1, \tilde{h}_1)$ from $(h_0, \tilde{h}_0)$.

```
/**** Reconstruction from Fine Approximation ****/

//-get fine approximation of stream function
// from multiscale coefficients
//-------------------
{a^div_{(F,F),k}}_k = IW{a^div_{(0,0),(0,0)}, b^div_{(0,j),(0,k)}, c^div_{(j,0),(k,0)}, d^div_{j,k}}; //filters (h1 ,g1)

//-build {a^1_{(F,F),k}}_k by finite difference along 2nd dimension
//-------------------
for (int k1=0 ; k1 < 2^{-F} ; k1++){
    for (int k2=1 ; k2 < 2^{-F} ; k2++){
        a^1_{(F,F),(k1,k2)} = 2^{-F} [a^div_{(F,F),(k1,k2)} - a^div_{(F,F),(k1,k2-1)}];
    }
    //periodization for first element
    a^1_{(F,F),(k1,0)} = 2^{-F} [a^div_{(F,F),(k1,0)} - a^div_{(F,F),(k1,2^{-F}-1)}];
}
//-return to v1
//-------------------
v1 = 2^{-F} * a^1_{(F,F)}; //this is Mallat's approximation

//-build {a^2_{(F,F),k}}_k by finite difference along 1st dimension
//-------------------
for (int k2=0 ; k2 < 2^{-F} ; k2++){
    for (int k1=1 ; k1 < 2^{-F} ; k1++){
        a^2_{(F,F),(k1,k2)} = -2^{-F} [a^div_{(F,F),(k1,k2)} - a^div_{(F,F),(k1-1,k2)}];
    }
    //periodization for first element
    a^2_{(F,F),(0,k2)} = -2^{-F} [a^div_{(F,F),(0,k2)} - a^div_{(F,F),(2^{-F}-1,k2)}];
}
// return to v2
//-------------------
v2 = 2^{-F} * a^2_{(F,F)}; //again Mallat's approximation
```

Figure A.6: Algorithm for $\boldsymbol{v}$ reconstruction from z fine approximation coefficients.

```
/**** Reconstruction from Multiscale Coefficients ****/

//v1 component
//————————————————
//wavelet coefficients
for (int j1=0; j1 < F; j1++){
    for (int k1=0 ; k1 < 2^-j1 ; k1++){
        for (int j2=0 ; j2 < F ; j2++){
            for (int k2=0 ; k2 < 2^-j2 ; k2++){
                d^1_{j,k} = (2^{-j2+2}) d^{div}_{j,k};
            }
        }
    }
}
//approx coefficients
for (int j=0 ; j < F ; j++){
    for (int k=0 ; k < 2^-j ; k++){
        b^1_{(0,j),(0,k)} = (2^{-j+2}) b^{div}_{(0,j),(0,k)};
        c^1_{(j,0),(k,0)} = 0;
    }
}
a^1_{(0,0)(0,0)} = 0;
//inverse transform
v1 = IW{a^1_{(0,0),(0,0)}, b^1_{(0,j),(0,k)}, c^1_{(j,0),(k,0)}, d^1_{j,k}}; //filters (h1 ,g1) and (h0 ,g0)

//v2 component
//————————————————
//wavelet coefficients
for (int j1=0; j1 < F; j1++){
    for (int k1=0 ; k1 < 2^-j1 ; k1++){
        for (int j2=0 ; j2 < F ; j2++){
            for (int k2=0 ; k2 < 2^-j2 ; k2++){
                d^2_{j,k} = (-2^{-j1+2}) d^{div}_{j,k};
            }
        }
    }
}
//approx coefficients
for (int j=0 ; j < F ; j++){
    for (int k=0 ; k < 2^-j ; k++){
        b^2_{(0,j),(0,k)} = 0;
        c^2_{(j,0),(k,0)} = (2^{-j+2}) b^{div}_{(j,0),(k,0)};
    }
}
a^2_{(0,0)(0,0)} = 0;
//inverse transform
v2 = IW{a^2_{(0,0),(0,0)}, b^2_{(0,j),(0,k)}, c^2_{(j,0),(k,0)}, d^2_{j,k}}; //filters (h0 ,g0) and (h1 ,g1)
```

Figure A.7: Algorithm for $\boldsymbol{v}$ reconstruction from z multiscale coefficients.

```
        /**** Decomposition ****/

//get v1, v2 multiscale coefficients
//--------------------
//forward transform
{a^1_{(0,0),(0,0)}, b^1_{(0,j),(0,k)}, c^1_{(j,0),(k,0)}, d^1_{j,k}} = FW{v1}; //filters (h̃1, g̃1) and (h̃0, g̃0)
{a^2_{(0,0),(0,0)}, b^2_{(0,j),(0,k)}, c^2_{(j,0),(k,0)}, d^2_{j,k}} = FW{v2}; //filters (h̃0, g̃0) and (h̃1, g̃1)

//make z multiscale coefficients
//--------------------
//wavelet coefficients
for (int j1=0; j1 < F; j1++){
    for (int k1=0 ; k1 < 2^{-j1} ; k1++){
        for (int j2=0 ; j2 < F ; j2++){
            for (int k2=0 ; k2 < 2^{-j2}; k2++){
                d^div_{j,k} = 1/(2^{-j2+2} + 2^{-j1+2}) [d^1_{j,k} - d^2_{j,k}];
            }
        }
    }
}
//approx coefficients
for (int j=0 ; j < F ; j++){
    for (int k=0 ; k < 2^{-j}; k++){
        b^div_{(0,j),(0,k)} = 1/(2^{-j+2}) b^1_{(0,j),(0,k)};
        c^div_{(j,0),(k,0)} = -1/(2^{-j+2}) c^2_{(j,0),(k,0)};
    }
}
a^div_{(0,0)(0,0)} = 0;
```

Figure A.8: Algorithm for $\boldsymbol{v}$ decomposition.

Appendix B

Computational Costs of Filterbanks

B.1 Usual Filterbanks

We consider filters h, g of length K and a current approximation a_j at a given scale $j \geq 0$ (i.e. with $2^j \times 2^j$ coefficients). Applying one step of the decomposition filter bank (Figure 3.6a) to compute $\{a_{j-1}, d^1_{j-1}, d^2_{j-1}, d^3_{j-1}\}$ from a_j requires $8K2^{2(j-1)}$ multiplications, using the factorized form:

(i) for each of the 2^j rows:
- $2^{(j-1)}$ convolutions with $\bar{h}$;
- $2^{(j-1)}$ convolutions with $\bar{g}$;

which give $4K2^{2(j-1)}$ multiplications;

(ii) for each of the $2 \times 2^{(j-1)}$ columns:
- $2^{(j-1)}$ convolutions with $\bar{h}$;
- $2^{(j-1)}$ convolutions with $\bar{g}$;

giving again $4K2^{2(j-1)}$ multiplications;

So that the grand total is $8K2^{2(j-1)}$ multiplications. For an input image of $N = 2^F \times 2^F$ pixels, iteration of the filter bank to reach coarse scale C gives:

$$\begin{aligned}
\sum_{j=C+1}^{F} 8K2^{2(j-1)} &= 2K(2^F)^2 \sum_{j=C+1}^{F} \left(2^{(j-F)}\right)^2 \\
&= 2KN \sum_{j=C+1}^{F} \left(\frac{2^j}{2^F}\right)^2 \\
&= 2KN \left(1 + \frac{1}{4} + \frac{1}{16} + \cdots\right) \\
&< \frac{8}{3} KN \,. \qquad \text{(B.1)}
\end{aligned}$$

B.2 Modified Filterbanks

During the estimation process, at a given scale s, finer scales coefficients ($F < j < s$) are not considered. DFD gradient is computed by projecting the two terms terms (5.13) on the considered basis. By dropping computation of useless details d^i_j, $i = 1..3$ at fine scales $F < j < s$, there are:

(i) for each of the 2^j rows:
- $2^{(j-1)}$ convolutions with $\bar{h}$;

which give $2K2^{2(j-1)}$ multiplications;

(ii) for each of the $2^{(j-1)}$ columns left:
- $2^{(j-1)}$ convolutions with $\bar{h}$;

which give $K2^{2(j-1)}$ multiplications;

then the total number of required multiplications drops down to $3K2^{2(j-1)}$, hence saving $5/8 = 62.5\%$ of multiplications at each step involving useless details. A step of the reconstruction filter bank to get a_j from a_{j-1} (and eventually $\{d^1_{j-1}, d^2_{j-1}, d^3_{j-1}\}$) requires the same amount of operations for both cases. Therefore, at the reconstruction of motion $\boldsymbol{v}$ from a partial set of coefficients (up to scale s), an identical reduction of computations is achieved by explicitly neglecting contributions from finer scales $F < j < s$.