This CVPR paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

Task-Driven Wavelets using Constrained Empirical Risk Minimization

Eric Marcus[*,1, 2], Ray Sheombarsing[*,1], Jan-Jakob Sonke[1], Jonas Teuwen[1]

[1]Netherlands Cancer Institute, [2]University of Amsterdam

{e.marcus, r.sheombarsing, j.sonke, j.teuwen}@nki.nl

Abstract

Deep Neural Networks (DNNs) are widely used for their ability to effectively approximate large classes of functions. This flexibility, however, makes the strict enforcement of constraints on DNNs a difficult problem. In contexts where it is critical to limit the function space to which certain network components belong, such as wavelets employed in Multi-Resolution Analysis (MRA), naive constraints via additional terms in the loss function are inadequate. To address this, we introduce a Convolutional Neural Network (CNN) wherein the convolutional filters are strictly constrained to be wavelets. This allows the filters to update to task-optimized wavelets during the training procedure. Our primary contribution lies in the rigorous formulation of these filters via a constrained empirical risk minimization framework, thereby providing an exact mechanism to enforce these structural constraints. While our work is grounded in theory, we investigate our approach empirically through applications in medical imaging, particularly in the task of contour prediction around various organs, achieving superior performance compared to baseline methods.

1. Introduction

Empirical risk minimization (ERM) is currently the most prevalent framework for supervised learning. The goal is to find a function to map inputs to associated targets for all representative (potentially unseen) observations. To find such a mapping, one introduces a loss function to quantify the discrepancy between observed and predicted targets. An optimal map is then found by minimizing the expected loss. In large-scale settings, such as deep learning, the resulting minimization problem is solved using Stochastic Gradient Descent (SGD) and various variants thereof, see [15, 24, 25, 40] for instance. As deep learning applications become more specialized, domain-specific needs become increasingly vital. These are often formulated in terms of constraints on

[*]These authors contributed equally to this work.

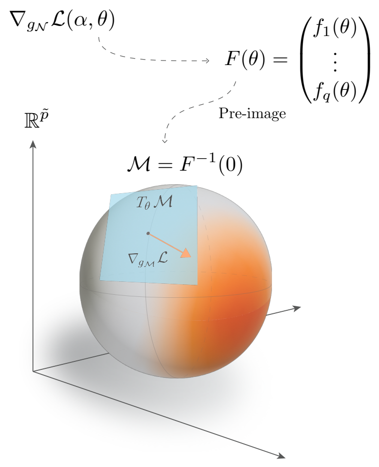

$\nabla_{g_{\mathcal{N}}}\mathcal{L}(\alpha,\theta)$
$F(\theta) = \begin{pmatrix} f_1(\theta) \\ \vdots \\ f_q(\theta) \end{pmatrix}$
Pre-image
$\mathbb{R}^{\tilde{p}}$
$\mathcal{M} = F^{-1}(0)$
$T_\theta\mathcal{M}$
$\nabla_{g_{\mathcal{M}}}\mathcal{L}$

Figure 1. An overview of the Constrained Empirical Risk Minimization (CERM) framework: on the top left side, the full gradient of the loss $\mathcal{L}$ is shown on the total space $\mathcal{N}$ (containing both the constrained and unconstrained space), with α are the unconstrained parameters, and θ the constrained ones. The constraints can be written in the form $F(\theta) = 0$; the solution set $F^{-1}(0)$ is an embedded submanifold $\mathcal{M}$ of $\mathbb{R}^{\tilde{p}}$. The constrained parameters are updated by following a path on $\mathcal{M}$ in the direction of the negative gradient on this manifold. This constrained part of the full gradient is contained in the tangent space $T_\theta\mathcal{M}$ of the submanifold. The color of the manifold indicates the value of the loss function. By restricting the relevant components of our descent trajectories to the embedded submanifold, we *always* satisfy the constraints imposed by F. The gradient and parameter updates for the unconstrained parameters α are computed as usual using standard SGD for flat space (not shown). In our wavelet scenario, the manifold thus consists of all admissible wavelets of a certain dimension.

the permissible mappings. For instance, the constraint for translation-equivariance led to the development and success of convolution neural networks (CNNs) [17]. In general, however, it is a highly non-trivial task to construct network architectures that satisfy a set of constraints, if they exist at all. It is, therefore, common practice to incorporate constraints directly into the loss function by including additional terms, usually referred to as “soft” constraints. This setup, however, has the drawback that the constraints are only approximately (on average) satisfied due to the formulation of ERM. Moreover, incorporating many different objectives in a loss function may lead to suboptimal results for the individual objectives. Another common strategy, used for many types of constraints, e.g., flip invariance, is based on data augmentation. However, not every constraint can be achieved using data augmentation, and it also leads, at best, to constraints being approximately satisfied. Furthermore, if we wish to constrain neural networks to specific function classes, such as wavelets, an additional loss term is no longer sufficient: a wavelet is only a wavelet whenever the constraints are satisfied *exactly*.

Approaches which circumvent loss-based (soft) constraints have been recently proposed, see [2, 4, 28, 30, 32] for example. Most notably, we mention the field of Geometric Deep Learning (GDL), which focuses on incorporating constraints arising from symmetries of the domain directly into the networks themselves, see [8, 9, 39] and the references therein. In GDL, one considers very specific but powerful types of constraints, namely that network layers are equivariant with respect to some group action. Not all constraints arise as equivariance principles, however. A large class of examples comes from highly specialized requirements on the output of a neural network, e.g., that the output is a divergence-free vector field, a contour, or perhaps a surface. For example, in medical image segmentation, a natural requirement is that the output of the segmentation network corresponds to a continuous (closed) curve. Creating architectures for such constraints is out of the realm of GDL and, in general, poses a challenging task; no prescribed methods exist to do so.

To facilitate these different types of constraints, we present a framework, called Constrained Empirical Risk Minimization (CERM), where we directly formulate and solve the constrained ERM problem in the space where the constraints are satisfied. This space forms, under mild conditions, a Riemannian manifold. As long as we stay on this manifold, the constraints will be satisfied *exactly* up to numerical precision. In particular, we explain how to perform SGD on such Riemannian manifolds arising from a finite-dimensional system of equations. Although we are motivated by the ability to constrain to wavelets, the framework, in principle, encompasses constraints of the form $F(W) = 0$, with W the network weights. Our method heavily relies on the Implicit Function Theorem, which is used to construct so-called graph-charts amenable to numerical computations. This allows for efficient evaluation of the (induced) Riemannian metric and gradients, which are vital for performing SGD on Riemannian manifolds using only the intrinsic geometry.

As mentioned, we will present the tools and implementations of convolutional networks whose filters will form wavelets. There are several tasks at which one expects wavelet-based neural networks to excel. Wavelet decompositions naturally lend themselves to representing continuous objects such as curves, images, vector fields, or other higher-dimensional objects. Hence any task where the object of interest can be identified with a smooth or continuous function is well-suited for wavelet-based neural networks. There is an abundance of such examples to be found in computer vision, e.g., boundary prediction, image registration, and so forth. Another family of exciting applications can be found in signal analysis, e.g., in compression and denoising, where wavelets are long-standing tools that have proven to be extremely efficient [26]. The main idea in these areas is to extract information about noise, smoothness, and even singularities, through analysis of the wavelet coefficients. Subsequently, by modifying a subset of the coefficients, e.g., through thresholding, the signal can be cleaned up or denoised. More recently, wavelets have also received increasing attention in the context of neural networks. The applications are broad; some references include [22, 33, 41], ranging from efficient normalizing flows, noise-robust training, scale-equivariant models, and many more. The wavelet layers we create in this work are included in our code, which we will make available publicly[*].

Related work An earlier attempt at incorporating constraints is described in [11]. However, this method is only able to deal with linear constraints, which is hence incompatible with the wavelet constraints we will derive. Other works include [20, 21, 27], which are related to the method of Lagrange multipliers and have their optimization and training dynamics largely determined by variants of Newton's method. We discuss the differences between our SGD-compatible method and that of Lagrange multipliers in more detail in the supplementary material 6.2. A comprehensive overview of all constrained optimization methods, however, is out of the scope of this paper, and we therefore focus our attention to techniques based on SGD on Riemannian manifolds.

Performing SGD on Riemannian manifolds is well-established and has been studied extensively, see [6, 14, 34, 35, 42]. To the best of our knowledge, current methods require an explicit description of charts on the underlying manifold and adopt a completely extrinsic point of view. This process involves manual computations on paper on a per-case basis. For instance, the authors of [34, 35] present methods for three specific cases: the space of positive definite matrices, Grassman, and Stiefel manifolds. We mention the method in [7, Chapter 7.7] in particular, which is most closely related to ours. This method adopts an extrinsic

[*]https://github.com/NKI-AI/CERM

perspective on RSGD and effectively performs all computations in the ambient (vector) space in which $\mathcal{M}$ is embedded. This extrinsic point of view leads to an algorithm for "projecting" paths in the ambient space onto descent paths on $\mathcal{M}$. This projection involves an additional nonlinear optimization problem and presents highly non-trivial issues for numerical implementation on general manifolds, as mentioned by the authors. In contrast, the method we present here is entirely intrinsic to $\mathcal{M}$ and uses a local coordinate system amenable to numerical computations, allowing us to deal with a general class of manifolds. In particular, it allows us to incorporate the constraints for the wavelet straightforwardly. Most importantly, we follow a specific descent path on $\mathcal{M}$; the geodesic in the direction of the negative gradient of the objective, a process made possible through our explicit understanding of the Riemannian metric. Our intrinsic perspective ensures that all computations are performed directly on the manifold, avoiding additional optimization problems. In particular, the dynamics of our optimization algorithm are solely driven by the gradient flow of the objective.

Overview The contributions of this paper are ordered as follows. In Sec. 2, we introduce the theory of the constrained optimization. In Sec. 3, we apply the framework to our topic of interest: constraining filters of a CNN to form wavelets. We will show experiments in Sec. 4. In particular, this section will show our novel application, where we find data-driven wavelets for contour prediction. Specifically, we use wavelet networks to perform contour prediction in the medical domain, where we outperform strong baselines.
We remark that an extensive, formal, and detailed investigation of the CERM framework, MRA, comparisons with other frameworks, and several implementation details are provided in [16], to which we refer the reader for more details.

2. Constrained Empirical Risk Minimization

In this section, we introduce a framework for performing ERM with constraints, which we will refer to as Constrained Empirical Risk Minimization (CERM). We start with a brief review of the traditional ERM setup [36, 37] introducing the necessary terminology, notation, and assumptions. Next, we explain how to incorporate constraints into the ERM framework in the form of a system of equations. We provide sufficient conditions on the system of equations to guarantee that the solution set is a Riemannian manifold. Finally, we provide an explanation for how the Riemannian metric and associated geometric quantities can be numerically evaluated, which enables us to perform SGD on the Riemannian manifold of interest directly. The discussion will be high-level; each step is explored in full mathematical detail in Sec. 6 [16].

2.1. ERM Setup

Let X and Y be random variables whose realizations are interpreted as input data and corresponding targets, respectively. The sample spaces of X and Y are denoted by $\mathcal{X}$ and $\mathcal{Y}$, respectively. As a side note, self-supervised settings also fall into this framework, in which case the target Y is created on the fly as a function of X.

Empirical risk minimization The goal of the ERM framework is to find a map $G : \mathcal{X} \to \mathcal{Y}$ such that $G(x) \approx y$ for most realizations of (X, Y). The discrepancy between predicted and observed targets is quantified using a loss function $L : \mathcal{Y} \times \mathcal{Y} \to [0, \infty)$. Without loss of generality, we assume that L assumes positive values only and that L decreases as the accuracy of predictions increases. The main objective of ERM is to find an optimal map G^*, which solves the minimization problem

$$\min_{G \in \mathcal{G}} \mathbb{E}\left(L(G(X), Y)\right). \tag{1}$$

Here $\mathcal{G}$ is a suitable subset of functions.

In all our applications, we assume that X and Y are random vectors with sample spaces $\mathcal{X} = \mathbb{R}^n$ and $\mathcal{Y} = \mathbb{R}^m$. Furthermore, we assume $\mathcal{G}$ is a parametric set that consists of mappings $\mathcal{G} = \{G(\cdot, \xi) : \xi \in \mathbb{R}^p\}$, where $G : \mathbb{R}^n \times \mathbb{R}^p \to \mathbb{R}^m$ is a continuously differentiable map. This includes, for example, ordinary neural networks, where the parameters are the weights. With these assumptions in place, the minimization problem in (1) is equivalent to minimizing over the parameters ξ instead. In practice, we have access to only a finite set of observations, which are assumed to be i.i.d., and the objective in (1) is replaced by an arithmetic average.

2.2. Imposing constraints

In this section, we explain how to incorporate constraints on a subset of the parameters ξ directly into the ERM framework. We consider constraints given in the form of a system of equations. To be very precise, let $F : \mathbb{R}^{\tilde{p}} \to \mathbb{R}^q$ be a twice continuously differentiable map, where $\tilde{p}$ denotes the number of constrained parameters, and q is the number of equations. We assume that $q < \tilde{p} \leq p$. For notational convenience, we decompose $\mathbb{R}^p = \mathbb{R}^{p-\tilde{p}} \oplus \mathbb{R}^{\tilde{p}}$, where the first and second subspaces correspond to the unconstrained and constrained parameters, respectively. We take $\pi_{p-\tilde{p}} : \mathbb{R}^p \to \mathbb{R}^{p-\tilde{p}}$ and $\pi_{\tilde{p}} : \mathbb{R}^p \to \mathbb{R}^{\tilde{p}}$ to be the projections onto the unconstrained and constrained parameter subspace, respectively. We will denote the unconstrained and constrained parameters by α and θ, respectively, i.e., $\alpha = \pi_{p-\tilde{p}}(\xi)$ and $\theta = \pi_{\tilde{p}}(\xi)$.

Definition 2.1 (CERM)**.** Let $G : \mathbb{R}^n \times \mathbb{R}^p \to \mathbb{R}^m$ be a continuously differentiable parameterization of mappings and $F : \mathbb{R}^{\tilde{p}} \to \mathbb{R}^q$ a twice continuously differentiable constraint, where $q < \tilde{p} \leq p$. Suppose $L : \mathbb{R}^m \times \mathbb{R}^m \to [0, \infty)$ is a continuously differentiable loss function. The *constrained* ERM problem for (X, Y) with respect to (G, F, L) is defined by

$$\begin{cases} \min\limits_{\xi \in \mathbb{R}^p} \mathbb{E}\left(L(G(X,\xi), Y)\right), \\ \text{s.t. } F(\pi_{\tilde{p}}(\xi)) = 0. \end{cases} \tag{2}$$

Note the generality of the admissible mappings G in Definition 2.1. Although we will focus on neural networks from now on, the proposed framework applies to any parametric model, e.g., logistic or polynomial regression models. Next, we show that the CERM problem in (2) can be reformulated as an ordinary ERM problem on a Riemannian manifold $(\mathcal{N}, g_{\mathcal{N}})$, provided that the system of equations satisfies a mild non-degeneracy condition. This result allows us to consider the admissible parameters as a geometric object in its own right, whose intrinsic geometry we use to solve (2). The proof of this theorem can be found in Sec. 6.1 [16].

Theorem 2.2. *If zero is a regular (i.e., non-singular) value of F, then the CERM problem in (2) is equivalent to solving an ordinary ERM problem on a Riemannian manifold $\mathcal{N}$ with metric $g_{\mathcal{N}}$ of dimension $p - q$. Here $\mathcal{N} = \mathbb{R}^{p-\tilde{p}} \times \mathcal{M}$ is an embedded C^2-submanifold of $\mathbb{R}^p$ and $\mathcal{M} := F^{-1}(0)$. The equivalent minimization problem is given by*

$$\min_{(\alpha,\theta)\in\mathcal{N}} \mathbb{E}\left(L\left(G\left(X, \alpha \oplus \iota(\theta)\right), Y\right)\right), \tag{3}$$

where $\iota : \mathcal{M} \to \mathbb{R}^{\tilde{p}}$ is the inclusion map which embeds $\mathcal{M}$ in the ambient space.

From now on, we refer to the objective in (3) as simply the loss $\mathcal{L}$.

Graph coordinates on $\mathcal{N}$ The main idea is to directly minimize the loss on $\mathcal{N}$ using SGD, see Figure 1 for an overview. To perform SGD on this space, however, we need to choose a coordinate system in which we can evaluate the gradient of the loss and subsequently follow descent trajectories on $\mathcal{N}$. Now, the proof of Theorem 2.2 involves showing that $\mathcal{M}$ is an embedded submanifold of the ambient space $\mathbb{R}^{\tilde{p}}$. This is accomplished by constructing a special coordinate system, a so-called graph chart or graph coordinate system. This chart is uniquely determined by the constraint F and a prescribed zero $\theta \in \mathcal{M}$. It can be numerically evaluated at the point θ by solving a linear system of equations. In turn, this enables us to construct a chart on the manifold in which we can evaluate the Riemannian metric and the gradient $\nabla_{g_{\mathcal{N}}}\mathcal{L}(\alpha, \theta)$.

The crucial observation is that we can numerically construct coordinate systems in which we can follow paths *on* the manifold $\mathcal{N}$ in the direction of $-\nabla_{g_{\mathcal{N}}}\mathcal{L}(\alpha, \theta)$. In particular, the system of equations determining these so-called graph coordinate systems is fully determined by the derivative DF of the constraint F and the constraint itself. This observation enables a general implementation of graph coordinate systems independent of the explicit form of F. We only require an implementation of F and DF, where the latter may be obtained using automatic differentiation. We provide a sketch of the algorithm for computing the gradient in Algorithm 1. For the interested reader, we mention that a complete and detailed explanation of all steps can be found in Sec. 6 [16]. Before moving on, we mention the limitations of the CERM framework for other generic constraints.

Algorithm 1 Sketch of the computation of $\nabla_{g_{\mathcal{N}}}\mathcal{L}(\alpha, \theta)$ given $(\alpha, \theta) \in \mathcal{N}$. See Algorithm 2 in Sec. 6.5 [16] the detailed formal version.

```
1: Compute the derivative $DF(\theta)$ of the constraint $F$.
2: Compute chart-related derivatives.
3: Compute the metric $g_{\mathcal{N}}$ using the chart-related derivatives.
4: Compute the partial derivatives of the loss $\mathcal{L}$ with respect
   to the constrained components $\theta$.
5: Compute the components of $\nabla_{g_{\mathcal{M}}}\mathcal{L}(\alpha, \theta)$ using the
   above ingredients.
6: Compute the ordinary (unconstrained) components
   $\nabla_{g_{\text{flat}}}\mathcal{L}(\alpha, \theta)$ by evaluating $D_\alpha\mathcal{L}(\alpha, \theta)$.
```

Limitations As with most applications of deep neural networks, the available amount of computational power imposes limitations. Specifically, if the number of constrained parameters is large relative to the number of constraints, the computational demands will rise. Furthermore, setting up well-posed constraints can be a non-trivial task, e.g., symmetries in the constraints may result in redundancies obstructing the solvability of the associated systems of equations. Finally, the topology of the constrained submanifold may pose limitations as well. For instance, while the constraints will generically define a Riemannian manifold, it need not be connected. In such cases, the initialization of the network parameters will play an important role, as the gradient flow will stay on the connected component it was initialized on.

3. Multiresolution Analysis and CERM

This section presents the application of the aforementioned CERM framework to a non-trivial application to learn optimal wavelet bases for a given task. Many different types of wavelets, such as Haar, Daubechies, Gabor, etc., are used for a wide array of tasks. The specifics of the task demand different characteristics of the wavelets, such as the degree of smoothness, or the support of the wavelet. For example,

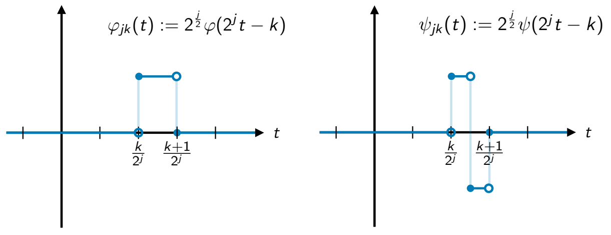
the CDF wavelets [10] are well-known due to their usage for both lossless and lossy compression in JPEG2000. In this section, we consider networks that will find task-driven wavelets, optimizing the wavelet characteristics to the task at hand.
Here, we will explain how to set up a system of equations (constraints) whose solution set corresponds to wavelets. We will apply these constraints in the next section to learn task-driven wavelets for predicting multi resolution decompositions of contours in the medical domain. While we only consider one-dimensional wavelets in this work, we stress that the framework is easily adapted to higher-dimensional domains, to decompose images, for instance, by using tensor products of one-dimensional bases.

3.1. Multiresolution Analysis

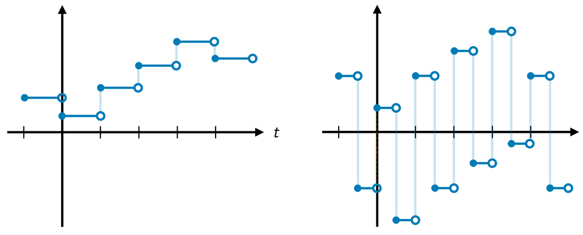
In this section, we briefly state what a Multiresolution Analysis (MRA) is. We closely follow the standard exposition, found in, e.g., [29, 31], and refer the reader to Sec. 7 [16] for a more in-depth and formal introduction. The MRA framework allows one to study a signal $\gamma \in L^2(\mathbb{R})$ at various resolution levels. The origin of this framework lies in the uncertainty principle of Fourier analysis, which states that a signal cannot be simultaneously localized in the time and frequency domain. MRA addresses this shortcoming by decomposing a signal on different discrete resolution levels. The idea is to construct a nested (exhausting) sequence of closed subspaces $V_j \subset L^2(\mathbb{R})$, each associated with a specific resolution level j. The subspaces V_j are spanned by integer shifts of an appropriate dilation of a so-called scaling map (or father wavelet) φ. The level of dilation determines the resolution level. Formally, we require that $(\varphi_{jk})_{k\in\mathbb{Z}}$ is an orthonormal basis for V_j, where $\varphi_{jk}(t) := 2^{\frac{j}{2}}\varphi(2^j t - k)$. In Fig. 2a and Fig. 2c, we show an example of that Haar-MRA; in particular, we show the father wavelet and an example of a subspace at a certain resolution level.

Decomposing a signal To analyze a signal γ at different resolution levels, we project it onto the subspaces V_j. The orthogonal projection of γ at level j is denoted by γ_j. The coefficients of γ_j with respect to the basis for V_j, denoted by $a_{jk}(\gamma)$, are called the *approximation coefficients* of γ. When a signal in V_{j+1} is projected onto V_j, the lost information is stored in the orthogonal complement W_j of V_j in V_{j+1}. This subspace is referred to as the detail subspace at level j. The decomposition of V_{j+1} into V_j and W_j allows the reconstruction of a signal at level $j+1$ from any lower level j_0 provided all the details in between are known. The corresponding decomposition and reconstruction algorithms give rise to the Discrete Wavelet Transform (DWT) and Mallat's Pyramid Algorithm. In Fig. 2b and Fig. 2d, we show examples of the mother wavelet and the detail subspace for the Haar wavelet.

A fundamental result, known as Mallat's Theorem, states



(a) Father wavelet (b) Mother wavelet



(c) V_j (d) W_j

Figure 2. Example of the Haar MRA: (a) Dilated translation of the Haar scaling map (or father wavelet) $\varphi = \mathbf{1}_{[0,1)}$. The approximation subspace V_j at level j consists of all step-functions with step-size 2^{-j}. (b) Dilated translation of the mother wavelet $\psi = \mathbf{1}_{[0,\frac{1}{2})} - \mathbf{1}_{[\frac{1}{2},1)}$. The detail subspace W_j is spanned by integer translations of the dilated mother wavelet ψ_j. (c) The approximation subspace at level j consists of all step-functions with step-size 2^{-j}. (d) Example of a function in the detail subspace at level j.

that the subspaces W_j can also be spanned by dilating and shifting a single map. More precisely, there exists a map $\psi \in W_0$, the so-called *mother wavelet*, such that the ψ_{jk} form an orthonormal basis for W_j, see [31]. The coefficients of the projection of (γ) onto W_j, are denoted by $d_j(\gamma) := (d_{jk}(\gamma))_{k\in\mathbb{Z}}$, and referred to as the *detail coefficients* of γ at resolution level j. We can summarize the MRA procedure roughly as

$$\gamma_j = \sum_k a_{j_0 k}\varphi_{j_0 k} + \sum_{j_0 \le l \le j-1} \sum_k d_{lk}\psi_{lk} , \qquad (4)$$

where φ denotes the father wavelet and ψ the mother wavelet. The MRA boils down to the specification of the first approximation coefficient and a number of detail coefficients.

Low and high pass filters As we mentioned before, an MRA is completely determined by a scaling map φ, which in turn is completely characterized by a so-called low-pass filter. We will use this observation to set up a system of constraints characterizing a finite-dimensional family of wavelets, such that we can use it in the CERM framework. To explain how this works, we first make a key observation. Since $V_0 \subset V_1$, there exists a unique sequence h characterizing φ in terms of the basis $(\varphi_{1k})_{k\in\mathbb{Z}}$ for V_1. The sequence h completely characterizes the scaling function and is called the *low-pass filter* or *wavelet filter* of the MRA. Similarly, since $\psi \in W_0 \subset V_1$, there exists a unique sequence g, the so-called *high-pass filter* associated to h, characterizing the mother wavelet. An example of an MRA is shown in Figure 2. In practice, to define an MRA, one only needs to specify an appropriate low-pass filter h, as for Mallat's mother wavelet, we have $g_k = (-1)^{k-1} h_{1-k}$. We refer the interested reader to the supplementary information for a detailed discussion of all these points [16].

3.2. MRA constraints

We are now ready to formulate and impose constraints on a sequence h to ensure that it is the low-pass filter of a scaling map φ. In practice, we only consider finite sequences. For notational convenience, we introduce the space $\mathcal{A}_M(\mathbb{R})$, which consists of one-dimensional real-valued two-sided sequences $(h_k)_{k=1-M}^{M-1}$ of order M; so $\mathcal{A}_M(\mathbb{R})$ is $2M-1$ dimensional.

Theorem 3.1. *Let $M \in \mathbb{N}_{\geq 3}$ be a prescribed order and define $F_M : \mathcal{A}_M(\mathbb{R}) \to \mathbb{R}^{M+1}$ by*

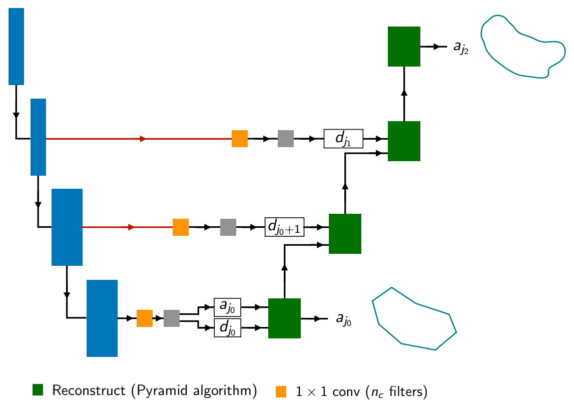
$$(F_M(h))_k := \begin{cases} \sum\limits_{|l| \leq M-1} |h_l|^2 - 1, & k = 0, \\ \sum\limits_{1-M+2k \leq l \leq M-1} h_{l-2k} h_l, & 1 \leq k \leq M-1, \\ -\sqrt{2} + \sum\limits_{|l| \leq M-1} h_l, & k = M. \end{cases}$$

If $F_M(h) = 0$, and h satisfies a certain non-degeneracy condition, then h is the low-pass filter of a scaling map φ generating an MRA of $L^2(\mathbb{R})$.

A full proof and detailed statement of the non-degeneracy condition can be found in Sec. 7.4 in the supplementary material. Heuristically, the first M equations encode the orthonormality of the scaling map. The last equation encodes the observation that the Fourier transform of the scaling map must be nonzero. The set of regular points in $F_M^{-1}(0)$ is a real-analytic $(M-2)$-dimensional submanifold of $\mathbb{R}^{2M-1}$. Hence, we can get as many degrees of freedom as desired, by choosing a sufficiently large order M. We now have an end-to-end way of learning wavelets for specific tasks, a wavelet network, by using the constraint F_M along with the CERM framework.

4. Experiments

In this section, we show some explicit examples of the wavelet networks with respect to contour prediction. In particular, we will utilize the derived wavelet filters of Sec. 3.2 to create a network for autocontouring. This task fits well with the setup of wavelets, as they naturally lend themselves to representing continuous objects such as curves.



a_{j_2}
d_{j_1}
d_{j_0+1}
a_{j_0}
d_{j_0}
a_{j_0}
🟩 Reconstruct (Pyramid algorithm)
🟧 1×1 conv (n_c filters)

Figure 3. A schematic picture of our network. The encoder consists of residual convolutional blocks depicted in blue. The first residual convolutional block uses n_f filters and is doubled after every other residual convolutional block. Attached to the encoder are fully connected layers to predict approximation and detail coefficients at the lowest resolution level j_0. The approximation and detail coefficients are supplied as input to the Pyramid Algorithm (the decoder) to predict a contour on high-resolution level. Each green block corresponds to an inverse discrete wavelet transform. Detail coefficients at higher levels are computed using skip-connections (arrows in red). We only predict detail coefficients up to level j_1. In this example, we have set $j_1 = j_0 + 2$ and $j_2 = j_0 + 3$. In reality, the decoder consists of two upsampling paths, one for each spatial component of the curve. We have only drawn one for notational convenience. During training, only the approximation coefficients at the highest resolution level are supervised.

4.1. Contour Prediction using MRA

In the context of contour prediction, wavelets can be used to represent the boundary of a region in an image using a simple closed curve. We consider two-dimensional gray-valued images $x \in \mathcal{X} := [0,1]^{n \times n}$. For each of our applications, we assume that these images contain an organ with a boundary. Formally speaking, this means that each image x contains a (uniquely identifiable) simply connected region $R(x)$, the organ, with boundary $\partial R(x)$. This boundary of the region, $\partial R(x)$, can be parameterized by a simple closed, piecewise C^2, curve $\gamma(x)$. We will develop a deep learning framework for computing such parameterizations by learning a multiresolution decomposition of $\gamma(x)$ using the methods developed in the previous sections.

4.1.1 Data

We have used public datasets from the Medical Decathlon Challenge [3]. The selected data consists of CT scans of the spleen of size 512×512 and T2-weighted MRI images of the prostate central gland, henceforth abbreviated as just the prostate, of size 320×320. The scans were cropped to size 224×224 and 192×192, respectively. Further-

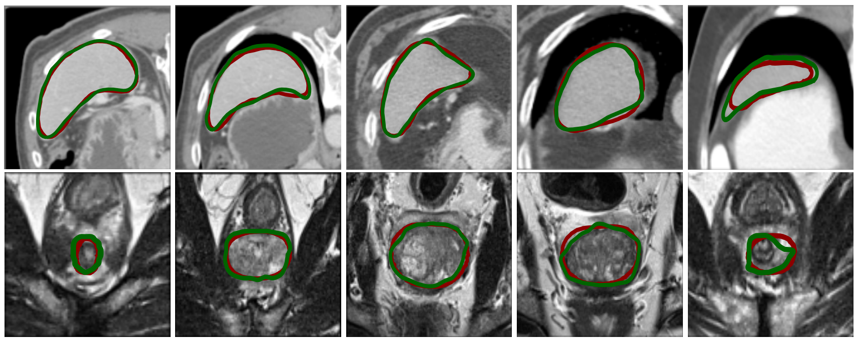
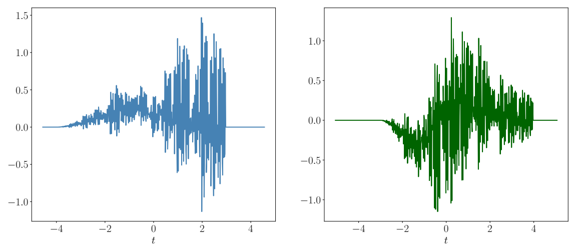
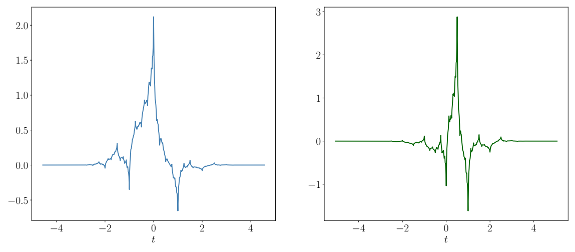


Figure 4. Examples of predictions on the test set for the spleen and prostate, depicted in the first and second row, respectively, for the best-performing wavelet models. The green curve corresponds to the ground truth, while the red curve is a prediction made by the wavelet network.



(a) Initial father and mother wavelet



(b) Task-optimized father and mother wavelet

Figure 5. Example of wavelets of order 5 learned during training of the spleen model: (a) The initial father (left) and mother (right) wavelets for the second spatial component. (b) The task-optimized father and mother wavelets after training for the second spatial component. We observe that the task-optimized wavelets are more simple and have "cleaned up" in a sense.

more, the images were resampled to the median sample spacing, which resulted in $(5.00\text{ mm}, 0.793\text{ mm}, 0.793\text{ mm})$ and $(3.6\text{ mm}, 0.625\text{ mm}, 0.625\text{ mm})$ spacings for the spleen and prostate, respectively. We refer the reader to Sec. 8.2 [16] for a detailed description of all the preprocessing steps and construction of ground truth curves. Examples of predictions on the test set, for both the spleen and prostate, are shown in Figure Fig. 4.

4.1.2 Model and Training

Model objective We parameterize the boundary $\partial R(x)$ by arc length, resulting in a curve $\gamma(x)$. The precise details of this parametrization are outlined in the supplementary information. The objective is to compute the relevant approximation coefficients of $\gamma(x)$. To this end, let $j_0, j_1, j_2 \in \mathbb{N}$ be resolution levels, where $j_0 \leq j_1 \leq j_2$. We will construct a convolutional neural network

$$G : \mathcal{X} \times \mathbb{R}^p \to \prod_{j=j_0}^{j_2} \mathbb{R}^{2^j} \times \mathbb{R}^{2^j} \times \prod_{j=j_0}^{j_1-1} \mathbb{R}^{2^j} \times \mathbb{R}^{2^j},$$

which predicts the wavelet decomposition of $\gamma(x)$. Here the subspaces $\mathbb{R}^{2^j}$ correspond to approximation and detail coefficients at level j, one for each spatial component. Furthermore, a subset of the parameters $\xi \in \mathbb{R}^p$ are constrained to be wavelet filters, one wavelet filter per spatial component, using Theorem 3.1. The map $G(\cdot, \xi)$ applied to an image x has output

$$\begin{aligned} G(x, \xi) = (&a_{j_0}(x, \xi), \ldots, a_{j_2}(x, \xi), \\ &d_{j_0}(x, \xi), \ldots, d_{j_1-1}(x, \xi)) . \end{aligned} \tag{5}$$

Here $a_j(x, \xi)$ and $d_j(x, \xi)$ are predictions for the approximation and detail coefficients of $\gamma(x)$ at level j, respectively. We only predict detail coefficients up to level j_1.

Architecture The network is a hybrid analog of the U-Net. It consists of a two-dimensional convolutional encoder, a bottleneck of fully connected layers, and a one-dimensional decoder. The encoder and decoder are connected through skip-connections. The approximation and detail coefficients at the lowest resolution level j_0 are predicted in the bottleneck. Afterward, the Pyramid Algorithm takes over to compute approximation coefficients at higher resolution levels (the decoder) using *learnable* wavelet filters. In Fig. 3, we provide a schematic overview of the network architecture.

Loss To measure the discrepancy between the ground truth and the predicted curve, we define

$$L(G(x,\xi),\tilde{a}(\gamma(x))) := \|[a_{j_2}]_1 - \tilde{a}_{j_2}([\gamma(x)]_1)\|_2 + \|[a_{j_2}]_2 - \tilde{a}_{j_2}([\gamma(x)]_2)\|_2. \tag{6}$$

The $\tilde{a}$ thus correspond to the ground-truth approximation coefficients, and the subscripts $1,2$ denote the spatial dimension. In other words, the loss corresponds to the component-wise L^2-error between the curves on resolution level j_2 with approximation coefficients $a_{j_2}(x,\xi)$ and $\tilde{a}_{j_2}(\gamma(x))$.

Optimization We use two different optimizers during training: one for the encoder and MLP network (unconstrained), and one for the decoder (constrained wavelet network). The reason for this is that the needed step sizes on the non-trivial submanifold may significantly differ from those on the unconstrained (flat) parameter space. For the training of the model, we use plain SGD for the first eight epochs for both the constrained and unconstrained parameters. During this period, the learning rate for the unconstrained parameters is linearly increased from 10^{-5} to $2 \cdot 10^{-4}$. The learning rate for the constrained parameters (wavelet filters) is linearly increased from 10^{-4} to 10^{-2}. After the initial warmup stage, we switch to the Adam optimizer for the unconstrained parameters. For both the constrained and unconstrained parameters, we use learning rate schedulers and decrease the learning rate by a factor of 0.85 if no significant improvements in the validation loss are observed during the last ten epochs. We train all models for 250 epochs using a batch size of 32 and use the last epoch for inference. The computations were performed in PyTorch on a Geforce RTX 2080 Ti. An average training session takes about two and a half hours for the wavelet network and a day for the nnUNet. The inference times for the wavelet network were 12.5ms and 14ms for the prostate and spleen, respectively, using a batch size of 32. The inference times for the nnUNet were 17.8ms and 23.4ms for the prostate and spleen, respectively, using a batch size of 32.

4.1.3 Results

We have evaluated the performance of both our wavelet networks for different orders and a state-of-the-art baseline 2d-nnUNet [13] on the unseen test data, see Table 1. For both the spleen and prostate, we observe that the best wavelet networks outperform the baseline. We note that the task-optimized wavelets differ significantly from the wavelets randomly initialized at the start of training. A comparison of an initial and task-optimized wavelet is depicted in Figure 5, see Sec. 8.4 for more examples.

Table 1. Mean and standard deviation (in parentheses) of the dice score on the unseen test sets. The number of parameters in our wavelet-based networks ranged from 5M to 10M parameters, while the nnUNet used approximately 40M and 18M parameters for the spleen and prostate, respectively.

Model	Dice Spleen	Dice Prostate
nnUNet	$0.914\ (1.74 \cdot 10^{-1})$	$0.896\ (1.27 \cdot 10^{-1})$
order 3	$0.911\ (7.38 \cdot 10^{-2})$	$0.929\ (4.66 \cdot 10^{-2})$
order 4	$0.911\ (7.85 \cdot 10^{-2})$	$\mathbf{0.935}\ (3.48 \cdot 10^{-2})$
order 5	$0.916\ (6.94 \cdot 10^{-2})$	$0.935\ (4.11 \cdot 10^{-2})$
order 6	$0.917\ (7.17 \cdot 10^{-2})$	$0.934\ (4.14 \cdot 10^{-2})$
order 7	$\mathbf{0.921}\ (6.91 \cdot 10^{-2})$	$0.928\ (4.03 \cdot 10^{-2})$
order 8	$0.919\ (6.64 \cdot 10^{-2})$	$0.934\ (3.62 \cdot 10^{-2})$

5. Conclusions

In this paper, we have introduced the CERM framework for creating task-driven wavelets. Although our aim is to find such wavelets, in principle the framework lends itself to imposing generic constraints on parametric models. The wavelet constraints can be formulated as a finite system of equations. Under mild smoothness and non-degeneracy conditions, the networks can be made to obey the constraints *exactly* throughout the entire training procedure by performing SGD on a curved space. We can then utilize these wavelet constraints in a convolutional network, where the filters of the network now parameterize wavelets themselves. We have applied these wavelet networks to the prediction of boundaries of simply connected regions in medical images, where they outperform strong baselines.

References

[1] P. Abry and P. Flandrin. On the initialization of the discrete wavelet transform algorithm. *IEEE Signal Processing Letters*, 1(2):32–34, 1994. 14

[2] Brandon Amos, Lei Xu, and J. Zico Kolter. Input convex neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, pages 146–155. PMLR, 2017. 2

[3] Michela Antonelli, Annika Reinke, Spyridon Bakas, Keyvan Farahani, Annette Kopp-Schneider, Bennett A Landman, Geert Litjens, Bjoern Menze, Olaf Ronneberger, Ronald M Summers, et al. The medical segmentation decathlon. *Nature communications*, 13(1):1–13, 2022. 6

[4] Randall Balestriero and Yann LeCun. Police: Provably optimal linear constraint enforcement for deep neural networks, 2022. 2

[5] D.P. Bertsekas and W. Rheinboldt. *Constrained Optimization and Lagrange Multiplier Methods*. Elsevier Science, 2014. 1

[6] Silvere Bonnabel. Stochastic gradient descent on riemannian manifolds. *IEEE Transactions on Automatic Control*, 58(9): 2217–2229, 2013. 2, 1, 5

[7] N. Boumal. *An Introduction to Optimization on Smooth Manifolds*. Cambridge University Press, 2023. 2

[8] Michael M Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, 2017. 2

[9] Michael M Bronstein, Joan Bruna, Taco Cohen, and Petar Veličković. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *arXiv preprint arXiv:2104.13478*, 2021. 2

[10] Albert Cohen, Ingrid Daubechies, and J-C Feauveau. Biorthogonal bases of compactly supported wavelets. *Communications on pure and applied mathematics*, 45(5):485–560, 1992. 5

[11] Priya Donti, David Rolnick, and J Zico Kolter. Dc3: A learning method for optimization with hard constraints. In *International Conference on Learning Representations*, 2021. 2

[12] M.W. Frazier. *An Introduction to Wavelets Through Linear Algebra*. Springer New York, 2001. 12

[13] Fabian Isensee, Paul F Jaeger, Simon AA Kohl, Jens Petersen, and Klaus H Maier-Hein. nnu-net: a self-configuring method for deep learning-based biomedical image segmentation. *Nature methods*, 18(2):203–211, 2021. 8

[14] Hiroyuki Kasai, Pratik Jawanpuria, and Bamdev Mishra. Riemannian adaptive stochastic gradient algorithms on matrix manifolds. In *International Conference on Machine Learning*, pages 3262–3271. PMLR, 2019. 2, 1, 5

[15] Diederik P. Kingma and Jimmy Lei Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*. International Conference on Learning Representations, ICLR, 2015. 1

[16] FirstName LastName. Task-driven wavelets using constrained empirical risk minimization, 2023. Supplied as supplemental material `supplementary-notes.pdf`. 3, 4, 5, 6, 7

[17] Yann LeCun, Yoshua Bengio, et al. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995, 1995. 1

[18] John M Lee. *Riemannian manifolds: an introduction to curvature*. Springer Science & Business Media, 2006. 6, 8

[19] John M Lee. Smooth manifolds. In *Introduction to smooth manifolds*. Springer, 2013. 2, 6, 8

[20] Benedict Leimkuhler, Timothée Pouchon, Tiffany Vlaar, and Amos Storkey. Constraint-based regularization of neural networks. *arXiv preprint arXiv:2006.10114*, 2020. 2

[21] Benedict Leimkuhler, Tiffany Vlaar, Timothée Pouchon, and Amos Storkey. Better training using weight-constrained stochastic dynamics. *arXiv preprint arXiv:2106.10704*, 2021. 2

[22] Qiufu Li, Linlin Shen, Sheng Guo, and Zhihui Lai. Wavelet integrated cnns for noise-robust image classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7245–7254, 2020. 2

[23] Cho-Chun Liu, Yaohui Liu, Zhengding Qiu, and Xiyu Du. On the initialization of the discrete wavelet transform algorithm. *1997 IEEE International Conference on Intelligent Processing Systems (Cat. No.97TH8335)*, 2:1220–1222 vol.2, 1997. 14

[24] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 1

[25] Ilya Loshchilov and Frank Hutter. Fixing weight decay regularization in adam, 2018. 1

[26] Stephane Mallat. *A Wavelet Tour of Signal Processing, Third Edition: The Sparse Way*. Academic Press, Inc., USA, 3rd edition, 2008. 2, 10

[27] Pablo Márquez-Neila, Mathieu Salzmann, and Pascal Fua. Imposing hard constraints on deep networks: Promises and limitations. *arXiv preprint arXiv:1706.02025*, 2017. 2

[28] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations*, 2018. 2

[29] L. Montefusco and L. Puccio. *Wavelets: Theory, Algorithms, and Applications*. Elsevier Science, 2014. 5, 11, 12, 14

[30] Deepak Pathak, Philipp Krahenbuhl, and Trevor Darrell. Constrained convolutional neural networks for weakly supervised segmentation. In *Proceedings of the IEEE international conference on computer vision*, pages 1796–1804, 2015. 2

[31] M.C. Pereyra and L.A. Ward. *Harmonic Analysis: From Fourier to Wavelets*. American Mathematical Society, 2012. 5, 11, 12

[32] Jack Richter-Powell, Yaron Lipman, and Ricky T. Q. Chen. Neural conservation laws: A divergence-free perspective, 2022. 2

[33] David W Romero, Erik J Bekkers, Jakub M Tomczak, and Mark Hoogendoorn. Wavelet networks: Scale equivariant learning from raw waveforms. *arXiv preprint arXiv:2006.05259*, 2020. 2

[34] Soumava Kumar Roy, Zakaria Mhammedi, and Mehrtash Harandi. Geometry aware constrained optimization techniques for deep learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4460–4469, 2018. 2, 1, 5

[35] Hiroyuki Sato, Hiroyuki Kasai, and Bamdev Mishra. Riemannian stochastic variance reduced gradient algorithm with retraction and vector transport. *SIAM Journal on Optimization*, 29(2):1444–1472, 2019. 2, 1, 5

[36] Vladimir Vapnik. Principles of risk minimization for learning theory. *Advances in neural information processing systems*, 4, 1991. 3

[37] Vladimir Vapnik. *The nature of statistical learning theory*. Springer science & business media, 1999. 3

[38] D.F. Walnut. *An Introduction to Wavelet Analysis*. Birkhäuser Boston, 2002. 12, 15

[39] Maurice Weiler, Patrick Forré, Erik Verlinde, and Max Welling. Coordinate independent convolutional networks–isometry and gauge equivariant convolutions on riemannian manifolds. *arXiv preprint arXiv:2106.06020*, 2021. 2

[40] Yang You, Jing Li, Sashank Reddi, Jonathan Hseu, Sanjiv Kumar, Srinadh Bhojanapalli, Xiaodan Song, James Demmel, Kurt Keutzer, and Cho-Jui Hsieh. Large batch optimization for deep learning: Training bert in 76 minutes. In *International Conference on Learning Representations*, 2020. 1

[41] Jason J Yu, Konstantinos G Derpanis, and Marcus A Brubaker. Wavelet flow: Fast training of high resolution normalizing

flows. *Advances in Neural Information Processing Systems*, 33:6184–6196, 2020. 2

[42] Hongyi Zhang, Sashank J. Reddi, and Suvrit Sra. Riemannian svrg: Fast stochastic optimization on riemannian manifolds. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2016. 2, 1, 5