Deep Sturm–Liouville: Learnable orthogonal basis functions parameterized by neural networks

David Vigouroux, Joseba Dalmau, Louis Béthune

DEEP STURM–LIOUVILLE: LEARNABLE ORTHOGONAL BASIS FUNCTIONS PARAMETERIZED BY NEURAL NETWORKS

A PREPRINT

David Vigouroux [*1], **Joseba Dalmau** [†1], **and Louis Béthune** [‡2]
[1]Institut de Recherche Technologique Saint Exupéry, Toulouse, France
[2]Institut de Recherche en Informatique de Toulouse, France

February 8, 2024

ABSTRACT

We introduce *Deep Sturm-Liouville* (DSL), a novel function approximator obtained by integrating the Sturm-Liouville theorem (SLT) into the deep learning framework. The Sturm-Liouville theorem deals with a class of eigenvalue problems having a wide range of applications in physics, which motivates us to explore its usage on machine learning tasks. The core idea of our work is to learn a vector field, crossing the input space $\Omega \subset \mathbb{R}^n$, such that the ML task along each of its field lines can be solved more easily due to the regularity of the problem on these field lines. A Sturm-Liouville Problem is solved along each field line to obtain orthogonal basis functions that, combined linearly, form the DSL function approximator. The vector field and the functions appearing in the SLT are parameterized by neural networks and they are learnt simultaneously. We also demonstrate that the DSL formulation appears naturally when solving a Rank-1 Parabolic Eigenvalue Problem. DSL is trained by stochastic gradient descent thanks to the implicit differentiation theorem, achieving comparable performances to neural networks on several multivariate datasets and the `MNIST` dataset.

1 Introduction

Neural networks have become indispensable in various applications, demonstrating their versatility by excelling in a wide range of tasks from image recognition to natural language processing. These practical results are also supported by theoretical works on the expressivity of neural networks. It has long been known that any function can be approximated by neural networks Hornik et al. [1989], Cybenko [1989] and recent works demonstrate exponential approximation accuracy Elbrächter et al. [2021].

Despite its remarkable achievements, deep learning presents notable drawbacks. Several works demonstrate that deep learning doesn't follow the same logic as humans: this difference is particularly highlighted by adversarial attack techniques Moosavi-Dezfooli et al. [2015] where a change in an image that is imperceptible to humans leads to a complete change in the predictions of the neural network; or when a domain shift appears and the neural network doesn't recognize an image despite being semantically similar. Even if whole fields of active research stem from these problems Rodriguez et al. [2023], Linsley et al. [2023], Szegedy et al. [2013], the underlying reasons are poorly understood: does the problem come from the optimization process, from the learning procedure, from the regularization of the networks, from the architectures themselves, or something else entirely? Each field of artificial intelligence tries to answer differently to these major issues.

These questions motivate us to explore new classes of function approximators where new regularizations can be defined. In this work, we introduce a new class of predictors by leveraging the power of the Sturm-Liouville theorem in high dimension, which allows us to learn orthogonal basis functions adapted to a machine learning task defined on an open domain $\Omega \subset \mathbb{R}^n$ with targets $Y \in \mathbb{R}^k$. We assume that $P_{XY} = \mathcal{P}(\Omega \times \mathbb{R}^k)$ is the joint distribution of data points

[*]david.vigouroux@irt-saintexupery.com
[†]joseba.dalmau@irt-saintexupery.com
[‡]louis.bethune@math.univ-toulouse.fr

and targets, and we consider a dataset $\mathcal{D} \sim P_{XY}^{\otimes n}$. We define the predictor F as a pair (θ, L) composed of a vector of basis functions $\boldsymbol{u^\theta} : \Omega \to \mathbb{R}^d$, d is a hyper-parameter of our method, parametrized by θ with a weight function $w^\theta : \Omega \to \mathbb{R}_*^+$, and a linear map $L : \mathbb{R}^d \to \mathbb{R}^k$:

$$\begin{aligned} &F\left(x, \theta, L\right) \stackrel{\text{def}}{=} L(\,\boldsymbol{u^\theta(x)}), \\ &\text{s.t} \int_\Omega w^\theta(x) u_i^\theta(x) u_j^\theta(x) dx = 0 \quad \forall i \neq j. \end{aligned} \tag{1}$$

Our goal is to minimize the empirical risk associated to a loss $\mathcal{L} : \mathbb{R}^k \times \mathbb{R}^k \to \mathbb{R}$, defined as

$$\min_{\theta, L} \frac{1}{n} \sum_{i=1} \mathcal{L}\big(y_i, F(x_i, \theta, L)\big), \tag{2}$$

by simultaneously learning the linear operator L and the orthogonal function basis $\boldsymbol{u^\theta(x)}$, in a *data-dependent fashion*. To avoid the curse of dimensionality incurred by fixed basis functions such as Fourier, polynomial or wavelet, the aim of this work is to create a flexible framework where the orthogonal basis functions are not fixed but they are learnt to adapt to a particular machine learning task.

The main idea behind our work is that there exists some vector field Feldman and Yeager [2018] crossing the input space where the task to be solved is simple on each field line. For example, in image classification task, we want to learn a field line which contains only images of cars in the domain of definition. In this line, the classification task will be simpler compared to a line which contains images of bananas, apples, cars and trains. During the training procedure, we don't explicitly train the vector field to be meaningful but we enforce implicitly a strong level of regularization to encourage the discovery of this kind of field lines. It is interesting to note that the DSL's formulation appears naturally when solving a Rank-1 Parabolic Eigenvalue Problem.

Our main contributions are:

- Introducing a new function approximator, called *Deep Sturm-Liouville* (DSL), built from a task-dependent orthogonal function basis in an open domain.
- Establishing a link between Deep Sturm-Liouville and a Rank-1 Parabolic Eigenvalue Problem.
- A training procedure using the implicit function theorem to train DSL with stochastic gradient descent.
- Implicitly enforcing the regularization of the function along each field lines by using only the d first elements of the basis functions that have the property of being more *regular* than higher order of the basis function.
- Explicitly regularizing the function via spectral regularization thanks to the local eigenvalues of the function.

First, we introduce the Sturm-Liouiville theorem (SLT) in its original 1D form. Then, we expose the Elliptic Eigenvalue Problem which extends SLT to the multidimensional case. This Elliptic Eigenvalue Problem being hard to solve due to its significant calculation time, we also present a related problem called Parabolic Eigenvalue Problem. In section 3, we introduce the Deep Sturm-Liouville method, which exploits a link between the Rank-1 Parabolic Eigenvalue Problem and the Sturm-Liouville problem to obtain a tractable solution for the high-dimensional case. We finish with an experimental evaluation of the DSL method.

1.1 Sturm-Liouville theorem

The Sturm-Liouville theorem Sturm and Liouville [1837] has a significant importance on the theory of eigenvalue problems for 1D ordinary differential equations (ODE). For instance, Sturm-Liouville theory (SLT) is employed in quantum mechanics to analyze the solutions of the Schrödinger equations Bender and Orszag [1978], in heat conduction problems Lützen [1984] or to compute vibrational modes Wang [1996]. Sturm-Liouville eigenvalue problems offer a systematic approach to discerning the characteristic frequencies and spatial patterns. This relationship between SLT and physics problems motivates us to explore the potential application of this theorem in machine learning. A wide range of 1D complete orthonormal function bases can be reinterpreted within this theory, common bases such as Fourier, Bessel or Chebyshev polynomials being are particular cases of this theorem.

The Sturm-Liouville theorem is formulated as an eigenvalue and eigenfunction problem satisfying the boundary conditions of an ODE:

Theorem 1.1 (Sturm-Liouville Theorem)**.** *For any given functions, $p, w : [a, b] \to \mathbb{R}_0^+$ and $q : [a, b] \to \mathbb{R}$ of classes $\mathcal{C}^1$, $\mathcal{C}^0$ and $\mathcal{C}^0$ respectively, and real numbers $\alpha_1, \alpha_2, \beta_1, \beta_2$, there exist a unique sequence $\{\lambda_i\}_{i\geq 1}$ (of eigenvalues) and associated eigenfunctions $y_i : [a, b] \to \mathbb{R}$ solving the ODE below, with the given boundary conditions:*

$$\begin{aligned} -\frac{d}{dt}\left[p(t)\frac{dy_i(t)}{dt}\right] + q(t)y_i(t) &= \lambda_i w(t)y_i(t), \\ \alpha_1 y_i(a) + \alpha_2 \frac{y_i}{dt}(a) &= 0 \qquad \alpha_1, \alpha_2 \textit{ not both 0}, \\ \beta_1 y_i(b) + \beta_2 \frac{y_i}{dt}(b) &= 0 \qquad \beta_1, \beta_2 \textit{ not both 0}. \end{aligned} \tag{3}$$

The sequence of eigenfunctions $\{y_i(t)\}$ forms an orthonormal basis in the Hilbert space $L^2([a, b])$ with the inner product weighted by w:

$$\int_a^b w(t)y_i(t)y_j(t)dt = \delta_{ij}.$$

The eigenvalues $\lambda_1, \lambda_2, ...$ are real and numbered so that $\lambda_1 < \lambda_2 < ...\lambda_n < ... \to \infty$. According to Egorov and Kondratiev [1996] Chapter 5-Theorem 19, the n^{th} basis function has exactly $n - 1$ zeros in the interval $]a, b[$, so that by tuning the number of basis functions being used, we can enforce the desired level of regularization in the function approximator.

For example, the Fourier basis is obtained for $p(t) = 1$, $w(t) = 1$, $q(t) = 0$, $a = 0$, $b = \pi$ and Dirichlet's conditions. The eigenfunctions are $\sin(nx)$ and eigenvalues are n^2.

Herein, we assume Dirichlet's boundary conditions:

$$y_i(a) = 0 \text{ and } y_i(b) = 0.$$

To compute the eigenvalues of the Strum-Liouville problem, a shooting method Stoer et al. [2002] will be used by performing a binary search between the lower and upper bounds of the eigenvalues Breuer and Gottlieb [1971] on an equivalent problem obtained by the Prüfer Substitution Prüfer [1926], Lebovitz [2019]. Details of this method can be found in Section 3.3.

For a one dimensional ML task, we can parameterize the functions p, q and w with neural networks. By solving the associated Sturm-Liouville Problem, we obtain the eigenfunctions $y_i(t)$ which form an orthogonal basis. A linear combination of the $y_i(t)$ can be used to predict the value on $x \in [a, b]$. The weights of p, q, w can be learnt to optimize (2). The main idea behind our work is to extend this procedure to the multidimensional case.

1.2 Elliptic Eigenvalue Problem

The Strum-Liouville theorem has its extension in dimension greater than one, more precisely on a open set Ω, thanks to the following Elliptic Eigenvalue Problem (EEP) Larsson [2003], Muthukumar [2014]:

Theorem 1.2 (Elliptic Eigenvalue Problem)**.** *For any continuous functions $A : \Omega \to \mathbb{R}^n \times \mathbb{R}^n$, symmetric, positive-definite, $q : \Omega \to \mathbb{R}$ and $w : \Omega \to \mathbb{R}_+^*$ of classes $\mathcal{C}^1$, $\mathcal{C}^0$ and $\mathcal{C}^0$ respectively, there exist a unique sequence of eigenvalues λ_i and associated eigenfunctions u_i satisfying:*

$$\begin{aligned} \nabla \cdot (A(x) \cdot \nabla u_i(x)) + q(x)u_i(x) &= -\lambda_i w(x)u_i(x). \\ \textit{with } u_i(x) &= 0 \qquad \forall x \in \partial\Omega. \\ \int_\Omega w(x)u_i(x)u_j(x)dx &= \delta_{ij} \qquad \forall i \neq j. \end{aligned} \tag{4}$$

This theorem could be useful to learn a basis of functions suited to a particular machine learning task in high dimension by optimizing (2) through the optimization of the functions A, q and w, typically surrogated by neural networks.

Solving these equations directly is quite challenging. First, even if recent works study the solutions of partial differential equations in high dimension Wu et al. [2023], efficiently solving this kind of partial differential equations (PDE) is

very costly. Secondly, solving the eigenvalue problem is very difficult even if numeric methods exists Larsson [2003]. Thirdly, without making any hypothesis on the form of the matrix A, whose size is the square of the size of input space, the matrix A can be too large for high dimension data.

To overcome these difficulties, we study a related problem where the matrix A is not full rank:

Definition 1.3. The Eigenvalue Problem (4) is called Parabolic when the matrix A is positive semi-definite.

Definition 1.4. The Parabolic Eigenvalue Problem is called Rank-1 when the matrix A is positive semi-definite and its rank is equal to 1.

In such case, the existence of eigenvalues is not guaranteed. However, its rank-1 structure will allow us to solve the Parabolic Eigenvalue Problem along a field line by using the 1D Sturm-Liouville theorem. This will allow us to combine the Sturm-Liouville theorem and deep neural networks, thus giving rise to Deep Sturm-Liouville, a means to compute orthogonal bases in high dimensions without the need to solve a high dimensional PDE.

2 Related work

In recent years, deep learning architectures have evolved quickly to obtain better approximating functions within high-dimensional spaces and exploit unique properties of complex data representations. ResNet He et al. [2015], a pioneer in convolutional neural networks (CNNs), remains influential for its simplicity and effectiveness in image classification. The MLP Mixer architecture Tolstikhin et al. [2021] has emerged as a concise alternative, utilizing multi-layer perceptrons to capture intricate patterns in data sequences without the need for extensive convolution layers. MixConv Tan and Le [2019] mixes up multiple kernel sizes in a single convolution. The Vision Transformer (ViT) Dosovitskiy et al. [2021] represents a breakthrough in image processing by relying on self-attention mechanisms Vaswani et al. [2017]. The aim of most new architectures is mainly to improve the performance of the neural networks. Other neural network architectures, such as Lipschitz neural networks Bethune et al. [2021] which controls the Lipschitz constant of the neural network, have specific properties to improve the robustness and the explainability of neural networks Serrurier et al. [2023]. The aim of our work is to build a function approximator with particular regularities to tackle limitations of deep learning previously outlined. DSL regularizes the function along a learnt field line where we expect that the task is simpler to solve which is not possible with usual architectures. The regularization in our method is done by controlling the number of times of the basis function changes sign. Contrary to the Gram-Schmidt process which could be used to define an orthogonality on the neighborhood of a sample, DSL enforce the orthogonality along a field line and by extension to the whole domain Ω 3.4.

Neural Ordinary Differential Equations introduced in Chen et al. [2018a] parameterize the derivative of the hidden state using a neural network. Unlike the more classical architectures formed of discrete sequences of hidden layers, Neural ODEs define the evolution of hidden states as solutions to ODEs. Neural ODEs have been successfully applied to normalizing flows which were introduced by Tabak and Vanden-Eijnden [2010] and popularised by Rezende and Mohamed [2015]. The first part of our algorithm has some similarities with Neural ODEs. Neural ODEs solve an autonomous ODE from an initial point to a fixed final time independent of each sample while DSL computes the final time for each sample such as the final state reaches the boundary of the domain Ω.

Continuous-time variable models are a popular topic in generative models such as normalizing flow Chen et al. [2018a], diffusion models Sohl-Dickstein et al. [2015], Flow matching Dao et al. [2023] and energy based models Hinton [2002]. Even if our work is not an explicit generative model, the first part of Deep Sturm-Liouville can be seen as a generative component that projects the sample distribution to the boundary of the domain. This property could be exploited in future work to couple generative models and classifiers such as in generative classifier Grathwohl et al. [2020], Jaini et al. [2023].

Some works solve partial differential equations with neural networks, e.g. for magnetic field estimation Khan et al. [2019], fluid simulations Kim et al. [2019], eigenvalue functions problems Kovacs et al. [2022] or PDEs similar to the Elliptic Eigenvalues Problems Marwah et al. [2023]. Inspired by complex natural phenomena which can be simulated by PDEs, our work takes the opposite point of view of these works by using Rank-1 Parabolic Eigenvalue Problems to propose a new function approximator.

3 Deep Sturm-Liouville Method

The purpose of this work is to introduce a new kind of orthogonal basis in an open domain $\Omega \subset \mathbb{R}^n$. Leveraging deep learning techniques and the Sturm-Liouville Theorem, Deep Sturm-Liouville is a new predictor which adapts orthogonal basis functions to a particular machine learning task. DSL can be used in any machine learning task that can be formulated as the optimization problem in (2).

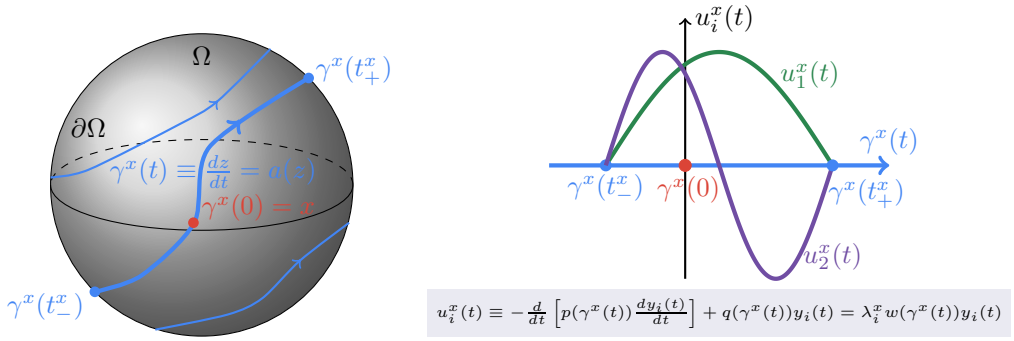

Figure 1: **Deep Sturm-Liouville.** For a given point x, the field line $\gamma^x(t)$ is defined by equation (5), it is such that $\gamma^x(0) = x$ and reaches the two points at the boundary of Ω at time t_-^x and t_+^x. On the field line $\gamma^x(t)$, the Sturm-Liouville Problem 6 is solved with the parameter functions $p(\gamma^x(t))$, $q(\gamma^x(t))$ and $w(\gamma^x(t))$ to obtain an orthogonal function basis that, combined linearly, form the DSL function approximator along the field line. The prediction at x is obtained by taking the value of this function at $t = 0$.

3.1 Deep Sturm-Liouville

First, we define the field line $\gamma^x(t)$ satisfying the following equation parameterized by the function $a : \Omega \to \mathbb{R}^n$:

$$\frac{dz}{dt} = a(z), \quad z(0) = x. \tag{5}$$

This equation is similar to the one in Neural Ordinary Differential Equations Chen et al. [2018b]. Nevertheless, equation (5) will be used differently to project the sample distribution to the boundaries of the domain Ω. Neural ODEs have a fixed final time while DSL has a different final time for each x. In Deep Sturm-Liouville, for any $x \in \Omega$, we impose that the field line $\gamma^x(t)$ passing through x shall cross the boundary $\partial\Omega$ in two unique points, i.e. there exist two unique times $t_-^x < 0$ and $t_+^x > 0$ such that $\gamma^x(t_-^x)$ and $\gamma^x(t_+^x) \in \partial\Omega$.

To obtain the uniqueness and the existence of t_-^x and t_+^x, we shall assume that (5) has an unique solution ($a(x)$ is Lipschitz), that there are no limit cycles and that $a(x)$ is nowhere tangent to $\partial\Omega$. The existence of limit cycles is a complex problem for which no general solution is known for $n > 2$; for n=2 the Bendixson–Dulac theorem describes sufficient conditions to have no limit cycles Burton and Burton [1983]. However, special cases exists where the absence of limit cycles has been demonstrated. For example Johnston [2015]:

- a is a strictly positive continuous function and Ω is convex,
- a is a gradient of a function with no singular point and the gradient is not vanishing anywhere in the domain[4].

The field line $\gamma^x(t)$ is defined by the equation (5) on the the interval $[t_-^x, t_+^x]$.

For a given x, thanks to the Sturm-Liouville theory applying along the field line $\gamma^x(t)$ with $p : \Omega \to \mathbb{R}_*^+$, $q : \Omega \to \mathbb{R}$ and $w : \Omega \to \mathbb{R}_*^+$, we can solve the eigenvalue problem of the following system to obtain the 1D orthogonal basis functions $u_i^x(t)$ and the eigenvalues λ_i^x:

$$-\frac{d}{dt}\left[p(\gamma^x(t))\frac{du_i(t)}{dt}\right] + q(\gamma^x(t))u_i(t) = \lambda_i^x w(\gamma^x(t))u_i(t),$$
$$u_i(t_-^x) = 0, \qquad u_i(t_+^x) = 0, \qquad \frac{du_i(t_-^x)}{dt} = 1. \tag{6}$$

[4]This equation has some similarities with energy-based models Hinton [2002].

Remark 3.1. The first three equations are defined up to a multiplying coefficient. The 4^{th} equation on the u_i^x derivative at t_- ensures the uniqueness of the solution. The value of this derivative is not important for building an orthogonal basis, but would matter if we wanted to build an orthonormal basis on Ω instead.

We define the function $u_i : \Omega \to \mathbb{R}$ by setting $u_i(x) = u_i^x(0)$. The $u_i(x)$ are well defined and form an orthogonal basis along the field line γ^x. Indeed, for all $x_1 \in \Omega$, if $x_2 = \gamma^{x_1}(s)$, we can show that $u_i^{x_1}(t) = u_i^{x_2}(t-s)$. As a consequence,

$$u_i(\gamma^{x_1}(s)) = u_i(x_2) = u^{x_2}(0) = u^{x_1}(s), \tag{7}$$

which holds for any $s \in [t_-^{x_1}, t_+^{x_1}]$. This implies that we can rewrite the equality

$$\int_{t_-^x}^{t_+^x} w(\gamma^x(t)) u_i^x(t) u_j^x(t) dt = 0$$

in terms of the functions u_i:

$$\int_{t_-^x}^{t_+^x} w(\gamma^x(t)) u_i(\gamma^x(t)) u_j(\gamma^x(t)) dt = 0.$$

In the Sturm-Liouville Theorem, the functions p, q and w depend only on the variable t. In Deep Strum-Liouville, the key idea is that p, q and w depend on the field line $\gamma^x(t)$. The purpose of this dependence is to couple equations (5) and (6) through the variable t. For two samples x^1 and x^2, which belong to two different field lines γ^{x^1} and γ^{x^2}, two different *local* orthogonal 1D basis functions are estimated. Consequently, the function approximator on the whole domain Ω is composed of 1D basis functions which are locally orthogonal.
To find the eigenvalues λ_i^x for a given x, we apply the shooting method along the field line $\gamma^x(t)$ by applying the Prüner substitution (9) (details can be found in the section 3.3). Finally, we can compute the prediction at a given x by solving the previous ordinary differential equations to obtain $u_i^x(0)$. The function approximator is defined thanks to the linear map $L : \mathbb{R}^d \to \mathbb{R}^k$, where d is the number of eigenvalues (a parameter of our method) and k is the dimension of the output of the predictor $F : \Omega \to \mathbb{R}^k$:

$$F^{\theta,L}(x) = L\left(\boldsymbol{u}^\theta(x)\right) \quad \text{with } \theta = [a, p, q, w].$$

Algorithm 1 Deep Sturm-Liouville - Prediction

1: Compute t_-^x and t_+^x with equation (5)
2: Find eigenvalues λ_i^x along the field line $\gamma^x(t)$ in (6) using a shooting method and the Prüner substitution (9)
3: Resolve equation (6) from t_-^x to compute $u_i(x)$
4: Compute the prediction at x: $F^{\theta,L}(x) = L\left(\boldsymbol{u}^\theta(x)\right)$

The optimization problem (2) can be rewritten by minimizing the parametric functions of the Sturm-Liouville problem:

$$\min_{L,\theta} \mathcal{L}(Y, F^{\theta,L}(X)).$$

In our experiments, the functions $a(x)$, $p(x)$, $q(x)$, $w(x)$ will typically be neural networks.

3.2 Regularizations

The main idea of Deep Strum-Liouville is that there exists some vector field crossing the input space Ω where the task to solve along each field lines can be solved more easily due to the regularity of the problem on these field lines. In DSL, we don't train explicitly $a(x)$ to encourage to share common features among all samples along γ^x. We rather enforce a strong regularization of the function along each field line γ^x that indirectly encourages $a(x)$ to find the desired vector field. This regularization is done both implicitly and explicitly.

Implicit regularization The implicit regularization is the most important regularization of this work, coming naturally from the mathematical formulation. This regularization is obtained by selecting the first few elements of the basis alone; which is a very natural regularization, since similarly to what happens in a Fourier basis, the first elements of the DSL basis *oscillate* less than higher elements of the basis. In the Sturm-Liouville framework, the *oscillation* is defined by the number of times where the basis functions change sign: the n^{th} base function changes sign exactly $n-1$ times. By selecting the d first elements of the basis function, DSL guaranties an implicit regularization along each field line γ^x.

Spectral regularization To avoid strong variations on the derivatives of the basis along the field line γ^x, the absolute value of the eigenvalues of the Sturm-Liouville Theorem, computed for each field line $\gamma^x(t)$, are added in the loss as a regularization term:

$$\min_{L,\theta} \mathcal{L}(Y, F^{\theta,L}(X)) + \alpha \cdot \mathbb{E}\left(\frac{1}{n}\sum_{i=0}^{d} |\lambda_i(X)|\right). \tag{8}$$

3.3 Computation of the eigenvalues

To compute the eigenvalues of the Strum-Liouville problem (3), we use a shooting method Stoer et al. [2002]. The aim of the shooting method is to optimize the eigenvalue λ such as the boundary condition $y(b) = 0$ is satisfied. Several techniques exist to solve this optimization problem such as gradient descent. In our work, we perform a binary search between the lower and upper bounds of the eigenvalues $[\lambda_n^-, \lambda_n^+]$ Breuer and Gottlieb [1971], see the appendix A for more details. We choose the binary search method to avoid tuning some hyper-parameters such as the learning rate if we had chosen the gradient descent.

Unfortunately, it is not possible to perform the binary search directly in the interval $[\lambda_i^-, \lambda_i^+]$. Indeed, the different intervals for each λ_i may overlap meaning that there might be multiple eigenvalues λ_j in the interval $[\lambda_i^-, \lambda_i^+]$. So the binary search is not guaranteed to find the correct eigenvalue λ_i.

To resolve this issue and to guarantee the monotonicity of the eigenvalue problem in the interval $[\lambda_i^-, \lambda_i^+]$, the Prüfer Substitution Prüfer [1926], Lebovitz [2019] is used to ensure that there is a unique solution for each eigenvalue. The equations (3) are substituted by the following equations thanks to the change of variables:

$$\begin{cases} u_i(t) = r(t)\sin(\theta(t)), \\ \dfrac{du_i(t)}{dt} = \dfrac{r(t)}{p(t)}\cos(\theta(t)). \end{cases}$$

If λ_n is the n^{th} eigenvalue given by the Sturm-Liouville theorem, the equations can be re-expressed as:

$$\begin{aligned} \frac{d\theta(t)}{dt} &= (\lambda_n w(t) + q(t))\sin^2(\theta(t)) + \cos^2(\theta(t))\frac{1}{p(t)}, \\ \frac{dr(t)}{dt} &= \left[\frac{1}{p(t)} - (\lambda_n w(t) + q(t))\right]\frac{r(t)}{2}\sin(2\theta(t)), \\ \theta(a) &= 0, \quad \theta(b) = n\pi. \end{aligned} \tag{9}$$

The boundary conditions are dependent on the parameter n, relating to the n^{th} eigenvalue, which is not the case with the initial formulation (3). This is what allows us to overcome the overlapping intervals problem.

Let $g(\lambda)$ be the function that maps each λ to the value $\theta(b) - n\pi$ obtained by solving the equation (9) with the initial boundary condition $\theta(a) = 0$, for a given λ. The function g is a strictly increasing function, so that a binary search can be applied between $[\lambda_i^-, \lambda_i^+]$ to find the λ such that $g(\lambda) = 0$.

The computation of λ_i^x is done in a similar way by using the shooting method along the field line γ^x with binary search and by applying the Prüner substitution on equation (6).

3.4 Gradient computation

The computation of the gradients of Deep Sturm-Liouville over the weights of the function a, p, q and w is not straightforward due to the estimation of eigenvalues the λ_i^x and the times t_-^x and t_+^x associated to each prediction. In fact, the computation through the shooting process and the stop conditions are not differentiable. To overcome this difficulty, we use the implicit differentiation theorem Krantz and Parks [2012].

We define the mapping $H^{\theta,\lambda,t_-^x,t_+^x} : \Omega \to \mathbb{R}^{d+2}$, capturing the optimal conditions of the problem:

$$H_k^{\theta,\lambda,t_-^x,t_+^x}(x) = \begin{cases} u_k^{\theta,\lambda}(\gamma^x(t_+^x)), & \text{if } 1 \le k \le d, \\ \min\limits_j \gamma_j^x(t_-^x), & \text{if } k = d+1, \\ \max\limits_j \gamma_j^x(t_+^x) - 1 & \text{if } k = d+2. \end{cases}$$

Remark 3.2. The two last conditions materialize the intersection of the field line γ^x with $\partial\Omega$ for the specific domain $\Omega =]0,1[^n$ that we use in our experiment. It should be defined differently for other convex domains such as the sphere.

Remark 3.3. $\forall x \in \Omega, \quad H^{\theta,\lambda,t_-^x,t_+^x}(x) = 0.$

Following the implicit differentiation theorem Krantz and Parks [2012]:

$$\begin{aligned}\nabla u_i^\theta(x) &= \nabla_\theta u_i^\theta(x) \\ &\quad - \nabla_{\lambda,t_-,t_+} u_i^\theta(x) \mathbb{J}^{-1}_{\lambda,t_-,t_+} H^{\theta,\lambda,t_-^x,t_+^x}(x) \mathbb{J}_\theta H^{\theta,\lambda,t_-^x,t_+^x}(x).\end{aligned}$$

3.5 Deep Sturm-Liouville is an orthogonal basis on Ω

Let us state the main theorem of Deep Sturm-Liouville:

Theorem 3.4. *The functions $u_i(x)$ form an orthogonal basis of functions on the open domain Ω:*

$$\int_\Omega v(x) u_i(x) u_j(x) dx = 0.$$

The intuition behind the proof of this theorem is simple. Along the field line γ^x the basis functions $u_i(x)$ are orthogonal. By applying a Fubini-like Nicolaescu [2011] result to the integral over the whole domain Ω, we rewrite the integral over Ω as a double integral: over the points in the boundary $\partial\Omega$ of the form $\gamma^x(t_-^x)$, and along the field line γ^x, thus obtaining the orthogonality over the whole domain Ω. Details of the proof can be found in appendix B.

3.6 Link between Deep Sturm-Liouville and Rank-1 Parabolic Eigenvalue Problems

Before stating the main result of this section, we define a sub-class of Deep Sturm-Liouville problems.

Definition 3.5. Deep Sturm-Liouville Problem (6) is uniform if all eigenvalues are independent of x.

Theorem 3.6. *The Uniform Deep Sturm-Liouville Problem can be rewritten as a Dirichelt Rank-1 Parabolic Eigenvalue Problem when assuming $a_i(x) > 0$.*

$$\begin{aligned}&\nabla \cdot (a(x) a^t(x) \cdot \nabla u_i(x)) + q(x) u_i(x) = -\lambda_i w(x) u_i(x). \\ \Leftrightarrow &\begin{cases} \frac{\partial}{\partial t}\left(p(x)\frac{\partial v_i(t,\boldsymbol{y})}{\partial t}\right) + \tilde{q}(x) v_i(t,\boldsymbol{y}) = -\lambda_i \tilde{w}(x) v_i(t,\boldsymbol{y}), \\ \frac{dx}{dt} = \tilde{a}(x). \end{cases}\end{aligned}$$

The idea is to define a direction $a(x)$ where the PDE can be solved thanks to ODEs. This can be achieved by observing that any positive semi-definite rank-1 matrix $A(x)$ can be written as $A(x) = a(x)a^t(x)$ and by applying a change of variables where all gradients of this new system of variables are orthogonal to $a(x)$. Details of the proof can be found in appendix C.

Deep Sturm-Liouville can be seen as a relaxed version of the Rank-1 Parabolic Eigenvalues Problem when no *global* eigenvalues exists.

4 Experiments

To evaluate our work, Deep Sturm-Liouville has been trained on three multivariate datasets: Adult Becker and Kohavi [1996], Dry Bean UCI Machine Learning Repository and Bank Marketing Frank [2010], as well as the MNIST image dataset LeCun and Cortes [2010].

4.1 Experimental Setup

Ordinary differential equation solvers. Deep Sturm-Liouville is implemented on `jax` Bradbury et al. [2018] and uses `diffrax` Kidger [2021] to solve the ODEs involved. The solver `dopri8` Prince and Dormand [1981] is used with a relative tolerance of $1\mathrm{e}{-6}$ and an absolute tolerance of $1\mathrm{e}{-6}$.

Computation of the times t_- and t_+. At the time of writing of this paper, even if the diffrax library supports *event* stopping conditions, diffrax does not provide a way of approximating t_- and t_+ within a given tolerance as in Chen et al. [2020]. To obtain a good approximation of the times t_- and t_+ to reach the boundary of the domain Ω, we perform a binary search. There is no need for this binary search procedure to be differentiable thanks to the implicit differentiation theorem.

4.2 Computation of eigenvalues

To solve the eigenvalue problem, the values of q, p and w are computed along the field line γ to obtain a spline which is dependent only on t (avoiding numerous calls to the neural networks during the shooting phase). In these experiments, a piecewise linear function of 2000 parts is used to approximate these functions along the field line.
The binary search is done with a tolerance of $1\mathrm{e}{-4}$ for the tabular experiments and $1\mathrm{e}{-8}$ for the image dataset. For all experiments, the number of eigenfunctions was fixed to 10.

Eigenvalues regularization. The value of the regularization coefficient of the equation (8) was fixed to $1\mathrm{e}{-4}$.

For the MNIST dataset, to ensure that the eigenvalues are not too large at initialization, the output domain of each of the functions a, p, q and w was bounded. The choice of the appropriate bounds for each function is guided by the lower and upper bounds in the equations (10). To limit the eigenvalues to belong to the interval $\approx [-100, 100\ n^2\pi^2]$, the following constraints were implemented:

	$a(x)$	$q(x)$	$\frac{1}{p(x)}$	$w(x)$
DOMAIN	$(0.01, 1)$	$(-10, 10)$	$(1, 10)$	$(0.1, 10)$

Remark 4.1. The eigenvalues bounds were taken experimentally to let enough range to the variation of the eigenvalues while maintaining a reasonable computation times.

These constraints are enforced by using sigmoid activations at the end of the model for a, p and w and a hyperbolic tangent activation for q.

Architectures, losses and optimizers. The optimizers of `optax` library are adam Kingma and Ba [2014] with learning rate $2\mathrm{e}{-3}$ for the tabular datasets and fromage (lr=$1\mathrm{e}{-2}$) Bernstein et al. [2020] for the MNIST dataset. The losses are the hinge loss for the tabular datatsets and the categorical cross-entropy for the MNIST dataset.

For the tabular datasets, the functions $q(x)$, $\frac{1}{p(x)}$ and $w(x)$ are defined by a MLP with the features [128, 64, 32, 1] and leaky relu activations. The function $a(x)$ is a MLP [128, 64, 32, k] with tanh activations, where k is the dimension of the input size of the data.

For the MNIST dataset, the functions $q(x)$, $\frac{1}{p(x)}$ and $w(x)$ are defined by a convolutional neural network with [32,64,128] features and kernel (3,3), the features of the MLP are [32, 32, 16, 1] with tanh activations. The function $a(x)$ is an auto-encoder with [32,64] convolutions features and kernel (3,3) with 32 features and tanh activations.

Domain and Data normalization. As defined in the remark (3.2), the domain Ω is defined to be $]0, 1[^n$. Data are normalized so that they belong to $[0.25, 0.75]^n$, thus ensuring that they are included in Ω, and that no example is too close to the boundary $\partial\Omega$, where the basis functions equal to 0 due to the Dirichlet conditions.

4.3 Results

To verify that DSL learns a different local basis for each example x, the local basis along the field line $\gamma^x(t)$ is analyzed for several different samples of the Dry Bean Dataset. As observed in figure 2, the local basis is different for the each of the examples represented, illustrating the local expressiveness of Deep Sturm-Liouville. Additionally, as expected by the Sturm-Liouville theory, the boundaries satisfy the Dirichlet conditions and the i^{th} base function crosses the x-axis exactly $i - 1$ times.

To demonstrate that Deep Sturm-Liouville can reach comparable performance to a Neural Network, DSL and NN were trained on several classification tasks. For this experiment, Neural Networks have similar architectures. For tabular dataset, the NNs are MLP with [128, 64, 32, n_o] features where n_o is the number of output. Table 1 demonstrates that DSL achieves comparable result than NN with only 10 eigenfunctions.

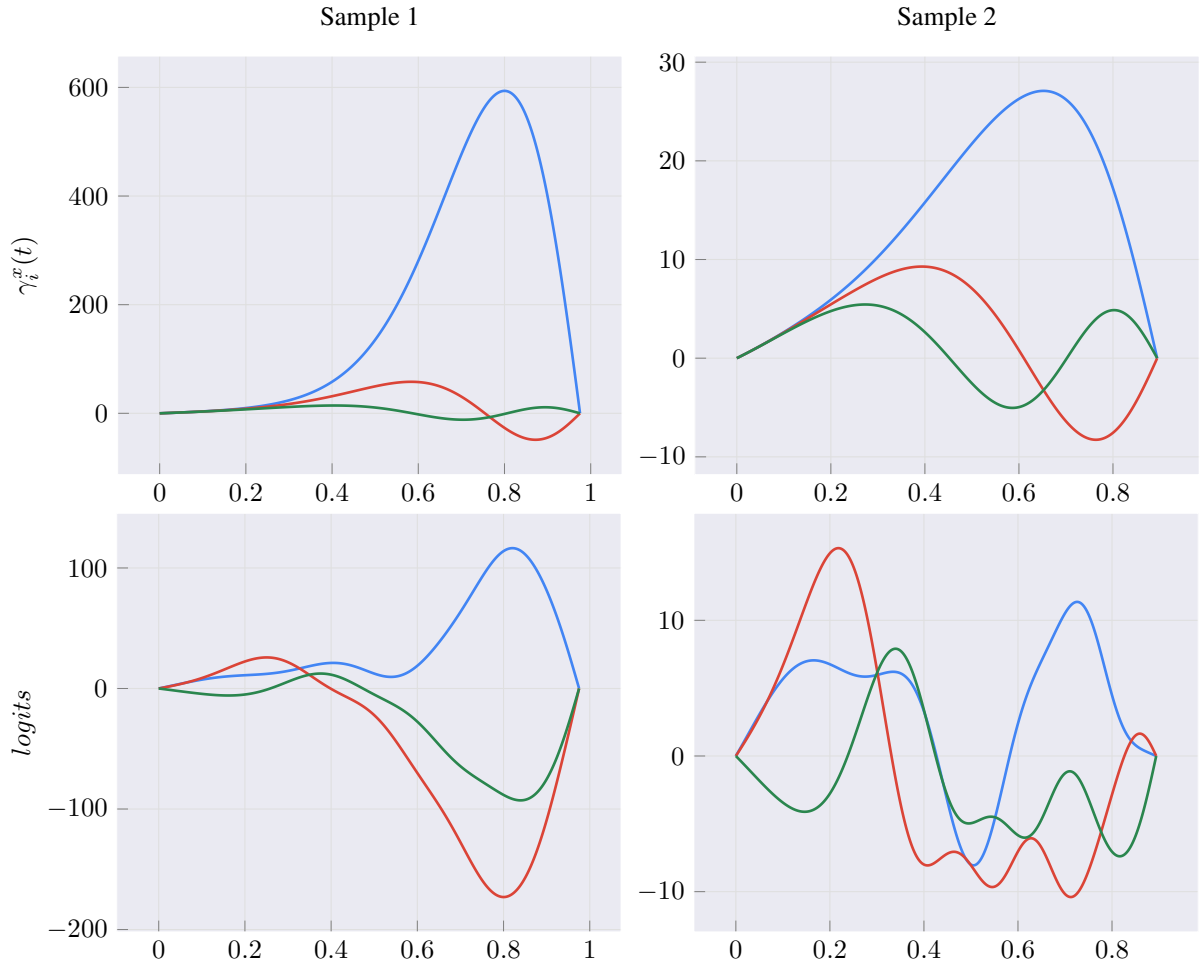

Figure 2: **Eigenfunctions on Dry Bean dataset.** For two samples, on the top, the first three eigenfunctions and on the bottom the first three logits. The x-axis represents the time t of the field line $\gamma^x(t)$.

DATA SET	DEEP STURM-LIOUVILLE	NN
ADULT	84.28%	84.06%
DRY BEAN	91.14%	91.45%
BANK MARKETING	83.10%	83.77%
MNIST	97.93%	99.83%

Table 1: **Evaluation.** Classification accuracies for Deep Sturm-Liouville and Neural Networks

Finally, for the Dry Bean dataset, the impact of the number of eigenfunctions on the performance of the classifier was analyzed. As illustrated in 3, 10 eigenfunctions are sufficient to obtain comparable performance to an MLP.

5 Limitations

Despite the promising results, Deep Sturm-Liouville suffers from several flaws. **Scalability.** Even if Deep Sturm-Liouville is scalable to high dimension, the gradient computation can be expensive due to the form of the problem to solve . It is not particularly due to the implicit differentiation theorem because we observe that the computation of the gradient on the weights of neural networks are the same order of magnitude than the computation of the jacobian on the eigenvalues and times to boundaries. Prediction computation can also be expensive. Even if the binary search

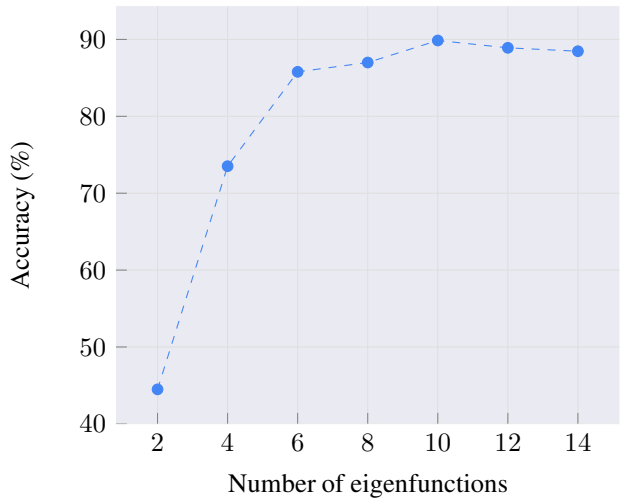

Figure 3: **Impact of number of eigenfunctions on Dry Bean dataset.** Validation accuracy as function of eigenfunctions basis size. Trained with fromage and cross-entropy.

itself is quick, the approximation of p, q, w along the field line γ can be costly despite their smoothness[5].
Stability. The estimation of the gradient of the ODE could be noisy if the solver is not precise enough. During training, in rare configurations, the ODE solver takes too much time to solve the ODE with a good precision and training fails.

6 Conclusion

A mathematical formulation has been developed to introduce the Sturm-Liouville Theory in the deep learning framework. We demonstrate the link between the Deep Strum-Liouville formula and the Rank-1 Parabolic Eigenvalues problem. A trainable procedure based on implicit differentiation was implemented, successfully achieving comparable results to those of neural networks on several datasets and `MNIST`. We hope that our work paves the way for novel avenues in function regularization.

7 Broader Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

References

Barry Becker and Ronny Kohavi. Adult. UCI Machine Learning Repository, 1996. DOI: https://doi.org/10.24432/C5XW20.

C. M. Bender and S. A. Orszag. *Advanced Mathematical Methods for Scientists and Engineers*. McGraw-Hill, 1978.

Jeremy Bernstein, Arash Vahdat, Yisong Yue, and Ming-Yu Liu. On the distance between two neural networks and the stability of learning. In *Neural Information Processing Systems*, 2020.

Louis Bethune, Thibaut Boissin, Mathieu Serrurier, Franck Mamalet, Corentin Friedrich, and Alberto Gonzalez-Sanz. Pay attention to your loss : understanding misconceptions about lipschitz neural networks. In *Neural Information Processing Systems*, 2021. URL `https://api.semanticscholar.org/CorpusID:249375220`.

James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL `http://github.com/google/jax`.

[5]Smoother functions allow for bigger stepsize in numerical integration.

Shlomo Breuer and David Gottlieb. Upper and lower bounds on eigenvalues of sturm-liouville systems. *Journal of Mathematical Analysis and Applications*, 36(3):465–476, 1971. ISSN 0022-247X. doi: https://doi.org/10.1016/0022-247X(71)90032-1. URL https://www.sciencedirect.com/science/article/pii/0022247X71900321.

T.A. Burton and T.A. Burton. *Volterra Integral and Differential Equations*. Mathematics in science and engineering. Academic Press, 1983. ISBN 9780121473808. URL https://books.google.fr/books?id=DRvAlAEACAAJ.

Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, page 6572–6583, Red Hook, NY, USA, 2018a. Curran Associates Inc.

Ricky T. Q. Chen, Brandon Amos, and Maximilian Nickel. Learning neural event functions for ordinary differential equations. *ArXiv*, abs/2011.03902, 2020. URL https://api.semanticscholar.org/CorpusID:226282371.

Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018b. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/69386f6bb1dfed68692a24c8686939b9-Paper.pdf.

G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2(4):303–314, Dec 1989. ISSN 1435-568X. doi: 10.1007/BF02551274. URL https://doi.org/10.1007/BF02551274.

Quan Dao, Hao Phung, Binh Nguyen, and Anh Tran. Flow matching in latent space. *arXiv preprint arXiv:2307.08698*, 2023.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=YicbFdNTTy.

Yuri Egorov and Vladimir Kondratiev. *Spectral Properties of Elliptic Operators*, pages 133–151. Birkhäuser Basel, Basel, 1996. ISBN 978-3-0348-9029-8. doi: 10.1007/978-3-0348-9029-8_4. URL https://doi.org/10.1007/978-3-0348-9029-8_4.

Dennis Elbrächter, Dmytro Perekrestenko, Philipp Grohs, and Helmut Bölcskei. Deep neural network approximation theory. *IEEE Transactions on Information Theory, invited feature paper*, 67(5), May 2021. URL http://www.nari.ee.ethz.ch/pubs/p/deep-it-2019.

Rechnitzer Feldman and Yeager. *Vector Calculus*. 2018. URL https://math.libretexts.org/Bookshelves/Calculus/CLP-4_Ve

Andrew Frank. Uci machine learning repository. *http://archive. ics. uci. edu/ml*, 2010.

Will Grathwohl, Kuan-Chieh Wang, Jörn-Henrik Jacobsen, David Duvenaud, Mohammad Norouzi, and Kevin Swersky. Your classifier is secretly an energy based model and you should treat it like one. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=HkxzxONtDB.

Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2015. URL https://api.semanticscholar.org/CorpusID:206594692.

Geoffrey E. Hinton. Training Products of Experts by Minimizing Contrastive Divergence. *Neural Computation*, 14(8):1771–1800, 08 2002. ISSN 0899-7667. doi: 10.1162/089976602760128018. URL https://doi.org/10.1162/089976602760128018.

Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989. ISSN 0893-6080. doi: https://doi.org/10.1016/0893-6080(89)90020-8. URL https://www.sciencedirect.com/science/article/pii/0893608089900208.

Priyank Jaini, Kevin Clark, and Robert Geirhos. Intriguing properties of generative classifiers. *arXiv preprint arXiv:2309.16779*, 2023.

M. D. Johnston. Limit cycles, van der pol oscillator, poincar´e-bendixson theorem, 2015. URL https://johnstonmd.files.wordpress.com/2015/03/math415-w10.pdf.

Arbaaz Khan, Vahid Ghorbanian, and David Lowther. Deep learning for magnetic field estimation. *IEEE Transactions on Magnetics*, 55(6):1–4, 2019. doi: 10.1109/TMAG.2019.2899304.

Patrick Kidger. *On Neural Differential Equations*. PhD thesis, University of Oxford, 2021.

Byungsoo Kim, Vinicius C. Azevedo, Nils Thuerey, Theodore Kim, Markus Gross, and Barbara Solenthaler. Deep Fluids: A Generative Network for Parameterized Fluid Simulations. *Computer Graphics Forum (Proc. Eurographics)*, 38(2):59–70, 2019.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Alexander Kovacs, Lukas Exl, Alexander Kornell, Johann Fischbacher, Markus Hovorka, Markus Gusenbauer, Leoni Breth, Harald Oezelt, Masao Yano, Noritsugu Sakuma, Akihito Kinoshita, Tetsuya Shoji, Akira Kato, and Thomas Schrefl. Conditional physics informed neural networks. *Communications in Nonlinear Science and Numerical Simulation*, 104:106041, 2022. ISSN 1007-5704. doi: https://doi.org/10.1016/j.cnsns.2021.106041. URL https://www.sciencedirect.com/science/article/pii/S1007570421003531.

S. G. Krantz and H. R. Parks. The implicit function theorem: history, theory, and applications. *Springer Science and Business Media*, 2012. URL https://link.springer.com/book/10.1007/978-1-4612-0059-8#book-header.

Thomée Larsson. *The Elliptic Eigenvalue Problem*, pages 77–94. Springer Berlin Heidelberg, Berlin, Heidelberg, 2003. ISBN 978-3-540-88706-5. doi: 10.1007/978-3-540-88706-5_6. URL https://doi.org/10.1007/978-3-540-88706-5_6.

N Lebovitz. Oscillation theory and the spectra of eigenvalues. *Ordinary Differential Equations, http://people. cs. uchicago. edu/lebovitz/odes. html*, 2019. URL https://people.cs.uchicago.edu/~lebovitz/Eodesbook/sl.pdf.

Yann LeCun and Corinna Cortes. MNIST handwritten digit database. http://yann.lecun.com/exdb/mnist/, 2010. URL http://yann.lecun.com/exdb/mnist/.

Drew Linsley, Pinyuan Feng, Thibaut Boissin, Alekh Karkada Ashok, Thomas Fel, Stephanie Olaiya, and Thomas Serre. Adversarial alignment: Breaking the trade-off between the strength of an attack and its relevance to human perception, 2023.

Jesper Lützen. Sturm and liouville's work on ordinary linear differential equations. the emergence of sturm-liouville theory. *Archive for History of Exact Sciences*, 29(4):309–376, Dec 1984. ISSN 1432-0657. doi: 10.1007/BF00348405. URL https://doi.org/10.1007/BF00348405.

Tanya Marwah, Zachary Chase Lipton, Jianfeng Lu, and Andrej Risteski. Neural network approximations of pdes beyond linearity: A representational perspective. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 24139–24172. PMLR, 2023. URL https://proceedings.mlr.press/v202/marwah23a.html.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: A simple and accurate method to fool deep neural networks. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2574–2582, 2015. URL https://api.semanticscholar.org/CorpusID:12387176.

T. Muthukumar. Second order elliptic pde. *Course notes*, 2014. URL http://home.iitk.ac.in/~tmk/courses/minicourse/FAPDE/BasicPDE.pdf.

Liviu I Nicolaescu. The coarea formula. In *seminar notes. Citeseer*, 2011.

P. J Prince and J. R. Dormand. High order embedded Runge–Kutta formulae. *J. Comp. Appl. Math*, 7(1):67–75, 1981.

H. Prüfer. Neue herleitung der sturm-lionvilleschen reihenentwicklung stetiger funktionen. *Mathematische Annalen*, 95:499–518, 1926. URL http://eudml.org/doc/159142.

Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, page 1530–1538. JMLR.org, 2015.

Ivan Felipe Rodriguez, Drew Linsley, Jay Gopal, Thomas Fel, Michael J Acaro, Saloni Sharma, Margaret Livingstone, and Thomas Serre. Harmonizing the visual strategies of image-computable models with humans yields more performant and interpretable models of primate visual system function. *Journal of Vision*, 23(9):5768–5768, 2023.

Mathieu Serrurier, Franck Mamalet, Thomas Fel, Louis Béthune, and Thibaut Boissin. On the explainable properties of 1-lipschitz neural networks: An optimal transport perspective. 2023.

Jascha Narain Sohl-Dickstein, Eric A. Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. *ArXiv*, abs/1503.03585, 2015. URL https://api.semanticscholar.org/CorpusID:14888175.

J. Stoer, R. Bartels, W. Gautschi, R. Bulirsch, and C. Witzgall. *Introduction to Numerical Analysis*. Texts in Applied Mathematics. Springer New York, 2002. ISBN 9780387954523. URL https://books.google.fr/books?id=1oDXWLb9qEkC.

C. Sturm and J. Liouville. Extrait d'un mémoire sur le développement des fonctions en séries dont les différents termes sont assujettis à satisfaire à une même équation différentielle linéaire, contenant un paramètre variable. *J. Math. Pures Appl. 2*, page 220–223, 1837. URL http://www.numdam.org/item/JMPA_1837_1_2__220_0.pdf.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

Esteban G. Tabak and Eric Vanden-Eijnden. Density estimation by dual ascent of the log-likelihood. *Communications in Mathematical Sciences*, 8:217–233, 2010. URL https://api.semanticscholar.org/CorpusID:17933194.

Mingxing Tan and Quoc Le. Mixconv: Mixed depthwise convolutional kernels. In Kirill Sidorov and Yulia Hicks, editors, *Proceedings of the British Machine Vision Conference (BMVC)*, pages 116.1–116.13. BMVA Press, September 2019. doi: 10.5244/C.33.116. URL https://dx.doi.org/10.5244/C.33.116.

Ilya O Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, Mario Lucic, and Alexey Dosovitskiy. Mlp-mixer: An all-mlp architecture for vision. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 24261–24272. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/cba0a4ee5ccd02fda0fe3f9a3e7b89fe-Paper.pdf.

UCI Machine Learning Repository. Dry Bean Dataset. UCI Machine Learning Repository, 2020. DOI: https://doi.org/10.24432/C50S4B.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Quanfeng Wang. Sturm-liouville equation for free vibration of a tube-in-tube tall building. *Journal of Sound and Vibration*, 191(3):349–355, 1996. ISSN 0022-460X. doi: https://doi.org/10.1006/jsvi.1996.0126. URL https://www.sciencedirect.com/science/article/pii/S0022460X96901261.

Haixu Wu, Tengge Hu, Huakun Luo, Jianmin Wang, and Mingsheng Long. Solving high-dimensional pdes with latent spectral models. In *International Conference on Machine Learning*, 2023.

A Eigenvalues lower and upper bounds

The lower and upper bounds of Sturm-Liouville Problem are:

$$\begin{aligned}\lambda_n^+ &\stackrel{\text{def}}{=} \frac{n^2\pi^2}{\min_t w(t)p(t)\cdot\left(\int_b^a \frac{1}{p(t)}dt\right)^2} + \max_t \frac{-q(t)}{w(t)}.\\ \lambda_n^- &\stackrel{\text{def}}{=} \frac{n^2\pi^2}{\max_t w(t)p(t)\cdot\left(\int_b^a \frac{1}{p(t)}dt\right)^2} + \min_t \frac{-q(t)}{w(t)}.\end{aligned} \tag{10}$$

B Proof Theorem 3.4

$u_i(x)$ form an orthogonal basis function on a open Ω:

$$\int_\Omega v(x)u_i(x)u_j(x)dx = 0.$$

Proof. By Sturm-Liouville Theory, we have for all $x \in \Omega$ and for all $i \neq j$:

$$\int_{t_-^x}^{t_+^x} w(\gamma^x(t))u_i^x(t)u_j^x(t)dt = 0.$$

We will reformulate the integral over the field line γ^x and by applying the line integrals change of variable and by (7):

$$\int_{\gamma^x} \frac{w(z)}{\|a(z)\|}u_i(z)u_j(z)d\mathcal{H}_\Omega^1(z) = 0.$$

We define the manifold $\partial\Omega_- \subset \mathbb{R}^n$ which is $(n-1)$-rectifiable:

$$\partial\Omega_- = \{\gamma^x(t_-^x) \quad \forall x \in \Omega\}.$$

By integrating over Ω_- we get:

$$\int_{\partial\Omega_-}\int_{\gamma^v} \frac{w(z)}{\|a(z)\|}u_i(z)u_j(z)d\mathcal{H}_\Omega^1(z)d\mathcal{H}_{\partial\Omega_-}^{n-1}(v) = 0.$$

We define:

$$\begin{aligned}P &: \Omega \to \partial\Omega_-\\ P(x) &= \gamma^x(t_-^x).\end{aligned}$$

Since P is Lipschitz we apply the co-area formula Nicolaescu [2011]:

$$\int_\Omega \frac{w(x)}{\|a(x)\|}u_i(x)u_j(x)det|\mathbb{J}_P(x)|dx = 0.$$

We let:

$$v(x) = \frac{w(x)}{\|a(x)\|}\, det|\mathbb{J}_P(x)|.$$

Then:

$$\int_\Omega v(x)u_i(x)u_j(x)dx = 0.$$

$u_i(x)$ form an orthogonal basis functions under the weight function $v(x)$ on the domain Ω. □

C Proof Theorem 3.6

Uniform Deep Sturm-Liouville can be rewritten to a Dirichelt Rank-1 Parabolic Eigenvalues Problem when assuming $a_i(x) > 0$.

Proof. From the equation (4), we will take the special case where:

$$A(x) = a(x)a^t(x).$$

Then we will develop the first component of the equation:

$$\nabla \cdot (a(x)a^t(x) \cdot \nabla u_i(x)) + q(x)u_i(x) = -\lambda_i w(x)u_i(x). \tag{11}$$

We will introduce the following change of variable $(t, \boldsymbol{y})$ such that:

$$\begin{aligned} v_i(t, \boldsymbol{y}) &= u_i(x) \\ \nabla_x t &= a(x) \\ \nabla_x y_k &= a_{n_k}(x) \quad \forall k \in [1, n-1] \end{aligned}$$

With $a_{n_k}(x)$ are built using Gram-Schmidt procedure to obtain an orthogonal base $a_{n_k}(x) \perp a(x)$.

Then, we can expand $\nabla_x \cdot (a(x)a^t(x)\nabla_x u_i(x))$ as

$$\begin{aligned} &= \nabla_x \cdot \left(a(x)a^t(x) \left(a(x)\frac{\partial v_i(t, \boldsymbol{y})}{\partial t} + \sum_{k=1}^{n-1} \left(a_{n_k}(x)\frac{\partial v_i(t, \boldsymbol{y})}{\partial y_k} \right) \right) \right) \\ &= \nabla_x \cdot \left(\|a(x)\|^2 a(x)\frac{\partial v_i(t, \boldsymbol{y})}{\partial t} \right) \\ &= \|a(x)\|^2 a(x) \left(\frac{\partial^2 v_i(t, \boldsymbol{y})}{\partial t^2} a(x) + \sum_{k=1}^{n-1} \left(a_{n_k}(x)\frac{\partial^2 v_i(t, \boldsymbol{y})}{\partial y_k \partial t} \right) \right) \\ &\quad + \|a(x)\|^2 \frac{\partial v_i(t, \boldsymbol{y})}{\partial t} \nabla_x \cdot a(x) \\ &\quad + 2\left(\mathbb{J}_x a(x) \cdot a(x)\right) \cdot a(x)\frac{\partial v_i(t, \boldsymbol{y})}{\partial t} \\ &= \|a(x)\|^4 \frac{\partial^2 v_i(t, \boldsymbol{y})}{\partial t^2} \\ &\quad + \left(2\left(\mathbb{J}_x a(x) \cdot a(x)\right) \cdot a(x) + \|a(x)\|^2 \nabla_x \cdot a(x) \right) \frac{\partial v_i(t, \boldsymbol{y})}{\partial t} \end{aligned}$$

We let:

$$b(x) = 2\left(\mathbb{J}_x a(x) \cdot a(x)\right) \cdot a(x) + \|a(x)\|^2 \nabla_x \cdot a(x)$$

Then because $a_i(x) > 0$, the equation (11) can be rewritten:

$$\Leftrightarrow \begin{cases} \|a(x)\|^4 \frac{\partial^2 v_i(t, \boldsymbol{y})}{\partial t^2} + b(x)\frac{\partial v_i(t, \boldsymbol{y})}{\partial t} + q(x)v_i(t, \boldsymbol{y}) = -\lambda_i w(x)v_i(t, \boldsymbol{y}) \\ \frac{dx}{dt} = a^{\circ -1}(x) \qquad \text{(Hadamard inverse of } a(x)) \end{cases} \tag{12}$$

$$\Leftrightarrow \begin{cases} \frac{\partial}{\partial t}\left(p(x)\frac{\partial v_i(t, \boldsymbol{y})}{\partial t} \right) + \tilde{q}(x)v_i(t, \boldsymbol{y}) = -\lambda_i \tilde{w}(x)v_i(t, \boldsymbol{y}) \\ \frac{dx}{dt} = \tilde{a}(x) \end{cases} \tag{13}$$

□