On the Fundamental Limits of Overparameterized Basis Expansion Machine Learning Models

Anonymous authors
Paper under double-blind review

Abstract

We study the generalization behavior of over-parameterized fixed basis regression models, which subsumes random feature models, extreme learning machines and are a special version of adaptive basis regression models like feed-forward neural networks. We distinguish between strict generalization, which requires recovery of the true target structure through recovery of the true coefficients in basis expansion of it, and weak generalization, which requires the minimization of test error alone. To characterize these, we introduce the sampling and expressivity thresholds, which complement the well-known interpolation threshold, which compares the training data size with model complexity. Our analysis shows that strict generalization which enables out-of-domain approximation i.e extrapolation, are unattainable in over-parameterized regimes, while weak generalization remains feasible for in-domain tasks. Moreover, using Bernstein bases and the Weierstrass Approximation Theorem, we further prove that weak generalization is theoretically always achievable for closed and bounded continuous one-dimensional functions within the training domain, a result re-emphasized from approximation theory. We also study condition number of feature matrix and reveal insights into choice of basis of the model vs stability. Our work refines the understanding of generalization in over-parameterized learning and connects classical approximation theory with modern machine learning. Finally, we discuss applications for deep neural networks and quantum machine learning. While limited to one-dimensional continuous functions with fixed bases, this analysis offers simple and refined insights into the fundamental trade-offs for over-parameterized models beyond comparison of model complexity and sample size.

1 Introduction

The recent success of deep learning has renewed interest in the surprising generalization capabilities of over-parameterized models. Classical learning theory suggests that models with more parameters than data points should overfit, yet in practice such models often generalize well. This paradox motivates fundamental questions about the nature of generalization in modern machine learning. Several phenomena are discussed today by ML/AI experts in trying to understand the generalization properties of deep neural networks. These include:

- **Double Descent**: Traditionally, increasing model complexity indefinitely is expected to improve training performance indefinitely but worsen test performance due to overfitting. This can be found in many articles and textbooks, including Bishop and Nasrabadi (2006); Geman et al. (1992); Kleinberg and Weinberger (2015). However, in some learning scenarios, test error has been observed to first increase (the classical overfitting regime) and then decrease again as model complexity keeps growing, thus resulting in a Double Descent curve. This phenomenon appears across different axes of complexity, leading to classifications such as model-wise Double Descent, sample-wise Double Descent Nakkiran (2019), and epoch-wise Double Descent Nakkiran et al. (2021). Variants such as sparse Double Descent have also been reported, further highlighting the ubiquity and richness of the behavior. It remains as one of the candidates to explain the success of deep-learning Belkin et al. (2019); Schaeffer et al. (2023); Stephenson and Lee (2021). However, it does not completely explain

the success of deep learning or general as despite double descent a model can generalize badly as seen in Figure 1.

- **Benign Overfitting**: The phenomenon of overfitting has not only been a curiosity for mathematicians and machine learning experts, but also physicists. There is a humorous take on overfitting coined as "Von-Neumann's elephant" which was termed when Enrico Fermi was once discussing his results with Von-Neumann Dyson et al. (2004): *"with four parameters I can fit an elephant, and with five I can make him wiggle his trunk"*. Traditionally, overfitting is associated with poor generalization. However, recent work has shown that in certain conditions we can achieve near perfect fit on training data and still generalize well, especially in the presence of over-parameterization Belkin et al. (2019). This counterintuitive behavior in comparison to traditional understanding is known as benign overfitting Bartlett et al. (2020). We can conclude from Figure 1, that Double Descent does not always lead to Benign Overfitting. In that case it is either model dependent or more specifically conditional on more factors than the model. However, it is clear that Benign overfitting with Double Descent is certainly the requirement of over-parameterized architectures, there does not exist universal conditions that guarantee such. The definition of benign overfitting does not have a universally accepted metric and it depends on the context of acceptable error for the task. In Mallinar et al. (2022) there are attempts to distinguish levels of overfitting and classify them as benign, tempered and catastrophic overfitting.

- **Implicit regularization during optimization**: When we are in over-parameterized regime, there are infinitely many solutions to the problem, as we are dealing with an under-determined problem. It has been shown that under certain conditions, gradient descent and several other optimization algorithms tend towards minimum norm solutions of these infinitely possible solutions, even when we are in over-parametrized regime. For linearized problems the minimum norm solution is unique and equal to the closed-form solution obtained using the pseudo-inverse method ((Neyshabur, 2017; Gunasekar et al., 2018; Soudry et al., 2018; Woodworth et al., 2020)). However, the results are still folklore with limited theoretical proof. There is no proof explaining if it is guaranteed with any or specific weight initializations. Moreover, training aspect alone cannot explain the success of deep-learning without understanding its' feature learning aspects. Also, generalization in over-parameterized regimes is not only limited to models which are trained by optimization methods, it can also be seen if we use Pseudo-inverse methods or other methods as such.

- **Grokking**: Grokking refers to a surprising behavior where a model initially overfits over the training data, but after continued training over several epochs, without any changes in architecture, data, or optimization parameters, suddenly transitions to delayed generalization. This delayed emergence of generalization remains an open problem and has sparked renewed interest in the temporal dynamics of learning in neural networks Power et al. (2022); de Mello Koch and Ghosh (2025).

- **Neural Network Gaussian Process (NNGP) Correspondence**: From a theoretical standpoint, the correspondence between neural networks and Gaussian processes (NN-GP) has provided valuable insights into the inductive biases of deep learning models Neal (2012); Lee et al. (2017); Yang (2019). However, this perspective is limited to the infinite-width regime and without accounting the training of the networks. While the NN-GP correspondence helps characterize the prior over functions induced by specific architectures and activation functions, it does not address the dynamics of training or the learnability of features. For that we need to linearize the neural networks through NTK limit Jacot et al. (2018); Vyas et al. (2022), where essentially the networks loose its' feature learning capabilities.

These are just a few different cases in the plethora of different phenomena and directions, discussed in deep learning literature today. Despite progress, key theoretical questions remain. For example, why good generalization sometimes cannot be observed in the over-parameterized regime, despite double descent? An example is shown in Figure 1, where there is double descent, but the model has wrongly approximated the function in the over-parameterized regime. Other fundamental questions include the role of the model function class and out-of-domain generalization, where the trained model is tested over a different domain than that of the training data, e.g. for extrapolation tasks.

In our article we aim to understand generalization in the over-parameterized regime in more details. We introduce two distinct notions of generalization: *strict generalization*, which requires both low residual error and accurate recovery of the target function's underlying coefficients or features irrespective of the training domain; and *weak generalization*, which focuses solely on minimizing test error, regardless of whether the true target structure is learned. Weak generalization is the usual notion of generalization in machine learning. To characterize strict generalization, we define two novel conceptual thresholds: the *sampling threshold* and the *expressivity threshold*, which complement the well-known interpolation threshold in the context of double descent.

We demonstrate that, within the over-parameterized regime, strict generalization is fundamentally unattainable in the fixed basis regression setting. As a consequence, out-of-domain approximation, which depends on true function recovery, is also unachievable in such regimes. Nonetheless, we show that in-domain approximation, where the training and test domains coincide, remains feasible through weak generalization i.e. even without learning the true underlying features/coefficients.

We further investigate whether weak generalization can be achieved by indefinitely increasing the model complexity. Using Bernstein basis functions and a version of the Weierstrass Approximation Theorem, we show that for any closed and bounded continuous function in one dimension, weak generalization is theoretically always achievable in the over-parameterized regime, but only for in-domain approximation. The proof of the Weierstrass approximation theorem is well known in the mathematics community, but a perspective on its implications on over-parameterized machine learning is mostly missing.

This work provides a refined perspective on the generalization behavior of over-parameterized machine learning models by studying the in-domain and out-of-domain approximation capabilities of a specific machine learning model and tries to contribute to closing the theoretical gap in understanding the success and limits of deep learning.

1.1 Related Works

In Muthukumar et al. (2020); Subramanian (2022), the problem of signal reconstruction and regression using basis expansion models in machine learning is investigated. In this work they study over-parameterized models from the perspective of learning the coefficients of the true function in the basis expansion. These works examine test error across various basis functions and sampling schemes with respect to the increase in model complexity(Fig. 2 in (Subramanian, 2022)). Peters and Schuld (2023) is an article in similar spirit directed towards quantum machine learning. It explores the effect of various encoding schemes for the quantum states for a quantum neural network. From the insights gained in our study, it is evident that these articles primarily focus on strict generalization defined in our article. Furthermore, the conditions required for achieving strict generalization is lightly discussed in these articles.

Our work differs from these studies and extends their findings in several key ways. First, we provide a refined classification of generalization into strict and weak forms, which has not been addressed in prior works. While Muthukumar et al. (2020); Subramanian (2022) predominantly emphasizes strict generalization, we demonstrate that generalization can also be achieved through an alternative approach, which we term weak generalization. Additionally, we establish necessary conditions for strict generalization and show that out-of-domain approximation of ground truth is not achievable in over-parameterized regimes. We also introduce new thresholds; namely sampling and expressivity thresholds, which are missing from the discussion on the generalization capabilities of over-parameterized models. Finally, we prove that for in-domain approximation using the Bernstein basis as the model basis, it can be guaranteed for any one-dimensional, closed, and bounded continuous function in highly over-parameterized regime, through weak generalization. It is a result in approximation theory which is re-emphasized in Machine learning context in this article.

Previous works like that of Poggio et al. (2019) have studied double descent in the condition number of the feature matrix, but it was limited to only random matrices. We study the behavior of condition number for various feature matrices and study if there is any correlation between the behavior of condition number and test error with respect to increasing model complexity. We attempt to answer how important a lower condition number is (in comparison to choice of the model basis and the conditions derived for strict gen-

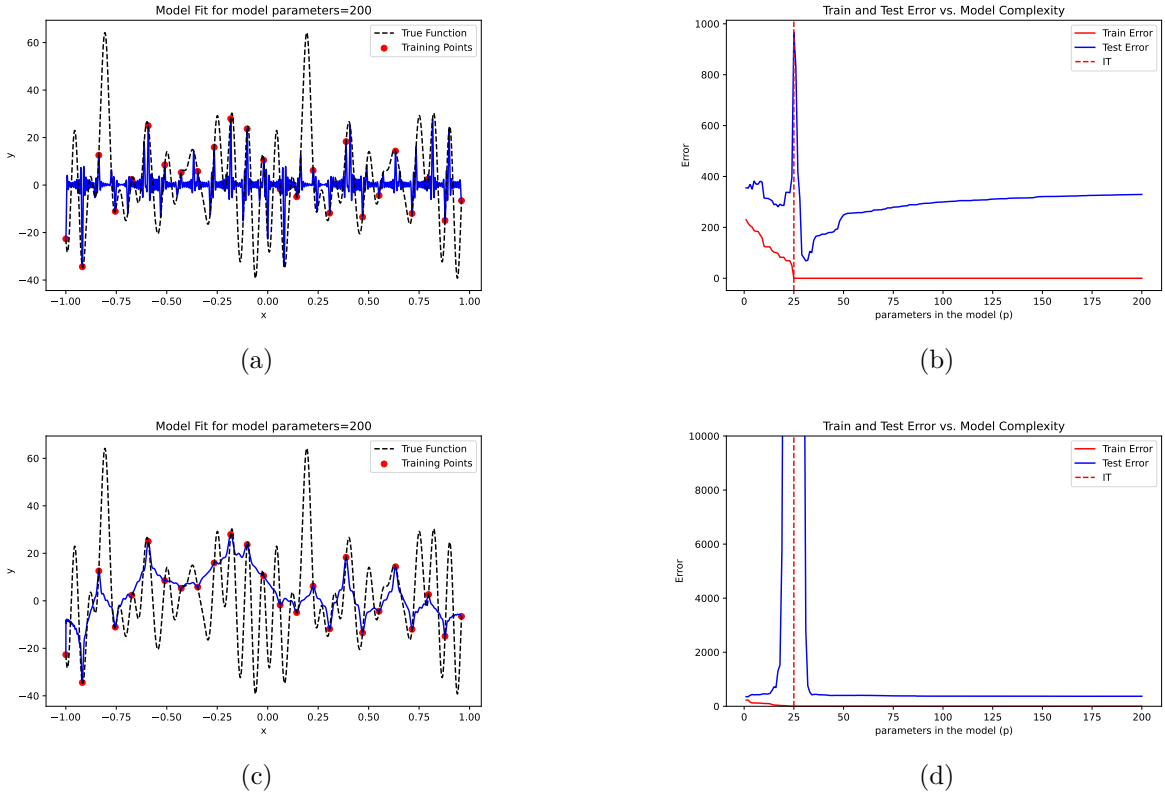


Figure 1: We approximate a 1D continuous target function (black dashed curve) using (a) a partial Fourier series and (c) linear combination of Legendre polynomial basis, in a basis expansion model. We discuss the model in more detail in section 2. The function is sampled at discrete training points (red), and the resulting fits are shown in (blue). Panel (b) shows the test error versus model complexity (up to $p = 200$) for the Fourier series fit, trained on $n = 25$ points. The characteristic Double Descent phenomenon is observed; however, the approximation remains poor in the highly over-parameterized regime, despite Double Descent. Panel (d) presents the corresponding error curve for the Legendre polynomial fit under the same training conditions. A modest improvement in generalization is observed with this alternative basis. Beyond the choice of basis can other important factors significantly influence the behavior of the error curve ?

eralization) for a good approximation of the ground truth, which is unanswered in the respective previous article.

The intent of this article is primarily theoretical, but its impact lies in clarifying concepts that affect both practical ML and theory.

1.2 Outline of the article

Our manuscript is organized as follows. In section 2 we discuss the setup of the regression task and the model used. We define important terms used in the article, like *strict generalization* and *weak generalization* and the newly introduced thresholds *sampling threshold* and *expressivity threshold*. We also discuss the sampling schemes as we will learn later about their impact on generalization. In section 3, we explore the concept of strict generalization. We prove the conditions needed to obtain it and also prove an important result that it is not possible to achieve it in the over-parameterized regime for basis expansion models. We also argue that hence, extrapolation (i.e. learning features outside training domain) is not possible in such models in over-parameterized regimes. In section 4 we discuss the concept of weak generalization and showcase that it is possible to achieve it in the over-parameterized regime, however only when the problem is not

of extrapolation. Moreover, we prove that good approximation is guaranteed in weak sense by Bernstein basis expansion model, which already well known in mathematical community but not emphasized enough in machine learning. In section 5 the effect of sampling noise on the model with respect to the choice of basis in the basis expansion. In this section we argue that the ill-conditioning of the model is secondary to the choice of the model basis. Once a model basis is chosen it can be regularized. In section 6 we discuss the insights we gain from this article into deep-learning. How our work is related to the insights from the infinite width expansion of the neural networks and how ours is a bottom-up approach while the previous is a top-down approach. How violation of the conditions in Theorem 3.1 leads them to weak generalization rather than strict one. section 7 discusses the insights gained in Quantum Machine Learning from our work. In Appendix: Appendix A we discuss the properties of various basis used and the sampling procedures. In Appendix: Appendix B we prove the corollary related to Theorem 3.1. In Appendix: Appendix C we showcase a proof related to noise and stability of the model. Appendix: Appendix D discusses a real-world example and puts our results into effect.

2 Setup

Consider the problem of approximating a 1-d continuous function on a closed and bounded domain ($[a, b]$) denoted by $g(x)$. We call it as the *target/true function* or *ground truth*.

Let n_{train} training points be sampled from the function, restricted to domain $[a_{train}, b_{train}]$, with some sampling error ϵ, such that the sampled training points are $\tilde{g}(x_i^{train}) = g(x_i^{train}) + \epsilon_i$, where $i = 0, ..., n_{train} - 1$. This gives us the training dataset $(x_i^{train}, \tilde{g}(x_i^{train}))$. The test of the approximation capability of a model is in the domain $[a, b]$. If $[a, b] = [a_{train}, b_{train}]$, then it is considered *in-domain approximation* and if $[a_{train}, b_{train}] \subset [a, b]$ it is considered *out-of-domain approximation* (extrapolation). The test set is given as $(x_i^{test}, g(x_i^{test}))$. Note, we do not consider any sampling error in the test set as we intend to understand the theoretical approximation capabilities of a model.

Before moving further let us recall the difference between interpolation and approximation. While interpolating a function we intend our model to pass through the training points, while in approximation we care more about approximating the underlying function represented by the data. Overfitting happens when the function passes through the training points, but does not approximate the function. The latter part is important. Hence, as you can predict already, overfitting is a nuisance in the approximation tasks and not for interpolation. If there is no sampling noise, a good approximation should also pass through the training points.

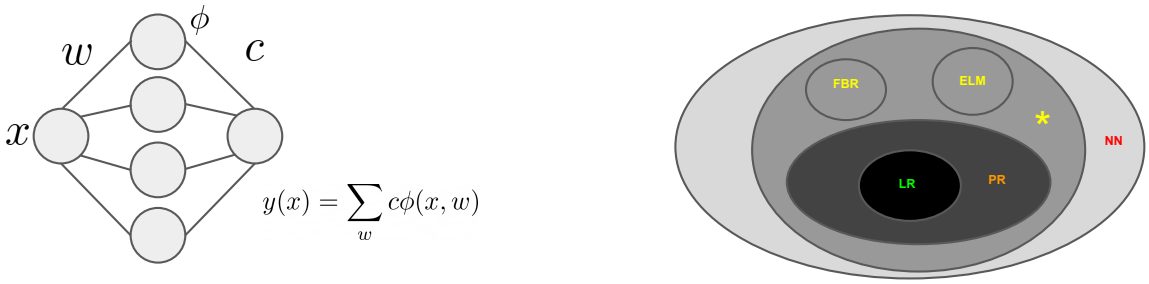
2.1 Sampling

Let the true function (in the domain $[a, b]$) be

$$g(x) = \sum_{j=0}^{d-1} \phi_t(x; w_j) c_j^{\text{true}}, \tag{1}$$

where $d \in \mathbb{Z}_+$ is the number of "continuous basis functions" ($\phi_t(x, w)$) in linear expansion of the true function. There is no restriction on the value of d and it can be infinite too. The combination of d, $\phi_t(x; w_j)$ and c^{true} generates different continuous functions in the domain of the basis used. If ϕ are polynomial bases, w represents the degree of the polynomial or if it is non-polynomial basis like sine or cosine, it is the frequency. More generally, in our article we consider w to be fixed set of hyper-parameters and belonging to the set $\{w_0, ..., w_{d-1}\}$.

Note, that basis expansion type models for regression like polynomial regression, neural networks or kernel methods (implicitly) ideally should fit functions which can be exactly expanded into a certain basis. Hence we consider functions of the type Equation 1. However, most functions in applications cannot be expanded as such. In such scenarios, we perform *surrogate modeling*, i.e., approximate the ground truth by some basis expansion, given by the Equation 1. Hence, in practice, we are indirectly approximating functions of the type Equation 1. For example, approximation of gravitational potential of Earth which is a complicated function,



(a) (b)

Figure 2: (a) Schematic comparison of fixed basis regression model and feed-forward neural network. The neural networks can be called as adaptive basis regression model. In the latter case we learn the weights (and bias), while in the previous case it is fixed. This understanding can help generalize the results and understanding of this article to deep learning. (b) Diagram showing taxonomy of basis expansion models. Linear regression model is the most restricted case of it. *LR* represents Linear Regression, *PR* represents Polynomial (monomial basis) regression, *FBR* represents Fixed basis regression (the model we use in this article), *ELM* represents Extreme Learning Machine model, * represents the group of all model which can be non-linear in features but linear in learnable weights. When we generalize outside these models we achieve the neural network model.

but we approximate it by a Legendre polynomial of certain order according to the type of application the approximation will be used Hofmann-Wellenhof and Moritz (2006). We discuss such a scenario when the ground truth is not of the type Equation 1 through a real-world example in more detail in Appendix D.

The sampled training point vector can be written as

$$\tilde{\bar{g}}^{train} = \bar{g}^{train} + \bar{E} = \Phi_t^{train}\bar{c}^{true} + \bar{E}, \tag{2}$$

where $\bar{E} \in \mathbb{R}^{n_{train}}$ represents the error vector in sampling, $\Phi_t^{train} \in \mathbb{R}^{n_{train} \times d}$ and $c^{true} \in \mathbb{R}^d$, and Φ_t^{train} is the true feature matrix defined in training domain. In general in this article we use the symbol Φ, possibly with subscripts and superscripts, to denote a matrix with entries $\Phi_{ij} = \phi(x_i, w_j)$.

The inputs can be sampled uniformly spaced, randomly or with certain sequence, i.e. at nodes of polynomials. We define it in Appendix A.

2.2 Model

To approximate the functions consider a model defined as

$$y(x_i) = \sum_{j=0}^{p-1} \phi_m(x_i; w_j)c_j \qquad\qquad \bar{y} = \Phi_m\bar{c}, \tag{3}$$

where $p \in \mathbb{Z}_+$. The model is a linear combination of the basis functions ϕ_m generated at points x. If $x = x_{train/test}$ then $\Phi_m = \Phi_m^{train/test}$ and $\Phi_m^{train/test} \in \mathbb{R}^{n_{train/test} \times p}$. The coefficient vector is $\bar{c} \in \mathbb{R}^p$. At the training points we consider

$$\bar{y}^{\text{train}} = \tilde{\bar{g}}^{\text{train}}. \tag{4}$$

Let us try to understand the relation of this model to other well-known methods in machine-learning Figure 2.

We consider w to be fixed (in both the target function and the model) in this article for simplicity. However, the coefficients c are learnable. Hence, we call it the "*fixed basis regression*" model.

If we restrict the basis to $\phi_m(x_i, w_0) = 1$ and $\phi_m(x_i, w_1) = x_i$, $(y(x_i) = w_0 + w_1x_i)$ then it is a *Linear regression model*. If ϕ_m is a monomial basis, then it is nothing but *Polynomial regression*. If w is random

	Pseudo-Inverse ($\Phi^{\#}_{p\times n}$)	Identity
$(n > p)$	$(\Phi^{\dagger}\Phi)^{-1}_{p\times p}\Phi^{\dagger}_{p\times n}$	$\Phi^{\#}\Phi = I$
$(n = p)$	$\Phi^{-1}_{n=p\times n=p}$	$\Phi^{-1}(x)\Phi(x) = \Phi\Phi^{-1} = I$
$(n < p)$	$\Phi^{\dagger}_{p\times n}(\Phi\Phi^{\dagger})^{-1}_{n\times n}$	$\Phi\Phi^{\#} = I$

Table 1: This table gives expression of Moore-Penrose Pseudo Inverse in different regions, and their properties. † represents transpose-conjugate of the matrix.

and c is learnable, it represents an *Extreme Learning Machine model*. If the ϕ_m were Fourier basis, it would represent *Random Fourier features model*. If w was learnable by the model, it would be a single-layer feed-forward neural network (without bias term). We showcase the relation in increasing level of complexity in Figure 2. Hence, the understanding of the model considered in this article leads us to gain insights into many other related models.

Note, as w are fixed, the number of learnable parameters in this model are p. Moreover, the choice of Φ_m can be different from the true Φ_t and, normally, $\bar{c}$ is different from $\bar{c}^{true}$ and it is to be learned. Crucially, the choice of model parameters (p) can also be different to the number of features in the true function (d). This is because both Φ_t and d (and also $\bar{c}^{true}$) are unknown to the learner, in general. In some examples though d can be known. For instance, for the problem of signal reconstruction (a type of approximation problem where both the true basis Φ_t and the band-limit d is known) in signal processing the band-limit d is e.g. 22 kHz for audio signals.

2.3 Training

Once, the model basis is chosen we should perform rescaling of the domain of the dataset to the domain in which the model basis is defined. This process is a homeomorphism and does not affect the structure and topological properties of the true function. For example, Legendre basis is only defined between the domain $x \in [-1.0, 1.0]$.

Now, the task of training is to find the optimal coefficients of the model called $\bar{c}^{opt}$ such that we fit the model y to resemble the true function g. This can be done by various methods, like using pseudo-inverse or with optimization methods like gradient descent. Pseudo-Inverse methods provide a closed form solution. However, they are only theoretically possible to be applied in cases where the model is linear in parameters. Moreover, where there are large datasets or many parameters, pseudo-inverse methods are not practical due to computational limitations. On the other hand, gradient descent methods solve the issues affecting the pseudo-inverse methods. However, they do not provide closed form true solution always and can get stuck in poor local minima (for non-convex loss functions) and provide wrong solutions. As our model is linear in parameters and we are trying to understand the theoretical basis of generalization, we use the pseudo-inverse solution, so as to obtain the true solution of the problem always.

Using the Pseudo-inverse method the optimal set of coefficients to fit the model to true function is

$$\bar{c}^{opt} = \Phi_m^{train^{\#}}\bar{y}^{train} = \Phi_m^{train^{\#}}(\Phi_t^{train}\bar{c}^{true} + \bar{E}) \tag{5}$$

We used Equation 2, Equation 3 and Equation 4. The symbol # denotes the pseudo-inverse, and we remind that $\bar{c}^{opt} \in \mathbb{R}^{p\times 1}$ and $\Phi_m^{train^{\#}} \in \mathbb{R}^{p\times n_{train}}$. There are three different cases of pseudo-inverse that may arise depending on its dimension, shown in Table 1, considering that Φ is always full rank, which is guaranteed if w are unique. The expression in the second equality shows the mathematical relation between the true coefficients $\bar{c}^{true}$ and the trained coefficients $\bar{c}^{opt}$. In general, these can only be equal if the model basis functions and the true basis functions agree and there is no noise.

We can check if the learned $\bar{c}^{opt}$ effectively predicts the training points via

$$\hat{\bar{y}}^{train} = \Phi_m^{train}\bar{c}^{opt} = \Phi_m^{train}\Phi_m^{train^{\#}}(\Phi_t^{train}\bar{c}^{true} + \bar{E}), \tag{6}$$

where we used Equation 5 and Equation 2 and we denoted with $\hat{\bar{y}}^{train} \in \mathbb{R}^{n_{train}}$ the predicted vector. From now on, we will use the hat symbol ($\hat{}$) to denote predicted quantities.

It is well known that good training can sometimes be misleading for approximation task. What matters in a good machine learning model for approximation is that it performs well on unknown points. These unknown points can be of two types, one inside domain and other outside domain as discussed earlier. As we will prove and demonstrate, the latter can only be achieved by strict generalization, while the previous one can be achieved by both strict and weak generalization. We will define these in a mathematical form once we discuss more nuances.

Along these lines, consider unknown data points that were not used in training the model

$$\hat{\bar{y}}^{test} = \Phi_m^{test} \bar{c}^{opt} = \Phi_m^{test} \Phi_m^{train^{\#}} (\Phi_t^{train} \bar{c}^{true} + \bar{E}). \tag{7}$$

where $\hat{\bar{y}}^{test} \in \mathbb{R}^{n_{test}}$ is the predicted vector. Here, $\Phi^{test} \in \mathbb{R}^{n_{test} \times p}$. Note that Φ_m^{test} is defined on test inputs lying in the range $[a, b]$, which can be different from $[a_{train}, b_{train}]$. We consider $n_{test} \gg n_{train}$ so as to theoretically better evaluate the capabilities of the model.

Now, we can check the training and testing performance of the model in unknown parts by using the metric called “residual" as given below

$$\bar{R}^{train/test} = \hat{\bar{y}}^{train/test} - \bar{g}^{train/test} \tag{8}$$

Note that in case of training the second term is Equation 4 while in case of testing it is nothing but the true function at the test points without any sampling error, as discussed earlier for theoretical purposes.

Having said this, let us classify generalization. This nuanced understanding of generalization, leads us to understand the conundrum around generalization capabilities of over-parameterized machine learning models. It also showcases the limits of the over-parameterized models in terms of approximation.

Definition 2.1. We define **strict generalization** when $\bar{R}^{test} = \bar{0}$ and $\bar{c}_{p\times 1}^{opt} = \bar{c}_{p\times 1}^{true,pad}$, namely when the residual is zero and we have also perfectly learnt the coefficients.

In the above definition

$$\bar{c}_{p\times 1}^{true,pad} = \begin{pmatrix} \bar{c}_{d\times 1}^{true} \\ 0_{(p-d)\times 1} \end{pmatrix}_{p\times 1} \tag{9}$$

Since the objective is to learn the true coefficients of the underlying function, such generalization enables extrapolation, that is, predicting function behavior outside the training domain. This kind of out-of-domain approximation is powerful because it allows us to make predictions even in regions where no data was observed during training.

However, we must be cautious: extrapolation is only reliable in cases where the function exhibits repeating structure across the domain. For example, linear and periodic functions possess globally consistent patterns, which makes them inherently suitable for extrapolation. In contrast, non-linear, non-periodic functions typically lack such regularity, so extrapolation is only an approximation and becomes increasingly unreliable the farther we move from the training domain. In mathematical terms to represent the same function in a different domain we need a different set of $\bar{c}^{true}$. It changes the farther we go away from the training domain.

This limitation arises because the training domain is smaller than the test domain, and the optimal coefficients learned from the training data may not generalize well to unseen regions. As a result, the learned coefficients may diverge from the true coefficients, leading to poor fits or entirely incorrect predictions in regions far from the training set. For this reason, when working with complex, non-linear, non-periodic functions, we usually assume that the test domain is near the training domain. That said, when out-of-domain approximation is achieved, it represents a strong and meaningful form of generalization—it implies that the model has captured some deeper structure of the function that holds even in regions it has never seen before.

However, generalization can also be achieved without learning the true coefficients (features), as we will showcase later in this article.

	Parametrization	Sampling	Expressivity
Under	$n < p$	$n < d$	$p < d$
Threshold	$n = p$	$n = d$	$p = d$
Over	$n > p$	$n > d$	$p > d$

Table 2: Table depicting various regions of the learning space.

Definition 2.2. We define **weak generalization** when $\bar{R}^{test} = 0$ even if we do not learn the true coefficients.

We showcase that such generalization then is only restricted to the training domain (i.e. in-domain regression) and we cannot extrapolate in such a case if we do not learn the true coefficients.

Let us now understand the criteria required for strict generalization and weak generalization. Also, let us prove/disprove which type of generalization we can observe in over-parameterized regime with experimental proof.

3 Strict Generalization

To understand the generalization properties in more detail we need to define new thresholds apart from the interpolation threshold (also called "parametrization threshold" in Table 2). We call the condition where $n = d$, the "sampling threshold" (ST) and the condition where $p = d$, the "expressivity threshold" (ET) and, as already mentioned, when $n = p$ we call it the "interpolation threshold" (IT). See also Table 2 for a summary of the various regimes. As we sweep the number of parameters p while keeping n constant we pass through the ET, but the condition that we are below ST or not is decided when we choose the number of training data points (n), and it is not visible on the learning curve plots.

Let us derive the conditions needed for strict generalization and understand the regions in which we can obtain it.

Theorem 3.1. *The necessary conditions for $\bar{c}^{opt} = \bar{c}^{true,pad}$ in the fixed basis regression model are*

1. *Zero sampling-noise contribution:* $\Phi_m^{train^{\#}} \bar{E} = \bar{0}$
2. *Enough Expressivity:* $p \geq d$.
3. *Sampling sufficiency:* $n \geq d$.
4. *Under-parameterization:* $n \geq p$.
5. *Span-inclusion:* $\text{Span}\{\phi_t(x_i; w)\}_{i=1}^n \subseteq \text{Span}\{\phi_m(x_i; w)\}_{i=1}^n$.

The most important implication in terms of approximation capabilities in over-parameterized regimes is that we cannot have strict generalization in an over-parametrized regime with the fixed basis regression model.

Proof. Let us start with Equation 9. We know that from Equation 5 and Equation 2 that

$$\bar{c}^{opt} = \Phi_m^{train^{\#}} \Phi_t^{train} \bar{c}^{true} + \Phi_m^{train^{\#}} \bar{E} \tag{10}$$

For Equation 9 to be true we need either $\Phi_m^{\#} \bar{E} = \bar{0}$ or the sampling noise $\bar{E} = \bar{0}$. This gives us the first criterion. So Equation 9 reduces to obtaining conditions for

$$\bar{c}^{opt} = \Phi_m^{train^{\#}} \Phi_t^{train} \bar{c}^{true} = \bar{c}^{true,pad}. \tag{11}$$

This implies that the necessary condition for it to be true is

$$\Phi_m^{train^{\#}} \Phi_t^{train} = \begin{pmatrix} I_{d\times d} \\ 0_{(p-d)\times d} \end{pmatrix}_{p\times d} = I_{p\times d}^{pad}. \tag{12}$$

As $\text{Rank}(I^{pad}) = d$, we see that we should have $p \geq d$. This gives us the second criteria, that the model has to be expressive enough, and it is decided by the "expressive threshold".

Let us write

$$\Phi_m^{train^{\#}} = \left(\frac{A_{d\times n}}{B_{(p-d)\times n}}\right)_{p\times n}, \tag{13}$$

so according to Equation 12 we should have

$$A\Phi_t^{train} = I_{d\times d}, \tag{14}$$

$$B\Phi_t^{train} = 0_{(p-d)\times d}. \tag{15}$$

The first equation implies that A must be left inverse of Φ_t^{train}, only then can it be satisfied. As left inverse only exists if $n \geq d$, i.e. the number of training points needs to cross the sampling threshold, for $\bar{c}^{opt} = \bar{c}^{true,pad}$. This leads us to the third condition for obtaining strict generalization. This criteria is similar to the "Shannon-Nyquist sampling theorem" in signal reconstruction theory and Sampling complexity in PAC theory Kearns and Vazirani (1994); Ehrenfeucht et al. (1989). Moreover,

$$\text{Im}(\Phi_t^{train}) \subseteq \text{Ker}(B), \tag{16}$$

namely the column space of Φ_t^{train} is contained in the null-space/kernel of B. A null-space of B is a set of all vectors $\bar{v}$ such that $B\bar{v} = \bar{0}$. Note, that $\bar{v}$ need not be zero vector always, for it to be true. Now, let $\text{Rank}(B) = r$, then by the Rank-Nullity Theorem,

$$\text{Rank}(B) + \text{Nullity}(B) = r + \dim(\text{Ker}(B)) = \dim(\text{Dom}(B)) = n \tag{17}$$

Here Dom means "Domain" and dim is to represent "dimensions". This gives us $\dim(\ker(B)) = n - r$, i.e. any of set of vectors in $\text{Ker}(B)$ can have atmost $n - r$ linearly independent vectors. Now according to Equation 16, $\dim(\text{Im}(\Phi_t(x)) \leq \dim(\text{Ker}(B))$. This means $d \leq n - r$. Now, the rank r can be either n or $p - d$, as we are considering full rank assumption. Suppose $r = n$, then we get $d \leq 0$, which is not possible as d is an integer greater than or equal to 1, hence, for strict generalization r cannot be equal to n. Suppose we use $r = p - d$, then $d \leq n - (p - d)$, this gives us the criteria that $n \geq p$, for Equation 15 to be satisfied. Satisfying all these conditions leads us to obtain $\bar{c}^{opt} = \bar{c}^{true,pad}$.

Merely satisfying Equation 9 does not lead to strict generalization, we also need according to Equation 8, $\bar{R}^{test} = \Phi_m^{test}\bar{c}^{opt} - \Phi_t^{test}\bar{c}^{true} = \Phi_m^{test}\bar{c}^{true,pad} - \Phi_t^{test}\bar{c}^{true} = \bar{0}$, for this along with Equation 9 we need $\Phi_m^{train} = (\Phi_t^{train}|\Phi'_{n\times(p-d)})_{n\times p}$ (i.e. $\text{Span}\{\phi_{\text{t}}^{train}(x;w)\} \subseteq \text{Span}\{\phi_{\text{m}}^{test}(x;w)\}$) .

If we achieve all the conditions mentioned above and learn the true coefficients, then $\bar{R}^{test} \approx \bar{0}$, leading to strict generalization. However, we may not be able to learn it if the size of training domain is very small compared to the test domain for a general set of functions (except linear and periodic functions), as we discussed in previous section. This finishes the proof. □

Let us now justify the previous result using examples. Before going ahead we need to emphasize that to compare behavior across datasets and scales we will use "normalized root mean square" rather than "root mean square" error. Note that it does not change the behavior of the error except that it rescales the error for comparison. The equation for normalized root mean square is given as

$$\text{NRMSE}_{\text{train}} = \frac{\sqrt{\frac{1}{n_{train}}\sum_{i=1}^{n} R_i^{train^2}}}{(y_{max} - y_{min})^{train}}, \tag{18}$$

where $\bar{R}^{train}$ is the residual in the equation Equation 8. Similarly we can define the NRMSE_{test} for the test error.

We illustrate our result using one-dimensional continuous function constructed from a Chebyshev polynomial basis of maximum degree $d = 13$. The training dataset consists of $n = 26$ noise-free points, restricted in the domain $[-0.5, 0.5]$. We intend to perform out-of-domain approximation in the region $[-1.0, 1.0]$. Note that

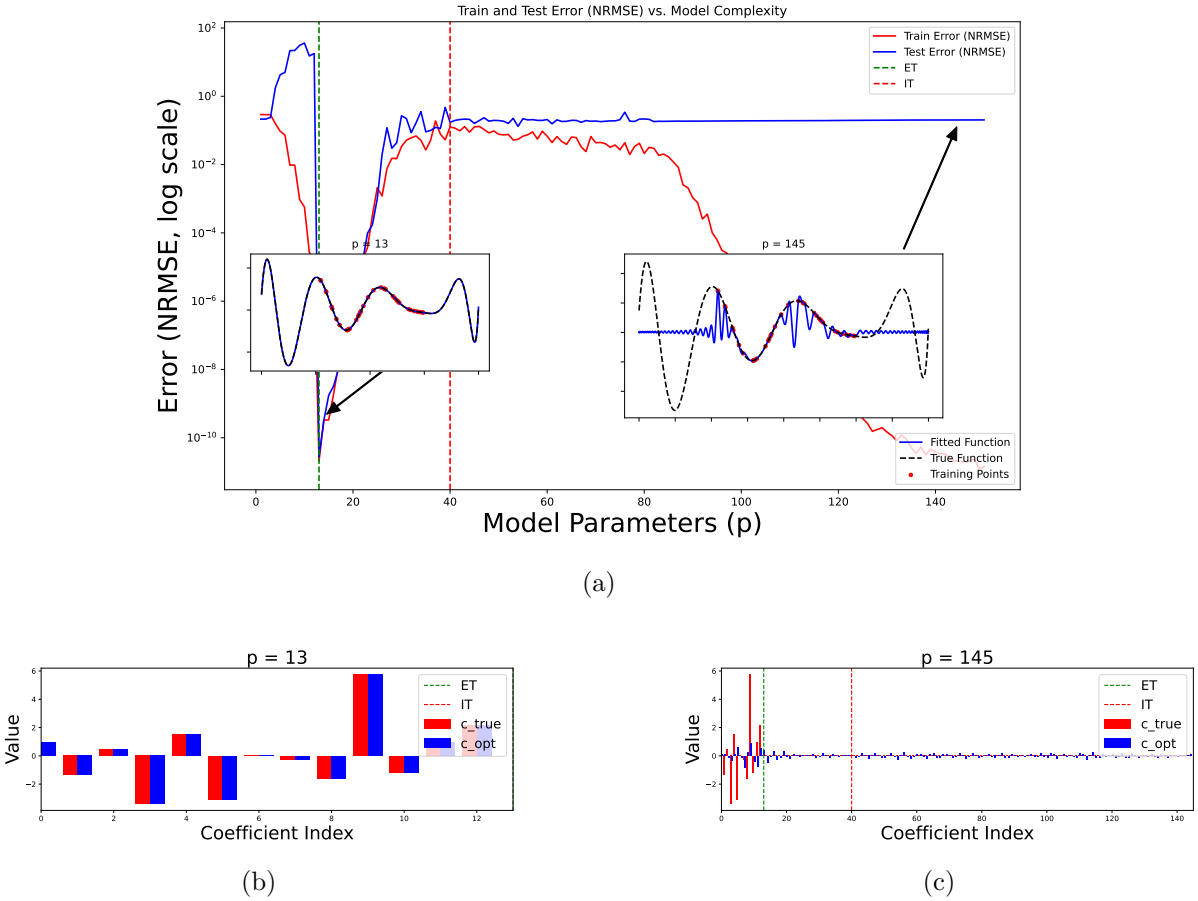


Figure 3: (a) NRMSE (log-scale) v/s model parameters (p). We are interested in out-of-domain approximation and demonstrating the observations of Theorem 3.1. The true function is generated from linear combination of Chebyshev polynomial basis. It can however be any other type of 1-d continuous function. The order of Chebyshev polynomial is $d = 13$. We consider $n = 40$ randomly spaced noise-less training points, restricted to the domain $[-0.5, 0.5]$ and test the approximation in the domain $[-1.0, 1.0]$. We are above the sampling threshold ($(n = 40) \geq (d = 13)$). Inset shows approximation at $p = 13$ and $p = 145$. We observe that $p = d = 13$ the model extrapolates, while at $p = 145$ (over-parameterized regime) the extrapolation capability is lost. (b–c) Histogram comparing $\bar{c}^{opt}$ and $\bar{c}^{true}$ at $p = 13$ and $p = 145$. At $p = 13$, the model learns the true coefficients as expected from Theorem 3.1 and looses it in the over-parameterized regime at $p = 145$.

we are well above the sampling threshold ($n = d$). The model is built using a Chebyshev basis, identical to that of the true function. In particular, as we increase the complexity of the model (p), and when it is equal to the highest degree of the true function (d) ($p = d$), all the conditions required for strict generalization are met, as stated in Theorem 3.1. In this regime, the model achieves perfect extrapolation. The corresponding results are presented in Figure 3. We plot the normalized root mean square error (NRMSE) as a function of the model complexity p. Additionally, we compare the true coefficient vector ($\bar{c}^{\text{true}}$) with the learned coefficients ($\bar{c}^{\text{opt}}$) both at $p = d$ and at an over-parameterized setting $p \gg d$. The associated fitted functions at these complexities are also shown to illustrate the behavior of the model. We can also observe that we do not achieve strict generalization and hence out-of-domain approximation in the over-parametrized regime, which is one of the main results of the theorem above.

We are still unaware of the precise structure of the matrix $\Phi'_{n\times(p-d)}$ that is required for achieving *sustained strict generalization* in the regime $d < p < n$. While we observe generalization exactly at $p = d$ satisfying the conditions above, increasing model complexity beyond this point does not guarantee that the learned coefficients will match the true coefficients—unless additional conditions are satisfied.

We refer to this intermediate region, where $d < p < n$ but strict generalization can still occur under specific structural constraints, as the "extrapolation regime". This regime represents a phase in the learning process where *strict generalization* is possible but not guaranteed, depending on the nature of Φ' and the alignment between the model and the true function.

Corollary 3.2. *Strict generalization in the extrapolation regime ($d < p < n$) occurs only for an orthogonal basis in the fixed basis regression model, even if all the conditions proved in Theorem 3.1 are satisfied.*

Check Appendix B for the proof.

We should note that when we are dealing with discrete orthogonal polynomial basis; they are usually orthogonal with respect to some weighting function (W) which depends on the input (x), i.e.

$$\sum_{i=0}^{n-1} \phi^a(x_i)\phi^b(x_i)W(x_i) = c_b\delta_{a,b} \tag{19}$$

Where c_b is a real number. Some orthogonal polynomials (like Legendre) have a weighting function equal to 1, even the basis like Fourier basis have a weighting function equal to 1. However, orthogonal polynomials like Chebyshev have a non-constant weighting function. Monomial basis do not form an orthogonal polynomial basis. Hence we cannot expect strict generalization for them in the described region, unless we consider it into account. We can always absorb the weighting function inside the orthogonal polynomial basis and avoid the weighting function, when choosing a model.

Let us demonstrate this understanding in Figure 4 while approximating the function outside the training domain in the region $[-1.0, 1.0]$. We consider a function generated from the Fourier basis. The highest frequency of the basis considered is $f_{max} = 6$, this leads to the true complexity of the function to $d = 2f_{max} + 1 = 13$. We consider a model with a Fourier basis for regression and consider $n = 35$ noiseless training points which are randomly spaced in the domain $[-0.5, 0.5]$. This way we are above the sampling threshold (i.e. $n = d$). As we increase the model-complexity (p) and we surpass the expressive threshold ($p = d$), we satisfy all the conditions needed for strict generalization. However, as we increase the model complexity in the region ($d < p \leq n$), because the Fourier basis is orthogonal to each other, we can sustain the strict generalization in the extrapolation regime discussed in Corollary Corollary 3.2.

To conclude the discussion on strict generalization, let us observe, what happens when we violate one of the assumptions in Theorem 3.1. Let us consider that we do not sample enough Figure 5(a) i.e. do not surpass the sampling threshold ($n = d$). We consider another case where we do not choose a right basis Figure 5(b). What happens if we violate both Figure 5(c). All of the scenarios show that we achieve Double Descent but we do not generalize well in the over-parameterized regime.

The code for these demonstrations can be found (to be added later)

4 Weak Generalization

We previously demonstrated that strict generalization is not achievable in the over-parameterized regime, which in turn led us to conclude that even out-of-domain approximation fails under such conditions. Now, let us explore whether in-domain approximation is feasible in the over-parameterized setting.

Can this be achieved without recovering the true coefficients (or features) of the target function? Is it necessary to adhere to the specific conditions we derived earlier for in-domain approximation, or can we succeed without them? Furthermore, in the previous section, we observed that strict generalization requires the model's basis to closely match that of the target function. Does this constraint also apply here? Or is it sufficient to use a more flexible basis—such as one with universal approximation capabilities?

Ultimately, this discussion helps us understand the core puzzle: Why is it that an over-parameterized model, despite being under-determined can still approximate a target function effectively within the training domain?

Let us recall "Weierstrass approximation theorem" from approximation theory Rudin (1976). It states that

Theorem 4.1. *For any real continuous function $g(x)$ defined on a closed and bounded interval $[a, b]$, there exist polynomials $y_p(x)$, where p is highest degree in the polynomial, such that $\lim_{p\to\infty} y_p(x) = g(x)$ uniformly for all points in the domain.*

In the language of Machine learning, y represents the model we use for learning and g is the true function. As the order of the polynomial basis increase to infinity, (in our fixed basis regression model, it is the number of parameters) $\lim_{p\to\infty} y_p(x) = g(x)$ for such functions; $\bar{R}^{test} = \bar{0}$, provided that sampling noise is negligible ($\bar{E} = \bar{0}$). Note that this theorem is applicable when the domain of the polynomial is same as the domain of the function to be approximated. This theorem strengthens the notion of generalization in over-parametrized regime for in-domain generalization of continuous functions. Moreover, we recall from approximation theory that Bernstein polynomial basis achieves this irrespective of the target function (it has to be continuous). However, other polynomial basis like Chebyshev basis or Legendre basis cannot achieve this.

Note, this theorem is different from the Universal approximation theorem Cybenko (1989), which does not concern itself with the number of parameters/neurons used to approximate a target function.

It shows us that it is not surprising that we observe generalization in over-parameterized regime, despite it contradicting the conventional notion of "statistical learning theory".

4.1 Heuristic proof of weak generalization in over-parameterized regime using Bernstein basis

Let us study the notable case of using Bernstein basis for fixed basis regression model. The proof of Weierstrass approximation is available in textbooks, but we prove it again here in the machine learning perspective and showcase that it (in-domain) approximates continuous functions in over-parameterized regimes, irrespective of learning the true coefficients of the true function and irrespective to the type of the target function (except that it has to be continuous and lie in the range $[0, 1]$).

The Bernstein basis is defined as

$$\Phi_{m_{i,j}} = b_{j,p-1}(x_i) = \binom{p-1}{j} x_i^j (1-x_i)^{p-1-j}, \tag{20}$$

where here $x \in [0, 1]$. Note, through out this section, we will consider the training domain as same as test domain. It can be observed that the Bernstein basis is nothing but the probability mass function of a binomial distribution. This plays an important role in its being a universal model basis for in-domain generalization. The feature matrix of this basis is

$$\Phi_m = \begin{bmatrix} b_{0,p-1}(x_0) & b_{1,p-1}(x_0) & \cdots & b_{p-1,p-1}(x_0) \\ \vdots & & & \\ b_{0,p-1}(x_{n-1}) & b_{1,p-1}(x_{n-1}) & \cdots & b_{p-1,p-1}(x_{n-1}) \end{bmatrix}_{n\times p}. \tag{21}$$

By De-Moivre Laplace theorem, for $p \to \infty$ such polynomials can be approximated by a Gaussian distribution

$$\lim_{p\to\infty} b_{j,p-1}(x_i) = \frac{e^{-(j-\mu_i)^2/2\sigma_i^2}}{\sqrt{2\pi\sigma_i^2}}, \tag{22}$$

where $\mu_i = (p-1)x_i$ and $\sigma_i^2 = (p-1)x_i(1-x_i)$. We demonstrate this behavior in Figure 6, for various sampling schemes. We can write

$$\lim_{p\to\infty} \Phi_{m_{i,j}} \approx a_i \delta_{j,\lfloor (p-1)x_i \rfloor}, \tag{23}$$

where a_i is some positive real number and $\lfloor\ \rfloor$, represent the floor function, as the index has to be an integer. So,

$$\lim_{p\to\infty} \Phi_m \bar{c} \approx \lim_{p\to\infty} \sum_{j=0}^{p-1} a_i\, \delta_{\lfloor (p-1)x_i \rfloor, j}\, c_j, \quad \text{where } \delta \in \mathbb{R}^{n\times p},\ c \in \mathbb{R}^p. \tag{24}$$

The right inverse of δ matrix given here, when $p > n$, is approximately equal to δ^T. We showcase this in the Figure 7.

The optimal coefficients using the training inputs can then be written as using the Pseudo-inverse methods as

$$\lim_{p\to\infty} c^{opt} \approx a_i^{-1}\delta_{j,\lfloor(p-1)x_i^{train}\rfloor}y(x_i^{train}), \tag{25}$$

$$\lim_{p\to\infty} c_j^{opt} \approx a_j^{-1}y(\frac{j}{p-1}). \tag{26}$$

This indicates that c^{opt} passes through the sampled points. We showcase this behavior in Figure 8.

Now definition of test residual Equation 8 and Equation 23

$$\bar{R}^{test} = \hat{\bar{y}}^{test} - \bar{g}^{test} = \Phi_m^{test}\bar{c}^{opt} - \bar{g}^{test} \tag{27}$$

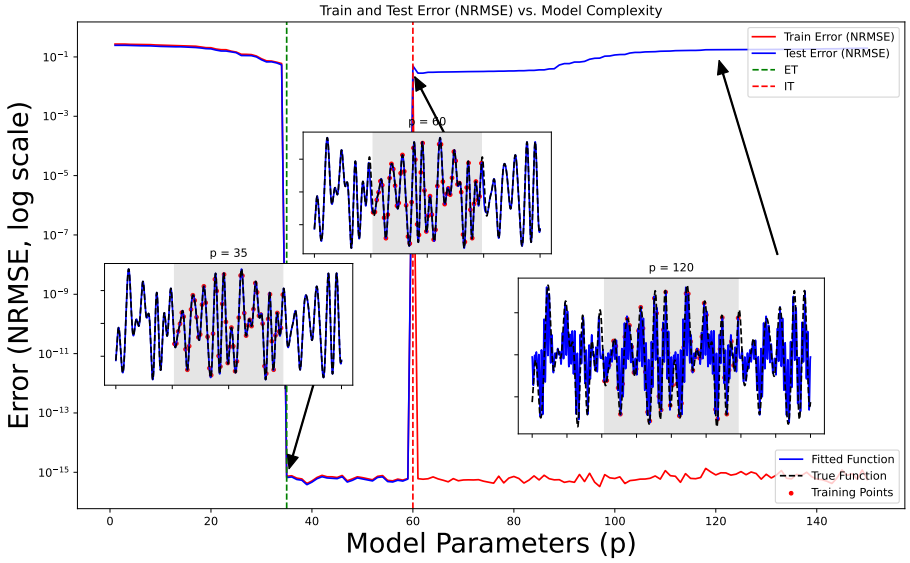
$$= a_k(\delta_{\lfloor(p-1)x_k^{test}\rfloor,j})y(\frac{j}{(p-1)})a_j^{-1} - \bar{g}^{test} = y(x_k^{test}) - g(x_k^{test}). \tag{28}$$

If there was no error in sampling the training points, it will not propagate into the test prediction and $\bar{y}^{test}$ becomes equal to $\bar{g}^{test}$, making the residual equal to zero (otherwise non-zero according to the noise in sampling).

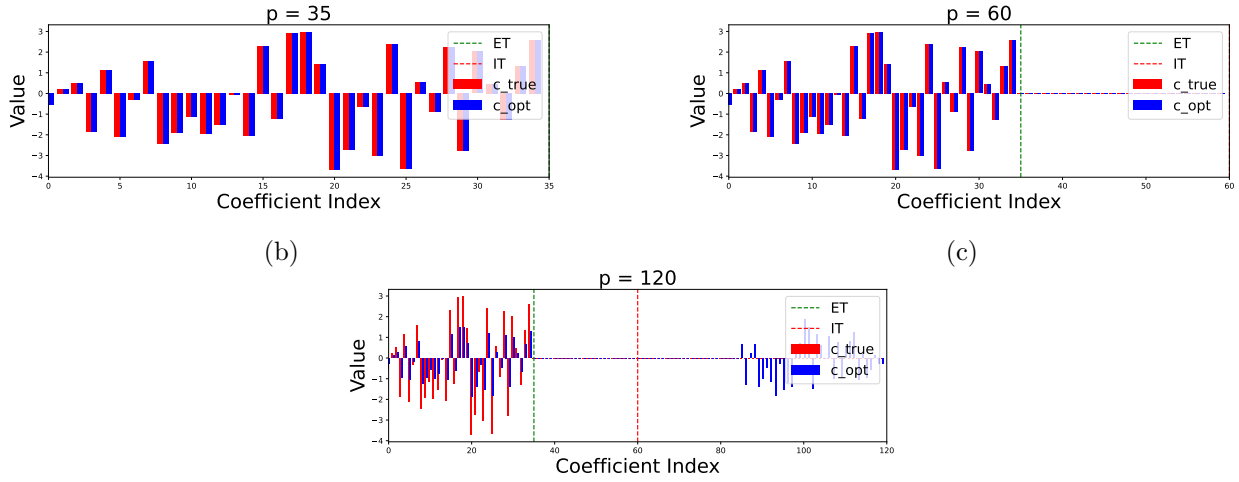
We illustrate this phenomenon in Figure 9. The true underlying function is constructed using a Legendre basis of maximum degree $d = 17$. Using $n = 50$ training points sampled uniformly from the domain $[0, 1]$, we fit a fixed-basis regression model. We use the domain because, Bernstein basis is restricted to this domain. When evaluating this model in the same domain using an over-parameterized representation via the Bernstein basis, we observe that the model continues to generalize well despite the increased model complexity. Backed by the proof above we can see that this behavior is a direct consequence of the Weierstrass approximation theorem, stated above. While this result is fundamental in approximation theory, it is often overlooked in discussions within the machine learning community. Recognizing its implications sheds light on why in-domain approximation of function in over-parameterized regimes should not be surprising.

This behavior is a direct consequence of the Weierstrass approximation theorem, stated above. While this result is fundamental in approximation theory, it is often overlooked in discussions within the machine learning community. Recognizing its implications sheds light on why in-domain approximation of function in over-parameterized regimes should not be surprising. Figure 11 uses the same function as in Figure 10. We can observe that Chebyshev polynomial basis does not gener- alize well in the over-parametrized regime. Showcasing that Weierstrass approximation showcases that generalization in over-parameterized regimes for in-domain approximation of continuous functions can be achieved, but not with any type of polynomial.

Figure 10 uses the same function as in Figure 9. We can observe that Chebyshev polynomial basis does not generalize well in the over-parametrized regime. Showcasing that Weierstrass approximation showcases that generalization in over-parameterized regimes for in-domain approximation of continuous functions can be achieved, but not with any type of polynomial.



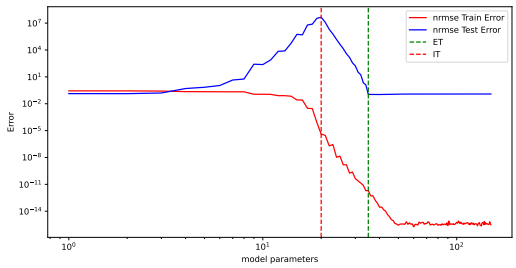
(a)



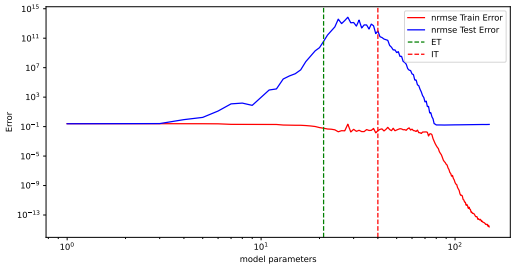
(b) (c)

(d)

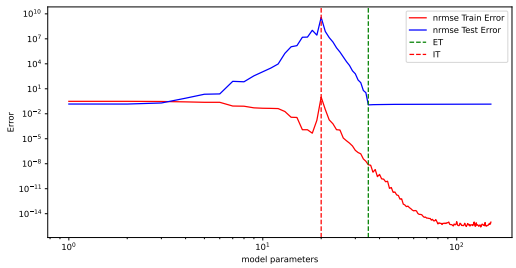
Figure 4: We showcase sustained strict generalization in the "extrapolation regime" for out-of-domain approximation. Main plot considers NRMSE (log-scale) v/s model parameters (p). The true function is a periodic function which can be written using Fourier basis which is orthogonal basis. In this case $d = 2 * f_{max} + 1 = 17 * 2 + 1 = 35$. We consider $n = 60$ uniformly spaced training points, restricted to the domain $[-0.48, 0.48]$ and the test domain is $[-1.0, 1.0]$. We are above the sampling threshold ($(n = 60) \geq (d = 35)$). (a) Showcases error ($NRMSE$) v/s model complexity as we increase the number of parameter p. Inset plots show the fit at $p = 35, 60, 120$. We can observe that we loose the extrapolation capability at $p = 120$ as expected from the Theorem 3.1. The sustained strict generalization between ET and IT is guaranteed in this case for Fourier basis (as it is orthogonal) by Corollary 3.2. (b–d) Histograms comparing $\overline{c}^{opt}$ and $\overline{c}^{true}$. They demonstrate the coefficients are exactly learned at $p = 35$ and $p = 60$ (which is in the region $d < p \leq n$), but it is lost in over-parameterized regime $p = 120$ as expected by Theorem 3.1.



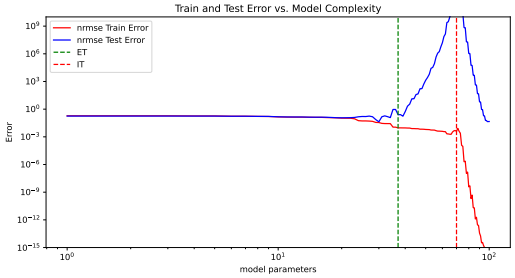
(a) Not having enough samples ($n < d$)



(b) $\text{Span}\{\phi_{\text{t}}(x; w)\} \not\subseteq \text{Span}\{\phi_{\text{m}}(x; w)\}$

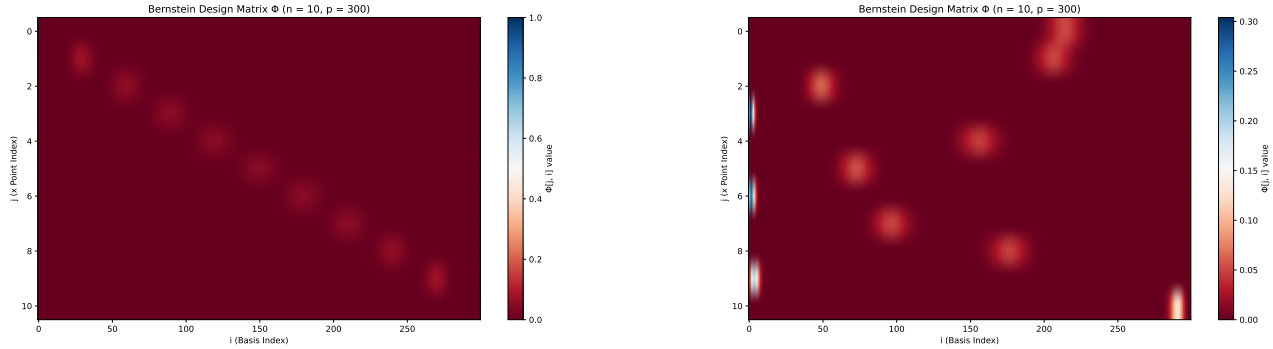


(c) Not having enough samples + basis mismatch



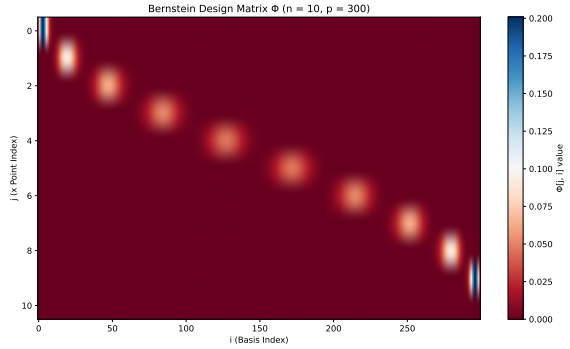
(d) Noisy samples + basis mismatch

Figure 5: Illustration of different types of model violations. In (d) we observe triple descent behavior at ($n = d$) as observed in d'Ascoli et al. (2020).

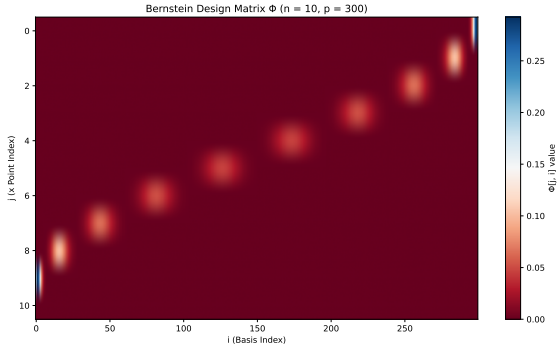


(a) Uniform

(b) Random

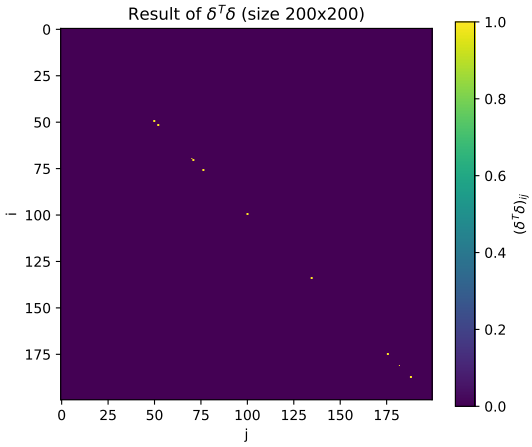


(c) Legendre

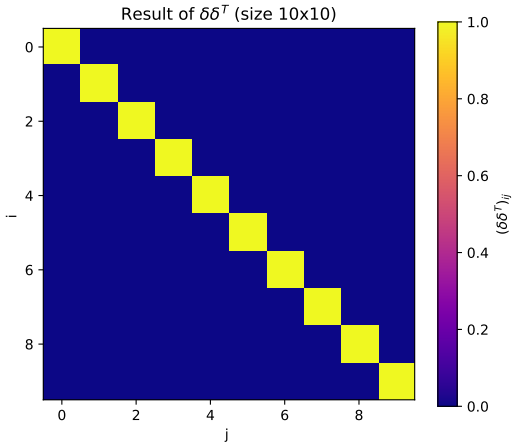


(d) Chebyshev

Figure 6: We plot the values of the Bernstein basis design matrix. It can be seen that we observe the gaussian distribution with mean at points given by $\lim_{p\to\infty} b_{j,p-1}(x_i) = \frac{e^{-(j-\mu_i)^2/2\sigma_i^2}}{\sqrt{2\pi\sigma_i^2}}$



(a)



(b)

Figure 7: We consider a $\delta_{n\times p}$ ($n = 10$ and $p = 200$) matrix defined in Equation 23, where x_i are ($n = 10$) generated randomly in the domain $[0, 1]$. (b) Showcases that the right inverse of such a matrix is its' transpose when $p \geq n$. (a) Showcases the transpose of δ matrix is not its left inverse.

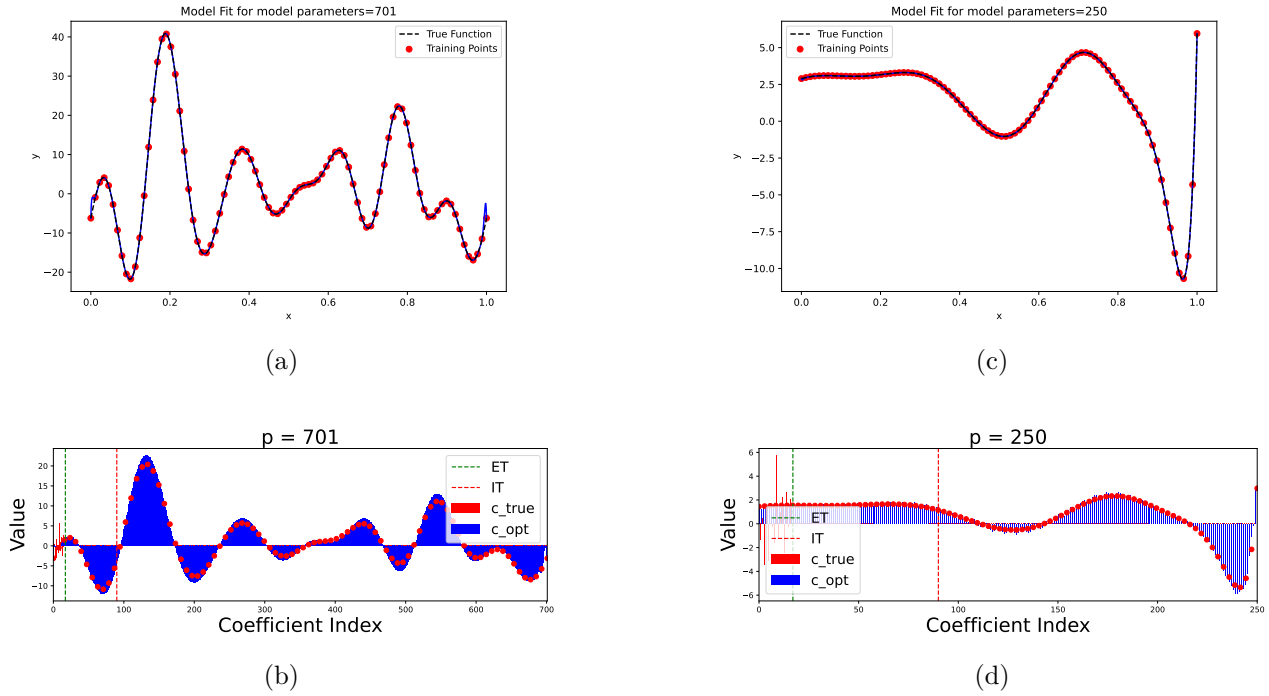
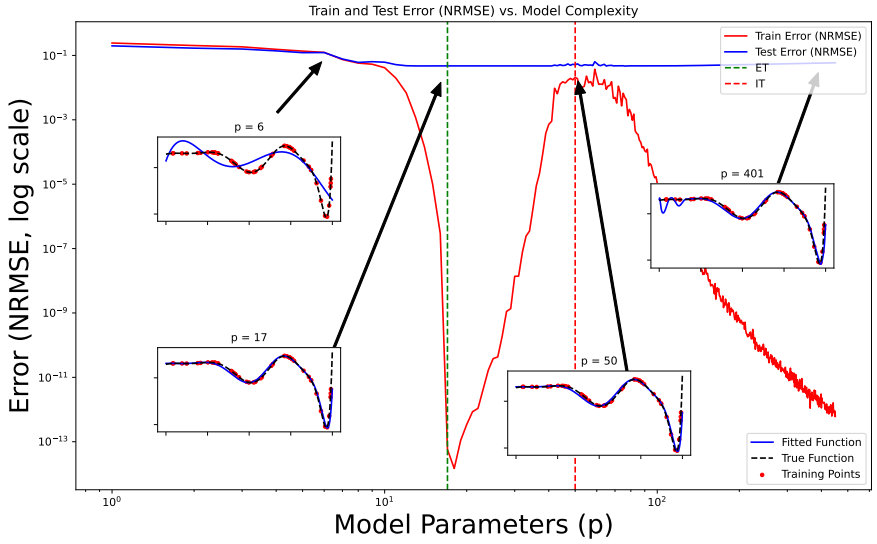
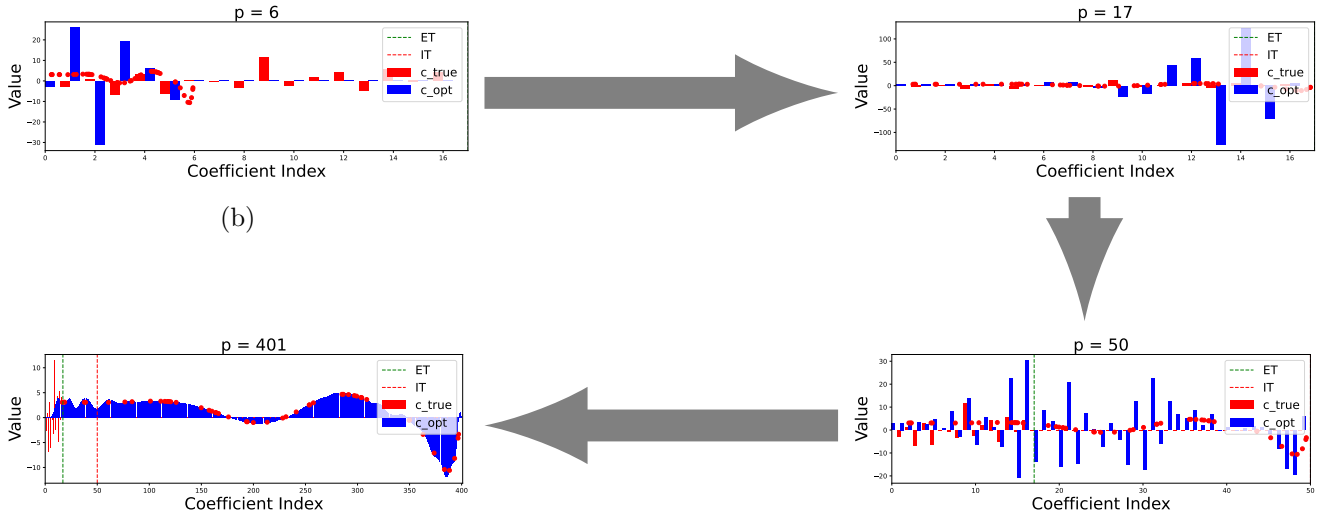


Figure 8: In these plots we showcase that the coefficients become approximately equivalent to the sampled training data points, when we use Bernstein basis for regression. (a) Shows the in-domain approximation of function generated using Fourier basis, with highest frequency equal to 17 and 90 training data points. The points are sampled randomly. The plot is generated at 700 model parameters. (b) Shows the histogram of true and learned coefficients. It can be seen that their values are equivalent to that of the training data points. (c) Shows the in-domain approximation of function generated using Legendre basis, with highest frequency equal to 17 and 50 training data points. The plot is generated at 150 model parameters. (d) Shows the histogram of true and learned coefficients. It can be seen again that their values are equivalent to that of the training data points. In both (b) and (d) plots, we should point out that the true coefficients are restricted behind the expressive threshold and are not visible as their values are way less than the learned coefficients.



(a)

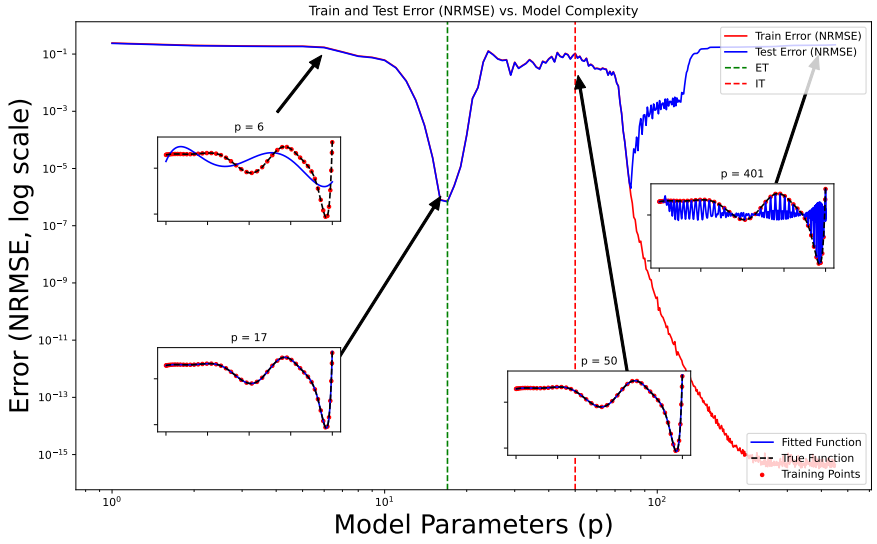


(b)

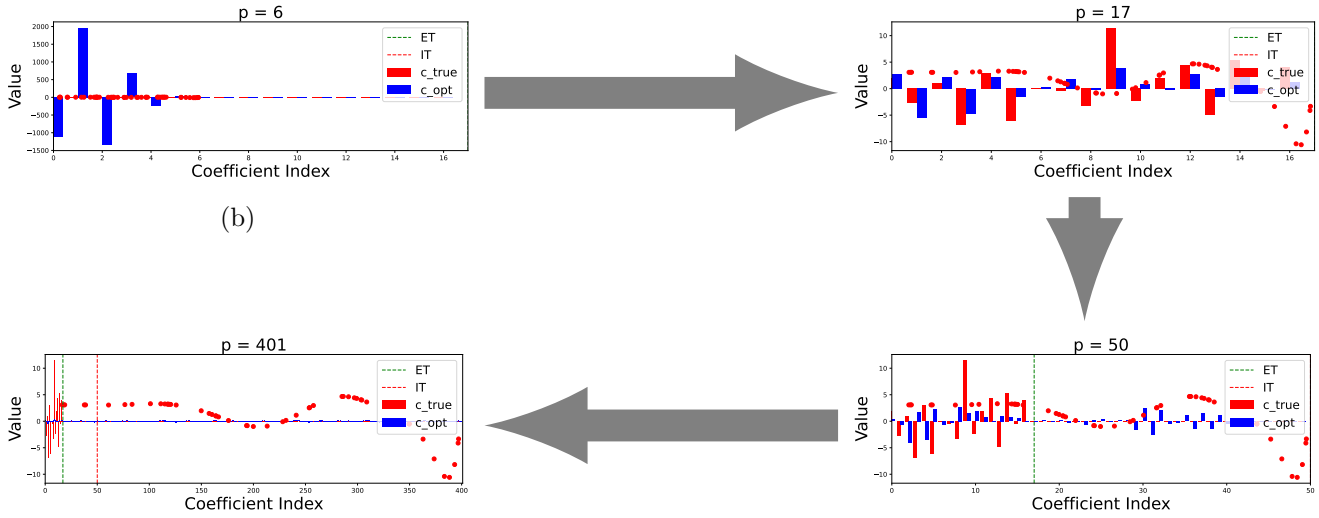
(e)

(d)

Figure 9: (a) In this figure we plot NRMSE error v/s Model complexity, in log scale. The problem is of approximating a function generated using Legendre basis of highest degree $d = 17$ in the domain $[0, 1]$. We sample $n = 50$ samples at random points. We use Bernstein basis for defining the model. The figures in the inset showcase the approximations over various model complexities. It can be seen that the model generalizes well even beyond over-parameterized regime. This is a demonstration of Weierstrass approximation theorem, which strengthens our observation of phenomenon like benign overfitting and generalization in over-parameterized regime. (b) In this subfigure we showcase histogram of learned coefficients (c^{opt}) and true coefficients (c^{true}) superimposed by the values of the training data points, at various model complexities. We can observe the property of learned coefficients when using Bernstein coefficients in the histograms at over-parameterized regime.



(a)

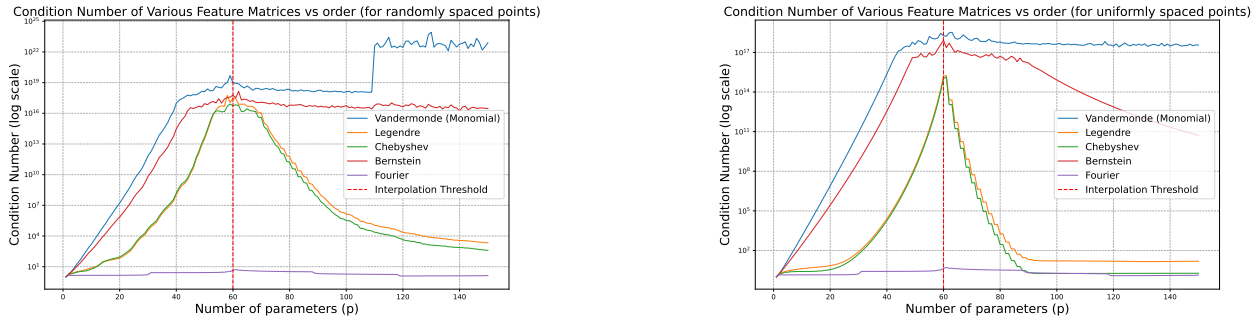


(b)

(e)

(d)

Figure 10: (a) In this figure we plot NRMSE error v/s Model complexity, in log scale. The problem is of approximating a function generated using Legendre basis of highest degree $d = 17$ in the domain $[0, 1]$. We sample $n = 50$ samples at random points. We use Chebyshev basis for defining the model, rather than Legendre. The figures in the inset showcase the approximations over various model complexities. It can be seen that this time the model does not generalize well beyond over-parameterized regime, unlike in the case of Bernstein basis. This showcases that not all polynomials can achieve the approximation capability in the over-parametrized regime. (b) In this subfigure we showcase histogram of learned coefficients (c^{opt}) and true coefficients (c^{true}) superimposed by the values of the training data points, at various model complexities. We can observe that the learned coefficients (c_{opt}) do not match the value of the training points unlike in the case of Bernstein basis.



(a) (b)

Figure 11: We consider various feature matrices used in the model and plot their condition number v/s parameters (highest degree of the basis) in the model. We consider $n = 60$ training points which are randomly spaced in (a) and uniformly spaced in (b). The red dashed line represents the interpolation threshold. The plot shows Fourier basis to be most stable while Monomial basis to be ill-conditioned even after the interpolation threshold. We can observe double descent behavior in the condition number for other bases.

5 Importance of stability and noise sensitivity

In Theorem 3.1 we sidelined the noise in sampling, let us understand the effect of noise in estimating the true coefficients. This is necessary for getting a complete picture.

According to Equation 10, the second term $\Phi_m^{train^\#}\bar{E}$ is the error in the learning of true coefficients if there is error in the inputs themselves. It can be shown that the ratio of the relative error in learned coefficients to the relative error in training data is bounded by the condition number Strang (2012) of the feature matrix itself. For derivation check Appendix C

$$\frac{\|\Phi_m^{train^\#}\bar{E}\|}{\|\bar{E}\|}\frac{\|\bar{y}^{train}\|}{\|\Phi_m^{train^\#}\bar{y}^{train}\|} \leq \kappa(\Phi_m^{train}). \tag{29}$$

Let us plot the condition number for various model feature matrices, in their respective domains they are defined in, with randomly as well as uniformly spaced inputs x^{train} in that domain Figure 11.

The error part in the test residual $\bar{R}_{test}$ due to sampling error in the training data is $\Phi_m^{test}\Phi_m^{train^\#}$. As $\Phi_m^{train^\#}$ appears in the error in the test residual, and $\kappa(\Phi_m^{train^\#}) = \kappa(\Phi_m^{train})$ we currently only care about its' condition number, by exploiting the sub-multiplicativity property of the condition numbers.

Let us understand the result in detail. It shows that the maximum of the ratio of the relative error in the learned coefficients to the relative error in the training data is highly regular for the Fourier feature matrix, while for the Monomial feature matrix it is highest, irrespective of the sampling choice. Moreover, we see that for the case of monomials it does not show significant double descent, while for others it is significant. We surprisingly observe double descent in the condition numbers of these feature matrices, just like the observation of test errors. A similar observation was also reported for *Random Matrices* and *Radial Kernel* in Poggio et al. (2019).

However, we should note that despite the model improving its condition number in over-parameterized, we cannot learn the true coefficients according to Theorem 3.1, in that regime. The condition number of the feature matrix is only important until and unless the conditions of Theorem 3.1 are met for out-of-domain approximation tasks.

Moreover, in the case of in-domain approximation, we observe that despite Chebyshev basis or Fourier features being highly conditioned in over-parameterized regimes, the condition number does not predict the

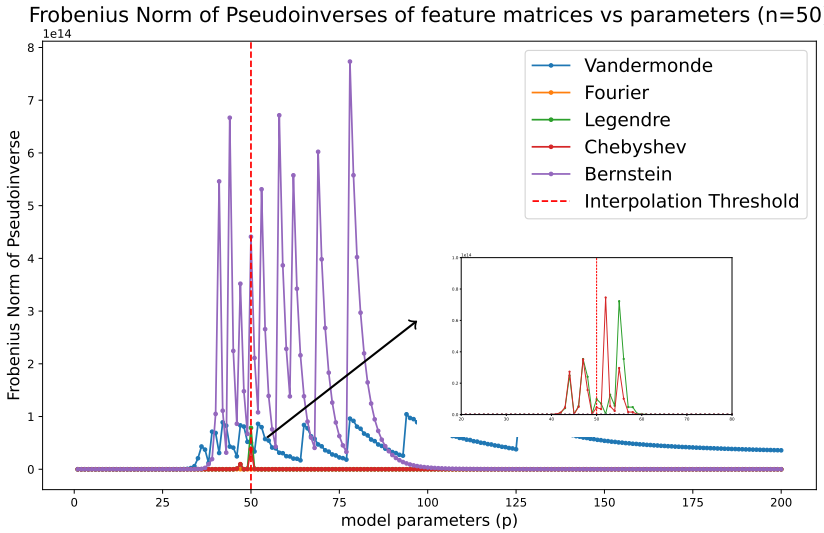


Figure 12: Frobenius norm of the pseudo-inverse of the feature matrices of the model v/s model parameters. The inputs are randomly sampled in the respective domains. The Frobenius norm measures the expected variance in the output with respect to error in the input, if it is a gaussian noise. We sample points It can be seen that it also showcases double descent behavior.

test error behavior. We can observe that a model with Bernstein features is able to approximate well in the over-parameterized regime Figure 9 rather than the Chebyshev features Figure 10. We observe similar behavior for Fourier basis Figure 1. This shows that the choice of basis for approximation is more important than the condition number of the feature matrices, both for in-domain and out-of-domain approximation tasks. Condition numbers, alternatively, only decide the effect of noise in training data on the approximation. Moreover, the ill-conditioning of the model can be regularized by different techniques once the model basis is chosen for specific task, taking into account the condition number of the feature matrix for that number of parameters and training data.

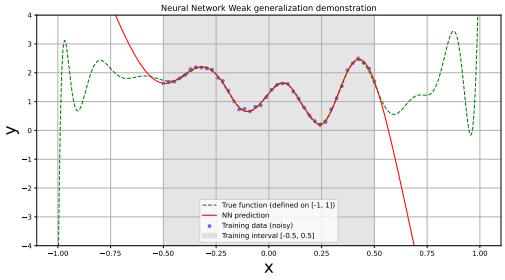
Condition number of the Hessian matrix of the loss function can affect the convergence properties while training using gradient descent, however we do not delve into this in this article Recht (2013), as we have used closed form solution using pseudo-inverse.

Let us now calculate the variance in the learned coefficients in comparison to the noise in the training data. If $\bar{E} \sim \mathcal{N}(0, \epsilon\mathbb{I})$ (i.e. Gaussian noise) and if $\bar{v} = \Phi_m^{train^\#} \bar{E}$ then

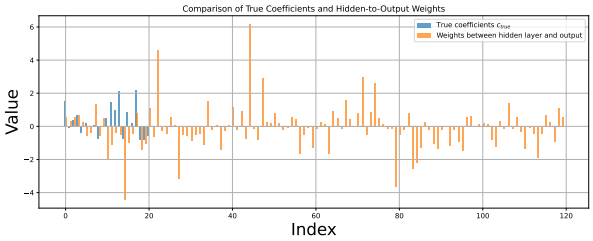
$$\mathbb{E}(\|\bar{v}\|^2) = \epsilon^2 Tr(\Phi_m^{train^\#}(\Phi_m^{train^\#})^T) = \epsilon^2 \|\Phi_m^{train^\#}\|_F^2 = \epsilon^2 \sum_{i=1}^{r} \sigma_i^2, \tag{30}$$

where $\|\Phi_m^{train^\#}\|_F$ is known as the *Frobenius Norm* of the matrix and it is equal to trace or root of sum of the square of singular values of that matrix Strang (2012). In contrast to the condition number we now have an idea of how much training data noise is mapped to learned coefficients. Let us find it for various features Figure 12.

The stability of the model through condition number and its relation to double descent is studied in Poggio et al. (2019); Chen and Schaeffer (2021); Rangamani et al. (2020).



(a)



(b)

Figure 13: (a) We consider a true function defined by a Legendre polynomial basis defined in the domain $[-1, 1]$. Let it be of order $d = 20$. We sample $n = 40$ training points in the domain $[-0.5, 0.5]$ at uniform spacing with Gaussian sampling noise mean 0 and noise standard deviation 0.05. The true coefficients are randomly generated. The plot represents the approximation with a single hidden layer neural network with $p = 120$ neurons. Which means the model is over-parameterized. We train using BFGS and it can be seen that the model fits the function very well (b) It plots the true coefficients with the weights of the neural network between the hidden layer and output. We can observe that they do not match. Indicating that feedforward neural networks generalize weakly.

6 Insights into deep learning

Let us understand if neural networks, when they approximate well, do they learn the coefficients defining the function (i.e. they obtain strict generalization or not). We use a single hidden layer neural network with one single input and output and '$tanh()$' activation function. It gives us a model function

$$y(x_i^{train}) = \sum_w \phi_m(x_i; w_j, b_j)c_j \tag{31}$$

$$\bar{y}^{train} = \Phi_m^{train}\bar{c} \tag{32}$$

This model is different from Equation 3, in the sense that in this case w is learnable and we have an added parameter b (i.e. bias). In the neural network model sense the weights between the input and the hidden layer are represented by w and the weights between hidden layer and output are represented by c. The function $\phi()$ in this model cannot be a polynomial as the degree of a polynomial basis cannot be learned through any known training algorithms like Gradient descent or Pseudo-Inverse.

To demonstrate that feedforward neural networks perform "weak generalization", let us consider a true function defined by Equation 1. Let us consider that it is defined by a Legendre polynomial basis defined in the domain $[-1, 1]$. Let it be of order $d = 20$. We sample $n = 40$ training points at uniform spacing with Gaussian sampling noise mean 0 and noise standard deviation 0.05. We consider $p = 120$ neurons, leading us to $3 \times p$ unknowns. The true coefficients are randomly generated. Let us perform training from the sampled data using “BFGS" method (rather than Gradient Descent) for better convergence to the solution. The results are shown in Figure 13.

It can be observed that the model generalizes very well in the training domain. However, it does not learn the true coefficients, leading us to conclude that it generalizes weakly according to definition Definition 2.2. Moreover, as we have proven that there is no restriction for weak generalization in over-parameterized regime, we can observe good in-domain approximation. However, according to Theorem 3.1 as we cannot truly learn features in over-parameterized regimes, it hints us towards the explanation that neural networks, in most cases, would not learn the true features of the target function, outside the training domain in the over-parameterized regime, despite generalizing well during in-domain approximation, because according to condition 5 of Theorem 3.1 the activation function of the neural network in most cases does not match the

features representing the true function. Neural network models like *SIREN* which use periodic activation functions, strengthen our viewpoint Ziyin et al. (2020); Sitzmann et al. (2020).

Our work is complementary to the study of deep learning models in *infinite-width limit* Jacot et al. (2018). However, our approach is a bottoms-up approach rather than top-down approach of the infinite-width framework. In that framework the neural network turns into a model which performs linear regression with non-linear fixed features. Let us denote the network model with parameters θ and input x as $f(x;\theta)$. The Taylor expansion around initial parameters θ_0 gives

$$f(x;\theta) = f(x;\theta_0) + \nabla_\theta f(x;\theta_0)^T(\theta-\theta_0) + \frac{1}{2!}(\theta-\theta_0)^T H_f(\tilde{\theta})(\theta-\theta_0) + \cdots \tag{33}$$

in the infinite width limit and with particular initializations the Jacobian term becomes constant in parameters and the Hessian term (and other higher order terms) becomes negligible Lee et al. (2019). This leads us to

$$f(x;\theta) \approx \sum_j \phi_j(x;\theta_0)c_j, \tag{34}$$

where the first term in Equation 33 can be considered a constant offset term and the Jacobian can be written as $\nabla_\theta f(x;\theta_0)^T \approx \phi_j(x;\theta_0)$. As $j \to \infty$, this model is always over-parameterized when trying to approximate a function, hence it leads to weak generalization, i.e. the network looses the ability to extrapolate and performs weak generalization.

7 Insights into Quantum Machine Learning

Generalization in quantum machine learning is also not completely understood Gil-Fuster et al. (2024). While bounds based on "conventional" statistical learning theory can be applied when the model class is suitably constrained Banchi et al. (2021; 2024); Caro et al. (2021; 2022); Du et al. (2021; 2023), they are known to be vacuous in the over-parametrized regime. On the other hand, benign overfitting and double descent were both observed in the quantum setting Peters and Schuld (2023); Kempkes et al. (2025).

It is important to remark that it is not fully clear yet whether over-parametrization is readily applicable to the quantum setting. This is due to several challenges: on one hand, training gets complicated because of the lack of a simple backpropagation algorithm Abbas et al. (2023); on the other hand, hardware constraints make extremely challenging to significantly increase either the depth or the width of quantum neural networks Schuld and Petruccione (2021). Even to address these challenges, a hardware-friendly approach to achieve over-parametrization with quantum models has been recently proposed Tognini et al. (2025), which is based on a mixture of quantum experts.

We now discuss the applications of our findings for the quantum machine learning community. First, our results are directly applicable to quantum extreme learning machines and quantum reservoir computing Innocenti et al. (2023); Mujal et al. (2021); Nakajima et al. (2019), since both of them can be expressed as in our Equation 3, where $\phi_m(x,w)$ are the "reservoir functions" that, in the quantum setting, are obtained by creating a suitably complex quantum state, e.g. by letting a simple quantum state evolve according to a complex Hamiltonian, and then performing fixed measurements. Here the parameters w define the internal dynamics of the quantum system, e.g. the Hamiltonian parameters, and the measurement settings. Our results can be applied to better understand which reservoir functions can guarantee better generalization and stability against noise.

Another application is for models constructed via classical shadows Jerbi et al. (2024), since some of them can also be expressed as a linear model Equation 3 where the basis functions are obtained by first loading data into quantum states (in a non-linear way) and then measuring certain observables.

Finally we discuss applications for quantum neural networks, which are among the most popular quantum machine learning models Schuld and Petruccione (2021). Even such models can be linearized via natural tangent kernel Liu et al. (2022); Shirai et al. (2024); Girardi and De Palma (2025) and thus be expressed as Equation 3, though the validity of such approximation is not completely understood. General quantum

neural networks defined through common quantum gates, even with reuploading layers Pérez-Salinas et al. (2020), can always be expressed as a partial Fourier series Schuld et al. (2021)

$$f(x, \theta) = \sum_{\omega \in \Omega} c_\omega(\theta) e^{i\omega x}, \tag{35}$$

where the expansion coefficients $c_\omega(\theta)$ depend in a complex way on the network parameters θ (e.g. qubit rotations) and the space of possible frequencies Ω may grow exponentially with the number of qubits and circuit layers. Because of Equation 35, most common quantum neural networks can be expressed as a linear model in a Fourier-like space, though with expansion coefficients that depend in a non-linear way on the trainable parameters. Nonetheless, quantum neural networks often display generalization properties similar to the ones that we observe with Fourier models, or the related Chebyshev polynomials, e.g. the "spiky" behaviour that may eventually lead to benign overfitting Peters and Schuld (2023).

8 Conclusion

Despite observation of Double Descent we could observe that models did not generalize well in the over-parameterized regime. We also try to understand that despite the problem being under-determined in this regime it approximates well. This led us to understand that there are two types of generalization possibilities. One, where we not only limit the residual error to a very small value but also learn the features of the true function globally, we call this as "strict generalization". On the other hand we can achieve very small residual error irrespective of learning the true features of the function to be approximated. We call this as "weak generalization". We argue that while with strict generalization we can achieve out-of-domain approximation of a continuous function, with weak generalization we only achieve in-domain approximation.

We derive the necessary conditions for strict generalization in Theorem 3.1. In doing so, we learn that strict generalization cannot be achieved in the over-parametrized regime for a fixed basis regression model Figure 2. It holds insights into the adaptive basis regression model, which is nothing but a vanilla neural network. We additionally define new thresholds, i.e. "Sampling threshold" (which indicates the minimum number of samples needed in comparison to the highest complexity of the function to be approximated) and also "Expressive threshold" (which indicates the minimum model complexity needed to approximate the true function). These thresholds bring fresh perspective to the theory of over-parameterized models.

We then study weak generalization, which can be achieved by both satisfying the conditions for strict generalization or without satisfying them, when performing in-domain approximation specifically. When we do not satisfy the conditions for strict generalization we can still achieve approximation even in the over-parametrized regime for in-domain approximation. We showcase this using a model which uses Bernstein basis. We remind that this is backed by the Weierstrass approximation theorem, which is well known in approximation theory but often ignored in Machine learning literature. We finally concluded by showcasing that not all polynomials (unlike Bernstein basis) can guarantee generalization in the over-parameterized regime. Which we can also observe in the Double Descent literature, where despite drop in the test error, the model does not generalize well. Moreover, we do not observe significant Double Descent in the test error in our approximation using the Bernstein basis, which emphasizes that observation of Double Descent is not absolutely necessary for in-domain approximation in over-parameterized regime.

Later we study the stability and noise sensitivity of the model for various basis. We realize the condition number of the Pseudo-inverse of the feature matrix (worst case amplification of noise) and its Frobenius norm (Expected variance in solution due to Gaussian noise), both showcase double descent. We also notice that the double descent in these metrics need not follow the same dynamics for test residuals. It leads us to showcase that a well-conditioned model is important but it is secondary to having the conditions for generalization being met, i.e. the choice of model basis with respect to the problem. Once the basis is chosen and there is enough expressivity and samples we can regularize the model to make it well-conditioned.

These observations lead us to gain insights in the success and limitations of deep-learning, despite them being over-parameterized. We showcase empirically that feed-forward neural networks achieve weak generalization in most cases (when the activation functions do not match the features of the target function). As weak

generalization has no restrictions, unlike strict generalization, to be obtained in the over-parameterized regime, we can observe it. As weak generalization restricts us to in-domain approximations only, we can observe why over-parameterized neural networks in those cases are not capable of feature learning outside training domain.

Our work is limited to 1-d continuous functions, though it can be generalized to higher dimensions. We would also need a rigorous exploration of the cases where the function to be approximated is discontinuous. Moreover, we only focus supervised learning model like fixed basis regression model. We choose this model as (a) it is fundamental to Machine learning (b) it is easy to find the closed form solution for this model without getting involved in the issues with optimization problems and (c) it is a special case of adaptive basis regression models, which are nothing but vanilla feed-forward neural networks. Having said this, there are some non-basis expansion models like symbolic regression Dick and Owen (2024) and self-supervised learning models Lupidi et al. (2023), which are speculated to show no double descent behavior. These speculations are empirical and a rigorous proof is lacking. We believe that adopting the concepts in this article to study feed-forward neural networks should be a natural extension. However, in doing so we would get involved in the complexities of backpropagation as no closed form solution method exists for such method. There are several alternatives to backpropagation in deep learning literature which can provide a way to study over-parameterized generalization in deep learning models Lillicrap et al. (2016); Hinton (2022); Li et al. (2025)

We expect that our work will bring a fresh perspective in the quest to understand generalization in over-parametrized regimes and close the gap on understanding generalization in deep learning, thus showing us direction to making even more efficient deep learning algorithms.

Acknowledgements

CW and LB have been supported by the European Union – NextGenerationEU under the National Recovery and Resilience Plan (PNRR) – Mission 4 Education and research – Component 2 From research to business – Investment 1.1 Notice Prin 2022 – DD N. 104 del 2/2/2022, from title "understanding the LEarning process of QUantum Neural networks (LeQun)", proposal code 2022WHZ5XH – CUP J53D23003890006. LB has also been supported the European Union's Horizon Europe research and innovation program under EPIQUE Project GA No. 101135288.

References

Christopher M Bishop and Nasser M Nasrabadi. *Pattern recognition and machine learning*, volume 4. Springer, 2006.

Stuart Geman, Elie Bienenstock, and René Doursat. Neural networks and the bias/variance dilemma. *Neural computation*, 4(1):1–58, 1992.

Robert Kleinberg and Kilian Weinberger. Lecture 13: Bias-variance tradeoff. `https://www.cs.cornell.edu/courses/cs4780/2015fa/web/lecturenotes/lecturenote13.html`, 2015. CS4780: Machine Learning, Fall 2015, Cornell University.

Preetum Nakkiran. More data can hurt for linear regression: Sample-wise double descent. *arXiv preprint arXiv:1912.07242*, 2019.

Preetum Nakkiran, Gal Kaplun, Yamini Bansal, Tristan Yang, Boaz Barak, and Ilya Sutskever. Deep double descent: Where bigger models and more data hurt. In *International Conference on Learning Representations (ICLR)*, 2021. URL `https://arxiv.org/abs/1912.02292`. arXiv:1912.02292.

Mikhail Belkin, Daniel Hsu, Siyuan Ma, and Saurabh Mandal. Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proceedings of the National Academy of Sciences*, 116(32):15849–15854, 2019. doi: 10.1073/pnas.1903070116.

Rylan Schaeffer, Mikail Khona, Zachary Robertson, Akhilan Boopathy, Kateryna Pistunova, Jason W Rocks, Ila Rani Fiete, and Oluwasanmi Koyejo. Double descent demystified: Identifying, interpreting & ablating the sources of a deep learning puzzle. *arXiv preprint arXiv:2303.14151*, 2023.

Cory Stephenson and Tyler Lee. When and how epochwise double descent happens. *arXiv preprint arXiv:2108.12006*, 2021.

Freeman Dyson et al. A meeting with enrico fermi. *Nature*, 427(6972):297–297, 2004.

Peter L Bartlett, Philip M Long, Gábor Lugosi, and Alexander Tsigler. Benign overfitting in linear regression. *Proceedings of the National Academy of Sciences*, 117(48):30063–30070, 2020. doi: 10.1073/pnas.1907378117.

Neil Mallinar, James Simon, Amirhesam Abedsoltan, Parthe Pandit, Misha Belkin, and Preetum Nakkiran. Benign, tempered, or catastrophic: Toward a refined taxonomy of overfitting. *Advances in neural information processing systems*, 35:1182–1195, 2022.

Behnam Neyshabur. Implicit regularization in deep learning. *arXiv preprint arXiv:1709.01953*, 2017.

Suriya Gunasekar, Jason D Lee, Daniel Soudry, and Nathan Srebro. Implicit bias of gradient descent on linear convolutional networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018. URL `https://arxiv.org/abs/1806.00468`. arXiv:1806.00468.

Daniel Soudry, Elad Hoffer, Mor Nacson, Suriya Gunasekar, and Nathan Srebro. The implicit bias of gradient descent on separable data. *Journal of Machine Learning Research*, 19(70):1–57, 2018. URL `http://jmlr.org/papers/v19/18-188.html`.

Blake Woodworth, Suriya Gunasekar, Jason D Lee, and Nathan Srebro. Kernel and rich regimes in overparameterized learning. In *International Conference on Machine Learning (ICML)*, 2020. URL `https://arxiv.org/abs/1906.05827`. arXiv:1906.05827.

Adam Power, Yuri Burda, Harri Edwards, Igor Babuschkin, and Vedant Misra. Grokking: Generalization beyond overfitting on algorithmic tasks, 2022. URL `https://arxiv.org/abs/2201.02177`. arXiv:2201.02177.

Robert de Mello Koch and Animik Ghosh. A two-phase perspective on deep learning dynamics. *arXiv e-prints*, pages arXiv–2504, 2025.

Radford M Neal. *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media, 2012.

Jaehoon Lee, Yasaman Bahri, Roman Novak, Samuel S Schoenholz, Jeffrey Pennington, and Jascha Sohl-Dickstein. Deep neural networks as gaussian processes. *arXiv preprint arXiv:1711.00165*, 2017.

Greg Yang. Wide feedforward or recurrent neural networks of any architecture are gaussian processes. *Advances in Neural Information Processing Systems*, 32, 2019.

Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. *Advances in neural information processing systems*, 31, 2018.

Nikhil Vyas, Yamini Bansal, and Preetum Nakkiran. Limitations of the ntk for understanding generalization in deep learning. *arXiv preprint arXiv:2206.10012*, 2022.

Vidya Muthukumar, Kailas Vodrahalli, Vignesh Subramanian, and Anant Sahai. Harmless interpolation of noisy data in regression. *IEEE Journal on Selected Areas in Information Theory*, 1(1):67–83, 2020.

Vignesh Subramanian. *When do machine learning models generalize well? A signal-processing perspective.* PhD thesis, University of California, Berkeley, USA, 2022. URL `https://www.escholarship.org/uc/item/0282k6n4`.

Evan Peters and Maria Schuld. Generalization despite overfitting in quantum machine learning models. *Quantum*, 7:1210, 2023.

Tomaso Poggio, Gil Kur, and Andrzej Banburski. Double descent in the condition number. *arXiv preprint arXiv:1912.06190*, 2019.

Bernhard Hofmann-Wellenhof and Helmut Moritz. *Physical geodesy*. Springer, 2006.

Michael J Kearns and Umesh Vazirani. *An introduction to computational learning theory*. MIT press, 1994.

Andrzej Ehrenfeucht, David Haussler, Michael Kearns, and Leslie Valiant. A general lower bound on the number of examples needed for learning. *Information and Computation*, 82(3):247–261, 1989.

Stéphane d'Ascoli, Levent Sagun, and Giulio Biroli. Triple descent and the two kinds of overfitting: Where & why do they appear? *Advances in neural information processing systems*, 33:3058–3069, 2020.

Walter Rudin. Principles of mathematical analysis. *3rd ed.*, 1976.

George Cybenko. Approximations by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2:183–192, 1989.

Gilbert Strang. *Linear algebra and its applications*. New York: Academic Press,, 2012.

Benjamin Recht. Optimization I. Workshop at Big Data Boot Camp, Simons Institute, September 2013. URL `https://simons.berkeley.edu/talks/optimization-i`. University of California, Berkeley.

Zhijun Chen and Hayden Schaeffer. Conditioning of random feature matrices: Double descent and generalization error. *arXiv preprint arXiv:2110.11477*, 2021.

Akshay Rangamani, Lorenzo Rosasco, and Tomaso Poggio. For interpolating kernel machines, minimizing the norm of the erm solution minimizes stability. *arXiv preprint arXiv:2006.15522*, 2020.

Liu Ziyin, Tilman Hartwig, and Masahito Ueda. Neural networks fail to learn periodic functions and how to fix it. *Advances in Neural Information Processing Systems*, 33:1583–1594, 2020.

Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. *Advances in neural information processing systems*, 33: 7462–7473, 2020.

Jaehoon Lee, Lechao Xiao, Samuel Schoenholz, Yasaman Bahri, Roman Novak, Jascha Sohl-Dickstein, and Jeffrey Pennington. Wide neural networks of any depth evolve as linear models under gradient descent. *Advances in neural information processing systems*, 32, 2019.

Elies Gil-Fuster, Jens Eisert, and Carlos Bravo-Prieto. Understanding quantum machine learning also requires rethinking generalization. *Nature Communications*, 15(1):2277, 2024.

Leonardo Banchi, Jason Pereira, and Stefano Pirandola. Generalization in quantum machine learning: A quantum information standpoint. *PRX Quantum*, 2(4):040321, 2021.

Leonardo Banchi, Jason Luke Pereira, Sharu Theresa Jose, and Osvaldo Simeone. Statistical complexity of quantum learning. *Advanced Quantum Technologies*, page 2300311, 2024.

Matthias C Caro, Elies Gil-Fuster, Johannes Jakob Meyer, Jens Eisert, and Ryan Sweke. Encoding-dependent generalization bounds for parametrized quantum circuits. *Quantum*, 5:582, 2021.

Matthias C Caro, Hsin-Yuan Huang, Marco Cerezo, Kunal Sharma, Andrew Sornborger, Lukasz Cincio, and Patrick J Coles. Generalization in quantum machine learning from few training data. *Nature communications*, 13(1):4919, 2022.

Yuxuan Du, Min-Hsiu Hsieh, Tongliang Liu, Shan You, and Dacheng Tao. Learnability of quantum neural networks. *PRX quantum*, 2(4):040337, 2021.

Yuxuan Du, Yibo Yang, Dacheng Tao, and Min-Hsiu Hsieh. Problem-dependent power of quantum neural networks on multiclass classification. *Physical Review Letters*, 131(14):140601, 2023.

Marie Kempkes, Aroosa Ijaz, Elies Gil-Fuster, Carlos Bravo-Prieto, Jakob Spiegelberg, Evert van Nieuwenburg, and Vedran Dunjko. Double descent in quantum machine learning. *arXiv preprint arXiv:2501.10077*, 2025.

Amira Abbas, Robbie King, Hsin-Yuan Huang, William J Huggins, Ramis Movassagh, Dar Gilboa, and Jarrod McClean. On quantum backpropagation, information reuse, and cheating measurement collapse. *Advances in Neural Information Processing Systems*, 36:44792–44819, 2023.

Maria Schuld and Francesco Petruccione. *Machine learning with quantum computers*, volume 676. Springer, 2021.

Paolo Alessandro Xavier Tognini, Leonardo Banchi, and Giacomo De Palma. Solving mnist with a globally trained mixture of quantum experts. *arXiv preprint arXiv:2505.14789*, 2025.

Luca Innocenti, Salvatore Lorenzo, Ivan Palmisano, Alessandro Ferraro, Mauro Paternostro, and G Massimo Palma. Potential and limitations of quantum extreme learning machines. *Communications Physics*, 6(1): 118, 2023.

Pere Mujal, Rodrigo Martínez-Peña, Johannes Nokkala, Jorge García-Beni, Gian Luca Giorgi, Miguel C Soriano, and Roberta Zambrini. Opportunities in quantum reservoir computing and extreme learning machines. *Advanced Quantum Technologies*, 4(8):2100027, 2021.

Kohei Nakajima, Keisuke Fujii, Makoto Negoro, Kosuke Mitarai, and Masahiro Kitagawa. Boosting computational power through spatial multiplexing in quantum reservoir computing. *Physical Review Applied*, 11 (3):034021, 2019.

Sofiene Jerbi, Casper Gyurik, Simon C Marshall, Riccardo Molteni, and Vedran Dunjko. Shadows of quantum machine learning. *Nature Communications*, 15(1):5676, 2024.

Junyu Liu, Francesco Tacchino, Jennifer R Glick, Liang Jiang, and Antonio Mezzacapo. Representation learning via quantum neural tangent kernels. *PRX Quantum*, 3(3):030323, 2022.

Norihito Shirai, Kenji Kubo, Kosuke Mitarai, and Keisuke Fujii. Quantum tangent kernel. *Physical Review Research*, 6(3):033179, 2024.

Filippo Girardi and Giacomo De Palma. Trained quantum neural networks are gaussian processes. *Communications in Mathematical Physics*, 406(4):1–146, 2025.

Adrián Pérez-Salinas, Alba Cervera-Lierta, Elies Gil-Fuster, and José I Latorre. Data re-uploading for a universal quantum classifier. *Quantum*, 4:226, 2020.

Maria Schuld, Ryan Sweke, and Johannes Jakob Meyer. Effect of data encoding on the expressive power of variational quantum-machine-learning models. *Physical Review A*, 103(3):032430, 2021.

Grant Dick and Caitlin Owen. Characterising the double descent of symbolic regression. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, pages 2050–2057, 2024.

Alisia Lupidi, Yonatan Gideoni, and Dulhan Jayalath. Does double descent occur in self-supervised learning? *arXiv preprint arXiv:2307.07872*, 2023.

Timothy P Lillicrap, Daniel Cownden, Douglas B Tweed, and Colin J Akerman. Random synaptic feedback weights support error backpropagation for deep learning. *Nature communications*, 7(1):13276, 2016.

Geoffrey Hinton. The forward-forward algorithm: Some preliminary investigations. *arXiv preprint arXiv:2212.13345*, 2(3):5, 2022.

Qinyu Li, Yee Whye Teh, and Razvan Pascanu. Noprop: Training neural networks without back-propagation or forward-propagation. *arXiv preprint arXiv:2503.24322*, 2025.

Gen Nakamura and Roland Potthast. *Inverse modeling: an introduction to the theory and methods of inverse problems and data assimilation*. IOP Publishing, 2015.

Alexander Forrester, Andras Sobester, and Andy Keane. *Engineering design via surrogate modelling: a practical guide*. John Wiley & Sons, 2008.

A Various feature matrices and sampling schemes

Let us define the various features used in the article, the domain in which they are defined and if they are orthogonal, the weight metric under which the basis is orthogonal.

Property	**Legendre Basis**	**Chebyshev Basis (First Kind)**	**Fourier Basis (Real Form)**	**Monomial Basis**
Definition Domain	$x \in [-1, 1]$	$x \in [-1, 1]$	$x \in [0, 1]$ or $[-\pi, \pi]$ (We feature scale to $[-1, 1]$)	$x \in \mathbb{R}$ (We feature scale to $[-1, 1]$)
Weight Function	$w(x) = 1$	$w(x) = \frac{1}{\sqrt{1-x^2}}$	$w(x) = 1$	Not orthogonal
Basis functions	$P_j(x) = 1, x, \frac{1}{2}(3x^2 - 1), \dots$	$T_j(x) = 1, x, 2x^2 - 1, \dots$	$F_j(x) = 1, \cos(2\pi x), \sin(2\pi x) \dots$	$M_j(x) = 1, x, x^2, ...$

Table 3: Comparison of Legendre, Chebyshev, and Fourier bases

Using the information above, various feature matrices can be generated. For example for the Legendre polynomial basis, the Feature matrix is given as

$$\Phi = \begin{bmatrix} P_0(x_1) & P_1(x_1) & P_2(x_1) & \cdots & P_{d-1}(x_1) \\ P_0(x_2) & P_1(x_2) & P_2(x_2) & \cdots & P_{d-1}(x_2) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ P_0(x_n) & P_1(x_n) & P_2(x_n) & \cdots & P_{d-1}(x_n) \end{bmatrix} = \begin{bmatrix} 1 & x_1 & \frac{1}{2}(3x_1^2 - 1) & \frac{1}{2}(5x_1^3 - 3x_1) & \frac{1}{8}(35x_1^4 - 30x_1^2 + 3) & \cdots \\ 1 & x_2 & \frac{1}{2}(3x_2^2 - 1) & \frac{1}{2}(5x_2^3 - 3x_2) & \frac{1}{8}(35x_2^4 - 30x_2^2 + 3) & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \\ 1 & x_n & \frac{1}{2}(3x_n^2 - 1) & \frac{1}{2}(5x_n^3 - 3x_n) & \frac{1}{8}(35x_n^4 - 30x_n^2 + 3) & \cdots \end{bmatrix} \tag{36}$$

Similarly, other feature matrices can be generated for the model. from Table 3.

B Proof of Corollary 3.2

We add the proof of Corollary 3.2 here, which is derived from the Theorem 3.1

Proof. In general, a vector $\bar{v} \in Ker(M)$, iff x is orthogonal to each row of M. This is according to the definition of orthogonality itself.

Now, according to Equation 16 for ensuring sustained strict generalization in the region between $d < p \leq n$ columns of $\Phi_t(x)$ need to be orthogonal to rows of B.

That means that 'Φ_t' and 'Φ_m' need to have an orthogonal basis for strict generalization in the strict generalization regime, for fixed basis regression model.

If the basis are not orthogonal, we cannot extend the strict generalization in this region and it would only occur at $p = d$ i.e. at expressive threshold (provided that we are above sampling threshold $n \geq d$), provided that other requirements are also fulfilled. □

C Proof of how worst case noise amplification is bounded by condition number

The ratio of the relative error in learned coefficients to the relative error in training data is given as

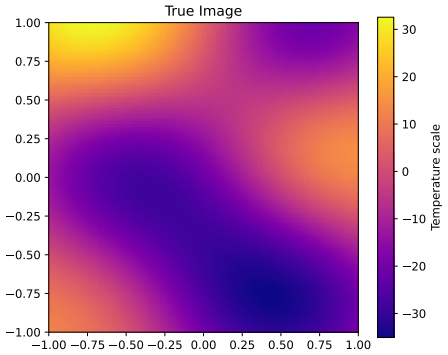


Figure 14: Representation of the temperature field given by Equation 42. WE scaled the boundaries to lie between $[-1, 1; -1, 1]$

$$\frac{\|\Phi_m^{train^{\#}}\bar{E}\|}{\|\bar{E}\|}\frac{\|\bar{y}^{train}\|}{\|\Phi_m^{train^{\#}}\bar{y}^{train}\|} = \frac{\|\Phi_m^{train^{\#}}\bar{E}\|}{\|\bar{E}\|}\frac{\|\Phi_m^{train}\bar{c}\|}{\|\bar{c}\|} \tag{37}$$

We used the relation Equation 3 in the previous equation. We also know that, $\frac{\|\Phi_m^{train^{\#}}\bar{E}\|}{\|\bar{E}\|} \leq \sigma_{max}(\Phi_m^{train^{\#}})$ and $\frac{\|\Phi_m^{train}\bar{c}\|}{\|\bar{c}\|} \leq \sigma_{max}(\Phi_m^{train})$, where $\sigma_{max}()$ represents the largest singular value of that matrix. The largest simgular value is nothing but the *2-norm* of the matrix. The definition of condition number of Φ_m^{train} is such that

$$\kappa(\Phi_m^{train}) = \sigma_{max}(\Phi_m^{train})\sigma_{max}(\Phi_m^{train^{\#}}) \tag{38}$$

which leads us to the relation that the ratio of the relative errors in the learned coefficients to the relative error in training data to be bounded by the condition number Equation 29

D Real world example

D.1 Strict Generalization and surrogate model reconstruction of a temperature field

As an illustrative real-world case of strict generalization, we consider the reconstruction of the steady-state temperature field on a 2-D surface. Temperature field reconstruction refers to estimating the spatial distribution of temperature over a domain using sparse or noisy observations. This arises in simulations, inverse problems, meteorology, and sensor networks, where full field measurements are often impractical.

This is formulated as an inverse problem, where we aim to infer hidden properties of a system from observed measurements, in contrast to a direct problem Nakamura and Potthast (2015). In our setup, we generate a synthetic temperature field (known to us but hidden from the reconstruction model), and task the model with recovering it from partial observations. Moreover, as an added complexity, we pose this as an extrapolation problem where we are unable to get enough observations in some parts of the problem, maybe due to limitations of the instrument.

We use a Chebyshev basis to first approximate the true field. This makes our problem that of a surrogate model Forrester et al. (2008). A surrogate model (or metamodel) is any computationally inexpensive approximation of a more complex, expensive-to-evaluate function. Instead of solving full physics-based PDEs (like Navier-Stokes, heat equation, etc.), we project the field onto a set of Chebyshev polynomials, reducing the problem to a small set of coefficients. We can then reconstruct or predict the field values efficiently using this surrogate.

Let us understand when we can surrogate a model. Let us consider a function $g(x)$ (for example e^{-x^2}). This function is not a basis expansion of Chebyshev basis. However, we can approximate it in this basis, calling the expansion as *surrogate* of the actual function. The Chebyshev expansion of a function $g(x)$ on $[-1, 1]$ is

$$g(x) \sim g_{surrogate}(x) = \frac{a_0}{2} + \sum_{i=1}^{\infty} a_i T_i(x), \tag{39}$$

where the coefficients are given by

$$a_i = \frac{2}{\pi} \int_{-1}^{1} \frac{g(x)\, T_i(x)}{\sqrt{1-x^2}}\, dx, \qquad i \geq 0. \tag{40}$$

Using numerical integration we can find the coefficient of expansion as Table 4. Note, the value of the coefficients reduces as the degree i increases of the expansion. In such scenarios we can safely surrogate the ground truth into another basis expansion.

We emphasize that this expansion is not the same as the approximation using regression methods. This is because for evaluating the expansion coefficients we need to know the ground truth as evident in Equation 40. In Pseudo-inverse methods of approximating the ground truth we have no knowledge of the true function.

Table 4: Chebyshev coefficients a_i for $f(x) = e^{-x^2}$ on $[-1, 1]$

i	a_i
0	1.290070540898
1	0.000000000000
2	-0.312841606370
3	0.000000000000
4	0.038704115419
5	0.000000000000
6	-0.003208683015
7	0.000000000000
8	0.000199919238
9	0.000000000000
10	-0.000009975211
11	0.000000000000
12	0.000000415017

The example of surrogate models also serves as a demonstration of the limits of extrapolation capabilities of the model, when the model features do not match the exact true features of the ground truth, as given by the conditions in Theorem 3.1.

In the 2-d case (assuming that the function represents a temperature field) we can expand the true function as given below

$$T(x, y) \approx \sum_{i=0}^{d} \sum_{j=0}^{e} c_{ij}^{true}\, T_i(x)\, T_j(y) \tag{41}$$

where $T(x, y)$ is approximated temperature field as a function of spatial coordinates x and y,. $T_i(x)$, $T_j(y)$ are Chebyshev polynomials of the first kind (orthogonal basis functions). c_{ij}^{true} are Spectral coefficients to be determined from data or projection. d, e are Order (or degree) of the polynomial expansion in x and y, respectively.

Now, to evaluate generalization performance in inverse modeling, we construct a synthetic temperature field $T(x, y)$ defined over a 2-D spatial domain. The field is designed to exhibit a combination of smooth global

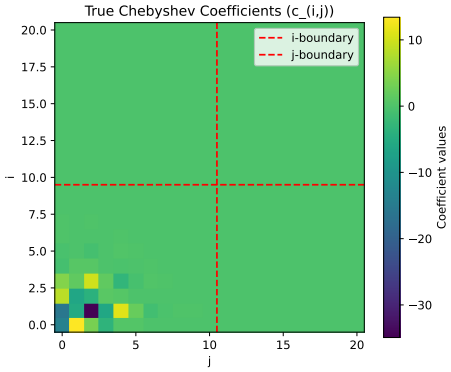


Figure 15: Values of the coefficients in Chebyshev expansion of the true temperature field. We can see that the values of $c_{i,j}$ are approximately equal to zero for values above $i = 10$ and $j = 10$ in the Equation 41. Hence, we consider the true coefficients for surrogate modeling to be $d = 10$ and $e = 10$.

trends, localized sources and sinks, spatial oscillations, and stochastic perturbations, mimicking complex real-world thermodynamic behavior. This field is known to us, but to the approximating algorithm later, we will feed the samples from this field, to showcase our understanding on approximation and strict generalization in the main article.

D.2 Temperature field generation

The temperature at location $(x, y) \in \mathbb{R}^2$ is given by:

$$\begin{aligned} T(x,y) = T_{\text{ambient}} &+ (T_s - T_{\text{ambient}})\, e^{-\lambda_c (x^2+y^2)} \\ &+ A_1 \sin(2x)\cos(3y)\, e^{-\lambda_1((x-2)^2+(y-1)^2)} \\ &+ A_2\, e^{-\lambda_2((x+3)^2+(y-4)^2)} \\ &+ A_3\, e^{-\lambda_x x^2 - \lambda_y y^2} \\ &+ \alpha_x x + \alpha_y y \\ &+ \beta \sin\left(0.5x^2 + 0.3y^2\right) \\ &+ \sigma \cdot \eta(x,y) \\ &+ A_{\text{sink}}\, e^{-\lambda_s((x-x_s)^2+(y-y_s)^2)} \end{aligned} \tag{42}$$

where:

- T_{ambient} is the ambient background temperature.
- T_s is the strength of the central Gaussian heat source, with decay rate λ_c.
- The second term models a spatially oscillatory source centered at $(0.3, 1)$, with amplitude A_1 and decay λ_1.
- The third term models a distant source centered at $(-3, 4)$, with amplitude A_2.
- The fourth term is an anisotropic Gaussian source with independent decay rates λ_x, λ_y, and amplitude $A_3 \sim \mathcal{U}(0, 1)$.
- The fifth term introduces a linear gradient field with coefficients $\alpha_x, \alpha_y \sim \mathcal{U}(0, 2)$.

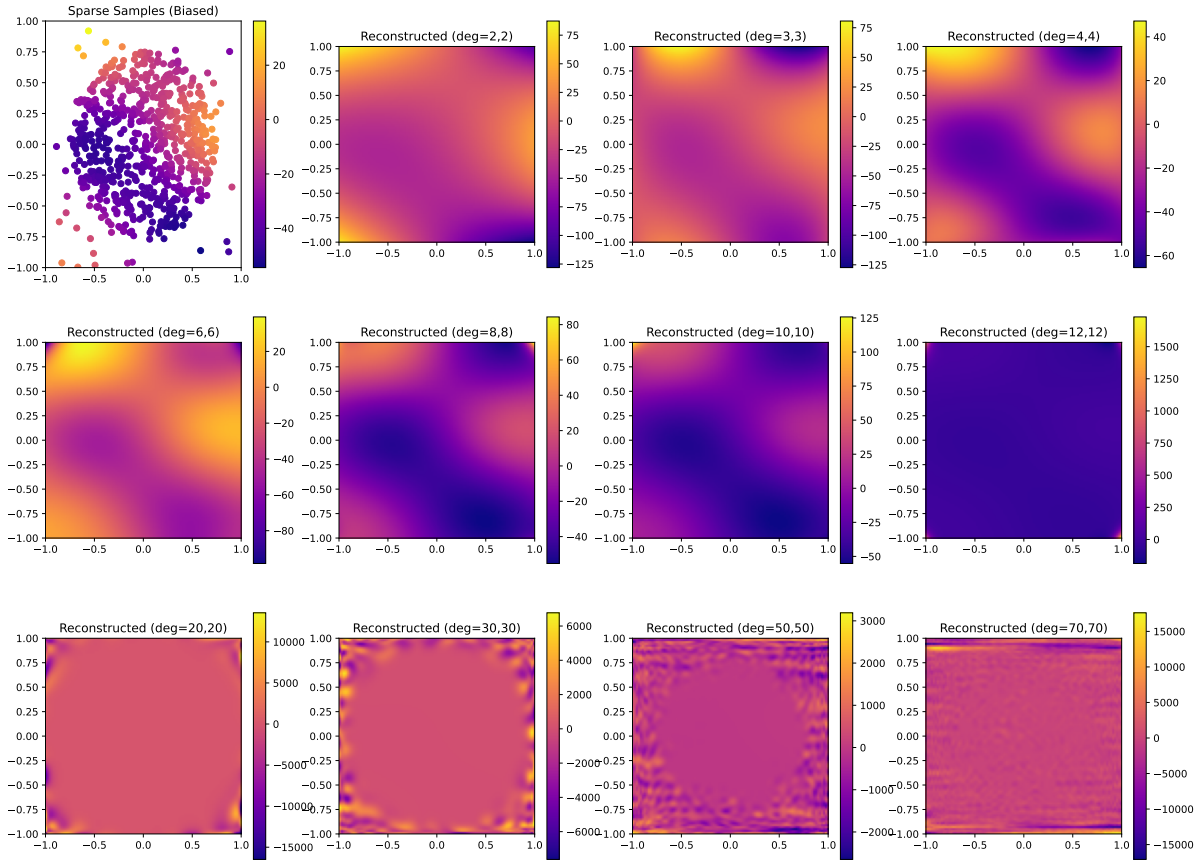


Figure 16: Reconstructed temperature fields from the sampled points. The samples are sparse outside a certain radius in this case it is $r = 0.75$. We can see that at $p, q = 10$, that is the model complexity, we see best reconstruction, the model also extrapolates in the sparsely sampled region. The model looses it reconstruction capability if we increase the parameters further than that.

- The sixth term is a smooth nonlinear perturbation with amplitude $\beta \sim \mathcal{U}(0, 1)$.
- The seventh term is spatial white noise, where $\eta(x, y) \sim \mathcal{N}(0, 1)$ and noise scale $\sigma \sim \mathcal{U}(0, 1)$.
- The final term is a temperature sink centered at $(x_s, y_s) = (0.25, -0.5)$, with amplitude $A_{\text{sink}} < 0$ and decay λ_s.

This construction ensures that the true field is challenging test case for generalization in reconstruction models. The field looks as Figure 14.

Let us see if surrogate modeling using Chebyshev basis works for such a temperature field. If we take this ground truth and decompose it into Chebyshev basis, we get Figure 15, using the method given in Equation 41.

It can be seen that $c_{i,j}$ values above $i = 10$ and $j = 10$ are essentially zero. Hence, we consider the true coefficients for surrogate modeling to be $d = 10$ and $e = 10$.

D.3 Reconstruction

Let us now reconstruct the field from samples taken from the temperature field. We emphasize that the approximation algorithm does not have the information of the true coefficients. We generate the learned coefficients using the training model using Chebyshev basis over the samples from the temperature field.

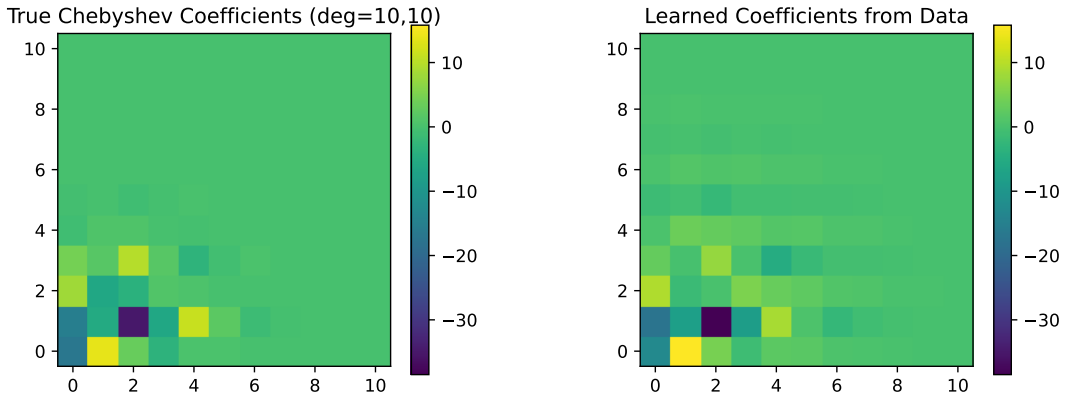


Figure 17: Comparison of true and learned coefficients. We see that the learned coefficients approximately match

Moreover, we consider that the samples are sparse in a region outside a certain radius from the center. Consider, it to be limitation due to instrumentation. It also makes this a problem of extrapolation or out-of-domain approximation.

We take $n_{train} = 700$ samples. We train using pseudo-inverse, as we require a closed-form solution for theoretical insights. The percentage of samples outside the radius $r = 0.75$ is 5% of the total samples. Let us see if we can reconstruct as well as extrapolate outside this radius.

We plot the reconstruction as we increase the number of parameters of the model p and q and compare the true coefficients to learned coefficients at $p = 10$ and $q = 10$, which is the actual effective order of the temperature field in the Chebyshev basis.

We consider the number of parameters according to the conditions of Theorem 3.1. We demonstrate the results in Figure 16

It can be seen that we have $n > d, e$ that is, we were well above the sampling threshold. The noise was sufficiently low, the surrogate approximation basis was appropriately chosen and that when the model parameters were $p, q = d, e$ we saw the best reconstruction as well as extrapolation in the sparsely sampled domain. We see that for lower model parameter values we do not have enough expressivity and for more parameter values we observe that the reconstruction degrades. This is the demonstration of the conditions given by Theorem 3.1. In fact, we also learn the exact true coefficients Figure 17.

If we just increase the order of the polynomial model. We observe that the test error is least at the order $(10, 10)$, as expected, as this is the true order of the surrogate ground truth of the true function Figure 18, or the expressive threshold of the surrogate. In short, when we approximate the true function $g(x)$ using a surrogate model $\tilde{g}(x)$, which itself is a function from a restricted function class. This introduces an approximation error due to the limited expressiveness of the surrogate itself. This adds extra error and it is very high in comparison to our observations of ideal strict generalization, as we are working with a surrogate model, where the ground truth is not exactly decomposed into a linear combination of certain basis Φ_t.

This finishes our demonstration of real world application demonstration of our findings. We used a surrogate model setting to also demonstrate the limitation in which the model basis do not match exactly to the ground truth in an ideal scenario.

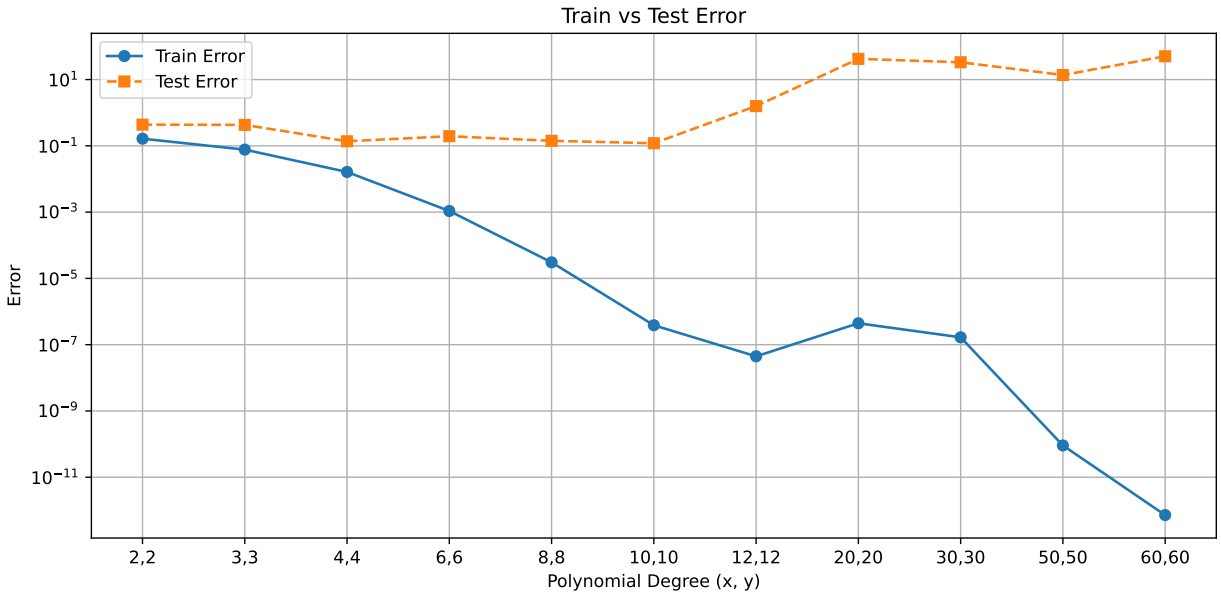


Figure 18: Train/Test error v/s order of the model basis of the surrogate for the ground truth.