Generalization Error of Generalized Linear Models in High Dimensions

Melikasadat Emami [1] **Mojtaba Sahraee-Ardakan** [1 2] **Parthe Pandit** [1 2] **Sundeep Rangan** [3] **Alyson K. Fletcher** [1 2]

Abstract

At the heart of machine learning lies the question of generalizability of learned rules over previously unseen data. While over-parameterized models based on neural networks are now ubiquitous in machine learning applications, our understanding of their generalization capabilities is incomplete and this task is made harder by the non-convexity of the underlying learning problems. We provide a general framework to characterize the asymptotic generalization error for single-layer neural networks (i.e., generalized linear models) with arbitrary non-linearities, making it applicable to regression as well as classification problems. This framework enables analyzing the effect of (i) over-parameterization and non-linearity during modeling; (ii) choices of loss function, initialization, and regularizer during learning; and (iii) mismatch between training and test distributions. As examples, we analyze a few special cases, namely linear regression and logistic regression. We are also able to rigorously and analytically explain the *double descent* phenomenon in generalized linear models.

1. Introduction

A fundamental goal of machine learning is *generalization*: the ability to draw inferences about unseen data from finite training examples. Methods to quantify the generalization error are therefore critical in assessing the performance of any machine learning approach.

This paper seeks to characterize the generalization error for a class of generalized linear models (GLMs) of the form

$$y = \phi_{\text{out}}(\langle \boldsymbol{x}, \boldsymbol{w}^0 \rangle, d), \tag{1}$$

where $\boldsymbol{x} \in \mathbb{R}^p$ is a vector of input features, y is a scalar output, $\boldsymbol{w}^0 \in \mathbb{R}^p$ are weights to be learned, $\phi_{\text{out}}(\cdot)$ is a known link function, and d is random noise. The notation $\langle \boldsymbol{x}, \boldsymbol{w}^0 \rangle$ denotes an inner product. We use the superscript "0" to denote the "true" values in contrast to estimated or postulated quantities. The output may be continuous or discrete to model either regression or classification problems.

We measure the generalization error in a standard manner: we are given training data $(\boldsymbol{x}_i, y_i)$, $i = 1, \ldots, N$ from which we learn some parameter estimate $\widehat{\boldsymbol{w}}$ via a regularized empirical risk minimization of the form

$$\widehat{\boldsymbol{w}} = \operatorname*{argmin}_{\boldsymbol{w}} F_{\text{out}}(\boldsymbol{y}, \mathbf{X}\boldsymbol{w}) + F_{\text{in}}(\boldsymbol{w}), \tag{2}$$

where $\mathbf{X} = [\boldsymbol{x}_1\, \boldsymbol{x}_2\, \ldots\, \boldsymbol{x}_N]^{\mathsf{T}}$, is the data matrix, F_{out} is some output loss function, and F_{in} is some regularizer on the weights. We are then given a new test sample, $\boldsymbol{x}_{\text{ts}}$, for which the true and predicted values are given by

$$y_{\text{ts}} = \phi_{\text{out}}(\langle \boldsymbol{x}_{\text{ts}}, \boldsymbol{w}^0 \rangle, d_{\text{ts}}), \quad \widehat{y}_{\text{ts}} = \phi(\langle \boldsymbol{x}_{\text{ts}}, \widehat{\boldsymbol{w}} \rangle), \tag{3}$$

where d_{ts} is the noise in the test sample, and $\phi(\cdot)$ is a postulated inverse link function that may be different from the true function $\phi_{\text{out}}(\cdot)$. The generalization error is then defined as the expectation of some expected loss between y_{ts} and $\widehat{y}_{\text{ts}}$ of the form

$$\mathbb{E}\, f_{\text{ts}}(y_{\text{ts}}, \widehat{y}_{\text{ts}}), \tag{4}$$

for some test loss function $f_{\text{ts}}(\cdot)$ such as squared error or prediction error.

Even for this relatively simple GLM model, the behavior of the generalization error is not fully understood. Recent works (Montanari et al., 2019; Deng et al., 2019; Mei & Montanari, 2019; Salehi et al., 2019) have characterized the generalization error of various linear models for classification and regression in certain large random problem instances. Specifically, the number of samples N and number of features p both grow without bound with their ratio satisfying $p/N \to \beta \in (0, \infty)$, and the samples in the training data $\boldsymbol{x}_i$ are drawn randomly. In this limit, the generalization error can be exactly computed. The analysis can

Code available at: https://github.com/melikaemami/Generalization-Error-of-GLMs

[1]Department of Electrical and Computer Engineering, University of California, Los Angeles, Los Angeles, USA [2]Department of Statistics, University of California, Los Angeles, Los Angeles, USA [3]Department of Electrical and Computer Engineering, New York University, Brooklyn, New York, USA. Correspondence to: Melikasadat Emami <emami@ucla.edu>.

Proceedings of the 37th International Conference on Machine Learning, Online, PMLR 119, 2020. Copyright 2020 by the author(s).

explain the so-called *double descent* phenomena (Belkin et al., 2019a): in highly under-regularized settings, the test error may initially *increase* with the number of data samples N before decreasing. Perhaps the first empirical evidence of the double descent curve can be traced back to (Bös & Opper, 1997). See the prior work section below for more details.

Summary of Contributions. Our main result (Theorem 1) provides a procedure for exactly computing the asymptotic value of the generalization error (4) for GLM models in a certain random high-dimensional regime called the Large System Limit (LSL). The procedure enables the generalization error to be related to key problem parameters including the sampling ratio $\beta = p/N$, the regularizer, the output function, and the distributions of the true weights and noise. Importantly, our result holds under very general settings including: (i) arbitrary test metrics f_{ts}; (ii) arbitrary training loss functions F_{out} as well as decomposable regularizers F_{in}; (iii) arbitrary link functions ϕ_{out}; (iv) correlated covariates $\boldsymbol{x}$; (v) underparameterized ($\beta < 1$) and overparameterized regimes ($\beta > 1$); and (vi) distributional mismatch in training and test data. Section 4 discusses in detail the general assumptions on the quantities f_{ts}, F_{out}, F_{in}, and ϕ_{out} under which Theorem 1 holds.

Prior Work. Many recent works characterize generalization error of various machine learning models, including special cases of the GLM model considered here. For example, the precise characterization for asymptotics of prediction error for least squares regression has been provided in (Belkin et al., 2019b; Hastie et al., 2019; Muthukumar et al., 2019). The former confirmed the double descent curve of (Belkin et al., 2019a) under a Fourier series model and a noisy Gaussian model for data in the over-parameterized regime. The latter also obtained this scenario under both linear and non-linear feature models for ridge regression and min-norm least squares using random matrix theory. Also, (Advani & Saxe, 2017) studied the same setting for deep linear and shallow non-linear networks.

The analysis of the the generalization for max-margin linear classifiers in the high dimensional regime has been done in (Montanari et al., 2019). The exact expression for asymptotic prediction error is derived and in a specific case for two-layer neural network with random first-layer weights, the double descent curve was obtained. A similar double descent curve for logistic regression as well as linear discriminant analysis has been reported by (Deng et al., 2019). Random feature learning in the same setting has also been studied for ridge regression in (Mei & Montanari, 2019). The authors have, in particular, shown that highly over-parametrized estimators with zero training error are statistically optimal at high signal-to-noise ratio (SNR). The asymptotic performance of regularized logistic regression in high dimensions is studied in (Salehi et al., 2019) using the Convex Gaussian Min-max Theorem in the under-parametrized regime. The results in the current paper can consider all these models as special cases. Bounds on the generalization error of over-parametrized linear models are also given in (Bartlett et al., 2019; Neyshabur et al., 2018).

Although this paper and several other recent works consider only simple linear models and GLMs, much of the motivation is to understand generalization in deep neural networks where classical intuition may not hold (Belkin et al., 2018; Zhang et al., 2016; Neyshabur et al., 2018). In particular, a number of recent papers have shown the connection between neural networks in the over-parametrized regime and kernel methods. The works (Daniely, 2017; Daniely et al., 2016) showed that gradient descent on over-parametrized neural networks learns a function in the RKHS corresponding to the random feature kernel. Training dynamics of over-parametrized neural networks has been studied by (Jacot et al., 2018; Du et al., 2018; Arora et al., 2019; Allen-Zhu et al., 2019), and it is shown that the function learned is in an RKHS corresponding to the neural tangent kernel.

Approximate Message Passing. Our key tool to study the generalization error is approximate message passing (AMP), a class of inference algorithms originally developed in (Donoho et al., 2009; 2010; Bayati & Montanari, 2011) for compressed sensing. We show that the learning problem for the GLM can be formulated as an inference problem on a certain multi-layer network. Multi-layer AMP methods (He et al., 2017; Manoel et al., 2018; Fletcher et al., 2018; Pandit et al., 2019) can then be applied to perform the inference. The specific algorithm we use in this work is the multi-layer vector AMP (ML-VAMP) algorithm of (Fletcher et al., 2018; Pandit et al., 2019) which itself builds on several works (Opper & Winther, 2005; Fletcher et al., 2016; Rangan et al., 2019; Cakmak et al., 2014; Ma & Ping, 2017). The ML-VAMP algorithm is not necessarily the most computationally efficient procedure for the minimization (2). For our purposes, the key property is that ML-VAMP enables exact predictions of its performance in the large system limit. Specifically, the error of the algorithm estimates in each iteration can be predicted by a set of deterministic recursive equations called the *state evolution* or SE. The fixed points of these equations provide a way of computing the asymptotic performance of the algorithm. In certain cases, the algorithm can be proven to be Bayes optimal (Reeves, 2017; Gabrié et al., 2018; Barbier et al., 2019; Advani & Ganguli, 2016).

This approach of using AMP methods to characterize the generalization error of GLMs was also explored in (Barbier et al., 2019) for i.i.d. distributions on the data. The explicit formulae for the asymptotic mean squared error for the regularized linear regression with rotationally invarient data matrices is proved in (Gerbelot et al., 2020). The ML-VAMP method in this work enables extensions to correlated features and to mismatch between training and test distributions.

2. Generalization Error: System Model

We consider the problem of estimating the weights $\boldsymbol{w}$ in the GLM model (1). As stated in the Introduction, we suppose we have training data $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^N$ arranged as $\mathbf{X} := [\boldsymbol{x}_1\, \boldsymbol{x}_2 \ldots \boldsymbol{x}_N]^\intercal \in \mathbb{R}^{N\times p}$, $\boldsymbol{y} := [y_1\, y_2 \ldots y_N]^\intercal \in \mathbb{R}^N$. Then we can write

$$\boldsymbol{y} = \boldsymbol{\phi}_{\rm out}(\mathbf{X}\boldsymbol{w}^0, \mathbf{d}), \tag{5}$$

where $\boldsymbol{\phi}_{\rm out}(\mathbf{z}, \mathbf{d})$ is the vector-valued function such that $[\boldsymbol{\phi}_{\rm out}(\mathbf{z}, \mathbf{d})]_n = \phi_{\rm out}(z_n, d_n)$ and $\{d_n\}_{n=1}^N$ are general noise.

Given the training data $(\mathbf{X}, \boldsymbol{y})$, we consider estimates of $\boldsymbol{w}^0$ given by a regularized empirical risk minimization of the form (2). We assume that the loss function $F_{\rm out}$ and regularizer $F_{\rm in}$ are separable functions, i.e., one can write

$$F_{\rm out}(\boldsymbol{y}, \mathbf{z}) = \sum_{n=1}^N f_{\rm out}(y_n, z_n), \quad F_{\rm in}(\boldsymbol{w}) = \sum_{j=1}^p f_{\rm in}(w_j), \tag{6}$$

for some functions $f_{\rm out} : \mathbb{R}^2 \to \mathbb{R}$ and $f_{\rm in} : \mathbb{R} \to \mathbb{R}$. Many standard optimization problems in machine learning can be written in this form: logistic regression, support vector machines, linear regression, Poisson regression.

Large System Limit: We follow the LSL analysis of (Bayati & Montanari, 2011) commonly used for analyzing AMP-based methods. Specifically, we consider a sequence of problems indexed by the number of training samples N. For each N, we suppose that the number of features $p = p(N)$ grows linearly with N, i.e.,

$$\lim_{N\to\infty} \frac{p(N)}{N} \to \beta \tag{7}$$

for some constant $\beta \in (0, \infty)$. Note that $\beta > 1$ corresponds to the over-parameterized regime and $\beta < 1$ corresponds to the under-parameterized regime.

True parameter: We assume the *true* weight vector $\boldsymbol{w}^0$ has components whose empirical distribution converges as

$$\lim_{N\to\infty} \{w_n^0\} \overset{PL(2)}{=} W^0, \tag{8}$$

for some limiting random variable W^0. The precise definition of empirical convergence is given in Appendix A. It means that the empirical distribution $\frac{1}{p}\sum_{i=1}^p \delta_{w_i}$ converges, in the Wasserstein-2 metric (see Chap. 6 (Villani, 2008)), to the distribution of the finite-variance random variable W^0. Importantly, the limit (8) will hold if the components $\{w_i^0\}_{i=1}^p$ are drawn i.i.d. from the distribution of W^0 with $\mathbb{E}(W^0)^2 < \infty$. However, as discussed in Appendix A, the convergence can also be satisfied by correlated sequences and deterministic sequences.

Training data input: For each N, we assume that the training input data samples, $\boldsymbol{x}_i \in \mathbb{R}^p$, $i = 1, \ldots, N$, are i.i.d. and drawn from a p-dimensional Gaussian distribution with zero mean and covariance $\boldsymbol{\Sigma}_{\rm tr} \in \mathbb{R}^{p\times p}$. The covariance can capture the effect of features being correlated. We assume the covariance matrix has an eigenvalue decomposition,

$$\boldsymbol{\Sigma}_{\rm tr} = \tfrac{1}{p}\mathbf{V}_0^\intercal {\rm diag}(\mathbf{s}_{\rm tr}^2)\mathbf{V}_0, \tag{9}$$

where $\mathbf{s}_{\rm tr}^2$ are the eigenvalues of $\boldsymbol{\Sigma}_{\rm tr}$ and $\mathbf{V}_0 \in \mathbb{R}^{p\times p}$ is the orthogonal matrix of eigenvectors. The scaling $\frac{1}{p}$ ensures that the total variance of the samples, $\mathbb{E}\|\boldsymbol{x}_i\|^2$, does not grow with N. We will place a certain random model on $\mathbf{s}_{\rm tr}$ and $\mathbf{V}_0$ momentarily.

Using the covariance (9), we can write the data matrix as

$$\mathbf{X} = \mathbf{U}\, {\rm diag}(\mathbf{s}_{\rm tr})\mathbf{V}_0, \tag{10}$$

where $\mathbf{U} \in \mathbb{R}^{N\times p}$ has entries drawn i.i.d. from $\mathcal{N}(0, \frac{1}{p})$. For the purpose of analysis, it is useful to express the matrix $\mathbf{U}$ in terms of its SVD:

$$\mathbf{U} = \mathbf{V}_2\mathbf{S}_{\rm mp}\mathbf{V}_1, \quad \mathbf{S}_{\rm mp} := \begin{bmatrix} {\rm diag}(\mathbf{s}_{\rm mp}) & \mathbf{0} \\ \mathbf{0} & * \end{bmatrix} \tag{11}$$

where $\mathbf{V}_1 \in \mathbb{R}^{N\times N}$ and $\mathbf{V}_2 \in \mathbb{R}^{p\times p}$ are orthogonal and $\mathbf{S}_{\rm mp} \in \mathbb{R}^{N\times p}$ with non-zero entries $\mathbf{s}_{\rm mp} \in \mathbb{R}^{\min\{N,p\}}$ only along the principal diagonal. $\mathbf{s}_{\rm mp}$ are the singular values of $\mathbf{U}$. A standard result of random matrix theory is that, since $\mathbf{U}$ is i.i.d. Gaussian with entries $\mathcal{N}(0, \frac{1}{p})$, the matrices $\mathbf{V}_1$ and $\mathbf{V}_2$ are Haar-distributed on the group of orthogonal matrices and $\mathbf{s}_{\rm mp}$ is such that

$$\lim_{N\to\infty} \{s_{{\rm mp},i}\} \overset{PL(2)}{=} S_{\rm mp}, \tag{12}$$

where $S_{\rm mp} \geq 0$ is a non-negative random variable such that $S_{\rm mp}^2$ satisfies the Marcencko-Pastur distribution. Details on this distribution are in Appendix H.

Training data output: Given the input data $\mathbf{X}$, we assume that the training outputs $\boldsymbol{y}$ are generated from (5), where the noise $\mathbf{d}$ is independent of $\mathbf{X}$ and has an empirical distribution which converges as

$$\lim_{N\to\infty} \{d_i\} \overset{PL(2)}{=} D. \tag{13}$$

Again, the limit (13) will be satisfied if $\{d_i\}_{i=1}^N$ are i.i.d. draws of random variable D with bounded second moments.

Test data: To measure the generalization error, we assume now that we are given a test point $\boldsymbol{x}_{\rm ts}$, and we obtain the true output y_{ts} and predicted output $\widehat{y}_{\text{ts}}$ given by (3). We assume that the test data inputs are also Gaussian, i.e.,

$$\boldsymbol{x}_{\text{ts}}^{\mathsf{T}} = \mathbf{u}^{\mathsf{T}}\text{diag}(\mathbf{s}_{\text{ts}})\mathbf{V}_0, \tag{14}$$

where $\mathbf{u} \in \mathbb{R}^p$ has i.i.d. Gaussian components, $\mathcal{N}(0, \frac{1}{p})$, and $\mathbf{s}_{\text{ts}}$ and $\mathbf{V}_0$ are the eigenvalues and eigenvectors of the test data covariance matrix. That is, the test data sample has a covariance matrix

$$\boldsymbol{\Sigma}_{\text{ts}} = \tfrac{1}{p}\mathbf{V}_0^{\mathsf{T}}\text{diag}(\mathbf{s}_{\text{ts}}^2)\mathbf{V}_0. \tag{15}$$

In comparison to (9), we see that we are assuming that the eigenvectors of the training and test data are the same, but the eigenvalues may be different. In this way, we can capture distributional mismatch between the training and test data. For example, we will be able to measure the generalization error when the test sample is outside a subspace explored by the training data.

To capture the relation between the training and test distributions, we assume that components of $\mathbf{s}_{\text{tr}}$ and $\mathbf{s}_{\text{ts}}$ converge as

$$\lim_{N\to\infty} \{(s_{\text{tr},i}, s_{\text{ts},i})\} \overset{PL(2)}{=} (S_{\text{tr}}, S_{\text{ts}}), \tag{16}$$

to some non-negative, bounded random vector $(S_{\text{tr}}, S_{\text{ts}})$. The joint distribution on $(S_{\text{tr}}, S_{\text{ts}})$ captures the relation between the training and test data.

When $S_{\text{tr}} = S_{\text{ts}}$, our model corresponds to the case when the training and test distribution are matched. Isotropic Gaussian features in both training and test data correspond to covariance matrices $\boldsymbol{\Sigma}_{\text{tr}} = \frac{1}{p}\sigma_{\text{tr}}^2\mathbf{I}$, $\boldsymbol{\Sigma}_{\text{ts}} = \frac{1}{p}\sigma_{\text{ts}}^2\mathbf{I}$, which can be modeled as $S_{\text{tr}} = \sigma_{\text{tr}}$, $S_{\text{ts}} = \sigma_{\text{ts}}$. We also require that the matrix $\mathbf{V}_0$ is uniformly distributed on the set of $p \times p$ orthogonal matrices.

Generalization error: From the training data, we obtain an estimate $\widehat{\boldsymbol{w}}$ via a regularized empirical risk minimization (2). Given a test sample $\boldsymbol{x}_{\text{ts}}$ and parameter estimate $\widehat{\boldsymbol{w}}$, the true output y_{ts} and predicted output $\widehat{y}_{\text{tr}}$ are given by equation (3). We assume the test noise is distributed as $d_{\text{ts}} \sim D$, following the same distribution as the training data. The postulated inverse-link function $\phi(\cdot)$ in (3) may be different from the true inverse-link function $\phi_{\text{out}}(\cdot)$.

The generalization error is defined as the asymptotic expected loss,

$$\mathcal{E}_{\text{ts}} := \lim_{N\to\infty} \mathbb{E} f_{\text{ts}}(\widehat{y}_{\text{ts}}, y_{\text{ts}}), \tag{17}$$

where $f_{\text{ts}}(\cdot)$ is some loss function relevant for the test error (which may be different from the training loss). The expectation in (17) is with respect to the randomness in the training as well as test data, and the noise. Our main result provides a formula for the generalization error (17).

3. Learning GLMs via ML-VAMP

There are many methods for solving the minimization problem (2). We apply the ML-VAMP algorithm of (Fletcher et al., 2018; Pandit et al., 2019). This algorithm is not necessarily the most computationally efficient method. For our purposes, however, the algorithm serves as a constructive proof technique, i.e., it enables exact predictions for generalization error in the LSL as described above. Moreover, in the case when loss function (2) is strictly convex, the problem has a unique global minimum, whereby the generalization error of this minimum is agnostic to the choice of algorithm used to find this minimum. To that end, we start by reformulating (2) in a form that is amicable to the application of ML-VAMP, Algorithm 1.

Multi-Layer Representation. The first step in applying ML-VAMP to the GLM learning problem is to represent the mapping from the true parameters $\boldsymbol{w}^0$ to the output $\boldsymbol{y}$ as a certain multi-layer network. We combine (5), (10) and (11), so that the mapping $\boldsymbol{w}^0 \mapsto \boldsymbol{y}$ can be written as the following sequence of operations (as illustrated in Fig. 1):

$$\begin{aligned} \mathbf{z}_0^0 &:= \boldsymbol{w}^0, & \mathbf{p}_0^0 &:= \mathbf{V}_0\mathbf{z}_0^0, \\ \mathbf{z}_1^0 &:= \boldsymbol{\phi}_1(\mathbf{p}_0^0, \boldsymbol{\xi}_1), & \mathbf{p}_1^0 &:= \mathbf{V}_1\mathbf{z}_1^0, \\ \mathbf{z}_2^0 &:= \boldsymbol{\phi}_2(\mathbf{p}_1^0, \boldsymbol{\xi}_2), & \mathbf{p}_2^0 &:= \mathbf{V}_2\mathbf{z}_2^0, \\ \mathbf{z}_3^0 &:= \boldsymbol{\phi}_3(\mathbf{p}_2^0, \boldsymbol{\xi}_3) = \boldsymbol{y}, \end{aligned} \tag{18}$$

where $\boldsymbol{\xi}_\ell$ are the following vectors:

$$\boldsymbol{\xi}_1 := \mathbf{s}_{\text{tr}}, \quad \boldsymbol{\xi}_2 := \mathbf{s}_{\text{mp}}, \quad \boldsymbol{\xi}_3 := \mathbf{d}, \tag{19}$$

and the functions $\boldsymbol{\phi}_\ell(\cdot)$ are given by

$$\begin{aligned} \boldsymbol{\phi}_1(\mathbf{p}_0, \mathbf{s}_{\text{tr}}) &:= \text{diag}(\mathbf{s}_{\text{tr}})\mathbf{p}_0, \\ \boldsymbol{\phi}_2(\mathbf{p}_1, \mathbf{s}_{\text{mp}}) &:= \mathbf{S}_{\text{mp}}\mathbf{p}_1, \\ \boldsymbol{\phi}_3(\mathbf{p}_2, \mathbf{d}) &:= \boldsymbol{\phi}_{\text{out}}(\mathbf{p}_2, \mathbf{d}). \end{aligned} \tag{20}$$

We see from Fig. 1 that the mapping of true parameters $\boldsymbol{w}^0 = \mathbf{z}_0^0$ to the observed response vector $\boldsymbol{y} = \mathbf{z}_3^0$ is described by a multi-layer network of alternating orthogonal operators $\mathbf{V}_\ell$ and non-linear functions $\boldsymbol{\phi}_\ell(\cdot)$. Let $L = 3$ denote the number of layers in this multi-layer network.

The minimization (2) can also be represented using a similar signal flow graph. Given a parameter candidate $\boldsymbol{w}$, the mapping $\boldsymbol{w} \mapsto \mathbf{X}\boldsymbol{w}$ can be written using the sequence of vectors

$$\begin{aligned} \mathbf{z}_0 &:= \boldsymbol{w}, & \mathbf{p}_0 &:= \mathbf{V}_0\mathbf{z}_0, \\ \mathbf{z}_1 &:= \mathbf{S}_{\text{tr}}\mathbf{p}_0, & \mathbf{p}_1 &:= \mathbf{V}_1\mathbf{z}_1, \\ \mathbf{z}_2 &:= \mathbf{S}_{\text{mp}}\mathbf{p}_1, & \mathbf{p}_2 &:= \mathbf{V}_2\mathbf{z}_2 = \mathbf{X}\boldsymbol{w}. \end{aligned} \tag{21}$$

There are $L = 3$ steps in this sequence, and we let

$$\mathbf{z} = \{\mathbf{z}_0, \mathbf{z}_1, \mathbf{z}_2\}, \quad \mathbf{p} = \{\mathbf{p}_0, \mathbf{p}_1, \mathbf{p}_2\}$$

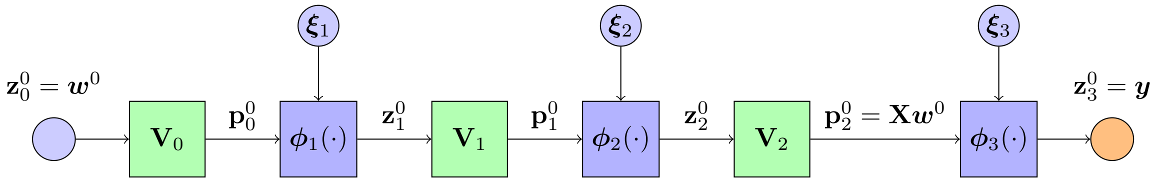
$\mathbf{z}_0^0 = \boldsymbol{w}^0$
$\mathbf{V}_0$
$\mathbf{p}_0^0$
$\boldsymbol{\xi}_1$
$\boldsymbol{\phi}_1(\cdot)$
$\mathbf{z}_1^0$
$\mathbf{V}_1$
$\mathbf{p}_1^0$
$\boldsymbol{\xi}_2$
$\boldsymbol{\phi}_2(\cdot)$
$\mathbf{z}_2^0$
$\mathbf{V}_2$
$\mathbf{p}_2^0 = \mathbf{X}\boldsymbol{w}^0$
$\boldsymbol{\xi}_3$
$\boldsymbol{\phi}_3(\cdot)$
$\mathbf{z}_3^0 = \boldsymbol{y}$

Figure 1. **Sequence flow representing the mapping from the unknown parameter values $\boldsymbol{w}^0$ to the vector of responses $\boldsymbol{y}$ on the training features X.** $\mathbf{V}_\ell$ blocks represent multiplication by orthogonal operators and $\boldsymbol{\phi}_\ell(\cdot)$ blocks are non-linear functions acting coordinatewise. For the GLM learning problem we have $\boldsymbol{\xi}_1 = \mathbf{s}_{\rm tr}$ and, $\boldsymbol{\xi}_2 = \mathbf{s}_{\rm mp}, \boldsymbol{\xi}_3 = \mathbf{d}$. Also, $\boldsymbol{\phi}_1(\mathbf{p}_0, \mathbf{s}_{\rm tr}) = \mathrm{diag}(\mathbf{s}_{\rm tr})\mathbf{p}_0$, $\boldsymbol{\phi}_2(\mathbf{p}_1, \mathbf{s}_{\rm mp}) = \mathrm{diag}(\mathbf{s}_{\rm mp})\mathbf{p}_1$, and $\boldsymbol{\phi}_3(\mathbf{p}_2, \mathbf{d}) = \boldsymbol{\phi}_{\rm out}(\mathbf{p}_2, \mathbf{d})$.

denote the sets of vectors across the steps. The minimization in (2) can then be written in the following equivalent form:

$$\begin{aligned} \min_{\mathbf{z},\mathbf{p}} \; & F_0(\mathbf{z}_0) + F_1(\mathbf{p}_0, \mathbf{z}_1) + F_2(\mathbf{p}_1, \mathbf{z}_1) + F_3(\mathbf{p}_2) \\ & \text{subject to} \quad \mathbf{p}_\ell = \mathbf{V}_\ell \mathbf{z}_\ell, \quad \ell = 0, 1, 2, \end{aligned} \tag{22}$$

where the *penalty functions* F_ℓ are defined as

$$\begin{aligned} F_0(\cdot) &= F_{\rm in}(\cdot), & F_1(\cdot,\cdot) &= \delta_{\{\mathbf{z}_1 = \mathbf{S}_{\rm tr}\mathbf{p}_0\}}(\cdot,\cdot), \\ F_2(\cdot,\cdot) &= \delta_{\{\mathbf{z}_2 = \mathbf{S}_{\rm mp}\mathbf{p}_1\}}(\cdot,\cdot), & F_3(\cdot) &= F_{\rm out}(\boldsymbol{y}, \cdot), \end{aligned} \tag{23}$$

where $\delta_{\mathcal{A}}(\cdot)$ is 0 on the set $\mathcal{A}$, and $+\infty$ on $\mathcal{A}^c$.

ML-VAMP for GLM Learning. Using this multi-layer representation, we can now apply the ML-VAMP algorithm from (Fletcher et al., 2018; Pandit et al., 2019) to solve the optimization (22). The steps are shown in Algorithm 1. These steps are a special case of the "MAP version" of ML-VAMP in (Pandit et al., 2019), but with a slightly different set-up for the GLM problem. We will call these steps the ML-VAMP GLM Learning Algorithm.

The algorithm operates in a set of iterations indexed by k. In each iteration, a "forward pass" through the layers generates estimates $\widehat{\mathbf{z}}_{k\ell}$ for the hidden variables $\mathbf{z}_\ell^0$, while a "backward pass" generates estimates $\widehat{\mathbf{p}}_{k\ell}$ for the variables $\mathbf{p}_\ell^0$. In each step, the estimates $\widehat{\mathbf{z}}_{k\ell}$ and $\widehat{\mathbf{p}}_{k\ell}$ are produced by functions $\mathbf{g}_\ell^+(\cdot)$ and $\mathbf{g}_\ell^-(\cdot)$ called *estimators* or *denoisers*.

For the MAP version of ML-VAMP algorithm in (Pandit et al., 2019), the denoisers are essentially proximal-type operators defined as

$$\mathrm{prox}_{F/\gamma}(\boldsymbol{u}) := \operatorname*{argmin}_{\boldsymbol{x}} F(\boldsymbol{x}) + \tfrac{\gamma}{2}\left\|\boldsymbol{x} - \boldsymbol{u}\right\|^2. \tag{24}$$

An important property of the proximal operator is that for separable functions F of the form (6), we have $[\mathrm{prox}_{F/\gamma}(\boldsymbol{u})]_i = \mathrm{prox}_{f/\gamma}(\boldsymbol{u}_i)$.

In the case of the GLM model, for $\ell = 0$ and L, on lines 1 and 1, the denoisers are proximal operators given by

$$\mathbf{g}_0^+(\mathbf{r}_0^-, \gamma_0^-) = \mathrm{prox}_{F_{\rm in}/\gamma_0^-}(\mathbf{r}_0^-), \tag{25a}$$

$$\mathbf{g}_3^-(\mathbf{r}_2^+, \boldsymbol{y}, \gamma_2^+) = \mathrm{prox}_{F_{\rm out}/\gamma_2^+}(\mathbf{r}_2^+). \tag{25b}$$

Note that in (25b), there is a dependence on $\boldsymbol{y}$ through the term $F_{\rm out}(\boldsymbol{y}, \cdot)$. For the *middle* terms, $\ell = 1, 2$, i.e., lines 1 and 1, the denoisers are given by

$$\mathbf{g}_\ell^+(\mathbf{r}_{\ell-1}^+, \mathbf{r}_\ell^-, \gamma_{\ell-1}^+, \gamma_\ell^-) := \widehat{\mathbf{z}}_\ell, \tag{26a}$$

$$\mathbf{g}_\ell^-(\mathbf{r}_{\ell-1}^+, \mathbf{r}_\ell^-, \gamma_{\ell-1}^+, \gamma_\ell^-) := \widehat{\mathbf{p}}_{\ell-1}, \tag{26b}$$

where $(\widehat{\mathbf{p}}_{\ell-1}, \widehat{\mathbf{z}}_\ell)$ are the solutions to the minimization

$$\begin{aligned} (\widehat{\mathbf{p}}_{\ell-1}, \widehat{\mathbf{z}}_\ell) := \operatorname*{argmin}_{(\mathbf{p}_{\ell-1}, \mathbf{z}_\ell)} F_\ell(\mathbf{p}_{\ell-1}, \mathbf{z}_\ell) + \frac{\gamma_\ell^-}{2}\|\mathbf{z}_\ell - \mathbf{r}_\ell^-\|^2 \\ + \frac{\gamma_{\ell-1}^+}{2}\|\mathbf{p}_{\ell-1} - \mathbf{r}_{\ell-1}^+\|^2. \end{aligned} \tag{27}$$

The quantity $\langle \partial \boldsymbol{v}/\partial \boldsymbol{u} \rangle$ on lines 1 and 1 denotes the empirical mean $\frac{1}{N}\sum_{n=1}^N \partial v_n/\partial u_n$.

Thus, the ML-VAMP algorithm in Algorithm 1 reduces the joint constrained minimization (22) over variables $(\mathbf{z}_0, \mathbf{z}_1, \mathbf{z}_2)$ and $(\mathbf{p}_0, \mathbf{p}_1, \mathbf{p}_2)$ to a set of proximal operations on pairs of variables $(\mathbf{p}_{\ell-1}, \mathbf{z}_\ell)$. As discussed in (Pandit et al., 2019), this type of minimization is similar to ADMM with adaptive step-sizes. Details of the denoisers $\mathbf{g}_\ell^\pm$ and other aspects of the algorithm are given in Appendix B.

4. Main Result

We make two assumptions. The first assumption imposes certain regularity conditions on the functions $f_{\rm ts}$, ϕ, $\phi_{\rm out}$, and maps $\mathbf{g}_\ell^\pm$ appearing in Algorithm 1. The precise definitions of pseudo-Lipschitz continuity and uniform Lipschitz continuity are given in Appendix A of the supplementary material.

Assumption 1. The denoisers and link functions satisfy the following continuity conditions:

(*a*) The proximal operators in (25),

$$\mathbf{g}_0^+(\mathbf{r}_0^-, \gamma_0^-), \quad \mathbf{g}_3^-(\mathbf{r}_2^+, \boldsymbol{y}, \gamma_2^+),$$

are uniformly Lipschitz continuous in $\mathbf{r}_0^-$ and $(\mathbf{r}_2^+, \boldsymbol{y})$ over parameters γ_0^- and γ_2^+.

(*b*) The link function $\phi_{\rm out}(p, d)$ is Lipschitz continuous in (p, d). The test error function $f_{\rm ts}(\phi(\widehat{z}), \phi_{\rm out}(z, d))$ is pseduo-Lipschitz continuous in $(\widehat{z}, z, d)$ of order 2.

Algorithm 1 ML-VAMP GLM Learning Algorithm

```
1:  Initialize $\gamma_{0\ell}^- > 0$, $\mathbf{r}_{0\ell}^- = 0$ for $\ell = 0, \dots, L-1$
2:
3:  for $k = 0, 1, \dots$ do
4:      // Forward Pass
5:      for $\ell = 0, \dots, L-1$ do
6:          if $\ell = 0$ then
7:              $\widehat{\mathbf{z}}_{k0} = \mathbf{g}_0^+(\mathbf{r}_{k0}^-, \gamma_{k0}^-)$
8:          else
9:              $\widehat{\mathbf{z}}_{k\ell} = \mathbf{g}_\ell^+(\mathbf{r}_{k,\ell-1}^+, \mathbf{r}_{k\ell}^-, \gamma_{k,\ell-1}^+, \gamma_{k\ell}^-)$
10:         end if
11:         $\alpha_{k\ell}^+ = \langle \partial \widehat{\mathbf{z}}_{k\ell} / \partial \mathbf{r}_{k\ell}^- \rangle$
12:         $\mathbf{r}_{k\ell}^+ = \dfrac{\mathbf{V}_\ell(\widehat{\mathbf{z}}_{k\ell} - \alpha_{k\ell}^+ \mathbf{r}_{k\ell}^-)}{1 - \alpha_{k\ell}^+}$
13:         $\gamma_{k\ell}^+ = (1/\alpha_{k\ell}^+ - 1)\gamma_{k\ell}^-$
14:     end for
15:
16:     // Backward Pass
17:     for $\ell = L, \dots, 1$ do
18:         if $\ell = L$ then
19:             $\widehat{\mathbf{p}}_{k,L-1} = \mathbf{g}_L^-(\mathbf{r}_{k,L-1}^+, \gamma_{k,L-1}^+)$
20:         else
21:             $\widehat{\mathbf{p}}_{k,\ell-1} = \mathbf{g}_\ell^-(\mathbf{r}_{k,\ell-1}^+, \mathbf{r}_{k+1,\ell}^-, \gamma_{k,\ell-1}^+, \gamma_{k+1,\ell}^-)$
22:         end if
23:         $\alpha_{k,\ell-1}^- = \left\langle \partial \widehat{\mathbf{p}}_{k,\ell-1} / \partial \mathbf{r}_{k,\ell-1}^+ \right\rangle$
24:         $\mathbf{r}_{k+1,\ell-1}^- = \dfrac{\mathbf{V}_{\ell-1}^\intercal(\widehat{\mathbf{p}}_{k,\ell-1} - \alpha_{k,\ell-1}^- \mathbf{r}_{k,\ell-1}^+)}{1 - \alpha_{k,\ell-1}^-}$
25:         $\gamma_{k+1,\ell-1}^- = (1/\alpha_{k,\ell-1}^- - 1)\gamma_{k,\ell-1}^+$
26:     end for
27: end for
```

Our second assumption is that the ML-VAMP algorithm converges. Specifically, let $\boldsymbol{x}_k = \boldsymbol{x}_k(N)$ be any set of outputs of Algorithm 1, at some iteration k and dimension N. For example, $\boldsymbol{x}_k(N)$ could be $\widehat{\mathbf{z}}_{k\ell}(N)$ or $\widehat{\mathbf{p}}_{k\ell}(N)$ for some ℓ, or a concatenation of signals such as $\begin{bmatrix}\mathbf{z}_\ell^0(N) & \widehat{\mathbf{z}}_{k\ell}(N)\end{bmatrix}$.

Assumption 2. Let $\boldsymbol{x}_k(N)$ be any finite set of outputs of the ML-VAMP algorithm as above. Then there exist limits

$$\boldsymbol{x}(N) = \lim_{k\to\infty} \boldsymbol{x}_k(N) \tag{28}$$

satisfying

$$\lim_{k\to\infty} \lim_{N\to\infty} \frac{1}{N} \|\boldsymbol{x}_k(N) - \boldsymbol{x}(N)\|^2 = 0. \tag{29}$$

We are now ready to state our main result.

Theorem 1. *Consider the GLM learning problem (2) solved by applying Algorithm 1 to the equivalent problem (22) under the assumptions of Section 2 along with Assumptions 1 and 2. Then, there exist constants $\tau_0^-, \overline{\gamma}_0^+ > 0$ and $\mathbf{M} \in \mathbb{R}_{\succ 0}^{2\times 2}$ such that the following hold:*

(a) The fixed points $\{\widehat{\mathbf{z}}_\ell, \widehat{\mathbf{p}}_\ell\}$, $\ell = 0, 1, 2$ of Algorithm 1 satisfy the KKT conditions for the constrained optimization problem (22). Equivalently $\widehat{\boldsymbol{w}} := \widehat{\mathbf{z}}_0$ is a stationary point of (2).

(b) The true parameter $\boldsymbol{w}^0$ and its estimate $\widehat{\boldsymbol{w}}$ empirically converge as

$$\lim_{N\to\infty} \{(w_i^0, \widehat{w}_i)\} \overset{PL(2)}{=} (W^0, \widehat{W}), \tag{30}$$

where W^0 is the random variable from (8) and

$$\widehat{W} = \operatorname{prox}_{f_{\text{in}}/\overline{\gamma}_0^+}(W^0 + Q_0^-), \tag{31}$$

with $Q_0^- = \mathcal{N}(0, \tau_0^-)$ independent of W^0.

(c) The asymptotic generalization error (17) with $(y_{\text{ts}}, \widehat{y}_{\text{ts}})$ defined as (3) is given by

$$\mathcal{E}_{\text{ts}} = \mathbb{E}\, f_{\text{ts}}\Big(\phi_{\text{out}}(Z_{\text{ts}}, D), \phi(\widehat{Z}_{\text{ts}})\Big), \tag{32}$$

where $(Z_{\text{ts}}, \widehat{Z}_{\text{ts}}) \sim \mathcal{N}(\mathbf{0}_2, \mathbf{M})$ and independent of D.

Part (a) shows that, similar to gradient descent, Algorithm 1 finds the stationary points of problem (2). These stationary points will be unique in strictly convex problems such as linear and logistic regression. Thus, in such cases, the same results will be true for any algorithm that finds such stationary points. Hence, the fact that we are using ML-VAMP is immaterial – our results apply to any solver for (2). Note that the convergence to the fixed points $\{\widehat{\mathbf{z}}_\ell, \widehat{\mathbf{p}}_\ell\}$ is assumed from Assumption 2.

Part (b) provides an exact description of the asymptotic statistical relation between the true parameter $\boldsymbol{w}^0$ and its estimate $\widehat{\boldsymbol{w}}$. The parameters $\tau_0^-, \overline{\gamma}_0^+ > 0$ and $\mathbf{M}$ can be explicitly computed using a set of recursive equations called the *state evolution* or SE described in Appendix C in the supplementary material.

We can use the expressions to compute a variety of relevant metrics. For example, the $PL(2)$ convergence shows that the MSE on the parameter estimate is

$$\lim_{N\to\infty} \frac{1}{N} \sum_{n=1}^{N} (w_n^0 - \widehat{w}_n)^2 = \mathbb{E}(W^0 - \widehat{W})^2. \tag{33}$$

The expectation on the right hand side of (33) can then be computed via integration over the joint density of $(W^0, \widehat{W})$ from part (b). In this way, we have a simple and exact method to compute the parameter error. Other metrics such as parameter bias or variance, cosine angle or sparsity detection can also be computed.

Part (c) of Theorem 1 similarly exactly characterizes the asymptotic generalization error. In this case, we would compute the expectation over the three variables $(Z, \widehat{Z}, D)$. In this way, we have provided a methodology for exactly predicting the generalization error from the key parameters of the problems such as the sampling ratio $\beta = p/N$, the regularizer, the output function, and the distributions of the true weights and noise. We provide several examples such as linear regression, logistic regression and SVM in the Appendix G. We also recover the result by (Hastie et al., 2019) in Appendix G.

Remarks on Assumptions. Note that Assumption 1 is satisfied in many practical cases. For example, it can be verified that it is satisfied in the case when $f_{\rm in}(\cdot)$ and $f_{\rm out}(\cdot)$ are convex. Assumption 2 is somewhat more restrictive in that it requires that the ML-VAMP algorithm converges. The convergence properties of ML-VAMP are discussed in (Fletcher et al., 2016). The ML-VAMP algorithm may not always converge, and characterizing conditions under which convergence is possible is an open question. However, experiments in (Rangan et al., 2019) show that the algorithm does indeed often converge, and in these cases, our analysis applies. In any case, we will see below that the predictions from Theorem 1 agree closely with numerical experiments in several relevant cases.

In some special cases equation (32) simplifies to yield quantitative insights for interesting modeling artifacts. We discuss these in Appendix G in the supplementary material.

5. Experiments

Training and Test Distributions. We validate our theoretical results on a number of synthetic data experiments. For all the experiments, the training and test data is generated following the model in Section 2. We generate the training and test eigenvalues as i.i.d. with lognormal distributions,

$$S_{\rm tr}^2 = A(10)^{0.1u_{\rm tr}}, \quad S_{\rm ts}^2 = A(10)^{0.1u_{\rm ts}},$$

where $(u_{\rm tr}, u_{\rm ts})$ are bivariate zero-mean Gaussian with

$$\mathrm{cov}(u_{\rm tr}, u_{\rm ts}) = \sigma_u^2 \left[\begin{array}{cc} 1 & \rho \\ \rho & 1 \end{array} \right].$$

In the case when $\sigma_u^2 = 0$, we obtain eigenvalues that are equal, corresponding to the i.i.d. case. With $\sigma_u^2 > 0$ we can model correlated features. Also, when the correlation coefficient $\rho = 1$, $S_{\rm tr} = S_{\rm ts}$, so there is no training and test mismatch. However, we can also select $\rho < 1$ to experiment with cases when the training and test distributions differ. In the examples below, we consider the following three cases:

(1) i.i.d. features ($\sigma_u = 0$);

(2) correlated features with matching training and test distributions ($\sigma_u = 3$ dB, $\rho = 1$); and

(3) correlated features with train-test mismatch ($\sigma_u = 3$ dB, $\rho = 0.5$).

For all experiments below, the true model coefficients are generated as i.i.d. Gaussian $w_j^0 \sim \mathcal{N}(0,1)$ and we use standard L2-regularization, $f_{\rm in}(w) = \lambda w^2/2$ for some $\lambda > 0$. Our framework can incorporate arbitrary i.i.d. distributions on w_j and regularizers, but we will illustrate just the Gaussian case with L2-regularization here.

Under-regularized linear regression. We first consider the case of under-regularized linear regression where the output channel is $\phi_{\rm out}(p,d) = p + d$ with $d \sim \mathcal{N}(0, \sigma_d^2)$. The noise variance σ_d^2 is set for an SNR level of 10 dB. We use a standard mean-square error (MSE) output loss, $f_{\rm out}(y,p) = (y-p)^2/(2\sigma_d^2)$. Since we are using the L2-regularizer, $f_{\rm in}(w) = \lambda w^2/2$, the minimization (2) is standard ridge regression. Moreover, if we were to select $\lambda = 1/\mathbb{E}(w_j^0)^2$, then the ridge regression estimate would correspond to the minimum mean-squared error (MMSE) estimate of the coefficients $\boldsymbol{w}^0$. However, to study the under-regularized regime, we take $\lambda = (10)^{-4}/\mathbb{E}(w_j^0)^2$.

Fig. 2 plots the test MSE for the three cases described above for the linear model. In the figure, we take $p = 1000$ features and vary the number of samples n from $0.2p$ (over-parameterized) to $3p$ (under-parameterized). For each value of n, we take 100 random instances of the model and compute the ridge regression estimate using the sklearn package and measure the test MSE on the 1000 independent test samples. The simulated values in Fig. 2 are the median test error over the 100 random trials. The test MSE is plotted in a normalized dB scale,

$$\text{Test MSE (dB)} = 10 \log_{10} \left(\frac{\mathbb{E}(\widehat{y}_{\rm ts} - y_{\rm ts})^2}{\mathbb{E} y_{\rm ts}^2} \right).$$

Also plotted is the state evolution (SE) theoretical test MSE from Theorem 1.

In all three cases in Fig. 2, the SE theory exactly matches the simulated values for the test MSE. Note that the case of match training and test distributions for this problem was studied in (Hastie et al., 2019; Mei & Montanari, 2019; Montanari et al., 2019) and we see the *double descent* phenomenon described in their work. Specifically, with highly under-regularized linear regression, the test MSE actually *increases* with more samples n in the over-parametrized regime ($n/p < 1$) and then decreases again in the under-parametrized regime ($n/p > 1$).

Our SE theory can also provide predictions for the correlated feature case. In this particular setting, we see that in the correlated case the test error is slightly lower in the over-parametrized regime since the energy of data is concentrated in a smaller sub-space. Interestingly, there is minimal difference between the correlated and i.i.d. cases for the

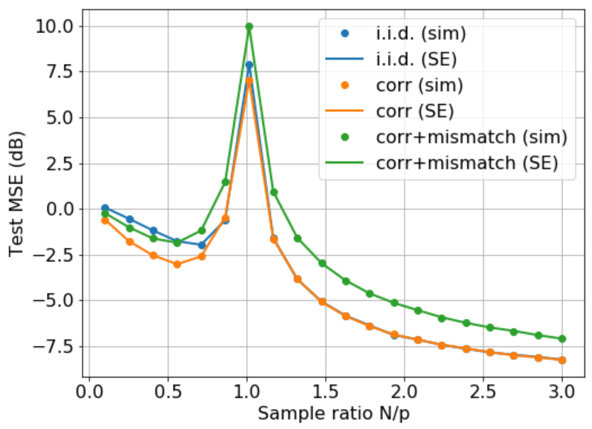

Figure 2. **Test error for under-regularized linear regression under various train and test distributions.** Noise variance σ_d^2 is set to SNR = 10 dB. The number of features $p = 1000$, and the number of samples n vary from $0.2 \times p$ (over-parameterized) to $3 \times p$ (under-parameterized). The experiment is averaged over 100 random instances of the model for each n and test MSE is calculated with 1000 independent test samples.

under-parametrized regime when the training and test data match. When the training and test data are not matched, the test error increases. In all cases, the SE theory can accurately predict these effects.

Logistic Regression. Fig. 3 shows a similar plot as Fig. 2 for a logistic model. Specifically, we use a logistic output $P(y = 1) = 1/(1 + e^{-p})$, a binary cross entropy output loss $f_{\text{out}}(y, p)$, and ℓ_2-regularization level λ, so that the output corresponds to the MAP estimate (we do not perform ridgeless regression in this case). Other values of λ would correspond to M-estimators with a mismatched prior.

The mean of the training and test eigenvalues $\mathbb{E}S_{\text{tr}}^2 = \mathbb{E}S_{\text{ts}}^2$ are selected such that, if the true coefficients $\boldsymbol{w}^0$ were known, we could obtain a 5% prediction error. As in the linear case, we generate random instances of the model, use the sklearn package to perform the logistic regression, and evaluate the estimates on 1000 new test samples. We compute the median error rate (1− accuracy) and compare the simulated values with the SE theoretical estimates. The i.i.d. case was considered in (Salehi et al., 2019). Fig. 3 shows that our SE theory is able to predict the test error rate exactly in i.i.d. cases along with a correlated case and a case with training and test mismatch.

Nonlinear Regression. The SE framework can also consider non-convex problems. As an example, we consider a non-linear regression problem where the output function is

$$\phi_{\text{out}}(p, d) = \tanh(p) + d, \quad d \sim \mathcal{N}(0, \sigma_d^2). \tag{34}$$

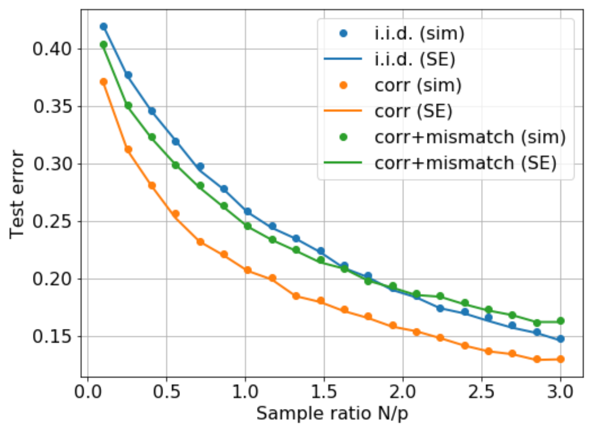

Figure 3. **Classification error rate with logistic regression under various train and test distributions.** F_{out} is the Binary cross-entropy loss, and F_{in} is the ridge penalty. The median error rate (1- accuracy) is estimated from 1000 new test samples.

The $\tanh(p)$ models saturation in the output. Corresponding to this output, we use a non-linear MSE output loss

$$f_{\text{out}}(y - p) = \frac{1}{2\sigma_d^2}(y - \tanh(p))^2. \tag{35}$$

This output loss is non-convex. The data is generated as in the previous experiments and we scale the data matrix so that the input $\mathbb{E}(p^2) = 9$ so that the $\tanh(p)$ is driven well into the non-linear regime. We also take $\sigma_d^2 = 0.01$.

For the simulation, the non-convex loss is minimized using Tensorflow where the non-linear model is described as a two-layer model. We use the ADAM optimizer (Kingma & Ba, 2014) with 200 epochs to approach a local minimum of the objective (2). Fig. 4 plots the median test MSE for the estimate along with the SE theoretical test MSE. We again see that the SE theory is able to predict the test MSE in all cases even for this non-convex problem. Note that Figures 3 and 4 do not show a double descent because we apply regularization in those experiments.

6. Conclusions

In this paper we provide a procedure for exactly computing the asymptotic generalization error of a solution in a generalized linear model (GLM). This procedure is based on scalar quantities which are fixed points of a recursive iteration. The formula holds for a large class of generalization metrics, loss functions, and regularization schemes. Our formula allows analysis of important modeling effects such as (i) overparameterization, (ii) dependence between covariates, and (iii) mismatch between train and test distributions, which play a significant role in the analysis and design of

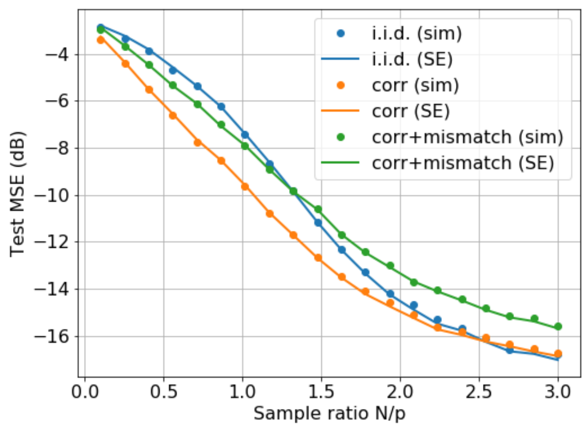

Figure 4. **Test MSE under a non-linear least square estimation.** The tanh($\cdot$) output function is used with ℓ_2-regularization. Noise variance $\sigma_d^2 = 0.01$. The ADAM optimizer is used for simulations.

machine learning systems. We experimentally validate our theoretical results for linear as well as non-linear regression and logistic regression, where a strong agreement is seen between our formula and simulated results.

Acknowledgements

The work of M. Emami, M. Sahraee-Ardakan, P. Pandit, and A. K. Fletcher was supported in part by the National Science Foundation under Grants 1254204 and 1738286, and the Office of Naval Research under Grant N00014-15-1-2677. S. Rangan was supported in part by the National Science Foundation under Grants 1116589, 1302336, and 1547332, NIST, the industrial affiliates of NYU WIRELESS, and the SRC.

References

Advani, M. and Ganguli, S. An equivalence between high dimensional bayes optimal inference and m-estimation. In *Advances in Neural Information Processing Systems*, pp. 3378–3386, 2016. 2

Advani, M. S. and Saxe, A. M. High-dimensional dynamics of generalization error in neural networks. *arXiv preprint arXiv:1710.03667*, 2017. 2

Allen-Zhu, Z., Li, Y., and Liang, Y. Learning and generalization in overparameterized neural networks, going beyond two layers. In *Advances in Neural Information Processing Systems*, pp. 6155–6166, 2019. 2

Arora, S., Du, S. S., Hu, W., Li, Z., and Wang, R. Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. *arXiv preprint arXiv:1901.08584*, 2019. 2

Barbier, J., Krzakala, F., Macris, N., Miolane, L., and Zdeborová, L. Optimal errors and phase transitions in high-dimensional generalized linear models. *Proc. National Academy of Sciences*, 116(12):5451–5460, 2019. 2

Bartlett, P. L., Long, P. M., Lugosi, G., and Tsigler, A. Benign overfitting in linear regression. *arXiv preprint arXiv:1906.11300*, 2019. 2

Bayati, M. and Montanari, A. The dynamics of message passing on dense graphs, with applications to compressed sensing. *IEEE Trans. Inform. Theory*, 57(2):764–785, February 2011. 2, 3, 11

Belkin, M., Ma, S., and Mandal, S. To understand deep learning we need to understand kernel learning. *arXiv preprint arXiv:1802.01396*, 2018. 2

Belkin, M., Hsu, D., Ma, S., and Mandal, S. Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proc. National Academy of Sciences*, 116(32):15849–15854, 2019a. 2

Belkin, M., Hsu, D., and Xu, J. Two models of double descent for weak features. *arXiv preprint arXiv:1903.07571*, 2019b. 2

Bös, S. and Opper, M. Dynamics of training. In Mozer, M. C., Jordan, M. I., and Petsche, T. (eds.), *Advances in Neural Information Processing Systems 9*, pp. 141–147. MIT Press, 1997. 2

Cakmak, B., Winther, O., and Fleury, B. H. S-AMP: Approximate message passing for general matrix ensembles. In *Proc. IEEE ITW*, 2014. 2

Daniely, A. Sgd learns the conjugate kernel class of the network. In *Advances in Neural Information Processing Systems*, pp. 2422–2430, 2017. 2

Daniely, A., Frostig, R., and Singer, Y. Toward deeper understanding of neural networks: The power of initialization and a dual view on expressivity. In *Advances In Neural Information Processing Systems*, pp. 2253–2261, 2016. 2

Deng, Z., Kammoun, A., and Thrampoulidis, C. A model of double descent for high-dimensional binary linear classification. *arXiv preprint arXiv:1911.05822*, 2019. 1, 2, 20

Donoho, D. L., Maleki, A., and Montanari, A. Message-passing algorithms for compressed sensing. *Proc. National Academy of Sciences*, 106(45):18914–18919, 2009. 2

Donoho, D. L., Maleki, A., and Montanari, A. Message passing algorithms for compressed sensing. In *Proc. Inform. Theory Workshop*, pp. 1–5, 2010. 2

Du, S. S., Zhai, X., Poczos, B., and Singh, A. Gradient descent provably optimizes over-parameterized neural networks. *arXiv preprint arXiv:1810.02054*, 2018. 2

Fletcher, A., Sahraee-Ardakan, M., Rangan, S., and Schniter, P. Expectation consistent approximate inference: Generalizations and convergence. In *Proc. IEEE Int. Symp. Information Theory (ISIT)*, pp. 190–194. IEEE, 2016. 2, 7

Fletcher, A. K., Rangan, S., and Schniter, P. Inference in deep networks in high dimensions. *Proc. IEEE Int. Symp. Information Theory*, 2018. 2, 4, 5

Gabrié, M., Manoel, A., Luneau, C., Barbier, J., Macris, N., Krzakala, F., and Zdeborová, L. Entropy and mutual information in models of deep neural networks. In *Proc. NIPS*, 2018. 2

Gerbelot, C., Abbara, A., and Krzakala, F. Asymptotic errors for convex penalized linear regression beyond gaussian matrices. *arXiv preprint arXiv:2002.04372*, 2020. 3

Givens, C. R., Shortt, R. M., et al. A class of wasserstein metrics for probability distributions. *The Michigan Mathematical Journal*, 31(2):231–240, 1984. 16

Hastie, T., Montanari, A., Rosset, S., and Tibshirani, R. J. Surprises in high-dimensional ridgeless least squares interpolation. *arXiv preprint arXiv:1903.08560*, 2019. 2, 7, 19

He, H., Wen, C.-K., and Jin, S. Generalized expectation consistent signal recovery for nonlinear measurements. In *2017 IEEE International Symposium on Information Theory (ISIT)*, pp. 2333–2337. IEEE, 2017. 2

Jacot, A., Gabriel, F., and Hongler, C. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in neural information processing systems*, pp. 8571–8580, 2018. 2

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 8

Ma, J. and Ping, L. Orthogonal amp. *IEEE Access*, 5: 2020–2033, 2017. 2

Manoel, A., Krzakala, F., Varoquaux, G., Thirion, B., and Zdeborová, L. Approximate message-passing for convex optimization with non-separable penalties. *arXiv preprint arXiv:1809.06304*, 2018. 2

Mei, S. and Montanari, A. The generalization error of random features regression: Precise asymptotics and double descent curve. *arXiv preprint arXiv:1908.05355*, 2019. 1, 2, 7

Montanari, A., Ruan, F., Sohn, Y., and Yan, J. The generalization error of max-margin linear classifiers: High-dimensional asymptotics in the overparametrized regime. *arXiv preprint arXiv:1911.01544*, 2019. 1, 2, 7

Muthukumar, V., Vodrahalli, K., and Sahai, A. Harmless interpolation of noisy data in regression. In *2019 IEEE International Symposium on Information Theory (ISIT)*, pp. 2299–2303. IEEE, 2019. 2

Neyshabur, B., Li, Z., Bhojanapalli, S., LeCun, Y., and Srebro, N. Towards understanding the role of overparametrization in generalization of neural networks. *arXiv preprint arXiv:1805.12076*, 2018. 2

Opper, M. and Winther, O. Expectation consistent approximate inference. *Journal of Machine Learning Research*, 6(Dec):2177–2204, 2005. 2

Pandit, P., Sahraee, M., Rangan, S., and Fletcher, A. K. Asymptotics of MAP inference in deep networks. In *Proc. IEEE Int. Symp. Information Theory*, pp. 842–846, 2019. 4, 5, 12, 15

Pandit, P., Sahraee-Ardakan, M., Rangan, S., Schniter, P., and Fletcher, A. K. Inference with deep generative priors in high dimensions. *arXiv preprint arXiv:1911.03409*, 2019. 2, 13, 15

Rangan, S., Schniter, P., and Fletcher, A. K. Vector approximate message passing. *IEEE Trans. Information Theory*, 65(10):6664–6684, 2019. 2, 7

Reeves, G. Additivity of information in multilayer networks via additive gaussian noise transforms. In *Proc. 55th Annual Allerton Conf. Communication, Control, and Computing (Allerton)*, pp. 1064–1070. IEEE, 2017. 2

Salehi, F., Abbasi, E., and Hassibi, B. The impact of regularization on high-dimensional logistic regression. *arXiv preprint arXiv:1906.03761*, 2019. 1, 2, 8, 20

Tulino, A. M., Verdú, S., et al. Random matrix theory and wireless communications. *Foundations and Trends® in Communications and Information Theory*, 1(1):1–182, 2004. 18, 20

Villani, C. *Optimal transport: old and new*, volume 338. Springer Science & Business Media, 2008. 3, 11

Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530*, 2016. 2