Approximate Cross-Validation with Low-Rank Data in High Dimensions

William T. Stephenson
MIT
wtstephe@mit.edu

Madeleine Udell
Cornell University
udell@cornell.edu

Tamara Broderick
MIT
tamarab@mit.edu

Abstract

Many recent advances in machine learning are driven by a challenging trifecta: large data size N, high dimensions, and expensive algorithms. In this setting, cross-validation (CV) serves as an important tool for model assessment. Recent advances in approximate cross validation (ACV) provide accurate approximations to CV with only a single model fit, avoiding traditional CV's requirement for repeated runs of expensive algorithms. Unfortunately, these ACV methods can lose both speed and accuracy in high dimensions — unless sparsity structure is present in the data. Fortunately, there is an alternative type of simplifying structure that is present in most data: approximate low rank (ALR). Guided by this observation, we develop a new algorithm for ACV that is fast and accurate in the presence of ALR data. Our first key insight is that the Hessian matrix — whose inverse forms the computational bottleneck of existing ACV methods — is ALR. We show that, despite our use of the *inverse* Hessian, a low-rank approximation using the largest (rather than the smallest) matrix eigenvalues enables fast, reliable ACV. Our second key insight is that, in the presence of ALR data, error in existing ACV methods roughly grows with the (approximate, low) rank rather than with the (full, high) dimension. These insights allow us to prove theoretical guarantees on the quality of our proposed algorithm — along with fast-to-compute upper bounds on its error. We demonstrate the speed and accuracy of our method, as well as the usefulness of our bounds, on a range of real and simulated data sets.

1 Introduction

Recent machine learning advances are driven at least in part by increasingly rich data sets — large in both data size N and dimension D. The proliferation of data and algorithms makes cross-validation (CV) [Stone, 1974, Geisser, 1975, Musgrave et al., 2020] an appealing tool for model assessment due its ease of use and wide applicability. For high-dimensional data sets, leave-one-out CV (LOOCV) is often especially accurate as its folds more closely match the true size of the data [Burman, 1989]; see also Figure 1 of Rad and Maleki [2020]. Traditionally many practitioners nonetheless avoid LOOCV due its computational expense; it requires re-running an expensive machine learning algorithm N times. To address this expense, a number of authors have proposed approximate cross-validation (ACV) methods [Beirami et al., 2017, Rad and Maleki, 2020, Giordano et al., 2019]; these methods are fast to run on large data sets, and both theory and experiments demonstrate their accuracy. But these methods struggle in high-dimensional problems in two ways. First, they require inversion of a $D \times D$ matrix, a computationally expensive undertaking. Second, their accuracy can degrade in high dimensions; see Fig. 1 of Stephenson and Broderick [2020] for a classification example and Fig. 1 below for a count-valued regression example. Koh and Liang [2017], Lorraine et al. [2020] have investigated approximations to the matrix inverse for problems similar to ACV, but these approximations do not work well for ACV itself; see [Stephenson and Broderick, 2020, Appendix B].

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

Stephenson and Broderick [2020] demonstrate how a practitioner might avoid these high-dimensional problems in the presence of sparse data. But sparsity may be a somewhat limiting assumption.

We here consider approximately *low-rank* (ALR) data. Udell and Townsend [2019] argue that ALR data matrices are pervasive in applications ranging from fluid dynamics and genomics to social networks and medical records — and that there are theoretical reasons to expect ALR structure in many large data matrices. For concreteness and to facilitate theory, we focus on fitting generalized linear models (GLMs). We note that GLMs are a workhorse of practical data analysis; as just one example, one of many popular books on GLMs [McCullaugh, 1989] has been cited over 9,000 times since 2015 as of this writing. While accurate ACV methods for GLMs alone thus have potential for great impact, we expect many of our insights may extend beyond both GLMs and LOOCV (i.e. to other CV and bootstrap-like "retraining" schemes).

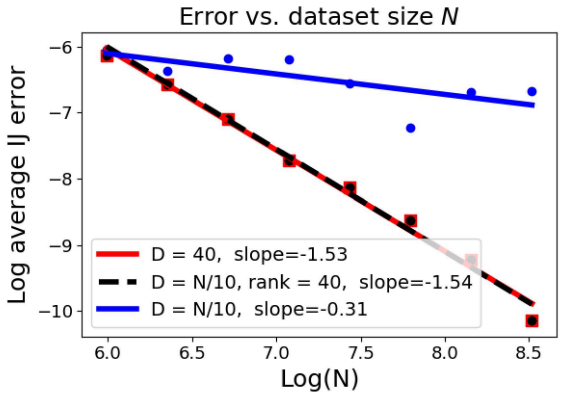

Figure 1: Accuracy of the IJ approximation in Eq. (4) for a synthetic Poisson regression problem versus the dataset size N. Red shows the accuracy when the data dimension is fixed at $D = 40$, blue when the dimension grows as $D = N/10$, and black when the dimension grows as $D = N/10$ but with a fixed rank of 40. High-dimensional yet low-rank data has identical performance to low-dimensional data.

In particular, we propose an algorithm for fast, accurate ACV for GLMs with high-dimensional covariate matrices — and provide computable upper bounds on the error of our method relative to exact LOOCV. Two major innovations power our algorithm. First, we prove that existing ACV methods automatically obtain high accuracy in the presence of high-dimensional yet ALR data. Our theory provides cheaply computable upper bounds on the error of existing ACV methods. Second, we notice that the $D \times D$ matrix that needs to be inverted in ACV is ALR when the covariates are ALR. We propose to use a low-rank approximation to this matrix. We provide a computable upper bound on the extra error introduced by using such a low-rank approximation. By studying our bound, we show the surprising fact that, for the purposes of ACV, the matrix is well approximated by using its *largest* eigenvalues, despite the fact that ACV uses the matrix inverse. We demonstrate the speed and accuracy of both our method and bounds with a range of experiments.

2 Background: approximate CV methods

We consider fitting a generalized linear model (GLM) with parameter $\theta \in \mathbb{R}^D$ to some dataset with N observations, $\{x_n, y_n\}_{n=1}^N$, where $x_n \in \mathbb{R}^D$ are covariates and $y_n \in \mathbb{R}$ are responses. We suppose that the x_n are approximately low rank (ALR); that is, the matrix $X \in \mathbb{R}^{N \times D}$ with rows x_n has many singular values near zero. These small singular values can amplify noise in the responses. Hence it is common to use ℓ_2 regularization to ensure that our estimated parameter $\hat{\theta}$ is not too sensitive to the subspace with small singular values; the rotational invariance of the ℓ_2 regularizer automatically penalizes any deviation of θ away from the low-rank subspace [Hastie et al., 2009, Sec. 3.4]. Thus we consider:

$$\hat{\theta} := \arg\min_{\theta \in \mathbb{R}^D} \frac{1}{N} \sum_{n=1}^{N} f(x_n^T \theta, y_n) + \frac{\lambda}{2} \|\theta\|_2^2, \tag{1}$$

where $\lambda \geq 0$ is some regularization parameter and $f : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ is convex in its first argument for each y_n. Throughout, we assume f to be twice differentiable in its first argument. To use leave-one-out CV (LOOCV), we compute $\hat{\theta}_{\backslash n}$, the estimate of θ after deleting the nth datapoint from the sum, for each n. To assess the out-of-sample error of our fitted $\hat{\theta}$, we then compute:

$$\frac{1}{N} \sum_{n=1}^{N} \mathrm{Err}(x_n^T \hat{\theta}_{\backslash n}, y_n), \tag{2}$$

where $\mathrm{Err} : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ is some function measuring the discrepancy between the observed y_n and its prediction based on $\hat{\theta}_{\setminus n}$ — for example, squared error or logistic loss.

Computing $x_n^T \hat{\theta}_{\setminus n}$ for every n requires solving N optimization problems, which can be a prohibitive computational expense. Approximate CV (ACV) methods aim to alleviate this burden via one of two principal approaches below. Denote the Hessian of the objective by $H := (1/N) \sum_{n=1}^N \nabla_\theta^2 f(x_n^T \hat{\theta}, y_n) + \lambda I_D$ and the kth scalar derivative of f as $\hat{D}_n^{(k)} := d^k f(z, y_n)/dz^k|_{z=x_n^T \hat{\theta}}$. Finally, let $Q_n := x_n^T H^{-1} x_n$ be the nth quadratic form on H^{-1}. The first approximation, based on taking a Newton step from $\hat{\theta}$ on the objective $(1/N) \sum_{m \neq n}^N f(x_m^T \theta, y_m) + \lambda \|\theta\|_2^2$, was proposed by Obuchi and Kabashima [2016, 2018], Rad and Maleki [2020], Beirami et al. [2017]. We denote this approximation by $\mathrm{NS}_{\setminus n}$; specializing to GLMs, we have:

$$x_n^T \hat{\theta}_{\setminus n} \approx x_n^T \mathrm{NS}_{\setminus n} := x_n^T \hat{\theta} + \frac{\hat{D}_n^{(1)}}{N} \frac{Q_n}{1 - \hat{D}_n^{(2)} Q_n}. \tag{3}$$

Throughout we focus on approximating $x_n^T \hat{\theta}_{\setminus n}$, rather than $\hat{\theta}_{\setminus n}$, since $x_n^T \hat{\theta}_{\setminus n}$ is the argument of Eq. (2). See Appendix A for a derivation of Eq. (3). The second approximation we consider is based on the infinitesimal jackknife [Jaeckel, 1972, Efron, 1982]; it was conjectured as a possible ACV method by Koh and Liang [2017], used in a comparison by Beirami et al. [2017], and studied in depth by Giordano et al. [2019]. We denote this approximation by $\mathrm{IJ}_{\setminus n}$; specializing to GLMs, we have:

$$x_n^T \hat{\theta}_{\setminus n} \approx x_n^T \mathrm{IJ}_{\setminus n} := x_n^T \hat{\theta} + (\hat{D}_n^{(1)}/N) Q_n. \tag{4}$$

See Appendix A for a derivation. We consider both $\mathrm{NS}_{\setminus n}$ and $\mathrm{IJ}_{\setminus n}$ in what follows as the two have complementary strengths. In our experiments in Section 6, $\mathrm{NS}_{\setminus n}$ tends to be more accurate; we suspect that GLM users should generally use $\mathrm{NS}_{\setminus n}$. On the other hand, $\mathrm{NS}_{\setminus n}$ requires the inversion of a different $D \times D$ matrix for each n. In the case of LOOCV for GLMs, each matrix differs by a rank-one update, so standard matrix inverse update formulas allow us to derive Eq. (3), which requires only a single inverse across folds. But such a simplification need not generally hold for models beyond GLMs and data re-weightings beyond LOOCV (such as other forms of CV or the bootstrap). By contrast, even beyond GLMs and LOOCV, the IJ requires only a single matrix inverse for all n.

In any case, we notice that existing theory and experiments for both $\mathrm{NS}_{\setminus n}$ and $\mathrm{IJ}_{\setminus n}$ tend to either focus on low dimensions or show poor performance in high dimensions; see Appendix C for a review. One problem is that error in both approximations can grow large in high dimensions. See [Stephenson and Broderick, 2020] for an example; also, in Fig. 1, we show the $\mathrm{IJ}_{\setminus n}$ on a synthetic Poisson regression task. When we fix $D = 40$ and N grows, the error drops quickly; however, if we fix $D/N = 1/10$ the error is substantially worse. A second problem is that both $\mathrm{NS}_{\setminus n}$ and $\mathrm{IJ}_{\setminus n}$ rely on the computation of $Q_n = x_n^T H^{-1} x_n$, which in turn relies on computation[1] of H^{-1}. The resulting $O(D^3)$ computation time quickly becomes impractical in high dimensions. Our major contribution is to show that both of these issues can be avoided when the data are ALR.

3 Methodology

We now present our algorithm for fast, approximate LOOCV in GLMs with ALR data. We then state our main theorem, which (1) bounds the error in our algorithm relative to exact CV, (2) gives the computation time of our algorithm, and (3) gives the computation time of our bounds. Finally we discuss the implications of our theorem before moving on to the details of its proof in the remainder of the paper.

Our method appears in Algorithm 1. To avoid the $O(D^3)$ matrix inversion cost, we replace H by $\widetilde{H} \approx H$, where $\widetilde{H}$ uses a rank-K approximation and can be quickly inverted. We can then use $\widetilde{H}$ to compute $\widetilde{Q}_n \approx Q_n$, which enters into either the NS or IJ approximation, as desired.

Before stating Theorem 1, we establish some notation. We will see in Proposition 2 of Section 4 that we can provide computable upper bounds $\eta_n \geq |\widetilde{Q}_n - Q_n|$; η_n will enter directly into the error

[1] In practice, for numerical stability, we compute a factorization of H so that $H^{-1} x_n$ can be quickly evaluated for all n. However, for brevity, we refer to computation of the inverse of H throughout.

Algorithm 1 Approximation to $\{x_n^T \hat{\theta}_{\backslash n}\}_{n=1}^N$ for low-rank GLMs

procedure APPXLOOCV($\hat{\theta}, X, \lambda, \{\hat{D}_n^{(1)}\}_{n=1}^N, \{\hat{D}_n^{(2)}\}_{n=1}^N, K$)
  $B \leftarrow X^T \text{diag}\{\hat{D}_n^{(2)}\}_{n=1}^N X$ ▷ The Hessian, H, equals $B + \lambda I_D$
  $\{\widetilde{Q}_n\}_{n=1}^N \leftarrow$ APPXQN(B, K, λ) ▷ Uses rank-K decomposition of B (Section 5)
  for $n = 1, \ldots, N$ **do**
    either $x_n^T \widetilde{\text{NS}}_{\backslash n} \leftarrow x_n^T \text{NS}_{\backslash n}(\widetilde{Q}_n)$ ▷ i.e., compute Eq. (3) using $\widetilde{Q}_n$ instead of Q_n
    or $x_n^T \widetilde{\text{IJ}}_{\backslash n} \leftarrow x_n^T \text{IJ}_{\backslash n}(\widetilde{Q}_n)$ ▷ i.e., compute Eq. (4) using $\widetilde{Q}_n$ instead of Q_n
  end for
  return $\{x_n^T \widetilde{\text{NS}}_{\backslash n}\}_{n=1}^N$ **or** $\{x_n^T \widetilde{\text{IJ}}_{\backslash n}\}_{n=1}^N$ ▷ User's choice
end procedure

bound for $x_n^T \widetilde{\text{IJ}}_{\backslash n}$ in Theorem 1 below. To bound the error of $x_n^T \widetilde{\text{NS}}_{\backslash n}$, we need to further define

$$E_n := \max \left\{ \left| \frac{\widetilde{Q}_n + \eta_n}{1 - \hat{D}_n^{(2)}(\widetilde{Q}_n + \eta_n)} - \frac{\widetilde{Q}_n}{1 - \hat{D}_n^{(2)} \widetilde{Q}_n} \right|, \left| \frac{\widetilde{Q}_n - \eta_n}{1 - \hat{D}_n^{(2)}(\widetilde{Q}_n - \eta_n)} - \frac{\widetilde{Q}_n}{1 - \hat{D}_n^{(2)} \widetilde{Q}_n} \right| \right\}.$$

Additionally, we will see in Proposition 1 of Section 4 that we can bound the "local Lipschitz-ness" of the Hessian related to the third derivatives of f evaluated at some z, $\hat{D}_n^{(3)}(z) := d^3 f(z, y_n)/dz^3|_{z=z}$. We will denote our bound by M_n:

$$M_n \geq \left(\frac{1}{N} \sum_{m \neq n} \|x_m\|_2^2 \right) \max_{s \in [0,1]} \left| \hat{D}_n^{(3)} \left(x_n^T ((1-s)\hat{\theta} + s\hat{\theta}_{\backslash n}) \right) \right|, \tag{5}$$

We are now ready to state, and then discuss, our main result — which is proved in Appendix D.3.

Theorem 1. ***(1) Accuracy:*** *Let $\eta_n \geq |Q_n - \tilde{Q}_n|$ be the upper bound produced by Proposition 2 and M_n the local Lipschitz constants computed in Proposition 1. Then the estimates $x_n^T \widetilde{\text{NS}}_{\backslash n}$ and $x_n^T \widetilde{\text{IJ}}_{\backslash n}$ produced by Algorithm 1 satisfy:*

$$|x_n^T \widetilde{\text{NS}}_{\backslash n} - x_n^T \hat{\theta}_{\backslash n}| \leq \frac{M_n}{N^2 \lambda^3} |\hat{D}_n^{(1)}|^2 \|x_n\|_2^3 + |\hat{D}_n^{(1)}| E_n \tag{6}$$

$$|x_n^T \widetilde{\text{IJ}}_{\backslash n} - x_n^T \hat{\theta}_{\backslash n}| \leq \frac{M_n}{N^2 \lambda^3} |\hat{D}_n^{(1)}|^2 \|x_n\|_2^3 + \frac{1}{N^2 \lambda^2} |\hat{D}_n^{(1)}| \hat{D}_n^{(2)} \|x_n\|_2^4 + |\hat{D}_n^{(1)}| \eta_n. \tag{7}$$

(2) Algorithm computation time: *The runtime of Algorithm 1 is in $O(NDK + K^3)$.* ***(3) Bound computation time:*** *The upper bounds in Eqs.* (6) *and* (7) *are computable in $O(DK)$ time for each n for common GLMs such as logistic and Poisson regression.*

To interpret the running times, note that standard ACV methods have total runtime in $O(ND^2 + D^3)$. So Algorithm 1 represents a substantial speedup when the dimension D is large and $K \ll D$. Also, note that our bound computation time has no worse behavior than our algorithm runtime. We demonstrate in our experiments (Section 6) that our error bounds are both computable and useful in practice. To help interpret the bounds, note that they contain two sources of error: (A) the error of our additional approximation relative to existing ACV methods (i.e. the use of $\widetilde{Q}_n$) and (B) the error of existing ACV methods in the presence of ALR data. Our first corollary notes that (A) goes to zero as the data becomes exactly low rank.

Corollary 1. *As the data becomes exactly low rank with rank R (i.e., X's lowest singular values $\sigma_d \to 0$ for $d = R+1, \ldots, D$), we have $\eta_n, E_n \to 0$ if $K \geq R$.*

See Appendix D.4 for a proof. Our second corollary gives an example for which the error in existing (exact) ACV methods (B) vanishes as N grows.

Corollary 2. *Suppose the third derivatives $\hat{D}_n^{(3)}$ and the x_n are both bounded and the data are exactly low-rank with constant rank R. Then with $N \to \infty$, D growing at any rate, and K arbitrary, the right hand sides of Eqs.* (6) *and* (7) *reduce to $|\hat{D}_n^{(1)}| E_n$ and $|\hat{D}_n^{(1)}| \eta_n$, respectively.*

We note that Corollary 2 is purely illustrative, and we strongly suspect that none of its conditions are necessary. Indeed, our experiments in Section 6 show that the bounds of Theorem 1 imply reasonably low error for non-bounded derivatives with ALR data and only moderate N.

4 Accuracy of exact ACV with approximately low-rank data

Recall that the main idea behind Algorithm 1 is to compute a fast approximation to existing ACV methods by exploiting ALR structure. To prove our error bounds, we begin by proving that the exact ACV methods $\mathrm{NS}_{\backslash n}$ and $\mathrm{IJ}_{\backslash n}$ approximately (respectively, exactly) retain the low-dimensional accuracy displayed in red in Fig. 1 when applied to GLMs with approximately (respectively, exactly) low-rank data. Let us first define low-rank data. Let $X = U\Sigma V^T$ be the singular value decomposition of X, where $U \in \mathbb{R}^{N\times D}$ has orthonormal columns, $\Sigma \in \mathbb{R}^{D\times D}$ is a diagonal matrix, and $V \in \mathbb{R}^{D\times D}$ is an orthonormal matrix.

Definition 1. *We say that a matrix X with singular value decomposition $X = U\Sigma V$ is of* exactly low-rank *R if $\Sigma_{dd} = 0$ for all $d > R$. We say that X is of* approximately low rank (ALR) *R if $\Sigma_{dd} \approx 0$ for all $d > R$.*

We note that this definition of ALR is different from that in Udell and Townsend [2019], which we gave in Section 1 as a motivation for considering ALR data. In particular, Udell and Townsend [2019] define a matrix X to be of ALR if it is entry-wise ε-close to some matrix of exactly low-rank R; such a matrix can be very different from our definition of ALR, as such a matrix can have $\Sigma_{R+1,R+1} = D\varepsilon$. While we only consider Udell and Townsend [2019] as general motivation, we note that in our work below, we will consider the case of $\Sigma_{dd} \to 0$, making the two definitions of ALR equivalent.

We now show that existing ACV methods are accurate in the presence of exactly low-rank data. Let $V_{:R}$ be the top R right singular vectors of X (i.e. the first R columns of V), and fit a model restricted to R dimensions as:

$$\hat{\phi} := \underset{\phi\in\mathbb{R}^R}{\arg\min} \frac{1}{N}\sum_{n=1}^{N} f((V_{:R}^T x_n)^T\phi) + \frac{\lambda}{2}\|\phi\|_2^2.$$

Let $\hat{\phi}_{\backslash n}$ be the nth leave-one-out parameter estimate from this problem, and let $\mathrm{RNS}_{\backslash n}$ and $\mathrm{RIJ}_{\backslash n}$ be Eq. (3) and Eq. (4) applied to this restricted problem. We can now show that the error of $\mathrm{IJ}_{\backslash n}$ and $\mathrm{NS}_{\backslash n}$ applied to the full D-dimensional problem is exactly the same as the error of $\mathrm{RNS}_{\backslash n}$ and $\mathrm{RIJ}_{\backslash n}$ applied to the restricted $R \ll D$ dimensional problem.

Lemma 1. *Assume that the data matrix X is exactly low-rank R. Then $|x_n^T\mathrm{NS}_{\backslash n} - x_n^T\hat{\theta}_{\backslash n}| = |(V_{:R}^T x_n)^T\mathrm{RNS}_{\backslash n} - (V_{:R}^T x_n)^T\hat{\phi}_{\backslash n}|$ and $|x_n^T\mathrm{IJ}_{\backslash n} - x_n^T\hat{\theta}_{\backslash n}| = |(V_{:R}^T x_n)^T\mathrm{RIJ}_{\backslash n} - (V_{:R}^T x_n)^T\hat{\phi}_{\backslash n}|$.*

See Appendix D.1 for a proof. Based on previous work (e.g., [Beirami et al., 2017, Rad and Maleki, 2020, Giordano et al., 2019]), we expect the ACV errors $|(V_{:R}^T x_n)^T\mathrm{RNS}_{\backslash n} - (V_{:R}^T x_n)^T\hat{\phi}_{\backslash n}|$ and $|(V_{:R}^T x_n)^T\mathrm{RIJ}_{\backslash n} - (V_{:R}^T x_n)^T\hat{\phi}_{\backslash n}|$ to be small, as they represent the errors of $\mathrm{NS}_{\backslash n}$ and $\mathrm{IJ}_{\backslash n}$ applied to an R-dimensional problem. We confirm Lemma 1 numerically in Fig. 1, where the error for the $D = 40$ problems (red) exactly matches that of the high-D but exact low-rank $R = 40$ problems (black).

However, real-world covariate matrices X are rarely exactly low-rank. By adapting results from Wilson et al. [2020], we can give bounds that smoothly decay as we leave the exact low-rank setting of Lemma 1. To that end, define:

$$L_n := \left(\frac{1}{N}\sum_{m:\, m\neq n}^{N} \|x_m\|_2^2\right) \max_{s\in[0,1]} \hat{D}_n^{(3)}\left(x_n^T((1-s)\hat{\theta} + s\hat{\theta}_{\backslash n})\right). \tag{8}$$

Lemma 2. *Assume that $\lambda > 0$. Then, for all n:*

$$|x_n^T\mathrm{NS}_{\backslash n} - x_n^T\hat{\theta}_{\backslash n}| \leq \frac{L_n}{N^2\lambda^3}|\hat{D}_n^{(1)}|^2\,\|x_n\|_2^3 \tag{9}$$

$$|x_n^T\mathrm{IJ}_{\backslash n} - x_n^T\hat{\theta}_{\backslash n}| \leq \frac{L_n}{N^2\lambda^3}|\hat{D}_n^{(1)}|^2\,\|x_n\|_2^3 + \frac{1}{N^2\lambda^2}|\hat{D}_n^{(1)}|\hat{D}_n^{(2)}\,\|x_n\|_2^4. \tag{10}$$

Furthermore, these bounds continuously decay as the data move from exactly to approximately low rank in that they are continuous in the singular values of X.

The proofs of Eqs. (9) and (10) mostly follow from results in Wilson et al. [2020], although our results removes a Lipschitz assumption on the $\hat{D}_n^{(2)}$; see Appendix D.2 for a proof.

Our bounds are straightforward to compute; we can calculate the norms $\|x_n\|_2$ and evaluate the derivatives $\hat{D}_n^{(1)}$ and $\hat{D}_n^{(2)}$ at the known $x_n^T\hat{\theta}$. The only unknown quantity is L_n. However, we can upper bound the L_n using the following proposition.

Proposition 1. *Let $\mathcal{Z}_n$ be the set of $z \in \mathbb{R}$ such that $|z| \leq |x_n^T\hat{\theta}| + |\hat{D}_n^{(1)}|\|x_n\|_2^2/(N\lambda)$. For L_n as defined in Eq.* (8)*, we have the upper bound:*

$$L_n \leq M_n := \max_{z \in \mathcal{Z}_n} |\hat{D}_n^{(3)}(z)| \left(\frac{1}{N} \sum_{m:\, m \neq n}^{N} \|x_m\|_2^2 \right). \tag{11}$$

To compute an upper bound on the M_n in turn, we can optimize $\hat{D}_n^{(3)}(z)$ for $|z| \leq |x_n^T\hat{\theta}| + |\hat{D}_n^{(1)}|\|x_n\|_2^2/(N\lambda)$. This scalar problem is straightforward for common GLMs: for logistic regression, we can use the fact that $|\hat{D}_n^{(3)}| \leq 1/4$, and for Poisson regression with an exponential link function (i.e., $y_n \sim \text{Poisson}(\exp(x_n^T\theta))$), we maximize $\hat{D}_n^{(3)}(z) = e^z$ with the largest $z \in \mathcal{Z}_n$.

5 Approximating the quadratic forms Q_n

Algorithm 2 Estimate $Q_n = x_n^T(B + \lambda I_D)^{-1}x_n$ via a rank-K decomposition of PSD matrix B. *Note*: as written, this procedure is not numerically stable. See Appendix E.3 for an equivalent but numerically stable version.

procedure APPXQN(B, K, λ)
  for $k = 1, \ldots, K$ **do**
    $\mathcal{E}_k \leftarrow \mathcal{N}(0_D, I_D)$ ▷ $\mathcal{E} \in \mathbb{R}^{D \times K}$ has i.i.d. $\mathcal{N}(0,1)$ entries
  end for
  $\Omega \leftarrow$ ORTHONORMALIZECOLUMNS($\text{diag}\{1/(B_{dd} + \lambda)\}_{d=1}^D X^T X \mathcal{E}$) ▷ Proposition 3
  $M \leftarrow B\Omega$
  $\widetilde{H} \leftarrow M(\Omega^T M)^{-1}M^T + \lambda I_D$ ▷ Rank-K Nyström approximation of B
  for $n = 1, \ldots, N$ **do**
    $\widetilde{Q}_n \leftarrow \min\left\{x_n^T\widetilde{H}^{-1}x_n,\ \|x_n\|_2^2/(\lambda + \hat{D}_n^{(2)}\|x_n\|_2^2)\right\}$ ▷ Proposition 4
  end for
  return $\{\widetilde{Q}_n\}_{n=1}^N$
end procedure

The results of Section 4 imply that existing ACV methods achieve high accuracy on GLMs with ALR data. However, in high dimensions, the $O(D^3)$ cost of computing H^{-1} in the Q_n can be prohibitive. Koh and Liang [2017], Lorraine et al. [2020] have investigated an approximation to the matrix inverse for problems similar to ACV; however, in our experiments in Appendix B, we find that this method does not work well for ACV. Instead, we give approximations $\widetilde{Q}_n \approx Q_n$ in Algorithm 2 along with computable upper bounds on the error $|\widetilde{Q}_n - Q_n|$ in Proposition 4. When the data has ALR structure, so does the Hessian H; hence we propose a low-rank matrix approximation to H. This gives Algorithm 2 a runtime in $O(NDK + K^3)$, which can result in substantial savings relative to the $O(ND^2 + D^3)$ time required to exactly compute the Q_n. We will see that the main insights behind Algorithm 2 come from studying an upper bound on the approximation error when using a low-rank approximation.

Observe that by construction of Ω and $\widetilde{H}$ in Algorithm 2, the approximate Hessian $\widetilde{H}$ exactly agrees with H on the subspace Ω. We can compute an upper bound on the error $|x_n^T\widetilde{H}^{-1}x_n - Q_n|$ by recognizing that any error originates from components of x_n orthogonal to Ω:

Proposition 2. *Let $\lambda > 0$ and suppose there is some subspace $\mathcal{B}$ on which H and $\widetilde{H}$ exactly agree: $\forall v \in \mathcal{B}, Hv = \widetilde{H}v$. Then H^{-1} and $\widetilde{H}^{-1}$ agree exactly on the subspace $\mathcal{A} := H\mathcal{B}$, and*

$$|x_n^T\widetilde{H}^{-1}x_n - Q_n| \leq \frac{\left\|P_{\mathcal{A}}^{\perp}x_n\right\|_2^2}{\lambda}, \qquad \textit{for all } n = 1, \ldots, N, \tag{12}$$

where $P_{\mathcal{A}}^{\perp}$ denotes projection onto the orthogonal complement of $\mathcal{A}$.

For a proof, see Appendix E.1. The bound from Eq. (12) is easy to compute in $O(DK)$ time given a basis for $\mathcal{B}$. It also motivates the choice of Ω in Algorithm 2. In particular, Proposition 3 shows that Ω approximates the rank-K subspace $\mathcal{B}$ that minimizes the average of the bound in Eq. (12).

Proposition 3. *Let $V_{:K} \in \mathbb{R}^{D\times K}$ be the matrix with columns equal to the right singular vectors of X corresponding to the K largest singular values. Then the rank-K subspace $\mathcal{B}$ minimizing $\sum_n \|P_{\mathcal{A}}^{\perp} x_n\|_2^2$ is an orthonormal basis for the columns of $H^{-1}V_{:K}$.*

Proof. $\sum_n \|P_{\mathcal{A}}^{\perp} x_n\|_2^2 = \|(I_D - P_{\mathcal{A}})X^T\|_F^2$, where $\|\cdot\|_F$ denotes the Frobenius norm. Noting that the given choice of $\mathcal{B}$ implies that $\mathcal{A} = V_{:K}$, the result follows from the Eckart-Young theorem. □

We now see that the choice of Ω in Algorithm 2 approximates the optimal choice $H^{-1}V_{:K}$. In particular, we use a single iteration of the subspace iteration method [Bathe and Wilson, 1973] to approximate $V_{:K}$ and then multiply by the diagonal approximation $\operatorname{diag}\{1/H_{dd}\}_{d=1}^{D} \approx H^{-1}$. This approximation uses the top singular vectors of X. We expect these directions to be roughly equivalent to the largest eigenvectors of $B := \sum_n \hat{D}_n^{(2)} x_n x_n^T$, which in turn are the largest eigenvectors of $H = B + \lambda I_D$. Thus we are roughly approximating H by its *largest* K eigenvectors.

Why is it safe to neglect the small eigenvectors? At first glance this is strange, as to minimize the operator norm $\|H^{-1} - \widetilde{H}^{-1}\|_{op}$, one would opt to preserve the action of H along its *smallest* K eigenvectors. The key intuition behind this reversal is that we are interested in the action of H^{-1} in the direction of the datapoints x_n, which, on average, tend to lie near the largest eigenvectors of H.

Algorithm 2 uses one additional insight to improve the accuracy of its estimates. In particular, we notice that, by the definition of H, each x_n lies in an eigenspace of H with eigenvalue at least $\hat{D}_n^{(2)}\|x_n\|_2^2 + \lambda$. This observation undergirds the following result, which generates our final estimates $\widetilde{Q}_n \approx Q_n$, along with quickly computable bounds on their error $|\widetilde{Q}_n - Q_n|$.

Proposition 4. *The $Q_n = x_n^T H^{-1} x_n$ satisfy $0 < Q_n \leq \|x_n\|_2^2/(\lambda + \hat{D}_n^{(2)}\|x_n\|_2^2)$. Furthermore, letting $\widetilde{Q}_n := \min\{x_n^T \widetilde{H}^{-1} x_n,\ \|x_n\|_2^2/(\lambda + \hat{D}_n^{(2)}\|x_n\|_2^2)\}$, we have the error bound*

$$|\widetilde{Q}_n - Q_n| \leq \min\left\{\frac{\left\|P_{\mathcal{A}}^{\perp} x_n\right\|_2^2}{\lambda}, \frac{\|x_n\|_2^2}{\lambda + \hat{D}_n^{(2)}\|x_n\|_2^2}\right\}. \tag{13}$$

See Appendix E.1 for a proof. We finally note that Algorithm 2 strongly resembles algorithms from the randomized numerical linear algebra literature. Indeed, the work of Tropp et al. [2017] was the original inspiration for Algorithm 2, and Algorithm 2 can be seen as an instance of the algorithm presented in Tropp et al. [2017] with specific choices of various tuning parameters optimized for our application. For more on this perspective, see Appendix E.2.

6 Experiments

Algorithm 1 on real data. We begin by confirming the accuracy and speed of Algorithm 1 on real data compared to both exact CV and existing ACV methods. We apply logistic regression to two datasets (`p53` and `rcv1`) and Poisson regression to one dataset (`blog`). `p53` has a size of roughly $N = 8{,}000, D = 5{,}000$, and the remaining two have roughly $N = D = 20{,}000$; see Appendix G for more details. For all experiments we fix $\lambda = 5.0$. We choose this moderate value of λ to make the underlying optimization problems sufficiently regular so that exact CV's runtime would still be reasonable for our larger experiments. In Appendix H, we show that these results are not sensitive to the particular value of λ. To further speed up computation of exact LOOCV, we only run over twenty randomly chosen datapoints. We report average percent error, $(1/20)\sum_{b=1}^{20}|x_b^T \text{appx.} - x_b^T\hat{\theta}_{\setminus b}|/|x_b^T\hat{\theta}_{\setminus b}|$ for each exact ACV algorithm and the output of Algorithm 1. For the smaller dataset `p53`, the speedup of Algorithm 1 over exact $\text{NS}_{\setminus n}$ or $\text{IJ}_{\setminus n}$ is marginal; however, for the larger two datasets, our methods provide significant speedups: for the `blog` dataset, we estimate the runtime of full exact CV to be nearly ten months. By contrast, the runtime of $\text{NS}_{\setminus n}$ is nearly five minutes, and the runtime of $\widetilde{\text{NS}}_{\setminus n}$ is forty seconds. In general, the accuracy of $\widetilde{\text{IJ}}_{\setminus n}$ closely mirrors the accuracy of $\text{IJ}_{\setminus n}$, while the accuracy of $\widetilde{\text{NS}}_{\setminus n}$ can be somewhat worse than that

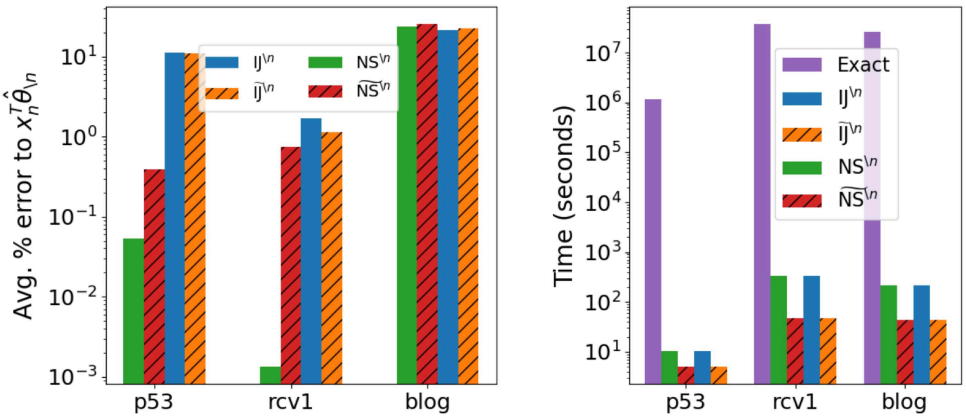

Figure 2: Experiments on real datasets. (*Left*): average percent error compared to exact CV on a subset of datapoints, $(1/20)\sum_{b=1}^{20}|x_b^T\text{approx.} - x_b^T\hat{\theta}_{\backslash b}|/|x_b^T\hat{\theta}_{\backslash b}|$, where approx. denotes $\text{NS}_{\backslash n}$, $\widetilde{\text{NS}}_{\backslash n}$, $\text{IJ}_{\backslash n}$, or $\widetilde{\text{IJ}}_{\backslash n}$. (*Right*): ACV runtimes with exact CV runtimes for comparison. ACV runtimes are given for all N datapoints. Exact CV runtimes are estimated runtimes for all N datapoints.

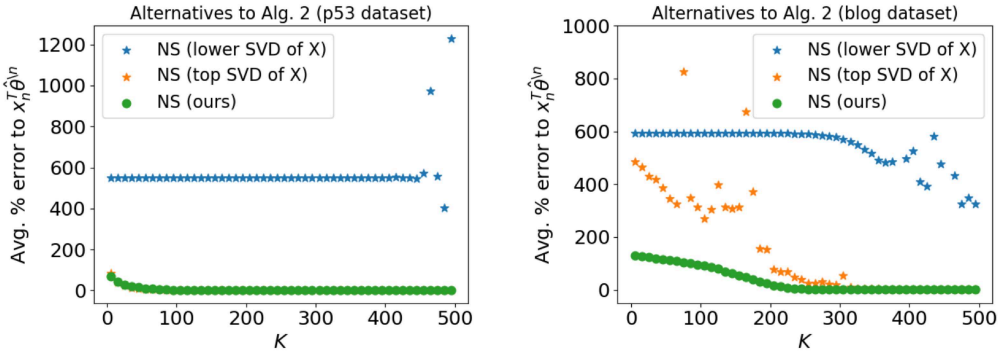

Figure 3: Comparisons to alternatives to Algorithm 1 for estimating $x_n^T\hat{\theta}_{\backslash n}$ via the NS approximation on small versions of two different real datasets: the `p53` dataset (a logistic regression task) and the `blog` dataset (a Poisson regression task). See Appendix G for dataset details. For the `p53` dataset (left), the results of our algorithm (green) and the top SVD of X (orange) are visually indistinguishable. For the `blog` dataset (right), both the lower and upper SVD of X obtain errors larger than the displayed scale; we cut off the vertical scale at 1,000% error so that finer details are visible.

of $\text{NS}_{\backslash n}$ on the two logistic regression tasks; however, we note that in these cases, the error of $\widetilde{\text{NS}}_{\backslash n}$ is still less than 1%.

Alternatives for estimating Q_n. Given our use of low-rank approximations to $H = X^T\text{diag}\{\hat{D}_n^{(2)}\}_{n=1}^N X + \lambda I_D$ to approximate Q_n, one might first consider a more straightforward option before reaching for Algorithm 2. In particular, one might consider using principle components analysis, which is a common method for dimensionality reduction in generalized linear models. Here, this corresponds to taking the SVD of X and then computing H^{-1} using only the top-K singular vectors and values. Additionally, as discussed in Section 5, one might consider using the *lower* K singular vectors and values given our use of H^{-1}. In Fig. 3, we study the performance of these options on smaller versions of the `p53` and `blog` datasets. We see the value of our analysis in Propositions 2 and 3, as the lower singular vectors give an extremely poor approximation to CV. Further, we see the necessity of the truncation present in our final estimate $\widetilde{Q}_n$ in Proposition 4. In particular, recall that the $\text{NS}_{\backslash n}$ approximation depends on $1/(1-\hat{D}_n^{(2)}Q_n)$. Thus if, for any values of

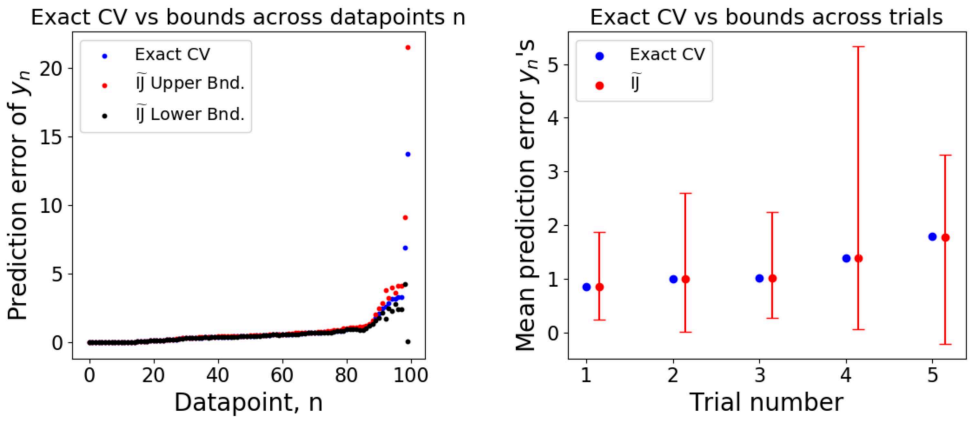

Figure 4: Error bounds implied by Theorem 1 on $\widetilde{\mathrm{IJ}}_{\setminus n}$'s estimate of out-of-sample error using squared loss, $\mathrm{Err}(x_n^T\theta, y_n) = (e^{x_n^T\theta} - y_n)^2$. (*Left*): Per datapoint error bounds. The bound is fairly loose for datapoints with larger squared loss, but tighter for points with lower squared loss. (*Right*): Five trials with estimates averaged across all n. We compute the upper (respectively, lower) error bars for $\widetilde{\mathrm{IJ}}_{\setminus n}$ by averaging the upper (respectively, lower) bounds. While our bounds overstate the difference between exact CV and $\widetilde{\mathrm{IJ}}_{\setminus n}$, they still give non-vacuous information on value of exact CV.

n and K, the estimate of $\hat{D}_n^{(2)}Q_n$ passes near 1, the resulting estimate of $\mathrm{NS}_{\setminus n}$ will become unstable. We observe this instability for $K \leq \sim 300$ for the top-K SVD of X on the right of Fig. 3.

Accuracy of error bounds. We next empirically check the accuracy of the error bounds from Theorem 1. We generate a synthetic Poisson regression problem with i.i.d. covariates $x_{nd} \sim \mathcal{N}(0,1)$ and $y_n \sim \mathrm{Poisson}(e^{x_n^T\theta^*})$, where $\theta^* \in \mathbb{R}^D$ is a true parameter with i.i.d. $\mathcal{N}(0,1)$ entries. We generate a dataset of size $N = 800$ and $D = 500$ with covariates of approximate rank 50. To speed up the runtime of exact CV, we choose a moderate value of $\lambda = 1.0$. In Fig. 4, we illustrate the utility of the bounds from Theorem 1 by estimating the out-of-sample loss with $\mathrm{Err}(x_n^T\theta, y_n) = (e^{x_n^T\theta} - y_n)^2$. Across five trials, we show the results of exact LOOCV, our estimates provided by $\widetilde{\mathrm{IJ}}_{\setminus n}$, and the bounds on the error of $\widetilde{\mathrm{IJ}}_{\setminus n}$ given by Theorem 1. While our error bars in Fig. 4 tend to overestimate the difference between $\widetilde{\mathrm{IJ}}_{\setminus n}$ and exact CV, they typically provide upper bounds on exact CV on the order of the exact CV estimate itself. In some cases, we have observed that the error bars can overestimate exact CV by many orders of magnitude (see Appendix F), but this failure is usually due to one or two datapoints n for which the bound is vacuously large. As these failure cases are easy to spot by inspection, a simple fix is to resort to exact CV just for these datapoints.

7 Conclusions

We provide an algorithm to approximate CV accurately and quickly in high-dimensional GLMs with ALR structure. Additionally, we provide quickly computable upper bounds on the error of our algorithm. We see two major directions for future work. First, while our theory and experiments focus on ACV for model assessment, the recent work of Wilson et al. [2020] has provided theoretical results on ACV for model selection (e.g. choosing λ). It would be interesting to see how dimensionality and ALR data plays a role in this setting. Second, as noted in the introduction, we hope that the results here will provide a springboard for studying ALR structure in models beyond GLMs and CV schemes beyond LOOCV.

Broader Impact

In general, we feel that work assessing the accuracy of machine learning models will have a positive impact on society. As machine learning is deployed in areas in which mistakes could have adverse effect on peoples' lives, it is important that we understand the error rate of such decisions before deployment. On the other hand, machine learning models can (and are) used for harm and the methods in this paper may assist in the development in such models. Additionally, there is always a risk in introducing any sort of approximation, as it may fail silently and unexpectedly in practice; e.g., our approximations might incorrectly lead a practitioner to conclude that their machine learning model has very small error when the opposite is in fact true. While we believe the computable upper bounds provided here somewhat mitigate this issue, we still remain cautious (though optimistic) about applying ACV methods in practice. Finally, we note that an implicit assumption throughout our work is that computing exact CV is something we want; that is, exact CV provides a good estimate of out-of-sample error. While this seems to be generally true, this does add another failure mode to our algorithm. In particular, even if we provide an accurate approximation to exact CV, it may be that exact CV itself is misleading.

Acknowledgements

The authors thank Zachary Frangella for helpful conversations. WS and TB were supported by the CSAIL-MSR Trustworthy AI Initiative, an NSF CAREER Award, an ARO Young Investigator Program Award, ONR Award N00014-17-1-2072, and Amazon. MU was supported by NSF Award IIS-1943131, the ONR Young Investigator Program, and DARPA Award FA8750-17-2-0101. The Broderick Group is also supported by the Sloan Foundation, ARPA-E, United States Department of the Air Force, and MIT Lincoln Laboratory.

References

N. Agarwal, B. Bullins, and E. Hazan. Second-order stochastic optimization in linear time. *Journal of Machine Learning Research*, 2017.

K. J. Bathe and E. L. Wilson. Solution methods for eigenvalue problems in structural mechanics. *International Journal for Numerical Methods in Engineering*, 6:213–226, 1973.

A. Beirami, M. Razaviyayn, S. Shahrampour, and V. Tarokh. On optimal generalizability in parametric learning. In *Advances in Neural Information Processing Systems (NIPS)*, pages 3458–3468, 2017.

P. Burman. A comparative study of ordinary cross-validation, v-fold cross-validation and the repeated learning-testing methods. *Biometrika*, 76, September 1989.

K. Buza. Feedback prediction for blogs. In *Data Analysis, Machine Learning and Knowledge Discovery*, pages 145–152. Springer International Publishing, 2014.

S. A. Danziger, S. J. Swamidass, J. Zeng, L. R. Dearth, Q. Lu, J. H. Chen, J. Cheng, V. P. Hoang, H. Saigo, R. Luo, P. Baldi, R. K. Brachmann, and R. H. Lathrop. Functional census of mutation sequence spaces: the example of p53 cancer rescue mutants. *IEEE/ACM transactions on computational biology and bioinformatics*, 3, 2006.

S. A. Danziger, J. Zeng, Y. Wang, R. K. Brachmann, and R. H. Lathrop. Choosing where to look next in a mutation sequence space: active learning of informative p53 cancer rescue mutants. *Bioinformatics*, 23, 2007.

S. A. Danziger, R. Baronio, L. Ho, L. Hall, K. Salmon, G. W. Hatfield, P. Kaiser, and R. H. Lathrop. Predicting positive p53 cancer rescue regions using most informative positive (MIP) active learning. *PLOS computational biology*, 5, 2009.

B. Efron. *The Jackknife, the Bootstrap, and Other Resampling Plans*, volume 38. Society for Industrial and Applied Mathematics, 1982.

S. Geisser. The predictive sample reuse method with applications. *Journal of the American Statistical Association*, 70(350):320–328, June 1975.

R. Giordano, W. T. Stephenson, R. Liu, M. I. Jordan, and T. Broderick. A Swiss army infinitesimal jackknife. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, April 2019.

T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer-Verlag, 2009.

L. Jaeckel. The infinitesimal jackknife, memorandum. Technical report, MM 72-1215-11, Bell Lab. Murray Hill, NJ, 1972.

P. W. Koh and P. Liang. Understanding black-box predictions via influence functions. In *International Conference in Machine Learning (ICML)*, 2017.

P. W. Koh, K. S. Ang, H. Teo, and P. Liang. On the accuracy of influence functions for measuring group effects. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

D. D. Lewis, Y. Yang, T. G. Rose, and F. Li. RCV1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research*, 5, 2004.

J. Lorraine, P. Vicol, and D. Duvenaud. Optimizing millions of hyperparameters by implicit differentiation. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2020.

P. G. Martinsson and J. Tropp. Randomized numerical linear algebra: foundations & algorithms. *arXiv Preprint*, aug 2020.

P. McCullaugh. *Generalized Linear Models*. Chapman and Hall/CRC, 2nd edition, 1989.

K. Musgrave, S. Belongie, and S. N. Lim. A machine learning reality check. *arXiv Preprint*, March 2020.

T. Obuchi and Y. Kabashima. Cross validation in LASSO and its acceleration. *Journal of Statistical Mechanics*, May 2016.

T. Obuchi and Y. Kabashima. Accelerating cross-validation in multinomial logistic regression with l1-regularization. *Journal of Machine Learning Research*, September 2018.

K. R. Rad and A. Maleki. A scalable estimate of the extra-sample prediction error via approximate leave-one-out. *arXiv Preprint*, January 2020.

W. T. Stephenson and T. Broderick. Approximate cross-validation in high dimensions with guarantees. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, June 2020.

M. Stone. Cross-validatory choice and assessment of statistical predictions. *Journal of the American Statistical Association*, 36(2):111–147, 1974.

J. Tropp, A. Yurtsever, M. Udell, and V. Cevher. Fixed-rank approximation of a positive-semidefinite matrix from streaming data. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

M. Udell and A. Townsend. Why are big data matrices approximately low rank? *SIAM Journal on Mathematics of Data Science (SIMODS)*, 2019.

A. Wilson, M. Kasy, and L. Mackey. Approximate cross-validation: guarantees for model assessment and selection. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2020.

A Derivation of $x_n^T \mathrm{NS}_{\backslash n}$ and $x_n^T \mathrm{IJ}_{\backslash n}$

Here, we derive the expressions for $x_n^T \mathrm{NS}_{\backslash n}$ and $x_n^T \mathrm{IJ}_{\backslash n}$ given in Eqs. (3) and (4). We recall from previous work (e.g., see Stephenson and Broderick [2020, Appendix C] for a summary) that the LOOCV parameter estimates given by the Newton step and infinitesimal jackknife approximations are given by

$$\hat{\theta}_{\backslash n} \approx \mathrm{NS}_{\backslash n} := \hat{\theta} + \frac{1}{N}\left(\sum_{m:\, m \neq n}^{N} \hat{D}_n^{(2)} x_m x_m^T + \lambda I_D\right)^{-1} \hat{D}_n^{(1)} x_n \tag{14}$$

$$\hat{\theta}_{\backslash n} \approx \mathrm{IJ}_{\backslash n} := \hat{\theta} + \frac{1}{N}\left(\sum_{n=1}^{N} \hat{D}_n^{(2)} x_n x_n^T \lambda I_D\right)^{-1} \hat{D}_n^{(1)} x_n. \tag{15}$$

Taking the inner product of $\mathrm{IJ}_{\backslash n}$ with x_n immediately gives Eq. (4). To derive Eq. (3), define $H := \sum_n \hat{D}_n^{(2)} x_n x_n^T + \lambda I_D$ and note that we can rewrite $\mathrm{NS}_{\backslash n}$ using the Sherman-Morrison formula:

$$\mathrm{NS}_{\backslash n} = \hat{\theta} + \frac{1}{N}\left[\hat{D}_n^{(1)} H^{-1} x_n + \hat{D}_n^{(1)} \hat{D}_n^{(2)} \frac{H^{-1} x_n x_n^T H^{-1}}{1 - \hat{D}_n^{(2)} Q_n} x_n\right].$$

Taking the inner product with x_n and reorganizing gives:

$$x_n^T \mathrm{NS}_{\backslash n} = x_n^T \hat{\theta} + \frac{\hat{D}_n^{(1)}}{N}\left[\frac{Q_n - \hat{D}_n^{(2)} Q_n^2}{1 - \hat{D}_n^{(2)} Q_n} + \frac{\hat{D}_n^{(2)} Q_n^2}{1 - \hat{D}_n^{(2)} Q_n}\right] = x_n^T \hat{\theta} + \frac{\hat{D}_n^{(1)}}{N} \frac{Q_n}{1 - \hat{D}_n^{(2)} Q_n}.$$

B Comparison to existing Hessian inverse approximation

We note that two previous works have used inverse Hessian approximations for applications similar to ACV. Koh and Liang [2017] use influence functions to estimate behavior of black box models, and Lorraine et al. [2020] use the implicit function theorem to optimize model hyperparameters. In both papers, the authors need to multiply an inverse Hessian by a gradient. To deal with the high dimensional expense associated with this matrix inverse, both sets of authors use the method of Agarwal et al. [2017], who propose a stochastic approximation to the *Neumann series*. The Neumann series writes the inverse of a matrix H with operator norm $\|H\|_{op} < 1$ as:

$$H^{-1} = \sum_{k=0}^{\infty} (I - H)^k.$$

The observation of Agarwal et al. [2017] is that this series can be written recursively, as well as estimated stochastically if one has random variables A_s with $\mathbb{E}[A_s] = H$. In the general case of empirical risk minimization with an objective of $(1/N)\sum_{n=1}^{N} f_n(\theta)$, Agarwal et al. [2017] propose using $A_s = \nabla^2 f_s(\theta)$ for some $s \in [N]$ chosen uniformly at random. In the GLM setting we are interested in here, we choose an index $s \in [N]$ uniformly at random and set $A_s = \hat{D}_n^{(2)} x_s x_s^T + (\lambda/N) I_D$. Then, for $s = 1, \ldots, S$, we follow Agarwal et al. [2017] to recursively define:

$$H^{-1} \approx \bar{H}_s^{-1} := I_D + (I - A_s)\bar{H}_{s-1}^{-1}.$$

The final recommendation of Agarwal et al. [2017] is to repeat this process M times and average the results. We thus have two hyperparameters to choose: S and M.

To test out the Agarwal et al. [2017] approximation against our approximation in Algorithm 1, we generate Poisson regression datasets of increasing sizes N and D. We generate approximately low-rank covariates x_n by drawing $x_{nd} \sim N(0, 1)$ for $d = 1, \ldots, 1{,}000$ and $x_{nd} \sim N(0, 0.01)$ for $d = 1{,}001, \ldots, D$; for our dataset with $D = 40$, we follow the same procedure but with $R = 20$ instead. For each dataset, we compute $\mathrm{IJ}_{\backslash n}$, as well as our approximation $\widetilde{\mathrm{IJ}}_{\backslash n}$ from Algorithm 1. We run Algorithm 1 for $K = 1, 100, 200, \ldots, D$ and run the stochastic Neumann series approximation with all combinations of $M \in \{2, 5\}$ and $S \in \{1, 5, 10, 15, \ldots, 200\}$. We measure the accuracy of

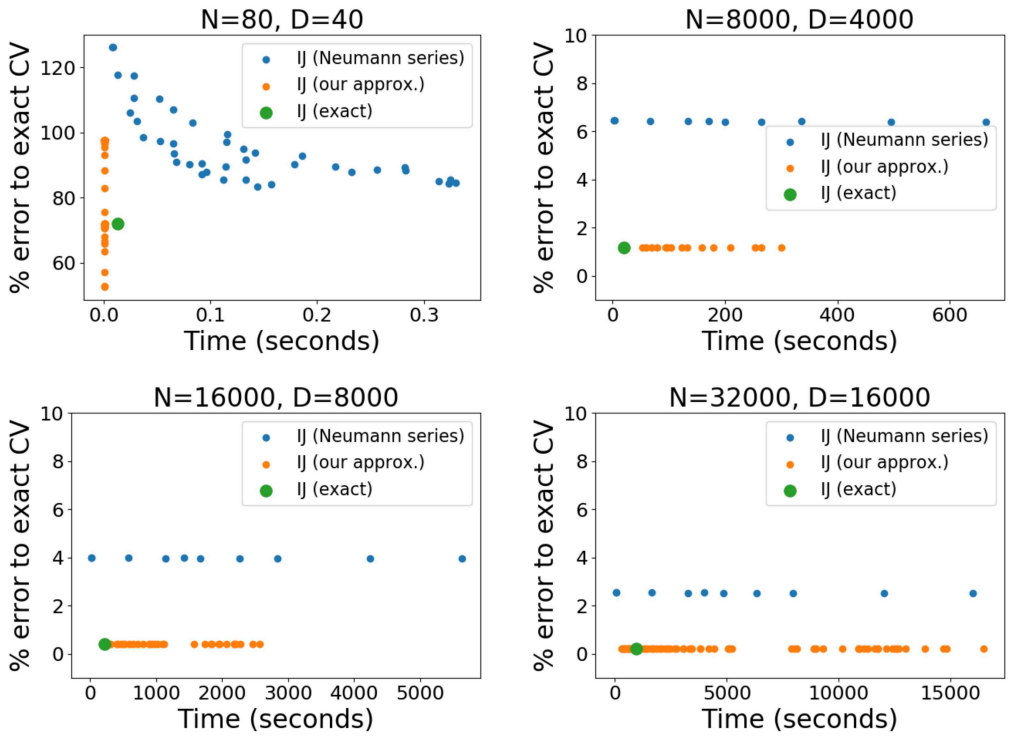

Figure 5: Experiment from Appendix B. Across four different dataset sizes, using the Neumann series approximation (orange) does not show any noticeable improvement on the time scale of running our approximation (green) for all possible values of K.

all approximations as percent error to exact CV ($x_n^T \hat{\theta}_{\backslash n}$). We show in Fig. 5 that our approximation has far improved error in far less time. Notably, this phenomenon becomes more pronounced as the dimension gets higher; while spending more computation on the Neumann series approximation does noticeably decrease the error for the $N = 80, D = 40$ case, we see that as soon as we step into even moderate dimensions (D in the thousands), spending more computation on the Neumann approximation does not noticeably decrease the error. In fact, in the three lowest-dimensional experiments here, the dimension is so low that exactly computing H^{-1} via a Cholesky decomposition is the fastest method.

We also notice that in the $N = 80, D = 40$ experiment, $\widetilde{\text{IJ}}_{\backslash n}$ is a better approximation of exact CV than is $\text{IJ}_{\backslash n}$ for intermediate values of K (i.e. some orange dots sit below the large green dot). We note that we have observed this behavior in a variety of synthetic and real-data experiments. We do not currently have further insight into this phenomenon and leave its investigation for future work.

C Previous ACV theory

We briefly review pre-existing theoretical results on the accuracy of ACV. Theoretical results for the accuracy of $\text{IJ}_{\backslash n}$ are given by Giordano et al. [2019], Koh et al. [2019], Wilson et al. [2020]. Giordano et al. [2019] give a $O(1/N^2)$ error bound for unregularized problems, which Stephenson and Broderick [2020, Proposition 2] extends to regularized prolems; however, in our GLM case here, both results require the covariates and parameter space to be bounded. Koh et al. [2019] give a similar bound, but require the Hessian to be Lipschitz, and their bounds rely on the inverse of the minimum singular value of H, making them unsuited for describing the low rank case of interest here. The bounds of Wilson et al. [2020] are close to our bounds in Lemma 2. The difference to our work is that Wilson et al. [2020] consider generic (i.e. not just GLM) models, but also require a Lipschitz assumption on the Hessian. We specialize to GLMs, avoid the Lipschitz assumption by noting that it only need hold locally, and provide fully computable bounds.

Various theoretical guarantees also exist for the quality of $\mathrm{NS}_{\backslash n}$ from Eq. (3). Rad and Maleki [2020] show that the error $\|\mathrm{NS}_{\backslash n} - \hat{\theta}_{\backslash n}\|_2$ is $o(1/N)$ as $N \to \infty$ and give conditions under which the error is a much slower $O(1/\sqrt{N})$ as both $N, D \to \infty$ with N/D converging to a constant. Beirami et al. [2017] show that the error is $O(1/N^2)$, but require fairly strict assumptions (namely, boundedness of the covariates and parameter space). Koh et al. [2019], Wilson et al. [2020] provide what seem to be the most interpretable bounds, but, as is the case for $\mathrm{IJ}_{\backslash n}$, both require a Lipschitz assumption on the Hessian and the results of Koh et al. [2019] depend on the lowest singular value of the Hessian.

D Proofs from Sections 3 and 4

D.1 Proving accuracy of $\mathrm{NS}_{\backslash n}$ and $\mathrm{IJ}_{\backslash n}$ under exact low-rank data (Lemma 1)

Here, we prove that, when the covariate matrix is exactly rank $R \ll D$, the accuracy of $\mathrm{NS}_{\backslash n}$ and $\mathrm{IJ}_{\backslash n}$ behaves exactly as in a dimension $R \ll D$ problem. Let $X = U\Sigma V$ be the singular value decomposition of X, where Σ is a diagonal matrix with only R non-zero entries; let $V_{:R} \in \mathbb{R}^{D\times R}$ be the right singular vectors of X corresponding to these R non-zero singular values. We define the restricted, R-dimensional problem with covariates $\tilde{x}_n := V_{:R}^T x_n$ as:

$$\hat{\phi} := \underset{\phi\in\mathbb{R}^R}{\arg\min} \frac{1}{N}\sum_{n=1}^{N} f(\tilde{x}_n^T\phi) + \frac{\lambda}{2}\|\phi\|_2^2 . \tag{16}$$

Let $\hat{\phi}_{\backslash n}$ be the solution to the leave-one-out version of this problem and $\mathrm{RIJ}_{\backslash n}$ and $\mathrm{RNS}_{\backslash n}$ the application of Eqs. (3) and (4) to this problem. We then have the following proposition, which implies the statement of Lemma 1.

Proposition 5 (Generalization of Lemma 1)**.** *The following hold for all datapoints n:*

$$\begin{aligned} x_n^T\hat{\theta}_{\backslash n} &= \tilde{x}_n^T\hat{\phi}_{\backslash n} \\ x_n^T\mathrm{IJ}_{\backslash n} &= \tilde{x}_n^T\mathrm{RIJ}_{\backslash n} \\ x_n^T\mathrm{NS}_{\backslash n} &= \tilde{x}_n^T\mathrm{RNS}_{\backslash n}. \end{aligned}$$

In particular, $|x_n^T\mathrm{NS}_{\backslash n} - x_n^T\hat{\theta}_{\backslash n}| = |\tilde{x}_n^T\mathrm{RNS}_{\backslash n} - \tilde{x}_n^T\hat{\phi}_{\backslash n}|$ and $|x_n^T\mathrm{IJ}_{\backslash n} - x_n^T\hat{\theta}_{\backslash n}| = |\tilde{x}_n^T\mathrm{RIJ}_{\backslash n} - \tilde{x}_n^T\hat{\phi}_{\backslash n}|$, as claimed in Lemma 1.

Proof. First, note that if $\hat{\phi}$ is an optimum of Eq. (16), then $(1/N)\sum_n \hat{D}_n^{(1)} V_{:R}^T x_n + \lambda\hat{\phi} = 0$. As $V_{:R}V_{:R}^T x_n = x_n$, we have that $\hat{\theta} = V_{:R}\hat{\phi}$ is optimal for the full, D-dimensional, problem. This implies that $\hat{\phi} = V_{:R}^T\hat{\theta}$, and thus $x_n^T\hat{\theta} = \tilde{x}_n^T\hat{\phi}$. The same reasoning shows that $x_n^T\hat{\theta}_{\backslash n} = \tilde{x}_n^T\hat{\phi}_{\backslash n}$.

Now, notice that the Hessian of the restricted problem, H_R, is given by $H_R = (1/N)\sum_n V_{:R}^T x_n x_n^T V_{:R}\hat{D}_n^{(2)} + \lambda I_R \implies H_R^{-1} = V_{:R}^T H^{-1} V_{:R}$, where the $\hat{D}_n^{(2)}$ are evaluated at $\tilde{x}_n^T\hat{\phi} = x_n^T\hat{\theta}$. Also, the gradients of the restricted problem are given by $\nabla_\phi f(\tilde{x}_n^T\hat{\phi}, y_n) = \hat{D}_n^{(1)} V_{:R}^T x_n$. Thus the restricted IJ is:

$$\mathrm{RIJ}_{\backslash n} = \hat{\phi} + H_R^{-1}V_{:R}^T x_n \hat{D}_n^{(1)} = V_{:R}^T\left(\hat{\theta} + H^{-1}x_n\hat{D}_n^{(1)}\right) = V_{:R}^T\mathrm{IJ}_{\backslash n}.$$

Thus, we have $\tilde{x}_n^T\mathrm{RIJ}_{\backslash n} = x_n^T V_{:R}V_{:R}^T\mathrm{IJ}_{\backslash n} = (V_{:R}V_{:R}^T x_n)^T\mathrm{IJ}_{\backslash n}$. Using $V_{:R}V_{:R}^T x_n = x_n$, we have that $\tilde{x}_n^T\mathrm{RIJ}_{\backslash n} = x_n^T\mathrm{IJ}_{\backslash n}$. The proof that $\tilde{x}_n^T\mathrm{RNS}_{\backslash n} = x_n^T\mathrm{NS}_{\backslash n}$ is identical. □

D.2 Proving accuracy of $\mathrm{NS}_{\backslash n}$ and $\mathrm{IJ}_{\backslash n}$ under ALR data (Lemma 2)

We will first need a few lemmas relating to how the exact solutions $\hat{\theta}_{\backslash n}$ and $\hat{\theta}$ vary as we leave datapoints out and move from exactly low-rank to ALR. We start by bounding $\|\hat{\theta} - \hat{\theta}_{\backslash n}\|_2$; this result and its proof are from Wilson et al. [2020, Lemma 16] specialized to our GLM context.

Lemma 3. *Assume that $\lambda > 0$. Then:*

$$\left\|\hat{\theta} - \hat{\theta}_{\backslash n}\right\|_2 \leq \frac{1}{N\lambda}|\hat{D}_n^{(1)}|\,\|x_n\|_2 . \tag{17}$$

Proof. Let $F^{\setminus n}$ be the leave-one-out objective, $F^{\setminus n}(\theta) = (1/N)\sum_{m:\, m\neq n} f(x_m^T\theta, y_m) + (\lambda/2)\|\theta\|_2^2$. As $F^{\setminus n}$ is strongly convex with parameter λ, we have:

$$\lambda \left\|\hat{\theta} - \hat{\theta}_{\setminus n}\right\|_2^2 \leq \langle \hat{\theta} - \hat{\theta}_{\setminus n}, \nabla F^{\setminus n}(\hat{\theta}) - \nabla F^{\setminus n}(\hat{\theta}_{\setminus n})\rangle.$$

Now, use the fact that $\nabla F^{\setminus n}(\hat{\theta}_{\setminus n}) = \nabla F(\hat{\theta}) = 0$ and then that $F^{\setminus n} - F = (1/N) f(x_n^T\theta)$ to get:

$$\begin{aligned} &= \langle \hat{\theta} - \hat{\theta}_{\setminus n}, \nabla F^{\setminus n}(\hat{\theta}) - \nabla F(\hat{\theta})\rangle = \langle \hat{\theta} - \hat{\theta}_{\setminus n}, \nabla f(x_n^T\hat{\theta})\rangle \\ &\leq \left\|\hat{\theta} - \hat{\theta}_{\setminus n}\right\|_2 |\hat{D}_n^{(1)}|\, \|x_n\|_2 \,. \end{aligned}$$

□

We will need a bit more notation to discuss the ALR and exactly low-rank versions of the same problem. Suppose we have a $N \times D$ covariate matrix X that is exactly low-rank (ELR) with rows $x_{n,ELR} \in \mathbb{R}^D$. Then, suppose we form some approximately low-rank (ALR) covariate matrix by adding $\varepsilon_n \in \mathbb{R}^D$ to all x_n such that $X\varepsilon_n = 0$ for all ε_n. Let $x_{n,ALR}$ be the rows of this ALR matrix. Let $\hat{\theta}_{ELR}$ be the fit with the ELR data and $\hat{\theta}_{ALR}$ the fit with the ALR data. Finally, define the scalar derivatives:

$$\begin{aligned} \hat{D}_{n,ELR}^{(1)}(\theta) &:= \frac{df(z, y_n)}{dz}\Big|_{z=\langle x_{n,ELR},\theta\rangle} \\ \hat{D}_{n,ALR}^{(1)}(\theta) &:= \frac{df(z, y_n)}{dz}\Big|_{z=\langle x_{n,ALR},\theta\rangle} \end{aligned}$$

We can now give an upper bound on the difference between the ELR and ALR fits $\|\hat{\theta}_{ELR} - \hat{\theta}_{ALR}\|_2$. Our bound will imply that the $\hat{\theta}_{ALR}$ is a continuous function of the ε_n, which in turn are continuous functions of the singular values of the ALR covariate matrix.

Lemma 4. *Assume $\lambda > 0$. We have:*

$$\left\|\hat{\theta}_{ELR} - \hat{\theta}_{ALR}\right\|_2 \leq \frac{1}{N\lambda}\left\|\sum_{n=1}^N \hat{D}_{n,ELR}^{(1)}(\hat{\theta}_{ELR})\varepsilon_n\right\|_2$$

In particular, $\hat{\theta}_{ALR}$ is a continuous function of the ε_n around $\varepsilon_1, \ldots, \varepsilon_N = 0$.

Proof. Denote the ALR objective by $F_{ALR}(\theta) = (1/N)\sum_n f(x_{n,ALR}^T\theta) + \lambda\|\theta\|_2^2$. Then, via a Taylor expansion of its gradient around $\hat{\theta}_{ALR}$:

$$\nabla_\theta F_{ALR}(\hat{\theta}_{ELR}) = \nabla_\theta F_{ALR}(\hat{\theta}_{ALR}) + \nabla_\theta^2 F_{ALR}(\tilde{\theta})(\hat{\theta}_{ELR} - \hat{\theta}_{ALR}),$$

where $\tilde{\theta} \in \mathbb{R}^D$ satisfies $\tilde{\theta}_d = (1 - s_d)\theta_{ALR,d} + s_d\theta_{ELR,d}$ for some $s_d \in [0, 1]$ for each $d = 1, \ldots, D$. Via strong convexity and $\nabla_\theta F_{ALR}(\hat{\theta}_{ALR}) = 0$, we have:

$$\left\|\hat{\theta}_{ELR} - \hat{\theta}_{ALR}\right\|_2 \leq \frac{1}{\lambda}\left\|\nabla_\theta F_{ALR}(\hat{\theta}_{ELR})\right\|_2.$$

Now, note that the gradient on the right hand side of this equation is equal to

$$\nabla_\theta F_{ALR}(\hat{\theta}_{ELR}) = \frac{1}{N}\sum_{n=1}^N \hat{D}_{n,ALR}^{(1)}(\hat{\theta}_{ELR})x_{n,ELR} + \frac{1}{N}\sum_{n=1}^N \hat{D}_{n,ALR}^{(1)}(\hat{\theta}_{ELR})\varepsilon_n + \lambda\hat{\theta}_{ELR}. \quad (18)$$

By the optimality of $\hat{\theta}_{ELR}$ for the exactly low-rank problem, we must have that $\langle \varepsilon_n, \hat{\theta}_{ELR}\rangle = 0$ for all n; in particular, this implies that $\langle x_{n,ELR}, \hat{\theta}_{ELR}\rangle = \langle x_{n,ALR}, \hat{\theta}_{ELR}\rangle$, which in turn implies $\hat{D}_{n,ALR}^{(1)}(\hat{\theta}_{ELR}) = \hat{D}_{n,ELR}^{(1)}(\hat{\theta}_{ELR})$ for all n. Also by the optimality of $\hat{\theta}_{ELR}$, we have $(1/N)\sum_n \hat{D}_{n,ELR}^{(1)}(\hat{\theta}_{ELR})x_n + \lambda\hat{\theta}_{ELR} = 0$. Thus we have that Eq. (18) reads:

$$\nabla_\theta F_{ALR}(\hat{\theta}_{ELR}) = \frac{1}{N}\sum_{n=1}^N \hat{D}_{n,ELR}^{(1)}(\hat{\theta}_{ELR})\varepsilon_n,$$

which completes the proof. □

We now restate and prove Lemma 2.

Lemma 2. *Assume that $\lambda > 0$ and recall the definition of L_n from Eq.* (8)*. Then, for all n:*

$$|x_n^T \mathrm{NS}_{\backslash n} - x_n^T \hat{\theta}_{\backslash n}| \leq \frac{L_n}{N^2\lambda^3} |\hat{D}_n^{(1)}|^2 \left\|x_n\right\|_2^3 \tag{19}$$

$$|x_n^T \mathrm{IJ}_{\backslash n} - x_n^T \hat{\theta}_{\backslash n}| \leq \frac{L_n}{N^2\lambda^3} |\hat{D}_n^{(1)}|^2 \left\|x_n\right\|_2^3 + \frac{1}{N^2\lambda^2} |\hat{D}_n^{(1)}| |\hat{D}_n^{(2)}| \left\|x_n\right\|_2^4. \tag{20}$$

Furthermore, these bounds continuously decay as the data move from exactly to approximately low rank in that they are continuous in the singular values of X.

Proof. The proof of Eqs. (19) and (20) strongly resembles the proof of Wilson et al. [2020, Lemma 17] specialized to our current context. We first prove Eq. (19). We begin by applying the Cauchy-Schwarz inequality to get:

$$|x_n^T \mathrm{NS}_{\backslash n} - x_n^T \hat{\theta}_{\backslash n}| \leq \left\|x_n\right\|_2 \left\|\mathrm{NS}_{\backslash n} - \hat{\theta}_{\backslash n}\right\|_2.$$

The remainder of our proof focuses on bounding $\|\mathrm{NS}_{\backslash n} - \hat{\theta}_{\backslash n}\|_2$. Let $\widetilde{F^{\backslash n}}$ be the second order Taylor expansion of $F^{\backslash n}$ around $\hat{\theta}$; then $\mathrm{NS}_{\backslash n}$ is the minimizer of $\widetilde{F^{\backslash n}}$. By the strong convexity of $\widetilde{F^{\backslash n}}$:

$$\lambda \left\|\hat{\theta}_{\backslash n} - \mathrm{NS}_{\backslash n}\right\|_2^2 \leq \langle \mathrm{NS}_{\backslash n} - \hat{\theta}_{\backslash n}, \nabla \widetilde{F^{\backslash n}}(\mathrm{NS}_{\backslash n}) - \nabla \widetilde{F^{\backslash n}}(\hat{\theta}_{\backslash n}) \rangle \tag{21}$$

$$= \langle \mathrm{NS}_{\backslash n} - \hat{\theta}_{\backslash n}, \nabla F^{\backslash n}(\hat{\theta}_{\backslash n}) - \nabla \widetilde{F^{\backslash n}}(\hat{\theta}_{\backslash n}) \rangle \tag{22}$$

Now the goal is to bound this quantity as the remainder in a Taylor expansion. To this end, define $r(\theta) := \langle \hat{\theta}_{\backslash n} - \mathrm{NS}_{\backslash n}, \nabla F^{\backslash n}(\theta) \rangle$. To apply Taylor's theorem with integral remainder, define $g(t) := r((1-t)\hat{\theta} + t\hat{\theta}_{\backslash n})$ for $t \in [0, 1]$. Then, by a zeroth order Taylor expansion:

$$g(1) = g(0) + g'(0) + \int_0^1 (g'(s) - g'(0))\, ds.$$

Putting in the values of g and its derivatives:

$$\langle \hat{\theta}_{\backslash n} - \mathrm{NS}_{\backslash n}, \nabla F^{\backslash n}(\hat{\theta}_{\backslash n}) \rangle = \langle \hat{\theta}_{\backslash n} - \mathrm{NS}_{\backslash n}, \nabla F^{\backslash n}(\hat{\theta}) \rangle + \langle \hat{\theta}_{\backslash n} - \mathrm{NS}_{\backslash n}, \nabla^2 F^{\backslash n}(\hat{\theta}) \left(\hat{\theta}_{\backslash n} - \hat{\theta}\right) \rangle$$
$$+ \int_0^1 \langle \hat{\theta}_{\backslash n} - \mathrm{NS}_{\backslash n}, \left(\nabla^2 F^{\backslash n}((1-s)\hat{\theta} + s\hat{\theta}_{\backslash n}) - \nabla^2 F^{\backslash n}(\hat{\theta})\right) \left(\hat{\theta}_{\backslash n} - \hat{\theta}\right) \rangle ds$$

Now, subtracting the first two terms on the right hand side from the left, we get can identify the left with Eq. (22). Thus, Eq. (22) is equal to:

$$= \int_0^1 \langle \hat{\theta}_{\backslash n} - \mathrm{NS}_{\backslash n}, \left(\nabla^2 F^{\backslash n}((1-s)\hat{\theta} + s\hat{\theta}_{\backslash n}) - \nabla^2 F^{\backslash n}(\hat{\theta})\right) \left(\hat{\theta}_{\backslash n} - \hat{\theta}\right) \rangle ds.$$

We can upper bound this by taking an absolute value, then applying the triangle inequality and Cauchy-Schwarz to get

$$\leq \left\|\hat{\theta}_{\backslash n} - \mathrm{NS}_{\backslash n}\right\|_2 \left\|\hat{\theta}_{\backslash n} - \hat{\theta}\right\| \int_0^1 \left\|\left(\nabla^2 F^{\backslash n}((1-s)\hat{\theta} + s\hat{\theta}_{\backslash n}) - \nabla^2 F^{\backslash n}(\hat{\theta})\right)\right\|_{op} ds. \tag{23}$$

Using the fact that, on the line segment $(1-s)\hat{\theta} + s\hat{\theta}_{\backslash n}$, the $\hat{D}_n^{(2)}$ are lipschitz with constant C_n:

$$C_n := \max_{s = in [0,1]} \left|\hat{D}_n^{(3)}\left((1-s)\hat{\theta} + s\hat{\theta}_{\backslash n}\right)\right|,$$

we can upper bound the integrand by:

$$\left\|\left(\nabla^2 F^{\backslash n}((1-s)\hat{\theta} + s\hat{\theta}_{\backslash n}) - \nabla^2 F^{\backslash n}(\hat{\theta})\right)\right\|_{op}$$
$$= \frac{1}{N} \left\|\sum_{m \neq n} \left(\hat{D}_n^{(2)}\left((1-s)\hat{\theta} + s\hat{\theta}_{\backslash n}\right) - \hat{D}_n^{(2)}\left(\hat{\theta}\right)\right) x_m x_m^T\right\|_{op}$$
$$\leq \frac{C_n \left\|\hat{\theta}_{\backslash n} - \hat{\theta}\right\|_2}{N} \sum_{m \neq n} \left\|x_m\right\|_2^2.$$

Putting this into Eq. (23) and using Lemma 3 gives the result Eq. (19) with $L_n = C_n/N \sum_{m \neq n} \|x_m\|_2^2$.

Now Eq. (20) follows from the triangle inequality $\|\mathrm{IJ}_{\setminus n} - \hat{\theta}_{\setminus n}\|_2 \leq \|\mathrm{NS}_{\setminus n} - \hat{\theta}_{\setminus n}\|_2 + \|\mathrm{IJ}_{\setminus n} - NS\|_2$. The bound on $\|\mathrm{IJ}_{\setminus n} - \mathrm{NS}_{\setminus n}\|_2$ follows from Wilson et al. [2020, Lemma 20].

Finally, the continuity of the bounds in Eqs. (19) and (20) follows from Lemma 4. In particular, the $\hat{D}_n^{(1)}$, $\hat{D}_n^{(2)}$, and $\hat{D}_n^{(3)}$ in both bounds are evaluated at $\hat{\theta}_{ALR}$, which is shown to be a continuous function of the ε_n in Lemma 4. The ε_n are, in turn, continuous functions of the lower singular values of the covariate matrix. □

D.3 Proof of Theorem 1

Proof. We first note that the runtime claim is immediate, as Algorithm 2 runs in $O(NDK + K^3)$ time. That the bounds are computable in $O(DK)$ time for each n follows as all derivatives $\hat{D}_n^{(1)}$ and $\hat{D}_n^{(2)}$ need only the inner product of x_n and $\hat{\theta}$, which takes $O(D)$ time. Each norm $\|x_n\|_2$ is computable in $O(D)$. For models for which the optimization problem in Proposition 1 can be quickly solved – such as Poisson or logistic regression – we need only to compute a bound on $\|\hat{\theta}_{\setminus n} - \hat{\theta}\|_2$, which we can do in $O(D)$ via Lemma 3. The only remaining quantity to compute is the η_n, which, by Proposition 2, is computed via a projection onto the orthogonal complement of a K-dimensional subspace. We can compute this projection in $O(DK)$. Thus our overall runtime is $O(DK)$ per datapoint.

To prove Eq. (7), we use the triangle inequality $|x_n^T \widetilde{\mathrm{IJ}}_{\setminus n} - x_n^T \hat{\theta}_{\setminus n}| \leq |x_n^T \mathrm{IJ}_{\setminus n} - x_n^T \hat{\theta}_{\setminus n}| + |x_n^T \mathrm{IJ}_{\setminus n} - x_n^T \widetilde{\mathrm{IJ}}_{\setminus n}|$. We upper bound the first term by using Lemma 2. For the latter, we note that $|x_n^T \mathrm{IJ}_{\setminus n} - x_n^T \widetilde{\mathrm{IJ}}_{\setminus n}| = |\hat{D}_n^{(1)}||Q_n - \tilde{Q}_n|$, which we can bound via the η_n of Proposition 2. The proof for $\mathrm{NS}_{\setminus n}$ is similar. □

D.4 Proof of Corollary 1

Proof. Notice that Ω from Algorithm 2 captures a rank-K subspace of the column span of X. The error bound η_n is the norm of x_n projected outside of this subspace divided by λ. Now, assume that we have $K \geq R$. Then, as the singular values σ_d for $d = R+1, \ldots, D$ go to zero, the norm of any x_n outside this subspace must also go to zero. Thus η_n goes to zero. As E_n is a continuous function of η_n, we also have $E_n \to 0$. □

E Proofs and discussion from Section 5

E.1 Proofs

For convenience, we first restate each claimed result from the main text before giving its proof.

Proposition 2. *Let $\lambda > 0$ and suppose there is some subspace $\mathcal{B}$ on which H and $\widetilde{H}$ exactly agree. Then H^{-1} and $\widetilde{H}^{-1}$ agree exactly on the subspace $\mathcal{A} := H\mathcal{B}$, and for all $n = 1, \ldots, N$:*

$$|x_n^T \widetilde{H}^{-1} x_n - Q_n| \leq \frac{\left\|P_{\mathcal{A}}^{\perp} x_n\right\|_2^2}{\lambda}, \tag{24}$$

where $P_{\mathcal{A}}^{\perp}$ denotes projection onto the orthogonal complement of $\mathcal{A}$.

Proof. First, if H and $\widetilde{H}$ agree on $\mathcal{B}$, then for $\mathcal{A} = H\mathcal{B} = \widetilde{H}\mathcal{B}$, we have $H^{-1}\mathcal{A} = \mathcal{B} = \widetilde{H}^{-1}\mathcal{A}$, as claimed. Then:

$$\begin{aligned}
|Q_n - x_n^T \widetilde{H}^{-1} x_n| &= |x_n^T H^{-1} x_n - x_n^T \widetilde{H}^{-1} x_n| \\
&\leq |(P_{\mathcal{A}}^{\perp} x_n)(H^{-1} - \widetilde{H}^{-1})(P_{\mathcal{A}}^{\perp} x_n)| \\
&\leq \left\|P_{\mathcal{A}}^{\perp} x_n\right\|_2^2 \left\|H^{-1} - \widetilde{H}^{-1}\right\|_{op, \mathcal{A}^{\perp}},
\end{aligned}$$

where $\|\cdot\|_{op,\mathcal{A}^\perp}$ is the operator norm of a matrix restricted to the subspace $\mathcal{A}^\perp$. On this subspace, the action of $\widetilde{H}^{-1}$ is $1/\lambda$ times the identity, whereas all eigenvalues of H^{-1} are all between 0 and $1/\lambda$. Thus:

$$\begin{aligned}\left\|\widetilde{H}^{-1} - H^{-1}\right\|_{op,\mathcal{A}^\perp} &= \max_{v\in\mathcal{A}^\perp,\ \|v\|_2=1}\left[v^T\widetilde{H}^{-1}v - v^T H^{-1}v\right] \\ &= \max_{v\in\mathcal{A}^\perp,\ \|v\|_2=1}\left[\frac{1}{\lambda} - v^T H^{-1}v\right] \leq \frac{1}{\lambda}.\end{aligned}$$

□

We next restate and proof Proposition 4.

Proposition 4. *The $Q_n = x_n^T H^{-1} x_n$ satisfy $0 \leq Q_n \leq \|x_n\|_2^2/(\lambda + \hat{D}_n^{(2)}\|x_n\|_2^2)$. Furthermore, letting $\widetilde{Q}_n := \min\{x_n^T\widetilde{H}^{-1}x_n,\ \|x_n\|_2^2/(\lambda + \hat{D}_n^{(2)}\|x_n\|_2^2)\}$, we have the error bound*

$$|\widetilde{Q}_n - Q_n| \leq \min\left\{\frac{\left\|P_{\mathcal{A}}^\perp x_n\right\|_2^2}{\lambda}, \frac{\|x_n\|_2^2}{\lambda + \hat{D}_n^{(2)}\|x_n\|_2^2}\right\}. \tag{25}$$

Proof. Let $b_n := \sqrt{\hat{D}_n^{(2)}}x_n$. Let $\{v_d\}_{d=1}^D$ be the eigenvectors of H with eigenvalues $\{\gamma_d + \lambda\}_{d=1}^D$ with $\gamma_1 \geq \gamma_2 \geq \cdots \geq \gamma_D$. The quantity $b_n^T H^{-1} b_n$ is maximized if b_n is parallel to v_D; in this case, $b_n^T H^{-1} b_n = \|b_n\|_2^2/(\gamma_D + \lambda)$. Now, recall that the γ_d are the eigenvalues of $\sum_n b_n b_n^T$, meaning $\sum_n \langle b_n, v_D\rangle^2 = \gamma_D$. So, if b_n is parallel to v_D, it must be that $\gamma_D \geq \|b_n\|_2^2$. Thus, $b_n^T H^{-1} b_n \leq \|b_n\|_2^2/(\|b_n\|_2^2 + \lambda)$. Dividing by $\hat{D}_n^{(2)}$ gives that $Q_n = x_n^T H^{-1} x_n$ satisfies:

$$0 \leq Q_n \leq \frac{\|x_n\|_2^2}{\lambda + \hat{D}_n^{(2)}\|x_n\|_2^2}.$$

If we estimate Q_n by the minimum of this upper bound and $x_n^T\widetilde{H}^{-1}x_n$, the error bound from Proposition 2 implies the error bound claimed here. □

E.2 Relation of Algorithm 2 to techniques from randomized linear algebra

As noted, our Algorithm 2 bears a resemblance to techniques from the randomized numerical linear algebra literature. Indeed, our inspiration for Algorithm 2 was the work of Tropp et al. [2017]. Tropp et al. [2017] propose a method to find a randomized top-K eigendecomposition of a positive-semidefinite matrix B. Their method follows the basic steps of (1) produce a random orthonormal matrix $\Omega \in \mathbb{R}^{D\times(S+K)}$, where $S \geq 0$ is an *oversampling* parameter to ensure the stability of the estimated eigendecomposition, (2) compute the Nyström approximation of $B_{nys} \approx B$ using Ω, and (3) compute the eigendecomposition of B_{nys} and throw away the lowest S eigenvalues.

Our Algorithm 2 can be seen as using this method of Tropp et al. [2017] to obtain a rank-K decomposition of the matrix $B = (1/N)\sum_n \hat{D}_n^{(2)} x_n x_n^T$ with specific choices of S and Ω. First, we notice that $S = 0$ (i.e., no oversampling) is optimal in our application – the error bound of Proposition 2 decreases as the size of the subspace $\mathcal{A}$ increases. As $S > 0$ only decreases the size of this subspace, we see that our specific application is only hurt by oversampling. Next, while Tropp et al. [2017] recommend completely random matrices Ω for generic applications (e.g., the entries of Ω are i.i.d. $\mathcal{N}(0,1)$), we note that the results of Proposition 3 suggest that we can improve upon this choice. With the optimal choice of $S = 0$, we note that $\mathcal{A} = H\Omega$. In this case, Proposition 3 implies it is optimal to set $\Omega = H^{-1}V_{:K}$, where $V_{:K}$ are the top-K right singular vectors of X. Algorithm 2 provides an approximation to this optimal choice.

We illustrate the various possible choices of Ω, including i.i.d. $\mathcal{N}(0,1)$, in Fig. 6. We generate a synthetic Poisson regression problem with covariates $x_{nd} \overset{i.i.d.}{\sim} \mathcal{N}(0,1)$ and $y_n \sim \text{Poisson}(e^{x_n^T\theta^*})$, where $\theta^* \in \mathbb{R}^D$ is a true parameter with i.i.d. $\mathcal{N}(0,1)$ entries. We generate a dataset of size $N = 200$. The covariates have dimension $D = 150$ but are of rank 50. We compute $\widetilde{I}J$ for various settings of K and Ω, as shown in Fig. 6. As suggested by the above discussion, we use no oversampling

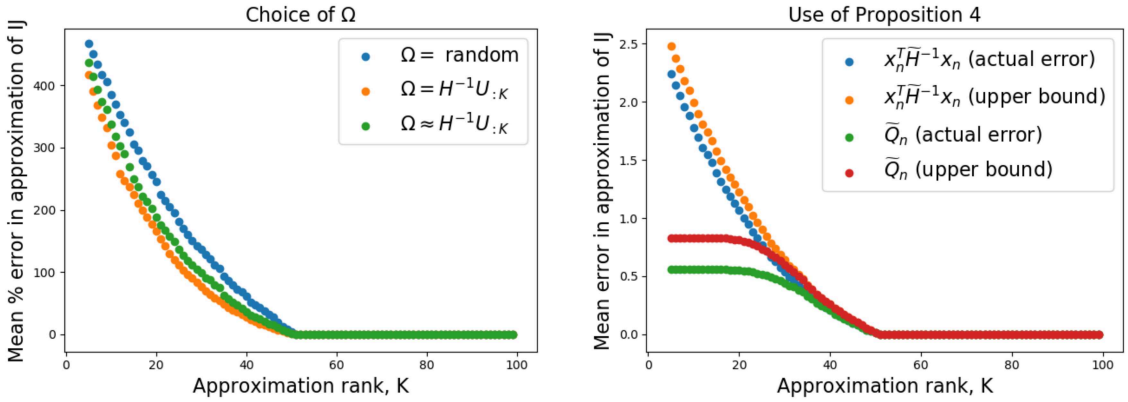

Figure 6: Quality of approximation of $\text{IJ}_{\setminus n}$ on a synthetic Poisson regression problem using the methods from Section 5. (*Left*): We show three options for the choice of the matrix Ω. Blue shows the choice of Ω having orthonormal columns selected uniformly at random, orange the optimal choice of Ω from Proposition 3, and green our approximation to this optimal choice. Percent error $|\text{IJ}_{\setminus n} - \widetilde{\text{IJ}_{\setminus n}}|/|\text{IJ}_{\setminus n}|$ is reported to give a sense of scale. (*Right*): Importance of Proposition 4 for approximating Q_n. We show two approximations along with our upper bounds on their error: (1) $Q_n \approx x_n^T\widetilde{H}^{-1}x_n$ and (2) our recommended $Q_n \approx \widetilde{Q}_n$ from Proposition 4. We report absolute error $|\text{IJ}_{\setminus n} - \widetilde{\text{IJ}_{\setminus n}}|$ so that both actual and estimated error can be plotted.

(i.e., $S = 0$). On the left, we see that using a diagonal approximation to H^{-1} and a single subspace iteration gives a good approximation to the optimal setting of Ω. On the right, we see the improvement made by use of the upper bound on $x_n^T H^{-1} x_n$ from Proposition 4.

E.3 Implementation of Algorithm 2

As noted by Tropp et al. [2017], finding the decomposition of B in Algorithm 2 as-written can result in numerical issues. Instead, Tropp et al. [2017] present a numerically stable version which we use in our experiments. For completeness, we state this implementation here, which relies on computing the Nyström approximation of the shifted matrix $B_\nu = B + \nu I_D$, for some small $\nu > 0$:

1. Construct the shifted matrix sketch $G_\nu := B\Omega + \nu\Omega$.
2. Form $C = \Omega^T G_\nu$.
3. Compute the Cholesky decomposition $C = \Gamma\Gamma^T$.
4. Compute $E = G_\nu \Gamma^{-1}$.
5. Compute the SVD $E = U\Sigma V^T$.
6. Return U and $\Sigma^2 - \nu I$ as the approximate eigenvectors and eigenvalues of B.

F Error bound experiments

Here, we provide more details on our investigation of the error bounds of Theorem 1 from Fig. 4. In Section 6, we showed that, over five randomly generated synthetic datasets, our error bound on $x_n^T\widetilde{\text{IJ}}_{\setminus n}$ implies upper bounds on out-of-sample error that are reasonably tight. However, we noted that these bounds can occasionally be vacuously loose. On the left of Appendix F, we show this is the case by repeating the experiment in Fig. 4 for an additional fifteen trials. While most trials have similar behavior to the first five, trial 16 finds an upper bound of the out-of-sample error that is too loose by two orders of magnitude. However, we note that this behavior is mostly due to two offending points n. Indeed, on the right of Appendix F, we show the same results having replaced the two largest bound values with those from exact CV.

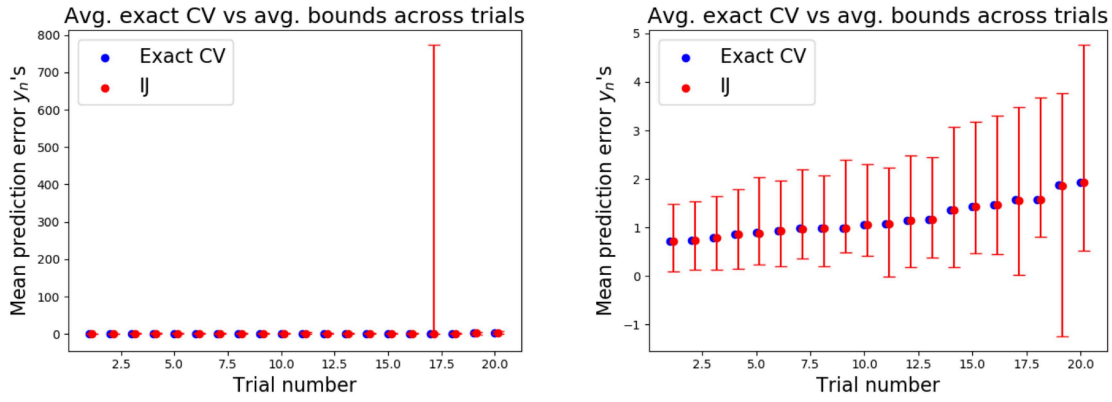

Figure 7: Experiments from Appendix F. (*Left*): Error bounds can be vacuously large; for trial number 16, our bound exceeds exact CV by two orders of magnitude. (*Right*): By computing our bound for all n and re-running exact CV for the two largest values, we obtain estimates that are much closer to exact CV.

G Real data experiments

Here we provide more details about the three real datasets used in Section 6.

1. The `p53` dataset is from Danziger et al. [2009, 2007, 2006]. The full dataset contains $D = 5{,}408$ features describing attributes of mutated p53 proteins. The task is to classify each protein as either "transcriptionally competent" or inactive. To keep the dimension high relative to the number of observations N, we subsampled $N = 8{,}000$ datapoints uniformly at random for our experiments here. We fix $K = 500$ to compute $\widetilde{Q}_n$ for both $x_n^T \widetilde{\mathrm{IJ}}_{\setminus n}$ and $x_n^T \widetilde{\mathrm{NS}}_{\setminus n}$.
2. The `rcv1` dataset is from Lewis et al. [2004]. The full dataset is of size $N = 20{,}242$ and $D = 47{,}236$. Each datapoint corresponds to a Reuters news article given one of four labels according to its subject: "Corporate/Industrial," "Economics," "Government/Social," and "Markets." We use a pre-processed binarized version from `https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/binary.html`, which combines the first two categories into a "positive" label and the latter two into a "negative" label. We found running CV on the full dataset to be too computationally intensive, and instead formed a smaller dataset of size $N = D = 20{,}000$. The data matrix is highly sparse, so we chose our $20{,}000$ dimensions by selecting the most common (i.e., least sparse) features. We then chose $N = 20{,}000$ datapoints by subsampling uniformly at random. We fix $K = 1{,}000$ in this experiment.
3. The `blog` dataset is from Buza [2014]. The base dataset contains $D = 280$ features about $N = 52{,}397$ blogs. Each feature represents a statistic about web traffic to the given blog over a 72 hour period. The task is to predict the number of unique visitors to the blog in the subsequent 24 hour period. We first generate a larger dataset by considering all possible pairwise features $x_{nd_1} x_{nd_2}$ for $d_1, d_2 \in \{1, \ldots, D\}$. The resulting problem has too high N and D to run exact CV on in a reasonable amount of time, so we again subsample to $N = 20{,}000$ and $D = 20{,}280$. We again choose the $20{,}000$ least sparse parwise features and then add in the original 280 features. Finally, we choose our $20{,}000$ datapoints uniformly at random.

H Sensitivity of results to λ

In our main real-data experiments in Section 6, we chose a moderate value of $\lambda = 5.0$ to speed up the convergence of exact CV. Here, we investigate how sensitive our results are to the choice of this parameter. We randomly select $N = 600, D = 400$ subsets of the `p53` and `blog` datasets. We exactly compute Q_n, as well as compute our approximation $\widetilde{Q}_n$ from Algorithm 2 for $K \in$

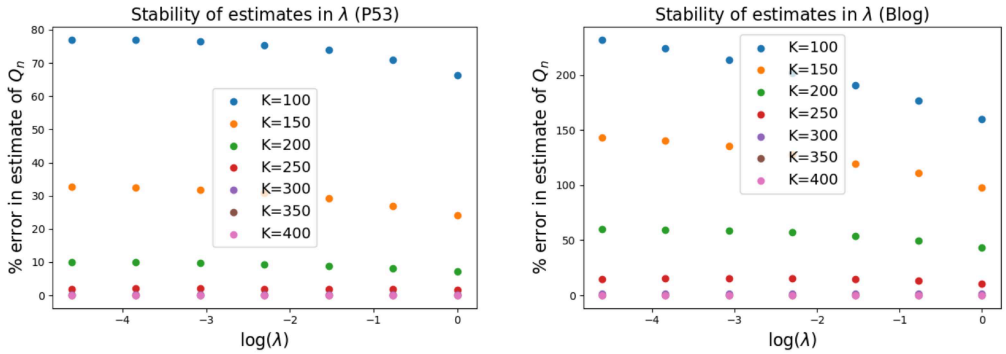

Figure 8: Experiment from Appendix H. We report average error in the estimate of Q_n, $(1/N)\sum_n |Q_n - \widetilde{Q}_n|/|Q_n|$ for both the `p53` and `blog` datasets. We note that the errors when using $K = 300, 350, 400$ are visually indistinguishable from one another.

$\{100, 150, 200, 250, 300, 350, 400\}$. In Fig. 8, we see that the choice of λ has only a mild effect on the results.