Sparse Feature Selection by Information Theory

Guangyao Zhou, Stuart Geman
Division of Applied Mathematics, Brown University
Providence, RI, USA
guangyao_zhou@brown.edu, stuart_geman@brown.edu

Joachim M. Buhmann
Department of Computer Science
ETH Zurich, Switzerland
jbuhmann@inf.ethz.ch

***Abstract*—Learning sparse structures in high dimensions defines a combinatorial selection problem of e.g. informative feature dimensions and a subsequent estimation task to determine adequate model parameters. The arguably simplest sparse inference problem requires estimating a sparse mean in high dimensions when most dimensions should be discarded [6]. We reduce sparse mean estimation to a pure selection problem by restricting the source to binary values that are contaminated with various noise models. The model selection principle of *Approximation Set Coding* [2], [3] is rigorously applied, and generalization capacity is used to evaluate different algorithms. Simulation results demonstrate the effectiveness of generalization capacity compared to traditional model selection approaches. Sampling-based approximation yields insights into the behavior of algorithms in high dimensions at different noise levels.**

I. Introduction

Estimating sparse structures or patterns in very high dimensional data requires selecting informative dimensions and estimating corresponding model parameters. Sparse linear regression [1], sparse mean estimation [6] and compressive sensing [5] belong to this category of statistical problems with a distinct combinatorial flavor.

Arguably, the simplest case of sparse estimation is defined by the sparse centroid problem where we estimate a location parameter $\mu = (\mu_1, \ldots, \mu_d) \in \mathbb{R}^d$ based on n samples $\{X_1, \ldots, X_n\} \in \mathcal{X} \subset (\mathbb{R}^d)^n$. This statistical task is well posed by the assumption that most dimensions j are irrelevant for the estimation and the respective components vanish ($\mu_j = 0$). The informative dimensions with "signal" in the data are encoded by $\mu_j \neq 0$.

In this paper, we will study the sparse centroid problem under a simplified set-up:

Data Generating Mechanism: The data are assumed to be generated by a sparse source that has binary coordinates with exactly p ones, i.e., $\forall 1 \leq j \leq d$, $\mu_j^0 \in \{0,1\} =: \mathbb{B}$, $|\mu^0|_1 = p$

$$X_j = \mu_j^0 + \epsilon_j(\sigma)\,. \tag{1}$$

$\epsilon(\sigma)$ is the noise part, whose noise level is controlled by $\sigma \geq 0$. The simplification $\mu^0 \in \mathbb{B}^d$ restricts sparse centroid estimation to a support recovery problem which isolates the combinatorial selection part of the inference from the parameter estimation part.

Hypothesis Space: The solution space or *hypothesis space* $\mathcal{C}$ of this inference problem is assumed to be a subset of the d-dimensional Boolean cube $\mathcal{C} \subset \mathbb{B}^d$ with the sparsity constraint $\forall \mu \in \mathcal{C}, |\mu|_1 = p$.

Goal and Algorithms: Our goal here is to recover the true sparse centroid from the noisy data. Algorithms to solve this inference problem can often be characterized by cost minimization where the cost $R(\mu, X)$ measures the quality of a particular hypothesis $\mu \in \mathcal{C}$ for a given data set X.

If we choose different cost functions, we get different algorithms, and a natural question to ask is: how to decide among these algorithms? There are some established approaches in classical statistics, e.g., the generalization error. In this paper, we advocate an information-theoretic approach called *Approximation Set Coding* [2], [3], in which we use a key quantity called "generalization capacity" (denoted as GC in the rest of the paper) to evaluate different algorithms.

The main contributions of the paper are twofold: (1) It gives the first simple enough yet interesting example to which *Approximation Set Coding* can be rigorously applied.[1] (2) Instead of conditioning on two specific data sets in deriving GC, as we did in the previous works on *Approximation Set Coding*, we average over all data set pairs by taking expectation w.r.t. the joint distribution of independent data set pairs in order to make GC more robust. This new approach gives rise to a quenched averaging problem, whose difficulty has long been recognized, and a sampling-based approximation method provides us with some interesting insights into the behavior of GC.

The rest of the paper is organized as follows: in Section II, we motivate the main idea of *Approximation Set Coding*, and rigorously derive GC under the above set-up. In Section III, we present some simulation results to illustrate how we can use GC to evaluate different algorithms, and also compare its performance with other model selection approaches. We present the sampling-based approximation of GC for the simple Gaussian noise model and a quadratic loss function in Section IV, before concluding in Section V.

II. Information Content of Algorithms

An algorithm $\mathcal{A} : \mathcal{X} \to \mathcal{C}$ can be seen as a mapping from the data space $\mathcal{X}$ to the hypothesis space $\mathcal{C}$. $\mathcal{A}$ accepts a data set X as input and outputs what $\mathcal{A}$ considers the optimal hypothesis. For example, the inference principle of *empirical risk minimization* suggests selecting the hypothesis with minimal

[1] There might be some subtleties for more complicated problems (e.g. the assumption of existence of a corresponding transformation on the data space for every transformation on the hypothesis space, which is discussed in Section II), but the derivations of the error bound are mathematically rigorous for the problem considered in this paper.

cost on the data set X, i.e., $\mu^* \in \arg\min_{\mu\in\mathcal{C}} R(\mu, X)$. Since data X is random, μ^* also fluctuates and might be unstable.

Intuitively, fluctuations in the input X induce fluctuations in the output of $\mathcal{A}$, implying that we infer different hypotheses for different data sets, but all of them should be considered reasonable hypotheses returned by $\mathcal{A}$. In some sense, all these hypotheses are statistically indistinguishable when employing $\mathcal{A}$ for inference. The size of this generalization set of statistically indistinguishable hypotheses depends on properties of $\mathcal{A}$, and it is an indicator of its "local" robustness, i.e., $\mathcal{A}$'s ability to generalize for such input data X. A small size implies that the output of $\mathcal{A}$ is fairly stable, $\mathcal{A}$ can tolerate a relatively high level of noise in the data, and $\mathcal{A}$ performs better at generalizing than alternative procedures with large generalization sets. This size, from our perspective, is a fundamental property of $\mathcal{A}$, and GC is our way of quantifying it.

Formally, suppose we have two data sets X', X'' which are produced independently by the data generating mechanism (1), the hypothesis space $\mathcal{C}$, and a cost function $R(\mu, X)$. From the above analysis, selecting a single best hypothesis might be unwarranted. As an alternative inference strategy, we suggest determining a weight function $w(\mu, X) \in [0, 1]$ which encodes the plausibility of a hypothesis given the data. The normalized version of these weights can be interpreted as a posterior $p(\mu|X) = \frac{w(\mu,X)}{\sum_{\mu\in\mathcal{C}} w(\mu,X)}$. Given a cost function $R(\mu, X)$, a natural choice for the weight function is the Boltzmann weight $w(\mu, X) := \exp(-\beta R(\mu, X))$[2]. With this terminology, we can define the partition function $Z_\beta(X) = \sum_{\mu\in\mathcal{C}} w_\beta(\mu, X)$ and $\Delta Z_\beta(X', X'') = \sum_{\mu\in\mathcal{C}} w_\beta(\mu, X')w_\beta(\mu, X'')$. Our GC is defined to be

$$I = \max_{\beta\in[0,+\infty)} \mathbb{E} I_\beta = \max_{\beta\in[0,+\infty)} \mathbb{E} \log \frac{|\mathcal{C}|\Delta Z_\beta(X', X'')}{Z_\beta(X')Z_\beta(X'')} \quad (2)$$

The expectation is taken w.r.t. (X', X'').

The idea of using a Gibbs measure over $\mathcal{C}$, with the energy related to the cost of each hypothesis, is a central concept of the PAC-Bayesian approach [4], [7]. However, PAC-Bayesian Analysis bounds the true generalization error with its empirical estimate and a penalty term, while we just approximate the expected generalization capacity (2) by its empirical counterpart.

To see how this formula (2) is derived and why it is a good measure of the generalization ability of an algorithm, we now set up the underlying communication problem. First, we define the set of allowed transformations $\mathbb{T}$, which is the analog of the set of possible codebook vectors in Shannon's information theory. Here, by transformation, we mean a bijective mapping $\tau : \mathcal{C} \to \mathcal{C}$ from $\mathcal{C}$ to itself. Since we want the transformation to move the generalization set around in $\mathcal{C}$, we discard all transformations that map a hypothesis to itself, i.e., $\mathbb{T} =: \{\tau : \mathcal{C} \to \mathcal{C} : \tau \text{ is bijective}, \forall\mu \in \mathcal{C}, \tau \circ \mu \neq \mu\}$. We introduce an additional assumption that $\forall\tau \in \mathbb{T}, \exists$ a bijective mapping τ^D from the set of data sets to itself, s.t. $w(\tau \circ \mu, \tau^D \circ X) = w(\mu, X), \forall\mu \in \mathcal{C}$ and data set X. This assumption is essential for the following error analysis of the communication problem, and it holds in this specific example. In fact, we can transform any of the hypotheses to any other hypothesis by permuting the dimensions, and a corresponding transformation (permuting dimensions in the same way on the data space) obviously exists and satisfies the requirement.

The communication problem is stated as follows: given a number M, we uniformly sample M transformations and generate a codebook $\mathfrak{C} = \{\tau_1, \cdots, \tau_M\} \subset \mathbb{T}$. Both the sender and the receiver know the codebook, the data set X', the cost function $R(\mu, X)$, and the parameter β. The sender sends a transformation τ_s. The receiver accepts a data set $\tilde{X} = \tau_s^D \circ X''$, and tries to estimate the sent transformation. The challenge for the receiver is to distinguish between the random fluctuations in X'' in its comparison to X' and the unknown transformation τ_s. For this, the receiver has to exploit the knowledge of X'. We suggest the decoding rule

$$\begin{aligned}\hat{\tau} \;&\in\; \arg\max_{\tau\in\mathfrak{C}} E_{\mu|X'} w(\tau \circ \mu, \tilde{X}) \\ &=\; \arg\max_{\tau\in\mathfrak{C}} \frac{\sum_{\mu\in\mathcal{C}} w(\mu, X')w(\tau \circ \mu, \tilde{X})}{Z_\beta(X')}\end{aligned}$$

The decoding rule selects the transformation with the largest expected weight on the transformed second data set $\tilde{X}$. The probability of a decoding error is

$$\begin{aligned}&P(\hat{\tau} \neq \tau_s|\tau_s) \;= \\ &P\Big(\max_{i\neq s} \sum_{\mu\in\mathcal{C}} w(\mu, X')w(\tau_i \circ \mu, \tilde{X}) > \sum_{\mu\in\mathcal{C}} w(\mu, X')w(\tau_s \circ \mu, \tilde{X})|\tau_s\Big) \\ &\leq \sum_{i\neq s} P\Big(\sum_{\mu\in\mathcal{C}} w(\mu, X')w(\tau_i \circ \mu, \tilde{X}) > \sum_{\mu\in\mathcal{C}} w(\mu, X')w(\mu, X'')|\tau_s\Big) \\ &= (M-1)P\Big(\sum_{\mu\in\mathcal{C}} w(\mu, X')w(\tau_s^{-1} \circ \tau \circ \mu, X'') > \Delta Z_\beta(X', X'')|\tau_s\Big)\end{aligned}$$

where τ is uniformly distributed on $\mathbb{T}$ and is the only source of randomness. By Markov's inequality, we have

$$\begin{aligned}&P(\hat{\tau} \neq \tau_s|\tau_s) \\ &\leq (M-1)\frac{\mathbb{E}_\tau \sum_{\mu\in\mathcal{C}} w(\mu, X')w(\tau_s^{-1} \circ \tau \circ \mu, X'')}{\Delta Z_\beta(X', X'')} \\ &= \frac{M-1}{|\mathbb{T}|\Delta Z_\beta(X', X'')} \sum_{\tau\in\mathbb{T}} \sum_{\mu\in\mathcal{C}} w(\mu, X')w(\tau_s^{-1} \circ \tau \circ \mu, X'') \\ &= \frac{M-1}{|\mathbb{T}|\Delta Z_\beta(X', X'')} \sum_{\mu\in\mathcal{C}} w(\mu, X') \sum_{\tau\in\mathbb{T}} w(\tau \circ \mu, X'')\end{aligned}$$

By the definition of $\mathbb{T}$ and symmetry, if $\forall\mu, \mu' \in \mathcal{C}, \mu \neq \mu'$, we define $a(\mu, \mu') = |\{\tau \in \mathbb{T} : \tau \circ \mu = \mu'\}|$, then $a(\mu, \mu')$ is a constant (denoted by a) and does not change with μ, μ', and $|\mathbb{T}| = (|\mathcal{C}| - 1)a$.

$$\begin{aligned}P(\hat{\tau} \neq \tau_s|\tau_s) &\leq \frac{M-1}{|\mathcal{C}|-1} \frac{\sum_{\mu\in\mathcal{C}} w(\mu, X') \sum_{\mu\in\mathcal{C}} w(\mu, X'')}{\Delta Z_\beta(X', X'')} \\ &\leq \frac{M}{|\mathcal{C}|} \frac{Z_\beta(X')Z_\beta(X'')}{\Delta Z_\beta(X', X'')} = M\exp(-I_\beta)\end{aligned}$$

[2]We assume for simplicity that $\min_\mu R(\mu, X) = 0$. If this minimization is computationally intractable then weights $w(\mu, X) \in [0, +\infty)$ are admissible.

The second inequality holds since $M < |\mathcal{C}|$. If I_β is large, we can reliably communicate more transformations than for small I_β. Since β is a parameter, we can maximize I_β w.r.t β. We define I_β based on two specific data sets. The condition of asymptotic error free communication should hold for all typical pairs (X', X'') [3] and, therefore, we calculate $\mathbb{E}I_\beta$ before maximizing over β. This derivation motivates GC as defined in equation (2) as a reasonable characterization of algorithm robustness, and illustrates how it measures an algorithm's generalization ability.

III. Some Simulations

To illustrate how we can use GC to evaluate algorithms, we carry out some simulation experiments. For a given noise model, we estimate the expectation $\mathbb{E}I_\beta$ by sampling to calculate GC: (i) sample a number of pairs of data sets, (ii) calculate an I_β for each pair and (iii) average these I_β values to estimate $\mathbb{E}I_\beta$. We perform this procedure for a grid of β values, and we calculate the maximum $\mathbb{E}I_\beta$ as the estimate of GC. A $\mu^0 \in \mathcal{C}$ is assumed to be given in the data generating mechanism (1).

Suppose we are provided with a pair of data sets X', X''. To calculate the corresponding I_β, we have to calculate three partition functions $Z_\beta(X')$, $Z_\beta(X'')$ and $\Delta Z_\beta(X', X'')$. If we use d to denote the number of dimensions and k to denote the number of 1's, then calculating each partition function involves a summation of $|\mathcal{C}| = \binom{d}{k}$ terms. This number grows quickly as d and k grows, so to make the numerical calculations tractable, we consider a relatively simple case where $d = 50$ and $k = 4$.

A. Noise Models

To evaluate and compare different algorithms, we consider three different noise models ($1 \le j \le d$): (a) Gaussian Noise Model (GN): $\epsilon_j(\sigma) \overset{i.i.d.}{\sim} N(0, \sigma^2)$, (b) Laplacian Noise Model (LN): $\epsilon_j(\sigma) \overset{i.i.d.}{\sim}$ Laplace$(0, \sigma)$, where Laplace$(0, \sigma)$ is the Laplace distribution with density function $\exp(-\frac{|x|}{\sigma})/(2\sigma),\ x \in \mathbb{R}$, and (c) Gaussian Mixture Noise Model (GMN): $\epsilon_j(\sigma) \overset{i.i.d.}{\sim} pN(0, \sigma^2) + (1-p)N(0, \alpha\sigma^2)$, where $\alpha > 1$ and $N(0, \alpha\sigma^2)$ contaminates the Gaussian noise part. In our experiment, we choose $p = 0.2$ and $\alpha = 100$.

B. Algorithms

The ML estimator for the location parameter is the mean for the Gaussian distribution, and the median for the Laplace distribution. The density function of the Gaussian distribution has a quadratic form in the exponent, while the density function of the Laplace distribution has an absolute value in the exponent. So it is reasonable to expect that an algorithm using the mean as the location parameter and the L_2 loss should perform better for GN, while an algorithm using the median as the location parameter and the L_1 loss should perform better for LN. GMN is just a contaminated version of GN, so we expect an algorithm using the median as the location parameter, which is robust to outliers, and the L_2 loss should outperform an algorithm using the mean and the L_1 loss. Based on these observations, we design four different algorithms, which correspond to four different cost functions:

$$
\begin{aligned}
R_{1\text{bar}}(\mu, X) &= \textstyle\sum_{j=1}^{d} |\mu_j - \bar{X}_j|, \\
R_{1\text{med}}(\mu, X) &= \textstyle\sum_{j=1}^{d} |\mu_j - X_j^{\text{med}}|, \\
R_{2\text{bar}}(\mu, X) &= \textstyle\sum_{j=1}^{d} (\mu_j - \bar{X}_j)^2, \\
R_{2\text{med}}(\mu, X) &= \textstyle\sum_{j=1}^{d} (\mu_j - X_j^{\text{med}})^2 ,
\end{aligned}
$$

where $X = \{X^{(1)}, \cdots, X^{(n)}\}$ is the data set, and $\bar{X}_j, X_j^{\text{med}}$ are the mean and median of $\{X_j^{(1)}, \cdots, X_j^{(n)}\}$, respectively.

Since we assume the data are $i.i.d$ generated and we are just using the mean/median in the algorithm, varying the size of the data set n is essentially equivalent to varying the level of noise σ. We compare the algorithms at different noise levels, so we simply set the size of the data set $n = 100$. We use the noise levels $\sigma = 1, 2, \cdots, 10, 20$. At each σ, we sample 100 pairs of data sets and compute I_β for β from 0.1 to 20 with a step size of 0.1. We then get the average of these I_β's to use as $\mathbb{E}I_\beta$ and maximize over these β's to get GC at this σ.

C. Other Model Selection Approaches

To gain insights into how GC actually performs, we compare it to established model selection approaches. The pair of data sets X', X'' suggests using X' as the training set and X'' as the validation set. Empirical risk minimization requires first finding $\mu^* \in \arg\min_{\mu \in \mathcal{C}} R(\mu, X')$ and then measuring the generalization error $R(\mu^*, X'')$ to estimate the generalization ability of the algorithm. We repeat the experiment 100 times at each noise level and calculate the average.

Since we know the true mean in this example, we can also compare the algorithms by calculating the expected overlap between the learned mean and the true mean. This *support recovery* procedure finds $\mu^* \in \arg\min_{\mu \in \mathcal{C}} R(\mu, X')$ first, and then uses the inner product $\mu^* \cdot \mu^0$ to measure the generalization ability. The experiment is repeated 100 times, and the average $\mu^* \cdot \mu^0$ serves as our expected overlap.

D. Experimental Results

The experimental results are shown in Table I with the best results depicted in bold. From the tables, we can see that R_{2bar} consistently performs the best for GN, while R_{2med} generally performs the best for LN and GMN. This matches the results given by the expected overlap, but differs from our expectation for LN in that GC favors the L_2 loss function over the L_1 loss function. This seemingly unexpected result indeed shows the effectiveness of GC in evaluating algorithms, since in this specific example, the expected overlap should serve as the ultimate criterion for evaluating algorithms because it uses the true mean.

But the expected generalization error gives us a different perspective. While R_{2bar} remains the best for GN for small σ, it starts to lose to other cost functions for larger σ. Specifically, according to generalization error, it is no longer true that L_2 loss is better than L_1 loss under GN when σ is large. The same effect happens with R_{2med} under LN: R_{2med} starts to

perform worse than R_{1bar} when σ is large, suggesting that the median performs worse than the mean. Under GMN, R_{2med} loses to R_{1med} when σ is large, which again implies L_1 loss performs better than L_2 loss under a Gaussian-type noise model. These results are inconsistent with the results from expected overlap, which should be a better approach in this example. This difference in the performance of GC and the generalization error further demonstrates the advantage of GC in evaluating algorithms.

IV. Sampling-based Approximation

The simulations in Section III are based on directly computing the partition functions, but this is not practical for large d and k. To analyze the behavior of GC in high dimensions, we develop some sampling-based approximation methods. For illustration, we only consider GN and the cost function

$$R_{2bar}(\mu, X) = \sum_{j=1}^{d}(\mu_j - \bar{X}_j)^2 = \sum_{j=1}^{d}\mu_j(1-2\bar{X}_j) + \sum_{j=1}^{d}\bar{X}_j^2 \tag{3}$$

From the set-up, we have $\bar{X}_j \overset{i.i.d.}{\sim} N(\mu_j^0, \frac{\sigma^2}{n})$, so combining equation (2) with (3) gives us

$$\mathbb{E}I_\beta = \log(\binom{d}{k}) + \mathbb{E}_Z \log(\sum_{\mu\in\mathcal{C}} e^{4\beta(\mu\cdot\mu^0)+\frac{2\sqrt{2}\beta\sigma}{\sqrt{n}}(\mu\cdot Z)}) - 2\mathbb{E}_Z \log(\sum_{\mu\in\mathcal{C}} e^{2\beta(\mu\cdot\mu^0)+\frac{2\beta\sigma}{\sqrt{n}}(\mu\cdot Z)}) =: f(\beta) \tag{4}$$

where $Z = (Z_1, \cdots, Z_d)$ and $Z_1, \cdots, Z_d \overset{i.i.d.}{\sim} N(0,1)$.

A. Generalization Capacity Approximation

To approximate GC, we need to be able to evaluate $f(\beta)$ for any β. It is not hard to see, from equation (4), that all we need is the ability to evaluate an expectation of the form $\mathbb{E}(a,b) := \mathbb{E}_Z \log(\sum_{\mu\in\mathcal{C}} e^{a(\mu\cdot\mu^0)+b(\mu\cdot Z)})$, which can be written as $\mathbb{E}_Z \log(|\mathcal{C}|\mathbb{E}_{u(\mu)} e^{a(\mu\cdot\mu^0)+b(\mu\cdot Z)})$, where $u(\mu) = \frac{1}{|\mathcal{C}|}, \forall \mu \in \mathcal{C}$ and a, b are some constants. The latter expression gives us a sampling-based method for approximating $\mathbb{E}(a,b)$: we first sample a large set of Z's, and for each Z, we sample μ's uniformly from $\mathcal{C}$. We then average over the samples, and multiply the result by $|\mathcal{C}|$ to get an approximation of $\mathbb{E}(a,b)$. For a given β, we can repeat this procedure twice for the two expectations in $f(\beta)$ to approximately evaluate $f(\beta)$.

With the ability to approximately evaluate $f(\beta)$, there are two ways of getting GC: (1) evaluate $f(\beta)$ for a grid of β values and use the maximum as GC and (2) get the optimal β value β^* first, and then use $f(\beta^*)$ as GC. Getting β^* involves calculating $\frac{df(\beta)}{d\beta}$, which is $\mathbb{E}_Z[\mathbb{E}_{p_1(\mu|Z)}\mu\cdot(4\mu^0 + \frac{2\sqrt{2}\sigma}{\sqrt{n}}Z) - 2\mathbb{E}_{p_2(\mu|Z)}\mu\cdot(2\mu^0 + \frac{2\sigma}{\sqrt{n}}Z)]$ where $p_1(\mu|Z), p_2(\mu|Z)$ are Gibbs distributions with Boltzmann weights $e^{4\beta(\mu\cdot\mu^0)+\frac{2\sqrt{2}\beta\sigma}{\sqrt{n}}(\mu\cdot Z)}, e^{2\beta(\mu\cdot\mu^0)+\frac{2\beta\sigma}{\sqrt{n}}(\mu\cdot Z)}$ *resp.*. This equation gives us a sampling-based method of evaluating $\frac{df(\beta)}{d\beta}$: we first sample a large set of Z's, and for each Z, we sample μ's from $p_1(\mu|Z), p_2(\mu|Z)$, which are both exponential family distributions, and then approximate $\frac{df(\beta)}{d\beta}$ by averaging over the samples. The simulation results from Section III suggest that the $(\beta, \mathbb{E}I_\beta)$ curve is concave in β, so a reasonable way to get the optimal β is to use gradient ascent.

B. Behavior of Generalization Capacity in High Dimensions

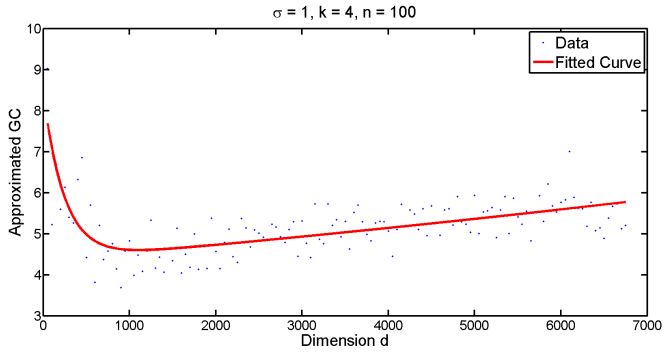

Fig. 1. Approximated Generalization Capacity at Noise Level $\sigma = 1$

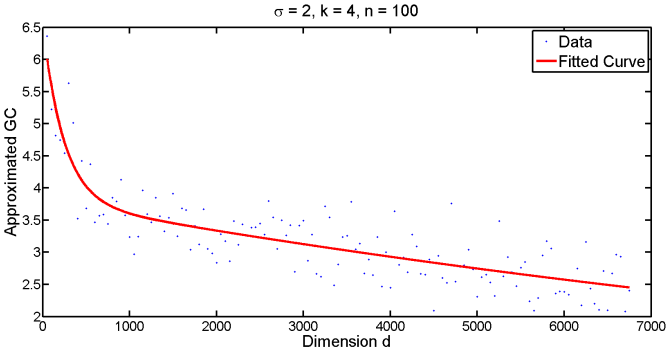

Fig. 2. Approximated Generalization Capacity at Noise Level $\sigma = 2$

In this part, we use the sampling-based methods outlined above to study the behavior of GC in high dimensions. Since it's not very time consuming to evaluate $f(\beta)$, and we haven't proved that the $(\beta, \mathbb{E}I_\beta)$ curve is concave in β, we use the first method to approximate GC. As an illustrative example, we fix $k = 4, n = 100$ and calculate GC at $\sigma = 1, 2$. The results are shown in Figures 1 and 2. Since we are using sampling-based methods, there are fluctuations in the results. So instead of directly connecting different points (the blue dots in the figure), we fit a two term exponential model ($y = ae^{bx} + ce^{dx}$), which is the red curve in the figure.

Intuitively, we expect GC to decrease as d increases, since increasing the number of irrelevant dimensions should make identifying the feature dimensions harder. But as shown in Figure 1, this is not the case: as d increases, GC first drastically decreases, but then slowly increases. This is likely because at a low noise level, when d increases, it is not significantly harder to identify the feature dimensions, but the $\log(\binom{d}{k})$ term in GC increases at a faster rate when d is large. At a moderate noise level (e.g. $\sigma = 2$), GC decreases, first at a very fast rate, then at a much slower rate, as d increases, but remains relatively large in high dimensions. At higher noise levels (e.g. $\sigma = 5, 10$),

TABLE I

Generalization Capacity

σ	Gaussian Noise Model				Laplacian Noise Model				Gaussian Mixture Noise Model			
	1bar	1med	2bar	2med	1bar	1med	2bar	2med	1bar	1med	2bar	2med
1	12.347	12.347	12.347	12.347	12.347	12.347	12.347	12.347	3.082	12.347	3.288	12.347
2	12.226	11.315	12.226	11.323	9.705	11.867	9.744	11.867	0.179	8.538	0.308	**8.611**
3	8.964	5.724	**9.004**	5.910	3.430	7.277	3.965	**7.394**	0.023	3.040	0.065	**3.409**
4	4.742	2.058	**5.229**	2.564	1.281	3.805	1.789	**4.091**	0.002	1.098	0.004	**1.571**
5	2.175	0.778	**2.464**	1.174	0.407	1.713	0.622	**1.941**	0.000	0.307	0.000	**0.446**
6	1.169	0.326	**1.509**	0.555	0.229	1.193	0.407	**1.413**	0.002	0.118	0.000	**0.189**
7	0.524	0.217	**0.751**	0.338	0.124	0.407	0.246	**0.503**	0.000	0.100	0.000	**0.227**
8	0.235	0.126	**0.355**	0.148	0.048	0.132	0.164	**0.255**	0.000	0.017	0.000	**0.045**
9	0.204	0.197	**0.364**	0.180	0.008	0.063	0.014	**0.134**	0.000	0.052	0.000	**0.072**
10	0.043	0.029	**0.109**	0.067	0.016	0.033	0.062	**0.153**	0.000	0.002	0.000	**0.013**
20	0.010	0.000	**0.020**	0.001	0.000	0.013	0.010	**0.016**	0.000	0.003	0.000	0.000

Expected Generalization Error (standard deviations in parentheses)

1	4(0.4)	5(0.5)	**0(0.1)**	1(0.2)	6(0.6)	4(0.5)	1(0.2)	**1(0.1)**	20(2.2)	6(0.6)	13(2.8)	**1(0.2)**
2	8(0.8)	10(1.2)	**2(0.5)**	3(1.0)	12(1.5)	8(1.2)	5(1.5)	**2(0.8)**	39(4.4)	13(1.5)	48(9.7)	**6(1.7)**
3	13(1.6)	16(2.1)	**5(1.6)**	9(2.5)	19(2.2)	14(1.7)	12(2.9)	**6(1.8)**	56(6.2)	20(2.4)	98(20.6)	**13(3.2)**
4	18(2.2)	22(2.9)	**11(2.8)**	16(4.3)	25(2.6)	18(2.2)	19(4.0)	**11(2.7)**	74(7.6)	26(2.8)	170(34.0)	**23(4.7)**
5	22(2.2)	27(2.7)	**15(3.4)**	23(4.6)	31(3.1)	23(2.6)	30(5.3)	**18(4.0)**	90(10.0)	**33(3.2)**	262(56.7)	34(6.9)
6	26(3.0)	32(3.0)	**22(4.6)**	32(5.7)	37(3.7)	28(3.2)	41(8.6)	**25(5.6)**	110(12)	**39(4.3)**	388(83.0)	49(9.7)
7	32(3.5)	38(4.5)	32(6.7)	47(9.8)	42(4.5)	**32(3.5)**	56(11.8)	34(7.6)	129(14)	**46(4.7)**	523(103.9)	64(12.1)
8	**34(3.7)**	42(4.5)	37(7.3)	55(12.1)	47(4.8)	**37(4.1)**	68(13.3)	43(10.2)	143(16)	**51(4.6)**	654(138.2)	79(13.3)
9	**39(3.8)**	47(4.8)	46(8.6)	69(14.1)	52(6.0)	**40(4.6)**	85(18.0)	51(11.5)	164(19)	**56(5.7)**	844(187.4)	98(18.2)
10	**42(4.3)**	51(5.3)	55(10.5)	82(16.4)	60(6.6)	**45(4.4)**	112(24)	65(13.3)	186(20)	**63(6.7)**	1078(218.8)	123(24)
20	**81(8.2)**	100(10)	202(43)	314(61)	114(11)	**86(9)**	405(74)	241(48)	361(39)	**122(12)**	4132(860.5)	476(90)

Expected Overlap (standard deviations in parentheses)

1	4.00(0.0)	4.00(0.0)	4.00(0.0)	4.00(0.0)	4.00(0.0)	4.00(0.0)	4.00(0.0)	4.00(0.0)	2.50(0.8)	4.00(0.0)	2.49(0.8)	4.00(0.0)
2	3.98(0.1)	3.82(0.4)	3.98(0.1)	3.82(0.4)	3.65(0.5)	3.91(0.3)	3.65(0.5)	3.91(0.3)	1.06(1.0)	3.43(0.6)	1.32(0.8)	3.43(0.6)
3	3.48(0.6)	2.88(0.7)	3.48(0.6)	2.89(0.7)	2.53(0.8)	3.27(0.6)	2.56(0.8)	**3.29(0.6)**	0.35(0.6)	2.45(0.8)	0.77(0.8)	**2.49(0.8)**
4	2.92(0.7)	2.36(0.9)	2.91(0.7)	2.32(0.9)	1.97(0.9)	2.82(0.7)	2.13(0.9)	**2.84(0.7)**	0.26(0.5)	1.88(0.9)	0.60(0.7)	**1.96(0.9)**
5	2.30(0.8)	1.86(0.8)	**2.38(0.8)**	2.02(0.8)	1.36(0.8)	2.19(0.8)	1.71(0.8)	**2.26(0.8)**	0.20(0.4)	1.35(1.0)	0.50(0.6)	**1.49(0.8)**
6	1.69(0.9)	1.19(0.8)	**1.81(0.8)**	1.50(0.8)	0.84(0.7)	1.58(0.8)	1.49(0.7)	**1.81(0.8)**	0.11(0.3)	0.95(0.8)	0.53(0.7)	**1.31(0.8)**
7	1.36(0.8)	1.09(0.9)	**1.52(0.9)**	1.32(0.9)	0.60(0.6)	1.06(0.8)	1.20(0.8)	**1.46(0.8)**	0.14(0.4)	0.55(0.7)	0.55(0.7)	**1.21(0.8)**
8	1.16(0.8)	0.73(0.8)	**1.29(0.9)**	1.15(0.8)	0.54(0.6)	0.79(0.7)	1.00(0.8)	**1.20(0.8)**	0.07(0.3)	0.34(0.6)	0.43(0.6)	**1.06(0.8)**
9	0.92(0.7)	0.52(0.6)	**1.22(0.8)**	1.01(0.7)	0.43(0.5)	0.77(0.7)	0.88(0.7)	**1.11(0.8)**	0.06(0.2)	0.52(0.7)	0.49(0.6)	**0.93(0.7)**
10	0.71(0.8)	0.39(0.6)	**1.07(0.8)**	0.85(0.8)	0.53(0.5)	0.65(0.6)	0.89(0.7)	**1.09(0.8)**	0.07(0.3)	0.33(0.5)	0.41(0.6)	**0.81(0.7)**
20	0.13(0.4)	0.17(0.4)	**0.71(0.7)**	0.65(0.7)	0.27(0.4)	0.38(0.5)	0.53(0.6)	**0.65(0.7)**	0.02(0.1)	0.13(0.4)	0.37(0.5)	**0.50(0.6)**

GC shows similar behavior, but the initial decreasing rate is higher, and it approaches zero in high dimensions.

V. Conclusion

This paper discusses the combinatorial aspect of feature selection when computing the sparse centroid of high-dimensional data. By restricting the feature values of the data source to be binary, we separate the estimation problem from the combinatorial search problem. Contrary to traditional model selection principles, *Approximation Set Coding* identifies a set of hypotheses that are equally plausible given the uncertain input data, and uses GC to evaluate the robustness of different algorithms. In the simplified version of the sparse centroid problem, GC shows us the superiority of L_2 loss and mean in the Gaussian noise case, and of L_2 loss and median in the Laplacian and Gaussian mixture noise case, which are verified by the expected overlap criterion. The behavior of GC in high dimensions shows R_{2bar} works much better in low dimensions than in high dimensions, and its ability to work, to some extent, in high dimensions at a low noise level. These insights from GC, some of which match our expectation while others don't, help us better understand the sparse centroid problem and demonstrate the value of *Approximation Set Coding* in understanding algorithms.

Acknowledgment: J.M. Buhmann would like to thank the Division of Applied Mathematics of Brown University for its hospitality during his sabbatical in Fall, 2013. His work was partially funded by SNF Grant # 200021_138117. The work of S. Geman and G. Zhou was partially supported by ONR under contract N00014101933 and DARPA under contract FA8650-11-1-7151.

References

[1] Peter Bühlmann and Sara van de Geer. *Statistics for High-Dimensional Data Methods, Theory and Applications*. Springer Series in Statistics. Springer Verlag, Heidelberg, Berlin, New York, 2011.

[2] Joachim M. Buhmann. Information theoretic model validation for clustering. In *International Symposium on Information Theory, Austin Texas*. IEEE, 2010. (http://arxiv.org/abs/1006.0375).

[3] Joachim M. Buhmann. SIMBAD: emergence of pattern similarity. In *Similarity-Based Pattern Analysis and Recognition*, Advances in Vision and Pattern Recognition, pages 45–64. Springer Berlin, 2013.

[4] O. Catoni. Pac-Bayesian Supervised Classification: The Thermodynamics of Statistical Learning. *ArXiv e-prints*, December 2007.

[5] David L. Donoho. Compressed sensing. *IEEE Tr. on Information Theory*, 52(4):1289, 2006.

[6] Iain M. Johnstone. On minimax estimation of a sparse normal mean vector. *Ann. Statist.*, 22(1):271–289, 1994.

[7] David A. McAllester. Some pac-bayesian theorems. *Machine Learning*, 37(3):355–363, 1999.