The *xyz* algorithm for fast interaction search in high-dimensional data

Gian-Andrea Thanei[1], Nicolai Meinshausen[1], and Rajen D. Shah[*2]

[1]ETH Zürich
[2]University of Cambridge

October 18, 2016

Abstract

When performing regression on a dataset with p variables, it is often of interest to go beyond using main linear effects and include interactions as products between individual variables. For small-scale problems, these interactions can be computed explicitly but this leads to a computational complexity of at least $\mathcal{O}(p^2)$ if done naively. This cost can be prohibitive if p is very large.

We introduce a new randomised algorithm that is able to discover interactions with high probability and under mild conditions has a runtime that is subquadratic in p. We show that strong interactions can be discovered in almost linear time, whilst finding weaker interactions requires $\mathcal{O}(p^\alpha)$ operations for $1 < \alpha < 2$ depending on their strength. The underlying idea is to transform interaction search into a closest-pair problem which can be solved efficiently in subquadratic time. The algorithm is called *xyz* and is implemented in the language `R`. We demonstrate its efficiency for application to genome-wide association studies, where more than 10^{11} interactions can be screened in under 280 seconds with a single-core 1.2 GHz CPU.

1 Introduction

Given a response vector $\mathbf{Y} \in \mathbb{R}^n$ and matrix of associated predictors $\mathbf{X} = (\mathbf{X}_1, \dots, \mathbf{X}_p) \in \mathbb{R}^{n \times p}$, finding interactions is often of great interest as they may reveal important relationships and improve predictive power. When the number of variables p is large, fitting a model involving interactions can involve serious computational challenges. The simplest form of

[*]Supported by the Isaac Newton Trust Early Career Support Scheme.

interaction search consists of screening for pairs (j, k) with high inner product between the outcome of interest $\mathbf{Y}$ and the point-wise product $\mathbf{X}_j \circ \mathbf{X}_k$:

$$\text{Keep all pairs } (j, k) \text{ for which } \mathbf{Y}^T(\mathbf{X}_j \circ \mathbf{X}_k)/n > \gamma. \tag{1}$$

This search is of complexity $\mathcal{O}(np^2)$ in a naive implementation and quickly becomes infeasible for large p. Of course one would typically be interested in maximising (absolute values of) correlations rather than dot products in (1), an optimisation problem that would be at least as computationally intensive.

Even more challenging is the task of fitting a linear regression model involving pairwise interactions:

$$Y_i = \mu + \sum_{j=1}^{p} X_{ij}\beta_j + \sum_{k=1}^{p}\sum_{k=1}^{j-1} X_{ij}X_{ik}\theta_{jk} + \varepsilon_i. \tag{2}$$

Here $\mu \in \mathbb{R}$ is the intercept and β_j and θ_{jk} contain coefficients for main effects and interactions respectively, and ε_i is random noise.

In this paper, we make several contributions to the problem of searching for interactions in high-dimensional settings.

(a) We first establish a form of equivalence between (1) and closest-pair problems [Shamos and Hoey, 1975, Agarwal et al., 1991]. Assume for now that all predictors and outcomes are binary: $X_{ij}, Y_i \in \{-1, 1\}$ (we will later relax this assumption) and define $\mathbf{Z} \in \{-1, 1\}^{n \times p}$ as $X_{ij} = Y_i X_{ij}$. Then it is straightforward to show that (1) is equivalent to

$$\text{Keep all pairs } (j, k) \text{ for which } \|\mathbf{X}_j - \mathbf{Z}_k\|_2 < \gamma' \tag{3}$$

for some γ'. This connects the search for interactions to literature in computational geometry [Shamos and Hoey, 1975, Agarwal et al., 1991] on problems of finding closest pairs of points.

(b) We introduce the *xyz* algorithm to solve (3) based on randomly projecting each of the columns in $\mathbf{X}$ and $\mathbf{Z}$ to a one-dimensional space. By exploiting the ability to sort the resulting $2p$ points with $\mathcal{O}(p\log(p))$ computational cost, we achieve a runtime that is always subquadratic in p and can even reach a linear complexity $\mathcal{O}(np)$ when γ is much larger than the interaction strengths $|\mathbf{Y}^T(\mathbf{X}_j \circ \mathbf{X}_k)|/n$ of the bulk of the pairs (j, k). We show that our approach can be viewed as an example of locality sensitive hashing [Leskovec et al., 2014] optimised for our specific problem.

(c) We show how any method for solving (1) can be used to fit regression models with interactions (2) by building it into an algorithm for the Lasso [Tibshirani, 1996]. The use of *xyz* thus leads to a procedure for applying the Lasso to all main effects and interactions with computational cost that scales subquadratically in p.

(d) We provide implementations of both the core *xyz* algorithm and its extension to the Lasso in the R package `xyz`, available on CRAN.

Our work here is thus related to "closest pairs of points" algorithms in computational geometry as well as an extensive literature on modelling interactions in statistics, both of which we now review.

1.1 Related work

A common approach to avoid the quadratic cost in p of searching over all pairs of variables (1) is to restrict the search space: one can first seek a small number of important main effects, and then only consider interactions involving these discovered main effects. More specifically, one could fit a main effects Lasso [Tibshirani, 1996] to the data first, add interactions between selected main effects to the matrix of predictors, and then run the Lasso once more on the augmented design matrix in order to produce the final model (see Wu et al. [2010] for example). Tree-based methods such as CART [Breiman et al., 1984] work in a similar fashion by aiming to identify an important main effect and then only considering interactions involving this discovered effect.

However it is quite possible for the signal to be such that main effects corresponding to important interactions are hard to detect. As a concrete example of this phenomenon, consider the setting where $\mathbf{X}$ is generated randomly with all entries independent and having the uniform distribution on $\{-1, 1\}$. Suppose the response is given by $Y_i = X_{i1}X_{i2}$, so there is no noise. Since the distribution $Y_i|X_{ij}$ is the same for all k, main effects regressions would find it challenging to select variables 1 and 2. Note that by reparametrising the model by adding one to each entry of $\mathbf{X}$ for example, we obtain $Y_i = (X_{i1}-1)(X_{i2}-1) = 1-X_{i1}-X_{i2}+X_{i1}X_{i2}$. The model now respects the so-called strong hierarchical principle [Bien et al., 2013] that interactions are only present when their main effects are. The hierarchical principle is useful to impose on any fitted model. However, imposing the principle on the model does not imply that the interactions will easily be found by searching for main effects first. The difficulty of the example problem is due to interaction effects masking main effects: this is a property of the signal $\mathbb{E}(Y_i)$ and of course no reparametrisation can make the main effects any easier to find. Approaches that increase the set of interactions to be considered iteratively, can help to tackle this sort of issue in practice [Bickel et al., 2010, Hao and Zhang, 2014, Friedman, 1991, Shah, 2016]. Or those that randomise the search procedure [Breiman, 2001]. However they cannot eliminate the problem of missing interactions, nor do these approaches offer guarantees of how likely it is that they discover an interaction.

As alluded to earlier, the pure interaction search problem (3) is related to close pairs of points problems, and more specifically the close bichromatic pairs problem in computational geometry [Agarwal et al., 1991]. Most research in this area has focused on algorithms that lead to computationally optimal results in the number of points p whilst considering the dimension n to be constant. This has resulted in algorithms where the scaling of the computational complexity with n is at least of order 2^n [Shamos and Hoey, 1975]. Since for meaningful statistical results one would typically require $n \gg \log(p)$, these approaches would not lead to subquadratic complexity. In the special case where $n = p$ and $Z_{ij}, X_{ij} \in \{-1, 1\}$, (3) may be seen to be equivalent to searching for large magnitude entries in the product of square matrices $\mathbf{X}$ and $\mathbf{Z}^T$. This latter problem is amenable to fast matrix multiplication algorithms, which in theory can deliver a subquadratic complexity of roughly $\mathcal{O}(p^{2.4}) = \mathcal{O}(np^{1.4})$ [Williams, 2012, Davie and Stothers, 2013, Le Gall, 2012]. However the constants hidden in the order notation are typically very large, and practical implementations are unavailable. The Strassen algorithm [Strassen, 1969] is the only fast matrix multiplication algorithm used regularly in practice to the best of our knowledge. With a complexity of roughly $\mathcal{O}(p^{2.8}) = \mathcal{O}(np^{1.8})$, the improvement over a brute force close pairs search is only slight.

The strategy we use is most closely related to locality sensitive hashing (LSH) [Indyk and Motwani, 1998] which encompasses a family of hashing procedures such that similar points are mapped to the same bucket with high probability. A close pair search can then be conducted by searching among pairs mapped to the same bucket. In fact, our proposed algorithm for solving (3) can be thought of as an example of LSH optimised for our particular problem setting.

An approach that bears some similarity with our procedure is *epiq* [Arkin et al., 2014]. This works by projecting the data and then searches through a lower dimensional representation for close pairs. This appears to improve upon a naive brute force empirically but there are no proven guarantees that the runtime improves on the $\mathcal{O}(np^2)$ complexity of a naive search.

The *Random Intersection Trees* algorithm of Shah and Meinshausen [2014] searches for potentially deeper interactions in data with both $\mathbf{X}$ and $\mathbf{Y}$ binary. In certain cases with strong interactions a complexity close to linear in p is achieved; however it is not clear how to generalise the approach to continuous data or embed it within a regression procedure.

The idea of Kong et al. [2016] is to first transform the data by forming $\tilde{\mathbf{Y}} = \mathbf{Y} \circ \mathbf{Y}$ and $\tilde{\mathbf{X}}_j = \mathbf{X}_j \circ \mathbf{X}_j$ for each predictor. Next $\tilde{\mathbf{X}}_j$ and $\tilde{\mathbf{Y}}$ are tested for independence using the distance correlation test. In certain settings, this can reveal important interactions with a computational cost linear in p. However, the powers of these tests depend on the distributions of the transformed variables $\tilde{\mathbf{X}}_j$. For example in the binary case when $\mathbf{X} \in \{-1, 1\}^{n \times p}$, each transformed variable will be a vectors of 1's and the independence tests will be unhelpful. We will see that our proposed approach works particularly well in this setting.

1.2 Organisation of the paper

In Section 2 we consider the case where both the response $\mathbf{Y}$ and the predictors $\mathbf{X}$ are binary. We first demonstrate how (2) may be converted to a form of closest pair of points problem. We then introduce a general version of the *xyz* algorithm which solves this based on random projections. As we show in Section 2.1 there is a particular random projection distribution that is optimal for our purposes. This leads to our final version of the *xyz* algorithm which we present in Section 2.3 along with an analysis of its runtime and probabilistic guarantees that it recovers important interactions. In Section 3 we extend the *xyz* algorithm to continuous data. These ideas are then used in Section 4 to demonstrate how the *xyz* algorithm can be embedded within common algorithms [Friedman et al., 2010] for high dimensional regression allowing high-dimensional regression models with interactions to be fitted with subquadratic complexity in p. Section 5 contains a variety of numerical experiments on real and simulated data that complements our theoretical results and demonstrate the effectiveness of our proposal in practice. We conclude with a brief discussion in Section 6 and all proofs are collected in the Appendix.

2 The *xyz* algorithm for binary data

In this section, we present a version of the *xyz* algorithm applicable in the special case where both $\mathbf{X}$ and $\mathbf{Y}$ are binary, so $X_{ij} \in \{-1, 1\}$ and $Y_i \in \{-1, 1\}$. We build up to the algorithm in stages, giving the final version in Section 2.2.

Define $\mathbf{Z} \in \{-1, 1\}^{n \times p}$ by $Z_{ij} = Y_i X_{ij}$ and

$$\gamma_{jk} = \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}_{\{Y_i = X_{ij} X_{ik}\}}. \tag{4}$$

We call γ_{jk} the interaction strength of the pair (j, k). It is easy to see that the interaction search problem (1) can be expressed in terms of either the γ_{jk} or the normalised squared distances. Indeed

$$2\gamma_{jk} - 1 = \mathbf{Y}^T (\mathbf{X}_j \circ \mathbf{X}_k)/n = \mathbf{Z}_j^T \mathbf{X}_k / n = 1 - \|\mathbf{Z}_j - \mathbf{X}_k\|_2^2 / (2n). \tag{5}$$

Thus those pairs (j, k) with $Y^T (\mathbf{X}_j \circ \mathbf{X}_k)/n$ large will have γ_{jk} large, and $\|\mathbf{Z}_j - \mathbf{X}_k\|_2^2$ small.

This equivalence suggests that to solve (1), we can search for pairs (j, k) of columns $\mathbf{Z}_j, \mathbf{X}_k$ that are close in ℓ_2 distance. At first sight, this new problem would also appear to involve a search across all pairs, and would thus incur an $\mathcal{O}(np^2)$ cost. As mentioned in the introduction, close pair searches that avoid a quadratic cost in p incur typically an exponential cost in n. Since n would typically be much larger than $\log(p)$, such searches would be computationally infeasible.

We can however project each of the n-dimensional columns of $\mathbf{X}$ and $\mathbf{Z}$ to a lower dimensional space and then perform a close pairs search. The Johnson–Lindenstrauss Lemma, which states roughly that one can project p points into a space of dimension $\mathcal{O}(\log(p))$ and faithfully preserve distances, may appear particularly relevant here. The issue is that the projected dimension suggested by the Johnson–Lindenstrauss Lemma is still too large to allow for an efficient close pairs search. The following observation however gives some encouragement: if we had $Y = \mathbf{X}_j \circ \mathbf{X}_k$ so $\mathbf{X}_j = \mathbf{Z}_k$, even a one-dimensional projection $\mathbf{R} \in \mathbb{R}^n$ will have $|\mathbf{R}^T (\mathbf{X}_j - \mathbf{Z}_k)| = 0 = \|\mathbf{X}_j - \mathbf{Z}_k\|_2$, which implies that a perfect interaction will have zero distance in the projected space. We will later see that our approach leads to a linear runtime in such a case. Importantly, we are only interested in using a projection that preserves the distances between the close pairs rather than all pairs, which makes our problem very different to the setting considered in the Johnson–Lindenstrauss Lemma.

With this in mind, consider the following general strategy. First project the columns of $\mathbf{X}$ and $\mathbf{Z}$ to one dimensional vectors x and z using a random projection $\mathbf{R}$: $\mathbf{x} = \mathbf{X}^T \mathbf{R}$, $\mathbf{z} = \mathbf{Z}^T \mathbf{R}$. Next for some threshold τ, collect all pairs (j, k) such that $|x_j - z_k| \leq \tau$ in the set C. By first sorting $\mathbf{x}$ and $\mathbf{z}$, a step requiring only $\mathcal{O}(p \log(p))$ computations (see for example Sedgewick [1998]), this close pairs search can be shown to be very efficient. Given this set of candidate interactions, we can check for each $(j,k) \in C$ whether we have $\mathbf{Y}^T(\mathbf{X}_j \circ \mathbf{X}_k)/n > \gamma$. The process can be repeated L times with different random projections, and one would hope that given enough repetitions, any given strong interaction would be present in one of the candidate sets $C_1, \dots, C_L$ with high probability. This approach is summarised in Algorithm 1, which we term the general form of the *xyz* algorithm.

Algorithm 1 A general form of the *xyz* algorithm.

```
   Input: $\mathbf{X} \in \{-1,1\}^{n \times p}$, $\mathbf{Y} \in \{-1,1\}^n$
   Parameters: $\xi = (G, L, \tau, \gamma)$. Here $G$ is the joint distribution for the projection vector
   $\mathbf{R}$, $L$ is the number of projections $L$, and $\tau$ and $\gamma$ are the thresholds for close pairs and
   interactions strength respectively..
   Output: $I$ set of strong interactions.
1: Form $\mathbf{Z}$ via $Z_{ij} = Y_i X_{ij}$ and set $I := \emptyset$.
2: for $l \in \{1, \dots, L\}$ do
3:     Draw random vector $\mathbf{R} \in \mathbb{R}^n$ with distribution $G$ and project the data using $\mathbf{R}$, to
   form
               $\mathbf{x} = \mathbf{X}^T\mathbf{R}$ and $\mathbf{z} = \mathbf{Z}^T\mathbf{R}$.
4:     Find all pairs $C_l$ for which $(j,k)$ such that $|x_j - z_k| \leq \tau$
5:     Add to $I$ those pairs in $C_l$ for which $|\mathbf{X}_j^T \mathbf{Z}_k|/n > \gamma$.
6: end for
```

There are several parameters that must be selected, and a key choice to be made is the form of the random projection $\mathbf{R}$. For the joint distribution G of $\mathbf{R}$ we consider the following general class of distributions, which includes both dense and sparse projections. We sample a random or deterministic number M of indices from the set $\{1, \dots, n\}$, $i_1, \dots, i_M$, either with or without replacement. Then, given a distribution $F \in \mathcal{F}$ where $\mathcal{F}$ is a class of distributions to be specified later, we form a vector $\mathbf{D} \in \mathbb{R}^M$ with independent components each distributed according to F. We then define the random projection vector $\mathbf{R}$ by

$$R_i = \sum_{m=1}^{M} D_m \mathbb{1}_{\{i_m = i\}}, \qquad i = 1, \dots, n. \tag{6}$$

Each configuration of the *xyz* algorithm is characterised by fixing the following parameters:

(i) G, a distribution for the projection vector R which is determined through (6) by $F \in \mathcal{F}$, a distribution for the subsample size M and whether sampling is with replacement or not;

(ii) $L \in \mathbb{N}$, the number of projection steps;

(iii) $\tau \geq 0$, the close pairs threshold;

(iv) $\gamma \in (0,1)$, the interaction strength threshold.

We will denote the collection of all possible parameter levels by Ξ. This includes the following subclasses of interest. Fix $F \in \mathcal{F}$.

(a) **Dense projections**. Let $\mathbf{R} \in \mathbb{R}^n$ have independent components distributed according to F and denote the distribution of $\mathbf{R}$ by G. This falls within our general framework above with M set to n and sampling without replacement. Let

$$\Xi_{\text{dense}} := \{\xi \in \Xi \text{ with joint distribution equal to G}\}.$$

(b) **Subsampling**. Let $\mathcal{G}_{\text{subsample}}$ be the set of distributions for R obtained through (6) when subsampling with replacement. Let

$$\Xi_{\text{subsample}} := \{\xi \in \Xi : \text{joint distribution } G \in \mathcal{G}_{\text{subsample}}\}.$$

(c) **Minimal subsampling**. Let Ξ_{minimal} be the set of all parameters in $\Xi_{\text{subsample}}$ such that the close pairs threshold is $\tau = 0$ and M takes randomly values in the set $\{m, m+1\}$ for some positive integer m.

$$\Xi_{\text{minimal}} := \{\xi \in \Xi_{\text{subsample}} \text{ with } \tau = 0 \text{ and } M \in \{m, m+1\} \text{ for some } m \in \mathbb{N}\}.$$

Note that we have suppressed the dependence of the classes above on the fixed distribution $F \in \mathcal{F}$ for notational simplicity. We define $\mathcal{F}$ to be the set of all univariate absolutely continuous and symmetric distributions with bounded density and finite third moment. The restriction to continuous distributions in $\mathcal{F}$ ensures that Ξ_{minimal} is invariant to the choice of F: when $\tau \equiv 0$, every $F \in \mathcal{F}$ with L and the distribution for M fixed yields the same algorithm. Moreover the set of close pairs in C_l is simply the set of pairs (j,k) that have $X_{i_m j} = Z_{i_m k}$ for all $m = 1, \ldots, M$, that is the set of pairs that are equal on the subsampled rows. We note that the symmetry and boundedness of the densities in $\mathcal{F}$ and finiteness of the third moment are mainly technical conditions necessary for the theoretical developments in the following section. We will assume without loss of generality that the second moment is equal to 1. This condition places no additional restriction on Ξ since a different second moment may be absorbed into the choice of τ.

Minimal subsampling represents a very small subset of the much larger class of randomised algorithms outlined above. However, Theorem 1 below shows that minimal subsampling is essentially always at least as good as any algorithm from the wider class, which is perhaps surprising. A beneficial consequence of this result is that we only need to search for the optimal ways of selecting M and L; the threshold τ is fixed at $\tau = 0$ and the choice of the continuous distribution F is inconsequential for minimal subsampling. The choices we give in Section 2.2 yield a subquadratic runtime that approaches linear in p when the interactions to be discovered are much stronger than the bulk of the remaining interactions.

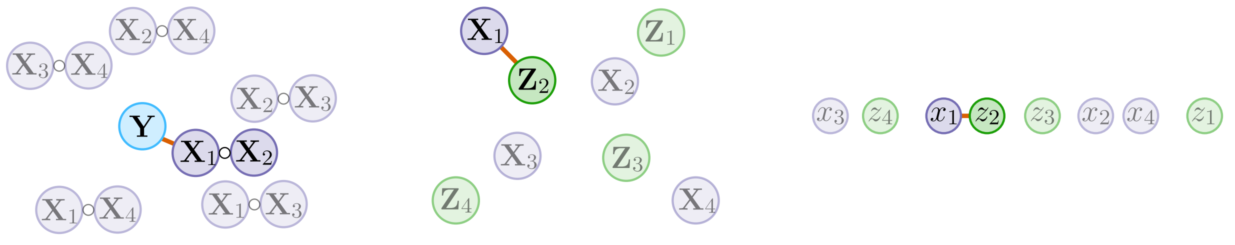

Figure 1: Illustration of the general *xyz* algorithm. The strongest interaction is the pair $(1, 2)$ and $p = 4$. Panel a) illustrates the interaction search among $\mathbf{Y}$ and $\mathbf{X}_j \circ \mathbf{X}_k$, panel b) shows the closest pair problem after the transformation $Z_{ij} = X_{ij}Y_i$ and panel c) depicts the closest pair problem after the data has been projected.

2.1 Optimality of minimal subsampling

In this section, we compare the runtime of the algorithms in $\xi \in \Xi_{\text{dense}}, \Xi_{\text{subsample}}$ and Ξ_{minimal} that return strong interactions with high probability. Let (j^*, k^*) be the indices of a strongest interaction pair, i.e. $\gamma_{j^*k^*} = \max_{j,k \in \{1,\ldots,p\}} \gamma_{jk}$. We will consider algorithms ξ with γ set to $\gamma_{j^*k^*}$. Define the power of $\xi \in \Xi$ as

$$\text{Power}(\xi) := \mathbb{P}_\xi((j^*, k^*) \in I).$$

For $\eta \in (0, 1)$, let

$$\Xi_{\text{dense}}(\eta) = \{\xi \in \Xi_{\text{dense}} : \text{Power}(\xi) \geq \eta\},$$

and define $\Xi_{\text{subsample}}(\eta)$ and $\Xi_{\text{minimal}}(\eta)$ analogously. Note that these classes depend on the underlying $F \in \mathcal{F}$, which is considered to be fixed, and moreover that we are fixing $\gamma = \gamma_{j^*k^*}$. We consider an asymptotic regime where we have a sequence of response–predictor matrix pairs $(\mathbf{Y}^{(n)}, \mathbf{X}^{(n)}) \in \mathbb{R}^n \times \mathbb{R}^{n \times p_n}$. Write $\gamma_{jk}^{(n)}$ for the corresponding interaction strengths, and let $\gamma_1^{(n)} = \max_{j,k} \gamma_{jk}^{(n)}$. Let $f_{\boldsymbol{\gamma}^{(n)}}$ be the probability mass function corresponding to drawing an element of $\boldsymbol{\gamma}^{(n)}$ uniformly at random. Note that $f_{\boldsymbol{\gamma}^{(n)}}$ has domain $\{0, 1/n, 2/n, \ldots, 1\}$. We make the following assumptions about the sequence of interaction strength matrices $\boldsymbol{\gamma}^{(n)}$.

(A1) There exists c_0 such that $|\{(j,k) : \gamma_{jk}^{(n)} = \gamma_1^{(n)}\}| \leq c_0 p_n$.

(A2) There exists $\gamma_l > 0$, $\gamma_u < 1$ such that $\gamma_u \geq \gamma_1^{(n)} \geq \gamma_l$ for all n.

(A3) There exists $\rho < 1$ such that $f_{\boldsymbol{\gamma}^{(n)}}$ is non-increasing on $[\rho\gamma_1^{(n)}, \gamma_1^{(n)}) \cap \{0, 1/n, \ldots, 1\}$.

Assumption (A1) is rather weak: typically one would expect the maximal strength interaction to be essentially unique, while (A1) requires that at most of order p_n interactions have maximal strength. (A2) requires the maximal interaction strength to be bounded away from 0 and 1, which is the region where complexity results for the search of interactions are of interest. As mentioned earlier, if the maximal interaction strength is 1, it will always be retained in the close-pair sets C_l, whilst if its strength is too close to 0, then it is near impossible to distinguish it from the remaining interactions. (A3) ensures a certain form of separation between maximal strength interactions and the bulk of the interactions.

To aid readability, in the following we suppress the dependence of quantities on n in the notation. Given $\mathbf{X}$ and $\mathbf{Y}$, we may define $T(\xi)$ as the expected number of computational operations performed by the algorithm corresponding to ξ. We have the following result.

Theorem 1. *Given $F \in \mathcal{F}$ and $\eta \in (0,1)$, there exists n_0 such that for all $n \geq n_0$ we have*

$$\inf_{\xi \in \Xi_{minimal}(\eta)} T(\xi) = \inf_{\xi \in \Xi_{subsample}(\eta)} T(\xi), \tag{7}$$

$$\inf_{\xi \in \Xi_{minimal}(\eta)} \frac{T(\xi)}{np^2} \to 0, \tag{8}$$

and there exists $c > 0$ such that

$$\inf_{\xi \in \Xi_{dense}(\eta)} \frac{T(\xi)}{np^2} > c. \tag{9}$$

The theorem shows that the optimal runtime is achieved when using minimal subsampling. The last point is surprising: setting $\mathbf{R} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ will not improve the computational complexity over the brute-force approach and dense Gaussian projections hence do not reduce the complexity of the search.

2.2 The final version of xyz

The optimality properties of minimal subsampling presented in the previous section suggest the approach set out in Algorithm 2, which we will refer to as the *xyz* algorithm. Here we

Algorithm 2 Final version of the *xyz* algorithm.

```
    Input: $\mathbf{X} \in \{-1,1\}^{n \times p}$, $\mathbf{Y} \in \{-1,1\}^n$, subsample size $M$, number of projections $L$, threshold for interaction strength $\gamma$.
    Output: $I$ set of strong interactions.
1:  Form $\mathbf{Z}$ via $Z_{ij} = Y_i X_{ij}$.
2:  for $l \in \{1, \ldots, L\}$ do
3:      Form $\mathbf{R} \in \mathbb{R}^n$ as in (6) with distribution $F = U[0,1]$ and set $\mathbf{x} = \mathbf{X}^T\mathbf{R}$, $\mathbf{z} = \mathbf{Z}^T\mathbf{R}$.
4:      Find all pairs $(j,k)$ such that $x_j = z_k$ and store these in $C_l$.
5:      Add to $I$ those pairs in $C_l$ for which $|\mathbf{X}_j^T\mathbf{Z}_k|/n > \gamma$.
6:  end for
```

are using a simplified version of the minimal subsampling proposal given in the previous section where we keep M fixed rather than allowing it to be random. The reason is that the potential additional gain from allowing M to be any one of two consecutive numbers with certain probabilities is minimal but necessary for Theorem 1 and so the simpler approach is preferable. We note that the uniform distribution in line 3 may be replaced with any continuous distribution to yield identical results.

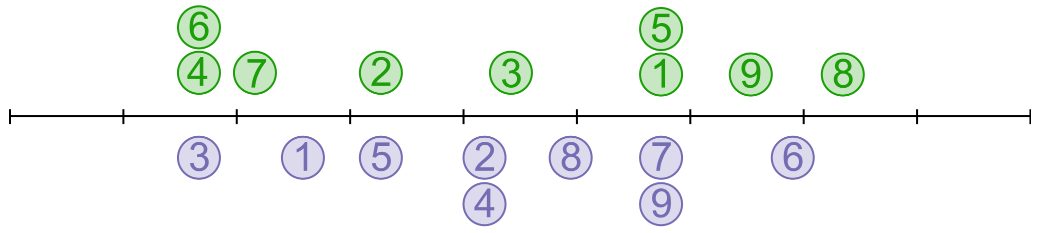

Figure 2: Illustration of an equal pairs search among components of $\mathbf{x}, \mathbf{z} \in \mathbb{R}^p$ when $p = 9$. The horizontal locations of blue and green circles numbered j give x_j and z_j respectively. Sorting of $(\mathbf{x}, \mathbf{z})$ allows traversal of the unique locations. At each of these it is checked whether points of both colours are present, and if so, the indices are recorded. Here the set of close pairs $(\{3\} \times \{4, 6\}) \cup (\{7\} \times \{1, 5\}) \cup (\{9\} \times \{1, 5\})$ would be returned.

To perform the equal pairs search in line 4, we sort the concatenation $(\mathbf{x}, \mathbf{z}) \in \mathbb{R}^{2p}$ to determine the unique elements of $\{x_1, \dots, x_p, z_1, \dots, z_p\}$. At each of these locations, we can check if there are components from both x and z lying there, and if so record their indices. This procedure, which is illustrated in Figure 2, gives us the set of equal pairs C in the form of a union of Cartesian products. The computational cost is $\mathcal{O}(p \log(p))$. This complexity is driven by the cost of sorting whilst the recording of indices is linear in p. We note, however, that looping through the set of equal pairs in order to output a list of close pairs of the form $(j_1, k_1), \dots, (j_{|C|}, k_{|C|})$ would incur an additional cost of the size of C, though in typical usage we would have $|C| = o(p)$. Readers familiar with locality sensitive hashing can find a short interpretation of equal pairs search as an LSH-family in the appendix. In the next section, we discuss in detail the impact of minimal subsampling on the complexity of the *xyz* algorithm and the discovery probability it attains.

2.3 Computational and statistical properties of xyz

We have the following upper bound on the expected number of computational operations performed by *xyz* (Algorithm 2) when the subsample size and number of repetitions are M and L:

$$C(M, L) := \underset{\text{(i)}}{np} + L\{\underset{\text{(ii)}}{Mp} + \underset{\text{(iii)}}{p\log(p)} + \underset{\text{(iv)}}{n\mathbb{E}_\xi(|C_1|)}\}. \tag{10}$$

The terms may be explained as follows: (i) construction of $\mathbf{Z}$; (ii) multiplying M subsampled rows of $\mathbf{X}$ and $\mathbf{Z}$ by $\mathbf{R} \in \mathbb{R}^n$; (iii) finding the equal pairs; (iv) checking whether the interactions exceed the interaction strength threshold γ. Note we have omitted a constant factor from the upper bound $C(M, L)$. There is a lower bound only differing from (10) in the equal pairs search term (iii), which is p instead of $p \log(p)$. It will be shown that (iv) is the dominating term and therefore the upper and lower bound are asymptotically equivalent, implying the bounds are tight.

An interaction with strength γ is retained in C_1 with probability γ^M. Hence it is present in the final set of interactions I with probability

$$\eta(M, L, \gamma) = 1 - (1 - \gamma^M)^L. \tag{11}$$

The following result demonstrates how the *xyz* algorithm can be used to find interactions whilst incurring only a subquadratic computational cost.

Theorem 2. *Let F_Γ be the distribution function corresponding to a random draw from the set of interaction strengths $\{\gamma_{jk}\}_{j,k\in\{1,\ldots,p\}}$. Given an interaction strength threshold γ, let $1 - F_\Gamma(\gamma) = c_1/p$. Define $\gamma_0 = p^{-1/M}$ and let c_2 be defined by $1 - F_\Gamma(\gamma_0) = c_2 p^{\log(\gamma)/\log(\gamma_0)-1}$. We assume that $\gamma_0 < \gamma$. Finally given a discovery threshold $\eta' \in [1/2, 1)$ let L be the minimal L' such that $\eta(M, L', \gamma) \geq \eta'$. Ignoring constant factors we have*

$$C(M, L) \leq \log\{1/(1 - \eta')\}(1 + c_1 + c_2)[\{1 + 1/\log(\gamma_0^{-1})\}\log(p) + n]p^{1+\log(\gamma)/\log(\gamma_0)}.$$

If $n \gg \log(p)$ and γ_0 is bounded away from 1 we see that the dominant term in the above is

$$cnp^{1+\log(\gamma)/\log(\gamma_0)}, \tag{12}$$

where $c = \log\{1/(1 - \eta')\}(1 + c_1 + c_2)$. Typically we would expect γ to be such that $|\{\gamma_{jk} : \gamma_{jk} > \gamma\}| \sim p$ as only the largest interactions would be of interest: thus we may think of c_1 as relatively small. If M is such that γ_0 is also larger than the bulk of the interactions, we would also expect c_2 to be small. Indeed, suppose that the proportion of interactions whose strengths are larger than γ_0 is $1 - F_\Gamma(\gamma_0) = c_1'/p$. Then $c_2 = c_1'/p^{\log(\gamma)/\log(\gamma_0)} < c_1'$. As a concrete example, if $\gamma = 0.9$ and M is such that $\gamma_0 = 0.55$, the exponent in (12) is around 1.17, which is significantly smaller than the exponent of 2 that a brute-force approach would incur; see also the examples in Section 5. Note also that when $\gamma = 1$, the exponent is 1 for all $\gamma_0 < 1$: if we are only interested in interactions whose strength is as large as possible, we have a runtime that is linear in p.

It is interesting to compare our results here with the runtime of approaches based on fast matrix multiplication. By computing $\mathbf{X}^T\mathbf{Z}$ we may solve the interaction search problem (1). Naive matrix multiplication would require $\mathcal{O}(np^2)$ operations, but there are faster alternatives when $n = p$. The fastest known algorithm [Williams, 2012] gives a theoretical runtime of $\mathcal{O}(np^{1.37})$ when $n = p$. For *xyz* to achieve such a runtime when $\gamma_0 = 0.55$ for example, the target interaction strength would have to be $\gamma \geq 0.81$: a somewhat moderate interaction strength. For $\gamma > 0.81$, *xyz* is strictly better; we also note that fast matrix multiplication algorithms tend to be unstable and are therefore rarely used in practice. A further advantage is that the *xyz* algorithm has an optimal memory usage of $\mathcal{O}(np)$.

We also note that whilst Theorem 2 concerns the the discovery of any single interaction with strength at least γ, the runtime required to discover a fixed number interactions with strength at least γ would only differ by a multiplicative constant. If we however want a guarantee of discovering the p strongest pairs the bound in Theorem 2 would no longer hold.

To minimise the runtime in (12), we would like γ_0 to be larger than most of the interactions in order that c_2 and hence c be small, yet a smaller γ_0 yields a more favourable exponent. Thus a careful choice of M, on which γ_0 depends, is required for *xyz* to enjoy good performance. In the following we show that a choice M exists, and we discuss how this optimal M may be estimated based on the data.

Clearly if for some pair (M, L), we find another pair (M', L') with $\eta(M'.L') > \eta(M, L)$ but $C(M', L') \leq C(M, L)$, we should always use (M', L') rather than (M, L). It turns out that there is in fact an optimal choice of M such that the parameter choice is not dominated by any others in this fashion. Define

$$M^* = \underset{M \in \mathbb{N}}{\arg\min} \left\{ -\frac{1}{\log(1-\gamma^M)} \left(Mp + p\log(p) + n \sum_{j,k} \gamma_{jk}^M \right) \right\}, \tag{13}$$

where it is implicitly assumed that the minimiser is unique. This will always be the case except for peculiar values of γ and γ_{jk}.

Proposition 3. *Let $L \in \mathbb{N}$. If $(M', L') \in \mathbb{N}^2$ has $\eta(M', L') \geq \eta(M^*, L)$, then also $C(M', L') \geq C(M^*, L)$ with the final inequality being strict if $M' \neq M^*$ and M^* is a unique minimiser.*

Thus there is a unique Pareto optimal M. Although the definition of M^* involves the moments of F_Γ, this can be estimated by sampling from $\{\gamma_{jk}\}$. We can then numerically optimise a plugin version of the objective to arrive at an approximately optimal M.

3 Interaction search on continuous data

In the previous section we demonstrated how the *xyz* algorithm can be used to efficiently solve the simplest form of interaction search (1) when both $\mathbf{X}$ and $\mathbf{Y}$ are binary. In this section we show how small modifications to the basic algorithm can allow it to do the same when $\mathbf{Y}$ is continuous, and also when $\mathbf{X}$ is continuous. We consider the regression setting in Section 4.

3.1 Continuous Y and binary $\mathbf{X}$

We begin by considering the setting where $\mathbf{X} \in \{-1, 1\}^{n \times p}$, but where we now allow real-valued $\mathbf{Y} \in \mathbb{R}^n$. Without loss of generality, we will assume $\|\mathbf{Y}\|_1 = 1$. The approach we take is motivated by the observation that the inner product $\mathbf{Y}^T(\mathbf{X}_j \circ \mathbf{X}_k)$ can be interpreted as a weighted inner product of $\mathbf{X}_j \circ \mathbf{X}_k$ with the sign pattern of $\mathbf{Y}$, using weights $w_i = |Y_i|$.

With this in mind, we modify *xyz* in the following way. We set $\mathbf{Z}$ to have $Z_{ij} = \mathrm{sgn}(Y_i)X_{ij}$. Let $i_1, \ldots, i_M \in \{1, \ldots, n\}$ be i.i.d. such that $\mathbb{P}(i_s = i) = w_i$. Forming the projection vector $\mathbf{R}$ using (6), we then find the probability of (j, k) being in the equal pairs set may be computed as follows.

$$\begin{aligned}\{\mathbb{P}(\mathbf{R}^T\mathbf{X}_j = \mathbf{R}^T\mathbf{Z}_k)\}^{1/M} &= \mathbb{P}(X_{i_1 j} = \mathrm{sgn}(Y_{i_1})X_{i_1 k}) \\ &= \sum_{i=1}^{n} \mathbb{P}(X_{i_1 j} = \mathrm{sgn}(Y_{i_1})X_{i_1 k} | i_1 = i)\mathbb{P}(i_1 = i) \\ &= \sum_{i=1}^{n} |Y_i| \mathbb{1}_{\{X_{ij} = \mathrm{sgn}(Y_i)X_{ik}\}} \\ &= \sum_{i:\mathrm{sgn}(Y_i) = X_{ij}X_{ik}} Y_i X_{ij} X_{ik} =: \tilde{\gamma}_{jk},\end{aligned}$$

where $\mathbb{P}$ is always with respect to $\mathbf{R}$ (and, equivalently, the random indices $i_1, \dots, i_M$) with $\mathbf{Y}$ and $\mathbf{X}$ considered fixed. The calculation above shows that the runtime bound of Theorem 2 continues to hold in the setting with continuous Y provided we replace the interaction strengths γ_{jk} with their continuous analogues $\tilde{\gamma}_{jk}$.

As a simple example, consider the model

$$Y_i = X_{i1}X_{i2} + \varepsilon_i,$$

with $\varepsilon_i \sim \mathcal{N}(0, \sigma^2)$ and $\mathbf{X}$ generated randomly having each entry drawn independently from $\{-1, 1\}$ each with probability $1/2$. Then for a non interacting pair $j \neq 1, 2$ or $k \neq 1, 2$, we have $\tilde{\gamma}_{jk} \approx 0.5$. For the pair $(1, 2)$ we calculate an interaction strength of

$$\begin{aligned}\tilde{\gamma}_{12} &= \mathbb{P}(\mathrm{sgn}(Y_{i_1}) = X_{i_1 1}X_{i_1 2}) = \mathbb{P}(\mathrm{sgn}(X_{i_1 1}X_{i_1 2} + \varepsilon_i) = X_{i_1 1}X_{i_1 2}) \\ &= \mathbb{P}(|\varepsilon_i| < 1) + \frac{1}{2}\mathbb{P}(|\varepsilon_i| > 1) = \frac{1}{2}(1 + \mathbb{P}(|\varepsilon_i| < 1)).\end{aligned}$$

A quick simulation gives the following table:

σ^2	0.1	0.25	0.5	1	2	5
$\tilde{\gamma}_{12}$	0.99	0.98	0.92	0.84	0.76	0.67

Using Theorem 2 and the above table we can estimate the computational complexity needed to discover the pair $(1, 2)$ given a value of σ^2.

3.2 Continuous Y and continuous $\mathbf{X}$

If both $\mathbf{X}$ and $\mathbf{Y}$ are continuous the idea is to transform $\mathbf{X}$ to a binary matrix and then use the method discussed in the preceding section. Such a transformation would have the following aim: given the original matrix $\mathbf{X}$, find a binary representation $\tilde{\mathbf{X}}$, such that the strongest interactions in $\mathbf{X}$ remain the strongest interactions under $\tilde{\mathbf{X}}$. We describe two transformations below.

- **Unbiased transform:** We transform $\mathbf{X}$ such that the inner product $\mathbf{Y}^T(\mathbf{X}_j \circ \mathbf{X}_k)$ remains the same. Assume there exists $a_j, b_j \in \mathbb{R}$ so that $\mathbf{X}_j \in [a_j, b_j]^n$. Define the distribution of $\tilde{X}_{ij}$ as:

$$\mathbb{P}(\tilde{X}_{ij} = a_j) = \frac{b_j - X_{ij}}{b_j - a_j} \quad \text{and} \quad \mathbb{P}(\tilde{X}_{ij} = b_j) = \frac{X_{ij} - a_j}{b_j - a_j}.$$

It follows $\mathbb{E}[\tilde{X}_{ij}] = X_{ij}$ and if each variable is transformed independently from another then we have

$$\mathbb{E}[\sum_{i=1}^{n} Y_i \tilde{X}_{ij} \tilde{X}_{ik}] = \sum_{i=1}^{n} Y_i \mathbb{E}[\tilde{X}_{ij} \tilde{X}_{ik}] = \sum_{i=1}^{n} Y_i \mathbb{E}[\tilde{X}_{ij}]\mathbb{E}[\tilde{X}_{ik}] = \sum_{i=1}^{n} Y_i X_{ij} X_{ik},$$

since the only randomness comes from $\tilde{X}_{ij}$ and we sample the variables independently. Thus this transform is unbiased. Using $\mathbb{E}[\tilde{X}_{ij}^2] = X_{ij}(b_j + a_j) - a_j b_j$, the mean square error is then:

$$\begin{aligned}\mathbb{E}[(Y_i \tilde{X}_{ij} \tilde{X}_{ik} - Y_i X_{ij} X_{ik})^2] &= Y_i^2(\mathbb{E}[\tilde{X}_{ij}^2]\mathbb{E}[\tilde{X}_{ik}^2] - X_{ij}^2 X_{ik}^2) \\ &= Y_i^2((X_{ij}(b_j + a_j) - a_j b_j)(X_{ik}(b_k + a_k) - a_k b_k) - X_{ij}^2 X_{ik}^2) \\ &\leq Y_i^2(\max(a_j^2, b_j^2)\max(a_k^2, b_k^2) - \min(a_j^2, b_j^2)\min(a_k^2, b_k^2)).\end{aligned}$$

To decrease the variance of this transform one might consider to not take $a_j = \min_i(X_{ij})$, but instead one could take the first quartile. This way the variance gets reduced at the cost of an increased bias.

- **Deterministic transform:** An alternative to the unbiased transform is to transform the variables in a biased way but without any variance. Given $a_j, b_j, c_j \in \mathbb{R}$ (a_j, b_j are not related to the unbiased transform above), define the transformed variable

$$\tilde{X}_{ij} = \begin{cases} a_j & \text{if } X_{ij} \leq c_j \\ b_j & \text{if } X_{ij} > c_j. \end{cases}$$

We fix the parameters a_j, b_j and c_j such that

$$(\hat{a}_j, \hat{b}_j, \hat{c}_j) = \underset{(a_j, b_j, c_j) \in \mathbb{R}^3}{\text{argmin}} \|\mathbf{X}_j - \tilde{\mathbf{X}}_j\|_1.$$

This is similar 2-means clustering on $\mathbf{X}_j$, with two clusters around $\{a_j, b_j\}$. This transform is not unbiased any more but has zero variance.

Both these transforms can be efficiently conducted in $\mathcal{O}(np)$. In practice we think one might test both transformation to check which one works better on a given dataset. Again for these transforms to be compatible with our method one would need $a_j = -1$ and $b_j = 1$. We observed that in most applications it is fine to enforce such a parametrisation. If data is short-tailed and symmetric around 0, this will not affect the outcome much. To see this in practice, consider the regression experiment in section 5.

4 Application to Lasso regression

Thus far we have only considered the simple version of the interaction search problem (1) involving finding pairs of variables whose interaction has a large dot product with $\mathbf{Y}$. In this section we show how any solution to this, and in particular the *xyz* algorithm, may be used to fit the Lasso [Tibshirani, 1996] to all main effects and pairwise interactions in an efficient fashion.

Given a response $\mathbf{Y} \in \mathbb{R}^n$ and a matrix of predictors $\mathbf{X} \in \mathbb{R}^{n\times p}$, let $\mathbf{W} \in \mathbb{R}^{n\times p(p+1)/2}$ be the matrix of interactions defined by

$$\mathbf{W} = (\mathbf{X}_1 \circ \mathbf{X}_1, \mathbf{X}_1 \circ \mathbf{X}_2, \cdots, \mathbf{X}_1 \circ \mathbf{X}_p, \mathbf{X}_2 \circ \mathbf{X}_2, \mathbf{X}_2 \circ \mathbf{X}_3, \cdots, \mathbf{X}_p \circ \mathbf{X}_p).$$

We will assume that the $\mathbf{Y}$ and the columns of $\mathbf{X}$ have been centred. Let $\tilde{\mathbf{W}}$ be a version of $\mathbf{W}$ with centred columns. Consider the Lasso objective function

$$(\hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\theta}}) = \underset{\boldsymbol{\beta}\in\mathbb{R}^p, \boldsymbol{\theta}\in\mathbb{R}^{p(p+1)/2}}{\operatorname{argmin}} \left\{ \frac{1}{2n} \|\mathbf{Y} - \mathbf{X}\boldsymbol{\beta} - \tilde{\mathbf{W}}\boldsymbol{\theta}\|_2^2 + \lambda(\|\boldsymbol{\beta}\|_1 + |\boldsymbol{\theta}\|_1) \right\}. \tag{14}$$

Note that since the entire design matrix in the above is column-centred, any intercept term would always be zero.

In order to avoid a cost of $\mathcal{O}(np^2)$ it is necessary to avoid explicitly computing $\mathbf{W}$. To describe our approach, we first review in Algorithm 3 the active set strategy employed by several of the fastest Lasso solvers such as `glmnet` [Friedman et al., 2010]. We use the notation that for a matrix $\mathbf{M}$ and set of column indices H, $\mathbf{M}_H$ is the submatrix of $\mathbf{M}$ formed from those columns indexed by H. Similarly for a vector $\mathbf{v}$ and component indices H, $\mathbf{v}_H$ is the subvector of $\mathbf{v}$ formed from the components of $\mathbf{v}$ indexed by H.

Algorithm 3 Active set strategy for Lasso computation

```
   Input: $\mathbf{X}$, $\mathbf{Y}$ and grid of $\boldsymbol{\lambda}$ values $\lambda_1 < \cdots < \lambda_L$.
   Output: Lasso solutions $\hat{\boldsymbol{\beta}}_{\lambda_l}$ and $\hat{\boldsymbol{\theta}}_{\lambda_l}$ at each $\lambda$ on the grid.
1: for $l \in \{1, \dots, L\}$ do
2:     If $l = 1$ set $A, B = \emptyset$; otherwise set $A = \{k : \hat{\beta}_{\lambda_{l-1},k} \neq 0\}$ and $B = \{k : \hat{\theta}_{\lambda_{l-1},k} \neq 0\}$.
3:     Compute the Lasso solution $(\hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\theta}})$ when $\lambda = \lambda_l$ under the additional constraint that
       $\hat{\boldsymbol{\beta}}_{A^c} = 0$ and $\hat{\boldsymbol{\theta}}_{B^c} = 0$.
4:     Let $U = \{k : |\mathbf{X}_k^T(\mathbf{Y} - \mathbf{X}_A\hat{\boldsymbol{\beta}}_A - \tilde{\mathbf{W}}_B\hat{\boldsymbol{\theta}}_B)|/n > \lambda_l\}$ and $V = \{k : |\tilde{\mathbf{W}}_k^T(\mathbf{Y} - \mathbf{X}_A\hat{\boldsymbol{\beta}}_A -$
       $\tilde{\mathbf{W}}_B\hat{\boldsymbol{\theta}}_B)|/n > \lambda_l\}$ be the set of coordinates that violate the KKT conditions when $(\hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\theta}})$
       is taken as a candidate solution.
5:     If $U$ and $V$ are empty, we set $\hat{\boldsymbol{\beta}}_{\lambda_l} = \hat{\boldsymbol{\beta}}$, $\hat{\boldsymbol{\theta}}_{\lambda_l} = \hat{\boldsymbol{\theta}}$. Else we update $A = A \cup U$ and
       $B = B \cup V$ and return to line 3.
6: end for
```

As the sets A and B would be small, computation of the Lasso solution in line 3 is not too expensive. Instead line 4, which performs a check of the KKT conditions involving dot products of all interaction terms and the residuals, is the computational bottleneck: a naive approach would incur a cost of $\mathcal{O}(np^2)$ at this stage.

There is however a clear similarity between the KKT condition check for the interactions and the simple interaction search problem (1). Indeed the computation of V may be expressed in the following way:

$$\text{Keep all pairs } (j,k) \text{ for which } |(\mathbf{Y} - \mathbf{X}_A\hat{\boldsymbol{\beta}}_A - \tilde{\mathbf{W}}_B\hat{\boldsymbol{\theta}}_B)^T(\mathbf{X}_j \circ \mathbf{X}_k)/n| > \lambda_l. \tag{15}$$

Note that since $\mathbf{Y} - \mathbf{X}_A\hat{\boldsymbol{\beta}}_A - \tilde{\mathbf{W}}_B\hat{\boldsymbol{\theta}}_B$ is necessarily centred, there is no need to centre the interactions in (15). In order to solve (15) we can use the *xyz* algorithm, setting γ in Algorithm 2 to λ_l and $\mathbf{Y}$ to each of $\pm(\mathbf{Y} - \mathbf{X}_A\hat{\boldsymbol{\beta}}_A - \tilde{\mathbf{W}}_B\hat{\boldsymbol{\theta}}_B)$ in turn.

Precisely the same strategy of performing KKT condition checks using *xyz* can be used to accelerate computation for interaction modelling for a variety of variants of the Lasso such as the elastic net [Zou and Hastie, 2005] and ℓ_1-penalised generalised linear models. Note also that it is straightforward to use a different scaling for the penalty on the interaction coefficients in (14), which may be helpful in practice.

5 Experiments

To test the algorithm and theory developed in the previous sections, we run a sequence of experiments on real and simulated data.

5.1 Comparison of minimal subsampling and dense projections

In this experiment we consider dense Gaussian projections ξ_{Gauss} and minimal subsampling $\xi_{minimal}$ to compare their probability to retain strong interactions in C_1; the set of close or equal pairs. For simplicity assume that there is one strong interaction with strength γ and the remainder of the pairs have an interaction strength of $\gamma_0 = 0.5 < \gamma$. For a comparison on even grounds we set the parameters so that the average size of C_1 is equal to p (i.e. $\mathbb{E}[|C_1|] = p$).

- ξ_{Gauss}: $L = 1$ and $\tau \geq 0$ is the $1/p$ quantile of the distribution of $|W|$ when $W \sim N(0, n(1-\gamma_0))$.
- $\xi_{minimal}$: $L = 1$ and $M = \log(\gamma_0)/\log(1/p)$.

This way we ensure that the same number of pairs is considered after each projection. The size of L is set to 1 as it does not matter for the discovery probability how many projections are conducted, as long as we conduct the same number we can compare the two approaches. We then plot η, the probability of discovering the interaction of strength γ, as a function of γ for different values of p (Figure 3). For $\xi_{minimal}$, η is given in equation (11). For ξ_{Gauss}, η is the $1/p$ quantile of the distribution of $|W|$ when $W \sim N(0, n(1-\gamma))$. We simulate η for values of p ranging from 10 up to 10^6. The results clearly show that in terms of discovery probability

$\xi_{minimal}$ outperforms ξ_{Gauss} by magnitudes. The impact of this different behaviour on the runtime is also highlighted by Theorem 1.

5.2 Scaling

In this experiment we test how the *xyz* algorithm scales on a simple test example as we increase the dimension p. We generate data $\mathbf{X} \in \mathbb{R}^{n \times p}$ with each entry sampled independently uniformly from $\{-1, 1\}$. We do this for different values of p, ranging from 1000 to 30000: this way for the largest p considered there are more than 400 million possible interactions. Then for each $\mathbf{X}$ we construct response vectors Y such that only the pair $(1, 2)$ is a strong interaction with an interaction strength taking values in $\{0.7, 0.8, 0.9\}$. Through this construction, if n is large enough, all the pairs except $(1, 2)$ will have an interaction strength around 0.5, and very few will have one above 0.55. We thus have $\gamma_0 = 0.55$ and we set M so that $\gamma_0 = p^{-1/M} = 0.55$. Since the only strong interaction is $(1, 2)$, we get $\gamma = \gamma_{12}$.

Each data set configuration (a combination of p and γ_{12}) is simulated 300 times and we measure the time it takes *xyz* to find the pair $(1, 2)$. We plot the average runtime against the dimension p. The different choices for γ_{12} are highlighted in different colours (Figure 3).

Theorem 2 indicates that the runtime should be of the order $np^{1+\log(\gamma)/\log(\gamma_0)}$. We see that the experimental results here are in close agreement with this prediction.

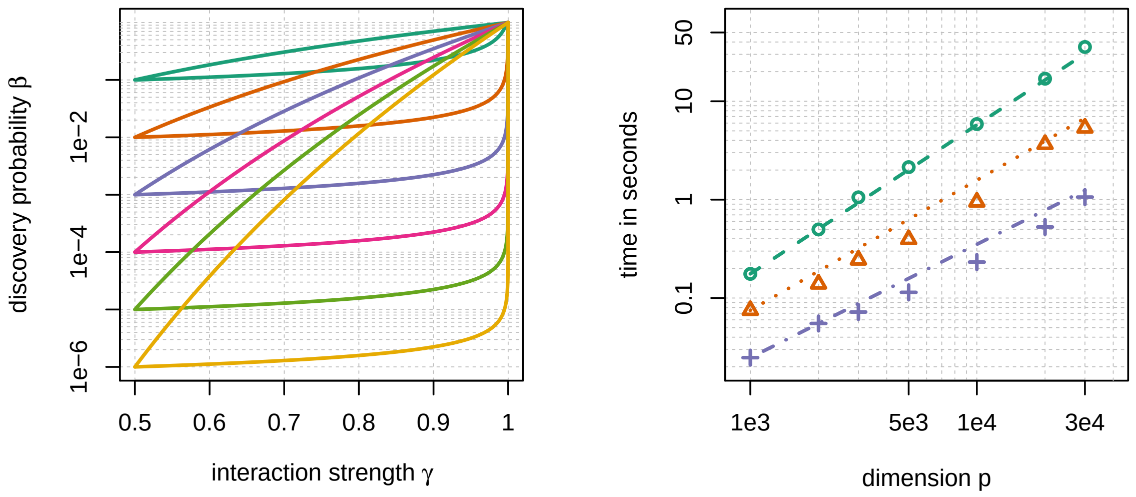

Figure 3: Left panel: Discovery probability as a function of γ for different values of $p \in \{10^1, \ldots, 10^6\}$ (colours decreasing in p from yellow $p = 10^6$ to green $p = 10$). The lower lines correspond to the dense Gaussian projections, the upper lines to minimal subsampling. Right panel: Time to discover the interaction pair as a function of the dataset dimension p. Lines correspond to the theoretical prediction and symbols are the actual measured runtime. Colour coding: green $\gamma = 0.7$, orange $\gamma = 0.8$ and purple $\gamma = 0.9$.

5.3 Run on SNP data

In the next experiment we compare the performance of *xyz* to its closest competitors on a real dataset. For each method we measure the time it takes to discover strong interactions. We consider the LURIC dataset [Winkelmann et al., 2001], which contains data of patients that were hospitalised for coronary angiography. We use a preprocessed version of the dataset that is made up of $n = 859$ observations and 687253 predictors. The dataset is binary. The response $\mathbf{Y}$ indicates coronary disease (1 corresponding to affected and -1 healthy) and $\mathbf{X}$ contains SNPs (Single Nucleotide Polymorphism) which are variations of base-pairs on DNA. The response vector $\mathbf{Y}$ is strongly unbalanced: there are 681 affected cases ($Y_i = 1$) and 178 unaffected ($Y_i = -1$). For a fair comparison among the classes -1 and 1 we generate a subsample of the dataset with equal class distributions. We repeat the subsampling a few times and pick the interactions that consistently appear over many subsamples.

To get a contrast of the performance of *xyz* we compare it to *epiq* [Arkin et al., 2014], another method for fast high dimensional interaction search. In order for *epiq* to detect interactions it needs to assume the model

$$Y_i = \alpha_{j^*k^*} X_{ij^*} X_{ik^*} + \varepsilon_i, \tag{16}$$

where $\varepsilon_i \sim \mathcal{N}(0, \sigma^2)$. It then searches for interactions by considering the test statistics

$$T_{jk} = (\mathbf{R}^T(\mathbf{Y} \circ \mathbf{X}_j))(\mathbf{R}^T \mathbf{X}_k)$$

where $\mathbf{R} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. These are used to try to find the pair (j^*, k^*), which is assumed to be the pair for which the inner product $\mathbf{Y}^T(\mathbf{X}_j \circ \mathbf{X}_k)$ is maximal. It is an easy calculation to show that $\mathbb{E}[T_{jk}] = \mathbf{Y}^T(\mathbf{X}_j \circ \mathbf{X}_k)$. To maximise the inner product on the right, *epiq* considers pairs where T_{jk}^2 is large by looking at pairs where both $(\mathbf{R}^T(\mathbf{Y} \circ \mathbf{X}_j))^2$ and $(\mathbf{R}^T \mathbf{X}_k)^2$ are large. While the approach of *epiq* is somewhat related to *xyz*, there are no bounds available for the time it takes to find strong interactions.

We also compare both methods to a naive approach where we subsample a fixed number of interactions uniformly at random, and retain the strongest one. We refer to this as *naive search*.

At fixed time intervals we check for the strongest interaction found so far with all three methods. We plot the interaction strength as a function of the computational time (Figure 4). All three methods eventually discover interactions of very similar strength and it would be a hasty judgement to say whether one significantly outperforms the others. *xyz* nevertheless discovers the strongest interactions on average for a fixed runtime compared to the other two approaches. To get a clearer picture we run two additional experiments on a slight modification of the LURIC dataset. We implant artificial interactions where we set the strength to $\gamma_{12} = 0.8$ and another example with $\gamma_{12} = 0.9$. In these two experiments *xyz* clearly outperforms all other methods considered (Figure 4; panels 3 and 4). Besides *xyz* being the fastest at interaction search, it also offers a probabilistic guarantee that there are no strong interactions left in the data. This guarantee comes out of Theorem 2. To run *xyz* we have to calculate the optimal subsample size (13) for use of minimal subsampling:

$$M^* = \arg\min_{M \in \mathbb{N}} \left\{ -\frac{1}{\log(1-\gamma^M)} \left(Mp + p\log(p) + n\sum_{j,k} \gamma_{jk}^M \right) \right\} = 21.$$

The sum in this optimisation can be approximated by uniformly sampling over pairs. Assume we have an interaction pair (j^*, k^*) with interaction strength $\gamma_{j^*k^*} = 0.85$. The probability that we discover this pair in one run $(L = 1)$ of the *xyz* algorithm is $\gamma_{j^*k^*}^{21}$. Therefore the probability of missing this pair after $L = 100$ runs is given by

$$(1 - \gamma_{j^*k^*}^{21})^L \approx 0.03.$$

Note that the number of possible interactions is $p(p-1)/2 \approx 10^{11}$. The whole search took 280 seconds. Naive search offers a similar guarantee, however it is extremely weak. The probability of not discovering the pair after drawing pL samples (with $L = 100$) is bounded by $(1 - 2/(p(p-1)))^{Lp} \approx 0.999$. If we consider the runtime guarantee from theorem 2, the

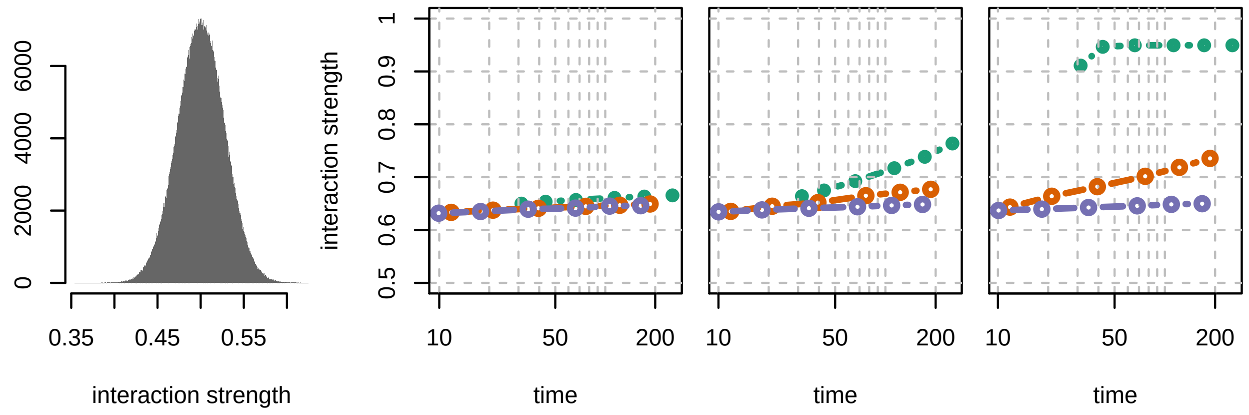

Figure 4: Left: Histogram of interaction strength of 10^6 interaction pairs, sampled at random from the more than 10^{11} existing pairs. The other three panels show the interaction strength of the discovered pairs as a function of the computation time (*xyz* (green line), *epiq* (orange dashed) and naive search (violet points)). The left panel is on the original LURIC dataset, the center one has an implanted interaction of strength $\gamma_{12} = 0.8$ and the right one $\gamma_{12} = 0.95$.

complexity of *xyz* in terms of p we have

$$p^{1+\frac{\log(0.85)}{\log(0.5)}} \approx p^{1.23}.$$

Compare this to the expected runtime of order p^2 for naive search, which means that *xyz* is about 30000-times faster than naive search (when $p = 687253$). In the empirical comparison this factor is around 20000.

5.4 Regression

Finally we want to demonstrate the capabilities of *xyz* in interaction search for continuous data as explained in Section 3. We simulate two different models of the form (2):

$$Y_i = \mu + \sum_{j=1}^{p} X_{ij}\beta_j + \sum_{k=1}^{p}\sum_{k=1}^{j-1} X_{ij}X_{ik}\theta_{jk} + \varepsilon_i.$$

We consider three settings. For all three settings we have $n = 1000$. We let $p \in \{250, 500, 750, 1000\}$. Each row of $\mathbf{X}$ is generated i.i.d. as $\mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$. The magnitudes of both the main and interaction effects are chosen uniformly from the interval $[2, 6]$ (20 main effects and 10 interaction effects) and we set $\varepsilon_i \sim \mathcal{N}(0, 1)$. The three settings we consider are as follows.

1. $\mathbf{\Sigma} = \mathbf{I} \in \mathbb{R}^{p\times p}$, we generate a hierarchical model: $\theta_{jk} \neq 0 \Rightarrow \beta_j \neq 0$ and $\beta_k \neq 0$. We first sample the main effects and then pick interaction effects uniformly from the pairs of main effects.

2. $\mathbf{\Sigma} = \mathbf{I} \in \mathbb{R}^{p\times p}$, we generate a strictly non-hierarchical model: $\theta_{jk} \neq 0 \Rightarrow \beta_j = 0$ and $\beta_k = 0$. We first sample the main effects and then pick interaction effects uniformly from all pairs excluding main effects as coordinates.

3. We repeat the setting 2 with a dataset that contains strong correlations. We create a dependence structure in $\mathbf{X}$, by first generating a DAG with on average 10 edges per node. Each node is sampled so that it is a linear function of its parents plus some independent centred Gaussian noise, with a variance of 10% the variance coming from the direct parents. The resulting correlation matrix then unveils for each variable $\mathbf{X}_j$ a substantial number of variables strongly correlated to $\mathbf{X}_j$ (There is usually around 10 variables with a correlation of above 0.9). Such a correlation structure will make it easier to detect pairs of variables whose product can serve as strong predictor of $\mathbf{Y}$, even though it has not been included in the construction of $\mathbf{Y}$.

We run three different procedures to estimate the main and interaction effects.

- **Two-stage Lasso:** We fit the Lasso to the data, and then run the Lasso once more on an augmented design matrix containing interactions between all selected main effects. Complexity analysis of the LAR algorithm [Efron et al., 2004] suggests the computational cost would be $\mathcal{O}(np\min(n, p))$, making the procedure very efficient. However, as the results show, it struggles in situations such as that given by model 2, where a main effects regression will fail to select variables involved in strong interactions.

- **Lasso with all interactions:** Building the full interaction matrix and computing the standard Lasso on this augmented data matrix. Analysis of the LAR algorithm would suggest the computational complexity would be in the order $\mathcal{O}(np^2\min(n, p^2))$. Nevertheless, for small p, this approach is feasible.

- **xyz:** This is Algorithm 3. The user has to set the number of projections L, conducted in the *xyz* algorithm. The run time is roughly $\min(n,p)C(M,L)$, since the runtime to fit a sparse linear model is given by $np\min(n,p)$ and each interaction search incurs a cost of $C(M,L)$. One is usually only interested in the strong interaction effects, therefore a small L will suffice (we choose $L=\sqrt{p}$). Additionally probabilistic guarantees such as in section 5.3 assure the user that all strong interactions have been found with high probability.

The experiment shows that *xyz* enjoys the favourable properties of both its competitors: it is as fast as the two-stage Lasso that gives an almost linear runtime in p, and it is about as accurate as the estimator calculated from screening all pairs (brute-force).

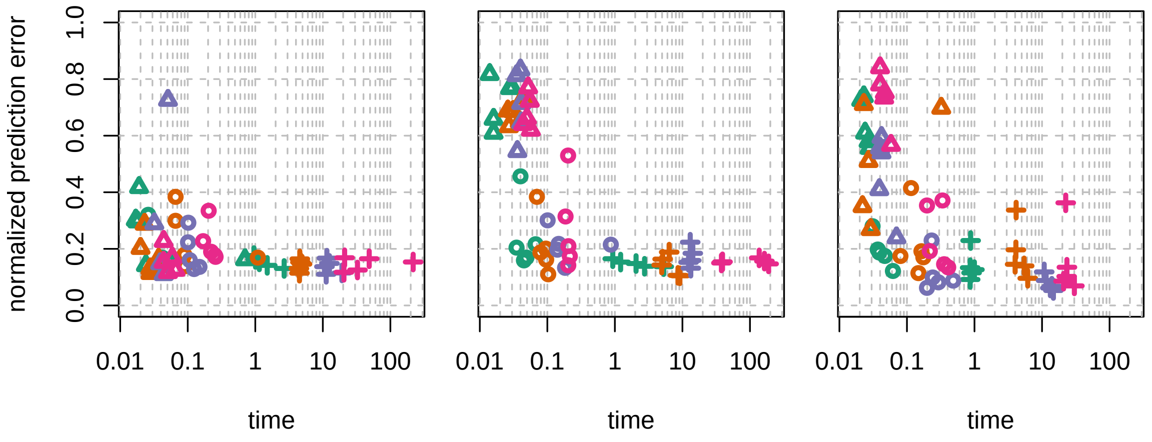

Figure 5: Normalised ℓ_2 prediction error as a function of time in seconds. Triangle: Two-stage Lasso. Circle: *xyz*-regression. Cross: Brute-force. The different colours correspond to different values of p: green $p=250$ up to pink $p=1000$. The left panel shows the results on setting 1, center panel shows setting 2 and right panel setting 3.

6 Discussion

We proposed the *xyz* algorithm for efficient search of product interactions. It is based on the translation of the interaction search to the closest pair search and the relation of the latter to sorting in one dimension. The connection between high dimensional closest pair search and its efficient one-dimensional equivalent is established through random projections.

To demonstrate the efficiency of this approach we gave a full runtime analysis in the case of binary data and showed that it enjoys subquadratic runtime. Our results hand the user an optimal way of choosing the parameters of the *xyz* algorithm to minimise its runtime.

We extended the *xyz* algorithm to continuous data and show how it can be built into the Lasso to give an efficient algorithm that can fit high dimensional models with both main and interaction effects.

We demonstrate the correctness of our results in an experimental section and use *xyz* on both real and simulated data to uncover interactions in subquadratic time.

7 Appendix

Here we include proofs that were omitted earlier.

7.1 Proof of Theorem 1

In the following, we fix the following notation for convenience:

$$\begin{aligned}\Psi &= \Xi_{\text{minimal}}, &\quad \Psi(\eta) &= \Xi_{\text{minimal}}(\eta),\\ \Xi &= \Xi_{\text{subsample}}, &\quad \Xi(\eta) &= \Xi_{\text{subsample}}(\eta).\end{aligned}$$

Note that both $\Psi(\eta)$ and $\Xi(\eta)$ depend on F though this is suppressed in the notation. Also define $\Xi_{\text{all}} = \Xi \cup \Xi_{\text{dense}}$ and $\Xi_{\text{all}}(\eta) = \Xi(\eta) \cup \Xi_{\text{dense}}(\eta)$. We will reference the parameters levels contained in $\xi \in \Xi_{\text{all}}$ as ξ_L and ξ_τ. If $\xi \in \Xi$ then we will write ξ_M for the distribution of the subsample sise M.

If we let V denote the complexity of the search for τ-close pairs, similarly to (10) we have that

$$T(\xi) = c_1 np + L(c_2 \mathbb{E}_\xi M p + \mathbb{E}_\xi V + c_3 n \mathbb{E}_\xi C_1), \tag{17}$$

where c_1, c_2, c_3 are constants. Suppose $\psi \in \Psi$ and $\xi \in \Xi$ have $\mathbb{E}_\xi C_1 = \mathbb{E}_\psi C_1$. Then since searching for τ-close pairs is at least as computationally difficult as finding equal pairs we know that $\mathbb{E}_\xi V \geq \mathbb{E}_\psi V$.

Similarly for $\xi \in \Xi_{\text{dense}}$ we have

$$T(\xi) = c_1 np + L(c_2 np + \mathbb{E}_\xi V + c_3 n \mathbb{E}_\xi C_1). \tag{18}$$

For $\xi \in \Xi_{\text{all}}$, define

$$\alpha(\xi) = \mathbb{E}_\xi C_1 / p^2, \qquad \beta(\xi) = \mathbb{P}_\xi((j^*, k^*) \in I_1)$$

where I_1 is the set of candidate interactions I when $L = 1$. Note that

$$\mathbb{P}_\xi((j^*, k^*) \in I) = 1 - \{1 - \beta(\xi)\}^{\xi_L}.$$

Thus any $\xi \in \Xi_{\text{all}}(\eta)$ with $T(\xi)$ minimal must have ξ_L as the smallest L such that $1 - \{1 - \beta(\xi)\}^{\xi_L} \geq \eta$, whence

$$\xi_L = \lceil \log(1-\eta) / \log\{1 - \beta(\xi)\} \rceil . \tag{19}$$

Note that $\beta(\xi)$ does not depend on ξ_L, so the above equation completely determines the optimal choice of L once other parameters have been fixed. We will therefore henceforth assume that L has been chosen this way so that the discovery probability of all the algorithms is at least η.

The proofs of (8) and (9) are contained in Lemmas 7 and 8 respectively. The proof of (7) is more involved and proceeds by establishing a Neyman–Pearson type lemma (Lemmas 5 and 6) showing that given a constraint on the 'size' α that is sufficiently small, minimal subsampling enjoys maximal 'power' β. To complete the argument, we show that any sequence of algorithms with size α remaining constant as $p \to \infty$ cannot have a subquadratic complexity, whilst Lemma 7 attests that in contrast minimal subsampling does have subquadratic complexity under the assumptions of the theorem. Several auxiliary technical lemmas are collected in Section 7.4

Our proofs Lemmas 5 and 6 make use of the following bound on a quantity related to the ratio of the size to the power of minimal subsampling.

Lemma 4. *Suppose $\psi \in \Psi$ has distribution for M placing mass on M and $M+1$. Under the assumptions of Theorem 1,*

$$\frac{\alpha(\psi)}{\gamma_1^M} \leq \frac{2}{1-\rho}\frac{1}{M+1}.$$

Proof. We have

$$\frac{\alpha(\psi)}{\gamma_1^M} \leq \frac{1}{p^2}\sum_{j,k}(\gamma_{jk}/\gamma_1)^M \leq \frac{c_0}{p} + \sum_{i=0}^{n\gamma_1-1}\left(\frac{i}{n\gamma_1}\right)^M f_n(i/n).$$

Now the sum on the RHS is maximised over f_n obeying constraints (A1) and (A2) in the following way. If $\rho\gamma_1 n > \gamma_1 n - 1$ then f_n places all available mass on $\gamma_1 - 1/n$. Otherwise f_n should be as close to constant as possible on $\lceil \rho\gamma_1 n \rceil/n, \ldots, (\gamma_1 n - 1)/n$, and zero below $\lceil \rho\gamma_1 n \rceil/n$. In both cases it can be seen that

$$\sum_{i=0}^{n\gamma_1-1}\left(\frac{i}{n\gamma_1}\right)^M f_n(i/n) \leq \frac{2}{1-\rho}\int_{(1+\rho)/2}^{1} x^M dx \leq \frac{2}{1-\rho}\frac{1}{M+1}. \qquad \square$$

The following Neyman–Pearson-type lemma considers only non-randomised algorithms in Ξ. In Lemma 6 we extend this result to randomised algorithms.

Lemma 5. *Let Ξ_0 be the set of $\xi \in \Xi$ such that ξ_M places mass only on a single M, so the subsample size is not randomised. There exists an α_0 independent of n such that for all $\alpha' \leq \alpha_0$, we have*

$$\sup_{\psi \in \Psi : \alpha(\psi) \leq \alpha'} \beta(\psi) = \sup_{\xi \in \Xi_0 : \alpha(\xi) \leq \alpha'} \beta(\xi).$$

Moreover the suprema are achieved.

Proof. Each $\xi \in \Xi_0$ is parametrised by its close pairs threshold τ and subsample size M. Given a $\xi \in \Xi_0$ with parameter values τ and M we compute $\alpha(\xi)$ as follows. Note that by replacing the threshold τ by $\tau/2$, we may assume that $\mathbf{X}$ and $\mathbf{Z}$ have entries in $\{-1/2, 1/2\}$.

Thus $\mathbf{X}_j - \mathbf{Z}_k$ has components in $\{-1, 0, 1\}$. Let J_{jk} be the number of non-zero components of $(X_{i_m j} - Z_{i_m k})_{m=1}^M$. Then $J_{jk} \sim \text{Binom}(M, 1 - \gamma_{jk})$. Thus

$$\mathbb{P}\biggl(\biggl|\sum_{m=1}^M D_m(X_{i_m j} - Z_{i_m k})\biggr| \le \tau\biggr) = \mathbb{P}(J_{jk} = 0) + \sum_{r=1}^M \mathbb{P}\biggl(\biggl|\sum_{m=1}^r D_m\biggr| \le \tau\biggr)\mathbb{P}(J_{jk} = r),$$

noting that $D_m \overset{d}{=} -D_m$. By Lemma 9 we know there exists an $a > 0$ such that for all $\tau \le a\sqrt{M}$ the RHS is bounded below by

$$\gamma_{jk}^M + \sum_{r=r_0}^M \frac{c_1 \tau}{\sqrt{r}} \binom{M}{r} \gamma_{jk}^{M-r}(1 - \gamma_{jk})^r \tag{20}$$

for M sufficiently large. Here the constants $a, c_1 > 0$ and $r_0 \in \mathbb{N}$ depend only on F.

Consider $\tau > a\sqrt{M}$. In this case, for $r \le M$ sufficiently large we have by Lemma 9

$$\mathbb{P}\biggl(\biggl|\sum_{m=1}^r D_m\biggr| \le \tau\biggr) \ge \mathbb{P}\biggl(\biggl|\sum_{m=1}^r D_m\biggr| \le a\sqrt{r}\biggr) \ge c_1 a.$$

However then for M sufficiently large,

$$\mathbb{P}(J_{jk} = 0) + \sum_{r=1}^M \mathbb{P}\biggl(\biggl|\sum_{m=1}^r D_m\biggr| \le \tau\biggr)\mathbb{P}(J_{jk} = r) \ge c_1 a/2,$$

so $\alpha(\xi) \ge c_1 a/2$. Note also that we must have $\alpha_0 \ge \alpha(\xi) \ge \gamma_l^M$, so $M \ge \log(\alpha_0)/\log(\gamma_l)$. Thus by choosing $0 < \alpha_0 < c_1 a/2$ sufficiently small, we can rule out $\tau > a\sqrt{M}$ and so we henceforth assume that $\tau \le a\sqrt{M}$, and that M is sufficiently large such that (20) holds for all (j, k).

We have

$$\alpha(\xi) \ge \frac{1}{p^2} \sum_{j,k} \biggl\{\gamma_{jk}^M + \tau \sum_{r=r_0}^M \frac{c_1}{\sqrt{r}} \binom{M}{r} \gamma_{jk}^{M-r}(1 - \gamma_{jk})^r\biggr\}. \tag{21}$$

Similarly we have

$$\beta(\xi) \le \gamma_1^M + \tau \sum_{r=1}^M \frac{c_2}{\sqrt{r}} \binom{M}{r} \gamma_1^{M-r}(1 - \gamma_1)^r. \tag{22}$$

Now substituting the upper bound on τ implied by (21) into (22), we get

$$\beta(\xi) \le \gamma_1^M + Q_M\biggl(\alpha(\xi) - \frac{1}{p^2}\sum_{j,k} \gamma_{jk}^M\biggr)$$

where

$$Q_M = \frac{c_2 \sum_{r=1}^{M} r^{-1/2}\binom{M}{r}\gamma_1^{M-r}(1-\gamma_1)^r}{c_1 p^{-2}\sum_{j,k}\sum_{r=r_0} r^{-1/2}\binom{M}{r}\gamma_{jk}^{M-r}(1-\gamma_{jk})^r}.$$

Now by Lemma 10, for M sufficiently large and some constant Q we have

$$Q_M \leq Q\frac{\sqrt{1-\gamma_1}}{\sum_{j,k}\sqrt{1-\gamma_{jk}}/p^2} \leq Q.$$

Thus

$$\beta(\xi) \leq \gamma_1^M + Q\biggl(\alpha(\xi) - \frac{1}{p^2}\sum_{j,k}\gamma_{jk}^M\biggr) \tag{23}$$

for all M sufficiently large. Now given α_0, let M_0 be such that

$$\frac{1}{p^2}\sum_{j,k}\gamma_{jk}^{M_0} \geq \alpha_0 \geq \frac{1}{p^2}\sum_{j,k}\gamma_{jk}^{M_0+1}.$$

Consider the minimal subsampling algorithm ψ that chooses subsample size as either M_0 or M_0+1 with probabilities b and $1-b$ such that

$$\alpha(\psi) = \frac{1}{p^2}\sum_{j,k}\{b\gamma_{jk}^{M_0} + (1-b)\gamma_{jk}^{M_0+1}\} = \alpha_0.$$

Then we have $\beta(\psi) = b\gamma_1^{M_0} + (1-b)\gamma_1^{M_0+1}$. Now suppose $\xi \in \Xi_0$ has $\alpha(\xi) \leq \alpha_0$. Then in particular $M \geq M_0+1$. We first examine the case where $M = M_0+1$. Then

$$\begin{aligned}\frac{1}{\gamma_1^{M_0}}\{\beta(\psi)-\beta(\xi)\} &\geq b + (1-b)\gamma_1 - \gamma_1 - \frac{Q}{\gamma_1^{M_0}}\biggl(\alpha_0 - \frac{1}{p^2}\sum_{j,k}\gamma_{j,k}^{M_0+1}\biggr)\\ &= b + (1-b)\gamma_1 - \gamma_1 - \frac{aQ}{\gamma_1^{M_0}}\frac{1}{p^2}\sum_{j,k}(\gamma_{j,k}^{M_0} - \gamma_{j,k}^{M_0+1})\\ &\geq b\biggl((1-\gamma_u) - \frac{2Q}{1-\rho}\frac{1}{M_0+1}\biggr),\end{aligned}$$

using Lemma 4 in the final line. Note this is non-negative for M_0 sufficiently large. When $M \geq M_0+2$ we instead have

$$\frac{\beta(\xi)}{\beta(\psi)} \leq \frac{\beta(\xi)}{\gamma_1^{M_0+1}} \leq \gamma_1 + \frac{2Q}{\gamma_1(1-\rho)}\frac{1}{M_0+1} \leq \gamma_u + \frac{2Q}{\gamma_l(1-\rho)}\frac{1}{M_0+1} < 1$$

for M_0 sufficiently large. Recall that by making α_0 sufficiently small, we can force M_0 to be arbitrarily large. Thus the result is proved. □

Lemma 6. *There exists an α_0 independent of n such that for all $\alpha' \leq \alpha_0$, we have*

$$\sup_{\psi \in \Psi : \alpha(\psi) \leq \alpha'} \beta(\psi) = \sup_{\xi \in \Xi : \alpha(\xi) \leq \alpha'} \beta(\xi).$$

Moreover the suprema are achieved.

Proof. With a slight abuse of notation, write $\xi(M', \tau')$ for the element of $\xi \in \Xi$ that fixes $M = M'$ and $\tau = \tau'$. Using the notation of Lemma 5, define function $f : [0, 1] \to [0, 1]$ by

$$f(\alpha') = \sup_{\xi \in \Xi_0 :\, \alpha(\xi) \leq \alpha'} \beta(\xi).$$

Note that for $\xi \in \Xi$ we have

$$\beta(\xi) \leq \mathbb{E}_{M \sim \xi_M} f[\alpha\{\xi(M, \xi_\tau)\}]. \tag{24}$$

Now by Lemma 5 we know there exists α_0 (depending on F) such that on $[0, \alpha_0]$, f is the linear interpolation of points

$$\left(\frac{1}{p^2} \sum_{j,k} \gamma_{j,k}^M,\, \gamma_1^M \right)_{M=1}^{\infty}.$$

We claim that f is concave on $[0, \alpha_0]$. Indeed, it suffices to show that the slopes of the successive linear interpolants are decreasing in this region, or equivalently that their reciprocals are increasing. We have

$$\frac{1}{p^2} \sum_{j,k} \frac{\gamma_{jk}^{M+1} - \gamma_{jk}^M}{\gamma_1^{M+1} - \gamma_1^M} = \frac{1}{p^2} \sum_{j,k} \left(\frac{\gamma_{j,k}}{\gamma_1} \right)^M \frac{\gamma_{jk} - 1}{\gamma_1 - 1} \tag{25}$$

which increases as M decreases, thus proving the claim.

Note also that the RHS of (25) is at most $\alpha(\psi)/\{(1 - \gamma_u)\gamma_1^M\}$ when ψ has subsample size fixed at M. Thus by Lemma 4 we see the derivatives of the linear interpolants approach infinity as they get closer to the origin. This implies the existence of an $0 < \alpha_1 < \alpha_0$ such that $-\sup\big(\partial(-f)(\alpha_1)\big) \geq \{1 - f(\alpha_1)\}/(\alpha_0 - \alpha_1)$, where $\partial(-f)(\alpha_1)$ denotes the subdifferential of the function $-f$ at α_1. We may therefore invoke Lemma 11 to conclude that for ξ with $\alpha(\xi) \leq \alpha_1$

$$\mathbb{E}_{M \sim \xi_M} f[\alpha\{\xi(M, \xi_\tau)\}] \leq f[\mathbb{E}_{M \sim \xi_M} \alpha\{\xi(M, \xi_\tau)\}] = f(\alpha(\xi)) \leq f(\alpha_1) = \max_{\psi \in \Psi : \alpha(\psi) \leq \alpha_1} \beta(\psi).$$

Combining with (24) gives the result. $\square$

The next lemma establishes subquadratic complexity of minimal subsampling.

Lemma 7. *Under the assumptions of Theorem 1, we have* $\inf_{\psi \in \Psi(\eta)} T(\psi)/(np^2) \to 0$.

Proof. Let $\psi \in \Psi$ be such that ψ_M places all mass on M. We have that $\beta(\psi) = \gamma_1^M$. Thus using the inequality $-x \leq \log(1-x)$ for $x \in (0,1)$, we have

$$\psi_L \leq -\gamma_1^{-M} \log(1-\eta).$$

Lemma 4 gives an upper bound on $\psi_L \mathbb{E}_\psi C_1$. Note that $\mathbb{E}_\psi V = \mathcal{O}(p \log(p))$. Thus ignoring constant factors, we have

$$T(\psi)/(np^2) \leq \frac{M + \log(p)}{\gamma_1^M np} + \frac{1}{M+1}.$$

Taking $M = \lfloor \log(1/\sqrt{p})/\log(\gamma_1) \rfloor$ then ensures $T(\psi)/(np^2) \to 0$. $\square$

Lemma 8. *Let $\xi \in \Xi_{dense}$. There exists $c > 0$ and $n_0 \in \mathbb{N}$ such that for all $n \geq n_0$,*

$$\inf_{\xi \in \Xi_{dense}} T(\xi)/(np^2) > c.$$

Proof. Each $\xi \in \Xi_{\text{dense}}$ is parametrised by its close pairs threshold τ. Given a $\xi \in \Xi_{\text{dense}}(F)$ with close pairs threshold τ we compute $\alpha(\xi)$ as follows. Similarly to Lemma 5 we may assume without loss of generality that $\mathbf{X}$ and $\mathbf{Z}$ have entries in $\{-1/2, 1/2\}$ so $\mathbf{X}_j - \mathbf{Z}_k$ has components in $\{-1, 0, 1\}$. Since $R_i \overset{d}{=} -R_i$ as $F \in \mathcal{F}$, we have

$$\mathbb{P}\left(\left| \sum_{i=1}^n R_i (X_{ij} - Z_{ik}) \right| \leq \tau \right) = \mathbb{P}\left(\left| \sum_{i=1}^{n(1-\gamma_{jk})} R_i \right| \leq \tau \right).$$

We now use Lemma 9. For $n(1-\gamma_u)$ sufficiently large, when $\tau \leq a\sqrt{n}$ the RHS is bounded below by

$$\frac{c_1 \tau}{\sqrt{n(1-\gamma_{jk})}}.$$

Here constant $a, c_1 > 0$ also depend only on F. Thus

$$\alpha(\xi) \geq \frac{1}{p^2} \sum_{j,k} \frac{c_1 \tau}{\sqrt{n(1-\gamma_{jk})}} \geq c_1 \tau / \sqrt{n}. \tag{26}$$

Similarly we have

$$\beta(\xi) \leq \frac{c_2 \tau}{\sqrt{n(1-\gamma_1)}}. \tag{27}$$

Note that from (26), when $\tau > a\sqrt{n}$ we have $\alpha(\xi) \geq c_1 a$. Thus from (18) we know there exists n_0 such that for all $n \geq n_0$, we have

$$\inf_{\xi \in \Xi_{\text{dense}}(\eta) : \xi_\tau > a\sqrt{n}} T(\xi)/(np^2) \geq \inf_{\xi \in \Xi_{\text{dense}}(\eta) : \xi_\tau > a\sqrt{n}} \xi_L \alpha(\xi) \geq \xi_L c_1 a > 0. \tag{28}$$

We therefore need only consider the case where $\tau \leq a\sqrt{n}$ and where $\alpha(\xi) \to 0$.

Substituting the upper bound on τ implied by (26) into (27), we get

$$\beta(\xi) \leq \alpha(\xi) \frac{c_2}{c_1\sqrt{1-\gamma_u}}.$$

Note that then

$$\xi_L \geq \frac{\log(1-\eta)}{\log\{1-\alpha(\xi)c_2/(c_1\sqrt{1-\gamma_u})\}} \geq c_3 \frac{\log\bigl(1/1-\eta\bigr)}{\alpha(\xi)}$$

for some $c_3 > 0$ provided $\alpha(\xi) < 1/2$ say. However this gives us

$$\inf_{\xi\in\Xi_{\text{dense}}(\eta):\xi_\tau\leq a\sqrt{n}} T(\xi)/(np^2) \geq \inf_{\xi\in\Xi_{\text{dense}}(\eta):\xi_\tau\leq a\sqrt{n}} \xi_L\alpha(\xi) \geq \min\{1/2, c_3\log\bigl(1/1-\eta\bigr)\} > 0.$$

Combined with (28) this give the result. $\square$

With the previous lemmas in place, we are in a position to prove (7) of Theorem 1.

Proof of Theorem 1

The proofs of (8) and (9) are contained in Lemmas 7 and 8 respectively. To show (7) we argue as follows. Given F and η, suppose for contradiction that there exists a sequence $\xi^{(1)}, \xi^{(2)}, \ldots$ and $n_1 < n_2 < \cdots$ such that (making the dependence on n of the computational time explicit)

$$\inf_{\psi\in\Psi(\eta)} T^{(n_k)}(\psi) > T^{(n_k)}(\xi^{(k)})$$

for all k. By Lemma 7, we must have $T^{(n_k)}(\xi^{(k)})/(np^2) \to 0$. This implies that $\alpha(\xi^{(k)}) \to 0$. By Lemma 6, we know that for k sufficiently large

$$\sup_{\psi\in\Psi:\alpha(\psi)=\alpha(\xi^{(k)})} \beta(\psi) \geq \beta(\xi^{(k)}).$$

Let $\psi^{(k)}$ be the maximiser of the LHS. In order for $T^{(n_k)}(\psi^{(k)}) > T^{(n_k)}(\xi^{(k)})$, it must be the case that $\mathbb{E}_{M\sim\psi_M^{(k)}} M > \mathbb{E}_{M\sim\xi_M^{(k)}} M$. However we claim that $\xi = \psi^{(k)}$ minimises $\mathbb{E}_{M\sim\xi_M} M$ among all $\xi \in \Xi$ with $\alpha(\xi) \leq \alpha(\xi^{(k)}) =: \alpha_0$, which gives a contradiction and completes the proof. Let f be the function that linearly interpolates the points

$$\left(\frac{1}{p^2}\sum_{j,k}\gamma_{j,k}^M,\ M\right)_{M=1}^{\infty}.$$

Note that f is decreasing. By considering the inverse of f it is clear that f is convex. With a slight abuse of notation, write $\xi(M,\tau)$ for the element of $\xi \in \Xi$ such that ξ_M places all mass on M and $\xi_\tau = \tau$. Note that

$$\mathbb{E}_{M\sim\xi_M} M = \mathbb{E}_{M\sim\xi_M} f[\alpha\{\xi(M,0)\}] \geq \mathbb{E}_{M\sim\xi_M} f[\alpha\{\xi(M,\xi_\tau)\}].$$

Now suppose ξ has $\alpha(\xi) \leq \alpha_0$. Then from the above and Jensen's inequality,

$$\mathbb{E}_{M\sim\xi_M} M \geq f\bigl(\mathbb{E}_{M\sim\xi_M}\alpha(\xi(M,\xi_\tau))\bigr) \geq f(\alpha_0) = \mathbb{E}_{M\sim\psi_M^{(k)}} M. \quad \square$$

7.2 Proof of Theorem 2

First note that from (11) we have $L \leq \log(1-\eta')/\log(1-\gamma^M)+1$. Then using the inequality $\log(1-x) \leq -x$ for $x \in (0,1)$, we have

$$L \leq \frac{\log\{1/(1-\eta')\}+1}{\gamma^M}.$$

Note that from the definition of γ_0 we have $\gamma^{-M} = p^{\log(\gamma)/\log(\gamma_0)}$. We then see that

$$\begin{aligned}
\gamma^{-M}\mathbb{E}(C_1) &= \gamma^{-M}\sum_{j,k}\gamma_{jk}^M \\
&\leq \gamma^{-M}\Big(\sum_{j,k:\gamma_{jk}>\gamma}\gamma_{jk}^M + \sum_{j,k:\gamma_0<\gamma_{jk}\leq\gamma}\gamma_{jk}^M + \sum_{j,k:\gamma_{jk}\leq\gamma_0}\gamma_{jk}^M\Big) \\
&\leq c_1 p\gamma^{-M} + c_2 p^{1+\log(\gamma)/\log(\gamma_0)} + p^2\gamma_0^M\gamma^{-M} \\
&\leq (c_1+c_2+1)p^{1+\log(\gamma)/\log(\gamma_0)}.
\end{aligned}$$

Collecting together the terms in (10) we have

$$C(M,L) \leq np + [\log\{1/(1-\eta')\}+1][\log(p)\{1+1/\log(\gamma_0^{-1})\} + n(c_1+c_2+1)]p^{1+\log(\gamma)/\log(\gamma_0)}$$

from which the result easily follows.

7.3 Proof of Proposition 3

Let $\eta^* = \eta(M^*, L)$. Note that in order for $\eta(M', L') \geq \eta^*$ it must be the case that $L' \geq \log(1-\eta^*)/\log(1-\gamma^{M'})$. Therefore

$$\begin{aligned}
C(M',L') - np &\geq \frac{\log(1-\eta^*)}{\log(1-\gamma^{M'})}\Big(M'p + p\log(p) + n\sum_{j,k}\gamma_{jk}^{M'}\Big) \\
&\geq \min_{M\in\mathbb{N}}\frac{\log(1-\eta^*)}{\log(1-\gamma^{M})}\Big(Mp + p\log(p) + n\sum_{j,k}\gamma_{jk}^{M}\Big) \qquad (29) \\
&= \frac{\log(1-\eta^*)}{\log(1-\gamma^{M^*})}\Big(M^*p + p\log(p) + n\sum_{j,k}\gamma_{jk}^{M^*}\Big) = C(M^*, L).
\end{aligned}$$

Moreover, the inequality leading to (29) is strict if M^* is the unique minimiser and $M' \neq M^*$.

7.4 Technical lemmas

Lemma 9. *Let $F \in \mathcal{F}$ and suppose $(R_i)_{i=1}^\infty$ is an i.i.d. sequence with $R_i \sim F$.*

Then for all $a > 0$, there exists $c_1, c_2 > 0$ and $l_0 \in \mathbb{N}$ such that for all $l \geq l_0$ and $0 \leq \tau \leq a\sqrt{l}$ we have

$$\frac{c_1\tau}{\sqrt{l}} \leq \mathbb{P}\Big(\big|\sum_{i=1}^l R_i\big| \leq \tau\Big) \leq \frac{c_2\tau}{\sqrt{l}}.$$

Proof. Let f_l be the density of $\sum_{i=1}^{l} R_i/\sqrt{l}$. Note that as $\mathbb{E}(|R_1|^3) < \infty$, we must have $\mathbb{E}(R_1^2) < \infty$, so we may assume without loss of generality that $\mathbb{E}(R_1^2) = 1$. Then by Theorem 3 of Petrov [1964] we have that for sufficiently large l,

$$|f_l(t) - \phi(t)| \leq \frac{c}{\sqrt{l}(1 + |t|^3)}. \tag{30}$$

Here c is a constant and $\phi(t) = e^{-t^2/2}/\sqrt{2\pi}$ is the standard normal density. Now by the mean value theorem, we have

$$2 \inf_{0 \leq t \leq \tau/\sqrt{l}} \{f_l(t)\} \frac{\tau}{\sqrt{l}} \leq \mathbb{P}\Big(\big|\sum_{i=1}^{l} R_i\big|/\sqrt{l} \leq \tau/\sqrt{l}\Big) \leq 2 \sup_{0 \leq t \leq \tau/\sqrt{l}} \{f_l(t)\} \frac{\tau}{\sqrt{l}}.$$

Thus from (30), for l sufficiently large we have

$$\mathbb{P}\Big(\big|\sum_{i=1}^{l} R_i\big| \leq \tau\Big) \geq \frac{\tau}{\sqrt{l}} \Big(\frac{\sqrt{2}}{\sqrt{\pi}} \exp\{-\tau^2/(2l)\} - \frac{2c}{\sqrt{l}} \Big).$$

Note that for $a > 0$ and l sufficiently large we have $\sqrt{2/\pi} e^{-a^2/2} > 2c/\sqrt{l}$, whence

$$\mathbb{P}\Big(\big|\sum_{i=1}^{l} R_i\big| \leq \tau\Big) \geq \frac{c_1 \tau}{\sqrt{l}}$$

for $0 \leq \tau \leq a\sqrt{l}$, some $c_1 > 0$. A similar argument yields the upper bound in the final result. □

Lemma 10. *Suppose $\gamma \in [0, 1)$. For all $M \in \mathbb{N}$ we have*

$$\sum_{r=1}^{M} \frac{1}{\sqrt{r}} \binom{M}{r} (1-\gamma)^r \gamma^{M-r} \leq \frac{\sqrt{2}}{\sqrt{(1-\gamma)M}}. \tag{31}$$

Given $r_0 \in \mathbb{N}$ and $\gamma \in [0, 1)$, there exists $c > 0$ and $M_0 \in \mathbb{N}$ such that for all $M \geq M_0$ we have

$$\sum_{r=r_0}^{M} \frac{1}{\sqrt{r}} \binom{M}{r} (1-\gamma)^r \gamma^{M-r} \geq \frac{c}{\sqrt{(1-\gamma)M}}. \tag{32}$$

Proof. First we show the upper bound (31). Let $J \sim \text{Binomial}(M, 1-\gamma)$.

$$\begin{aligned} \sum_{r=1}^{M} \frac{1}{\sqrt{r}} \binom{M}{r} (1-\gamma)^r \gamma^{M-r} &\leq \sqrt{2} \sum_{r=1}^{M} \frac{1}{\sqrt{r+1}} \binom{M}{r} (1-\gamma)^r \gamma^{M-r} \\ &\leq \sqrt{2}\, \mathbb{E}(1/\sqrt{J+1}). \end{aligned}$$

Next, by Jensen's inequality we have $\mathbb{E}(1/\sqrt{J+1}) \leq \sqrt{\mathbb{E}\{1/(J+1)\}}$. We now compute $\mathbb{E}\{1/(J+1)\}$ as follows.

$$\begin{aligned}
\mathbb{E}\left(\frac{1}{J+1}\right) &= \sum_{r=0}^{M} \frac{1}{r+1}\binom{M}{r}(1-\gamma)^r\gamma^{M-r} \\
&= \frac{1}{M+1}\sum_{r=0}^{M}\binom{M+1}{r+1}(1-\gamma)^r\gamma^{M-r} \\
&= \frac{1}{(1-\gamma)(M+1)}\sum_{r=0}^{M}\binom{M+1}{r+1}(1-\gamma)^{r+1}\gamma^{M-r} \\
&= \frac{1-\gamma^{M+1}}{(1-\gamma)(M+1)} \leq \frac{1}{(1-\gamma)(M+1)}.
\end{aligned}$$

Putting things together gives (31).

Turning now to (32), we see that the LHS equals

$$\mathbb{E}(1/\sqrt{J}\mathbb{1}_{\{J\geq r_0\}}) = \mathbb{E}(1/\sqrt{J}|J \geq r_0)\mathbb{P}(J \geq r_0).$$

By Jensen's inequality we have

$$\mathbb{E}(1/\sqrt{J}|J \geq r_0) \geq \frac{1}{\sqrt{\mathbb{E}(J|J \geq r_0)}} = \frac{\sqrt{\mathbb{P}(J \geq r_0)}}{\sqrt{\mathbb{E}(J\mathbb{1}_{\{J\geq r_0\}})}} \geq \frac{\sqrt{\mathbb{P}(J \geq r_0)}}{\sqrt{(1-\gamma)M}}.$$

But as $M \to \infty$, $\mathbb{P}(J \geq r_0) \to 1$, which easily gives the result. □

Lemma 11. *Let $f : [0, \infty) \to [0, 1]$ be non-decreasing. Suppose there exists $0 < \alpha_1 < \alpha_0$ such that:*

(i) f is concave on $[0, \alpha_0]$;

(ii) $-\sup\big(\partial(-f)(\alpha_1)\big) \geq \{1 - f(\alpha_1)\}/(\alpha_0 - \alpha_1)$, where $\partial(-f)(\alpha_1)$ denotes the subdifferential of the function $-f$ at α_1.

Then if random variable X has $\mathbb{E}(X) \leq \alpha_0$, then $f(\mathbb{E}X) \geq \mathbb{E}f(X)$.

Proof. Write $m = -\sup\big(\partial(-f)(\alpha_1)\big)$ Let function $g : [0, \infty) \to [0, \infty)$ be defined as follows.

$$g(x) = \begin{cases} f(x) & \text{if } 0 \leq x \leq \alpha_1 \\ f(\alpha_1) + m(x - \alpha_1) & \text{if } x > \alpha_1. \end{cases}$$

Note that g thus defined has $g(\alpha_0) \geq 1$. We see that g is convex and $g \geq f$. Thus if $\mathbb{E}(X) \leq \alpha_1$, by Jensen's inequality we have

$$f(\mathbb{E}X) = g(\mathbb{E}X) \geq \mathbb{E}g(X) \geq \mathbb{E}f(X).$$ □

7.5 Connection to LSH

Minimal subsampling as considered in Algorithm 2 is closely related to the locality-sensitive hashing (LSH) framework: Define $h(j) = \mathbf{R}^T\mathbf{X}_j$ (R corresponds to the minimal subsampling projection) to be the hashing function and $\mathcal{H}$ to be the family of such functions, from which we sample uniformly. Then $\mathcal{H}$ is $(\gamma, c\gamma, p_1, p_2)$-sensitive, i.e.:

- if $\gamma_{jk} \geq \gamma$ then $\mathbb{P}(h(j) = h(k)) \geq p_1$
- if $\gamma_{jk} \leq c\gamma$ then $\mathbb{P}(h(j) = h(k)) \leq p_2$,

where $0 < c < 1$. In the case of the minimal subsampling we have $p_1 = \gamma^M$ and $p_2 = \gamma^M c^M$. However, the typical LSH machinery cannot be applied directly to the equal pairs problem above. In our setting, we are not interested in preserving close pairs but rather the closest pairs. Theorem 1 establishes that the family $\mathcal{H}$ leads to the maximal ratio p_1/p_2 among all linear hashing families.

References

P. K. Agarwal, H. Edelsbrunner, O. Schwarzkopf, and E. Welzl. Euclidean minimum spanning trees and bichromatic closest pairs. *Discrete & Computational Geometry*, 1991.

Y. Arkin, E. Rahmani, M. E. Kleber, R. Laaksonen, W. März, and E. Halperin. Epiq: efficient detection of snp–snp epistatic interactions for quantitative traits. *Bioinformatics*, 2014.

P. Bickel, Y. Ritov, and A. Tsybakov. Hierarchical selection of variables in sparse high-dimensional regression. *IMS Collections*, 2010.

J. Bien, J. Taylor, R. Tibshirani, et al. A lasso for hierarchical interactions. *The Annals of Statistics*, 2013.

L. Breiman. Random Forests. *Machine Learning*, 2001.

L. Breiman, J. Friedman, R. Olshen, and C. Stone. *Classification and Regression Trees.* Wadsworth, Belmont, 1984.

A. M. Davie and A. J. Stothers. Improved bound for complexity of matrix multiplication. *Proceedings of the Royal Society of Edinburgh: Section A Mathematics*, 2013.

B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least Angle Regression. *Annals of Statistics*, 2004.

J. Friedman. Multivariate adaptive regression splines. *Annals of Statistics*, 1991.

J. Friedman, T. Hastie, and R. Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 2010.

N. Hao and H. H. Zhang. Interaction screening for ultrahigh-dimensional data. *Journal of the American Statistical Association*, 2014.

P. Indyk and R. Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 604–613. ACM, 1998.

Y. Kong, D. Li, Y. Fan, and J. Lv. Interaction Pursuit with Feature Screening and Selection. *arXiv preprint arXiv:1605.08933*, 2016.

F. Le Gall. Faster algorithms for rectangular matrix multiplication. In *Foundations of Computer Science (FOCS), 2012 IEEE 53rd Annual Symposium on*. IEEE, 2012.

J. Leskovec, A. Rajaraman, and J. D. Ullman. *Mining of massive datasets*. Cambridge University Press, 2014.

V. V. Petrov. On local limit theorems for sums of independent random variables. *Theory of Probability & Its Applications*, 1964. doi: 10.1137/1109044. URL `http://dx.doi.org/10.1137/1109044`.

R. Sedgewick. Algorithms in c. *Algorithms in C*, 1998.

R. D. Shah. Modelling interactions in high-dimensional data with backtracking. *arXiv preprint arXiv:1208.1174*, 2016.

R. D. Shah and N. Meinshausen. Random intersection trees. *The Journal of Machine Learning Research*, 2014.

M. I. Shamos and D. Hoey. Closest-point problems. In *Foundations of Computer Science, 1975., 16th Annual Symposium on*. IEEE, 1975.

V. Strassen. Gaussian elimination is not optimal. *Numerische Mathematik*, 1969.

R. Tibshirani. Regression Shrinkage and Selection via the Lasso. *Journal of the Royal Statistical Society, Series B*, 1996.

V. V. Williams. Multiplying matrices faster than coppersmith-winograd. In *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*. ACM, 2012.

B. R. Winkelmann, W. März, B. O. Boehm, R. Zotz, J. Hager, P. Hellstern, and J. Senges. Rationale and design of the luric study-a resource for functional genomics, pharmacogenomics and long-term prognosis of cardiovascular disease. *Pharmacogenomics*, 2001.

J. Wu, B. Devlin, S. Ringquist, M. Trucco, and K. Roeder. Screen and clean: a tool for identifying interactions in genome-wide association studies. *Genetic Epidemiology*, 2010.

H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society, Series B*, 2005.