

Fast Feature Selection with Fairness Constraints

Francesco Quinzan¹, Rajiv Khanna², Moshik Hershcovitch³,
Sarel Cohen⁴, Daniel G. Waddington³, Tobias Friedrich⁴, Michael W. Mahoney⁵

¹KTH Royal Institute of Technology

²Purdue University

³IBM Research

⁴Hasso Plattner Institute

⁵ICSI and UC Berkeley

Abstract

We study the fundamental problem of selecting *optimal* features for model construction. This problem is computationally challenging on large datasets, even with the use of greedy algorithm variants. To address this challenge, we extend the adaptive query model, recently proposed for the greedy forward selection for submodular functions, to the faster paradigm of Orthogonal Matching Pursuit for non-submodular functions. The proposed algorithm achieves exponentially fast parallel run time in the adaptive query model, scaling much better than prior work. Furthermore, our extension allows the use of downward-closed constraints, which can be used to encode certain fairness criteria into the feature selection process. We prove strong approximation guarantees for the algorithm based on standard assumptions. These guarantees are applicable to many parametric models, including Generalized Linear Models. Finally, we demonstrate empirically that the proposed algorithm competes favorably with state-of-the-art techniques for feature selection, on real-world and synthetic datasets.

1 Introduction

We study the fundamental problem of selecting a few features out of many for a given modeling problem, while satisfying additional side constraints. An application, we also study how this setup can be used to encode certain notions of fairness in feature selection in a principled way.¹ Formally, given a function $l : \mathbb{R}^n \rightarrow \mathbb{R}_{>0}$ expressing the goodness of fit, we search for a set of features S maximizing the function

$$f(S) := l(\beta^{(S)}) - l(\mathbf{0}). \quad (1)$$

Here, $\mathbf{0}$ represents the n -dimensional zero vector, and $\beta^{(S)}$ is a vector maximizing $l(\cdot)$ with support in S . If we denote by \mathcal{I} the set of all acceptable solution sets that satisfy the side constraints, then the feature selection optimization problem with side constraints can be formalized as

$$\arg \max_{S \subseteq [n]: S \in \mathcal{I}} f(S) = \arg \max_{S \subseteq [n]: S \in \mathcal{I}} l(\beta^{(S)}) - l(\mathbf{0}), \quad (2)$$

¹In this paper, we consider several fairness constraints proposed in the literature, but there may be other notions of fairness which do not fall within our theoretical framework.

where $[n]$ is the index set of all the features. Often, \mathcal{I} corresponds to an r -sparsity constraint, i.e., a solution \mathbf{S} is feasible if it contains at most r features. Several algorithms have been proposed for feature selection under sparsity constraints. Some examples include forward step-wise selection methods [Elenberg et al., 2018, Qian and Singer, 2019], the Orthogonal Matching Pursuit [Elenberg et al., 2018, Needell and Tropp, 2010, Sakaue, 2020], forward-backward methods [Jalali et al., 2011, Liu et al., 2014], Pareto optimization [Qian et al., 2015, 2020], Exponential Screening [Rigollet and Tsybakov, 2010], and gradient-based methods [Jain et al., 2014, Yuan et al., 2017]. The aforementioned algorithms, however, are computationally inefficient on large datasets. Furthermore, there are a limited number of studies that take into account additional side concerns (e.g. by encoding them as matroids [Chen et al., 2018, Gatmiry and Gomez-Rodriguez, 2018]), which can be crucial when deploying machine learning systems in the real world.

Recently, there has been a growing effort towards developing fair algorithms for many fundamental problems, such as regression and classification [Agarwal et al., 2018, Feldman et al., 2015, Grgic-Hlaca et al., 2018b, Kim et al., 2018, Zafar et al., 2017a], matching [Chierichetti et al., 2019], and summarization [Halabi et al., 2020]. Several definitions of fairness can be incorporated in the learning process as additional side constraints [Agarwal et al., 2018, 2019, Chierichetti et al., 2019, Donini et al., 2018, Grgic-Hlaca et al., 2016, Grgic-Hlaca et al., 2018a,b, Halabi et al., 2020, Woodworth et al., 2017, Zafar et al., 2017a]. Motivated by this line of research, we consider the following question: *How can we efficiently perform feature selection, while taking into account additional constraints such as fairness?*

We address this question by proposing a novel algorithm that combines the paradigms of matching pursuit and the adaptive query model for the problem (2). This algorithm is much faster than previously known techniques. At the same time, it allows incorporation of certain notions of fairness in the learning process, via a reduction to the p -system side constraints (see Jenkyns [1976] and Section 2.2).

We recognize that quantifying the full meaning of the notion of fairness with its societal context, as understood by human beings, may be hard to achieve through mathematical formalism alone [Selbst et al., 2019]. Indeed, p -systems may not explicate all current or future codifications of fairness. Our claim is not to *solve* the problem of fairness itself, but rather to provide a theoretically sound framework for some of its currently accepted manifestations, with the hope of serving as a blueprint for further developments along similar lines.

Optimization problems as in (2) under general side constraints are computationally challenging in practice. A major bottleneck lies in the evaluation of $\beta^{(\mathbf{S})}$ for a given set \mathbf{S} , since this operation requires to re-train the model onto every candidate set \mathbf{S} . Another challenge is enforcing the side constraint \mathcal{I} that encodes the selection criteria, except in the trivial cases where the protected classes are known a priori [Beutel et al., 2019, Lahoti et al., 2020]. Our proposed algorithm is efficient with respect to these challenges.

Our Contribution. We propose a novel matching pursuit algorithm for the constrained feature selection problem (2). This algorithm uses oracle access to the gradient $\nabla l(\beta^{(\mathbf{S})})$, and to an oracle for the evaluation of the feasibility of an input solution set. The feasibility oracle is well-aligned with previous related work [Elenberg et al., 2018, Sakaue, 2020].

Our algorithm is based on a general technique called adaptive sequencing, that was recently

proposed for submodular functions [Balkanski et al., 2019, Breuer et al., 2020]. However, previously proposed adaptive sequencing algorithms fail on problem (2), as shown in Appendix A. On the other hand, our algorithm converges after poly-logarithmic rounds in the adaptive query model. In each round, calls to the oracle functions can be performed in parallel, resulting in a dramatic speed-up.

The main technical contributions of this paper are two-fold: (i) we extend the adaptive query model to a class of non-submodular objectives using the paradigm of gradient based pursuit algorithms; (ii) we incorporate general downward-closed constraints in the optimization process beyond the standard sparsity constraints. Our main result can be stated as follows.

Theorem 1.1 (informal). *Denote with OPT the global maximum of the function $f(\cdot)$ as in (2). There exists a randomized algorithm that outputs a set of features \mathbf{S}^* such that*

$$\frac{\mathbb{E}[f(\mathbf{S}^*)]}{\text{OPT}} \geq \frac{1}{1+p} (1 - \exp\{-\alpha(1-\varepsilon)^2\}),$$

for tolerance $0 < \varepsilon < 1$. Here, p is a parameter that depends on \mathcal{I} , and α is a parameter that depends on $l(\cdot)$. For n total number of features, this algorithm uses $\mathcal{O}(\varepsilon^{-2} \log n)$ rounds of calls to $\nabla l(\beta^{(\mathbf{S})})$. Furthermore, this algorithm uses expected $\mathcal{O}(\varepsilon^{-2} \sqrt{r} \log n)$ rounds of calls to the oracle for the feasibility of an input solution set, where r is the size of the largest feasible solution.

To the best of our knowledge, our algorithm is the fastest known algorithm for the general setting of maximizing non-submodular functions with side constraints, with provable guarantees and strong empirical performance (see Section 5). Qian and Singer [2019] propose another algorithm for maximizing non-submodular functions that converges after poly-logarithmic rounds. However, their algorithm cannot handle general side constraints \mathcal{I} beyond sparsity by design. Hence, the algorithm of Qian and Singer [2019] is unsuitable for more complex applications such as learning with fairness constraints [Grgic-Hlaca et al., 2018b]. Furthermore, for the standard r -sparsity constraint, we improve upon their approximation guarantee (see Theorem 4.1).

Technical Overview. In our analysis, we face two major technical challenges. The first challenge is that of re-purposing the adaptive sequencing for functions that are not submodular, without a significant loss in the approximation guarantee. Adaptive sequencing has been so far employed only for maximizing submodular functions. Interestingly, it is known that standard adaptive sampling techniques do not guarantee constant factor approximation for functions with weak diminishing returns, which are typically invoked for feature selection theoretical studies [Elenberg et al., 2018].

The second challenge consists of integrating a constrained selection process based on orthogonal projections in the above adaptive sequencing framework. Common objective functions for feature selection do not have certain desirable properties (e.g., an antitone gradient) and the standard analysis for adaptive sequencing fails in our setting. To resolve these issues, we build upon the work of Elenberg et al. [2018] to establish a connection between gradient evaluations of functions that are restricted strong concave, and their marginal contributions to the optimization cost. This connection allows us to bound the gradient evaluations in terms of a discrete function, which is in turn used in the analysis to obtain the desired approximation guarantees. See Theorem 4.1 for the formal result and Appendix G for the proof.

2 Preliminaries

Notation. We denote with n the number of features, i.e., the dimension of the domain of $l(\cdot)$, and we define $[n] := \{1, 2, \dots, n\}$. For any $s \in [n]$, we denote with \mathbf{e}_s the unit vector, with a 1 for the coefficient indexed by s , and 0 otherwise. Feature sets are represented by sans script fonts, i.e., \mathbf{S}, \mathbf{T} . Vectors are represented by lower-case bold letters as $\mathbf{x}, \mathbf{y}, \boldsymbol{\beta}$ and matrices are represented by upper-case bold letters, i.e., $\mathbf{X}, \mathbf{Y}, \boldsymbol{\Sigma}$. For a feature set \mathbf{S} , we denote with $\boldsymbol{\beta}^{(\mathbf{S})}$ a vector maximizing $l(\cdot)$ with non-zero entries indexed by the set \mathbf{S} . For a feature set \mathbf{T} and a parameter vector $\boldsymbol{\beta}$, we define $\nabla l(\boldsymbol{\beta})_{\mathbf{T}} := \langle \nabla l(\boldsymbol{\beta}), \sum_{s \in \mathbf{T}} \mathbf{e}_s \rangle$. We denote with OPT the optimal value attained by the function $f(\cdot)$ as in (2). We denote with \mathcal{I} the p -system side constraint, and with r its rank, as defined in Section 2.1. The notation $\text{Cond}(\mathbf{T})$ denotes the set $\{s \in [n] \setminus \mathbf{T} : \mathbf{T} \cup \{s\} \in \mathcal{I}\}$.

2.1 Problem Formulation

We study optimization tasks as in the problem (2) under some additional assumptions on $l(\cdot)$, which are often satisfied in practical applications (see Appendix C). Define an r -sparse subdomain as a set of the form $\Omega_r := \{(\mathbf{x}, \mathbf{y}) \in \mathbb{R}^n \times \mathbb{R}^n : \|\mathbf{x}\|_0 \leq r, \|\mathbf{y}\|_0 \leq r, \|\mathbf{x} - \mathbf{y}\|_0 \leq r\}$. We now define the notions of Restricted Strong Concavity (RSC) and Restricted Smoothness (RSM) of a function.

Definition 2.1 (RSC, RSM [Negahban et al., 2010]). A function $l(\cdot)$ is said to be restricted strong concave (RSC) with parameter m and restricted smooth (RSM) with parameter M on a subdomain Ω_r iff, for all $(\mathbf{x}, \mathbf{y}) \in \Omega_r$ it holds that $-\frac{m}{2}\|\mathbf{y} - \mathbf{x}\|_2 \geq l(\mathbf{y}) - l(\mathbf{x}) - \langle \nabla l(\mathbf{x}), \mathbf{y} - \mathbf{x} \rangle \geq -\frac{M}{2}\|\mathbf{y} - \mathbf{x}\|_2$.

We say that l is (M, m) -(smooth, restricted concave), if it fulfills the conditions as in Definition 2.1 with parameters M and m . RSC/RSM often hold in practice, we refer the reader to Appendix C for further discussion of these properties.

We model the side constraints as p -systems. In order to give an axiomatic definition of p -systems, we introduce additional terminology. Given a collection of feasible solutions \mathcal{I} over a ground set $[n]$ and a set $\mathbf{T} \subseteq [n]$, we denote with $\mathcal{I}|_{\mathbf{T}}$, the restricted feasible solution set, as the collection consisting of all sets $\mathbf{S} \subseteq \mathbf{T}$ s.t. $\mathbf{S} \in \mathcal{I}$. We define $\text{Cond}(\mathbf{T})$ as the set of all $s \in [n] \setminus \mathbf{T}$ such that $\mathbf{T} \cup \{s\} \in \mathcal{I}$. A set \mathbf{T} is a maximal independent set if it holds that $\text{Cond}(\mathbf{T}) = \emptyset$. A base for \mathcal{I} is any maximal set $\mathbf{T} \in \mathcal{I}$.

Definition 2.2 (p -Systems [Jenkyns, 1976]). A p -system \mathcal{I} over $[n]$ is a collection of subsets of $[n]$ such that: (i) $\emptyset \in \mathcal{I}$; (ii) for any two sets $\mathbf{S} \subseteq \mathbf{T} \subseteq [n]$, if $\mathbf{T} \in \mathcal{I}$ then $\mathbf{S} \in \mathcal{I}$; (iii) for any set $\mathbf{T} \subseteq [n]$ and any bases $\mathbf{S}, \mathbf{U} \in \mathcal{I}|_{\mathbf{T}}$ it holds $|\mathbf{S}| \leq p|\mathbf{U}|$.

The second defining axiom is commonly referred to as subset-closure or downward-closed property. The rank r of a p -system \mathcal{I} is defined as the maximum cardinality of any feasible solution $\mathbf{T} \in \mathcal{I}$.

Armed with these definitions, we can re-visit the problem 2 with additional assumptions, where the set of feasible solutions \mathcal{I} is a p -system, and $l(\cdot)$ is restricted strong concave and smooth. Our problem formulation is a strict generalization of previous related works [Chierichetti et al., 2019, Elenberg et al., 2018, Sakaue, 2020] which considered the r -sparsity constraints encoded as $\mathcal{I} := \{\mathbf{T} \subseteq [n] : |\mathbf{T}| \leq r\}$ which is a special case of the more general p -system constraints.

Further, p -systems are also a strict generalization of the matroid-type constraints considered in submodular literature [Chen et al., 2018, Gatmiry and Gomez-Rodriguez, 2018].

2.2 Embedding Fairness via p -Systems

In our framework, the p -system \mathcal{I} enumerates which sets of features are considered “fair”. That is, a set of features \mathbf{S} is acceptable as fair if and only if $\mathbf{S} \in \mathcal{I}$. Our framework is very flexible, and can handle a large variety of constraints, including many constraints used to enumerate notions of fairness (see, e.g., Section 4 by Chierichetti et al. [2019] and Section 2 by Grgic-Hlaca et al. [2018b]). Non-trivial examples of p -systems side constraints are also found in the context of data summarization [Feldman et al., 2017, Mirzasoileman et al., 2016, 2018, Quinzan et al., 2021]. We now describe how some additional well-established notions of fairness can also be embedded as p -systems.

Procedural fairness metrics: Procedural fairness focuses on selecting features based on perceived notion of fairness as envisioned by human beings during the process of decision making, rather than on the fairness of the outcome. It is measured by gauging “the degree to which people consider various features to be fair” [Grgic-Hlaca et al., 2016]. This is in contrast to measuring fairness of the *outcomes* of such decisions, for example, by down weighing decisions that affect users of protected groups (e.g., race, gender).

In this work, we consider measures for procedural fairness studied by Grgic-Hlaca et al. [2016] and Grgic-Hlaca et al. [2018b]. However, our framework is not specific to these definitions. These measures consist of monotone set functions $h : 2^{[n]} \rightarrow [0, 1]$. For an input feature set $\mathbf{T} \subseteq [n]$, the value $h(\mathbf{T})$ quantifies the perceived fairness of \mathbf{T} , with $h(\mathbf{T}) = 0$ corresponding to maximum fairness and $h(\mathbf{T}) = 1$ corresponding to maximum unfairness. We can take a specific example, where $h(\cdot)$ is the *feature-apriori* unfairness Grgic-Hlaca et al. [2016]. For a given feature $s \in [n]$, denote with \mathcal{U}_s the set of users that perceive a feature to be fair. For a set of features $\mathbf{T} \subseteq [n]$, Grgic-Hlaca et al. [2016] define the feature-apriori unfairness as

$$h(\mathbf{T}) := 1 - \frac{|\bigcap_{s \in \mathbf{T}} \mathcal{U}_s|}{|\mathcal{U}|}.$$

Sets of features $\mathbf{T} \subseteq [n]$ are selected only if the value $h(\mathbf{T})$ is below a certain threshold. Given a monotone set function h as described above, we can enumerate “fair” sets of features as $\mathcal{I}_{\text{acc}}^\lambda := \{\mathbf{T} \in [n] : h(\mathbf{T}) \leq \lambda\}$. Here, $\lambda \in [0, 1]$ is a user-defined parameter, which determines the trade-off between fairness and accuracy. Since, we require $h(\cdot)$ to be monotone, the $\mathcal{I}_{\text{acc}}^\lambda$ satisfies the downward closed property and is a p -system (see Definition 2.2). The similar notions of feature-disparity fairness and feature-accuracy fairness can be embedded as p -systems in a similar fashion. While procedural fairness may not imply fairness of the outcome, it has been observed that in some cases procedurally fair feature sets maintain good outcome fairness [Grgic-Hlaca et al., 2016, Grgic-Hlaca et al., 2018b].

Feature partitions: Our proposed approach also includes as a special case the framework proposed by Celis et al. [2018]. Here, features are grouped into disjoint clusters $[n] = \mathbf{X}_1 \cup \dots \cup \mathbf{X}_\ell$. The constraints are specified using λ_j which encodes the maximum number of features that can be selected from cluster \mathbf{X}_j . In other words, features $\mathbf{T} \subseteq [n]$ is then feasible if the number of data-points intersecting a class \mathbf{X}_j does not exceed the corresponding threshold λ_j . Formally,

we define the set of constraints $\mathcal{I}_{\text{cl}}^\lambda := \{\mathbb{T} \subseteq [n]: |\mathbb{T} \cap \mathbf{X}_j| \leq \lambda_j \forall j \in [\ell]\}$. This set of constraints is a *matroid*, which is a p -system with $p = 1$.

A generalization of the aforementioned partition matroid was considered by Halabi et al. [2020]. For each element in any partition set \mathbf{X}_j , we are given a lower- and an upper-bound on the number of elements that can be selected from this set. Bounds are denoted by ℓ_j and u_j respectively. The set of constraints can be written as $\mathcal{I}_p := \{\mathbb{T} \subseteq [n]: \ell_j \leq |\mathbb{T} \cap \mathbf{X}_j| \leq c_j \forall j \in [\ell]\}$. This set of constraints is, in general, not a p -system. However, Halabi et al. [2020] show that one can consider a relaxation of the constraint set $\bar{\mathcal{I}}_p := \{\mathbb{T} \subseteq [n]: \mathbb{T} \subseteq \mathbb{S} \text{ for any set } \mathbb{S} \in \mathcal{I}_p\}$, which is equivalent from the optimization perspective. Any monotone optimization objective (as in (1)) yields the same solution sets on \mathcal{I}_p and $\bar{\mathcal{I}}_p$. The set of constraints $\bar{\mathcal{I}}_p$ is a *matroid*, i.e., a p -system with $p = 1$ [Edmonds, 1970].

2.3 The Computational Model

We assume access to an oracle that returns $\nabla l(\boldsymbol{\beta}^{(\mathbb{T})})$, for a given input set \mathbb{T} . Elenberg et al. [2018] highlight the benefits of using this oracle model for feature selection, since access to the gradient is available from the inner optimization. In the case of a linear model, the gradient $\nabla l(\boldsymbol{\beta}^{(\mathbb{T})})$ can be easily estimated for various metrics l expressing the goodness of fit. For instance, if l is the log-likelihood function, then the gradient can be computed in explicit form. For more complex models, stochastic lower-bounds of $\log \mathbb{P}_{\boldsymbol{\beta}^{(\mathbb{S})}}(\mathbf{x})$ can be used, and then differentiated [Bamler et al., 2017, Nowozin, 2018]. Similar considerations hold for other metrics, such as the R^2 objective [Elenberg et al., 2018].

We also assume access to the independence oracle of the underlying p -system \mathcal{I} . The independence oracle takes as input a set \mathbb{T} , and returns as output a Boolean value, true if $\mathbb{T} \in \mathcal{I}$ and false otherwise. This oracle is often assumed for optimizing functions over p -systems [Mirzasoleiman et al., 2016, Quinzan et al., 2021]. Our method also works assuming access to a rank oracle, or a span oracle. We refer the reader to [Chekuri and Quanrud, 2019] for a description of these oracle models.

We evaluate performance using the notion of adaptivity. The adaptivity refers to the number of sequential rounds of the algorithm, wherein polynomial number of parallel queries are made in each round [Balkanski and Singer, 2018]. Formally, given an oracle f , an algorithm is r -adaptive if every query q to the oracle f occurs at a round $i \in [r]$ such that q is independent of the answers $f(q')$ to all other queries q' at round i . This notion is closely related to the Parallel Random Access Machines (PRAM) model, as shown in Appendix D. We evaluate empirical speedup by the adaptivity of the oracle to evaluate $\nabla l(\boldsymbol{\beta}^{(\mathbb{T})})$. We also evaluate the adaptivity of the independence oracle for the p -system \mathcal{I} .

3 Algorithmic Overview

Our algorithm, which we call FASTOMP, is presented in Algorithm 1. This algorithm is based on a technique called adaptive sequencing [Balkanski et al., 2019, Breuer et al., 2020], which was recently proposed for highly scalable maximization of submodular functions.

Say \mathbf{X} is the complete set of candidate features, and \mathbb{S} is the current solution. Starting from $\mathbb{S} \leftarrow \emptyset$, the FASTOMP iteratively generates a random sequence of features $\{a_1, a_2, \dots, a_k\}$ with the RNDSEQ sub-routine (details in Appendix E), such that the set $\{a_1, a_2, \dots, a_k\} \cup \mathbb{S}$ is a

Algorithm 1 FASTOMP

$S \leftarrow \emptyset$;
while the number of iterations is less than ε^{-1} and $\text{Cond}(S) \neq \emptyset$ **do**
 $X \leftarrow \{s \in [n]: \{s\} \cup S \in \mathcal{I}\}$;
 $t \leftarrow (1 - \varepsilon) \frac{m}{|T|M} \|\nabla l(\beta^{(S)})_T\|_2^2$ with $T \subseteq X$ maximizing $\|\nabla l(\beta^{(S)})_T\|_2^2$ s.t. $|T| \leq r$;
 while $X \neq \emptyset$ and $\text{Cond}(S) \neq \emptyset$ **do**
 $\{a_1, a_2, \dots, a_k\} \leftarrow \text{RNDSEQ}(X, S)$ and define $S_j \leftarrow S \cup \{a_1, \dots, a_j\}$;
 observe $X_j \leftarrow \{s \in X: \langle \nabla l(\beta^{(S_j)}), \mathbf{e}_s \rangle^2 \geq t \text{ and } s \in \text{Cond}(S_j)\}$;
 $j^* \leftarrow \min_j \{j: |X_j| < (1 - \varepsilon) |X|\}$;
 $X \leftarrow X_{j^*}$ and $S \leftarrow S_{j^*}$;
 end
end
return S ;

maximal independent set of \mathcal{I} . After a sequence is generated, the FASTOMP identifies a prefix $\{a_1, \dots, a_{j^*}\}$ that is added to the current solution. The index j^* defining this prefix is chosen such that it holds $|X_j| \geq (1 - \varepsilon) |X|$, for all $0 \leq j < j^*$. This inequality ensures that any point added to the current solution yields $\langle \nabla l(\beta^{(S)}), \mathbf{e}_s \rangle^2 \geq t$ in expected value. Finally, the ground set X is updated as to include only those points that yield a good improvement to the new solution. The RNDSEQ sub-routine used to generate $\{a_1, a_2, \dots, a_k\}$ corresponds to Algorithm A by Karp et al. [1988]. Here, k is the size of the independent set returned in the current iteration.

Adaptive sequencing via matching projections. Our proposed algorithm differs from previous adaptive sequencing techniques, since it does not use queries to the function f as defined in (1). Instead, our algorithm uses oracle access to the function $\nabla l(\beta^{(S)})$, and features $s \in [n]$ are added to the current solution if it holds $\langle \nabla l(\beta^{(S_j)}), \mathbf{e}_s \rangle \geq t$ in expected value, with t a threshold updated during run time. As such, our approach extends the applicability of adaptive sequencing to gradient-based pursuit methods, as opposed to the standard value-oracle based methods in earlier works. Finding optimal solutions with this technique requires much less computation than the previous approach [Balkanski et al., 2019], by which points s are selected if it holds $f(S_j \cup \{s\}) \geq t$. This is because it is usually much faster to compute an inner product, than evaluating $f(\cdot)$ for every candidate S .

Implicit estimates of OPT. All algorithms based on adaptive sampling techniques proposed so far require an estimate of OPT, the value of the optimal solution set. This value is typically not known a priori. To circumvent this problem, previous algorithms perform multiple runs for various guesses of OPT [Breuer et al., 2020], or they use additional preprocessing steps [Fahrbach et al., 2019]. Our algorithm has the significant advantage that OPT is estimated implicitly. For a function $l(\cdot)$ that is (m, M) -(smooth, restricted concave), we have that $\max_{\{T: |T| \leq k\}} \|\nabla l(\beta^{(S)})_T\|_2^2 \geq 2m(\text{OPT} - f(S))$. A proof of this result is deferred to Appendix G, and it is based on Elenberg et al. [2018]. Hence, the FASTOMP estimates OPT with a single oracle valuation, and it does not require multiple runs or preprocessing steps.

Finding the set X_{j^*} . Common optimization functions $l(\cdot)$ for feature selection lack certain desirable properties, such as an antitone gradient. Hence, in contrast to the submodular case, the sequence $\{|X_j|\}_j$ as in Line 7 of Algorithm 1 is not monotonic. For this reason, it is not possible to estimate the index j^* with a binary search, as other adaptive sequencing algorithms do [Breuer et al., 2020]. We require $\mathcal{O}(r)$ phases of re-training to estimate j^* , whereas the general adaptive sequencing technique would require $\mathcal{O}(rn)$ phases of re-training [Balkanski et al., 2019], due to the different oracle model.

4 Approximation Guarantees

In this section, we study the approximation guarantees for Algorithm 1, when solving Problem (2).

Theorem 4.1. *Define the support selection function $f(\cdot)$ as in (1), for the given function $l(\cdot)$ that is (M, m) -(restricted smooth, restricted strong concave), on the sparse sub-domain Ω_{2r} . Consider a p -system \mathcal{I} of rank r over $[n]$, and let S^* be the output of Algorithm 1 while OPT is the optimum solution set for the Problem 2. Then,*

$$\frac{\mathbb{E}[f(S^*)]}{\text{OPT}} \geq \frac{1}{1+p} \left(1 - \exp \left\{ -(1-\varepsilon)^2 \frac{m^3}{M^3} \right\} \right),$$

for all $0 < \varepsilon < 1$. Furthermore, in the specific case when \mathcal{I} is r -sparsity constraint over $[n]$, then,

$$\frac{\mathbb{E}[f(S^*)]}{\text{OPT}} \geq \left(1 - \exp \left\{ -(1-\varepsilon)^2 \frac{m^2}{M^2} \right\} \right).$$

A full proof of this theorem is deferred to Appendix G. We remark that, if \mathcal{I} is a r -sparsity constraint, then the approximation guarantee of Theorem 4.1 is asymptotically better than the guarantee attained by other parallel algorithms for this problem, such as the DASH [Qian and Singer, 2019]. Specifically, as proven in Theorem 1 by Qian and Singer [2019], the DASH yields an approximation of $1 - \exp\{m^4/M^4\} - \varepsilon$ on this problem. Furthermore, the DASH cannot handle general p -system side constraints.

The parameter p in Theorem 4.1 is always upper-bounded by the maximum number of features r that we wish to select for model construction. This upper-bound still holds if one assumes additional underlying fairness constraints. Additional assumptions on the p -system may yield an improved bound on p . Finally, previous related work [Halabi et al., 2020] reduces some fairness constraints to a matroid, in which case our analysis applies with $p = 1$. We also provide bounds for the run time of the FAST_{OMP} as follows.

Theorem 4.2. *Algorithm 1 terminates after $\mathcal{O}(\varepsilon^{-2} \log n)$ rounds of calls to the oracle function, and it uses at most $\mathcal{O}(\varepsilon^{-2} r \log n)$ oracle queries. Furthermore, Algorithm 1 requires expected $\mathcal{O}(\varepsilon^{-2} \sqrt{r} \log n)$ independent calls to the oracle for the p -system \mathcal{I} , and the total expected number of calls to the oracle for the p -system \mathcal{I} is $\mathcal{O}(\varepsilon^{-2} nr \log n)$.*

The proof of Theorem 4.2 is deferred to Appendix H. The estimates on the rounds of adaptivity extends to the PRAM model. If we denote with d_l the depth required to evaluate the oracle function on a set, then the FAST_{OMP} has $\mathcal{O}(\varepsilon^{-2} d_l \log n)$ depth. Note that the rounds of independent calls to the oracle are sub-linear, but not poly-logarithmic in the problem size.

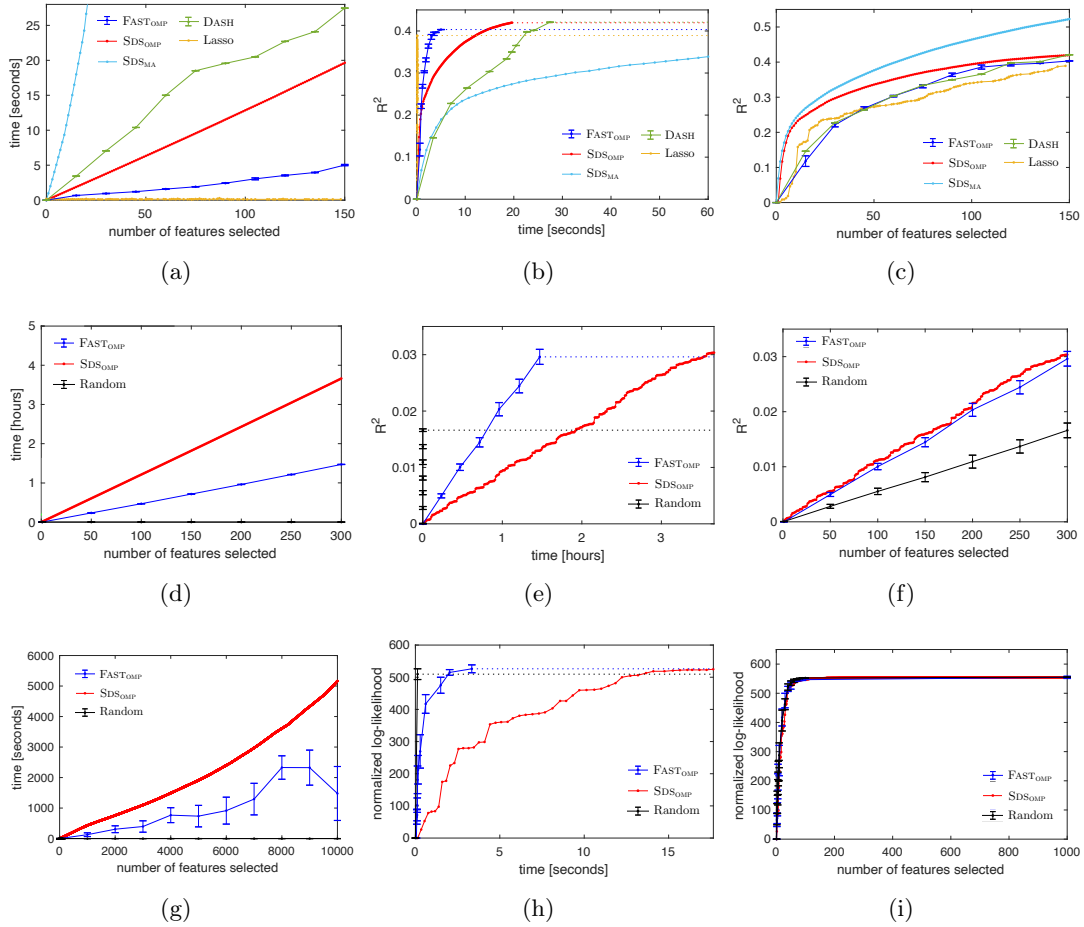


Figure 1: Results on the Synthetic Unconstrained Dataset (top row), the Synthetic Dataset with Constraints (mid row), and the Pan-Cancer Dataset (bottom row), as in Section 5.

The reason is that the RNDSEQ sub-routine requires expected $\mathcal{O}(\sqrt{n})$ rounds of independent calls to the oracle for the p -system (see Appendix E).

Some authors have proposed non-adaptive techniques for feature selection. If \mathcal{I} is an r -sparsity constraint, then Elenberg et al. [2018], Sakaue [2020] provide an algorithm that require $\mathcal{O}(1)$ sequential oracle calls. This algorithm selects r best points $s \in [n]$ independently, according to the values $f(\{s\})$, or the value of the inner product $\langle \nabla l(\mathbf{0}), \mathbf{e}_s \rangle$. However, oblivious feature selection techniques for general p -system side constraints require $\Omega(r)$ sequential calls to the independence oracle. These techniques are impractical on large dataset, when the feasibility of a solution set is computationally expensive.

5 Experiments

In this section, we present empirical evidence of the efficacy of the proposed algorithm. We provide extensive experiments on both synthetic and real world datasets. All experiments are performed on Python 3 on a server that runs Linux with Intel Xeon E5-2630 v4 with 40

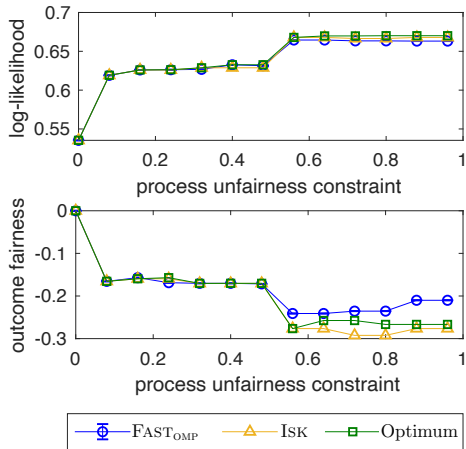


Figure 2: Experiments on the COMPAS Dataset, as detailed in Section 5. For the FAST_{OMP} we perform multiple runs and we report on the sample mean. On this dataset, the variance in accuracy and outcome fairness is negligible for FAST_{OMP}.

The CGARN Pan-Cancer Dataset. This dataset has about 2×10^4 features and 804 observations, with size $\sim 201\text{MB}$ [Cancer Genome Atlas Research Network et al., 2013]. In this dataset, the features embed information on the genome sequences of 802 patients affected with cancer, and the observations consists of a pseudo-Boolean array, with 1 if the corresponding patient has PRAD cancer and 0 otherwise.

The ProPublica COMPAS DataSet. We consider the well-known ProPublica COMPAS dataset, which is a pretrial risk assessment instrument [Larson et al., 2016]. It was constructed in 2016, using data of defendants from Broward County, FL, who had been arrested in 2013 or 2014. This dataset was intended to be used to predict if a criminal was likely to re-offend, based on previous arrest charges and demographic information. Predictions based on the COMPAS datasets were found to be racially biased [Angwin et al., 2016, Berk et al., 2021] (see Appendix J for details).

We use the COMPAS dataset to reproduce the experiments by Grgic-Hlaca et al. [2018b]. These experiments use the COMPAS dataset to predict if a defendant faces risks of recidivism, and study the trade-off between fairness and accuracy achieved by the feature-apriori accuracy, feature-accuracy fairness, and feature-disparity fairness (see Section 2.2). Our interest in using the COMPAS dataset is only to quantitatively compare our algorithm with previous related work. We do not claim that normalizing the task of recidivism prediction is something that can be made “fair” with this approach.

5.1 Benchmarks.

In this section, we describe the benchmark algorithms we use for comparison against the proposed framework. Our goal in empirical evaluation is to illustrate the accuracy vs speedup

CPUs at 2.2GHz. We also tested our algorithms on an emerging CPU-attached persistent memory technology using MCAS [Waddington et al., 2021a,b] (see Appendix I). We now describe the datasets we used for our experiments.

Synthetic Datasets. We generate two synthetic datasets of different sizes. The first dataset has 500 features, 1000 observations, and it has size $\sim 10\text{MB}$; the other dataset has 10^6 features, 10^4 observations, and it has size $\sim 19\text{GB}$. In both cases, features are sampled with a joint Gaussian distribution with mean vector $\boldsymbol{\mu} = \mathbf{0}$, and a covariance matrix $\boldsymbol{\Sigma}$ designed to ensure that 10% of the features are highly correlated with the response, and the remaining features have low correlation with the response variable. We then add posterior uniform noise, and normalize the observations.

trade-off that follows from our computational model of adaptive sampling based matching pursuit under constraints. The main purpose of using a technique like adaptive sampling is to obtain speedups by reducing the number of oracle evaluations compared to their non-adaptive counterparts, with some admissible loss in accuracy [Balkanski et al., 2019]. Indeed, there are many other feature selection algorithms. We choose feature selection benchmarks that help us illustrate the said speedup obtained when using our method while ensuring graceful degradation in accuracy in simulated and real-world use-cases. These algorithms include popular selection methods [Elenberg et al., 2018, Grgic-Hlaca et al., 2018b, Kalimeris et al., 2019, Krause and Cevher, 2010]. Other popular algorithms for feature selection include the Maximum Relevance Minimum Redundancy (mRMR) [Ding and Peng, 2005, Zhao et al., 2019] and the Conditional Mutual Information Maximisation (CMIM) filters [Torkkola, 2003] among others. These algorithms iteratively adds features by maximizing suitable objectives, such as scores based on the F-statistic, or Mutual Information gain. However, these algorithms for general p -system side constraints require $\Omega(r)$ sequential calls to the independence oracle for a solution size of r . Thus, such methods are impractical and will be trivially too slow compared to our method. We consider the following algorithms to compare against:

- **SDS_{MA}**: Starting from the empty set, this algorithm adds feasible points to the current solution in a greedy fashion [Elenberg et al., 2018, Krause and Cevher, 2010]. This algorithm uses oracle access to the function f as in (1).
- **SDS_{OMP}**: Starting from the empty set, this algorithm iteratively adds a feature s to the current solution S if it maximizes the dot product $\langle \nabla l(\beta^{(S)}), \mathbf{e}_s \rangle$ [Elenberg et al., 2018, Krause and Cevher, 2010].
- **DASH**: This algorithm follows a computationally similar model, and achieves strong approximation guarantees on the subset selection problem, under the RSC/RSM assumption [Qian and Singer, 2019]. It also uses oracle access to the function f but can only handle r -sparsity constraints.
- **ISK**: This algorithm is the iterated submodular-cost knapsack algorithm proposed by Iyer and Bilmes [2013]. It was used by Grgic-Hlaca et al. [2018b] to perform feature selection on the ProPublica COMPAS dataset. The ISK can only handle r -sparsity constraints, and it has no known guarantees for the problem (2).
- **Lasso**: We also compare against the popular Lasso regression. Tuning Lasso to obtain a solution of a desired size is hard [Mairal and Yu, 2012]. In our experiments, we vary the parameter manually and benchmark against the resulting solution size.
- **Random**: This simple algorithm outputs a maximum independent set of \mathcal{I} chosen uniformly at random. We use the RNDSEQ to generate this set (see Appendix E).

5.2 Results

In this section, we present extensive empirical evidence that validates our theory that the proposed algorithm FAST_{OMP} is significantly faster and more scalable than the baselines, while maintaining competitive performance in accuracy.

Unconstrained Synthetic Dataset. We consider a sparse linear regression task on the small synthetic dataset (~ 10 MB), where the goal is to search for a set of few features

maximizing the R^2 objective. Figure 1(a) showcases the run time of each algorithm for a fixed number of features selected. We observe that our algorithm significantly outperforms baselines. In Figure 1(b), we fix an upper-bound of $k = 150$ on the number of features selected, and we compare the solution accuracy of each algorithm, for a given time budget. We observe that the FASTOMP outperforms the non-oblivious algorithms. Furthermore, we observe that the SDSMA yields significantly worse performance than baselines. In Figure 1(c) we display the solution quality versus the number of features selected. We observe that the FASTOMP, the SDSOMP, the DASH, the Lasso achieve similar solution quality. The SDSMA yields best performance, but it is much slower than the other algorithms.

Synthetic Dataset with Constraints. We search for a set of features maximizing the R^2 objective, for the large synthetic dataset ($\sim 19\text{GB}$). We consider a randomly generated p -system on top of the features. We do not report on the results for the SDSMA, because it was too slow on account of the dataset being too large. We do not test the DASH, the ISK, and the Lasso since they cannot handle p -system side constraints by design.

To illustrate scalability on such a large dataset, we compare speed of feature selection while satisfying the side constraints (Figure 1(d)) and the corresponding solution accuracy with respect to time spent (Figure 1(e)) for selecting upto $k = 300$ features. We observe that our algorithm FASTOMP is significantly faster while maintaining a good quality solution when compared to the SDSOMP and the Random. In Figure 1(f), we further observe that the FASTOMP and the SDSOMP achieve similar solution quality for a given number of features.

CGARN Pan-Cancer Dataset We use the logistic regression to predict if patients have PRAD cancer, or other types of cancer. We enforce a randomly generated p -system on the features. We search for features maximizing the normalized log-likelihood. Again, SDSMA was too slow on this dataset and we do not report on the results for it. We also do not test the DASH, the ISK, and the Lasso since they cannot handle p -systems by design.

To illustrate the scalability and speed of FASTOMP, we show that it is much faster in selecting a given number of features (Figure 1(g)) and achieves much better accuracy for a given run time (1(h)) compared to the baselines. Figure 1(h) is presented till selection of $k = 50$ features, since adding more features did not improve the metric by much as seen in Figure 1(i) which shows that the algorithms FASTOMP and SDSOMP perform similarly in terms of the log-likelihood for a given number of selected features.

ProPublica COMPAS Dataset. In this section, we reproduce the experiments by Grgic-Hlaca et al. [2018b] on the COMPAS dataset. We use regularized logistic regression to predict the recidivism risk and use fairness constraints given by the feature-apriori accuracy [Grgic-Hlaca et al., 2018b]. These constraints are encoded as p -system side constraints $\mathcal{I}_{\text{acc}}^\lambda$ as described in Section 2.2. We report on the outcome fairness of each output feature set, by estimating the racial bias of the corresponding classifier. Following Grgic-Hlaca et al. [2018b], Kleinberg et al. [2017], Zafar et al. [2017a], we examine the false positive (FPR) and false negative (FNR) rates for whites (w) and non-whites (nw) as

$$\text{outcome fairness} = -|\text{FPR}_w - \text{FPR}_{nw}| - |\text{FNR}_w - \text{FNR}_{nw}|.$$

This measure of fairness varies between -2 and 0 , with -2 corresponding to maximum unfairness and 0 to maximum fairness.

The results for this set of experiments are displayed in Figure 2, where we plot the accuracy and outcome fairness as a function of the parameter λ for the constraints $\mathcal{I}_{\text{acc}}^\lambda$. We compare the solution quality and fairness of the FASTOMP, against the ISK, and the solution with best possible accuracy (optimum). We observe that the FASTOMP achieves nearly optimal solution. Although the outcome fairness inevitably decreases for increasing process unfairness, the FASTOMP maintains a more graceful degradation than the other algorithms.

6 Conclusion and Ethics Discussion

We have extended the adaptive sequencing framework, first proposed for maximizing submodular functions, to the setting of feature selection, via the matching pursuit paradigm. Our analysis yields strong performance and approximation guarantees, which are better than previously known results. Furthermore, our proposed formulation is more general than previously considered, as it can handle p -system constraints. While our main contributions are theoretical and algorithmic, we apply these results for fair feature selection. Our framework ensures approximation guarantees, as long as the constraints can be encoded as p -systems.

A major hurdle in further research into fair learning is the lack of gold standard benchmarks. The COMPAS dataset is used for empirical evaluations in several studies. However, its use for benchmarking has also been criticized [Bao et al., 2021b]. Furthermore, over-tuning notions of fairness for a single dataset could be problematic.

Fairness criteria should also take into account the contextual grounding of the dataset, and the trained model that operates within. As such, the mathematical formalism of fairness as constraints may evolve. We hope that our framework motivates further research into addressing the above concerns about fairness in an algorithmically scalable way.

References

- A. Agarwal, A. Beygelzimer, M. Dudík, J. Langford, and H. M. Wallach. A reductions approach to fair classification. In *Proc. of ICML*, volume 80, pages 60–69, 2018.
- A. Agarwal, M. Dudík, and Z. S. Wu. Fair regression: Quantitative definitions and reduction-based algorithms. In *Proc. of ICML*, volume 97, pages 120–129, 2019.
- J. Angwin, J. Larson, S. a. Mattu, and L. Kirchner. There’s software used across the country to predict future criminals. And it’s biased against blacks, 2016.
- S. Bahmani, B. Raj, and P. T. Boufounos. Greedy sparsity-constrained optimization. *Journal of Machine Learning Research*, 14(1):807–841, 2013.
- E. Balkanski and Y. Singer. The adaptive complexity of maximizing a submodular function. In *Proc. of STOC*, pages 1138–1151, 2018.
- E. Balkanski, A. Rubinstein, and Y. Singer. An optimal approximation for submodular maximization under a matroid constraint in the adaptive complexity model. In *Proc. of STOC*, pages 66–77, 2019.
- R. Bamler, C. Zhang, M. Opper, and S. Mandt. Perturbative black box variational inference. In *Proc. of NPIS*, pages 5079–5088, 2017.

- A. S. Bandeira, E. Dobriban, D. G. Mixon, and W. F. Sawin. Certifying the restricted isometry property is hard. *IEEE Transactions of Information Theory*, 59(6):3448–3450, 2013.
- M. Bao, A. Zhou, S. Zottola, B. Brubach, S. Desmarais, A. Horowitz, K. Lum, and S. Venkatasubramanian. It’s compaslicated: The messy relationship between RAI datasets and algorithmic fairness benchmarks. *CoRR*, abs/2106.05498, 2021a.
- M. Bao, A. Zhou, S. A. Zottola, B. Brubach, S. Desmarais, A. S. Horowitz, K. Lum, and S. Venkatasubramanian. It’s COMPASlicated: The messy relationship between RAI datasets and algorithmic fairness benchmarks. In *Proc. of NeurIPS Datasets and Benchmarks Track*, 2021b.
- R. Berk, H. Heidari, S. Jabbari, M. Kearns, and A. Roth. Fairness in criminal justice risk assessments: The state of the art. *Sociological Methods & Research*, 50(1):3–44, 2021.
- A. Beutel, J. Chen, T. Doshi, H. Qian, L. Wei, Y. Wu, L. Heldt, Z. Zhao, L. Hong, E. H. Chi, and C. Goodrow. Fairness in recommendation ranking through pairwise comparisons. In *Proc. of SIGKDD*, pages 2212–2220, 2019.
- A. Borodin, J. von zur Gathen, and J. E. Hopcroft. Fast parallel matrix and GCD computations. *Information and Control*, 52(3):241–256, 1982.
- A. Breuer, E. Balkanski, and Y. Singer. The FAST algorithm for submodular maximization. In *Proc. of ICML*, pages 1134–1143, 2020.
- Cancer Genome Atlas Research Network, J. N. Weinstein, E. A. Collisson, G. B. Mills, K. R. M. Shaw, B. A. Ozenberger, K. Ellrott, I. Shmulevich, C. Sander, and J. M. Stuart. The cancer genome atlas pan-cancer analysis project. *Nature Genetics*, 45(10):1113–20, 2013.
- L. E. Celis, V. Keswani, D. Straszak, A. Deshpande, T. Kathuria, and N. K. Vishnoi. Fair and diverse dpp-based data summarization. In *Proc. of ICML*, pages 715–724, 2018.
- C. Chekuri and K. Quanrud. Parallelizing greedy for submodular set function maximization in matroids and beyond. In *Proc. of STOC*, pages 78–89, 2019.
- L. Chen, M. Feldman, and A. Karbasi. Weakly submodular maximization beyond cardinality constraints: Does randomization help greedy? In J. Dy and A. Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 804–813. PMLR, 10–15 Jul 2018.
- F. Chierichetti, R. Kumar, S. Lattanzi, and S. Vassilvitskii. Matroids, matchings, and fairness. In *Proc. of AISTATS*, pages 2212–2220, 2019.
- A. L. Chistov. Fast parallel calculation of the rank of matrices over a field of arbitrary characteristic. In *Proc. of FCT*, pages 63–69, 1985.
- A. Chouldechova. Fair prediction with disparate impact: A study of bias in recidivism prediction instruments. *Big data*, 5(2):153–163, 2017.
- S. Corbett-Davies, E. Pierson, A. Feller, S. Goel, and A. Huq. Algorithmic decision making and the cost of fairness. In *Proc. of KDD*, pages 797–806, 2017.

- S. Corbett-Davies, E. Pierson, A. Feller, S. Goel, and A. Huq. Algorithmic decision making and the cost of fairness. In *Proc. of KDD*, pages 797–806, 2017.
- B. Cowgill and C. E. Tucker. Economics, fairness and algorithmic bias. *preparation for: Journal of Economic Perspectives*, 2019.
- A. Das and D. Kempe. Submodular meets spectral: Greedy algorithms for subset selection, sparse approximation and dictionary selection. In *Proc. of ICML*, pages 1057–1064, 2011.
- C. H. Q. Ding and H. Peng. Minimum redundancy feature selection from microarray gene expression data. *Journal of Bioinformatics and Computational Biology*, 3(2):185–206, 2005.
- M. Donini, L. Oneto, S. Ben-David, J. Shawe-Taylor, and M. Pontil. Empirical risk minimization under fairness constraints. In *Proc. of NeurIPS*, pages 2796–2806, 2018.
- J. Edmonds. *Submodular functions, matroids, and certain polyhedra.*, pages 69–87. Springer, 1970.
- E. R. Elenberg, R. Khanna, A. G. Dimakis, and S. N. Negahban. Restricted strong convexity implies weak submodularity. *The Annals of Statistics*, 46(6B):3539–3568, 2018.
- M. Fahrback, V. S. Mirrokni, and M. Zadimoghaddam. Submodular maximization with nearly optimal approximation, adaptivity and query complexity. In *Proc. of SODA*, pages 255–273, 2019.
- M. Feldman, S. A. Friedler, J. Moeller, C. Scheidegger, and S. Venkatasubramanian. Certifying and removing disparate impact. In *Proc. of SIGKDD*, pages 259–268, 2015.
- M. Feldman, C. Harshaw, and A. Karbasi. Greed is good: Near-optimal submodular maximization via greedy optimization. In *Proc. of COLT*, volume 65, pages 758–784, 2017.
- K. Gatmiry and M. Gomez-Rodriguez. On the network visibility problem. *CoRR*, abs/1811.07863, 2018.
- N. Grgic-Hlaca, M. B. Zafar, K. Gummadi, and A. Weller. The case for process fairness in learning: Feature selection for fair decision making. In *NeurIPS Symposium on Machine Learning and the Law*, 2016.
- N. Grgic-Hlaca, E. M. Redmiles, K. P. Gummadi, and A. Weller. Human perceptions of fairness in algorithmic decision making: A case study of criminal risk prediction. In *Proc. of WWW*, pages 903–912, 2018a.
- N. Grgic-Hlaca, M. B. Zafar, K. P. Gummadi, and A. Weller. Beyond distributive fairness in algorithmic decision making: Feature selection for procedurally fair learning. In *Proc. of AAAI*, pages 51–60, 2018b.
- M. E. Halabi, S. Mitrovic, A. Norouzi-Fard, J. Tardos, and J. Tarnawski. Fairness in streaming submodular maximization: Algorithms and hardness. In *Proc. of NeurIPS*, 2020.
- O. H. Ibarra, S. Moran, and L. E. Rosier. A note on the parallel complexity of computing the rank of order n matrices. *Information Processing Letters*, 11(4/5):162, 1980.

- R. K. Iyer and J. A. Bilmes. Submodular optimization with submodular cover and submodular knapsack constraints. In *Proc. of NIPS*, pages 2436–2444, 2013.
- P. Jain, A. Tewari, and P. Kar. On iterative hard thresholding methods for high-dimensional m-estimation. In *Proc. of NIPS*, pages 685–693, 2014.
- A. Jalali, C. C. Johnson, and P. Ravikumar. On learning discrete graphical models using greedy methods. In *Proc. of NIPS*, pages 1935–1943, 2011.
- T. A. Jenkyns. The efficacy of the "greedy" algorithm. In *Proceedings of the 7th Southeastern Conference on Combinatorics, Graph Theory and Computing, 1976*, pages 341–350, 1976.
- D. Kalimeris, G. Kaplun, and Y. Singer. Robust influence maximization for hyperparametric models. In *Proc. of ICML*, pages 3192–3200, 2019.
- R. M. Karp, E. Upfal, and A. Wigderson. The complexity of parallel search. *Journal of Computer and System Science*, 36(2):225–253, 1988.
- M. P. Kim, O. Reingold, and G. N. Rothblum. Fairness through computationally-bounded awareness. In *Proc. of NeurIPS*, pages 4847–4857, 2018.
- J. M. Kleinberg, S. Mullainathan, and M. Raghavan. Inherent trade-offs in the fair determination of risk scores. In *Proc. of ITCS*, volume 67, pages 43:1–43:23, 2017.
- A. Krause and V. Cevher. Submodular dictionary selection for sparse representation. In *Proc. of ICML*, pages 567–574, 2010.
- P. Lahoti, A. Beutel, J. Chen, K. Lee, F. Prost, N. Thain, X. Wang, and E. Chi. Fairness without demographics through adversarially reweighted learning. In *Proc. of NeurIPS*, 2020.
- J. Larson, M. Roswell, and V. Atlidakis. Data and analysis for ‘how we analyzed the COMPAS recidivism algorithm’. <https://github.com/propublica/compas-analysis>, 2016.
- J. Liu, J. Ye, and R. Fujimaki. Forward-backward greedy algorithms for general convex smooth functions over A cardinality constraint. In *Proc. of ICML*, volume 32, pages 503–511, 2014.
- J. Mairal and B. Yu. Complexity analysis of the lasso regularization path. In *Proc. of ICML*, 2012.
- B. Mirzasoleiman, A. Badanidiyuru, and A. Karbasi. Fast constrained submodular maximization: Personalized data summarization. In *Proc. of ICML*, volume 48, pages 1358–1367, 2016.
- B. Mirzasoleiman, S. Jegelka, and A. Krause. Streaming non-monotone submodular maximization: Personalized video summarization on the fly. In *Proc. of AAAI*, pages 1379–1386, 2018.
- K. Mulmuley. A fast parallel algorithm to compute the rank of a matrix over an arbitrary field. *Combinatorica*, 7(1):101–104, 1987.
- D. Needell and J. A. Tropp. Cosamp: iterative signal recovery from incomplete and inaccurate samples. *Communication of the ACM*, 53(12):93–100, 2010.

- S. Negahban, P. Ravikumar, M. Wainwright, and B. Yu. A unified framework for high-dimensional analysis of m-estimators with decomposable regularizers. *Statistical Science*, 27: 538–557, 10 2010.
- G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher. An analysis of approximations for maximizing submodular set functions - II. *Mathematical Programming*, 14(3):73–87, 1978.
- S. Nowozin. Debiasing evidence approximations: On importance-weighted autoencoders and jackknife variational inference. In *Proc. of ICLR*, 2018.
- C. Qian, Y. Yu, and Z. Zhou. Subset selection by pareto optimization. In *Proc. of NIPS*, pages 1774–1782, 2015.
- C. Qian, C. Bian, and C. Feng. Subset selection by pareto optimization with recombination. In *Proc. of AAAI*, pages 2408–2415, 2020.
- S. Qian and Y. Singer. Fast parallel algorithms for statistical subset selection problems. In *Proc. of NeurIPS*, pages 5073–5082, 2019.
- F. Quinzan, V. Doskoc, A. Göbel, and T. Friedrich. Adaptive sampling for fast constrained maximization of submodular functions. In *Proc. of AISTATS*, pages 964–972, 2021.
- P. Rigollet and A. Tsybakov. Exponential screening and optimal rates of sparse estimation. *The Annals of Statistics*, 39:731–771, 2010.
- C. Rudin, C. Wang, and B. Coker. The age of secrecy and unfairness in recidivism prediction. *arXiv:1811.00731*, 2018.
- S. Sakaue. On maximization of weakly modular functions: Guarantees of multi-stage algorithms, tractability, and hardness. In *Proc. of AISTATS*, pages 22–33, 2020.
- A. D. Selbst, D. Boyd, S. A. Friedler, S. Venkatasubramanian, and J. Vertesi. Fairness and abstraction in sociotechnical systems. *Proc. of FAccT*, 2019.
- K. Torkkola. Feature extraction by non-parametric mutual information maximization. *Journal of Machine Learning Research*, 3:1415–1438, 2003.
- D. Waddington, C. Dickey, M. Hershcovitch, and S. Seshadri. An architecture for memory centric active storage (MCAS). *CoRR*, abs/2103.00007, 2021a.
- D. Waddington, C. Dickey, L. Xu, M. Hershcovitch, and S. Seshadri. A high-performance persistent memory key-value store with near-memory compute. *CoRR*, abs/2104.06225, 2021b.
- B. E. Woodworth, S. Gunasekar, M. I. Ohannessian, and N. Srebro. Learning non-discriminatory predictors. In *Proc. of COLT*, volume 65, pages 1920–1953, 2017.
- X. Yuan, P. Li, and T. Zhang. Gradient hard thresholding pursuit. *Journal of Machine Learning Research*, 18:166:1–166:43, 2017.
- M. B. Zafar, I. Valera, M. Gomez-Rodriguez, and K. P. Gummadi. Fairness constraints: Mechanisms for fair classification. In *Proc. of AISTATS*, volume 54, pages 962–970, 2017a.

- M. B. Zafar, I. Valera, M. Gomez-Rodriguez, and K. P. Gummadi. Fairness beyond disparate treatment & disparate impact: Learning classification without disparate mistreatment. In *Proc. of WWW*, pages 1171–1180, 2017b.
- M. B. Zafar, I. Valera, M. Gomez-Rodriguez, K. P. Gummadi, and A. Weller. From parity to preference-based notions of fairness in classification. In *Proc. of NIPS*, pages 229–239, 2017c.
- Z. Zhao, R. Anand, and M. Wang. Maximum relevance and minimum redundancy feature selection methods for a marketing machine learning platform. In *Proc. of DSAA*, pages 442–452, 2019.

Appendix

A Motivating Example

In this section, we motivate our analysis by showing that the adaptive sequencing prototype by Balkanski et al. [2019] does not work for feature selection. This example can also be used to show that similar algorithms, such as FAST [Breuer et al., 2020] do not work for feature selection.

The adaptive sequencing prototype is presented in Algorithm 2. We refer to this algorithm as the FAST. Starting from the empty set, the FAST generates at every iteration a uniformly random sequence $\{a_1, \dots, a_k\}$ of the elements \mathbf{X} not yet discarded. This sequence can be sampled uniformly at random, or using the RNDSEQ sub-routine described in Appendix E. Afterwards, the FAST filters elements that have a high marginal contribution, when added to \mathbf{S} , and it determines the prefix of $\{a_1, \dots, a_k\}$ that is added to the current solution \mathbf{S} . This prefix has the property that there is a large fraction of elements in \mathbf{X} with high contribution to the current solution \mathbf{S} .

Following an example provided by Elenberg et al. [2018], we show that the FAST fails on a simple linear regression task with three features. For a fixed parameter $z > 0$, consider the following variables:

$$\begin{aligned} \mathbf{y} &= [1, 0, 0]^T & \mathbf{x}_2 &= [z, \sqrt{1 - z^2}, 0]^T \\ \mathbf{x}_1 &= [0, 1, 0]^T & \mathbf{x}_3 &= [\delta z, 0, \sqrt{1 - \delta^2 z^2}]^T \end{aligned}$$

Note that all variables have unit norm. Our goal is to choose two of the three variables $\{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3\}$ that best estimate \mathbf{y} , with respect to the R^2 objective. To this end, we introduce additional notation. For a given index set $\mathbf{S} \subseteq [3]$, we denote with $\mathbf{X}_{\mathbf{S}}$ the matrix whose columns consist of the features indexed by \mathbf{S} . For instance, for $\mathbf{S} = \{1, 3\}$ it holds

$$\mathbf{X}_{\{1,3\}} = \begin{bmatrix} 0 & \delta z \\ 1 & 0 \\ 0 & \sqrt{1 - \delta^2 z^2} \end{bmatrix}.$$

With this notation, we define the objective function for our problem as

$$f(\mathbf{S}) = R^2(\boldsymbol{\beta}^{(\mathbf{S})}) - R^2(\mathbf{0}) = (\mathbf{y}^T \mathbf{X}_{\mathbf{S}})(\mathbf{X}_{\mathbf{S}}^T \mathbf{X}_{\mathbf{S}})^{-1}(\mathbf{X}_{\mathbf{S}}^T \mathbf{y}), \quad (3)$$

with $R^2(\cdot)$ the R^2 objective evaluated on the model for an input parameter vector $\boldsymbol{\beta}^{(\mathbf{S})}$. Using this formula, one can easily see that it holds

$$\begin{aligned} f(\{1\}) &= 0 & f(\{1, 2\}) &= 1 \\ f(\{2\}) &= z^2 & f(\{1, 3\}) &= \delta^2 z^2 \\ f(\{3\}) &= \delta^2 z^2 & f(\{2, 3\}) &= (1 + \delta)z^2 + \delta^2 z^4 \end{aligned}$$

Clearly, the optimal solution is $f(\{1, 2\})$. However, on this instance the FAST outputs the solution $\mathbf{S} = \emptyset$, attaining an f -value of $f(\mathbf{S}) = 0$. The following lemma holds.

Algorithm 2 FAST

 $S \leftarrow \emptyset, t \leftarrow \max_{e \in [n]} f(e);$ **for** Δ *iterations* **do** $X \leftarrow [n];$ **while** $X \neq \emptyset$ **do** $\{a_1, a_2, \dots, a_k\} \leftarrow \text{RNDSEQ}(X, S);$ $X_i \leftarrow \{e \in X: f_{S \cup \{a_1, \dots, a_i\}}(e) \geq t \text{ and } S \cup \{a_1, \dots, a_i, e\} \in \mathcal{I}\} \quad i^* \leftarrow \min\{i: |X_i| \leq (1 - \varepsilon)|X|\};$ $S \leftarrow S \cup \{a_1, \dots, a_{i^*}\};$ $X = X_{i^*}$ **end** $t \leftarrow (1 - \varepsilon)t;$ **end****return** $S;$

Lemma A.1. *Consider the FAST optimizing the function $f(S)$ as in (3), over sets $S \subseteq [3]$ of size at most $|S| \leq k$ with $k = 2$. Then, for any constant $\varepsilon > 0$, the FAST outputs either the solution $\{1, 3\}$ or the solution $\{2, 3\}$ with probability at least $1 - (2/3)^{\varepsilon^{-1} \log 1/\delta}$. In particular, the expected approximation guarantee of FAST converges to zero, for $\delta, z \rightarrow 0$.*

Proof. At the beginning of the optimization process, the constant t is set to $t = \delta^2 z^2$ and a sequence $\{a_1, a_2\}$. This sequence yield $\{a_1\} = \{3\}$ at least with probability $1/3$. If $\{a_1\} = \{3\}$ sequence is sampled, then the point $\{3\}$ is added to the current solution. Otherwise, the value t decreases of a multiplicative factor of $(1 - \varepsilon)$, and a new sequence is sampled. As long as $t > z^2$, the FAST can only output a solution that contains the point $\{3\}$. Hence, the solution $\{1, 2\}$ can only be sampled after t decreases to a value $t < z$, which requires at least $\varepsilon^{-1} \log 1/\delta$ iterations of the outer loop. Hence, the probability of sampling the solution $\{1, 2\}$ can be upper-bounded by the probability that no sequence with $\{a_1\} = \{3\}$ is sampled during the first $\varepsilon^{-1} \log 1/\delta$ iterations of the outer-loop. This probability can be estimated as $2/3^{\varepsilon^{-1} \log 1/\delta}$. Hence, the FAST outputs either the solution $\{1, 3\}$ or the solution $\{2, 3\}$ with probability at least $1 - (2/3)^{\varepsilon^{-1} \log 1/\delta}$. \square

We remark that in this example, the FAST_{OMP} outputs the optimal solution in one iteration, at least with constant probability. We can bound the optimal parameters and in Line 4 of Algorithm 1 in a similar fashion as in Proposition C.5, and obtain that m and M are bounded as $m \geq 1 - \sqrt{1 - z^2}$ and $M \leq 1 + \sqrt{1 - z^2}$. It follows that the parameter in Line 4 of Algorithm 1 is upper-bounded as $t \leq (1 - \varepsilon)(1 - \sqrt{1 - z^2})/2 \leq z^2$, for δ sufficiently small. Suppose now that at the beginning of the iteration, a sequence $\{a_1, a_2\} = \{2, 1\}$. Since $t \leq z^2$, then the entire sequence is added to the current solution $\{2, 1\}$ and the algorithm outputs the optimum. Note that the desired sequence is sampled with probability $1/6$. It follows that the FAST_{OMP} outputs the optimum at least with constant probability. Hence, the FAST_{OMP} maintains a constant-factor approximation guarantee.

B Weak Submodularity

Consider a function l that is restricted smooth and restricted strong concave. It is well known that the corresponding function f as in (1) has weak diminishing return properties. These diminishing returns property is called *weak submodularity*, and it was first introduced by [Das and Kempe \[2011\]](#) to study statistical subset selection problems. Functions that exhibit weak submodularity, i.e., weakly submodular functions, are defined in terms of the submodularity ratio, as follows.

Definition B.1 (Weak submodularity, Definition 2.3 by [Das and Kempe \[2011\]](#)). Consider a function $f: 2^{[n]} \rightarrow \mathbb{R}_{\geq 0}$. The submodularity ratio is defined as the largest scalar γ such that

$$\sum_{j \in S} (f(L \cup \{j\}) - f(L)) \geq \gamma (f(L \cup S) - f(L)),$$

for all sets $L, S \in [n]$ such that $L \cap S \neq \emptyset$. We say that f is γ -weakly submodular if its submodularity ratio is γ .

There is a well-known connection between weak submodularity and Problem 2. This connection was discovered by [Elenberg et al. \[2018\]](#), and it can be formalized as follows.

Theorem B.2 (Theorem 1 by [Elenberg et al. \[2018\]](#)). Define f as in (1), with a function l that is $(m_{|U|+k}, M_{|U|+k})$ -(strongly concave, smooth) on $\Omega_{|U|+k}$, and $\tilde{M}_{|U|+1}$ smooth on $\Omega_{|U|+1}$. Fix a set $U \in [n]$, and denote with $\gamma_{U,k}$ the largest scalar such that

$$\sum_{j \in S} (f(L \cup \{j\}) - f(L)) \geq \gamma_{U,k} (f(L \cup S) - f(L)),$$

for all sets $L, S \in [n]$ such that $L \cap S \neq \emptyset, L \subseteq U, |S| \leq k$. Then, the constant $\gamma_{U,k}$ is lower-bounded as

$$\gamma_{|U|,k} \geq \frac{m_{|U|+k}}{\tilde{M}_{|U|+1}} \geq \frac{m_{|U|+k}}{M_{|U|+k}}.$$

C Restricted Strong Concavity and Smoothness

Given a set of observations and a parametric family of distributions $\{p(\cdot; \beta) \mid \beta \in \Omega\}$ with $\Omega \subseteq \mathbb{R}^n$, we wish to identify a vector of parameters β maximizing the goodness of fit for these observations, according to a chosen measure l . For generalized linear models, common measures for feature selection are restricted strong concave and restricted smooth. We study the log-likelihood and the coefficient of determination, although analogous results hold for other similar statistics [[Das and Kempe, 2011](#), [Qian and Singer, 2019](#)].

Maximizing the log-conditional. Assuming that the response follows a distribution in an exponential family, the log-conditional can be written as

$$\log p(\mathbf{y} \mid \mathbf{X}; \beta) = h^{-1}(\tau) - Z(\mathbf{X}, \beta) + g(\mathbf{y}, \tau) \quad (4)$$

with Z the log-partition function, and τ the dispersion parameter. The log-conditional is commonly used for learning. In the case of the simple linear model, it is possible to derive approximation guarantees in terms of the eigenvalues of \mathbf{X} . Additional assumptions on the random design of \mathbf{X} ensure that the log-conditional objective is restricted smooth and restricted strong concave. We refer the reader to Appendix C.1 for a discussion on these results.

Maximizing the R^2 objective. Consider a linear model with a normalized response variable \mathbf{y} . The R^2 objective is defined as

$$R_{\mathbf{X},\mathbf{y}}^2(\boldsymbol{\beta}) := 1 - \mathbb{E} [(\mathbf{y} - \langle \mathbf{X}, \boldsymbol{\beta} \rangle)^2]. \quad (5)$$

This function is a popular measure for the goodness of fit. The function $R_{\mathbf{X},\mathbf{y}}^2(\boldsymbol{\beta})$ is restricted strong concave and restricted smooth, with parameters depending on the properties of the matrix \mathbf{X} . A discussion on these results is deferred to Appendix C.2.

C.1 Restricted Strong Concavity and Smoothness of the Log-Conditional

In this section we discuss results concerning the (M, m) -(smoothness, strong concavity) of the log-conditional for generalized linear models. Consider a feature matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ and response variable \mathbf{y} . Assuming that the response follows a distribution in an exponential family, the log-conditional can be written as

$$\log p(\mathbf{y} | \mathbf{X}; \boldsymbol{\beta}) = h^{-1}(\tau) - Z(\mathbf{X}, \boldsymbol{\beta}) + g(\mathbf{y}, \tau) \quad (6)$$

with Z the log-partition function, and τ the dispersion parameter. We give approximation guarantees for the objective

$$l(\boldsymbol{\beta}) = \log p(\mathbf{y} | \mathbf{X}; \boldsymbol{\beta}) - \eta \|\boldsymbol{\beta}\|_2^2, \quad (7)$$

for some parameter $\eta \geq 0$. We show that the function l is restricted smooth and restricted strong concave under various assumptions. We start discussing the simplest case of a logistic regression. We then study the general case as in (6), under the assumption that l has a non-zero regularization term. We then conclude with the general case, i.e., no assumptions on the regularization term.

Logistic regression. The log-likelihood function for the logistic regression is defined as

$$l(\boldsymbol{\beta}^{(S)}) := \sum_i y_i \langle \mathbf{X}_S, \boldsymbol{\beta} \rangle - \log \left(1 + e^{\langle \mathbf{X}_S, \boldsymbol{\beta} \rangle} \right), \quad (8)$$

with \mathbf{X}_S the matrix of all features indexed by S and y_i the i -th observation, i.e., the i -th coefficient of the response \mathbf{y} . For this class of log-likelihood functions, the following result holds.

Proposition C.1. *The log-likelihood function l for the logistic regression as in (8), is (m, M) -(restricted smooth, restricted strong concave) with parameters*

$$m := \min_S \lambda_{\min}(\mathbf{X}_S^T \mathbf{X}_S) \text{ and } M := \max_S \lambda_{\max}(\mathbf{X}_S^T \mathbf{X}_S).$$

Log-conditional with non-zero regularization term. We now study log-conditional functions as in (6), under the assumption that the corresponding objective l has a regularization term with parameter $\eta > 0$. We introduce additional notation to this end. For any feature set $T \subseteq [n]$, denote with \mathcal{P}_T an operator that takes as input vectors $\mathbf{x} \in \mathbb{R}^n$, and it replaces all indices in $[n] \setminus T$ of \mathbf{x} by 0. For any vector $\mathbf{x} \in \mathbb{R}^p$, we define $\mathbf{x}_T := \mathcal{P}_T(\mathbf{x})$. We consider the following assumption on the distribution of the features.

Assumption C.2 (Assumption by Bahmani et al. [2013]). For fixed constants $r, R > 0$, we make the following assumption on the feature matrix $\mathbf{X} \in \mathbb{R}^{n \times p}$. The rows \mathbf{x} of \mathbf{X} are generated *i.i.d.*, such that the following additional conditions hold. For any set $\mathsf{T} \subseteq [p]$ of size $|\mathsf{T}| \leq r$,

- $\|\mathbf{x}_{\mathsf{T}}\|_2 \leq R$;
- none of the matrices $\mathcal{P}_{\mathsf{T}} \mathbb{E}[\mathbf{x}\mathbf{x}^T] \mathcal{P}_{\mathsf{T}}$ are the zero matrix.

Following this notation, define

$$\phi_{\max} := \max_{|\mathsf{T}| \leq k} \lambda_{\max}(\mathcal{P}_{\mathsf{T}} \mathbf{C} \mathcal{P}_{\mathsf{T}}) \quad \phi_{\min} := \min_{|\mathsf{T}| \leq k} \lambda_{\max}(\mathcal{P}_{\mathsf{T}} \mathbf{C} \mathcal{P}_{\mathsf{T}}),$$

with $\lambda_{\max}(\cdot)$ the largest eigenvalue. The following corollary holds.

Proposition C.3 (Corollary 4 by Elenberg et al. [2018]). Consider a function l as in (7), and suppose that $\eta > 0$. Suppose that Assumption C.2 holds with parameters r, R , and suppose that the number of samples s is lower-bounded as

$$s > \frac{R(\log r + r(1 + \log \frac{n}{k} - \log \delta))}{\phi_{\min}(1 + \varepsilon) \log(1 + \varepsilon) - \varepsilon}.$$

Then, with probability at least $1 - \delta$ the function l is (m, M) -(smooth, restricted concave), for all β with at most r non-zero coefficients. The parameters m and M are defined as $m = q(1 + \varepsilon)\phi_{\max} + \eta$ and $M = \eta$, with q a constant fulfilling $q \geq \max_i h^{-1}(\tau) Z''(\beta, \mathbf{x}_i)$ for $h^{-1}(\cdot), Z(\cdot, \cdot)$ as in (6).

Log-conditional with no assumptions on the regularization term. We now study log-conditional functions as in (6), with no additional assumption on the regularization term. For simplicity, we consider functions l as in (6) with $\eta = 0$. However, these results can easily be extended to the general case. The following lemma holds.

Lemma C.4 (Corollary 2 by Elenberg et al. [2018]). Denote with r an upper-bound on the sparsity of the feature sets. Following the notation introduced above, suppose that the feature matrix \mathbf{X} consists of samples drawn from a sub-Gaussian distribution with parameter σ^2 and covariance matrix Σ . Then, for $\eta = 0$ the function l as in (7) is (M, m) -(smooth, restricted concave) with parameters

$$M = \alpha_u \lambda_{\max}(\Sigma) \left(3 + \frac{2nr}{s} \right) \quad \text{and} \quad m = \alpha_\ell - \frac{c^2 \sigma^2 k \log n}{\alpha_\ell s},$$

with high probability, for $s > 0$ sufficiently large. The constant α_ℓ depends on (σ^2, Σ) and k ; the constant α_u yields $\alpha_u \geq \max_i h^{-1}(\tau)^{-1} Z''(\beta, \mathbf{x}_i)$, with $h^{-1}(\cdot), Z(\cdot, \cdot)$ as in (6).

C.2 Restricted Strong Concavity and Smoothness of the R^2 Objective

In this section, we prove guarantees for the R^2 objective. To this end, given a feature matrix consisting of n features and k observations, the R^2 objective with regularization can be written as

$$l(\beta) = R_{\mathbf{X}, \mathbf{y}}^2(\beta) = 1 - \frac{1}{k} \|\mathbf{y} - \langle \mathbf{X}, \beta \rangle\|_2^2. \quad (9)$$

The following lemma, similar to Lemma C.1, holds.

Algorithm 3 RNDSEQ(X, S)

$A \leftarrow \emptyset$ **while** $X \neq \emptyset$ **do**
 Let $\{x_1, \dots, x_n\}$ be a permutation of X chosen uniformly at random $j^* \leftarrow \max\{j: S \cup A \cup \{x_1, \dots, x_j\} \in \mathcal{I}\}$ $A \leftarrow A \cup \{x_1, \dots, x_{j^*}\}$ $X \leftarrow \{e \in X \setminus (S \cup A): S \cup A \cup e \in \mathcal{I}\}$
end
return A

Proposition C.5. *The R^2 objective l for the linear regression as in (9), is (m, M) -(restricted smooth, restricted strong concave) with parameters*

$$m := \min_S \lambda_{\min}(\mathbf{X}_S^T \mathbf{X}_S) \text{ and } M := \max_S \lambda_{\max}(\mathbf{X}_S^T \mathbf{X}_S).$$

This lemma can easily be extended to the case of an R^2 objective with regularization term.

D Adaptivity and the PRAM Model

Recall that the adaptivity is defined as follows [Balkanski and Singer, 2018].

Definition D.1 (Adaptivity). Given an oracle f , an algorithm is r -adaptive if every query q to the oracle f occurs at a round $i \in [r]$ such that q is independent of the answers $f(q')$ to all other queries q' at round i .

The notion of adaptivity is closely related to other models such as Parallel Random Access Machines (PRAM). The PRAM model consists of a set of processors, that communicate via a single shared memory and a memory access unit. The adaptivity extend to PRAM via the notion of depth. The depth is the number of parallel steps in an algorithm or the longest chain of dependencies. We remark that the PRAM model assumes that the input is loaded in memory, whereas the adaptive complexity model only assumes access to an oracle function.

E The RNDSEQ Sub-Routine

The RNDSEQ sub-routine is presented in Algorithm 3. This algorithm correspond to Algorithm A by Karp et al. [1988]. This algorithm solves the problem of sampling a maximum independent set of \mathcal{I} uniformly at random. To our knowledge, no other algorithm is known for this problem, with better adaptivity than Algorithm 3. Given as input a ground set X , a current solution S , and a p -system \mathcal{I} , this algorithm finds a random set A such that $S \cup A$ is a maximum independent set for \mathcal{I} . This algorithm iteratively shuffles the set X , and then it identifies the longest prefix of this sequence that can be added to S , without violating side constraints. This prefix is then added to A . This algorithm terminates when $S \cup A \in \mathcal{I}$ is a maximal independent set.

This algorithm uses parallel calls to the independence oracle of \mathcal{I} . In fact, the evaluations for the feasibility of prefixes of A can be preformed in parallel. Hence, with this algorithm the adaptivity of the independence oracle corresponds to the number of iterations until convergence. It is well-known that this algorithm converges after expected $\mathcal{O}(\sqrt{r})$ iterations, with r the rank of \mathcal{I} (see Theorem 6 by Karp et al. [1988]). This implies that the adaptivity of the independence oracle is $\mathcal{O}(\sqrt{r})$. Although it is not known if this upper-bound on the adaptivity

is tight, it is known that there is no algorithm that finds a maximum independent set of \mathcal{I} with less than $\tilde{\Omega}(n^{1/3})$ rounds (see Theorem 7 by [Karp et al. \[1988\]](#)).

F Parameter Tuning for Algorithm 1

If l is the log-likelihood of a linear model, or the R^2 objective of a linear model with normalized response variable, then m and M can be estimated in terms of the design of the feature matrix (see Appendix C.1-C.2). These estimates need not be tight, and certifying bounds for m and M is NP-hard [[Bandeira et al., 2013](#)]. In general, the constant m/M in Line 4 of Algorithm 1 requires tuning by making multiple runs of the algorithm. We remark that other known parallel algorithms for feature selection also require estimates of these parameters [[Qian and Singer, 2019](#)].

Parallel algorithms that estimate the rank are known for several p -systems. For instance, the rank of a graphic matroid can be estimated with parallel algorithms that compute spanning trees. Furthermore, parallel rank oracles are known for matroids that can be represented as independent sets of vectors in a given field [[Borodin et al., 1982](#), [Chistov, 1985](#), [Ibarra et al., 1980](#), [Mulmuley, 1987](#)]. These algorithms can also be used to estimate the rank of more complex constraints, such as the intersection of matroids or p -matchoids.

G Proof of Theorem 4.1

We prove the following theorem.

Theorem 4.1. *Define the support selection function $f(\cdot)$ as in (1), for the given function $l(\cdot)$ that is (M, m) -restricted smooth, restricted strong concave, on the sparse sub-domain Ω_{2r} . Consider a p -system \mathcal{I} of rank r over $[n]$, and let \mathbf{S}^* be the output of Algorithm 1 while OPT is the optimum solution set for the Problem 2. Then,*

$$\frac{\mathbb{E} [f(\mathbf{S}^*)]}{\text{OPT}} \geq \frac{1}{1+p} \left(1 - \exp \left\{ -(1-\varepsilon)^2 \frac{m^3}{M^3} \right\} \right),$$

for all $0 < \varepsilon < 1$. Furthermore, in the specific case when \mathcal{I} is r -sparsity constraint over $[n]$, then,

$$\frac{\mathbb{E} [f(\mathbf{S}^*)]}{\text{OPT}} \geq \left(1 - \exp \left\{ -(1-\varepsilon)^2 \frac{m^2}{M^2} \right\} \right).$$

The proof of this theorem is based on a few lemmas and propositions, which we discuss in Appendix G.1 before proving Theorem 4.1. On a high level, the proof of Theorem 4.1 is split into two separate cases. First we prove that Theorem 4.1 holds when Algorithm 1 terminates after ε^{-1} iterations of the outer While-loop of Algorithm 1. Then, we prove Theorem 4.1 under the assumption that Algorithm 1 finds a solution of size k . The first part of the proof is discussed in Appendix G.3 (see Theorem G.8), and the second case is discussed in Appendix G.2 (see Theorem G.5).

G.1 Preliminary Results

Our analysis is based on a few preliminary result, which we discuss in this section.

Theorem G.1. *Suppose the l is (M, m) -(smooth, strongly concave) on Ω_{2k} . Then, for each subsets $\mathsf{S}, \mathsf{T} \subseteq [p]$ of size at most k it holds*

$$2M \sum_{j \in \mathsf{T}} f_{\mathsf{S}}(j) \geq \|\nabla l(\boldsymbol{\beta}^{(\mathsf{S})})_{\mathsf{T}}\|_2^2 \geq 2m \sum_{j \in \mathsf{T}} f_{\mathsf{S}}(j).$$

Proof. We start proving the first inequality. Fix a point $j \in \mathsf{T}$. Then, for any scalar α it holds

$$\begin{aligned} f_{\mathsf{S}}(j) &= l(\boldsymbol{\beta}^{(\mathsf{S} \cup \{j\})}) - l(\boldsymbol{\beta}^{(\mathsf{S})}) \geq l(\boldsymbol{\beta}^{(\mathsf{S})} + \alpha \mathbf{e}_j) - l(\boldsymbol{\beta}^{(\mathsf{S})}) \quad (\text{maximality of } \boldsymbol{\beta}^{(\mathsf{S} \cup \{j\})}) \\ &\geq \langle \nabla l(\boldsymbol{\beta}^{(\mathsf{S})}), \alpha \mathbf{e}_j \rangle - \frac{M}{2} \alpha^2, \quad (\text{restricted smoothness}) \end{aligned} \quad (10)$$

By substituting $\alpha = \frac{1}{M} \langle \nabla l(\boldsymbol{\beta}^{(\mathsf{S})}), \mathbf{e}_j \rangle$ in (10), we get

$$2M \sum_{j \in \mathsf{T}} f_{\mathsf{S}}(j) \geq \sum_{j \in \mathsf{T}} \left\langle \nabla l(\boldsymbol{\beta}^{(\mathsf{S})}), \mathbf{e}_j \right\rangle^2 = \|\nabla l(\boldsymbol{\beta}^{(\mathsf{S})})_{\mathsf{T}}\|_2^2,$$

and the first inequality follows. To conclude the proof, note that it holds

$$\begin{aligned} f_{\mathsf{S}}(j) &= l(\boldsymbol{\beta}^{(\mathsf{S} \cup \{j\})}) - l(\boldsymbol{\beta}^{(\mathsf{S})}) \\ &\leq \langle \nabla l(\boldsymbol{\beta}^{(\mathsf{S})}), \boldsymbol{\beta}^{(\mathsf{S} \cup \{j\})} - \boldsymbol{\beta}^{(\mathsf{S})} \rangle - \frac{m}{2} \|\boldsymbol{\beta}^{(\mathsf{S} \cup \{j\})} - \boldsymbol{\beta}^{(\mathsf{S})}\|_2^2 \quad (\text{restricted strong concavity}) \\ &\leq \max_{\mathbf{v}: \mathbf{v}_{(\mathsf{S} \cup \{j\})} = 0} \langle \nabla l(\boldsymbol{\beta}^{(\mathsf{S})}), \mathbf{v} - \boldsymbol{\beta}^{(\mathsf{S})} \rangle - \frac{m}{2} \|\mathbf{v} - \boldsymbol{\beta}^{(\mathsf{S})}\|_2^2. \quad (\text{maximality of } \mathbf{v}) \end{aligned} \quad (11)$$

By setting $\mathbf{v} = \boldsymbol{\beta}^{(\mathsf{S})} + \frac{1}{m} \langle \nabla l(\boldsymbol{\beta}^{(\mathsf{S})}), \mathbf{e}_j \rangle$ in (11) we get

$$\frac{\|\nabla l(\boldsymbol{\beta}^{(\mathsf{S})})_{\mathsf{S}^*}\|_2^2}{2m} \geq l(\boldsymbol{\beta}^{(\mathsf{S} \cup \{j\})}) - l(\boldsymbol{\beta}^{(\mathsf{S})}) = f_{\mathsf{S}}(j).$$

By taking the sum over all $j \in \mathsf{T}$ and rearranging we get

$$\|\nabla l(\boldsymbol{\beta}^{(\mathsf{S})})_{\mathsf{T}}\|_2^2 = \sum_{j \in \mathsf{T}} \frac{\|\nabla l(\boldsymbol{\beta}^{(\mathsf{S})})_{\mathsf{S}^*}\|_2^2}{2m} \geq \sum_{j \in \mathsf{T}} f_{\mathsf{S}}(j),$$

and the claim follows. \square

In our analysis, we also use the following well-known result.

Theorem G.2 (Theorem 1 by [Elenberg et al. \[2018\]](#)). *Suppose the l is (M, m) -(smooth, strongly concave) on Ω_{2k} . Then, for each subsets $\mathsf{S}, \mathsf{T} \subseteq [p]$ of size at most k it holds*

$$2M f_{\mathsf{S}}(\mathsf{T}) \geq \|\nabla l(\boldsymbol{\beta}^{(\mathsf{S})})_{\mathsf{T}}\|_2^2 \geq 2m f_{\mathsf{S}}(\mathsf{T}).$$

By combining Theorem G.1 with Theorem G.2, we get the following corollary.

Corollary G.3. *Suppose the l is (M, m) -(smooth, strongly concave) on Ω_{2k} . Then, for each subsets $\mathsf{S}, \mathsf{T} \subseteq [p]$ of size at most k it holds*

$$\frac{M}{m} f_{\mathsf{S}}(\mathsf{T}) \geq \sum_{j \in \mathsf{T}} f_{\mathsf{S}}(j) \geq \frac{m}{M} f_{\mathsf{S}}(\mathsf{T}).$$

We also make use of the following technical proposition.

Proposition G.4 (Proposition 2.2. by [Nemhauser et al. \[1978\]](#)). *Let $\{x_1, \dots, x_m\}$ and $\{y_1, \dots, y_m\}$ be two sequences of non-negative real numbers. Suppose that it holds $\sum_j x_j \leq 1$ for all $j \in [m]$. Then,*

$$\sum_{j=1}^m y_j \geq \sum_{j=1}^m x_j y_j.$$

G.2 If Algorithm 1 Outputs a Maximum Independent Set

We now prove Theorem 4.1, assuming that Algorithm 1 outputs a solution \mathbf{S} of maximum size, before performing ε^{-1} iterations of the outer While-loop of Algorithm 1. Formally, we prove the following theorem.

Theorem G.5. *Define the function f as in (1), with a log-likelihood function that is (M, m) -smooth, strongly concave on Ω_{2r} . Suppose that Algorithm 1 outputs a solution \mathbf{S}^* such that $\text{Cond}(\mathbf{S}^*) = \emptyset$. Then,*

$$\frac{\mathbb{E} [f(\mathbf{S}^*)]}{\text{OPT}} \geq \frac{1}{1+p} \left(1 - \exp \left\{ -(1-\varepsilon)^2 \frac{m^3}{M^3} \right\} \right),$$

for all $0 < \varepsilon < 1$. Furthermore, in the specific case when \mathcal{I} is r -sparsity constraint over $[n]$, then,

$$\frac{\mathbb{E} [f(\mathbf{S}^*)]}{\text{OPT}} \geq 1 - \exp \left\{ -(1-\varepsilon)^2 \frac{m^2}{M^2} \right\}.$$

The proof of Theorem G.5 is based on the following two additional lemma.

Lemma G.6. *At any point during the optimization process it holds*

$$(1-\varepsilon)^{-1} \frac{rM}{m} t \geq 2m (\text{OPT} - f(\mathbf{S})),$$

with \mathbf{S} the current solution.

Proof. Denote with $\bar{\mathbf{S}}$ a solution of size at most $|\bar{\mathbf{S}}| \leq r$ maximizing $f(\mathbf{S} \cup \bar{\mathbf{S}})$, and let $\mathbf{T} \subseteq [n]$ be a set maximizing $\|\nabla l(\boldsymbol{\beta}^{(\mathbf{S})})_{\mathbf{T}}\|_2^2$, such that $|\mathbf{T}| \leq r$ and $\mathbf{S} \cup \{s\} \in \mathcal{I}$ for all $s \in \mathbf{T}$. Note that it holds

$$(1-\varepsilon)^{-1} \frac{rM}{m} t = \frac{r}{|\mathbf{T}|} \|\nabla l(\boldsymbol{\beta}^{(\mathbf{S})})_{\mathbf{T}}\|_2^2 \geq \|\nabla l(\boldsymbol{\beta}^{(\mathbf{S})})_{\mathbf{T}}\|_2^2, \quad (12)$$

where the first inequality follows by the definition of t , and the second one follows since $|\mathbf{T}| \leq r$. We first prove the claim when t is updated at the beginning of each iteration of the outer While-loop of Algorithm 1. It holds

$$\begin{aligned} & l(\boldsymbol{\beta}^{(\mathbf{S} \cup \bar{\mathbf{S}})}) - l(\boldsymbol{\beta}^{(\mathbf{S})}) \\ & \leq \langle \nabla l(\boldsymbol{\beta}^{(\mathbf{S})}), \boldsymbol{\beta}^{(\mathbf{S} \cup \bar{\mathbf{S}})} - \boldsymbol{\beta}^{(\mathbf{S})} \rangle - \frac{m}{2} \|\boldsymbol{\beta}^{(\mathbf{S} \cup \bar{\mathbf{S}})} - \boldsymbol{\beta}^{(\mathbf{S})}\|_2^2 \quad (\text{restricted strong concavity}) \\ & \leq \max_{\mathbf{v}: \mathbf{v}_{(\mathbf{S} \cup \bar{\mathbf{S}}) \neq 0}} \langle \nabla l(\boldsymbol{\beta}^{(\mathbf{S})}), \mathbf{v} - \boldsymbol{\beta}^{(\mathbf{S})} \rangle - \frac{m}{2} \|\mathbf{v} - \boldsymbol{\beta}^{(\mathbf{S})}\|_2^2. \quad (\text{maximality of } \mathbf{v}) \end{aligned} \quad (13)$$

By setting $\mathbf{v} = \frac{1}{m}\boldsymbol{\beta}^{(\mathcal{S})} + \nabla l(\boldsymbol{\beta}^{(\mathcal{S})})_{\bar{\mathcal{S}}}$ in the inequality above we get

$$\begin{aligned} f(\text{OPT}) - f(\mathcal{S}) &= l(\boldsymbol{\beta}^{(\mathcal{S} \cup \bar{\mathcal{S}})}) - l(\boldsymbol{\beta}^{(\mathcal{S})}) && \text{(maximality of } \bar{\mathcal{S}} \text{ and monotonicity)} \\ &\leq \frac{\|\nabla l(\boldsymbol{\beta}^{(\mathcal{S})})_{\bar{\mathcal{S}}}\|_2^2}{2m} && \text{(substituting } \mathbf{v} \text{ in (13))} \\ &\leq \frac{\|\nabla l(\boldsymbol{\beta}^{(\mathcal{S})})_{\mathcal{T}}\|_2^2}{2m}. && \text{(maximality of } \mathcal{T}) \end{aligned} \quad (14)$$

The claim follows by combining (14) and (12). Suppose now that the current solution \mathcal{S} is updated to \mathcal{S}' during the inner While-loop of Algorithm 1. Then, $f(\mathcal{S}) \geq f(\mathcal{S}')$ due to monotonicity, and the claim holds. \square

Lemma G.7. *At any given time step, suppose that the current solution \mathcal{S} is updated to $\mathcal{S} \cup \{a_1, \dots, a_{j^*}\}$, and define $\mathcal{S}_j = \mathcal{S} \cup \{a_1, \dots, a_j\}$ for all $j \in [j^*]$. Then it holds*

$$\mathbb{E} [f_{\mathcal{S}_{j-1}}(\mathcal{S}_j)] \geq (1 - \varepsilon)^2 \frac{m^2}{rM^2} (\text{OPT} - \mathbb{E} [f(\mathcal{S}_{j-1})]).$$

Proof. Fix all random decisions of Algorithm 1 until the point a_j is added to the current solution. Define the sets $\mathcal{X}_j^{\mathcal{I}} := \{e \in \mathcal{X} \setminus \{a_1, \dots, a_{j-1}\} : \{a_1, \dots, a_{j-1}\} \cup \{a\} \in \mathcal{I}\}$. Then, it holds

$$\begin{aligned} \mathbb{E}_{a_j} [f_{\mathcal{S}_{j-1}}(a_j)] &\geq \frac{1}{2M} \mathbb{E}_{a_j} [\langle \nabla l(\boldsymbol{\beta}^{(\mathcal{S}_{j-1})}), \mathbf{e}_{a_j} \rangle^2] && \text{(Theorem G.1)} \\ &\geq \frac{1}{2M} \mathbb{P}_{a_j}(\langle \nabla l(\boldsymbol{\beta}^{(\mathcal{S}_{j-1})}), \mathbf{e}_{a_j} \rangle^2 \geq t) t && \text{(Markov's inequality)} \end{aligned} \quad (15)$$

By design of the RNDSEQ subroutine, each point a_j is sampled uniformly at random from the set $\mathcal{X}_j^{\mathcal{I}}$, i.e. $a_j \sim \mathcal{U}(\mathcal{X}_j^{\mathcal{I}})$. Hence, it holds $\mathbb{P}_{a_j}(\langle \nabla l(\boldsymbol{\beta}^{(\mathcal{S}_{j-1})}), \mathbf{e}_{a_j} \rangle^2 \geq t) = |\mathcal{X}_{j-1}| / |\mathcal{X}_j^{\mathcal{I}}|$. Combining this observation with (15) we get

$$\begin{aligned} \mathbb{E}_{a_j} [f_{\mathcal{S}_{j-1}}(a_j)] &\geq \frac{1}{2M} \frac{|\mathcal{X}_{j-1}|}{|\mathcal{X}_j^{\mathcal{I}}|} t, && \text{(by the sub-sampling procedure)} \\ &\geq \frac{1}{2M} \frac{|\mathcal{X}_{j-1}|}{|\mathcal{X}|} t, && \text{(since } \mathcal{X}_j^{\mathcal{I}} \subseteq \mathcal{X}) \\ &\geq (1 - \varepsilon) \frac{1}{2M} t && \text{(since } |\mathcal{X}_{j-1}| \geq (1 - \varepsilon) |\mathcal{X}|) \\ &\geq (1 - \varepsilon)^2 \frac{m^2}{rM^2} (\text{OPT} - f(\mathcal{S}_{j-1})), && \text{(by Lemma G.6)} \end{aligned}$$

The claim follows by taking the expectation on both sides. \square

Using this lemma, we can now prove Theorem G.5.

Proof of Theorem G.5. Denote with $\{a_1, \dots, a_j\}$ the first j points added to the solution \mathcal{S}^* , sorted in the order that they were added to it, and define the constant $c := (1 - \varepsilon)^2 m^2 / rM^2$. Using an induction argument on j , we prove that it holds

$$\frac{\mathbb{E} [f(\{a_1, \dots, a_j\})]}{\text{OPT}} \geq (1 - (1 - c)^j). \quad (16)$$

The base case with $j = 0$ holds, due to the non-negativity of the function f . For the inductive case, we have that it holds

$$\begin{aligned}\mathbb{E} [f(\{a_1, \dots, a_j\})] &\geq \mathbb{E} [f(\{a_1, \dots, a_{j-1}\})] + c(\text{OPT} - f(\{a_1, \dots, a_{j-1}\})) \quad (\text{Lemma G.7}) \\ &\geq (1 - c) (1 - (1 - c)^{j-1}) \text{OPT} + c\text{OPT} \quad (\text{by induction}) \\ &\geq (1 - (1 - c)^j) \text{OPT},\end{aligned}$$

and (16) holds. It follows that for $j = |\mathbf{S}^*|$ we have

$$\frac{\mathbb{E} [f(\mathbf{S}^*)]}{\text{OPT}} \geq 1 - (1 - c)^{|\mathbf{S}^*|} \geq 1 - \exp \left\{ -\frac{|\mathbf{S}^*|}{k} c \right\}.$$

In the case of a r -sparsity constraint we have that $|\mathbf{S}^*| = r$, and the claim follows. In the case of a p -system constraint, we have that $|\mathbf{S}^*| \geq pr$, hence

$$\frac{\mathbb{E} [f(\mathbf{S}^*)]}{\text{OPT}} \geq 1 - \exp \left\{ -(1 - \varepsilon)^2 p \frac{m^2}{M^2} \right\} \geq \frac{1}{1 + p} \left(1 - \exp \left\{ -(1 - \varepsilon)^2 \frac{m^3}{M^3} \right\} \right),$$

and the claim also holds. \square

G.3 If Algorithm 1 terminates after ε^{-1} iterations

We now prove Theorem 4.1, assuming that the FASTOMP terminates after ε^{-1} iterations of the outer While-loop of Algorithm 1. Specifically, we prove the following theorem.

Theorem G.8. *Define the function f as in (1), with a log-likelihood function that is (M, m) - (smooth, strongly concave) on Ω_{2r} . Suppose that Algorithm 1 terminates after ε^{-1} iterations of the outer-While loop. Then,*

$$\frac{\mathbb{E} [f(\mathbf{S}^*)]}{\text{OPT}} \geq \frac{1}{1 + p} \left(1 - \exp \left\{ -(1 - \varepsilon)^2 \frac{m^3}{M^3} \right\} \right),$$

for all $0 < \varepsilon < 1$. Furthermore, in the specific case when \mathcal{I} is r -sparsity constraint over $[n]$, then,

$$\frac{\mathbb{E} [f(\mathbf{S}^*)]}{\text{OPT}} \geq 1 - \exp \left\{ -(1 - \varepsilon)^2 \frac{m^2}{M^2} \right\}.$$

In order to prove this theorem, we introduce additional notation. We denote with \mathbf{S}_i the current solution at the beginning of the i -th iteration of the outer While-loop of Algorithm 1. Furthermore, denote with $\bar{\mathbf{S}} \subseteq \mathbf{T} \setminus \mathbf{S}_i$ a feasible set, such that $f(\bar{\mathbf{S}} \cup \mathbf{S}_i) = \text{OPT}$, and denote with a_j the j -th element added to the solution \mathbf{S} . The proof of this theorem is based on the following lemma.

Lemma G.9. *It holds*

$$\frac{M}{m} f_{\mathbf{S}_i}(\mathbf{S}_{i+1}) + \sum_{e \in \bar{\mathbf{S}} \setminus \text{Cond}(\mathbf{S}_i)} \varepsilon f_{\mathbf{S}_i}(e) \geq \varepsilon \frac{m}{M} f_{\mathbf{S}_i}(\bar{\mathbf{S}}).$$

Proof. Fix all random decision of Algorithm 1, up to the $(i + 1)$ -th iteration of the outer While-loop of Algorithm 1. Let $\mathbf{T} \subseteq [n]$ be a set maximizing $\|\nabla l(\boldsymbol{\beta}^{(\mathbf{S}_i)})_{\mathbf{T}}\|_2^2$, such that $|\mathbf{T}| \leq r$

and $\mathbb{T} \subseteq \mathcal{I}$. Due to the assumption on the stopping criterion, it holds $\mathbf{X} = \emptyset$ at the end of iteration i . This means that each point $j \in (\mathbb{T} \setminus \mathcal{S}_i) \cap \text{Cond}(\mathcal{S}_i)$ was discarded at some point during the previous iteration. Denote with \mathbf{U}_j the current solution when j was discarded. Then, it holds

$$(1 - \varepsilon) \frac{2m}{r} \sum_{e \in \mathbb{T}} f_{\mathcal{S}_i}(e) \geq (1 - \varepsilon) \frac{m}{rM} \|\nabla l(\boldsymbol{\beta}^{(\mathcal{S}_i)})_{\mathbb{T}}\|_2^2 = t \quad (17)$$

where the first inequality follows by Theorem G.1, and the second one follows by the definition of t . Since the point j was discarded and since $j \in \text{Cond}(\mathcal{S}_i)$, then it must hold $t \geq \langle \nabla l(\boldsymbol{\beta}_j^{(\mathbf{U}_j)}), \mathbf{e}_j \rangle^2$. Note also that by the RSC/RSM properties of the function l it holds $\langle \nabla l(\boldsymbol{\beta}_j^{(\mathbf{U}_j)}), \mathbf{e}_j \rangle^2 \geq \langle \nabla l(\boldsymbol{\beta}^{(\mathcal{S}_{i+1})}), \mathbf{e}_j \rangle^2$ [Elenberg et al., 2018]. Combining these observations with (17) we get

$$(1 - \varepsilon) \frac{2m}{r} \sum_{e \in \mathbb{T}} f_{\mathcal{S}_i}(e) \geq \langle \nabla l(\boldsymbol{\beta}^{(\mathcal{S}_{i+1})}), \mathbf{e}_j \rangle^2 \geq 2m f_{\mathcal{S}_{i+1}}(j). \quad (18)$$

By taking the sum over all points $j \in (\mathbb{T} \setminus \mathcal{S}_i) \cap \text{Cond}(\mathcal{S}_i)$ and rearranging, we get

$$\begin{aligned} (1 - \varepsilon) \sum_{e \in \mathbb{T}} f_{\mathcal{S}_i}(e) &\geq (1 - \varepsilon) \frac{|(\mathbb{T} \setminus \mathcal{S}_i) \cap \text{Cond}(\mathcal{S}_i)|}{r} \sum_{e \in \mathbb{T}} f_{\mathcal{S}_i}(e) && \text{(by definition of } r) \\ &\geq \sum_{j \in (\mathbb{T} \setminus \mathcal{S}_i) \cap \text{Cond}(\mathcal{S}_i)} \bar{f}_{\mathcal{S}_{i+1}}(j) && \text{(it follows from (18))} \end{aligned} \quad (19)$$

By rearranging (19) we get

$$\begin{aligned} \sum_{e \in \mathcal{S}_{i+1}} f_{\mathcal{S}_i}(e) &\geq \sum_{e \in (\mathbb{T} \setminus \mathcal{S}_i) \cap \text{Cond}(\mathcal{S}_i)} \varepsilon f_{\mathcal{S}_i}(e) \\ &\geq \sum_{e \in \bar{\mathcal{S}} \cap \text{Cond}(\mathcal{S}_i)} \varepsilon f_{\mathcal{S}_i}(e) && \text{(by the definition of } \bar{\mathcal{S}}) \\ &\geq \sum_{e \in \bar{\mathcal{S}}} \varepsilon f_{\mathcal{S}_i}(e) - \sum_{e \in \bar{\mathcal{S}} \setminus \text{Cond}(\mathcal{S}_i)} \varepsilon f_{\mathcal{S}_i}(e) && \text{(by linearity)} \end{aligned} \quad (20)$$

Hence, it holds

$$\begin{aligned} \frac{M}{m} f_{\mathcal{S}_i}(\mathcal{S}_{i+1}) &\geq \sum_{e \in \mathcal{S}_{i+1}} f_{\mathcal{S}_i}(e) && \text{(Corollary G.3)} \\ &\geq \sum_{e \in \bar{\mathcal{S}}} \varepsilon f_{\mathcal{S}_i}(e) - \sum_{e \in \bar{\mathcal{S}} \setminus \text{Cond}(\mathcal{S}_i)} \varepsilon f_{\mathcal{S}_i}(e) && \text{(it follows from (20))} \\ &\geq \varepsilon \frac{m}{M} f_{\mathcal{S}_i}(\bar{\mathcal{S}}) - \sum_{e \in \bar{\mathcal{S}} \setminus \text{Cond}(\mathcal{S}_i)} \varepsilon f_{\mathcal{S}_i}(e). && \text{(Corollary G.3)} \end{aligned}$$

The claim follows by rearranging. \square

In order to continue with the proof, we also use the following lemma.

Lemma G.10. *It holds*

$$pf(\mathbf{S}_i) \geq (1 - \varepsilon) \frac{m^2}{M^2} \sum_{e \in \bar{\mathbf{S}} \setminus \text{Cond}(\mathbf{S}_i)} f_{\mathbf{S}_i}(e).$$

Proof. Fix an index $j \leq |\mathbf{S}_i|$, and fix all random decisions of Algorithm 1 until the point a_j is added to the current solution. For each element a_j , define the set $\mathbf{D}_j := (\bar{\mathbf{S}} \cap \text{Cond}(\{a_1, \dots, a_{j-1}\})) \setminus (\bar{\mathbf{S}} \cap \text{Cond}(\{a_1, \dots, a_j\}))$. Note that these sets consist of all points in $\bar{\mathbf{S}}$ that yield a feasible solution when added to $\{a_1, \dots, a_{j-1}\}$, but that violate side constraints when added to $\{a_1, \dots, a_j\}$. Note also that it holds $\mathbf{D}_1 \cup \dots \cup \mathbf{D}_j = \bar{\mathbf{S}} \setminus \text{Cond}(\{a_1, \dots, a_j\})$. Furthermore, define the sets

$$\mathbf{X}_j^{\mathcal{I}} := \{e \in \mathbf{X} \setminus \{a_1, \dots, a_{j-1}\} : \{a_1, \dots, a_{j-1}\} \cup \{a\} \in \mathcal{I}\}.$$

Then, it holds

$$\begin{aligned} \mathbb{E}_{a_j} \left[f_{\{a_1, \dots, a_{j-1}\}}(a_j) \right] &\geq \frac{1}{2M} \mathbb{E}_{a_j} \left[\langle \nabla l(\boldsymbol{\beta}^{\{a_1, \dots, a_{j-1}\}}), \mathbf{e}_{a_j} \rangle^2 \right] && \text{(by Lemma G.1)} \\ &\geq \frac{1}{2M} \mathbb{P}_{a_j}(\langle \nabla l(\boldsymbol{\beta}^{\{a_1, \dots, a_{j-1}\}}), \mathbf{e}_{a_j} \rangle^2 \geq t) && \text{(by Markov)} \end{aligned} \quad (21)$$

Note that the RNDSEQ subroutine samples points a_j uniformly at random $a_j \sim \mathcal{U}(\mathbf{X}_j^{\mathcal{I}})$. Hence,

$$\mathbb{P}_{a_j}(\langle \nabla l(\boldsymbol{\beta}^{\mathbf{S}_{j-1}}), \mathbf{e}_{a_j} \rangle^2 \geq t) = \frac{|\mathbf{X}_{j-1}|}{|\mathbf{X}_j^{\mathcal{I}}|} \geq \frac{|\mathbf{X}_{j-1}|}{|\mathbf{X}|}, \quad (22)$$

where the last inequality holds, since $\mathbf{X}_j^{\mathcal{I}} \subseteq \mathbf{X}$. Combining this observation with (21) we get

$$\begin{aligned} \mathbb{E}_{a_j} \left[f_{\{a_1, \dots, a_{j-1}\}}(a_j) \right] &\geq \frac{1}{2M} \frac{|\mathbf{X}_j|}{|\mathbf{X}|} t && \text{(it follows by (22))} \\ &\geq \frac{1 - \varepsilon}{2M} t && (|\mathbf{X}_{j-1}| \geq (1 - \varepsilon) |\mathbf{X}|) \\ &= \frac{1 - \varepsilon}{2M^2} \frac{m}{|\mathbf{T}|} \|\nabla l(\boldsymbol{\beta}^{\{a_1, \dots, a_{j-1}\}})_{\mathbf{T}}\|_2^2 && \text{(by the definition of } t) \\ &\geq \frac{1 - \varepsilon}{2M^2} \frac{m}{|\mathbf{D}_j|} \|\nabla l(\boldsymbol{\beta}^{\{a_1, \dots, a_{j-1}\}})_{\mathbf{D}_j}\|_2^2, && (\mathbf{T} \text{ is maximal}) \end{aligned}$$

By taking the expected value on both sides in the chain of inequalities above, we get

$$\sum_{j \leq |\mathbf{S}_i|} |\mathbf{D}_j| \mathbb{E}_{a_j} \left[\bar{f}_{\{a_1, \dots, a_{j-1}\}}(a_j) \right] \geq (1 - \varepsilon) \frac{m}{2M^2} \sum_{j \leq |\mathbf{S}_i|} \mathbb{E} \left[\|\nabla l(\boldsymbol{\beta}^{\mathbf{S}_i})_{\mathbf{D}_j}\|_2^2 \right]. \quad (23)$$

In order to continue with the proof, we give an upper-bound on the size of the sum $\sum_j |\mathbf{D}_j|$. To this end, note that the set \mathbf{S}_i is a maximum independent set over the ground set

$$\mathbf{S}_i \cup (\mathbf{D}_1 \cup \dots \cup \mathbf{D}_{|\mathbf{S}_i|}) = \mathbf{S}_i \cup (\bar{\mathbf{S}} \setminus \text{Cond}(\mathbf{S}_i)).$$

In fact, the set \mathbf{S}_i is independent by definition, and that any point $s \in (\bar{\mathbf{S}} \setminus \text{Cond}(\mathbf{S}_i)) \setminus \mathbf{S}_i$ yields $\mathbf{S}_i \cup \{s\} \notin \mathcal{I}$. Hence \mathbf{S}_i is a maximum independent set as claimed. Note also that

$D_1 \cup \dots \cup D_{|S_i|} \subseteq \bar{S}$ is an independent set, due to the subset-closure of \mathcal{I} . Since \mathcal{I} is a p -system, then it holds

$$|D_1| + \dots + |D_{|S_i|}| = |D_1 \cup \dots \cup D_{|S_i|}| \leq p|S_i|. \quad (24)$$

Hence, it holds

$$\begin{aligned} p \sum_{j \leq |S_i|} \mathbb{E}_{a_j} \left[f_{\{a_1, \dots, a_{j-1}\}}(a_j) \right] &\geq \sum_{j \leq |S_i|} |D_j| \mathbb{E}_{a_j} \left[f_{\{a_1, \dots, a_{j-1}\}}(a_j) \right] && \text{(it follows by (24))} \\ &\geq (1 - \varepsilon) \frac{m}{2M^2} \sum_{j \leq |S_i|} \mathbb{E} \left[\|\nabla l(\beta^{(S_i)})_{D_j}\|_2^2 \right] && \text{(it follows by (23))} \\ &\geq (1 - \varepsilon) \frac{m^2}{M^2} \sum_{e \in D_1 \cup \dots \cup D_{|S_i|}} f_{S_i}(e) && \text{(by Theorem G.1).} \end{aligned}$$

The claim follows since $D_1 \cup \dots \cup D_{|S_i|} = \bar{S} \setminus \text{Cond}(S_i)$. \square

We now have all necessary tools to prove Theorem G.8.

Proof of Theorem G.8. We first prove the claim, assuming that \mathcal{I} is a general p -system. In this case, by combining Lemma G.9 with Lemma G.10 it holds

$$\frac{M}{m} \mathbb{E} [f_{S_i}(S_{i+1})] + \varepsilon p \frac{M^2}{m^2} \mathbb{E} [f(S_i)] \geq \varepsilon(1 - \varepsilon) \frac{m}{M} \mathbb{E} [f_{S_i}(\bar{S})]. \quad (25)$$

To continue, define the constant $c = (1 - \varepsilon)m^3/M^3$. We prove by induction on i that it holds

$$(1 + \varepsilon ip) \mathbb{E} [f(S_i)] \geq (1 - (1 - \varepsilon c)^i) \text{OPT}. \quad (26)$$

The base case with $S_0 = \emptyset$ trivially follows, since the function f is non-negative. For the inductive case, suppose that the claim holds for $\mathbb{E} [f(S_{i-1})]$. Then,

$$\begin{aligned} (1 + \varepsilon ip) \mathbb{E} [f(S_i)] &\geq \mathbb{E} [f(S_i)] + \varepsilon ip \mathbb{E} [f(S_{i-1})] && \text{(by monotonicity)} \\ &\geq \mathbb{E} [f(S_{i-1})] + \varepsilon c \mathbb{E} [f_{S_i}(\bar{S})] + \varepsilon(i-1)p \mathbb{E} [f(S_{i-1})] && \text{(it follows by (25))} \\ &\geq (1 - \varepsilon c) \mathbb{E} [f(S_{i-1})] + \varepsilon c \text{OPT} + \varepsilon(i-1)p \mathbb{E} [f(S_{i-1})] && \text{(by monotonicity)} \\ &\geq (1 - \varepsilon c)(1 - (1 - \varepsilon c)^{i-1}) \text{OPT} + \varepsilon c \text{OPT} && \text{(by induction)} \\ &\geq (1 - (1 - \varepsilon c)^i) \text{OPT}. \end{aligned}$$

Hence, (26) holds. It follows that

$$\begin{aligned} \mathbb{E} [f(S^*)] &= \mathbb{E} [f(S_{\lfloor 1/\varepsilon \rfloor})] && \text{(by the stopping criterion)} \\ &\geq \frac{1}{1 + \varepsilon \lfloor 1/\varepsilon \rfloor p} \left(1 - \left(1 - \varepsilon(1 - \varepsilon) \frac{m^3}{M^3} \right)^{\lfloor 1/\varepsilon \rfloor} \right) \text{OPT} && \text{(it follows by (26))} \\ &\geq \frac{1}{1 + p} \left(1 - \exp \left\{ -(1 - \varepsilon) \frac{m^3}{M^3} \right\} \right) \text{OPT}, \end{aligned}$$

as claimed.

We conclude by proving the claim in the special case that \mathcal{I} is a r -sparsity constraint. Since the algorithm terminates before a solution of size r is found, we have that $\text{Cond}(S_i) = [n] \setminus S_i$ for all iterations i . Hence, $D_1 \cup \dots \cup D_i = \emptyset$ and Lemma G.9 yields

$$\mathbb{E} [f_{S_i}(S_{i+1})] \geq \varepsilon \frac{m^2}{M^2} \mathbb{E} [f_{S_i}(S^*)] .$$

With this inequality, we can use an inductive argument similar to the proof for the general case, and obtain an improved lower-bound on the solution quality. \square

H Proof of Theorem 4.2

We conclude by giving upper-bounds on the run time and adaptivity for Algorithm 1. Recall that the notion of adaptivity is given in Definition D.1. The following theorem holds.

Theorem 4.2. *Algorithm 1 terminates after $\mathcal{O}(\varepsilon^{-2} \log n)$ rounds of calls to the oracle function, and it uses at most $\mathcal{O}(\varepsilon^{-2} r \log n)$ oracle queries. Furthermore, Algorithm 1 requires expected $\mathcal{O}(\varepsilon^{-2} \sqrt{r} \log n)$ independent calls to the oracle for the p -system \mathcal{I} , and the total expected number of calls to the oracle for the p -system \mathcal{I} is $\mathcal{O}(\varepsilon^{-2} nr \log n)$.*

In order to prove this result, we use the following well-known estimate on the number of adaptive rounds of the RNDSEQ sub-routine (see Appendix E).

Theorem H.1 (Theorem 6 by Karp et al. [1988]). *Algorithm E terminates after expected $\mathcal{O}(\sqrt{r})$ steps, with r the rank of the independent system \mathcal{I} .*

Note that this theorem implies that the number of adaptive rounds of the independence oracle for Algorithm E is $\mathcal{O}(\sqrt{r})$ in expected value. In fact, in each step of Algorithm E, queries to the independence oracle can be performed in parallel. Using this result, we can now prove Theorem 4.2.

Proof of Theorem 4.2. We first give upper-bounds for the oracle function that accesses $\nabla l(\cdot)$. To this end, observe that there are two While-loops in Algorithm 1. The outer while-loop terminates after at most ε^{-1} iteration. The inner While-loop terminates after $\mathcal{O}(\varepsilon^{-1} \log n)$ iterations, since at each iteration the size of X decreases at least of a multiplicative factor of $1 - \varepsilon$. Hence, the rounds of calls to the oracle function is $\mathcal{O}(\varepsilon^{-2} \log n)$. Furthermore, at each iteration of the inner While-loop, at most r parallel calls to $\nabla l(\cdot)$ are performed. It follows that the total number of oracle calls is $\mathcal{O}(\varepsilon^{-2} r \log n)$.

We now estimate the number of adaptive rounds and run time for the calls to the independence oracle. To this end, note that is oracle is called by the RNDSEQ sub-routine, and it is also evaluated nr times in parallel during the inner While-loop of Algorithm 1. From Theorem H.1 it follows that the number of adaptive rounds is $\mathcal{O}(\varepsilon^{-2} \sqrt{r} \log n)$, and that the total number of calls to the oracle function is $\mathcal{O}(\varepsilon^{-2} nr \log n)$ as claimed. \square

I Feature selection on Non-volatile Memory (NVM)

The emergence of CPU-attached persistent memory technology, such as Intel's Optane Non-Volatile Memory (NVM), has opened opportunities for running large datasets on a single server.

A single server may have up to 6TB of NVM, that can be accessed through the memory bus. The use of NVM compared to DRAM is beneficial for two reasons. First, due to its large capacity, it is suitable to handle large datasets. Secondly, as NVM is non-volatile, it can easily support fault-tolerance and recovery of the computation when a server crashes. Furthermore, even though NVM latency is slightly slower than DRAM, it has much better latency than SSD of orders of magnitude.

For running our Python code on NVM, we used MCAS [Waddington et al., 2021a,b] (Memory Centric Active Storage), which is an advanced client-server in-memory object store designed from the ground up to leverage persistent memory. MCAS supports “pushdown” operations on the client-side which are termed Active Data Objects (ADO), and has a Python plugin that allows zero-copy for Numpy data access. Our benchmark for evaluating feature selection with persistent memory is done by integrating the algorithms to MCAS as an ADO using the above mentioned Python plugin. We use an Intel Xeon Gold 6248 server with 80 CPUs at 2.50GHz. The server is equipped with 384GB DDR4 DRAM and 1512GB Optane DC.

By experimenting with the same datasets with the `SDSOMP` and `FASTOMP`, we observed no significant variation in run time compared to our DRAM implementation. This is due to the fact that the time required to copy data from NVM to DRAM is negligible, in comparison with the compute time of training phases. However, when performing tasks on the DRAM that use more memory than the DRAM capacity, we might observe a significant decrease in the performance. For this reason, in future work we intend to investigate the trade-off between training time on different media (NVM versus DRAM and SSD) of tasks that use more memory than the DRAM capacity.

J The ProPublica COMPAS Dataset

The ProPublica COMPAS dataset was constructed in 2016, using data of defendants from Broward County, FL, who had been arrested in 2013 or 2014 and assessed with the COMPAS risk screening system. ProPublica then collected data on future arrests for these defendants through the end of March 2016, in order to study how the COMPAS score predicted recidivism [Angwin et al., 2016]. Based on its analysis, ProPublica concluded that the COMPAS risk score was racially biased [Berk et al., 2021].

The ProPublica COMPAS data has become one of the key bench-marking datasets for testing algorithmic fairness definitions and procedures [Chouldechova, 2017, Corbett-Davies et al., 2017, Corbett-Davies et al., 2017, Cowgill and Tucker, 2019, Rudin et al., 2018, Zafar et al., 2017b,c]. However, Bao et al. [2021a] notes that there are inaccuracies in the COMPAS dataset. For instance, COMPAS race categories lack Native Hawaiian or Other Pacific Islander, and it redefines Hispanic as race instead of ethnicity.

This dataset consists of the following features: “number of prior criminal offenses”, “arrest charge description”, “charge degree”, “number of juvenile felony offenses”, “juvenile misdemeanor offenses”, “other juvenile offenses”, “age”, “sex” and “race” of the defendant. The dataset also contains information on whether the defendant recidivated or not.