Master Degree in Data Science

Randomized Numerical Linear Algebra For Generalized Linear Models with Big Datasets

Robert Tjarko Lange

Directors:
Prof. Ioannis Kosmidis (UCL)
Prof. Omiros Paspapiliopoulos (UPF)

June 2017

ABSTRACT IN ENGLISH:

Data scientific questions face the fundamental trade-off between complexity, generalizability and computational feasibility. The need for quick estimation and evaluation of a vast amount of statistical models has given rise to a plethora of new and innovative algorithms in the field of randomized numerical linear algebra (RandNLA). They intend to decrease effective running time by approximating exact solutions. One commonly allows for some e-"slack" in order to make use of powerful subspace embedding ideas such as the Johnson-Lindenstrauss transform (JLT). In this way, one is able to significantly reduce the dimensionality of the problem, while preserving a substantial amount of the original structure. Petros Drineas and Michael Mahoney have been applying these ideas to a range of problems such as solving linear systems of equations (over-and under-constrained), matrix completion and low-rank matrix approximation.

ABSTRACT IN CATALAN:

Les qüestions científiques de dades afronten el compromís fonamental entre complexitat, generalizabilitat i viabilitat computacional. La necessitat d'una ràpida estimació i avaluació d'una gran quantitat de models estadístics ha donat lloc a una infinitat d'algorismes nous i innovadors en el camp de l'àlgebra lineal numèrica aleatòria (RandNLA). Tenen la intenció de disminuir el temps d'execució efectiu mitjançant l'aproximació de solucions exactes. Un comunament permet una mica de "descentralització" per a fer ús d'idees incrustantes potents de subespacio com la transformació Johnson-Lindenstrauss (JLT). D'aquesta manera, es pot reduir significativament la dimensionalitat del problema, tot conservant una quantitat substancial de l'estructura original. Petros Drineas i Michael Mahoney han estat aplicant aquestes idees a una sèrie de problemes com la solució de sistemes lineals d'equacions (sobre i subterfugis), la finalització de la matriu i la aproximació a la matriu de baix rang.

Robert Tjarko Lange
Student no. 138491

Randomized Numerical Linear Algebra For Generalized Linear Models with Big Datasets

Master Thesis

Supervisors:
Prof. Ioannis Kosmidis (UCL)
Prof. Omiros Paspapiliopoulos (UPF)

Submitted for the Master examination in
Data Science
Barcelona Graduate School of Economics

Barcelona, June 2017

Contents

List of Figures

Symbol directory

Symbols and abbreviations refer to the following, except where stated differently.

Abbreviations

RandNLA	Randomized Numerical Linear Algebra
GLM	Generalized Linear Model
LS	Least Squares
RSS	Residual Sum of Squares
JLT	Johnson-Lindenstrauss Transform
FJLT	Fast Johnson-Lindenstrauss Transform
WLS	Weighted Least Squares
IWLS	Iterative Weighted Least Squares

Essential Symbols

ϵ	Degree of approx. error	$\Pi \in \mathbb{R}^{r \times n}$	Sketching matrix
$\delta \in [0,1]$	Stopping Criterion IWLS	$\Pi_{JLT} \in \mathbb{R}^{r \times n}$	JLT
$y^{\perp} \in \mathbb{R}^n$	Residuals	$\Pi_{FJLT} \in \mathbb{R}^{r \times n}$	FJLT
$y \in \mathbb{R}^n$	Target vector	$\tilde{y} \in \mathbb{R}^r$	Sketched target vector
$X \in \mathbb{R}^{n \times d}$	Design matrix	$\tilde{X} \in \mathbb{R}^{r \times n}$	Sketched design matrix
$z \in \mathbb{R}^n$	Working variate	$\tilde{z} \in \mathbb{R}^r$	Sketched working variate
$W \in \mathbb{R}^{n \times n}$	Weighting matrix	$\tilde{W} \in \mathbb{R}^{r \times r}$	Sketched weighting matrix
$\beta \in \mathbb{R}^d$	True data-gen. parameter		
$\beta_{LS} \in \mathbb{R}^d$	Least Squares solution	$\tilde{\beta} \in \mathbb{R}^d$	Randomized LS solution
$\beta_{GLM} \in \mathbb{R}^d$	IWLS GLM solution	$\tilde{\beta}_{RandGLM} \in \mathbb{R}^d$	Randomized GLM solution
$l_i \in [0,1]$	Leverage score of obs. i		
$\tilde{l}_i \in [0,1]$	Approx. lev. score (FJLT and JLT)	$\hat{l}_i \in [0,1]$	Approx. lev. score (FJLT)

Landau Symbols

$r = o(f)$	r grows slower than f	$\iff \lim_{x \to x'} \left\lvert \frac{r(x)}{f(x)} \right\rvert = 0$
$r = O(f)$	r grows not signif. slower than f	$\iff \lim_{x \to x'} \sup \left\lvert \frac{r(x)}{f(x)} \right\rvert < \infty$
$r = \Theta(f)$	r grows exactly like f	$\iff 0 < \lim_{x \to x'} \inf \left\lvert \frac{r(x)}{f(x)} \right\rvert \leq \lim_{x \to x'} \sup \left\lvert \frac{r(x)}{f(x)} \right\rvert < \infty$
$r = \Omega(f)$	r doesn't always grow slower than f	$\iff \lim_{x \to x'} \inf \left\lvert \frac{r(x)}{f(x)} \right\rvert > 0$

1 Introduction

Data scientific questions face the fundamental trade-off between complexity, generalizability and computational feasibility. The need for quick estimation and evaluation of a vast amount of statistical models has given rise to a plethora of new and innovative algorithms in the field of randomized numerical linear algebra (RandNLA). They intend to decrease effective running time by approximating exact solutions. One commonly allows for some ϵ-"slack" in order to make use of powerful subspace embedding ideas such as the Johnson-Lindenstrauss transform (JLT). In this way, one is able to significantly reduce the dimensionality of the problem, while preserving a substantial amount of the original structure. Petros Drineas and Michael Mahoney have been applying these ideas to a range of problems such as solving linear systems of equations (over- and under-constrained), matrix completion and low-rank matrix approximation (see e.g. Drineas et al., 2006a,b,c).

This thesis is going to focus on the branch of RandNLA that approximates a statistical estimator. Common statistical methods obtain estimates of parameters which underly the data-generating process of the observed data. Randomized algorithms, on the other hand, are not directly interested in the underlying parameter itself. Instead they speed up the computation of the estimator by approximating it. This way one obtains a fast estimate of the original estimator.

We contribute in two ways to the existing literature: First, we strengthen the statistical analysis of the resulting randomized parameter estimator. RandNLA is a novel field deeply rooted in computer science. Most research endeavors solely focus on the quality of approximation and completely neglect the underlying statistical motivation of obtaining estimators: Inferring causal relationships based on empirical observations. Raskutti and Mahoney (2014) provide a first approach to unify the algorithmic and the statistical perspective on sketching exact estimator solutions. They show that the statistical properties of the randomized sketch depend heavily on the underlying objectives of the approximation. Ma et al. (2015), on the other hand, introduce regularization ideas to the leverage score sampling approach and show, that one can obtain similar bias-variance results as in the standard penalized likelihood setting. By introducing a small amount of bias to the "estimate of the estimator", one can obtain a large decrease in terms of variance. In the following analysis we inspect fundamental assumptions from a statistical point of view and introduce powerful ideas such as influence scores. This way we are able to reduce the variance of the quality of approximation. Second, we analyze and extend a framework for the application of randomized sampling to generalized linear models (GLM). In practice, GLMs can be conveniently estimated using iterative weighted least squares (IWLS). Hence, it is worth investigating how the standard random sampling scheme has to be adapted to an iterative procedure, which at each iteration works with a new approximation. Making use of different measures of self-sensitivity, we analyze and extend a randomized algorithm for estimating GLMs in the over-constrained setting ($n >> d$), first introduced by Jia (2014). We show that one is able to translate the original least squares (LS) quality of approximation result by

Drineas et al. (2011) into the iterative context of GLMs.
The following thesis is structured in the following way: Chapter 2 outlines the general problem of approximating the LS solution. The dimensionality reduction, which lies at the heart of complexity improvements, relies heavily on the notion of subspace embedding. In order to establish concentration results for the resulting estimator one has to formalize the extend of the geometrical "disruption" induced by this subspace embedding. This notion can be expressed in two "structural requirements". Afterwards, we will outline a first slow random sampling estimator for practical purposes, which was introduced by Mahoney (2016). Chapter 3 extends this estimation procedure by introducing a fast algorithm that makes use of a fast subspace embedding. It first approximates the statistical leverage scores of the input matrix and then down-samples according to a normalized importance sampling distribution. The lower-dimensional problem is then solved using ordinary techniques. This leverage scores-based technique is also known as algorithmic leveraging. We review the fundamental quality of approximation result by Drineas et al. (2011) and show simulation results for different data-generating processes. Furthermore, we discuss advantages and disadvantages of using leverage scores as a measure of "influence" on the solution fit. Chapter 4 generalizes this notion to the setting of GLMs. After discussing the maximum-likelihood estimation of GLMs via IWLS, we outline the problem formulation of a randomized estimator. We show that one can treat each iteration as a simple weighted normal equation problem which introduces some amount of approximation error. Assuming that these errors introduced at every iteration are independent, we are able to show that one can obtain an unbiased GLM estimator. Chapter 5 combines the previous results in a random sampling algorithm for GLMs. Using three different measures of influence on the solution of the likelihood equations, we introduce a randomized algorithm for IWLS. Simulating Logit estimators, reveals that one can obtain similar performing estimators as in the plain least squares case. Furthermore, we analyze how the randomized estimator behaves over the course of the iterations of the algorithm. Finally, Chapter 6 discusses possible extensions and concludes.

2 Problem Formulation: LS

The following chapter introduces the problem of sketching an exact solution to the Least Squares problem. First, the general notion of a sketching matrix is introduced. After differentiating between different approximation objectives, I discuss the structural requirements of a sketching matrix that allow us to obtain a concentration result. Finally, I describe a first slow random sampling algorithm.

2.1 Approximating Least Squares: Concept and Notation

The standard LS problem involves solving the following convex optimization problem

$$RSS_{min} = \min_{\beta \in \mathbb{R}^d} ||y - X\beta||_2^2 = ||y^{\perp}||_2^2, \tag{1}$$

where $y \in \mathbb{R}^n$, $X \in \mathbb{R}^{n \times d}$, $\beta \in \mathbb{R}^d$. $y^{\perp}$ denotes the residuals, which are the part of y that cannot be obtained by a linear combination of the column vectors in X. One minimizes the residual sum of squares (RSS) by orthogonally projecting y onto the column span of X. Taking the gradient with respect to β, one easily obtains the closed-form solution to the LS problem by solving the system of normal equations

$$(X^T X)\beta_{LS} = X^T y. \tag{2}$$

Usually, this system is solved using either Cholesky, QR or singular value decomposition (SVD) (see e.g. Golub and Van Loan, 2012). The theoretical complexity of all methods is $O(nd^2)$, while in practice there are different constant factors depending on how well-conditioned X is. Randomized algorithms for LS reduce this theoretical complexity to $O(ndlog(r))$ where $log(r) \leq d$.[1] But this decrease in running time comes at a price: We can only obtain an ϵ-approximation to the actual solution β_{LS}. Instead of working with the actual full $n \times d$ matrix X and n-dimensional response vector y, we want to preprocess and construct sketches, $\tilde{X} \in R^{r \times d}$ and $\tilde{y} \in R^r$, of X and y respectively. This sketching procedure drastically reduces the amount of rows to $r << n$. While doing this, we intend to maintain as much structural information of X and y for the relevant fit of the estimator (e.g. column span of X) as is possible. We can write the new optimization problem as

$$\begin{aligned} \widetilde{RSS}_{min} &= \min_{\beta \in \mathbb{R}^d} ||\Pi(y - X\beta)||_2^2 \,, \\ &= \min_{\beta \in \mathbb{R}^d} (y - X\beta)^T \Pi^T \Pi (y - X\beta) \,, \end{aligned} \tag{3}$$

where $\Pi \in \mathbb{R}^{r \times n}$ denotes a $(r \times n)$ sketching matrix (Drineas et al., 2011, p. 6). The original least squares problem now becomes a weighted one with weight matrix $\Pi^T \Pi$. After this preprocessing step, we solve the reduced system and obtain the vector $\tilde{\beta}_{LS}$. There are two requirements for a sufficient construction (Mahoney, 2016, p. 56):

Solution Certificate: $\tilde{\beta}_{LS} \approx \beta_{LS}$,

Optimized Objective Function: $\widetilde{RSS}_{min} \approx RSS_{min}$.

The second requirement is clearly weaker and implied by the first. Given convexity of the RSS, an approximation that is close to the optimal β_{LS}, is also going to give an $RSS(\tilde{\beta}_{LS})$ value that is close to the desired RSS_{min}. On the other hand, if the RSS is only

[1] Choosing r such that $log(r) = d$ would yield the same theoretical complexity as solving the LS problem exactly, while introducing an approximation error. Hence, in practice we require $log(r) << d << n$.

weakly convex, in the sense that the function $RSS(\beta) = ||y - X\beta||_2^2$ has flat regions, then it is possible to obtain a $\tilde{\beta}_{LS} = RSS^{-1}(\widetilde{RSS}_{min})$ that might not be close to the the desired $\tilde{\beta}_{LS}$. For a formal derivation of the above implication see remark 1 in appendix A. This situation comes up when the optimal solution lies in a very flat region of the RSS. Hence, first-order optimization methods, which only make use of the gradient information, would fail due to a diminishing gradient. From a statistical standpoint we are estimating an estimator and we exhibit large uncertainty in the "identification". Hence, in this case $\tilde{\beta}$ has large variance. This highlights the first difference between the algorithmic and the statistical perspective. From a theoretical computer science point of view an approximation in terms of the RSS might be a "good" approximation to the d-dimensional solution vector. As statisticians, on the other hand, we are also interested in inferring or approximating the true data-generating process. Hence, the solution certificate is of greater importance.

2.2 Structural Requirements for a Good Approximation

But how can we formalize the notion of a "good" sketching? A particular Π that fulfills both of the desired objectives can be found in a randomized version of the identity matrix. A matrix Π with

$$\mathbb{E}(\Pi^T\Pi) = I_n$$

leads to an unbiased estimator for the RSS_{min}, since

$$\mathbb{E}(\widetilde{RSS}_{min}) = \min_{\beta \in \mathbb{R}^d} (y - X\beta)^T \mathbb{E}(\Pi^T\Pi)(y - X\beta) = RSS_{min} \, .$$

Assume that $rank(X) = d$ and consider the thin singular value decomposition of X:

$$X = U^{(X)}\Sigma V^T \, ,$$

where $U^{(X)} \in \mathbb{R}^{n \times d}$ and $V \in \mathbb{R}^{d \times d}$ are the matrices of left- and right-singular vectors, respectively and $\Sigma \in \mathbb{R}^{d \times d}$ denotes the diagonal matrix whose elements are the non-zero singular values of X. Furthermore, let $\sigma(X)$ and $\lambda(X)$ denote a singular value and an eigenvalue of the matrix X.
Mahoney (2016, p. 57) states the following two structural conditions which are needed to obtain concentration results for the resulting estimator:

$$\textbf{Rotation/Isometry Requirement: } \sigma_{min}^2(\Pi U^{(X)}) = \lambda_{min}(U^{(X)T}\Pi^T\Pi U^{(X)}) \geq \frac{1}{\sqrt{2}} \, , \quad (4)$$

$$\textbf{Subspace Embedding: } ||U^{(X)}\Pi^T\Pi y^{\perp}||_2^2 \leq \frac{\epsilon}{2} RSS_{min} \, . \quad (5)$$

The first condition demands a lower bound on the singular values of $\Pi U^{(X)}$, where $U^{(X)}$ denotes the orthonormal basis of $span(X)$, which can be obtained from SVD or QR. This

captures the notion of Π being an approximate isometry/rotation after reducing the dimensionality to r (Mahoney, 2016, p. 57). If Π was an exact rotation matrix, it would be orthogonal and hence $\Pi^T = \Pi^{-1}$ would hold. Furthermore, if $\min\{r, d\} = d$ and since $U^{(X)}$ is also orthogonal by construction, the eigenvalue condition would always be fulfilled, since $\lambda_{min}(I_d) = \lambda_{max}(I_d) = 1$. If, on the other hand, $\min\{r, d\} = r$, then some of the eigenvalues are going to be zero and the lower bound of $\frac{1}{\sqrt{2}}$ is going to be violated. Hence, this requirement also implicitly imposes that $r > d$ and in order for this randomized approach to make sense we need a highly unbalanced problem, where $n >> r > d$.
One main characteristic of LS is that it orthogonally projects y onto the span of X in the sense that the residuals $y^{\perp}$ stand orthogonally on $span(X)$. We want to approximately maintain this property even after translating the problem into a much lower dimensional subspace. This is what the subspace embedding condition captures: $\Pi y^{\perp}$ has to be approximately orthogonal to $\Pi U^{(X)}$.
If Π was in fact constructed so that $\Pi^T\Pi$ was exactly equal to I_n ($r = n$), then both of the above requirements would be trivially fulfilled. Furthermore, it is also easy to show that a matrix with random independent Gaussian entries fulfills the desired property of $\mathbb{E}(\Pi^T\Pi) = I_n$. For this case, it is possible to establish a concentration result, that ensures that both requirements are going to be fulfilled with high probability if r is appropriately large (see e.g. Mahoney, 2016). Given the two structural requirements, we are now able to introduce the first slow random sampling algorithm for LS.

2.3 A Slow Random Sampling Algorithm for LS

Another way to interpret Π is as a random sampling matrix (Drineas et al., 2011). It subsamples r rows from the original design matrix X and y. In this case each row of Π contains one non-zero entry, which represents a rescaled sampled row, where sampling is done with replacement. While doing so, we want $\Pi X \in \mathbb{R}^{r \times d}$ to capture as much "fit-relevant structure" of $X \in \mathbb{R}^{n \times d}$ as possible. Intuitively, this means that we want to sample observations that highly influence the original LS fit more often. A natural measure of influence of each data point on the in-sample fit of itself is the statistical leverage score. It is defined in the following way:

Definition 1 (Statistical Leverage Score (Mahoney, 2016, p. 80))**.** *Let $X \in \mathbb{R}^{n \times d}$ and its SVD/QR decomposition be denoted by $X = U^{(X)}\Sigma V^T = Q^{(X)}R$. Furthermore, let e_i denote a standard basis vector acting as an indicator for row i. Also let $X^{\dagger} = V\Sigma^{-1}U^{(X)T}$ denote the generalized Moore-Penrose inverse. The leverage scores of X are then defined to be*

$$
\begin{aligned}
l_i &= \left(X(X^TX)^{-1}X^T\right)_{ii} \\
&= ||U_i^{(X)}||_2^2 = ||Q_i^{(X)}||_2^2 \\
&= ||e_i^TU^{(X)}||_2^2 = ||e_iU^{(X)}U^{(X)T}||_2^2 \\
&= ||e_iU^{(X)}\Sigma V^TV\Sigma^{-1}U^{(X)T}||_2^2 = ||e_iXX^{\dagger}||_2^2 \,.
\end{aligned}
$$

The leverage score, l_i, specifies how much a specific data point pulls the point mass from the center of gravity. Hence, it describes how much each data point pulls the linear model fit towards itself (Huber and Ronchetti, 2009, p. 154ff.). A data point with leverage 1 "occupies" one complete parameter/degree of freedom and the LS hyperplane goes straight through it. The construction of the leverage scores can be done in several different ways and does not depend on the response vector y (see section 3.4 for alternative measures) . They can either be extracted from the diagonal elements of the hat matrix or by computing the euclidean row norm of the orthogonal matrices $U^{(X)}$ and $Q^{(X)}$ from the SVD or QR, respectively.
A simple and slow randomized algorithm for LS is described by Mahoney (2016, p. 65) as follows:

Algorithm 1 Mahoney (2016, p. 65) - "Slow" Random Sampling Algorithm for LS

Input: LS problem with $X \in \mathbb{R}^{n \times d}$, $y \in \mathbb{R}^n$ and an ϵ-level of approximation
Output: Approximate LS solution, $\tilde{\beta}$

1: Compute the importance sampling distribution $p_i = \frac{1}{d}||U^{(X)}_{(i)}||^2_2 = \frac{1}{d}||Q^{(X)}_{(i)}||^2_2$, $\forall i = 1, ..., n$, where $U^{(X)}$ and $Q^{(X)}$ are the orthogonal matrices from either the SVD or the QR.
2: Randomly sample $r = O(\frac{dlog(d)}{\epsilon})$ rows of X and y. Rescale them by multiplying by $\frac{1}{\sqrt{rp_i}}$ and form $\tilde{X} \in R^{r \times d}, \tilde{y} \in R^r$.
3: Solve $(\tilde{X}'\tilde{X})\tilde{\beta} = \tilde{X}'\tilde{y}$ using any of the above mentioned methods.
4: **return** $\tilde{\beta}$, an ϵ-approximation of β_{LS}.

Algorithm 1 constructs a sampling distribution by computing the *exact* leverage scores. The distribution is normalized by dividing the leverage score by $d = ||U^{(X)}||^2_F = \sum_{i=1}^n l_i$. Observations with a high leverage score are sampled more frequently than observation with smaller leverage. Therefore, the algorithm exploits the well-known weakness of the LS estimator of being highly sensitive to outliers. The bottleneck of this algorithm lies in computing the SVD of QR of X which is already as expensive as solving the full problem in the first place. Hence, we need a fast way to compute an approximation to the leverage scores. Slow versions of an algorithmic leveraging estimator for LS, as the one introduced by Mahoney (2016, p. 65) suffer from this *exact* computation of the leverage scores. They take at least $O(nd^2)$ time which is slower than the desired $O(ndlog(r))$ time. The algorithms presented in the next section "widen" this bottleneck by making use of another approximation: They allow for ϵ-slack in computing the leverage scores and afterwards importance sample with respect to an approximate distribution.

3 Fast Algorithmic Leveraging Estimator for Least Squares

After having discussed a first slow algorithmic leveraging approach to approximating the LS solution, we now turn to a fast random sampling approach for Least Squares.

Therefore, we first discuss the notion of random subspace embeddings. This ultimately allows us to obtain a fast approximation to the leverage scores. After establishing the theoretical complexity of such an algorithm, we discuss the fundamental quality of approximation result by Drineas et al. (2011) and show simulation results. Finally, we dig into other measures of influence and discuss advantages and disadvantages of those.

3.1 Fast Subspace Johnson-Lindenstrauss Transform

Given the structural requirements stated in equations (4) and (5), how can we construct a matrix $\Pi \in \mathbb{R}^{r \times n}$ that speeds up the desired computations? All of the following fast constructions rely heavily on two fundamental subspace embedding results: The standard Johnson-Lindenstrauss transform (JLT) and the Fast Johnson-Lindenstrauss transform (FJLT),

Lemma 2 (Johnson-Lindenstrauss Lemma)**.** *Given d points $\{x_i\}_{i=1}^{d}$, each of which is in $\mathbb{R}^n$, there exists a linear function $f : \mathbb{R}^n \to \mathbb{R}^r$ such that $\forall i, j \in \{1, ..., d\}$ the mapped vectors fulfill*

$$(1-\epsilon)||x_i - x_j||_2^2 \leq ||f(x_i) - f(x_j)||_2^2 \leq (1+\epsilon)||x_i - x_j||_2^2$$

Such a mapping can be constructed by $f(x) = \Pi x$ where $\Pi = (\Pi_{ij})_{r \times n}$ and $\Pi_{ij} \sim \frac{1}{\sqrt{r}} N(0,1)$. Hence, we have that

$$(1-\epsilon) \leq \frac{||\Pi(x_i - x_j)||_2^2}{||x_i - x_j||_2^2} \leq (1+\epsilon).$$

Mappings that fulfill the Johnson-Lindenstrauss lemma are called Johnson-Lindenstrauss transforms (JLT) and can be easily constructed by either sampling scaled Gaussians (Frankl and Maehara, 1988) or by simply using ± 1 coin flips (Achlioptas, 2003). A Gaussian-based proof of the lemma can be found in appendix A and follows from applying Chernoff bounding methods and a simple union bound argument. For the Gaussian construction it holds that with probability $1 - \delta$ and choosing $r \geq \frac{4}{\epsilon^2} log \frac{d^2}{2\delta}$, the pairwise distances of all d points are going to be ϵ-approximately maintained in the reduced $r << n$ dimensional space. Furthermore, a matrix Π that fulfills the JLT is said to be a "random projection". But this transformation does not speed up any LS computations, because just the matrix multiplication ΠX is going to take $O(ndr)$ time.

Running time improvements can be obtained by making use of so-called structured random projections (Mahoney, 2016, p. 67), which are also called Fast Johnson-Lindenstrauss transforms (FJLT). At their core lies a preprocessing of the matrix X, which transforms its eigenvalue/singular value concentration. This allows us to either use sparse projection methods or uniform sampling methods to construct efficient sketches. The increase in speed comes from the fact that the FJLT is structured, so that fast Fourier/Walsh-Hadamard transforms can be applied to decrease the ΠX computation from $O(ndr)$ to $O(ndlog(r))$ (Mahoney, 2016, p. 67). Unlike the standard JLT, structured random

projections are not defined in terms of a discrete number of points, $x_1, ..., x_d$, but for a complete subspace. This means that one generalizes the JLT property by "putting an ϵ-net on the unit ball" (Mahoney, 2016, p. 68). The formal definition is the following:

Definition 3 (Fast Subspace Johnson-Lindenstrauss Transform - FJLT - Mahoney (2016, p. 68))**.** *Let $\epsilon > 0$ and $U \in \mathbb{R}^{n \times d}$ be an orthogonal matrix, viewed as d vectors in $\mathbb{R}^n$. A ϵ-FJLT or subspace Johnson-Lindenstrauss transform maps vectors from $\mathbb{R}^n \to R^r$ such that the orthogonality of U is preserved. $\Pi \in \mathbb{R}^{r \times n}$ is called an ϵ-FJLT if*

- **Orthogonality preservation:** $||I_d - U^T \Pi^T \Pi U||_2 \leq \epsilon$
- **Fast running time:** $\forall X \in \mathbb{R}^{n \times d}$ we can compute ΠX in $O(ndlog(r))$ time.

The orthogonality preservation condition states that the transformation does not disturb the orthogonality of $U^{(X)}$ by too much. The information captured in the column space of X survives the rotation and embedding done by Π. Ailon and Chazelle (2006) first proposed a Hadamard-based construction in the context of nearest-neighbor-search. In what follows I will make use of the adapted algorithm by Drineas et al. (2012), which computes fast leverage score approximations. It relies on fast Fourier/Randomized Hadamard preprocessing of the input matrix. They define Π in the following way:

$$\Pi_{FJLT} = PHD \ .$$

The individual components are as follows (Mahoney, 2016, p. 68f.):
$P \in \mathbb{R}^{r \times n}$ denotes a sparse Johnson-Lindenstrauss matrix or Uniform sampling matrix. The sparse Johnson-Lindenstruas construction is the following

$$P_{ij} = \begin{cases} 0 & \text{with prob. } 1 - q \\ N(0, q^{-1}) & \text{with prob. } q, \end{cases}$$

where $q = min\{1, \Theta(\frac{log^2(n)}{d})\}$. It alone cannot be used as an FJLT, since for a "spiky" vector x (imagine a column vector of the identity matrix), the random variable $||Px||$ would have large variance due to the increased weight given to a small number of random Gaussians/coin flips (Ailon and Chazelle, 2006, p. 307). This ultimately, prevents us from obtaining the necessary concentration.

$H_n \in \mathbb{R}^{n \times n}$ denotes a discrete Fourier Transform or normalized Hadamard matrix. It is structured in away such that Fast Walsh-Hadamard methods can be applied to compute them quickly (Mahoney, 2016, 69). Setting $\tilde{H}_0 = \tilde{H}_1 = 1$ the Hadamard matrix is defined for all n that are a power of two in a recursive fashion:[2]

$$\tilde{H}_{2m} = \begin{bmatrix} \tilde{H}_m & \tilde{H}_m \\ \tilde{H}_m & -\tilde{H}_m \end{bmatrix} = \tilde{H}_2 \otimes \tilde{H}_{m-2} \text{ where } \tilde{H}_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} .$$

[2]In what follows we will assume that n is a power of two and $n >> d$. This can be generalized by introducing the notion of best-rank-k-approximation (See e.g. Drineas et al. (2011, p. 3458 f.)).

The normalized Hadamard matrix is then defined as $H_n = \frac{\tilde{H}_n}{\sqrt{n}}$.

$D \in \mathbb{R}^{n \times n}$ takes values ± 1 with probability $\frac{1}{2}$ each and randomly spreads the vectors in the space.
The matrix product HD is called a randomized Hadamard transform (RHT) and fulfills a very important task (Drineas et al., 2012, p. 3449): It flattens spiky vectors (see theorem 1 in appendix A). Furthermore, the preprocessing matrix is orthogonal. Thereby, only the eigenvector/singular vector mass is spread out. This ultimately uniformizes the leverage scores and hence both uniform sampling as well as sparse projections are going to perform well (Mahoney, 2016, p. 67). A geometrical example of such a transformation can be found in figure 1. The first column plots the individual data points in red, $X \in \mathbb{R}^{128 \times 2}$, and their respective transformations, $HDX \in \mathbb{R}^{128 \times 2}$, in blue. Following Ma et al. (2015, p. 15), the three data-generating processes I consider are:

a) Multivariate normal distribution with mean 0 and covariance matrix $\Sigma_{ij} = 2 \times 0.5^{|i-j|}$. This distribution represents the case where leverage scores scores can be regarded as *nearly uniform*.
b) Multivariate t distribution with 1 degree of freedom and covariance structure, Σ, as before. In this case the leverage scores are *moderately nonuniform*.
c) Multivariate t distribution with 3 degrees of freedom and and covariance structure, Σ, as before. Here, the leverage scores are *very nonuniform*.

One can see that HD disperses the data points in a structured manner. The leverage/structural information is essentially "shared" by groups of data points. This pattern becomes stronger as the leverage scores become more nonuniform. The second column, on the other hand, plots the original leverage scores on the x-axis and the leverage scores of HDX on the y-axis. The leverage scores of HDX are distributed more uniformly on the y-axis. This uniformization property of HD becomes even more apparent when inspecting the kernel density of both of the matrices in column 3 of figure 1. The density mass is spread across the (0,1)-support of the leverage scores. The transformation has the strongest impact in the case where the original leverage scores are very nonuniform. Throughout the following analysis we will see that this represents a case in which the algorithmic leveraging procedure is especially effective. If r is large enough, this smoothing in combination with the sampling matrix P allows us to construct an FJLT with high probability. Furthermore, computing the r vectors of the FJLT transformed matrix can be done in $O(ndlog(r))$, as desired. Hence, $\Pi_{FJLT} = PHD$ can be viewed as a subsampled randomized Hadamard transform (Drineas et al., 2012, p. 3449) or as a FJLT with high probability.

The notion of Π_{FJLT} allows us to construct two different fast LS methodologies (Mahoney, 2016, p. 71):

(i) Random Projection Approach: Uniformize leverage scores and sample uniformly/use sparse projection matrix.

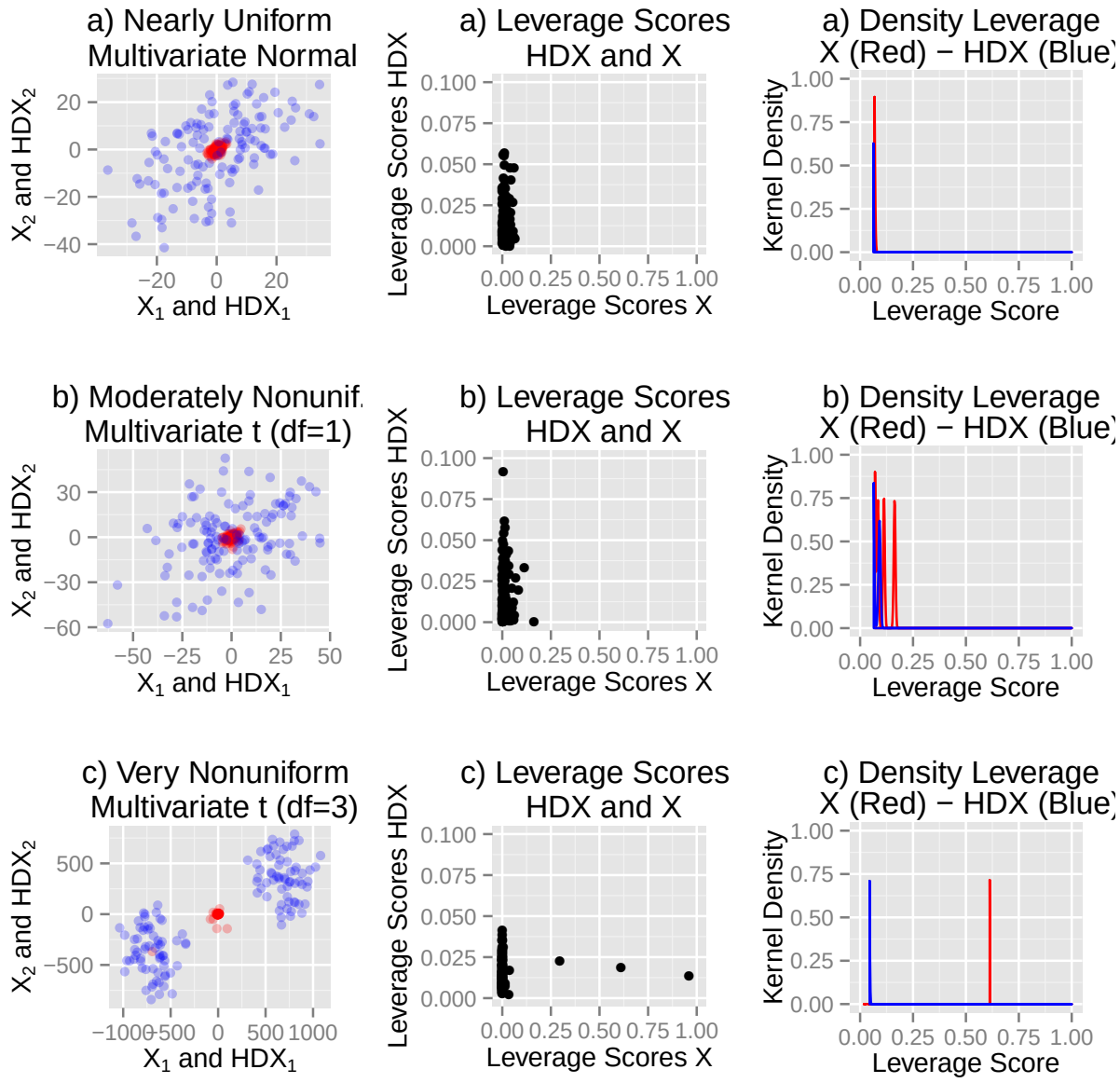

Figure 1: Randomized Hadamard Transform and Leverage Score Uniformization

First, Π can be viewed as a random projection matrix. In this case Π_{FJLT} is directly multiplied with X and y. HD uniformizes the leverage scores of X and P either uniformly samples from the original rows of HDX or randomly (sparse) projects the d points in $\mathbb{R}^n$ to $\mathbb{R}^r$. Since we preprocess the data directly, we are going to loose the interpretability of the rows as being observations in both cases. Again, this might not matter from a theoretical computer science perspective, but it might be crucial from a statistics point of view. On the other hand, it is data-agnostic in the sense, that we do not have to precompute the leverage scores. After we obtain $\tilde{X} = \Pi X$ and $\tilde{y} = \Pi y$, we simply solve the reduced normal equations problem.

(ii) Random Sampling Approach: Use FJLT to compute approximate leverage scores and sample accordingly.

Second, instead of sketching the whole matrix X we can sketch its leverage scores l_i. Recall that the bottleneck of the slow Algorithm 1 was the exact computation of the leverage scores. As we will see in the next section, we can also use Π_{FJLT} in combination with a regular JLT to construct a fast sketch of the leverage scores. These approximate leverage scores are afterwards used to construct an sampling distribution which in turn is used to sample rows from the original X and y, yielding $\tilde{X}, \tilde{y}$. Afterwards, again we solve the reduced normal equation problem.
The main difference between the two approaches is that the random projection approach can be regarded as a data-agnostic approach. One does not have to precompute any leverage scores. Furthermore, one looses the interpretability of the rows. By sampling/sparse random projecting from HDX, we obtain linear combinations of the original rows. Especially in a regression context, this may be regarded as a downside. From theoretical computer science point of view each row of the normal equations is interpreted as a simple linear constraint to the optimization problem. A statistician, on the other hand, interprets a row as an observation. Taking linear combinations of many interpretable features such as factors might potentially disturb any causal interpretability of the inferred estimators. Hence, in the effort of unifying both perspectives I will focus on the second approach in the following sections.

3.2 Fast Random Sampling Algorithm for LS

Again, fast versions of the random sampling estimator rely on the property that in high dimensional spaces a lot of flexibility can be induced by allowing for a small amount of slack.[3] We are going to deviate from the exact computation of the leverage scores and instead first compute an approximation to the leverage scores, $\tilde{l}_i$. Afterwards, we use these approximate scores to obtain an approximation to the LS objective and the solution vector. By optimally choosing the dimensions to which we reduce, we can we obtain an ϵ-approximation with high probability. Drineas et al. (2012, p. 3451) propose the following algorithm to obtain a fast ϵ-approximation to the statistical leverage scores:

[3] E.g. in $\mathbb{R}^d$ it is possible to generate a lot more (if d is large) than d orthogonal vectors when allowing for a small amount of ϵ-slack.

Algorithm 2 Drineas et al. (2012, p. 3451) - FJLT approximation for leverage scores

Input: $X \in \mathbb{R}^{n \times d}$ with SVD $X = U^{(X)}\Sigma V^T$ (not required as input) and an ϵ-level
Output: Approximate leverage scores, $\tilde{l}_i$, $i = 1, ..., n$

1: Let $\Pi_1 \in \mathbb{R}^{r_1 \times n}$ be an ϵ-FJLT for $U^{(X)}$ with $r_1 = \Omega\left(\frac{dlog(n)}{\epsilon^2} log\left(\frac{dlog(n)}{\epsilon^2}\right)\right)$ - precomputed (no additional complexity).
2: Compute $\Pi_1 X$ and its SVD/QR where $R^{(\Pi_1 X)} = \Sigma^{(\Pi_1 X)} V^{(\Pi_1 X)T}$.
3: Let $\Pi_2 \in \mathbb{R}^{d \times r_2}$ be an ϵ-JLT for n^2 vectors, with $r_2 = O(\frac{log(n)}{\epsilon^2})$ - precomputed (no additional complexity).
4: View the rows of $XR^{-1|(\Pi_1 X)} \in \mathbb{R}^{n \times d}$ as n vectors in $\mathbb{R}^d$.
5: **return** $\tilde{l}_i = ||(XR^{-1|(\Pi_1 X)}\Pi_2)_i||_2^2$, an ϵ-approximation of l_i.

First the algorithm applies a FJLT to X. Afterwards, one computes the SVD/QR of the transformed matrix $\Pi_1 X$. Multiplying X by the inverse of $R^{(\Pi_1 X)}$ one obtains a rough sketch of $U^{(X)}$. By applying another regular JLT and taking the euclidean row norm, one obtains the ϵ-approximation to the exact leverage scores.
The overall complexity of the algorithm is $O(ndlog(r_1) + r_1 d^2 + r_2 d^2 + ndr_2)$ and can be decomposed in the following way (Mahoney, 2016, p. 84f.):

- $\Pi_1 X$ takes $O(ndlog(r_1))$ since Π_1 is an ϵ-FJLT
- SVD/QR of $\Pi_1 X \in \mathbb{R}^{r_1 \times d}$ takes $O(r_1 d^2)$
- $R^{-1|(\Pi_1 X)}\Pi_2$ takes $O(r_2 d^2)$ since Π_2 is an ϵ-JLT
- Premultiplying by X takes $O(ndr_2)$ (plain matrix-matrix multiplication)

Theorem 4 shows that by choosing the parameters appropriately,[4] the algorithm has complexity $O(ndlog(d/\epsilon) + nd\epsilon^{-2}log(n) + d^3\epsilon^{-2}log(n)log(d\epsilon^{-1}))$. This implies that even choosing a reasonably small $\epsilon \approx 0.01$ will not result in complexity that is greater than $O(ndlog(r))$. Hence, the fast approximation can be incorporated in a fast version of Algorithm 1. Furthermore, one can obtain a concentration result for the leverage score approximation:

Theorem 4 (Approximation of the Leverage Scores (Drineas et al., p. 3443f., 2012))**.** *Algorithm 2 returns an approximation of the leverage scores, $\tilde{l}_i$ such that with probability at least 0.8*

$$|l_i - \tilde{l}_i| \leq \epsilon l_i \; \forall \; i = 1, ..., n$$

Furthermore, the complexity of the algorithm is

$$O(ndlog(\frac{d}{\epsilon}) + nd\epsilon^{-2}log(n) + d^3\epsilon^{-2}log(n)log(d\epsilon^{-1}))$$

A proof of this theorem can be found in the appendix. Note that the same concentration inequality result (even tighter) can be obtained without multiplying by the second

[4] $r_1 = O(\epsilon^{-2}dlog(n)(log(\epsilon^{-2}dlog(n))))$, $r_2 = O(\epsilon^{-2}log(n))$, ϵ constant, $\delta = 0.1$, $d \leq n \leq e^d$

JLT, Π_2 (Drineas et al., 2012, 3450). By multiplying X by $R^{-1|(\Pi_1 X)} \in \mathbb{R}^{d \times r_1}$ we are constructing a randomized sketch of $U^{(X)}$. Preprocessing by $\Pi_2 \in \mathbb{R}^{d \times r_2}$ where $r_2 < d$ only allows us to improve the running time. Since we are only interested in the euclidean row norm, we can reduce the column dimensionality by another JLT (Mahoney, 2016, p. 81). This way we do not have the full $O(ndr_1)$ running time from multiplying X by $R^{-1|(\Pi_1 X)} \in \mathbb{R}^{d \times r_1}$ but can instead do the $O(ndr_2)$ matrix-matrix multiplication of X times $R^{-1|(\Pi_1 X)}\Pi_2 \in \mathbb{R}^{d \times r_2}$. Furthermore, a simple example of this approximation for different data-generating processes can be found in figure 2. Instead of only constructing the leverage scores, I am interested in the approximation of the resulting sampling distribution. Therefore, I first compute an approximation to the leverage scores and afterwards construct the distribution by normalizing each score by the sum of all scores. For $\epsilon = 0.2$ and $X \in \mathbb{R}^{4098 \times 2}$ the results can be found in figure 2. The first row of plots displays the quality of approximation of the sampling probabilities if we only apply the ϵ-FJLT, Π_1. Hence, $\hat{l}_i = ||(XR^{-1|(\Pi_1 X)})_i||_2^2$ and $\hat{p}_i = \hat{l}_i / \sum_{i=1}^n \hat{l}_i$. The second row, on the other hand, plots the "faster" version which applies both Π_1 and and the ϵ-JLT, Π_2: $\tilde{l}_i = ||(XR^{-1|(\Pi_1 X)}\Pi_2)_i||_2^2$ and $\tilde{p}_i = \tilde{l}_i / \sum_{i=1}^n \tilde{l}_i$.
Comparing both procedures, we can observe that the approximation of the sampling distribution works especially well for the case, where we only apply the first FJLT, Π_1 to X. The distribution of the statistic is more positively skewed. Hence, again we find the same trade-off as before: computational speed versus quality of approximation. We might be able to improve the computational complexity of the algorithm at the cost of worsening the approximation of the leverage scores. The only exception can be found in the case where the data-generating process induces highly nonuniform leverage scores. Using this approximate sampling distribution from Algorithm 2 and one can obtain the following fast algorithmic leveraging algorithm, first introduced by Drineas et al. (2011):

Algorithm 3 Mahoney (2016, p. 85) - "Fast" Random Sampling Algorithm for LS

Input: LS problem with $X \in \mathbb{R}^{n \times d}$, $y \in \mathbb{R}^n$,

1: $r_1 = \Omega\left(\frac{dlog(n)}{\epsilon^2} log\left(\frac{dlog(n)}{\epsilon^2}\right)\right)$, $r_2 = O(\frac{log(n)}{\epsilon^2})$, and an ϵ-level

Output: Approximate LS solution, $\tilde{\beta}$

2: Let $\{\tilde{l}_i\}_{i=1}^n$ be an $1 \pm \epsilon$ approximation to the leverage score computed using Algorithm 2 with r_1 and r_2 as inputs.
3: Randomly sample $r = O(\frac{dlog(d)}{\epsilon})$ rows of X and y with probability depending on $\tilde{l}_i$, rescale them by $\frac{1}{\sqrt{rp_i}}$ and form $\tilde{X} \in R^{r \times d}, \tilde{y} \in R^r$.
4: Solve $(\tilde{X}'\tilde{X})\tilde{\beta} = \tilde{X}'\tilde{y}$ by SVD/QR/Cholesky.
5: **return** $\tilde{\beta}$, an ϵ-approximation of β_{LS} in $O(ndlog(r))$ time.

After constructing the sampling distribution, we use it to sample r rows randomly from it and construct the low-dimensional sketches of X and y. In the end, we simply solve this reduced system and obtain an ϵ-approximation to the optimal LS solution β_{LS}.
The quality of approximation can be summarized in the following theorem:

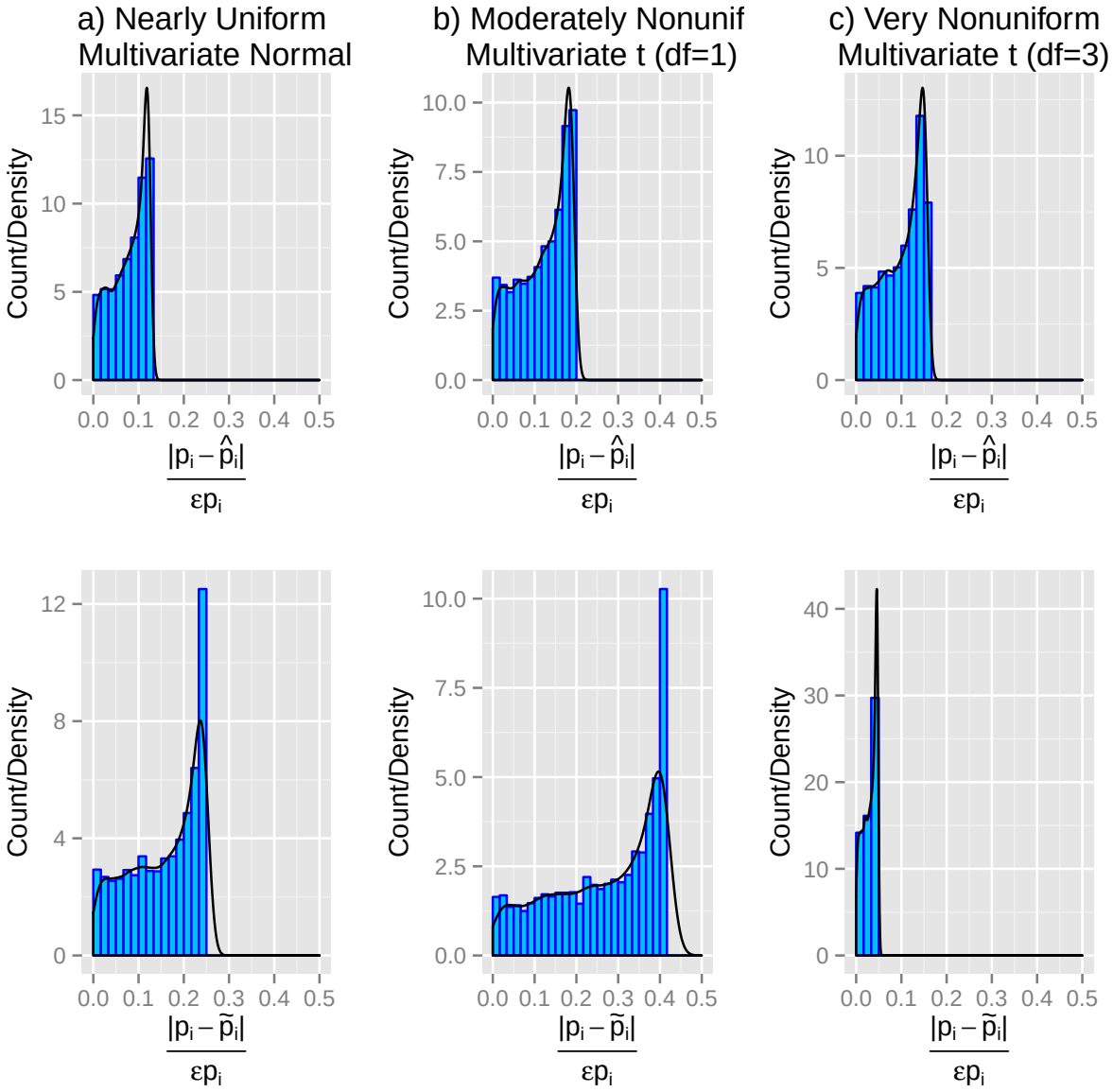

Figure 2: Leverage Score Sampling Distribution Approximation

Theorem 5 (Least Squares Quality of Approximation (Mahoney, 2016, p. 85))**.** *The returned vector $\tilde{\beta}$ of Algorithm 3 is such that with probability at least 0.8*

$$\textit{Solution Certificate: } ||\tilde{\beta} - \beta_{LS}||_2 \leq \sqrt{\epsilon}\kappa(X)\sqrt{\gamma^{-2} - 1}||\beta_{LS}||_2 \,, \tag{6}$$

$$\textit{Objective Function: } ||X\tilde{\beta} - y||_2 \leq (1+\epsilon)||X\beta_{LS} - y||_2 \,, \tag{7}$$

where $\kappa(X)$ is the condition number of X and γ denotes the fraction of the norm of y that lies in the column space of X

$$\gamma = \frac{||U^{(X)}U^{(X)T}y||_2}{||y||_2} \,.$$

For a proof see appendix A. We can summarize this result by noting two very important properties: First, the approximation of the solution vector is inversely proportional to the condition number of the design matrix X (ratio of the largest and the smallest eigenvalue of X). Second, γ denotes the portion of the euclidean norm of y that lies outside of the column span of X (Drineas et al., 2011, p. 2). Hence, it measures the in-sample fit of the LS solution. If we were able to find a linear combination of X that fits the target vector y almost perfectly, then $\gamma \lessapprox 1$. In this case the solution vector approximation is close to perfect.

3.3 Simulation Results

In what follows we analyze the statistical properties of the resulting estimators by conducting Monte Carlo experiments. We are not only interested in how the quality of approximation result behaves under different underlying data-generating processes but also in higher moments of the approximation error distribution. The Monte Carlo experiment is structured in the following way:

1. **Data Generation**: We simulate the data according to one of the three different data-generating processes introduced in section 3.1. Hence, we draw $n = 1000$ observations, $x_i \in \mathbb{R}^5, \;\; i = 1, \ldots, n$ according to one of the processes, which differ in terms of non-uniformity of the leverage scores. Afterwards, $X \in \mathbb{R}^{1000\times 5}$ is multiplied by a vector $\beta_0 \in \mathbb{R}^5$ and Gaussian noise is added in order to form $y \in \mathbb{R}^n$.
2. **Estimator Computation**: Afterwards, I first compute the leverage scores approximation according to Algorithm 2 with $\epsilon = 0.2$ (using only the first FJLT) and afterwards the random sampling estimator, $\tilde{\beta}_{RS} \in \mathbb{R}^5$, for $\epsilon = 0.01$ according to Algorithm 3 as well as the LS solution vector using the Cholesky decomposition, β_{LS}^{CHOL}.
3. **Statistic Computation**: Since we are interested in the approximation error distribution implied by equation 6 we compute the following two standardized

statistics:

$$\frac{||\tilde{\beta} - \beta||_2}{\sqrt{\epsilon}\kappa(X)\sqrt{\gamma^{-2} - 1}||\beta||_2}.$$

We are interested in both the approximation of the true underlying parameter as well as the estimator approximation, β denotes in one case the true data-generating parameter $\beta_0 \in \mathbb{R}^5$ and in the other case the LS solution vector obtained from the Cholesky decomposition solution, $\beta_{LS} \in \mathbb{R}^5$.

4. **Iterate**: We repeat steps 1 to 3 1000 times.

The plots in figure 3 display the empirical kernel/histogram of the statistic we simulate as described above. Since we divided the left-hand side of equation 6 by the right-hand side we draw a vertical line at 1. If 0.8 of the lies below 1, we found evidence that the quality of approximation result holds.
The columns of the figures differ in the uniformity of the leverage scores, I impose in the data generating process of step 1. The first column generates multivariate Gaussian samples, while columns 2 and 3 impose more nonuniform leverage scores by sampling from the multivariate t distribution with different degrees of freedom (1 and 3).
The first row of figure 3 depicts how well the random sampling estimator approximates the true data-generating β_0. The second row on the other hand, depicts the quality of approximation of the LS solution obtained by Cholesky decomposition. For both of the rows the sampling distribution is computed by normalizing the leverage scores. The third row, on the other hand, computes the sampling distribution from the influence scores as described in the next section.
For all three situations, we can directly see that the above theorem holds. More than 0.8 of the probability mass lies below 1. Hence, in at least 80 percent of the cases equation 6 holds. Furthermore, when comparing row 1 and 2, it becomes apparent that the random sampling estimator does a better job at approximating the estimator instead of the true data-generating vector β_0. Intuitively, this makes a lot of sense: One of LS well-known weaknesses is its sensitivity to outliers. Outliers often times have large leverage scores which translate into a higher weight in the sampling procedure, which underlies the randomized estimator. Hence, the weakness of LS is going to be forwarded to the randomized estimator. But this also means that in the case of strong nonuniform leverage scores (column c)), this weakness can be exploited to obtain a more accurate approximation to the LS solution vector. We can conclude, that the random sampling estimator does a very good job at what it is intended to do: Provide an approximation to the least-squares estimator.

3.4 Notes on other influence measures

Our general aim is to reduce a system of linear equations down to a minimum, while still maintaining an ϵ-approximation of the original β_{LS} solution. As argued before, it only seems natural to "value" data points according to their influence on the desired fit. Statistical leverage scores are only one option for doing so. But there are many others (see Chatterjee and Hadi (1986) for an overview). Examples include the following:

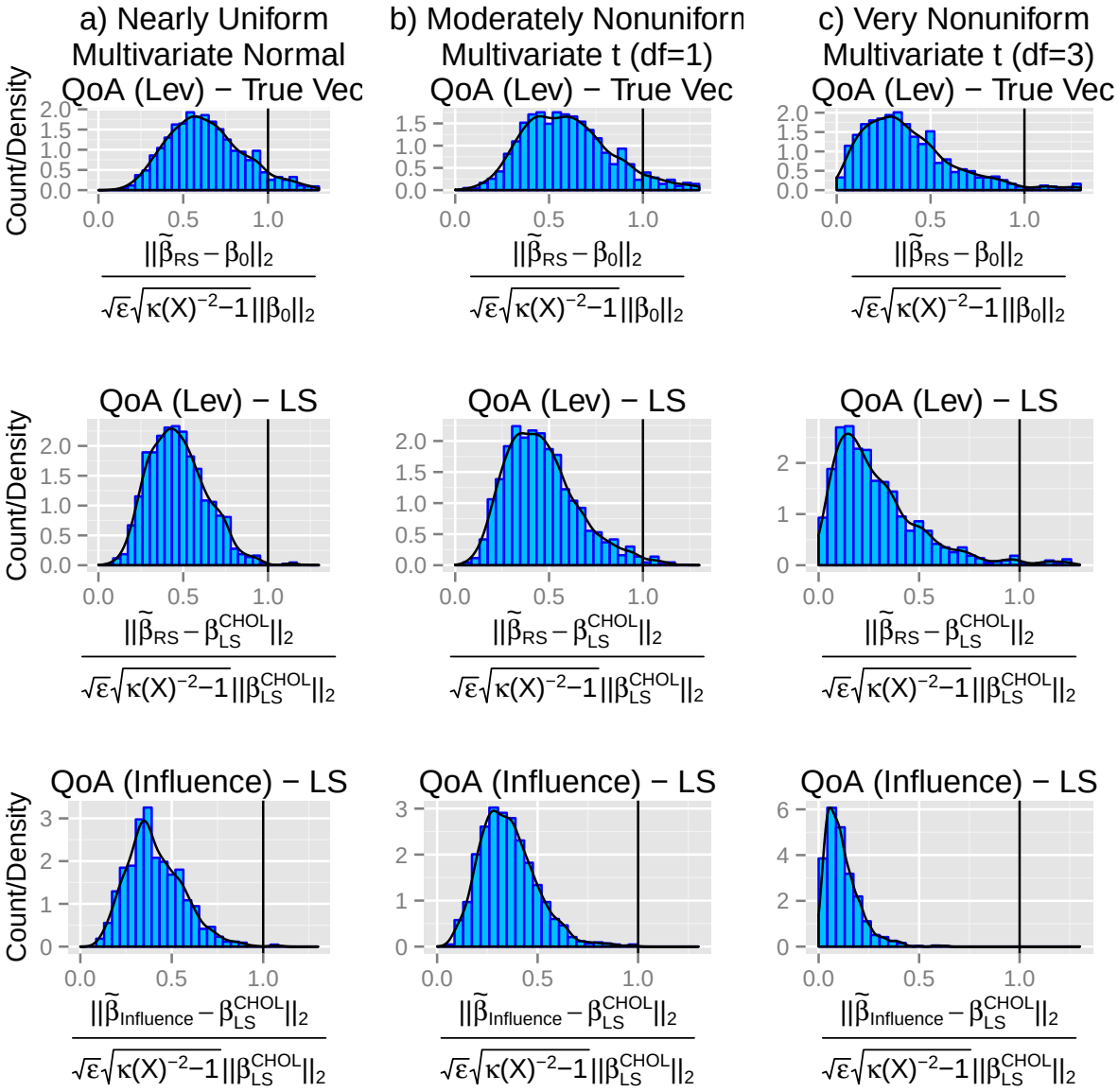

Figure 3: Quality of Approximation - Random Sampling LS Estimator

- Influence scores: $I_i = ||U_i^{(X,Y)}||_2^2 = L_i + \frac{\hat{e}_i}{\sum_{i=1}^n \hat{e}_i}$
- Weighted leverage scores (GLM): $wl_i = ||w_i U_i||_2^2$
- Leverage scores relative to best rank k approximation (Mahoney, 2016, p. 86):

$$l_i^{(k)} = ||U_i^{(X-k)}||_2^2 \text{ where } X = U_k \Sigma_k V_k^T + U_k^{\perp} \Sigma_k^{\perp} V_k^{T,\perp}$$

- Convex combination of uniform probability and leverage score probability (Ma et al. (2015) - leverage score regularization)

In the end for each of these options we form an importance sampling distribution from the scores S_i, $i = 1, ..., n$ by normalizing in the following way:

$$p_i = \frac{S_i}{\sum_{i=1}^n S_i}$$

It might be possible to obtain robustness improvements both in terms of sensitivity to outliers and in terms of decreased sampling variance by using the influence score instead. Furthermore, there has been some research done (Ma et al., 2015) on two particular forms of leverage score "regularization". Ma et al. (2015) show that one can obtain more efficient estimators (decreased variance) at the cost of a small bias in the approximation. This is done by either taking a convex combination between a uniform sampling and the leverage score importance sampling distribution or by not standardizing/reweighting the leverage scores before forming the sampling distribution.
Simulation results displayed in the final row of figure 3 show the performance of a random sampling estimator based on approximate influence scores. We can see that the quality of approximation of β_{LS}^{CHOL} improves for all three different data-generating processes. Computing the influence scores comes at a minimal computational cost. We have to compute the leverage scores for the concatenated $(X, Y) \in \mathbb{R}^{n \times d+1}$ matrix. Hence, the theoretical complexity increases to

$$O\left(n(d+1)log((d+1)/\epsilon) + n(d+1)\epsilon^{-2}log(n) + (d+1)^3\epsilon^{-2}log(n)log((d+1)\epsilon^{-1})\right).$$

We can conclude that both from a statistical as well as a computational point of view it seems preferable to use a more robust measure of influence on the LS fit, which come at small computational costs. In the following analysis we will see how this significant statistical improvement can be used to construct randomized IWLS algorithms.

4 Problem Formulation: GLM

Generalized linear models generalize the Gaussian linear regression model to cases where the target variable Y follows a general distribution from the exponential family in standard form. In this section I first introduce basic notation and the standard maximum likelihood estimation via IWLS. Afterwards, I show the simple connection between the LS problem introduced in the previous section and derive an analogous and more general framework for the GLM problem. The notation and GLM outline is for simplicity and completeness taken from Dobson and Barnett (2008, p. 51f., p. 64f.).

4.1 Generalized Linear Models and Iterative Weighted Least Squares

The general setup is the following: Consider a random variable Y that follows a distribution from the exponential family and let y denote a particular realization. Its density can be written as (Dobson and Barnett, 2008, p. 51):

$$f(y;\theta) = exp\{a(y)b(\theta) + c(\theta) + d(y)\} \, .$$

If $a(y) = y$ the distribution is said to be in canonical/standard form. Furthermore, in this scenario $b(\theta)$ is said to be the natural parameter of the distribution. The log-likelihood for exponential family distributions is the following:

$$l(\theta;y) = a(y)b(\theta) + c(\theta) + d(y) \, .$$

Its derivative with respect to θ is called the score statistic and takes the form

$$S(\theta;y) = \frac{dl(\theta;y)}{d\theta} = a(y)b'(\theta) + c'(\theta) \, .$$

The expectation and variance (also called information) of $S(\theta;y)$ are (Dobson and Barnett, 2008, p. 62f.)

$$\mathbb{E}(S) = b'(\theta)\mathbb{E}(a(Y)) + c'(\theta) = b'(\theta) - \frac{c'(\theta)}{b'(\theta)} + c'(\theta) = 0 \, ,$$

$$I = Var(S) = (b'(\theta)^2)Var(a(Y)) = \frac{b''(\theta)c'(\theta)}{b'(\theta)} - c''(\theta) \, .$$

GLMs assume that the independent random variables $Y_1, ..., Y_n$ follow a distribution of the exponential family which has the canonical form, which depends on the single parameter $\theta = (\theta_1, \ldots, \theta_n)$. Hence, the log-likelihood can be written as (Dobson and Barnett, 2008, 64)

$$l(y_1, \ldots, y_n; \theta_1, \ldots, \theta_n) = \sum_{i=1}^{n} l_i = \sum y_i b(\theta_i) + \sum c(\theta_i) + \sum d(y_i) \, .$$

In general, we are not interested in the parameter vector $\theta \in \mathbb{R}^n$ but in a lower dimension vector $\beta \in \mathbb{R}^d$ where $d << n$. GLMs assume that $\mathbb{E}(Y_i) = \mu_i$, where μ_i is some potentially complex function of θ_i. Furthermore, they introduce a monotone differentiable function, the link function, which maps μ_i to the predictor (Dobson and Barnett, 2008, p. 52):

$$g(\mu_i) = x_i^T \beta = \eta_i \, .$$

GLMs can be conveniently estimated by maximum likelihood estimation (MLE). Taking the gradient of the log-likelihood with respect to β and applying the chain rule, we obtain a system of likelihood equations. This system can be iteratively estimated by making use of scoring methods. Let k denote the k-th iteration of the updating procedure.

Scoring methods update estimates iteratively, using the following the formula (Dobson and Barnett, 2008, p. 65)

$$\beta_{(k+1)} = \beta_{(k)} + I_{(k)}^{-1} S_{(k)} \, . \tag{8}$$

The benefit of such a scheme is that it does not only make use of the first-order moment $S_{(k)}$ but also of the second-order moment $I_{(k)}$. The complete vectorized updating procedure can then be written as

$$I_{(k)}\beta_{(k+1)} = I_{(k)}\beta_{(k)} + S_{(k)} = X^T W_{(k)} z_{(k)} \, ,$$

where $W_{(k)}$ is a diagonal matrix with elements $w_{i(k)} = Var(\mu_i)^{-1} \left(\frac{\partial \mu_i}{\partial \eta_i}\right)^2$ and $z \in \mathbb{R}^n$ is the so-called vector of working variates with $z_{(k)} = X\beta_{(k)} + (y - \mu) diag\left(\frac{\partial \eta_i}{\partial \mu_i}\right)$, where both μ_i and $\frac{\partial \eta_i}{\partial \mu_i}$ are evaluated at $\beta_{(k)}$ (Dobson and Barnett, 2008, p. 65f.). It follows that

$$(X^T W_{(k)} X)\beta_{(k+1)} = X^T W_{(k)} z_{(k)} \, . \tag{9}$$

This can be viewed as a "weighted" version of the normal equations in (2). In general, this expression has to be solved iteratively, since z and W depend on $\beta_{(k)}$. A possible stopping criterion is $||\beta_{(k+1)} - \beta_{(k)}||_2^2 < \delta$. These results can be summarized in the standard IWLS algorithm. This algorithm solves multiple weighted least squares (WLS) problems until the solution β_{GLM} has been found. Each iteration of the algorithm has $O(nd^2)$ computational complexity, since the inversion can again be circumvented by using the Cholesky or QR decomposition for $W^{1/2}X$. The convergence and overall running time of the algorithm depends like all first-/second-order optimization methods on the behavior of the gradient.

Algorithm 4 Iterative Weighted Least Squares

Input: GLM problem with $X \in \mathbb{R}^{n \times d}$, $y \in \mathbb{R}^n$, initialization $\beta_{(0)}$, δ
Output: GLM solution, β_{GLM}

1: **while** $||\beta_{(k+1)} - \beta_{(k)}||_2^2 > \delta$ **do**
2: Compute $z_{(k)} = X\beta_{(k)} + (y - \mu) diag\left(\frac{\partial \eta_i}{\partial \mu_i}\right)$
3: Compute $W_{(k)} = diag\left(Var(\mu_i)^{-1} \left(\frac{\partial \mu_i}{\partial \eta_i}\right)^2\right)$ with $\left(\frac{\partial \mu_i}{\partial \eta_i}\right)^2$ evaluated at $\beta_{(k)}$
4: Solve $\beta_{(k+1)} = (X^T W_{(k)} X)^{-1} X^T W_{(k)} z_{(k)}$.
5: **end while**
6: **return** $\beta_{(k+1)}$, the final estimator β_{GLM}.

IWLS essentially solves multiple WLS problems. After each iteration the vector of working variates and the weighting matrix are updated and a new WLS problem is formulated. This procedure continues until a stationary point in the solver is reached.

Randomized algorithms might be able to speed up each single WLS problem at the cost of introducing an approximation error due to randomization in each iteration. This randomization comes from the fact that the subspace embedding is only exact up to an ϵ-degree (and that only with high probability). If we could find an algorithm that improves the theoretical computational complexity of each individual iteration, while still converging in the same amount of iterations, then this might lead to significant improvements in running times. Hence, we have three requirements for a potential randomized algorithm for solving the ML problem of GLMs:

1. The randomized algorithm should result in an ϵ-approximation of β_{GLM}, $\tilde{\beta}_{RandGLM}$.
2. The randomized algorithm should solve any individual iteration of the IWLS algorithm in at most $o(nd^2)$.
3. In order to obtain not only theoretical complexity but also running time improvements the algorithm should converge in a reasonable amount of iterations relative to the original IWLS algorithm.

4.2 Transformed Problem and Approximation Error Propagation

In general, weighted least squares problems can be written as

$$\min_{\beta \in \mathbb{R}^d} (y - X\beta)^T M (y - X\beta) \ , \tag{10}$$

where $M \in \mathbb{R}^{n \times n}$ is some form of weighting matrix. In the LS case introduced in section 2.2 M took the role of the identity matrix. The randomized algorithm coupled M with the randomized version of the identity $\Pi^T \mathbf{1}_n \Pi = \Pi^T \Pi$. Treating Π as a random sampling matrix, which selects rows according to their leverage scores, we saw that it is possible to obtain an ϵ-approximation to the solution vector in $O(ndlog(r))$. The solution boiled down to solving the transformed normal equations:

$$(X^T \Pi^T \Pi X)\beta = X^T \Pi^T \Pi y \ .$$

Since $\mathbb{E}(\Pi^T \Pi) = I_n$, this estimator is unbiased. In the generalized scenario we are interested in constructing an approximation of $\beta_{(k)}$ at every single iteration, $\tilde{\beta}_{(k)}$. Therefore, we have to derive sketches for X, $z_{(k)}$ and the weighting matrix $W_{(k)}$. Let $\tilde{W}_{(k)} \in \mathbb{R}^{r \times r}$ be the diagonal matrix of weights, which correspond to the r sampled observations by Π. E.g., if we sample rows 1, 3, 5 and 9, then $\tilde{W}_{(k)} = diag(w_{1(k)}, w_{3(k)}, w_{5(k)}, w_{9(k)})$. Hence, we are sketching all ingredients of equation 9, in order to reduce the dimensionality of the original problem.
A first formulation for the randomized IWLS problem might look as follows: At each iteration of the algorithm we want to solve the *weighted and randomized* system of equations:

$$\tilde{X}^T \tilde{W}_{(k)} \tilde{X} \tilde{\beta}_{(k+1)} = \tilde{X}^T \tilde{W}_{(k)} \tilde{z}_{(k)} \ , \tag{11}$$

where $\tilde{X} = \Pi X \in \mathbb{R}^{r \times d}$, $\tilde{W}_{(k)} \in \mathbb{R}^{r \times r}$ is as described above and $\tilde{z}_{(k)} = \Pi z_{(k)} \in \mathbb{R}^r$. This formulation makes use of a weighted randomized identity matrix, $\Pi^T \tilde{W}_{(k)} \Pi$. After solving equation 11 for $\tilde{\beta}_{(k+1)}$, the vector of working variates and the weighting matrix are updated. Let ϵ_1 denote the approximation error for the first period. For the transition from the first to the second iteration this works out as follows:

$$\hat{z}_{(2)} = X\tilde{\beta}_{(1)} + (y - \tilde{\mu}) diag\left(\frac{\partial \tilde{\eta}_i}{\partial \tilde{\mu}_i}\right) = (1 \pm \epsilon_1) z_{(2)}$$

$$\hat{W}_{(2)} = diag\left(Var(\tilde{\mu}_i)^{-1} \left(\frac{\partial \tilde{\mu}_i}{\partial \tilde{\eta}_i}\right)^2\right) = (1 \pm \epsilon_1) W_{(2)}$$

where

$$\tilde{\eta} = X\tilde{\beta}_{(1)} = (1 \pm \epsilon_1) X\beta_{(1)} \text{ and } \tilde{\mu} = \mathbb{E}(X\tilde{\beta}_{(1)}) = \mathbb{E}(X\beta_{(1)})$$

The quality-of-approximation result of theorem 6 applies directly to $\tilde{\beta}_{(1)}$ and the IWLS solution vector of the first iteration $\beta_{(1)}$. Due to the fact that $\tilde{\beta}_{(1)}$ is unbiased estimator of $\beta_{(1)}$ it follows that $\tilde{\mu} = \mu$. Hence, also $\tilde{\eta}$ and therefore $\hat{z}_{(2)}$ and $\hat{W}_{(2)}$ are going to be ϵ-approximations of their exact counterparts. Let ϵ_k denote the relative approximation error conducted in iteration k. In the following iterations these approximations are going to propagate:

$$\hat{z}_{(k+1)} = X\tilde{\beta}_{(k)} + (y - \tilde{\mu}) diag\left(\frac{\partial \tilde{\eta}_i}{\partial \tilde{\mu}_i}\right) = \prod_{j=1}^{k} (1 \pm \epsilon_j) z_{(k+1)}$$

$$\hat{W}_{(k+1)} = diag\left(Var(\tilde{\mu}_i)^{-1} \left(\frac{\partial \tilde{\mu}_i}{\partial \tilde{\eta}_i}\right)^2\right) = \prod_{j=1}^{k} (1 \pm \epsilon_j) W_{(k+1)}$$

where

$$\tilde{\eta} = X\tilde{\beta}_{(k)} = \prod_{j=1}^{k} (1 \pm \epsilon_j) X\beta_{(k)}$$

$$\tilde{\mu} = \mathbb{E}(X\tilde{\beta}_{(k)}) = \mathbb{E}(X \prod_{j=1}^{k} (1 \pm \epsilon_j) \beta_{(k)}) = \mathbb{E}(X\beta_{(k)})$$

But due to the fact that the construction of the random sampling estimator is unbiased and given the independence of the approximation errors as $k \to \infty$, $\prod_{k=1}^{K} (1 \pm \epsilon_k) = (1 \pm \epsilon)^k \to 1$. Hence, we can conclude that the randomized estimator is going to be asymptotically (in terms of the number of iterations) consistent. The following chapter therefore analyzes the performance of three different random sampling estimators for the IWLS algorithm that converges in a finite amount of iterations.

5 Algorithmic Leveraging for GLM

Based on the previous insights, this chapter introduces three different versions of an algorithm for a fast randomized computation of the IWLS estimator. All of them construct fast sketches of the input matrices similar to the LS case discussed in Chapter 3. They differ on how they utilize the information in the weighting matrix, $W \in \mathbb{R}^{n \times n}$ and the vector of working variates, $z \in \mathbb{R}^n$. Again, we analyze the empirical performance of all three randomized estimators for three different data-generating processes. In order to better understand the behavior of the approximation error, we discuss how the estimator evolves during the complete IWLS procedure. A possible problem to the procedure is the variance introduced by randomization in the sampling step. If the estimator varies too much from one iteration to the next, one might not be able to achieve convergence in a reasonable amount of time. Hence, we also discuss the convergence behavior of both algorithms and show the evolution of the estimator evolves over the course of the iterative procedure.

5.1 Random Sampling Algorithms for GLMs

The algorithm we present assumes Π to be a random sampling matrix, constructed from 3 different measures of influence: static leverage scores of X, dynamically weighted leverage scores of $W_{(k)}$ and X and so-called working variate influence scores of $(X, z_{(k)}) \in \mathbb{R}^{n \times (d+1)}$. The first naive randomized version of the IWLS algorithm computes the leverage scores of X, $\tilde{l}_i$ according to algorithm 2. Afterwards we sample r rows from $X \in \mathbb{R}^{n \times d}$, $z_{(k)} \in \mathbb{R}^n$ and diagonal elements from $W_k \in \mathbb{R}^{n \times n}$ according to the corresponding normalized discrete probability distribution. These sketches $\tilde{X} \in \mathbb{R}^{r \times d}$, $\tilde{W}_{(k)} \in \mathbb{R}^{r \times r}$ and $\tilde{z}_{(k)} \in \mathbb{R}^r$ are then used to compute $\tilde{\beta}_{(k+1)}$ according to equation 11. Finally, this $\tilde{\beta}_{(k+1)}$ is then used to update both the working variates and the weighting matrix, $z_{(k+1)} = X\tilde{\beta}_{(k+1)} \in \mathbb{R}^n$ and $W_{(k+1)} \in \mathbb{R}^{n \times n}$. In the following iteration the procedure is repeated again. Computationally this comes at the advantage of not having to compute a new leverage score distribution at every iteration. One can just compute it at the first iteration, save it and at each following iteration reuse it. This way we are able to obtain running time improvements if $n >> d$. This implementation of the randomized IWLS algorithm can be found in Algorithm 5, where the influence type is "Leverage". Since the bottleneck of one iteration of the IWLS algorithm is the inversion/solving of equation 9 which can be done in $O(nd^2)$, we are able to improve the complexity to $O(ndlog(r))$ by solving equation 11 instead.
As we noted in the previous chapter, the approximation of the solution vector in iteration k is directly passed into $z_{(k+1)}$ and $W_{(k+1)}$. Hence, also the influence of each observation on its corresponding working variate changes from iteration to iteration. The first naive leverage score version of the algorithm disregards these dynamic changes. This might lead to an unnecessary increase in the variance of the estimator due to the sampling variance.
One way to account for this is to directly incorporate the weighting done by W (*see*

"Weighted Leverage" version of Algorithm 5). Jia (2014) first proposed the notion of weighted leverage scores in this context:

Definition 6 (Weighted Leverage Scores (Jia, 2014))**.** *Given an orthonormal basis $U^{(X)}$ for $span(X)$ and a weighting matrix W, the weighted leverage scores of X are defined as*

$$wl_i = ||w_i U_i^{(X)}||_2^2,$$

where w_i is the i-th diagonal element of the weighting matrix W.

At each iteration of the algorithm we compute weighted leverage scores and standardize them to obtain an importance sampling distribution. The rest of the algorithm remains the same. This way we are able to dynamically incorporate the adaptive weighting of each observation during the IWLS procedure. An important difference to the previous leverage score version is that we standardize the weighted leverage scores by $||W_{(k)}\tilde{U}^{(X)}||_F^2$ instead of d. This is due to the fact that in the LS case, $W = I_n$ and $||U^{(X)}||_F^2 = d$. All of this can again be done in a fast way by adapting Algorithm 2. The expression $XR^{-1|(\Pi_1 X)}\Pi_2 = \tilde{U}^{(X)}$ is a fast approximation to the orthonormal basis of $span(X)$. Multiplying $\tilde{U}^{(X)}$ with the matrix W and taking the euclidean row norm of every row does not come at an additional cost.
The final version of the algorithm (*see "Influence" version of algorithm 5*) makes use of adaptive influence scores of X and $z_{(k)}$. They are defined in the following way:

Definition 7 (Working Variate Influence Scores)**.** *At iteration k of the Randomized IWLS scheme the working variate influence scores are defined as the leverage scores of the concatenated matrix $(X, z_{(k)}) \in \mathbb{R}^{n\times(d+1)}$,*

$$wvI_i = ||U_i^{(X,z_{(k)})}||_2^2 \,.$$

where $z_{(k)}$ denotes the working variate at the k-th iteration of the randomized

Again, at every iteration the design matrix is concatenated with the vector of working variates, which yields $(X, z_{(k)}) \in \mathbb{R}^{n\times(d+1)}$. Afterwards, the algorithm computes the leverage scores for this concatenated matrix. The rest of the procedure remains untouched. A benefit of the adaptive influence scores is that the weighting can be regarded as more direct, since the working variate is treated as part of the column span of X.
All three version are summarized in Algorithm 5.

5.2 Simulation Results

Again, we analyze the statistical properties of the resulting estimators by the means of simulation. The Monte Carlo experiments displayed in figure 4 depict how the randomized estimator performs compared to the baseline IWLS solution in a simple logistic regression scenario. The simulation proceeds in the following way:

Algorithm 5 IWLS using Random Sampling based on Influence Measures

Input: GLM problem with $X \in \mathbb{R}^{n \times d}$, $y \in \mathbb{R}^n$, initial $\beta_{(0)} \in \mathbb{R}^d$,

1: $r_1 = \Omega\left(\frac{dlog(n)}{\epsilon^2} log\left(\frac{dlog(n)}{\epsilon^2}\right)\right)$, $r_2 = O(\frac{log(n)}{\epsilon^2})$, $\epsilon \in (0,1)$, δ, one of three Influence Types

Output: Approximate GLM solution, $\tilde{\beta}_{RandGLM}$

2: **while** $||\beta_{(k+1)} - \beta_{(k)}||_2^2 > \delta$ **do**

3: Compute $z_{(k)} = X\beta_{(k)} + (y - \mu)diag\left(\frac{\partial \eta_i}{\partial \mu_i}\right)$ with $\eta = X\beta_{(k)}$ and $\mu = \mathbb{E}(\eta)$

4: Compute $W_{(k)} = diag\left(Var(\mu_i)^{-1}\left(\frac{\partial \mu_i}{\partial \eta_i}\right)^2\right)$ with $\left(\frac{\partial \mu_i}{\partial \eta_i}\right)^2$ evaluated at $\beta_{(k)}$

5: **if** Influence Type = "Leverage" **then**

6: Let $\{\tilde{l}_i\}_{i=1}^n$ be an $1 \pm \epsilon$ approx. to the leverage scores of X - call Algorithm 2.

7: **end if**

8: **if** Influence Type = "Weighted Leverage" **then**

9: Compute $p_{i(k)} = \frac{||w_{i(k)}\tilde{U}_i^{(X)}||_2^2}{||W_{(k)}\tilde{U}^{(X)}||_F^2}$, $\forall i = 1, ..., n$

10: **end if**

11: **if** Influence Type = "Influence" **then**

12: Let $\{\widetilde{wvI}_i\}_{i=1}^n$ be an $1 \pm \epsilon$ approx. to the leverage scores of $(X, z_{(k)})$ - call Algorithm 2.

13: **end if**

14: Randomly sample $r = O(\frac{dlog(d)}{\epsilon})$ rows of X, W and z. Rescale them by $\frac{1}{\sqrt{rp_i}}$, form $\tilde{X} \in R^{r \times d}, \tilde{W}_{(k)} \in \mathbb{R}^{r \times r}, \tilde{z}_{(k)} \in R^r$.

15: Solve $\beta_{(k+1)} = (\tilde{X}^T\tilde{W}_{(k)}\tilde{X})^{-1}\tilde{X}^T\tilde{W}_{(k)}\tilde{z}_{(k)}$.

16: **end while**

17: **return** $\tilde{\beta}_{RandGLM} = \beta_{(k+1)}$, an approximation of β_{GLM}.

1. **Data Generation**: After generating $X \in R^{1000\times 5}$ from one of the three data-generating processes introduced in section 3.1, we multiply it by a vector $\beta_0 \in \mathbb{R}^5$ and apply the inverse link function $g^{-1}(\eta) = \frac{exp(\eta)}{1+exp(\eta)}$. Afterwards, y is generated by comparing $g^{-1}(\eta)$ to a uniform distributed random variable. It is set to 1 if the draw is smaller than μ.
2. **Estimator Computation**: We compute the IWLS solution as well as the randomized IWLS estimator (for one of the chosen influence types) according to Algorithm 5. For this we choose to set ϵ to 0.01 and stop the algorithm after a maximum of 50 iterations.
3. **Statistic Computation**: Afterwards, we compute the following measure of quality of approximation:
$$\frac{||\tilde{\beta}_{RandGLM} - \beta_{IWLS}||_2}{||\beta_{IWLS}||_2}$$
It is a relative measure of how much vector length the approximation deviates from the converged IWLS estimate.
4. **Iterate**: We repeat steps 1 to 3 1000 times.

Row one of the figure shows how the simple leverage score estimator performs compared to the IWLS estimator. Row two on the other hand plots the performance of the random sampling estimator based on the weighted leverage scores. Finally, row three displays the quality of approximation for the influence score based estimator. All three estimator types exhibit a bimodal approximation error statistic distribution for the case of nonuniform leverage scores. This is likely to be due to the fact that the algorithm does not always converges. We can observe that the estimator based on the influence scores performs the best and that the first uniform leverage inducing data-generating process yields the best results for all three estimators. This might be related to the previous convergence observation. If the algorithm does not sample a high leverage observation at one iteration but samples it in the following iteration, this might induce too much variance for the algorithm to converge. Therefore, the following section is going to take an in-depth look at the convergence behavior of the algorithm.

5.3 Analyzing Estimator Trajectories and Convergence Behavior

From the previous section we concluded that the quality of approximation of a randomized sampling algorithm strongly depends on the data-generating process and the type of random sampling estimator, which we use. We saw that the more nonuniform the leverage scores, the worse the resulting approximation of the GLM estimator. Furthermore, the working variate influence score based random sampling estimator yielded the best approximation results in terms of concentration of probability mass. This section analyzes how the dynamics of the estimators behave and how the additional randomization effects the variance of the estimator. Figure 5 plots how one single element of the vector $\beta_{RandGLM|(k)}$ evolves from iteration to iteration, $k = 1, ..., K$. The first row

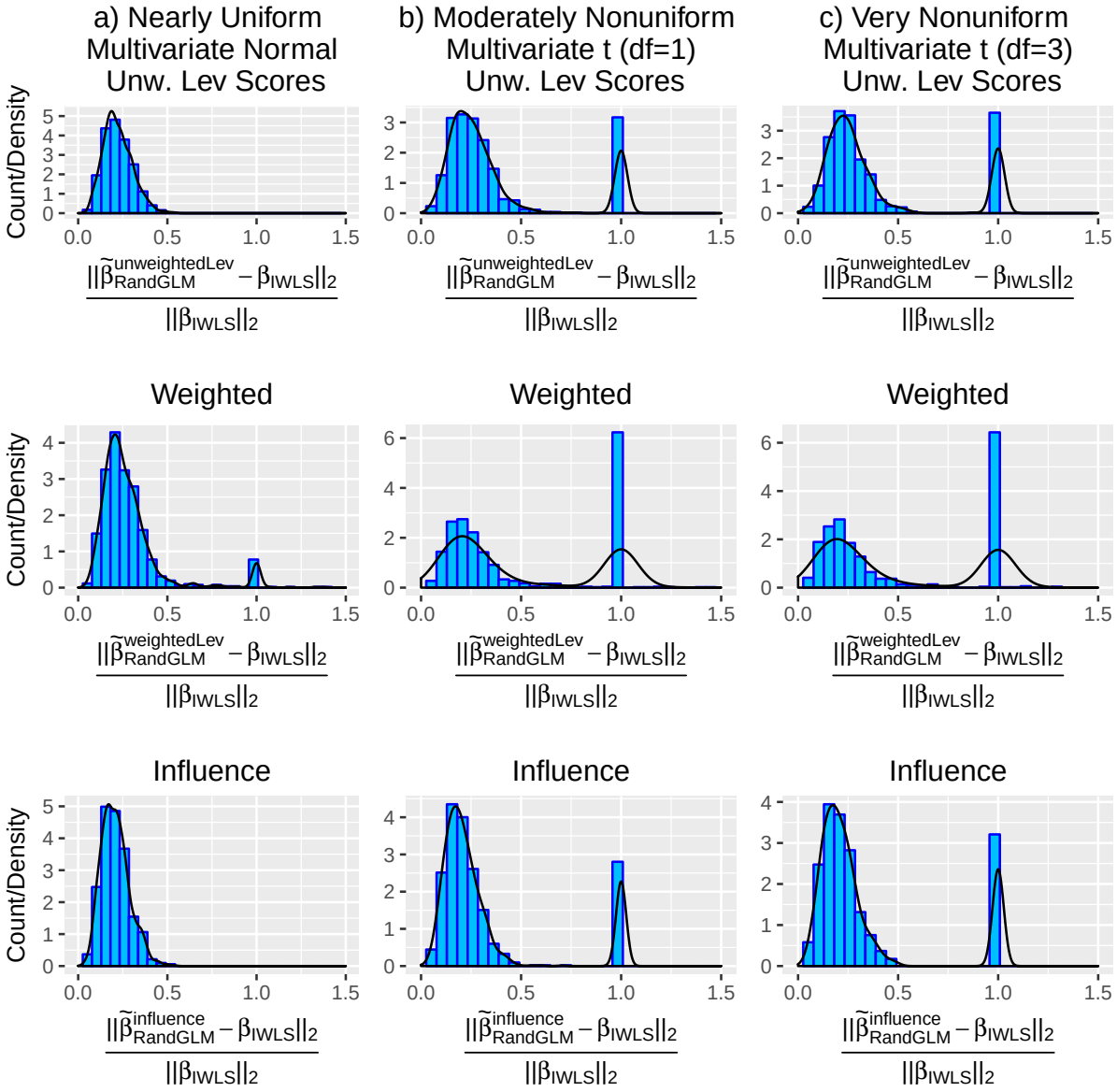

Figure 4: Euclidean Norm Error for the Random Sampling IWLS Estimator

plots the trajectory of the the 4 different estimators for the case where the design matrix is generated from the multivariate normal with mean 0 and $\Sigma_{ij} = 2 \times 0.5^{|i-j|}$. The black vertical line indicates the iteration at which the standard IWLS has converged. Afterwards, the red line which represents the IWLS coefficient is constant. We can see that the variance induced due to the sampling procedure is especially large for the unweighted leverage score random sampling estimator (green line). On the other hand, the random sampling estimator based on influence scores experiences the least variance around the converged IWLS estimator value. Furthermore, one can observe that the variance increases when the leverage scores become more and more nonuniform. The sampling variance is stronger for the case of more nonuniform leverage scores. This strengthens the hypothesis, that the sampling algorithm has to account even stronger for the leverage structure of the data. This way, the sampling variance from iteration to iteration should be decreased. One important research question is whether or not one is able to achieve this by dynamically increasing the amount of rows sampled from iteration to iteration.
Hence, I conclude that the convergence and stability of a random sampling estimator for GLMs depends not only on the nonuniformity of the leverage scores but as well on the mechanism which accounts for the sampling variance.

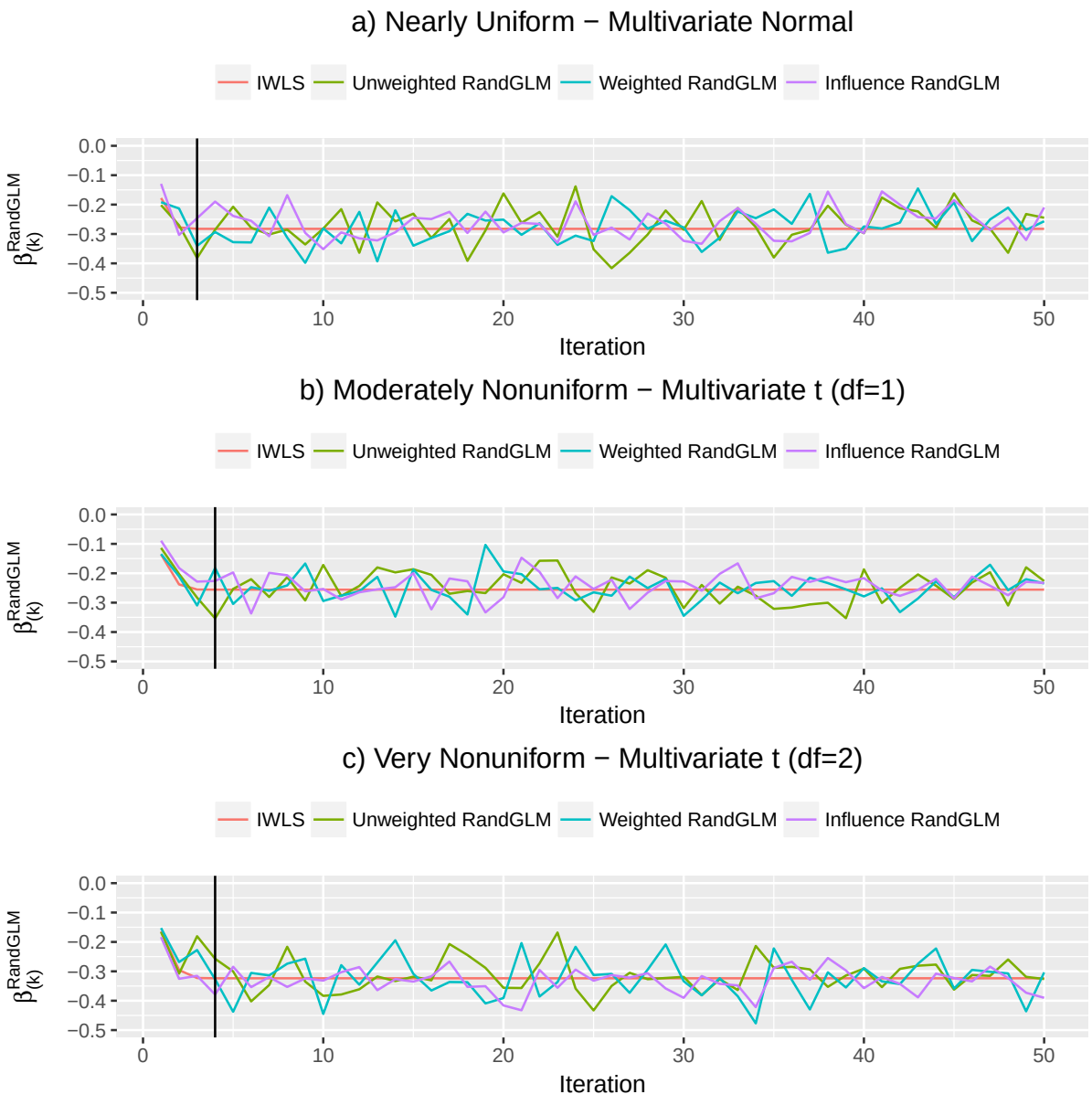

Figure 5: Trace of Estimators based on Random Sampling IWLS Algorithms

6 Conclusion

This thesis has theoretically and empirically analyzed potential extensions of the randomized numerical linear algebra framework to the application of Generalized Linear models. Furthermore, it has attempted to extend and unify this exciting field of computer science with a statistical learning perspective. At the center of this analysis was the fundamental research conducted by Michael Mahoney and Petros Drineas. We introduced the reader to their random sampling or "algorithmic leveraging" scheme for Least Squares. Furthermore, we showed that the essential dimensionality reduction step can be interpreted as a preprocessing of the original normal equation problem by a randomized version of the identity matrix, namely $\Pi^T\Pi$. It turned out that both structural requirements for concentration result could be directly expressed in terms of this randomized "information gateway". We defined and visualized how the Fast Subspace Johnson-Lindenstrauss Transform distorts the initial input space in a structured way. Even for very different data-generating processes the FJLT induces a more uniform leverage score structure. This was achieved by groups of data points socializing or sharing leverage scores. Afterwards, we reviewed the fast random sampling estimator introduced by Drineas et al. (2011). In order to assess the statistical properties of the approximation error, we simulated a statistic obtained from a concentration result. We concluded that the skewness and kurtosis of the distribution of the statistic of interest highly depend on the data-generating process. Furthermore, we showed that it is possible to obtain a better (in terms of probability concentration) random sampling estimator when making use of a sampling distribution based on influence scores. Afterwards, we presented a first formal framework for a random sampling Iterative Weighted Least Squares estimator. Assuming that the random sampling approximation error induced at every iteration is independent across iterations, we were able to show that the random sampling IWLS estimator converges in the amount of iterations to the IWLS solution. Again, we conducted Monte Carlo experiments and simulated random IWLS estimators to examine the statistical properties of the estimators. Finally, we graphically examined how the estimators behaves during the iterations of the algorithm. We concluded that one has to dynamically account for both the leverage score structure as well as the variance induced by the random sampling procedure. Understanding how the convergence behavior might be incorporated into an efficient algorithm design, is crucial to the success of the generalization of a random sampling estimator to any iterative estimation procedure. Furthermore, it might be possible to obtain improvements by adapting the amount of rows sampled at each iteration. This way one might circumvent potential variance increases. Another question worth investigating is whether one is able to obtain faster convergence by using different stopping criterion. This thesis has highlighted the potential of influence measures that dynamically adapt during the iteration of the IWLS procedure. I conclude that the intersection of numerical procedures and statistical inference has never been as important as of now and that more research has to be conducted in this field.

List of Appendices

A Useful Proofs and Derivations

A.1 Difference in Approximation Objectives

Remark 1 (Good approx. of RSS_{min} does not imply good β_{LS} approximation)**.** *Let $y \in \mathbb{R}^n$, $X \in \mathbb{R}^{n \times d}$ and $\beta \in \mathbb{R}^d$. Furthermore, let $\Pi \in \mathbb{R}^{r \times d}$ and $\tilde{\beta} \in \mathbb{R}^d$. Let*

$$RSS_{min} = ||y - X\beta_{LS}||_2^2 \text{ and } \widetilde{RSS}_{min} = ||y - X\tilde{\beta}||_2^2$$

Without further assumptions on the shape of the residual sum of squares function it follows that

$$\tilde{\beta} \approx \beta_{LS} \implies \widetilde{RSS}_{min} \approx RSS_{min} \tag{A.1.1}$$

and

$$\widetilde{RSS}_{min} \approx RSS_{min} \not\implies \tilde{\beta} \approx \beta_{LS} \tag{A.1.2}$$

For the first part let $\tilde{\beta} = (1+\epsilon)\beta_{LS}$. Then

$$\begin{aligned}
\widetilde{RSS}_{min} &= ||y - X(1+\epsilon)\beta_{LS}||_2^2 = ||y - X\beta_{LS} - \epsilon X\beta_{LS}||_2^2 \\
&\leq (||y - X\beta_{LS}||_2 + \epsilon||X\beta_{LS}||_2)^2 \\
&= RSS_{min} + 2\epsilon||y - X\beta_{LS}||_2||X\beta||_2 + \epsilon^2||X\beta_{LS}||_2^2
\end{aligned}$$

Since $\tilde{\beta} \approx \beta_{LS}$ we have that $\epsilon \approx 0$ and therefore it follows that $\widetilde{RSS}_{min} \approx RSS_{min}$.
The second part follows simply by the fact that the RSS is only weakly convex but not not λ-strong convex as in the following definition.

Definition 2 (λ-strong convex functions)**.** *Let f be a function. f is said to be λ-strong convex iff $\forall x, y, \alpha \in (0,1)$, $\exists!\ \lambda > 0$ such that*

$$f(\alpha x + (1-\alpha)y) \leq \alpha f(x) + (1-\alpha)f(y) - \frac{\lambda}{2}(1-\alpha)||x - y||$$

A.2 Johnson-Lindenstrauss Lemma

Lemma 1 (Gaussian-version of Johnson-Lindenstrauss Lemma)**.** *Given d points $\{x_i\}_{i=1}^d$, each of which is in $\mathbb{R}^n$, there exists a linear function $f : \mathbb{R}^n \to \mathbb{R}^r$ such that $\forall i, j = 1, ..., d$ the mapped vectors fulfill*

$$(1-\epsilon)||x_i - x_j||_2^2 \leq ||f(x_i) - f(x_j)||_2^2 \leq (1+\epsilon)||x_i - x_j||_2^2$$

Proof. We want to map the points into $\mathbb{R}^{r<<n}$. While doing so, we would like $f(.)$ to be such that the pointwise distances are preserved.[5] We are going to allow for slack in the pointwise distances and proof the claim by construction. Let $Pi \in \mathbb{R}^{r \times n}$ be such that $\Pi_{ij} = \frac{1}{\sqrt{r}} N(0,1)$. Furthermore, let $f(x) = \Pi x$, and hence $f(x_i) - f(x_j) = \Pi(x_i - x_j)$. For any vector $b \in \mathbb{R}^n$, $b = (b_1, ..., b_n)^T$ and therefore

$$\begin{aligned}
\mathbb{E}(||\Pi b||_2^2) &= \mathbb{E}(\sum_{i=1}^{r}(\sum_{j=1}^{n}\Pi_{ij}b_j)^2) = \sum_{i=1}^{r}\mathbb{E}(\sum_{j=1}^{n}\Pi_{ij}b_j)^2 \\
&= \sum_{i=1}^{r}\sum_{j=1}^{n}\mathbb{E}[(\Pi_{ij}b_j)^2] = \sum_{i=i}^{r}\frac{1}{d}\sum_{j=1}^{n}b_j^2 \\
&= ||b||_2^2
\end{aligned}$$

where the second and third equalities follows from the fact that we are summing over independent and identically distributed random variables. The fourth equality follows from $\mathbb{E}(\Pi_{ij}^2) = \frac{1}{r}$. It follows that for all vector pairs $i, j = 1, ..., d$ it holds that

$$\mathbb{E}(||\Pi(x_i - x_j)||_2^2) = ||x_i - x_j||_2^2$$

We need to proof that for all i, j

$$\begin{aligned}
&\left|\frac{||\Pi(x_i - x_j)||_2^2||}{||x_i - x_j||_2^2} - 1\right| < \epsilon \Leftrightarrow \left|||\Pi\frac{(x_i - x_j)}{||x_i - x_j||_2^2}||_2^2 - 1\right| < \epsilon \\
&\Leftrightarrow \left|||\Pi c_{ij}||_2^2 - 1\right| < \epsilon \text{ where } c_{ij} = \frac{(x_i - x_j)}{||x_i - x_j||_2^2} \text{ and } ||c_{ij}|| = 1
\end{aligned}$$

We can write $||\Pi c_{ij}||_2^2 - 1$ as

$$\sum_{i=1}^{r}(\sum_{j=1}^{n}\Pi_{ij}c_{ij})^2 - 1 = \sum_{i=1}^{r}(N_i^2 - \mathbb{E}(N_i^2))$$

where N_i is a Gaussian and hence $\sum_{i=1}^{r} N_i^2 \sim \chi_r^2$. Using Chernoff bounding methods, we can therefore obtain the following concentration bound

$$\mathbb{P}[|||\Pi c||_2^2 - 1| > \epsilon] \leq e^{-\epsilon^2\frac{r}{4}}$$

Let δ denote the failure probability. Then by the union bound

$$\mathbb{P}(\max_{i,j=1,...,d}|||\Pi c_{ij}||_2^2 - 1| > \epsilon) \leq \binom{d}{2}e^{-\epsilon\frac{r}{4}} < \delta$$

Noticing that $\binom{d}{2} \sim \frac{d(d-1}{2} \approx \frac{d^2}{2}$, the result follows for $\delta = 0.5$ from $r \geq \frac{4}{\epsilon^2}log\frac{d^2}{2\delta}$. □

[5] If $n = d$ and $x_1, ..., x_d$ were the standard basis vectors (all pointwise distances are equal), then we can only map to $\mathbb{R}^{n-1}$ and preserve distances but not less. Hence, we need a highly unbalanced setting in terms of dimensionality.

A.3 Leverage Score and Least Squares Approximation

Theorem 1 (Hadamard Preprocessing - Flattening (Mahoney, 2016, p. 69))**.** *Fix a set X of n vectors in $\mathbb{R}^d$ and let $HD \in \mathbb{R}^{n\times n}$ be a randomized Hadamard Transform. Then, w.p. at least $1 - \frac{1}{20}$, we have that*

$$\max_{x\in X} ||HDx||_\infty = O\left(\sqrt{\frac{log(n)}{d}}\right)$$

Proof. The proof is taken for completeness from Mahoney (2016, p. 69f.) Assume w.l.o.g. unit length of a fixed vector $x \in X$ ($||x||_2 = 1$). Afterwards the define the following random variable

$$u = HDx = (u_1, \dots, u_d)^T$$

where u_i is of the form $\sum_{i=1}^d a_i x_i$, where each $a_i = \frac{\pm 1}{\sqrt{d}}$ is chosen uniformly and independently. Apply Chernoff bound:

$$\begin{aligned}
\mathbb{E}[e^{\lambda d u_i}] &= \prod_i \mathbb{E}[e^{\lambda d a_i x_i}] \\
&= \prod_i \mathbb{E}[cosh(t\sqrt{d_i}x_i] && \text{(by cosinus hyperbolicus)} \\
&\leq exp(\lambda^2 d||x||_2^2/2) && \text{(by Chernoff bound)}
\end{aligned}$$

Apply Markov's Inequality $\forall s > 0$ and plug in $\lambda = sd$ (from Chernoff bounding method - minimize bound wrt. λ) into previously obtained bound

$$\begin{aligned}
\mathbb{P}(|u_1|) &= 2\mathbb{P}(e^{2du_1} \geq e^{s^2 d}) \\
&\leq 2\frac{\mathbb{E}[e^{sdu_1}]}{e^{s^2 d}} && \text{(by Chernoff Argument)} \\
&\leq 2e^{s^2 d||x||_2^2/2 - s^2 d} && \text{(by Markov's Inequality)} \\
&= 2e^{-s^2 d/2} && \text{(by } ||x||_2 = 1\text{)} \\
&\leq \frac{1}{20nd} && \text{(for } s = \Theta(\sqrt{\tfrac{log(n)}{d}})\text{)}
\end{aligned}$$

Perform union bound over all nd coordinates of the vectors $\{HDx : x \in X\}$ gives

$$\max_{x\in X} ||HDx||_\infty = O\left(\sqrt{\frac{log(n)}{d}}\right)$$

□

Lemma 2 (Structural Lemma FJLT (Mahoney, 2016, p. 78f.))**.** *Let $X \in \mathbb{R}^{n\times d}$ with $n >> d$ and $rank(X) = d$. Furthermore, let its SVD be $X = U^{(X)}\Sigma V^T$. Also, let $\Pi \in \mathbb{R}^{r\times n}$ be an FJLT for $U^{(X)}$ and $\Pi U^{(X)} = U^{(\Pi U)}\Sigma^{(\Pi U)}V^{T|(\Pi U)}$. Then the four following results hold:*

1. $rank(\Pi X) = rank(\Pi U^{(X)}) = rank(U^{(X)}) = rank(X) = d$
2. $||I_d - \Sigma^{(\Pi U)^{-2}}||_2 \leq \frac{\epsilon}{1-\epsilon}$
3. $(\Pi X)^\dagger = V\Sigma^{-1}(\Pi U^{(X)})^\dagger$
4. $||(\Pi U^{(X)})^\dagger - (\Pi U^{(X)})^T||_2 = ||\Sigma^{(\Pi U)} - \Sigma^{-1|(\Pi U)}||_2$

Proof. The proof is taken for completeness from Mahoney (2016, p. 78f.) Claim 1: Note that $\forall i \in \{1, ..., d\}$:

$$\begin{aligned} |1 - \sigma_i^2(\Pi U^{(X)})| &= |\sigma_i(U^{(X)T}U^{(X)}) - \sigma_i(U^{(X)T}\Pi^T\Pi U^{(X)})| \\ &\leq ||U^{(X)T}U^{(X)} - U^{(X)T}\Pi^T\Pi U^{(X)}||_2 \\ &\leq \epsilon \qquad \text{(By } \Pi \text{ being an FJLT)} \end{aligned}$$

where the first inequality follows from the fact that the spectral norm of a matrix is equal to the largest singular value of the matrix. Hence, the singular values won't be distorted too much. Since the number of strictly positive singular values equals the number of strictly positive eigenvalues, which in turn equals the rank of X, the rank of X is approximately preserved (given $r \geq \frac{dlog(d)}{\epsilon}$).

For all other claims: See Mahoney (2016, p. 78f.) □

Theorem 3 (Approximation of the Leverage Scores (Drineas et al., 2012, p. 3443f.))**.** *Algorithm 2 returns an approximation of the leverage scores, $\tilde{l}_i$ such that with probability at least 0.8*

$$|l_i - \tilde{l}_i| \leq \epsilon l_i \; \forall \; i = 1, ..., n$$

Proof. The proof is replicated for completeness from Mahoney (2016, p. 81f.) and Drineas et al. (2012, p. 3454f.). The setup is the same as in the algorithm.

Let $\Pi_1 \in \mathbb{R}^{r_1 \times n}$ be an ϵ-FJLT for $U^{(X)}$ and let $\Pi_2 \in \mathbb{R}^{r_1 \times r_2}$ be an ϵ-JLT for n^2 points in $\mathbb{R}^{r_1}$. Both "events" hold with constant probability. Let $X^\dagger = V\Sigma^{-1}U^{(X)T}$ denote the generalized Moore-Penrose inverse. Define the following quantities:

$$\hat{U}_i^{(X)} = e_i X(\Pi_1 X)^\dagger \text{ and } \hat{l}_i = ||\hat{U}_i^{(X)}||_2^2$$

$$\tilde{U}_i^{(X)} = e_i X(\Pi_1 X)^\dagger \Pi_2 \text{ and } \tilde{l}_i = ||\tilde{U}_i^{(X)}||_2^2$$

The aim of the proof is to show that choosing the parameters optimally one can obtain with high probability the following

$$U_i^{(X)T}U_j^{(X)} \approx \hat{U}_i^{(X)T}\hat{U}_j^{(X)} \text{ and } \hat{U}_i^{(X)T}\hat{U}_j^{(X)} \approx \tilde{U}_i^{(X)T}\tilde{U}_j^{(X)}$$

In order to establish the theorem, we need to make use of the following two results which are proven below:

1) For all $i, j \in \{1, ..., n\}$ it holds that $|U_i^{(X)T} U_j^{(X)} - \hat{U}_i^{(X)T} \hat{U}_j^{(X)}| \leq \frac{\epsilon}{1-\epsilon} ||U_i^{(X)}||_2 ||U_j^{(X)}||_2$. Establishes that the FJLT does not distort the leverage scores ($i = j$) by too much. It implies that

$$|l_i - \hat{l}_i| \leq \frac{\epsilon}{1-\epsilon} l_i$$

2) For all $i, j \in \{1, ..., n\}$ it holds that $|\hat{U}_i^{(X)T} \hat{U}_j^{(X)} - \tilde{U}_i^{(X)T} \tilde{U}_j^{(X)}| \leq 2\epsilon ||\hat{U}_i^{(X)}||_2 ||\hat{U}_j^{(X)}||_2$ Establishes that the JLT does not distort the leverage scores ($i = j$) by too much. It implies that

$$|\hat{l}_i - \tilde{l}_i| \leq 2\epsilon \hat{l}_i$$

Combining the two results, making use of the triangle inequality and the fact that $\epsilon \leq \frac{1}{2}$ gives the result

$$\begin{aligned} |l_i - \tilde{l}_i| &= |l_i + \hat{l}_i - \hat{l}_i - \tilde{l}_i| \\ &\leq |l_i - \hat{l}_i| + |\hat{l}_i - \tilde{l}_i| \\ &\leq \left(\frac{\epsilon}{1-\epsilon} + 2\epsilon \right) l_i \\ &\leq 4\epsilon l_i \end{aligned}$$

□

Lemma 4 (Drineas et al. (2012, p. 3454f.))**.** *For all $i, j \in \{1, ..., n\}$ it holds that*

$$|U_i^{(X)T} U_j^{(X)} - \hat{U}_i^{(X)T} \hat{U}_j^{(X)}| \leq \frac{\epsilon}{1-\epsilon} ||U_i^{(X)}||_2 ||U_j^{(X)}||_2$$

Proof. The proof is replicated for completeness from Drineas et al. (2012, p. 3454f.). Let $X = U^{(X)} \Sigma V^T$ and use that $(\Pi_1 X)^\dagger = V \Sigma^{-1} (\Pi_1 U^{(X)})^\dagger$, if Π_1 preserves the rank (full rotation). Furthermore, let e_i^T denote a standard basis vector, which acts as a row indicator. It follows that

$$\begin{aligned} \hat{U}_i^{(X)T} \hat{U}_j^{(X)} &= e_i X (\Pi X)^\dagger [(\Pi X)^\dagger]^T X^T e_j \\ &= e_i U^{(X)} \Sigma V^T V \Sigma^{-1} (\Pi U^{(X)})^\dagger [(\Pi U^{(X)})^\dagger]^T \Sigma^{-1} V^T V \Sigma U^{(X)T} e_j \\ &= e_i U^{(X)} (\Pi U^{(X)})^\dagger [(\Pi U^{(X)})^\dagger]^T U^{(X)T} e_j \end{aligned}$$

Using this, we obtain the following inequality

$$\begin{aligned} |U_i^{(X)T} U_j^{(X)} - \hat{U}_i^{(X)T} \hat{U}_j^{(X)}| &= |e_i U^{(X)} U^{(X)T} e_j - e_i U^{(X)} (\Pi U^{(X)})^\dagger [(\Pi U^{(X)})^\dagger]^T U^{(X)T} e_j| \\ &= |e_i U^{(X)} \left(I - (\Pi U^{(X)})^\dagger [(\Pi U^{(X)})^\dagger]^T \right) U^{(X)T} e_j| \\ &\leq ||I_d - (\Pi_1 U^{(X)})^\dagger ((\Pi_1 U^{(X)})^\dagger)^T||_2 ||U_i^{(X)}||_2 ||U_j^{(X)}||_2 \end{aligned}$$

Note the decomposition $\Pi_1 U^{(X)} = U^{(\Pi_1 U)} \Sigma^{(\Pi_1 U)} V^{(\Pi_1 U)T}$ and hence

$$(\Pi_1 U^{(X)})^{\dagger}((\Pi_1 U^{(X)})^{\dagger})^T = V^{(\Pi_1 U)} \Sigma^{-2|(\Pi_1 U)} V^{T|(\Pi_1 U)}$$

We can therefore obtain the proof by the following line of thought:

$$\begin{aligned}
|U_i^{(X)T} U_j^{(X)} - \hat{U}_i^{(X)T} \hat{U}_j^{(X)}| &\leq ||I_d - V^{(\Pi_1 U)} \Sigma^{-2|(\Pi_1 U)} V^{T|(\Pi_1 U)}||_2 ||U_i^{(X)}||_2 ||U_j^{(X)}||_2 \\
&= ||I_d - \Sigma^{-2|(\Pi_1 U)}||_2 ||U_i^{(X)}||_2 ||U_j^{(X)}||_2 \\
&= \frac{\epsilon}{1-\epsilon} ||U_i^{(X)}||_2 ||U_j^{(X)}||_2
\end{aligned}$$

□

Lemma 5 (Drineas et al. (2012, p. 3454f.))**.** *For all $i, j \in \{1, ..., n\}$ it holds that*

$$|\hat{U}_i^{(X)T} \hat{U}_j^{(X)} - \tilde{U}_i^{(X)T} \tilde{U}_j^{(X)}| \leq 2\epsilon ||\hat{U}_i^{(X)}||_2 ||\hat{U}_j^{(X)}||_2$$

Proof. See Drineas et al. (2012, p. 3454f.) □

Theorem 6 (Quality of Approximation Result for LS I (Drineas et al., 2011, p. 7f.))**.** *Consider the LS problem of equation (1). Let $U^{(X)} \in \mathbb{R}^{n \times d}$ denote a matrix that contains the top d singular vectors of the matrix $X \in \mathbb{R}^{n \times d}$. Let $\sigma_{min}(X)$ denote the smallest singular value of the matrix X. Furthermore, assume that the matrix $\Pi \in \mathbb{R}^{r \times n}$ satisfy the conditions stated in (4) and (5), for some $\epsilon \in (0, 1)$. Then, the solution vector $\tilde{\beta}$ (e.g. of algorithm 2 - but deterministic statement) satisfies*

$$||X\tilde{\beta} - y||_2 \leq (1 + \epsilon)||X\beta_{LS} - y||_2 \tag{A.3.1}$$

$$||\tilde{\beta} - \beta_{LS}||_2 \leq \frac{1}{\sigma_{min}(X)} \sqrt{\epsilon} ||X\beta_{LS} - y||_2 \tag{A.3.2}$$

Proof. The proof is replicated for completeness from Drineas et al. (2011, p. 7f.). The randomized LS problem stated in equation 3 can also be written as

$$\begin{aligned}
\min_{\beta \in \mathbb{R}^d} ||\Pi y - \Pi X \beta||_2^2 &= \min_{c \in \mathbb{R}^d} ||\Pi(X\beta_{LS} + y^{\perp}) - \Pi X(\beta_{LS} + c)||_2^2 \\
&\qquad\qquad (y = X\beta + y^{\perp}) \\
&= \min_{c \in \mathbb{R}^d} ||\Pi y^{\perp} - \Pi X c||_2^2 \\
&= \min_{z \in \mathbb{R}^d} ||\Pi y^{\perp} - \Pi U^{(X)} z||_2^2 \\
&\qquad\qquad (span(X) = span(U^{(X)}))
\end{aligned}$$

Let $z_{opt} \in \mathbb{R}^d$ be such that $U^{(X)} z_{opt} = X(\beta_{LS} - \tilde{\beta})$. Furthermore, z_{opt} minimizes the previously stated problem:

$$
\begin{aligned}
||\Pi y^{\perp} - \Pi X(\beta_{LS} - \tilde{\beta})||_2^2 &= ||\Pi y^{\perp} - \Pi(y - y^{\perp}) + \Pi X\tilde{\beta}||_2^2 \\
&= ||\Pi X\tilde{\beta} - \Pi y||_2^2
\end{aligned}
$$

The "randomized" normal equations can therefore be written as

$$
(\Pi U^{(X)})^T(\Pi U^{(X)})z_{opt} = (\Pi U^{(X)})^T \Pi y^{\perp}
$$

Taking the norm on both sides and noticing that by isometry requirement (condition (4))

$$
\sigma_i\left((\Pi U^{(X)})^T \Pi U^{(X)}\right) = \sigma_i^2(\Pi U^{(X)}) \geq \frac{1}{2}\ \forall i
$$

it follows by the subspace embedding requirement (condition (5)) that

$$
\begin{aligned}
\frac{||z_{opt}||_2^2}{2} &\leq ||(\Pi U^{(X)})^T(\Pi U^{(X)})z_{opt}||_2^2 \\
&= ||(\Pi U^{(X)})^T \Pi y^{\perp}||_2^2 = ||U^{(X)T}\Pi^T \Pi y^{\perp}||_2^2 \\
&\leq \frac{\epsilon}{2} RSS
\end{aligned}
$$

$$
\begin{aligned}
||y - X\tilde{\beta}||_2^2 &= ||y - X\beta_{LS} + X\beta_{LS} - X\tilde{\beta}||_2^2 \\
&= ||y - X\beta_{LS}||_2^2 + ||X\beta_{LS} - X\tilde{\beta}||_2^2 \\
&\qquad\qquad \text{(by Pythagorean - } y^{\perp} \text{ is orthog to } X) \\
&= RSS + ||U^{(X)} z_{opt}||_2^2 \\
&\leq RSS + \epsilon RSS
\end{aligned}
$$

The first claim follows from $\sqrt{1+\epsilon} \leq 1 + \epsilon$. In order to establish the second claim, note $X(\beta_{LS} - \tilde{\beta}) = U^{(X)} z_{opt}$. Taking the norm on both sides will yield

$$
\begin{aligned}
||\beta_{LS} - \tilde{\beta}||_2^2 &\leq \frac{||U^{(X)} z_{opt}||_2^2}{\sigma_{min}^2(X)} && \text{(by } rank(X) = d\text{)} \\
&\leq \frac{\epsilon RSS}{\sigma_{min}^2(X)} && \text{(by orthogonality of } U^{(X)}\text{)}
\end{aligned}
$$

Taking the square root, claim 2 follows.

□

Theorem 7 (Quality of Approximation Result for LS II (Drineas et al., 2011, p. 8))**.** *Using theorem 6 and assuming that* $||U^{(X)}U^{(X)T}y||_2 \leq \gamma||y||_2$ *it follows that*

$$
||\tilde{\beta} - \beta_{LS}||_2 \leq \sqrt{\epsilon}\kappa(X)\sqrt{\gamma^{-2} - 1}||\beta_{LS}||_2
$$

Proof. The proof is replicated for completeness from Drineas et al. (2011, p. 8f.). Since $\gamma||y||_2 \geq ||U^{(X)}U^{(X)T}y||_2$ it follows that

$$\begin{aligned} RSS &= ||y||_2^2 - ||U^{(X)}U^{(X)T}y||_2^2 \\ &\leq (\gamma^{-2} - 1)||U^{(X)}U^{(X)T}y||_2^2 \\ &\leq (\gamma^{-2} - 1)\sigma_{max}^2(X)||\beta_{LS}||_2^2 \end{aligned}$$

The last inequality follows from $U^{(X)}U^{(X)T}y = X\beta_{LS}$ and by the triangle inequality

$$||U^{(X)}U^{(X)T}y||_2 = ||X\beta_{LS}||_2 \leq ||X||_2||\beta_{LS}||_2 = \sigma_{max}^2(X)||\beta_{LS}||_2$$

Combining this with equation A.3.2, we obtain

$$\begin{aligned} ||\tilde{\beta} - \beta_{LS}||_2 &\leq \frac{1}{\sigma_{min}(X)}\sqrt{\epsilon}\sqrt{(\gamma^{-2} - 1)\sigma_{max}^2(X)||\beta_{LS}||_2^2} \\ &= \frac{\sigma_{max}(X)}{\sigma_{min}(X)}\sqrt{\epsilon}\sqrt{(\gamma^{-2} - 1)}||\beta_{LS}||_2 \end{aligned}$$

□

References

ACHLIOPTAS, D. (2003): "Database-friendly random projections: Johnson-Lindenstrauss with binary coins," *Journal of computer and System Sciences*, 66, 671–687.

AILON, N. AND B. CHAZELLE (2006): "Approximate nearest neighbors and the fast Johnson-Lindenstrauss transform," *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*, 557–563.

CHATTERJEE, S. AND A. S. HADI (1986): "Influential observations, high leverage points, and outliers in linear regression," *Statistical Science*, 379–393.

DOBSON, A. J. AND A. BARNETT (2008): *An introduction to generalized linear models*, CRC press.

DRINEAS, P., R. KANNAN, AND M. W. MAHONEY (2006a): "Fast Monte Carlo algorithms for matrices I: Approximating matrix multiplication," *SIAM Journal on Computing*, 36, 132–157.

——— (2006b): "Fast Monte Carlo algorithms for matrices II: Computing a low-rank approximation to a matrix," *SIAM Journal on Computing*, 36, 158–183.

——— (2006c): "Fast Monte Carlo algorithms for matrices III: Computing a compressed approximate matrix decomposition," *SIAM Journal on Computing*, 36, 184–206.

DRINEAS, P., M. MAGDON-ISMAIL, M. W. MAHONEY, AND D. P. WOODRUFF (2012): "Fast approximation of matrix coherence and statistical leverage," *Journal of Machine Learning Research*, 13, 3475–3506.

DRINEAS, P., M. W. MAHONEY, S. MUTHUKRISHNAN, AND T. SARLÓS (2011): "Faster least squares approximation," *Numerische Mathematik*, 117, 219–249.

FRANKL, P. AND H. MAEHARA (1988): "The Johnson-Lindenstrauss lemma and the sphericity of some graphs," *Journal of Combinatorial Theory, Series B*, 44, 355–362.

GOLUB, G. H. AND C. F. VAN LOAN (2012): *Matrix computations*, vol. 3, JHU Press.

HUBER, P. J. AND E. M. RONCHETTI (2009): "Robust Statistics. Hoboken," *NJ: Wiley. doi*, 10, 9780470434697.

JIA, J. (2014): "Influence Sampling for Generalized Linear Models," Workshop Presentation: MMDS.

MA, P., M. W. MAHONEY, AND B. YU (2015): "A statistical perspective on algorithmic leveraging," *Journal of Machine Learning Research*, 16, 861–911.

MAHONEY, M. W. (2016): "Lecture Notes on Randomized Linear Algebra," *arXiv preprint arXiv:1608.04481*.

RASKUTTI, G. AND M. MAHONEY (2014): "A statistical perspective on randomized sketching for ordinary least-squares," *arXiv preprint arXiv:1406.5986*.