A Randomized Algorithm to Reduce the Support of Discrete Measures

Francesco Cosentino
Mathematical Institute
University of Oxford
The Alan Turing Institute
francesco.cosentino@maths.ox.ac.uk

Harald Oberhauser
Mathematical Institute
University of Oxford
oberhauser@maths.ox.ac.uk

Alessandro Abate
Dept. of Computer Science
University of Oxford
The Alan Turing Institute
alessandro.abate@cs.ox.ac.uk

Abstract

Given a discrete probability measure supported on N atoms and a set of n real-valued functions, there exists a probability measure that is supported on a subset of $n+1$ of the original N atoms and has the same mean when integrated against each of the n functions. If $N \gg n$ this results in a huge reduction of complexity. We give a simple geometric characterization of barycenters via negative cones and derive a randomized algorithm that computes this new measure by "greedy geometric sampling". We then study its properties, and benchmark it on synthetic and real-world data to show that it can be very beneficial in the $N \gg n$ regime.

1 Introduction

Discrete probability measures are central to many inference tasks, for example as empirical measures. In the "big data" regime, where the number N of samples is huge, this often requires to construct a reduced summary of the original measure. Often this summary is constructed by sampling n points at random out of the N points, but Tchakaloff's theorem suggests that there is another way.

Theorem 1 (Tchakaloff [1])**.** *Let μ be a discrete probability measure that is supported on N points in a space $\mathscr{X}$. Let $\{f_1,\dots,f_n\}$ be a set of n real-valued functions $f_i : \mathscr{X} \to \mathbb{R}$, $n < N$. There exists a discrete probability measure $\hat{\mu}$ such that $\operatorname{supp}(\hat{\mu}) \subset \operatorname{supp}(\mu)$, $|\operatorname{supp}(\hat{\mu})| \leq n+1$, and*

$$\mathbb{E}_{X\sim\mu}[f_i(X)] = \mathbb{E}_{X\sim\hat{\mu}}[f_i(X)] \text{ for all } i \in \{1,\dots,n\}. \tag{1}$$

We introduce a randomized algorithm that computes $\hat{\mu}$ efficiently in the $n \ll N$ regime.

Previous work. Reducing the support of a (not necessarily discrete) measure subject to matching the mean on a set of functions is a classical problem, which goes back at least to Gauss' famous quadrature formula that matches the mean of monomials up to a given degree when μ is the Lebesgue measure on $\mathbb{R}$. In multi-dimensions this is known as cubature and Tchakaloff [1] showed the existence of a reduced measure for compactly supported, not necessarily discrete, measures, see [2]. When μ is discrete, the problem of computing $\hat{\mu}$ also runs under the name recombination. Algorithms to compute $\hat{\mu}$ for discrete measures μ go back at least to [3] and have been an intensive research topics ever since; we refer to [4] for an overview of the different approaches, and [4, 5, 6] for recent, state of the art algorithms and applications. Using randomness for Tchakaloff's Theorem has been suggested before [7, 8] but the focus there is to show that the barycenter lies in the convex hull when enough points from a continuous measure are sampled so that subsequently any of the algorithms [4, 5, 6] can be applied; in contrast, our randomness stems from the reduction algorithm itself. More generally, the topic of replacing a large data set by a small, carefully weighted subset is a vast field that has attracted many different communities and we mention, pars pro toto, computational geometry [9], coresets in computer science [10], scalable Bayesian statistics [11], clustering and optimisation [12].

Preprint. Under review.

Contribution. The above mentioned algorithms [4, 5, 6] use a divide and conquer approach that splits up points into groups, computes a barycenter for each group, and solves a constrained linear system several times. This leads to a deterministic complexity that is determined by N and n. In contrast, our approach uses the geometry of cones to "greedy" sample for candidates in the support of μ that are atoms for $\hat{\mu}$ and tries to construct the reduced measure in one go. Further, it can be optimized with classical black box reset strategies to reduce the variance of the run time. Our results show that this can be very efficient in the big data regime $N \gg n$ that is common in machine learning applications, such as least least square solvers when the number of samples N is very large. Moreover, our approach is complementary to previous work since it can be combined with it: by limiting the iterations for our randomized algorithm and subsequently running any of the deterministic algorithms above if a solution by "greedy geometric sampling" was not found.

Outline. We introduce the basic ideas in Section 2, where we derive a simple version of the greedy sampling algorithm, and study its theoretical properties. In Section 3 we optimize the algorithm to better use the cone geometry, combine with reset strategies to reduce the running time variance, and use the Woodbury formula to obtain a robustness result. In Section 4 we discuss numerical experiments that study the properties of the algorithms on two problems: (i) reducing the support of empirical measures; and (ii) least square solvers for large samples. In the Appendix we provide detailed proofs and more background on discrete geometry.

2 Negative cones and a naive algorithm

Background. As is well-known, Theorem 1 follows from Caratheodory's convex hull theorem

Theorem 2 (Caratheodory)**.** *Given a set of N points in $\mathbb{R}^n$ and a point x that lies in the convex hull of these N points. Then x is a linear combination of at most $n+1$ points from the N points.*

It is instructive to recall how Theorem 2 implies Theorem 1. Therefore define a $\mathbb{R}^n$-valued random variable $F : \Omega = \mathscr{X} \to \mathbb{R}^n$ as $F(\omega) := (f_1(\omega), \ldots, f_n(\omega))$ and note that Equation (1) is equivalent to

$$\int_\Omega F(\omega)\mu(d\omega) = \int_\Omega F(\omega)\hat{\mu}(d\omega).$$

Since μ has finite support, the left-hand side can be written as a sum $\sum_{\omega\in\mathrm{supp}(\mu)} F(\omega)\mu(\omega)$. This sum gives a point in the convex hull of the set of N (or less) points $\mathbf{x} := \{F(\omega) : \omega \in \mathrm{supp}(\mu)\}$ in $\mathbb{R}^n$. But by Caratheodory's theorem, this point must be a convex combination of a subset $\hat{\mathbf{x}}$ of only $n+1$ (or less) points of $\mathbf{x}$ and Theorem 1 follows. This proof of Theorem 1 is also constructive in the sense that it shows that computing $\hat{\mu}$ reduces to constructing the linear combination guaranteed by Caratheodory's theorem; e.g. by solving N times a constrained linear system, see [3].

Barycenters and cones. Key to Tchakaloff's theorem is to verify if two measures have the same mean. We now give a simple geometric characterization in terms of negative cones, Theorem 3.

Definition 1. *Let $\mathbf{x} \subset \mathbb{R}^n$ be a finite set of points in $\mathbb{R}^n$. We call the set $C(\mathbf{x}) := \{c \in \mathbb{R}^n \,|\, c = \sum_{x\in\mathbf{x}} \lambda_x x$, where $\lambda_x \geq 0\}$ the cone generated by $\mathbf{x}$ and we also refer to $\mathbf{x}$ as its basis. We call the set $C^-(\mathbf{x}) := \{c \,|\, c = \sum_{x\in\mathbf{x}} \lambda_x x$, where $\lambda_x \leq 0\}$ the negative cone generated by $\mathbf{x}$.*

For example, $C(x_1, x_2)$ is the "infinite" triangle created by the half-lines $\overline{0x_1}$, $\overline{0x_2}$ with origin in 0; $C(\{x_1, x_2, x_3\})$, is the infinite pyramid formed by the edges $\overline{0x_1}$, $\overline{0x_2}$, $\overline{0x_3}$, with vertex 0; etc.

Theorem 3. *Let $\mathbf{x} = \{x_1, \ldots, x_{n+1}\}$ be a set of $n+1$ points in $\mathbb{R}^n$ such that $\mathbf{x} \setminus \{x_{n+1}\}$ spans $\mathbb{R}^n$. Let A be the matrix that transforms $\mathbf{x} \setminus \{x_{n+1}\} = \{x_1, \ldots, x_n\}$ to the orthonormal basis $\{e_1, \ldots, e_n\}$ of $\mathbb{R}^n$, i.e. $Ax_i = e_i$. Further, let h_i be the unit vector such that $\langle h_i, x\rangle = 0$ for all $x \in \mathbf{x} \setminus \{x_i, x_{n+1}\}$ and $\langle h_i, x_i\rangle < 0$ and denote with $H_{\mathbf{x}\setminus\{x_{n+1}\}}$ a $n \times n$ matrix that has $h_1, \ldots, h_n$ as row vectors. It holds that*

1. *$C(\mathbf{x} \setminus \{x_{n+1}\}) = \{c | H_{\mathbf{x}\setminus\{x_{n+1}\}} c \leq 0\}$ and $C^-(\mathbf{x} \setminus \{x_{n+1}\}) = \{c | H_{\mathbf{x}\setminus\{x_{n+1}\}} c \geq 0\}$.*
2. *$Ax \geq 0$ if and only if $H_{\mathbf{x}\setminus\{x_{n+1}\}} x \leq 0$. and $Ax \leq 0$ if and only if $H_{\mathbf{x}\setminus\{x_{n+1}\}} x \geq 0$.*
3. *There exists a convex combination of $\mathbf{x}$ with 0 as barycentre, $\sum_{i=1}^{n+1} w_i x_i = 0$ for some $w_i > 0$, and $\sum_{i=1}^{n+1} w_i = 1$ if and only if $x_{n+1} \in C^-(\mathbf{x} \setminus \{x_{n+1}\})$.*

The above result could be formulated without the matrix A, only in terms of $H_{\mathbf{x}\setminus\{x_{n+1}\}}$. However, A is the inverse of the matrix with columns equal to the vectors $\{\mathbf{x}\setminus\{x_{n+1}\}\}$, hence computing A is more efficient than computing $H_{\mathbf{x}\setminus\{x_{n+1}\}}$, since matrix inversion is optimized in standard libraries.

A Naïve Algorithm. Item 3 of Theorem 3 implies a simple randomized algorithm. If $\mathbb{E}_\mu[X] \neq 0$ we can always study the points $\mathbf{x} - \mathbb{E}_\mu[X]$, this is equivalent in the proof of Theorem 3 to work with cones whose vertex is not 0.

Algorithm 1 Basic measure reduction algorithm

procedure REDUCE(A set $\mathbf{x}$ of N points in $\mathbb{R}^n$)
  Choose n points $\mathbf{x}^\star$ from $\mathbf{x}$
  while $C^-(\mathbf{x}^\star) \cap \mathbf{x} = \emptyset$ **do**
    Replace $\mathbf{x}^\star$ with n new random points $\mathbf{x}^\star$ from $\mathbf{x}$
  end while
  $\mathbf{x}^\star \leftarrow \mathbf{x}^\star \cup x^\star$ with an arbitrary $x^\star \in C^-(\mathbf{x}^\star) \cap \mathbf{x}$
  Solve the linear system $\sum_{x\in\mathbf{x}^\star} w_x^\star x = 0$ for $w^\star = (w_x^\star)_{x\in\mathbf{x}^\star}$
  return $(\mathbf{x}^\star, w^\star)$
end procedure

Corollary 1. *Algorithm 1 computes a reduced measure $\hat{\mu}$ as required by Theorem 1 in $\tau \cdot O(n^3 + Nn^2)$ computational steps. Here $\tau = \inf\{i \geq 1 : C^-(\mathbf{X}_i) \cap \mathbf{x} \neq \emptyset\}$, where $\mathbf{X}_1, \mathbf{X}_2, \ldots$ are random sets of n points sampled uniformly at random from $\mathbf{x}$.*

The complexity of Algorithm 1 in a single loop iteration is dominated by (i) computing the matrix A that defines the cones $C^-(\mathbf{x}^\star)$ and $C(\mathbf{x}^\star)$, (ii) checking if there are points inside the cones, (iii) solving a linear system to compute the weights $w_i^\star$. Respectively, the worst case complexities are $O(n^3)$, $O(Nn^2)$ and $O(n^3)$, since to check if there are points inside the cones we have to multiply A and $\mathbf{X}$, where $\mathbf{X}$ is the matrix whose rows are the vector in $\mathbf{x}$.

Proposition 1. *Let $N > n+1$ and μ be a discrete probability with finite support and $f_1, \ldots, f_n$ be as in Theorem 1. Moreover wlog assume $\mathbb{E}_{X\sim\mu}[f_i(X)] = 0$ for $i = 1, \ldots, n$. With $p := \frac{n\cdot n!(N-n)!}{N!}$ it holds that $\mathbb{E}[\tau] \leq \frac{1}{p}$ and $Var(\tau) \leq \frac{1-p}{p^2}$ and, for fixed n, $\lim_{N\to\infty} \mathbb{E}[\tau] = 1$.*

Not surprisingly, the worst case bound for $\mathbb{E}[\tau]$ are not practical, and it is easy to come up with examples where this τ will be very large with high probability, e.g. a cluster of points and one point far apart from this cluster would result in wasteful oversampling of points from the cluster. Such examples, suggest that one should be able to do better by taking the geometry of points into account which we will do in Section 3. However, before that, it is instructive to better understand the properties of Algorithm 1 when applied to empirical measures.

Application to empirical measures. Consider a random probability measure $\mu = \frac{1}{N}\sum_{i=1}^N \delta_{(f_1(X_i),\ldots,f_n(X_i))}$ where the $X_1, X_2, \ldots$ are independent and identically distributed.

Proposition 2. *Let $N > n+1$ and let $f_1, \ldots, f_n$ be n real-valued functions and $X_1, \ldots, X_N$ be N i.i.d. copies of a random variable X. Set $F(X) = (f_1(X), \ldots, f_n(X))$, assume $\mathbb{E}[F(X)] = 0$ and denote*

$$E := \{0 \in \mathrm{Conv}\{F(X_i), i \in \{1, \ldots, N\}\}\}. \tag{2}$$

1. *$\mathbb{E}[\tau|E] \leq \frac{1}{p}$ and $Var(\tau|E) \leq \frac{1-p}{p^2}$, where*

$$p = \max\left\{\frac{n\cdot n!(N-n)!}{N!}, 1 - \mathbb{P}\left(0 \notin \mathrm{Conv}\{F(X_1), \ldots, F(X_{n+1})\}\right)^{N-n}\right\}, \tag{3}$$

2. *If the law of $F(X)$ is invariant under reflection in the origin, then $\mathbb{P}(0 \notin \mathrm{Conv}\{F(X_1), \ldots, F(X_{n+1})\}) = 1 - 2^{-n}$,*

3. *For fixed n, as $N \to \infty$*

$$\mathbb{P}(\text{for } n \text{ uniformly at random chosen points } \mathbf{x}^\star \text{ from } \mathbf{x},\ \exists x \in \mathbf{x}\ s.t.\ x \in C^-(\mathbf{x}^\star)) \to 1,$$

where $\mathbf{x} = \{F(X_1), F(X_2), \ldots, F(X_N)\}$.

Proposition 2 conditions on the event (2) so that the recombination problem is well-posed, but this happens with probability one for large enough N, see Theorem 5 in Appendix and [8]. Not surprisingly, the worst case bounds of Algorithm 1 can be inconvenient, as equation (3) shows. Nevertheless, item 2 of Proposition 2 shows an interesting trade-off in computational complexity, since the total cost

$$\mathbb{E}[\tau]O(n^3+Nn^2) \leq C(n^3+Nn^2)\min\left\{\frac{N!}{n\cdot n!(N-n)!}, \frac{1}{1-(1-2^{-n})^{N-n}}\right\} \tag{4}$$

has a local minimum in N, see Figure 5 in the Appendix. Section 4 shows that this is observed in experiments and this minimum also motivates the divide and conquer strategy we use in the next section.

3 A geometrically greedy Algorithm

Algorithm 1 produces candidates for the reduced measure by random sampling and then accepts or rejects them via the characterization given in Theorem 3. We now optimize the first part of it, namely the selection of candidates, by exploiting better the geometry of cones.

Motivation in two dimensions. Having chosen $\mathbf{x}^\star$ points we know that we have found a solution if $C^-(\mathbf{x})\cap\mathbf{x}\neq\emptyset$. Hence, maximizing the volume of the cone increases the chance that this intersection is not empty. Indeed, when $n=2$ it is easy to show the following result.

Theorem 4. *Let* $\mathbf{x}$ *be a set of* $N\geq 3$ *points in* $\mathbb{R}^2$ *and* $x_1\in\mathbf{x}$. *Define* $\mathbf{x}^\star=(x_1^\star,x_2^\star)$, *where*

$$x_2^\star := \arg\max_{x\in\mathbf{x}\setminus\{x_1^\star\}}\left|\frac{\langle x_1^\star,x\rangle}{\|x_1^\star\|\|x\|}-1\right|. \tag{5}$$

There exists a convex combination $\sum_{x\in\mathbf{x}}w_x x$ *of* $\mathbf{x}$ *that equals* 0 *if and only if* $\mathbf{x}\cap C^-(\mathbf{x}^\star)\neq\emptyset$.

Hence, if we modify Algorithm 1 by selecting in step 4 the new point by maximizing the angle according to (5) then for $n=2$, Theorem 4 guarantees that $n+1=3$ points out of N are found that constitute a reduced measure $\hat{\mu}$ in $\tau\leq 2$ computational steps.

A geometrically greedy Algorithm. For general dimensions n, the intuition remains that we should maximize the likelihood that the negative cone is non-empty by maximizing the volume of the cone considered, see [13, Chapter 8]. Again, we do this by optimizing its angles, see [14]. This motivates the "geometrically greedy" Algorithm 2 that applies for any n. First note that the deletion

Algorithm 2 Optimized measure reduction algorithm

1: **procedure** REDUCE-OPTIMIZED(A set $\mathbf{x}$ of N points in $\mathbb{R}^n$)
2: Choose n points $\mathbf{x}^\star=\{x_1^\star,\ldots,x_n^\star\}$ from $\mathbf{x}$
3: $i\leftarrow 0$
4: **while** $C^-(\mathbf{x}^\star)\cap\mathbf{x}=\emptyset$ **do**
5: $\mathbf{x}\leftarrow\mathbf{x}\setminus\text{interior}\{C(\mathbf{x}^\star)\}$
6: **if** $i=0$ **then**
7: $x_{i+1}^\star\leftarrow\arg\max_{x\in\mathbf{x}\setminus\mathbf{x}^\star}\left|\langle x,\sum_{j=2}^n x_j^\star\rangle-1\right|$
8: **else**
9: $x_{i+1}^\star\leftarrow\arg\max_{x\in\mathbf{x}\setminus\mathbf{x}^\star}\left|\langle x,\sum_{j=1}^i x_j^\star\rangle-1\right|$
10: **end if**
11: $i\leftarrow((i+1) \mod n)$
12: **end while**
13: $\mathbf{x}^\star\leftarrow\mathbf{x}^\star\cup x^\star$ with an arbitrary $x^\star\in C^-(\mathbf{x}^\star)\cap\mathbf{x}$
14: Solve the linear system $\sum_{x\in\mathbf{x}^\star}w_x^\star x=0$ for $w^\star=(w_x^\star)_{x\in\mathbf{x}^\star}$
15: **return** $(\mathbf{x}^\star,w^\star)$
16: **end procedure**

of points in step 5 in Algorithm 2 does not throw away potential solutions: suppose we have deleted a point $\hat{x}$ in the previous step that was part of a solution, i.e. there exists a set of $n+1$ points $\hat{\mathbf{x}}^\star$ in

$\mathbf{x}$ such that $\hat{x} \in \hat{\mathbf{x}}^\star$, and there exist $n+1$ weights $\hat{w}_i$, $i \in [0,\dots,n+1]$, $\hat{w}_i \in [0,1]$, $\sum w_i = 1$ such that $\sum_{i=1}^{n+1} \hat{w}_i \hat{x}_i^* = 0$, $\hat{x}_i^* \in \hat{\mathbf{x}}^\star$. If we indicate with $\mathbf{x}_c$ the n vectors of the basis of the cone of the previous iteration, we know that $\hat{x} \in \text{interior}\{C(\mathbf{x}_c)\}$, which means there exist strictly positive values c_i such that $\hat{x} = \sum_{i=1}^n c_i x_i^c$, $x_i^c \in \mathbf{x}^c$. Therefore,

$$\sum_{i=1}^{n+1} \hat{w}_i \hat{x}_i^* = \hat{w}_{n+1}\hat{x} + \sum_{i=1}^{n} \hat{w}_i \hat{x}_i^* = \hat{w}_1 \sum_{i=1}^{n} c_i x_i^c + \sum_{i=1}^{n} \hat{w}_i \hat{x}_i^*,\ x_i^c \in \mathbf{x}^c \text{ and } \hat{x}_i^* \in \hat{\mathbf{x}}^\star.$$

Given that w_i and c_i are positive $0 \in \text{Conv}\{\hat{\mathbf{x}}^\star \cup \mathbf{x}_c\}$ and we can apply again the Caratheodory's Theorem 2, which implies that the deleted point $\hat{x}$ was not essential. The reason for the if clause in step 6 is simply that the first time the loop is entered we optimize using the randomly selected bases, but in subsequent runs it is intuitive that we should only optimize over the base points that were optimized in previous loops.

Complexity. We now discuss the complexity of Algorithm 2.

Proposition 3. *The complexity of Algorithm 2 to compute a reduced measure $\hat{\mu}$,as in Theorem 1, is*

$$O(n^3 + n^2 N) + (\tau - 1)O(n^2 + nN),$$

here $\tau = \inf\{i \geq 1 : C^-(\mathbf{X}_i) \cap \mathbf{x} \neq \emptyset\}$ where $\mathbf{X}_1, \mathbf{X}_2, \dots$ are obtained as in Algorithm 2.

In contrast, to the complexity of Algorithm 1, Corollary 1, the n^3 term that results from a matrix inversion is no longer proportional to τ, and the random runtime τ only affects the complexity proportional to $n^2 + nN$. For a generalization of Theorem 4 from $n = 2$ to general n, that is a statement of the form "in n dimensions the algorithm terminates after at most $\tau \leq f(n)$", one ultimately runs into the result of a "positive basis" from discrete geometry, see for example [15], that says if $n \geq 3$ it is possible to build positive independent set of vectors of any cardinality. Characterizing the probability of the occurrences of such sets is an ongoing discrete geometry research topic and we have nothing new to contribute to this. However, despite the existence of such "positive independent sets" for $n \geq 3$, the experiments in Section 4 underlines the intuition that in the generic case, maximizing the angles is hugely beneficial also in higher dimension. If a deterministic bound on the runtime is crucial, one can combine the strengths of Algorithm 2 (a good chance of finding $\mathbf{x}^\star$ quickly by repeatedly smart guessing) with the strength of deterministic algorithms such as [4, 5, 6] by running Algorithm 2 for at most k steps and if a solution is not found run a deterministic algorithms. Indeed, our experiments show that this is on average a very good trade-off since most of the probability mass of τ is concentrated at small k.

Robustness. An interesting question is how robust the measure reduction procedure is to the initial points. Therefore assume we know the solution of the recombination problem (RP) for $\mathbf{x} \subset \mathbb{R}^n$, i.e. a subset of $n+1$ points $\hat{\mathbf{x}} = (\hat{x}_1, \hat{x}_2, \dots, \hat{x}_{n+1}) \subset \mathbf{x}$ and a discrete measure $\hat{\mu}$ on $\hat{\mathbf{x}}$ such that $\hat{\mu}(\hat{\mathbf{x}}) = 0$. If a set of points $\mathbf{y}$ is close to $\mathbf{x}$ one would expect that one can use the solution of the RP for $\mathbf{x}$ to solve the RP for $\mathbf{y}$. The theorem below uses the Woodbury matrix identity to make this precise.

Proposition 4. *Assume that $\text{span}(\hat{\mathbf{x}}) = \text{span}(\hat{\mathbf{x}}_{-1}) = \mathbb{R}^n$, where $\hat{\mathbf{x}}_{-i} := \hat{\mathbf{x}} \setminus \hat{x}_i$. Denote with $\mathbf{X}$ a matrix which as has rows the vectors in $\mathbf{x}$. Suppose there exists an invertible matrix R and another matrix E, such that $\mathbf{X} = \mathbf{Y}R + E$. Denote $\gamma_1 := (\hat{\mathbf{X}}_{-1}^\top)^{-1} \hat{X}_1^\top$, where $\hat{\mathbf{x}}$ is a solution to the RP $\mathbf{x}$. Assuming that the inverse matrices exist, $\hat{\mathbf{X}}R + E_{\hat{\mathbf{x}}}$ is a solution to the RP $\mathbf{y}$ if and only if*

$$\gamma_1^\top + E_{\hat{x}_1} R^{-1} A_1^\top \leq \left(\gamma_1^\top + E_{\hat{x}_1} R^{-1} A_1^\top\right) E_{\hat{\mathbf{x}}_{-1}} \left(I + R^{-1} A_1^\top E_{\hat{\mathbf{x}}_{-1}}\right) R^{-1} A_1^T$$

where E_y indicates the part of the matrix E related to the set of vectors $y \subset \mathbf{x}$ and $A_1 = (\hat{\mathbf{X}}_{-1}^\top)^{-1}$.

This is not only of theoretical interest, since the initial choice of a cone basis in Algorithm 2 can be decisive. For example, if we repeatedly solve a "similar" RP, e.g. N points are sampled from the same distribution, then Proposition 4 suggests that after the first RP solution, we should use the points in the new set of points that are closest to the previous solution as initial choice of the cone basis.

Las Vegas resets. The only source of randomness in Algorithm 2 is the choice of $\mathbf{x}^\star$ in step 2. If this happens to be a bad choice, much computational time is wasted, or even worse, the Algorithm might not even terminate. However, like any other random "black box" algorithm one can stop Algorithm 2

if τ becomes large and then restart with a random basis $\mathbf{x}^\star$ sampled independently from the previous one. We call a sequence $\mathscr{S} = (t_1, t_2, \ldots)$ of integers a reset strategy where the ith entry t_i denotes the number of iterations we allow after the ith time the algorithm was restarted, e.g. $\mathscr{S} = (10, 3, 6, \ldots)$ means that if a solution is not found after 10 loops, we stop and restart; then wait for at most 3 loop iterations before restarting; then 6, etc. Surprisingly, the question which strategy $\mathscr{S}$ minimises the expected run time of Algorithm 2 has an explicit answer. A direct consquence of the main result in [16] is that $\mathscr{S}^\star = c \times (1,1,2,1,1,2,4,1,1,2,1,1,2,4,8,1\ldots)$ achieves the best expected running time up to a logarithmic factor, i.e. by following $\mathscr{S}^\star$ the expected running time is $O(\mathbb{E}[\tau^\star]\log\mathbb{E}[\tau^\star])$ where $\tau^\star$ denotes the minimal expected run time under *any* reset strategy. Thus, although in general we do not know the optimal reset strategy $\tau^\star$ (which will be highly dependent on the points $\mathbf{x}$), we can still follow $\mathscr{S}^\star$ which guarantees that Algorithm 2 terminates and that its expected running time is within a logarithmic factor of the best reset strategy. Since Algorithm 2 uses n updates of a cone basis, it is natural to take c proportional to n (in our experiments we fixed throughout $c = 2n$).

Divide and conquer. A strategy used in existing algorithms [4, 5, 6] is for a given $\mu = \sum_{i=1}^N w_i x_i$ to partition the N points into $2(n+1)$ groups $I_1, \ldots, I_{2(n+1)}$ of approximately equal size, compute the barycenter b_i for each group I_i, and then carry out any measure reduction algorithm to reduce the support of $\sum_{i=1}^{2(n+1)} \frac{w_i}{\sum_{j\in I_i} w_j}\delta_{b_i}$ to $n+1$ points. One can then repeat this procedure, see [4, Algorithm 2 on p1306] for details, and since each iteration halves the support, this terminates after $\log(N/n)$ iterations. Hence the total cost is $O(Nn + \log(N/n))C(2(n+1), n+1))$, where $C(2(n+1), n+1)$ denotes the cost to reduce the measure from $2(n+1)$ points to $n+1$ points. For the algorithm in [4] $C(2(n+1), n+1) = O(n^4)$, similarly to [6]; for the algorithm in [5] $C(2(n+1), n+1) = O(n^3)$. Similarly, we can also run our randomized Algorithm 2 on the $2(n+1)$ points. However, the situation is more subtle since the regime where Algorithm 2 excels is the big data regime $N \gg n$, so running the algorithm on smaller groups of points could reduce the effectiveness. Already for simple distributions and Algorithm 1, as in Proposition 2 item 2, we see that for the optimal choice we should divide the points in N_n^* groups with N_n^* denoting the argmin of the complexity in N. This decreases the computational cost to $O(Nn + \log_{N_n^*/n}(N)\bar{C}(N_n^*, n+1))$, where $\bar{C}$ denotes the computational cost of Algorithm 1 to reduce from N_n^* points to $(n+1)$ points. In general, the optimal N_n^* will depend on the distribution of the points, but an informal calculation shows that $N_n^* = 50(n+1)$ achieves the best trade-off; see Appendix 9 for details.

4 Experiments

We give two sets of experiments to demonstrate the properties of Algorithm 1 and Algorithm 2: (i) using synthetic data allows to study the behaviour in various regimes of N and n, (ii) on real-world data, following [6], for fast least square solvers. As baselines we consider two recent algorithms [5] (*det3*) and [4] resp. [6] (*det4*)[1].

Reducing empirical measures. We sampled $N \in \{2n, 20, 30, 50, \ldots, 10^6\}$ points (i) a $n = 15$-dimensional standard normal random variable (*symmetric1*), (ii) a $n = 20$-dimensional standard normal random variable (*symmetric2*), (iii) $n = 20$ dimensional mixture of exponential (*non symmetric*). We then ran Algorithm 1 (*basic*), Algorithm 2 (*optimized*), as well as Algorithm 2 with the optimal Las Vegas reset (*optimized-reset*) and the divide and conquer strategy (*log-opt*); the results are shown in Figure 1. The first row clearly shows that the performance gets best in the big data regime $N \gg n$. The biggest effect is how the angle/volume optimization of Algorithm 2 drastically reduces the number of iterations compared to Algorithm 1, and therefore the running time and the variance. From a theoretical perspective is interesting that the shape predicted in Proposition 2, for symmetric distributions such as the normal distribution (the two columns on the left) also manifests itself for the non-symmetric mixture (the right column); see also Figure 5 in the Appendix.

As expected, the regime when the number of points N is much higher than the number of dimensions n, yields the best performance for the randomized algorithm; moreover, Figure 2 shows that the run time is approximately $O(n)$ (in contrast to the runtime of *det1* and *det2* that is $O(n^4)$ resp. $O(n^3)$).

[1] Although the derivation is different, the resulting algorithms in [4, 6] are essentially identical; we use the implementation of [6] since do the same least mean square experiment as in [6]. All experiments have been run on a MacBook Pro, CPU: i7-7920HQ, RAM: 16 GB, 2133 MHz LPDDR3.

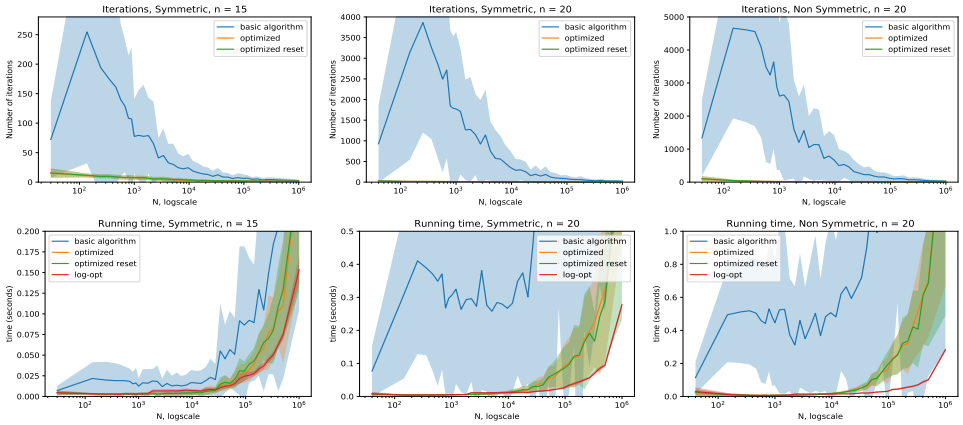

Figure 1: Running time and number of iterations of the randomized algorithms as N varies. The first two columns show the results for *symmetric1* and *symmetric2* , the right column for *non-symmetric*. The shaded area represents the standard deviation (from 70 repetitions of the experiment).

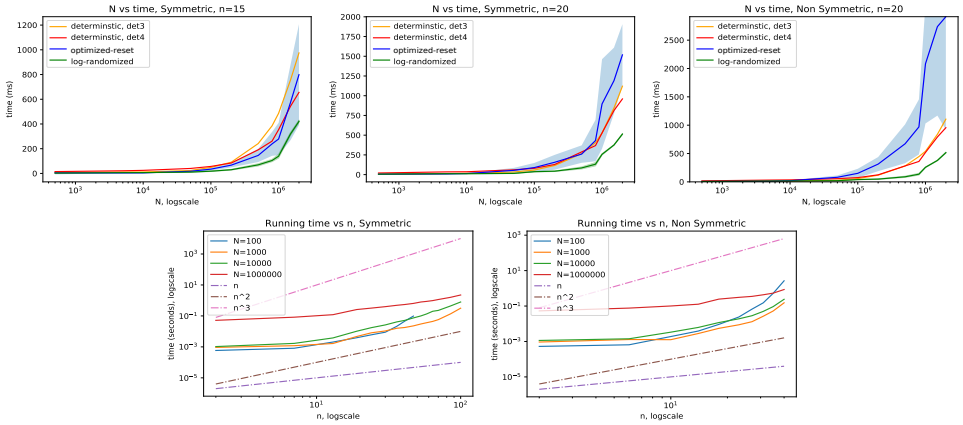

Figure 2: The top row compares the running time of randomized Algorithms against deterministic algorithms as N varies for *symmetric1* (left), *symmetric2* (middle) and *non-symmetric* (right). The shaded area represents the standard deviation (from 70 repetitions of the experiment). The bottom row running time of the *log-optimized* algorithm as n varies for different N (average of 70 samples).

Fast mean-square solvers. In [6] a measure reduction was used to accelerate the least squares method, i.e. the solution of the minimization problem $\min_w \|\mathbf{X}w-\mathbf{Y}\|^2$ where $\mathbf{X}\in\mathbb{R}^{N\times d}$ and $\mathbf{Y}\in\mathbb{R}^N$; as in Proposition 4, $\mathbf{X}$ denotes a matrix which has as row vectors the elements of $\mathbf{x}$, similarly for $\mathbf{Y}$. Theorem 2 guarantees the existence of a subset of $n+1=(d+1)(d+2)/2+1$ points $(\mathbf{x}^\star,\mathbf{y}^\star)$ of $\mathbf{x}$ and $\mathbf{y}$ such that $(\mathbf{X}|\mathbf{Y})^\top\cdot(\mathbf{X}|\mathbf{Y})=(\mathbf{X}^\star|\mathbf{Y}^\star)^\top\cdot(\mathbf{X}^\star|\mathbf{Y}^\star)$ where we denote with $(\mathbf{X}|\mathbf{Y})$ the element of $\mathbb{R}^{N\times(d+1)}$ formed by adding $\mathbf{Y}$ as a column. However, this implies that $\|\mathbf{X}w-\mathbf{Y}\|^2=\|\mathbf{X}^\star w-\mathbf{Y}^\star\|^2$ for every w, hence it is sufficient to solve the least square problem in much lower dimensions once $\mathbf{X}^\star$ and $\mathbf{Y}^\star$ have been found. We use the following datasets from [6](i) 3D Road Network [17] that contains 434874 records and use the two attributes longitude and latitude to predict height, (ii) Household power consumption [18] that contains 2075259 records and use the two 2 attributes active and reactive power to predict voltage. We also add a synthetic dataset, (iii) where $\mathbf{X}\in\mathbb{R}^{N\times n}$, $\boldsymbol{\theta}\in\mathbb{R}^n$ and $\boldsymbol{\varepsilon}\in\mathbb{R}^N$ are normal random variables, and $\mathbf{Y}=\mathbf{X}\boldsymbol{\theta}+\boldsymbol{\varepsilon}$ which allows to study various regimes of N and n. The first row in Figure 3 shows the performance of Algorithm 2 with the Las Vegas reset (but without the divide and conquer) on the datasets (i),(ii). The second row in Figure 3 shows the same but when Algorithm 2 is run with both, the Las Vegas reset and the divide and conquer optimization. We observe that already Algorithm 2 with Las Vegas resets is on average faster but the running time distribution has outliers where the algorithm takes longer than for the deterministic run

time algorithms; combined with divide and conquer the variance is reduced by a lot. Figure 4 shows the results on the synthetic dataset (iii) for various values of N and $n = (d+1)(d+2)/2$.

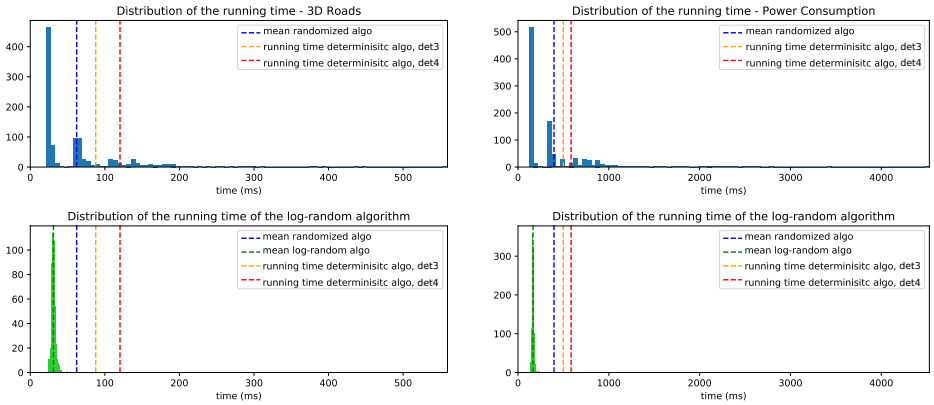

Figure 3: Histogram of the running time of Algorithm 2 with a reset strategy and of the "divide and conquer" variation algorithm (*log-random*).

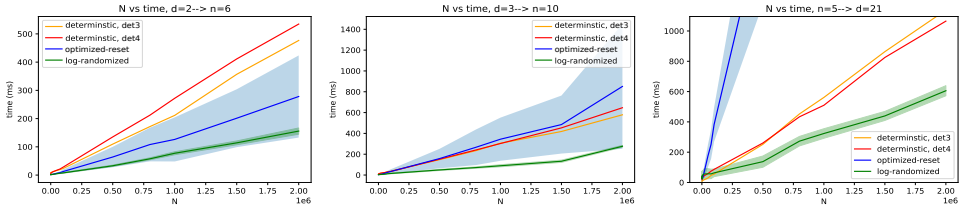

Figure 4: Performance of the different algorithms on the synthetic data set (iii) for various values of N and $n = (d+1)(d+2)/2$.

Breaking the randomized Algorithm. As the above experiments show, Algorithm 2 can lead to big speed ups. However, it is not uniformly better than the deterministic algorithms, and there are situations where one should not use it: firstly, Algorithm 2 was optimized to work well in the $N \gg n$ regime and while this is an important regime for data science, the recombination problem itself is of interest also in other regimes [4]. Secondly, the Las Vegas resets give a finite running time, but it is easy to construct examples where this can be much worse than the deterministic algorithms. Arguably the most practically relevant issue is when the independence hypothesis of Theorem 3 is not satisfied. This can appear in data sets with a high number of highly correlated categorical features, such as [19]. This can be overcome by using the Weyl Theorem, see Remark 1 in Appendix 6, but the computational cost is higher than computing the inverse of the cone basis (A in Theorem 3) and the benefits would be marginal, if not annulled, compared to the deterministic algorithms.

5 Summary

We introduced a randomized algorithm that reduces the support of a discrete measure supported on N atoms down to $n+1$ atoms while preserving the statistics captured in terms of n functions. The key was a characterization of the barycenter in terms of negative cones, that inspired a greedy sampling. Motivated by the geometry of cones this greedy sampling can be optimized, and finally combined with optimization methods for randomized algorithms. This yields a "greedy geometric sampling" that follows a very different strategy than previous deterministic algorithms, and that performs very well in the big data regime when $N \gg n$ as is often the case for large sample sizes common in machine learning applications.

Acknowledgements and Disclosure of Funding

The authors want to thank The Alan Turing Institute and the University of Oxford for the financial support given. FC is supported by The Alan Turing Institute, TU/C/000021, under the EPSRC Grant No. EP/N510129/1. HO is supported by the EPSRC grant "Datasig" [EP/S026347/1], The Alan Turing Institute, and the Oxford-Man Institute.

References

[1] V. Tchakaloff. Formules de cubature mécanique à coefficients non négatifs. *Bulletin des Sciences Mathématiques*, 81:123–134, 1957.

[2] Christian Bayer and Josef Teichmann. The proof of tchakaloff's theorem. *Proceedings of the American Mathematical Society*, 134(10):3035–3041, oct 2006.

[3] Philip J. Davis. A construction of nonnegative approximate quadratures. *Mathematics of Computation*, 21:578–582, 1967.

[4] Christian Litterer, Terry Lyons, et al. High order recombination and an application to cubature on wiener space. *The Annals of Applied Probability*, 22(4):1301–1327, 2012.

[5] Maria Tchernychova. *Caratheodory cubature measures*. PhD thesis, University of Oxford, 2016.

[6] Alaa Maalouf, Ibrahim Jubran, and Dan Feldman. Fast and accurate least-mean-squares solvers. In *Advances in Neural Information Processing Systems*, pages 8305–8316, 2019.

[7] Federico Piazzon, Alvise Sommariva, and Marco Vianello. Caratheodory-tchakaloff subsampling. *Dolomites Research Notes on Approximation*, 10(1), 2017.

[8] Satoshi Hayakawa. Monte carlo cubature construction. *arXiv preprint arXiv:2001.00843*, 2020.

[9] Pankaj K Agarwal, Sariel Har-Peled, and Kasturi R Varadarajan. Geometric approximation via coresets. *Combinatorial and computational geometry*, 52:1–30, 2005.

[10] Jeff M Phillips. Coresets and sketches. *arXiv preprint arXiv:1601.00617*, 2016.

[11] Jonathan H. Huggins, Trevor Campbell, and Tamara Broderick. Coresets for scalable bayesian logistic regression. *ArXiv*, abs/1605.06423, 2016.

[12] Dan Feldman, Melanie Schmidt, and Christian Sohler. Turning big data into tiny data: Constant-size coresets for k-means, pca and projective clustering. In *SODA*, 2013.

[13] Rolf Schneider and Wolfgang Weil. *Stochastic and integral geometry*. Probability and its Applications (New York). Springer-Verlag, Berlin, 2008.

[14] D. M. Y. Sommerville. The relations connecting the angle-sums and volume of a polytope in space of n dimensions. *Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character*, 115(770):103–119, 1927.

[15] Rommel G. Regis. On the properties of positive spanning sets and positive bases. *Optimization and Engineering. International Multidisciplinary Journal to Promote Optimization Theory & Applications in Engineering Sciences*, 17(1):229–262, 2016.

[16] Michael Luby, Alistair Sinclair, and David Zuckerman. Optimal speedup of Las Vegas algorithms. *Information Processing Letters*, 47(4):173–180, 1993.

[17] 3d road network, north jutland, denmark. https://archive.ics.uci.edu/ml/datasets/.

[18] Individual household electric power consumption. https://archive.ics.uci.edu/ml/datasets/.

[19] House sales in king county, usa. https://www.kaggle.com/harlfoxem/housesalesprediction.

[20] Günter M. Ziegler. *Lectures on polytopes*, volume 152 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 1995.

[21] James G Wendel. A problem in geometric probability. *Math. Scand*, 11:109–111, 1962.

Appendix

6 Properties of Algorithm 1

Background on polytopes. To prepare the proof of Theorem 3 we want to recall that a well-known tool from discrete geometry, is that polygons and polyhedral descriptions are equivalent. That is $C(\mathbf{x})$ is an affine span of the vectors $\mathbf{x}$ as in Definition 1, but equivalently $C(\mathbf{x})$ is the intersection of hyperplanes. In general this is the content of the celebrated Weyl–Minkowksi theorem and computing one representation from the other is non-trivial, see [20]. However, when restricted to n generic vectors in $\mathbb{R}^n$ (as is the case required in Theorem 3), one can immediately switch from one to the other, see item 2 of Theorem 3.

Proofs of Theorem 3 and Proposition 1

Theorem 3. *Let $\mathbf{x} = \{x_1, \dots, x_{n+1}\}$ be a set of $n+1$ points in $\mathbb{R}^n$ such that $\mathbf{x} \setminus \{x_{n+1}\}$ spans $\mathbb{R}^n$. Let A be the matrix that transforms $\mathbf{x} \setminus \{x_{n+1}\} = \{x_1, \dots, x_n\}$ to the orthonormal basis $\{e_1, \dots, e_n\}$ of $\mathbb{R}^n$, i.e. $Ax_i = e_i$. Further, let h_i be the unit vector such that $\langle h_i, x\rangle = 0$ for all $x \in \mathbf{x} \setminus \{x_i, x_{n+1}\}$ and $\langle h_i, x_i\rangle < 0$ and denote with $H_{\mathbf{x}\setminus\{x_{n+1}\}}$ a $n \times n$ matrix that has $h_1, \dots, h_n$ as row vectors. It holds that*

1. *$C(\mathbf{x} \setminus \{x_{n+1}\}) = \{c | H_{\mathbf{x}\setminus\{x_{n+1}\}} c \le 0\}$ and $C^-(\mathbf{x} \setminus \{x_{n+1}\}) = \{c | H_{\mathbf{x}\setminus\{x_{n+1}\}} c \ge 0\}$.*
2. *$Ax \ge 0$ if and only if $H_{\mathbf{x}\setminus\{x_{n+1}\}} x \le 0$. and $Ax \le 0$ if and only if $H_{\mathbf{x}\setminus\{x_{n+1}\}} x \ge 0$.*
3. *There exists a convex combination of $\mathbf{x}$ with 0 as barycentre, $\sum_{i=1}^{n+1} w_i x_i = 0$ for some $w_i > 0$, and $\sum_{i=1}^{n+1} w_i = 1$ if and only if $x_{n+1} \in C^-(\mathbf{x} \setminus \{x_{n+1}\})$.*

Proof. For item 1 first note that the h_i are well-defined since any set of $n-1$ independent points determines a hyperplane that includes 0 and that divides $\mathbb{R}^n$ into two parts. Each of these two parts is of the form $\{x : \langle h, x\rangle \le 0\}$ or $\{x : \langle h, x\rangle > 0\}$ and the additional condition $\langle h, x_i\rangle < 0$ selects one of the two parts. Now let $c = \sum_{x \in \mathbf{x}\setminus\{x_{n+1}\}} w_x x$ be a general vector. Since $H_{\mathbf{x}\setminus\{x_{n+1}\}} x \le 0$, for $x \in \mathbf{x} \setminus \{x_{n+1}\}$, it follows that $H_{\mathbf{x}\setminus\{x_{n+1}\}} c = \sum_{x \in \mathbf{x}\setminus\{x_{n+1}\}} w_x H_{\mathbf{x}\setminus\{x_{n+1}\}} x$ satisfies $H_{\mathbf{x}\setminus\{x_{n+1}\}} c \le 0$ if and only if the w_x, for $x \in \mathbf{x} \setminus \{x_{n+1}\}$, are positive.

For item 2, we can write $x = \sum_{i=1}^n w_i x_i$, for some $w_i \in \mathbb{R}$, hence $Ax = (w_1, \dots, w_n)^\top$. By definition

$$H_{\mathbf{x}\setminus\{x_{n+1}\}} x = \sum w_i H_{\mathbf{x}\setminus\{x_{n+1}\}} x_i = \sum w_i (\langle h_1, x_i\rangle, \dots, \langle h_n, x_i\rangle)^\top = (w_1 \langle h_1, x_1\rangle, \dots, w_n \langle h_n, x_n\rangle)^\top.$$

Note that $\langle h_i, x_i\rangle \le 0$ by definition, therefore sign $\{-Ax\}$ = sign $\{H_{\mathbf{x}\setminus\{x_{n+1}\}} x\}$. The statement with the reversed inequalities follows similarly.

For item 3($\Rightarrow$) assume there exists a convex combination of $\mathbf{x}$, this means that $x_{n+1} = -\frac{1}{w_{n+1}} \sum_{i=1}^n w_i x_i$, and

$$Ax_{n+1} = -\frac{1}{w_{n+1}} \sum_{i=1}^n w_i A x_i = \sum_{i=1}^n -\frac{w_i}{w_{n+1}} e_i.$$

Therefore, $Ax_{n+1} \le 0$ which is by item 2 equivalent to $H_{\mathbf{x}\setminus\{x_{n+1}\}} x \ge 0$. Thus, $x_{n+1} \in C^-(\mathbf{x} \setminus \{x_{n+1}\})$. Finally, for item 3($\Leftarrow$) assume that $x_{n+1} \in C^-(\mathbf{x} \setminus \{x_{n+1}\})$. The by item 2 $Ax_{n+1} \le 0$. Moreover, $\exists \lambda_i \in \mathbb{R}$ such that $x_{n+1} = \sum_{i=1}^n \lambda_i x_i$, therefore $Ax_{n+1} = \sum_{i=1}^n \lambda_i e_i = (\lambda_1, \dots, \lambda_n)^\top \le 0$. Let us call $\lambda^* := 1 - \sum_{i=1}^n \lambda_i$, by the decomposition of x_{n+1} we know that

$$\frac{1}{\lambda^*} x_{n+1} + \sum_{i=1}^n \frac{-\lambda_i}{\lambda^*} x_i = 0 \text{ and } \frac{1}{\lambda^*} + \sum_{i=1}^n \frac{-\lambda_i}{\lambda^*} = 1 \text{ and } \frac{1}{\lambda^*}, \frac{-\lambda_i}{\lambda^*} \ge 0.$$

□

Remark 1. *The assumption that $\{\mathbf{x} \setminus \{x_{n+1}\}\}$ spans $\mathbb{R}^n$ can be relaxed, indeed item 1 is a particular case of the Weyl's Theorem, which briefly does not require the independence of the cone basis.*
From an implementation point of view, however, item 2 gives an important boost, indeed the computation of $H_{\mathbf{x}\setminus\{x_{n+1}\}}$ is heavier than inverting a matrix, i.e. computing A, since it requires the computation of the coefficients of n different hyperplanes in $\mathbb{R}^n$. Moreover, speaking about the greedy searching strategy of Algorithm 2, using $H_{\mathbf{x}\setminus\{x_{n+1}\}}$ in place of A does not allow the use of the Sherman–Morrison formula weighing even more on the total computational cost.

Proposition 1. *Let $N > n+1$ and μ be a discrete probability with finite support and $f_1,\dots,f_n$ be as in Theorem 1. Moreover wlog assume $\mathbb{E}_{X\sim\mu}[f_i(X)] = 0$ for $i = 1,\dots,n$. With $p := \frac{n\cdot n!(N-n)!}{N!}$ it holds that $\mathbb{E}[\tau] \le \frac{1}{p}$ and $Var(\tau) \le \frac{1-p}{p^2}$ and, for fixed n, $\lim_{N\to\infty}\mathbb{E}[\tau] = 1$.*

Proof. Note that in every run through the loop, n points are randomly selected. However, by Theorem 3, Algorithm 1 finishes when the event

$$A := \{\text{for } n \text{ uniformly at random chosen points } \mathbf{x}^\star \text{ from } \mathbf{x},\ \exists x \in \mathbf{x}\ s.t.\ x \in C^-(\mathbf{x}^\star)\}.$$

occurs. Combined with Tchakaloff's Theorem guarantees this shows that there exists at least one set of n points $\mathbf{x}^\star$ such that $C^-(\mathbf{x}^\star) \neq \emptyset$ and therefore

$$\mathbb{P}(A) \ge \frac{\binom{n+1}{n}}{\binom{N}{n}} = \frac{n\cdot n!(N-n)!}{N!},$$

By independence, τ can be modelled by a geometric distribution with parameter $p = \mathbb{P}(A)$

$$\mathbb{P}(\tau = k) = (1-p)^{k-1}p$$

and the bounds for mean and variance follow.

□

7 Properties when applied to special measures

Proof of Proposition 2

Proposition 2. *Let $N > n+1$ and let $f_1,\dots,f_n$ be n real-valued functions and $X_1,\dots,X_N$ be N i.i.d. copies of a random variable X. Set $F(X) = (f_1(X),\dots,f_n(X))$, assume $\mathbb{E}[F(X)] = 0$ and denote*

$$E := \{0 \in \mathrm{Conv}\{F(X_i), i \in \{1,\dots,N\}\}\}. \tag{2}$$

1. *$\mathbb{E}[\tau|E] \le \frac{1}{p}$ and $Var(\tau|E) \le \frac{1-p}{p^2}$, where*

$$p = \max\left\{\frac{n\cdot n!(N-n)!}{N!}, 1 - \mathbb{P}\left(0 \notin \mathrm{Conv}\{F(X_1),\dots,F(X_{n+1})\}\right)^{N-n}\right\}, \tag{3}$$

2. *If the law of $F(X)$ is invariant under reflection in the origin, then $\mathbb{P}\left(0 \notin \mathrm{Conv}\{F(X_1),\dots,F(X_{n+1})\}\right) = 1 - 2^{-n}$,*

3. *For fixed n, as $N \to \infty$*

$$\mathbb{P}(\textit{for } n \textit{ uniformly at random chosen points } \mathbf{x}^\star \textit{ from } \mathbf{x},\ \exists x \in \mathbf{x}\ s.t.\ x \in C^-(\mathbf{x}^\star)) \to 1,$$

where $\mathbf{x} = \{F(X_1), F(X_2),\dots,F(X_N)\}$.

Proof. As in Proposition 1, the algorithm terminates when the event

$$A := \{\text{for } n \text{ uniformly at random chosen points } \mathbf{x}^\star \text{ from } \mathbf{x},\ \exists x \in \mathbf{x}\ s.t.\ x \in C^-(\mathbf{x}^\star)\}$$

happens, where $\mathbf{x} = \{F(X_1), F(X_2),\dots,F(X_N)\}$.
For item 1 denote $F_i := (f_1(X_{I_i}),\dots,f_n(X_{I_i}))$ where $\{I_1,\dots,I_N\}$ is a uniform shuffle of $\{1,\dots,N\}$, i.e. a random permutation of its elements that makes every rearrangement equally probable, then

$$A = \{\exists i \in \{n+1,\dots,N\} \text{ s.t. } F_i \in C^-(F_1,\dots,F_n)\}$$

and note that

$$\mathbb{P}(A|E) = \frac{\mathbb{P}(E|A)\mathbb{P}(A)}{\mathbb{P}(E)} \ge \mathbb{P}(A)$$

since by Theorem 3 $\mathbb{P}(E|A) = 1$ and $\mathbb{P}(E) > 0$ since $N \geq n+1$ and $\mathbb{E}F(X) = 0$. The estimate of Proposition 1 $\mathbb{P}(E|0 \in \text{Conv}\{F_i\}) \geq \frac{n \cdot n!(N-n)!}{N!}$ is still valid, moreover

$$\begin{aligned}\mathbb{P}(A|E) \geq \mathbb{P}(A) =& \mathbb{P}(\exists i \in \{n+1,\dots,N\} \text{ such that } F_i \in C^-(F_1,\dots,F_n)) \\ =& 1 - \prod_{j=n+1}^{N} \mathbb{P}(F_j \notin C^-(F_1,\dots,F_n)) \\ =& 1 - \mathbb{P}(F_{n+1} \notin C^-(F_1,\dots,F_n))^{N-n} \\ =& 1 - \mathbb{P}(0 \notin \text{Conv}(F_1,\dots,F_{n+1}))^{N-n}\end{aligned}$$

where the last equality follows from Theorem 3. We have therefore two different bounds for $\mathbb{P}(A|E)$, so we can take the maximum, i.e.

$$\mathbb{P}(A|E) \geq \max\left\{\frac{n \cdot n!(N-n)!}{N!}, 1 - \mathbb{P}\left(0 \notin \text{Conv}\{F_1,\dots,F_{n+1}\}\right)^{N-n}\right\}$$

Item 2. In [21] the author shows that when the F_i are distributed uniformly randomly on the unit sphere, then $\mathbb{P}(0 \notin \text{Conv}(F_1,\dots,F_{n+1}))^{N-n} = (1-2^{-n})^{N-n}$. In [13][Theorem 8.2.1] it is shown the same result, for all the symmetric distributions with respect to 0.
Now τ can be modelled by a geometric distribution with parameter $p = \mathbb{P}(A|E)$, i.e. $\mathbb{P}(\tau = i) \geq (1-p)^{i-1}p$ and the mean and variance follows.
For item 3, it is enough to show $\mathbb{P}(A) \to 1$:

$$\mathbb{P}(A) = \mathbb{P}(A\big|E) \times \mathbb{P}(E) + \mathbb{P}\left(A\big|E^C\right) \times \mathbb{P}\left(E^C\right) = \mathbb{P}(A\big|E) \times \mathbb{P}(E) + 0 \times \mathbb{P}\left(E^C\right) \to 1 \times 1,$$

as $N \to \infty$, where $\mathbb{P}\left(A\big|E^C\right) = 0$ is due to Theorem 3, the convergence $\mathbb{P}(E) \to 1$ is guaranteed by Theorem 5 and the convergence $\mathbb{P}(A\big|E) \to 1$, is guaranteed by the proof of item 1 for fixed n. □

As intuition suggest, the event that the mean is included in the convex hull occurs almost surely.

Theorem 5 ([8])**.** *Let $X_1,\dots,X_N$ be i.i.d. samples from a random variable X that has a first moment $\mathbb{E}[X] < \infty$. Then $\mathbb{P}(\mathbb{E}[X] \in \text{Conv}\{X_i\}_{i=1}^N) \to 1$, as $N \to \infty$.*

Sampling from empirical measures Often we do not know the distribution ϕ of the points or $\mathbb{E}[F(X)]$, moreover it could be that for the realized samples $\{F_i\}_{i=1}^N$, $\mathbb{E}[F(X)] \notin \text{Conv}\{F_i\}_{i=1}^N$. In these cases, due to Theorem 2, and since Algorithm 1 is based on Theorem 3, which assumes that the barycentre of the points given is 0, the input of the Algorithm is not the collection $\{F_i\}_{i=1}^N$, but

$$\hat{F}_i = (f_1(X_i),\dots,f_n(X_i)) - \left(\sum_{j=1}^{N} f_1(X_j)w_j,\dots,\sum_{j=1}^{N} f_n(X_j)w_j\right),$$

in this way we are sure that the barycentre is 0 and $0 \in \text{Conv}\{\hat{F}_i\}_{i=1}^N$, and the hypothesis of both Theorem 2 and 3 are satisfied. Unfortunately the $\{\hat{F}_i\}_{i=1}^N$ are not independent, which leads to an impossible analysis, even though the correlation between the $\hat{F}_i$ decreases when N becomes bigger and tends to 0. Thus, we can believe that the analysis of the proof of Proposition 2 is a good approximation of the complexity of the Algorithm 1, when the $\{\hat{F}_i\}_{i=1}^N$ are given as input and N is big "enough", as it is shown in Figure 5 and Figure 1 in case of symmetric distribution.

It is relevant to note at this point that we can always consider uniform measures, i.e. $\mu = \frac{1}{N}\sum_{i=1}^N \delta_{x_i}$, modifying the support of the measure, and then eventually go back to the original (not-uniform) measure.

Lemma 1. *Let us consider a set $\mathbf{x} = \{x_i\}_{i=1}^N$ in $\mathbb{R}^n$ and a sequence $\{\kappa_i\}_{i=1}^N$ of strictly positive numbers. There exists a measure μ on $\mathbf{x}$ such that $\mu(\mathbf{x}) = 0$ if and only if there exists a measure $\mu^\star$ on $\{\frac{x_i}{\kappa_i}\}_{i=1}^N$ such that $\mu^*(\{\frac{x_i}{\kappa_i}\}_{i=1}^N) = 0$.*

Proof. Let us assume that there exists μ on $\mathbf{x}$ such that $\mu(\mathbf{x}) = 0$, and let us call $\mu_i := \mu(x_i)$. It is enough to define $\mu^\star = \mu_i\kappa_i$. The other side of the equivalence is proved in the same way. □

Remark 2. *Lemma 1 is a consequence of the fact that a cone is defined only by the directions of the vectors of the "basis", and not from their length.*

Proposition 2 shows us a "universal strategy" to explore the space of all the combination of points more efficiently, i.e. choosing the basis of the cone to maximize the probability placed on its inverse. In other words, ideally we should try to maximize

$$\max_{F_i \in \mathbf{x}} \mathbb{P}\left(F(X) \in C^{-}(F_1, \ldots, F_n)\right).$$

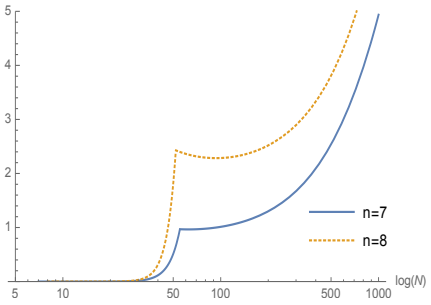

Figure 5: The plots shows the logplot of Equation (4). It can be seen that this is the same shape obtained with the experimental simulations in Section 4.

Figure 5 shows the complexity of Algorithm 1 in case of symmetric distributions; it can be noticed that it has a local minima N_n^*.

8 Properties of Algorithm 2

The case when $n = 2$.

Theorem 4. *Let* $\mathbf{x}$ *be a set of* $N \geq 3$ *points in* $\mathbb{R}^2$ *and* $x_1 \in \mathbf{x}$. *Define* $\mathbf{x}^\star = (x_1^\star, x_2^\star)$, *where*

$$x_2^\star := \arg\max_{x \in \mathbf{x} \setminus \{x_1^\star\}} \left| \frac{\langle x_1^\star, x \rangle}{\|x_1^\star\| \|x\|} - 1 \right|. \tag{5}$$

There exists a convex combination $\sum_{x \in \mathbf{x}} w_x x$ *of* $\mathbf{x}$ *that equals* 0 *if and only if* $\mathbf{x} \cap C^{-}(\mathbf{x}^\star) \neq \emptyset$.

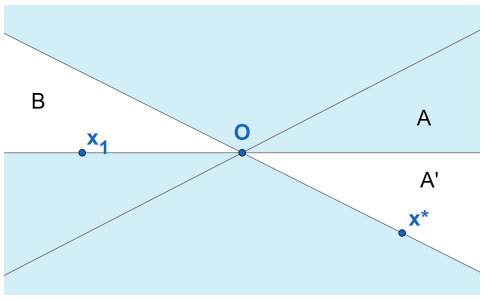

Figure 6: Proof of Theorem 4.

Proof. The proof follows for convex geometrical reasons. In Figure 6, we can see a general situation, the shaded areas indicate $C^{-}(x_1, x^\star)$ (on the top) and $C(x_1, x^\star)$ (on the bottom). Moreover, by definition of $x^\star$ on the region A and A' there are no points.

($\Rightarrow$) If there exists a point x_2 in $C^{-}(x_1, x^\star)$, then by simple convex arguments follow that there exists a convex combination of 0 for $x_1, x_2, x^\star$.

($\Leftarrow$) If does not exist a point in $C^{-}(x_1, x^\star)$, it means that there are points only in $B \cup C(x_1, x^\star)$, therefore, again for simple convex geometry arguments, it is impossible that there exists a convex combination of 0 for $x_1, x_2, x^\star$. □

Complexity of Algorithm 2

Proposition 3. *The complexity of Algorithm 2 to compute a reduced measure $\hat{\mu}$,as in Theorem 1, is*

$$O(n^3+n^2N)+(\tau-1)O(n^2+nN),$$

here $\tau=\inf\{i\geq 1 : C^-(\mathbf{X}_i)\cap\mathbf{x}\neq\emptyset\}$ where $\mathbf{X}_1,\mathbf{X}_2,\dots$ are obtained as in Algorithm 2.

Proof. The most expensive steps are the same ones as in Algorithm 1 and in addition the maximization problem. After the first step, given that we update one point of the basis at time, we can be more efficient using the Sherman–Morrison formula. Let us call $\mathbf{X}_t^\star$ the matrix whose rows are the vectors of the basis at the step t $\mathbf{x}_t^\star$, therefore we have $A_t=((\mathbf{X}_t^\star)^\top)^{-1}$, thus (shifting properly the vectors)

$$A_{t+1}=\left((\mathbf{X}_t^\star)^\top+(X^\star-X_1^\star)\cdot e_1^\top\right)^{-1}=A_t-\frac{A_t(X^\star-X_1^\star)e_1^\top A_t}{1+e_1^\top A_t(X^\star-X_1^\star)},$$

in the case we want to substitute the "first" vector of the basis $X_1^\star$ with the vector $X^\star$. Let us note that the only multiplications to be computed are $A_t(X^\star-X_1^\star)$ and $[A_t(X^\star-X_1^\star)]\cdot\left[e_1^\top A_t\right]$, which are done in $O(n^2)$ operations. To check if there are points inside the cone (or the inverse cone), we multiply the matrix A times the matrix $\mathbf{X}$ (of all the remaining vectors $\mathbf{X}$), and again after the first step costs $O(Nn^2)$, we can use the Sherman–Morrison formula as before and obtain

$$A_{t+1}\mathbf{X}^\top=A_t\mathbf{X}^\top-\frac{A_t(X^\star-X_1^\star)e_1^\top A_t\mathbf{X}^\top}{1+e_1^\top A_t(X^\star-X_1^\star)}.$$

Let us note that we have already computed $A_t\mathbf{X}^\top$ at the previous step, $A_t(X^\star-X_1^\star)$ to compute A_{t+1}, therefore the only cost is to compute $[A_t(X^\star-X_1^\star)]\cdot\left[e_1^\top A_t\mathbf{X}^\top\right]$, which is done in $O(nN)$ operations. The previous computations show us that after the first step, updating one element at a time, improves the computational efficiency of the successive steps of a factor n. Let us now tackle the maximization problem, it requires to compute the norm, i.e. $O(Nn^2)$ operations, which could be done only once. Moreover, the maximization problem requires to compute (part of) the sum of the vectors in $\mathbf{x}^\star$ and then the scalar product, which require $O(Nn)$. The last expensive operation we should consider is due to solve the last system to find the weights. The system we want to solve is

$$\begin{pmatrix}(\mathbf{X}^\star)^\top & X^\star\\ \mathbf{1} & 1\end{pmatrix}w=\begin{pmatrix}\mathbf{0}\\ 1\end{pmatrix},$$

where $\mathbf{0}$ is a $n\times 1$ vector of 0, and $\mathbf{1}$ is a $1\times n$ vector of 1. Using again the Sherman–Morrison formula, since we have already computed $A=(\mathbf{X}^\star)^{-1}$ the weights w_i can be computed as

$$\begin{aligned}w&=\begin{pmatrix}(\mathbf{X}^\star)^\top & X^\star\\ \mathbf{1} & 1\end{pmatrix}^{-1}\begin{pmatrix}\mathbf{0}\\ 1\end{pmatrix}=\begin{pmatrix}A^\top+A^\top X^\star c^{-1}\mathbf{1}A^\top, & -A^\top X^\star c^{-1}\\ -c^{-1}\mathbf{1}A^\top, & c^{-1}\end{pmatrix}\begin{pmatrix}\mathbf{0}\\ 1\end{pmatrix}\\ &=-\frac{1}{c}\begin{pmatrix}A^\top X^\star\\ 1\end{pmatrix},\end{aligned}$$

where $c=1-\mathbf{1}A^\top X^\star$ is a number. In this way we need $O(n^2)$ operations, not $O(n^3)$, i.e. the complexity of solving a linear system.
The total cost therefore is $O(n^3+n^2N)+(\kappa-1)O(n^2+nN)$. □

Remark 3. *The gain in the computational cost we obtain using the Sherman–Morrison formula has a cost in term of numerical stability.*

Robustness of the solution.

Proposition 4. *Assume that* $\operatorname{span}(\hat{\mathbf{x}})=\operatorname{span}(\hat{\mathbf{x}}_{-1})=\mathbb{R}^n$*, where $\hat{\mathbf{x}}_{-i}:=\hat{\mathbf{x}}\setminus\hat{x}_i$. Denote with $\mathbf{X}$ a matrix which as has rows the vectors in $\mathbf{x}$. Suppose there exists an invertible matrix R and another matrix E, such that $\mathbf{X}=\mathbf{Y}R+E$. Denote $\gamma_1:=(\hat{\mathbf{X}}_{-1}^\top)^{-1}\hat{X}_1^\top$, where $\hat{\mathbf{x}}$ is a solution to the RP $\mathbf{x}$. Assuming that the inverse matrices exist, $\hat{\mathbf{X}}R+E_{\hat{\mathbf{x}}}$ is a solution to the RP $\mathbf{y}$ if and only if*

$$\gamma_1^\top+E_{\hat{x}_1}R^{-1}A_1^\top\leq\left(\gamma_1^\top+E_{\hat{x}_1}R^{-1}A_1^\top\right)E_{\hat{\mathbf{x}}_{-1}}\left(I+R^{-1}A_1^\top E_{\hat{\mathbf{x}}_{-1}}\right)R^{-1}A_1^T$$

where E_y indicates the part of the matrix E related to the set of vectors $y\subset\mathbf{x}$ and $A_1=(\hat{\mathbf{X}}_{-1}^\top)^{-1}$.

Proof. From Theorem 3, we know that $\hat{\mathbf{X}}R + E_{\hat{\mathbf{x}}}$ is a solution if and only if $\left(\left(\hat{\mathbf{X}}_{-1}R + E_{\hat{\mathbf{x}}_{-1}}\right)^\top\right)^{-1}\left(\hat{X}_1 R + E_{\hat{x}_1}\right)^\top \leq 0$. Let us note that the last product is a vector, therefore we can study the transpose and using the Woodbury matrix identity we have that

$$\begin{aligned}\left(\hat{X}_1 R + E_{\hat{x}_1}\right)\left(\hat{\mathbf{X}}_{-1}R + E_{\hat{\mathbf{x}}_{-1}}\right)^{-1} &= \left(\hat{X}_1 + E_{\hat{x}_1}R^{-1}\right)\left(\hat{\mathbf{X}}_{-1} + E_{\hat{\mathbf{x}}_{-1}}R^{-1}\right)^{-1} \\ &= \left(\hat{X}_1 + E_{\hat{x}_1}R^{-1}\right)\left(I - A_1^T E_{\hat{\mathbf{x}}_{-1}}\left(I + R^{-1}A_1^T E_{\hat{\mathbf{x}}_{-1}}\right)R^{-1}\right)A_1^T.\end{aligned}$$

Setting the last equation less or equal than 0 shows the result. □

This also implies that the solution is invariant under rotations.

9 Divide and conquer, choice of the subgroup size

As mentioned in Section 3, to apply a divide and conquer strategy requires to balance the size of subgroups against the property of Algorithm 2 to exploit a large number of points as to maximize the likelihood of points being in the (inverse) cone. Let us explain how we have chosen $N_n^* = 50(n+1)$, which should be thought as linear approximation of the exact minimum for the complexity of Algorithm 2. Therefore first note, that as Figure 7 shows, choosing any number between 20 and 80,

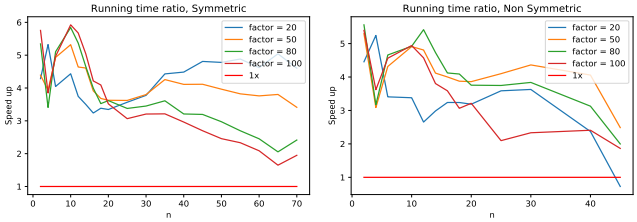

Figure 7: Bottom-right: it is shown how much the *optimized-reset* algorithm is faster than *det4* in case $N = (n+1) \times$ factor.

in place of 50, has similar effects if $n < 70$ in the case of symmetric distribution, whilst the same holds in the case of mixture of exponentials (non symmetric) if $n \leq 40$. We think that this effect is due to the fact that "experimentally" there exists a long plateau in the running time of the optimized Algorithms with and without reset, see Figure 1. Therefore, let us suppose that there exist k, K s.t. for any $k(n+1) \leq N^{(1)}, N^{(2)} \leq K(n+1)$ and $n < 40$, then $\bar{C}(N^{(1)}, n+1) \approx \bar{C}(N^{(2)}, n+1)$, where $\bar{C}(\cdot, n+1)$ is the computational cost to reduce $\cdot$ number of points in $\mathbb{R}^n$ using Algorithm 2. Moreover, we suppose that the $\arg\min_x \bar{C}(x, n+1) \in [k(n+1), K(n+1)]$. The previous two conditions are equivalent to the presence of the plateau in Figure 1, in correspondence with the minimum value of the running time, for the optimized Algorithms with and without reset. Under these assumptions, the best choice would be $N_n^* = K(n+1)$. Without knowing the value of K, however we can estimate the difference into the complexity for different group subdivisions: if we have $N \gg K(n+1)$ number of points, using the "divide and conquer" paradigm with $N^{(i)}$ groups, we can build two algorithms s.t. the difference of the computational costs is

$$\begin{aligned} O\left(Nn + \log_{N^{(1)}/n}(N/n)C(N^{(1)}, n+1)\right) - O\left(Nn + \log_{N^{(2)}/n}(N/n)C(N^{(2)}, n+1\right) \approx \\ \approx \bar{C}(Kn, n+1)\log_{N^{(2)}/n}(N/n)\left(\frac{1}{\log_{N^{(2)}/n}N^{(1)}/n} - 1\right). \end{aligned} \tag{6}$$

Therefore, we have that the difference depends on a factor $|1/\log_{N^{(2)}/n}(N^{(1)}/n) - 1|$. Given Figure 7, we have estimated approximately $k = 20$, $K = 80$ for the symmetric case, thus as a rule of thumb we assume that $N_n^* = 50(n+1)$ is a reasonable value, and in view of Equation (6) we can say that changing slightly 50 the running time would remain stable. The analogous argument can be made for the mixture of exponentials.

10 Implementation and benchmarking.

For *det1* we have used the code provided by the authors of [6] which is available at [*Github Link*] in Python; for *det2* the code of [5] has not been written in Python, therefore we have implemented it to allow for a fair comparison using standard Numpy libraries. We used throughout the same codeblocks for the Divide and Conquer strategy in all the implementations. We did not use the tree data-structure in [5] since it does not not change complexity bounds and is independent of the reduction procedure itself; however, it could be also used for *det1* and our randomized algorithms. Code for all experiments is available at [*Github Link*].