Mach Learn (2016) 104:129–148
DOI 10.1007/s10994-016-5548-x

Learning rotations with little regret

Elad Hazan[1] · Satyen Kale[2] · Manfred K. Warmuth[3]

Received: 7 October 2014 / Accepted: 25 January 2016 / Published online: 22 March 2016

Abstract We describe online algorithms for learning a rotation from pairs of unit vectors in $\mathbb{R}^n$. We show that the expected regret of our online algorithm compared to the best fixed rotation chosen offline over T iterations is $\sqrt{nT}$. We also give a lower bound that proves that this expected regret bound is optimal within a constant factor. This resolves an open problem posed in COLT 2008. Our online algorithm for choosing a rotation matrix is essentially an incremental gradient descent algorithm over the set of all matrices, with specially tailored projections. We also show that any deterministic algorithm for learning rotations has $\Omega(T)$ regret in the worst case.

Keywords Rotations · Online learning · Regret bounds · Bregman projection · Minimax

1 Introduction

Rotations are a fundamental object in robotics and vision and the problem of learning rotations, or finding the underlying rotation from a given set of examples, has numerous applications [see Arora (2009) for a summary of application areas, including computer vision, face

Editor: Alexander Rakhlin.

Supported by ISF Grant 810/11 and a Google Research Award. Supported by NSF Grant IIS-0917397.

✉ Manfred K. Warmuth
manfred@ucsc.edu

Elad Hazan
ehazan@cs.princeton.edu

Satyen Kale
satyen@satyenkale.com

[1] Department of Computer Science, Princeton University, Princeton, NJ, USA

[2] Yahoo Research, New York, NY, USA

[3] Department of Computer Science, UC Santa Cruz, Santa Cruz, CA, USA

recognition, robotics, crystallography, and physics]. Besides their practical importance, rotations have been shown to be powerful enough to capture seemingly more general mappings. Rotations can represent arbitrary Euclidean transformations via a conformal embedding by adding two special dimensions (Wareham et al. 2005). Also Doran et al. (1993) showed that the rotation group provides a universal representation for all Lie groups.

In the batch setting, the problem of learning a rotation was originally posed as the problem of estimating the altitude of satellites by Wahba (1966). The related problem of learning orthogonal, rather than rotation, matrices is known as the "orthogonal Procrustes problem" (see Schonemann 1966). Algorithms for optimizing a static cost function over the set of unitary matrices were proposed by Abrudan et al. (2008a, b) using descent over Riemannian manifolds.

The question of whether there are online algorithms for this problem was explicitly posed as an open problem in COLT 2008 (Smith and Warmuth 2008). An online algorithm for learning rotations was given by Arora (2009). This algorithm exploits the Lie group/Lie algebra relationship between rotation matrices and skew symmetric matrices respectively, and the matrix exponential and matrix logarithm that maps between these matrix classes. However, this algorithm deterministically predicts with a single rotation matrix in each trial. In this paper, we prove that any such deterministic algorithm can be forced to have regret at least $\Omega(T)$, where T is the number of trials. In contrast, we give an algorithm that produces a random rotation in each trial and has expected regret $\sqrt{nT}$, where n is the dimension of the matrices.

1.1 Technical contributions

The main technique used in this paper is a new variant of online gradient descent with Euclidean projections (see e.g. Herbster and Warmuth 2001; Zinkevich 2003) that uses what we call "lazy projections". A straightforward implementation of the original algorithm requires $O(n^3)$ time per iteration, because in each round we need to perform a Euclidean projection of the parameter matrix onto the convex hull of orthogonal matrices, and this projection requires the computation of a singular value decomposition. The crucial new idea is to project the parameter matrix onto a convex set determined by the current instance that contains the convex hull of all orthogonal matrices as a subset. The projection onto this larger set can be done easily in $O(n^2)$ time and needs to be performed only when the current parameter matrix is outside of the set. Surprisingly, our new algorithm based on this idea of "lazy projections" has the same optimal regret bound but requires only $O(n^2)$ time per iteration.

The loss function for learning rotations can be expressed as a linear function. Such linear losses are special because they are the least convex losses. The main case where such linear losses have been investigated is in connection with the Hedge and the Matrix Hedge algorithm (Freund and Schapire 1997; Warmuth and Kuzmin 2011). However for the latter algorithms the parameter space is one-norm or trace norm bounded, respectively. In contrast, the implicit parameter space of the main algorithm of this paper is essentially infinity norm bounded, i.e. the convex hull of orthogonal matrices consists of all square matrices with singular values at most one.

1.2 Outline of paper

We begin with some preliminaries in the next section, the precise online learning problem for rotations, and basic properties of rotations. We also show how to solve the corresponding off-line problem exactly. We then give in Sect. 3 our main probabilistic algorithm and prove a $\sqrt{nT}$ regret bound for it. This bound cannot be improved by more than a constant factor, since we can prove a lower bound of essentially $\sqrt{\frac{nT}{2}}$ (for the case when $T \geq n$).

For the sake of completeness we also consider deterministic algorithms in Sect. 4. In particular we show that any deterministic algorithm can be forced to have regret at least T in general. In Sect. 5 we then present the optimal randomized and deterministic algorithms for the special case when there is a rotation that is consistent with all examples. A number of open problems are discussed in final Sect. 6. In "Appendix 1" we prove a lemma that characterizes the solution to the batch optimization problem for learning rotations and in "Appendix 2" we show that the convex hull of orthogonal matrices consists of all matrices with maximum singular value one.

1.3 Historical note

A preliminary version of this paper appeared in COLT 2010 (Hazan et al. 2010b). Unfortunately the algorithm we presented in the conference (based on the Follow the Perturbed Leader paradigm) was flawed: the true regret bound it obtains is $O(n\sqrt{T})$ as opposed to the claimed bound of $O(\sqrt{nT})$. After noticing this we published a corrigendum [posted on the conference website, Hazan et al. (2010a)] with a completely different technique based on online gradient descent that obtained the optimal regret $O(\sqrt{nT})$. The algorithm presented in this paper is similar, but much more efficient, taking $O(n^2)$ time per iteration rather than $O(n^3)$.

2 Preliminaries and problem statement

2.1 Notation

In this paper, all vectors lie in $\mathbb{R}^n$ and all matrices in $\mathbb{R}^{n\times n}$. We use $\det(\mathbf{M})$ to denote the determinant of matrix $\mathbf{M}$. An *orthogonal matrix* is a matrix $\mathbf{R}$ whose columns and rows are orthogonal unit vectors, i.e. $\mathbf{R}^\top\mathbf{R} = \mathbf{R}\mathbf{R}^\top = \mathbf{I}$, where $\mathbf{I}$ is the identity matrix. We let $\mathcal{O}(n)$ denote the set of all orthogonal matrices of dimension $n \times n$ and $\mathcal{SO}(n)$ denote the special orthogonal group of *rotation matrices*, which are all orthogonal matrices of determinant one. Since for $n = 1$ there is exactly one rotation matrix (i.e. $\mathcal{SO}(1) = \{1\}$), the problem becomes trivial, so throughout this paper we assume that $n > 1$.

For any vector $\mathbf{x}$, $\|\mathbf{x}\|$ denotes the ℓ_2 norm and for any matrix $\mathbf{A}$, $\|\mathbf{A}\|_F = \sqrt{\mathrm{tr}(\mathbf{A}\mathbf{A}^\top)}$ is the Frobenius norm. All regret bounds of this paper immediately carry over to the complex domain: replace orthogonal by unitary matrices and rotation matrices by unitary matrices of determinant one.

2.2 Online learning of rotations problem

Learning proceeds in a series of trials. In every iteration for $t = 1, 2, \ldots, T$:

1. The online learner is given a unit instance vector $\mathbf{x}_t$ (i.e. $\|\mathbf{x}_t\| = 1$).
2. The learner is then required to predict (deterministically or randomly) with a unit vector $\hat{\mathbf{y}}_t$.
3. Finally, the algorithm obtains the "true" result, which is also a unit vector $\mathbf{y}_t$.

4. The loss to the learner then is half the squared norm of the difference between her predicted vector and the "true" rotated vector $\mathbf{y}_t$:

$$\tfrac{1}{2}\|\hat{\mathbf{y}}_t - \mathbf{y}_t\|^2 \;=\; 1 - \mathbf{y}_t^\top \hat{\mathbf{y}}_t \tag{1}$$

5. If $\hat{\mathbf{y}}_t$ is chosen probabilistically, then we define the expected loss as

$$\mathrm{E}\left[\tfrac{1}{2}\left\|\mathbf{y}_t - \hat{\mathbf{y}}_t\right\|^2\right] \;=\; \tfrac{1}{2}\mathrm{E}\left[\|\hat{\mathbf{y}}_t - \mathbf{y}_t\|^2\right] \;=\; 1 - \mathbf{y}_t^\top \mathrm{E}[\hat{\mathbf{y}}_t].$$

Note that the loss is linear in the prediction vector $\hat{\mathbf{y}}_t$ or the expected prediction vector $\mathrm{E}[\hat{\mathbf{y}}_t]$, respectively. The goal of the learner is to minimize the regret on all T examples against the best fixed rotation $\mathbf{R}$ chosen in hindsight:

$$\mathrm{Regret}_T \;=\; \sum_{t=1}^{T} \mathrm{E}\left[\tfrac{1}{2}\left\|\mathbf{y}_t - \hat{\mathbf{y}}_t\right\|^2\right] - \min_{\mathbf{R}\in\mathcal{SO}(n)} \sum_{t=1}^{T} \tfrac{1}{2}\left\|\mathbf{y}_t - \mathbf{R}\mathbf{x}_t\right\|^2. \tag{2}$$

In this paper we give a probabilistic online algorithm with worst-case regret $\sqrt{nT}$ and an adversary strategy that forces any probabilistic algorithm to have have regret essentially $\sqrt{\frac{1}{2}nT}$. Note that since our loss is linear in the rotation $\mathbf{R}$, the best total loss $\min_{\mathbf{R}\in\mathcal{SO}(n)} \sum_{t=1}^{T} \frac{1}{2}\left\|\mathbf{y}_t - \mathbf{R}\mathbf{x}_t\right\|^2$ cannot be decreased by allowing mixtures of rotations.

When $n > 1$, then any unit vector can be rotated into any other unit vector. Namely one can always produce an explicit rotation matrix $\mathbf{R}_t$ in Step 2 that rotates $\mathbf{x}_t$ to $\hat{\mathbf{y}}_t$ (that is, $\hat{\mathbf{y}}_t$ in the definition of regret (2) is replaced by $\mathbf{R}_t\mathbf{x}_t$). Such a rotation matrix can be computed in $O(n^2)$ time, as the following lemma shows.

Lemma 1 *Let* $\mathbf{x}$ *and* $\hat{\mathbf{y}}$ *be two unit vectors. Then we can find an explicit rotation matrix* $\mathbf{R}$ *that rotates* $\mathbf{x}$ *onto* $\hat{\mathbf{y}}$*, i.e.* $\mathbf{R}\mathbf{x} = \hat{\mathbf{y}}$*, in* $O(n^2)$ *time.*

Proof We first take care of the case when $\hat{\mathbf{y}} = \pm\mathbf{x}$: If $\hat{\mathbf{y}} = \mathbf{x}$ we can simply let $\mathbf{R} = \mathbf{I}$; if $\hat{\mathbf{y}} = -\mathbf{x}$ and n is even, then we can use $\mathbf{R} = -\mathbf{I}$; finally, if $\hat{\mathbf{y}} = -\mathbf{x}$ and n is odd, then choose $\mathbf{R} = -\mathbf{I} + 2\mathbf{z}\mathbf{z}^\top$, where $\mathbf{z}$ is an arbitrary unit vector orthogonal to $\mathbf{x}$. In all these cases, $\mathbf{R}\mathbf{x} = \hat{\mathbf{y}}$ and $|\mathbf{R}| = 1$.

For the remaining case ($\hat{\mathbf{y}} \neq \pm\mathbf{x}$), let d denote the dot product $\mathbf{x}\cdot\hat{\mathbf{y}}$ and let $\hat{\mathbf{y}}^\perp$ be the normalized component of $\hat{\mathbf{y}}$ that is perpendicular to $\mathbf{x}$, i.e. $\hat{\mathbf{y}}^\perp = \frac{\hat{\mathbf{y}} - d\mathbf{x}}{\|\hat{\mathbf{y}} - d\mathbf{x}\|}$. Let $\mathbf{U}$ be an orthogonal matrix with $\mathbf{x}$ and $\hat{\mathbf{y}}^\perp$ as its first two columns. Now define $\mathbf{R}$ as $\mathbf{U}\mathbf{C}\mathbf{U}^\top$, where

$$\mathbf{C} = \begin{pmatrix} d & -\sqrt{1-d^2} & 0 & 0 & \cdots & 0 \\ \sqrt{1-d^2} & d & 0 & 0 & \cdots & 0 \\ 0 & 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 1 \end{pmatrix}.$$

All unspecified off-diagonal entries are 0, and all diagonal entries starting from the third row are 1. Now $\mathbf{R}$ is a rotation matrix that satisfies the requirements because

$$\begin{aligned} \mathbf{R}\mathbf{x} &= \mathbf{U}\mathbf{C}\mathbf{U}^\top\mathbf{x} = \mathbf{U}\mathbf{C}\,(1,0,0,\ldots,0)^\top \\ &= \mathbf{U}\,(d, \sqrt{1-d^2}, 0, 0, \ldots, 0)^\top \\ &= d\,\mathbf{x} + \sqrt{1-d^2}\,\hat{\mathbf{y}}^\perp = \hat{\mathbf{y}}, \\ \mathbf{R}\mathbf{R}^\top &= \mathbf{U}\mathbf{C}\mathbf{C}^\top\mathbf{U}^\top = \mathbf{U}\mathbf{I}\mathbf{U}^\top = \mathbf{I} \quad \text{and} \quad \det(\mathbf{R}) = \det(\mathbf{U})\det(\mathbf{C})\det(\mathbf{U}) = \det(\mathbf{C}) = 1. \end{aligned}$$

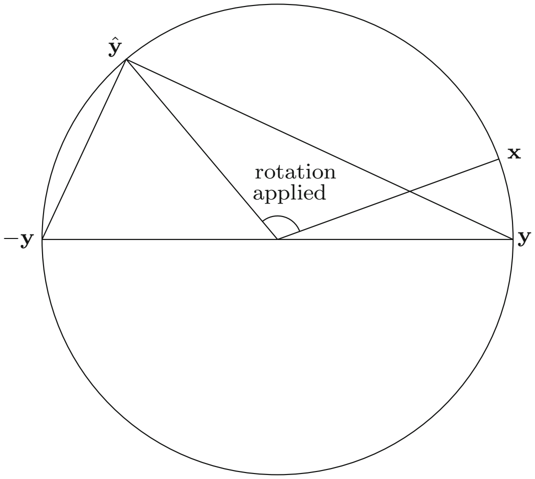

Fig. 1 For two examples $(\mathbf{x}, \mathbf{y})$ and $(\mathbf{x}, -\mathbf{y})$, the matrix $\mathbf{S} = \mathbf{x}\mathbf{y}^\top - \mathbf{x}\mathbf{y}^\top = 0$. Thus, any rotation matrix $\mathbf{R}$ has the same total loss. In particular, if $\hat{\mathbf{y}} = \mathbf{R}\mathbf{x}$ for any rotation matrix $\mathbf{R}$, then the total loss is exactly 2: $\frac{1}{2}\|\mathbf{y} - \hat{\mathbf{y}}\|^2 + \frac{1}{2}\|{-\mathbf{y}} - \hat{\mathbf{y}}\|^2 = 2 - \left(\mathbf{y}^\top - \mathbf{y}^\top\right)\hat{\mathbf{y}} = 2 - 0 = 2$. Thus the loss is at least 1 for at least one of the examples regardless of how $\hat{\mathbf{y}}$ was chosen. We exploit this kind of argument in our lower bounds

Note that $\mathbf{R}$ can be computed in $O(n^2)$ time by rewriting it as

$$\begin{aligned}\mathbf{R} = \mathbf{I} + \mathbf{U}(\mathbf{C} - \mathbf{I})\mathbf{U}^\top &= \mathbf{I} + (d-1)\left(\mathbf{x}\mathbf{x}^\top + \hat{\mathbf{y}}^\perp\left(\hat{\mathbf{y}}^\perp\right)^\top\right)\\ &\quad + \sqrt{1-d^2}\left(\hat{\mathbf{y}}^\perp\mathbf{x}^\top - \mathbf{x}\left(\hat{\mathbf{y}}^\perp\right)^\top\right).\end{aligned}$$

□

2.3 Solving the offline problem

Before describing our online algorithm, we need to understand how to solve the optimization problem of offline (batch) algorithm:

$$\underset{\mathbf{R}\in\mathcal{SO}(n)}{\operatorname{argmin}} \sum_{t=1}^T \frac{1}{2}\|\mathbf{y}_t - \mathbf{R}\mathbf{x}_t\|^2 = \underset{\mathbf{R}\in\mathcal{SO}(n)}{\operatorname{argmin}}\ T - \left(\sum_{t=1}^T \mathbf{y}_t^\top \mathbf{R}\mathbf{x}_t\right) \tag{3}$$

$$= \operatorname{argmax}_{\mathbf{R}\in\mathcal{SO}(n)} \operatorname{tr}\left(\left(\sum_{t=1}^T \mathbf{x}_t\mathbf{y}_t^\top\right)\mathbf{R}\right). \tag{4}$$

The first equality follows from rewriting the loss function $\frac{1}{2}\|\mathbf{y}_t - \mathbf{R}\mathbf{x}_t\|^2$ as in (1). Note that the T examples $(\mathbf{x}_t, \mathbf{y}_t)$ only enter into the optimization problem via the matrix $\mathbf{S} := \sum_{t=1}^T \mathbf{x}_t\mathbf{y}_t^\top$. This matrix functions as a "sufficient statistic" of the examples. As we shall see later, our randomized online algorithm will also be based on this sufficient statistic.

In general, an optimization problem of the form

$$\operatorname{argmax}_{\mathbf{R}\in\mathcal{SO}(n)} \operatorname{tr}(\mathbf{S}\mathbf{R}) \tag{5}$$

for some matrix $\mathbf{S}$, is a classical problem known as Wahba's problem (Wahba 1966). Figure 1 gives a simple example of the challenges in solving Wahba's problem: the optimum rotation matrix may not be unique (any rotation matrix $\mathbf{R}$ has the same loss on the two examples given in Fig. 1).

Nevertheless, the value of Wahba's problem (5) has a very elegant solution expressed i.t.o. the singular value decomposition (SVD) of $\mathbf{S}$.

Lemma 2 *Let* $\mathbf{S} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\top$ *be any SVD of* $\mathbf{S}$*, i.e.* $\mathbf{U}$ *and* $\mathbf{V}$ *are orthogonal matrices, and* $\boldsymbol{\Sigma} = diag(\sigma_1, \sigma_2, \ldots, \sigma_n)$ *is the diagonal matrix of non-negative singular values. Assume that* σ_n *is the smallest singular value and let* $s := \det(\mathbf{U})\det(\mathbf{V})$*. Since* $\mathbf{U}$ *and* $\mathbf{V}$ *are orthogonal,* $s \in \{+1, -1\}$*. Now if* $\mathbf{W} := diag(1, 1, \ldots, 1, s)$*, then*

$$\mathbf{V}\mathbf{W}\mathbf{U}^\top \in \operatorname{argmax}_{\mathbf{R} \in \mathcal{SO}(n)} \operatorname{tr}(\mathbf{S}\mathbf{R}),$$

and the value of the optimal solutions is $\sum_{i=1}^{n-1} \sigma_i + s\sigma_n$*, which is always non-negative.*

By (4), the solution to Wahba's problem is obtained by solving the argmax problem of the above lemma with $\mathbf{S} = \sum_{t=1}^{T} \mathbf{x}_t \mathbf{y}_t^\top$ and the value of the original optimization problem (3) is $T - \sum_{i=1}^{n-1} \sigma_i - s\sigma_n$.

Note that the solution $\mathbf{W}$ given in the lemma is a rotation matrix since it is a product of three orthogonal matrices, and its determinant equals $\det(\mathbf{U})\det(\mathbf{V})\det(\mathbf{W}) = 1$. The solution can be found in $O(n^3)$ time by constructing a SVD of $\mathbf{S}$.

We have been unable to find a complete proof of the above lemma for dimensions more than 3 in the literature and therefore, for the sake of completeness, we give a self-contained proof in "Appendix 1".

Note that if we are simply optimizing over all orthogonal matrices $\mathbf{R} \in \mathcal{O}(n)$, with no condition on $\det(\mathbf{R})$, then we arrive at another classical problem known as the "orthogonal Procrustes problem" (first solved by Schonemann 1966). The solution for this simpler problem is also given in "Appendix 2" for completeness:

Lemma 3 *Let* $\mathbf{S} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\top$ *be a SVD of* $\mathbf{S}$ *as in Lemma 2. Then*

$$\mathbf{V}\mathbf{U}^\top \in \operatorname{argmax}_{\mathbf{R} \in \mathcal{O}(n)} \operatorname{tr}(\mathbf{S}\mathbf{R})$$

and the value of the optimum solutions is $\sum_{i=1}^{n} \sigma_i$*.*

3 The randomized algorithm and main theorem

3.1 The algorithm

We begin by presenting our main randomized algorithm, called "Lazy Projection GD" (see Algorithm 1).

Algorithm 1 Lazy Projection GD

```
Let $\mathbf{W}_0$ be the all zero matrix
for $t = 1$ to $T$ do
    Obtain instance vector $\mathbf{x}_t$
    Compute $\mathbf{z}_t = \mathbf{W}_{t-1}\mathbf{x}_t$ and $\widetilde{\mathbf{z}}_t = \begin{cases} \frac{\mathbf{z}_t}{\|\mathbf{z}_t\|} & \text{if } \mathbf{z}_t \neq \mathbf{0} \\ \mathbf{e}_1 & \text{otherwise} \end{cases}$
    Choose prediction vector $\hat{\mathbf{y}}_t = \begin{cases} \pm\widetilde{\mathbf{z}}_t \text{ with probability } \frac{1\pm\|\mathbf{z}_t\|}{2}, & \text{if } \|\mathbf{z}_t\| \leq 1 \\ \widetilde{\mathbf{z}}_t \text{ otherwise} \end{cases}$
    Compute $\mathbf{W}_t^m = \begin{cases} \mathbf{W}_{t-1} & \text{if } \|\mathbf{z}_t\| \leq 1 \\ \mathbf{W}_{t-1}\left(\mathbf{I} - \left(1 - \frac{1}{\|\mathbf{z}_t\|}\right)\mathbf{x}_t\mathbf{x}_t^\top\right) & \text{otherwise} \end{cases}$
    Observe result vector $\mathbf{y}_t$ and suffer expected loss $\mathrm{E}[\frac{1}{2}\|\mathbf{y}_t - \hat{\mathbf{y}}_t\|^2] = 1 - \mathbf{y}_t^\top \mathrm{E}[\hat{\mathbf{y}}_t] = 1 - \mathbf{y}_t^\top \mathbf{W}_t^m \mathbf{x}_t$
    Update $\mathbf{W}_t = \mathbf{W}_t^m + \eta\, \mathbf{y}_t \mathbf{x}_t^\top$
end for
```

The algorithm maintains a parameter matrix $\mathbf{W}_{t-1}$, which intuitively models a distribution for rotation matrices that transform $\mathbf{x}_t$ onto $\hat{\mathbf{y}}_t$. The goal will be to predict with unit vector $\hat{\mathbf{y}}_t$ such that $\mathrm{E}(\hat{\mathbf{y}}_t) = \mathbf{W}_{t-1}\mathbf{x}_t$. This is not quite possible if $\|\mathbf{W}_{t-1}\mathbf{x}_t\| > 1$. Therefore during the t-th trial in Step 6 of the algorithm, $\mathbf{W}_{t-1}$ is updated to an intermediate parameter matrix $\mathbf{W}_t^m$ which makes it possible that the algorithm can predict with a unit vector $\hat{\mathbf{y}}_t$ such that $\mathrm{E}(\hat{\mathbf{y}}_t) = \mathbf{W}_t^m\mathbf{x}_t$. As shown in Lemma 4, the update rule for obtaining $\mathbf{W}_t^m$ from $\mathbf{W}_{t-1}$ in Step 6 is a Bregman projection with respect to the squared Frobenius norm onto the set of matrices $\mathbf{W}$ for which $\|\mathbf{W}\mathbf{x}\| \leq 1$:

$$\mathbf{W}_t^m = \operatorname*{argmin}_{\mathbf{W}:\|\mathbf{W}\mathbf{x}_t\|^2\leq 1} \tfrac{1}{2}\|\mathbf{W} - \mathbf{W}_{t-1}\|_F^2.$$

This ensures that $\|\mathbf{W}_t^m\mathbf{x}_t\| \leq 1$, making it possible to predict with unit vector $\hat{\mathbf{y}}_t$ such that $\mathrm{E}(\hat{\mathbf{y}}_t) = \mathbf{W}_t^m\mathbf{x}_t$.

The prediction $\hat{\mathbf{y}}_t$ is computed as follows. There are two main cases depending on the length of $\mathbf{z}_t = \mathbf{W}_{t-1}\mathbf{x}_t$. When $\|\mathbf{z}_t\| \leq 1$, then $\mathbf{z}_t$ is not too long and unit vector $\widetilde{\mathbf{z}}_t := \frac{\mathbf{z}_t}{\|\mathbf{z}_t\|}$ in direction $\mathbf{z}_t$ is used for choosing the prediction.[1] More precisely, the algorithm predicts with $\hat{\mathbf{y}}_t = \pm\widetilde{\mathbf{z}}_t$ with probability $\frac{1\pm\|\mathbf{z}_t\|}{2}$. In this case the intermediate parameter matrix $\mathbf{W}_t^m$ is set to be equal to the parameter matrix $\mathbf{W}_{t-1}$ and

$$\mathrm{E}(\hat{\mathbf{y}}_t) = \|\mathbf{z}_t\|\,\widetilde{\mathbf{z}}_t = \mathbf{z}_t = \mathbf{W}_{t-1}\mathbf{x}_t = \mathbf{W}_t^m\mathbf{x}_t. \tag{6}$$

In the degenerate case when $\|\mathbf{z}_t\| = 0$, then the direction $\widetilde{\mathbf{z}}_t = \frac{\mathbf{z}_t}{\|\mathbf{z}_t\|}$ of $\mathbf{z}_t$ is undefined and the algorithm arbitrarily sets $\widetilde{\mathbf{z}}_t$ to the unit vector $\mathbf{e}_1 = (1, 0, \ldots, 0)^\top$. Observe that the above equalities (6) remain valid in this case.

When $\|\mathbf{z}_t\| > 1$, then $\mathbf{z}_t$ is too long and the algorithm deterministically sets the prediction $\hat{\mathbf{y}}_t$ to the shortened unit direction $\widetilde{\mathbf{z}}_t = \frac{\mathbf{z}_t}{\|\mathbf{z}_t\|}$. Now the parameter matrix $\mathbf{W}_{t-1}$ also needs to be "shortened" or "projected" as we shall see in a moment. More precisely, $\mathbf{W}_t^m$ is set to equal $\mathbf{W}_{t-1}(\mathbf{I} - (1 - \frac{1}{\|\mathbf{z}_t\|})\mathbf{x}_t\mathbf{x}_t^\top)$ which ensures that

$$\hat{\mathbf{y}}_t = \widetilde{\mathbf{z}}_t = \frac{\mathbf{W}_{t-1}\mathbf{x}_t}{\|\mathbf{z}_t\|} = \mathbf{W}_t^m\mathbf{x}_t.$$

We conclude that in all cases the expected loss of the algorithm is

$$\mathrm{E}\left(\tfrac{1}{2}\left\|\mathbf{y}_t - \hat{\mathbf{y}}_t\right\|^2\right) = 1 - \mathbf{y}_t^\top\mathrm{E}(\hat{\mathbf{y}}_t) = 1 - \mathbf{y}_t^\top\mathbf{W}_t^m\mathbf{x}_t. \tag{7}$$

This means that the update of $\mathbf{W}_t$ from $\mathbf{W}_t^m$ in Step 8 of the algorithm is a standard gradient descent update with respect to the squared Frobenius norm and the above expected loss:

$$\mathbf{W}_t = \operatorname*{argmin}_{\mathbf{W}}\left(\tfrac{1}{2}\left\|\mathbf{W} - \mathbf{W}_t^m\right\|_F^2 + \eta(1 - \mathbf{y}_t^\top\mathbf{W}\mathbf{x}_t)\right).$$

We now prove that the update of $\mathbf{W}_t^m$ from $\mathbf{W}_{t-1}$ is a Bregman projection with respect to the squared Frobenius norm:

Lemma 4 *Let* $\mathbf{V}$ *be a matrix and* $\mathbf{x}$ *be a unit vector. Then the projection of* $\mathbf{V}$ *on the set* $\{\mathbf{W} : \|\mathbf{W}\mathbf{x}\|^2 \leq 1\}$ *is given by*

$$\operatorname*{argmin}_{\mathbf{W}:\|\mathbf{W}\mathbf{x}\|^2\leq 1} \tfrac{1}{2}\|\mathbf{W} - \mathbf{V}\|_F^2 = \begin{cases} \mathbf{V} & \textit{if } \|\mathbf{V}\mathbf{x}\| \leq 1 \\ \mathbf{V}\left(\mathbf{I} - \left(1 - \frac{1}{\|\mathbf{V}\mathbf{x}\|}\right)\mathbf{x}\mathbf{x}^\top\right) & \textit{otherwise.} \end{cases}$$

The projection can be computed in $O(n^2)$ *time.*

[1] In the special case when $\|\mathbf{z}_t\| = 1$, then $\widetilde{\mathbf{z}}_t = \mathbf{z}_t$ and $\hat{\mathbf{y}}_t$ is deterministically chosen as $\widetilde{\mathbf{z}}_t$.

Proof If $\|\mathbf{Vx}\| \leq 1$ then $\mathbf{V}$ is already in the set $\{\mathbf{W} : \|\mathbf{Wx}\|^2 \leq 1\}$ and hence the projection is $\mathbf{V}$ itself. So assume that $\|\mathbf{Vx}\| > 1$. Now note that the set $\{\mathbf{W} : \|\mathbf{Wx}\|^2 \leq 1\}$ is convex. Thus, the projection $\mathbf{W}$ of $\mathbf{V}$ onto this set lies on the boundary and satisfies $\|\mathbf{Wx}\| = 1$. The theory of Lagrange multipliers tells us that the optimum solution satisfies the following equation for some constant α:

$$\mathbf{W} - \mathbf{V} + \alpha \mathbf{W}\mathbf{x}\mathbf{x}^\top = 0. \tag{8}$$

By right multiplying the matrices on both sides of the equation with vector $\mathbf{x}$, moving $-\mathbf{Vx}$ to the right and squaring the length both sides we get

$$\|\mathbf{Wx}\|^2(1+\alpha)^2 = \|\mathbf{Vx}\|^2.$$

Since $\|\mathbf{Wx}\|^2 = 1$, $\alpha = \pm\|\mathbf{Vx}\| - 1$. From Eq. (8) it follows that

$$\mathbf{W} = \mathbf{V}\left(\mathbf{I} + \alpha\mathbf{x}\mathbf{x}^\top\right)^{-1} = \mathbf{V}\left(\mathbf{I} - \tfrac{\alpha}{1+\alpha}\mathbf{x}\mathbf{x}^\top\right) = \mathbf{V}\left(\mathbf{I} - \left(1 \mp \frac{1}{\|\mathbf{Vx}\|}\right)\mathbf{x}\mathbf{x}^\top\right).$$

The second equality follows from the Sherman–Morrison–Woodbury formula (see e.g. Bernstein 2009). Note that the inverse is defined because we are in the case $\|\mathbf{Vx}\| > 1$ and therefore both solutions for α are not equal to -1. The result now follows from the fact that for the $\alpha = +\|\mathbf{Vx}\| - 1$ solution, $\|\mathbf{W} - \mathbf{V}\|_F^2$ is smaller. □

We call our algorithm "Lazy Projection GD", because for the sake of efficiency we "project as little as necessary" while keeping the relationship $\mathrm{E}[\hat{\mathbf{y}}_t] = \mathbf{W}_t^m \mathbf{x}_t$. The algorithm takes $O(n^2)$ time per trial since all steps can be reduced to a small number of matrix additions and matrix vector multiplications. In the corrigendum to the conference version of this paper, an alternate but more time expensive projection is used which projects $\mathbf{W}_{t-1}$ onto the convex hull of all orthogonal matrices. As sketched below, this is more involved because it requires us to maintain the SVD decomposition of the parameter matrix.

Let B denote convex hull of all orthogonal matrices. In "Appendix 2" we show that B is the set of all square matrices with singular values at most one. Projecting onto B consists of computing the SVD of $\mathbf{W}_{t-1}$ and capping all singular values larger than one at one (see corrigendum to Hazan et al. 2010b). Next, the projected matrix $\mathbf{W}_t^m$ is randomly rounded to an orthogonal matrix $\mathbf{U}_t$ s.t. $\mathrm{E}[\mathbf{U}_t] = \mathbf{W}_t^m$ (see Lemma 6 for details), and then the prediction made is $\hat{\mathbf{y}}_t = \mathbf{U}_t\mathbf{x}_t$. Overall, the algorithm takes $O(n^3)$ time per iteration.

As shown in "Appendix 2", the convex hull B of all orthogonal matrices can be written as an intersection of convex constraints:

$$B = \bigcap_{\mathbf{x}:||\mathbf{x}||=1} \{\mathbf{W} : ||\mathbf{Wx}|| \leq 1\}.$$

So for the sake of efficiency we only project onto $\{\mathbf{W} : ||\mathbf{W}\mathbf{x}_t|| \leq 1\}$, where $\mathbf{x}_t$ is the current instance, and not onto the full intersection B. This new "lazy projection" method is a simpler method with update time $O(n^2)$ that avoids the need to maintain the SVD decomposition. The possibility of using delayed projections was discussed in Section 5.5 of Helmbold and Warmuth (2009). What is unique in our setting is that the convex set we project onto depends on the instance $\mathbf{x}_t$.

3.2 Analysis and main theorem

We are now ready to prove our main regret bound for our on-line algorithm based on lazy projections. Note that the learning rate depends on the number of trials T. However, it is easy to run the algorithms in stages based on a geometric series of upper bounds on T [see for example Algorithm G1 of Cesa-Bianchi et al. (1996)]. This increases the regret bound by at most a constant factor.

Theorem 1 *If Algorithm 1 is run with $\eta = \sqrt{\frac{n}{T}}$ on any sequence of T examples, then*

$$\sum_{t=1}^{T} E\left[\frac{1}{2}\left\|\mathbf{y}_t - \hat{\mathbf{y}}_t\right\|^2\right] - \min_{\mathbf{R}\in\mathcal{SO}(n)} \sum_{t=1}^{T} \frac{1}{2}\|\mathbf{y}_t - \mathbf{R}\mathbf{x}_t\|^2 \le \sqrt{nT}.$$

Proof For any rotation matrix $\mathbf{R}$,

$$\begin{aligned}
\frac{1}{2}\|\mathbf{W}_t - \mathbf{R}\|_F^2 &= \frac{1}{2}\left\|\mathbf{W}_t^m - \mathbf{R} + \eta\, \mathbf{y}_t\mathbf{x}_t^\top\right\|_F^2 \\
&= \frac{1}{2}\left\|\mathbf{W}_t^m - \mathbf{R}\right\|_F^2 + \eta \operatorname{tr}\left(\left(\mathbf{W}_t^m - \mathbf{R}\right)\mathbf{x}_t\mathbf{y}_t^\top\right) + \frac{\eta^2}{2}\|\mathbf{y}_t\mathbf{x}_t^\top\|_F^2 \\
&= \frac{1}{2}\left\|\mathbf{W}_t^m - \mathbf{R}\right\|_F^2 + \eta\left(\frac{1}{2}\|\mathbf{y}_t - \mathbf{R}\mathbf{x}_t\|^2 - \mathrm{E}\left[\frac{1}{2}\left\|\mathbf{y}_t - \hat{\mathbf{y}}_t\right\|^2\right]\right) + \frac{\eta^2}{2} \\
&\le \frac{1}{2}\|\mathbf{W}_{t-1} - \mathbf{R}\|_F^2 + \eta\left(\frac{1}{2}\|\mathbf{y}_t - \mathbf{R}\mathbf{x}_t\|^2 - \mathrm{E}\left[\frac{1}{2}\left\|\mathbf{y}_t - \hat{\mathbf{y}}_t\right\|^2\right]\right) + \frac{\eta^2}{2}.
\end{aligned}$$

The first equality uses the update in Step 8 of Algorithm 1. In the second equality we expand the square, and in the third we use $\frac{1}{2}\|\mathbf{y}_t - \mathbf{R}\mathbf{x}_t\|^2 = 1 - \operatorname{tr}(\mathbf{R}\mathbf{x}_t\mathbf{y}_t^\top)$ and $\mathrm{E}\left[\frac{1}{2}\left\|\mathbf{y}_t - \hat{\mathbf{y}}_t\right\|^2\right] = 1 - \operatorname{tr}(\mathbf{W}_t^m\mathbf{x}_t\mathbf{y}_t^\top)$, which is Eq. (7) above. The last inequality is the most interesting. It follows from the generalized Pythagorean theorem for Bregman divergences and the fact the update from $\mathbf{W}_{t-1}$ to $\mathbf{W}_t^m$ is a Bregman projection onto the set $\{\mathbf{W} : \|\mathbf{W}\mathbf{x}_t\|^2 \le 1\} = \{\mathbf{W} : \|\mathbf{W}\mathbf{x}_t\| \le 1\}$. The crucial fact is that this is a closed convex set that contains all rotation matrices. See Herbster and Warmuth (2001) for an extended discussion of the application of Bregman projections for obtaining regret bounds. By rearranging we get the following inequality for each trial:

$$\mathrm{E}\left[\frac{1}{2}\left\|\mathbf{y}_t - \hat{\mathbf{y}}_t\right\|^2\right] - \frac{1}{2}\|\mathbf{y}_t - \mathbf{R}\mathbf{x}_t\|^2 \le \frac{\|\mathbf{W}_{t-1} - \mathbf{R}\|_F^2 - \|\mathbf{W}_t - \mathbf{R}\|_F^2}{2\eta} + \frac{\eta}{2}.$$

By summing over all T trials we get

$$\begin{aligned}
\sum_t \mathrm{E}\left[\frac{1}{2}\left\|\mathbf{y}_t - \hat{\mathbf{y}}_t\right\|^2\right] - \sum_t \frac{1}{2}\|\mathbf{y}_t - \mathbf{R}\mathbf{x}_t\|^2 &\le \frac{\|\mathbf{W}_0 - \mathbf{R}\|_F^2 - \|\mathbf{W}_T - \mathbf{R}\|_F^2}{2\eta} + \frac{\eta T}{2} \\
&\le \frac{n}{2\eta} + \frac{\eta T}{2},
\end{aligned}$$

since $\|\mathbf{W}_0 - \mathbf{R}\|_F^2 = \|\mathbf{R}\|_F^2 = \operatorname{tr}\left(\mathbf{R}\mathbf{R}^\top\right) = \operatorname{tr}(\mathbf{I}) = n$ and $\|\mathbf{W}_T - \mathbf{R}\|_F^2 \ge 0$. The RHS is minimized at $\sqrt{nT}$ by choosing $\eta = \sqrt{\frac{n}{T}}$. □

As we shall see the above upper bound of $\sqrt{nT}$ on the regret bound of our randomized algorithm is rather weak in the noise-free case, i.e. when there is a rotation that has loss zero on the entire sequence. It is an open problem whether the upper bound on the regret can be strengthened to $O(\sqrt{nL}+n)$ where L is the loss of the best rotation on the entire sequence of trials. An $O(\sqrt{nL})$ regret bound was erroneously claimed in the conference paper of Hazan et al. (2010b).

3.3 Lower bounds on the regret

We now show a lower bound against any algorithm (including randomized ones).

Theorem 2 *For any integer $T \geq n \geq 2$, there is a fixed instance vector sequence on which any randomized online algorithm for learning rotations can be forced to have regret at least* $\sqrt{\frac{(T-1)(n-1)}{4}} = \Omega(\sqrt{nT})$.

Proof We first define the fixed instance sequence. Let $\mathbf{e}_i$ denote the i-th standard basis vector, i.e. the vector with 1 in its i-th coordinate and 0 everywhere else. In trial $t < T$, set $\mathbf{x}_t = \mathbf{e}_{f(t)}$, where $f(t) = (t \bmod n-1)+1$ (i.e. we cycle through the coordinates $1, 2, \ldots, n-1$). The last instance is $\mathbf{e}_n$.

We will now show that for any online algorithm for the rotations problem, there is a sequence of result vectors $\mathbf{y}_t$ of length T for which this algorithm has regret at least $\sqrt{\frac{(T-1)(n-1)}{4}} = \Omega(\sqrt{nT})$. Recall that $\mathbf{e}_{f(t)}$ is the instance at trial $1 \leq t < T$. To achieve our lower bound on the regret we set $\mathbf{y}_t$ equal to the instance at trial t or its reverse with equal probability, i.e. $\mathbf{y}_t = \sigma_t \mathbf{e}_{f(t)}$, where $\sigma_t \in \{-1, 1\}$ uniformly at random. For any coordinate $i \in 1, 2, \ldots, n-1$, let $X_i = \sum_{t:\ f(t)=i} \sigma_t$. For the final trial T, the instance $\mathbf{x}_T$ is $\mathbf{e}_n$, and we set the result vector to $\mathbf{y}_T = \sigma_T \mathbf{e}_n$, where $\sigma_T \in \{-1, 1\}$ is chosen in a certain way specified momentarily. First, note that

$$\mathbf{S}_T = \sum_{t=1}^{T} \mathbf{y}_t \mathbf{x}_t^\top = \mathrm{diag}(X_1, X_2, \ldots, X_{n-1}, \sigma_T).$$

We choose σ_T so that

$$\det(\mathbf{S}_T) = \sigma_T \prod_{i=1}^{n-1} X_i > 0.$$

In other words, $\sigma_T = \mathrm{sgn}(\prod_{i=1}^{n-1} X_i)$.

By Lemma 2, the solution to the offline problem is the rotation matrix $\mathbf{R}^\star = \mathrm{argmax}_{\mathbf{R}\in\mathcal{SO}(n)} \mathrm{tr}(\mathbf{S}_T\mathbf{R})$, where

$$\mathbf{R}^\star = \mathrm{diag}(\mathrm{sgn}(X_1), \mathrm{sgn}(X_2), \ldots, \mathrm{sgn}(X_{n-1}), \sigma_T),$$

and the loss of this matrix is

$$\sum_{t=1}^{T} \tfrac{1}{2} \left\|\mathbf{y}_t - \mathbf{R}^\star\mathbf{x}_t\right\|^2 = T - \mathrm{tr}(\mathbf{S}_T R^\star) = T - \sum_{i=1}^{n-1} |X_i| - 1.$$

Since each X_i is a sum of at least $\lfloor \frac{T-1}{n-1} \rfloor$ Rademacher variables, Khintchine's inequality (Haagerup 1982) implies that $\mathrm{E}_{\sigma_t}[|X_i|] \geq \sqrt{\frac{1}{2}\lfloor \frac{T-1}{n-1} \rfloor}$ where the expectation is taken over the choice of the σ_t's. Thus, the expected loss of the optimal rotation is bounded as follows:

$$\begin{aligned}\mathrm{E}_{\sigma_t}\left[\sum_{t=1}^{T} \tfrac{1}{2} \left\|\mathbf{y}_t - \mathbf{R}^\star\mathbf{x}_t\right\|^2\right] &= T - \mathrm{E}_{\sigma_t}[\mathrm{tr}(\mathbf{S}_T\mathbf{R}^\star)] \\ &\leq T - 1 - (n-1)\sqrt{\frac{1}{2}\left\lfloor \frac{T-1}{n-1} \right\rfloor}. \end{aligned} \quad (9)$$

We now show that for $t < T$ the expected loss of the algorithm is one, where the expectation is with respect to σ_t as well as the internal randomization of the algorithm. Note that both types of randomizations are independent. If $\mathrm{E}_{\text{alg}}[\hat{\mathbf{y}}_t]$ denotes the expected prediction vector of the algorithm at trial t after receiving instance vector $\mathbf{e}_{f(t)}$ and before receiving the result vector $\mathbf{y}_t = \sigma_t \mathbf{e}_t$, then

$$\mathrm{E}_{\sigma_t,\text{alg}}\left[\tfrac{1}{2}\left\|\mathbf{y}_t - \hat{\mathbf{y}}_t\right\|^2\right] = 1 - \mathrm{E}_{\sigma_t,\text{alg}}\left[\mathbf{y}_t^\top \hat{\mathbf{y}}_t\right] = 1 - \mathrm{E}_{\sigma_t}[\sigma_t]\,\mathbf{e}_{f(t)}^\top \mathrm{E}_{\text{alg}}\left[\hat{\mathbf{y}}_t\right] = 1.$$

In trial T, the algorithm might at best have an expected loss of 0. Thus, the expected loss of the algorithm is at least $T-1$, and hence by subtracting (9), its expected regret (with respect to both randomizations) is at least $(n-1)\sqrt{\frac{1}{2}\left\lfloor\frac{T-1}{n-1}\right\rfloor}$. Since for $a \geq 1$, $\lfloor a \rfloor \geq \max(a-1, 1) \geq (a-1+1)/2 = a/2$, the expected regret is lower bounded by

$$(n-1)\sqrt{\frac{1}{4}\left(\frac{T-1}{n-1}\right)} = \sqrt{\frac{(T-1)(n-1)}{4}}.$$

This implies that for any algorithm there is a choice of the σ_t's for which the algorithm has expected regret (with respect to its internal randomization only) at least $\sqrt{\frac{(T-1)(n-1)}{4}}$, as required. Since $T \geq n \geq 2$, this lower bounds is $\Omega(\sqrt{nT})$. □

As can be seen from the proof, the above lower bound is essentially $\sqrt{\frac{nT}{2}}$ for large n and T, s.t. $T \geq n$. Recall that the upper bound of our Algorithm 1 is $\sqrt{nT}$.

We now generalize the above lower bound on the worst-case regret that depends on the number of trials T to a lower bound that depends on the loss L of the best rotation in hindsight. Applying Theorem 6 of Sect. 5 below with n orthogonal instances shows that any randomized on-line algorithm can be forced to incur loss n on sequences of examples for which there is a consistent rotation (i.e. $L = 0$). Combined with the above lower bound, this immediately generalizes to the following lower bound for any $L \geq 0$:

Corollary 1 *For any $L \geq 0$ and $T \geq \lfloor L/2 \rfloor$, any randomized on-line algorithm can be forced to have regret $\Omega(\sqrt{nL} + n)$ for some sequence of T examples for which the loss of the best rotation is at most L.*

Proof If $L < 2n$, then the lower bound of n for the noise free case already implies a lower bound of $\Omega(\sqrt{nL} + n)$. If $L \geq 2n$, then we apply the lower bound of the above theorem for the first $T_0 = \lfloor L/2 \rfloor$ rounds. Note that $T \geq T_0 \geq n$. We then achieve a regret lower bound of $\Omega(\sqrt{\lfloor L/2 \rfloor n}) = \Omega(\sqrt{nL} + n)$ for the first T_0 rounds. Let $\mathbf{R}_0$ be the best rotation for the first T_0 rounds. Since the per trial loss of any rotation is at most 2, the loss of the best rotation on the sequence of the T_0 examples used in the construction of the above theorem is at most $2T_0 \leq L$.

For the remaining $T - T_0$ rounds, we simply use arbitrary pairs of unit vectors $(\mathbf{x}_t, \mathbf{y}_t)$ that are consistent with $\mathbf{R}_0$ (i.e. $\mathbf{y}_t = \mathbf{R}_0 \mathbf{x}_t$ for $T_0 < t \leq T$). Thus, the rotation $\mathbf{R}_0$ incurs 0 additional loss, and hence its total loss is bounded by L; the algorithm, on the other hand, also incurs 0 additional loss at best. Thus we have an $\Omega(\sqrt{nL} + n)$ lower bound on the regret for the sequence of T examples in this construction, and the best rotation has loss at most L. □

Note that the constant factors needed for the $\Omega(.)$ notation in the proofs of the lower bounds are independent of the algorithm and the values of n and L.

4 Deterministic online algorithms

We begin by showing that any deterministic algorithm can be forced to have regret T in T trials. We then show how to construct a deterministic algorithm from any probabilistic one with at most twice the loss.

Theorem 3 *For any deterministic algorithm, there is a sequence of T examples (which may be fixed before running the algorithm) such that the algorithm has regret at least T on it.*

Proof The adversary sets all instances $\mathbf{x}_t$ to $\mathbf{e}_1$. Since the algorithm is deterministic, the adversary can compute the prediction $\hat{\mathbf{y}}_t$, and then set the result vector $\mathbf{y}_t = -\hat{\mathbf{y}}_t$. The per trial loss is therefore

$$\tfrac{1}{2}\left\|\mathbf{y}_t - \hat{\mathbf{y}}_t\right\|^2 = \tfrac{1}{2}\|2\hat{\mathbf{y}}_t\|^2 = 2,$$

amounting to a loss $2T$ in all trials.

The loss of the optimum rotation on the produced sequence of examples is

$$\min_{\mathbf{R}\in\mathcal{SO}(n)} \tfrac{1}{2}\|\mathbf{y}_t - \mathbf{R}\mathbf{x}_t\|^2 = T - \max_{\mathbf{R}\in\mathcal{SO}(n)} \operatorname{tr}(\mathbf{S}_T\mathbf{R}),$$

where $\mathbf{S}_T = \sum_{t=1}^{T} \mathbf{e}_1\mathbf{y}_t^\top$. By Lemma 2, $\max_{\mathbf{R}\in\mathcal{SO}(n)} \operatorname{tr}(\mathbf{S}_T\mathbf{R}) \geq 0$. Thus, the loss of the optimum rotation is at most T. Hence, the algorithm has regret at least $2T - T = T$. □

A simple deterministic algorithm would be to predict with an optimal rotation matrix for the data seen so far using the algorithm of Lemma 2. However this Follow the Leader or Incremental Off-Line type algorithm along with the elegant deterministic algorithms based on Lie group mathematics (Arora 2009) all can be forced to have linear worst-case regret, whereas probabilistic algorithms can achieve worst-case regret at most $\sqrt{nT}$. We don't know the optimal worst-case regret achievable by deterministic algorithms. In some cases the worst-case regret of deterministic algorithm is at least F times the regret of the best probabilistic algorithm, where F grows with the size of the problem (see e.g. Warmuth et al. 2011). We show that for our rotation problem, the factor is at most 2.

Theorem 4 *Any randomized algorithm for learning rotations can be converted into a deterministic algorithm with at most twice the loss.*

Proof We construct a deterministic algorithm from the randomized one that in each trial has at most twice the loss. Let $\mathbf{z}_t$ be the expected prediction $\mathrm{E}[\hat{\mathbf{y}}_t]$ of the randomized algorithm. If $\mathbf{z}_t = \mathbf{0}$, we let the deterministic algorithm predict with $\mathbf{e}_1$. In this case the randomized algorithm has expected loss $1 - \mathbf{y}_t^\top\mathbf{z}_t = 1$ and the deterministic algorithm has loss at most 2.

If $\mathbf{z}_t \neq \mathbf{0}$, then the deterministic algorithm predicts with $\hat{\mathbf{y}}_t = \frac{\mathbf{z}_t}{\|\mathbf{z}_t\|}$. We need to show that

$$\begin{aligned} 1 - \mathbf{y}_t^\top\hat{\mathbf{y}}_t &\leq 2(1 - \mathbf{y}_t^\top\mathbf{z}_t) \\ \Leftrightarrow 0 &\leq 1 - 2\mathbf{y}_t^\top\mathbf{z}_t + \mathbf{y}_t^\top\frac{\mathbf{z}_t}{\|\mathbf{z}_t\|}. \end{aligned} \tag{10}$$

Next observe that $\|\mathbf{z}_t\|$ lies in the range $[|\mathbf{y}_t^\top\mathbf{z}_t|\,, 1]$: the upper bound holds because $\mathbf{z}_t$ lies in the unit ball and the lower bound follows from the fact that $|\mathbf{y}_t^\top\mathbf{z}_t|$ is the length of the projection of $\mathbf{z}_t$ onto unit vector $\mathbf{y}_t$. Now if $\mathbf{y}_t^\top\mathbf{z}_t \geq 0$, then we use the upper bound on the range to show that $1-2\mathbf{y}_t^\top\mathbf{z}_t+\mathbf{y}_t^\top\frac{\mathbf{z}_t}{\|\mathbf{z}_t\|} \geq 1-2\mathbf{y}_t^\top\mathbf{z}_t+\mathbf{y}_t^\top\mathbf{z}_t = 1-\mathbf{y}_t^\top\mathbf{z}_t \geq 0$. If $\mathbf{y}_t^\top\mathbf{z}_t \leq 0$, then we use the lower bound on the range to show that $1 - 2\mathbf{y}_t^\top\mathbf{z}_t + \mathbf{y}_t^\top\frac{\mathbf{z}_t}{\|\mathbf{z}_t\|} \geq 1 - 2\mathbf{y}_t^\top\mathbf{z}_t - 1 \geq 0$. □

5 Learning when there is a consistent rotation

In Sect. 3 we gave a randomized algorithm with at most $\sqrt{nT}$ worst-case regret (when $T \geq n$) and showed that this regret bound is tight within a constant factor. However regret bounds that are expressed as a function of number of trials T only guard against the high-noise case and are weak when there is a comparator with small loss. Ideally we want bounds that grow with the loss of the best comparator chosen in hindsight as was done in the original online learning papers (see e.g. Cesa-Bianchi et al. 1996a; Kivinen and Warmuth 1997). In an attempt to understand what regret bounds are possible when there is a comparator of low noise, we carefully analyze the case when there is a rotation $\mathbf{R}$ consistent with all T examples, i.e. $\mathbf{R}\mathbf{x}_t = \mathbf{y}_t$ for all $t = 1, 2, \ldots, T$. Even in this case, the online learner incurs loss until a unique consistent rotation is determined and the question is which algorithm has the smallest regret against such sequences of examples.

There is a randomized and a deterministic variant of the algorithm, but they only disagree in the first trial. The deterministic variant predicts with $\hat{\mathbf{y}}_1 = \mathbf{e}_1$ and the randomized variant with $\hat{\mathbf{y}}_1 = \pm\mathbf{e}_1$ with probability $\frac{1}{2}$ [i.e. $\mathrm{E}(\hat{\mathbf{y}}_1) = \mathbf{0}$)]. In later trials the algorithm (both variants) predict deterministically. At trial $t \geq 2$, it first decomposes the instance $\mathbf{x}_t$ in the part $\mathbf{x}_t^{\|}$ that lies in the span of the previous instances and $\mathbf{x}_t^{\perp}$ that is perpendicular to the span. By definition, $\mathbf{x}_t^{\|}$ is a linear combination of the past instances $\mathbf{x}_q$ for $1, \ldots, t-1$. Replacing the instance vectors $\mathbf{x}_q$ in the linear combination by the result vectors $\mathbf{y}_q$, we arrive at $\mathbf{y}_t^{\|}$ (See Step 5). The algorithm essentially predicts with the direction vector of $\mathbf{y}_t^{\|}$ until and including the first trial S in which the matrix $[\mathbf{x}_1, \ldots, \mathbf{x}_S]$ has rank $n-1$. From trial $S+1$ to T, the algorithm predicts with the unique consistent rotation.

Algorithm 2 Optimal noise-free algorithm

```
1: In trail 1, predict with $\hat{\mathbf{y}}_1 = \begin{cases} \mathbf{e}_1 & \text{if deterministic} \\ \pm\mathbf{e}_1 \text{ with probability } \frac{1}{2} & \text{if randomized} \end{cases}$
2: for trial $t = 2$ to and including the first trial $S$ in which $\mathrm{rank}([\mathbf{x}_1, \ldots, \mathbf{x}_S]) = n-1$
   or $S = T$ if this never happens do
3:   Decompose the instance vector $\mathbf{x}_t = \mathbf{x}_t^{\|} + \mathbf{x}_t^{\perp}$ and compute the linear combination $\mathbf{x}_t^{\|} = \sum_{q=1}^{t-1} \alpha_q \mathbf{x}_q$
4:   if $\mathbf{x}^{\|} = \mathbf{0}$ then predict with $\hat{\mathbf{y}}_t = \mathbf{y}_1$.
5:   else predict with $\hat{\mathbf{y}}_t = \frac{\mathbf{y}_t^{\|}}{\|\mathbf{y}_t^{\|}\|}$, where $\mathbf{y}_t^{\|} = \sum_{q=1}^{t-1} \alpha_q \mathbf{y}_q$.
6: end for
7: for $t = S+1$ to $T$ do
8:   Predict with $\hat{\mathbf{y}}_t = \mathbf{R}\mathbf{x}_t$, where $\mathbf{R}$ is the rotation consistent with trials 1 through $S$
9: end for
```

Theorem 5 *On any sequence of examples $(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_T, \mathbf{y}_T)$ that is consistent with a rotation, the randomized version of Algorithm 2 has expected loss $\sum_{t=1}^{S}(1 - \|\mathbf{x}_t^{\|}\|)$. The deterministic version has loss at most $\left(\sum_{t=1}^{S}\left(1 - \|\mathbf{x}_t^{\|}\|\right)\right) + 1$.*

Proof For proving the upper bound for the randomized algorithms, first note that in trial 1 the expected loss is $1 - \mathbf{y}_1^{\top}\mathrm{E}(\hat{\mathbf{y}}_1) = 1 - \mathbf{y}_1^{\top}\mathbf{0} = 1 = 1 - \|\mathbf{x}_1^{\|}\|$. In trials $2 \leq t \leq S$, the deterministic choice $\hat{\mathbf{y}}_t = \frac{\mathbf{y}_t^{\|}}{\|\mathbf{y}_t^{\|}\|}$ assures that the loss is

$$\frac{1}{2}\left\|\mathbf{y}_t - \hat{\mathbf{y}}_t\right\|^2 = 1 - \mathbf{y}_t^{\top}\hat{\mathbf{y}}_t = 1 - \|\mathbf{y}_t^{\|}\| = 1 - \|\mathbf{x}_t^{\|}\|. \tag{11}$$

In the special case when $\mathbf{x}_t^{\|} = \mathbf{0}$ then $\mathbf{x}_t^\top \mathbf{x}_1 = 0$, and so $\mathbf{y}_t^\top \mathbf{y}_1 = 0$ since we have a consistent rotation and rotations preserve angles. Thus the deterministic prediction $\hat{\mathbf{y}}_t = \mathbf{y}_1$ has loss $1 = 1 - \|\mathbf{x}_t^{\|}\|$ as well. If $S < T$, then since $\operatorname{rank}([\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_S]) = n - 1$, the vectors $\mathbf{x}_1^\perp, \mathbf{x}_2^\perp, \ldots, \mathbf{x}_S^\perp$ are mutually orthogonal, with exactly $n-1$ non-zero vectors. This gives two sequences of mutually orthogonal vectors $\mathbf{x}_1^\perp, \mathbf{x}_2^\perp, \ldots, \mathbf{x}_S^\perp$ and $\mathbf{y}_1^\perp, \mathbf{y}_2^\perp, \ldots, \mathbf{y}_S^\perp$ such that $\|\mathbf{x}_t^\perp\| = \|\mathbf{y}_t^\perp\|$ for $t \le S$, and exactly $n-1$ non-zero vectors in each sequence. By Lemma 5 below we conclude that there is a unique rotation matrix $\mathbf{R}$ such that $\mathbf{y}_t^\perp = \mathbf{R}\mathbf{x}_t^\perp$ for $t \le S$. Since the result vectors $\mathbf{y}_t$ are linear combinations of the orthogonal components $\mathbf{y}_q^\perp$ for $q \le t$, we have the rotation $\mathbf{R}$ is a unique rotation consistent with the first S examples. Hence we find this rotation and incur no more loss in any trial $t > S$. Overall, the expected loss of the randomized algorithm is

$$\sum_{t=1}^{S} (1 - \|\mathbf{x}_t^{\|}\|).$$

The deterministic algorithm may have loss at most 2 in the first trial. For the rest of the trials the deterministic algorithm is the same as the randomized one and has the same bound on the loss, leading to an upper bound of $\left(\sum_{t=1}^{S}\left(1 - \|\mathbf{x}_t^{\|}\|\right)\right) + 1$. □

We now give a simple lemma which states that a rotation is essentially determined by its action on $n-1$ mutually orthogonal vectors. The proof is straightforward, but we include it for the sake of completeness.

Lemma 5 *Let $\mathbf{x}_1^\perp, \mathbf{x}_2^\perp, \ldots, \mathbf{x}_T^\perp$ and $\mathbf{y}_1^\perp, \mathbf{y}_2^\perp, \ldots, \mathbf{y}_T^\perp$ be two orthogonal sequences of vectors in $\mathbb{R}^n$ such that $\|\mathbf{x}_t\| = \|\mathbf{y}_t\|$ and both sequences have at most $r \le n-1$ non-zero vectors. Then there is a rotation matrix $\mathbf{R}$ such that $\mathbf{y}_t = \mathbf{R}\mathbf{x}_t$. If $r = n-1$, then $\mathbf{R}$ is unique.*

Proof Pairs $(\mathbf{x}_t, \mathbf{y}_t)$ of length zero can be omitted and without loss of generality we can rescale the remaining vectors so that their length is one, since rotations are a linear transformation and preserve the scaling. Also we can always add more orthonormal vectors to both sequences and therefore it suffices to find a rotation when $r = n - 1 = T$. Let $\mathbf{x}_n^\perp$ and $\mathbf{y}_n^\perp$ be unit vectors orthogonal to the span of $\{\mathbf{x}_1^\perp, \mathbf{x}_2^\perp, \ldots, \mathbf{x}_{n-1}^\perp\}$ and $\{\mathbf{y}_1^\perp, \mathbf{y}_2^\perp, \ldots, \mathbf{y}_{n-1}^\perp\}$, respectively. Now, since rotations preserve angles and lengths, if there is a rotation $\mathbf{R}$ such that $\mathbf{R}\mathbf{x}_t^\perp = \mathbf{y}_t^\perp$ for $t = 1, 2, \ldots, n-1$, then we must have $\mathbf{R}\mathbf{x}_n^\perp = \pm\mathbf{y}_n^\perp$. We now determine the right sign of $\mathbf{y}_n^\perp$ as follows. Let $\mathbf{X} = [\mathbf{x}_1^\perp, \mathbf{x}_2^\perp, \ldots, \mathbf{x}_{n-1}^\perp, \mathbf{x}_n^\perp]$, $\mathbf{Y}^+ = [\mathbf{y}_1^\perp, \mathbf{y}_2^\perp, \ldots, \mathbf{y}_{n-1}^\perp, \mathbf{y}_n^\perp]$, and $\mathbf{Y}^- = [\mathbf{y}_1^\perp, \mathbf{y}_2^\perp, \ldots, \mathbf{y}_{n-1}^\perp, -\mathbf{y}_n^\perp]$. Note that $\mathbf{X}$, $\mathbf{Y}^+$ and $\mathbf{Y}^-$ are all orthogonal matrices. The desired rotation matrix $\mathbf{R}$ is a solution to one of following two linear systems $\mathbf{R}\mathbf{X} = \mathbf{Y}^+$ or $\mathbf{R}\mathbf{X} = \mathbf{Y}^-$, i.e. $\mathbf{R} = \mathbf{Y}^+\mathbf{X}^\top$ or $\mathbf{R} = \mathbf{Y}^-\mathbf{X}^\top$, since $\mathbf{X}^{-1} = \mathbf{X}^\top$. Note that both $\mathbf{Y}^+\mathbf{X}^\top$ and $\mathbf{Y}^-\mathbf{X}^\top$ are orthogonal matrices since they are products of orthogonal matrices, and the determinant of one is the negative of the other. The desired rotation matrix is then the solution with determinant 1. □

We now prove a lower bound which shows that Algorithm 2 given above has the strong property of being *instance-optimal*: viz., for any sequence of input vectors and for any algorithm, there is a sequence of result vectors on which the loss achieved of the algorithm is at least that of Algorithm 2:

Theorem 6 *For any online algorithm and for every instance sequence $\mathbf{x}_1, \ldots, \mathbf{x}_T$, there is an adversary strategy for choosing the result sequence $\mathbf{y}_1, \ldots, \mathbf{y}_T$ for which there is a consistent rotation, and the algorithm incurs loss $\sum_{t=1}^{S}(1 - \|\mathbf{x}_t^{\|}\|)$. Furthermore, if the algorithm is deterministic, then the adversary can force a loss of at least $\sum_{t=1}^{S}(1 - \|\mathbf{x}_t^{\|}\|) + 1$.*

Proof Let us begin with the lower bound for randomized algorithms. At trials $t \leq S$, the adversary finds some vector $\mathbf{y}_t^\perp$ of length $\|\mathbf{x}_t^\perp\| = \sqrt{1 - \|\mathbf{x}_t^\parallel\|^2}$ that is perpendicular to the span of $\{\mathbf{y}_1, \ldots, \mathbf{y}_{t-1}\}$ and chooses the result vector $\mathbf{y}_t = \mathbf{y}_t^\parallel \pm \mathbf{y}_t^\perp$ depending on which of the two examples $(\mathbf{x}_t, \mathbf{y}_t^\parallel \pm \mathbf{y}_t^\perp)$ has larger expected loss. The expected loss of both examples is

$$\begin{aligned} \mathrm{E}\left[\tfrac{1}{2}\|\mathbf{y}_t^\parallel + \mathbf{y}_t^\perp - \hat{\mathbf{y}}_t\|\right] + \mathrm{E}\left[\tfrac{1}{2}\|\mathbf{y}_t^\parallel - \mathbf{y}_t^\perp - \hat{\mathbf{y}}_t\|\right] &= 2 - \left(\mathbf{y}_t^\parallel + \mathbf{y}_t^\perp + \mathbf{y}_t^\parallel - \mathbf{y}_t^\perp\right)^\top \mathrm{E}\left[\hat{\mathbf{y}}_t\right] \\ &= 2 - 2\left(\mathbf{y}_t^\parallel\right)^\top \mathrm{E}\left[\hat{\mathbf{y}}_t\right] \\ &\geq 2 - 2\|\mathbf{y}_t^\parallel\| = 2 - 2\|\mathbf{x}_t^\parallel\|, \end{aligned}$$

where last inequality holds because $\|\mathrm{E}[\hat{\mathbf{y}}_t]\| \leq \mathrm{E}[\|\hat{\mathbf{y}}_t\|] \leq 1$, and therefore $(\mathbf{y}_t^\parallel)^\top \mathrm{E}[\hat{\mathbf{y}}_t] \leq \|\mathbf{y}_t^\parallel\|$. It follows that one of the result vectors $\mathbf{y}_t = \mathbf{y}_t^\parallel \pm \mathbf{y}_t^\perp$ incurs expected loss at least $1 - \|\mathbf{x}_t^\parallel\|$ and the adversary makes that choice. Overall, the loss for any algorithm is at least

$$\sum_{t=1}^{S} \left(1 - \|\mathbf{x}_t^\parallel\|\right).$$

For deterministic algorithms, the adversary can further force a loss of 2 on the first trial by choosing $\mathbf{y}_1 = -\hat{\mathbf{y}}_1$, leading to a lower bound of $1 + \sum_{t=1}^{S}(1 - \|\mathbf{x}_t^\parallel\|)$.

We next show that for the examples $(\mathbf{x}_t, \mathbf{y}_t)$ produced by the adversary, there always is a consistent rotation $\mathbf{R}$ such that $\mathbf{y}_t = \mathbf{R}\mathbf{x}_t$. Note that in the construction the vectors $\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_t$ lie in the span of the $\mathbf{y}_1^\perp, \mathbf{y}_2^\perp, \ldots, \mathbf{y}_t^\perp$. So it suffices to show that there is a rotation matrix $\mathbf{R}$ such that $\mathbf{y}_q^\perp = \mathbf{R}\mathbf{x}_q^\perp$ for all $q \leq t$. To show this we use Lemma 5. First, we note that since rank of $[\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_t]$ is at most $n-1$, and since $\mathbf{x}_1^\perp, \mathbf{x}_2^\perp, \ldots, \mathbf{x}_t^\perp$ are mutually orthogonal and span the same subspace, there can be at most $n-1$ non-zero $\mathbf{x}_q^\perp$ vectors. Further, by construction $\|\mathbf{y}_q^\perp\| = \|\mathbf{x}_q^\perp\|$ for all $q \leq t$. Thus, by Lemma 5, the desired rotation exists and we are done. □

The following algorithm seems more natural than Algorithm 2 above: Given $\mathbf{x}_t$, decompose it into $\mathbf{x}_t^\parallel$ and $\mathbf{x}_t^\perp$ as before. Then predict with a rotation that is consistent with the previous $t-1$ examples (i.e. takes $\mathbf{x}_t^\parallel$ to $\mathbf{y}_t^\parallel$) and rotates $\mathbf{x}_t^\perp$ to an arbitrary vector $\pm\mathbf{y}_t^\perp$ of the same length that is orthogonal to the previous result vectors and whose sign is chosen uniformly. Thus $\hat{\mathbf{y}}_t = \mathbf{y}_t^\parallel + \pm\mathbf{y}_t^\perp$ and the expected loss of this algorithm is

$$1 - y_t^\top \mathrm{E}\left[\hat{\mathbf{y}}_t\right] = 1 - \left(\mathbf{y}_t^\parallel + \mathbf{y}_t^\perp\right)^\top \left(\mathbf{y}_t^\parallel \pm \mathbf{y}_t^\perp\right) = 1 - ||\mathbf{y}_t^\parallel||^2 = 1 - ||\mathbf{x}_t^\parallel||^2.$$

However the above lower bound shows that when $\mathbf{y}_t^\parallel \neq \mathbf{0}$, then only the deterministic prediction $\hat{\mathbf{y}}_t = \frac{\mathbf{y}_t^\parallel}{\|\mathbf{y}_t^\parallel\|}$ has the optimal loss $1 - \|\mathbf{x}_t\|$ [i.e. it makes the equalities (11) tight]. This is also in contrast to Algorithm 1, which often employs non-optimal random predictions in trials when $\mathbf{y}_t^\parallel \neq \mathbf{0}$. It seems that less randomization should be used when the loss of the best offline rotation is small because a deterministic algorithm is essentially minimax optimal in the noise-free case.

The deterministic variant of Algorithm 2 also does not predict with a consistent rotation. It is easy to show that any deterministic algorithm that predicts with a consistent rotation can be forced to have loss $2(n-1)$ instead of the optimum n: when such an algorithm receives an instance vector that is orthogonal to all the previous instance vectors, the algorithm deterministically predicts with a unit vector $\hat{\mathbf{y}}_t$ that is orthogonal to all the previous result vectors. Since the adversary knows $\hat{\mathbf{y}}_t$, it can simply choose the result vector $\mathbf{y}_t$ as $-\hat{\mathbf{y}}_t$. This forces the algorithm to incur loss 2 and the can be repeated $n-1$ times, at which point the optimal rotation is completely determined.

6 Conclusions

We have presented a randomized online algorithm for learning rotation with a regret bound of $\sqrt{nT}$ and proved a lower bound of $\Omega(\sqrt{nT})$ for $T > n$. We also proved a lower bound of $\Omega(\sqrt{nL}+n)$ for the regret of any algorithm on sequences on which the best rotation has lost at most L. Recently, an upper bound was shown that matches this lower bound within a constant factor (Nie 2015). However the algorithm that achieves this upper bound on the regret is not the GD algorithm of this paper based on lazy projection but the GD algorithm that projects into the convex hull of orthogonal matrices at the end of each trial (Hazan et al. 2010a). The latter algorithm requires $O(n^3)$ per trial (Hazan et al. 2010a). It is open whether the $O(n^2)$ lazy projection version of GD has the same $\Omega(\sqrt{nL}+n)$ regret bound.

In general, we don't know the best way to maintain uncertainty over rotation matrices. The convex hull of orthogonal matrices seems to be a good candidate (as used in Hazan et al. 2010a). For the sake of efficiency, we used an even larger hypothesis space in this paper: we only require in Algorithm 1 that $\|\mathbf{W}_t^m\mathbf{x}_t\| \leq 1$ at trial t. The algorithm regularizes with the squared Frobenius norm and uses Bregman projections with respect to the inequality constraint $\|\mathbf{W}_t^m\mathbf{x}_t\| \leq 1$. However such projections "forget" information about past examples and the squared Frobenius norm does not take the group structure of $\mathcal{SO}(n)$ into account.

One way to resolve some of these questions is to develop the minimax optimal algorithm for rotations and the linear loss discussed in this paper. This was recently posted as an open problem in Kotłowski and Warmuth (2011). In the special case of $n=2$, the minimax regret for randomized algorithm and T trials was determined to be $\sqrt{T}$, whereas for our randomized algorithm we were only able to prove the larger regret bound of $\sqrt{2T}$. We have shown in the previous section that in the noise-free case, the minimax regret is $n-1$ for randomized algorithms and $2(n-1)$ for deterministic ones. Curiously enough the minimax algorithm for $n=2$ and T trials (Kotłowski and Warmuth 2011) makes heavy use of randomness, whereas in the noise-free case the optimal randomized algorithm only requires randomness in trials when $\mathbf{x}_t^{\|}=\mathbf{0}$.

Another direction is to make the problem harder and study learning rotations when the loss is non-linear. The question is whether for non-linear loss functions the GD algorithm that projects on the parameter space of the convex hull of orthogonal matrices still achieves a worst-case regret that is within a constant factor of optimal.

Acknowledgments This work was motivated by preliminary research done with Adam Smith (Smith and Warmuth 2010, 2008) and we greatly benefited from discussions with him. We thank Abhishek Kumar for allowing us to include a simplified proof of Wahba's problem which was worked out in collaboration with him. Our work greatly benefited from discussions with Raman Arora and Wojciech Kotłowski. In particularly, Wojciech showed us the $O(n^2)$ algorithm for rotating a unit vector onto another. Finally, thanks to Christfried Webers for pointing out a number of subtle inaccuracies in an earlier draft and to Nie Jiazhong for many helpful discussions.

Appendix 1: Solutions of Wahba's problem and the orthogonal Procrustes problem

We first prove Lemma 3, since it is simpler and it gives a reduction which will be useful for the proof of Lemma 2.

Proof of Lemma 3 Recall that we want to compute

$$\max_{\mathbf{R}\in\mathcal{O}(n)} \operatorname{tr}(\mathbf{SR}).$$

Let $\mathbf{S} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\top$ be an SVD of $\mathbf{S}$, so that $\mathbf{U}$ and $\mathbf{V}$ are orthogonal matrices, and $\boldsymbol{\Sigma} = \operatorname{diag}(\sigma_1, \sigma_2, \ldots, \sigma_n)$ is a diagonal matrix of the non-negative singular values σ_i.

Now, we do a change of variables. Instead of maximizing over an orthogonal matrix $\mathbf{R}$, we maximize over orthogonal matrix $\mathbf{W} = \mathbf{V}^\top\mathbf{RU}$. This lets us rewrite the dot product we are minimizing over

$$\operatorname{tr}(\mathbf{SR}) = \operatorname{tr}\left(\mathbf{SVWU}^\top\right) = \operatorname{tr}\left(\mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\top\mathbf{VWU}^\top\right) = \operatorname{tr}(\boldsymbol{\Sigma}\mathbf{W}). \tag{12}$$

Since $\mathbf{W}$ is an orthogonal matrix, we have $|W_{ii}| \le 1$ for all i. Hence, the linear expression $\operatorname{tr}(\boldsymbol{\Sigma}\mathbf{W}) = \sum_i \sigma_i W_{ii}$ is maximized when $\mathbf{W} = \mathbf{I}$, the identity matrix. We conclude that $\mathbf{R} = \mathbf{VU}^\top$ is an optimal solution to $\max_{\mathbf{R}\in\mathcal{O}(n)} \operatorname{tr}(\mathbf{SR})$ and the optimum value is $\sum_{i=1}^n \sigma_i$. □

We have been unable to find a complete, rigorous solution of Wahba's problem in the literature for dimensions more than 3. For the sake of completeness, we give a complete proof. This proof was obtained in collaboration with Abhishek Kumar, simplifying a previous version given in the submitted version of this paper.

Proof of Lemma 2 Recall that we want to compute

$$\max_{\mathbf{R}\in\mathcal{SO}(n)} \operatorname{tr}(\mathbf{SR}).$$

As in the proof of Lemma 3, let $\mathbf{S} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\top$ be an SVD of $\mathbf{S}$. As before, we do a change of variables from a rotation matrix $\mathbf{R}$ to an orthogonal matrix $\mathbf{W} = \mathbf{V}^\top\mathbf{RU}$, with the following condition on the determinant of $\mathbf{W}$:

$$\det(\mathbf{W}) = \det(\mathbf{V})\det(\mathbf{R})\det(\mathbf{U}) = \det(\mathbf{U})\det(\mathbf{V}) =: s \in \{+1, -1\}.$$

Using Eq (12), the problem now reduces to:

$$\max_{\mathbf{W}\in\mathcal{O}(n), \det(\mathbf{W})=s} \operatorname{tr}(\boldsymbol{\Sigma}\mathbf{W}). \tag{13}$$

The case $\det(\mathbf{W}) = 1$ is easy. We already showed in the previous lemma that

$$\mathbf{I} \in \operatorname{argmax}_{\mathbf{W}\in\mathcal{O}(n)} \operatorname{tr}(\boldsymbol{\Sigma}\mathbf{W}).$$

Thus in this case the constraint on the determinant of $\mathbf{W}$ is immaterial and the optimum value is $\sum_{i=1}^n \sigma_i = \sum_{i=1}^{n-1} \sigma_i + s\sigma_n$, where σ_n is the smallest singular value.

The case $\det(\mathbf{W}) = -1$ is considerably harder. We need to show $\mathbf{W} = \operatorname{diag}(1, 1, \ldots, 1, -1)$ is an optimal solution which has value $\sum_{i=1}^{n-1} \sigma_i - \sigma_n = \sum_{i=1}^{n-1} \sigma_i + s\sigma_n$, where σ_n is the smallest singular value.

Let $\mathbf{W}$ be an arbitrary orthogonal matrix of determinant -1. We make the following observations regarding $\mathbf{W}$. First, if $\lambda_1, \lambda_2, \ldots, \lambda_n$ are the n (real or complex) eigenvalues of $\mathbf{W}$, then we have

$$\prod_{i=1}^{n} \lambda_i = \det(\mathbf{W}) = -1.$$

Since $\mathbf{W} \in \mathcal{O}(n)$, all eigenvalues λ_i have magnitude $|\lambda_i| = 1$: this is because if λ is an eigenvalue of $\mathbf{W}$ with eigenvector $\mathbf{v}$, i.e. $\mathbf{W}\mathbf{v} = \lambda\mathbf{v}$, then

$$\|\mathbf{v}\| \;=\; \|\mathbf{W}\mathbf{v}\| \;=\; \|\lambda\mathbf{v}\| \;=\; |\lambda|\|\mathbf{v}\|,$$

where the second equality uses $\mathbf{W} \in \mathcal{O}(n)$.

We claim that at least one eigenvalue of the matrix $\mathbf{W}$ is -1. This is so because all the complex eigenvalues of the *real* matrix $\mathbf{S}$ must occur in complex conjugate pairs, $a + ib$ and $a - ib$, for some $b \neq 0$ and $a^2 + b^2 = 1$. Now, the product of any such complex conjugate pair of eigenvalues is $(a + ib)(a - ib) = a^2 + b^2 = 1$. Hence, the product of all complex eigenvalues is 1. Since the product of *all* eigenvalues (real or complex) is -1, and all real eigenvalues are either $+1$ or -1, we must have at least one eigenvalue being -1.

For convenience of notation, let $\lambda_n = -1$. Now, we have

$$\mathrm{tr}(\mathbf{W}) \;=\; \sum_{i=1}^{n-1} \lambda_i + \lambda_n \;=\; \sum_{i=1}^{n-1} \mathrm{real}(\lambda_i) - 1 \;\leq\; n - 1 - 1 \;=\; n - 2.$$

Here, $\mathrm{real}(z)$ is the real part of a complex number z, and we use the fact that the sum of two complex conjugate eigenvalues $a + ib$ and $a - ib$ is $2a$, which is the sum of their real parts. We also used the fact that for any eigenvalue λ_i, $\mathrm{real}(\lambda_i) \leq 1$ since $|\lambda_i| = 1$.

Finally, note that $|W_{ii}| \leq 1$ since $\mathbf{W}$ is an orthogonal matrix. Now, consider the following linear program which is a relaxation of the optimization problem (13) [This is a relaxation since the last inequality holds for all solutions of (13) and we drop the constraint that $\mathbf{W}$ is an orthogonal matrix of determinant -1]:

$$\max_{\forall i:\, -1 \leq W_{ii} \leq 1 \text{ and } \sum_{i=1}^{n} W_{ii} \leq n-2} \; \sum_{i=1}^{n} \sigma_i W_{ii}.$$

The optimal solution to this linear program is obtained at a vertex of the polytope defined by the constraints. We now characterize the vertices of the polytope as follows:

Claim 1 *Any vertex of the polytope defined by the constraints of the above linear program satisfies* $W_{ii} \in \{+1, -1\}$ *for all* i*, with at least one* W_{ii} *set to* -1.

Proof Any vertex is obtained by setting n of the inequalities to equalities.

Case 1: n of the $-1 \leq W_{ii} \leq 1$ inequalities are tight. Then all $W_{ii} \in \{-1, +1\}$, and to satisfy $\sum_{i=1}^{n} W_{ii} \leq n - 2$, we must have at least one -1.

Case 2: $\sum_{i=1}^{n} W_{ii}$ equals the integer $n-2$, exactly $n-1$ of the inequalities $-1 \leq W_{ii} \leq 1$ are tight for say $1 \leq i \leq n-1$, and the last one is not tight, i.e. $-1 < W_{nn} < 1$. Then for all $1 \leq i \leq n-1$, we have $W_{ii} \in \{+1, -1\}$, since $\sum_{i=1}^{n} W_{ii} = n - 2$, an integer, W_{nn} is also an integer, and hence must be zero. But then with $W_{ii} \in \{+1, -1\}$ for $1 \leq i \leq n-1$ the sum $\sum_{i=1}^{n-1} W_{ii}$ is either $n - 1$ or at most $n - 3$. Thus $\sum_{i=1}^{n} W_{ii} = n - 2$ can't be satisfied and case 2 does not give any more vertices. □

With this characterization of the vertices, since the σ_n is the smallest singular value, the optimal vertex for the linear program is the one where $W_{ii} = 1$ for $1 \leq i \leq n-1$, and $W_{nn} = -1$. Thus the optimum value of the linear program is $\sum_{i=1}^{n-1} \sigma_i - \sigma_n$. Since this is a relaxation to the original problem, this optimum value is only larger than the optimum of the original problem. However, by setting $\mathbf{W} = \text{diag}(1, 1, \ldots, 1, -1)$, which is an orthogonal matrix of determinant -1, we achieve the same value in the original problem as in the relaxed LP, and hence the optimal solution to the original problem is given by this $\mathbf{W}$. □

Appendix 2: A characterization of the convex hull of the orthogonal matrices

Let $||\mathbf{M}||_2$ denote the spectral norm of matrix $\mathbf{M}$ which is the maximum singular value of $\mathbf{M}$. Alternatively, this norm can be defined as: $||\mathbf{M}||_2 := \max\{||\mathbf{Mx}|| : ||\mathbf{x}|| = 1\}$.

Lemma 6 *The convex hull B of all $n \times n$ orthogonal matrices is exactly the set of matrices of spectral norm at most* 1*, that is*

$$B = \{\mathbf{M} \in \mathbf{R}^{n\times n} : \|\mathbf{M}\|_2 \leq 1\} = \bigcap_{\mathbf{x}:||\mathbf{x}||=1} \{\mathbf{W} : ||\mathbf{Wx}|| \leq 1\}. \tag{14}$$

Furthermore, given any matrix $\mathbf{M} \in B$, there is a polynomial time randomized rounding algorithm, which produces a random orthogonal matrix $\widetilde{\mathbf{M}}$ such that $\mathbf{E}[\widetilde{\mathbf{M}}] = \mathbf{M}$.

Proof The second equality of (14) follows from the alternate definition of the spectral norm. To show that the convex hull of $\mathcal{O}(n)$ is the set of matrices of spectral norm at most 1, first observe that since all singular values of any orthogonal matrix are equal to 1, we have $\mathcal{O}(n) \subseteq B$. Hence the convex hull of $\mathcal{O}(n)$ is contained in B. We now give the randomized rounding procedure, which automatically implies that B is contained in the convex hull of $\mathcal{O}(n)$, completing the characterization.

Let $\mathbf{M} \in B$ be any square matrix, and let $\mathbf{M} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$ be the SVD of $\mathbf{M}$, where $\mathbf{\Sigma} = \text{diag}(\sigma_1, \sigma_2, \ldots, \sigma_n)$. Since $\|\mathbf{M}\|_2 \leq 1$, the singular value $\sigma_i \in [0, 1]$ for all $i = 1, 2, \ldots, n$. Consider the random diagonal matrix $\widetilde{\mathbf{\Sigma}} = \text{diag}(\tilde{\sigma}_1, \tilde{\sigma}_2, \ldots, \tilde{\sigma}_n)$ defined as

$$\tilde{\sigma}_i = \begin{cases} 1 & \text{with probability } (1+\sigma_i)/2 \\ -1 & \text{with probability } (1-\sigma_i)/2 \end{cases}$$

Note that for all i, we have $\mathbf{E}[\tilde{\sigma}_i] = \sigma_i$ and therefore $\mathbf{E}[\widetilde{\mathbf{\Sigma}}] = \mathbf{\Sigma}$. Furthermore, $\widetilde{\mathbf{\Sigma}}$ is an orthogonal matrix, and hence $\widetilde{\mathbf{M}} = \mathbf{U}\widetilde{\mathbf{\Sigma}}\mathbf{V}^\top$ is an orthogonal matrix too. Finally, by the linearity of the expectation, $\mathbf{E}[\widetilde{\mathbf{M}}] = \mathbf{M}$. □

References

Abrudan, T., Eriksson, J., Koivunen, V. (2008a). Efficient Riemannian algorithms for optimization under unitary matrix constraint. In *IEEE international conference on acoustics, speech and signal processing (ICASSP '08)*, pp. 2353–2356. IEEE, March (2008).

Abrudan, T. E., Eriksson, J., & Koivunen, V. (2008b). Steepest descent algorithms for optimization under unitary matrix constraint. *IEEE Transactions on Signal Processing*, *56*(3), 1134–1146.

Arora, R. (2009). On learning rotations. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, & A. Culotta (Eds.), *Advances in neural information processing systems* (Vol. 22, pp. 55–63). Cambridge: MIT Press.

Bernstein, D. S. (2009). *Matrix mathematics: Theory, facts, and formulas*. Princeton: Princeton University Press.

Cesa-Bianchi, N., Long, P., & Warmuth, M. K. (1996a). Worst-case quadratic loss bounds for on-line prediction of linear functions by gradient descent. *IEEE Transactions on Neural Networks*, *7*(2), 604–619.

Cesa-Bianchi, N., Philip, M. L., & Manfred, K. W. (1996b). Worstcase quadratic loss bounds for prediction using linear functions and gradient descent. *IEEE Transactions on Neural Networks and Learning Systems*, *7*(3), 604–619.

Doran, C., Hestenes, D., Sommen, F., & Van Acker, N. (1993). Lie groups as spin groups. *Journal of Mathematical Physics*, *34*(8), 3642–3669.

Freund, Y., & Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, *55*, 119–139.

Haagerup, U. (1982). The best constants in the Khintchine inequality. *Studia Mathematica*, *70*(3), 427–485.

Hazan, E., Kale, S., Warmuth, M. K. (2010a). Corrigendum to "learning rotations with little regret". http://www.colt2010.org/papers/rotfixfinal.pdf, September (2010).

Hazan, E., Kale, S., Warmuth, M. K. (2010b). Learning rotations with little regret. In *COLT '10*, June (2010). A corrigendum can be found at the conference website.

Herbster, M., & Warmuth, M. K. (2001). Tracking the best linear predictor. *Journal of Machine Learning Research*, *1*, 281–309.

Helmbold, D. P., & Warmuth, M. K. (2009). Learning permutations with exponential weights. *Journal of Machine Learning Research*, *10*, 1705–1736.

Kivinen, J., & Warmuth, M. K. (1997). Additive versus exponentiated gradient updates for linear prediction. *Information and Computation*, *132*(1), 1–64.

Kotłowski, W., Warmuth, M. K. (2011). Minimax algorithm for learning rotations. In *Proceedings of the 24rd annual conference on learning theory (COLT '11)*, June (2011).

Nie, J. (2015). Optimal learning with matrix parameters. Ph.D. thesis, University of California, Santa Cruz.

Schonemann, P. (1966). A generalized solution of the orthogonal Procrustes problem. *Psychometrika 31*(1), 1–10.

Smith, A., Warmuth, M. K. (2008). Learning rotations. In: *Proceedings of the 21st annual conference on learning theory (COLT '08)*, p. 517, July (2008).

Smith, A. M., Warmuth, M. K. (2010). Learning rotations online. Technical Report UCSC-SOE-10-08, Department of Computer Science, University of California, Santa Cruz, February (2010).

Wahba, G. (1966). Problem 65–1, a least squares estimate of satellite attitude. *SIAM Review*, *8*(3), 384–386.

Wareham, R., Cameron, J., Lasenby, J. (2005). Applications of conformal geometric algebra in computer vision and graphics. In *6th international workshop IWMM 2004*, pp. 329–349 (2005).

Warmuth, M. K., & Kuzmin, D. (2011). Online variance minimization. *Journal of Machine Learning*, *87*(1), 1–32.

Warmuth, M. K., Koolen, W. M., Helmbold, D. P. (2011). Combining initial segments of lists. In *Proceedings of the 22nd international conference on algorithmic learning theory (ALT '11)*, pp. 219–233. Springer-Verlag, October (2011).

Zinkevich, M. (2003). Online convex programming and generalized infinitesimal gradient ascent. In *ICML*, pp. 928–936 (2003)