An Efficient and Stable Algorithm for Learning Rotations

Raman Arora
Deptartment of Electrical Engineering
University of Washington
Seattle, WA 98105

William A. Sethares
Dept. of Electrical & Comp. Engineering
University of Wisconsin-Madison
Madison, WI 53706

Abstract

This paper analyses the computational complexity and stability of an online algorithm recently proposed for learning rotations. The proposed algorithm involves multiplicative updates that are matrix exponentiated skew-symmetric matrices comprising the Lie algebra of the rotation group. The rank-deficiency of the skew-symmetric matrices involved in the updates is exploited to reduce the updates to a simple quadratic form. The Lyapunov stability of the algorithm is established and the Frobenius norm is shown to be a Lyapunov function for the updates. The application of the algorithm to registration of point-clouds in n-dimensional Euclidean space is discussed.

1 Introduction

Learning rotations is an important aspect of data processing in many areas including computer vision, robotics, graphics, physics and quantum mechanics. The problem of learning rotations can be broadly classified into batch and online version. Whereas in the batch version two point-clouds are given that can be related with a change-of-basis, in the online version a new pair of corresponding points is obtained at each instance. The batch version is well understood as the problem of orthogonal Procrustes [9] or least-squares linear-fitting [11, 4]. The online version, on the other hand, is challenging and was recently posed as an open problem [10].

Online learning of rotations is especially of interest when the rotations are changing continuously. The complexity of online learning of rotations stems from the fact that the set of rotation matrices is a curved space. The curved space associated with rotation matrices in n-dimensional Euclidean space is the unit sphere $\mathbf{S}^{n-1}$ in $\mathbb{R}^n$. Consequently, changing rotations may not be tracked using a standard Kalman filter. The gradient of a loss function on $\mathbf{S}^{n-1}$ gives the geodesic direction and velocity vector on $\mathbf{S}^{n-1}$. However, a naive steepest descent algorithms designed for $\mathbb{R}^n$ takes a step in direction of the gradient in $\mathbb{R}^n$, thereby steering off the manifold. This leads to updates that require repeated approximation and projection on to $\mathbf{S}^{n-1}$.

In [2], we proposed an algorithm for online learning of rotations and discussed its application to tracking rotations in n-dimensional Euclidean space. The key idea in the development of the updates in [2] was to utilize the parallel-transport mechanism along the geodesics on the unit sphere. This avoids the need for repeated approximation and projection on to $\mathbf{S}^{n-1}$. The resulting updates are similar to steepest descent algorithms on Riemannian manifolds for optimization under unitary matrix constraint [1]. However, since the updates involve expensive matrix-exponentiation, they are quite comparable in computational complexity to repeated projection and approximation methods [10]. One of the contributions in this paper is proving a reduction in computational complexity of the online algorithm presented in [2]. The second main result of the paper is proving a stability result for the updates proposed in this paper. We show that the Frobenius norm of the difference between the true rotation matrix and the estimated rotation matrix is non-increasing.

The proposed algorithm offers another desirable feature, that of averaging over the rotation group. This finds application in registration of noisy point-clouds in n-dimensional Euclidean space. Note that the group of rotation matrices does not admit the structure of a vector space. Consequently, the observation noise in the point-clouds cannot be dealt with by averaging rotation matrices. However, with the proposed updates, the averaging actually takes place in the Lie algebra (i.e. the tangent vector- space at the identity rotation) associated with the group of rotation matrices. The application of the algorithm to the registration of noisy point-clouds is in many ways similar to iterative closest point (ICP) method restricted to pure rotations [4]. It should be remarked though that ICP acts only on three dimensional data whereas the proposed updates apply to rotations of n-dimensional data.

Finally, a famous result by Doran [6] states that the rotation group provides a representation for all Lie groups. Therefore, the results presented here can be generalized to any Lie group under a suitable conformal map.

2 Online Algorithm

Let $\mathbf{D}_1 = \{\mathbf{x}_i\}_{i=1}^M$ and $\mathbf{D}_2 = \{\mathbf{y}_i\}_{i=1}^N$ be two point-clouds in $\mathbb{R}^n$. Without loss of generality, assume that $\mathbf{x}_i$ and $\mathbf{y}_i$ are unit vectors. Let $\mathbf{R}_*$ be an unknown $n \times n$ rotation matrix (or a change-of-basis transformation) that relates the two point-clouds. Recall that a rotation matrix $\mathbf{R}$ satisfies the properties that $\mathbf{R}^T\mathbf{R} = \mathbf{I}$, $\mathbf{R}^T\mathbf{R} = \mathbf{I}$ and $\det(\mathbf{R}) = 1$. At each instance, we learn a new corresponding pair of points $(\mathbf{x}_t, \mathbf{y}_t) \in \mathbf{D}_1 \times \mathbf{D}_2$ in the two point-clouds. The matrix $\mathbf{R}_*$ acts on $\mathbf{x}_t$ to give the rotated vector $\mathbf{y}_t = \mathbf{R}_*\mathbf{x}_t$. Let $\hat{\mathbf{R}}_t$ denote the estimate of $\mathbf{R}_*$ at instance t and $\hat{\mathbf{y}}_t = \hat{\mathbf{R}}_t\ \mathbf{x}_t$ represent the prediction for the rotated vector $\mathbf{y}_t$. Let $L_t(\hat{\mathbf{R}}_t) = d(\mathbf{y}_t, \hat{\mathbf{R}}_t\mathbf{x}_t)$ denote the loss incurred due to error in prediction of rotation of the input vector $\mathbf{x}_t$. The estimate of the rotation needs to be updated based on the loss incurred at every instance and the objective is to develop an algorithm for learning $\mathbf{R}_*$. Consider the online updates proposed in [2],

$$\hat{\mathbf{R}}_{t+1} = \hat{\mathbf{R}}_t\ \mathbf{exp}\left(-\eta\ \mathbf{skew}\ \left(\hat{\mathbf{R}}_t^T\ \nabla_{\hat{\mathbf{R}}_t} L_t(\hat{\mathbf{R}}_t)\right)\right),$$

where $\mathbf{skew}\ (\cdot)$ is the skew-symmetrization operator on the matrices, $\mathbf{skew}\ (\mathbf{A}) = \mathbf{A} - \mathbf{A}^T$. With the squared-error loss-function $L_t(\hat{\mathbf{R}}_t) = ||\hat{\mathbf{y}}_t - \mathbf{y}_t||^2$, the resulting updates are

$$\hat{\mathbf{R}}_{t+1} = \hat{\mathbf{R}}_t\ \mathbf{exp}\left(-2\eta\ \mathbf{skew}\left(\hat{\mathbf{R}}_t^T(\hat{\mathbf{y}}_t - \mathbf{y}_t)\mathbf{x}_t^T\right)\right). \quad (1)$$

It is easy to check that if $\hat{\mathbf{R}}_t$ is a rotation matrix then $\hat{\mathbf{R}}_{t+1}$ given by the updates in (1) is also a rotation matrix [2].

3 Computational complexity of updates

The updates presented in [2] (see eqn. (1) above) ensure that the estimates for the rotation matrix stay on the manifold associated with the rotation group at each iteration. However, with the matrix exponentiation at each step, the updates are computationally intensive and in fact the computational complexity of the updates is comparable to other approaches that would require repeated approximation and projection on to the manifold. The next result discusses a complexity reduction result to establish a simpler update by exploiting the eigen-structure of the update matrix.

Lemma 3.1. *The matrix exponentiated gradient updates in eqn.* (1) *are equivalent to the following updates,*

$$\hat{\mathbf{R}}_{t+1} \quad = \quad \hat{\mathbf{R}}_t \left(\mathbf{I} + \frac{\sin(\lambda)}{\lambda}\mathbf{S} + \frac{1-\cos(\lambda)}{\lambda^2}\mathbf{S}^2\right) \quad (2)$$

where $\mathbf{S} = -2\eta\ \boldsymbol{skew}\left(\hat{\mathbf{R}}_t^T(\hat{\mathbf{y}}_t - \mathbf{y}_t)\mathbf{x}_t^T\right)$ *is the skew-symmetric matrix in eqn.* (1) *with eigenvalues* $\pm j\lambda$, *for* $\lambda = 2\eta\sqrt{1 - \left(\hat{\mathbf{y}}_t^T\mathbf{y}_t\right)^2}$.

Proof. First observe that the matrix $\mathbf{S}$ can be written as

$$\begin{aligned}
\mathbf{S} &= -2\eta\ \mathbf{skew}\left(\hat{\mathbf{R}}_t^T(\hat{\mathbf{y}}_t - \mathbf{y}_t)\mathbf{x}_t^T\right), \\
&= -2\eta\ \mathbf{skew}\left(\hat{\mathbf{R}}_t^T(\hat{\mathbf{R}}_t\mathbf{x}_t - \mathbf{R}_*\mathbf{x}_t)\mathbf{x}_t^T\right), \\
&= -2\eta\ \mathbf{skew}\left(\mathbf{x}_t\mathbf{x}_t^T - \hat{\mathbf{R}}_t^T\mathbf{R}_*\mathbf{x}_t\mathbf{x}_t^T\right), \\
&= 2\eta\ \left(\hat{\mathbf{R}}_t^T\mathbf{R}_*\mathbf{x}_t\mathbf{x}_t^T - \mathbf{x}_t\mathbf{x}_t^T\mathbf{R}_*^T\hat{\mathbf{R}}_t\right), \\
&= 2\eta\ \left(\hat{\mathbf{R}}_t^T\mathbf{y}_t\mathbf{x}_t^T - \mathbf{x}_t\mathbf{y}_t^T\hat{\mathbf{R}}_t\right),
\end{aligned}$$

where $\mathbf{y}_t = \mathbf{R}_*\mathbf{x}_t$. Each term in the matrix $\mathbf{S}$ is a rank-one matrix. Thus $\mathbf{S}$ is at most rank-two. Since $\mathbf{S}$ is skew-symmetric, it has (at most) two eigenvalues in a complex conjugate pair $\pm j\lambda$ (and $n-2$ zero eigenvalues) [5]. Furthermore, Butler shows in (2.1) of [5] that the nonzero eigenvalues of the sum of two rank-one matrices $\mathbf{u}_1\mathbf{v}_1^T + \mathbf{u}_2\mathbf{v}_2^T$ can be expressed in closed form as

$$\tilde{\lambda} = \frac{1}{2}\Bigg(\mathbf{v}_1^T\mathbf{u}_1 + \mathbf{v}_2^T\mathbf{u}_2 \pm \sqrt{(\mathbf{v}_1^T\mathbf{u}_1 - \mathbf{v}_2^T\mathbf{u}_2)^2 + 4(\mathbf{v}_1^T\mathbf{u}_2)(\mathbf{v}_2^T\mathbf{u}_1)}\Bigg). \quad (3)$$

For matrix $\mathbf{S}$, write $\mathbf{u}_1 = 2\eta\hat{\mathbf{R}}_t^T\mathbf{y}_t$, $\mathbf{v}_1 = \mathbf{x}_t$, $\mathbf{u}_2 = \mathbf{x}_t$, and $\mathbf{v}_2 = -2\eta\hat{\mathbf{R}}_t^T\mathbf{y}_t$. Then the first term in parentheses in eqn. (3) can be written as

$$\begin{aligned}
\mathbf{v}_1^T\mathbf{u}_1 + \mathbf{v}_2^T\mathbf{u}_2 &= 2\eta(\mathbf{x}_t^T\hat{\mathbf{R}}_t^T\mathbf{y}_t - \mathbf{y}_t^T\hat{\mathbf{R}}_t\mathbf{x}_t) \\
&= 2\eta\left(\hat{\mathbf{y}}_t^T\mathbf{y}_t - \mathbf{y}_t^T\hat{\mathbf{y}}_t\right) \\
&= 0
\end{aligned}$$

And the term $4(\mathbf{v}_1^T\mathbf{u}_2)(\mathbf{v}_2^T\mathbf{u}_1)$ in (3) can be simplified as

$$\begin{aligned}
4(\mathbf{v}_1^T\mathbf{u}_2)(\mathbf{v}_2^T\mathbf{u}_1) &= 4(\mathbf{x}_t^T\mathbf{x}_t)(-4\eta^2\mathbf{y}_t^T\hat{\mathbf{R}}_t\hat{\mathbf{R}}_t^T\mathbf{y}_t) \\
&= -16\eta^2\mathbf{y}_t^T\mathbf{y}_t \\
&= -16\eta^2
\end{aligned}$$

Then the eigenvalues given by eqn. (3) are

$$
\begin{aligned}
\tilde{\lambda} &= \pm\frac{1}{2}\sqrt{\left(2\eta(\mathbf{x}_t^T\hat{\mathbf{R}}_t^T\mathbf{y}_t+\mathbf{y}_t^T\hat{\mathbf{R}}_t\mathbf{x}_t)\right)^2-16\eta^2} \\
&= \pm\frac{1}{2}\sqrt{4\eta^2\left(\hat{\mathbf{y}}_t^T\mathbf{y}_t+\mathbf{y}_t^T\hat{\mathbf{y}}_t\right)^2-16\eta^2} \\
&= \pm\eta\sqrt{\left(2\,\hat{\mathbf{y}}_t^T\mathbf{y}_t\right)^2-4} \\
&= \pm 2\eta\sqrt{\left(\hat{\mathbf{y}}_t^T\mathbf{y}_t\right)^2-1} \\
&= \pm j\; 2\eta\sqrt{1-\left(\hat{\mathbf{y}}_t^T\mathbf{y}_t\right)^2}
\end{aligned}
$$

In order to simplify the exponential of $\mathbf{S}$, a generalization of the Rodrigues' formula from [7] is useful. When $\mathbf{S}$ is skew-symmetric and rank 2

$$e^{\mathbf{S}}=\mathbf{I}+\frac{\sin(\lambda)}{\lambda}\mathbf{S}+\frac{1-\cos(\lambda)}{\lambda^2}\mathbf{S}^2$$

where $\lambda=2\eta\sqrt{1-\left(\hat{\mathbf{y}}_t^T\mathbf{y}_t\right)^2}$. □

Owing to the result above the matrix exponential reduces to a simple quadratic form involving an element from the Lie algebra of the rotation group.

4 Stability of updates

Finally, we establish the Lyapunov stability of the proposed updates and show that Frobenius norm is a Lyapunov function. First note that the iteration (1) has an equilibrium at $\mathbf{R}_*$ since

$$
\begin{aligned}
\mathbf{S} &= 2\eta\left(\hat{\mathbf{R}}_t^T\mathbf{y}_t\mathbf{x}_t^T-\mathbf{x}_t\mathbf{y}_t^T\hat{\mathbf{R}}_t\right) \\
&= 2\eta\left(\hat{\mathbf{R}}_t^T\mathbf{R}_*\mathbf{x}_t\mathbf{x}_t^T-\mathbf{x}_t\mathbf{x}_t^T\mathbf{R}_*^T\hat{\mathbf{R}}_t\right) \qquad (4)
\end{aligned}
$$

is equal to $\mathbf{0}_n$ (the $n\times n$ matrix of all zeroes) when $\hat{\mathbf{R}}_t=\mathbf{R}_*$. Since $\mathbf{exp}\,(\mathbf{0}_n)=\mathbf{I}$, $\hat{\mathbf{R}}_{t+1}=\hat{\mathbf{R}}_t$, and the iteration remains fixed.

A Lyapunov function [8] for a discrete iteration (such as (1)) about its equilibrium $\mathbf{R}_*$ is a function $V(\cdot)$ with the properties: (a) $V(\mathbf{R}_*-\hat{\mathbf{R}}_t)$ is positive definite, (b) $V(0)=0$, and (c) $V(\mathbf{R}_*-\hat{\mathbf{R}}_{t+1})\le V(\mathbf{R}_*-\hat{\mathbf{R}}_t)$ for all t. The equilibrium is Lyapunov stable if there exists such a V.

Theorem 4.1. *The Frobenius norm* $V(\mathbf{R})=||\mathbf{R}||_F=\sqrt{\sum_{i=1}^n\sum_{j=1}^n|\mathbf{R}_{ij}|^2}=\sqrt{tr\,(\mathbf{R}^T\mathbf{R})}$ *is a Lyapunov function for the algorithm* (1) *about its equilibrium* $\mathbf{R}_*$.

Proof. The first two properties follow immediately from the fact that $||\cdot||_F$ is a norm. The third property, that $||\mathbf{R}_*-\hat{\mathbf{R}}_{t+1}||_F\le||\mathbf{R}_*-\hat{\mathbf{R}}_t||_F\ \forall t$ remains to be shown. Because $\mathbf{R}_*$ and $\hat{\mathbf{R}}_t$ are rotation matrices, $\text{tr}\left(\mathbf{R}_*^T\mathbf{R}_*\right)=\text{tr}\left(\hat{\mathbf{R}}_t^T\hat{\mathbf{R}}_t\right)=\text{tr}\,(\mathbf{I})=n$, and the norm simplifies to

$$
\begin{aligned}
||\mathbf{R}_*-\hat{\mathbf{R}}_t||_F^2 &= \text{tr}\left((\mathbf{R}_*-\hat{\mathbf{R}}_t)^T(\mathbf{R}_*-\hat{\mathbf{R}}_t)\right) \\
&= \text{tr}\left(\mathbf{R}_*^T\mathbf{R}_*-\mathbf{R}_*^T\hat{\mathbf{R}}_t-\hat{\mathbf{R}}_t^T\mathbf{R}_*+\hat{\mathbf{R}}_t^T\hat{\mathbf{R}}_t\right) \\
&= 2n-2\,\text{tr}\left(\mathbf{R}_*^T\hat{\mathbf{R}}_t\right).
\end{aligned}
$$

Similarly, Lemma 1 shows that $\hat{\mathbf{R}}_{t+1}$ is a rotation matrix and so $\text{tr}\left(\hat{\mathbf{R}}_{t+1}^T\hat{\mathbf{R}}_{t+1}\right)=n$. Hence

$$||\mathbf{R}_*-\hat{\mathbf{R}}_{t+1}||_F^2=2n-2\text{tr}\left(\mathbf{R}_*^T\hat{\mathbf{R}}_{t+1}\right).$$

Thus the condition that $||\mathbf{R}_*-\hat{\mathbf{R}}_{t+1}||_F\le||\mathbf{R}_*-\hat{\mathbf{R}}_t||_F$ reduces to the condition that

$$\text{tr}\left(\mathbf{R}_*^T\hat{\mathbf{R}}_{t+1}\right)-\text{tr}\left(\mathbf{R}_*^T\hat{\mathbf{R}}_t\right)\ge 0.$$

Using lemma 3.1 the left hand side of the inequality above can be expressed as

$$
\begin{aligned}
&\text{tr}\left(\mathbf{R}_*^T\hat{\mathbf{R}}_{t+1}\right)-\text{tr}\left(\mathbf{R}_*^T\hat{\mathbf{R}}_t\right) \\
&\quad= \text{tr}\left(\tfrac{\sin(\lambda)}{\lambda}\mathbf{R}_*^T\hat{\mathbf{R}}_t\mathbf{S}+\tfrac{1-\cos(\lambda)}{\lambda^2}\mathbf{R}_*^T\hat{\mathbf{R}}_t\mathbf{S}^2\right), \\
&\quad= \tfrac{\sin(\lambda)}{\lambda}\text{tr}\,(\mathbf{AS})+\tfrac{1-\cos(\lambda)}{\lambda^2}\text{tr}\left(\mathbf{AS}^2\right), \qquad (5)
\end{aligned}
$$

where $\mathbf{A}=\mathbf{R}_*^T\hat{\mathbf{R}}_t$. Furthermore, define $\mathbf{X}=\mathbf{x}\mathbf{x}^T$. Then using (4), $\mathbf{S}$ can be re-written as $\mathbf{S}=2\eta(\mathbf{A}^T\mathbf{X}-\mathbf{XA})$. Treating the two terms in the right hand side separately, write

$$
\begin{aligned}
\text{tr}\,(\mathbf{AS}) &= 2\eta\cdot\text{tr}\left(\mathbf{A}(\mathbf{A}^T\mathbf{X}-\mathbf{XA})\right) \\
&= 2\eta\,(\text{tr}\,(\mathbf{X})-\text{tr}\,(\mathbf{AXA})) \\
&= 2\eta\,(1-\mathbf{x}^T\mathbf{A}^2\mathbf{x}). \qquad (6)
\end{aligned}
$$

The final equality follows because the arguments of the trace function can be circularly permuted. Therefore, $\text{tr}\,(\mathbf{AXA})=\text{tr}\left(\mathbf{A}\mathbf{x}\mathbf{x}^T\mathbf{A}\right)=\text{tr}\left(\mathbf{x}\mathbf{x}^T\mathbf{AA}\right)=\text{tr}\left(\mathbf{x}^T\mathbf{A}^2\mathbf{x}\right)$. Similarly, the identities $\mathbf{X}^2=\mathbf{X}$ and $\text{tr}\,(\mathbf{AX})=\text{tr}\left(\mathbf{XA}^T\right)$ help simplify

$$
\begin{aligned}
\text{tr}\left(\mathbf{AS}^2\right) &= 4\eta^2\text{tr}\left(\mathbf{A}(\mathbf{A}^T\mathbf{X}-\mathbf{XA})(\mathbf{A}^T\mathbf{X}-\mathbf{XA})\right) \\
&= 4\eta^2\text{tr}\left(\mathbf{XA}^T\mathbf{X}-\mathbf{AX}-\mathbf{XA}+\mathbf{AXAXA}\right) \\
&= 4\eta^2\text{tr}\,(-\mathbf{AX}+\mathbf{AXAXA}) \\
&= 4\eta^2\cdot(-\text{tr}\,(\mathbf{AX})+\text{tr}\,(\mathbf{AXAXA})) \\
&= 4\eta^2\cdot\left(-\mathbf{x}^T\mathbf{Ax}+(\mathbf{x}^T\mathbf{Ax})(\mathbf{x}^T\mathbf{A}^2\mathbf{x})\right) \\
&= -4\eta^2\mathbf{x}^T\mathbf{Ax}(1-\mathbf{x}^T\mathbf{A}^2\mathbf{x}). \qquad (7)
\end{aligned}
$$

Substituting (7) and (6) into (5) shows that

$$
\begin{aligned}
&\text{tr}\left(\mathbf{R}_*^T\hat{\mathbf{R}}_{t+1}\right)-\text{tr}\left(\mathbf{R}_*^T\hat{\mathbf{R}}_t\right) \\
&= 2\eta(1-\mathbf{x}^T\mathbf{A}^2\mathbf{x})\left(\tfrac{\sin(\lambda)}{\lambda}-2\eta\tfrac{1-\cos(\lambda)}{\lambda^2}\mathbf{x}^T\mathbf{Ax}\right). \qquad (8)
\end{aligned}
$$

Since $\mathbf{x}$ is a unit vector and $\mathbf{A}$ is the product of two rotations, $0 \leq \mathbf{x}^T\mathbf{A}^2\mathbf{x} \leq 1$. Accordingly, the positivity of (8) is determined by the second term on the right hand side. Letting $z = \mathbf{x}^T\mathbf{A}\mathbf{x}$, λ can be rewritten as $2\eta\sqrt{1-z^2}$. Making this substitution and multiplying through by $\lambda^2 = 1 - z^2$, the sign of the second term is the same as the sign of

$$f(z) = -z + z\cos(2\eta\sqrt{1-z^2}) + \sqrt{1-z^2}\sin(2\eta\sqrt{1-z^2}). \quad (9)$$

Consider the function $g(\theta) = f(\sin(\theta)) = -\sin(\theta) + \sin(\theta + 2\eta\cos(\theta))$ defined on the interval $\Theta = [0, \pi/2]$. First note that for $\eta \leq 1/2$, $h(\theta) = \theta + 2\eta\cos(\theta)$ is a monotonically increasing function on Θ because $h'(\theta) = 1 - 2\eta\sin(\theta) \geq 0$ for all $\theta \in \Theta$, given $\eta \leq 1/2$. Therefore, $\theta + 2\eta\cos(\theta) \leq \pi/2$. Also, $\cos(\theta) \geq 0$ on Θ and $\sin(\theta)$ is a monotonically increasing function on Θ. This implies $g(\theta) \geq 0$ on $[0, \pi/2]$ which means that $f(z)$ is nonnegative for $0 \leq z \leq 1$. Hence $\mathrm{tr}\left(\mathbf{R}_*^T\hat{\mathbf{R}}_{t+1}\right) - \mathrm{tr}\left(\mathbf{R}_*^T\hat{\mathbf{R}}_t\right) \geq 0$ and the Frobenius norm of $\mathbf{R}_* - \hat{\mathbf{R}}_t$ is a Lyapunov function. □

Since Theorem 4.1 holds for all initial states $\mathbf{R}_0$ and all possible input (unit) vectors $\mathbf{x}_t$, the result is a global Lyapunov stability. The Lyapunov stability is also evident from the simulation results for the noise-less setting in [2].

5 Experimental results

This section discusses the estimation performance of the proposed algorithm with respect to the number of observations for various step sizes of the updates. Figure 1 shows Frobenius norm between the true rotation matrix in $\mathbf{SO}(3)$ and the estimated rotation matrix using the updates in eqn. (2) for various step sizes. The observations are from a 3D face and are corrupted by additive white Gaussian noise with variance 0.01. The noise corresponds to a coarse initial registration which is typical in 3D face recognition where it is easier to identify facial landmarks (like nose or mouth region) in a pair of images than to establish exact registration (of nose tips, for instance). It is evident from the plot that for large step sizes, the noisy-observations can cause the estimation error to level-out rather than decrease at every step. For smaller step sizes a lower noise floor is achieved but requires a much larger number of instances to learn. Choosing a step-size so as to give a relatively proportional weight to the new observation (i.e the step size decreases as $\frac{1}{n}$ where n is the number of instances) leads to faster convergence and lower noise floor. This is intuitive since we are averaging over more and more data. For more details on experiments and code please check [3].

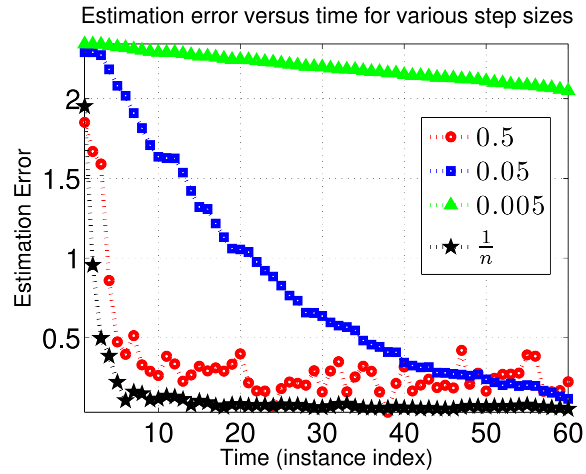

Figure 1. Estimation performance for constant step sizes ($0.5, 0.05, 0.005$) and variable step size that decreases as $\frac{1}{n}$.

6 Conclusion

This paper presented a computational complexity reduction result for online learning of rotations and also proved the Lyapunov stability of the proposed updates. The application of the algorithm to tracking rotations and registration of point-clouds in n-dimensional Euclidean space is discussed with emphasis on the choice of step-size for the gradient updates and choice of weights for computing true empirical mean over the rotation group.

References

[1] T. E. Abrudan, J. Eriksson, and V. Koivunen. Steepest descent algorithms for optimization under unitary matrix constraint. *IEEE Trans. Sig. Proc.*, March 2008.

[2] R. Arora. On learning rotations. In *NIPS*, 2009.

[3] R. Arora. Supp. Results. `http://www.cae.wisc.edu/~sethares/links/ICPR.html`, 2010.

[4] P. Besl and N. McKay. A method for registration of 3D shapes. *IEEE Trans. PAMI*, 14:239–256, 1992.

[5] L. J. Butler. *Applications of Matrix Theory to Approximation Theory*. MS Thesis, Texas Tech Univ., 1999.

[6] C. Doran, D. Hestenes, F. Sommen, and N. V. Acker. Lie groups as spin groups. *J. Math. Phys.*, Aug 1993.

[7] J. Gallier and D. Xu. Computing exponentials of skew-symmetric matrices and logarithms of orthogonal matrices. *Int. J. Robotics Automation*, 17(4), 2002.

[8] W. Hahn. *Stability of Motion*. Springer-Verlag, 1967.

[9] P. Schonemann. A generalized solution of the orthogonal procrustes problem. *Psychometrika*, Mar 1966.

[10] A. Smith and M. Warmuth. Learning rotations. In *Conference on Learning Theory*, Jun 2008.

[11] G. Wahba. Problem 65-1, a least squares estimate of satellite attitude. *SIAM Rev.*, 7(3), July 1965.