Formulation and Integration of Learning Differential Equations on the Stiefel Manifold

Simone Fiori

Abstract

The present Letter aims at illustrating the relevance of numerical integration of learning differential equations on differential manifolds. In particular, the task of learning with orthonormality constraints is dealt with, which is naturally formulated as an optimization task with the compact Stiefel manifold as neural parameter space. Intrinsic properties of the derived learning algorithms, such as stability and constraints preservation, are illustrated through experiments on minor and independent component analysis.

Keywords

Unsupervised neural network learning; Differential Geometry; Riemannian manifold; Riemannian gradient; Geodesics.

I. Introduction

In the scientific literature, it has recently been pointed out that a class of learning rules for artificial neural networks may be formulated in terms of differential equations of network's learnable parameters over smooth manifolds (see e.g. [1], [7], [14] and references therein, along with a forthcoming journal special issue's papers whose contents are summarized in [9]). The geometrical structure of the base manifolds intrinsically describes the restrictions that the network's learnable parameters are subjected to.

Within such framework, a way to formulate a learning algorithm for a given learning task consists in the following steps:

1. Describe the learning task as a constrained optimization problem, so that the desired network parameters values coincide to the set of variables values that maximize or minimize a criterion function under the specified (smooth) constraints. The constraints may be described via a suitable smooth manifold formed by all the feasible solutions.
2. Choose a suitable optimization method, that allows finding the optimum of the criterion function under the prescribed constraints. A typical method is based on the Riemannian gradient optimization, that follows the steepest descent/ascent direction in the manifold of all the feasible solutions. The structure of the gradient of the criterion function depends on the geometry of the base manifold. At this point, the gradient-based learning rule appears as a differential equation on a smooth manifold.
3. Numerically solve the mentioned differential equation through an appropriate numerical integration method that should allow us to preserve the manifold structure (up to reasonable precision). This is equivalent to defining a suitable discretization method in the time-domain that allows for converting a differential learning equation into a discrete-time algorithm.

It is important to stress out the substantial difference between the learning equations, that appear - in this context - as differential equations of variables belonging to smooth manifolds, and the learning algorithms, that appear as discrete-time rules well suited to be implemented on a computer. It is important to underline this difference because *the theoretical desirable properties of the learning differential equations, such as asymptotic convergence, do not necessarily copy into the learning algorithms, unless suitable integration methods are employed.*

With the present contribution, we aim at illustrating learning algorithms for a single neural layer whose connection matrix belongs to the Stiefel manifold, that is the set of "tall-skinny" orthonormal matrices (for a recent review, see e.g. [6]). As an appropriate approximation of the exact Riemannian gradient learning flow, the algorithms exploit approximate geodesic arcs suitably glued together. The structure of gradient and geodesic depends on the metrics that the manifold is equipped with: We illustrate the effects of two possible choices by discussing their analytical features as well as numerical experiments on minor component analysis and independent component analysis.

II. Learning algorithms on the Stiefel manifold

In the following section, we briefly recall from differential geometry the concepts of smooth manifold, metrics, gradient and geodesics. Then, these concepts are particularized to the Stiefel manifold.

S. Fiori is with the Faculty of Engineering of the Perugia University, Terni Campus, Loc. Pentima bassa, 21, I-05100 Terni (Italy). Email: fiori@unipg.it.

A. Few basic differential geometry concepts

A smooth manifold $\mathcal{M}$ of dimension q is defined as a topological Hausdorff space equipped with a covering of open sets and compatible homeomorphisms (coordinate maps) that establish a one-to-one correspondence between elements belonging to the open sets of $\mathcal{M}$ with elements of open sets of $\mathbb{R}^q$.

The tangent space to $\mathcal{M}$ in a point $x \in \mathcal{M}$ is denoted by $T_x\mathcal{M}$: It is a linear space of dimension q isomorphic to $\mathbb{R}^q$. A bilinear scalar product $g_x : T_x\mathcal{M} \times T_x\mathcal{M} \rightarrow \mathbb{R}$ turns the manifold $\mathcal{M}$ into a metric space, which is defined Riemannian manifold and is denoted with the pair $(\mathcal{M}, g)$.

Let us denote by $g^e : T_x\mathcal{M} \times T_x\mathcal{M} \rightarrow \mathbb{R}$ the Euclidean scalar product, that a tangent space may always be endowed with. In short, we also denote the derivative (or standard gradient) of a function as $\frac{\partial}{\partial x} f$. The gradient $\nabla_x f$ of a differentiable function $f : \mathcal{M} \rightarrow \mathbb{R}$ in a point x on a Riemannian manifold $(\mathcal{M}, g)$ is defined by the following two conditions: $\nabla_x f \in T_x\mathcal{M}$ (tangency condition) and $g_x(\nabla_x f, \mathbf{v}) = g^e(\frac{\partial}{\partial x} f, \mathbf{v})$ for all $\mathbf{v} \in T_x\mathcal{M}$ (compatibility condition).

A Riemannian-gradient equation for the constrained minimization/maximization of a function $f : \mathcal{M} \rightarrow \mathbb{R}$, where the smooth manifold $\mathcal{M}$ fully describes the constraints, is:

$$\dot{x}(t) = \pm \nabla_x f(x(t)) \ , \ x(0) = x_0 \in \ \mathcal{M} \ . \tag{1}$$

The exact solution $x = x(t) \in \mathcal{M}$ of the above differential equation on a manifold is termed gradient flow.

A geodesic on a curved manifold is intuitively looked upon as the generalization of a straight line over a flat space. A geodesic $\gamma_{x,\mathbf{v}}(t)$, with $t \in [0, T]$, $T > 0$, may also be thought of as the path followed by a particle departing from the point x on a smooth manifold $\mathcal{M}$ and sliding with constant speed $\mathbf{v} \in T_x\mathcal{M}$ over the manifold itself. The structure of geodesics on a Riemannian manifold does depend of the selected metrics.

The availability of geodesic for a manifold provides a suitable way for the numerical integration of the differential equation (1). In fact, as proposed e.g. in [5], [8], [14], provided we have approximately solved the equation up to point x_n, $n \in \mathbb{N}$, such approximate solution may be extended further to the point x_{n+1} by:

$$x_{n+1} = \gamma_{x_n, \pm \nabla_{x_n} f}(t_\star) \ , \tag{2}$$

where $[0, \ t_\star]$ is the time-interval that we deem it appropriate to prolong the geodesic-based solution along. Such numerical integration method on manifolds has the following features:

1. Irrespective of the extent that the geodesic-based solution is prolonged to from x_n to x_{n+1}, the approximate discrete flow $x(nt_\star) = x_n$, $n \in \mathbb{N}$, belongs completely to the base manifold $\mathcal{M}$. This is a noticeable advantage with respect to integration methods that do not exploit the underlying geometrical structure: The latter break the constraints and do not take advantage e.g. of the possible compactness of the base manifold which would ensure the *intrinsic stability of the algorithm*, that the former methods benefit from.
2. The exploitation of Riemannian gradient flow (either exact or numerical) for learning may help avoiding or mitigating the well-known problem of sticking onto local optima or plateaus [11]. Also, it is readily seen that those local optima of the criterion function f that do not belong to the search-space (that is $\mathcal{M}$) are definitely avoided. In other terms, only the local optima of the restriction $f_{|\mathcal{M}}$ are potentially dangerous.
3. The closed-form expression of geodesic is known only in a limited number of cases and, even for these, the numerical computation of geodesic flows raises several difficulties.

With regard to the last point, for the base Stiefel manifold we possess the closed-form solution of geodesic corresponding to two different metrics. However, in the section devoted to numerical results, it will be clear that the numerical evaluation of geodesic may be of serious concern both from a computational complexity and a numerical reliability viewpoints.

B. Gradients and geodesics of the Stiefel manifold

The compact real Stiefel manifold is defined by $\mathrm{St}(p, m) \stackrel{\text{def}}{=} \{\mathbf{X} \in \mathbb{R}^{p \times m} | \mathbf{X}^T\mathbf{X} = \mathbf{I}_m\}$, where $p \geq m$. It is a smooth manifold of dimension $pm - \frac{m(m+1)}{2}$. The tangent space in a given point is $T_{\mathbf{X}}\mathrm{St}(p, m) = \{\mathbf{H} \in \mathbb{R}^{p \times m} | \mathbf{H}^T\mathbf{X} + \mathbf{X}^T\mathbf{H} = \mathbf{0}_m\}$.

We may consider two metrics for the Stiefel manifold:

- Euclidean metrics [6], [17]: $g^e(\mathbf{H}_1, \mathbf{H}_2) \stackrel{\text{def}}{=} \mathrm{tr}(\mathbf{H}_1^T\mathbf{H}_2)$, $\forall \ \mathbf{H}_1, \mathbf{H}_2 \in T_{\mathbf{X}}\mathrm{St}(p, m)$.
- Canonical metrics [5], [6]: $g^c_{\mathbf{X}}(\mathbf{H}_1, \mathbf{H}_2) \stackrel{\text{def}}{=} \mathrm{tr}(\mathbf{H}_1^T(\mathbf{I}_m - \frac{1}{2}\mathbf{X}\mathbf{X}^T)\mathbf{H}_2)$, $\forall \ \mathbf{H}_1, \mathbf{H}_2 \in T_{\mathbf{X}}\mathrm{St}(p, m)$.

We next give the expressions of the gradients and of geodesics on the Stiefel manifold corresponding to the above metrics.

The gradient $\nabla^e_{\mathbf{X}} f$ of a differentiable function $f : \mathrm{St}(p, m) \rightarrow \mathbb{R}$ under the Euclidean metrics may be computed as follows: From the compatibility condition we know that $\mathrm{tr}\left(\left(\nabla^e_{\mathbf{x}} f - \frac{\partial}{\partial \mathbf{X}} f\right)^T \mathbf{H}\right) = 0$ for all $\mathbf{H} \in T_{\mathbf{X}}\mathrm{St}(p, m)$. Therefore,

$\nabla^e_{\mathbf{X}} f = \frac{\partial}{\partial \mathbf{X}} f + \mathbf{X}\mathbf{S}$, with $\mathbf{S} \in \mathbb{R}^{m\times m}$ symmetric. By plugging this expression for $\nabla^e_{\mathbf{X}} f$ into the tangency condition $(\nabla^e_{\mathbf{X}} f)^T \mathbf{X} + \mathbf{X}^T (\nabla^e_{\mathbf{X}} f) = \mathbf{0}_m$ we immediately get $\mathbf{S} = -\frac{1}{2}\left(\left(\frac{\partial}{\partial \mathbf{X}} f\right)^T \mathbf{X} + \mathbf{X}^T \left(\frac{\partial}{\partial \mathbf{X}} f\right)\right)$. In conclusion:

$$\nabla^e_{\mathbf{X}} f = \frac{\partial}{\partial \mathbf{X}} f - \frac{1}{2}\mathbf{X}\left(\left(\frac{\partial}{\partial \mathbf{X}} f\right)^T \mathbf{X} + \mathbf{X}^T \left(\frac{\partial}{\partial \mathbf{X}} f\right)\right) \ . \tag{3}$$

Analogously, the gradient $\nabla^c_{\mathbf{X}} f$ may be computed and shown to possess the structure:

$$\nabla^c_{\mathbf{X}} f = \frac{\partial}{\partial \mathbf{X}} f - \mathbf{X}\left(\frac{\partial}{\partial \mathbf{X}} f\right)^T \mathbf{X} \ . \tag{4}$$

The expression of the geodesic $\gamma^e_{\mathbf{X},\pm\nabla^e_{\mathbf{X}} f}(t)$ under the Euclidean metrics is given in closed form in [6] and reads:

$$\gamma^e_{\mathbf{X},\pm\nabla^e_{\mathbf{X}} f}(t) = [\mathbf{X} \ \pm\nabla^e_{\mathbf{X}} f] \left(\exp t \begin{pmatrix} \pm\mathbf{X}^T\nabla^e_{\mathbf{X}} f & (\nabla^e_{\mathbf{X}} f)^T(\nabla^e_{\mathbf{X}} f) \\ \mathbf{I}_m & \pm\mathbf{X}^T\nabla^e_{\mathbf{X}} f \end{pmatrix}\right) \mathbf{I}_{2m,m} \exp\left(\mp(\mathbf{X}^T\nabla^e_{\mathbf{X}} f)t\right) \ . \tag{5}$$

The composite matrix in the leftmost matrix exponent has size $2m \times 2m$, while the second matrix to be exponentiated has size $m \times m$.

The expression of the geodesic $\gamma^c_{\mathbf{X},\pm\nabla^c_{\mathbf{X}} f}(t)$ under the canonical metrics was given in closed form in [8] and reads:

$$\gamma^c_{\mathbf{X},\pm\nabla^c_{\mathbf{X}} f}(t) = \left(\exp\left[\mp t \left(\left(\frac{\partial}{\partial \mathbf{X}} f\right)\mathbf{X}^T - \mathbf{X}\left(\frac{\partial}{\partial \mathbf{X}} f\right)^T\right)\right]\right)\mathbf{X} \ . \tag{6}$$

In the above expression, the matrix to be exponentiated has size $p \times p$.

C. Effect of metrics selection in learning by optimization

As mentioned in the Introduction, the guideline of this Letter is that we suppose that a learning task may be formulated in terms of optimization of a smooth criterion, denoted here by function f, which drives network's learning or, equivalently, measures the network performance with respect to the given task. Then, network learning happens by means of the numerical algorithm (2) that provides a suitable approximation of the exact gradient flow.

As it is clear from the preceding section, the numerical flow on the constraints manifold relies on the computation of geodesic and gradient whose structure depends on the metrics that the (tangent space of the) base manifold is equipped with. It thus deserves to wonder if the choice of a specific metrics determines any advantage/disadvantage with respect to the learning task. The answer is mixed as we should distinguish between exact flow and piece-wise geodesic approximate flow:

- Ideally, the solutions $x(t) \in \mathcal{M}$ of the differential equation on manifold (1) pertaining to different norms induced by arbitrary scalar products on $T_x\mathcal{M}$ are completely equivalent. Such equivalence may be proven by the following argument: The velocity of change of function $f(x)$ may be evaluated through the chain-rule of differentiation as:

$$\frac{df}{dt} = g^e\left(\frac{\partial}{\partial x} f, \frac{dx}{dt}\right) \ . \tag{7}$$

Let $g_x(\cdot,\cdot)$ be any scalar product and $\nabla_x f$ the corresponding Riemannian gradient on $\mathcal{M}$. From equation (1) and from the compatibility condition of gradients we have:

$$\frac{df}{dt} = \pm g_x\left(\nabla_x f, \nabla_x f\right) \ , \tag{8}$$

which is independent of the selected metrics, therefore the course of the learning criterion function $f(x(t))$, is independent of the selected metrics for the base-manifold $\mathcal{M}$.

- In practical (computer-based) implementations, the computational complexity may be retained as a factor of prime importance in the discrimination of otherwise equivalent learning algorithms. For instance, the structure of the geodesics (5) and (6) and the evaluation of the sizes of the matrices to be exponentiated, suggest that the Euclidean metrics provides a computationally advantageous learning algorithm over the canonical metrics. (Of course, it is hypothesized that matrix exponentiation is the most burdensome operation. Also, the computational complexity of exponentiation as well as matrix multiplication and composite-matrices construction may be lessened by keenly exploiting appropriate numerical-geometrical techniques, which, however, are not the subject of the present contribution.)
- As a further evaluation index, it is necessary to consider the numerical stability/reliability of the selected geodesic-based numerical integration algorithms. Again, the numerical difficulties related to the evaluation of the matrix-to-matrix

function exp(·) play an important role: Basically, unless trivial cases are dealt with, in general exact computation of matrix exponential is impossible. The most popular approximation methods rely on high-order Padé expansions, truncated Taylor expansions or matrix-function evaluations based on eigenvalue decomposition of the matrix to be exponentiated (see e.g. [3], [12], [15]). Basic numerical considerations as well as experience suggest that if the geodesic arcs are prolonged for sufficiently short time-intervals (represented by the quantity $t_\star$ in equation (2)), the approximation error in matrix exponential computation may be negligible. Therefore, thanks to the special structure of the Stiefel manifold, we have:

$$\left(\gamma_{\mathbf{X},\pm\nabla_{\mathbf{X}}f}(t_\star)\right)^T\left(\gamma_{\mathbf{X},\pm\nabla_{\mathbf{X}}f}(t_\star)\right)=\mathbf{I}_m\ ,\ \forall\ \mathbf{X}\in\ \mathrm{St}(p,m)\ , \tag{9}$$

where a generic metrics has been allowed. This ensures the boundness of the learning parameters, hence the numerical stability of the related learning algorithm. Conversely, for large values of the integration step $t_\star$, the approximation error may be potentially destructive, in the sense that the network parameters might escape the base manifold ($x\notin\mathcal{M}$ at some time): This makes the learning algorithm loose desirable properties such as numerical stability. Still, geodesic-based algorithms prove far more numerically stable than conventional Euler-discretization-based ones, as shown by the wide-range numerical experiments illustrated in [10].

III. Numerical simulation results

In order to elucidate the above considerations, examples concerning minor component analysis (MCA) and kurtosis-optimization-based independent component analysis (ICA) are considered in what follows. In particular, MCA is considered in a on-line version: In this case, short integration steps is a sensible choice and we may verify the numerical equivalence of the two geodesic-based learning algorithms up to reasonable extent. Conversely, batch-type ICA is considered, for which large integration steps are allowable: In this case, the Euclidean-metrics-based algorithm shall prove to be almost completely unreliable. Of course, this depends on the way the matrix exponential is computed. In the experiments, we used MATLAB©'s 'expm' primitive.

In the following, we denote by $\mathbf{y}=\mathbf{M}^T\mathbf{x}$ the input-output relation describing a linear neural network with p inputs and m outputs (where $p>m$). Also, symbol $E[\cdot]$ denotes statistical expectation.

In the experiments, the generic numerical performances of the algorithms, run on a 128 MB 600 MHz machine, are monitored through the following indices:

- A measure of the orthonormality of $\mathbf{M}$, defined as $\mu\stackrel{\mathrm{def}}{=}\|\mathbf{M}^T\mathbf{M}-\mathbf{I}_m\|_{\mathrm{F}}^2$, where $\|\cdot\|_{\mathrm{F}}$ denotes the Frobenius norm.
- The average number of floating point operations (flops) per iteration.
- The total CPU-time that the algorithms take to run.

A. Problem/data and specific performance indices description

Minor component analysis deals with the problem of extracting the m eigenvectors of the covariance matrix $\mathbf{C}_x\stackrel{\mathrm{def}}{=}E[\mathbf{x}\mathbf{x}^T]$ of a stationary zero-mean multivariate random vector field $\mathbf{x}(t)$ corresponding to its smallest eigenvalues [16]. As $\mathbf{C}_x$ is a real symmetric matrix, its eigenvectors form an orthogonal frame $\mathbf{M}$ of $\mathbb{R}^p$, thus it holds $\mathbf{M}\in\mathrm{St}(p,m)$. Formally, we assume as learning criterion function $2f(\mathbf{M})\stackrel{\mathrm{def}}{=}\mathrm{tr}(\mathbf{M}^T\mathbf{C}_x\mathbf{M})=\mathrm{tr}(E[\mathbf{y}\mathbf{y}^T])$, which should be minimized. In on-line signal processing, it is customary to drop the expectation operator (this is equivalent to invoking the coarse approximation $\mathbf{C}_x\approx\mathbf{x}\mathbf{x}^T$). With this approximation we get $\frac{\partial}{\partial\mathbf{M}}f=\mathbf{x}\mathbf{y}^T$. The signal $\mathbf{x}(t)\in\mathbb{R}^p$ was generated as $\mathbf{x}(t)=\mathbf{V}\mathbf{D}\mathbf{s}(t)$, where $\mathbf{s}(t)$ is a zero-mean normally distributed random process, the on-diagonal entries of the diagonal matrix $\mathbf{D}$ are uniformly spaced in [0.1 2] and $\mathbf{V}\in\mathbb{R}^{p\times m}$ is a random matrix in $\mathrm{St}(p,m)$. As a specific network performance index, it was assumed:

$$\rho\stackrel{\mathrm{def}}{=}\frac{\mathrm{tr}(\mathbf{C}_1\mathbf{C}_1^T)}{\mathrm{tr}(\mathbf{C}_2\mathbf{C}_2^T)}\ ,\ \mathbf{C}_i\stackrel{\mathrm{def}}{=}\mathbf{M}^T\mathbf{E}_i\ , \tag{10}$$

where $\mathbf{E}_1$ is formed by the columns of the matrix $\mathbf{V}$ corresponding to the largest eigenvalues of covariance $\mathbf{C}_x$ and $\mathbf{E}_2$ is formed by the remaining eigenvectors.

Independent component analysis is a signal processing technique that allows for recovering independent random processes from their unknown linear combinations [13]. In particular, standard noiseless instantaneous ICA allows the decomposition of a random process $\mathbf{x}(t)$ as $\mathbf{x}(t)=\mathbf{A}\mathbf{s}(t)$, where $\mathbf{A}\in\mathbb{R}^{p\times p}$ is the mixing operator, $\mathbf{s}(t)\in\mathbb{R}^p$ is the source stream. The classical hypotheses on the involved quantities are that the mixing operator is full-rank and that the source signals are statistically independent at any time, at most one Gaussian source being allowed. Neural ICA consists in training an artificial neural network so that the network output signals become as statistically independent as possible. This problem may be cast as an optimization one, provided a suitable criterion function be defined. In the case that the source signal have same kurtosis sign, a valid criterion is defined as $4f\stackrel{\mathrm{def}}{=}\sum_{i=1}^m\kappa_i^y$, where $k_i^y\stackrel{\mathrm{def}}{=}\frac{E[y_i^4]}{E^2[y_i^2]}-3$. As the observations may always be normalized (through so-termed pre-whitening) so that $E[\mathbf{x}\mathbf{x}^T]=\mathbf{I}_p$, the mixing operator $\mathbf{A}$ may be supposed to be an orthonormal matrix without loss of generality. Therefore, the unmixing network connection matrix may be supposed $\mathbf{M}\in\mathrm{St}(p,m)$. In the present context, the statistical expectation is approximated

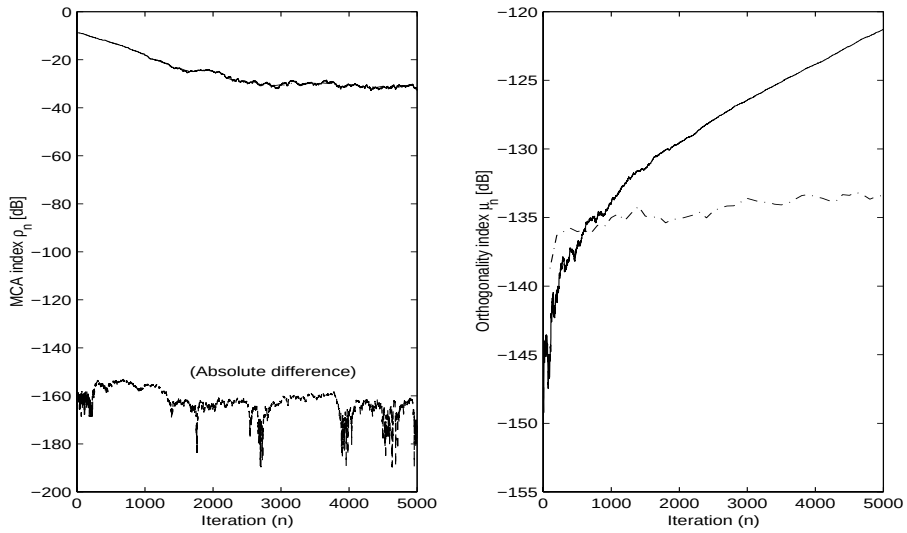

Fig. 1. Experiment on MCA. Left panel: Values of indices ρ (10) as well as their absolute difference on panel's bottom. Right-panel: Values of orthogonality index μ versus the number of iteration. Legend: Euclidean metrics in solid line; canonical metrics in dot-dashed line.

via sample mean and $\frac{\partial}{\partial \mathbf{M}} f = E[\mathbf{x}(\mathbf{y}^T)^3]$, where $(\cdot)^3$ is understood to operate component-wise. In order to define a network performance index, we consider that, at convergence, the separation product $\mathbf{P} \stackrel{\text{def}}{=} \mathbf{M}^T\mathbf{A} \in \mathbb{R}^{m\times p}$ should ideally exhibit only one entry per row (or column) different from zero, while the magnitude of non-zero values does not care. In a real-word situation, of course some residual interference should be tolerated. Therefore, a valid separation index is:

$$Q \stackrel{\text{def}}{=} \frac{1}{m}\|\mathbf{P}\mathbf{P}^T - \text{diag}(\mathbf{P}\mathbf{P}^T)\|_{\text{F}} \ . \tag{11}$$

As normally the index Q assumes very low values, it is worth normalizing to its initial value corresponding to the initial network-connection matrix $\mathbf{W} = \mathbf{I}_p$.

B. Experimental results

In the MCA experiment, it was assumed $p = 40$ and $m = 5$. The integration step was assumed as $t_\star = 0.001$. The obtained results are illustrated in the Figure 1. These results show that when the integration step assumes a relatively small value, there is no practical difference between the behavior of the Euclidean-metrics-based algorithm and the canonical-metrics-based algorithm. In fact, the learning curves shown in the left-hand panel of Figure 1 are nearly superimposed (as testified by their point-to-point absolute difference illustrated in the same panel). However, the right-hand panel illustrates the progressive loss of orthonormality of the columns of the matrix $\mathbf{M}$ due to numerical learning algorithm based on the Euclidean metrics as opposed to the good adherence to the Stiefel manifold exhibited by the canonical-metrics-based algorithm. This effect is due to the numerical difficulties related to the evaluation of matrix exponential for generic matrices. It is worth noting that, *in spite MCA is widely known to be a unstable counterpart of principal component analysis [4], when an appropriate numerical integration method of MCA learning differential equation is selected, no instabilities are observed.*

In the ICA experiment, it was assumed $p = 9$ real-world source-images and $m = 4$ component-images are sought for. The integration step was assumed as $t_\star = 0.1$. The obtained results are shown in the Figure 2. These results show that when the integration step assumes a relatively large value, there appears a noticeable difference between the behavior of the Euclidean-metrics-based algorithm and the canonical-metrics-based algorithm: The learning curves shown in the left-hand and middle panels of Figure 2 are nearly superimposed during the first phase of learning but the learning curves pertaining to the Euclidean-metrics-based algorithm tend to diverge as learning proceeds. This effect may be explained by observing the right-hand panel, which again illustrates the progressive and quick loss of orthonormality of the columns of the matrix $\mathbf{M}$ due to Euclidean-metrics-based algorithm. Also, it is worth noting that, even in the first phase of learning, the canonical-metrics-based algorithm exhibits a steadier convergence in comparison to the Euclidean-metrics-based one.

The empirical computational complexity evaluation results are reported in the Table I: They show that, in presence of a low-dimensional learning problem, the computational complexity of the canonical-metrics-based algorithm and Euclidean-metrics-based algorithm are nearly equivalent, while on a larger-scale problem the difference in computational burden becomes apparent: The canonical-metrics-based algorithm looks definitely more burdensome than the Euclidean-metrics-based one.

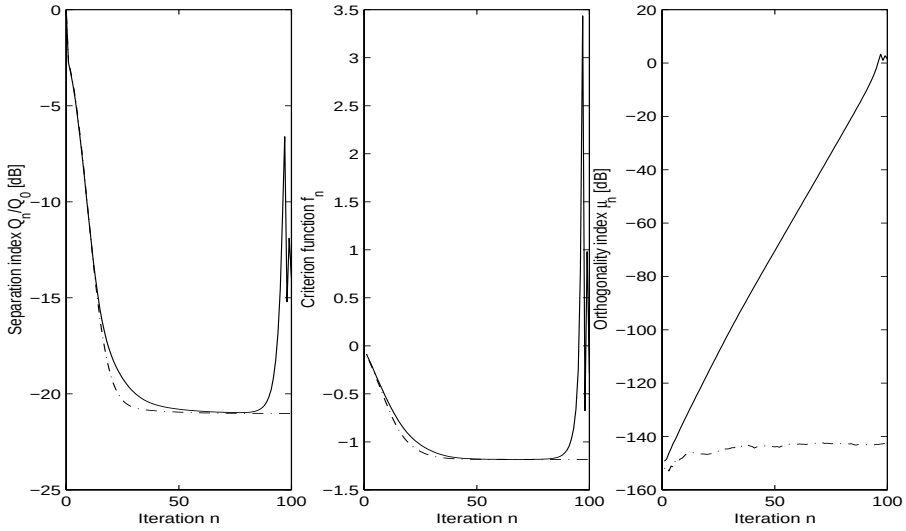

Fig. 2. Experiment on ICA. Left panel: Relative values of index Q (11). Middle panel: Course of the criterion function (sum of kurtosis) during iteration. Right panel: Values of orthogonality index μ versus the number of iteration. Legend: Euclidean metrics in solid line; canonical metrics in dot-dashed line.

TABLE I
EXPERIMENTS ON MCA AND ICA. AVERAGE FLOPS AND CPU-TIME COUNT (IN SEC.S).

ALGORITHM (APPLIC.)	AVE. FLOPS	ELAPSED TIME
Euclidean (MCA)	43597	4.21
Canonical (MCA)	910031	28.32
Euclidean (ICA)	2504025	20.19
Canonical (ICA)	2504198	27.34

A wide-purpose analysis of the computational complexity pertaining to the evaluation of the required matrix-exponential was given in the recent contribution [2], along with some efficient methods for its computation.

IV. CONCLUSION

The present Letter aimed at illustrating the relevance of numerical integration of learning differential equations on the Stiefel manifold. The basic geometry of the compact real Stiefel manifold was surveyed, including Riemannian gradient and geodesic pertaining to its Euclidean and canonical metrics.

The proposed learning algorithms are based on appropriate approximation of the exact Riemannian gradient learning flow that exploits geodesic arcs. As the structure of gradient and geodesic depend on the metrics that the manifold is equipped with, we illustrated the effects of the effected choices by discussing their analytical features and by presenting the results of some numerical experiments.

In particular, the experimental results allow us to conclude that, for small learning/integration stepsizes, the numerical performances of the two learning algorithms are nearly the same, therefore the Euclidean-metrics-based one is preferable being more convenient from a computational-complexity point of view. Conversely, when large learning/integration stepsize are selected, the canonical-metrics-based algorithm is to be preferred, it being reliable from the viewpoints of preservation of underlying constraints and of numerical stability.

ACKNOWLEDGMENT

The author wishes to gratefully thank N. Del Buono (Department of Mathematics, University of Bari, Italy) for the useful discussions and for kindly proofreading the first version of the present manuscript.

REFERENCES

[1] S.-I. AMARI, *Natural gradient works efficiently in learning*, Neural Computation, Vol. 10, pp. 251 – 276, 1998
[2] E. CELLEDONI AND S. FIORI, *Neural learning by geometric integration of reduced 'rigid-body' equations*, Journal of Computational and Applied Mathematics (JCAM), Vol. 172, No. 2, pp. 247 – 269, December 2004
[3] E. CELLEDONI AND A. ISERLES, *Approximating the exponential form of a Lie algebra to a Lie group*, Mathematics of Computation, Vol. 69, pp. 1457 – 1480, 2000
[4] T.-P. CHEN AND S.-I. AMARI, *Unified stabilization approach to principal and minor components extraction algorithms*, Neural Networks, Vol. 14, No. 10, pp. 1377 – 1387, 2001

[5] N. Del Buono and L. Lopez, *Runge-Kutta type methods based on Geodesics for systems of ODEs on the Stiefel manifold*, BIT – Numerical Mathematics, Vol. 41, No. 5, pp. 912 – 923, 2001
[6] A. Edelman, T.A. Arias and S.T. Smith, *The geometry of algorithms with orthogonality constraints*, SIAM Journal on Matrix Analysis and Applications, Vol. 20, No. 2, pp. 303 – 253, 1998
[7] S. Fiori, *A theory for learning by weight flow on Stiefel-Grassman manifold*, Neural Computation, Vol. 13, No. 7, pp. 1625 – 1647, July 2001
[8] S. Fiori, *A theory for learning based on rigid bodies dynamics*, IEEE Trans. on Neural Networks, Vol. 13, No. 3, pp. 521 – 531, May 2002
[9] S. Fiori and S.-i. Amari, *Editorial: Special issue on "Geometrical methods in neural networks and learning"*, Neurocomputing. Issue in preparation (2005)
[10] S. Fiori, *A minor subspace algorithm based on neural Stiefel dynamics*, International Journal of Neural Systems, Vol. 12, No. 5, pp. 339 – 350, 2002
[11] K. Fukumizu and S.-i. Amari, *Local minima and plateaus in hierarchical structures of multilayer perceptrons*, Neural Networks, Vol. 13, pp. 317 – 328, 2000
[12] G.H. Golub and C.F. Van Loan, *Matrix computations*, Second edition, Baltimore, MD: John Hopkins University Press, 1989
[13] T.-W. Lee, *Independent Component Analysis - Theory and Practice*, Kluwer Academic Publisher, 1998
[14] X. Liu, A. Srivastava and K. Gallivan, *Optimal linear representation of images for object recognition*, IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 26, No. 5, pp. 662 – 666, May 2004
[15] C. Moler and C. Van Loan, *Nineteen dubious ways to compute the exponential of a matrix, 25 years later*, SIAM Review, Vol. 45, pp. 3 – 49, 2003
[16] E. Oja, *Principal components, minor components, and linear neural networks*, Neural Networks, Vol. 5, pp. 927 – 935, 1992
[17] W.-Y. Yan, U. Helmke and J.B. Moore, *Global analysis of Oja's flow for neural networks*, IEEE Trans. on Neural Networks, Vol. 5, No. 5, pp, 674 – 683, Sept. 1994