16th Mediterranean Conference on Control and Automation
Congress Centre, Ajaccio, France
June 25-27, 2008

An Efficient Kullback-Leibler Optimization Algorithm for Probabilistic Control Design

Miguel Barão* João M. Lemos†‡

Abstract

This paper addresses the problem of iterative optimization of the Kullback-Leibler (KL) divergence on discrete (finite) probability spaces. Traditionally, the problem is formulated in the constrained optimization framework and is tackled by gradient like methods. Here, it is shown that performing the KL optimization in a Riemannian space equipped with the Fisher metric provides three major advantages over the standard methods: 1. The Fisher metric turns the original constrained optimization into an unconstrained optimization problem; 2. The optimization using a Fisher metric behaves asymptotically as a Newton method and shows very fast convergence near the optimum; 3. The Fisher metric is an intrinsic property of the space of probability distributions and allows a formally correct interpretation of a (natural) gradient as the steepest-descent method. Simulation results are presented.

1. Introduction

The problem considered in this paper is motivated by the design of probabilistic controllers [4, 5, 7] for controlled Markov chains, that imply the optimization of the Kullback-Leibler divergence. The optimization literature (*e.g.* [6]) refers many minimization algorithms that apply to smooth cost functions. Most methods are based on a cost function descent towards a local minimum. They provide, at each iteration, a direction computed from the cost functions local properties, usually the gradient and, sometimes, the hessian. Methods based on the gradient alone are usually slow, while methods using the second order information, such as quasi-Newton and conjugate gradient methods, yield faster convergence rates. Quasi-Newton methods are also called variable-metric methods since the Hessian can be seen as a metric in the parameter space. This "metric" is, however, a property of the cost function and not a characteristic of the underlying parameter space. Amari suggests [1, 2], instead, the use of the Fisher metric to compute the natural gradient in probability spaces and to proceed in the resulting direction.

In the present paper, Amari's natural gradient method is applied to the optimization of the Kullback-Leibler divergence on discrete (finite) probability spaces. The main results can be divided into two distinct domains. First, properties derived from the underlying space alone show that the Fisher metrics smoothly modifies the gradient direction, so that it flows within the feasible region and, therefore, constraints never become active. This property is an intrinsic property of the space of discrete probability functions and does not depend on the cost function to be optimized. If, for some reason, some constraints become active, the method behaves as the gradient projection method. The second result is more specific to the Kullback-Leibler divergence. It is shown that, since the Fisher information approaches the Hessian of the K-L divergence $D(p\|q)$ when p is close to q, the natural gradient can be seen as a quasi-Newon method for this specific cost function. With this approach the computational requirements are minimal: only marginally larger than the standard gradient; constant in time and space; and using only additions and multiplications.

The paper is organized as follows. Section 2 introduces the theoretical material used to treat the discrete probability distributions as points in a Riemannian space. Section 3 presents the definition of natural gradient and gives a numerically robust and efficient way to compute the natural gradient. Section 4 shows the stopping condition used. Section 5 analyzes some properties of the natural gradient in the current framework, namely

*M. Barão is with INESC-ID Lisboa and Informatics Department, Évora University, Portugal. mjsb@ramses.inesc-id.pt.

†J. M. Lemos is with INESC-ID Lisboa and Department of Electrical Engineering, Instituto Superior Técnico/UTL, Portugal. jlml@inesc-id.pt.

‡Supported by DynaMo - Dynamical modeling, control and optimization of metabolic networks PTDC/EEA-ACR/69530/2006.

978-1-4244-2505-1/08/$20.00 ©2008 IEEE

that the parameters evolution is (naturally) bounded by the constraints. Section 6 presents some simulations and section 7 draws conclusions.

2. Preliminaries

Let $\mathscr{P}$ denote the set of probability mass functions (p.m.f.) $p(x)$, where $x \in \{0,1,\ldots,N\}$. The p.m.f. $p(x)$ can be represented in many ways. One such way is to define the parameters as $\theta^i \stackrel{\text{def}}{=} \Pr\{X = i\} = p(i)$, for $i = 0,1,\ldots,N$. Since probabilities add to one, the $N+1$ parameters θ^i are not independent. Selecting N parameters from those gives a representation of $p(x)$ where all parameters are independent. This parametrization defines a *coordinate system* of $\mathscr{P}$ where θ^i are the coordinates. Let the N parameters $(\theta^1,\ldots,\theta^N)$ be the coordinates, while the remaining parameter θ^0 is automatically determined by $\theta^0 \stackrel{\text{def}}{=} 1 - \sum_{i=1}^{N} \theta^i$.

The probabilities are constrained to positive real numbers and thus $\theta^i > 0$, for $i = 0,1,\ldots,N$. The constraint on θ^0 can also be written in terms of the independent coordinates alone by $\sum_{i=1}^{N} \theta^i < 1$.

The region in the parameter space that satisfies all constraints is called the *feasible region*. It is a convex open set in $\mathbb{R}^N$. It has, however, a richer structure than $\mathbb{R}^N$, a feature captured by the metric tensor of a Riemannian space that tells how independent the dimensions are. It has been suggested [3] that the Fisher information matrix should be used as the metric tensor when dealing with probabilities.

2.1. The Fisher information metric

The Fisher information matrix is a symmetric positive (semi-)definite matrix $\mathbf{G}$, whose components are determined by

$$g_{ij}(\theta) \stackrel{\text{def}}{=} E_p\Big[\frac{\partial \log p(x)}{\partial \theta^i}\frac{\partial \log p(x)}{\partial \theta^j}\Big]. \tag{1}$$

The matrix $\mathbf{G}$ is not constant over the parameter space. The components depend on the particular point where the expectation is taken and the coordinate system used.

In the problem under consideration, the components of the Fisher information matrix specialize to

$$g_{ij}(\theta) = \frac{1}{1-\sum_{k=1}^{N}\theta^k} + \frac{\delta_{ij}}{\theta^i}, \tag{2}$$

where δ is the Kronecker delta function. The corresponding Fisher information matrix $\mathbf{G}$ is given by

$$\mathbf{G} = \frac{1}{1-\sum_{k=1}^{N}\theta^k}\begin{bmatrix} 1 & \cdots & 1 \\ \vdots & & \vdots \\ 1 & \cdots & 1 \end{bmatrix} + \begin{bmatrix} 1/\theta^1 & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & 1/\theta^N \end{bmatrix}. \tag{3}$$

It is a nonsingular matrix in the feasible region, but diverges as one or more probabilities approach zero.

2.2. Differentials and gradients

Given a function $F(p)$ defined on the space $\mathscr{P}$, the coordinate system θ allows a new function $F(\theta)$ to be defined. The function that gives the rate of change of $F(\theta)$ in an arbitrary direction v is the *differential* of F, denoted by $\mathrm{d}F(v)$. It is a linear operator on the linear space where v lives. If v is represented as a column matrix $\mathbf{v}$, then $\mathrm{d}F$ can be represented as the row matrix of partial derivatives of F, such that $\mathrm{d}F(v) = \begin{bmatrix}\frac{\partial F}{\partial \theta^1} & \cdots & \frac{\partial F}{\partial \theta^N}\end{bmatrix}\mathbf{v}$. The direction of greatest increase of F at some point θ is given by the *gradient vector* ∇F defined such that the following identity holds:

$$\langle \nabla F, v\rangle = \mathrm{d}F(v), \quad \forall v \neq 0. \tag{4}$$

In standard euclidean spaces, the inner product $\langle\cdot,\cdot\rangle$ is simply the dot product, and (4) yields, in matrix notation, $\nabla F = \begin{bmatrix}\frac{\partial F}{\partial \theta^1} & \cdots & \frac{\partial F}{\partial \theta^N}\end{bmatrix}^T$. In riemannian spaces, however, the inner product is defined by the metric tensor g_{ij} which, in matrix notation, reads $\langle v, w\rangle = \mathbf{v}^T\mathbf{G}\mathbf{w}$.

The gradient defined in (4) with respect to the riemannian inner product is called the *natural gradient*, and denoted by $\tilde{\nabla}F$. It is given in matrix notation by

$$\tilde{\nabla}F = \mathbf{G}^{-1}\begin{bmatrix}\frac{\partial F}{\partial \theta^1} & \cdots & \frac{\partial F}{\partial \theta^N}\end{bmatrix}^T. \tag{5}$$

It coincides with the standard gradient only for euclidean spaces, where the metric $\mathbf{G}$ is the identity matrix.

2.3. The steepest descent method

Optimization of a p.m.f. $p(x)$ can be formulated as a constrained optimization problem in $\mathbb{R}^N$, where the coordinates are confined to the feasible region. One optimization method is the steepest descent method. It states that parameters should follow the gradient flow – or its negative – on the parameter space. Ideally, this means that $\dot{\theta} = -\nabla F(\theta)$, where $\dot{\theta}$ is a velocity vector denoting the rate of change of θ. In practice, the discretized version

$$\theta_{[k+1]} = \theta_{[k]} - \eta_{[k]}\nabla F(\theta) \tag{6}$$

is used, where η is the discretization step size.

In euclidean spaces the steepest descent direction is given by ∇F, but in riemannian spaces the steepest descent direction is given by the natural gradient $\tilde{\nabla}F$ defined in (5). Thus, in riemannian spaces, the steepest descent method becomes

$$\theta_{[k+1]} = \theta_{[k]} - \eta_{[k]}\mathbf{G}^{-1}\nabla F, \tag{7}$$

where $\mathbf{G}^{-1}$ and ∇F are taken at $\theta_{[k]}$.

3. Fast natural gradient computation

The natural gradient defined in (5) requires an inversion of the metric $\mathbf{G}$. For discrete p.m.f the metric given in (3) is badly conditioned and numerical problems arise quite frequently. To overcome this limitations, the matrix $\mathbf{G}$ is written as $\mathbf{G} = \mathbf{A} + \mathbf{b}c\mathbf{b}^T$, where

$$\mathbf{A} = \begin{bmatrix} 1/\theta^1 & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & 1/\theta^N \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}, \qquad (8)$$

and $c = 1/(1 - \sum_{i=1}^{N} \theta^i)$. Using the Woodbury identity, the inverse $\mathbf{G}^{-1}$ can be written as

$$\mathbf{G}^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{b}(\mathbf{b}^T\mathbf{A}^{-1}\mathbf{b} + c^{-1})^{-1}\mathbf{b}^T\mathbf{A}^{-1}, \quad (9)$$

which can be further simplified to

$$\mathbf{G}^{-1} = \begin{bmatrix} \theta^1 & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \theta^N \end{bmatrix} - \begin{bmatrix} \theta^1 \\ \vdots \\ \theta^N \end{bmatrix} \begin{bmatrix} \theta^1 & \cdots & \theta^N \end{bmatrix}. \qquad (10)$$

As can be seen, the inverse metric $\mathbf{G}^{-1}$ can be exactly computed without any inversions. Even scalar inversions are avoided. This allows a numerically robust implementation of the natural gradient method described in section 2.3.

If the parameters θ are in column matrix form, then the natural gradient $\tilde{\nabla}F$ can be computed from ∇F by

$$\tilde{\nabla}F = \theta \circ \nabla F - \theta \left(\theta \cdot \nabla F\right), \qquad (11)$$

where $\circ$ denotes the Hadamard (a.k.a. Shur or element-wise) product and $\cdot$ denotes the dot product. Thus, it is not required to explicitly build the whole matrix $\mathbf{G}^{-1}$ to compute the product $\mathbf{G}^{-1}\nabla F$.

Noting that the dot product can be obtained by summing together the components of $\theta \circ \nabla F$, then (11) only requires $2N$ multiplications and $2N-1$ additions. It scales linearly with the problem size and, therefore, is computationally time efficient. Memory requirements are also minimal. Space for vectors θ, ∇F and $\tilde{\nabla}F$ is all that is necessary. This amounts to $3N$ floating point numbers, scales linearly with the problem size, and therefore is also computationally space efficient.

4. Stopping criteria

The iterative procedure requires a stopping criteria. One possibility is to compare the gradient vector components to a precision threshold ε, but this method has two problems: it does not take into account the geometric nature of the underlying space; and, due to curvature of F, parameters can be more sensible in some directions than in others. To tackle this issues, the natural gradient norm is used instead.

The riemannian norm is taken from the Fisher metric by $\|\mathbf{v}\|_g \stackrel{\text{def}}{=} \mathbf{v}^T\mathbf{G}\mathbf{v}$. When this norm is applied to the natural gradient, it becomes

$$\|\tilde{\nabla}F\|_g = \tilde{\nabla}F^T\mathbf{G}\tilde{\nabla}F = \nabla F^T\mathbf{G}^{-1}\nabla F. \qquad (12)$$

A more computational efficient way to compute it is available, that does not make explicit use of either $\mathbf{G}$ or $\mathbf{G}^{-1}$ and requires only a single dot product between two previously available vectors:

$$\|\tilde{\nabla}F\|_g = \nabla F^T\tilde{\nabla}F. \qquad (13)$$

Finally, the norm is compared against a threshold ε to decide when to stop. Equation (13) can also be interpreted as

$$\|\tilde{\nabla}F\|_g = \mathrm{d}F(\tilde{\nabla}F), \qquad (14)$$

i.e., the greatest rate of change of F, from the riemannian point of view. Thus, the iterative procedure stops when the greatest (riemannian) rate of change of F is below ε.

5. Properties

The natural gradient exhibits a remarkable property that is not provided by the euclidean gradient. It can be shown that the natural gradient flow is bounded to the admissible region for any admissible initial condition, *i.e.*, it intrinsically satisfies the constraints

$$\theta^i > 0, \qquad \sum_{i=1}^{N} \theta^i < 1. \qquad (15)$$

To prove this assertion, define the matrices

$$\Lambda = \begin{bmatrix} \theta^1 & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \theta^N \end{bmatrix}, \qquad \sqrt{\theta} = \begin{bmatrix} \sqrt{\theta^1} \\ \vdots \\ \sqrt{\theta^N} \end{bmatrix}, \qquad (16)$$

and let $\sqrt{\Lambda}$ denote the unique positive-definite matrix such that $\Lambda = \sqrt{\Lambda}\sqrt{\Lambda}$. Then, since $\theta = \sqrt{\Lambda}\sqrt{\theta}$, equation (10) can be factorized into

$$\mathbf{G}^{-1} = \sqrt{\Lambda}\left(\mathbf{I} - \|\sqrt{\theta}\|^2 \frac{\sqrt{\theta}}{\|\sqrt{\theta}\|}\frac{\sqrt{\theta}^T}{\|\sqrt{\theta}\|}\right)\sqrt{\Lambda}^T. \qquad (17)$$

Furthermore,

$$\|\sqrt{\theta}\|^2 = \sqrt{\theta}^T\sqrt{\theta} = \sum_{i=1}^{N} \theta^i = 1 - \theta^0. \qquad (18)$$

This fact suggests that it is possible to factorize (17) as a product of matrices resembling projection transformations[1]. Defining

$$\mathbf{H}_0 \stackrel{\text{def}}{=} \mathbf{I} - (1-\theta^0)\frac{\sqrt{\theta}}{\|\sqrt{\theta}\|}\frac{\sqrt{\theta}^T}{\|\sqrt{\theta}\|}, \tag{19}$$

$$\mathbf{H}_i \stackrel{\text{def}}{=} \mathbf{I} - (1-\theta^i)\mathbf{e}_i\mathbf{e}_i^T, \quad i = 1,\ldots,N, \tag{20}$$

where $\mathbf{e}_i$ denotes the i-th standard basis. Then, equation (17) can be written in either of the following forms:

$$\mathbf{G}^{-1} = \left(\sqrt{\Lambda}\right)^{-1}\mathbf{H}_N\cdots\mathbf{H}_2\mathbf{H}_1\mathbf{H}_0\left(\sqrt{\Lambda}\right), \tag{21}$$

$$\mathbf{G}^{-1} = \left(\sqrt{\Lambda}\right)\mathbf{H}_0\mathbf{H}_1\mathbf{H}_2\cdots\mathbf{H}_N\left(\sqrt{\Lambda}\right)^{-1}. \tag{22}$$

Equations (21) and (22) allow a better understanding of the metrics role in the natural gradient. First, the matrix $\sqrt{\Lambda}$ is performing a change of basis vectors at each point. In the new basis, the transformed vectors suffer a rescaling given by the product $\mathbf{H}_0\cdots\mathbf{H}_N$. Each one of the factors produce an incomplete projection onto the subspace defined by the respective constraint. For instance, if $\theta^1 = 0$, then

$$\mathbf{H}_1 = \begin{bmatrix} 0 & & & \\ & 1 & & \\ & & \ddots & \\ & & & 1 \end{bmatrix} \tag{23}$$

is a projection into the subspace defined by $\theta^1 = 0$. Otherwise, if $\theta^1 \neq 0$, the projection is incomplete and the resulting vector will have the first component scaled by θ^1, leaving the other components unchanged. The same goes for each one of the factors from $\mathbf{H}_2$ up to $\mathbf{H}_N$. The factor $\mathbf{H}_0$ is different, however. It scales by θ^0 along the direction $\sqrt{\theta}$. Although, it may seem different, the scaling role of $\mathbf{H}_0$ is similar to the others, since it produces a projection onto the "subspace" (strictly speaking it is not a linear subspace as it does not contain the origin) defined by $\theta^0 = 0$, or equivalently, to $\sum_{i=1}^N \theta^i = 1$, which in the new coordinates is the unit sphere $\|\sqrt{\theta}\| = 1$ (see (18)). Basically, the the new coordinates allow a scaling to be performed independently on each of the $N+1$ constraints, *i.e.*, constraints intersect at 90°.

To see that the natural gradient (continuous) flow does not violate the constraints, it suffices to check that the constraints define invariant subspaces. Thus, since the continuous flow can not cross these invariant subspaces, it is confined to the feasible region.

Assume for instance that $\theta^1 = 0$. Then, rewriting (21) as $\mathbf{G}^{-1} = \mathbf{H}_1\mathbf{H}_N\cdots\mathbf{H}_2\left(\sqrt{\Lambda}\right)^{-1}\mathbf{H}_0\left(\sqrt{\Lambda}\right)$, it is clear that $\mathbf{H}_1$ cancels any possible change in θ^1. The constraints on the remaining parameters θ^i, $i > 1$, work the same way. Here, use was made of the property that, since all matrices except $\mathbf{H}_0$ are diagonal, they can be reordered arbitrarily (with exception of $\mathbf{H}_0$).

To check the invariance on the constraint $\theta^0 = 0$, or equivalently, $\sum_{i=1}^N \theta^i = 1$, equation (22) is used in the form $\mathbf{G}^{-1} = \left(\sqrt{\Lambda}\right)\mathbf{H}_0\left(\sqrt{\Lambda}\right)^{-1}\mathbf{H}_N\cdots\mathbf{H}_2\mathbf{H}_1$. Now, the natural gradient flow given by $\dot{\theta} = -\mathbf{G}^{-1}\nabla F$, when started with the constraint $\theta^0 = 0$ active, will stay active implying $\sum_{i=1}^N \dot{\theta}^i = 0$. It suffices to prove that

$$\begin{bmatrix} 1 & \cdots & 1 \end{bmatrix}\dot{\theta} = \begin{bmatrix} 1 & \cdots & 1 \end{bmatrix}\mathbf{G}^{-1}\nabla F = 0. \tag{24}$$

Collapsing $\mathbf{H}_N\cdots\mathbf{H}_1\nabla F$ into a single vector $\mathbf{v}$, yields

$$\begin{aligned} \begin{bmatrix} 1 & \cdots & 1 \end{bmatrix}\sqrt{\Lambda}\mathbf{H}_0\sqrt{\Lambda}^{-1}\mathbf{v} &= \\ &= \begin{bmatrix} 1 & \cdots & 1 \end{bmatrix}\sqrt{\Lambda}(\mathbf{I} - \sqrt{\theta}\sqrt{\theta}^T)\sqrt{\Lambda}^{-1}\mathbf{v} \\ &= \begin{bmatrix} 1 & \cdots & 1 \end{bmatrix}\left(\mathbf{I} - \theta\begin{bmatrix} 1 & \cdots & 1 \end{bmatrix}\right)\mathbf{v} \\ &= \left(1 - \sum_{i=1}^N \theta^i\right)\begin{bmatrix} 1 & \cdots & 1 \end{bmatrix}\mathbf{v} \\ &= 0. \end{aligned} \tag{25}$$

Thus, if the constraint $\sum_{i=1}^N \theta^i = 1$ is active, then the flow emanating from θ will maintain this property.

From the above observations, it can be concluded that the feasible region has an invariant boundary. Thus, under conditions of existence and unicity of solutions of ODEs, the gradient flow is continuous and consequently bounded to the feasible region.

6. Simulation results

To illustrate the application of the natural gradient method, three examples are presented. In the first, a p.m.f. $p(x)$ is sought to minimize the K-L divergence $D(p\|q)$ to a given target p.m.f. $q(x)$. This is an artificial problem as the answer $p(x) = q(x)$ is already known. Its interest stems from being a problem where both gradient and natural gradient methods can be easily compared. In the second example, the same K-L divergence $D(p\|q)$ is minimized, but p is not completely free. It is assumed that p is a joint p.m.f. generated by the chain rule of probabilities $p_{X,Y}(x,y) = p_X(x)\,p_{Y|X}(y|x)$, where only $p_X(x)$ is free. This problem is related in part to closed-loop optimization common in probabilistic control systems. The third example considers a closed loop probabilistic control problem and shows the viability of the algorithm in large scale optimizations.

Example 1 *Using the parametrization introduced in*

[1] A projection matrix $\mathbf{P} = \mathbf{I} - \mathbf{u}\mathbf{u}^T$, where $\mathbf{u}$ are unitary vectors, eliminates the components of a vector v along the direction $\mathbf{u}$.

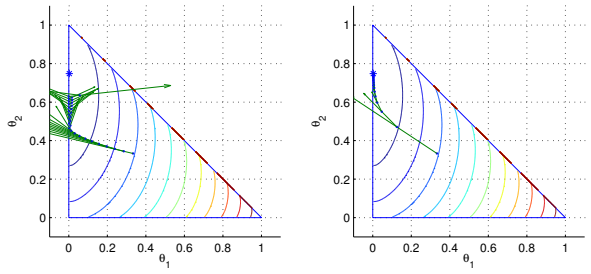

Figure 1. Optimization of $D(p\|q)$ using the standard and natural gradient methods.

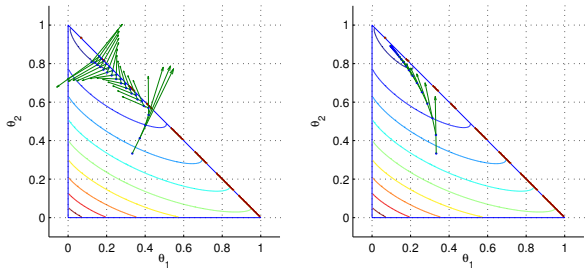

Figure 2. Optimization of $D(p\|q)$ where p is jointly distributed $p(x,y) = p_\theta(x)p(y|x)$.

the previous sections, the p.m.f $p(x)$ is, in matrix notation, $\mathbf{p} = \begin{bmatrix}\theta^0 & \theta^1 & \cdots & \theta^N\end{bmatrix}^T$, where $(\theta^1,\ldots,\theta^N)$ are free parameters and $\theta^0 = 1 - \sum_{i=1}^N \theta^i$ is a dependent parameter. The gradient of D is given by

$$\nabla D = \begin{bmatrix} -1 & 1 & & \mathbf{0} \\ \vdots & & \ddots & \\ -1 & \mathbf{0} & & 1 \end{bmatrix} (\log \mathbf{p} - \log \mathbf{q}), \qquad (26)$$

the logarithms being taken elementwise over the probability vectors $\mathbf{p}$ and $\mathbf{q}$.

Let $\mathbf{q} = [0.2494; 0.0025; 0.7481]$ be the target p.m.f. and $\mathbf{p} = [\frac{1}{3}; \frac{1}{3}; \frac{1}{3}]$ the initial guess. Use of the standard gradient method with fixed step size yields erratic behavior on the parameter space. Figure 1 shows, on the left, a simulation using a constant step-size of $\eta = 0.01$. This example shows two known weakness of the standard gradient: when the curvature differs greatly in different directions a small step is required and slow convergence is obtained; if η is large then instability occurs. Figure 1 shows, on the right, the same problem solved by the natural gradient method with a bigger step size $\eta = 0.18$. The convergence is very fast in this case. The justification is due to the Fisher information matrix being also the Hessian of the K-L divergence at $p = q$. Thus, the natural gradient behaves as a quasi-Newton, variable metric, method as it approaches the optimum.

Example 2 *Another example is the optimization of $D(p\|q)$, where p is generated by*

$$p_{X,Y}(x,y) = p_X(x)\, p_{Y|X}(y|x). \qquad (27)$$

Here, $p_{Y|X}(y|x)$ is assumed to be previously specified and fixed, while $p_X(x)$ is free. The optimization is performed on the later distribution. For the sake of compactness, the subscripts in p_X, $p_{Y|X}$ and $p_{X,Y}$ will be dropped in the foregoing expressions whenever they are understood from context. The gradient of $D(p\|q)$ with respect to parameters θ which specify $p(x)$ has components given by

$$\frac{\partial D}{\partial \theta^i} = \sum_{x=0}^{N}\sum_{y=0}^{M} p(y|x) \log \frac{p(y|x)p(x)}{q(x,y)} \frac{\partial p(x)}{\partial \theta^i}. \qquad (28)$$

In matrix notation, let $\mathbf{p}_X$ denote the column matrix of probabilities $p(x)$, $\mathbf{P}_{Y|X}$ the $M \times N$ transition matrix, and $\mathbf{P}_{Y,X}$ and $\mathbf{Q}_{Y,X}$ the joint probability matrices of $p(x,y)$ and $q(x,y)$, respectively. Subscript positions of X and Y are used to index rows and columns, so $\mathbf{P}_{Y,X} = (\mathbf{P}_{X,Y})^T$. Using this notation, the gradient ∇D can be computed by

$$\mathbf{P}_{Y,X} = \mathbf{P}_{Y|X} \circ (\mathbf{p}_X \begin{bmatrix}1 & \cdots & 1\end{bmatrix})^T, \qquad (29)$$

$$\mathbf{T}_{Y,X} = \mathbf{p}_{Y|X} \circ (\log \mathbf{P}_{Y,X} - \log \mathbf{Q}_{Y,X}), \qquad (30)$$

$$\nabla D = \begin{bmatrix} -1 & 1 & & \mathbf{0} \\ \vdots & & \ddots & \\ -1 & \mathbf{0} & & 1 \end{bmatrix} \mathbf{T}_{X,Y} \begin{bmatrix}1 \\ \vdots \\ 1\end{bmatrix}, \qquad (31)$$

where the logarithms are taken elementwise over the matrices $\mathbf{P}_{Y,X}$ and $\mathbf{Q}_{Y,X}$.

The $\mathbf{G}^{-1}$ matrix is the same as in the previous sections, since it is computed for $p(x)$ alone. Then, the natural gradient is given, as in equation (11).

A low dimensional problem was setup for illustrative purposes where $\mathbf{P}_{X,Y}$ and $\mathbf{Q}_{X,Y}$ are 3×2 matrices. Figure 2 shows the parameter evolutions in both methods (standard and natural gradient).

Here, the cost $D\big(p(x,y)\|q(x,y)\big)$ does not converge to zero since there is no distribution $p(x)$ satisfying $p(x)p(y|x) = q(x,y)$, as was the case in example 1.

Example 3 *This example addresses a more complex optimization problem. We seek to minimize, as before, the K-L divergence $D(p\|q)$. The difference, here, is the highly dimensional nature of p and q and the intricate structure of p. Assume that the distribution p results*

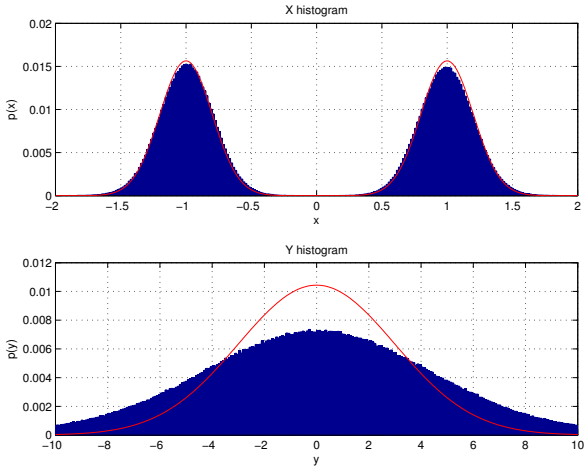

Figure 3. Target $q(x_{0:T}, y_{1:T})$ (continuous curve) and histogram obtained in closed loop simulation using the optimal controller.

from the Markov chain

$$p(x_{0:T}, y_{1:T}) = p(x_0) \prod_{i=1}^{T} p(y_i|x_{i-1}) p(x_i|x_{i-1}, y_i), \quad (32)$$

where $p(y_i|x_{i-1})$ is free for all $i = 1, \ldots, T$, while the remaining transition probabilities are fixed. This kind of formulation arises in design of probabilistic control systems, where $p(y_i|x_{i-1})$ is the controller generating a signal Y given the observation X. The remaining fixed transition probabilities $p(x_i|x_{i-1}, y_i)$ are a model of the process for which the control system is designed. This process has an external input Y, generated by the controller, and an internal state X.

Without diving into the details of the solution, it briefly consists in the division of the control horizon T into a sum of costs. This can be achieved using K-L decomposition properties and dynamic programming. The resulting algorithm requires an iterative approach for each instant i in the horizon. This is where the natural gradient enters into action: optimizing $p(y_i|x_{i-1})$, T times, one for each i in the control horizon.

For illustrative purposes, a first order unstable system was sampled by an 8 bit A/D converter, for which a "controller" $p(y_i|x_{i-1})$ is to be found such that the closed loop behavior is as close as possible (in K-L sense) to a desired $q(x_{0:T}, y_{1:T})$.

With the above framework, a target p.m.f q was defined such that variables X and Y have probabilities as indicated in figure 3, where the stationary marginal distribution $q(x)$ was selected to be bimodal. In this example, the initial optimization by the natural gradient method discovered a 256×256 transition matrix in 9 iterations, starting from a uniform distribution and using adaptive step-size ensuring a decreasing cost (not line search). The following optimizations for the remaining control horizon took successively less iterations. This amounts to a grand total of $256 \times 256 \times 200 = 13107200$ parameters being discovered in under 13 seconds on a today's desktop computer.

7. Conclusions

This paper presented an application of the natural gradient learning method to discover discrete probability mass functions. It was shown that the natural gradient can be obtained in a computationally very efficient and numerically robust way. It was also shown that the natural gradient is strongly tied to the geometrical structure of the underlying space and, as a consequence, its flow is always feasible. It is, for this reason, much easier to apply than constrained optimization methods.

Furthermore, when the optimizing function is the K-L divergence, the natural gradient method becomes a quasi-Newton method and is therefore much faster than the standard gradient near the solution.

Three examples were presented with increasing level of complexity. The first two, of low dimension, illustrate the trajectories followed in comparsion to the standard gradient, while the third showed that the method works well for highly dimensional problems.

References

[1] S. Amari and S.C. Douglas. Why natural gradient? In *Acoustics, Speech, and Signal Processing, 1998. ICASSP '98. Proceedings of the 1998 IEEE International Conference on*, volume 2, pages 1213 – 1216 vol.2, 12-15 May 1998.

[2] Shun-Ichi Amari. Natural gradient works efficiently in learning. *Neural Computation*, 10(2):251–276, 1998.

[3] Shun-Ichi Amari and Hiroshi Nagaoka. *Methods of Information Theory*, volume 191 of *Translations of Mathematical Monographs*. American Mathematical Society, 2000.

[4] Miguel Barão and João M. Lemos. Parametric probabilistic control: An information-geometric approach. In *Mediterranean Conference on Control and Automation*, Lisbon, Portugal, 2002.

[5] M Kárný. Towards fully probabilistic control design. *Automatica*, 32(12):1719–1722, Dec. 1996.

[6] David G. Luenberger. *Introduction to Linear and Nonlinear Programming, Second Edition*. Addison-Wesley, 1989.

[7] E. Nováková and M. Kárný. Fully probabilistic control design for markov chains. In *European Control Conference*, 1997.