An Iterative Nonlinear Gaussianization Algorithm for Image Simulation and Synthesis

Jen-Jen Lin* Naoki Saito† Richard A. Levine‡

submitted to Journal of Computational and Graphical Statistics

June 12, 2001

Abstract

We propose an Iterative Nonlinear Gaussianization Algorithm (INGA) which seeks a nonlinear map from a set of dependent random variables to independent Gaussian random variables. A direct motivation of INGA is to extend principal component analysis (PCA), which transforms a set of correlated random variables into uncorrelated (independent up to second order) random variables, and Independent Component Analysis (ICA), which linearly transforms random variables into variates that are "as independent as possible." A modified INGA is then proposed to nonlinearly transform ICA coefficients into statistically independent components. To quantify the performance of each algorithm: PCA, ICA, INGA, and modified INGA, we study the Edgeworth Kullback-Leibler Distance (EKLD) which serves to measure the "distance" between two distributions in multi-dimensions. Several examples are presented to demonstrate the superior performance of INGA (and its modified version) in situations where PCA and ICA poorly simulate the images of interest.

Keywords: dimension reduction, data compression, principal component analysis (PCA), Independent Component Analysis (ICA), Edgeworth Kullback-Leibler distance, image analysis.

*Corresponding author: Department of Applied Statistics, Ming Chuan University, Taipei, Taiwan; email: jjlin@mcu.edu.tw

†Department of Mathematics, University of California, Davis, CA 95616, USA; email: saito@math.ucdavis.edu

‡Department of Statistics, University of California, Davis, CA 95616, USA; email: levine@wald.ucdavis.edu

1 INTRODUCTION

Given n samples of a random vector $\boldsymbol{X}$ of p dimensions ($n > p$), we are interested in resampling/simulating the dependent components of $\boldsymbol{X}$ in such a way that the resampled/simulated data and the observed samples obey the same p-variate distribution. In the case of one dimension, the classical bootstrap (Efron and Tibshirani, 1993) "random resampling with replacement" has been a popular choice. Another direction is to use copulas (Nelsen, 1998), which join multivariate distributions to their one-dimensional marginal distributions. In the high dimensional case, however, both methods have limitations. Only linear models have been used in bootstrap analyses (Efron and Tibshirani, 1993), and the copula-based algorithms are hard to apply for problems of dimension greater than three (Nelsen, 1998).

Separately from these resampling ideas, the method of Independent Component Analysis (ICA) (Comon, 1994) has become very popular, in particular, in the neural network community (Bell and Sejnowski, 1995). The objective of ICA is to find a linear transformation so that the given random vector can be represented "as independent as possible." ICA methodologies were originally motivated by the problems of blind source separation and blind deconvolution (see e.g., Cardoso, 1998, for a comprehensive review).

In this paper, we propose a statistical algorithm called the Iterative Nonlinear Gaussianization Algorithm (INGA) — an extension to principal components analysis (PCA). While PCA merely transforms a set of correlated random variables into a set of uncorrelated random variables, INGA nonlinearly transforms them to the standard multivariate Gaussian variables in an attempt to minimize the statistical dependence among the transformed coordinates, at a similar computational cost to PCA. The difference between INGA and ICA lies in two aspects, although both seek statistically-independent coordinate systems. First, INGA seeks a *nonlinear* transform whereas ICA seeks a linear one. Second, the motivation of INGA is really *resampling* and *simulation* rather than blind source separation and blind deconvolution.

There are two parts to INGA: the forward and backward processes. The forward process iteratively transforms a given set of dependent random variables by: 1) applying PCA to decorrelate the random components; 2) matching their one-dimensional marginal distributions to the standard Gaussian distribution $\mathcal{N}(0,1)$; and 3) transforming the components linearly to improve the joint Gaussianity. At each iteration, the closeness of the resulting transformed variables to the standard

joint Gaussian variates $\mathcal{N}(0, I)$ is checked by statistical tests evaluated with the empirical P-value of certain distance measures such as the squared Mahalanobis distance, multivariate skewness, and kurtosis. The backward process generates new samples which presumably obey the same distribution as the original samples at our disposal. Section 3, 4 and 5 discuss these in details.

Evaluation of the resamples or simulated data is often done subjectively by visually comparing them with the original samples. This subjective evaluation is particularly dominating in the area of texture modeling and simulation (e.g., (Portilla and Simoncelli, 1999), (Zhu et al., 1998)). In order to *objectively* evaluate the similarity (or difference) between the original samples and the generated resamples, we use the Kullback-Leibler distance (Lin et al., 2001), which is summarized in Section 6.

Our motivation to develop INGA lies in image modeling and simulation from the given samples without knowledge of the underlying true probability model. However, the most difficult problem in image modeling is the "curse of dimensionality." In particular, a reliable estimate of probability density functions (pdf's) of high dimensional data from a finite (and potentially small) number of samples are hard to obtain in general. Also, INGA itself is a relatively expensive algorithm (i.e., a constant multiple of the cost of PCA). Therefore, before actually applying INGA to image samples, we need to efficiently compress images (dimension reduction). Either PCA or wavelet transforms can be used for this dimension reduction. However, it is important to realize that PCA achieves only decorrelation and the wavelet transforms in general produce highly statistically dependent coefficients across different scales (Portilla and Simoncelli, 1999), (Saito et al., 2000). On the other hand, ICA provides a less statistically dependent coordinate system to be successful for samples that come from a linear combination of statistically independent sources (Comon, 1994), (Hyvärinen, 1999).

The organization of this paper is as follows. In Section 2, we use a few simple synthetic datasets to motivate the development of INGA by showing the failure of the simple simulation methods using PCA and ICA. Then, Section 3 describes the basic structure of our proposed algorithm, INGA, which consists of the forward and backward (or analysis and synthesis) processes. This is followed by the detail analysis of the nonlinear Gaussianization step in the forward process in Section 4. Section 5 proposes statistical tests to check the closeness of the resulting transformed variables in each iteration in the INGA forward process to the standard joint Gaussian variables $\mathcal{N}(0, I)$. These

tests are important to check the convergence of the iteration. Section 6 describes our strategy to objectively quantify the similarity between the original samples and the simulated samples using the sample distributions of the EKLDs between the original samples and simulated samples. We then present several simulation results with INGA and compare its performance with those based on PCA and ICA in Section 7. Since INGA is an iterative algorithm, the speed of the convergence relies on how good the initial coordinate system is (or how close the initial coordinate system is to the statistically independent one). Therefore, for a certain stochastic process, much faster convergence is obtained if we start with the ICA coordinates rather than the PCA coordinates. In Section 8, we propose a modified version of INGA, which replaces the initialization step using PCA by ICA, with the hope to speed up the convergence.

We note that a preliminary version of this paper was presented at the Second International Workshop on Independent Component Analysis and Blind Signal Separation that was held in June 2000 at Helsinki, Finland (Lin et al., 2000).

2 MOTIVATION OF INGA THROUGH SYNTHETIC EXAMPLES

In this section, we first formally define PCA and ICA, and then describe the shortfalls of PCA and ICA using a few synthetic datasets and motivate the development of INGA.

2.1 PCA and ICA

The Principal Component Analysis (PCA), also known as transformation through a Karhunen-Loève basis (KLB), provides a decorrelated coordinate system. Watanabe (1965) showed that the PCA basis is characterized by minimizing the entropy of the energy distributions over its coordinates:

$$B_{\mathrm{PCA}} = \arg\min_{B \in \mathcal{L}} \mathcal{C}_{\mathrm{PCA}}(B \mid \mathcal{T}),$$

where $\mathcal{L}$ is a set of possible bases under consideration, and

$$\mathcal{C}_{\mathrm{PCA}}(B \mid \mathcal{T}) = \sum_{i=1}^{n} h(\hat{\gamma}[B]).$$

The entropy function h is defined by

$$h(\gamma[B]) = -\sum_{i=1}^{n} \gamma_i[B] \log \gamma_i[B],$$

where $\gamma_i[B]$ is a normalized energy of the ith coordinate of B. In practice, we use the sample estimate $\hat{\gamma}_i[B]$ of $\gamma_i[B]$ from a training dataset $\mathcal{T}$.

To lift the PCA from its limitation to the second order statistics, Comon (1994) proposed the so-called Independent Component Analysis (ICA); see also Jutten and Herault (1991), Cardoso (1998). Bell and Sejnowski (1995) discussed the closely related concept of "information maximization" and its neural network implementation. Given a training dataset $\mathcal{T}$, the ICA tries to find an invertible linear transformation that minimizes the statistical dependence among its coordinates. In our notation, ICA can be written as

$$B_{\mathrm{ICA}} = \arg\min_{B \in \mathcal{L}} \mathcal{C}_{\mathrm{ICA}}(B \,|\, \mathcal{T}),$$

where $\mathcal{C}_{\mathrm{ICA}}(B \,|\, \mathcal{T})$ is a measure of statistical dependence, and often the mutual information or its approximation by the higher-order moments are used. In fact,

$$\mathcal{C}_{ICA}(B \,|\, \mathcal{T}) = I(\boldsymbol{Y}) = -H(\boldsymbol{Y}) + \sum_{i=1}^{n} H(Y_i)$$

where $\boldsymbol{Y} = B^{-1}\boldsymbol{X}$ is a random vector represented in the basis B, $I(\boldsymbol{Y})$ is the mutual information of the random vector in the transformed coordinates, and $H(\boldsymbol{Y}) = -\int f_{\boldsymbol{Y}}(\boldsymbol{y}) \log f_{\boldsymbol{Y}}(\boldsymbol{y}) \mathrm{d}\boldsymbol{y}$ is its differential entropy. See Cover and Thomas (1991) for more details on the differential entropy and mutual information. In this paper, we use a particular version of ICA algorithm called *fastICA* developed by Hyvärinen (1999), which is a computationally highly efficient method for estimating the independent components from given multidimensional signals. He used a fixed-point iteration scheme to provide a general-purpose data analysis method that can be used both in an exploratory fashion and for estimation of independent components (source). The term "source" means the original unknown independent component that can be represented by a random variable in general. The fastICA package for MATLAB®[1] is available at the web site `http://www.cis.hut.fi/projects/ica/fastica/`.

[1] MATLAB is a registered trademark of The MathWorks, Inc., 3 Apple Hill Drive, Natick, MA 01760-2098, USA.

2.2 Motivating Examples: Cigar, Boot, and Square Datasets

The development of INGA was motivated by the failure of the simple simulation methods based on PCA and ICA. In the following three examples, we, using PCA and ICA, simulate the data as follows. Let $\boldsymbol{X}$ be a random vector in the standard basis, and let $\boldsymbol{Y} = B^{-1}\boldsymbol{X}$ be a random vector relative to the coordinate system B, where B is the PCA basis or the ICA basis. Let $F_{N,i}(y)$ be an empirical cumulative distribution function (ecdf) of the ith coordinate Y_i and let $F_i(y)$ be an interpolated version of $F_{N,i}$ so that the inverse exists. If $U \sim \text{unif}(0,1)$, then $F_i^{-1}(U)$ obeys F_i, i.e., Y_i and $F_i^{-1}(U)$ share the same ecdf F_i since $\Pr\{F_i^{-1}(U) < y\} = \Pr\{U < F_i(y)\} = F_i(y)$. Once we sample all the coordinates to get $\tilde{\boldsymbol{Y}} = (F_1^{-1}(U_1), \ldots, F_n^{-1}(U_n))'$ where $U_1, \ldots, U_n$ are different realizations of the unif$(0,1)$ distribution, then we can simulate typical data by the inverse transform $\tilde{\boldsymbol{X}} = B\tilde{\boldsymbol{Y}}$. We call this simple simulation method mCDF (the marginal cdf-based) method in this paper. See Ripley (1987) for simulation methods other than the inversion method.

We demonstrate the failure of the mCDF methods using the following relatively simple examples.

2.2.1 Cigar Data

The "cigar" data, two obliquely overlapping distributions, illustrate a situation where PCA fails to capture the correct coordinate system. The cigar dataset is obtained as follows.

1. Let $\boldsymbol{X} = (X_1, X_2)'$ be a two-dimensional random sample where X_1 and X_2 are drawn independently from unif$(0,1)$ and $\mathcal{N}(0,1)$ distributions respectively.

2. Form the cigar-shaped data. One leaf of it is obtained by

$$\boldsymbol{Y} = \begin{pmatrix} Y_1 \\ Y_2 \end{pmatrix} = \begin{pmatrix} \cos(\pi/12) & -\sin(\pi/12) \\ \sin(\pi/12) & \cos(\pi/12) \end{pmatrix} \begin{pmatrix} X_1 \\ X_2 \end{pmatrix},$$

   the other leaf is by

$$\boldsymbol{Z} = \begin{pmatrix} Z_1 \\ Z_2 \end{pmatrix} = \begin{pmatrix} \cos(5\pi/12) & -\sin(5\pi/12) \\ \sin(5\pi/12) & \cos(5\pi/12) \end{pmatrix} \begin{pmatrix} X_1 \\ X_2 \end{pmatrix}.$$

3. Let $\boldsymbol{W} = (W_1, W_2)'$ be a random vector that selects $\boldsymbol{Y}$ and $\boldsymbol{Z}$ with equal probability, i.e., $1/2$.

Figure 1 shows the original cigar samples and the simulations obtained by the mCDF method with PCA and ICA. Note the poor quality of the simulations. The reason is that the mCDF-PCA method cannot capture the correct basis and the mCDF-ICA method generates the coefficients marginally which cannot represent the joint information among the original coefficients.

2.2.2 Boot Data

This is an example of a nonlinear combination of two independent sources (random variables). Once we applied the sample matrix to the fastICA package in MATLAB, the output showed the information of "non-convergence". This example illustrates a situation where ICA fails to give convergent independent sources. Note that the key assumption of ICA is that the input random vectors consist of a *linear* combination of statistically independent sources while this example is a *nonlinear* combination of sources. The boot dataset is obtained by the following procedure:

1. Let $\boldsymbol{X} = (X_1, X_2)'$ be a two-dimensional random sample where both X_1 and X_2 are independently drawn from a unif$(0, 1)$ distribution.

2. Form the boot-shaped data $\boldsymbol{Y} = (Y_1, Y_2)'$ by the following nonlinear combinations of X_1 and X_2:

$$\begin{aligned} Y_1 &= \frac{\mathrm{e}^{-3X_2}}{X_1} + 1, \\ Y_2 &= \frac{\mathrm{e}^{X_1}}{X_2^2} + 1. \end{aligned}$$

Figure 2 shows the original boot samples and simulated data obtained by PCA. Note that the simulated boot is a poor representation of the original data since PCA cannot handle a nonlinear structure like this dataset.

2.2.3 Square Data

The square data illustrate a situation where the ICA basis essentially provides the independent coordinate system whereas the PCA basis provides merely a decorrelated coordinate system. A random vector $\boldsymbol{X} = (X_1, X_2)'$ representing this dataset is obtained by sampling X_1 and X_2 independently from the unif$(0, 1)$ distribution.

Figure 3 shows the original square samples and the simulations obtained by the mCDF method with PCA and ICA. Note that the mCDF-PCA simulation poorly represents the square data since PCA uses the wrong basis. The mCDF-ICA simulation looks better although the angles of the upper and lower right corners deviate from the right angle. In other words, ICA could not provide the exact statistically independent basis, which is the standard basis in this case. This is due to the approximate numerical procedure from a finite number of samples.

3 ITERATIVE NONLINEAR GAUSSIANIZATION ALGORITHM

INGA extends PCA by seeking a nonlinear map of a set of dependent random variables to a set of independent Gaussian random variables. An immediate advantage of deriving independent standard Gaussian components is that we can simulate a general multivariate stochastic process by sampling univariate independent standard Gaussian variables.

There are two parts of INGA: the forward and backward processes. The forward process iteratively transforms a given set of dependent random variables; the backward process generates new samples which presumably obey the same distribution as the original samples at our disposal.

3.1 The INGA Forward Process

Let $\boldsymbol{X} = (X_1, \ldots, X_p)'$ be a dependent random (column) p-vector of interest which obeys a cumulative distribution function (cdf) $F_{\boldsymbol{X}}$. Let $\boldsymbol{x}$ be a realization (or an observed version) of this random vector. Let Φ denote the cdf of the standard p-variate Gaussian distribution $\mathcal{N}(0, I_p)$, where I_p is the $p \times p$ identity matrix. The INGA forward process consists of the following steps:

(i) `Initialize` $\boldsymbol{x}^{(0)} = \boldsymbol{x}$ and set $k = 0$.

(ii) `Apply` PCA to the images, i.e., $\boldsymbol{y} = B'\boldsymbol{x}^{(k)}$, where B is an orthogonal matrix for decorrelating $\boldsymbol{x}^{(k)}$.

(iii) `Transform` $\boldsymbol{u} = F_{\boldsymbol{Y}}(\boldsymbol{y})$ so that each of the p random variates $\boldsymbol{u} = (u_1, \ldots, u_p)'$ are uniformly distributed on the interval $(0, 1)$.

(iv) `Transform` $\boldsymbol{z} = \Phi^{-1}(\boldsymbol{u})$ so that each of the p random variates $\boldsymbol{z} = (z_1, \ldots, z_p)'$ obeys the standard Gaussian distribution $\mathcal{N}(0, 1)$.

(v) `Transform` $\tilde{\boldsymbol{z}} = A\boldsymbol{z}$ through a linear operator A such that the distribution of $\tilde{\boldsymbol{z}}$ becomes closer to the standard multivariate Gaussian distribution $\mathcal{N}(0, I_p)$.

(vi) `Check` the convergence to the multivariate Gaussian distribution. If a satisfactory convergence is not attained, then set $k \leftarrow k+1$, $\boldsymbol{x}^{(k)} = \tilde{\boldsymbol{z}}$, and go to (ii).

Note that step (iv) does not guarantee to generate the *joint* Gaussian random variates even if each z_i is *marginally* Gaussian. That is the reason why we need to apply the transformation of step (v) to ensure the random variates are at least approximately jointly Gaussian.

The operator A in Step (v) is obtained by minimizing the L_2 error between the characteristic function of $\tilde{\boldsymbol{z}}$ and that of the standard multivariate Gaussian distribution $\mathcal{N}(0, I_p)$. Section 4 describes how to compute such an operator A in detail.

The variables transformed by A may be neither truly multivariate Gaussian nor independent at a given iteration. We thus need to iterate over the $\tilde{\boldsymbol{z}}$ variates until the maximum correlation of the components of $\tilde{\boldsymbol{z}}$ (i.e., the size of the off-diagonal components of the covariance matrix of $\tilde{\boldsymbol{z}}$) becomes less than a given tolerance ϵ and the tests of multivariate Gaussianity using the Monte Carlo estimates of the multivariate skewness, kurtosis, and squared Mahalanobis distance reach satisfactory levels. Section 5 describes these validation procedures in detail. We note that we do not have a proof of convergence of this iterative nonlinear procedure. However, all the examples we have examined so far, INGA reached to the satisfactory levels of joint Gaussianity within several iterations.

3.2 INGA Backward Process

With the records of the INGA forward process (i.e., storing all the information about A and $F_{\boldsymbol{Y}}$ for each iteration), we can go backwards: starting from the INGA coefficients, we can obtain the original sample $\boldsymbol{x}$ by reversing the forward steps (ii)–(v). If we assume that the final INGA coefficients obey the standard multivariate Gaussian distribution, it is easy to generate a new set of such coefficients. We then can proceed backwards to synthesize or simulate the data in the starting coordinates.

(i) `Generate` INGA coefficients.

(ii) `Invert` all the iterations of the INGA forward process to synthesize the data in the starting coordinates.

We will, in Section 7, present several simulation examples and compare the performance of INGA with that of mCDF-PCA and mCDF-ICA.

4 ANALYSIS OF THE GAUSSIANIZATION STEP

4.1 Theoretical Analysis of the Linear Operator A

After we have the marginal Gaussian variables Z_i in step (iv) of the INGA forward process, step (v) finds a linear transform A so that $\tilde{\boldsymbol{Z}} = A\boldsymbol{Z}$ closely obeys the standard multivariate Gaussian distribution. How can we find such an A? Let $\phi(t_i) = \exp(-t_i^2/2)$ be the characteristic function of Z_i and let $f(A \mid \boldsymbol{t}) = E[\exp(\mathrm{i}\boldsymbol{t}'\tilde{\boldsymbol{Z}})] = E[\exp(\mathrm{i}\boldsymbol{t}'A\boldsymbol{Z})]$ be the characteristic function of $\tilde{\boldsymbol{Z}}$. Define a cost function measuring the L_2 error between these two characteristic functions:

$$D(A) = \left\| f(A \mid \boldsymbol{t}) - \prod_{i=1}^{p} \phi(t_i) \right\|^2. \tag{1}$$

The operator A is obtained by minimizing $D(A)$. A straightforward way to minimize (1) is to use the perturbation of $D(A)$ about A_0, the orthogonal matrix used in PCA to decorrelate the Z_i's, namely

$$A = A_0 + \delta A. \tag{2}$$

Here δA is a perturbation matrix with components δa_{ij} which are the elements obtained from the Taylor expansion of $D(A)$ about A_0,

$$\delta \boldsymbol{a} = -H^{-1}(A_0) \cdot \frac{\partial D}{\partial \boldsymbol{a}}(A_0), \tag{3}$$

where $\frac{\partial D}{\partial \boldsymbol{a}} = (\frac{\partial D}{\partial a_{11}}, \frac{\partial D}{\partial a_{12}}, \cdots, \frac{\partial D}{\partial a_{pp}})'$, and H is the $p^2 \times p^2$ Hessian matrix,

$$H = \begin{pmatrix} \frac{\partial^2 D}{\partial a_{11}\partial a_{11}} & \frac{\partial^2 D}{\partial a_{11}\partial a_{12}} & \cdots & \frac{\partial^2 D}{\partial a_{11}\partial a_{pp}} \\ \vdots & \vdots & & \vdots \\ \frac{\partial^2 D}{\partial a_{pp}\partial a_{11}} & \frac{\partial^2 D}{\partial a_{pp}\partial a_{12}} & \cdots & \frac{\partial^2 D}{\partial a_{pp}\partial a_{pp}} \end{pmatrix}.$$

Note that $\delta \boldsymbol{a} = (\delta a_{11}, \cdots, \delta a_{pp})' \in \mathbb{R}^{p^2}$ is a perturbation vector whose components δa_{ij} are the elements of δA. Although this perturbation is straightforward, one can easily see that this becomes computationally impractical even for a moderate size of p.

4.2 Numerical Minimization of $D(A)$

There is an additional challenge for numerically minimizing (1) via the perturbation in (2) and (3). Let us first rewrite $D(A)$ in (1) as

$$D(A) = \int \left| f(A \,|\, \boldsymbol{t}) - \prod_{i=1}^{p} \phi(t_i) \right|^2 \mathrm{d}t_1 \cdots \mathrm{d}t_p. \tag{4}$$

To find the minimizer A to (4), we need numerical evaluation of both (4) and (3). To perform these, we use the Gaussian quadrature method, i.e., a weighted sum of the integrand evaluated at the nodes $t_{i,j},\ i = 1, \ldots, p,\ j = 1, \ldots, J$ of $\boldsymbol{t}$, where J is the required number of nodes. This sum can be written as

$$D(A) \approx \sum_{j_1, \cdots, j_p} w_{j_1, \cdots, j_p} \left| f(A \,|\, t_{1,j_1}, \ldots, t_{p,j_p}) - \prod_{i=1}^{p} \phi(t_{i,j_i}) \right|^2. \tag{5}$$

This approximation is also used to compute the quantities in (3). In this problem, we use the standard quadrature Gauss-Chebyshev quadrature method which can be found in any textbook on numerical analysis, for example, Kincaid and Cheney (1991).

For the two-dimensional case, the required number of nodes is nine. For the general p dimensional case, $\boldsymbol{t} = (t_1, \cdots, t_p)$, we need 3^p nodes. This numerical approach is again only feasible for small to moderate p. One reason is that the limitation of an INTEGER variable representable in a computer: This is usually $(2^{31}) - 1 \approx 2.1475 \times 10^9$ which is even smaller than $3^{20} \approx 3.4868 \times 10^9$ (for $p = 20$). The other reason is the CPU time: in the case of $p = 20$ dimension, the estimated CPU time is 2.71 days (on a Pentium III 500 MHz (586 series) PC with 256MB RAM); for $p = 100$, it is out of question.

Instead of finding A by going through the 3^p nodes all at once, which we call the direct optimization process, we use a *sequential* process to find A *approximately*. This sequential process consists of the following steps:

[Step 1] Let $\boldsymbol{Z}_2 = (Z_1, Z_2)'$ be the first two components of $\boldsymbol{Z}$ obtained in the step (iv) of INGA forward process. Find the 2×2 matrix

$$A_2 = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}$$

by minimizing the cost function (5) using the direct optimization process. Then the transformed vector $\tilde{\boldsymbol{Z}}_2 = A_2\,\boldsymbol{Z}_2$ has a two-dimensional distribution with mean zero and identity covariance matrix.

Note that in order to obtain A_2 from

$$A_2 = A_2^{(0)} + \delta A_2,$$

we need to use the 2-dimensional 9-point Gaussian quadrature to evaluate all the differentials in H and $\frac{\partial D}{\partial \boldsymbol{a}}$. Here $\delta A_2 = -H^{-1}A_2^{(0)} \cdot \frac{\partial D}{\partial \boldsymbol{a}}A_2^{(0)}$ and $A_2^{(0)}$ is the PCA matrix decorrelating $\boldsymbol{Z}_2$.

[Step 2] By appending Z_3 to $\tilde{\boldsymbol{Z}}_2$, we let $\boldsymbol{Z}_3 = (\tilde{\boldsymbol{Z}}_2', Z_3)' = (\tilde{Z}_1, \tilde{Z}_2, Z_3)'$ be the next three-dimensional vector. With these new vectors, we want to find the linear transformation with the special structure

$$\tilde{A}_3 = \begin{pmatrix} I_2 & 0 \\ \boldsymbol{a}_2' & a_{33} \end{pmatrix},$$

where $\boldsymbol{a}_2' = (a_{31}, a_{32})$, so that the transformed vector

$$\tilde{\boldsymbol{Z}}_3 = \begin{pmatrix} \tilde{\boldsymbol{Z}}_2 \\ \tilde{Z}_3 \end{pmatrix} = \begin{pmatrix} I_2 & 0 \\ \boldsymbol{a}_2' & a_{33} \end{pmatrix} \begin{pmatrix} \tilde{\boldsymbol{Z}}_2 \\ Z_3 \end{pmatrix} = \tilde{A}_3\,\boldsymbol{Z}_3$$

minimizes

$$D(\tilde{A}_3) = \left\| f(\tilde{A}_3 \,|\, \boldsymbol{t}_3) - \prod_{i=1}^{3} \phi(t_i) \right\|^2,$$

where $\boldsymbol{t} = (t_1, t_2, t_3)' = (\boldsymbol{t}_2', t_3)'$. Assume we may approximate

$$f(\tilde{A}_3 \,|\, \boldsymbol{t}_3) = E\left(\mathrm{e}^{\mathrm{i}\boldsymbol{t}'\tilde{\boldsymbol{Z}}_3}\right) \approx E\left(\mathrm{e}^{\mathrm{i}\boldsymbol{t}_2'\tilde{\boldsymbol{Z}}_2}\right) E\left(\mathrm{e}^{\mathrm{i}t_3\tilde{Z}_3}\right).$$

This is the key assumption, but is justifiable because $\tilde{\boldsymbol{Z}}_3$ is eventually expected to be joint Gaussian. $D(\tilde{A}_3)$ can then be rewritten as follows:

$$D(\tilde{A}_3) \approx \mathrm{e}^{-(t_1^2+t_2^2)} \left\| E\left(\mathrm{e}^{\mathrm{i}t_3(a_{31}\tilde{Z}_1 + a_{32}\tilde{Z}_2 + a_{33}Z_3)}\right) - \mathrm{e}^{-t_3^2/2} \right\|^2. \tag{6}$$

Let

$$\tilde{D}(\boldsymbol{a}) = \left\| E\left(\mathrm{e}^{\mathrm{i}t_3(a_{31}\tilde{Z}_1 + a_{32}\tilde{Z}_2 + a_{33}Z_3)}\right) - \mathrm{e}^{-t_3^2/2} \right\|^2 \tag{7}$$

be the the norm term in (6). Then a_{31}, a_{32}, and a_{33} can be found by minimizing $D(\tilde{A}_3)$ in (6), or equivalently, minimizing $\tilde{D}(\boldsymbol{a})$ in (7).

Since only one characteristic variable t_3 is involved, we need only to find $\boldsymbol{a} = (a_{31}, a_{32}, a_{33})'$ through one-dimensional three-point Gaussian quadrature. Consider the Taylor expansion $G(\delta\boldsymbol{a})$ of $\tilde{D}(\boldsymbol{a})$ around $\boldsymbol{a}_0 = (0, 0, 1)'$:

$$G(\delta\boldsymbol{a}) = \tilde{D}(\boldsymbol{a}_0) + \sum_{i,j} \left(\frac{\partial \tilde{D}}{\partial a_{ij}}(\boldsymbol{a}_0) \right) (\delta a_{ij}) + \frac{1}{2} \sum_{i,j,k,l} \left(\frac{\partial^2 \tilde{D}}{\partial a_{ij} \partial a_{kl}}(\boldsymbol{a}_0) \right) (\delta a_{ij} \delta a_{kl}) \ .$$

Then minimizing $\tilde{D}(\boldsymbol{a})$ is carried out by minimizing $G(\delta\boldsymbol{a})$. Similarly, by (3), we have

$$\delta\boldsymbol{a} = -H^{-1}(\boldsymbol{a}_0) \cdot \frac{\partial \tilde{D}}{\partial \boldsymbol{a}}(\boldsymbol{a}_0)$$

where H is an 9×9 symmetric matrix. The updated $\boldsymbol{a}_2$ will be

$$\boldsymbol{a}_2 = (a_{31}, a_{32}, a_{33})' = \boldsymbol{a}_0 + \delta\boldsymbol{a}.$$

Combining $\boldsymbol{a}_2$ with the previous A_2, we have the 3×3 transformation matrix with the structure

$$A_3 = \begin{pmatrix} A_2 & 0 \\ \boldsymbol{a}_2' & a_{33} \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & 0 \\ a_{21} & a_{22} & 0 \\ a_{31} & a_{32} & a_{33} \end{pmatrix}.$$

[Step 3]

The following procedure, which produces the $m \times m$ transformation matrix, is repeated for $m = 4, \ldots, p$.

1. Once A_{m-1} and $\tilde{\boldsymbol{Z}}_{m-1}$ are obtained, append Z_m to $\tilde{\boldsymbol{Z}}_{m-1}$ to form $\boldsymbol{Z}_m = (\tilde{\boldsymbol{Z}}_{m-1}', Z_m)' = (\tilde{Z}_1, \cdots, \tilde{Z}_{m-1}, Z_m)'$.

2. Look for the linear transformation with the special structure $\tilde{A}_m = \begin{pmatrix} I_{m-1} & 0 \\ \boldsymbol{a}_{m-1}' & a_{mm} \end{pmatrix}$ where $\boldsymbol{a}_{m-1}' = (a_{m1}, \cdots, a_{m,m-1})$.

3. The transformed random vector $\tilde{\boldsymbol{Z}}_m = \tilde{A}_m \boldsymbol{Z}_m$ then minimizes

$$D(\tilde{A}_m) = \left\| f(\tilde{A}_m \,|\, \boldsymbol{t}_m) - \prod_{i=1}^{m} \phi(t_i) \right\|^2 , \tag{8}$$

for which we make the approximation

$$f(\tilde{A}_m \,|\, \boldsymbol{t}_m) = E\left(\mathrm{e}^{\mathrm{i}\boldsymbol{t}_m' \tilde{\boldsymbol{Z}}_m} \right) \approx E\left(\mathrm{e}^{\mathrm{i}\boldsymbol{t}_{m-1}' \tilde{\boldsymbol{Z}}_{m-1}} \right) E\left(\mathrm{e}^{\mathrm{i}t_m \tilde{Z}_m} \right),$$

where $\boldsymbol{t}_m = (t_1, \cdots, t_m)'$. The last row vector $\boldsymbol{a}'_m = (\boldsymbol{a}'_{m-1}, a_{mm})$ of $\tilde{A}_m$ in (8) can be found by minimizing

$$\tilde{D}(\boldsymbol{a}_m) = \left\| E\left(\mathrm{e}^{\mathrm{i}t_m(\boldsymbol{a}'_{m-1}\tilde{\boldsymbol{Z}}_{m-1} + a_{mm}Z_m)}\right) - \mathrm{e}^{-t_m^2/2} \right\|^2 .$$

4. Since only one characteristic variable t_m is involved, we only need to find $\boldsymbol{a}_m$ by one-dimensional 3-point Gaussian quadrature using the perturbation $\boldsymbol{a}_m = \boldsymbol{a}_{0m} + \delta\boldsymbol{a}_m$ in a similar manner as (2), where $\boldsymbol{a}_{0m} = (0, 0, \cdots, 0, 1)'$. Combining this optimal $\boldsymbol{a}_m$ with the previous A_{m-1}, we have the $m \times m$ transformation matrix with the structure

$$A_m = \begin{pmatrix} A_{m-1} & 0 \\ \boldsymbol{a}'_{m-1} & a_{mm} \end{pmatrix} .$$

Note the following relationship $\tilde{\boldsymbol{Z}}_m = \tilde{A}_m \boldsymbol{Z}_m = A_m(Z_1, \cdots, Z_m)'$. More specifically, the matrix A_m has the form:

$$A_m = \begin{pmatrix} a_{11} & a_{12} & 0 & \cdots & \cdots & 0 \\ a_{21} & a_{22} & 0 & \cdots & \cdots & 0 \\ a_{31} & a_{32} & a_{33} & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \ddots & \vdots \\ \vdots & \vdots & \vdots & & \ddots & 0 \\ a_{m1} & \cdots & \cdots & \cdots & \cdots & a_{mm} \end{pmatrix} .$$

5 TESTING INDEPENDENCE AND MULTIVARIATE GAUSSIANITY IN THE INGA FORWARD PROCESS

In step (vi) of the INGA forward process in Section 3.1, at each iteration, we diagnose independence and Gaussianity (i.e., normality) of the random variables using the multivariate skewness, multivariate kurtosis, and Mahalanobis distance measures. Recall that the multivariate skewness and kurtosis for a p-variate Gaussian distribution are zero and three respectively. We use the Mahalanobis distance to study the distance from the distribution of $\tilde{\boldsymbol{z}}$ of step (v) to the multivariate Gaussian distribution. We seek a distance measurement close to zero. Evaluation of these measures is performed through the empirical estimates of the skewness, kurtosis, and Mahalanobis distance to these known values under the null Gaussian distribution hypothesis.

In this section, we will first define these three measures, multivariate skewness, multivariate kurtosis, and the Mahalanobis distance, briefly describing how each fits into INGA. We then illustrate the independence and Gaussian goodness of fit tests to evaluate the INGA forward process. We note that INGA is modular in that if the user wishes to use alternative diagnostics measures, we merely substitute or add these alternative routines in INGA step (vi) of Section 3.1. We are thus not limited to the study of skewness, kurtosis, and Mahalanobis distance, though we choose to study them here.

5.1 Multivariate Skewness and Kurtosis

Malkovich and Afifi (1973) proposed the multivariate skewness and multivariate kurtosis to be the primary tools for checking whether given samples obey a multivariate Gaussian distribution. The definitions of multivariate skewness and multivariate kurtosis are as follows.

1. The distribution of a random vector $\boldsymbol{X} \in \mathbb{R}^p$ is said to have multivariate skewness if for some nonzero vector $\boldsymbol{c} \in \mathbb{R}^p$
$$\beta_1(\boldsymbol{c}) = \frac{[E\{(\boldsymbol{c}'\boldsymbol{X} - \boldsymbol{c}'E(\boldsymbol{X}))^3\}]^2}{[\mathrm{Var}(\boldsymbol{c}'\boldsymbol{X})]^3} > 0$$
2. The distribution has multivariate kurtosis if for some $\boldsymbol{c}$
$$[\beta_2(\boldsymbol{c})]^2 = \left[\frac{E\{(\boldsymbol{c}'\boldsymbol{X} - \boldsymbol{c}'E(\boldsymbol{X}))^4\}}{[\mathrm{Var}(\boldsymbol{c}'\boldsymbol{X})]^2}\right]^2 > 9.$$

Let us use the following notation: $Z_j = \boldsymbol{c}'\boldsymbol{X}_j$, and $\bar{Z} = (1/n)\sum_{j=1}^n Z_j$. Then the sample estimates of multivariate skewness and kurtosis are as follows.

1. sample multivariate skewness:
$$b_1(\boldsymbol{c}) = n\frac{[\sum_{j=1}^n (Z_j - \bar{Z})^3]^2}{[\sum_{j=1}^n (Z_j - \bar{Z})^2]^3} \tag{9}$$
2. sample multivariate kurtosis:
$$b_2(\boldsymbol{c}) = \frac{n\sum_{j=1}^n (Z_j - \bar{Z})^4}{[\sum_{j=1}^n (Z_j - \bar{Z})^2]^2}. \tag{10}$$

Using Roy's *union-intersection* principle (Roy, 1957), the hypothesis of no skewness and no kurtosis of the distribution of $\boldsymbol{X}$ is accepted if

$$b_1^* = \max_{\boldsymbol{c}\neq\boldsymbol{0}} b_1(\boldsymbol{c}) \leq K_{b_1},$$

$$(b_2^*)^2 = \max_{\boldsymbol{c}\neq\mathbf{0}}[b_2(\boldsymbol{c}) - K]^2 \leq K_{b_2},$$

where $K \to 3$ as $n \to \infty$ (Fisher, 1929). The constants K_{b_1}, K_{b_2} are obtained by: 1) generating synthetic Gaussian samples whose mean and variance are the same as the sample mean and variance of the data $\{Z_j\}$; and 2) compute the sample skewness and kurtosis of those samples using the formulas the formulas (9) and (10) to set K_{b_1} and K_{b_2}, respectively. Then, we also have

$$(b_2^*)^2 = \max\{[b_2^*(\min) - K]^2, [b_2^*(\max) - K]^2\}$$

where

$$b_2^*(\min) = \min_{\boldsymbol{c}\neq\mathbf{0}} b_2(\boldsymbol{c}) \quad \text{and} \quad b_2^*(\max) = \max_{\boldsymbol{c}\neq\mathbf{0}} b_2(\boldsymbol{c}).$$

5.2 Squared Mahalanobis Distance

The squared Mahalanobis distance is commonly used to examine multivariate Gaussianity (Johnson and Wichern, 1998). Let $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ be independent observations from a p-dimensional multivariate Gaussian population with mean $\boldsymbol{\mu}$ and a finite (nonsingular) covariance matrix Σ. The squared Mahalanobis distance is defined by

$$d_i^2 = (\boldsymbol{x}_i - \bar{\boldsymbol{X}})' S^{-1} (\boldsymbol{x}_i - \bar{\boldsymbol{X}}), \qquad i = 1, \ldots, n \tag{11}$$

where $\bar{\boldsymbol{X}}$ and S^2 are the sample estimates of $\boldsymbol{\mu}$ and Σ, respectively. Then the random variable D with observations $d_1^2, \ldots, d_n^2$ obeys the χ^2-distribution with p degrees of freedom (Johnson and Wichern, 1998).

5.3 Evaluating the Forward INGA Process

In this section, we will evaluate the forward INGA process using the cigar, boot and square datasets. All the datasets were already introduced in Section 2. We will show how to use the skewness, kurtosis, and Mahalanobis distance measures to evaluate step (v) of INGA. During the forward INGA process, we use Monte Carlo tests to compare the empirical estimates of the multivariate skewness, multivariate kurtosis, and Mahalanobis distance to the corresponding known values (zero, three, and zero respectively) under the null Gaussian distribution hypothesis. We choose to terminate the forward INGA iterations when the empirical p-value is greater than 0.8, at which we are reasonably satisfied with the multivariate Gaussian assumption.

Figure 4 presents the empirical p-values for the cigar, boot and square data. From these plots, we may conclude that the INGA forward process successfully transforms the original random variables to independent standard Gaussian variables within four to five iterations.

6 MODEL VALIDATION MEASURES

In essence, INGA builds a probability model of the underlying unknown distribution from the available data. This model is then used to simulate new data samples that we wish to behave as the original samples. This type of simulation can be used for a variety of important tasks such as diagnostics and classification. It is then of critical importance to know the accuracy of our model in a quantitative manner. An important question then is how to judge the closeness of these simulated samples to the originals. In information theory, the Kullback-Leibler distance is a useful validation measure to evaluate the similarity (or difference) between original samples and simulated samples. The Kullback-Leibler (KL) distance (Kullback, 1959), a measure of the difference between two probability models, is defined as

$$J(f,g) = \int f(\boldsymbol{u}) \log \frac{f(\boldsymbol{u})}{g(\boldsymbol{u})} \mathrm{d}\boldsymbol{u}, \tag{12}$$

where f and g are two pdf's of interest.

Under some regularity conditions, the Gaussian approximation to f and g, denoted by $\hat{f}$ and $\hat{g}$, may be obtained through the Edgeworth expansion (Barndorff-Nielsen and Cox, 1989), (Kendall et al., 1987). The Edgeworth KL distance (EKLD) expansion of f and g is

$$J_E(\hat{f},\hat{g}) = \int \hat{f}(\boldsymbol{u}) \log \frac{\hat{f}(\boldsymbol{u})}{\hat{g}(\boldsymbol{u})} \mathrm{d}\boldsymbol{u}.$$

Of course, our main goal is to derive computationally feasible model validation measures. We may use the results from Lin et al. (2001) to compute EKLD in practice. Let $\boldsymbol{X}$ and $\boldsymbol{Y}$ be random vectors whose pdf's are f and g, respectively. Under some regularity conditions on the estimators $\hat{f}$ and $\hat{g}$, Lin et al. (2001) showed that

$$J_E(\hat{f},\hat{g}) = J(\phi_{\boldsymbol{X}}, \phi_{\boldsymbol{Y}}) + O(n^{-1}). \tag{13}$$

In this equation, $\phi_{\boldsymbol{X}}$ is the multivariate Gaussian whose mean and covariance are the same as those

of $\boldsymbol{X}$ obeying the pdf f (similarly for $\phi_{\boldsymbol{Y}}$), and then we have $J(\phi_{\boldsymbol{X}}, \phi_{\boldsymbol{Y}}) = (a+b+c-1)/2$, where

$$\begin{aligned} a &= \log \frac{\det(\tilde{K})}{\det(K)}, \\ b &= \sum_i \frac{(\kappa^{i,i})^2 + (\kappa^i - \tilde{\kappa}^i)^2}{(\tilde{\kappa}^{i,i})^2}, \\ c &= \sum_{i \neq j} \frac{2\tilde{\kappa}^{i,j}}{\tilde{\kappa}^{i,i}\tilde{\kappa}^{j,j}} \{\kappa^{i,i}\kappa^{i,j} + (\kappa^i - \tilde{\kappa}^i)(\kappa^j - \tilde{\kappa}^j)\}, \end{aligned}$$

where $K = [\kappa^{i,j}]$ and $\tilde{K} = [\kappa^{i,j}]$ denote the covariance matrices of $\boldsymbol{X}$ and $\boldsymbol{Y}$; that is, $\kappa^{i,j} = E(X_i - \kappa^i)(X_j - \kappa^j)$; $\kappa^i = E(X_i)$, $\tilde{\kappa}^{i,j} = E(Y_i - \tilde{\kappa}^i)(Y_j - \tilde{\kappa}^j)$, and $\tilde{\kappa}^i = E(Y_i)$. The central limit theorem guarantees that the sampling distribution of the sample mean (called sample mean distribution) tends to a Gaussian distribution. Therefore, we can apply this model validation measure to the sample mean distribution of the simulated samples. In the INGA simulation, $\boldsymbol{X}$ and $\boldsymbol{Y}$ correspond to the original and the INGA simulated samples, respectively.

Our model validation procedure may be summarized as follows.

1. `Simulate` the stochastic process of interest with INGA and generate 100 datasets each of which contains 100 simulated samples.

2. `Compute` the EKLD in (13) between the original distribution and the simulated distribution using the sample means and the sample covariance matrices computed from the original samples and a simulated dataset containing 100 simulated samples. Perform this EKLD computation for each dataset. This results in 100 EKLDs.

3. `Display` the distribution of these 100 EKLDs by its boxplot. The smaller EKLD is, the closer (or more similar) the simulated samples are to the originals.

7 SIMULATIONS

We now demonstrate the simulation capabilities and limitations of INGA using several examples ranging from simple synthetic stochastic processes to a real-life image database. We will also compare the simulated data by INGA with those by the mCDF method on the PCA and ICA coordinates, which we explained in Section 2. To study the quantitative difference between the original samples and these simulated samples, we will consider the sampling distribution of the

EKLD between the simulated samples and the originals. Namely, we generate 100 simulated samples and evaluate the EKLD on each simulated sample. Typically, if the value of EKLD is close to zero with large probability, then we may conclude that the corresponding simulation method provides realizations from the same probability model as the original data. In order to obtain Monte Carlo or empirical estimates of the EKLD sampling distribution and the probability EKLD is in a neighborhood of zero, we repeat this simulation process 1000 times.

7.1 Simulation Procedures

In this section, we describe the INGA simulation method. We already explained the mCDF method based on PCA or ICA in Section 2 where we used the inversion method in each coordinate. Now, the simulation by INGA works as follows:

Step 1 (optional) Compress the original data using PCA (or wavelets or any other sparsifying transformation).

Step 2 Apply INGA to generate new samples relative to the coordinates used in Step 1.

Step 3 (optional) Rotate the new samples back to the original (i.e., standard) coordinate system.

7.2 Analysis of INGA Simulations of the Cigar, Boot, and Square Data

Here, we perform the INGA simulation on the cigar, boot, and square data, and compare the simulation results with those of mCDF-PCA and mCDF-ICA by Monte Carlo estimates of the EKLD sampling distribution.

7.2.1 Cigar Data

The top two rows of Figure 5 show the original cigar samples and simulations obtained by INGA, mCDF-PCA, and mCDF-ICA. The INGA simulation looks better than the others. Moreover, the third row of Figure 5 shows the boxplots of the EKLD of the INGA, PCA, and ICA. Note that INGA outperforms PCA and ICA.

7.2.2 Boot Data

In a similar manner, the top row of Figure 6 shows the case of the boot data. Again, the INGA simulation looks better than those of PCA. Moreover, the second row of Figure 6 shows the boxplots of the EKLD of the INGA and mCDF-PCA. Note that INGA outperforms mCDF-PCA. Recall that the ICA algorithm cannot be applied in this example. If we proceed with the ICA algorithm, the program will not converge since the boot data is a nonlinear mixture of two independent sources which conflicts with the assumption of ICA algorithm.

7.2.3 Square Data

The top two rows of Figure 7 show the case of the square data. The INGA simulation looks better again than the others. The third row of Figure 7 shows the boxplots of the EKLD of the INGA, PCA, and ICA. Figure 12 column 2 shows the PCA and ICA bases for the square dataset. As one can easily see, ICA provides the "most" independent coordinate system that coincides well with the truly independent system. PCA provides only a decorrelated coordinate system, which is still far from being independent. On the other hand, as far as INGA is concerned, there is no reason to prevent us from using ICA instead of PCA as the initialization step to start the algorithm. These observations motivate us to modify INGA even though the present simulations are not bad. We will introduce the modified INGA in Section 8.

7.3 Analysis of the Eye Data

The data set of eye images consists of digitized pictures of right eyes from 143 people. We randomly select 100 eyes from this set of 143 eyes as the training set. Each eye image consists of 144 (12×12) pixels. In the terminology of a data matrix, we have $n = 100$ samples of $p = 144$ dimensional vectors in the eye data set. In order to meet the requirements of INGA (sample size is greater than the dimension), before performing the forward INGA process, we need to compress the data. In this experiment, we use PCA to reduce the dimension down to 25 from 144. The top two rows of Figure 8 show the simulations by INGA, mCDF-PCA, and mCDF-ICA. It is really hard to compare the simulations visually in this example. The third row of Figure 8 shows the boxplot of the EKLDs of INGA, mCDF-PCA, and mCDF-ICA. We may conclude that the simulation by INGA is superior to the others. The simulations by mCDF-PCA and mCDF-ICA are essentially the same.

7.4 Ramp Data

INGA does not uniformly outperform ICA. The ramp process, a very simple stochastic process, illustrates such a situation. This process is expressed as:

$$X(t) = t - H(t - \tau), \quad 0 \leq t \leq 1,$$

where $H(\cdot)$ is the Heaviside step function, and τ obeys the uniform distribution unif$(0, 1)$. For more detailed discussion about the ramp process including an interesting historical remarks, see Saito et al. (2000), Buckheit and Donoho (1996), Donoho et al. (1998), and Cardoso and Donoho (1999). A p-dimensional discrete version of this process can be described as follows. Let ω be uniformly and randomly chosen from the set of integers $\{1, \ldots, p\}$. Then the discrete ramp process $\boldsymbol{X} = (X_1, \ldots, X_p)' \in \mathbb{R}^p$ can now be generated by the following formula:

$$X_j = \begin{cases} (j-1)/(p-1) & \text{if } 1 \leq j < \omega \\ -1 + (j-1)/(p-1) & \text{if } \omega \leq j \leq p. \end{cases}$$

For each realization of this discrete ramp process, a very small white Gaussian noise ($\sigma^2 = 10^{-14}$) is added for the numerical stability. The ramp signal dataset is then obtained by randomly sampling the 32-dimensional ramp process 1000 times. The top two rows of Figure 9 show the original ramp signal and the simulations obtained by INGA, mCDF-PCA, and mCDF-ICA. The mCDF-ICA simulation appears better than the others. The third row of Figure 9 denotes the boxplots of the EKLD of INGA, mCDF-PCA, and mCDF-ICA. mCDF-PCA is significantly worse than mCDF-ICA and INGA. mCDF-ICA outperforms INGA though the difference is not statistically significant.

We note that INGA projects the original data on the PCA coordinate system which is a set of sinusoidal functions—an inefficient basis for the ramp process (see also Donoho et al. 1998). The ICA basis for the ramp example (Figure 10) is more appropriate for compression and modeling. In this example, unlike the other cases in the previous subsections, the PCA initialization in INGA turns out to be defective and make the subsequent INGA iterative process ineffective. There is no reason to prevent us from using ICA instead of PCA in the initialization step. These observations motivate us to modify INGA, which we will describe in the next section.

8 MODIFIED INGA

As we saw in the previous section, INGA did not produce a satisfactory simulation in the case of the ramp data. Furthermore, the mCDF-ICA simulation was better than INGA. In general, ICA provides the less statistically dependent coordinate system than PCA does. Therefore, it is natural to replace PCA in step (ii) of the INGA forward process by ICA, which we call the modified INGA.

[**Step 1**] Project the original data to the ICA coordinate system to obtain the corresponding coefficients.

[**Step 2**] Proceed with the Nonlinear Gaussianization (NG) process (step (iii) – step (vi) in the original INGA forward process) on the coefficients obtained in Step 1.

To contrast the original and modified INGA, we first examine the simulations of the ramp dataset. Figure 11 shows these simulations. It is clear to see that: 1) the modified INGA produces better simulated samples than the mCDF simulation does using the PCA or ICA coefficients; 2) the simulation using the ICA coefficients (the middle column of Figure 11) is better than that using the PCA coefficients (the right column of Figure 11); and 3) the simulation by the modified INGA is better than that of the original INGA.

Next, we use the square example from Section 2.2 to further study the differences between the original and modified INGA. The ICA basis (the top row, the 2nd column of Figure 12) is close to the truly independent coordinate system, but the PCA basis (the bottom row, the 2nd column of Figure 12) is not. Therefore, there is no wonder why the mCDF-ICA simulation (the top row, the 3rd column of Figure 12) is clearly superior to that with PCA (the bottom row, the 3rd column of Figure 12).

Applying NG on the ICA coefficients (i.e., the modified INGA) lead to a very good simulation even with only one iteration of the NG process (the top row, the 4th column of Figure 12). On the contrary, the original INGA (i.e., with PCA) could not produce a satisfactory simulation by one iteration (the bottom row, the 3rd column of Figure 12), and it required several iterations to improve the simulation (the bottom row, the 4th column of Figure 12). Figure 13 shows the improvement of the simulations by the original INGA as the number of iterations increases.

To summarize our findings, we may conclude that INGA actually consists of two parts: one is the PCA/ICA to obtain an appropriate initial coordinate system and the NG process to make it less

statistically dependent. The function of the PCA/ICA is to seek a best coordinate system under the assumption that the given samples are derived from a multivariate Gaussian distribution (PCA) or a linear combination of statistically independent random variables. It is therefore expected that PCA/ICA may produce reasonably good simulations for those cases that meet these assumptions. However, there is no guarantee that the coefficients of the samples represented in the PCA/ICA basis are statistically independent in general; in fact they are often dependent. The NG process thus plays a critical role in transforming these coefficients to statistically independent components, although we do not have a proof of the convergence of this iterative process. By generating variates from these independent components and reversing INGA, we have developed an algorithm that is generally successful in statistical image simulations.

Acknowledgment

This work was partially supported by grants from NSF-EPA DMS-99-78321 (JJL, NS, RAL), NSF DMS-99-73032 (NS), and ONR YIP N00014-00-1-046 (NS).

References

Barndorff-Nielsen, O. E. and Cox, D. R. (1989). *Asymptotic Techniques for Use in Statistics.* Chapman and Hall, London.

Bell, A. J. and Sejnowski, T. J. (1995). An information-maximization approach to blind separation and blind deconvolution. *Neural Computation*, 7:1129–1159.

Buckheit, J. B. and Donoho, D. L. (1996). Time-frequency tilings which best expose the non-Gaussian behavior of a stochastic process. In *Proc. International Symposium on Time-Frequency and Time-Scale Analysis*, pages 1–4. IEEE. Jun. 18–21, 1996, Paris, France.

Cardoso, J.-F. (1998). Blind signal separation: statistical principles. *Proc. IEEE*, 90:2009–2025.

Cardoso, J.-F. and Donoho, D. L. (1999). Some experiments on independent component analysis of non-Gaussian processes. In *Proc. IEEE Signal Processing International Workshop on Higher Order Statistics, Ceasarea, Israel*, pages 74–77.

Comon, P. (1994). Independent component analysis, a new concept? *Signal Processing*, 36:287–314.

Cover, T. M. and Thomas, J. A. (1991). *Elements of Information Theory*. Wiley Interscience, New York.

Donoho, D. L., Vetterli, M., DeVore, R. A., and Daubechies, I. (1998). Data compression and harmonic analysis. *IEEE Trans. Inform. Theory*, 44(6):2435–2476. Invited paper.

Efron, B. and Tibshirani, R. J. (1993). *An Introduction to the Bootstrap*. Chapman & Hall, Inc., New York.

Fisher, R. A. (1929). The moments of the distribution for normal samples of measures of departures from normality. *Proc. London Math. Soc.*, Ser. 2, 30:199–238.

Hyvärinen, A. (1999). Fast and robust fixed-point algorithms for independent component analysis. *IEEE Trans. Neural Networks*, 10(3):626–634.

Johnson, R. A. and Wichern, D. W. (1998). *Applied Multivariate Statistics*. Prentice-Hall, Englewood Cliffs, N.J.

Jutten, C. and Herault, J. (1991). Blind separation of sources, Part I: An adaptive algorithm based on neuromimatic architecture. *Signal Processing*, 24(1):1–10.

Kendall, M., Stuart, A., and Ord, J. K. (1987). *Kendall's Advanced Theory of Statistics, Vol. I: Distribution Theory*. Oxford Univ. Press, 5th edition.

Kincaid, D. and Cheney, W. (1991). *Numerical Analysis*, Wadsworth & Brooks/Cole, Pacific Grove, CA.

Kullback, S. (1959). *Information Theory and Statistics*. John Wiley & Sons, New York. Republished by Dover in 1997.

Lin, J.-J., Saito, N., and Levine, R. A. (2000). An iterative nonlinear Gaussianization algorithm for resampling dependent components. In Pajunen, P. and Karhunen, J., editors, *Proc. 2nd International Workshop on Independent Component Analysis and Blind Signal Separation*, pages 245–250. IEEE. June 19–22, 2000, Helsinki, Finland.

Lin, J.-J., Saito, N., and Levine, R. A. (2001). On approximation of the Kullback-Leibler information by Edgeworth expansion. Technical report, Division of Statistics, Univ. California, Davis. submitted to J. Roy. Statist. Soc. Ser. B.

Malkovich, J. F. and Afifi, A. A. (1973). On tests for multivariate normality. *J. Amer. Statist. Assoc.*, 69:730–737.

Nelsen, R. B. (1998). *An Introduction to Copulas*, volume 139 of *Lecture Notes in Statistics*. Springer-Verlag, New York.

Portilla, J. and Simoncelli, E. P. (1999). Texture modeling and synthesis using joint statistics of complex wavelet coefficients. IEEE Workshop on Statistical and Computational Theories of Vision, Fort Collins, CO.

Ripley, B. D. (1987). *Stochastic Simulation*. John Wiley & Sons, Inc., New York.

Roy, S. N. (1957). *Some Aspects of Multivariate Analysis*. John Wiley & Sons, Inc., New York.

Saito, N., Larson, B. M., and Bénichou, B. (2000). Sparsity and statistical independence from a best-basis viewpoint. In Aldroubi, A., Laine, A. F., and Unser, M. A., editors, *Wavelet Applications in Signal and Image Processing VIII*, volume Proc. SPIE 4119, pages 474–486. Invited paper.

Watanabe, S. (1965). Karhunen-Loève expansion and factor analysis: Theoretical remarks and applications. In *Trans. 4th Prague Conf. Inform. Theory, Statist. Decision Functions, Random Processes*, Publishing House of the Czechoslovak Academy of Sciences, pages 635–660.

Zhu, S. C., Wu, Y., and Mumford, D. (1998). FRAME: Filters, Random fields And Maximum Entropy–Toward a unified theory for texture modeling. *Intern. J. Comput. Vision*, 27(2):1–20.

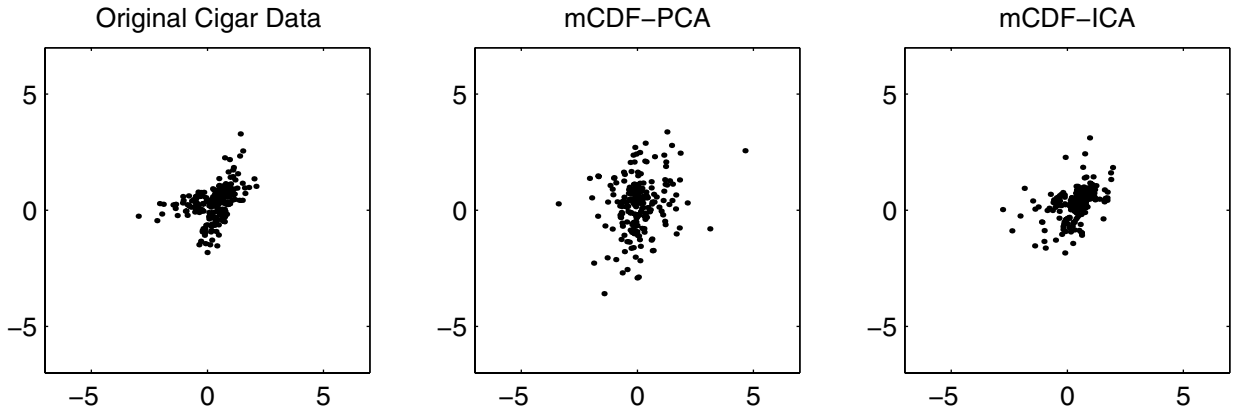

Figure 1: Simulations of the "cigar" data by mCDF-PCA and mCDF-ICA. From left to right: original samples; simulation by mCDF-PCA; simulation by mCDF-ICA.

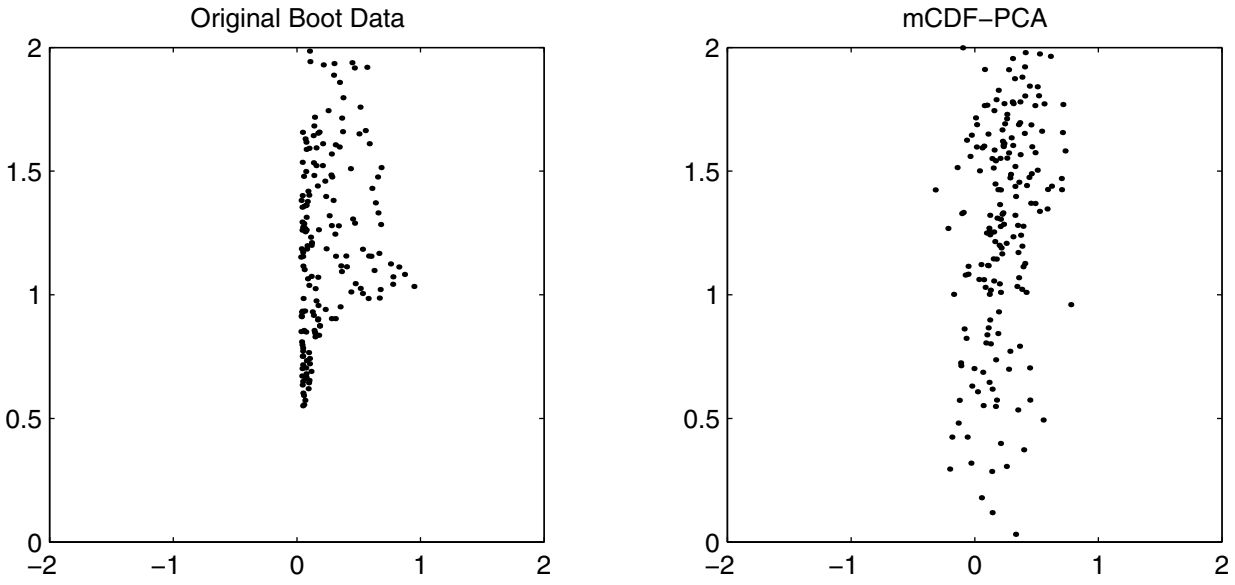

Figure 2: Simulation of the "boot" data by mCDF-PCA. From left to right: original samples; simulation by mCDF-PCA.

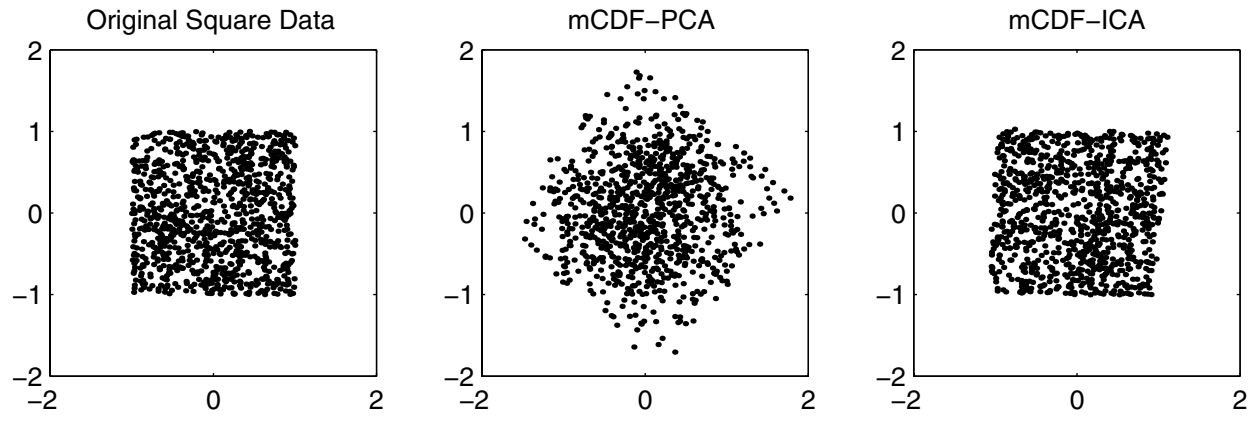

Figure 3: Simulations of the square data by mCDF-PCA and mCDF-ICA. From left to right: The original square data; simulation by mCDF-PCA; and simulation by mCDF-ICA.

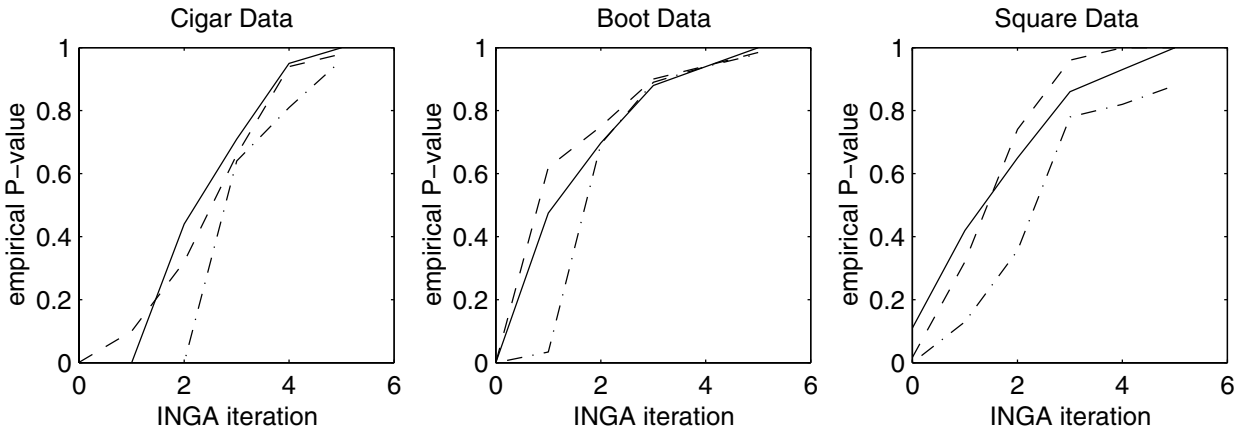

Figure 4: Empirical P-values of the squared Mahalanobis distance, multivariate kurtosis, and multivariate skewness of the INGA simulations of Cigar (left), Boot (middle), and Square (right) datasets. Solid lines: the squared Mahalanobis distance, Dashed lines: the multivariate skewness, Dashed-dotted lines: the multivariate kurtosis.

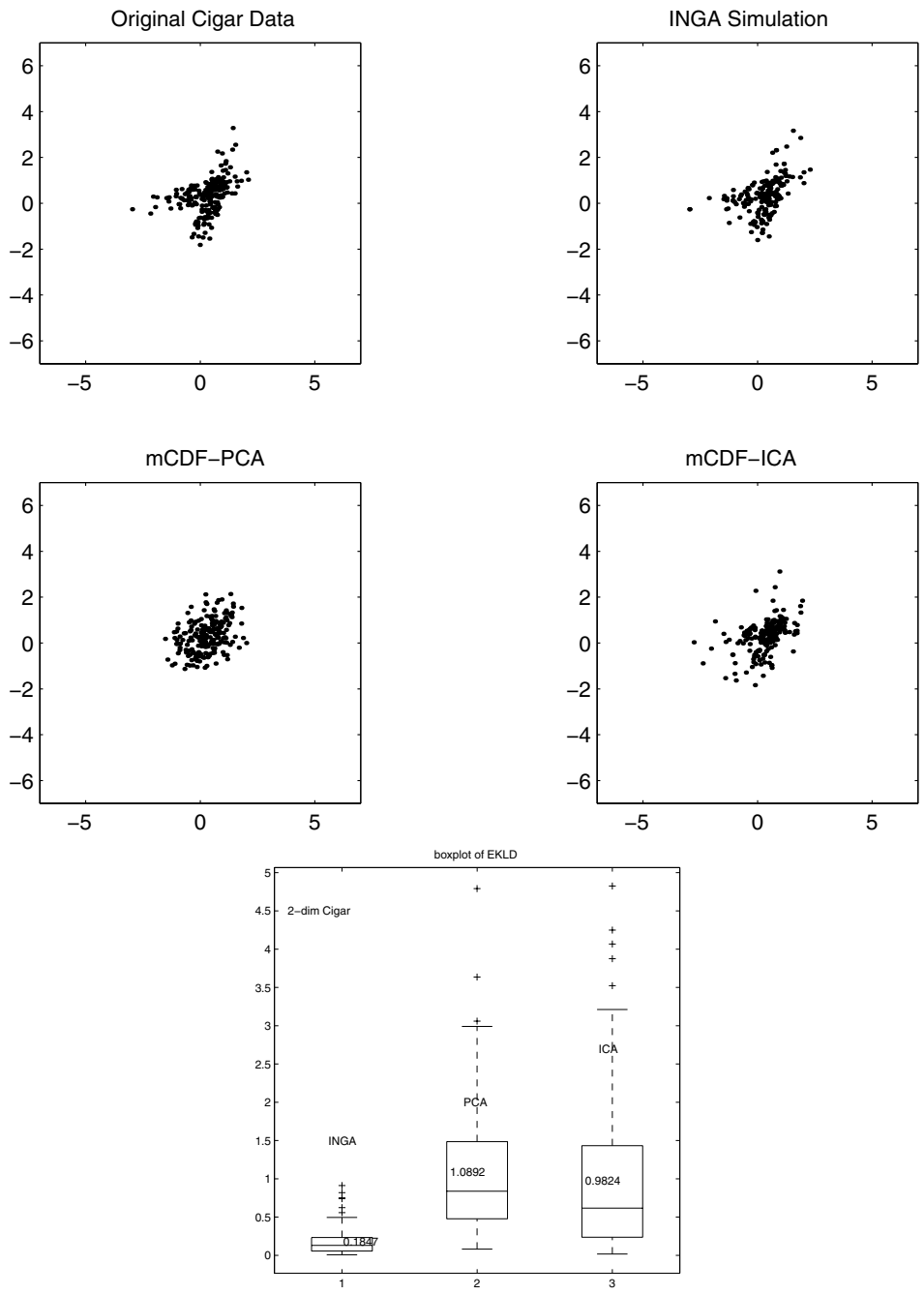

Figure 5: Simulations of the cigar data by INGA, mCDF-PCA, and mCDF-ICA. Top row from left to right: the original samples; simulation by INGA. Middle row from left to right: simulations by mCDF-PCA and mCDF-ICA. Bottom row : boxplot of EKLD of INGA, PCA and ICA.

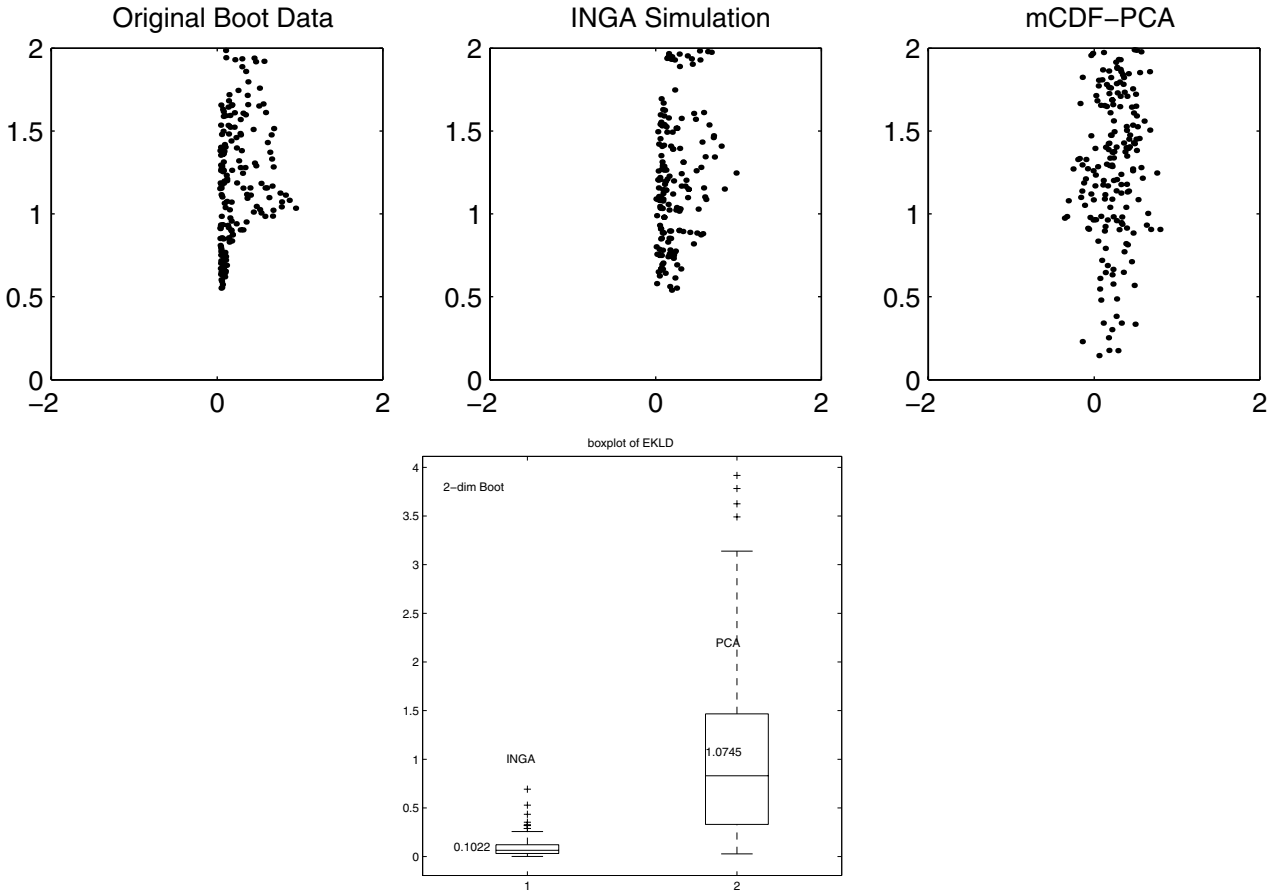

Figure 6: Simulations of the boot data by INGA and mCDF-PCA. Top row from left to right: the original samples; simulation by INGA; and simulation by mCDF-PCA. Bottom row : boxplot of EKLD of INGA, PCA and ICA.

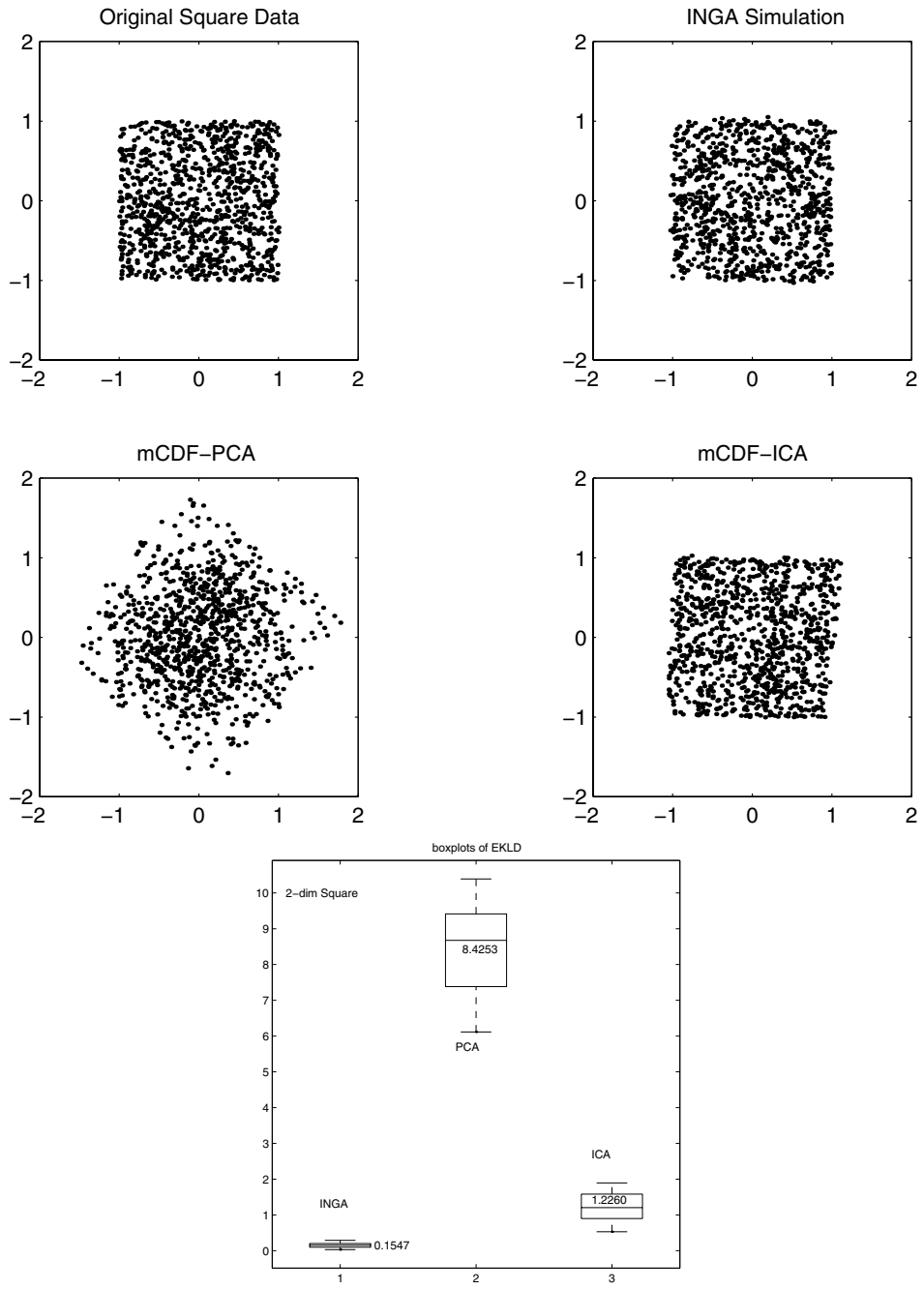

Figure 7: Simulations of the square data by INGA, mCDF-PCA, and mCDF-ICA. Top row from left to right: the original samples; simulation by INGA. Bottom row from left to right: simulations by mCDF-PCA and mCDF-ICA.

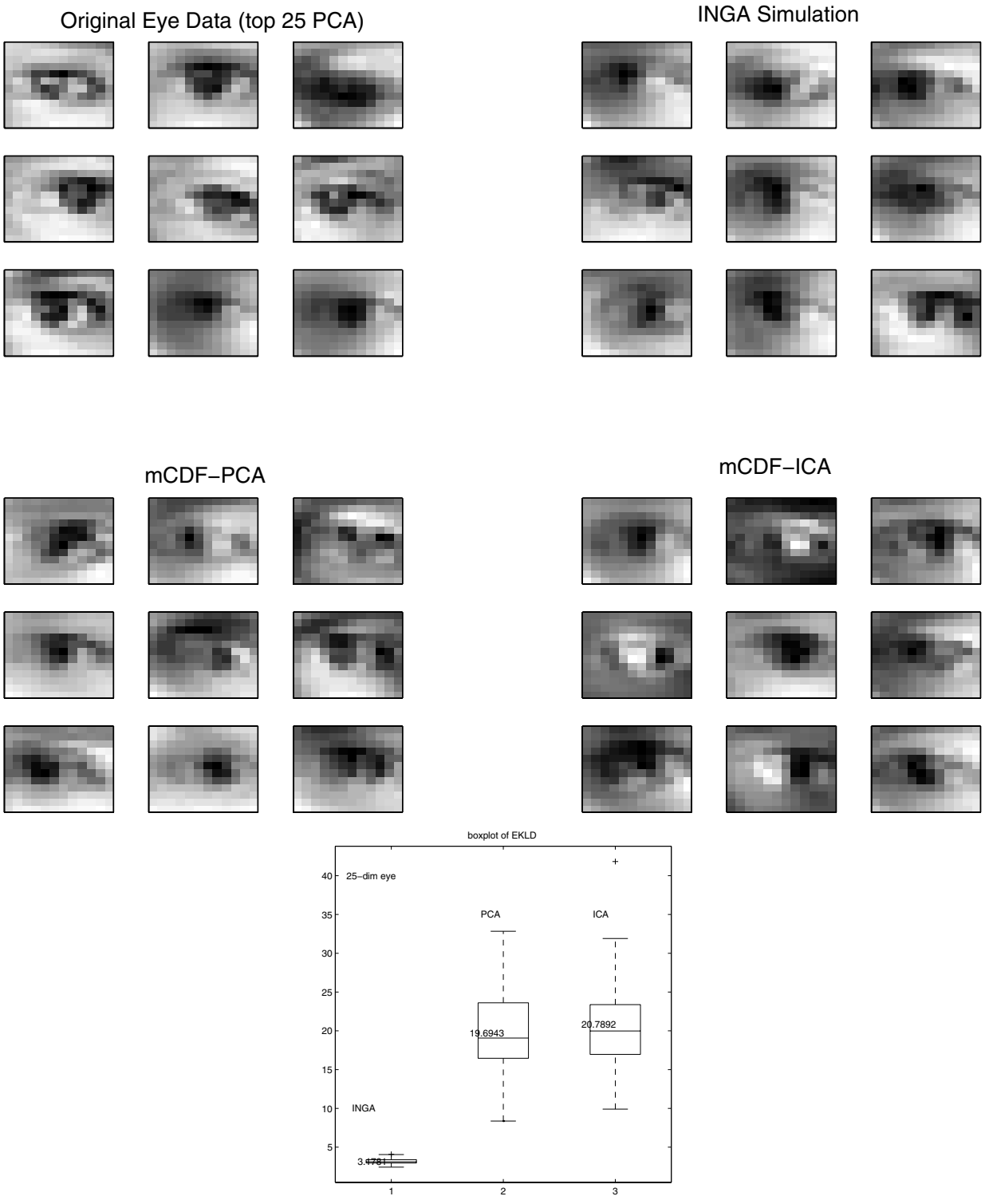

Figure 8: Simulations of the eye image database by INGA, mCDF-PCA, and mCDF-ICA. Each original eye image is represented by the top 25 PCA coordinates. Top row from left to right: the original samples; simulation by INGA. Middle row from left to right: simulations by mCDF-PCA and mCDF-ICA. Bottom row : boxplot of EKLD of INGA, PCA and ICA.

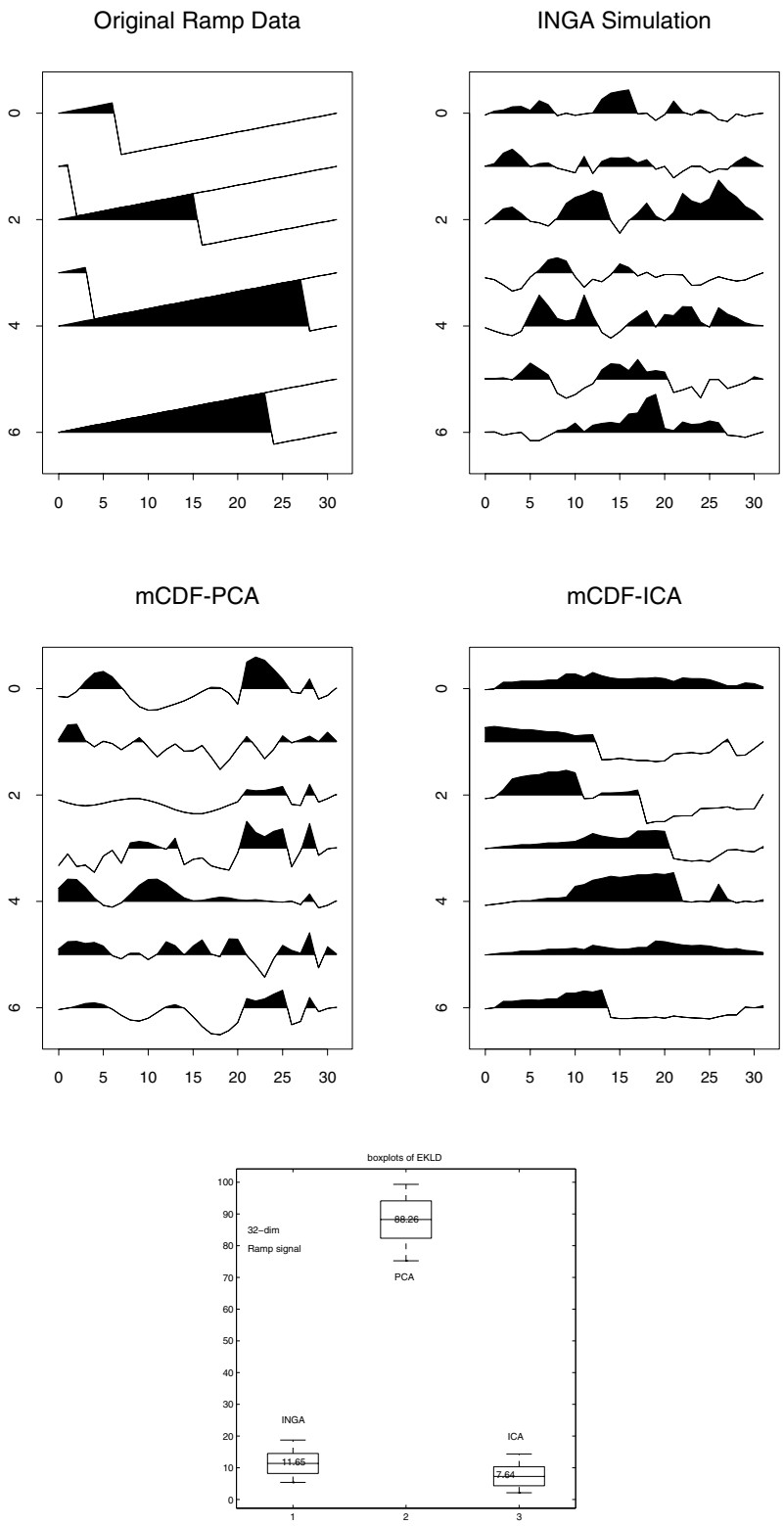

Figure 9: Simulations of the ramp data by INGA, mCDF-PCA, and mCDF-ICA. Top row from left to right: the original samples; simulation by INGA. Bottom row from left to right: simulations by mCDF-PCA and mCDF-ICA.

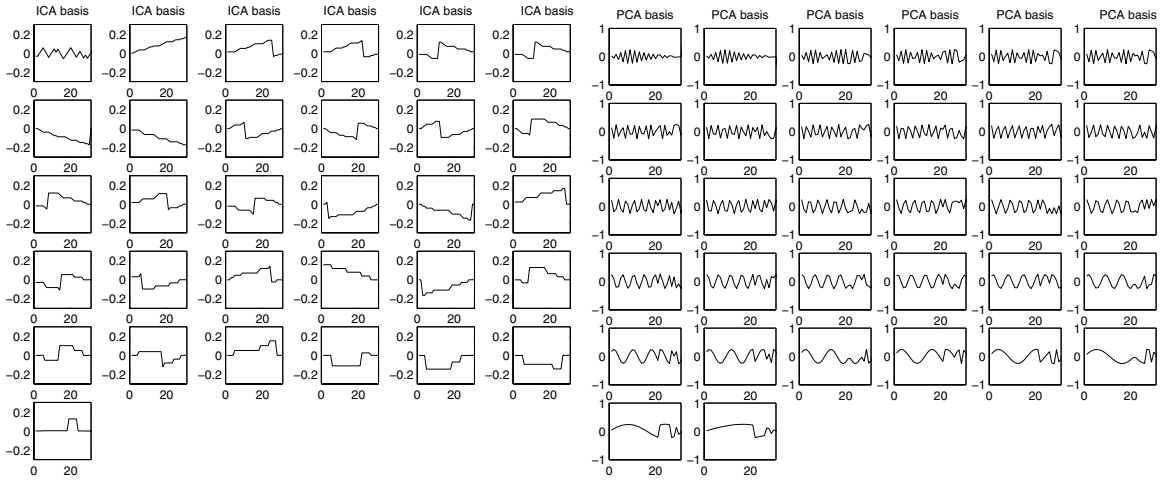

Figure 10: The ICA basis vectors (left) and the PCA basis vectors (right) of the ramp dataset.

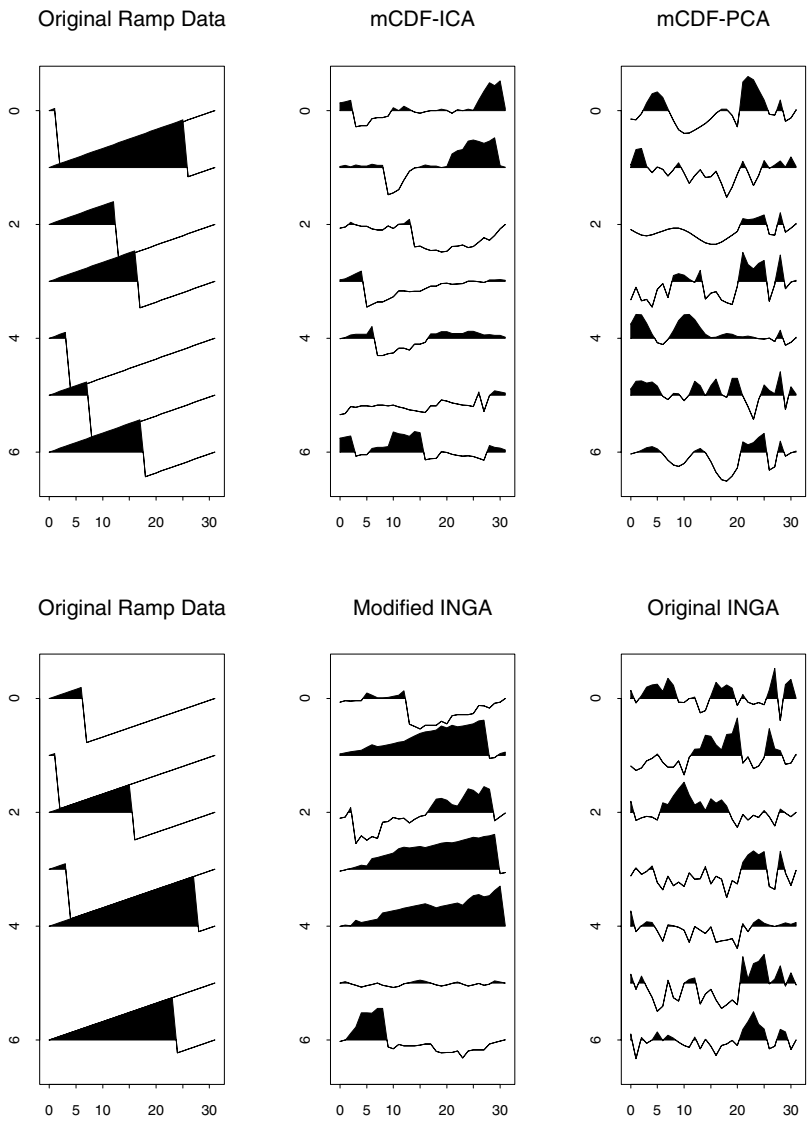

Figure 11: Comparison of various simulations. Top row from left to right: The original ramp data samples; simulations by mCDF-ICA and mCDF-PCA. Bottom row from left to right: Another set of the original samples; simulation by the modified INGA; simulation by the original INGA

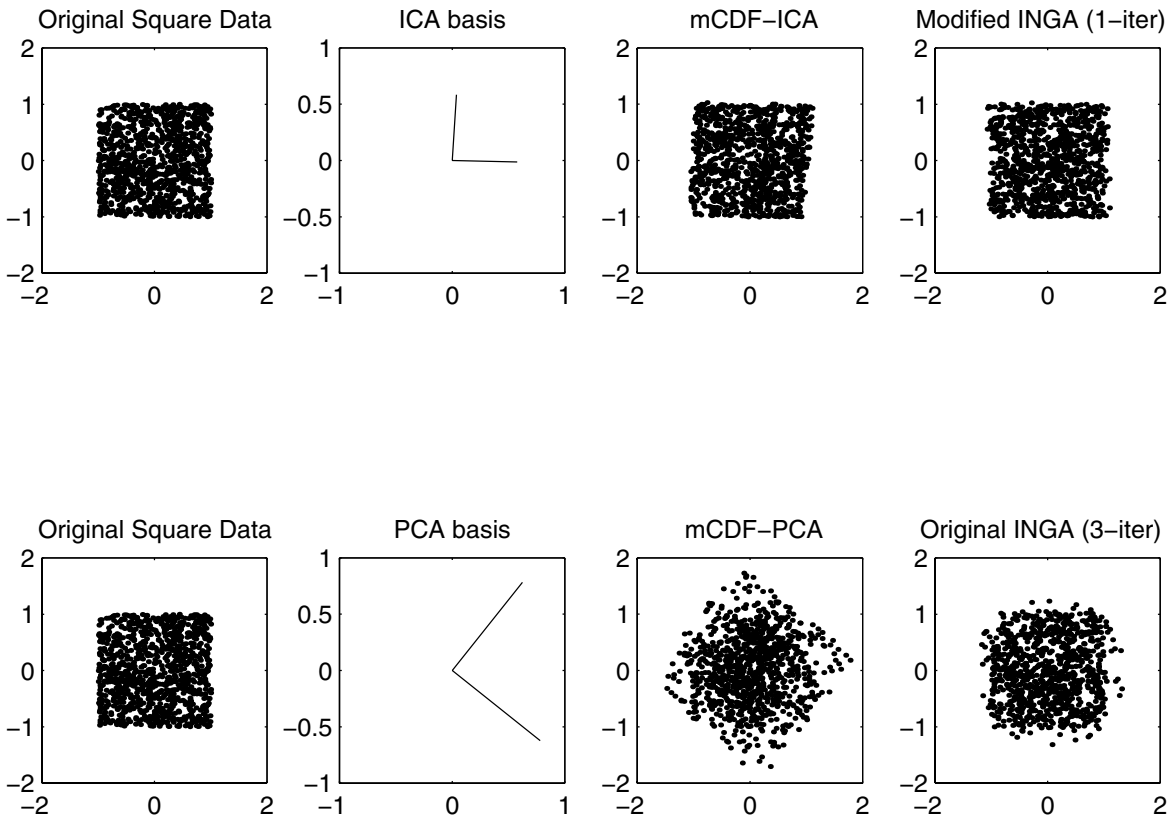

Figure 12: Effect of the initial coordinate system for the square data. Top row from left to right: The original square data; the ICA basis; simulation by mCDF-ICA; the modified INGA after one iteration. Bottom row from left to right: The original square data; the PCA basis; simulation by mCDF-PCA; the original INGA after three iterations.

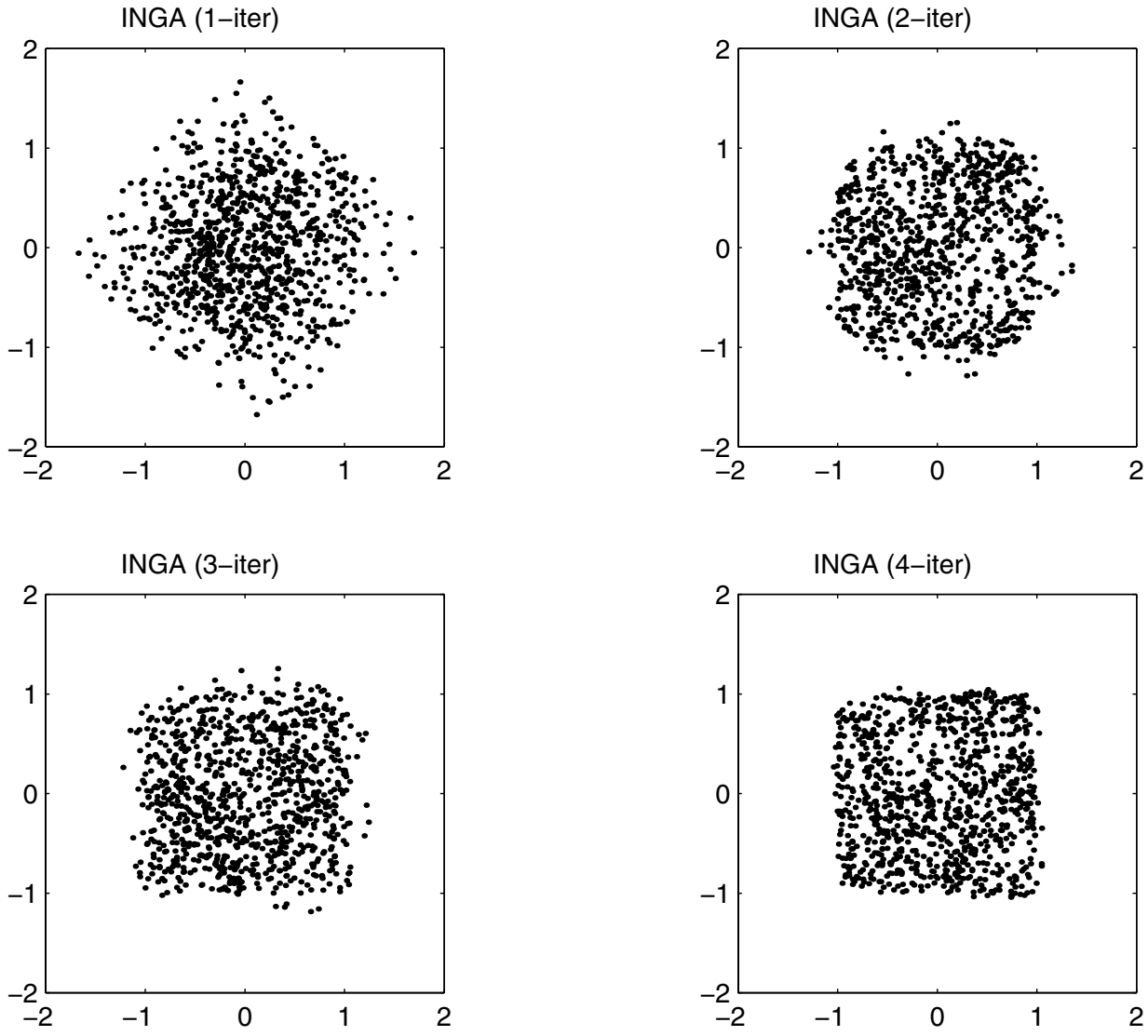

Figure 13: Iterative improvement of simulations by the original INGA from first to fourth iteration.