RESEARCH REPORT SERIES
(*Statistics #2002-03)*

# Working Papers for Mixture Model Additive Noise for Microdata Masking

William E. Yancey

Statistical Research Division
U.S. Bureau of the Census
Washington D.C. 20233

*Report Issued*: May 28, 2002

**Abstract**

We consider some aspects of using an additive mixture noise model for real microdata masking as a generalization of using normally distributed masking noise introduced by Roque [Roque]. We introduce a simplified procedure for computing additive mixture noise and consider the effectiveness of this approach from the point of view of information loss measures and record re-identification. We concentrate on the information loss statistics for the variance/covariance matrix of the full data set and of arbitrary subsets. We consider some of the information loss statistics introduced by Domingo-Ferrer [Domingo-Ferrer] and we introduce some analytic alternatives. We see that for the full data sets, the analytic properties are well preserved and the data masking is effective. The analytic properties are less well preserved on subsets of this highly skewed data. We include some SAS programs used in the study.

Key words: microdata masking, information loss statistics, record linkage re-identification

# Working Papers for Mixture Model Additive Noise for Microdata Masking

William E. Yancey
Statistical Research Division
Bureau of the Census
william.e.yancey@census.gov

May 28, 2002

# Contents

# Chapter 1

# Introduction

We collect here some of our notes involving the study of mixture models to produce additive noise for masking microdata. We consider our microdata set to be an array $X$ of real numbers, containing $n$ rows of individual records and $m$ columns of variables. A masked microdata set $Z$ is an $n \times m$ array obtained by transforming $X$,

$$f : X \to Z$$

so that ideally the masked data set $Z$ preserves the analytic validity of $X$ while being safe from disclosure. Unfortunately these are not only conflicting goals, they are not even well defined. By analytic validity, we might ideally hope that an analyst would obtain statistically equivalent results from analyzing $Z$ that he or she would from analyzing $X$. However, since there is no predicting what sort of analytic procedures the data might be subjected to, one cannot devise a test for the masked data $Z$ to see whether it will always perform comparably to $X$. By disclosure security, we mean ideally that an analyst cannot use the data in $Z$ to identify the individuals who supplied the data records in $X$. However, we cannot anticipate what possible knowledge or methodology a future analyst might bring to bear on $Z$ to attempt to re-identify individuals from $X$.

Since there is not a definitive answer to what is meant by analytic validity, as a basic requirement, we concentrate on testing to see whether first and second order statistics (sample means and covariances) are preserved. Again we need to define in a quantitative way what we mean by preserving means and covariances. To this end we have used and adapted some of the information loss measures proposed by Domingo-Ferrer.

To try to evaluate disclosure security, we can only apply a re-identification method and see how we do. We have used probabalistic record linkage software derived from the Fellegi-Sunter model to try to match the set of records in $Z$ with the set of records in $X$ to see what proportion of records can be reliably re-identified ([Winkler94],[Winkler95],[Winkler98]). We believe that this is a pretty stringent test for disclosure security since for methodology it uses sophisticated record linkage software developed at the Census Bureau and for knowledge it uses the complete data set $X$ from which the masked data set $Z$ was derived.

The empirical results of these tests applied to additive mixture model masking are discussed in Chapter 4. A comparison of these empirical results with those from other data masking methods are given in [Yancey].

## 1.1 Additive Mixture Noise Masking

### 1.1.1 Additive Noise Masking

The idea of additive noise masking was explored by Jay Kim [Kim86]. If we think of our data $X$ as being $n$ independent samples of a random vector $x$ with mean $\mu$ and covariance $\Sigma$, then we can use a (normal) random vector $y$ with mean 0 and covariance $\Sigma$ to additively produce a masked random variable $z$, where

$$z = x + \sqrt{d}y$$

for a scalar $d$. The resulting random vector $z$ has mean $\mu$ and covariance $(1+d)\,\Sigma$. If we prefer to have a masked random variable with the same covariance as $x$, we can always rescale to get

$$\begin{aligned} z_s &= \frac{z-\mu}{\sqrt{1+d}} + \mu \\ &= \frac{1}{\sqrt{1+d}} z + \left(1 - \frac{1}{\sqrt{1+d}}\right)\mu \end{aligned}$$

so that $z_s$ has mean $\mu$ and covariance $\Sigma$, the same as $x$. Thus by taking $n$ independent samples of the random vector $y$, we can generate a masked data set

$$Z = X + \sqrt{d}Y$$

where the masked data $Z$ theoretically has the same mean and proportional covariance to the raw data $X$. Since we are taking samples, we can only

compute the sample mean and sample covariance, but for reasonably large samples we expect fairly close agreement.

In order to generate the random noise vector, we can use a random number function that generates independent sample of a standard normal random variable. Thus we can get a random vector $\varepsilon$ that has mean 0 and covariance $I$. In order to get the appropriate covariance for $y$, we can multiply by a square root of the covariance, that is a matrix $\Sigma^{\frac{1}{2}}$ with the property that

$$\Sigma^{\frac{1}{2}} \left(\Sigma^{\frac{1}{2}}\right)^T = \Sigma.$$

Then the random variable

$$y = \Sigma^{\frac{1}{2}} \varepsilon$$

also has mean 0 and covariance given by

$$\begin{aligned} \mathsf{E}\left[yy^T\right] &= \Sigma^{\frac{1}{2}} \mathsf{E}\left[\varepsilon\varepsilon^T\right] \left(\Sigma^{\frac{1}{2}}\right)^T \\ &= \Sigma^{\frac{1}{2}} I \left(\Sigma^{\frac{1}{2}}\right)^T \\ &= \Sigma \end{aligned}$$

Such square root matrices exist for positive definite covariance matrices $\Sigma$. One way to compute one is via the Cholesky decomposition algorithm which produces a square root $\Sigma^{\frac{1}{2}}$ in lower triangular form. Another way is to compute the eigenvalues and eigenvectors of $\Sigma$, so that one has

$$\Sigma = UDU^T$$

where $D$ is a diagonal matrix of eigenvalues of $\Sigma$ and $U$ is an orthogonal matrix of eigenvectors of $\Sigma$. Since $\Sigma$ is positive definite, all of the eigenvectors are positive, so we may also compute a square root as

$$\Sigma^{\frac{1}{2}} = UD^{\frac{1}{2}}$$

where $D^{\frac{1}{2}}$ is the diagonal matrix of the square roots of the eigenvalues of $\Sigma$.

The virtue of this data masking method is that means and covariances can be preserved. Moreover, as shown by Jay Kim, means and covariances are theoretically preserved on arbitrary subpopulations [Kim90]. Thus this method should rate pretty well for preserving analytic validity. However, a shortcoming of the method, is that since the masking terms are generated

from a normal random variable of 0 mean, most of the samples generated tend to be near 0, and thus the corresponding masked values are not greatly changed from the unmasked values. The result is that such masked data can have a fairly high re-identification rate when using the Census probabilistic record linkage software, and hence may not rate sufficiently highly for data security [Kim95]. Methods has been tried to either edit the additive masking terms to only use sufficiently large ones or to use rank swapping subsequent to additive noise masking in order to disguise the most easily re-identified records. Of course these adaptations have their price on the analytic validity.

### 1.1.2 Mixture Noise

The approach investigated by Roque [Roque] was to replace the distribution of the additive noise term by a mixture of distributions. Thus instead of $y$ having a normal distribution of mean 0 and covariance $\Sigma$, we let $y$ have a density function $f$ of the form

$$f\left(x;0,\Sigma\right)=\sum_{j=1}^{k}\omega_{j}f_{j}\left(x;\mu_{j},\Sigma_{j}\right)$$

where we could take each density function $f_j\left(x;\mu_j,\Sigma_j\right)$ to be a (normal) density of mean $\mu_j$ and covariance $\Sigma_j$. We can see some basic properties of mixture distributions.

Let $X_j$ be a random variable with distribution $f_j\left(x;\mu_j,\Sigma_j\right)$ with mean $\mu_j$ and covariance matrix $\Sigma_j$, so that

$$\begin{aligned}1 &= \int f_j dx\\ \mu_j &= \int x f_j dx\\ \Sigma_j &= \int \left(x-\mu_j\right)\left(x-\mu_j\right)^T f_j dx\end{aligned}$$

so that

$$\begin{aligned}\Sigma_j &= \int \left(x-\mu_j\right)\left(x-\mu_j\right)^T f_j dx \\ &= \int xx^T f_j dx - \left(\int x f_j dx\right)\mu^T - \mu\left(\int x^T f_j dx\right) + \mu_j\mu_j^T \\ &= \int xx^T f_j dx - \mu_j\mu_j^T\end{aligned}$$

so

$$\int xx^T f_j dx = \Sigma_j + \mu_j\mu_j^T. \tag{1.1}$$

Let $Y$ be a random variable with mixture distribution

$$f = \sum_{j=1}^{k} \omega_j f_j,\ \omega_j \geq 0.$$

In order for $f$ to be a probability distribution, we must have

$$1 = \int f = \sum_{j=1}^{k} \omega_j \int f_j$$

so that

$$\sum_{j=1}^{k} \omega_j = 1.$$

The mean $\mu$ of $Y$ is the first moment

$$\begin{aligned}\mu &= \int xf \\ &= \sum_{j=1}^{k} \omega_j \int x f_j \\ &= \sum_{j=1}^{k} \omega_j \mu_j\end{aligned}$$

while the covariance matrix $\Sigma$ of $Y$ is the second moment about the mean

$$\begin{aligned}\Sigma &= \int (x-\mu)(x-\mu)^T f \\ &= \int xx^T f - \left(\int xf\right)\mu^t - \mu\left(\int x^T f\right) + \mu\mu^T \\ &= \int xx^T f - \mu\mu^T\end{aligned}$$

and

$$\begin{aligned}\int xx^T f &= \sum_{j=1}^{k} \omega_j \int xx^T f_j \\ &= \sum_{j=1}^{k} \omega_j \left(\Sigma_j + \mu_j\mu_j^T\right)\end{aligned}$$

from (1.1), so that

$$\Sigma = \sum_{j=1}^{k} \omega_j \left(\Sigma_j + \mu_j\mu_j^T\right) - \mu\mu^T.$$

In the special case where we have chosen our mixture distribution to have zero mean, *i.e.*

$$\mu = \sum_{j=1}^{k} \omega_j\mu_j = 0,$$

we have that the covariance matrix of $Y$ is given by

$$\Sigma = \sum_{j=1}^{k} \omega_j \left(\Sigma_j + \mu_j\mu_j^T\right).$$

We can make the simplifications that all component covariances are proportional to the total covariance

$$\Sigma_j = \sigma_j\Sigma$$

where $\sigma_j > 0$ is a scalar, or even more simply that they are all the same

$$\Sigma_j = \sigma\Sigma.$$

Also there seems to be no reason not to make all weights equal

$$\omega_j = \frac{1}{k}.$$

In this case, the nonlinear system for the covariance simplifies to

$$\begin{aligned} \Sigma &= \frac{1}{k}\sum_{j=1}^{k}\left(\sigma\Sigma + \mu_j\mu_j^T\right) \\ k\left(1-\sigma\right)\Sigma &= \sum_{j=1}^{k}\mu_j\mu_j^T \end{aligned}$$

However, even with these simplifications, in order to solve the (underdetermined) nonlinear system for the mean vectors $\mu_j$ requires some sort of numerical solution techniques. In Chapter 2, we give a simplification analogous to the standard noise model to avoid these numerical complications.

# Chapter 2

# A Factored Approach to Computing Mixture-Model Noise

## 2.1 Problem Standardization

Let $X$ be a random vector with mean $\mu$ and covariance $\Sigma$. We wish to produce a new random variable $Z$ with mean $\mu$ and covariance $(1+d)\,\Sigma$ by adding an independent random vector $Y$ with mean 0 and covariance $d\Sigma$,

$$Z = X + Y.$$

To standardize the problem, we can scale by the square root $\Sigma^{\frac{1}{2}}$ of the covariance matrix where

$$\Sigma^{\frac{1}{2}} \left(\Sigma^{\frac{1}{2}}\right)^T = \Sigma$$

to have the equivalent condition

$$\Sigma^{-\frac{1}{2}} Z = \Sigma^{-\frac{1}{2}} X + \Sigma^{-\frac{1}{2}} Y.$$

Since we had that

$$\Sigma = \mathsf{E}\left[(X-\mu)\,(X-\mu)^T\right]$$

we now have

$$\begin{aligned}\mathsf{E}\left[\Sigma^{-\frac{1}{2}}\,(X-\mu)\,(X-\mu)^T \left(\Sigma^{-\frac{1}{2}}\right)^T\right] &= \Sigma^{-\frac{1}{2}}\Sigma\left(\Sigma^{-\frac{1}{2}}\right)^T \\ &= I\end{aligned}$$

so that

$$\tilde{X} = \Sigma^{-\frac{1}{2}} X$$

has mean $\tilde{\mu} = \Sigma^{-\frac{1}{2}} \mu$ and covariance $I$. Likewise, the transformed variable

$$\tilde{Y} = \Sigma^{-\frac{1}{2}} Y$$

has mean 0 and variance $dI$.

Thus we wish to solve the standardized problem

$$\tilde{Z} = \tilde{X} + \tilde{Y}.$$

To produce $\tilde{Y}$, let

$$\tilde{Y} = \begin{pmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_p \end{pmatrix}$$

where the $\{Y_i\}$ are independent scalar random variables of mean 0 and variance $d$. For the Kim-Winkler method, we can use

$$Y_i = \sqrt{d} W_i$$

where $W_i$ has a standard normal distribution and the $\{W_i\}$ are independent. However, any distribution with zero mean and unit variance will do. The Roque mixture approach is to use mixture distributions where $W_i$ has probability density function $f_i$ where

$$W_i \sim f_i = \sum_{j=1}^{k} \omega_j f_j \left( x; \theta_j, d_i \right),$$

where $f_j \left( x; \theta_j, d_i \right)$ is a scalar probability density with mean $\theta_j$ and variance $d_i$. In order for this to be a probability density, we must have $\omega_i > 0$ and

$$\begin{aligned} 1 &= \int f_i \\ &= \sum_{j=1}^{k} \omega_j \int f_j \\ &= \sum_{j=1}^{k} \omega_j \end{aligned}$$

The mean of $W_i$ is given by

$$\begin{aligned}\int x f_i &= \sum_{j=1}^{k} \omega_j \int x f_j \\ &= \sum_{j=1}^{k} \omega_j \theta_j\end{aligned}$$

so if $W_i$ is to have zero mean, we must have

$$\sum_{j=1}^{k} \omega_j \theta_j = 0.$$

Likewise, for zero mean $W_i$, the variance of $W_i$ is given by

$$\begin{aligned}\int x^2 f_i &= \sum_{j=1}^{k} \omega_j \int x^2 f_j \\ &= \sum_{j=1}^{k} \omega_j \left(d_i + \theta_j^2\right)\end{aligned}$$

Thus if we want $W_i$ to equal 1, then we must have

$$\begin{aligned}\sum_{j=1}^{k} \omega_j \theta_j^2 &= 1 - \sum_{j=1}^{k} \omega_j d_i \\ &= 1 - d_i\end{aligned}$$

The simplest case for the mixture is if $\omega_j = \frac{1}{k}$, in which case

$$\sum_{j=1}^{k} \theta_j^2 = k\left(1 - d_i\right).$$

## 2.2 Computational Approach

So it looks like the simplest way to produce a mixture component is as follows. Choose a component distortion factor $\sigma, 0 < \sigma < 1$ and a number $k \geq 2$ of

mixture components. Choose arbitrary numbers $\psi_j$, $1 \leq j \leq k-1$. Let

$$\psi_k = -\sum_{j=1}^{k-1} \psi_j.$$

Compute

$$S = \sum_{j=1}^{k} \psi_j^2$$

and let

$$\theta_j = \sqrt{\frac{k\left(1-\sigma^2\right)}{S}}\psi_j$$

so that

$$\sum_{j=1}^{k} \theta_j = 0$$

and

$$\sum_{j=1}^{k} \theta_j^2 = k\left(1-\sigma^2\right).$$

Let scalar random variable $Y$ have mixture density

$$\frac{1}{k}\sum_{j=1}^{k} f_j\left(x; \theta_j, \sigma^2\right)$$

where the $f_j\left(x; \theta_j, \sigma^2\right)$ are scalar probability densities with mean $\theta_j$ and variance $\sigma^2$. Then we have seen that $Y$ is a random variable with mean 0 and variance

$$\sigma^2 + \frac{1}{k}\sum_{j=1}^{k} \theta_j^2 = 1.$$

If we create our random vector

$$\tilde{Y} = \begin{pmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_p \end{pmatrix}$$

by using independent scalar components $Y_i$ generated as above, then $\tilde{Y}$ has mean 0 and covariance $I$. Thus if we let

$$Y = \sqrt{d}\Sigma^{\frac{1}{2}}\tilde{Y}$$

then $Y$ has mean 0 and covariance $d\Sigma$.

**Example 1** *The smallest case is $k = 2$. In this case, $\psi_2 = -\psi_1$ and $S = 2\psi_1^2$, so that*

$$\theta_1 = \sqrt{\frac{2\left(1-\sigma^2\right)}{2\psi_1^2}}\psi_1 = \sqrt{1-\sigma^2}$$

*and $\theta_2 = -\sqrt{1-\sigma^2}$. Thus the only parameter choice is $0 < \sigma < 1$. As $\sigma \to 1^-$, both $\theta_1, \theta_2 \to 0$. If we are using normal densities in the mixture, then the mixture tends toward the standard normal density. As $\sigma \to 0^+$, the symmetric mixture becomes more pronouncedly bimodal, in the limit approaching binary density centered at $\pm 1$. For smaller $\sigma$, as the mixture density gets more concentrated at near $\pm 1$, the density tails diminish.*

# Chapter 3

# Jay Kim's Method for Computing Subpopulation Estimates from Masked Data

## 3.1 Masked Data and Scaled Masked Data

Let $X$ be a data set with mean $\mu$ and covariance $\Sigma$. We generate independent noise $Y$ with mean 0 and covariance $\Sigma$. That is, for each record $x \in X$, we obtain the masked record $z$ by computing

$$z = x + \sqrt{d}y$$

where $y$ is an independent random vector with mean $\mathsf{E}[y] = 0$ and covariance $\mathrm{Cov}(y) = \Sigma$. The masked data is then

$$Z = X + \sqrt{d}Y$$

which has mean $\mathsf{E}[z] = \mu$ and covariance $\mathrm{Cov}(z) = (1+d)\Sigma$. If we prefer, we can get scaled masked data set $Z'$ with covariance equal to the raw data by using

$$\begin{aligned} z' &= \frac{z-\mu}{\sqrt{1+d}} + \mu \\ &= \frac{1}{\sqrt{1+d}} z + \left(1 - \frac{1}{\sqrt{1+d}}\right)\mu \end{aligned}$$

since then

$$\begin{aligned} \mathsf{E}[z'] &= \frac{1}{\sqrt{1+d}}\mathsf{E}[z] + \left(1 - \frac{1}{\sqrt{1+d}}\right)\mu \\ &= \mu \end{aligned}$$

and

$$\begin{aligned} \mathrm{Cov}(z') &= \mathsf{E}\left[(z'-\mu)(z'-\mu)^T\right] \\ &= \frac{1}{1+d}\mathsf{E}\left[(z-\mu)(z-\mu)^T\right] \\ &= \frac{1}{1+d}\mathrm{Cov}(z) \\ &= \Sigma \end{aligned}$$

## 3.2 Statistics for Subpopulations of Masked Data

Now suppose that we want to consider the subpopulation $Z_s$ of the masked data set $Z$ where the records have been obtained from the subpopulation $X_s$ of the unmasked data set $X$ by

$$z_s = x_s + \sqrt{d}y.$$

We see that we still have, as in (1) in Kim,

$$\begin{aligned} \mathsf{E}[z_s] &= \mathsf{E}[x_s] + \sqrt{d}\mathsf{E}[y] \\ &= \mathsf{E}[x_s] \\ &= \mu_s \end{aligned}$$

but that the covariance is

$$\begin{aligned} \mathrm{Cov}(z_s) &= \mathsf{E}\left[(z_s-\mu_s)(z_s-\mu_s)^T\right] \\ &= \mathsf{E}\left[\left((x_s-\mu_s)+\sqrt{d}y\right)\left((x_s-\mu_s)+\sqrt{d}y\right)^T\right] \\ &= \mathsf{E}\left[(x_s-\mu_s)(x_s-\mu_s)^T\right] + \mathsf{E}\left[\left(\sqrt{d}y\right)\left(\sqrt{d}y\right)^T\right] \\ &= \mathrm{Cov}(x_s) + d\Sigma \end{aligned}$$

Thus if we want to recover the covariance of the original subpopulation covariance from the masked data, since

$$\mathrm{Cov}\,(z) = (1+d)\,\Sigma,$$

we have that

$$\begin{aligned}\mathrm{Cov}\,(x_s) &= \mathrm{Cov}\,(z_s) - d\Sigma \\ &= \mathrm{Cov}\,(z_s) - \frac{d}{1+d}\,\mathrm{Cov}\,(z)\end{aligned}$$

as in (2) and (5) in [Kim90].

## 3.3 Statistics for Subpopulations of Scaled Masked Data

If we choose our subpopulation from the covariance-corrected masked data, so that

$$\begin{aligned}z'_s &= \frac{1}{\sqrt{1+d}}z_s + \left(1 - \frac{1}{\sqrt{1+d}}\right)\mu \\ &= \frac{1}{\sqrt{1+d}}\left(x_s + \sqrt{d}y\right) + \left(1 - \frac{1}{\sqrt{1+d}}\right)\mu\end{aligned}$$

we see that we have the sample mean

$$\begin{aligned}\mathsf{E}\,[z'_s] &= \frac{1}{\sqrt{1+d}}\left(\mathsf{E}\,[x_s] + \sqrt{d}\mathsf{E}\,[y]\right) + \left(1 - \frac{1}{\sqrt{1+d}}\right)\mu \\ &= \frac{1}{\sqrt{1+d}}\mu_s + \left(1 - \frac{1}{\sqrt{1+d}}\right)\mu\end{aligned}$$

so we may recover $\mathsf{E}\,[x_s] = \mu_s$ from

$$\begin{aligned}\mu_s &= \sqrt{1+d}\mathsf{E}\,[z'_s] - \left(\sqrt{1+d} - 1\right)\mu \\ &= \sqrt{1+d}\mathsf{E}\,[z'_s] - \left(\sqrt{1+d} - 1\right)\mathsf{E}\,[z']\end{aligned}$$

as in [Kim90] (7). So since

$$\begin{aligned}z'_s - \mathsf{E}\,[z'_s] &= \frac{1}{\sqrt{1+d}}\left(x_s + \sqrt{d}y\right) + \left(1 - \frac{1}{\sqrt{1+d}}\right)\mu - \left(\frac{1}{\sqrt{1+d}}\mu_s + \left(1 - \frac{1}{\sqrt{1+d}}\right)\mu\right) \\ &= \frac{1}{\sqrt{1+d}}\left(x_s - \mu_s\right) + \frac{\sqrt{d}}{\sqrt{1+d}}y\end{aligned}$$

then the covariance is

$$\begin{aligned}
\mathrm{Cov}\,(z'_s) &= \mathsf{E}\left[(z'_s - \mathsf{E}\,[z'_s])\,(z'_s - \mathsf{E}\,[z'_s])^T\right] \\
&= \mathsf{E}\left[\left(\frac{1}{\sqrt{1+d}}\,(x_s - \mu_s) + \frac{\sqrt{d}}{\sqrt{1+d}}y\right)\left(\frac{1}{\sqrt{1+d}}\,(x_s - \mu_s) + \frac{\sqrt{d}}{\sqrt{1+d}}y\right)^T\right] \\
&= \frac{1}{1+d}\,\mathrm{Cov}\,(x_s) + \frac{d}{1+d}\,\mathrm{Cov}\,(y)
\end{aligned}$$

If we want to recover $\mathrm{Cov}\,(x_s)$ from the masked data, we can use

$$\begin{aligned}
\mathrm{Cov}\,(y) &= \Sigma \\
&= \mathrm{Cov}\,(z')
\end{aligned}$$

so that

$$\begin{aligned}
\mathrm{Cov}\,(x_s) &= (1+d)\,\mathrm{Cov}\,(z'_s) - d\,\mathrm{Cov}\,(y) \\
&= (1+d)\,\mathrm{Cov}\,(z'_s) - d\,\mathrm{Cov}\,(z')
\end{aligned}$$

## 3.4 Revised Computation of Subpopulation Estimates from Masked Data

Jay Kim has observed that we do not really subtract the mean of the distributions but only the sample mean, which contributes to the variance of the estimates. Here we recalulate the above variances using the sample mean. The asymptotic formulas remain the same.

### 3.4.1 Masked Data and Scaled Masked Data

Let $X$ be a data set

$$X = \begin{pmatrix} x_1^T \\ x_2^T \\ \vdots \\ x_n^T \end{pmatrix}$$

that is a set of $n$ independent samples of a random vector $x$ of dimension $m$, where $x$ has mean

$$\mu = \mathsf{E}\,[x]$$

and variance/covariance matrix

$$\Sigma = \mathrm{Var}\,(x) = \mathsf{E}\left[(x-\mu)\,(x-\mu)^T\right].$$

We note that the sample mean

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

has mean

$$\mathsf{E}\,[\bar{x}] = \frac{1}{n}\sum_{i=1}^{n} \mathsf{E}\,[x] = \mu$$

and variance

$$\mathrm{Var}\,(\bar{x}) = \frac{1}{n}\Sigma.$$

The sample $X$ has sample variance

$$S = \frac{1}{n}\sum_{i=1}^{n} (X-\mu)^T\,(X-\mu)$$

where we denote

$$X - \mu = \begin{pmatrix} (x_1-\mu)^T \\ (x_2-\mu)^T \\ \vdots \\ (x_n-\mu)^T \end{pmatrix}.$$

Since

$$\mathsf{E}\left[(x_i-\mu)\,(x_j-\mu)^T\right] = \begin{cases} \Sigma & \text{if } i=j \\ 0 & \text{if } i\neq j \end{cases} \tag{3.1}$$

due to the independence of the samples, we see that

$$\mathsf{E}\,[S] = \Sigma.$$

Let $y$ be a masking random vector, independent of $x$, of dimension $m$ with mean 0 and variance $\Sigma$. We define the masked random variable $z$ by

$$z = x + \sqrt{d}y$$

for a given choice of the scalar parameter $d > 0$. We see that $z$ has mean

$$\begin{aligned} \mathsf{E}[z] &= \mathsf{E}\left[x + \sqrt{d}y\right] \\ &= \mathsf{E}[x] + \sqrt{d}\mathsf{E}[y] \\ &= \mu \end{aligned}$$

and variance

$$\begin{aligned} \mathsf{E}\left[(z-\mu)(z-\mu)^T\right] &= \mathsf{E}\left[\left(x-\mu+\sqrt{d}y\right)\left(x-\mu+\sqrt{d}y\right)^T\right] \\ &= \mathsf{E}\left[(x-\mu)(x-\mu)^T\right] + \sqrt{d}\left(\mathsf{E}[x-\mu]\,\mathsf{E}\left[y^T\right] + \mathsf{E}[y]\,\mathsf{E}\left[(x-\mu)^T\right]\right) + \\ &\quad d\mathsf{E}\left[yy^T\right] \\ &= (1+d)\,\Sigma. \end{aligned}$$

Thus the masked data set

$$Z = \begin{pmatrix} z_1 \\ z_2 \\ \vdots \\ z_n \end{pmatrix}$$

has a sample mean with expected value $\mu$ and sample variance with expected value $(1+d)\,\Sigma$.

Now suppose that we want to rescale our masked data so that its sample variance is also $\Sigma$. If we knew the mean $\mu$ of $x$, then the simple way to do that would be to compute

$$\begin{aligned} z' &= \frac{z-\mu}{\sqrt{1+d}} + \mu \\ &= \frac{1}{\sqrt{1+d}}z + \left(1 - \frac{1}{\sqrt{1+d}}\right)\mu \end{aligned}$$

where it is clear that $z'$ has mean $\mu$ and variance $\Sigma$. However, in practice $\mu$ is generally unknown, so we need to use an approximation, such as the sample mean $\bar{z}$. So instead, if we define our scaled masked variable by

$$z' = \frac{1}{\sqrt{1+d}}z + \left(1 - \frac{1}{\sqrt{1+d}}\right)\bar{z}$$

we see that we still have

$$\begin{aligned} \mathsf{E}[z'] &= \frac{1}{\sqrt{1+d}}\mathsf{E}[z] + \left(1 - \frac{1}{\sqrt{1+d}}\right)\mathsf{E}[\bar{z}] \\ &= \mu \end{aligned}$$

but since $\bar{z}$ is not constant, but has it's own variance, the variance of $z'$ is more complicated. To simplify notation somewhat, let

$$a = \frac{1}{\sqrt{1+d}}$$

so that

$$\frac{1}{a^2} = 1 + d.$$

We can then compute

$$\begin{aligned} \mathrm{Var}(z') &= \mathsf{E}\left[(z'-\mu)(z'-\mu)^T\right] \\ &= \mathsf{E}\left[(a(z-\mu) + (1-a)(\bar{z}-\mu))(a(z-\mu) + (1-a)(\bar{z}-\mu))^T\right] \end{aligned}$$

We recall that

$$\mathrm{Var}(z) = \mathsf{E}\left[(z-\mu)(z-\mu)^T\right] = \frac{1}{a^2}\Sigma$$

and similarly

$$\mathrm{Var}(\bar{z}) = \frac{1}{na^2}\Sigma$$

For any sample element $z_i$ from our masked data, we have

$$\begin{aligned} \mathsf{E}\left[(z_i-\mu)(\bar{z}-\mu)^T\right] &= \frac{1}{n}\sum_{j=1}^{n}\mathsf{E}\left[(z_i-\mu)(z_j-\mu)^T\right] \\ &= \frac{1}{na^2}\Sigma \end{aligned}$$

similarly to (3.1) due to the independence of the $z_i$. Thus we can see that

$$\begin{aligned} \mathrm{Var}\left(z_i'\right) &= a^2\,\mathrm{Var}(z) + \frac{2a(1-a)}{na^2}\Sigma + \frac{(1-a)^2}{n}\,\mathrm{Var}(\bar{z}) \\ &= \left(1 + \frac{2(1-a)}{na} + \frac{(1-a)^2}{na^2}\right)\Sigma \end{aligned}$$

Note that

$$\begin{aligned}\frac{2(1-a)}{na}+\frac{(1-a)^2}{na^2} &= \frac{1}{n}\left(\left(\frac{1-a}{a}+1\right)^2-1\right)\\ &= \frac{1}{n}\left(\frac{1}{a^2}-1\right)\\ &= \frac{d}{n}\end{aligned}$$

so that

$$\mathrm{Var}\left(z_i^{'}\right)=\left(1+\frac{d}{n}\right)\Sigma$$

and $\mathrm{Var}\left(z_i^{'}\right)$ approaches $\Sigma$ asymptotically with increasing sample size $n$.

### 3.4.2 Statistics for Subpopulations of Masked Data

Now suppose that we want to consider the subpopulation $Z_s$ of the masked data $Z$ where the records have been obtained from the subpopulation $X_s$ of the unmasked data set $X$ by

$$z_s = x_s + \sqrt{d}y,$$

where $y$ is still random noise of mean 0 and variance $\Sigma$. We see that we still have, as in (1) in Kim,

$$\begin{aligned}\mathsf{E}[z_s] &= \mathsf{E}[x_s]+\sqrt{d}\mathsf{E}[y]\\ &= \mathsf{E}[x_s]\\ &= \mu_s\end{aligned}$$

but that the covariance is

$$\begin{aligned}\mathrm{Var}(z_s) &= \mathsf{E}\left[(z_s-\mu_s)(z_s-\mu_s)^T\right]\\ &= \mathsf{E}\left[\left((x_s-\mu_s)+\sqrt{d}y\right)\left((x_s-\mu_s)+\sqrt{d}y\right)^T\right]\\ &= \mathsf{E}\left[(x_s-\mu_s)(x_s-\mu_s)^T\right]+\mathsf{E}\left[\left(\sqrt{d}y\right)\left(\sqrt{d}y\right)^T\right]\\ &= \mathrm{Var}(x_s)+d\Sigma\end{aligned}$$

Thus if we want to recover the covariance of the original subpopulation covariance from the masked data, since

$$\mathrm{Var}\,(z) = (1+d)\,\Sigma,$$

we have that

$$\begin{aligned}\mathrm{Var}\,(x_s) &= \mathrm{Var}\,(z_s) - d\Sigma \\ &= \mathrm{Var}\,(z_s) - \frac{d}{1+d}\,\mathrm{Var}\,(z)\end{aligned}$$

as in (2) and (5) in Kim. The best estimates of these variances are given by the sample variances of $Z_s$ and $Z$ respectively.

### 3.4.3 Statistics for Subpopulations of Scaled Masked Data

If we choose the subpopulation from the scaled masked data

$$z' = az + (1-a)\,\bar{z}$$

where

$$a^2 = \frac{1}{1+d}$$

then the subpopulation has samples

$$z'_s = az_s + (1-a)\,\bar{z}.$$

We have that the subpopulation mean is given by

$$\begin{aligned}\mathsf{E}\,[z'_s] &= a\mathsf{E}\,[z_s] + (1-a)\,\mathsf{E}\,[\bar{z}] \\ &= a\mu_s + (1-a)\,\mu\end{aligned}$$

Thus from the masked scaled subpopulation sample mean

$$\bar{z}'_s = \frac{1}{|S|}\sum_{s\in S} z'_s$$

if we compute

$$\frac{1}{a}\bar{z}'_s - \frac{1-a}{a}\bar{z} = \sqrt{1+d}\bar{z}'_s - \left(\sqrt{1+d}-1\right)\bar{z} \tag{3.2}$$

we see that

$$\mathsf{E}\left[\frac{1}{a}\bar{z}'_s - \frac{1-a}{a}\bar{z}\right] = \mathsf{E}\left[z_s\right] = \mathsf{E}\left[x_s\right] = \mu_s$$

That is, (3.2) adjusts the scaled masked subpopulation mean to give an estimate of the raw data subpopulation mean, as in Kim (7). Likewise, to compute the variance, since

$$z'_s - \mathsf{E}\left[z'_s\right] = a\left(z_s - \mathsf{E}\left[z_s\right]\right) + (1-a)\left(\bar{z} - \mu\right)$$

to evaluate

$$\mathrm{Var}\left(z'_s\right) = \mathsf{E}\left[\left(z'_s - \mathsf{E}\left[z'_s\right]\right)\left(z'_s - \mathsf{E}\left[z'_s\right]\right)^T\right]$$

we have seen that

$$\begin{aligned}\mathrm{Var}\left(z_s\right) &= \mathsf{E}\left[\left(z_s - \mathsf{E}\left[z_s\right]\right)\left(z_s - \mathsf{E}\left[z_s\right]\right)^T\right] \\ &= \mathrm{Var}\left(x_s\right) + d\Sigma\end{aligned}$$

and

$$\begin{aligned}\mathrm{Var}\left(\bar{z}\right) &= \mathsf{E}\left[\left(\bar{z} - \mu\right)\left(\bar{z} - \mu\right)^T\right] \\ &= \frac{1+d}{n}\Sigma\end{aligned}$$

We further note that

$$\begin{aligned}\mathsf{E}\left[\left(z_s - \mathsf{E}\left[z_s\right]\right)\left(\bar{z} - \mu\right)^T\right] &= \mathsf{E}\left[\left(z_s - \mu + \mu - \mathsf{E}\left[z_s\right]\right)\left(\bar{z} - \mu\right)^T\right] \\ &= \mathsf{E}\left[\left(z_s - \mu\right)\left(\bar{z} - \mu\right)^T\right] + \left(\mu - \mathsf{E}\left[z_s\right]\right)\mathsf{E}\left[\left(\bar{z} - \mu\right)^T\right] \\ &= \frac{1+d}{n}\Sigma + 0\end{aligned}$$

so that altogether

$$\begin{aligned}\mathrm{Var}\left(z'_s\right) &= a^2\left(\mathrm{Var}\left(x_s\right) + d\Sigma\right) + 2a\left(1-a\right)\left(\frac{1+d}{n}\Sigma\right) + \left(1-a\right)^2\left(\frac{1+d}{n}\Sigma\right) \\ &= a^2\left(\mathrm{Var}\left(x_s\right) + d\Sigma\right) + \frac{d}{n}\Sigma \\ &= \frac{1}{1+d}\mathrm{Var}\left(x_s\right) + \left(\frac{d}{1+d} + \frac{d}{n}\right)\Sigma\end{aligned}$$

Thus

$$\mathrm{Var}\left(x_s\right) = (1+d)\,\mathrm{Var}\left(z'_s\right) - d\left(1+\frac{1+d}{n}\right)\Sigma$$

Since for the scaled data $z'$,

$$\mathrm{Var}\left(z'\right) = \Sigma,$$

we have that

$$\mathrm{Var}\left(x_s\right) = (1+d)\,\mathrm{Var}\left(z'_s\right) - d\left(1+\frac{1+d}{n}\right)\mathrm{Var}\left(z'\right)$$

and thus asyptotically

$$\mathrm{Var}\left(x_s\right) = (1+d)\,\mathrm{Var}\left(z'_s\right) - d\,\mathrm{Var}\left(z'\right).$$

Again, The best estimates of these variances are given by the sample variances of $Z_s$ and $Z$ respectively.

# Chapter 4

# Note on Additive Mixture Noise Data Masking

## 4.1 Introduction

We have considered some "information loss" scoring methods and some re-identification rate computations to arrive at some overall data masking scores when applied to two data sets. The information loss scores and data masking scoring methods are based on those of Domingo-Ferrer [Domingo-Ferrer], but we suggest some modifications. The data sets are the one used by Domingo-Ferrer in his study and some individual income statistics previously studied by Kim and Winkler [Kim95].

### 4.1.1 Information Loss Scores

Let $X$ be a $n \times m$ array of the original raw data and $Y$ be a $n \times m$ array of masked data. One information loss statistic proposed by Domingo-Ferrer to measure the change in the raw data is given by

$$\frac{1}{mn} \sum_{j=1}^{m} \sum_{i=1}^{n} \frac{|x_{ij} - y_{ij}|}{|x_{ij}|}.$$

This measure has some problems when the data set has values equal to 0, as is the case with the Kim-Winkler data, where for some variables (*e.g.* rental income, tax-exempt income) zero is the mode value. If one replaces the denominator by some constant when $x_{ij} = 0$, then the value of this expression

can be arbitrarily changed by the choice of this constant. Furthermore, this statistic tends to be several orders of magnitude larger that the other information loss statistics, and thus tends to swap all other comparisons. In order to reduce these problems, we have considered the modification

$$il1 = \frac{1}{mn} \sum_{j=1}^{m} \sum_{i=1}^{n} \frac{|x_{ij} - y_{ij}|}{0.5\left(|x_{ij}| + |y_{ij}|\right)}.$$

This does result in restricting the size of the statistic to $il1 \leq 2$. However, it still tends to dominate the other information loss statistics and it does not totally eliminate the problem of dividing by zero. While for the case of additive noise, it is unlikely that both $x_{ij} = 0$ and $y_{ij} = 0$, for other strategies such as data swapping in a file where a lot of the records have zero values, this could easily happen.

However, beyond these problems, by scaling the statistic by the individual data points, the resulting statistic is not very stable. Leaving aside zero data values, when the data values are very small, then small adjustments in the masked data produce large effects on the summary statistic. If we view our data set as independent samples from a common distribution, it is more stable to measure variations in the sample values by scaling them all by a value common to the variable. Thus if $X$ and $Y$ are independent random variables both with mean $\mu$ and variance $\sigma^2$, then the random variable $Z = X - Y$ has mean 0 and variance $2\sigma^2$. Hence a common scale for $Z$ would be its standard deviation $\sqrt{2}\sigma$. In our case, we can estimate that standard deviation with the sample standard deviation $S$. This motivates the proposed modification for the data perturbation information loss statistic given by

$$il1s = \frac{1}{mn} \sum_{j=1}^{m} \sum_{i=1}^{n} \frac{|x_{ij} - y_{ij}|}{\sqrt{2}S_j}.$$

This uses a common scale for all values of the same variable in the data set, the denominator is not zero unless the values of the variable are constant throughout the data set, and while the statistic does not have an a priori upper bound, the values in our empirical studies tend to be closer in magnitude to the other information loss statistics.

The other information loss statistics that we compute are the same as some of those suggested by Domingo-Ferrer. To measure the variation in

the sample means, we compute

$$il2 = \frac{1}{m}\sum_{j=1}^{m}\frac{|\bar{x}_j - \bar{y}_j|}{|\bar{x}_j|}.$$

For variations in the sample covariance matrix, we compute

$$il3 = \frac{2}{m\,(m+1)}\sum_{j=1}^{m}\sum_{k=1}^{j}\frac{\left|\mathrm{Cov}\,(X)_{jk} - \mathrm{Cov}\,(Y)_{jk}\right|}{\left|\mathrm{Cov}\,(X)_{jk}\right|}$$

for variations in the sample variances, we compute

$$il4 = \frac{1}{m}\sum_{j=1}^{m}\frac{\left|\mathrm{Cov}\,(X)_{jj} - \mathrm{Cov}\,(Y)_{jj}\right|}{\left|\mathrm{Cov}\,(X)_{jj}\right|}$$

and for variations in the sample correlation matrix, we compute

$$il5 = \frac{2}{m\,(m-1)}\sum_{j=1}^{m}\sum_{k=1}^{j-1}\left|\mathrm{Cor}\,(X)_{jk} - \mathrm{Cor}\,(Y)_{jk}\right|.$$

We wish to combine these information loss statistics into a summary information loss score. While it's not clear what sense it really makes to combine these numbers, and even if we do, its not clear what appropriate weighting we should give to them, in the absence of deeper insight, we just compute a straight average. However, we may choose which statistics we wish to include. As we have noted, the data perturbation measure $il1$ is somewhat numerically problematic. Moreover, for the purposes of data masking, it is not clear if one cares how much individual data records are perturbed as long as the overall statistical structure of the data set is preserved. Thus one information loss penalty score can be computed by leaving out $il1$ to get

$$s0 = \frac{il2 + il3 + il4 + il5}{4}.$$

On the other hand, one can leave it in to get

$$s1 = \frac{il1 + il2 + il3 + il4 + il5}{5}.$$

Another objection to combining all theses scores is that $il3$, $il4$, and $il5$ are redundant. With the covariance, variance, and correlation, if we know two of these things, then we know the third. Furthermore, the covariance score $il3$ is to a lesser degree subject to the same kind of scaling instability as found with $il1$, namely that the smallest values make the largest contributions to the score. In particular, we observe that the score tends to be dominated by those components corresponding to the smallest correlations. Thus as an alternative summary information loss statistic, we suggest using the rescaled data perturbation score and leaving out the covariance score to get

$$s2 = \frac{il1s + il2 + il4 + il5}{4}.$$

### 4.1.2 Additive Mixture Noise

For each of the following data sets $X$, we produced a noise data set $Y$ with a mixture distribution designed to have 0 mean and the same covariance as $X$. The masked data set $Z$ is then computed by

$$Z = X + dY$$

where we can vary the parameter $d$ to adjust the concentration of noise. We do this by simulating an uncorrelated mixture noise random variable $W$ with mean 0 and covariance $I$ and then multiplying

$$Y = \Sigma^{\frac{1}{2}} W \tag{4.1}$$

where $\Sigma^{\frac{1}{2}}$ is a square root factor of the covariance matrix $\Sigma = \mathrm{Cov}\,(X)$ where

$$\Sigma = \Sigma^{\frac{1}{2}} \left(\Sigma^{\frac{1}{2}}\right)^T.$$

The data set $W$ is generated from independent random samples of a mixture distribution with density function

$$f\,(x) = \frac{1}{2} N\left(x; \sqrt{1-\sigma^2}, \sigma^2\right) + \frac{1}{2} N\left(x; -\sqrt{1-\sigma^2}, \sigma^2\right)$$

where $N\,(x; \mu, \sigma^2)$ is the normal density function with mean $\mu$ and variance $\sigma^2$. We use a parameter value of $\sigma^2 = 0.025$. We consider masked data sets generated using $d = 0.01, 0.05, 0.10, 0.20$.

The resulting random variable has mean $\mu$ and covariance $(1+d)\,\Sigma$. We can get a scaled data set with the same mean and covariance $\Sigma$ by rescaling

$$\begin{aligned} Z_{sc} &= \frac{Z-\mu}{\sqrt{1+d}} + \mu \\ &= \frac{1}{\sqrt{1+d}} Z + \left(1 - \frac{1}{\sqrt{1+d}}\right)\mu \end{aligned}$$

We thus consider eight masked data sets computed as unscaled and scaled versions from the four choices of the noise proportion $d$, and compute information loss and re-identification rates for each masked data set.

## 4.2 Domingo Data Statistics

The Domingo data set consists of 1080 records from which we have masked 13 real variables. We can observe here how the information loss scores increase with increasing noise level $d$. Rescaling the masked data has no mathematical effect on the mean and correlation. Since the rescaling somewhat contracts the data, there tends to be some decrease in the data perturbation scores. The effects of rescaling are most significant in the covariance and especially the variance scores.

**Domingo Data Information Loss Statistics**

| | il1 | il1s | il2 | il3 | il4 | il5 | s0 | s1 | s2 |
|---|---|---|---|---|---|---|---|---|---|
| mixadd01 | 0.2041 | 0.0628 | 0.0019 | 0.0281 | 0.0115 | 0.0017 | 0.0108 | 0.0495 | 0.0195 |
| mixadd05 | 0.3257 | 0.1404 | 0.0041 | 0.0876 | 0.0533 | 0.0037 | 0.0372 | 0.0949 | 0.0504 |
| mixadd10 | 0.3984 | 0.1985 | 0.0059 | 0.1520 | 0.1047 | 0.0051 | 0.0669 | 0.1332 | 0.0786 |
| mixadd20 | 0.4886 | 0.2807 | 0.0083 | 0.2731 | 0.2066 | 0.0069 | 0.1237 | 0.1887 | 0.1256 |
| scalmixadd01 | 0.2023 | 0.0625 | 0.0019 | 0.0213 | 0.0033 | 0.0017 | 0.0071 | 0.0461 | 0.0174 |
| scalmixadd05 | 0.3158 | 0.1372 | 0.0041 | 0.0477 | 0.0073 | 0.0037 | 0.0157 | 0.0757 | 0.0381 |
| scalmixadd10 | 0.3785 | 0.1900 | 0.0059 | 0.0669 | 0.0101 | 0.0051 | 0.0220 | 0.0933 | 0.0528 |
| scalmixadd20 | 0.4485 | 0.2584 | 0.0083 | 0.0945 | 0.0135 | 0.0069 | 0.0308 | 0.1143 | 0.0718 |

The matching software has two methods for measuring agreement between two real values. In either case, it interpolates between the agreement weight and the disagreement weight. For the $d$ method, the slope of the interpolation line is given by

$$\frac{|x_i - z_i|}{\max\left(|x_i|, 0.1\right)}$$

and for the $l$ method, the slope of the interpolation line is given by

$$\frac{|\log x_i - \log z_i|}{\max(\log x_i, 0.1)}.$$

In this case, we see that when the perturbations are small, the $d$ method does a little better than the $l$ method. However, when the perturbations get large, the $l$ method is better able to see past moderate perturbations to large values.

The re-identification software produces a list of linked pairs in decreasing matching weight. For this small data set, the re-identification rate is computed as the total number of correctly linked pairs out of the total number of records in the data file. This is a rather optimistic re-identification score since most of the true matches are mixed among many false matches, and an analyst would probably have difficulty picking many of them out. In any event, we can see that for this data set with so few records and so many matching variables, a 1% noise level does not provide adequate masking, but the re-identification rate drops off rapidly with increasing noise level.

**Domingo Data Re-identification Rates**

| | $d$ metric | $l$ metric |
|---|---|---|
| mixadd01 | 0.7667 | 0.7176 |
| mixadd05 | 0.1482 | 0.3556 |
| mixadd10 | 0.0574 | 0.2194 |
| mixadd20 | 0.0139 | 0.1009 |
| scalmixadd01 | 0.7704 | 0.7370 |
| scalmixadd05 | 0.1602 | 0.3537 |
| scalmixadd10 | 0.0648 | 0.2417 |
| scalmixadd20 | 0.0269 | 0.1241 |

For an overall data masking score, we combine the information loss score with the re-identification score. Since we computed three data loss scores, we compute three overall scores:

$$\begin{aligned} Ascore &= 100\left(\frac{S0 + reid}{2}\right) \\ Dscore &= 100\left(\frac{S1 + reid}{2}\right) \\ Sscore &= 100\left(\frac{S2 + reid}{2}\right) \end{aligned}$$

**Domingo Data Scoring Metrics**

| | $d$ Metric | | | $l$ Metric | | |
|---|---|---|---|---|---|---|
| | Ascore | Dscore | Sscore | Ascore | Dscore | Sscore |
| mixadd01 | 38.88 | 40.81 | 39.31 | 36.42 | 38.36 | 36.86 |
| mixadd05 | 9.27 | 12.16 | 9.93 | 19.64 | 22.53 | 20.30 |
| mixadd10 | 6.22 | 9.53 | 6.80 | 14.32 | 17.63 | 14.90 |
| mixadd20 | 6.88 | 10.13 | 6.98 | 11.23 | 14.48 | 11.33 |
| scalmixadd01 | 38.95 | 40.90 | 39.46 | 37.21 | 39.16 | 37.72 |
| scalmixadd05 | 8.94 | 11.94 | 10.06 | 18.47 | 21.47 | 19.59 |
| scalmixadd10 | 4.43 | 8.00 | 5.97 | 13.19 | 16.75 | 14.73 |
| scalmixadd20 | 2.89 | 7.06 | 4.94 | 7.75 | 11.92 | 9.80 |

## 4.3 Kim-Winkler Data Statistics

The Kim-Winkler data consists of 59,315 records each containing 11 real variables for income data. Here we show the information loss statistics for our additive mixed noise masking for the whole data set. We note that the $il1$ date perturbation statistic tends to higher than that for the Domingo data, possibly due to the large number of zero entries in the Kim-Winkler data, whereas the $il1s$ data perturbation metric is about the same as for the Domingo data. In general the scaled data tends to get better results reducing the covariance measures $il3, il4$ than in the Domingo data case.

**Kim-Winkler Data Information Loss Statistics, 11 Variables**

| | il1 | il1s | il2 | il3 | il4 | il5 | s0 | s1 | s2 |
|---|---|---|---|---|---|---|---|---|---|
| mixadd01 | 1.1649 | 0.0604 | 0.0022 | 0.0142 | 0.0102 | 0.0004 | 0.0068 | 0.2384 | 0.0183 |
| mixadd05 | 1.3081 | 0.1350 | 0.0049 | 0.0563 | 0.0505 | 0.0009 | 0.0282 | 0.2841 | 0.0478 |
| mixadd10 | 1.3807 | 0.1909 | 0.0070 | 0.1056 | 0.1007 | 0.0013 | 0.0537 | 0.3191 | 0.0750 |
| mixadd20 | 1.4565 | 0.2700 | 0.0099 | 0.2013 | 0.2010 | 0.0017 | 0.1035 | 0.3741 | 0.1207 |
| scalmixadd01 | 1.1633 | 0.0601 | 0.0022 | 0.0081 | 0.0007 | 0.0004 | 0.0029 | 0.2349 | 0.0159 |
| scalmixadd05 | 1.3016 | 0.1321 | 0.0049 | 0.0180 | 0.0015 | 0.0009 | 0.0063 | 0.2654 | 0.0349 |
| scalmixadd10 | 1.3695 | 0.1830 | 0.0070 | 0.0248 | 0.0020 | 0.0013 | 0.0088 | 0.2809 | 0.0483 |
| scalmixadd20 | 1.4375 | 0.2488 | 0.0099 | 0.0332 | 0.0026 | 0.0017 | 0.0119 | 0.2970 | 0.0658 |

For our re-identification, we only used eight of the income variables, so we computed the information loss scores based on just these eight variables.

They show a generally slight increase over the eleven variable scores.

**Kim-Winkler Data Information Loss Statistics, 8 Variables**

| | il1 | il1s | il2 | il3 | il4 | il5 | s0 | s1 | s2 |
|---|---|---|---|---|---|---|---|---|---|
| mixadd01 | 1.3037 | 0.0612 | 0.0026 | 0.0118 | 0.0102 | 0.0004 | 0.0063 | 0.2657 | 0.0186 |
| mixadd05 | 1.4430 | 0.1369 | 0.0059 | 0.0522 | 0.0505 | 0.0009 | 0.0274 | 0.3105 | 0.0486 |
| mixadd10 | 1.5121 | 0.1936 | 0.0083 | 0.1014 | 0.1008 | 0.0012 | 0.0529 | 0.3448 | 0.0760 |
| mixadd20 | 1.5823 | 0.2737 | 0.0117 | 0.1986 | 0.2012 | 0.0015 | 0.1033 | 0.3971 | 0.1220 |
| scalmixadd01 | 1.3023 | 0.0609 | 0.0026 | 0.0048 | 0.0008 | 0.0004 | 0.0022 | 0.2622 | 0.0162 |
| scalmixadd05 | 1.4374 | 0.1339 | 0.0059 | 0.0108 | 0.0018 | 0.0009 | 0.0049 | 0.2914 | 0.0369 |
| scalmixadd10 | 1.5028 | 0.1853 | 0.0083 | 0.0150 | 0.0024 | 0.0012 | 0.0067 | 0.3059 | 0.0493 |
| scalmixadd20 | 1.5669 | 0.2516 | 0.0117 | 0.0203 | 0.0031 | 0.0015 | 0.0092 | 0.3207 | 0.0670 |

For the re-identification scores, we computed the proportion of correctly linked pairs out of the total number of records, as in the case of the Domingo data. In this case, we only compute the results using the $l$ interpolation metric, since it is much more effective on this data. However, reporting the total number of correct matches in the full link file is probably even more misleadingly optimistic than in the Domingo data case. For the Domingo data, the true matches tend to be distributed throughout the link file. As the noise level of the masking increases, this distribution becomes more sparse and random. In the case of this data set, there are twenty or so records that are extreme outliers with one or more income categories much higher than the values for the mass of the records. Many of these records fail to be successfully masked from the re-identification through most noise levels, especially using the $l$ metric. Thus there are always several clearly true matches at the top of the match-weight sorted link file. However, as the matching weights decrease, the proportion of true matches rapidly drops off as we include more and more false links in with a decreasing number of true matches. Thus is seems reasonable to cut off the count of true matches at some point, since beyond this point, any true matches will only appear sporadically among the preponderance of false matches and are unlikely to be discerned by the analyst. Here we choose a rather low cutoff point of 20%. This means that at this point, the number of linked pairs at this matching weight or higher contain 20% true matches and 80% false links. Below this

point, the true matches become much rarer.

**Kim-Winkler Data Re-identification Rates, $l$ Metric**

| | Total File Matches | 20% Zone |
|---|---|---|
| mixadd01 | 0.0841 | 0.0096 |
| mixadd05 | 0.0346 | 0.0027 |
| mixadd10 | 0.0240 | 0.0018 |
| mixadd20 | 0.0149 | 0.0011 |
| scalmixadd01 | 0.0844 | 0.0098 |
| scalmixadd05 | 0.0355 | 0.0031 |
| scalmixadd10 | 0.0249 | 0.0022 |
| scalmixadd20 | 0.0174 | 0.0016 |

Here are the overall data masking scores for the Kim-Winkler data. The data masking tends to be more effective here, especially at lower additive noise levels. Again inclusion of the $il1$ data perturbation score tends to dominate and obscure the rest of the results.

**Kim-Winker Data Scoring Metrics**

| | Full File Matches | | | 20% Zone Matches | | |
|---|---|---|---|---|---|---|
| | Ascore | Dscore | Sscore | Ascore | Dscore | Sscore |
| mixadd01 | 4.52 | 17.49 | 5.14 | 0.80 | 13.77 | 1.41 |
| mixadd05 | 3.10 | 17.26 | 4.16 | 1.51 | 15.66 | 2.57 |
| mixadd10 | 3.85 | 18.44 | 5.00 | 2.74 | 17.33 | 3.84 |
| mixadd20 | 5.92 | 19.45 | 6.78 | 5.23 | 18.76 | 6.09 |
| scalmixadd01 | 4.33 | 17.33 | 5.03 | 0.60 | 13.60 | 1.30 |
| scalmixadd05 | 2.02 | 16.35 | 3.62 | 0.40 | 15.45 | 2.00 |
| scalmixadd10 | 1.58 | 16.54 | 3.71 | 0.45 | 15.41 | 2.58 |
| scalmixadd20 | 1.33 | 16.91 | 4.22 | 0.49 | 16.12 | 3.43 |

### 4.3.1 Subpopulation Information Loss Statistics

The additive noise procedures are supposed to preserve means and covariances on arbitrary subpopulations, at least when these statistics are properly corrected, as discussed by Kim. Here we compute the information loss scores for two subpopulations. We see that even for the corrected means and covariances, there are still generally somewhat higher scores than for the full data set. We especially note that the scaled data sets fail to recover the original data covariance and variance values as well.

**S4 Return Type Subpopulation Information Loss, 8 Variables, 5885 Records**

| | il1 | il1s | il2 | il3 | il4 | il5 | s0 | s1 | s2 |
|---|---|---|---|---|---|---|---|---|---|
| mixadd01 | 1.3992 | 0.2457 | 0.0572 | 0.1008 | 0.0082 | 0.0056 | 0.0430 | 0.3142 | 0.0792 |
| mixadd05 | 1.5591 | 0.5495 | 0.1280 | 0.2319 | 0.0236 | 0.0136 | 0.0993 | 0.3912 | 0.1787 |
| mixadd10 | 1.6312 | 0.7771 | 0.1810 | 0.3419 | 0.0396 | 0.0207 | 0.1458 | 0.4429 | 0.2546 |
| mixadd20 | 1.6970 | 1.0989 | 0.2559 | 0.5390 | 0.0692 | 0.0328 | 0.2242 | 0.5188 | 0.3642 |
| scalmixadd01 | 1.3974 | 0.2445 | 0.0572 | 0.1297 | 0.0082 | 0.0056 | 0.0502 | 0.3196 | 0.0789 |
| scalmixadd05 | 1.5522 | 0.5362 | 0.1280 | 0.2981 | 0.0236 | 0.0136 | 0.1158 | 0.4031 | 0.1754 |
| scalmixadd10 | 1.6206 | 0.7411 | 0.1810 | 0.4397 | 0.0396 | 0.0207 | 0.1703 | 0.4603 | 0.2456 |
| scalmixadd20 | 1.6813 | 1.0039 | 0.2559 | 0.6931 | 0.0692 | 0.0328 | 0.2628 | 0.5465 | 0.3405 |

Here we see slightly better information loss scores for a slightly larger subpopulation.

**Schedule C Subpopulation Information Loss, 8 Variables, 7819 Records**

| | il1 | il1s | il2 | il3 | il4 | il5 | s0 | s1 | s2 |
|---|---|---|---|---|---|---|---|---|---|
| mixadd01 | 1.4096 | 0.2201 | 0.0119 | 0.0651 | 0.0092 | 0.0049 | 0.0228 | 0.2884 | 0.0615 |
| mixadd05 | 1.5597 | 0.4921 | 0.0267 | 0.1753 | 0.0258 | 0.0120 | 0.0600 | 0.3599 | 0.1392 |
| mixadd10 | 1.6266 | 0.6960 | 0.0378 | 0.2991 | 0.0430 | 0.0185 | 0.0996 | 0.4050 | 0.1988 |
| mixadd20 | 1.6884 | 0.9843 | 0.0535 | 0.5306 | 0.0752 | 0.0299 | 0.1723 | 0.4755 | 0.2857 |
| scalmixadd01 | 1.4076 | 0.2190 | 0.0120 | 0.0838 | 0.0092 | 0.0049 | 0.0275 | 0.3035 | 0.0613 |
| scalmixadd05 | 1.5534 | 0.4803 | 0.0267 | 0.2253 | 0.0258 | 0.0120 | 0.0725 | 0.3686 | 0.1362 |
| scalmixadd10 | 1.6170 | 0.6639 | 0.0378 | 0.3845 | 0.0430 | 0.0185 | 0.1210 | 0.4202 | 0.1908 |
| scalmixadd20 | 1.6741 | 0.8994 | 0.0535 | 0.6822 | 0.0752 | 0.0299 | 0.2102 | 0.5030 | 0.2645 |

## 4.4 Summary Mixture Noise Statistics

To create the colored noise with the desired covariance, one multiplies the simulated white noise by a matrix that is the square root of the raw data covariance matrix $\Sigma$, as in (4.1). However, the square root of a positive definite symmetric matrix is not unique. One way to compute a square root is to compute the Cholesky factor, which is an upper triangular matrix $C$ with the property that $C^T C = \Sigma$. The computation of the Cholesky factor has a straightforward algorithm. Another approach is to compute the eigenvalues and eigenvectors of $\Sigma$ to get a decomposition

$$\Sigma = UDU^T$$

where $D$ is a diagonal matrix of (positive) eigenvalues and $U$ is an orthogonal matrix of eigenvectors. Thus one can obtain a square root by computing the matrix $UD^{\frac{1}{2}}$ where $D^{\frac{1}{2}}$ is the diagonal matrix of the square roots of the eigenvalues. To test to see if there were any numerical consequences of the matrix square root computation, we used both forms and compared the information loss scores. That is, for each square root form, we take the colored noise matrix

$$Y = \Sigma^{\frac{1}{2}} W$$

and compare the means of the variables to 0 and the covariance of $Y$ to the covariance of the raw data matrix $X$. Based on these examples, the Cholesky factor tends to do a bit better on the mean and somewhat worse on the variance and covariance. Both methods do worse on the Domingo data than on the Kim-Winkler data. This may partly be directly attributable to the relative sizes of the data sets, but more specifically, the eigenvalues of the covariance matrices reveal than the Domingo data covariance is much worse conditioned (more nearly singular) which probably makes any matrix square root calculation numerically less accurate.

**Domingo Data, 13 Variables, 1080 Records**

| | Mean | Variance | Covariance | Correlation |
|---|---|---|---|---|
| Eigenvalue | 0.0185 | 0.0154 | 0.4109 | 0.0191 |
| Cholesky | 0.0178 | 0.0450 | 0.5116 | 0.0181 |

**Kim-Winkler Data, 11 Variables, 59,315 Records**

| | Mean | Variance | Covariance | Correlation |
|---|---|---|---|---|
| Eigenvalue | 0.0221 | 0.0038 | 0.0432 | 0.0021 |
| Cholesky | 0.0102 | 0.0019 | 0.0526 | 0.0024 |

**Kim-Winkler Data, 8 Variables, 59,315 Records**

| | Mean | Variance | Covariance | Correlation |
|---|---|---|---|---|
| Eigenvalue | 0.0336 | 0.0027 | 0.0303 | 0.0020 |
| Cholesky | 0.0215 | 0.0023 | 0.0385 | 0.0022 |

We note one more time that the covariance information loss score $il3$ is the least stable of these summary information loss scores. The terms contributing the largest proportional covariance error terms are the terms that correspond to terms of the smallest correlation. For example, in the Domingo data, the largest proportional covariance error terms come from the variable

pairs $(1,4)$ and $(1,7)$. The Domingo data covariance for these terms is

$$\begin{aligned} \text{Cov}\,(1,4) &= -1774438 \\ \text{Cov}\,(1,7) &= 8246860.9 \end{aligned}$$

while the eigenvalue factorization produces corresponding covariances

$$\begin{aligned} \text{Cov}\,(1,4) &= 16779788 \\ \text{Cov}\,(1,7) &= 52443168 \end{aligned}$$

which produce substantial error terms of

$$\begin{aligned} \frac{|-1774438 - 16779788|}{|-1774438|} &\doteq 10.4537 \\ \frac{|8246860.9 - 52443168|}{8246860.9} &\doteq 5.3592 \end{aligned}$$

and the Cholesky factorization produces corresponding covariances

$$\begin{aligned} \text{Cov}\,(1,4) &= 245566307 \\ \text{Cov}\,(1,7) &= 110528796 \end{aligned}$$

which produce error terms

$$\begin{aligned} \frac{|-1774438 - 245566307|}{|-1774438|} &\doteq 14.8407 \\ \frac{|8246860.9 - 110528796|}{|8246860.9|} &\doteq 12.4025 \end{aligned}$$

However, the correlations for these variables in the original data are

$$\begin{aligned} \text{Cor}\,(1,4) &= -0.003577 \\ \text{Cor}\,(1,7) &= 0.003841 \end{aligned}$$

while the correlations from the eigenvalue noise data are

$$\begin{aligned} \text{Cor}\,(1,4) &= 0.0335967 \\ \text{Cor}\,(1,7) &= 0.0242244 \end{aligned}$$

and the correlations from the Cholesky noise data are

$$\begin{aligned} \mathrm{Cor}\,(1,4) &= 0.048866 \\ \mathrm{Cor}\,(1,7) &= 0.051542 \end{aligned}$$

While these correlation terms contribute to the total correlation error estimation, they are not out of proportion with several other correlation error terms. On the other hand these covariance error terms substantially increase the total covariance error estimate. The problem again is that the scaling factor varies with the term. The smaller covariance terms that correspond to the smallest correlations are used to scale the covariance errors, resulting in a high proportion value.

# Chapter 5

# Second Order Information Loss Measures

## 5.1 Information Loss Measures

We wish to consider alternative measures to the Domingo-Ferrer information loss measures for covariance and correlation. If $\Sigma_X = \left(s_{ij}^X\right)$ is the (sample) variance/covariance matrix for the raw data set and $\Sigma_Z = \left(s_{ij}^Z\right)$ is the sample variance/covariance matrix for the masked data, then the Domingo-Ferrer variance information loss measure is given by

$$\frac{1}{m}\sum_{i=i}^{m}\frac{\left|s_{ii}^X - s_{ii}^Z\right|}{s_{ii}^X} \tag{5.1}$$

and the covariance information loss measure is given by

$$\frac{1}{m\left(m+1\right)}\sum_{i=1}^{m}\sum_{j=i}^{m}\frac{\left|s_{ij}^X - s_{ij}^Z\right|}{\left|s_{ij}^X\right|}. \tag{5.2}$$

If we let $S = \text{diag}\left(s_{ii}\right)$, the diagonal matrix consisting of the variances, then the correlation matrix $C = \left(\rho_{ij}\right)$ is given by

$$C = S^{-\frac{1}{2}}\Sigma S^{-\frac{1}{2}}$$

and the correlation information loss measure is given by

$$\frac{1}{m\left(m-1\right)}\sum_{i=1}^{m}\sum_{j=i+1}^{m}\left|\rho_{ij}^X - \rho_{ij}^Z\right|. \tag{5.3}$$

We have noted that these three measures are redundant and that the covariance matrix one is the poorest scaled.

We wish to consider alternative measures that may be more intrinsic or stable. We explore using the eigenvalues and eigenvectors of the matrices to form alternative information loss measures. Since the covariance matrix $\Sigma$ is symmetric positive definite, we can write $\Sigma$ in the form

$$\Sigma = UDU^T$$

where $D$ is a diagonal matrix of positive eigenvalues and $U$ is an orthogonal matrix where the columns are the corresponding eigenvectors. We may assume that the eigenvalues appear in order of decreasing magnitude. Comparably to the variance information loss measure, an alternative change-of-scale information loss measure can be given by the average proportional eigenvalue change

$$\frac{1}{m}\sum_{i=1}^{m}\frac{\left|d_i^X - d_i^Z\right|}{d_i^X}. \tag{5.4}$$

Geometrically, for any choice of positive constant $\kappa$, the set of points $x$ satisfying

$$x^T\Sigma x = \kappa$$

for positive definite symmetric $\Sigma$ is an ellipsoid. The eigenvalues of $\Sigma$ determine the length of the axes of the ellipsoid and the eigenvectors of $\Sigma$ determine the direction of these axes. The eigenvalue information loss measure measures the scale change for the ellipsoids for $\Sigma_X$ and $\Sigma_Z$. To determine the extent of orientation change for the ellipsoids, we can measure the angle change of the eigenvectors. For the corresponding unit column vectors of $U_X$ and $U_Z$, we have that

$$0 \leq \arccos\left(\left(u_i^X\right)^T u_i^Z\right) \leq \pi$$

so that we can have an eigenvector orientation information loss measure

$$\frac{1}{m\pi}\sum_{i=1}^{m}\arccos\left(\left(u_i^X\right)^T u_i^Z\right) \tag{5.5}$$

where we adjust the scale to be between 0 and 1.

| Noise Level | Variance | Covariance | Eigenvalues | Eigenvectors |
|---|---|---|---|---|
| 1% | 0.010041 | 0.015799 | 0.010103 | 0.000651 |
| 5% | 0.050091 | 0.060726 | 0.050228 | 0.001404 |
| 10% | 0.100129 | 0.114940 | 0.100321 | 0.001900 |
| 20% | 0.200182 | 0.221128 | 0.200450 | 0.002469 |

Table 5.1: Mixture Noise Covariance Information Loss

## 5.2 Eigenvalue Results

We apply these eigenvalue information loss measures to the Kim-Winkler data. In Table 5.1, we compare the Domingo-Ferrer information loss measures for variance (5.1) and covariance (5.2) with the information loss measures for the eigenvalues (5.4) and eigenvectors (5.5) for the variance/covariance matrices $\Sigma_X$ and $\Sigma_Z$ where the masked data set $Z$ has been generated by additive mixture noise

$$Z = X + \sqrt{d}Y$$

where $Y$ is random noise with a mixture distribution with mean 0 and covariance $\Sigma_X$ for different noise levels $d = 0.01, 0.05, 0.1, 0.2$. Hence we expect $Z$ to have covariance

$$\Sigma_Z = (1 + d)\,\Sigma_X.$$

We see that the variance (5.1) and eigenvalue (5.4) information loss measures give comparable results, reflecting the expected proportion of inflation of the scale of the matrix. The covariance information loss measure (5.2) trends to be somewhat inflated, resulting from variable scaling for differently correlated off-diagonal terms. The eigenvector information loss measure (5.5) is smaller and shows less percentage change from one noise level to the next than does the other measures.

In Table 5.2, we compare the correlation information loss measure (5.3) with the eigenvalue and eigenvector information loss measures computed from the correlation matrices $C_X$ and $C_Z$. Here all three of the information loss measures are fairly comparable. By using the well-scaled correlation matrices, the quite different information loss measures give pretty consistent results.

| Noise Level | Correlation | Eigenvalues | Eigenvectors |
|---|---|---|---|
| 1% | 0.000424 | 0.000435 | 0.000503 |
| 5% | 0.000911 | 0.000935 | 0.001082 |
| 10% | 0.001230 | 0.001262 | 0.001461 |
| 20% | 0.001594 | 0.001636 | 0.001895 |

Table 5.2: Mixture Noise Correlation Information Loss

| Noise Level | Variance | Covariance | Eigenvalues | Eigenvectors |
|---|---|---|---|---|
| 1% | 0.000532 | 0.008533 | 0.000576 | 0.000651 |
| 5% | 0.001144 | 0.018353 | 0.001236 | 0.001404 |
| 10% | 0.001545 | 0.024777 | 0.001667 | 0.001900 |
| 20% | 0.002002 | 0.032120 | 0.002159 | 0.002469 |

Table 5.3: Scaled Mixture Noise Covariance Information Loss

In Table 5.3, we compare information loss measures between the covariance matrix $\Sigma_X$ and the covariance matrix $\Sigma_{Z'}$ where $Z'$ is the scaled masked data

$$Z' = \frac{1}{\sqrt{1+d}} Z + \left(1 - \frac{1}{\sqrt{1+d}}\right) \mu \mathbf{1}^T$$

where $\mu$ is the (sample) mean of the variables of $Z$ and $\mathbf{1}$ is a vector of $n$ 1's. That is, $Z'$ has been scaled to have the same mean as $X$ and also the same covariance matrix $\Sigma$. This rescaling to make $\Sigma_{Z'} \approx \Sigma_X$ is seen in the two scale information loss measures, variance (5.1) and eigenvalue (5.4), compared with the analogous values in Table 5.1. As expected, the values here are much smaller than those in Table 5.1 and they are still fairly close to each other. Here we see more dramatically that the covariance measure (5.2) has failed to respond proportionally to the rescaling. On the other hand, since the eigenvector measure (5.5) is a measure of orientation not scale, it has remained unchanged from Table 5.1 by the rescaling of the masked data.

In Table 5.4, we compare information-loss measures for the correlation matrix of the scaled masked data. We see that the results are the same as in Table 5.2 because the centered data only differ by a scale factor and this scale factor is eliminated in the correlation matrix.

| Noise Level | Correlation | Eigenvalues | Eigenvectors |
|---|---|---|---|
| 1% | 0.000424 | 0.000435 | 0.000503 |
| 5% | 0.000911 | 0.000935 | 0.001081 |
| 10% | 0.001230 | 0.001262 | 0.001461 |
| 20% | 0.001594 | 0.001636 | 0.001895 |

Table 5.4: Scaled Mixture Noise Correlation Information Loss

## 5.3 Conclusions

If we are trying to summarize changes between the second order statistics of the raw and masked data sets in the form of some scalar measurements, there seem to be two main dimensions to account for: scale and orientation. The variance information loss measure (5.1) measures changes in scale. Another scale measure is the covariance eigenvalue measure (5.4). We are somewhat reassured by the empirical results that these two measures give quite similar results, with the variance formula being simpler to compute. For a measure of the orientation or variable interrelation change, it is interesting to note that all of the other above scalar measures give similar results with the exception of the covariance measure (5.2). Thus it appears that the correlation information loss measure (5.3) is a well-scaled measure of this orientation change and is more straightforward to compute that the eigenvector or eigenvalues measures. We note again that the covariance measure (5.2) seems to be both redundant to these other two and less stable due to scaling problems.

# Chapter 6

# Some Possible Modifications and Extensions for Additive Mixture Noise Application for Data Masking

In reviewing Ramesh Dandekar's [Dandekar] and Ruth Brand's [Brand] papers, we found a few techniques that we could try to appropriate or adapt to additive noise data masking. We can easily use a transformation to improve the whiteness of the additive noise. We might want to consider transformations to approximate the distributions of the data.

## 6.1 Improving Noise Statistics

In the somewhat different context of target rank correlation matrices, Dandekar uses a transformation to remove correlations. By doing the analogous transformation, we can get the additive noise to conform more closely to theoretical standards.

Let $X$ be an $n \times m$ data array,

$$X = \begin{pmatrix} x_1^T \\ x_2^T \\ \vdots \\ x_n^T \end{pmatrix}$$

which we view as $n$ independent samples of a random vector $x$, which has an unknown mean $\mu$ and covariance $\Sigma$, which we approximate by the sample mean

$$\mu_X = \frac{1}{n} \sum_{i=1}^{n} x_i$$

and sample covariance

$$\Sigma_X = \frac{1}{n} \sum_{i=1}^{n} (x_i - \mu_X) (x_i - \mu_X)^T .$$

We wish to produce a masked data set $Z$ by using additive noise

$$Z = X + dY$$

where $Y$ is a masking array with mean 0 and covariance $\Sigma_X$. If we do this, the resulting masked data set $Z$ has mean $\mu_X$ and covariance $(1 + d) \Sigma_X$. To produce the masking array $Y$, we compute an array $W$

$$W = \begin{pmatrix} w_1^T \\ w_2^T \\ \vdots \\ w_n^T \end{pmatrix}$$

which is formed by sampling a random vector $w$ with mean 0 and covariance $I$, then multiplying by a square root $\Sigma_X^{\frac{1}{2}}$ of the covariance matrix $\Sigma_X$,

$$\Sigma_X^{\frac{1}{2}} \left( \Sigma_X^{\frac{1}{2}} \right)^T = \Sigma_X,$$

to get

$$Y = W \left( \Sigma_X^{\frac{1}{2}} \right)^T .$$

When we use our mixture model for random noise, the resulting sample array $W$ does not have exactly 0 mean and $I$ covariance, nor can it be expected to have. However, we can adjust the noise array so that it conforms more exactly to the theoretical standards. If our given sample noise array $W$ has sample mean $\mu_W$ and sample covariance $\Sigma_W$, then we can "bleach"

the noise by subtracting the mean and transforming the covariance to get an improved, whiter white noise array

$$W' = (W - \mu_W)\left(\Sigma_W^{\frac{1}{2}}\right)^{-T}$$

where $\Sigma_W^{\frac{1}{2}}$ is a square root of the sample covariance

$$\Sigma_W^{\frac{1}{2}}\left(\Sigma_W^{\frac{1}{2}}\right)^T = \Sigma_W.$$

(Note that $W - \mu_W$ is sloppy notation for subtracting the sample mean row vector $\mu_W^T$ from each row $w_i^T$ of the matrix $W$.) The resulting array $W'$ will have more nearly mean 0 and covariance $I$, so that the colored noise array

$$Y = W'\left(\Sigma_X^{\frac{1}{2}}\right)^T$$

should also have more nearly mean 0 and sample covariance $\Sigma_X$. Since $W$ already has mean near 0 and covariance near $I$, the changes are fairly minor and there should be no problems with matrix conditioning or positive definiteness in computing the square root.

As a computational note, it the square root is computed as a Cholesky triangular factor, then multiplying by the inverse square root would be best accomplished by back substitution. If one uses an eigenvalue decomposition,

$$\Sigma = UDU^T$$

where $D$ is the diagonal matrix of eigenvalues and $U$ is the orthogonal matrix of eigenvectors, then the inverse square root is obtained from multiplying

$$\Sigma^{-\frac{1}{2}} = D^{-\frac{1}{2}}U^T$$

where $D^{-\frac{1}{2}}$ is the diagonal matrix of the reciprocals of the square roots of the eigenvalues of $\Sigma$ and

$$\left(\Sigma^{-\frac{1}{2}}\right)^T\Sigma^{-\frac{1}{2}} = \Sigma^{-1}.$$

A preliminary test shows improvement in the Domingo-Ferrer information loss statistics for the colored noise array $Y$ compared to the original data $X$. In this particular run, if we derive $Y$ from our standard mixture noise

model, the variance and correlation measures are $O(10^{-3})$ and the covariance measure is $O(10^{-2})$. When we compute the colored noise array $Y$ from the whitened noise array $W'$, the Domingo-Ferrer information loss measure for correlation is $O(10^{-15})$ and for variance and covariance are $O(10^{-14})$. These differences probably represent basic machine floating point error. Thus we can make the additive noise array $Y$ has nearly perfect first and second order statistics. It remains to be seen if this in turn produces a similar improvement in the information loss statistics for the masked data set $Z$. More interestingly, we should see if there is an improvement in the information loss statistics for subpopulations. We can also check what effect the whitened noise has on re-identification, but since the corrections to the original white noise array $W$ are presumably small, we expect the re-identification effects to be slight.

### 6.1.1 Empirical Results

We tried the noise whitening transformation of the Kim-Winkler data to see what effect it has on information loss measures. As intended from the direct calculation, the transformation produces white noise $W'$ with virtually zero mean and identity covariance. The SAS output yields means of magnitude less than $6 \times 10^{-17}$, variances exactly 1, and off-diagonal correlation term magnitudes less than $3.5 \times 10^{-15}$. When we color the noise to produce $Y$ and then obtain the masked data $Z = X + dY$, we compute the Domingo-Ferrer information loss statistics. We summarize the results compared to those for masked data without the extra whitening in tables below. In general, the means show a good deal of improvement, while the second order statistics are not greatly affected.

The tables summarize the effects of using the whiten random noise to generate the masked data. The unscaled masked data is the result $Z = X + dY$, while the scaled data is the result of scaling the masked data by $\sqrt{1+d}$ to correct for the covariance inflation. In Table 6.1, we see that the whitened data seems to result in generally slightly smaller perturbations of the original data. At smaller noise levels, the reduction is greater than the reduction resulting from the covariance scaling. As the noise level increases, and thus the scale factor increases, the scaling effects are greater than the whitening effects. We suppose that the reduction in the average standardized data perturbation due to the whitening is primarily a result of improved centering of the whitened data.

| | Unscaled | | Scaled | |
|---|---|---|---|---|
| | Standard | Whitened | Standard | Whitened |
| 1% | 0.0604 | 0.0575 | 0.0601 | 0.0573 |
| 5% | 0.1350 | 0.1286 | 0.1321 | 0.1260 |
| 10% | 0.1909 | 0.1818 | 0.1830 | 0.1746 |
| 20% | 0.2700 | 0.2572 | 0.2488 | 0.2377 |

Table 6.1: Average Standardized Data Difference

| | Unscaled | | Scaled | |
|---|---|---|---|---|
| | Standard | Whitened | Standard | Whitened |
| 1% | 0.0022 | $4.604 \times 10^{-15}$ | 0.0022 | $5.305 \times 10^{-15}$ |
| 5% | 0.0049 | $1.625 \times 10^{-15}$ | 0.0049 | $4.404 \times 10^{-15}$ |
| 10% | 0.0070 | $6.721 \times 10^{-15}$ | 0.0070 | $4.113 \times 10^{-15}$ |
| 20% | 0.0099 | $6.872 \times 10^{-15}$ | 0.0099 | $3.239 \times 10^{-15}$ |

Table 6.2: Average Proportional Mean Difference

In any event, there is certainly improved centering for the whitened data, as we can see from Table 6.2. While the deviations from the mean are small in the standard data, they have been reduced to virtually nothing in the whitened data. We also note that while the mean deviations increase with increasing noise levels for the standard data, they change irregularly for increasing noise or for changing from unscaled to scaled for the whitened data. This appears to indicate that the whitened data deviations from the mean are attributable to machine noise.

The remaining tables show that the improvement due to whitening is less significant for the second order statistics. In Table 6.3 we see that the whitened masked data produces a very small improvement in the average proportional variance error, much smaller than that produced by the covariance inflation scaling. In Table 6.4 we see that the average proportional covariance error increases somewhat for the whitened masked noise. This mostly supports the assertion that this measure of information loss is poorly scaled as well as being redundant. When we return to the properly scaled average correlation difference, we see in Table 6.5 that the whitened data does result in smaller correlation errors, although again these reductions are quite small.

| | Unscaled | | Scaled | |
|---|---|---|---|---|
| | Standard | Whitened | Standard | Whitened |
| 1% | 0.0102 | 0.0098 | 0.0007 | 0.0006 |
| 5% | 0.0505 | 0.0497 | 0.0015 | 0.0013 |
| 10% | 0.1007 | 0.0995 | 0.0020 | 0.0017 |
| 20% | 0.2010 | 0.1993 | 0.0026 | 0.0022 |

Table 6.3: Average Proportional Variance Difference

| | Unscaled | | Scaled | |
|---|---|---|---|---|
| | Standard | Whitened | Standard | Whitened |
| 1% | 0.0142 | 0.0166 | 0.0081 | 0.0088 |
| 5% | 0.0563 | 0.0634 | 0.0180 | 0.0190 |
| 10% | 0.1056 | 0.1188 | 0.0248 | 0.0257 |
| 20% | 0.2013 | 0.2266 | 0.0332 | 0.0333 |

Table 6.4: Average Proportional Covariance Difference

| | Unscaled | | Scaled | |
|---|---|---|---|---|
| | Standard | Whitened | Standard | Whitened |
| 1% | 0.00043 | 0.00041 | 0.00043 | 0.00041 |
| 5% | 0.00093 | 0.00089 | 0.00093 | 0.00089 |
| 10% | 0.00126 | 0.00120 | 0.00126 | 0.00120 |
| 20% | 0.00166 | 0.00156 | 0.00166 | 0.00156 |

Table 6.5: Average Correlation Difference

## 6.2 Data Probability Distributions

In Dandekar's method and in Sullivan's method as described in Brand's paper, efforts are made to try to produce masked data where the variables maintain the same probability distribution as those in the raw data. Presumably masked data sets can be considered more analytically valid when they replicate the original distribution than when they just preserve the first and second central moments. On the other hand, it is not clear that these methods preserve distributions for arbitrary subpopulations.

As long as we use a method of additive noise, we cannot presume that we are preserving the original data distributions. If the raw data $X$ is normally distributed and the additive noise $Y$ is normally distributed, then the masked data $Z = X + dY$ will be normally distributed as well. However, if $Y$ has a mixture distribution, as in a mixture of normals, then it is hard to say what distribution the resulting masked data $Z$ has. More importantly, the raw data $X$ may very likely not be normally distributed. Real life data, such as the Kim-Winkler data, can be highly skewed with large isolated outliers. If we were to add noise $Y$ whose distribution resembles that of $X$, then the resulting masked data set $Z$ may be distributed reasonably similarly to $X$ and the re-identification masking might be more effective as well.

The first step is to remove correlations between the variables of $X$. Thus we can compute the sample covariance matrix of $X$,

$$\Sigma_X = \frac{1}{n} \sum_{i=1}^{n} (x_i - \mu_X) (x_i - \mu_X)^T$$

and then compute a square root $\Sigma_X^{\frac{1}{2}}$ of the sample covariance. We can then get a scaled, centered, and decoupled data set

$$X' = (X - \mu_X) \left( \Sigma_X^{\frac{1}{2}} \right)^{-T}$$

that should have mean 0 and covariance $I$. Since the variables of $X'$ are now uncorrelated, we can consider the (empirical) distributions of each variable separately.

For each variable, we find an empirical distribution. There are probably a lot of ways to do this. At the most basic level, an empirical distribution could be a sorted list of the values of the variable. In the case of repeated values especially, a cumulative sum of the number of times the value (or

less) occurs is also recorded. Then a measure of $\Pr(X_i' \leq \alpha)$ can be just the proportion of times that such values have occurred. For intermediate values of $\alpha$, one can interpolate, using either linear interpolation or perhaps a smoother higher order interpolation. For the simulation, one could use a uniform random variable to generate a cumulative probability value $p$, then use the interpolated table to compute an inverse distribution value $\alpha$ satisfying $\Pr(X_i' \leq \alpha) = p$. In this manner, we can independently fill up a table $W$ where the columns represent random samples from the individual variable empirical probability distributions. We can then get a masked data set $Z$ by computing

$$Z = (X' + dW)\left(\Sigma_X^{\frac{1}{2}}\right)^T + \mu_X.$$

Since $X'$ and $W$ should both have mean 0, $Z$ should have mean $\mu_X$. If $X'$ and $W$ are independent, we expect that the covariance of $Z$ should be about $(1 + d^2)\,\Sigma_X$ so that correlations should be nearly preserved. Meanwhile, the distribution of $Z$ should be nearer to the distribution of $X$ than if $W$ were generated by some unrelated distribution. In particular, it the distribution of $X$ is characterized by several large outliers that skew the distribution, then the simulation may create some new outliers for $W$ which could make some re-identifications less certain.

# Bibliography


[Brand] Brand, Ruth. "Microrddata Protection through Noise Addition." *Statistical Data Confidentiality* (J. Domingo-Ferrer, Ed.). *Lecture Notes on Artificial Intelligence*. Springer-Verlag. New York, (2002, to appear).

[Dandekar] Dandekar, Ramesh, Michael Cohen, and Nancy Kirkendall. "Sensitive Micro Data Protection Using Latin Hypercube Sampling Technique." *Statistical Data Confidentiality* (J. Domingo-Ferrer, Ed.). *Lecture Notes on Artificial Intelligence*. Springer-Verlag. New York, (2002, to appear).

[Domingo-Ferrer] Domingo-Ferrer, J., J. Mateo-Sanz and Vincenc Torra. "Comparing SDC Methods for Microdata on the Basis of Information Loss and Disclosure Risk." *Proceedings of ETK-NTTS* . (2001, to appear).

[Kim86] Kim, J. J. "A Method for Limiting Disclosure in Microdata Based on Random Noise and Transformation." *American Statistical Association, Proceedings of the Section on Survey Research Methods*, (1986) 303–308.

[Kim90] Kim, J. J. "Subdomain Estimation for the Masked Data." *American Statistical Association, Proceedings of the Section on Survey Research Methods*, (1990) 456–461.

[Kim95] Kim, J. J. and W. E. Winkler. "Masking Microdata Files." *American Statistical Association, Proceedings of the Section on Survey Research Methods*, (1995) 114–119.

[Roque] Roque, G. M. , *Masking Microdata Files with Mixtures of Multivariate Normal Distributions*, Unpublished Ph.D. dissertation, Department of Statistics, University of California–Riverside (2000).

[Winkler94] Winkler, W. E. "Advanced Methods for Record Linkage." *American Statistical Association, Proceedings of the Section on Survey Research Methods*, (1994) 467–472.

[Winkler95] Winkler, W. E. "Matching and Record Linkage" in B. G. Cox (Ed.) *Business Survey Methods*, New York: J. Wiley, (1995) 355–384.

[Winkler98] Winkler, W. E. "Re-identification Methods for Evaluating the Confidentiality of Analytically Valid Microdata." *Research in Official Statistics,* **1**, (1998) 87–104

[Yancey] Yancey, W. E., W. E. Winkler, and Robert Creecy. "Disclosure Risk Assessment in Perturbative Microdata Protection via Record Linkage", *Statistical Data Confidentiality* (J. Domingo-Ferrer, Ed.). *Lecture Notes on Artificial Intelligence*. Springer-Verlag. New York, 2002 (to appear).

# Appendix A

# SAS Programs

## A.1 SAS Program to Create and Evaluate Mixture Noise Masked Data

The following SAS program uses PROC IML to compute data sets masked by mixture model additive noise. It reads in the specified raw data set and computes mean and covariance statistics. It computes the square root of the covariance matrix, using either the eigenvalue of the Cholesky decomposition. It then computes a data set of "white" mixture model noise as described in Chapter 2 and an array of "colored" mixture noise using the covariance square root, checking mean and covariance statistics. It adds the colored noise to the raw data to produce a masked data set. It then computes information loss statistics using formulas derived from Domingo-Ferrer. It then computes a rescaled masked data set to get the masked data covariance to agree with the original data covariance, and then checks the information loss statistics of the scaled masked data set. It also computes a "whitened" mixture noise data set as proposed in Chapter 6, masks the data using this noise, and computes the information loss statistics for both the unscaled and scaled masked data. The program also saves the masked data set to a flat file.

```
/*  William E. Yancey, Statistics Research Division   */
/*  SAS Program to produce data files masked by mixture noise
      and compute data statistics  */
/*  Output data statisistcs include Domingo-Ferrer-like
```

```
      information loss summary statistics   */
/*  Computes and outputs masked data for both additive noise
      model and rescaled additive noise model
      to compensate for covariance inflation   */
/*  Current version (11/15/01) also uses translation and square
      root transformation to ''whiten'' random noise mixture and
      compare results  */
/*  Computations carried out using SAS Proc IML to do matrix
      calculations   */

/*  Read in raw data file  */
data kimwin;
   infile '/home/bwinkler/psa/jk2001/srawid4.dat';
   input @1 seq 6. @7 tot_inc 8. @15 adj_gr 8. @23 wage 7.
      @30 tax_int 7. @37 divid 7. @44 rent 7. @51 non_int 7.
      @58 ssn_inc 6. @64 ret_type 1. @70 agec 2. @73 race 1.
      @75 sex 1. @77 cpswage 8. @86 cpsprop 8. @95 cpsagi 8.;

proc iml;
   /*   Create matrix with raw numerical data and compute mean,
         covariance, correlation  */
   use kimwin;
   read all var {tot_inc adj_gr wage tax_int divid rent non_int
      ssn_inc cpswage cpsprop cpsagi seq ret_type agec race sex};

   X = tot_inc || adj_gr || wage || tax_int || divid || rent ||
         non_int || ssn_inc || cpswage || cpsprop || cpsagi;

   nrec = nrow(X);

   nvar = ncol(X);

   meanX = X[+,]/nrec;

   print meanX ''Mean of raw data'';

   X0 = X - repeat(meanX,nrec,1);
```

```
   covX = (t(X0) * X0)/nrec;

   varX = vecdiag(covX);

   print varX ''Variance of raw data'';

   siginvX = diag(1/sqrt(varX));

   corX = siginvX*covX*siginvX;

   print corX ''Correlation of raw data'';

/* chol = root(covX);  */  /*  Cholesky factor   */

   call eigen(eigs,U,covX);

   eigsqr = sqrt(eigs);

   print eigsqr  ''The square roots of the covariance eigenvalues'';

   print U  ''Orthonormal eigenvector transform'';

   Seeds = {895907 234789 684111 905241 380625 746913 435673 502462
            265362 979324 846265 114473 988584 940020 414342 271842
            314161 557215 664901 532861 606512 985420};

   Useed = 646928;

   sig2 = 0.025;     /*  Component distortion factor (sigma)  */

   d = 0.01;      /*  Masked data covariance dilation term   */

   dsd = sqrt(d);

   MixN = J(nrec,nvar,0);    /*  Matrix to hold uncorrelated mixture noise  */

   MixNew = J(nrec,nvar,0);   /*  Matrix to hold whitened mixture noise   */
```

```
   Y = J(nrec, nvar, 0);  /*  Matrix to hold colored mixture noise  */

   Ynew = J(nrec,nvar,0);  /*  Matrix to hold colored mixture noise from
                                 whitened noise  */

   /*  Generate mixture noise   */

   do i = 1 to nrec;

      do j = 1 to nvar;

         choice = uniform(Useed);

         if (choice < 0.5) then do;

            w = normal(Seeds[1,j]);

            MixN[i,j] = sqrt(sig2)*w + sqrt(1 - sig2);

         end;

         else do;

            w = normal(Seeds[2,j]);

            MixN[i,j] = sqrt(sig2)*w - sqrt(1 - sig2);

         end;
      end;

/*    Y[i,] = MixN[i,]*chol;  */

      Y[i,] = MixN[i,]*diag(eigsqr)*t(U);

   end;

   /*   Test noise statistics   */
```

```
meanMix = MixN[+,]/nrec;

print meanMix ''Mean of pure mixed noise'';

MixN0 = MixN - repeat(meanMix,nrec,1);

covMix = (t(MixN0)*MixN0)/nrec;

call eigen(eignois,Unois,covMix);

eignsqr = sqrt(eignois);

print eignsqr  ''The square roots of the covariance eigenvalues'';

Dinv = diag(1/eignsqr);

varMix = vecdiag(covMix);

print varMix ''Variance of mixed noise'';

print covMix ''Covariance of mixed noise'';

siginvMx = diag(1/sqrt(varMix));

corMix = siginvMx*covMix*siginvMx;

print corMix ''Correlation of mixed noise'';

meanY = Y[+,]/nrec;

print meanY ''Mean of colored noise'';

Y0 = Y - repeat(meanY,nrec,1);

covY = (t(Y0)*Y0)/nrec;

varY = vecdiag(covY);
```

```
siginvY = diag(1/sqrt(varY));

corY = siginvY*covY*siginvY;

print corY ''Correlation of colored noise'';

Z = X + dsd#Y;       /*   Compute masked data   */

/*   Test masked data statisitcs   */

meanZ = Z[+,]/nrec;

print meanZ ''Mean of masked data'';

Z0 = Z - repeat(meanZ,nrec,1);

covZ = (t(Z0)*Z0)/nrec;

varZ = vecdiag(covZ);

print varZ ''Variance of masked data'';

siginvZ = diag(1/sqrt(varZ));

corZ = siginvZ*covZ*siginvZ;

print corZ ''Correlation of masked data'';

/*   Comupte Domingo measures for mixed data   */

Xdiff = abs(X - Z);

Xscal = 0.5#(abs(X) + abs(Z));

Xscal = Xscal + (Xscal = 0);

Xcmp = Xdiff/Xscal;
```

```
normX = Xcmp[:];
print normX ''Average proportional element difference'';
Xscal = repeat(t(sqrt(vecdiag(covX))),nrec,1);
Xcmp = Xdiff/Xscal;
normX = Xcmp[:]/sqrt(2);
print normX ''Average standardized difference from masked data'';
meandiff = abs(meanX - meanZ);
meancmp = meandiff/abs(meanX);
normmean = meancmp[:];
print normmean ''Average proportional mean difference'';
vardiff = abs(varX - varZ);
varcmp = vardiff/abs(varX);
normvar = varcmp[:];
print normvar ''Average proportional variance difference'';
normcov = 0;
do i = 1 to nvar;
   do j = i to nvar;
      normcov = normcov + abs(covX[i,j] - covZ[i,j])/abs(covX[i,j]);
   end;
```

```
end;
normcov = normcov/((nvar*(nvar + 1))/2);
print normcov ''Average proportional covariance difference'';
normcor = 0;
do i = 1 to nvar;
   do j = (i+1) to nvar;
      normcor = normcor + abs(corX[i,j] - corZ[i,j]);
   end;
end;
normcor = normcor/((nvar*(nvar - 1))/2);
print normcor ''Average correlation difference'';
/*   Compute masked data with rescaled covariance  */
Zpr = (1/sqrt(1 + d))#Z0 + repeat(meanZ,nrec,1);
/*   Compute statistics for rescaled masked data   */
meanZpr = Zpr[+,]/nrec;
print meanZpr ''Mean of rescaled masked data'';
Zpr0 = Zpr - repeat(meanZpr,nrec,1);
covZpr = (t(Zpr0)*Zpr0)/nrec;
print covZpr ''Covariance of rescaled masked data'';
```

```
varZpr = vecdiag(covZpr);

print varZpr ''Variance of rescaled masked data'';

siginvZp = diag(1/sqrt(varZpr));

corZpr = siginvZp*covZpr*siginvZp;

print corZpr ''Correlation of rescaled masked data'';

/*   Compute Domingo measures for rescaled masked data   */

Xdiffpr = abs(X - Zpr);

Xscal = 0.5#(abs(X) + abs(Zpr));

Xscal = Xscal + (Xscal = 0);

Xcmppr = Xdiffpr/Xscal;

normXpr = Xcmppr[:];

print normXpr ''Average proportional element difference from scaled data'';

Xscal = repeat(t(sqrt(vecdiag(covX))),nrec,1);

Xcmppr = Xdiffpr/Xscal;

normX = Xcmppr[:]/sqrt(2);

print normX ''Average standardized difference from scaled masked data'';

meandifp = abs(meanX - meanZpr);

meancmpr = meandifp/abs(meanX);

normmeap = meancmpr[:];
```

```
print normmeap ''Average proportional mean difference from scaled data'';

vardiffp = abs(varX - varZpr);

varcmppr = vardiffp/abs(varX);

normvarp = varcmppr[:];

print normvarp ''Average proportional variance difference from scaled data'';

normcovp = 0;

do i = 1 to nvar;

   do j = i to nvar;

      normcovp = normcovp + abs(covX[i,j] - covZpr[i,j])/abs(covX[i,j]);

   end;

end;

normcovp = normcovp/((nvar*(nvar + 1))/2);

print normcovp ''Average proportional covariance difference from scaled data'

normcorp = 0;

do i = 1 to nvar;

   do j = (i+1) to nvar;

      normcorp = normcorp + abs(corX[i,j] - corZpr[i,j]);

   end;

end;
```

```
normcorp = normcorp/((nvar*(nvar - 1))/2);
print normcorp ''Average correlation difference from scaled data'';
do i = 1 to nrec;
   MixNew[i,] = (MixN[i,] - meanMix)*Unois*Dinv;
   Ynew[i,] = MixNew[i,]*diag(eigsqr)*t(U);
end;
/*   Test noise statistics   */
meanMix = MixNew[+,]/nrec;
print meanMix ''Mean of pure mixed noise'';
MixN0 = MixNew - repeat(meanMix,nrec,1);
covMix = (t(MixN0)*MixN0)/nrec;
call eigen(eignois,Unois,covMix);
eignsqr = sqrt(eignois);
print eignsqr  ''The square roots of the covariance eigenvalues'';
Dinv = diag(1/eignsqr);
varMix = vecdiag(covMix);
print varMix ''Variance of mixed noise'';
print covMix ''Covariance of mixed noise'';
siginvMx = diag(1/sqrt(varMix));
```

```
corMix = siginvMx*covMix*siginvMx;

print corMix ''Correlation of mixed noise'';

meanY = Ynew[+,]/nrec;

print meanY ''Mean of colored noise'';

Y0 = Ynew - repeat(meanY,nrec,1);

covY = (t(Y0)*Y0)/nrec;

varY = vecdiag(covY);

siginvY = diag(1/sqrt(varY));

corY = siginvY*covY*siginvY;

print corY ''Correlation of colored noise'';

Z = X + dsd#Ynew;       /*   Compute masked data   */

/*   Test masked data statisitcs   */

meanZ = Z[+,]/nrec;

print meanZ ''Mean of masked data'';

Z0 = Z - repeat(meanZ,nrec,1);

covZ = (t(Z0)*Z0)/nrec;

varZ = vecdiag(covZ);

print varZ ''Variance of masked data'';

siginvZ = diag(1/sqrt(varZ));
```

```
corZ = siginvZ*covZ*siginvZ;

print corZ ''Correlation of masked data'';

/*   Comupte Domingo measures for mixed data   */

Xdiff = abs(X - Z);

Xscal = 0.5#(abs(X) + abs(Z));

Xscal = Xscal + (Xscal = 0);

Xcmp = Xdiff/Xscal;

normX = Xcmp[:];

print normX ''Average proportional element difference'';

Xscal = repeat(t(sqrt(vecdiag(covX))),nrec,1);

Xcmp = Xdiff/Xscal;

normX = Xcmp[:]/sqrt(2);

print normX ''Average standardized difference from masked data'';

meandiff = abs(meanX - meanZ);

meancmp = meandiff/abs(meanX);

normmean = meancmp[:];

print normmean ''Average proportional mean difference'';

vardiff = abs(varX - varZ);

varcmp = vardiff/abs(varX);
```

```
normvar = varcmp[:];
print normvar ''Average proportional variance difference'';
normcov = 0;
do i = 1 to nvar;
   do j = i to nvar;
      normcov = normcov + abs(covX[i,j] - covZ[i,j])/abs(covX[i,j]);
   end;
end;
normcov = normcov/((nvar*(nvar + 1))/2);
print normcov ''Average proportional covariance difference'';
normcor = 0;
do i = 1 to nvar;
   do j = (i+1) to nvar;
      normcor = normcor + abs(corX[i,j] - corZ[i,j]);
   end;
end;
normcor = normcor/((nvar*(nvar - 1))/2);
print normcor ''Average correlation difference'';
/*   Compute masked data with rescaled covariance  */
```

```
Zpr = (1/sqrt(1 + d))#Z0 + repeat(meanZ,nrec,1);

/*   Compute statistics for rescaled masked data   */

meanZpr = Zpr[+,]/nrec;

print meanZpr ''Mean of rescaled masked data'';

Zpr0 = Zpr - repeat(meanZpr,nrec,1);

covZpr = (t(Zpr0)*Zpr0)/nrec;

print covZpr ''Covariance of rescaled masked data'';

varZpr = vecdiag(covZpr);

print varZpr ''Variance of rescaled masked data'';

siginvZp = diag(1/sqrt(varZpr));

corZpr = siginvZp*covZpr*siginvZp;

print corZpr ''Correlation of rescaled masked data'';

/*   Compute Domingo measures for rescaled masked data   */

Xdiffpr = abs(X - Zpr);

Xscal = 0.5#(abs(X) + abs(Zpr));

Xscal = Xscal + (Xscal = 0);

Xcmppr = Xdiffpr/Xscal;

normXpr = Xcmppr[:];

print normXpr ''Average proportional element difference from scaled data'';
```

```
Xscal = repeat(t(sqrt(vecdiag(covX))),nrec,1);

Xcmppr = Xdiffpr/Xscal;

normX = Xcmppr[:]/sqrt(2);

print normX ''Average standardized difference from scaled masked data'';

meandifp = abs(meanX - meanZpr);

meancmpr = meandifp/abs(meanX);

normmeap = meancmpr[:];

print normmeap ''Average proportional mean difference from scaled data'';

vardiffp = abs(varX - varZpr);

varcmppr = vardiffp/abs(varX);

normvarp = varcmppr[:];

print normvarp ''Average proportional variance difference from scaled data'';

normcovp = 0;

do i = 1 to nvar;

   do j = i to nvar;

      normcovp = normcovp + abs(covX[i,j] - covZpr[i,j])/abs(covX[i,j]);

   end;

end;

normcovp = normcovp/((nvar*(nvar + 1))/2);
```

```
print normcovp ''Average proportional covariance difference from scaled data'

normcorp = 0;

do i = 1 to nvar;

   do j = (i+1) to nvar;

      normcorp = normcorp + abs(corX[i,j] - corZpr[i,j]);

   end;

end;

normcorp = normcorp/((nvar*(nvar - 1))/2);

print normcorp ''Average correlation difference from scaled data'';

/*   Output the masked data sets */

filename out1 '/home/yance003/noise/mixnoise/mixw01.dat';
filename out2 '/home/yance003/noise/mixnoise/mixw01pr.dat';

file out1;

   do i = 1 to nrec;

      put @1 (seq[i]) 6. @7 (Z[i,1]) 8. @15 (Z[i,2]) 8.
         @23 (Z[i,3]) 7. @30 (Z[i,4]) 7. @37 (Z[i,5]) 7.
         @44 (Z[i,6]) 7. @51 (Z[i,7]) 7. @58 (Z[i,8]) 6.
         @64 (ret_type[i]) 1. @70 (agec[i]) 2. @73 (race[i]) 1.
         @75 (sex[i]) 1. @77 (Z[i,9]) 8. @86 (Z[i,10]) 8.
         @95 (Z[i,11]) 8.;

   end;

close out1;
```

```
   file out2;

      do i = 1 to nrec;

         put @1 (seq[i]) 6. @7 (Zpr[i,1]) 8. @15 (Zpr[i,2]) 8.
            @23 (Zpr[i,3]) 7. @30 (Zpr[i,4]) 7. @37 (Zpr[i,5]) 7.
            @44 (Zpr[i,6]) 7. @51 (Zpr[i,7]) 7. @58 (Zpr[i,8]) 6.
            @64 (ret_type[i]) 1. @70 (agec[i]) 2. @73 (race[i]) 1.
            @75 (sex[i]) 1. @77 (Zpr[i,9]) 8. @86 (Zpr[i,10]) 8.
            @95 (Zpr[i,11]) 8.;

      end;

   close out2;

quit;

run;
```

## A.2 SAS Program to Evaluate Subpopulation Statistics for Mixture Model Masked Microdata

This SAS program reads in a raw data set and unscaled and scaled masked data sets as computed by the previous program and selects a specified subpopulation from each set. The program then compares mean and covariance statistics for the data sets and computes the information loss statistics. The means and covariances for the subpopulation are corrected using the formuas discussed in Chapter 3.

```
/*   William E. Yancey, Statistical Research Division   */
/*   Sample program to test subpopulation of masked data for
   Domingo-Ferrar type information loss statistics
   for masked data produced form additive mixture noise and
   rescaled masked data for covariance correction   */
```

```
/*   Additive noise masked data subpopulation means and covariance
   corrections based on Jay Kim's ''Subpopulation for Masked
   Data'' papter   */



TITLE‘    Data Subset ret_type = 4,  Masking = 0.20;

data kimwin;
   infile '/home/bwinkler/psa/jk2001/srawid4.dat';
   input @1 seq 6. @7 tot_inc 8. @15 adj_gr 8. @23 wage 7.
   @30 tax_int 7. @37 divid 7. @44 rent 7. @51 non_int 7.
   @58 ssn_inc 6. @64 ret_type 1. @70 agec 2. @73 race 1.
   @75 sex 1. @77 cpswage 8. @86 cpsprop 8. @95 cpsagi 8.;

data kimwins;
   set kimwin;
   if (ret_type = 4);


data mask;
   infile '/home/yance003/noise/mixnoise/mixw20.dat';
   input @1 seq 6. @7 tot_inc 8. @15 adj_gr 8. @23 wage 7.
   @30 tax_int 7. @37 divid 7. @44 rent 7. @51 non_int 7.
   @58 ssn_inc 6. @64 ret_type 1. @70 agec 2. @73 race 1.
   @75 sex 1. @77 cpswage 8. @86 cpsprop 8. @95 cpsagi 8.;

data masks;
   set mask;
   if (ret_type = 4);


data maskpr;
   infile '/home/yance003/noise/mixnoise/mixw20pr.dat';
   input @1 seq 6. @7 tot_inc 8. @15 adj_gr 8. @23 wage 7.
   @30 tax_int 7. @37 divid 7. @44 rent 7. @51 non_int 7.
   @58 ssn_inc 6. @64 ret_type 1. @70 agec 2. @73 race 1.
   @75 sex 1. @77 cpswage 8. @86 cpsprop 8. @95 cpsagi 8.;
```

```
data maskprs;
   set maskpr;
   if (ret_type = 4);

proc iml;

   d = 0.20;

   use kimwin;
   read all var {tot_inc adj_gr wage tax_int divid rent non_int
      ssn_inc cpswage cpsprop cpsagi seq ret_type agec race sex};

   X = tot_inc || adj_gr || wage || tax_int || divid || rent ||
         non_int || ssn_inc;

   nrec = nrow(X);

   nvar = ncol(X);

   print nrec  ''Number of data records'';

   meanX = X[+,]/nrec;

   print meanX ''Mean of raw data'';

   X0 = X - repeat(meanX,nrec,1);

   covX = (t(X0) * X0)/nrec;

   print covX  ''Covariance of raw data'';

   varX = vecdiag(covX);

   print varX ''Variance of raw data'';

   sdX = sqrt(varX);
```

```
print sdX  ''Standard deviations of raw data'';

siginvX = diag(1/sqrt(varX));

corX = siginvX*covX*siginvX;

print corX ''Correlation of raw data'';

use kimwins;
read all var {tot_inc adj_gr wage tax_int divid rent non_int
      ssn_inc cpswage cpsprop cpsagi seq ret_type agec race sex};

Xs = tot_inc || adj_gr || wage || tax_int || divid || rent ||
      non_int || ssn_inc;

nrecsu = nrow(Xs);

print nrecsu  ''Number of data subset records'';

meanXs = Xs[+,]/nrecsu;

print meanXs ''Mean of raw data subset'';

Xs0 = Xs - repeat(meanXs,nrecsu,1);

covXs = (t(Xs0) * Xs0)/nrecsu;

print covXs  ''Covariance of raw data subset'';

varXs = vecdiag(covXs);

print varXs ''Variance of raw data subset'';

sdXs = sqrt(varXs);

print sdXs  ''Standard deviations of raw data subset'';

siginvXs = diag(1/sqrt(varXs));
```

```
corXs = siginvXs*covXs*siginvXs;

print corXs ''Correlation of raw data subset'';

use mask;
read all var {tot_inc adj_gr wage tax_int divid rent non_int
      ssn_inc cpswage cpsprop cpsagi seq ret_type agec race sex};

Z = tot_inc || adj_gr || wage || tax_int || divid || rent ||
      non_int || ssn_inc;

nrec = nrow(Z);

meanZ = Z[+,]/nrec;

print meanZ ''Mean of masked data'';

Z0 = Z - repeat(meanZ,nrec,1);

covZ = (t(Z0) * Z0)/nrec;

print covZ ''Covariance of masked data'';

varZ = vecdiag(covZ);

print varZ ''Variance of masked data'';

siginvZ = diag(1/sqrt(varZ));

corZ = siginvZ*covZ*siginvZ;

print corZ ''Correlation of masked data'';

use masks;
read all var {tot_inc adj_gr wage tax_int divid rent non_int
      ssn_inc cpswage cpsprop cpsagi seq ret_type agec race sex};
```

```
Zs = tot_inc || adj_gr || wage || tax_int || divid || rent ||
      non_int || ssn_inc;

nrecsu = nrow(Zs);

meanZs = Zs[+,]/nrecsu;

print meanZs ''Mean of masked data'';

Zs0 = Zs - repeat(meanZs,nrecsu,1);

covZs = (t(Zs0) * Zs0)/nrecsu;

print covZs  ''Covariance of masked data subset'';

covZsc = covZs - (d/(d + 1))#covZ;

print covZsc  ''Corrected covariance of masked data subset'';

varZsc = vecdiag(covZsc);

print varZsc ''Corrected Variance of masked data'';

siginvZs = diag(1/sqrt(varZsc));

corZsc = siginvZs*covZsc*siginvZs;

print corZsc ''Corrected Correlation of masked data'';

use maskpr;

read all var {tot_inc adj_gr wage tax_int divid rent non_int
      ssn_inc cpswage cpsprop cpsagi seq ret_type agec race sex};

Zpr = tot_inc || adj_gr || wage || tax_int || divid || rent ||
      non_int || ssn_inc;

nrec = nrow(Zpr);
```

```
meanZpr = Zpr[+,]/nrec;

print meanZpr ''Mean of masked scaled data'';

Zpr0 = Zpr - repeat(meanZpr,nrec,1);

covZpr = (t(Zpr0) * Zpr0)/nrec;

print covZpr  ''Covariance of masked scaled data'';

varZpr = vecdiag(covZpr);

print varZpr ''Variance of masked scaled data'';

siginvZp = diag(1/sqrt(varZpr));

corZpr = siginvZp*covZpr*siginvZp;

print corZpr ''Correlation of masked scaled data'';

use maskprs;
read all var {tot_inc adj_gr wage tax_int divid rent non_int
      ssn_inc cpswage cpsprop cpsagi seq ret_type agec race sex};

Zprs = tot_inc || adj_gr || wage || tax_int || divid || rent ||
      non_int || ssn_inc;

nrecsu = nrow(Zprs);

meanZprs = Zprs[+,]/nrecsu;

print meanZprs  ''Mean of masked scaled data subset'';

menZprsc = sqrt(1 + d)#meanZprs - (sqrt(1 + d) - 1)#meanZpr;

print menZprsc ''Corrected Mean of masked scaled data subset'';
```

```
Zprs0 = Zprs - repeat(meanZprs,nrecsu,1);

covZprs = (t(Zprs0) * Zprs0)/nrecsu;

print covZprs  ''Covariance of masked scaled data subset'';

covZprsc = (1 + d)#covZprs - d#covZpr;

print  covZprsc  ''Corrected covariance of masked scaled data
                    subset'';

varZprsc = vecdiag(covZprsc);

print varZprsc ''Corrected Variance of masked scaled data
                  subset'';

siginvZp = diag(1/sqrt(varZprsc));

corZprsc = siginvZp*covZprsc*siginvZp;

print corZprsc ''Corrected Correlation of masked scaled data
                  subset'';

/*   Information measures  */

Xdiff = abs(X - Z);

Xscal = 0.5#(abs(X) + abs(Z));

Xscal = Xscal + (Xscal = 0);

Xcmp = Xdiff/Xscal;

normX = Xcmp[:];

print normX ''Average proportional element difference from masked
               data'';
```

```
Xscal = repeat(t(sqrt(vecdiag(covX))),nrec,1);

Xcmp = Xdiff/Xscal;

normX = Xcmp[:]/sqrt(2);

print normX ''Average standardized difference from masked data'';

meandiff = abs(meanX - meanZ);

meancmp = meandiff/abs(meanX);

print meancmp  ''Mean proportional errors'';

normmean = meancmp[:];

print normmean ''Average proportional mean difference from masked
                 data'';

vardiff = abs(varX - varZ);

varcmp = vardiff/abs(varX);

normvar = varcmp[:];

print normvar ''Average proportional variance difference from
                 masked data'';

coverr = abs(covX - covZ)/abs(covX);

print coverr  ''Covariance proportional errors'';

normcov = 0;

do i = 1 to nvar;

   do j = i to nvar;
```

```
      normcov = normcov + abs(covX[i,j] - covZ[i,j])/abs(covX[i,j]);

   end;

end;

ncovs = (nvar*(nvar + 1))/2;

normcov = normcov/ncovs;

print normcov ''Average proportional covariance difference from
                 masked data'';

normcor = 0;

do i = 1 to nvar;

   do j = (i+1) to nvar;

      normcor = normcor + abs(corX[i,j] - corZ[i,j]);

   end;

end;

ncors = (nvar*(nvar - 1))/2;

normcor = normcor/ncors;

print normcor ''Average correlation difference from masked data'';

Xdiffpr = abs(X - Zpr);

Xscal = 0.5#(abs(X) + abs(Zpr));

Xscal = Xscal + (Xscal = 0);

Xcmppr = Xdiffpr/Xscal;
```

```
normXpr = Xcmppr[:];

print normXpr ''Average proportional element difference from scaled
                  masked data'';

Xscal = repeat(t(sqrt(vecdiag(covX))),nrec,1);

Xcmppr = Xdiffpr/Xscal;

normX = Xcmppr[:]/sqrt(2);

print normX ''Average standardized difference from scaled masked
               data'';

meandifp = abs(meanX - meanZpr);

meancmpr = meandifp/abs(meanX);

normmeap = meancmpr[:];

print normmeap ''Average proportional mean difference from scaled
                  masked data'';

vardiffp = abs(varX - varZpr);

varcmppr = vardiffp/abs(varX);

normvarp = varcmppr[:];

print normvarp ''Average proportional variance difference from
                  scaled masked data'';

normcovp = 0;

do i = 1 to nvar;

   do j = i to nvar;
```

```
      normcovp = normcovp + abs(covX[i,j] - covZpr[i,j])/abs(covX[i,j]);

   end;

end;

normcovp = normcovp/ncovs;

print normcovp ''Average proportional covariance difference from
                  scaled masked data'';

normcorp = 0;

do i = 1 to nvar;

   do j = (i+1) to nvar;

      normcorp = normcorp + abs(corX[i,j] - corZpr[i,j]);

   end;

end;

normcorp = normcorp/ncors;

print normcorp ''Average correlation difference from scaled masked
                  data'';

Xdiff = abs(Xs - Zs);

Xscal = 0.5#(abs(Xs) + abs(Zs));

Xscal = Xscal + (Xscal = 0);

Xcmp = Xdiff/Xscal;

normX = Xcmp[:];
```

```
print normX ''Average proportional element difference from masked
              data subset'';

Xscal = repeat(t(sqrt(vecdiag(covXs))),nrecsu,1);

Xcmp = Xdiff/Xscal;

normX = Xcmp[:]/sqrt(2);

print normX ''Average standardized difference from masked data
              subset'';

meandiff = abs(meanXs - meanZs);

meancmp = meandiff/abs(meanXs);

print meancmp  ''Subset proportional mean errors'';

normmean = meancmp[:];

print normmean ''Average proportional mean difference from masked
                 data subset'';

vardiff = abs(varXs - varZsc);

varcmp = vardiff/abs(varXs);

normvar = varcmp[:];

print normvar ''Average proportional variance difference from
                 masked data subset'';

coverrs = abs(covXs - covZsc)/abs(covXs);

print coverrs  ''Subset coveriance proportional errors'';

normcov = 0;
```

```
do i = 1 to nvar;

   do j = i to nvar;

      normcov = normcov + abs(covXs[i,j] - covZsc[i,j])/abs(covXs[i,j]);

   end;

end;

normcov = normcov/ncovs;

print normcov ''Average proportional covariance difference from
                  masked data subset'';

normcor = 0;

do i = 1 to nvar;

   do j = (i+1) to nvar;

      normcor = normcor + abs(corXs[i,j] - corZsc[i,j]);

   end;

end;

normcor = normcor/ncors;

print normcor ''Average correlation difference from masked data
                  subset'';

Xdiffpr = abs(Xs - Zprs);

Xscal = 0.5#(abs(Xs) + abs(Zprs));

Xscal = Xscal + (Xscal = 0);
```

```
Xcmppr = Xdiffpr/Xscal;

normXpr = Xcmppr[:];

print normXpr ''Average proportional element difference from scaled
                  masked data subset'';

Xscal = repeat(t(sqrt(vecdiag(covXs))),nrecsu,1);

Xcmppr = Xdiffpr/Xscal;

normX = Xcmppr[:]/sqrt(2);

print normX ''Average standardized difference from scaled masked
               data subset'';

meandifp = abs(meanXs - menZprsc);

meancmpr = meandifp/abs(meanXs);

normmeap = meancmpr[:];

print normmeap ''Average proportional mean difference from scaled
                  masked data subset'';

vardiffp = abs(varXs - varZprsc);

varcmppr = vardiffp/abs(varXs);

normvarp = varcmppr[:];

print normvarp ''Average proportional variance difference from
                  scaled masked data subset'';

normcovp = 0;

do i = 1 to nvar;
```

```
      do j = i to nvar;

         normcovp = normcovp + abs(covXs[i,j] - covZprsc[i,j])/abs(covXs[i,j]);

      end;

   end;

   normcovp = normcovp/ncors;

   print normcovp ''Average proportional covariance difference from
                    scaled masked data subset'';

   normcorp = 0;

   do i = 1 to nvar;

      do j = (i+1) to nvar;

         normcorp = normcorp + abs(corXs[i,j] - corZprsc[i,j]);

      end;

   end;

   normcorp = normcorp/ncors;

   print normcorp ''Average correlation difference from scaled
                    masked data subset'';

quit;

run;
```