MAT 4376 Foundations of Data Privacy

Rafał Kulik

Department of Mathematics and Statistics (University of Ottawa)

Lectures 10, 11, 12

1. Synthetic data generation
   - Introduction
   - Simulation from univariate continuous distributions
   - Multivariate Inverse Transform Sampling
   - Generating Multivariate Distribution with Decomposition-Based Method
   - Bootstrap

2. Synthpop package: Description, Challenges and Solutions

During these lectures we will learn:

- basic simulation techniques for univariate and multivariate data;
- sequential procedure to create synthetic data;
- how to use some R packages.

Some resources

- Webpage support for the textbook: `https://programming-dp.com/ch14.html`
- MSc thesis of Chang Qu.

Introduction

Synthetic data generation is an approach to create anonymized data that share properties with the original dataset. This is nothing else, but a special case of simulation.
An output of the algorithm is a synthetic dataset with the same shape, i.e. same set of columns and same number of rows (although nothing prevents us from creating additional rows in the synthetic dataset). In addition, we would like the values in the synthetic dataset to have the same properties as the corresponding values in the original dataset.

Univariate continuous random variables

In what follows, we will need the following terminology.

- F is the cumulative distribution function (CDF) of a rv X .
- f is the density of F (if it exists).
- The left and right endpoints a, b of F are defined by $a = \inf\{t \in R : F(t) > 0\}$, $b = \sup\{t \in R : F(t) < 1\}$.
- The quantile function or generalized inverse function of F is defined as

$$Q(u) := F^{\leftarrow}(u) := \inf\{x \in \mathbb{R} : F(x) \geq u\} .$$

- If we need to emphasize which random variable we are dealing with, we are going to write F_X, Q_X.
- $\widehat{F}_n$ is the empirical distribution based on data $X_1, \ldots, X_n$, defined as

$$\widehat{F}_n(x) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}\{X_i \leq x\}.$$

Univariate continuous random variables

Theorem 1

Let F be a CDF. Assume that F is continuous and strictly increasing.

- *Let U be a random variable, uniformly distributed on $[0, 1]$. Then the CDF of the random variable $Q(U)$ is F .*
- *Let X be a random variable with CDF F. Then $F(X)$ is uniformly distributed on $[0, 1]$.*

Proof.

- Let U be a random variable, uniformly distributed on $[0, 1]$. Then for $x \in (a, b)$,

$$\mathbb{P}(Q(U) \leq x) = \mathbb{P}(U \leq F(x)) = F(x).$$

Univariate continuous random variables

Simulation Algorithm:

- Generate a sample of size n from the uniform distribution on $[0, 1]$. Denote it by u_i, $i = 1, \ldots, n$.
- For each u_i, calculate $x_i = Q(u_i)$.
- Then x_i, $i = 1, \ldots, n$, is a sample from the CDF F.

Univariate continuous random variables

Example 2

We want to generate data from an exponential distribution with parameter $\lambda = 1$, that is $F(x) = 1 - e^{-x}$, $x \geq 0$.

- We first find $Q(p) = -\frac{1}{\lambda} \ln(1-p)$.
- We generate n observations u_i from the standard uniform distribution.
- Then we calculate $x_i = -\frac{1}{\lambda} \ln(1-u_i)$ for each i.
- Since U and $1-U$ have the same distribution, we can also calculate $x_i = -\frac{1}{\lambda} \ln(u_i)$.

Parametric families

Example 3

Suppose that we have a dataset and we assume that the data follow a normal distribution with mean μ and variance σ^2, denoted by $\mathcal{N}(\mu, \sigma^2)$.

- First, we estimate the parameters μ and σ^2 using the sample mean $\hat{\mu}$ and sample variance $\hat{\sigma}^2$.
- Then, we generate data from the distribution $\mathcal{N}(\hat{\mu}, \hat{\sigma}^2)$ using the inverse transformation method.

□

Univariate discrete random variables

We want to generate random numbers from a non-continuous random variable X. This means that the associated cumulative distribution function (CDF) F is right-continuous only, but not continuous. Also the CDF is not strictly increasing.

Univariate discrete random variables

Example 4

Consider a random variable X where $\mathbb{P}(X = 0) = \mathbb{P}(X = 1) = \frac{1}{2}$. The cumulative distribution function (CDF) and the quantile function for X are as follows: The CDF, $F(x)$, is given by:

$$F(x) = \begin{cases} 0 & \text{if } x < 0, \\ \frac{1}{2} & \text{if } 0 \leq x < 1, \\ 1 & \text{if } x \geq 1. \end{cases}$$

The quantile function, $Q_X(p)$, is given by:

$$Q(p) = \begin{cases} 0 & \text{if } 0 \leq p < \frac{1}{2}, \\ 1 & \text{if } \frac{1}{2} \leq p \leq 1. \end{cases}$$

□

Univariate discrete random variables

Proposition 5

- *Let a and b be the left and right endpoints of F. Then* $\lim_{u\to 0} Q(u) = a$, $\lim_{u\to 1} Q(u) = b$. *The results also hold if* $a = -\infty$ *or* $b = +\infty$.
- *For* $x, x' \in [a, b]$, $F(x) \leq F(x') \Rightarrow Q(F(x)) \leq x'$.
- $F(x) \geq u \Leftrightarrow x \geq Q(u)$, *for all* $x \in (a, b)$ *and* $u \in (0, 1)$.

Univariate discrete random variables

We present the simulation algorithm for a discrete random variable, where $\mathbb{P}(X = x_i) = f_i$ for $i = 1, \ldots, q$, with $\sum_{i=1}^{q} f_i = 1$ and $f_i > 0$. Then, the CDF is

$$F(x_i) = \mathbb{P}(X \leq x_i) = F_i = \sum_{j=1}^{i} f_j,$$

and $F(x) = F(x_i)$ whenever $x_i \leq x < x_{i+1}$. The corresponding quantile function is given by:

$$Q(u) = x_k \quad \text{where } k = \min\{i : F(x_i) \geq u\} .$$

Univariate discrete random variables

We have the following algorithm when dealing with a discrete cumulative distribution function that has a finite domain of possible values.

Simulation Algorithm for Finite Domain:

- Generate a sample of size n from the uniform distribution on $[0, 1]$. Denote it by u_j, $j = 1, \ldots, n$.
- For each $j = 1, \ldots, n$, if $F_{i-1} < u_j \leq F_i$, then set x_i as a random number from the CDF F.

Multivariate distributions

Our goal is to generate numbers from d-variate distribution by applying the inverse transformation method. This approach requires knowledge of the conditional inverse cumulative distribution functions:

$$F_{X_1}^{-1}, F_{X_2|X_1}^{-1}, \ldots, F_{X_d|X_1,\ldots,X_{d-1}}^{-1}.$$

These functions allow us to convert uniformly distributed random numbers into numbers that follow the target multivariate distribution.

Multivariate distributions

Simulation Algorithm: Multivariate Inverse Transform Sampling

- For each dimension k from 1 to d repeat the following:
  - Generate a random number $u_{i,k}$ from a uniform distribution over $[0, 1]$.
  - If $k = 1$: compute $x_{i,} = F_{X_1}^{-1}(u_{i,1})$.
  - Otherwise if $k = 2, \ldots, d$: compute $x_{i,k} = F^{-1}_{X_k|X_1,\ldots,X_{k-1}}(u_{i,k}, x_{i,1}, \ldots, x_{i,k-1})$.
- Repeat above N times.

Multivariate distributions

Example 6

Suppose that we want to generate samples from a bivariate distribution that follows the following form:

$$f_{X_1}(x_1) = \lambda_1 e^{-\lambda_1 x_1}, \quad x_1 \geq 0,$$

and

$$f_{X_2|X_1}(x_2|x_1) = (\lambda_2 + \alpha x_1) e^{-(\lambda_2 + \alpha x_1) x_2}, \quad x_2 \geq 0.$$

The first variable, X_1, is exponential and has the rate λ_1, and the second variable, X_2, is exponential conditionally on X_1, and has a rate $\lambda_2(X_1) = \lambda_2 + \alpha X_1$. The parameter α determines the strength of dependence between X_1 and X_2.

Multivariate distributions

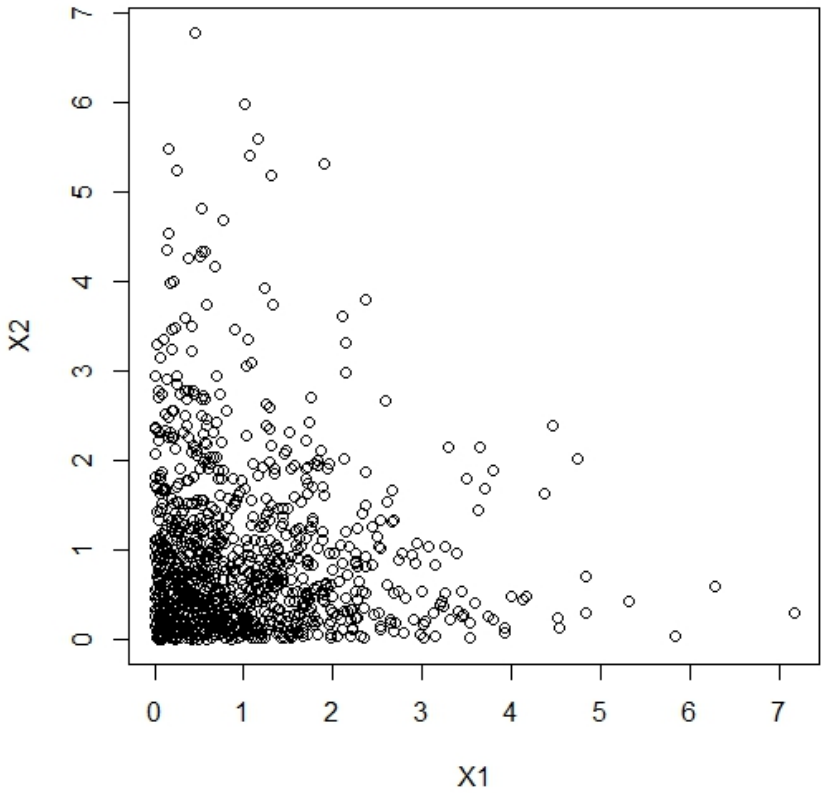
We apply the Multivariate Inverse Transform Sampling as follows:

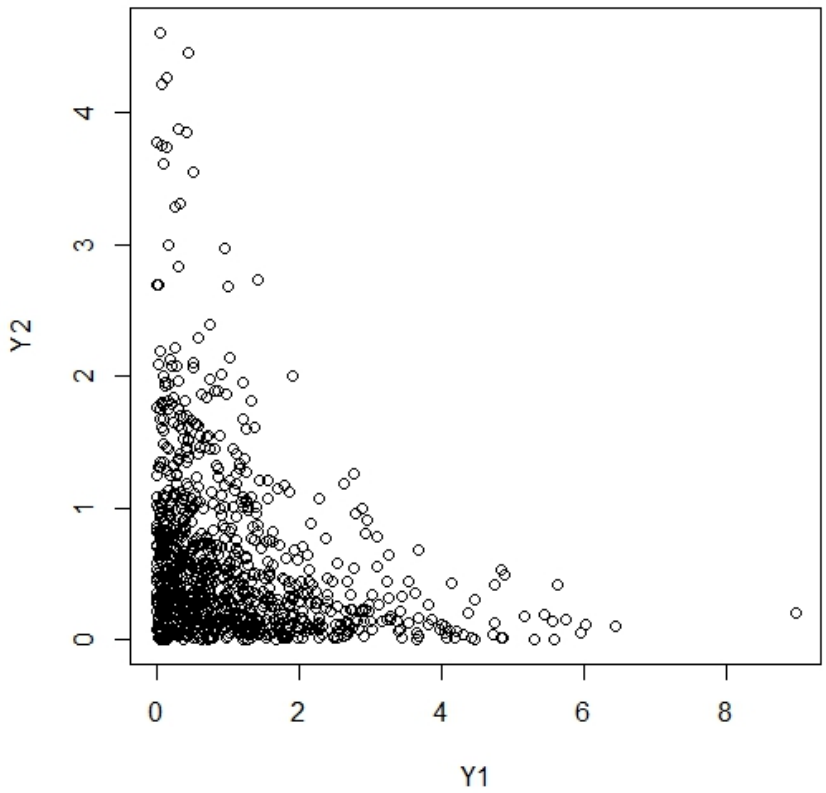
- Simulate a random number U from the uniform distribution on $[0, 1]$ and compute $X_1 = -\frac{1}{\lambda_1} \log(1 - U)$, which is the inverse transform sampling for an exponential distribution.
- Independently simulate another random number V from the uniform distribution on $[0, 1]$ and calculate X_2 using the inverse CDF conditioned on X_1, calculated as $X_2 = -\frac{1}{\lambda_2 + \alpha X_1} \log(1 - V)$.

Multivariate distributions

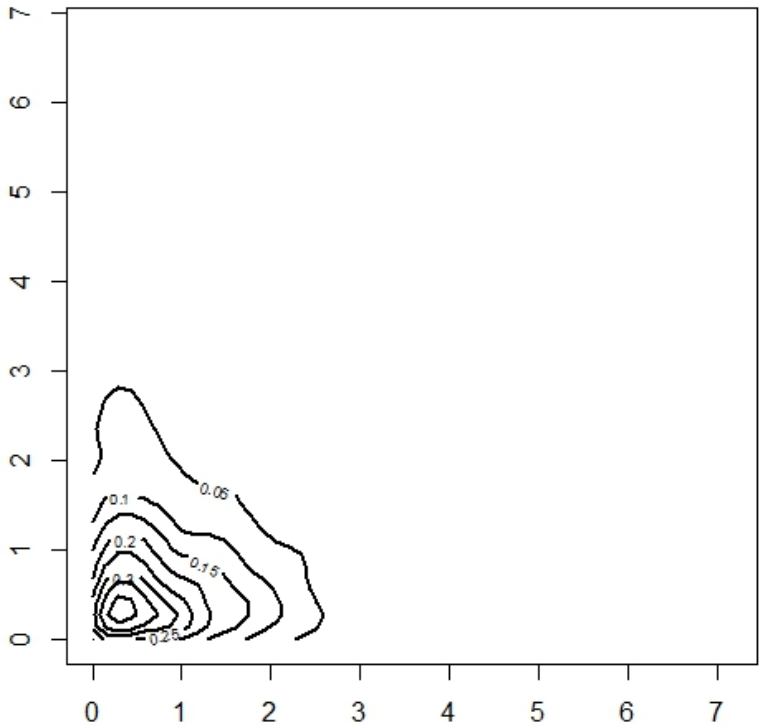
```
set.seed(1);
n=1000; lambda1=1; lambda2=1;
alpha=0.1; X1=NULL; X2=NULL;
for(i in 1:n)
 {
 x=rexp(1,lambda1); X1=c(X1,x);
 y=rexp(1,lambda2+alpha*x); X2=c(X2,y);
 }
par(mfrow=c(1,2))
plot(X1,X2)

alpha=1; Y1=NULL; Y2=NULL;
for(i in 1:n)
 {
 x=rexp(1,lambda1); Y1=c(Y1,x);
 y=rexp(1,lambda2+alpha*x); Y2=c(Y2,y);
 }
plot(Y1,Y2)
```

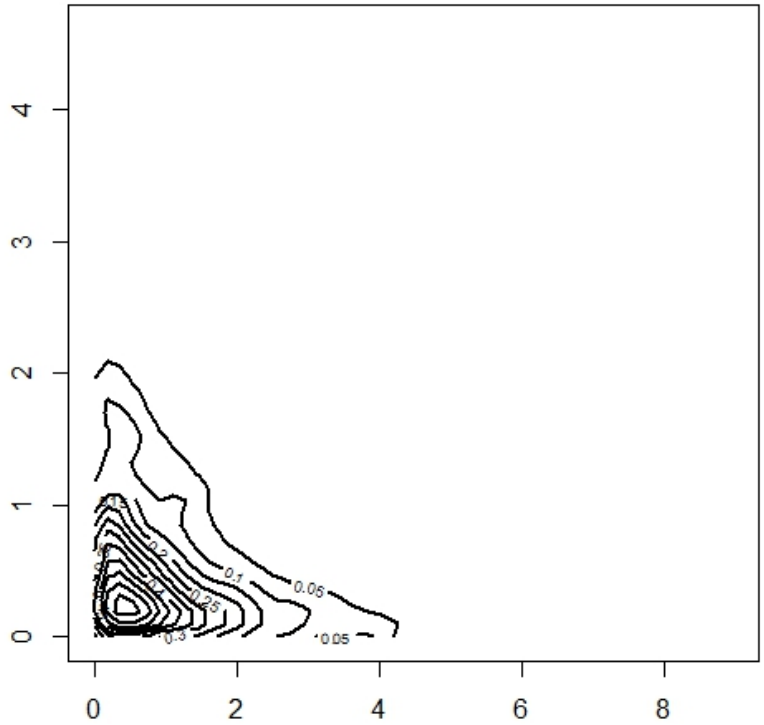
Multivariate distributions





Multivariate distributions





Decomposition methods

To generate a multivariate vector $\mathbf{Y} = (Y_1, \ldots, Y_m)^T$ with dependent components, follow this approach: If the transformed vector $\mathbf{X} = L\mathbf{Y} + \mu$ retains the same type of distribution as the independent components $\mathbf{Y}$, where L is a transformation matrix and μ is a vector that adjusts the mean, then employ this method.

Decomposition methods

Theorem 7 (Cholesky Decomposition)

Let $\mathbf{Y}$ be a random vector in $\mathbb{R}^m$ with independent components and variance 1, and mean vector μ. Let L be a deterministic matrix of size $m \times m$. Then $\mathbf{X} = L\mathbf{Y} + \mu^T$ has a multivariate distribution with the mean vector $L\mu + \mu^T \in \mathbb{R}^m$ and the covariance matrix $\Sigma = LL^T$.

Proof.

- *Mean of $\mathbf{X}$:* $\mathbb{E}[\mathbf{X}] = \mathbb{E}[L\mathbf{Y} + \mu^T] = L\mathbb{E}[\mathbf{Y}] + \mu^T = L\mu + \mu^T$.
- *Covariance Matrix of $\mathbf{X}$:*

$$\begin{aligned}\Sigma = \mathrm{Cov}(\mathbf{X}) &= \mathbb{E}[(\mathbf{X} - \mathbb{E}[\mathbf{X}])(\mathbf{X} - \mathbb{E}[\mathbf{X}])^T] \\ &= \mathbb{E}[L(\mathbf{Y} - \mu)(\mathbf{Y} - \mu)^T L^T] \\ &= L\mathbb{E}[(\mathbf{Y} - \mu)(\mathbf{Y} - \mu)^T]L^T = LIL^T = LL^T.\end{aligned}$$

Decomposition methods

Simulation Algorithm:

(a) For given Σ, apply the Cholesky method to get L.

(b) Generate an m-dimensional vector $\mathbf{Y}$ with independent standard normal components.

(c) Calculate $\mathbf{X} = L\mathbf{Y} + \mu^T$.

(d) Repeat (b)-(c) n times. As a result, we obtain a sample of size n from a multivariate normal distribution with the covariance matrix $\Sigma = LL^T$.

Decomposition methods

Example 8 (Simulation from Bivariate Normal)

Suppose that we want to generate samples from a bivariate normal distribution with mean vector $\mu = [0, 1]^T$ and covariance matrix

$$\Sigma = \begin{bmatrix} 1 & 0.7 \\ 0.7 & 1 \end{bmatrix}.$$

- The Cholesky decomposition of Σ is:

$$L = \begin{bmatrix} 1 & 0 \\ 0.7 & \sqrt{0.51} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0.7 & 0.714 \end{bmatrix}.$$

- Transform standard normal samples (y_1, y_2):

$$(x_1, x_2)^T = L\,[y_1, y_2] = \begin{bmatrix} 1 & 0 \\ 0.7 & 0.714 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} y_1 \\ 0.7y_1 + 0.714y_2 \end{bmatrix}.$$

Bootstrap

If we do not have a model assumption, we can apply non-parametric data generation techniques. Our objective is to randomly choose individuals from a dataset $\mathcal{D} = \{X_1, \ldots, X_n\}$. Each individual should have the same probability of being selected, and we repeat the selection process N times.

```
X=rnorm(1000);
Y=sample(X,1000,replace=TRUE);
qqplot(X,Y)
```

Bootstrap

Inverse transformation method applied to the empirical cumulative distribution function (ECDF) is equivalent to the bootstrap procedure.
Let $r(x)$ denote the number of times a value x appears in the dataset:

$$r(x) = \sum_{j=1}^{n} 1\{X_j = x\}.$$

The empirical CDF at the point x is:

$$\widehat{F}_n(x) = \frac{1}{n} \sum_{j=1}^{n} 1\{X_j \leq x\}.$$

Define x^- as the largest value less than x in the dataset (hence, both x^- and x are random numbers). The probability that U falls within this interval, thus selecting x by the inverse CDF, is:

$$\mathbb{P}\left(U \in \left(\widehat{F}_n(x^-), \widehat{F}_n(x)\right]\right) = \frac{r(x)}{n}.$$

This probability is the same as the bootstrap method in which each data point is equally likely to be chosen.

Synthpop package

The `synthpop` package for `R` is a tool specifically designed for generating synthetic datasets from the original dataset, trying to keep the statistical properties of individual-level data. The methodology used is based on the concept of multiple imputation, originally designed to handle missing data. This approach has been adapted in the `synthpop` package to facilitate the generation of synthetic datasets that allow the analysis of sensitive datasets without compromising individual privacy.

Synthpop package

The package implements data generation using a sequential procedure. We will discuss potential issues with its implementation. If our goal is to generate synthetic data from a multivariate distribution, we need to provide a proper sequence in which the variables are generated. In other words, we would like to preserve not only the univariate characteristics of the original dataset, but also the dependence structure between the variables. The package does not allow to verify if the dependence structure is well-preserved. Furthermore, if there is no sequential relation between the variables in the original dataset, the package will not generate the proper synthetic dataset.

We propose a solution to this problem. Rather than sequentially fitting models to the data, which might lead to compounding errors and potential privacy leaks, we propose a method that involves fitting each variable independently, using all other variables in the data set as predictors.

Sequential modeling

- We have observations $(X_{j1}, \ldots, X_{jp})$, where $j = 1, \ldots, n$, drawn from a multivariate vector $(X_1, \ldots, X_p)$.
- We aim to generate synthetic data $(Y_{j1}, \ldots, Y_{jp})$, $j = 1, \ldots, m$, which should ideally retain the statistical properties of the original data $(X_{j1}, \ldots, X_{jp})$.
- The function $f_{i+1}(\cdot \mid X_i, \ldots, X_1)$ represents the conditional distribution of X_{i+1} given all previously observed variables. The choice of the model for f is versatile and can depend on the nature of the data.
- The visit sequence $(1', \ldots, p')$ is an ordered list that determines the sequence in which the variables are synthesized, ensuring that the dependencies between the variables are respected in the synthetic data set.

Sequential modeling

- The predictor matrix $\mathbf{P}$ is a square matrix with dimensions $p \times p$, where p is the number of variables in the dataset. Each row and column of the matrix correspond to one of the variables $X_1, X_2, \ldots, X_p$. Each entry $\mathbf{P}_{ij}$ in the matrix is binary: $\mathbf{P}_{ij} = 1$ indicates that variable X_j is used as a predictor of variable X_i.
- Model fitted for the $(i+1)'$-th variable in the visiting sequence is denoted by $f_{(i+1)'}(\cdot \mid \mathbf{X}_{i'})$, where the set of predictors $\mathbf{Y}_{i'}$ is composed of those variables from the set $(X_{1'}, X_{2'}, \ldots, X_{i'})$ that are indicated by the input predictor matrix $\mathbf{P}$.

Sequential modeling

The following simulation algorithm is designed to generate synthetic data by sequentially modeling each variable in the dataset based on the previously synthesized variables. Ideally, this method ensures that the dependencies between variables are maintained, reflecting the structure present in the original data. However, we will demonstrate that this is not always the case if we use the package. As such, we propose some solutions to deal with this issue.

The process begins by selecting an initial variable from the observed data, generating its synthetic counterpart by e.g. bootstrapping, and then progressively modeling and generating synthetic values for each subsequent variable using a fitted model. The approach is iterative, with each synthetic variable conditioned on all previously generated synthetic variables. Note that the sequence of selecting of variables depend on the input visiting sequence.

Sequential modeling

- According to the visit sequence denoted by $1', \ldots, p'$, select the initial variable $X_{1'}$. Generate Y_1 by random sampling with replacement from $X_{1'}$.
- For each subsequent variable $X_{(i+1)'}$ in the visit sequence, select a subset of variables from $X_{1'}, X_{2'}, \ldots, X_{i'}$ according to the input predictor matrix $\mathbf{P}$, and fit a model $f_{(i+1)'}(\cdot \mid X_{i'}, \ldots, X_{1'})$. The predictor matrix specifies which variables among $X_{i'}, X_{(i-1)'}, \ldots, X_{1'}$ are used to predict $X_{(i+1)'}$. This model represents the conditional distribution of $X_{(i+1)'}$ given the selected predictors.
- Draw synthetic values $Y_{(i+1)'}$ from the model $f_{(i+1)'}(\cdot \mid X_{i'}, \ldots, X_{1'})$.
- Repeat the process for all variables in the dataset.

Sequential modeling

- For each fitted model, specific model assumptions are necessary. For continuous data, linear regression may be used, while for categorical data, models such as logistic regression may be appropriate.
- The choice of the initial variable and the sequence of selection variables must be specified. The predictor matrix that indicates the relationships between the variables should be specified.
- Random noise can be added to each predicted variable. There are two methods to incorporate this noise: one method is to add random noise directly to the predicted model, such as

$$Y_{(i+1)} = \hat{\beta}_i Y_{i'} + \cdots + \hat{\beta}_1 Y_{1'} + \varepsilon,$$

where $\hat{\beta}_1, \ldots, \hat{\beta}_i$ are estimates of the original model. Alternatively, a Bayesian approach can be used to estimate the distribution of β, from which samples are drawn, assuming normal noise. For example, this method would use $Y_{(i+1)'} = \tilde{\beta} Y_{i'}$, where $\tilde{\beta} \sim \mathcal{N}(\hat{\beta}, \sigma(\hat{\beta}))$.

Sequential modeling

The method used by `synthpop` involves generating synthetic data by sequentially modeling each variable based on previously synthesized variables. The algorithm introduced above is used, and the models can be chosen from various options. For a description of several commands from the package, refer to MSc thesis of Chang Qu.

Example 1

We illustrate the implementation of the method using a specific model. Assume that we have two random variables, age (X_1) and income (X_2). The sequence for this example is as follows: first, we generate the age and then generate income based on the age. Therefore, $1' = 1$ and $2' = 2$. The predictor matrix $\mathbf{P}$ is

$$\mathbf{P} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix}.$$

This matrix indicates that income (X_2) is predicted using age (X_1), but age (X_1) is not predicted using any other variable.

Example 1

The basic implementation is as follows.

1. Create Y_1 by bootstrapping X_1.
2. Fit the linear model $X_2 = \beta X_1 + Z$, where Z is $\mathcal{N}(0, \sigma_Z^2)$. As a result, we obtain the estimate $\hat{\beta}$. Calculate the residuals to estimate σ_Z^2; denote it by $\hat{\sigma}_Z^2$.
3. Simulate Y_2 using the formula $Y_2 = \hat{\beta} Y_1 + \tilde{Z}$, where $\tilde{Z}$ is centered normal with variance $\hat{\sigma}_Z^2$.

Example 1

Assume that we have a dataset with 10 observations:

ID	age	income
1	22	30000
2	25	35000
3	28	40000
4	30	45000
5	35	50000
6	40	60000
7	45	70000
8	50	80000
9	55	85000
10	60	90000

Example 1

In the first step, we select the initial variable: we randomly sample age_{syn} with replacement from the observed age values and obtain:

ID	age_{syn}
1	22
2	25
3	28
4	22
5	35
6	40
7	45
8	50
9	28
10	60

Example 1

In the next step, we fit the linear model:

$$\text{income} = 1230.77 + 1500 \cdot \text{age} + Z, \quad Z \sim N(0, \sigma_Z^2).$$

In the final step, we generate synthetic values: we draw synthetic values $\text{income}_{\text{syn}}$ from the model $f(\text{income}_{\text{syn}} \mid \text{age}_{\text{syn}})$ using normal noise with estimated σ_Z^2. The final synthetic dataset is:

Observation	age_{syn}	$\text{income}_{\text{syn}}$
1	40	60257.33
2	22	33786.17
3	30	45652.61
4	30	40345.76
5	50	75937.89
6	60	95796.99
7	30	49990.02
8	40	60031.50
9	28	47308.36
10	35	49256.03

Example 2

Given the variables X_1, X_2, X_3, X_4 and the visiting sequence $1' = 3, 2' = 4, 3' = 2, 4' = 1$, the input predictor matrix $\mathbf{P}$ is as follows:

$$\mathbf{P} = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}.$$

The rows and columns correspond to the variables X_1, X_2, X_3, X_4, and the matrix indicates which variables are used as predictors for the others.
In row 1 of the predictor matrix, we have:

$$P_{11} = 0, \quad P_{12} = 1, \quad P_{13} = 1, \quad P_{14} = 1,$$

indicating that X_2, X_3, and X_4 are predictors of X_1.
In row 2, we have:

$$P_{21} = 0, \quad P_{22} = 0, \quad P_{23} = 1, \quad P_{24} = 1,$$

indicating that X_3 and X_4 are predictors of X_2.

Example 2

In row 3, we have:

$$P_{31} = 0, \quad P_{32} = 0, \quad P_{33} = 0, \quad P_{34} = 0,$$

indicating that no variables predict X_3, so we resample it.
In row 4, we have:

$$P_{41} = 0, \quad P_{42} = 0, \quad P_{43} = 1, \quad P_{44} = 0,$$

indicating that X_3 is a predictor of X_4.
Combining the predictor matrix with the visiting sequence $3, 4, 2, 1$, as introduced before, we have:

- Y_3 is generated first by resampling;
- Y_4 is generated as the second, with Y_3 as predictor.
- etc.

In conclusion, the visiting sequence determines the order in which the variables are synthesized, while the predictor matrix defines the variables used as predictors for each step of the process.

Example 3

We can also apply a different visiting sequence for the same input predictor matrix. For example, if we define the visiting sequence as $4, 3, 1, 2$, the synthesis process changes accordingly. When synthesizing Y_4, no predictors are used, even though $P_{43} = 1$ in the input predictor matrix, (because Y_4 is visited before Y_3). For Y_3, the input predictor matrix indicates that no other predictors are used, so Y_3 is also resampled. When synthesizing Y_1, the predictor matrix shows that Y_2, Y_3, and Y_4 are predictors, but since only Y_3 and Y_4 have been visited, they are used as predictors. Finally, when synthesizing Y_2, the input predictor matrix indicates that Y_3 and Y_4 are predictors and both have been visited. Therefore, the output predictor matrix is as follows:

$$\mathbf{P}_{\text{out}} = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}.$$

Sequential modeling

In conclusion, the input predictor matrix and the output predictor matrix may be different. The input predictor matrix reflects the relationships or dependencies between variables, as defined before data generation. The output predictor matrix represents the actual model fitting during the synthesis, based on the visiting sequence and the predictors that have been visited up to that point. Therefore, while the input matrix indicates potential predictors, the output matrix shows which predictors were used in model fitting during the generation of synthetic data.

Sequential modeling

- The selection of the initial variable, the visiting sequence in which variables are chosen, and the design of the predictor matrix usually have a huge influence on the utility of the synthetic data generated, impacting both the marginal distributions and the dependency structures. Before modeling the data set, it is important to analyze the data to verify that the model assumptions are valid. For example, we need to identify which random variables X_i's affect X_j and how to model their dependence.
- In the `synthpop` implementation, it is assumed that there is a sequential dependency, that is, there exists an ordered subset $i_1 < \cdots < i_k$ of $\{1, \ldots, n\}$ such that for all $i \in \{1, \ldots, n\}$ the variable X_i is a function of all the variables $X_{1_i}, \ldots, X_{i_k}$. First, this sequence has to be chosen by the user. The package does not provide any tools to choose the sequence. Furthermore, the existence of such a sequential dependence structure is not guaranteed. In this case, the package may model the relationship inaccurately.

Sequential modeling

- Compared to other data generation, Machine Learning-type models, such as variational autoencoders (VAEs), generative adversarial networks (GANs) statistical data generation methods such as those in `synthpop` usually provide statistical inference tools like confidence intervals, hypothesis testing etc. The `synthpop` provides a very limited number of inferential tools and those that are provided do not seem to be implemented correctly; see our discussion on Kolmogorov-Smirnov test. This lack can be a significant drawback when the goal is not only to generate synthetic data, but also to derive statistically robust inferences from the data, which are important for understanding the variability and reliability of the data generated by the model.

Example 4

In this example we show an influence of the choice of the sequence in which variables are selected. We generate three variables which are treated as observed data:

- X_1: Generate 1000 observations from an exponential distribution with mean 5.
- X_2: Define X_2 as a linear function of X_1, specifically $X_2 = \frac{1}{2}X_1 + \varepsilon$, where ϵ follows a normal distribution with mean 0 and standard deviation 0.1.
- Define X_3 as a linear function of X_1 and X_2, specifically $X_3 = 2X_1 + 3X_2 + \varepsilon$, where ϵ follows a normal distribution with mean 0 and standard deviation 0.1.

Example 4

Using the proposed algorithm, the correct order selected is $(1, 2, 3)$. Additionally, we consider an incorrect order $(3, 2, 1)$. We then use the Synthpop package to generate data based on both orders. Notice that if the sequence is wrong, the marginal distribution and the contour plot do not match the original data. But if the sequence is correct, they match pretty well. Recall again that we cannot do a formal hypothesis testing of equality between the marginal distributions, due to dependence. Furthermore, we produce contour plots that indicate that the dependence structure between the variables is not preserved, if the sequence is not chosen in the correct order. We recall that this issue is not addressed in the package at all.

Example 4

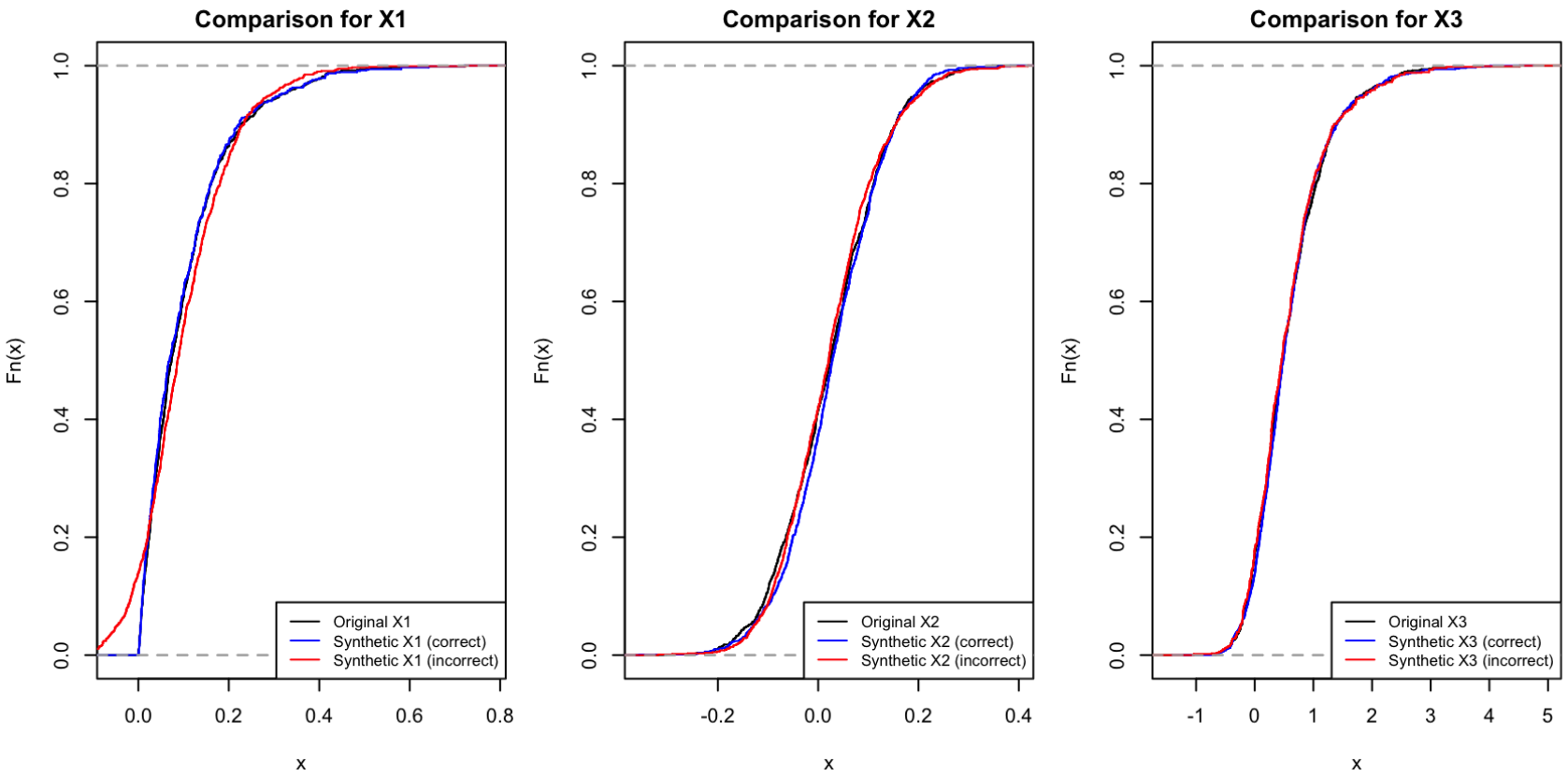
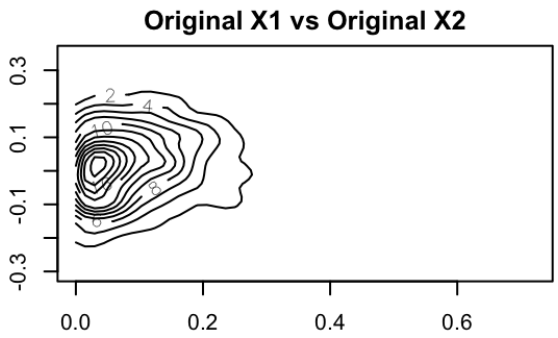
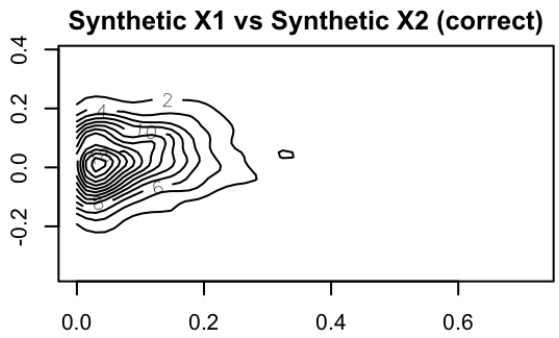
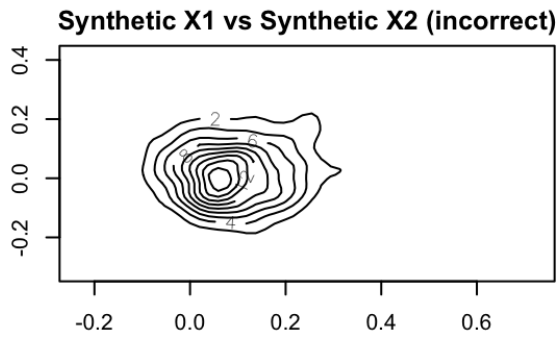


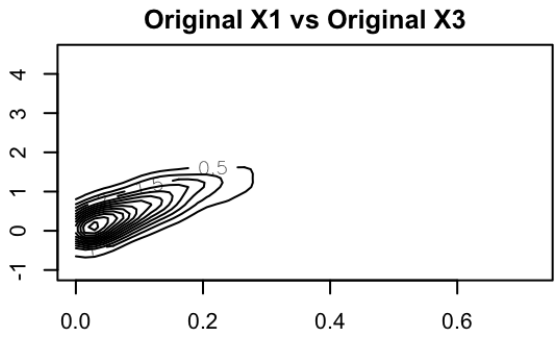
Figure: Marginal distributions: comparison between the correct and the incorrect sequence

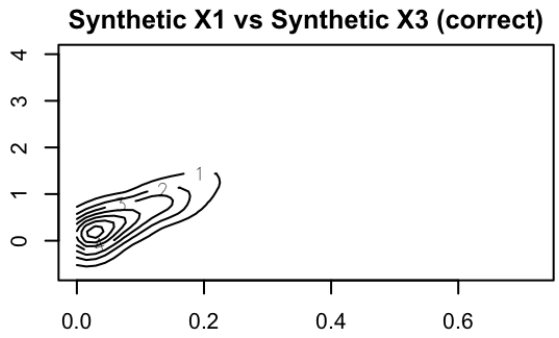
Example 4

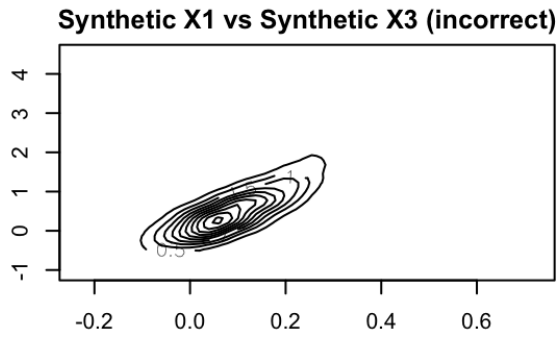

Original X1 vs Original X2
0.0
0.2
0.4
0.6
-0.3
-0.1
0.1
0.3

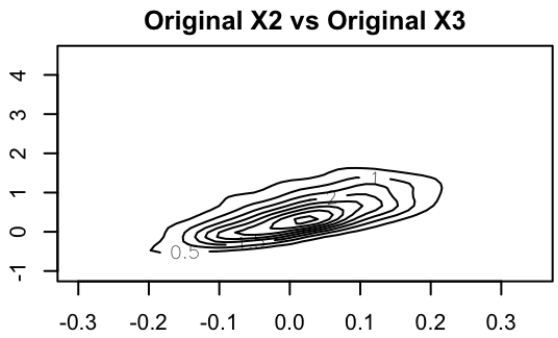

Synthetic X1 vs Synthetic X2 (correct)
0.0
0.2
0.4
0.6
-0.2
0.0
0.2
0.4

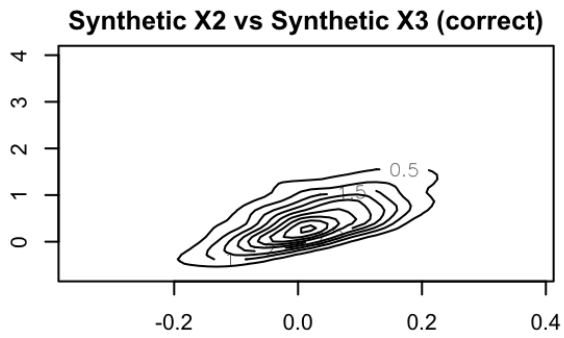

Synthetic X1 vs Synthetic X2 (incorrect)
-0.2
0.0
0.2
0.4
0.6

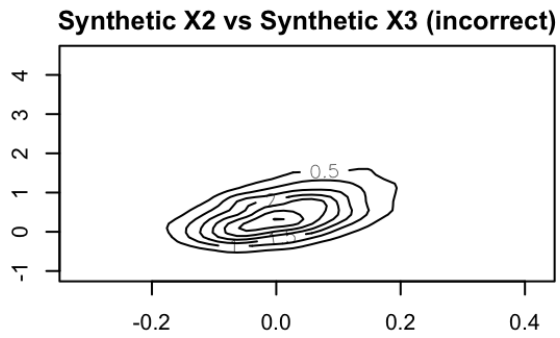

Original X1 vs Original X3
0.0
0.2
0.4
0.6
-1
0
1
2
3
4


Synthetic X1 vs Synthetic X3 (correct)
0.0
0.2
0.4
0.6
0
1
2
3
4


Synthetic X1 vs Synthetic X3 (incorrect)
-0.2
0.0
0.2
0.4
0.6
-1
0
1
2
3
4


Original X2 vs Original X3
-0.3
-0.2
-0.1
0.0
0.1
0.2
0.3
-1
0
1
2
3
4


Synthetic X2 vs Synthetic X3 (correct)
-0.2
0.0
0.2
0.4
0
1
2
3
4


Synthetic X2 vs Synthetic X3 (incorrect)
-0.2
0.0
0.2
0.4
-1
0
1
2
3
4

Example 5

In this example, the original data follow the regression model

$$X_2 = \rho X_1 + \varepsilon.$$

We ignore the dependence relationship and create the synthetic data as follows. Y_1 is bootstrapped from X_1, while $Y_2 = X_2$.

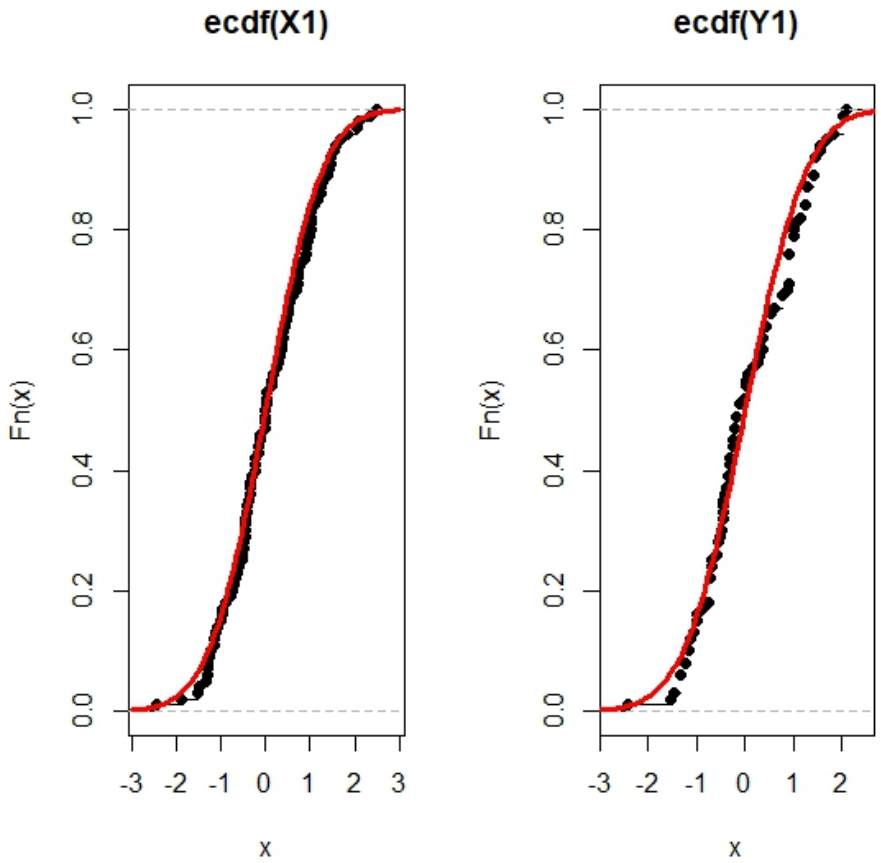
Example 5

```
library(MASS)
set.seed(15)
X1=rnorm(100); X2=0.9*X1+sqrt(1-0.9^2)*rnorm(100);
Y1=sample(X1,100,replace=TRUE); Y2=X2;
cont.1 <- kde2d(X1, X2, n = 50);
cont.2 <- kde2d(Y1, Y2, n = 50)
par(mfrow=c(1,2))
plot(ecdf(X1),lwd=1); curve(pnorm(x),xlim=c(-3,3),col=
plot(ecdf(Y1),lwd=1); curve(pnorm(x),xlim=c(-3,3),col=
contour(cont.1, lwd = 2); contour(cont.2, lwd = 2)
ks.test(X1, "pnorm",0,1); ks.test(Y1, "pnorm",0,1);
```

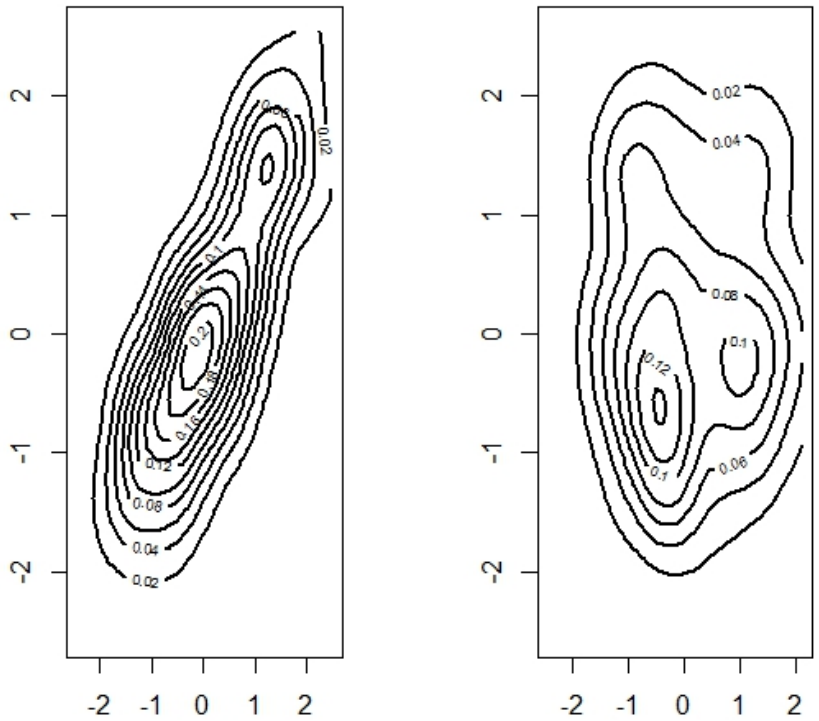
Example 5

```
        Asymptotic one-sample Kolmogorov-Smirnov test
data:  X1
D = 0.065193, p-value = 0.789
alternative hypothesis: two-sided

        Asymptotic one-sample Kolmogorov-Smirnov test
data:  Y1
D = 0.11519, p-value = 0.1407
alternative hypothesis: two-sided

Warning message:
In ks.test.default:
ties should not be present for the Kolmogorov-Smirnov
```

Example 5


ecdf(X1)
Fn(x)
0.0
0.2
0.4
0.6
0.8
1.0
-3
-2
-1
0
1
2
3
x
ecdf(Y1)
Fn(x)
0.0
0.2
0.4
0.6
0.8
1.0
-3
-2
-1
0
1
2
x

Example 5


2
1
0
-1
-2
-2 -1 0 1 2
0.02
0.04
0.06
0.08
0.1
0.12
0.2
2
1
0
-1
-2
-2 -1 0 1 2
0.02
0.04
0.08
0.1
0.12
0.1
0.06

Sequential modeling

These experiments show the importance of choosing variables in the right order when using the `synthpop` package to generate synthetic data. If the order is chosen correctly, the synthetic data look very similar to the original data, both in the marginal distribution and in their dependence structure. This indicates that the relationship between the variables is well maintained. However, when we tried the wrong order $(3, 2, 1)$, the synthetic data did not match the original well. There are differences in both marginal and dependence structures, which shows that the generated data may not be valid.

Selecting the sequence

We propose some solutions to the issue of choosing a proper sequence. In what follow, the p-value for the model refers to the null hypothesis that all parameters in the model are equal to zero, versus the alternative hypothesis that at least one parameter is not equal to zero. That is for paremeters β_i:

$$H_0 : \beta_1 = \beta_2 = \cdots = \beta_k = 0,$$

$$H_1 : \exists\ \beta_i \neq 0.$$

Selecting the sequence - Algorithm 1

- Build regression models f_i for each variable X_i as a function of all other variables X_k, where $k \neq i$. Compute for each i: $X_i = f_i(\{X_k \text{ for all } k \neq i\})$. Choose the model f_{i^*} with the lowest p-value, set the corresponding X_{i^*} aside as the last variable.
- Begin the process to order the remaining variables X_i, $i \neq i^*$:
  - For each remaining variables X_i, model it as a function of all other variables that have not yet been set aside.
  - Continue with the remaining variables:
    - Build regression models:

$$X_i = f_i(\{X_k \text{ for all } k \neq i \text{ and } X_k \text{ has not been set aside yet}\}).$$

    - Evaluate each model based on its p value. Select the model with the lowest p value and set the corresponding variable aside.
  - Once the sequence of variables $(X_1, X_2, \ldots, X_p)$ is established, use `synthpop` to generate data reflecting this sequence.

Selecting the sequence - Algorithm 1

The correct correlation between the variables will be maintained if there is an existing a sequential correlation.
The above procedure is described specifically for the linear regression.
However, at each step we can choose any other *parametric* regression model, possibly with penalization.

Example 6

In this experiment, we generate data from a linear model with three variables: X_1, X_2, and X_3. The relationships are defined as follows:

- X_2 is linearly dependent on X_1;
- X_3 is dependent on both X_1 and X_2.

Specifically, we simulate the data with $X_1 \sim \text{Exp}(2)$,

$$X_2 = 2X_1 + \varepsilon$$

with $\varepsilon \sim \mathcal{N}(0, 1)$, and

$$X_3 = 3X_1 + 5X_2 + \eta$$

with $\eta \sim \mathcal{N}(0, 1)$.

Example 6

```
set.seed(5)
X1=rexp(100,2);
X2=2*X1+rnorm(100);
X3=3*X1+5*X2+rnorm(100);
a3<-lm(X3~X1+X2);
a2<-lm(X2~X1+X3);
a1<-lm(X1~X2+X3);
```

Example 6

```
Call:
lm(formula = X3 ~ X1 + X2)

Residuals:
     Min       1Q   Median       3Q      Max
-3.4200  -0.5406  -0.0728   0.6449   2.2520

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.003716   0.142742  -0.026    0.979
X1           2.878458   0.285251  10.091   <2e-16 ***
X2           5.045631   0.089971  56.081   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' '
1
```

Example 6

```
pf(summary(a1)$fstatistic[1],summary(a1)$fstatistic[2],
summary(a1)$fstatistic[3],lower.tail=FALSE);
        value
3.823904e-30
pf(summary(a2)$fstatistic[1],summary(a2)$fstatistic[2],
summary(a2)$fstatistic[3],lower.tail=FALSE);
        value
6.080917e-89
pf(summary(a3)$fstatistic[1],summary(a3)$fstatistic[2],
summary(a3)$fstatistic[3],lower.tail=FALSE);
        value
7.707436e-94
```

Example 6

We apply Algorithm 1, which iteratively fits linear models and selects the variable with the lowest p-value, stopping when one variable remains. The final sequence is reversed to correctly preserve the dependencies for synthetic data generation. This process assumes a linear model. First, we fit the model for X_1 as a function of X_2 and X_3:

$$X_1 = \beta_0 + \beta_1 X_2 + \beta_2 X_3.$$

The p-value for this model is $p_{X_1} = 3.823904 * 10^{-30}$.
Next, we fit the model for X_2 as a function of X_1 and X_3:

$$X_2 = \beta_0 + \beta_1 X_1 + \beta_2 X_3.$$

The p-value for this model is $p_{X_2} = 6.080917 * 10^{-89}$.
Finally, we fit the model for X_3 as a function of X_1 and X_2:

$$X_3 = \beta_0 + \beta_1 X_1 + \beta_2 X_2.$$

The p-value for this model is $p_{X_3} = 7.707436 * 10^{-94}$.

Example 6

Since X_3 had the lowest p-value, we selected it as the last variable to fit. We then proceeded by fitting X_1 as a function of X_2 and vice versa. Both had the same p-value of $4.048605 * 10^{-17}$, indicating that the order of the last two variables does not matter. In linear regression, the sequence of selecting the final two variables is irrelevant. The resulting sequence is X_3, X_1, X_2, which is the correct sequence for the data.