Using Gaussian copula to generate a synthetic population

Yijun Wei
NISS, USDA-NASS
Yijun.Wei@nass.usda.gov
Luca Sartore
NISS, USDA-NASS
Luca.Sartore@nass.usda.gov
Nell Sedransk
NISS
NSedransk@niss.org

Disclaimer

The Findings and Conclusions in This Preliminary Presentation Have Not Been Formally Disseminated by the U.S. Department of Agriculture and Should Not Be Construed to Represent Any Agency Determination or Policy

Outline

- Purpose of research
- Census of Agriculture
- Gaussian copula
- R package dplyr and ggplot2
  - Comparing dplyr and ggplot2 with base R
  - Generating synthetic population using dplyr and ggplot2
- Conclusion

Purpose of presentation

- Generating a complex synthetic population
    - To protect confidential information provided by responders
    - To maintain pairwise statistical relationships among variables
    - To handle continuous variable and categorical variable simultaneously

- Introducing statistical R packages ggplot2, dplyr that are used for visualization, data processing respectively

Census of Agriculture

- Every five years, USDA's National Agricultural Statistics Service (NASS) conducts the Census of Agriculture
  - The Census provides a detailed picture of U.S. farms, ranches and the people who operate them
  - It is the only source of uniform, comprehensive agricultural data for every state and county in the United States
  - NASS also obtains information on most commodities from administrative sources or surveys of non-farm populations (e.g. cotton ginning data)

Census of Agriculture data overview

- Generate a synthetic population based on a subset of the Census of Agriculture data
  - The subset contains 25 predictors
    - For example, principal operator’s race, principal operator’s sex, etc…, but they are relabeled as X1, …, X25
  - There are continuous and categorical variables
  - The total number of observations in the subset is 800,000
  - No missing value

Gaussian copula

- Copulas are used to describe the dependence among random variables
- A copula is a multivariate probability distribution for which the marginal probability distribution of each variable is uniform
  - The marginal CDFs $F_i(x)$ of a random vector *X*$(X_1, X_2, \ldots, X_d)$ follows a uniform distribution U_i
  - The copula is defined as

$$C(u_1, u_2, \ldots, u_d) = P\,(U_1 \le u_1, U_2 \le u_2, \ldots, U_d \le u_d) = P(F_1^{-1}(u_1), F_2^{-1}(2) \ldots, F_d^{-1}(u_d))$$

- Gaussian copula is constructed from a multivariate normal distribution with correlation matrix *P*:

$$C_P(u_1, u_2, \ldots, u_d) = \Phi_P(\Phi_1^{-1}(u_1), \Phi_2^{-1}(2) \ldots, \Phi_d^{-1}(u_d))$$

where Φ denotes the standard normal distribution function, and Φ_P denotes the multivariate standard normal distribution function with correlation matrix *P*

Gaussian copula applied in generating synthetic population

- Perform a cholesky decomposition of correlation matrix *P*, and set *A* as the resulting lower triangular matrix

- Repeat the following steps *n* times
  - Generate a vector $Z = (Z_1, \ldots, Z_d)'$ of independent standard normal deviates
  - Set $X = AZ$
  - Return $U = (\Phi(X_1), \ldots, \Phi(X_d))'$

- Can be achieved using *mvrnorm* in *MASS* library

USDA

R packages introduction

- A Grammar of Data Manipulation (dplyr)
  - R-package used for data processing
  - Transform and summarize tabular data with rows and columns.
  - Contain a set of functions (or “verbs”) that perform common data manipulation operations
- Create Elegant Data Visualizations Using the Grammar of Graphics (ggplot2)
  - R-package used for data visualization
  - Consistent underlying grammar of graphics (a graphic version of dplyr)
  - Plot specification at a high level of abstraction
  - Very flexible and elegant

R packages introduction - dplyr

- Commonly used command:
  - mutate() adds new variables that are functions of existing variables
  - select() picks variables based on their names
  - filter() picks cases based on their values
  - summarise() reduces multiple values down to a single summary
  - arrange() changes the ordering of the rows
  - group_by() allows for group operations in the “split-apply-combine” concept

R packages Introduction – dplyr - Continue

- Base R:
  - Create new dataset:

```
pop_census$new1 <- pop_census$X10 + 0.5 * pop_census$X11
pop_census$new2 <- pop_census$X13 + 3 * pop_census$X14
pop_census$new3 <- log(pop_census$X15)
pop_census$new4 <- pop_census$X16 * 4
```

  - Select Variable:

```
pop_census[ ,c('X14','X15', 'X16', 'new1', 'X18')]
```

  - Filter variable:

```
pop_census [pop_census$X10 >=3 & pop_census$X16 >=1 & pop_census$X17 <=7,]
```

  - Sort by variable value:

```
pop_census[ order(pop_census$X19),]
```

USDA

R packages Introduction – dplyr - Continue

- dplyr
  - This can be done with one function using dplyr:

```
pop_census <- pop_census %>% mutate(new1 = X10 + 0.5 * X11,
                                    new2 = X13 + 3 * X14,
                                    new3 = log(X15),
                                    new4 = X16 * 4) %>%
                             dplyr::select (X14 , X15, X16, new1, X18)  %>%
                             filter (X10 >=1, X16 >=1 , X17 <=7) %>%
                             arrange(X19)
```

```
pop_census %>%
  group_by(X2) %>%
  summarise(avg_X15 = mean(X15),
            min_X15 = min(X15),
            max_X15 = max(X15),
            total = n())
```

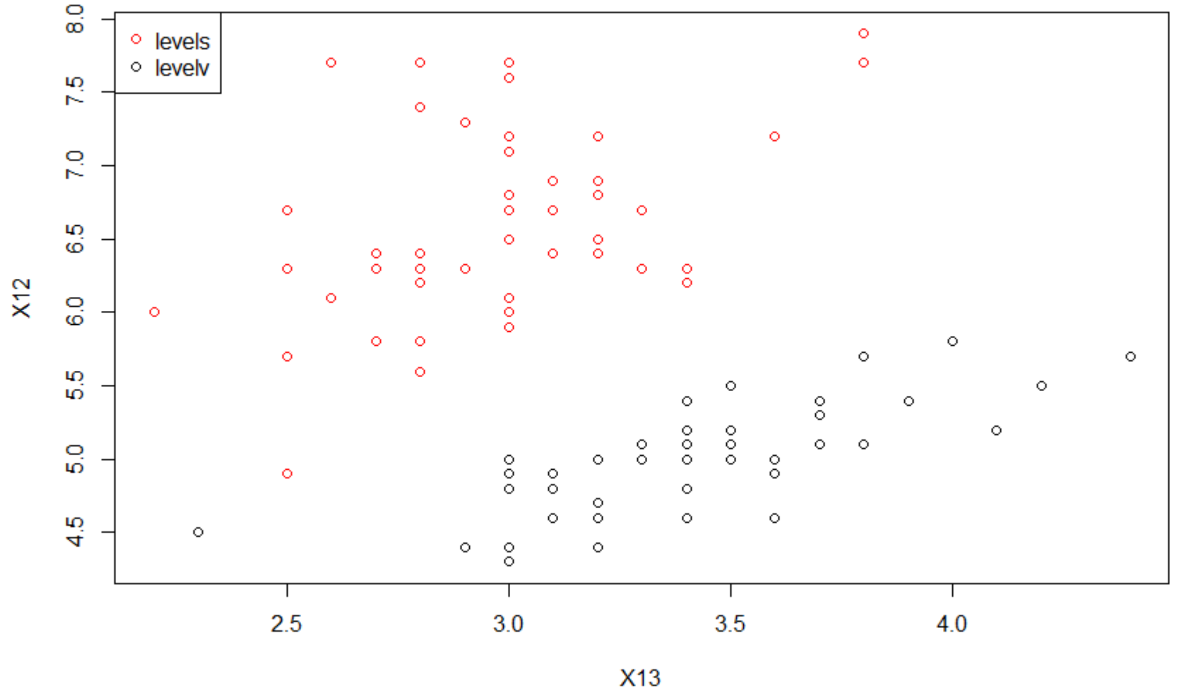
R packages introduction – ggplot2

- Graphic version of dplyr (using ‘+’ to replace ‘%>%’)
- Building blocks of a graph include:
  - data
  - aesthetic mapping
  - geometric object
  - statistical transformations
  - scales
  - coordinate system
  - position adjustments
  - faceting

R packages introduction – ggplot2 - Continue

- base R

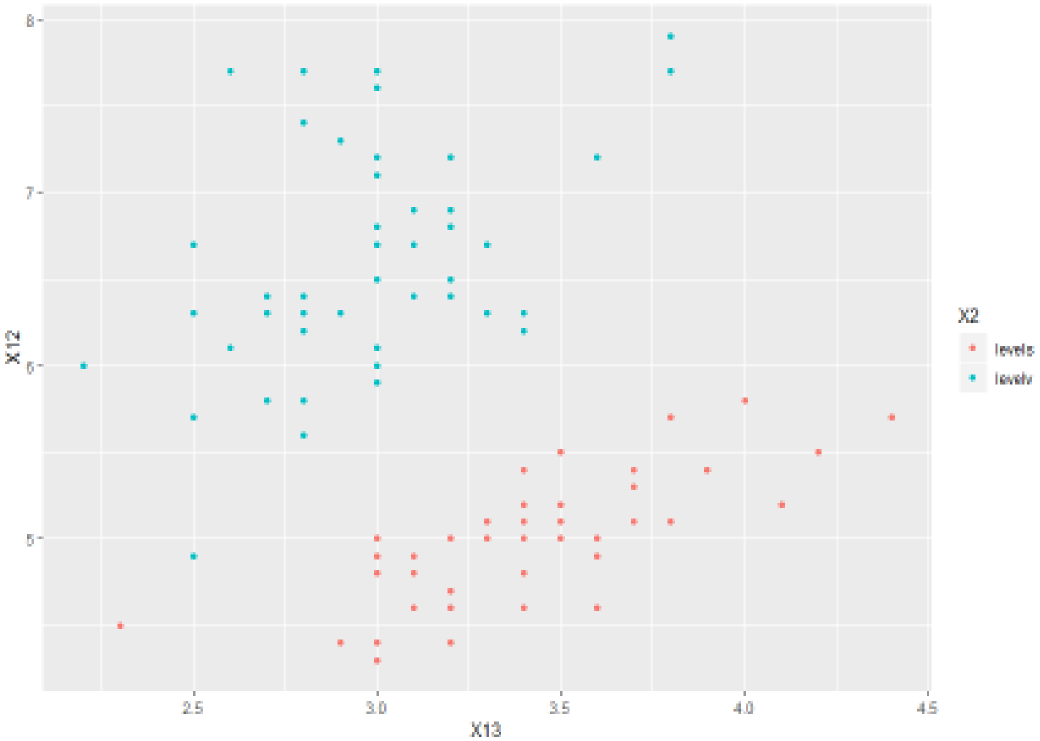
```
plot(X12 ~ X13,
     col = factor(X2),
     data = filter(pop_census, X2 %in% c("levels", "levelv")))
legend("topleft",
       legend = c("levels", "levelv"),
       col = c("red", "black"),
       pch = 1)
```

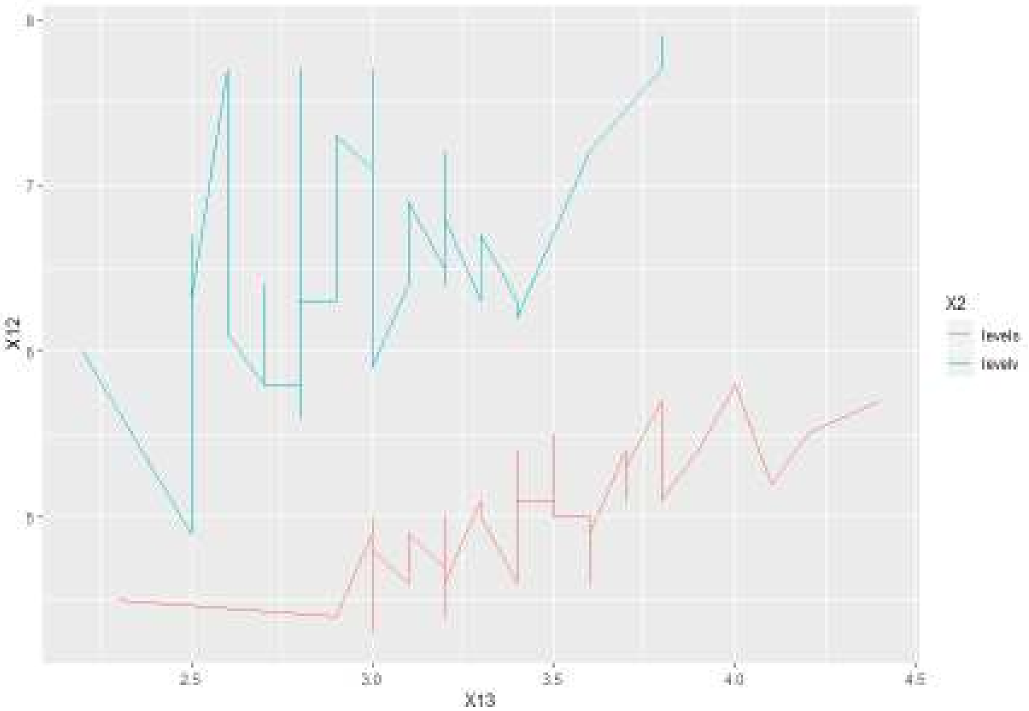


R packages Introduction – dplyr - Continue

- ggplot2

```
ggplot(filter(pop_census, X2 %in% c("levels", "levelv")),
           aes(x=X13,
                y=X12,
                color=X2))+
   geom_point()
```





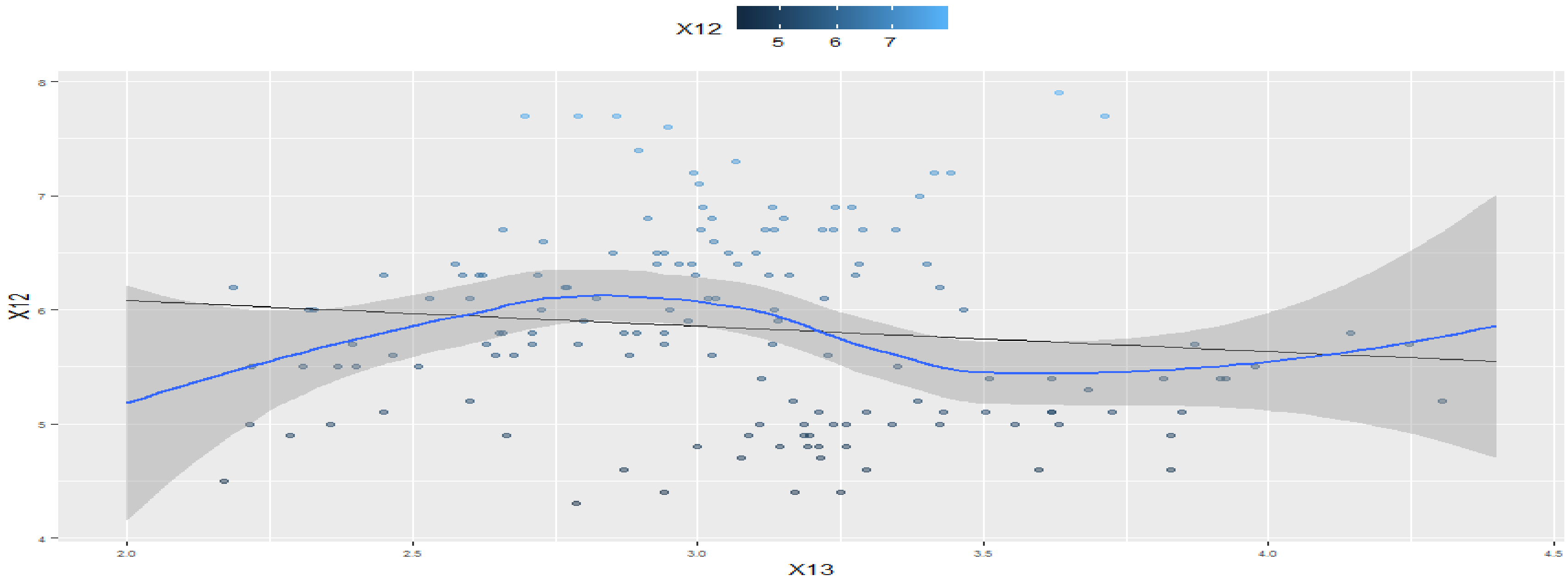
R packages Introduction – ggplot2 - Continue

- More advanced features

```
pop_census$pred.X12 <- predict(lm(X12 ~ X13, data = pop_census))

p1 <- ggplot(pop_census, aes(x = X13, y = X12)) +  theme(legend.position="top",
                                                          axis.text=element_text(size = 6))

p1 + geom_point(aes(color = X12),alpha = 0.5,
                size = 1.5,
                position = position_jitter(width = 0.25, height = 0)) +
  geom_line(aes(y = pred.X12)) +   geom_smooth()
```



Step by step R function

- Categorical variables are converted to continuous before using copula
  - The frequency of the categorical variables' levels are used as the value for that level

```
dat[,cat] = apply(dat[,cat],2,function(x){
                t = as.data.frame(table(x))
                t$Freq[match(x,t[,1])]/length(x)
              })
```

- CDF of the variable are transformed to be used in copula
  - U_i is transformed to $\Phi_i^{-1}(u_i)$ using *ecdf* and *qnorm*

```
prepare_copula_qnorm <- function (var){
  #This function is designed to calculate the copula
  new_var = qnorm(ecdf(var)(var)*0.99 + 0.005)
  return (new_var)
}
```

Step by step R function - Continue

```
dat <- dat %>% mutate (New_X1 = case_when( X1 == 1 ~ 1,
                                           X1 ==1 ~ 2,
                                         TRUE~ 0 ),
                       New_X2 = case_when((X2>=9 & X2 <=16) ~ 2,
                                     TRUE ~ 1),
                       New_X3= case_when(X3<=9 ~ 1,
                                   (X3>=10 & X3 <=49) ~ 2,
                                   (X3>=50 & X3 <=69) ~ 3,
                                   (X3>=70 & X3 <=99) ~ 4,
                                   (X3>=100 & X3 <=139) ~ 5,
                                   (X3>=140 & X3 <=179) ~ 6,
                                   (X3>=180 & X3 <=219) ~ 7,
                                   (X3>=220 & X3 <=259) ~ 8,
                                   (X3>=260 & X3 <=499) ~ 9,
                                   (X3>=500 & X3 <=999) ~ 10,
                                   (X3>=1000 & X3 <=1999) ~ 11,
                                   (X3>=2000) ~ 12))
dat[,cat] = apply(dat[,cat],2,function(x){
                t = as.data.frame(table(x))
                t$Freq[match(x,t[,1])]/length(x)
              })
```

Step by step R function - Continue

```
  dat <- dat %>% mutate (
                        New_X1_continuous = prepare_copula_qnorm(dat$New_X1),
                        New_X2_continuous= prepare_copula_qnorm(dat$New_X2),
                        New_X3_continuous = prepare_copula_qnorm(dat$New_X3),
                        X4_continuous = prepare_copula_qnorm(dat$X4),X5_continuous= prepare_copula_qnorm(dat$X5),
                        X6_continuous = prepare_copula_qnorm(dat$X6),X7_continuous= prepare_copula_qnorm(dat$X7),
                        X8_continuous = prepare_copula_qnorm(dat$X8),X9_continuous= prepare_copula_qnorm(dat$X9),
                        X10_continuous = prepare_copula_qnorm(dat$X10),X11_continuous= prepare_copula_qnorm(dat$X11),
                        X12_continuous = prepare_copula_qnorm(dat$X12),X13_continuous= prepare_copula_qnorm(dat$X13),
                        X14_continuous = prepare_copula_qnorm(dat$X14),X15_continuous= prepare_copula_qnorm(dat$X15),
                        X16_continuous = prepare_copula_qnorm(dat$X16),X17_continuous= prepare_copula_qnorm(dat$X17),
                        X18_continuous = prepare_copula_qnorm(dat$X18),X19_continuous= prepare_copula_qnorm(dat$X19),
                        X20_continuous = prepare_copula_qnorm(dat$X20),X21_continuous= prepare_copula_qnorm(dat$X21),
                        X22_continuous = prepare_copula_qnorm(dat$X22),X23_continuous= prepare_copula_qnorm(dat$X23),
                        X24_continuous = prepare_copula_qnorm(dat$X24),X25_continuous= prepare_copula_qnorm(dat$X25)
                        )
  return (dat)
}
```

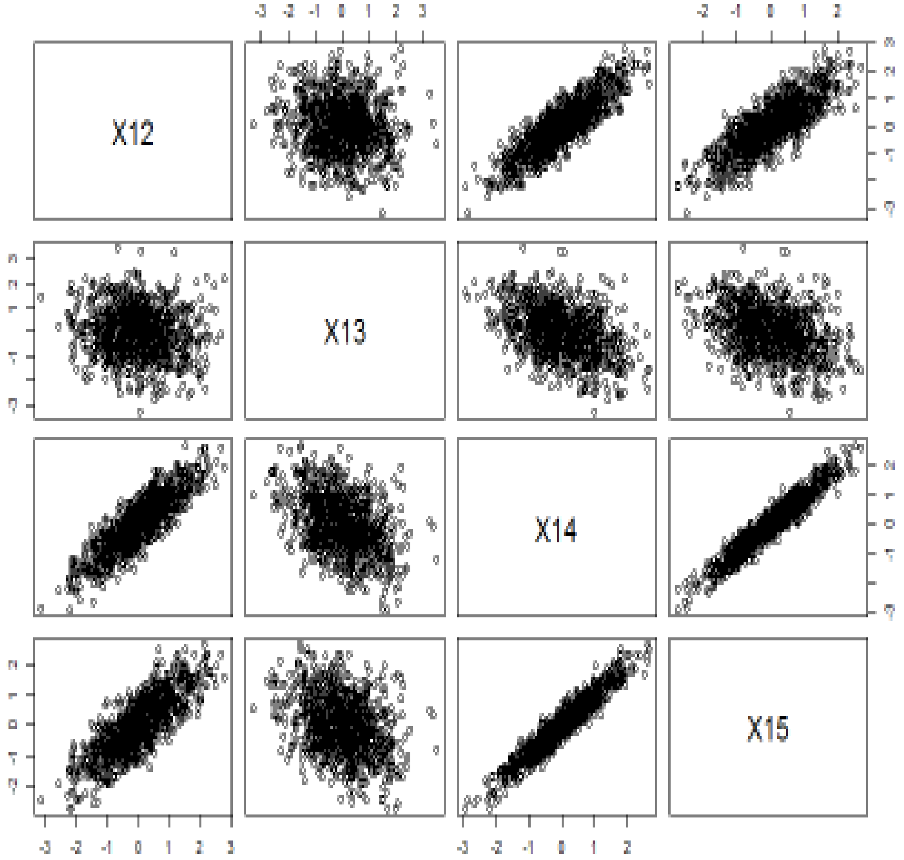
USDA

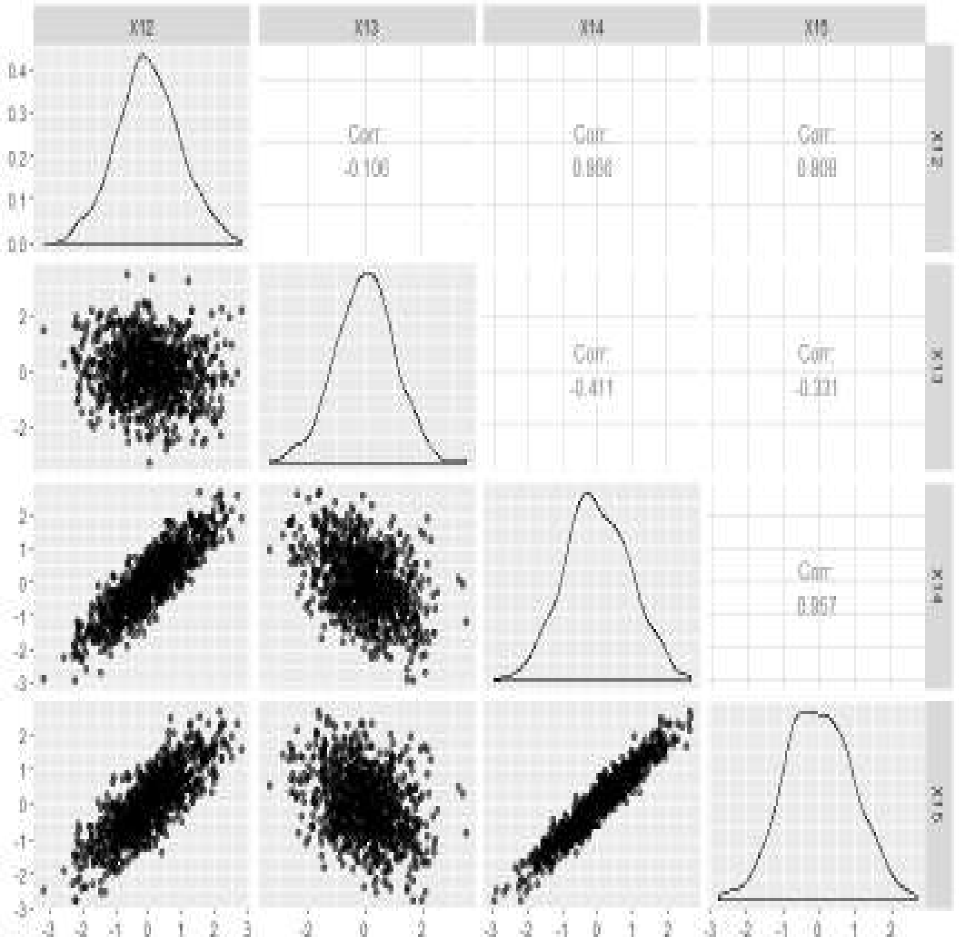
Step by step R function - Continue

```
create_newpop_sizetype<- function(m,n, pop_num, dat_tol){
  #This function is designed to create synthetic population within each New_X2 and New_X3
  dat_continuous <- dat_tol %>%
    filter (New_X2 ==m, New_X3==n) %>%
    dplyr::select(New_X1_continuous,  X4_continuous, X5_continuous,
                  X6_continuous, X7_continuous, X8_continuous, X9_continuous, X10_continuous,
                  X11_continuous, X12_continuous, X13_continuous, X14_continuous, X15_continuous,
                  X16_continuous, X17_continuous, X18_continuous, X19_continuous, X20_continuous,
                  X21_continuous, X22_continuous, X23_continuous, X24_continuous, X25_continuous)
  dat_cor <- cor(dat_continuous)
  #generating new population
  new_pop <- mvrnorm(n = pop_num, mu = rep(0,nrow(dat_cor)), dat_cor, tol = 1e-6, empirical = FALSE, EISPACK = FALSE)
  return (new_population)
}
```

USDA

Pairwise correlation comparison





Summary

- This study
  - Generating synthetic population by Gaussian copula
  - Adopting two R packages dplyr and ggplot2 simplifed the tasks for this study

- dplyr and ggplot2 are two useful R packages
  - More convenient to manage
  - Easy to read
  - Decrease the workload
  - My personal preference to use over base R

References

Nelsen, R. B. (2007). *An introduction to copulas*. Springer Science & Business Media.

Wickham, H., Francois, R., Henry, L., & Müller, K. (2015). dplyr: A grammar of data manipulation. *R package version 0.4, 3*.

Wickham, H. (2016). *ggplot2: elegant graphics for data analysis*. Springer.

Any Questions?

Thank you!