PROGRAMAÇÃO ESTATÍSTICA
Exercício 3

Leonardo de Salles Amaral
RA 770617

UFSCAR

São Carlos, 2022

Método congruencial multiplicativo:

$$x_i = ax_{i-1} \bmod m, \quad 0 < x_i < m.$$

Temos inicialmente dois cenários, um com $a = 17$ e outro com $a = 85$. Em ambos usamos $m = 2^{13} - 1$, e geramos 500 valores de x_i em cada cenário. Todo o código `R` [R Core Team, 2021] utlizado é apresentado.

```
mc <- function(x=617, a, m=2**13-1, length=500)
{
    xs    <- rep(NA, length)
    xs[1] <- x
    for (i in 2:length) xs[i] <- a * xs[i-1] %% m
    return(xs)
}
xa17 <- mc(a=17)
xa85 <- mc(a=85)
```

Abaixo temos um resumo dos valores gerados em cada cenário. Em ambos cenários começamos com o mesmo valor, $x_1 = 617$ (os três últimos algarismos do meu RA). Vemos que com um multiplicador a maior, os valores gerados são consequentemente maiores.

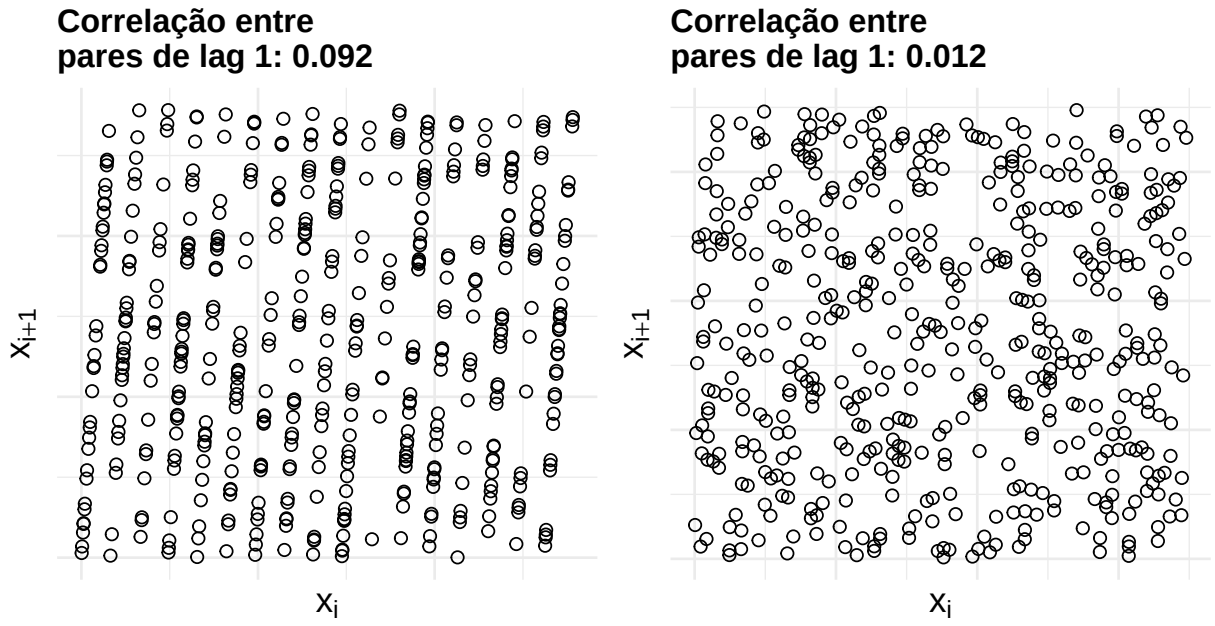
```
cbind('a'=c(17, 85), rbind(summary(xa17), summary(xa85)))
```

```
      a Min.   1st Qu. Median      Mean  3rd Qu.   Max.
[1,] 17   85  33281.75  67694  68954.76 102284.8 139111
[2,] 85  617 168725.00 335835 347587.14 532057.5 695725
```

A seguir, representamos em um gráfico os pares consecutivos de números x_{i+1} e x_i. Além disso, calculamos a correlação de *Pearson* entre tais pares.

```
library(tibble)
library(ggplot2)
library(patchwork)

mc.cor <- function(mc_seq, lag=1)
{
    size <- length(mc_seq)
    x    <- mc_seq[lag:size - lag]
    y    <- mc_seq[(lag + 1):size]
    tibble::tibble(x, y)|>
        ggplot(aes(x, y))+
        geom_point(size=2, shape=21)+
        labs(x=bquote(x[i]),
             y=bquote(x[i+.(lag)]),
             title=paste0('Correlação entre\npares de lag ', lag, ': ',
                          round(cor(x, y), 3)))+
        theme_minimal()+
        theme(axis.text.x =element_blank(),
              axis.title.x=element_text(size=14),
              axis.text.y =element_blank(),
              axis.title.y=element_text(size=14),
              plot.title  =element_text(face='bold'))
}
mc.cor(xa17) + mc.cor(xa85)
```

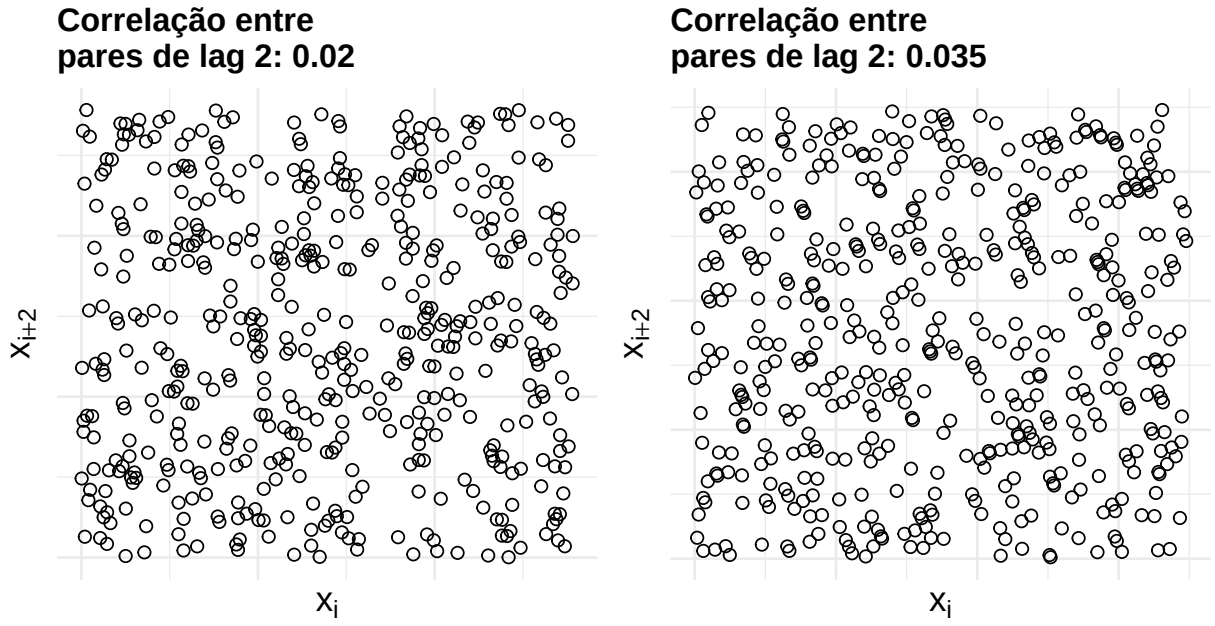


Na esquerda temos os pares gerados com o multiplicador $a = 17$, e na direita os pares

gerados com $a = 85$. Com as pequenos temos maiores correlações entre os pares, e consequentemos os pontos parecem ser menos aleatórios/com um padrão mais claro. Conforme aumentamos o valor do multiplicador a, mais aleatório se parece o gráfico dos pares consecutivos de números gerados pelo mcm. No gráfico da esquerda os pontos estão dispostos em 17 retas, o valor do multiplicador a.

Quando olhamos para os pares x_i e x_{i+2}, gráficos abaixo, temos um comportamento muito mais aleatório e com menor correlação no cenário com multiplicador $a = 17$ (gráfico da esquerda).

```
mc.cor(xa17, lag=2) + mc.cor(xa85, lag=2)
```



Agora, o método congruencial matricial:

$$\begin{bmatrix} x_{1i} \\ x_{2i} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} x_{1,i-1} \\ x_{2,i-1} \end{bmatrix} \mod m,$$

com $m = 2^{13} - 1$, $a_{11} = 17$, $a_{22} = 85$, e a_{12} e a_{21} variando separadamente entre 0 e 17.

Utilizando os seguintes valores para a_{12} e a_{21}, temos 20 pares de combinações.

```
(a12 <- c(0, 0:17, 17))
(a21 <- c(0, 17:0, 17))
```

```
 [1]  0  0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 17
 [1]  0 17 16 15 14 13 12 11 10  9  8  7  6  5  4  3  2  1  0 17
```

Para cada par de a_{12} e a_{21}, 500 vetores sã gerados.

```
mc.mat<- function(x1=617, x2=770, a11=17, a22=85, a12, a21,
                  m=2**13-1, length=500)
{
    xs      <- matrix(NA, nrow=length, ncol=2)
    xs[1, ] <- c(x1, x2)
    A <- matrix(c(a11, a21, a12, a22), nrow=2, ncol=2)
    for (i in 2:length) xs[i, ] <- A %*% xs[i-1, ] %% m
    return(xs)
}
xs_mat  <- vector('list', 20)
for (i in seq(20)) xs_mat[[i]] <- mc.mat(a12=a12[i], a21=a21[i])
```

Para cada par de a_{12} e a_{21}, suas variâncias e covariâncias amostrais são calculadas em termos de suas matrizes de variância e covariância.

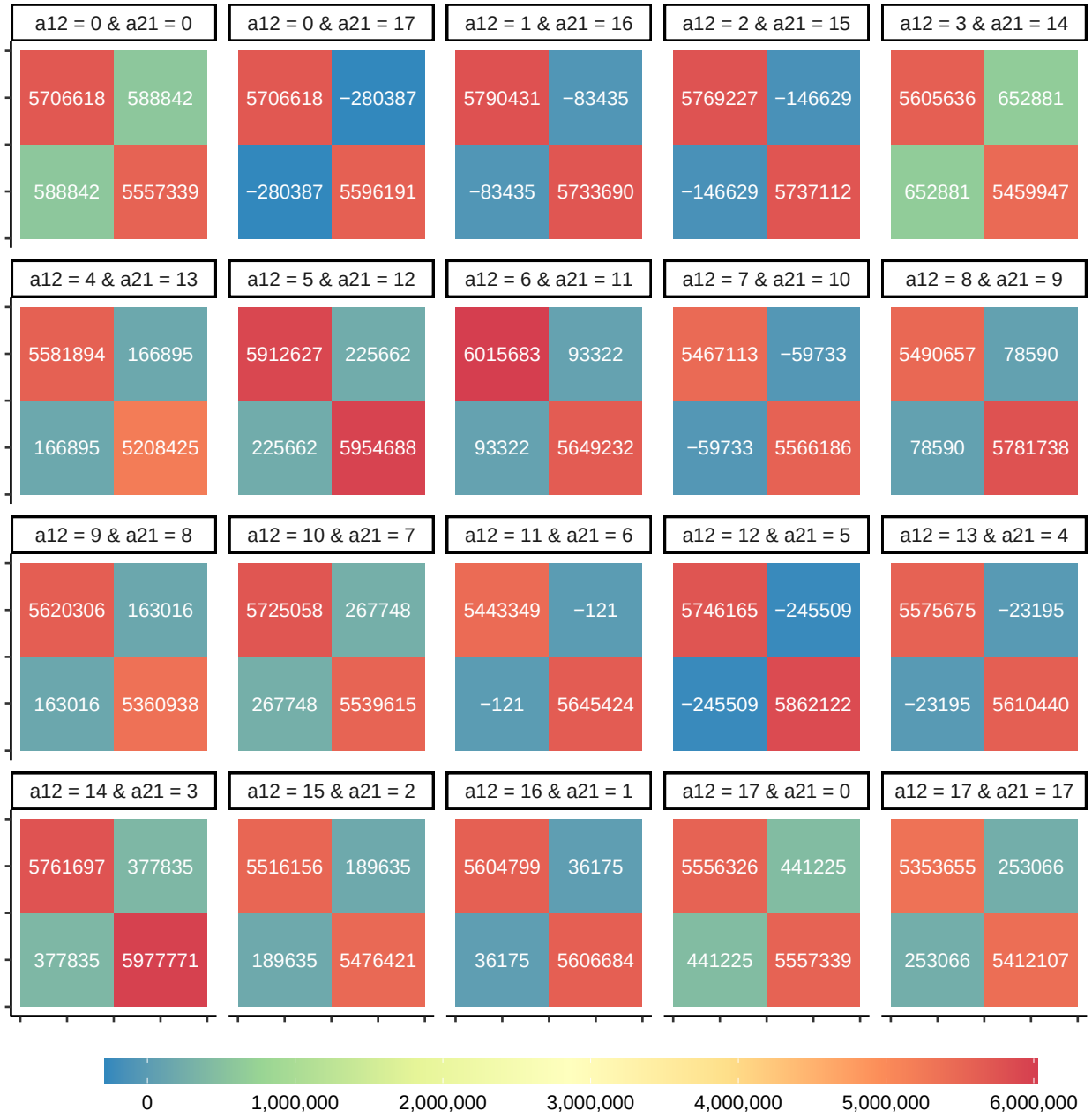
```
xs_vcov <- vector('list', 20)
for (i in seq(20)) xs_vcov[[i]] <- var(xs_mat[[i]])
```

Apresentamos as matrizes de variância e covariância de maneira gráfica.

```
value <- numeric(80)
k     <- 1
id    <- numeric(20)

for (i in seq(20))
{
    for (j in c(2, 1, 4, 3)) ## a12, a11, a22, a21 ---------------------
    {
        value[k] <- xs_vcov[[i]][j]
        k        <- k+1
    }
    id[i] <- paste('a12 =', a12[i], '& a21 =', a21[i])
}
library(scales)
library(dplyr)
library(haven)
dat <- tibble::tibble(id   =rep(id, each=4),
                      x    =rep(rep(1:2, each =2), times=20),
                      y    =rep(rep(1:2, times=2), times=20),
                      value=value)|>
    dplyr::mutate(id=haven::as_factor(id))
dat|>
    ggplot(aes(x, y, fill=value))+
    facet_wrap(~ id)+
    geom_tile()+
    scale_fill_distiller(palette='Spectral',
```

```
                    labels=scales::comma, n.breaks=6)+
geom_text(aes(label=round(value, 0)), color='white', size=3.3)+
labs(x=NULL, y=NULL, fill=NULL)+
theme_classic()+
theme(legend.position      ='bottom',
      legend.justification=c(0.65, 0),
      legend.key.width     =ggplot2::unit(3, 'cm'),
      legend.key.height    =ggplot2::unit(0.4, 'cm'),
      axis.text.x          =element_blank(),
      axis.text.y          =element_blank())
```



A seguir, respondemos algumas perguntas.

- As variâncias dos elementos em seus vetores são constantes quando variam-se a_{12} e a_{21}?

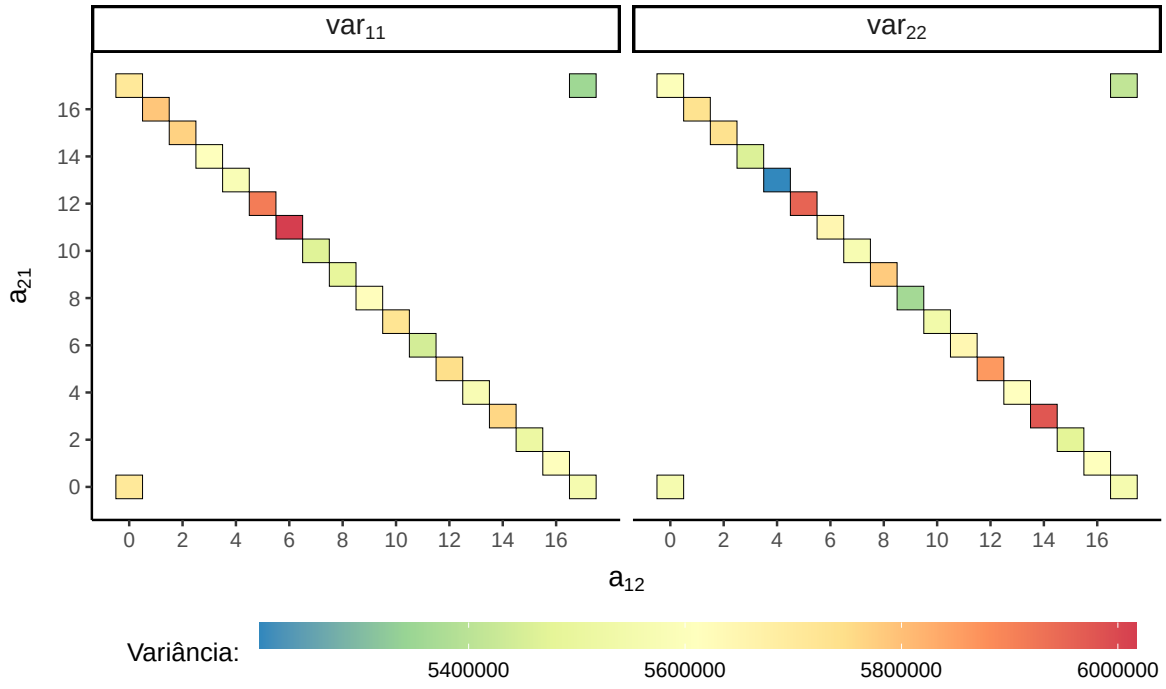
  De modo geral, sim. Obviamente, variando os valores de a_{12} e a_{21} temos variâncias diferentes. Contudo, os valores são muito similares/constantes. Com exceção de um cenário que resultou numa variância de 6015683, todas as demais variâncias ficam entre 5 e 6 milhões.

  Com o gráfico a seguir fica mais fácil de visualizarmos isso.

```
library(tidyr)

value_var <- dat|>
    dplyr::mutate(var=ifelse(x != y, value, NA))|>
    tidyr::drop_na()|>
    dplyr::pull(value)

tibble::tibble(a12, a21,
               '11'=value_var[seq(1, 40, by=2)],
               '22'=value_var[seq(2, 40, by=2)])|>
    tidyr::pivot_longer(`11`:`22`, names_to='var')|>
    ggplot(aes(x=a12, y=a21, fill=value))+
    geom_tile(color='black')+
    facet_wrap(~ var,
               labeller=label_bquote(var[.(var)]))+
    scale_fill_distiller(palette='Spectral')+
    scale_x_continuous(breaks=seq(0, 16, by=2))+
    scale_y_continuous(breaks=seq(0, 16, by=2))+
    labs(x=bquote(a[12]),
         y=bquote(a[21]), fill='Variância: ')+
    theme_classic()+
    theme(legend.position     ='bottom',
          legend.justification=c(0.65, 0),
          legend.key.width    =ggplot2::unit(2.65, 'cm'),
          legend.key.height   =ggplot2::unit(0.4, 'cm'),
          axis.title.x        =element_text(
              size=12,
              margin=ggplot2::unit(c(t=3, r=0, b=-2, l=0), 'mm')
          ),
          axis.title.y        =element_text(
              size=12,
              margin=ggplot2::unit(c(t=0, r=3, b=0, l=0), 'mm')
          ),
          strip.text.x        =element_text(size=12))
```



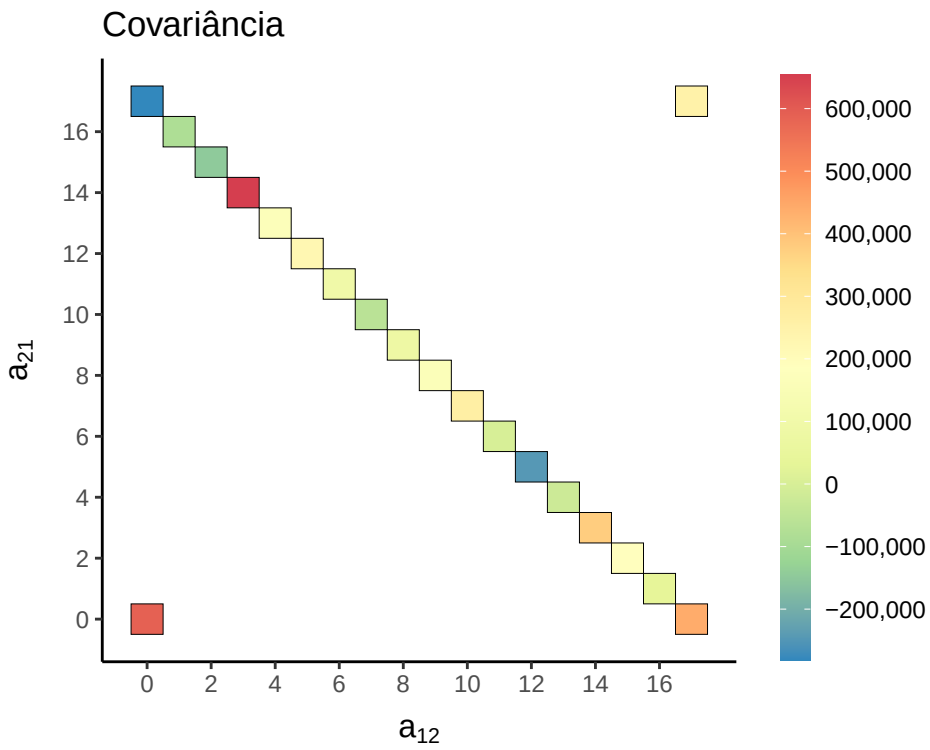
- E as covariâncias? Você consegue perceber alguma relação entre as covariâncias e os valores de a_{12} e a_{21}?

  O gráfico a seguir nos ajuda a visualizar.

```
value_cov <- dat|>
    dplyr::mutate(var=ifelse(x == y, value, NA))|>
    tidyr::drop_na()|>
    dplyr::pull(value)

tibble::tibble(a12, a21,
               cov=value_cov[seq(1, 40, by=2)])|>
    ggplot(aes(x=a12, y=a21, fill=cov))+
    geom_tile(color='black')+
    scale_fill_distiller(palette='Spectral',
                         n.breaks=8, labels=scales::comma)+
    scale_x_continuous(breaks=seq(0, 16, by=2))+
    scale_y_continuous(breaks=seq(0, 16, by=2))+
    labs(x=bquote(a[12]),
         y=bquote(a[21]), fill=NULL, title='Covariância')+
    theme_classic()+
    theme(legend.key.width    =ggplot2::unit(0.4, 'cm'),
          legend.key.height   =ggplot2::unit(1.55, 'cm'),
          axis.title.x        =element_text(
              size=12,
              margin=ggplot2::unit(c(t=3, r=0, b=0, l=0), 'mm')
          ),
          axis.title.y        =element_text(
```

```
    size=12,
    margin=ggplot2::unit(c(t=0, r=3, b=0, l=0), 'mm')
))
```



De modo geral, também não vemos nenhum padrão claro. Aparentemente, temos aleatoriedade.

- a_{12} e a_{21} com valores iguais produzem resultados diferentes em relação a quando eles possuem valores diferentes?

  Não, como podemos ver pelas figuras fornecidas. Nos dois casos em que os valores de a_{12} e a_{21} são iguais, não obtivemos nem variâncias e nem covariâncias discrepantes.

- Uma matriz triangular inferior (ou seja, uma com $a_{21} = 0$) é capaz de produzir os mesmos resultados que matrizes com valores variantes de a_{21}?

  Sim, é capaz. Como podemos ver pelas figuras fornecidas acima.

Os seguintes pacotes `R` [R Core Team, 2021] foram utilizados:

{`tibble`} [Müller and Wickham, 2021], {`ggplot2`} [Wickham, 2016], {`patchwork`} [Pedersen, 2020], {`scales`} [Wickham and Seidel, 2020], {`dplyr`} [Wickham et al., 2022], {`haven`} [Wickham and Miller, 2021], e {`tidyr`} [Wickham, 2021].

Referências

[Müller and Wickham, 2021] Müller, K. and Wickham, H. (2021). *tibble: Simple Data Frames*. R package version 3.1.2, https://CRAN.R-project.org/package=tibble.

[Pedersen, 2020] Pedersen, T. L. (2020). *patchwork: The Composer of Plots*. R package version 1.1.1, https://CRAN.R-project.org/package=patchwork.

[R Core Team, 2021] R Core Team (2021). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing. Vienna, Austria. https://www.R-project.org/.

[Wickham, 2016] Wickham, H. (2016). *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York. https://ggplot2.tidyverse.org.

[Wickham, 2021] Wickham, H. (2021). *tidyr: Tidy Messy Data*. R package version 1.1.3, https://CRAN.R-project.org/package=tidyr.

[Wickham et al., 2022] Wickham, H., François, R., Henry, L., and Müller, K. (2022). *dplyr: A Grammar of Data Manipulation*. R package version 1.0.8, https://CRAN.R-project.org/package=dplyr.

[Wickham and Miller, 2021] Wickham, H. and Miller, E. (2021). *haven: Import and Export 'SPSS', 'Stata' and 'SAS' Files*. R package version 2.4.3, https://CRAN.R-project.org/package=haven.

[Wickham and Seidel, 2020] Wickham, H. and Seidel, D. (2020). *scales: Scale Functions for Visualization*. R package version 1.1.1, https://CRAN.R-project.org/package=scales.