Proposta A

Contextualização

Minha proposta consiste em ajudar o usuário a selecionar e analisar o melhor modelo para os seus dados. A função irá: (1) montar os modelos lineares ou lineares mistos, (2) selecionar o melhor modelo (se houver mais de um), (3) analisar a distribuição dos resíduos do modelo selecionado, (4) fazer um summary ou anova (dependendo do tipo de modelo) do modelo selecionado.

Entrada:

```
mayara ← function(data, x, y, k = F , z = F, ylevels = F, klevels = F , nulo = T, AICc = F)
```

• data: `data.frame` com os dados

• x: coluna da variável resposta

• y: coluna da variável preditora 1

• k: coluna da variável preditora 2 (se não for preenchido o default será F)

• z: coluna da variável aleatória (se não for preenchido o default será F)

• `ylevels`: níveis de y, caso seja uma variável categórica (se não for preenchido o default será F, ou seja, y será numérico e não um fator)

• `klevels`: níveis de k, caso seja uma variável categórica (se não for preenchido o default será F, ou seja, k será numérico e não um fator)

• `nulo`: caso o usuário deseje usar o modelo 1 como modelo mais simples, ele deverá colocar `nulo = F` (se não for preenchido o default será T)

• `AICc`: se o n amostral for pequeno, recomendamos que seja usado o AICc (AIC corrigido para pequenas amostras), colocando `AICc = T` (se não for preenchido o default será F).

Saída:

Lista com: (1) tabela de AIC ou AICc, (2) modelo selecionado e (3) um objeto chamado resultado com uma anova e summary do modelo. Além disso, serão criados gráficos para análise visual da distribuição dos resíduos do modelo ao longo da análise.

Premissas que estarão no help da função:

• Para usar essa função é necessário que os dados estejam organizados em um `data.frame`

• É necessário instalar os pacotes `RVAideMemoire` (para o teste de Shapiro e plots), `bbmle` (para AIC), `lme4` (para lmer), `lattice` (para qqmath) e `rcompanion` (para histoplotresid) antes de usar a função.

• Essa função só lida com modelos lineares ou lineares mistos

• Essa função só lida com 1 variável resposta (y) e até 2 variáveis preditoras de efeito fixo (x e k), além de ser possível inserir 1 variável aleatória (z).

• Os argumentos `ylevels` e `klevels` terão que ser preenchidos da seguinte forma: Ex.: `ylevels = c("1", "2", "3")` neste exemplo temos 3 níveis.

Pseudo-código:

1. Com a função `if` verifica se a variável y possui níveis → se tiver, os níveis serão informados colocando `factor(data$y, levels = c(ylevels)`

2. Com a função `if`, verifica se a variável k possui níveis → se tiver, os níveis serão informados colocando `factor(data$y, levels = c(klevels)`

3. Com a função `if` verifica se `nulo==T` (ou seja, o usuário escolheu que o modelo nulo criado seja usado). Se for, continua no item a seguir (as chaves permanecem abertas até testar todas as opções dentro de `nulo ==T`). Se não for, vai para o item 4.

3.1 Com a função `if` testa se z==F&k==F (ou seja, só 1 variável preditora, x, e nenhuma aleatória) → se for, será criado o modelo linear `modelo_escolhido ← lm(y ~ x, data)`. E então, será feita uma anova do modelo. Em seguida, serão feitos testes de normalidade:

```
plotresid(modelo_escolhido, shapiro = T)

r = residuals(modelo_escolhido)

plotNormalHistogram de r

qqmath(modelo_escolhido)
```

Ainda dentro desse `if`, será feito o teste `if(teste_normalidade$p.value < = 0.05)` → se der, a seguinte mensagem aparecerá:

```
{message("parece que os resíduos do seu modelo não possuem uma distribuição normal. Sugerimos que verifique visualmente e, se necessário, mude o tipo de modelagem para uma que use uma distribuição que se adeque melhor aos seus dados")}.
```

E será fechado o `if`.

3.2 Com a função `if` testa se z==T&k==F (ou seja, existe variável aleatória, mas não uma segunda variável preditora de efeito fixo) → se for, serão criados os modelos

```
m_nulo ← lmer(y ~ 1 + (1|z), data, REML = F) m_1 ← lmer(y ~ x + (1|z), data, REML = F)
```

Ainda dentro desse `if`, será feito o teste `if(AICc==F)` → se for, será criada uma tabela de AIC `tabela.AIC ← AICtab(m_nulo,m_1, weights=T, delta=TRUE, base=TRUE)`

Mas, caso o usuário tenha colocado AICc=T no argumento, o teste `if(AICc==T)` gerará a tabela `tabela.AIC ← AICctab(m_nulo,m_1, weights=T, delta=TRUE, base=TRUE)`

Após gerar a tabela, será criado o objeto `modelo_escolhido`, com o modelo de dAIC ou dAICc igual ou menor que 2, para selecionar o modelo. Em seguida, são feitos os testes de normalidade:

```
plotresid(modelo_escolhido, shapiro = T)
```

```
r = residuals(modelo_escolhido)

plotNormalHistogram de r

qqmath(modelo_escolhido)
```

```
Ainda dentro desse ''if'', será feito o teste ''if(teste_normalidade$p.value
< = 0.05)'' -> se der, a seguinte mensagem aparecerá:

''{message("parece que os resíduos do seu modelo não possuem uma
distribuição normal. Sugerimos que verifique visualmente e, se necessário,
mude o tipo de modelagem para uma que use uma distribuição que se adeque
melhor aos seus dados. Análise continua")}''
```

Então, será criado o objeto `resultado ← summary(modelo_escolhido)` com o sumário do modelo selecionado e será encerrado o `if`.

3.3 Com a função `if` será feito o teste `(z==T&k==T)` (ou seja, há uma variável aleatória, z, e segunda variável de efeito fixo, k) → se for, serão criados os modelos

```
''m_nulo <- lmer(y ~ 1 + (1|z), data, REML = F)
m_1 <- lmer(y ~ x + (1|z),  data, REML = F)
m_2 <- lmer(y ~ k + (1|z), data=, REML = F)
m_3 <- lmer(y ~ x + k + (1|z), data, REML = F)
m_4 <- lmer(y ~ x:k + (1|z), data, REML = F)
m_5 <- lmer(y ~ x:k + x + (1|z), data, REML = F)
m_6 <- lmer(y ~ x:k + k + (1|z), data, REML = F)
m_7 <- lmer(y ~ x:k + x + k + (1|z), data, REML = F)''
```

Ainda dentro desse `if`, será feito o teste `if(AICc==F)` → se for, será criada uma tabela de AIC `tabela.AIC ← AICtab(m_nulo,m_1, m_2, m_3, m_4,m_5,m_6,m_7, weights=T, delta=TRUE, base=TRUE)`

Mas, caso o usuário tenha colocado AICc=T no argumento, o teste `if(AICc==T)` gerará a tabela `tabela.AIC ← AICctab(m_nulo,m_1, m_2, m_3, m_4,m_5,m_6,m_7, weights=T, delta=TRUE, base=TRUE)`

Após gerar a tabela, será criado o objeto `modelo_escolhido`, com o modelo de dAIC ou dAICc igual ou menor que 2, para selecionar o modelo.

Em seguida, são feitos os testes de normalidade

```
plotresid(modelo_escolhido, shapiro = T)

r = residuals(modelo_escolhido)

plotNormalHistogram de r

qqmath(modelo_escolhido)
```

```
Ainda dentro desse ''if'', será feito o teste ''if(teste_normalidade$p.value
< = 0.05)'' -> se der, a seguinte mensagem aparecerá:
```

```
{message("parece que os resíduos do seu modelo não possuem uma distribuição normal. Sugerimos que verifique visualmente e, se necessário, mude o tipo de modelagem para uma que use uma distribuição que se adeque melhor aos seus dados. Análise continua")}''
```

Então, será criado o objeto `resultado ← summary(modelo_escolhido)` com o sumário do modelo selecionado e será encerrado o `if`.

3.4 Com a função `if` será feito o teste `(z==F&K==T)` (ou seja, não há variável aleatória, mas uma segunda variável preditora de efeito fixo) → se tiver, serão criados os modelos

```
''m_nulo <- lm(y ~ 1,  data, REML = T)
m_1 <- lm(y ~ x, data, REML = T)
m_2 <- lm(y ~ k, data, REML = T)
m_3 <- lm(y ~ x + k, data, REML = T)
m_4 <- lm(y ~ x:k, data, REML = T)
m_5 <- lm(y ~ x:k + x, data, REML = T)
m_6 <- lm(y ~ x:k + k, data, REML = T)
m_7 <- lm(y ~ x:k + x + k, data, REML = T)''
```

Ainda dentro desse `if`, será feito o teste `if(AICc==F)` → se for, será criada uma tabela de AIC `tabela.AIC ← AICtab(m_nulo,m_1, m_2, m_3, m_4,m_5,m_6,m_7, weights=T, delta=TRUE, base=TRUE)`

Mas, caso o usuário tenha colocado AICc=T no argumento, o teste `if(AICc==T)` gerará a tabela `tabela.AIC ← AICctab(m_nulo,m_1, m_2, m_3, m_4,m_5,m_6,m_7, weights=T, delta=TRUE, base=TRUE)`

Após gerar, a tabela será criado o objeto `modelo_escolhido`, com o modelo de dAIC ou dAICc igual ou menor que 2, para selecionar o modelo.

Em seguida, são feitos os testes de normalidade

```
plotresid(modelo_escolhido, shapiro = T)
```

```
r = residuals(modelo_escolhido)
```

`plotNormalHistogram` de `r`

```
qqmath(modelo_escolhido)
```

Ainda dentro desse `if`, será feito o teste `if(teste_normalidade$p.value < = 0.05)` → se der, a seguinte mensagem aparecerá:

```
{message(“parece que os resíduos do seu modelo não possuem uma distribuição normal. Sugerimos que verifique visualmente e, se necessário, mude o tipo de modelagem para uma que use uma distribuição que se adeque melhor aos seus dados. Análise continua”)}
```

Então, será criado o objeto `resultado ← summary(modelo_escolhido)` com o sumário do modelo selecionado e será encerrado o `if`.

Então a chave, e portanto, o `if`, criada inicialmente (item 3.) é fechada.

4. Com a função `if` verifica se `nulo==F` (ou seja, o usuário escolheu que o modelo mais simples não seja o modelo nulo, mas o modelo 1). As chaves permanecem abertas até testar todas as opções dentro de `nulo==F`.

4.1 Com a função `if` será feito o teste `(z==T&k==F)` (ou seja, há uma variável aleatória, mas não uma segundo variável preditora de efeito fixo → se for, será criado o modelo `m_1 ← lmer(y ~ x + (1|z), data, REML = F)`. Em seguida, será criado o objeto `resultado ← summary(m_1)`. Então, as chaves são fechadas (a função `if` é encerrada).

4.2 Com a função `if` será feito o teste `(z==T&k==T)` (ou seja, há variável aleatória e uma segunda preditora de efeito fixo) → se for, serão criados os modelos

```
''m_1 <- lmer(y ~ x + (1|z),  data, REML = F)
  m_2 <- lmer(y ~ k + (1|z), data=, REML = F)
  m_3 <- lmer(y ~ x + k + (1|z), data, REML = F)
  m_4 <- lmer(y ~ x:k + (1|z), data, REML = F)
  m_5 <- lmer(y ~ x:k + x + (1|z), data, REML = F)
  m_6 <- lmer(y ~ x:k + k + (1|z), data, REML = F)
  m_7 <- lmer(y ~ x:k + x + k + (1|z), data, REML = F)''
```

Ainda dentro desse `if`, será feito o teste `if(AICc==F)` → se for, será criada uma tabela de AIC `tabela.AIC ← AICtab(m_1, m_2, m_3, m_4,m_5,m_6,m_7, weights=T, delta=TRUE, base=TRUE)`

Mas, caso o usuário tenha colocado AICc=T no argumento, o teste `if(AICc==T)` gerará a tabela `tabela.AIC ← AICctab(m_1, m_2, m_3, m_4,m_5,m_6,m_7, weights=T, delta=TRUE, base=TRUE)`

Após gerar, a tabela será criado o objeto `modelo_escolhido`, com o modelo de dAIC ou dAICc igual ou menor que 2, para selecionar o modelo.

Em seguida, são feitos os testes de normalidade

```
''plotresid(modelo_escolhido, shapiro = T)''
```

```
r = residuals(modelo_escolhido)
```

`plotNormalHistogram` de `r`

```
qqmath(modelo_escolhido)
```

Ainda dentro desse `if`, será feito o teste `if(teste_normalidade$ p.value < = 0.05)` → se der, a seguinte mensagem aparecerá: `{message("parece que os resíduos do seu modelo não possuem uma distribuição normal. Sugerimos que verifique visualmente e, se necessário, mude o tipo de modelagem para uma que use uma distribuição que se adeque melhor aos seus dados. Análise continua")}`

Então, será criado o objeto `resultado ← summary(modelo_escolhido)` com o sumário do modelo selecionado e será encerrado o `if`.

4.3 Com a função `if` será feito o teste `(z==F&K==T)` (ou seja, não há uma variável aleatória, mas há uma segunda variável preditora de efeito fixo) → se for, os seguintes modelos serão criados

```
''m_1 <- lm(y ~ x, data, REML = T)
  m_2 <- lm(y ~ k, data, REML = T)
  m_3 <- lm(y ~ x + k, data, REML = T)
  m_4 <- lm(y ~ x:k, data, REML = T)
  m_5 <- lm(y ~ x:k + x, data, REML = T)
  m_6 <- lm(y ~ x:k + k, data, REML = T)
  m_7 <- lm(y ~ x:k + x + k, data, REML = T)''
```

Ainda dentro desse `if`, será feito o teste `if(AICc==F)` → se for, será criada uma tabela de AIC `tabela.AIC ← AICtab(m_1, m_2, m_3, m_4,m_5,m_6,m_7, weights=T, delta=TRUE, base=TRUE)`

Mas, caso o usuário tenha colocado AICc=T no argumento, o teste `if(AICc==T)` gerará a tabela `tabela.AIC ← AICctab(m_1, m_2, m_3, m_4,m_5,m_6,m_7, weights=T, delta=TRUE, base=TRUE)`

Após gerar, a tabela será criado o objeto `modelo_escolhido`, com o modelo de dAIC ou dAICc igual ou menor que 2, para selecionar o modelo.

Em seguida, são feitos os testes de normalidade

`plotresid(modelo_escolhido, shapiro = T)`

`r = residuals(modelo_escolhido)`

`plotNormalHistogram` de `r`

`qqmath(modelo_escolhido)`

Ainda dentro desse `if`, será feito o teste `if(teste_normalidade$p.value < = 0.05)` → se der, a seguinte mensagem aparecerá: `{message("parece que os resíduos do seu modelo não possuem uma distribuição normal. Sugerimos que verifique visualmente e, se necessário, mude o tipo de modelagem para uma que use uma distribuição que se adeque melhor aos seus dados. Análise continua")}`

Então, será criado o objeto `resultado ← summary(modelo_escolhido)` com o sumário do modelo selecionado e será encerrado o `if`.

5. Por fim, será encerrada função mayara com a saída de uma lista com a tabela de AIC ou AICc, o modelo escolhido e o objeto resultado (que pode ser uma anova ou summary do modelo escolhido)

```
''return(list(tabela.AIC,modelo_escolhido,resultado))''
```

Julia Barreto

Oi, Mayara! Sua proposta A está bem estruturada e organizada, você me pareceu empenhada e interessada em realizar então podemos prosseguir com ela. Eu acredito que ainda é possível ser mais inovadora nessa tarefa, vamos tentar pensar em passos que podem tornar a função mais completa e

interessante?

Andei pensando sobre as fórmulas dos modelos que você se propôs criar, as preditoras seriam montadas apenas como termos aditivos (“+”) ou também interativos (“:”)? Talvez seja interessante criar esse argumento.

Além disso, normalidade não é o único aspecto dos resíduos que deve se atentar no diagnóstico, não seria interessante também analisar heteroscedasticidade, alavancagem e/ou sobredispersão?

Com as informações que você passou, não me parece que é qualquer planilha de dados de entrada que vá funcionar e isso já traz dificuldades na importação. Por exemplo, uma matrix de dados que contivesse espécies nas colunas ocorrências por sítios de coleta nas linhas já seria um empecilho na forma atual que planejou. Vale pensar sobre como resolveria isso ou considerar ser mais específica sobre o formato dos dados de entrada.

Me escreve se quiser conversar sobre suas ideias ou qualquer dúvida, estou à disposição Julia Barreto

Mayara Jordano

Olá Julia! Muito obrigada pelas suas sugestões! Então, sobre sua primeira pergunta “Andei pensando sobre as fórmulas dos modelos que você se propôs criar, as preditoras seriam montadas apenas como termos aditivos (”+“) ou também interativos (”:“)?” Eu montei as preditoras tanto como termos aditivos quanto interativos. No meu quadro de modelos com mais de uma variável preditora, existem ambas possibilidades, não se é a isso que você se refere.

Sobre essa pergunta “Além disso, normalidade não é o único aspecto dos resíduos que deve se atentar no diagnóstico, não seria interessante também analisar heteroscedasticidade, alavancagem e/ou sobredispersão?”

Realmente quando eu expliquei o código falei que era só para verificar a normalidade, mas as funções que coloquei também verificam a heteroscedasticidade dos resíduos, e boa ideia, vou adicionar o gráfico para verificar a alavancagem. Sobre essa pergunta “Com as informações que você passou, não me parece que é qualquer planilha de dados de entrada que vá funcionar e isso já traz dificuldades na importação. Por exemplo, uma matrix de dados que contivesse espécies nas colunas ocorrências por sítios de coleta nas linhas já seria um empecilho na forma atual que planejou. Vale pensar sobre como resolveria isso ou considerar ser mais específica sobre o formato dos dados de entrada.” Bom, eu coloquei que tinha que ser um data.frame e o que tinha que ter em cada coluna quando expliquei os argumentos da função. Você acha que precisa especificar mais alguma coisa? Talvez eu não tenha deixado claro como seria a planilha... obrigada vou verificar.

Julia Barreto

Oi Mayara! Acho que está claro sim. Se pensar em mais maneiras de se desafiar ou complementar ainda mais a função, estamos aí :)

Proposta B

Contextualização

Número de Reynolds (*Re*) é uma razão entre forças inerciais e viscosas, sendo, portanto, um número adimensional. É usado na mecânica de fluidos para o cálculo do regime de escoamento de determinado fluido sobre uma superfície. É utilizado, por exemplo, em projetos de tubulações industriais, de asas de aviões e para entender a interação animal-fluido de animais aquáticos.

É calculado da seguinte maneira:

onde V é a velocidade do escoamento, L é uma dimensão típica, ρ é a densidade da água, μ é a viscosidade dinâmica e ν é a viscosidade cinemática $\nu=\mu/\rho$.

Essa proposta consiste em calcular o número de Reynolds da superfície de algum objeto (*e.g.*, cano, asa de avião ou estrutura de um animal) a partir das posições de objetos, como partículas, que circulam ao redor deste objeto que ser calcular o *Re*.

Para isso, o usuário terá que fazer uma filmagem do fluxo ao redor do objeto desejado e transformar esse vídeo em imagens, no qual irá acompanhar as partículas do fluxo e coletar as posições dessas partículas ao longo do tempo (s).

Referência

Annual Review of Fluid Mechanics. Vol. 22:1-12 (Volume publication date January 1990)
https://doi.org/10.1146/annurev.fl.22.010190.000245

Entrada:

`Re ← function(data , f, v, l, n)`

Data deverá ser um `data.frame` com as seguintes colunas: `p, x, y`

`p` – partícula

`x` – posição x da partícula

`y` – posição y da partícula

`f` – frequência da filmagem em segundos

`v` - viscosidade cinemática do meio que se quer calcular o Re

`l` – espessura do objeto para o qual se deseja calcular o Re em metros

`n` – número de posições (comprimento de x e y)

Verificando parâmetros

`x` e `y` possuem o mesmo tamanho? Se não, irá aparecer `x e y possuem tamanhos diferentes` utilizando a função `stop`.

Saída `data.frame` com a distância percorrida, velocidade e o *Re* de cada posição de cada partícula.

Pseudo-código:

(1) Cálculo da distância percorrida pela partícula (S)

```
     Usando duas funções ''for'', será calculada a distância percorrida de
cada partícula. Será criada uma matriz (''S'') de ''NA'', com 1 coluna
(''S'') e ''n'' (será indicado pelo usuário no argumento) linhas.
```

No primeiro `for` terá `for(i in 1:n-1)`, e no segundo `for(j in (i-1):n)`

Dentro do segundo for terá:

```
sqrt((data$x[i]-data$x[j])^2 + (data$y[i]- data$y[j])^2)
           O resultado será guardado no objeto ''S''
```

(2) Cálculo da velocidade de fluxo (V) `V ← S/(1/f)`

Onde V foi calculado no item anterior e `f` foi informado pelo usuário como argumento.

(3) Cálculo do número de Reynolds `Reynolds ← l*velocidade/v`

Onde `l` e v foram informados pelo usuário como argumento

(4) Com a função `return` será retornado um `data.frame` com S,V e Re de cada posição de cada partícula.

Julia Barreto

Oi de novo! Achei que essa ideia seria também seria um pouco simples mas me parece um desafio interessante por extrair informação analisando imagens. Sei que já selecionamos a A mas vale dizer que essa teria potencial se fosse aprimorada para ficar mais completa, a comerçar por mais detalhe na proposta e pseudo-código. Por exemplo, não ficou claro pra mim o que exatamente propõe como entrada já que os argumentos não apontam a uma pasta onde contem os arquivos extraídos do vídeo. Fiquei curiosa em saber o que teria em mente. Outro aspecto que seria um potencial desafio seria imagem de qualidade ruim. Como não te conheci durante as semanas de aulas, fica difícil opinar sobre habilidade de executar mas já temos a primeira proposta que me parece melhor pensada! Julia Barreto

Mayara Jordano

Oi! Que bom que também gostou dessa! Então, não sei se essa é a sua dúvida, mas eu coloquei como argumentos da função um data.frame já com as posições das partículas (especifiquei o que seria cada coluna) que o usuário teria que fornecer. Eu não iria tirar os dados das imagens, então não seria um problema para a função se a qualidade da imagem não fosse boa. Se não respondi sua pergunta é só me falar haha Mas como você preferiu a A, vou focar nela.

Julia Barreto

Hum.. acho que eu havia entendido errado mesmo, então. Achei que entraria com uma sequência de imagens e a função detectaria a posição da partícula. Mas beleza, vamos que vamos com a A. Abs,Ju

FUNÇÃO

```
No final da página se encontram os arquivos .r da função e help.
```

Função que ajuda o usuário a selecionar e analisar o melhor modelo para os seus dados. os argumentos são:

- `data`: data.frame com os dados
- x: coluna da variável resposta
- y: coluna da variável preditora 1
- k: coluna da variável preditora 2
- z: coluna da variável aleatória
- `xlevels`: níveis de x, caso seja uma variável categórica
- `klevels`: níveis de k, caso seja uma variável categórica
- `nulo`: caso o usuário deseje usar o modelo 1 como modelo mais simples, deverá colocar nulo = F
- `AICc`: se o n amostral for pequeno, recomendamos que seja usado o AICc (AIC corrigido para pequenas amostras), colocando AICc = T.

Os passos serão:

- As variáveis x e k (se presente) possuem níveis?
- O modelo nulo será incluído na análise? (o usuário terá que avaliar se faz sentido, na sua análise, incluir ou não o modelo nulo como sendo o mais simples)
- As variáveis k e z estão presentes?
- Gerar modelo por AIC ou AICc
- Selecionar modelo com delta AIC menor ou igual a 2
- Realizar teste e gerar gráficos para análise visual da distribuição e homocedasticidade dos resíduos do modelo e, se necessário, gerar mensagem de alerta para possível necessidade de mudança do tipo de modelagem
- Realizar um summary ou anova do modelo selecionado

Para rodar esta função é necessário instalar os seguintes pacotes

- library(RVAideMemoire)# shapiro do modelo e plots
- library(bbmle)#AICc
- library(lme4)#lmer
- library(rcompanion) #histoplotresid
- library(dplyr) #transformar nome de linhas em colunas

Aqui começa a função

```
 mayara <- function(data, x, y, k = NULL, z = NULL, xlevels = NULL, klevels
= NULL , nulo = T, AICc=F)
suppressWarnings (#suprimi algumas mensagens de aviso do r)
  {#Verifica se os dados estao organizados em um data.frame. Se nao estiver,
a funcao para e aparece a seguinte mensagem
    if(class(data)!="data.frame")
    {
      stop("Data não é um data.frame")
    }
    ####sao feitos testes logicos para as variaveis x e k ####
    if(!is.null(xlevels)) #verificando se a variavel x eh categorica
    {data$x <- factor(data$x, levels = xlevels)} #informando os niveis de x
    if(!is.null(klevels)) #verificando se a variavel k eh categorica
    {data$k <- factor(data$k, levels = klevels)} #informando os niveis de k
    ##Teste logico para saber se o modelo nulo sera incluido na analise
    if(nulo==T) #se o modelo nulo for incluido na analise
    { ##Serie de Testes logicos para saber se tem variavel z e K
      if(is.null(z) & is.null(k))
      { #Se nao tiver variavel aleatoria nem segunda variavel resposta
        modelo_escolhido <- lm(y ~ x, data)   # Fazer modelo linear de y em
funcao de x (aqui nao foi feito um modelo nulo, porque a anova ja vai
comparar com o modelo nulo)
        ##Testes de distribuicao e homocedasticidade dos residuos do modelo
        teste_normalidade <- plotresid(modelo_escolhido, shapiro = T) # gera
graficos para verificar visualmente a ditribuicao dos residuos e a
homogeneidade de variancia dos residuos (homocedasticidade), além fazer o
teste de shapiro para verificar se os residuos do modelo seguem uma
distribuicao normal.
        x11() #para abrir uma janela de gráfico
        # gera um histograma para verificar se os residuos do modelo seguem
uma distribuicao normal
        r = residuals(modelo_escolhido)
        plotNormalHistogram(r)
        x11() #para abrir uma janela de gráfico
        par(mfrow=c(2,2)) #para aparecer os 4 graficos de uma vez
        plot(modelo_escolhido) # plota 4 graficos para averiguar a
distribuicao, homocedasticidade e alavancagem dos residuos do modelo
        #Teste para saber se o teste de shapiro deu significativo
        if(teste_normalidade$p.value<=0.05) #se der, aparece a mensagem
abaixo
        {message("Parece que os resíduos do seu modelo não possuem uma
```

```
distribuição normal. Sugerimos que verifique visualmente e, se necessário,
mude o tipo de modelagem para uma que use uma distribuição que se adeque
melhor aos seus dados.")}
        #Apesar da mensagem, a funcao continua. O usuario tera que avaliar,
nao soh com base no teste de shapiro, mas tambem pelas figuras geradas, se
deve ou nao usar outro tipo de modelo, talvez um modelo linear nao seja o
mais adequado
        tabela.AIC = "Não é aplicável"
        modelo_topo = "Não é aplicável"
        resultado <- anova(modelo_escolhido) # obejeto criado com a anova do
modelo
      }
      ## Continua a serie de Testes logicos para saber se tem variavel z e K
caso a anterior nao tenha dado positivo
      if(!is.null(z) & is.null(k)) #Se tiver variavel aleatoria, mas nao uma
segunda variavel resposta
        #fazer modelos lineres mistos (coloquei REML=F porque eh preciso
ajustar o modelo por máxima verossimilhança (ML), porque é o indicado para
comparar modelos com diferentes efeitos fixos mas com mesmo efeito
aleatório)
      {
        m_nulo <- lmer(y ~ 1 + (1|z), data, REML = F)#modelo linear misto
nulo (com variavel aletoria)
        m_1 <- lmer(y ~ x + (1|z),  data, REML = F)#modelo linear misto com
y em funcao de x (com variavel aletoria)
        if(AICc==F)
        {
          tabela.AIC <- AICtab(m_nulo,m_1, weights=T, delta=TRUE,
base=TRUE)#teste para saber se sera feita selecao de modelos por AIC. Entao
eh gerada a tabela de AIC
          class(tabela.AIC) <- "data.frame" #transformei em data.frame para
poder transformar o nome das linhas em coluna
          tabela.AIC <- tibble::rownames_to_column(tabela.AIC, "modelo")
#transformei os nomes das linhas em coluna
          modelo_topo <- tabela.AIC$modelo[tabela.AIC$dAIC<=2] #objeto
criado com o modelo selecionado, i.e., com delta AIC menor ou igual a 2.
Pode ser mais de um modelo.
          if((length(modelo_topo))>1) #criar mensagem de aviso caso tenha
mais de um modelo com dAIC<=2.
          {
            message("Atenção, mais de um modelo com dAIC<=2. No entanto,
apenas o primeiro será analisado.")
          }
          if(modelo_topo=="m_nulo")# criar objeto "modelo escolhido" com
modelo nulo
          {modelo_escolhido <- m_nulo}
          if(modelo_topo=="m_1") # ou criar objeto "modelo escolhido" com
modelo 1
          {modelo_escolhido <- m_1}
        }
```

```
        if(AICc==T) {tabela.AIC <- AICctab(m_nulo,m_1, weights=T,
delta=TRUE, base=TRUE)#teste para saber se sera feita selecao de modelos por
AICc. Entao eh gerada a tabela de AICc (para pequenas amostras)
        class(tabela.AIC) <- "data.frame" #transformei em data.frame para
poder transformar o nome das linhas em coluna
        tabela.AIC <- tibble::rownames_to_column(tabela.AIC, "modelo")
#transformei os nomes das linhas em coluna
        modelo_topo <- tabela.AIC$modelo[tabela.AIC$dAICc<=2]#objeto criado
com o modelo selecionado, i.e., com delta AICc menor ou igual a 2. Pode ser
mais de um modelo.
        if((length(modelo_topo))>1) #criar mensagem de aviso caso tenha mais
de um modelo com dAIC<=2.
        {
          message("Atenção, mais de um modelo com dAIC<=2. No entanto,
apenas o primeiro será analisado.")
        }
        if(modelo_topo=="m_nulo")# criar objeto "modelo escolhido" com
modelo nulo
        {modelo_escolhido <- m_nulo}
        if(modelo_topo=="m_1") # ou criar objeto "modelo escolhido" com
modelo 1
        {modelo_escolhido <- m_1}
        }
        ##Testes de distribuicao e homocedasticidade dos residuos do modelo
        teste_normalidade <- plotresid(modelo_escolhido, shapiro = T) # gera
graficos para verificar visualmente a ditribuicao dos residuos e a
homogeneidade de variancia dos residuos (homocedasticidade), além fazer o
teste de shapiro para verificar se os residuos do modelo seguem uma
distribuicao normal.
        x11() #para abrir uma janela de gráfico
        # gera um histograma para verificar se os residuos do modelo seguem
uma distribuicao normal
        r = residuals(modelo_escolhido)
        plotNormalHistogram(r)
        if(teste_normalidade$p.value<=0.05)  #Teste para saber se o teste de
shapiro deu significativo. Se der, aparece a mensagem abaixo
        {message("Parece que os resíduos do seu modelo não possuem uma
distribuição normal. Sugerimos que verifique visualmente e, se necessário,
mude o tipo de modelagem para uma que use uma distribuição que se adeque
melhor aos seus dados. Análise continua.")}#Apesar da mensagem, a funcao
continua. O usuario tera que avaliar, nao soh com base no teste de shapiro,
mas tambem pelas figuras geradas, se deve ou nao usar outro tipo de modelo,
talvez um modelo linear misto nao seja o mais adequado
        resultado <- summary(modelo_escolhido)} #objeto criado com o sumario
do modelo selecionado
      ## Continua a serie de Testes logicos para saber se tem variavel z e K
caso a anterior nao tenha dado positivo
      if(!is.null(z) & !is.null(k)) #Se tiver variavel aleatoria e segunda
variavel resposta
        # cria modelos lineraes mistos (coloquei REML=F porque eh preciso
ajustar o modelo por máxima verossimilhança (ML), porque é o indicado para
```

```
comparar modelos com diferentes efeitos fixos mas com mesmo efeito
aleatório). Todos com efeito aletorio (z)
      {
        m_nulo <- lmer(y ~ 1 + (1|z), data, REML = F)#modelo linear misto
nulo
        m_1 <- lmer(y ~ x + (1|z),  data, REML = F)#modelo de y em funcao de
x
        m_2 <- lmer(y ~ k + (1|z), data=, REML = F)#modelo de y em funcao de
k
        m_3 <- lmer(y ~ x + k + (1|z), data, REML = F)#modelo de y em funcao
de x e efeito aditivo de k
        suppressMessages(m_4 <- lmer(y ~ x:k + (1|z), data, REML = F))#
modelo de y em funcao da interacao de x e k. (#suprimi algumas mensagens de
aviso do r)
        m_5 <- lmer(y ~ x:k + x + (1|z), data, REML = F)#modelo de y em
funcao da intercao de x e k e com efeito aditivo de x
        m_6 <- lmer(y ~ x:k + k + (1|z), data, REML = F)#modelo de y em
funcao da interacao entre x e k com efeito aditivo de k
        m_7 <- lmer(y ~ x:k + x + k + (1|z), data, REML = F)#modelo de y em
funcao da interacao de x e k com efeito aditivo de x e k
        if(AICc==F)
        {
          tabela.AIC <- AICtab(m_nulo, m_1, m_2, m_3, m_4, m_5, m_6, m_7,
weights=T, delta=TRUE, base=TRUE) #teste para saber se sera feita selecao de
modelos por AIC. Entao eh gerada a tabela de AIC
          class(tabela.AIC) <- "data.frame" #transformei em data.frame para
poder transformar o nome das linhas em coluna
          tabela.AIC <- tibble::rownames_to_column(tabela.AIC, "modelo")
#transformei os nomes das linhas em coluna
          modelo_topo <- tabela.AIC$modelo[tabela.AIC$dAIC<=2]# objeto
criado com o modelo selecionado pelo teste logico se delta AIC eh menor ou
igual a 2. Pode selecionar mais de um modelo
          if((length(modelo_topo))>1) #criar mensagem de aviso caso tenha
mais de um modelo com dAIC<=2.
          {
            message("Atenção, mais de um modelo com dAIC<=2. No entanto,
apenas o primeiro será analisado.")
          }
          if(modelo_topo=="m_nulo")# criar objeto "modelo escolhido" com
modelo nulo
          {modelo_escolhido <- m_nulo}
          if(modelo_topo=="m_1") # ou criar objeto "modelo escolhido" com
modelo 1
          {modelo_escolhido <- m_1}
          if(modelo_topo=="m_2")# ou criar objeto "modelo escolhido" com
modelo 2
          {modelo_escolhido <- m_2}
          if(modelo_topo=="m_3") # ou criar objeto "modelo escolhido" com
modelo 3
          {modelo_escolhido <- m_3}
```

```
          if(modelo_topo=="m_4")# ou criar objeto "modelo escolhido" com
modelo 4
          {modelo_escolhido <- m_4}
          if(modelo_topo=="m_5") # ou criar objeto "modelo escolhido" com
modelo 5
          {modelo_escolhido <- m_5}
          if(modelo_topo=="m_6")# ou criar objeto "modelo escolhido" com
modelo 6
          {modelo_escolhido <- m_6}
          if(modelo_topo=="m_7") # ou criar objeto "modelo escolhido" com
modelo 7
          {modelo_escolhido <- m_7}
        }
        if(AICc==T)
        {
          tabela.AIC <- AICctab(m_nulo,m_1, m_2, m_3, m_4,m_5,m_6,m_7,
weights=T, delta=TRUE, base=TRUE) #teste para saber se sera feita selecao de
modelos por AIC. Entao eh gerada a tabela de AICc (para pequenas amostras)
          class(tabela.AIC) <- "data.frame" #transformei em data.frame para
poder transformar o nome das linhas em coluna
          tabela.AIC <- tibble::rownames_to_column(tabela.AIC, "modelo")
#transformei os nomes das linhas em coluna
          modelo_topo <- tabela.AIC$modelo[tabela.AIC$dAICc<=2]#objeto
criado com o modelo selecionado pelo teste logico se delta AICc eh menor ou
igual a 2. Pode selecionar mais de um modelo
          if((length(modelo_topo))>1) #criar mensagem de aviso caso tenha
mais de um modelo com dAIC<=2.
          {
            message("Atenção, mais de um modelo com dAIC<=2. No entanto,
apenas o primeiro será analisado.")
          }
          if(modelo_topo=="m_nulo")# criar objeto "modelo escolhido" com
modelo nulo
          {modelo_escolhido <- m_nulo}
          if(modelo_topo=="m_1") # ou criar objeto "modelo escolhido" com
modelo 1
          {modelo_escolhido <- m_1}
          if(modelo_topo=="m_2")# ou criar objeto "modelo escolhido" com
modelo 2
          {modelo_escolhido <- m_2}
          if(modelo_topo=="m_3") # ou criar objeto "modelo escolhido" com
modelo 3
          {modelo_escolhido <- m_3}
          if(modelo_topo=="m_4")# ou criar objeto "modelo escolhido" com
modelo 4
          {modelo_escolhido <- m_4}
          if(modelo_topo=="m_5") # ou criar objeto "modelo escolhido" com
modelo 5
          {modelo_escolhido <- m_5}
          if(modelo_topo=="m_6")# ou criar objeto "modelo escolhido" com
modelo 6
```

```
          {modelo_escolhido <- m_6}
          if(modelo_topo=="m_7") # ou criar objeto "modelo escolhido" com
modelo 7
          {modelo_escolhido <- m_7}
        }
        ##Testes de distribuicao e homocedasticidade dos residuos do modelo
        teste_normalidade <- plotresid(modelo_escolhido, shapiro = T) # gera
graficos para verificar visualmente a ditribuicao dos residuos e a
homogeneidade de variancia dos residuos (homocedasticidade), além fazer o
teste de shapiro para verificar se os residuos do modelo seguem uma
distribuicao normal.
        x11() #para abrir uma janela de gráfico
        # gera um histograma para verificar se os residuos do modelo seguem
uma distribuicao normal
        r = residuals(modelo_escolhido)
        plotNormalHistogram(r)
        if(teste_normalidade$p.value<=0.05) #Teste para saber se o teste de
shapiro deu significativo. se der, aparece a mensagem abaixo
        {message("Parece que os resíduos do seu modelo não possuem uma
distribuição normal. Sugerimos que verifique visualmente e, se necessário,
mude o tipo de modelagem para uma que use uma distribuição que se adeque
melhor aos seus dados. Análise continua.")}#Apesar da mensagem, a funcao
continua. O usuario tera que avaliar, nao soh com base no teste de shapiro,
mas tambem pelas figuras geradas, se deve ou nao usar outro tipo de modelo,
talvez um modelo linear misto nao seja o mais adequado
        resultado <- summary(modelo_escolhido)
      } #objeto criado com o sumario do modelo selecionado
      ##Continua a serie de Testes logicos para saber se tem variavel z e K
caso a anterior nao tenha dado positivo
      if(is.null(z) & !is.null(k)) #Se nao tiver varaiavel aleatoria e tiver
segunda variavel resposta
        #modelos lineares serao gerados, nao tem efeito aleatorio.
      {
        m_nulo <- lm(y ~ 1,  data) #modelo nulo
        m_1 <- lm(y ~ x, data)#modelo de y em funcao de x
        m_2 <- lm(y ~ k, data)#modelo de y em funcao de k
        m_3 <- lm(y ~ x + k, data)#modelo de y em funcao de x com efeito
aditivo de k
        m_4 <- lm(y ~ x:k, data)#modelo de y em funcao de x com interacao
com k
        m_5 <- lm(y ~ x:k + x, data)#modelo de y em funcao de x com
interacao com k e com efeito aditivo de x
        m_6 <- lm(y ~ x:k + k, data)#modelo de y em funcao de x com
interacao com k com efeito aditivo de k
        m_7 <- lm(y ~ x:k + x + k, data) #modelo de y em funcao de x com
interacao com k e efeito aditivo de x e k
        if(AICc==F)
        {
          tabela.AIC <- AICtab(m_nulo,m_1, m_2, m_3, m_4,m_5,m_6,m_7,
weights=T, delta=TRUE, base=TRUE) #teste para saber se sera feita selecao de
```

```
modelos por AIC. Entao eh gerada a tabela de AIC
          class(tabela.AIC) <- "data.frame" #transformei em data.frame para
poder transformar o nome das linhas em coluna
          tabela.AIC <- tibble::rownames_to_column(tabela.AIC, "modelo")
#transformei os nomes das linhas em coluna
          modelo_topo <- tabela.AIC$modelo[tabela.AIC$dAIC<=2] #objeto
criado com o modelo selecionado pelo teste se delta AIC for menor ou igual a
2
          if((length(modelo_topo))>1) #criar mensagem de aviso caso tenha
mais de um modelo com dAIC<=2.
          {
            message("Atenção, mais de um modelo com dAIC<=2. No entanto,
apenas o primeiro será analisado.")
          }
          if(modelo_topo=="m_nulo")# criar objeto "modelo escolhido" com
modelo nulo
          {modelo_escolhido <- m_nulo}
          if(modelo_topo=="m_1") # ou criar objeto "modelo escolhido" com
modelo 1
          {modelo_escolhido <- m_1}
          if(modelo_topo=="m_2")# ou criar objeto "modelo escolhido" com
modelo 2
          {modelo_escolhido <- m_2}
          if(modelo_topo=="m_3") # ou criar objeto "modelo escolhido" com
modelo 3
          {modelo_escolhido <- m_3}
          if(modelo_topo=="m_4")# ou criar objeto "modelo escolhido" com
modelo 4
          {modelo_escolhido <- m_4}
          if(modelo_topo=="m_5") # ou criar objeto "modelo escolhido" com
modelo 5
          {modelo_escolhido <- m_5}
          if(modelo_topo=="m_6")# ou criar objeto "modelo escolhido" com
modelo 6
          {modelo_escolhido <- m_6}
          if(modelo_topo=="m_7") # ou criar objeto "modelo escolhido" com
modelo 7
          {modelo_escolhido <- m_7}
        }
        if(AICc==T)
        {
          tabela.AIC <- AICctab(m_nulo, m_1, m_2, m_3, m_4, m_5, m_6, m_7,
weights=T, delta=TRUE, base=TRUE) #teste para saber se sera feita selecao de
modelos por AICc. Entao eh gerada a tabela de AICc (para pequenas amostras)
          class(tabela.AIC) <- "data.frame" #transformei em data.frame para
poder transformar o nome das linhas em coluna
          tabela.AIC <- tibble::rownames_to_column(tabela.AIC, "modelo")
#transformei os nomes das linhas em coluna
          modelo_topo <- tabela.AIC$modelo[tabela.AIC$dAICc<=2] #objeto
criado com o modelo selecionado pelo teste se delta AICc for menor ou igual
a 2
```

```
          if((length(modelo_topo))>1) #criar mensagem de aviso caso tenha
mais de um modelo com dAIC<=2.
          {
            message("Atenção, mais de um modelo com dAIC<=2. No entanto,
apenas o primeiro será analisado.")
          }
          if(modelo_topo=="m_nulo")# criar objeto "modelo escolhido" com
modelo nulo
          {modelo_escolhido <- m_nulo}
          if(modelo_topo=="m_1") # ou criar objeto "modelo escolhido" com
modelo 1
          {modelo_escolhido <- m_1}
          if(modelo_topo=="m_2")# ou criar objeto "modelo escolhido" com
modelo 2
          {modelo_escolhido <- m_2}
          if(modelo_topo=="m_3") # ou criar objeto "modelo escolhido" com
modelo 3
          {modelo_escolhido <- m_3}
          if(modelo_topo=="m_4")# ou criar objeto "modelo escolhido" com
modelo 4
          {modelo_escolhido <- m_4}
          if(modelo_topo=="m_5") # ou criar objeto "modelo escolhido" com
modelo 5
          {modelo_escolhido <- m_5}
          if(modelo_topo=="m_6")# ou criar objeto "modelo escolhido" com
modelo 6
          {modelo_escolhido <- m_6}
          if(modelo_topo=="m_7") # ou criar objeto "modelo escolhido" com
modelo 7
          {modelo_escolhido <- m_7}
        }
        ##Testes de distribuicao e homocedasticidade dos residuos do modelo
        teste_normalidade <- plotresid(modelo_escolhido, shapiro = T) # gera
graficos para verificar visualmente a ditribuicao dos residuos e a
homogeneidade de variancia dos residuos (homocedasticidade), além fazer o
teste de shapiro para verificar se os residuos do modelo seguem uma
distribuicao normal.
        x11() #para abrir uma janela de gráfico
        # gera um histograma para verificar se os residuos do modelo seguem
uma distribuicao normal
        r = residuals(modelo_escolhido)
        plotNormalHistogram(r)
        x11() #para abrir uma janela de gráfico
        par(mfrow=c(2,2)) #para mostrar os 4 graficos de uma vez
        plot(modelo_escolhido) # plota 4 graficos para averiguar a
distribuicao, homocedasticidade e alavancagem dos residuos do modelo
        if(teste_normalidade$p.value<=0.05)#Teste para saber se o teste de
shapiro deu significativo. se der, aparece a mensagem abaixo
        {message("Parece que os resíduos do seu modelo não possuem uma
distribuição normal. Sugerimos que verifique visualmente e, se necessário,
```

```
mude o tipo de modelagem para uma que use uma distribuição que se adeque
melhor aos seus dados. Análise continua.")}#Apesar da mensagem, a funcao
continua. O usuario tera que avaliar, nao soh com base no teste de shapiro,
mas tambem pelas figuras geradas, se deve ou nao usar outro tipo de modelo,
talvez um modelo linear nao seja o mais adequado
        resultado <- summary(modelo_escolhido) #objeto criado com o sumario
do modelo selecionado
      }
    }
    ##Teste logico para saber se o modelo nulo nao sera incluido na analise
(caso o teste sobre modelo nulo la do inicio nao tenha dado positivo ele
pula direto para esse)
    if(nulo==F) #se o modelo nulo nao for incluso na analise
    {
      if(is.null(z) & is.null(k))
      { #Se nao tiver variavel aleatoria nem segunda variavel resposta
        modelo_escolhido <- lm(y ~ x, data)   # Fazer modelo linear de y em
funcao de x (na verdade esse if nao precisaria ser criado, ja que mesmo que
o usuario coloque nulo=F, a anova vai comparar com o modelo nulo, alem disso
esse mesmo modelo ja foi feito anteriormente, mas caso o usuario nao tenha
essa percepcao, esse modelo foi incluido para nao dar erro caso o usuario
coloque nulo=F, z=F e k=F)
        ##Testes de distribuicao e homocedasticidade dos residuos do modelo
        teste_normalidade <- plotresid(modelo_escolhido, shapiro = T) # gera
graficos para verificar visualmente a ditribuicao dos residuos e a
homogeneidade de variancia dos residuos (homocedasticidade), além fazer o
teste de shapiro para verificar se os residuos do modelo seguem uma
distribuicao normal.
        x11() #para abrir uma janela de gráfico
        # gera um histograma para verificar se os residuos do modelo seguem
uma distribuicao normal
        r = residuals(modelo_escolhido)
        plotNormalHistogram(r)
        x11() #para abrir uma janela de gráfico
        par(mfrow=c(2,2)) #para aparecer os 4 graficos de uma vez
        plot(modelo_escolhido) # plota 4 graficos para averiguar a
distribuicao, homocedasticidade e alavancagem dos residuos do modelo
        #Teste para saber se o teste de shapiro deu significativo
        if(teste_normalidade$p.value<=0.05) #se der, aparece a mensagem
abaixo
        {message("Parece que os resíduos do seu modelo não possuem uma
distribuição normal. Sugerimos que verifique visualmente e, se necessário,
mude o tipo de modelagem para uma que use uma distribuição que se adeque
melhor aos seus dados.")}
        #Apesar da mensagem, a funcao continua. O usuario tera que avaliar,
nao soh com base no teste de shapiro, mas tambem pelas figuras geradas, se
deve ou nao usar outro tipo de modelo, talvez um modelo linear nao seja o
mais adequado
        tabela.AIC = "Não é aplicável"
        modelo_tpo = "Não é aplicável"
        resultado <- anova(modelo_escolhido) # obejeto criado com a anova do
```

```
modelo
      }
      #serie de Testes logicos para saber se tem variavel z e K
      if(!is.null(z) & is.null(k)) #Se tiver variavel aleatoria, mas nao uma
segunda variavel resposta
      {
        modelo_escolhido <- lmer(y ~ x + (1|z),  data, REML = F) # cria
modelo linear misto de y em funcao de x.
        ##Testes de distribuicao e homocedasticidade dos residuos do modelo
        teste_normalidade <- plotresid(modelo_escolhido, shapiro = T) # gera
graficos para verificar visualmente a ditribuicao dos residuos e a
homogeneidade de variancia dos residuos (homocedasticidade), além fazer o
teste de shapiro para verificar se os residuos do modelo seguem uma
distribuicao normal.
        x11() #para abrir uma janela de gráfico
        # gera um histograma para verificar se os residuos do modelo seguem
uma distribuicao normal
        r = residuals(modelo_escolhido)
        plotNormalHistogram(r)
        if(teste_normalidade$p.value<=0.05)#Teste para saber se o teste de
shapiro deu significativo. se der, aparece a mensagem abaixo
        {message("Parece que os resíduos do seu modelo não possuem uma
distribuição normal. Sugerimos que verifique visualmente e, se necessário,
mude o tipo de modelagem para uma que use uma distribuição que se adeque
melhor aos seus dados. Análise continua.")}#Apesar da mensagem, a funcao
continua. O usuario tera que avaliar, nao soh com base no teste de shapiro,
mas tambem pelas figuras geradas, se deve ou nao usar outro tipo de modelo,
talvez um modelo linear misto nao seja o mais adequado
        resultado <- summary(m_1)} #objeto criado com o sumario do modelo
      ####Continua a serie de Testes logicos para saber se tem variavel z e
K caso a anterior nao tenha dado positivo#####
      if(!is.null(z) & !is.null(k)) #Se tiver variavel aleatoria e segunda
variavel resposta
      {
        #sao criados modelos lineares mistos (com efeito aleatorio, z).
Coloquei REML=F porque eh preciso ajustar o modelo por máxima
verossimilhança (ML), porque é o indicado para comparar modelos com
diferentes efeitos fixos mas com mesmo efeito aleatório
        m_1 <- lmer(y ~ x + (1|z),  data, REML = F) #modelo de y em funcao
de x
        m_2 <- lmer(y ~ k + (1|z), data=, REML = F) #modelo de y em funcao k
        m_3 <- lmer(y ~ x + k + (1|z), data, REML = F) #modelo de y em
funcao de x com efeito aditivo de k
        m_4 <- lmer(y ~ x:k + (1|z), data, REML = F)#modelo de y em funcao
de x com interacao com k
        m_5 <- lmer(y ~ x:k + x + (1|z), data, REML = F)#modelo de y em
funcao de x com interacao com k e efeito aditivo de x
        m_6 <- lmer(y ~ x:k + k + (1|z), data, REML = F)#modelo de y em
funcao de x com interacao com k com efeito aditivo de k
        m_7 <- lmer(y ~ x:k + x + k + (1|z), data, REML = F)#modelo de y em
```

```
funcao de x com interacao com k com efeito aditivo de x e k
        if(AICc==F)
        {
          tabela.AIC <- AICtab(m_1, m_2, m_3, m_4,m_5,m_6,m_7, weights=T,
delta=TRUE, base=TRUE) #teste para saber se sera feita selecao de modelos
por AIC. Entao eh gerada a tabela de AIC
          class(tabela.AIC) <- "data.frame" #transformei em data.frame para
poder transformar o nome das linhas em coluna
          tabela.AIC <- tibble::rownames_to_column(tabela.AIC, "modelo")
#transformei os nomes das linhas em coluna
          modelo_topo <- tabela.AIC$modelo[tabela.AIC$dAIC<=2] #objeto com
modelo selecionado pelo AIC, usando o teste se delta AIC eh menor ou igual a
2
          if(length(modelo_topo>1))
          {
            message("Atenção, mais de um modelo com dAIC<=2. No entanto,
apenas o primeiro será analisado.")
          }
          if(modelo_topo=="m_1") # criar objeto "modelo escolhido" com
modelo 1
          {modelo_escolhido <- m_1}
          if(modelo_topo=="m_2")# ou criar objeto "modelo escolhido" com
modelo 2
          {modelo_escolhido <- m_2}
          if(modelo_topo=="m_3") # ou criar objeto "modelo escolhido" com
modelo 3
          {modelo_escolhido <- m_3}
          if(modelo_topo=="m_4")# ou criar objeto "modelo escolhido" com
modelo 4
          {modelo_escolhido <- m_4}
          if(modelo_topo=="m_5") # ou criar objeto "modelo escolhido" com
modelo 5
          {modelo_escolhido <- m_5}
          if(modelo_topo=="m_6")# ou criar objeto "modelo escolhido" com
modelo 6
          {modelo_escolhido <- m_6}
          if(modelo_topo=="m_7") # ou criar objeto "modelo escolhido" com
modelo 7
          {modelo_escolhido <- m_7}
        }
        if(AICc==T)
        {
          tabela.AIC <- AICctab(m_1, m_2, m_3, m_4,m_5,m_6,m_7, weights=T,
delta=TRUE, base=TRUE) #teste para saber se sera feita selecao de modelos
por AICc. Entao eh gerada a tabela de AICc (para pequenas amostras)
          class(tabela.AIC) <- "data.frame" #transformei em data.frame para
poder transformar o nome das linhas em coluna
          tabela.AIC <- tibble::rownames_to_column(tabela.AIC, "modelo")
#transformei os nomes das linhas em coluna
          modelo_topo <- tabela.AICc$modelo[tabela.AICc$dAICc<=2] #objeto
com modelo selecionado pelo AICc, usando o teste se delta AICc eh menor ou
```

```
igual a 2
          if((length(modelo_topo))>1) #criar mensagem de aviso caso tenha
mais de um modelo com dAIC<=2.
          {
            message("Atenção, mais de um modelo com dAIC<=2. No entanto,
apenas o primeiro será analisado.")
          }
          if(modelo_topo=="m_1") # criar objeto "modelo escolhido" com
modelo 1
          {modelo_escolhido <- m_1}
          if(modelo_topo=="m_2")# ou criar objeto "modelo escolhido" com
modelo 2
          {modelo_escolhido <- m_2}
          if(modelo_topo=="m_3") # ou criar objeto "modelo escolhido" com
modelo 3
          {modelo_escolhido <- m_3}
          if(modelo_topo=="m_4")# ou criar objeto "modelo escolhido" com
modelo 4
          {modelo_escolhido <- m_4}
          if(modelo_topo=="m_5") # ou criar objeto "modelo escolhido" com
modelo 5
          {modelo_escolhido <- m_5}
          if(modelo_topo=="m_6")# ou criar objeto "modelo escolhido" com
modelo 6
          {modelo_escolhido <- m_6}
          if(modelo_topo=="m_7") # ou criar objeto "modelo escolhido" com
modelo 7
          {modelo_escolhido <- m_7}
        }
        ##Testes de distribuicao e homocedasticidade dos residuos do modelo
        teste_normalidade <- plotresid(modelo_escolhido, shapiro = T) # gera
graficos para verificar visualmente a ditribuicao dos residuos e a
homogeneidade de variancia dos residuos (homocedasticidade), além fazer o
teste de shapiro para verificar se os residuos do modelo seguem uma
distribuicao normal.
        x11() #para abrir uma janela de gráfico
        # gera um histograma para verificar se os residuos do modelo seguem
uma distribuicao normal
        r = residuals(modelo_escolhido)
        plotNormalHistogram(r)
        if(teste_normalidade$p.value<=0.05)#Teste para saber se o teste de
shapiro deu significativo. se der, aparece a mensagem abaixo
        {message("Parece que os resíduos do seu modelo não possuem uma
distribuição normal. Sugerimos que verifique visualmente e, se necessário,
mude o tipo de modelagem para uma que use uma distribuição que se adeque
melhor aos seus dados. Análise continua.")}#Apesar da mensagem, a funcao
continua. O usuario tera que avaliar, nao soh com base no teste de shapiro,
mas tambem pelas figuras geradas, se deve ou nao usar outro tipo de modelo,
talvez um modelo linear misto nao seja o mais adequado
        resultado <- summary(modelo_escolhido) #objeto criado com o sumario
```

```
do modelo selecionado
      }
      ##Continua a serie de Testes logicos para saber se tem variavel z e K
caso a anterior nao tenha dado positivo
      if(is.null(z) & !is.null(k))#Se nao tiver variavel aleatoria e tiver
segunda variavel resposta
      {
        #modelos lineares serao criados
        m_1 <- lm(y ~ x, data) #modelo de y em funcao de x
        m_2 <- lm(y ~ k, data) #modelo de y em funcao de k
        m_3 <- lm(y ~ x + k, data) #modelo de y em funcao de x com efeito
aditivo de k
        m_4 <- lm(y ~ x:k, data) #modelo de y em funcao de x com interacao
com k
        m_5 <- lm(y ~ x:k + x, data) #modelo de y em funcao de x com
interacao com k e efeito aditivo de x
        m_6 <- lm(y ~ x:k + k, data) #modelo de y em funcao de x com
intercao com k e efeito aditivo de k
        m_7 <- lm(y ~ x:k + x + k, data) #modelo de y em funcao de x com
interacao com k e efeito aditivo de x e k
        if(AICc==F)
        {
          tabela.AIC <- AICtab(m_1, m_2, m_3, m_4,m_5,m_6,m_7, weights=T,
delta=TRUE, base=TRUE)#teste para saber se sera feita selecao de modelos por
AIC. Entao eh gerada a tabela de AIC
          class(tabela.AIC) <- "data.frame" #transformei em data.frame para
poder transformar o nome das linhas em coluna
          tabela.AIC <- tibble::rownames_to_column(tabela.AIC, "modelo")
#transformei os nomes das linhas em coluna
          modelo_topo <- tabela.AIC$modelo[tabela.AIC$dAIC<=2] #objeto
criado com o modelo selecionado por AIC, usando o teste se delta AIC for
menor ou igual a 2
          if((length(modelo_topo))>1) #criar mensagem de aviso caso tenha
mais de um modelo com dAIC<=2.
          {
            message("Atenção, mais de um modelo com dAIC<=2. No entanto,
apenas o primeiro será analisado.")
          }
          if(modelo_topo=="m_1") # criar objeto "modelo escolhido" com
modelo 1
          {modelo_escolhido <- m_1}
          if(modelo_topo=="m_2")# ou criar objeto "modelo escolhido" com
modelo 2
          {modelo_escolhido <- m_2}
          if(modelo_topo=="m_3") # ou criar objeto "modelo escolhido" com
modelo 3
          {modelo_escolhido <- m_3}
          if(modelo_topo=="m_4")# ou criar objeto "modelo escolhido" com
modelo 4
          {modelo_escolhido <- m_4}
          if(modelo_topo=="m_5") # ou criar objeto "modelo escolhido" com
```

```
modelo 5
          {modelo_escolhido <- m_5}
          if(modelo_topo=="m_6")# ou criar objeto "modelo escolhido" com
modelo 6
          {modelo_escolhido <- m_6}
          if(modelo_topo=="m_7") # ou criar objeto "modelo escolhido" com
modelo 7
          {modelo_escolhido <- m_7}
        }
        if(AICc==T)
        {
          tabela.AIC <- AICctab(m_1, m_2, m_3, m_4,m_5,m_6,m_7, weights=T,
delta=TRUE, base=TRUE)#teste para saber se sera feita selecao de modelos por
AICc. Entao eh gerada a tabela de AICc (para pequenas amostras)
          class(tabela.AIC) <- "data.frame" #transformei em data.frame para
poder transformar o nome das linhas em coluna
          tabela.AIC <- tibble::rownames_to_column(tabela.AIC, "modelo")
#transformei os nomes das linhas em coluna
          modelo_topo <- tabela.AICc$modelo[tabela.AICc$dAICc<=2] #objeto
com modelo selecionado por AICc, usando o teste se delta AICc for menor ou
igual a 2
          if((length(modelo_topo))>1) #criar mensagem de aviso caso tenha
mais de um modelo com dAIC<=2.
          {
            message("Atenção, mais de um modelo com dAIC<=2. No entanto,
apenas o primeiro será analisado.")
          }
          if(modelo_topo=="m_1") # criar objeto "modelo escolhido" com
modelo 1
          {modelo_escolhido <- m_1}
          if(modelo_topo=="m_2")# ou criar objeto "modelo escolhido" com
modelo 2
          {modelo_escolhido <- m_2}
          if(modelo_topo=="m_3") # ou criar objeto "modelo escolhido" com
modelo 3
          {modelo_escolhido <- m_3}
          if(modelo_topo=="m_4")# ou criar objeto "modelo escolhido" com
modelo 4
          {modelo_escolhido <- m_4}
          if(modelo_topo=="m_5") # ou criar objeto "modelo escolhido" com
modelo 5
          {modelo_escolhido <- m_5}
          if(modelo_topo=="m_6")# ou criar objeto "modelo escolhido" com
modelo 6
          {modelo_escolhido <- m_6}
          if(modelo_topo=="m_7") # ou criar objeto "modelo escolhido" com
modelo 7
          {modelo_escolhido <- m_7}
        }
        ##Testes de distribuicao e homocedasticidade dos residuos do modelo
```

```
        teste_normalidade <- plotresid(modelo_escolhido, shapiro = T) # gera
graficos para verificar visualmente a ditribuicao dos residuos e a
homogeneidade de variancia dos residuos (homocedasticidade), além fazer o
teste de shapiro para verificar se os residuos do modelo seguem uma
distribuicao normal.
        x11() #para abrir uma janela de gráfico
        # gera um histograma para verificar se os residuos do modelo seguem
uma distribuicao normal
        r = residuals(modelo_escolhido)
        plotNormalHistogram(r)
        x11() #para abrir uma janela de gráfico
        par(mfrow=c(2,2)) #para mostrar os 4 graficos de uma vez
        plot(modelo_escolhido) # plota 4 graficos para averiguar a
distribuicao, homocedasticidade e alavancagem dos residuos do modelo
        if(teste_normalidade$p.value<=0.05) #Teste para saber se o teste de
shapiro deu significativo. se der, aparece a mensagem abaixo
        {message("Parece que os resíduos do seu modelo não possuem uma
distribuição normal. Sugerimos que verifique visualmente e, se necessário,
mude o tipo de modelagem para uma que use uma distribuição que se adeque
melhor aos seus dados. Análise continua.")}#Apesar da mensagem, a funcao
continua. O usuario tera que avaliar, nao soh com base no teste de shapiro,
mas tambem pelas figuras geradas, se deve ou nao usar outro tipo de modelo,
talvez um modelo linear nao seja o mais adequado
        resultado <- summary(modelo_escolhido)} #objeto com o sumario do
modelo selecionado
    }
return(list(tabela.AIC=tabela.AIC,modelo_topo=modelo_topo,modelo_escolhido=m
odelo_escolhido,resultado=resultado)) #a funcao foi finalizada, retornando a
tabela de AIC ou AICc, o(s) modelos(s) topo, modelo escolhido e resultado
(anova ou summario do modelo escolhido)
  })
```

```
                                        HELP DA FUNÇÃO
```

```
    mayara                package:unknown                R Documentation
```

~~function to create and analize models ~~

```
 Description:

A funcao mayara é usada para criar e analisar o melhor modelo para um
conjunto de dados. A função irá: (1) montar os modelos lineares ou lineares
mistos, (2) selecionar o melhor modelo (se houver mais de um), (3) fazer um
teste de distribuição normal e criar gráficos para o usuário analisar a
distribuição e homocedatisticidade dos resíduos do modelo selecionado, (4)
realizar um summary ou anova (dependendo do tipo de modelo) do modelo
selecionado.
```

```
 Usage:
```

```
mayara(data, x, y, k = NULL , z = NULL, xlevels = NULL, klevels = NULL ,
nulo = T, AICc = F)
```

```
Arguments:


 ''data''
data.frame com os dados. Deve conter coluna com a variavel resposta, coluna
com variavel preditora 1, variavel preditora 2 (se tiver) e variável
aleatória (se tiver)
```

```
 ''x''
coluna da variável resposta
```

```
''y''
```

coluna da variável preditora 1

```
''k''
```

coluna da variável preditora 2 (se não for preenchido, o default será NULL))

```
 ''z''
 coluna da variável aleatória categórica (se não for preenchido, o default
será NULL)
```

xlevels níveis de x, caso seja uma variável categórica. Este argumento deverá ser preenchido da seguinte forma: Ex.: xlevels = c(“1”, “2”, “3”) neste exemplo temos 3 níveis. Se não for preenchido o default será NULL (i.e., x será numérico e não um fator).

```
''klevels''
```

níveis de k, caso seja uma variável categórica. Este argumento deverá ser preenchido da seguinte forma: Ex.: klevels = c(“1”, “2”, “3”) neste exemplo temos 3 níveis. Se não for preenchido o default será NULL (i.e., k será numérico e não um fator).

```
 ''nulo''
```

caso o usuário deseje usar o modelo 1 como modelo mais simples, deverá colocar nulo = F.

```
 ''AICc''
```

se o n amostral for pequeno, recomendamos que seja usado o AICc (AIC corrigido para pequenas amostras), colocando AICc = T.

```
Details:

 Para usar essa função é necessário que os dados estejam organizados em um
data.frame (outro formato, como uma matriz, por exemplo, não irá funcionar)
```

```
com os nomes das colunas especificados conforme os argumentos da função. Se, por exemplo, no lugar de k estiver "temperatura" ou "w", não irá funcionar.



 Para os argumentos "x" e "y" o usuário pode tanto colocar como argumento da função o nome da coluna do data.frame, por exemplo, x = data$área e y = data$temperatura e manter no data.frame os nomes das variáveis, área e temperatura, como mudar os nomes das colunas no data.frame para "x" e "y". No entanto, para os argumentos k e z é necessário que estejam k e z como nomes das colunas no data.frame.
```

```
 Nos argumentos é necessário indexar as colunas (usando $). Exemplo: x=data$x ou x=data$temperatura.
```

É necessário instalar os pacotes RVAideMemoire (para o teste de Shapiro e plots), bbmle (para AIC), lme4 (para lmer), rcompanion (para histoplotresid) e dplyr (para transformar o nome das linhas em colunas) antes de usar a função.

```
 Essa função só lida com modelos lineares ou lineares mistos.
```

```
Essa função só lida com 1 variável resposta (y) e até 2 variáveis preditoras de efeito fixo (x e k), além de ser possível inserir 1 variável aleatória (z).
```

```
A variável resposta não pode ser categórica. As variáveis preditoras de efeito fixo podem ser catregóricas ou numéricas. A variável preditora de efeito aleatório tem que ser categórica.
```

Value:

```
Gráficos para análise visual da distribuição e homocedasticidade dos resíduos do modelo ao longo da análise.
```

```
Lista com:
```

comp1 : tabela de AIC ou AICc

comp2 : modelos plausíveis (chamado de modelo_topo)

comp3 : modelo selecionado (modelo que foi analisado)

comp4 : um objeto chamado resultado com uma anova (para modelo linear) ou summary (para modelo linear misto) do modelo

```
 Warning:
```

```
 Recomendamos aos usuarios que verifiquem se seus dados estão corretos, com todas as variáveis que desejem incluir no modelo e organizados em um data.frame. Sugerimos aos usuários que usem a função name() para alterar o
```

```
nome das colunas do data.frame para os respectivos nomes dos argumentos.
```

```
Author(s):

Mayara de Almeida Jordano
mayara.jordano@usp.br
```

```
References:
```

```
Bates, D., Maechler, M., Bolker, B., Walker, S., 2015. Fitting linear mixed-
effects models using lme4. J. Stat. Softw. 67, 1–48.
https://doi.org/10.18637/jss.v067.i01.

Ben Bolker and R Development Core Team (2017). bbmle: Tools for General
Maximum Likelihood
Estimation. R package version 1.0.20.
https://CRAN.R-project.org/package=bbmle

Hadley Wickham, Romain François, Lionel Henry and Kirill Müller (2019).
dplyr: A Grammar of
Data Manipulation. R package version 0.8.1.
https://CRAN.R-project.org/package=dplyr

Maxime Hervé (2019). RVAideMemoire: Testing and Plotting Procedures for
Biostatistics. R
package version 0.9-73. https://CRAN.R-project.org/package=RVAideMemoire

Salvatore Mangiafico (2019). rcompanion: Functions to Support Extension
Education Program
Evaluation. R package version 2.2.1.
https://CRAN.R-project.org/package=rcompanion
```

Examples:

```
 link de acesso aos dados:
```

http://ecologia.ib.usp.br/bie5782/doku.php?id=dados:dados-mudas

```
Abrindo o arquivo

  mudas <- read.table("altura-mudas.csv", header=TRUE, sep=",", dec=".",
as.is=TRUE)
  names(mudas) <- c("k", "z", "x", "y")
```

```
 Exemplo com as variáveis preditoras de efeito fixo 2 (k)  e de efeito
aleatório (z)
 mayara(data=mudas, x=mudas$x, y=mudas$y, k=mudas$k, z=mudas$z, klevels=
c("paineira", "tamboril"), xlevels =
 c("1", "2", "3", "4", "5", "6", "7", "8", "9", "10" ), nulo=T, AICc = T)
```

```
 Exemplo sem a variável preditora de efeito fixo 2 (k)
 muda <- mudas[-1]
 mayara(data=muda, x=muda$x, y=muda$y, z=muda$z, k= NULL, xlevels = c("1",
"2", "3", "4", "5", "6", "7", "8", "9", "10" ), nulo=T, AICc = T)
```

```
 Exemplo sem a variável preditora de efeito aletório (z)
 muda_3 <- mudas[-2]
 mayara(data=muda_3, x=muda_3$x, y=muda_3$y, k=muda_3$k, klevels=
c("paineira", "tamboril"), xlevels = c("1", "2", "3", "4", "5", "6", "7",
"8", "9", "10" ), z = NULL, nulo=F, AICc = F)
```

```
 Exemplo sem as variáveis preditoras de efeito fixo 2 (k) e de efeito
aleatório (z)
 muda_4 <- mudas[c(-1,-2)]
 mayara(data=muda_4, x=muda_4$x, y=muda_4$y, xlevels = c("1", "2", "3", "4",
"5", "6", "7", "8", "9", "10" ),k= NULL, z = NULL, nulo=T, AICc = F)
```

```
Teste para data!=data.frame
muda_matriz <- as.matrix(mudas)
mayara(data=muda_matriz, x=muda_matriz[3], y=muda_matriz[4],
k=muda_matriz[1], z=muda_matriz[2], klevels= c("paineira", "tamboril"),
xlevels = c("1", "2", "3", "4", "5", "6", "7", "8", "9", "10" ), nulo=T,
AICc = T)
```

Atenção

```
Para rodar os exemplos baixe o arquivo
```

altura-mudas.csv

```
 Para facilitar seguem arquivos da função e help
```

funcao_mayara_final.r

help_funcao_mayara_final.r