Claire Pauline Röpke Ferrando - apaixonada por roedores!

Sou aluna de doutorado no PPG Ecologia e Conservação de Recursos Naturais da Universidade Federal de Uberlândia - MG orientada pela professora Natália O. Leiner. Minha linha de pesquisa atual envolve estudos sobre estrutura social de roedores através de dados de área de vida, parentesco e estrutura populacional. https://lemaufu.wixsite.com/labecomammals

exec

————— meus exercícios —————

exercicio_1_respondendo_topicos_1_a_3.r aula4_exercicios_so_1_e_3.r
aula5_exercicios_construcaograficos.r
notar_nao_leu_aula6_exercicios_2_e_3_anova_unha_e_facil.raula7b_exercicios.r
aula7_exercicio_2_modelo_mais_simples.r aula8_exercicios.r aula9_exercicio_2..r

————— Trabalho Final —————

PROPOSTA A - Quantas amostras são necessárias?

Contextualização:

Ecólogos e cientistas em geral trabalham com réplicas de amostras a fim de melhor representar a população ou a comunidade geral. Dentro desse contexto, os pesquisadores sempre procuram por obter os melhores resultados (i.e. mais representativos da realidade) em um menor tempo, custo e esforço. Poucas e muitas amostras podem levar a resultados enviesados e a custos desnecessários, respectivamente (Eckblad 1991). Para se detectar diferenças entre grupos amostrais depende da variabilidade do parâmetro e o tamanho da diferença que se pretende detectar (Gotelli & Ellison 2004). Segundo Gotelli & Ellison (2004) 10 amostras seria o mínimo para fins de comparação entre categorias ou tratamentos (Rule of 10), porém isso depende, por exemplo, dos organismos de estudo, tipo de amostragem e objetivo do estudo. Com base nessa problemática, Eckblad (1991) apresentou uma forma de estimar o número de amostras necessárias para estimar a média populacional em uma porcentagem de acurácia da média real.

A função

Com base no trabalho de Eckblad (1991), a função `n.samples()` estimará o número de amostras necessárias para uma determinada porcentagem de interesse de acurácia, assim como a acurácia em si da média do tamanho amostral. Essas estimativas são baseadas no valor de t com nível de significância (0,05) correspondente ao grau de liberdade (i.e. número de amostras – 1), média, variância e erro padrão da média das amostras e tamanho da amostra (i.e. número de amostras).

Ainda, os dados devem seguir uma distribuição normal. Os cálculos das estimativas propostos por Eckblad (1991) estão representados pelas duas fórmulas a seguir (imagens retiradas de Eckblad (1991)):

$$\text{sample size} \cong \frac{(\text{t-value})^2\ (\text{sample variance})}{(\text{accuracy} \times \text{sample mean})^2}$$

$$\text{accuracy} \cong \frac{(\text{coefficient of variation})(\text{t-value})}{\sqrt{\text{sample size}}}$$

Além das estimativas descritas acima, a função também gerará um gráfico de relação entre a média acurada e o número de amostras necessárias para um ou mais grupos como mostra a figura abaixo (imagens retiradas de Eckblad (1991))

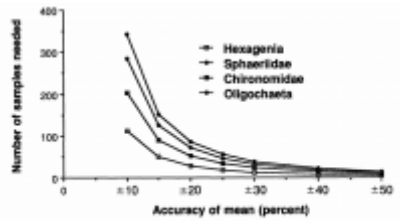

Figure 2. Number of samples required to sample four benthic taxa within a certain percent accuracy of the mean.

Nome da função

`n.samples()`

Argumentos da função - Entrada

Argumento1:x, objeto da classe vetor ou data.frame com observações numéricas.

Argumento2: `accuracy`, porcentagem de acurácia da média do tamanho amostral (de 10 a 50%).

Argumento3: `tvalue` , valor de t correspondente ao número de amostras e nível de significância 0,05.

Argumento4: `graphic`, TRUE ou FALSE (default TRUE).

Pseudo-código:

1.Verifica que existem valores faltantes nos dados.

se sim:

stop(“There are missing values in your data”)

2.Premissa1: se objeto é diferente de vetor ou data frame

stop (“your object is not a vector or even a data.frame”)

3.Se, o objeto for um vetor:

a.Premissa3.1: vetor é de classe numérica

se não não for:

stop (“your vector is not numeric”)

b.Premissa3.2: número de amostras ≥ 10

se não for:

stop (“your object has less than 10 observations”)

c.Premissa3.3: distribuição normal do vetor

d.Cria objeto `norm` com o teste da normalidade dos dados (shapiro.test())

e.Se P menor do que 0,05.

stop(“your data doesn’t follow a normal distribution”)

f.Cria objeto `med` com a média da amostra (com mean())

g.Cria objeto `vari` com a variância da amostra (com var())

h.Cria objeto `er.pad` com o erro padrão da média da amostra

i.Cria objeto n com o tamanho da amostra (número de amostras) (com length())

j.Cria objeto `coef.var` com o coeficiente de variação da amostra

k.Cria objeto `sample.size` com a estimativa do número de amostras necessário com base na porcentagem de acurácia e valor de t informados nos argumentos `accuracy` e `tvalue`, respectivamente.

l.Cria objeto `accur` com a estimativa da acurácia da média da amostra com base no valor de t informado no argumento `tvalue` (com apply())

m.return (list(`sample.size`, accuracy=`accur`))

n.Se argumento `graphic` != TRUE

stop (“the end”)

o.Cria objeto `x.axis` com as porcentagens de acurácia de 10 a 50%.

p.Cria objeto `y.axis` com os números de amostras necessárias para as diferentes porcentagens de acurácia da média da variável.

q.Cria objeto `sample.plot` com o plot da relação entre número de amostras para diferentes acurácias da média da amostra (com plot())

r.Return(`sample.plot`)

4.Se o objeto for um data.frame

a.Premissa4.1:data.frame somente com colunas numéricas

install.packages(`dplyr`)

b.seleciona somente as colunas com valores numéricos do data.frame com a função select_if

c.Premissa4.2: se número de amostras < 10 por variável (colunas)

stop ("your object has less than 10 observations")

d.Premissa 4.3: distribuição normal das variáveis

e.Cria objeto `norm` com o teste da normalidade das variáveis (colunas) (com apply())

f.Se P < 0,05.

g.stop("there are variables that don't fit a normal distribution")

h.Cria objeto `med` com a média dos dados de cada variável do data.frame (com apply())

i.Cria objeto `vari` com a variância dos dados de cada variável do data.frame (com apply())

j.Cria objeto `er.pad` com o erro padrão da média da amostra de cada variável do data.frame (com apply())

k.Cria objeto ncom o tamanho da amostra de cada variável do data.frame (número de amostras) (com apply())

l.Cria objeto `coef.var` com o coeficiente de variação da amostra de cada variável do data.frame (com apply())

m.Cria objeto `samples.size` com o número de amostras necessário com base na porcentagem de acurácia e valor de t informados nos argumentos `accuracy` e `tvalue`, respectivamente (com apply()).

n.Cria objeto `accur` com a estimativa da porcentagem de acurácia da média das amostras de cada variável do data.frame com base no valor de t informado no argumento `tvalue` (com apply())

o.return (list(`sample.size`, accuracy=`accur`))

p.Se argumento graphic != TRUE

stop (" the end")

q.Cria objeto `x.axis` com as porcentagens de acurácia de 10 a 50% .

r.Cria objeto `y.axis` com os números de amostras necessárias para diferentes porcentagens de acurácia da média.

s.Cria objeto `samples.plot` com o plot relação entre número de amostras para diferentes acurácias da média das amostras de cada variável (com plot())

t.Return(`samples.plot`)

Saída: Lista com as estimativas de número de amostras necessárias para uma determinada porcentagem de acurácia da média de interesse e de acurácia da média do tamanho amostral. Ainda, um gráfico de relação entre número de amostras necessárias e médias acuradas para uma ou mais variáveis.

> Julia Barreto
>
> Oi Claire, tudo bem? Gostei da ideia da função, o objetivo está muito claro mas acho que ainda precisamos trabalhar em detalhar melhor a proposta. Tem algumas informações que faltaram mas que você parece ter em mente. Pormenores que, assim que começar a montar, você vai notar e terá que decidir como seguir. Por exemplo, os argumentos na função, pensar o que fazer se não forem preenchidos (função para? Ou haverá um valor default?). Pensa também que o primeiro passo da função seria conferir se o objeto de entrada atende aos requisitos. Então, não faz sentido conferir se tem NAs antes de ver se é vetor ou data frame, considere inverter essas estapas.
>
> Vale tentar ensaiar suas habilidades, procure maneiras de tornar a tarefa mais instigante, por exemplo com uso de controle de fluxo vistos nas últimas aulas. Uma outra ideia que complementaria bastante a função, seria resultar em mais de uma estimativa como sugestões de outras maior quantidade de amostras para aumentar a acurácia. Compreende o que quero dizer? Isso seria particularmente interessante para casos em que um número um pouco maior de amostras já aumentasse bastante a acurácia. Seria uma saída interessante sobretudo quando o usuário não apontasse a desejada (deixando o argumento sem preencher, optando pelo default). Me escreve se quiser conversar sobre essas ou outras ideias que venha a ter, e qualquer dúvida também, claro!
> Estou à disposição Julia Barreto

> Claire Ferrando
>
> Olá Julia! Muito obrigada pelas dicas. Ainda estou incerta sobre como aplicar as tuas sugestões. Mas vou começara mexer nisso e as dúvidas vão surgir e entro em contato contigo. Com certeza terei dúvidas! Obrigada!

> Julia Barreto
>
> Massa! Boa sorte e qualquer coisa vc já sabe aonde me achar :)

Fontes:

Eckblad, J. W. (1991) How Many Samples Should Be Taken? BioScience 41, 346-348.

Gotelli, N. J. & Ellison, A. M. (2004) A Primer of Ecological Statistics. Sunderland, MA: Sinauer & Associates, 2004

McCabe, D. J. (2011) Sampling Biological Communities. Nature Education Knowledge 3(10):63

PROPOSTA B - Uma função para várias

Contextualização:

O programa R é estruturado por diferentes funções para diferentes finalidades. Elas resumem várias etapas/passos de análises, facilitando a vida do usuário.

A função

Seguindo nessa idéia, a função fast.resume() tem como finalidade, através de um único argumento (o próprio objeto), retornar as características básicas do objeto de dados independentemente da sua classe.

Nome da função:

fast.resume (x)

Argumento da função - Entrada

Argumento1:x,objeto de qualquer classe/estrutura.

Pseudo-código:

1.Se o objeto for um vetor, isto é “numeric” ou “factor” ou “character” ou “integer” ou “logic”

1.1. Mensagem (“your object is of class vector”)

a.Se os dados forem numéricos (i.e. “numeric”)

i.mensagem (“your object is of class numeric”)

ii.objeto `n.na` com número de observações ausentes (i.e. NAs) (com sum(is.na()))

iii.cria objeto `su` com o sumário do objeto (com summary())

iv.cria objeto `le` com o número de elementos no vetor (com length ())

v.return(list(missing_values=`n.na`, summary=`su`, length=`le`))

b.Se os dados forem fatores (i.e. “factor”)

i.mensagem (“your object is of class factor”)

ii.cria objeto `n.na` com número de observações ausentes (i.e. NAs) (com sum(is.na()))

iii.cria um objeto `tab` com o número de observações para cada nível do fator (com table())

iv.cria objeto `le` com o número de elementos no vetor (com length ())

v.return(list(missing_values=`n.na`, table=`tab`, length=`le`))

c.Se os dados forem caracteres (i.e. “character”)

i.mensagem (“your object is of class character”)

ii.cria objeto `n.na` com o número de observações ausentes (i.e. NAs) (com sum(is.na()))

iii.cria objeto `tab` com número de vezes de um mesmo caracter

iv.cria objeto `le` com o número de caracteres diferentes no vetor (com length())

v.return(list(missing_values=`n.na`, table=`tab`, length=`le`))

d.Se os dados forem do tipo “integer”

i.mensagem (“your object is of class integer”)

ii.cria objeto `n.na` com o número de observações ausentes (i.e. NAs) (com sum (is.na()))

iii.cria objeto `su` com o sumário do objeto (com summary())

iv.cria objeto `le` com o número de elementos no vetor (com length ())

v.return(list(missing_values=`n.na`, summary=`su`, length=`le`))

e.Se os dados forem do tipo “logical”

i.mensagem (“your object is of class logical”)

ii.cria objeto `n.na` com o número de observações ausentes (i.e. NAs) (com sum (is.na()))

iii.cria objeto `su` com o sumário do objeto (com summary())

iv.cria objeto `le` com o número de elementos no vetor (com length ())

v.return(list(missing_values=`n.na`, summary=`su`,length=`le`))

2.Se o objeto for um data frame

a.mensagem (“your object is of class data.frame”)

b.cria objeto `n.na` com o número de observações ausentes (i.e. NAs) (com sum (is.na()))

c.cria objeto `st` com a estrutura do objeto (com str())

d.cria objeto `col` com o número de colunas no objeto (com ncol())

e.cria objeto `row` com o número de linhas no objeto (com nrow())

f.cria objeto he com as três primeiras linhas do data.frame (com head()) (selecionando só as três primeiras linhas).

g.cria objeto `nam` com os nomes das variáveis (com names())

h.cria objeto `su` com o sumário do objeto (com summary())

i.return(list(missing_values=n.na, str=st, columns=col, rows=row, head=he, names=nam, summary=su)

3.Se objeto for um “matrix”

a.mensagem (“your object has a matrix structure”)

b.cria objeto mod com a natureza dos dados que a compõem

c.cria objeto n.na com o número de observações ausentes (i.e. NAs) (com sum (is.na()))

d.cria objeto st com a estrutura do objeto (com str())

e.cria objeto n.col com o número de colunas (com ncol())

f.cria objeto n.row com o número de linhas (com nrows())

g.cria objeto col.name com os nomes das colunas (com colnames())

h.cria objeto row.name com os nomes das linhas (com rownames())

i.return(list(mode=mod, missing_values=n.na, str=st, number_columns=n.col, number_rows=n.row, columns_names= col.name, rows_names=row.name)

4.Se objeto for um “table”

a.mensagem (“your object is of class table”)

b.cria objeto n.na com o número de observações ausentes (i.e. NAs)(com sum (is.na()))

c.cria objeto st com a estrutura do objeto (com str())

d.cria objeto n.col com o número de colunas (com ncol())

e.cria objeto n.row com o número de linhas (com nrows())

f.cria objeto col.name com os nomes das colunas (com colnames())

g.cria objeto row.name com os nomes das linhas (com rownames())

h.cria objeto he com as três primeiras linhas da tabela (com head()) (selecionando só as três primeiras linhas)

i.cria objeto nam com os nomes das variáveis (com names())

j.cria objeto su com o sumário do objeto (com summary())

k.return(list(missing_values=n.na, str=st, number_columns=n.col, number_rows=n.row, columns_names= col.name, rows_names=row.name, head=he, names=nam, summary=su)

5.Se objeto for um “array”

a.Mensagem (“your object has an array structure”)

b.cria objeto `n.na` com o número de observações ausentes (i.e. NAs) (com sum (is.na()))

c.cria objeto `dimens` com as dimensões do objeto (com dim())

d.cria objeto `dim.name` com os nomes das dimensões (com dimnames())

e.cria objeto `mod` com a natureza dos dados que a compõem (com mode())

f.cria objeto `st` com a estrutura do objeto (com str())

g.return(list(missing_values=`n.na`, dimentions=`dimens`, dimentions_name=`dim.name`, mode=`mod`, str=`st`)

6.Se objeto for um “list”

a.Executa o próprio objeto x e expõe o que tem na lista

b.Cria um objeto `list_name` com os nomes dos objetos presentes na lista (com names())

c.Return(list(in_the_list=x, objects_names=`list_name`))

Saída: uma lista do resumo do objeto.

Julia Barreto

Oi de novo! Sobre essa segunda proposta, está bastante simples além de pouco detalhada. Eu acredito que você poderia se desafiar mais do que apenas entregar resumo e classe do objeto de entrada. Sei que já optamos pela proposta A que parece mais elaborada mas, certamente, você poderia pensar em como complementar essa ideia B para que contivesse alguma novidade e maior utilidade. Julia Barreto

Claire Ferrando

Olá! Eu concordo que essa proposta é mais simples, mas fiz essa proposta por acreditar ser útil. Se depois de finalizar a proposta A eu ainda tiver tempo, vou repensar nela. Obrigada! Claire

---

Código da PROPOSTA A - Quantas amostras são necessárias?

```
n.samples=function(x, accuracy, tvalue){                    #função
n.samples.
if(missing(x)){                                  #testa se tem objeto
com dados.
    stop("There is no object")                          #caso o argumento
'x' não foi atendido, a função por default pára e envia uma mensagem ao
usuário da falta de dados para executar a função.
    }
if(missing(tvalue)){                             #testa se o valor de t
foi informado.
```

```
    stop("You need to inform the tvalue")                    #caso o
argumento 'tvalue' não foi atendido, a função, por default pára e envia uma
mensagem ao usuário dizendo que sem essa informação, a análise não
    }                                                        #pode ser feita já que
é uma informação relacionada ao objeto dos dados.

#ARGUMENTO 'accuracy' não inserido

if(class(x)=="numeric"){                                  #testando se o objeto
é um vetor, classe 'numeric' pois quando a função class é usada para um
vetor, ele dá a natureza dos dados. Dessa forma, aqui também já
                                                         #se testa se os valores
são numéricos, que é uma das premissas da função.
    if(missing(accuracy)){                                 #caso o usuário
não inserir o argumento 'accuracy' a função retornará somente a estimativa
da acurácia da média da amostra, e um data frame com o número de
    {                                                    #amostras necessárias
para diferentes acurácias da média da população com base do objeto com os
dados.
    message("By default, estimate of sample size                #mensagem:
caso o argumento 'accuracy' não foi inserido, a função, por default informa
todos os valores de
    needed will be given from 10 to 50% of                       #acurácia
para os dados de 10 a 50% a cada 5%.
    accuracy every 5%")
        }
        if(any(is.na(x))==TRUE){                           #verificando se no
vetor tem valores faltantes.
        stop("There are missing values in your data")            #se houver
valores faltantes no vetor, por default a função não será executada, senão
as estimativas de acurácia da média da amostra e o número de amostras serão
comprometidas
        }                                                  #i.e. estarão erradas.
        if(length(x)<10){                                    #seguindo a regra
dos 10 de Gotelli & Ellison (2004)como número mínimo para o cálculo de
estimativas mais acuradas, nessa linha verifica-se se existem 10 elemtos no
objeto.
        stop ("Your object has less than 10 samples")             #caso não
haja pelo menos 10 elementos no vetor, a função não será executada.
        }
        test.norm=shapiro.test(x)                           #testando a
normalidade dos dados, pois uma das premissas para o cálculo das estimativas
é de que os dados sigam uma distribuição normal.
        if(test.norm$p.value <= 0.05){                       #se o valor do
teste for menor do que 0,05, quer dizer que a distribuição dos dados não
seguem uma distribuição normal.
        stop ("Your vector doesn't follow a normal distribution")     #não
seguindo uma distribuição normal, a função pára e envia a mensagem ao
usuário informando que os seus dados não seguem uma distribuição normal como
é a premissa para o uso da função.
```

```
        }
        vari=var(x)                                    #criando objeto com a
variância da amostra.
        se.mean=sqrt(vari)                             #criando objeto com o
erro padrão da média.
        n=length(x)                                    #criando objeto com o
tamanho da amostra (número de amostras).
        coef.var=sqrt(se.mean/mean(x))                    #criando objeto
com o coeficiente de variação da amostra.
        accur=(coef.var*tvalue)/sqrt(n)                    #criando objeto
com a estimativa da acurácia da média da amostra.
        accuracies=seq(from=0.05, to=0.5, by=0.05)            #criando
objeto com os valores de acurácia de 10 a 15% a cada 5%.
        n.samples=(((tvalue)^2)*vari)/(accuracies*(mean(x)))^2
#criando objeto com os valores de número de amostras correspondente às
acurácias da média da população.
        return (list(accuracy=accur,data.frame(accuracies,n.samples)))
#retorna ao usuário uma lista com a estimativa do número de amostras
necessárias dentro da porcentagem de acurácia desejada, a estimativa da
acurácia da

        #grafico

        accuracies=seq(from=0.05, to=0.5, by=0.05)            #criando o
objeto com os valores do eixo x do gráfico (acurácias de 10 a 50% a cada
5%).
        n.samples=(((tvalue)^2)*vari)/(accuracies*(mean(x)))^2    #criando o
objeto com os valores de número de amostras correspondente a acurácia (eixo
y)
        par(mar=c(5, 5, 2, 2))                         #estabelecendo as
margens da imagem.
        plot(accuracies,n.samples, cex.axis= 1.5, las=1,         #criando um
plot com a relação entre as acurácias e a estimativa dos números de amostras
necessárias correspondentes.
        family="serif", tcl=0.3,col="black", xlab="", ylab="",
        pch=19, cex=1.5,bty="l")
        lines(accuracies[order(accuracies)],                 #adicionando
linhas ligando os pontos do gráfico para que o usuário possa ter uma ideia
da estimativa do número de amostras necessárias para as acurácias
            n.samples[order(accuracies)], xlim=range(accuracies),
#entre 10 a 50%.
        ylim=range(n.samples), lwd = 2)
        mtext("Accuracy of mean", line=3,side= 1,            #adicionando
título ao eixo x do gráfico.
        cex= 1.7, family="serif")
        mtext("Number of samples needed",line=3,side=2,cex=1.7,
#adicionando título ao eixo y do gráfico.
            family="serif")

            }
        }else{                                      #se o usuário inseriu o
```

```
argumento 'accuracy' com o uso de um objeto do tipo vetor, a função seguirá
a partir daqui.

#ARGUMENTO 'accuracy' inserido

    if(any(is.na(x))==TRUE){                              #verificando se no
vetor tem valores faltantes.
    stop("There are missing values in your data")                #se houver
valores faltantes no vetor, por default a função não será executada, senão,
as estimativas de acurácia e número de amostras serão comprometidas.
    }                                                   #i.e. gerando
estimativas erradas.
    if(length(x)<10){                                    #seguindo a regra
dos 10 de Gotelli & Ellison (2004)como número mínimo para o cálculo de
estimativas mais acuradas, nessa linha verifica-se se existem 10 elemtos no
objeto.
    stop ("Your object has less than 10 samples")                #caso não
haja pelo menos 10 elementos no objeto, a função não será executada.
    }
    test.norm=shapiro.test(x)                             #testando a
normalidade dos dados, pois uma das premissas para o cálculo das estimativas
é de que os dados sigam uma distribuição normal.
    if(test.norm$p.value <= 0.05){                        #se o valor do
teste for menor do que 0,05, quer dizer que a distribuição dos dados não
seguem uma distribuição normal.
    stop ("Your vector doesn't follow a normal distribution")        #não
seguindo uma distribuição normal, a função pára e envia a mensagem ao
usuário informando que os seus dados não seguem uma distribuição normal como
é a premissa para o uso da função.
    }
    vari=var(x)                                          #criando objeto com a
variância da amostra.
    se.mean=sqrt(vari)                                   #criando objeto com o
erro padrão da média.
    n=length(x)                                          #criando objeto com o
tamanho da amostra (número de amostras).
    coef.var=sqrt(se.mean/mean(x))                       #criando objeto
com o coeficiente de variação da amostra.
    sample.size=(((tvalue)^2)*vari)/(accuracy*(mean(x)))^2       #criando
objeto com a estimativa do número de amostras necessário dentro da
porcentagem de acurácia informada no argumento accuracy.
    accur=(coef.var*tvalue)/sqrt(n)                      #criando objeto
com a estimativa da acurácia da média da amostra.
    accuracies=seq(from=0.05, to=0.5, by=0.05)               #criando
objeto com os valores de acurácia de 10 a 15% a cada 5%.
    n.samples=(((tvalue)^2)*vari)/(accuracies*(mean(x)))^2       #criando
objeto com os valores de número de amostras correspondente às acurácias da
média da população.
    return (list(sample.size, accuracy=accur,                #retorna ao
usuário uma lista com a estimativa do número de amostras necessárias dentro
```

```
da porcentagem de acurácia desejada, a estimativa da acurácia da
        data.frame(accuracies,n.samples)))                          #média da
amostra, e ainda um data frame com o número de amostras necessárias com base
no objeto de dados para diferentes acurácias (de 10 a 50% a cada 5%).

    #gráfico

    accuracies=seq(from=0.05, to=0.5, by=0.05)                  #criando o
objeto com os valores do eixo x do gráfico (acurácias de 10 a 50% a cada
5%).
    n.samples=(((tvalue)^2)*vari)/(accuracies*(mean(x)))^2        #criando o
objeto com os valores de número de amostras correspondente a acurácia (eixo
y)
    par(mar=c(5, 5, 2, 2))                                      #estabelecendo as
margens da imagem.
    plot(accuracies,n.samples, cex.axis= 1.5, las=1,              #criando um
plot com a relação entre as acurácias e a estimativa dos números de amostras
necessárias correspondentes.
        family="serif", tcl=0.3,col="black", xlab="", ylab="",
        pch=19, cex=1.5,bty="l")
    lines(accuracies[order(accuracies)],                        #adicionando
linhas ligando os pontos do gráfico para que o usuário possa ter uma ideia
da estimativa do número de amostras necessárias para as acurácias
    n.samples[order(accuracies)], xlim=range(accuracies),             #entre
10 a 50%.
    ylim=range(n.samples), lwd = 2)
    mtext("Accuracy of mean", line=3,side= 1,                   #adicionando
título ao eixo x do gráfico.
        cex= 1.7, family="serif")
    mtext("Number of samples needed",line=3,side=2,cex=1.7,
#adicionando título ao eixo y do gráfico.
        family="serif")
        }
    }

n.samples()
```

Help da PROPOSTA A - Quantas amostras são necessárias?

```
modelo                  package:unknown                  R Documentation

ESTIMATIVAS DE NÚMERO DE AMOSTRAS NECESSÁRIAS E ACURÁCIA DA MÉDIA DA AMOSTRA

Description:

    Função para estimar o número de amostras necessárias para uma
determinada porcentagem de interesse de acurácia da média,
    assim como a acurácia da média do tamanho amostral. Além disso, ela
```

```
oferece um data frame e um gráfico com
```