Gabriel Massami Kayano

Mestrando em Ecologia no Instituto de Biociências da USP

Título da dissertação: Diferenças arquiteturais entre gramíneas africanas e nativas e implicações no padrão de uso da luz e no potencial invasivo

Orientação: Prof. Dr. Sérgio Tadeu Meirelles

Exercícios

exec

Proposta de Trabalho Final

Proposta I

Função para dividir gastos de viagens ou eventos para um grupo de pessoas

A função irá computar os gastos individuais de cada pessoa feitos durante um período e realizar a divisão equitativa desses gastos, sumarizando ao final o valor que cada pessoa deve pagar ou receber. Esse cálculo leva em conta as pessoas que participam ou não de cada evento de gastos monetários, computando o valor que cada pessoa deve pagar referente apenas ao que consumiu.

O objeto de entrada da função deve ser um data frame contendo os valores gastos de cada pessoa (linhas) por evento de compra (colunas). Se a pessoa usufruiu dessa compra e ajudou a pagar o valor deve ser aquele que foi pago no ato da compra. Caso essa pessoa tenha usufruído da compra, mas não pagou nada, o valor deverá ser zero, e se a pessoa não usufruiu do que foi comprado, coloca-se NA. Dessa forma, é possível calcular quanto cada participante gastou, e a diferença deste valor com a média de gasto de um determinado evento. Nas linhas do data.frame são discriminadas pelas pessoas que integram o grupo pelo qual se dividirão os gastos (ex: José, Rebeca, Bruno, Marcelo, Amanda etc). Nas colunas, os valores estarão discriminados por cada evento de gasto durante o período (ex: compra 1, compra 2, supermercado, gasolina, cerveja, aluguel, faxina, etc).

Os argumentos que devem ser fornecidos são: a) número de pessoas do grupo e b) número de eventos de compras.

A função devolverá uma impressão na tela, informando os valores que determinadas pessoas do grupo deverão pagar a outras, caso os gastos tenham sido desiguais. Esses valores são calculados para que a transferência entre pessoas seja a mais eficiente possível, evitando que pessoas tenham que pagar pequenas quantias a várias outras pessoas.

Proposta II

A função construirá um modelo tridimensional de um ramo vegetal utilizando dados sobre orientação espacial e angulação das folhas e pecíolos, área foliar, comprimento dos pecíolos e altura dos nós. O ramo deve ser ereto, e conter somente pecíolos e folhas, sem ramificações. Os dados devem ser fornecidos em um data frame com linhas representando cada folha medida, e as colunas contendo os dados sobre sua morfologia, de maneira que seja possivel caracterizar a estrutura tridimensional deste ramo.

A função conterá argumentos que permitam especificar qual o formato das folhas (eliptico, filiforme etc.) e o calculo adequado para cada opção. Ela retornará um gráfico de três eixos com as representações das estruturas aéreas do ramo em três dimensões, gráficos circulares contendo a distribuição angular e distribuição das orientações das folhas, e uma impressão na tela contendo o índice de área foliar.

COMENTÁRIOS PROPOSTAS MELINA

em 28/abril/2016

As duas propostas são razoáveis, mas achei a proposta A fácil e simples demais. A proposta B parece mais desafiadora e poderia ser melhor desenvolvida. Quais seriam os dados de morfologia úteis? Sugiro ficar com a B, e que você pense em todas as possibilidades e limitações da função de acordo com os diferentes tipos de plantas que poderiam ser 'desenhados' por ela.

Propostas Reformuladas

Em 04/05/2016

Melina, após analisar cuidadosamente a proposta B para reformular seu conteúdo concluí que para tornar viáveis as proposições, ao meu ver, eu teria que me valer de um conhecimento matemático que não possuo. Eu poderia estudar essa matéria para realizar aquilo que me propus a fazer. No entanto, isso significaria que o maior esforço na elaboração da função seria no sentido de me apropriar de fundamentos matemáticos e não de exercitar o que aprendi em programação em R. Além disso, o tempo disponível para isso provavelmente não seria suficiente. Portanto, resolvi esse problema da seguinte maneira: reelaborei as duas propostas, de modo a tornar a proposta A um pouco mais complexa, e simplifiquei as tarefas da proposta B, de modo que elas se tornem factíveis. À proposta A incorporei alguns elementos como subcategorias de gastos, conversão de moedas utilizando informação online e gráficos informativos. A proposta B teve que ser reduzida em termos de aplicação, pois encontrei sérias dificuldades em construir planos geométricos localizados em três dimensões sem precisar informar todos as coordenadas de todos os pontos necessários, o que tornaria o objeto de input uma matriz consideravelmente grande e de pouca praticidade. Dessa maneira, peço que reconsidere as duas proposições na reavaliação, e me indique qual delas eu devo executar, assim como as suas sugestões. Valeu!

Proposta A:

Função para dividir gastos de viagens ou eventos para um grupo de pessoas

A função irá computar os gastos individuais de cada pessoa feitos durante um período e realizar a divisão equitativa desses gastos, sumarizando ao final o valor que cada pessoa deve pagar ou receber. Esse cálculo leva em conta as pessoas que participam ou não de cada evento de gastos monetários, computando o valor que cada pessoa deve pagar referente apenas ao que consumiu.

Outras funcionalidades acionáveis por meio dos argumentos: a função também permite que se coloquem gastos em diferentes moedas, e que se especifique em qual moeda a divisão deve ser feita. As conversões monetárias serão feitas importando valores de câmbio de um site de referencia na internet. Outra opção é a de plotar ou não gráficos como o de gastos cumulativos por data com todos os participantes, um gráfico de setores com os valores totais de cada categoria de gasto (ex: comida, transporte, entretenimento, saúde, etc), e um gráfico de barras com o total de gastos de cada um e uma linha com a média.

O objeto de entrada da função deve ser um array de 5 dimensões contendo os valores gastos de cada pessoa (dim1) por evento de compra (dim2), a moeda utilizada por cada pessoa (dim3), a categoria da compra – o usuário define (dim4) e a data (dim5). Se a pessoa usufruiu dessa compra e ajudou a pagar o valor deve ser aquele que foi pago no ato da compra. Caso essa pessoa tenha usufruído da compra, mas não pagou nada, o valor deverá ser zero, e se a pessoa não usufruiu do que foi comprado, coloca-se NA.

Os argumentos que devem ser fornecidos são: a) o objeto array sobre o qual se aplicará a função; b) número de participantes; c) número de gastos; d) a moeda utilizada (apenas reais, dólares ou euros); e por fim um argumento lógico para plotar ou não os gráficos.

A função devolverá uma impressão na tela, informando os valores que determinadas pessoas do grupo deverão pagar a outras, caso os gastos tenham sido desiguais. Se o argumento de plotagem dos gráficos estiver assinalado como TRUE, a função também devolverá os gráficos supracitados em uma nova janela.

Proposta B:

A função deverá produzir um modelo simples tridimensional contendo a distribuição das folhas de uma planta com folhas filiformes em arranjo de touceira ou de ramo ereto, principalmente voltada para ser usada com gramíneas. Além disso, a função também fornece gráficos circulares de distribuição angular e das orientações das áreas foliares.

Limitações: a representação gráfica de outros formatos de folha requer a construção de um plano em um meio tridimensional o que requer ou o uso de muitos pontos (e um input bastante elaborado) ou um conhecimento matemático que não possuo. Portanto, a representação será feita através de segmentos de retas, que pretendem representar o arranjo das folhas. O modelo também despreza as curvaturas dos órgãos vegetais, assumindo-os como segmentos e retas, e também não permite a representação de ramos vegetais com ramificações (os pecíolos devem estar ligados ao ramo axial)

Argumentos: a) objeto de input; b) forma da folha para o cálculo da área (filiforme, laminar ou elíptica); c) escolha da distribuição das orientações: se elas serão fornecidas no input por quatro pontos (N, S, L e O), oito pontos (N, NE, NW, E, W, SE, SW) ou por graus (sendo 0º o Norte); d) se a

representação é de um ramo ou uma touceira.

Se a planta escolhida é um ramo, o input da função deve ser um data frame com as folhas medidas representadas nas linhas, e nas colunas os dados morfológicos para cada folha: distância internodal (do nó abaixo ao nó onde a folha se encontra), comprimento do pecíolo, ângulo do pecíolo, orientação do contínuo pecíolo-face adaxial da folha (admitindo as orientações das folhas e pecíolos são as mesmas), maior largura da folha, comprimento da folha, ângulo da superfície foliar em relação ao zênite.

Se a planta escolhida é uma touceira, o input da função deve ser um data frame com as folhas medidas representadas nas linhas, e nas colunas os dados na seguinte ordem: maior largura da folha, comprimento da folha, orientação da face adaxial da folha, inclinação da folha em relação ao zênite.

A função por fim devolverá o modelo tridimensional juntamente com os gráficos de distribuição angular e de orientações.

COMENTÁRIOS MELINA

em 6/5/2016

Oi gabriel, a proposta A ficou mais legal agora! Uma dúvida é que você coloca o objeto de entrada como um array, não consigo imaginar um array com as informações que você dá. E mesmo se desse, acho que uma pessoa criar um array para colocar os dados na função iria ser tão complicado que inviabilizaria o uso da função. Por acaso vc queria dizer uma lista? Também não indico usar uma lista para entrada de objetos, talvez fosse mais fácil dividir os objetos de entradas e "chamá-los" em diferentes argumentos. O que eu consigo ver dos objetos de entrada são:

- um data frame com as pessoas nas linhas e os eventos de compra nas colunas.
- um argumento na função para que a pessoa indique a moeda para conversão (e não um objeto de entrada)
- para a dim 4 (a categoria da compra - o usuário define) e 5 (data) eu não sei se entendi muito bem, mas acho que você quer relacionar cada evento de compra (os nomes das colunas do dataframe) com a sua categoria de compra e data, é isso? Se for isso, pensei em 3 maneiras dessa informação ser incluída:

1. você colocar no data frame original linhas acima do evento de compra (ou seja o cabeçalho) que indicariam a data e a categoria de compra. Essa forma seria boa de ver no excel, mas no R isso poderia ficar estranho e talvez seja difícil fazer o R ler um objeto com 3 cabeçalhos. Enfim, avalie.
2. você poderia construir um outro data frame que contem nas linhas o evento de compra e duas colunas indicando a data e a categoria da compra. assim você poderia associar essas linhas (que teriam os mesmos nomes das colunas do primeiro data frame) às colunas do data frame dos gastos.
3. ou você poderia inverter a ordem de linhas e colunas no data frame dos gastos, ou seja colocar as pessoas como colunas e os eventos de compra nas linhas, e daí incluir duas colunas, uma contendo a categoria do gasto e outra contendo a data.

Sugiro pensar melhor nesses dados de entrada, e ver qual fica melhor e mais fácil de trabalhar na função. Se você for usar dois data frames de objeto de input, esses dois data frames deveria ser chamados em argumentos diferentes para não confundir. Se ficar muito complicado, pode simplificar

a função nessa parte.

Depois de pensar nos dados de entrada reveja os argumentos da função. É necessário informar o número de participantes ou gastos? Essa informação pode ser acessada a partir do data frame dos dados, certo?

A proposta B também está razoável. Me pareceu que você se inclina mais a fazer a proposta A, principalmente pela dificuldade que você já adianta que vai ter na B. Então deixo pra você escolher qual e mandar ver!!

Boa função!

proposta A - dividir.gastos

Arquivos

Arquivo da função: dividir.gastos.r

Página de ajuda: help.txt

Arquivos usados para os exemplos:

exemplo1.csv, exemplo2.csv, exemplo3.csv

Página de Ajuda

```
dividir.gastos                package:unknown                R Documentation

Divide igualmente gastos compartilhados durante um período por duas ou mais
pessoas.

Description:

     A função é uma ferramenta para resolver divisões de gastos entre grupos
de pessoas que compartilham eventuais gastos durante um período, como
viagens, festas, churrascos e eventos em geral. Utilizando um data.frame
contendo os gastos efetuados por cada pessoa, a categoria do gasto e a data,
calcula as transações que devem ser feitas para que as contas possam ser
equilibradas após o término do período. Como opção, é possível plotar
gráficos (gastos cumulativos, gastos por categoria e gastos por pessoa) e
realizar conversões de moedas(real, dolar e euro).

Usage:

     dividir.gastos=function(x, moeda.entrada="real", moeda.saida="real",
graficos=TRUE)

Arguments:
```

```
       x        data.frame contendo os eventos de gastos nas linhas,
contendo
                os valores pagos pelas pessoas discriminadas nas colunas.
Nas
                primeiras colunas devem constar as pessoas e seus nomes no
                cabeçalho. A penúltima coluna deve se chamar "categoria" e
                deve conter as categorias de cada gasto (character). A
última
                coluna deve se chamar "data" e conter as datas dos
respectivos
                gastos, em formato dd/mm/yy.

 moeda.entrada  a moeda correspondente aos valores apresentados em x. Pode
ser
                "real"(default), "dolar" ou "euro".

  moeda.saida   a moeda que se deseja na devolução dos valores pela função.
                Pode ser "real" (default), "dolar" ou "euro".

   graficos argumento lógico. Se TRUE(default), habilita a plotagem dos
                gráficos da função. Se FALSE, desabilita a plotagem.



Details:

     Os nomes das pessoas devem estar explicitados no data.frame como
cabeçalho, assim como
     a penúltima e última colunas devem ter como cabeçalho "categoria" e
"data", respectivamente
     (e conter tais dados, obrigatoriamente).
     Regras para o preenchimento dos valores no data.frame: em cada linha,
valores de gastos efetuados
     serão colocados de acordo com a pessoa que gastou. Pessoas que
efetuaram de fato um gasto recebem
     o valor do gasto efetuado. Pessoas que participam do consumo mas não
gastaram no ato recebem o
     o valor zero. E por fim, pessoas que não participam do consumo daquele
gasto devem receber NA.

Value:

     A função retorna uma matriz de pagamentos. Para saber se uma pessoa
deve pagar algo a outra,
     basta olhar na linha que contém seu nome. Nas colunas estarão dispostos
os valores que essa
     pessoa deve pagar a todas as outras. Se ela não deve nada, o valor é
zero.

Warning:
```

```
     Caso os parâmetros especificados para o data.frame não esteja
corretamente preenchidos, a função
     não irá funcionar corretamente.

Author(s):

     Gabriel Massami Kayano

References:

   Google Finance Converter
     https://www.google.com/finance/converter

Examples:

## Exemplo 1 : divisão de gastos sem plotagem e sem conversão de moeda
# criando data.frame com os dados:
exemplo1=read.table("exemplo1.csv", header=TRUE, sep=",", as.is=TRUE)
# aplicação da função com graficos=FALSE e moeda.entrada=moeda.saida
dividir.gastos(exemplo1, graficos=FALSE)

## Exemplo 2: divisão de gastos com plotagem e sem conversão de moeda
# criando data.frame com os dados:
exemplo2=read.table("exemplo2.csv", header=TRUE, sep=",", as.is=TRUE)
# aplicação da função com graficos=TRUE
dividir.gastos(exemplo2, graficos=TRUE)

## Exemplo 3: divisão de gastos com plotagem e com conversão de moeda (BRL-
USD)
# criando data.frame com os dados:
exemplo3=read.table("exemplo3.csv", header=TRUE, sep=",", as.is=TRUE)
# aplicação da função com graficos=TRUE
dividir.gastos(exemplo3, moeda.entrada="real", moeda.saida="dolar",
graficos=TRUE)
```

Código da Função

```
### FUNCAO DIVIDIR.GASTOS ###

# esta funcao foi feita para ajudar a dividir gastos igualmente entre
participantes durante um periodo de compartilhamento de gastos (como
viagens, churrascos, festas, por exemplo)
# a funcao permite que se coloque no objeto de entrada valores em ate tres
moedas: real, dolar e euro, e converte os valores fornecidos de acordo com o
especificado nos argumentos "moeda.entrada" e "moeda.saida"
# a funcao tambem plota os seguintes graficos, caso o argumento
```

```
"graficos"esteja marcado como TRUE: gastos cumulativos por dia, total de
gastos por categoria e gastos totais por pessoa.
# o objeto de entrada deve ser um data.frame cujas linhas representem cada
evento de gasto. Nas primeiras colunas do data.frame estarao especificadas
as pessoas que participam do evento, e a cada evento de gasto(linha) o
montante gasto por cada pessoa. Se a pessoa nao participou de uma compra (ou
seja, nao consumiu o produto), deve-se colocar NA ao inves de zero
(reservado para as pessoas que participam da compra mas nao pagaram nada). A
penultima coluna deve conter as categorias correspondentes a cada evento de
gasto, e a ultima a data no formato dd/mm/yy.

dividir.gastos=function(x, moeda.entrada="real", moeda.saida="real",
graficos=TRUE)
{
## PARTE I - INDEXACAO E VERIFICACAO DO DATA.FRAME DE ENTRADA
  # objeto colunas com o numero de colunas do objeto de entrada.
  colunas=ncol(x)
  # objeto eventos com o numero de linhas
  eventos=nrow(x)
  # objeto pessoas com o numero de participantes
  pessoas=colunas-2
  # objeto categoria com o numero da coluna correspondente, para indexacao
  categoria=colunas-1
  # objeto data com o numero da coluna correspondente, para indexacao
  data=colunas
  # verificacao da classe da coluna contendo as datas
  if(class(x[,data])!="Date")
  {
    # conversao da coluna data para a classe Date
    x[,data]=as.Date(x[,data], format="%d/%m/%y")
    # o padrao deve ser dd-mm-aa
  }


  # verificacao das unidades de conversao
  if(moeda.saida=="real")
  {
    # objeto com a sigla de real
    sigla="BRL"
  }
  # se a moeda de saida escolhida for dolar
  if(moeda.saida=="dolar")
  {
    # objeto com a sigla de dolar
    sigla="USD"
  }
  # e se for euro...
  if(moeda.saida=="euro")
  {
    # objeto com a sigla de euro!
```

```
    sigla="EUR"
  }
## PARTE II - CALCULO DAS TAXAS DE CONVERSAO E CONVERSAO DOS VALORES
  # se a moeda inicial for REAL
  if(moeda.entrada=="real")
  {
    # calculo da taxa de conversao para dolar
    if(moeda.saida=="dolar")
    {
      # objeto com o endereco onde consta a taxa de conversao desejada
url="https://www.google.com/finance/converter?a=1&from=BRL&to=USD&meta=ei%3D
e6Q4V_nUKou7eoyQl4gK"
      # objeto com o texto extraido do endereco
      fonte=readLines(url)
      # objeto com a linha que contem o valor de conversao (no caso, a busca
foi pela expressao com a sigla da moeda de entrada BRL)
      linha=grep("1 BRL =", fonte)
      # objeto text contendo somente a linha com o valor de conversao
      text=fonte[linha]
      # extracao dos caracteres numericos do objeto text
      extract=gsub("[^0-9]","",text)
      # calculo do valor maximo de caracteres numericos
      nmax=nchar(extract)
      # objeto contendo somente os numeros de interesse (o primeiro numero
do objeto text eh o numero 1 da expressao 1 BRL = 0...USD por exemplo)
      numeros=as.numeric(substr(extract,nmax-4,nmax))
      # divisao por dez mil para recolocar o ponto decimal no lugar certo(o
site sempre utiliza quatro casas decimais)
      taxa=numeros/10000
    }
# para o caso da moeda de saida ser EURO:
    if(moeda.saida=="euro")
    {
      # objeto contendo o site com a taxa de conversao desejada BRL - EUR
url="https://www.google.com/finance/converter?a=1&from=BRL&to=EUR&meta=ei%3D
BAo5V-HUA4iimAHwwrKIDA"
      # objeto contendo o texto do site
      fonte=readLines(url)
      # localizacao da linha contendo o valor de interesse no texto
      linha=grep("1 BRL =", fonte)
      # indexacao do texto com a linha obtida
      text=fonte[linha]
      # extracao dos algarismos numericos do texto indexado
      extract=gsub("[^0-9]","",text)
      # numero de caracteres dos algarismos extraidos
      nmax=nchar(extract)
      # isolando os numeros de interesse e convertendo-os a classe numerica
      numeros=as.numeric(substr(extract,nmax-4,nmax))
      # calculo da taxa com posicionamento da casa decimal
      taxa=numeros/10000
    }
```

```
# caso o usuario opte por nao fazer conversao
    if(moeda.saida=="real")
    {
      # a taxa de transformacao eh 1
      taxa=1
      }
  }
  # se a moeda de entrada for DOLAR
  if(moeda.entrada=="dolar")
  {
    # calculo da taxa para conversao USD - BRL
    if(moeda.saida=="real")
    {
      # site contendo a taxa de conversao desejada USD - BRL
url="https://www.google.com/finance/converter?a=1&from=USD&to=BRL&meta=ei%3D
ZQs5V7BtgrSYAe3NrbgM"
      # transformando o site em texto no R
      fonte=readLines(url)
      # encontrando a linha contendo a taxa desejada
      linha=grep("1 USD =", fonte)
      # indexacao do texto pela linha encontrada
      text=fonte[linha]
      # extracao dos algarismos numericos do texto ja indexado
      extract=gsub("[^0-9]","",text)
      # obtencao do numero de algarismos obtidos na extracao
      nmax=nchar(extract)
      # selecao dos algarismos desejados e conversao para a classe numeric
      numeros=as.numeric(substr(extract,nmax-4,nmax))
      # obtencao da taxa, dividindo por dez mil para o posicionamento
correto do ponto decimal
      taxa=numeros/10000
    }
    # calculo da taxa de conversao USD-EUR
    if(moeda.saida=="euro")
    {
      # site contendo a taxa de conversao USD-EUR
url="https://www.google.com/finance/converter?a=1&from=USD&to=EUR&meta=ei%3D
QQw5V4HQD4HlmAHAvKvIAQ"
      # conversao do site para texto em R
      fonte=readLines(url)
      # obtencao da linha contendo o valor desejado
      linha=grep("1 USD =", fonte)
      # indexacao do texto pela linha obtida
      text=fonte[linha]
      # extracao dos algarismos do texto
      extract=gsub("[^0-9]","",text)
      # calculo do numero de algarismos extraidos
      nmax=nchar(extract)
      # selecao dos algarismos desejados e conversao em classe numeric
      numeros=as.numeric(substr(extract,nmax-4,nmax))
```

```
      # obtencao da taxa, dividindo por dez mil para o correto
posicionamento do ponto decimal
      taxa=numeros/10000
    }
    # se o usuario optar por nao fazer conversao...
    if(moeda.saida=="dolar")
    {
      # o valor da taxa eh 1
      taxa=1
    }
  }
  # se a moeda de entrada for EURO:
  if(moeda.entrada=="euro")
  {
    # calculo da taxa de conversao EUR - REAL:
    if(moeda.saida=="real")
    {
      # site contendo a taxa de conversao desejada
url="https://www.google.com/finance/converter?a=1&from=EUR&to=BRL&meta=ei%3D
NQ05V_nRK4qVmAHtzJzIAQ"
      # conversao do texto do site em texto do R
      fonte=readLines(url)
      # obtencao da linha contendo o valor de conversao desejado
      linha=grep("1 EUR =", fonte)
      # indexacao do texto presente pela linha obtida
      text=fonte[linha]
      # extracao dos algarismos presentes no texto
      extract=gsub("[^0-9]","",text)
      # calculo do numero de algarismos
      nmax=nchar(extract)
      # extracao dos algarismos de interesse e conversao a classe numeric
      numeros=as.numeric(substr(extract,nmax-4,nmax))
      # obtencao da taxa e divisao por dez mil para correto posicionamento
do ponto decimal
      taxa=numeros/10000
    }
    # calculo da taxa de conversao para DOLAR:
    if(moeda.saida=="dolar")
    {
      # site contendo a conversao EUR-USD
url="https://www.google.com/finance/converter?a=1&from=EUR&to=USD&meta=ei%3D
aA45V5HuH4HlmAHAvKvIAQ"
      # extracao do texto do site para o R
      fonte=readLines(url)
      # obtencao da linha contendo o valor desejado
      linha=grep("1 EUR =", fonte)
      # indexacao do texto pela linha obtida
      text=fonte[linha]
      # extracao dos algarismos presentes na linha
      extract=gsub("[^0-9]","",text)
      # calculo do numero de algarismos da linha
```

```
      nmax=nchar(extract)
      # extracao dos algarismos desejados e conversao a classe numeric
      numeros=as.numeric(substr(extract,nmax-4,nmax))
      # obtencao da taxa, dividindo por dez mil para o correto
posicionamento do ponto decimal
      taxa=numeros/10000
    }
    # caso o usuario nao queira fazer nenhuma conversao...
    if(moeda.saida=="euro")
    {
      # o valor da taxa eh 1!
      taxa=1
    }
  }
  # CONVERSAO DOS VALORES DO DATA.FRAME
  # indexacao do data.frame - remocao das colunas contendo categoria e data
  gastos=x[,1:pessoas]
  # multiplicacao do data.frame contendo somente valores pela taxa
encontrada acima. Valores convertidos!
  valores=gastos*taxa
  valores
## PARTE III CALCULO DOS VALORES DE TRANSACAO E MATRIZ DE PAGAMENTOS
  # calculo da media para gasto, considerando apenas aqueles que
participaram da compra - removendo aqueles que constam NA
  medias=apply(valores, 1, mean, na.rm=TRUE)
  # calculo do total por evento de gasto (linhas)
  total.gasto=apply(valores, 1, sum, na.rm=TRUE)
  # obtencao da diferenca entre os valores(ja convertidos) do data.frame e a
media para cada gasto. Valores negativos indicam que a pessoa pagou menos do
que deveria pelo evento (e portanto deve), e positivos indicam que a pessoa
pagou mais do que deveria (e portanto deve receber)
  diferencas=valores-medias
  # encontrando os devedores e os recebedores
  # somando todos os valores de diferencas para cada pessoa, obtem-se o
SCORE final, um valor final considerando dividas e creditos. Mais uma vez,
NAs sao removidos para pessoas que nao participaram da compra.
  score.final=apply(diferencas, 2, sum, na.rm=TRUE)
  # sao definidos recebedores como aqueles com um score positivo, que sao
arranjados em ordem decrescente de valor a receber.
  recebedores=sort(score.final[score.final>0], decreasing=TRUE)
  # sao definidos os devedores como aqueles com score negativo, que sao
arranjados em ordem decrescente pelo modulo do valor que devem.
  devedores=sort(score.final[score.final<0])
  # calculado o numero de recebedores
  nrec=length(recebedores)
  # calculado o numero de devedores
  ndev=length(devedores)
  # criacao de uma matriz vazia. Nela, as linhas representam o valor que uma
pessoa deve pagar e as colunas a quem elas devem pagar.
  matriz=matrix(NA, nrow=pessoas, ncol=pessoas)
```

```
  # atribuicao dos nomes das colunas e linhas extraidas do data.frame de
entrada.
  dimnames(matriz)=list(names(valores), names(valores))
  # criacao de um objeto score.temp temporario igual ao score.final, que
sera alterado durante os calculos das transacoes.
  score.temp=score.final
  # criacao de um ciclo para calculo, utilizando como valor maximo o produto
do numero de recebedores e devedores
  for(i in 1:(nrec*ndev))
  {
    # valor de indexacao para o maior score - recebedor que deve receber a
maior quantia
    ind.max=which.max(score.temp)
    # valor de indexacao para o menor score - devedor que deve a maior
quantia
    ind.min=which.min(score.temp)
    # obtencao do maior score recebedor
    val.max=max(score.temp)
    # obtencao do maior score devedor
    val.min=min(score.temp)
    # calculo da diferenca entre o maior score recebedor e maior score
devedor (somatoria pois o score devedor ja tem sinal negativo). Representa a
transacao feita entre o maior recebedor e o maior devedor.
    dif.1=val.max+val.min
    # se o valor da diferenca for maior que zero, significa que apos essa a
transacao o maior recebedor ainda precisa receber.
    if(dif.1>0)
    {
      # o valor que resta apos a transacao eh colocado na posicao indexada
do mesmo recebedor (era o maior recebedor)
      score.temp[ind.max]=dif.1
      # como a divida foi quitada, o valor zero eh colocado na posicao
indexada do devedor, que agora nao deve mais nada.
      score.temp[ind.min]=0
      # na matriz de pagamentos, na posicao que relaciona as duas pessoas em
questao, eh colocado o valor que o devedor devia(valor cujo sinal ja era
negativo)
      matriz[ind.min,ind.max]=-val.min
    }
    # se o valor da diferenca for menor que zero, significa que o valor pago
por este devedor eh mais do que suficiente para quitar essa divida.
    if(dif.1<0)
    {
      # neste caso, o novo score do recebedor eh zero (ja recebeu o que
precisava)
      score.temp[ind.max]=0
      # o score do devedor torna-se a mera diferenca encontrada, que tambem
eh o valor que ele ainda precisa pagar pra quitar sua divida por completo.
      score.temp[ind.min]=dif.1
      # na matriz de pagamentos eh colocado o valor total que o recebedor
precisava receber, que estava definido no objeto val.max
```

```
      matriz[ind.min,ind.max]=val.max
    }
    # agora, caso o valor dessa diferenca de zero (o devedor pagou
integralmente o que o recebedor deveria receber e ainda por cima quitou
totalmente sua divida)
    if(round(dif.1)==0)
    {
      # o valor do score do recebedor eh modificado a zero...
      score.temp[ind.max]=0
      # e o valor do score do devedor tambem.
      score.temp[ind.min]=0
      # na matriz de pagamentos, eh colocado o valor que o recebedor
precisava receber, armaeznado em val.max.
      matriz[ind.min,ind.max]=val.max
    }
  }
  # por mero capricho, substituem-se os indices nao utilizados da matriz de
NA pra zero.
  matriz[is.na(matriz)]=0
  # a seguir, arrendondam-se os valores da matriz para que fiquem
compativeis com quantias monetarias. Apos rodar todos os ciclos, a matriz de
pagamentos esta pronta!
  matriz=round(matriz, digits=2)

# IMPRESSAO DO RESULTADO NA TELA
  # cria-se um ciclo para as linhas da matriz de pagamentos.
  for(i in 1:pessoas)
  {
    # cria-se outro ciclo para as colunas da matriz de pagamentos.
    for(j in 1:pessoas)
    {
      # para todos os valores diferentes de zero(como so existem valores
positivos, o sinal utilizado foi >)...
      if(matriz[i,j]>0)
      {
        # sera feita uma impressao no formato de 'FULANO deve X para
CICRANO', buscando os nomes das pessoas na matriz e buscando tambem a sigla
correspondente a escolha do usuario (definida em um dos primeiros passos da
funcao)
        cat(paste(dimnames(matriz)[[1]][i], "deve" ,matriz[i,j],
sigla,"para", dimnames(matriz)[[2]][j],"\n"))
      }
    }
  }

## PARTE IV - PLOTAGEM DOS GRAFICOS
  # caso o usuario tenha optado por plotar os graficos:
  if(graficos==TRUE)
  {
  # serao plotados tres graficos, entao eh necessario um layout que permita
```

```
tres graficos simultaneos
  par(mfrow=c(2,2))
  # GRAFICO I - GASTOS CUMULATIVOS definidos por data
  # o total gasto por evento eh colocado em uma nova coluna no data.frame
  x$total=total.gasto
  # objeto cumulativo contendo datas e gastos totais por evento. Indexacao
buscando a ultima e a penultima colunas do data.frame.
  cumulativo=x[,c(ncol(x)-1, ncol(x))]
  # ordenacao das datas em ordem cronologica
  cumulativo=cumulativo[order(cumulativo$data),]
  # criacao de uma nova coluna no objeto cumulativo, contendo a soma
cumulativa dos gastos totais.
  cumulativo$soma=cumsum(cumulativo$total)
  # scatterplot de soma cumulativa por data, onde se pode acompanhar o
crescente volume de gastos durante o periodo, a cada dia.
  plot(cumulativo$data, cumulativo$soma, type="o", main=paste("Gastos
acumulados por dia"), xlab="", ylab=sigla, bty="l", pch=16, col="orange",
cex.axis=0.8, las=2)
  # GRAFICO II - SETORES POR CATEGORIA
  # indexacao do data.frame com o total gasto(calculado para o ultimo
grafico), selecionando apenas a ultima e a antipenultima colunas (total
gasto e categoria, respectivamente)
  setores=x[,c(ncol(x)-2,ncol(x))]
  # conversao da coluna categoria em classe factor
  as.factor(setores$categoria)
  # calculo da media por categoria do total gasto - armazenamento no objeto
novo "dados"
  dados=aggregate(setores$total, by=list(setores$categoria), FUN=mean)
  # somatoria total de gastos(tdos os eventos, todas as pessoas, todas as
categorias)
  som.set=sum(dados[,2])
  # calculo do gasto percentual por categoria - valores sao colocados em uma
nova coluna do objeto "dados"
  dados$percent=round(dados[,2]/som.set*100, digits=2)
  # definicao de um numero de cores do grupo terrain.colors compativel com o
numero de categorias.
  cor=terrain.colors(n=length(dados[,1]))
  # plot de um grafico de setores utilizando os gastos percentuais
  pie(dados[,2], labels=dados[,3], col=cor, main="Gastos por Categoria (%)",
radius=0.9)
  # legenda posicionada no canto superior direito, com dados extraidos do
objeto "dados".
  legend("topright", legend=c(dados[,1]), fill=cor, bty="n", cex=0.8)
  # GRAFICO III - BARPLOT COM GASTO TOTAL POR PESSOA E MEDIA GERAL
  # calculo do valor total gasto por pessoa, excluindo os NAs da conta
  tot.ind=apply(valores, 2, sum, na.rm=TRUE)
  # calculo da media geral de gastos por pessoa
  media.pessoa=sum(tot.ind)/pessoas
  # definicao de um numero de cores compativel com o numero de pessoas
  bar.col=heat.colors(n=pessoas)
  # barplot com os valores de gasto total por pessoa, atentando para o uso
```

```
da sigla correta, definida previamente,
  barplot(tot.ind, col=bar.col, ylim=c(0,max(tot.ind)*1.5), las=2,
border=FALSE, main=paste("Gastos (", sigla, ") por pessoa"))
  # plot de uma linha tracejada horizontal na media geral de gasto por
pessoa
  abline(h=media.pessoa, lty=2)
  # posicionamento da legenda no canto superior direito
  legend("topright", lty=2, legend=paste(round(media.pessoa, digits=2),
sigla), bty="o", cex=1, title=paste("media/pessoa"))
  }
  # apos a plotagem dos tres graficos, o argumento mfrow que havia sido
alterado eh devolvido ao estado original
  par(mfrow=c(1,1))
  # a funcao por fim, retorna a matriz de pagamentos, sendo possivel
atribui-la a um objeto.
  return(matriz)
}
```