Renata Pereira Beco

Mestranda em Zoologia pelo Instituto de Biociências, USP

exec

Trabalho final

Proposta A

Organização e análise de dados de coloração em organismos

A função tem como objetivo organizar o data frame de análise de coloração que são gerados no ImageJ (software de processamento e análise de imagens), calcular a luminância a partir dos próprios dados e a média dos valores obtidos por espécie.

Contexto

Uma das técnicas utilizadas para analisar cores em seres vivos é a fotografia digital. A técnica consiste em tirar fotos com condições padronizadas e analisar as imagens em softwares de processamento de imagem como o ImageJ. Existe uma ferramenta nesse software que permite selecionar as diferentes regiões de interesse do organismo estudado (eg. costas, garganta e barriga) e gerar uma tabela com os valores médios de coloração de acordo com o sistema de cores RGB (vermelho, verde e azul), que é o sistema adotado em câmeras fotográficas. As linhas da tabela correspondem às diferentes regiões medidas e as colunas contêm os valores de RGB e outros valores de variáveis que não são utilizados nos estudos exclusivos de coloração. Se objetivo do trabalho é comparar as diferentes regiões de interesse entre espécies distintas, a tabela precisa ser reorganizada de forma que cada linha seja um indivíduo de uma espécie e as colunas contenham os valores de RGB das diferentes regiões estudadas. Por exemplo, se estamos estudando as regiões 1, 2 e 3 de dois indivíduos da espécies A e B, a tabela de dados gerada pelo ImageJ teria um padrão assim:

Mas para que possamos comparar as diferentes regiões entre as duas espécies, a tabela de dados ideal seria assim:

Sendo assim, a função irá omitir as variáveis que não forem os valores de RGB, transpor os dados de forma que cada linha corresponda a um indivíduo de uma espécie e as colunas contenham os valores de RGB das diferentes regiões analisadas. Além disso, os valores de RGB de cada região serão somados para se calcular a luminância (R+G+B= luminância, de acordo com Endler, 2012) e todos esses valores (RGB e luminância) de cada região por indivíduo terão sua média calculada por espécie.

O input da função seria o data frame com os dados de coloração gerados no ImageJ e o output seriam dois data frame:

- Data frame 1: As linhas correspondem aos indivíduos de cada espécie a as colunas correspondem aos valores de RGB de cada região. Uma coluna com os valores de luminância será criada e as outras variáveis que não correspondem aos valores de RGB e luminância serão

omitidas.
- Data frame 2: Semelhante ao primeiro, mas contém apenas os valores médios de RGB e luminância por espécie.

Mais informações sobre a ferramenta de análise de cor estão disponíveis nesse link: http://www.jolyon.co.uk/myresearch/image-analysis/image-analysis-tools/

Referências:

Endler, J. A. (2012). A framework for analysing colour pattern geometry: Adjacent colours. Biological Journal of the Linnean Society 107:233–253.

Troscianko, J., and M. Stevens (2015). Image calibration and analysis toolbox - a free software suite for objectively measuring reflectance, colour and pattern. Methods in Ecology and Evolution 6:1320–1331.

> *— Diogo Melo 2017/06/05 16:20*
>
> Proposta legal, mas pra ficar boa mesmo precisa ser mais genérica. Esse procedimento que vc escreveu é conhecido por ai como transformar uma tabela narrow em wide, e é um problema que aparece o tempo todo em quando estamos lidando com modelos lineares em vários pacotes estatítisticos.
>
> Acho que sua função deveria receber um data.frame qualquer, uma lista de colunas onde estão os dados a serem “expandidos” em mais colunas, e uma coluna onde está a informação de qual a categoria daquela linha em particular. No seu caso, as colunas a serem expandidas seriam a vT vG e VB, e a coluna categórica seria uma coluna com as informações de indivíduo e região.
>
> Quanto as outras colunas, o ideal seria mantê-lás e depois usar indexação caso elas não sejam necessárias. Mas isso complica um pouco e não é estritamente necessário.
>
> Não é uma função fácil de ser feita de forma genérica, mas vai te dar muita segurança pra lidar com indexação. Proposta desafiadora e bacana.

> Oi Diogo, obrigada pelas sugestões e comentários! Vc acha melhor então eu reformular essa minha proposta A de uma forma mais genérica, né? Apenas para confirmar se eu entendi direito, uma das possibilidades de data frame de entrada nessa função que pretendo criar deveria ser composto por pelo menos três colunas de informação diferentes:

- Coluna 1= coluna categórica que pretendo manter do mesmo jeito
- Coluna 2= coluna categórica que pretendo expandir junto com as variáveis presentes nas próximas colunas
- Coluna 3= coluna com um tipo de variável que pretendo expandir com a Coluna 2

A ideia então seria a pessoa poder expandir as colunas de variáveis (coluna 3 e outras colunas que existirem nesse mesmo estilo) com as diferentes categorias existentes na coluna 2.

Eu acredito que essa função de forma mais genérica seria útil para uma maior quantidade de pessoas do que a que eu estava propondo antes. Mas aí eu tenho outra dúvida. Eu entendo que não vai ser uma tarefa fácil fazer essa função de forma genérica, mas vc acha que uma função que faz apenas essa expansão de tabelas é uma proposta legal de trabalho final para disciplina? Fiquei um pouco na dúvida sobre isso agora.

Proposta B

Cálculo do volume de reagentes de PCR

A proposta da função é calcular o volume de cada reagente utilizado no PCR para preparar um mix de acordo com o número de reações a ser realizado. Esses valores calculados levarão em conta o controle negativo e o erro de pipetagem.

Contexto

O PCR (Reação em cadeia da polimerase) é uma técnica que consiste em fazer várias cópias de uma região específica do DNA. Para isso, é preciso usar certos reagentes em quantidades especificadas por protocolos padronizados. Quando é necessário fazer mais de uma reação de PCR, geralmente é preparado um mix com os reagentes num valor multiplicado pelo número de reações que serão realizadas. Ou seja, se uma pessoa precisa fazer PCR de três amostras, ela pode preparar um mix com reagentes para três amostras e depois distribuir em três tubos diferentes. Além disso, toda reação realizada deve ser acompanhada de um negativo, que seria uma reação com todos os reagentes menos a amostra de DNA, a fim de saber se não há contaminação na reação. Outra questão importante é que o volume de mix preparado deve levar em conta um erro associado à pipetagem - distribuição dos volumes de reação em cada tubo.

A função deve conter os seguintes argumentos:

- agua = volume de água ultrapura necessária para reação
- tampao = volume do tampão (buffer)
- dntps= volume de dNTPs
- primer.f = volume de primer foward
- primer.r = volume de primer reverse
- mgcl = volume cloreto de magnésio
- taq = volume da enzima taq polimerase
- erro = porcentagem de erro
- n = número de reações + 1 (controle negativo)

Como os volumes utilizados de cada reagente podem variar de acordo com o protocolo adotado, os argumentos terão valores padrão de um protocolo de PCR já utilizado por mim, mas que podem ser modificados de acordo com o interesse do usuário da função.

Alguns protocolos podem não utilizar algum reagente que eu adicionei como argumento na função, ou até mesmo pode ter outros reagentes que eu não disponibilizei. Assim, a ideia seria inserir o máximo possível de reagentes opcionais utilizados em PCR como argumentos da função, mas que não fossem obrigatórios para a função rodar.

O output da reação da reação seria um vetor com os volumes de cada reagente necessário para a reação, de acordo com o número de reações escolhido e levando em conta o negativo e o erro de pipetagem.

> *— Diogo Melo 2017/06/05 16:28*
>
> Essa proposta é um pouco simples demais. Seria basicamente uma calculadora pra somar os volumes? Acho que precisa de mais um pouco de funcionalidade.

> Sim, concordo que essa proposta é muito simples. Estava com dificuldades em pensar em uma proposta B e sei que ela seria útil para quem trabalha com molecular. Como os reagentes tem volumes diferentes que devem ser multiplicados pelo números de reações + negativo + erro de pipetagem, muita gente acaba se atrapalhando na hora de montar a tabela de reagentes e calcular esses valores quando tem muitas amostras para fazer de uma vez. Daria para pensar em outras funcionalidades que seriam úteis, mas acho que mesmo assim continuaria uma função simples. Visto isso, posso investir na proposta A então?

Proposta A reformulada

A função tem como objetivos:

- Expandir um data frame com valores de RGB (coloração) de acordo com as diferentes regiões amostradas por indivíduo de cada espécie
- Calcular a luminância de cada região por indivíduo e espécie
- Calcular a PCA dos valores de RGB e luminância, a fim de saber qual das variáveis de qual região explica melhor a variação de cores nos indivíduos e espécies amostradas

Para isso, a função possui os seguintes argumentos:

- dfin=data frame de entrada da função
- spp=coluna de espécies do dfin
- ind=coluna de indivíduos do dfin

- reg=coluna das regioes do dfin
- R=coluna de valores de R do dfin
- G=coluna de valores de G do dfin
- B=coluna de valores de B do dfin

A função devolve na saída um data frame com os valores de RGB e luminância de cada região por indivíduo e espécie, os valores calculados da PCA e um gráfico da PCA calculada.

Arquivo da função: color.alys

Código da função

```
 #Funcao que realiza tres tarefas: 1) expande data.frame com dados de cores
de RGB de acordo com as diferentes regioes amostradas de cada individuo de
cada especie, 2)calcula a luminancia de cada regiao por individuo e especie,
3) calcula a PCA dos valores de RGB e luminancia
color.alys<-function(dfin,spp,ind,reg,R,G,B)
{
  dfin=data.frame(spp,ind,reg,R,G,B) #dataframe de entrada da funcao
  spp=dfin$spp #coluna de especies
  ind=dfin$ind #coluna de individuos
  reg=dfin$reg #coluna das regioes
  R=dfin$R     #coluna de valores de R
  G=dfin$G     #coluna de valores de G
  B=dfin$B     #coluna de valores de B
  lum=R+G+B    #criacao de um vetor que calcula os valores de luminancia
  dfinR=as.data.frame(matrix(ncol=length(unique(reg)),
nrow=(length(R)/length(unique(reg))))) #dataframe vazio onde serao guardados
os valores de R para cada regiao diferente
  colnames(dfinR)<-paste(unique(reg)) #nomeacao das colunas das diferentes
regioes
  dfinR
  for(i in unique(reg)) #for que ira preencher os valores de R para cada
regiao no dataframe dfinR
  {
    dfinR[i]=R[reg==i]
  }
  dfinG=as.data.frame(matrix(ncol=length(unique(reg)),
nrow=(length(G)/length(unique(reg))))) #dataframe vazio onde serao guardados
os valores de G para cada regiao diferente
  colnames(dfinG)<-paste(unique(reg)) #nomeacao das colunas das diferentes
regioes
  for(j in unique(reg)) #for que ira preencher os valores de G para cada
regiao no dataframe dfinG
  {
    dfinG[j]=G[reg==j]
  }
  dfinB=as.data.frame(matrix(ncol=length(unique(reg)),
nrow=(length(B)/length(unique(reg))))) #dataframe vazio onde serao guardados
os valores de B para cada regiao diferente
  colnames(dfinB)<-paste(unique(reg)) #nomeacao das colunas das diferentes
```

```
regioes
  for(k in unique(reg)) #for que ira preencher os valores de B para cada
regiao no dataframe dfinB
  {
    dfinB[k]=B[reg==k]
  }
  dflum=as.data.frame(matrix(ncol=length(unique(reg)),
nrow=(length(lum)/length(unique(reg))))) #dataframe vazio onde serao
guardados os valores de luminancia para cada regiao diferente
  colnames(dflum)<-paste(unique(reg)) #nomeacao das colunas das diferentes
regioes
  dflum
  for(o in unique(reg))
  {
    dflum[o]=lum[reg==o]
  }
  dfout=data.frame(spp=NA,ind=NA,dfinR,dfinG,dfinB,dflum) #dataframe de
saida com uma coluna vazia de especies, uma coluna vazia de individuos e as
outra colunas com os valores de R,G,B e luminancia para cada regiao
  dfout
  for (e in unique(reg)) #Criacao de dois for que irao preencher as colunas
de especies e individuos, de acordo com as colunas de R,G,B e luminancia por
regiao
  {
    for (f in unique(reg))
    {
      dfout[,1]=spp[reg==e]
      dfout[,2]=ind[reg==f]
    }
  }
  dfout #Dataframe de saida com as colunas de especies e individuos
devidamente preenchidas de acordo com as colunas de R,G,B e luminancia por
regiao
  colnames(dfout)<-c(paste("spp"),paste("ind"),paste("R", unique(reg),sep =
"."),paste("G", unique(reg),sep = "."),paste("B", unique(reg),sep =
"."),paste("lum", unique(reg),sep = ".")) #nomeacao correta de cada coluna
de acordo com a quantidade de regioes: spp=especies,ind=individuos,R+nome da
regiao, G+nome da regiao, B+nome da regiao, lum+nome da regiao
  dfout.log<-log(dfout[,-c(dfout$spp,dfout$ind,dfout$reg)]) #calculo do log
dos valores de R,G,B e luminancia de cada regiao
  PCA<-prcomp(dfout.log) #PCA comum feito a partir dos valores de log
calculados anteriormente
  x11() #abrir uma janela de grafico
  biplot(prcomp(dfout.log)) #plotagem da PCA na janela aberta anteriormente
  outlist<-list(dfout,PCA) #criacao de uma lista com os objetos que serao
retornados: dataframe de saida com as colunas de especies e individuos
devidamente preenchidas de acordo com as colunas de R,G,B e luminancia por
regiao, e os valores da PCA
  return(outlist)  #retorno da lista com os objetos
```

```
}
```

Help da função

```
 color.alys                package:unknown                R Documentation

Função que expande data.frame com dados de cores de RGB de acordo com as
diferentes regiões amostradas de cada indivíduo de cada espécie, calcula a
luminância de cada região por indivíduo e espécie, e calcula a PCA dos
valores de RGB e luminância


Description:

     A função irá utilizar um data frame de entrada com valores de RGB de
acordo com a região, espécie e indivíduo e irá gerar um data frame de saída
com os valores de RGB e luminância por região de cada indivíduo e espécie.
     Além disso, irá gerar um objeto com a PCA calculada dos valores de RGB
e luminância, e o gráfico da PCA calculada.

Usage:

     color.alys<-function(dfin,spp,ind,reg,R,G,B)

Arguments:

  dfin  data frame de entrada da função
  spp   coluna de espécies do dfin
  ind   coluna de indivíduos do dfin
  reg   coluna das regiões do dfin
  R     coluna de valores de R do dfin
  G     coluna de valores de G do dfin
  B     coluna de valores de B do dfin

Details:

  A luminância é calculada a partir da soma dos valores de R, G e B. Ela
basicamente representa a intensidade de claro e escuro de cada região
amostrada.
  A PCA é calculada a fim de saber qual das variáveis de qual região explica
melhor a variação de cores nos indivíduos e espécies amostradas. Ela é
realizada apenas como um análise preliminar, que irá permitir o usuário
explorar seus dados e escolher as variáveis e regiões de interesse.

Value:

  comp1 : Data frame com os valores de RGB e luminância por região de cada
indivíduo e espécie.

  comp2 : Resultado da PCA calculada com os valores de RGB e luminância
```

```
  comp2 : Gráfico da PCA calculada


Author(s):

     Renata Beco
     Contact Info: renata.beco@usp.br

References:

     Endler, J. A. (2012). A framework for analysing colour pattern
geometry: Adjacent colours. Biological Journal of the Linnean Society
107:233–253.

     Troscianko, J., and M. Stevens (2015). Image calibration and analysis
toolbox - a free software suite for objectively measuring reflectance,
colour and pattern. Methods in Ecology and Evolution 6:1320–1331.


Examples:

     ## Usando a função com exemplos diferentes de data frame
df.c.spp=data.frame(spp=rep(c("M_klagesi","M_cherriei"),each=6),ind=rep(1:3,
each=2),cat = c("throat", "belly"),
Rmean=c(18230.608,22795.743,17455.498,19821.663,24921.679,21802.825,14138.32
3,21588.183,10946.01,19106.609,8670.45,12657.933),
Gmean=c(15602.707,18894.751,16338.547,18674.353,21601.441,18370.581,13443.51
6,20600.13,9600.023,16144.581,8596.683,12297.234),
Bmean=c(10963.59,12567.816,14189.35,16050.867,15623.025,12508.535,11910.386,
18521.985,7222.401,11145.361,8166.308,11318.783)) #dataframe de exemplo com
duas regiões distintas: "throat" e "belly"
color.alys(dfin=df.c.spp,spp=df.c.spp$spp,ind=df.c.spp$ind,reg=df.c.spp$cat,
R=df.c.spp$Rmean,G=df.c.spp$Gmean,B=df.c.spp$Bmean) #usando a função com o
exemplo acima
df.c.spp2=data.frame(spp=rep(c("M_klagesi","M_cherriei"),each=6),ind=rep(1:2
,each=3),cat = c("throat", "belly","cheek"),
Rmean=c(18230.608,22795.743,17455.498,19821.663,24921.679,21802.825,14138.32
3,21588.183,10946.01,19106.609,8670.45,12657.933),
Gmean=c(15602.707,18894.751,16338.547,18674.353,21601.441,18370.581,13443.51
6,20600.13,9600.023,16144.581,8596.683,12297.234),
Bmean=c(10963.59,12567.816,14189.35,16050.867,15623.025,12508.535,11910.386,
18521.985,7222.401,11145.361,8166.308,11318.783)) #dataframe de exemplo com
três regiões distintas: "throat", "belly" e "cheek"

color.alys(dfin=df.c.spp2,spp=df.c.spp2$spp,ind=df.c.spp2$ind,reg=df.c.spp2$
cat,R=df.c.spp2$Rmean,G=df.c.spp2$Gmean,B=df.c.spp2$Bmean) #usando a função
com o exemplo acima
```