Objetivo: apresentar um esquema para replicar as análises de imagens usada na minha tese (de doutorado disponível em: https://repositorio.unb.br/handle/10482/38765), usando o pacote bwimage.

Ambiente de desenvolvimento: Versão dos softwares/pacotes usados nesse tutorial:

- R version 4.1.2
- R studio 2021.09.1
- Bwimage 1.3

Comentários sobre as análises: Nesse tutorial vamos pegar imagens feitas ao redor de ninhos de tiziu. Serão imagens de 3 ninhos, dos quais foram feitas 5 fotos da vegetação ao redor de cada ninho (portanto serão 15 imagens analisadas). Para cada imagem vamos coletar o dado de densidade total de vegetação (proporção de pixeis pretos/total de pixel da imagem) e índice de agregação (veja mais detalhes do que é essa métrica nas referências abaixo).

Biagolini-Jr, C., & Macedo, R. H. (2019). bwimage: A package to describe image patterns in natural structures. F1000Research, 8. https://f1000research.com/articles/8-1168/v3.

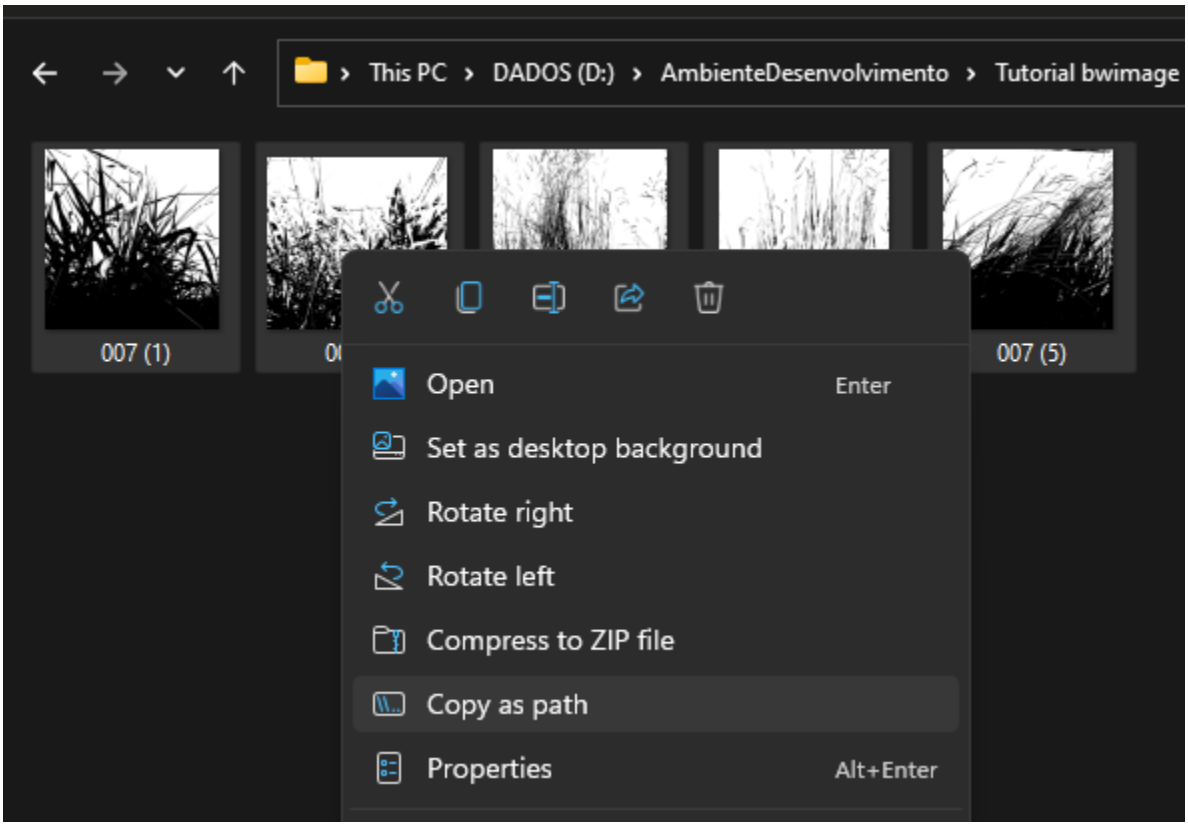
Biagolini-Jr, C. Using bwimge R package to describe patterns in images of natural structures. https://r-posts.com/using-bwimge-r-package-to-describe-patterns-in-images-of-natural-structures/

Extra, se você quiser saber mais informações sobre como eu apresentei essa analise em artigos publicados, veja:

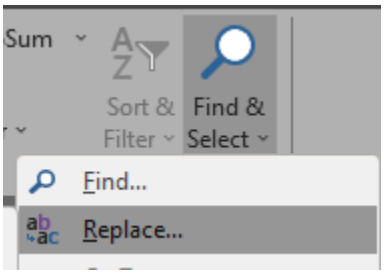
Biagolini-Jr, Carlos, et al. "Food, shadow, and fire influence a tropical bird´ s display." Behavioral Ecology and Sociobiology 75.5 (2021): 1-15. https://link.springer.com/article/10.1007/s00265-021-03015-2

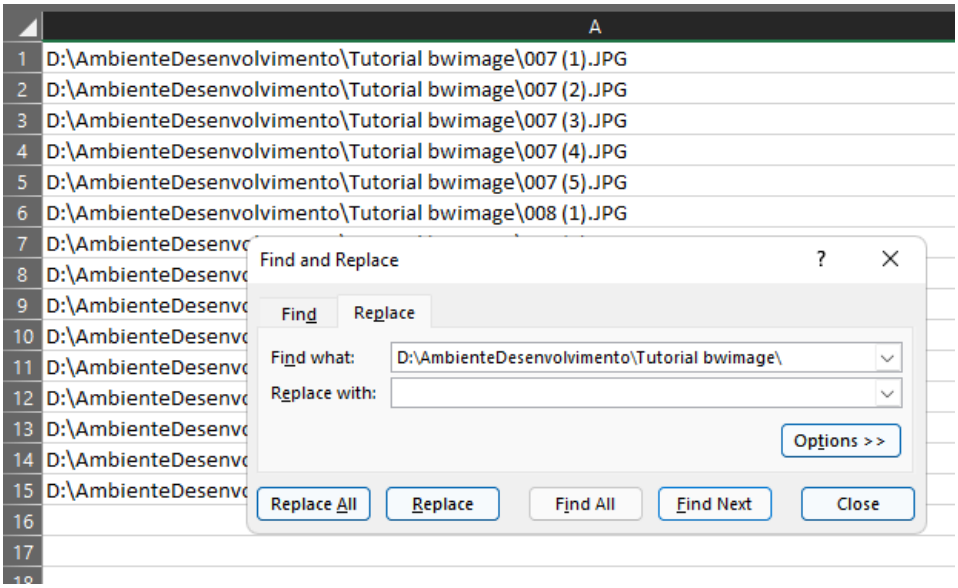
Biagolini Jr, C., & Macedo, R. H. (2021). Philornis infection in blue-black grassquits: impact on nestlings and risk factors involved. Journal of Avian Biology, 52(1). https://doi.org/10.1111/jav.02592.

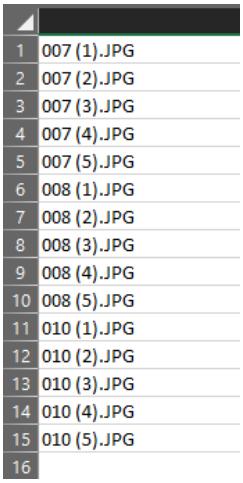
Passo 1: Vá na pasta onde você tem suas imagens salvas (no meu caso D:\AmbienteDesenvolvimento\Tutorial bwimage) selecione todas as imagens (CTRL + A), clique com botão direito, e vá em “Copy as path” (se você estiver usando o Windows 10, você precisa segurar o botão shift enquanto clica com botão direito do mouse).



Passo 2: Abra o Excel, cole o path (caminho das imagens no seu HD). Depois vá em “Localizar e substituir”. Coloque no campo de busca o início do path (no meu exemplo abaixo é “D:\AmbienteDesenvolvimento\Tutorial bwimage\”), e no campo de “substituir por” você deixa em branco. Em seguida clique em substituir todos. A ideia é que você fique com uma tabela com a lista de todas as figuras que você vai analisar. Em seguida, salve a lista de nome de imagens na mesma pasta onde vão estar suas imagens. No meu caso, usei o nome “Lista_figuras”, se você for seguir o código do R que vou apresentar mais adiantes, recomendo usar o mesmo nome.







Passo 5: Abra o R studio, para rodamos os códigos a seguir (códigos também disponíveis no arquivo “Tutorial bwimage.R”, disponível para acesso em https://github.com/biagolini/tutoriais.

Limpar a memória do R

```
rm(list=ls())
gc()
```

Definir diretório de trabalho

Opção 1 - passar o path na mão

```
setwd("D:/AmbienteDesenvolvimento/Tutorial bwimage/")
```

Opção 2 - usar uma tela de navegação

```
setwd(choose.dir())
```

Instalar os pacotes necessários - Se já tiver instalado pode pular essa parte

```
install.packages("bwimage")
install.packages("openxlsx")
```

Carregar pacotes necessários

```
library(bwimage)
library(openxlsx)
```

Carregar uma lista com o nome das figuras

```
lista_arquivos<- read.xlsx("Lista_figuras.xlsx", sheet = 1,colNames = FALSE)
```

Criar um objeto com a lista de nome de arquivos

```
lista_imagens<-as.character(lista_arquivos[,1])
```

Converter as imagens em matrizes binarias. Essa parte pode demorar, tenha paciência.

```
lista_matizes<-threshold_image_list(lista_imagens, filetype
="jpg",compress_method ="frame_fixed", target_width=1000,
target_height=1000)
```

Criar matrizes em branco para enviar o resultado da análise de dados

```
matriz_resposta<-matrix(NA,ncol=2,nrow=length(lista_imagens))
colnames(matriz_resposta)<-c("DV","Agregacao")
```

Loop para análise das figuras em cada conjunto de imagens. Essa parte vai demorar ainda mais que a de antes, aqui o hardware do seu computador será levado ao máximo que o R consegue usar. Minha experiência com R, diz que normalmente o gargalo está na velocidade da sua memória Ram. Por isso, às vezes um computador mais novo com uma memória mais rápida, pode rodar a análise mais rápida que em um computador antigo, mesmo que o antigo tenha um processador potente.

```
for(i in 1:length(lista_imagens)){
  matriz_resposta[i,1]<-denseness_total(lista_matizes[[i]])
  matriz_resposta[i,2]<-aggregation_index(lista_matizes[[i]])[1]
}
```

Exportar resultados

```
resultado_completo<-as.data.frame(matriz_resposta)
resultado_completo$Identificacao<-lista_arquivos[,1]
write.xlsx(resultado_completo,    file    =    "Resultado    completo.xlsx",
overwrite=TRUE)
```

Aqui já temos uma tabela com o resultado de DV e agregação de todas as imagens, agora vamos aproveitar e resumir os dados.

```
n_figuras<-5  numero de figuras por ninho
matriz_media5<-
matrix(NA,ncol=3,nrow=length(matriz_resposta[,1])/n_figuras)
colnames(matriz_media5)<-c("Identificação do ponto","DV","Agregacao")

for( i in 1: length(matriz_media5[,1])){
  matriz_media5[i,2:3]<-apply(matriz_resposta[((5*i)-4):(5*i),],2 ,mean)
  matriz_media5[i,1]<-lista_arquivos[,1][(5*i)-4]
}
```

Exportar resultados resumidos

```
matriz_media5<-data.frame(matriz_media5)
write.xlsx(matriz_media5, file = "Resultado resumido.xlsx", overwrite=TRUE)
```

NOTA

Por limitações do R, se sua lista de imagens for grande, o R não vai conseguir lhe dar com as listas de matrizes (porque existe um tamanho máximo de arquivo que o R consegue trabalhar, e sua lista com dados de todas as figuras pode ultrapassar esse limite).

Isso aconteceu comigo, nas análises da minha tese (os artigos que indiquei no início do tutorial são capítulos da tese). Para solucionar esse problema, eu dividi a lista de arquivos em grupos de 300 imagens. Dependendo dos valores que você escolher em “target_width” e “target_height”, você pode mudar o tamanho dos grupos (quanto maior o target, maior o tamanho da matriz de 1s e 0s, portanto menos imagens podem ser colocadas em cada grupo. Neste novo cenário, as analises ficariam assim:

Particionar as figuras em grupos de até 300s imagens.

```
lista_imagens1<-as.character(lista_arquivos[1:300,1])
lista_imagens2<-as.character(lista_arquivos[301:600,1])
lista_imagens3<-as.character(lista_arquivos[601:900,1])
lista_imagens4<-as.character(lista_arquivos[901:1120,1])
```

Converter as imagens em matrizes binarias

```
lista_matizes1<-threshold_image_list(lista_imagens1,              filetype
="jpg",compress_method          ="frame_fixed",          target_width=1000,
target_height=1000)
lista_matizes2<-threshold_image_list(lista_imagens2,              filetype
="jpg",compress_method          ="frame_fixed",          target_width=1000,
target_height=1000)
lista_matizes3<-threshold_image_list(lista_imagens3,              filetype
="jpg",compress_method          ="frame_fixed",          target_width=1000,
target_height=1000)
lista_matizes4<-threshold_image_list(lista_imagens4,              filetype
="jpg",compress_method          ="frame_fixed",          target_width=1000,
target_height=1000)
```

Criar matrizes em branco para o resultado da análise de dados

```
matriz_resposta1<-matrix(NA,ncol=2,nrow=length(lista_imagens1))
matriz_resposta2<-matrix(NA,ncol=2,nrow=length(lista_imagens2))
matriz_resposta3<-matrix(NA,ncol=2,nrow=length(lista_imagens3))
matriz_resposta4<-matrix(NA,ncol=2,nrow=length(lista_imagens4))
colnames(matriz_resposta4)<-colnames(matriz_resposta3)<-
colnames(matriz_resposta2)<-colnames(matriz_resposta1)<-
c("DV","Agregacao")
```

```
rownames(matriz_resposta1)<-lista_imagens1
rownames(matriz_resposta2)<-lista_imagens2
rownames(matriz_resposta3)<-lista_imagens3
rownames(matriz_resposta4)<-lista_imagens4
```

Loop para análise das figuras em cada conjunto de imagens

```
for(i in 1:length(lista_matizes1)){
  matriz_resposta1[i,1]<-denseness_total(lista_matizes1[[i]])
  matriz_resposta1[i,2]<-aggregation_index(lista_matizes1[[i]])[1]
}

for(i in 1:length(lista_matizes2)){
  matriz_resposta2[i,1]<-denseness_total(lista_matizes2[[i]])
  matriz_resposta2[i,2]<-aggregation_index(lista_matizes2[[i]])[1]
}

for(i in 1:length(lista_matizes3)){
  matriz_resposta3[i,1]<-denseness_total(lista_matizes3[[i]])
  matriz_resposta3[i,2]<-aggregation_index(lista_matizes3[[i]])[1]
}

for(i in 1:length(lista_matizes4)){
  matriz_resposta4[i,1]<-denseness_total(lista_matizes4[[i]])
  matriz_resposta4[i,2]<-aggregation_index(lista_matizes4[[i]])[1]
}

resultado_final<-
rbind(matriz_resposta1,matriz_resposta2,matriz_resposta3,matriz_resposta4)
```