Tutorial GEE e R para processamento

Fulbert Gnonlonfoun

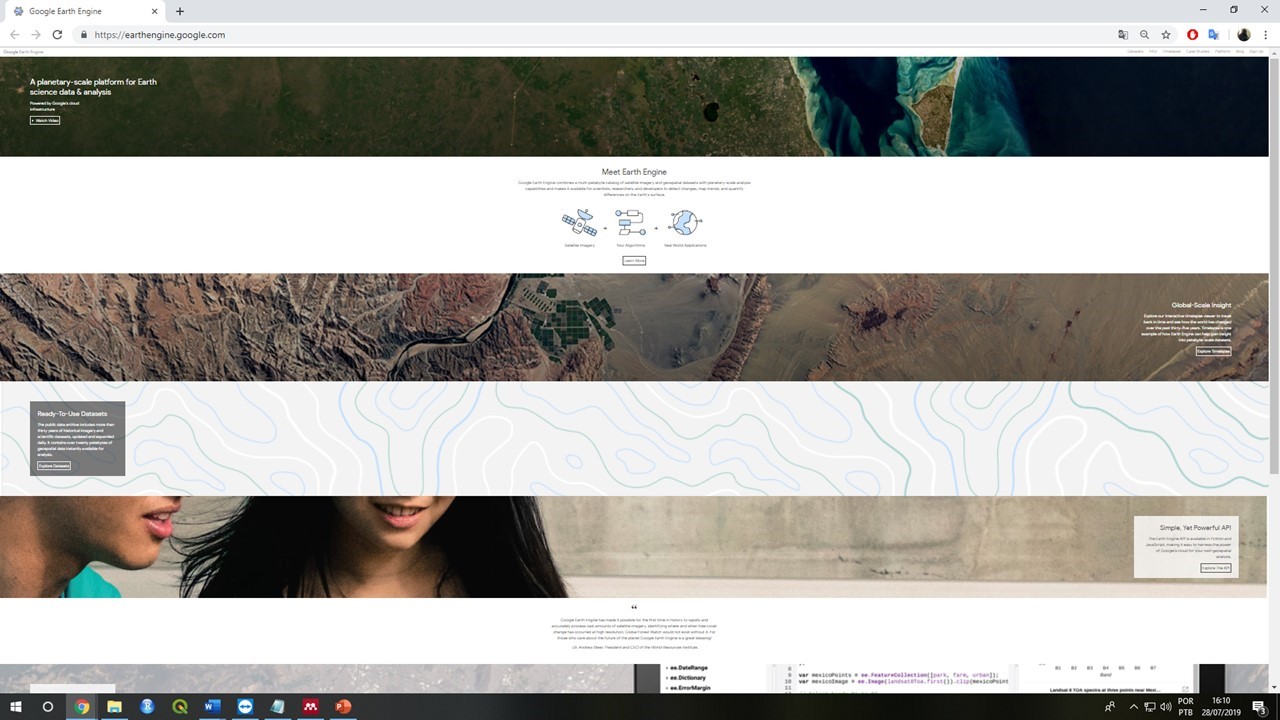
20/08/2019

## Desenvolvimento de documentação didática

Google Earth Engine, é um dos vários programas lancado pelo gigante da technologia Americano Google. Ele foi projetado para análise de dados ambientais em escala planetária. auxiliando na organização de informações geoespaciais para análise. Ele tem a particularidade de ser uma plataforma de alta qualidade na elaboração de processamento geoespacial de dados planetária baseada em nuvem (LI et al., 2019). Combinando mais de trinta anos de imagens de satélite e conjunto de dados geospacias atualizado a cada dia, disponibilizando múltiplas petabytes de imagens para professionais das áreas de geoprocessamento visualizar , manipular , editar e criar de maniera eficinte.Tendo como principal vantagem, a deteçao das mudanças, mapeamento das tendências e quantificação das diferenças na superfície da Terra. Para ter acesso a Google Earth Engine, vai precisar ter uma contar de gmail et preencher um formulário no site: signup.earthengine.google.com.

## Página de apresentação do Google Earth Engine:

Quando digite Google Earth Engine no google, a primeira pagina que aparece é a página de apresentação <https://earthengine.google.com/>, ele é uma galeria de imagens de mapas, de citações e de links que te levar a outras páginas de explicações (Figura 1). Esses links levam para vídeos de explicação, respostas para diversos perguntas , diversos aplicações, e guias.



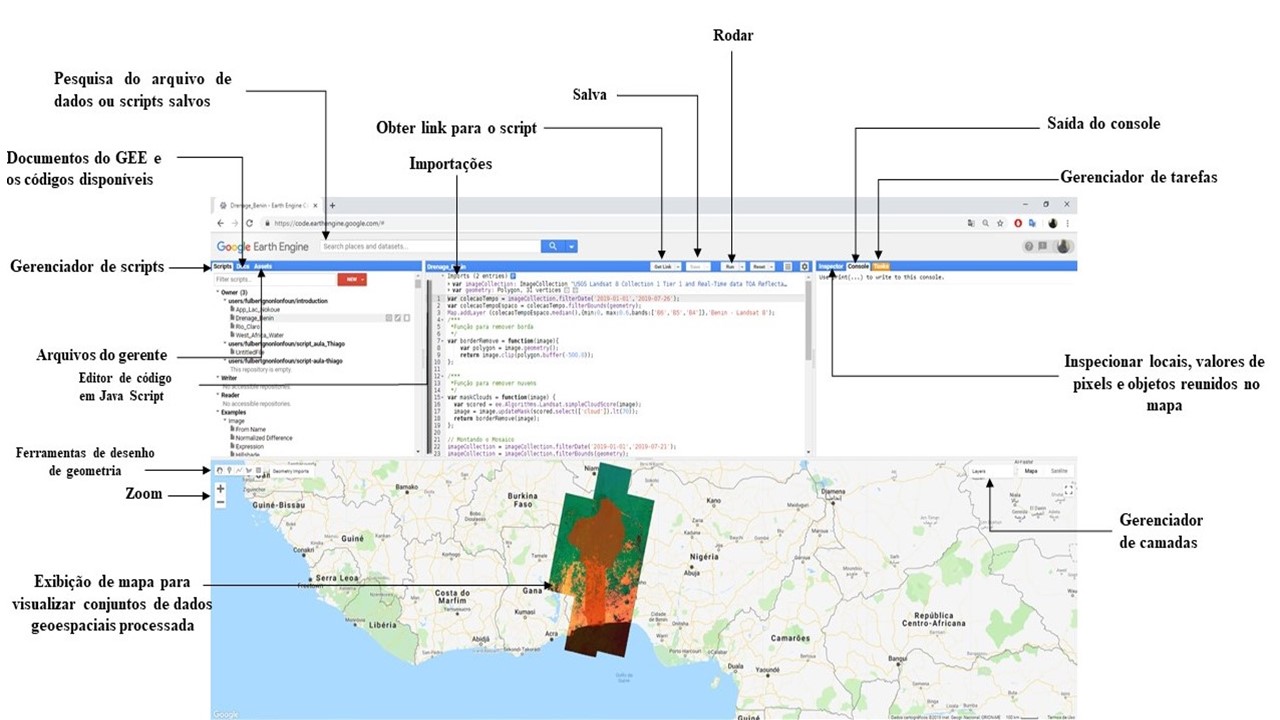
Pagina de apresentação GEE

### Editor de Códigos do Google Earth Engine:

O editor de Códigos Google Earth Engine, é baseado no site <https://code.earthengine.google.com/> , é a plataforma onde acontece as análises e visualização dos dados geoespaciais. Ela possui várias abas, tais como:

Editor de código JavaScript

Exibição de mapa para visualizar conjuntos de dados geoespaciais  
  
Documentação de referência da API (guia Documentos)  
  
Gerenciador de scripts (guia Scripts)  
  
Saída do console (guia console)  
  
Gerenciador de tarefas (guia Tarefas) para lidar com consultas demoradas  
  
Consulta de mapa interativo (guia Inspetor)  
  
Pesquisa do arquivo de dados ou scripts salvos  
  
Ferramentas de desenho de geometria como representado na figura 2.



Editor de Códigos GEE

### Aprender os princípios básicos do JavaScript

Google Earth Engine usa a linguagem de programação JavaScript. o que não falta é informações, e quando falamos de programação, a base se aprende sempre com os dados disponíveis na internet. Há vários suporte on-line para pode iniciar a aprendizagem do JavaScript. Mas como nosso foco não é a programação em si, vamos procurar tutorais no Google para JavaScript do Google Earth Engine. Mesmo assim, alguns sintaxe são importantíssimos quando se falar de JavaScript em GEE: Tudo as linhas de códigos em JavaScript termina sempre com um “;”, assim sendo, o código pode ser distribuído em varias linhas, mas no fim da última linha do código tem que colocar sempre um “;”.

Map.addLayer(parques);  
  
Map.addLayer(gfc);

Para definir novas variáveis, se usar:

var novo\_variável = ...

O “var” é sempre usado no GEE para criar uma nova camada, calcula alguns valores, importa novos conjuntos de dados. Porque o GEE precisa dessas variáveis para poder fazer qualquer tipo de operação como armazenamento, mapeamentos e exportações… Programando é sempre bom ter alguns comentários ou titulo para poder explicar o script ou o código redigido. No JavaScript para pode fazer isso é usado “//”.

// Montando o Mosaico

Em JavaScript, você tem que executar todo o seu script de uma só vez - ou seja, você não pode, por exemplo, selecionar duas linhas do seu script e executar apenas aquelas, você tem que executar a coisa toda. Você “executa” um script pressionando o Run botão. Isso significa que, durante todo o tutorial, à medida que você adiciona mais linhas ao seu script, é necessário continuar pressionando Run para ver os resultados do novo código que você adicionou.

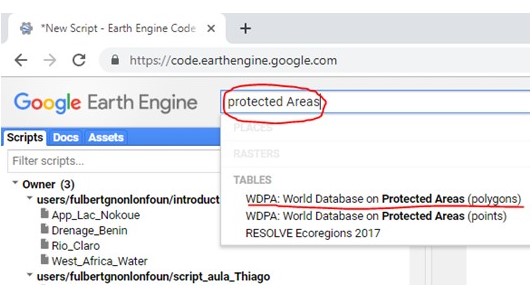


Rodar código no GEE

## Estudo de caso:

#### Como importar e explorar as mudanças da cobertura florestais nas áreas protegidas.

Nosso objetivo vai ser visualizar como as coberturas florestais mudarão ao longo dos tempos em quatros áreas protegidas de Benin e da Nigéria na África ocidental. Vamos começar tentando importar os dados das áreas que são protegidas no mundo. Para isso, vamos procurar na aba de pesquisa do GEE: protected area (polygono).



Importar Variaveis das Aréas protegidas

Depois de aparecer uma janela informativa sobre a variável, precisar ler para interder a variável e importar lá.



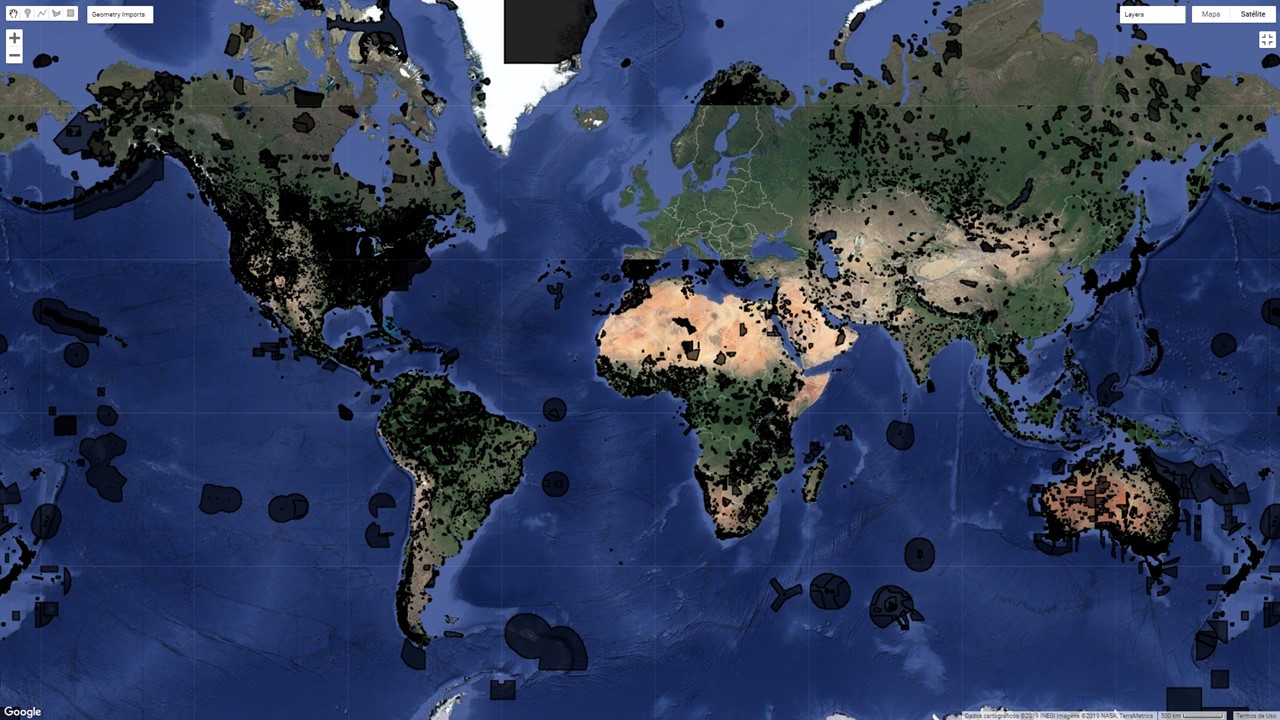
Importar uma variávels

Os dados importados vão aparece na parte superior da aba script. Vamos transforma o nome table em parques.

Variavel importado Vamos caregar a mapa.

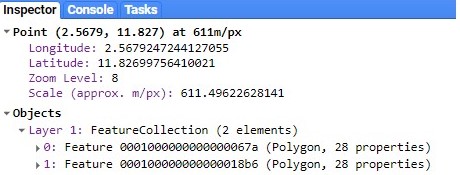
//vamos visualizar os parques do mundo  
Map.addLayer(parques);

Isso vai mapear os conjuntos de dados e adicionar as camadas dos parques do mundo.



Mapa dos parques do mundo

Na aba inspector, dando um click no mapa, podemos ter as informações geográficas daquele ponto em específica.



Inspector informado sobre as coordenadas geográficas de um ponto

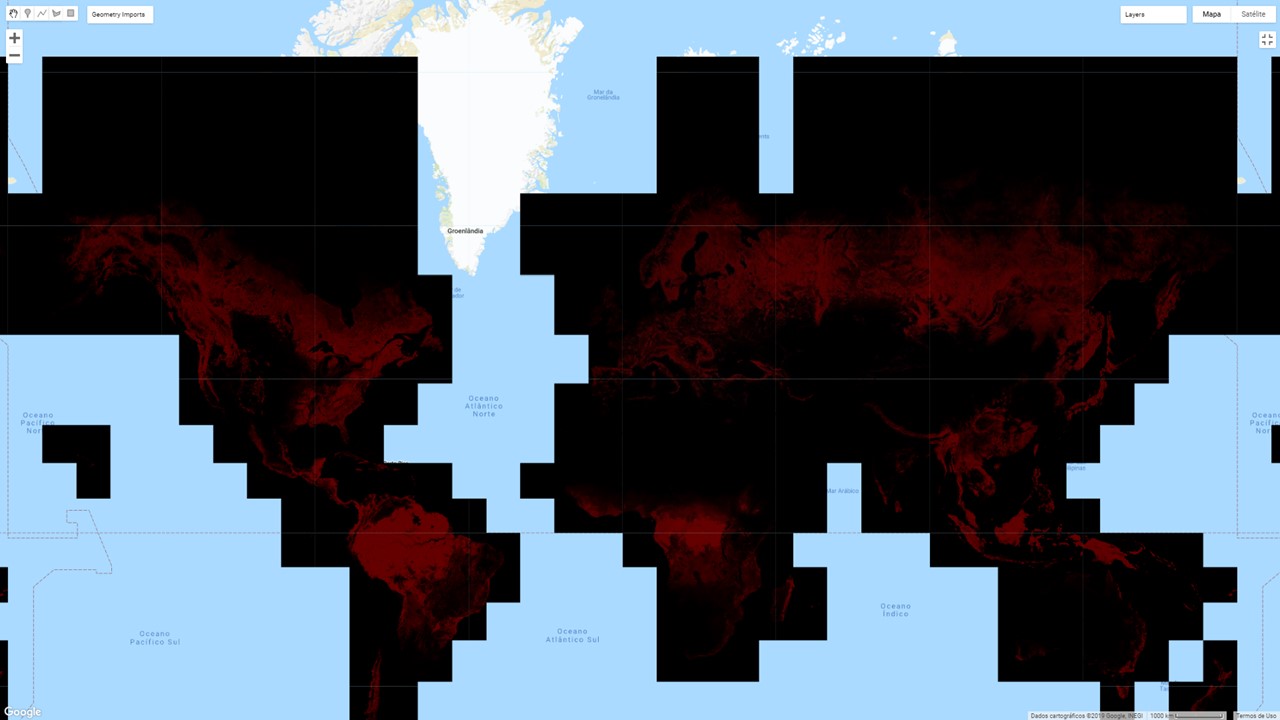
### Importação e exploração dos conjuntos de dados da cobertura florestal:

Do mesmo jeito que foi importado os dados das áreas protegidas, vamos importar os dados das mudanças globas das florestas dos anos 200 até 2018. Vamos digitar na Aba de pesquisa: Hansen Global Forest Change v1.6 (2000-2018), ler para inteder como funcionar e importar lo. Chama o objeto “Image” de gfc.

.

Agora, vamos adicionar o mapa Global Forest Change no nosso conjunto de dados.

//vamos mapear gfc  
Map.addLayer(gfc);



Mapa do Global forest change.

Portanto que temos um mapa, ele aparece em cor vermelha para onde ocorre floresta e a cor preta onde está ausente. E isso não ajudar em quase nada. Então vamos melhorar a mapa para uma fácil visualização.

### Visualização das mudanças da cobertura florestal

A escala padrão dos dados do Global forest Change é de 30m, então se um pixel sofre uma alteração ao longo de tempo, isso vai ser interpretado como uma perda ou um ganho da cobertura florestal. Seguindo essa lógica, pode ser definido uma escala para detectar automaticamente a resolução dos dados e usá-la como nosso escalar. No nosso script, vamos digitar o seguinte código:

// Definimos a escala padrão dos dados de Hansen que é de 30 m  
var scale = gfc.projection().nominalScale();

O passo seguinte vai ser a criação de variáveis para a cobertura da florestal dos anos 2000, para comparar com a perda da cobertura até 2018. E também criar variáveis para o ganho dos anos 2000 até 2018. E como sabemos, nos raster as imagens têm em geral três “bandas”: vermelho, verde, UV.

// vamos criar a variavel da cobertura original nos anos 2000  
var treeCover = gfc.select(['treecover2000']);  
  
// vamos converter as coberturas florestais em hectares  
treeCover = treeCover.divide(100);  
  
//vamos criar uma variavel "loss" para a perda de floresta  
var loss = gfc.select(['loss']);  
  
//vamos criar uma variavel "gain" para ganho de florestas  
var gain = gfc.select(['gain']);

Vamos mapear a cobertura florestal global com a perda e o ganho de floresta ao longo de 18anos. Nós vamos usar a função Map.addLayer de novo para adicionar as três variáveis; floresta original em 2000, a perda e o ganho até 2018.

// Adiciona a camada de cobertura de árvore em cinza claro  
Map.addLayer(treeCover.updateMask(treeCover),  
 {palette: ['D0D0D0', '00FF00'], max: 100}, 'Forest Cover');  
   
// Adiciona a camada de perda em vemelho  
Map.addLayer(loss.updateMask(loss),  
 {palette: ['#fd0a4d']}, 'Loss');  
   
// Adiciona a camada de ganho em amarelo  
Map.addLayer(gain.updateMask(gain),  
 {palette: ['#f3f700']}, 'Gain');

Nas imagens, podemos observar claramente, onde teve perda e ganho florestal no globo. Assim, tivemos uma extensão clara das mudanças florestal, mas não sobre a magnitude dessas mudanças. Para isso, vamos converter o numero de pixel quem tem de colar valores 0 para ausente e 1 para presença ou ocorrência de floresta em km². Para cada uma das 3 variáveis criado anteriormente, vamos criar valores correspondente em outras unidades, saindo de ha (hectares) para km². Vamos usar o código seguinte:

// As unidades das variáveis são números de pixels  
// Aqui estamos convertendo os pixels na área real  
// Dividindo por 10 000 para que o resultado final seja em km²  
var areaCover = treeCover.multiply(ee.Image.pixelArea())  
 .divide(10000).select([0],["areacover"]);  
   
var areaLoss = loss.gt(0).multiply(ee.Image.pixelArea()).multiply(treeCover)  
 .divide(10000).select([0],["arealoss"]);  
   
var areaGain = gain.gt(0).multiply(ee.Image.pixelArea()).multiply(treeCover)  
 .divide(10000).select([0],["areagain"]);

Como quantificar e calcular o ganho e a perda total da cobertura florestal de locais em específicos: Nosso objetivo inicial era extrair valores de dados geoespaciais para 4 parques do Benin e da Nigéria na África ocidental. Para responder a essa pergunta, precisamos calcular quantas mudanças ocorreu apenas em esses 4 parques. Para conseguir esses dados, vamos filtra nossas 4 áreas de interesse. Com os seguintes códigos:

// Vamos Cria uma variável que tenha os polígonos por apenas alguns parques do Benin e Nigéria  
var parques = parques.filter(ee.Filter.or(  
 ee.Filter.eq("NAME", "Boucle de la Pendjari"),  
   
 ee.Filter.eq("NAME", "W"),  
   
 ee.Filter.eq("NAME", "Kambari"),  
   
 ee.Filter.eq("NAME", "Kashimbila")));

Esses processamentos, nos dar a possibilidade de calcular as áreas de perda e de ganho dos nossos parques usando a função “reducer”. Vamos também armazenar nossos resultados em duas novas variáveis statsLosse statsGain.

// Soma dos valores dos pixels de perda.  
var statsLoss = areaLoss.reduceRegions({  
 reducer: ee.Reducer.sum(),  
 collection: parques,  
 scale: scale  
});  
  
// Soma dos valores dos pixels de ganho.   
var statsGain = areaGain.reduceRegions({  
 reducer: ee.Reducer.sum(),  
 collection: parques,  
 scale: scale  
});

### Exportação dos resultados e das imagens:

Para poder analisar as perdas e os ganhos das vegetaçoes florestais, vamos exportar os arquivos“.csv.” no google drive, Usaremos o seguinte código:

// vamos exportar nossos dados no drive  
Export.table.toDrive({  
 collection: statsLoss,  
 description: 'NP\_forest\_loss'});  
   
Export.table.toDrive({  
 collection: statsGain,  
 description: 'NP\_forest\_gain'});

para analisar variação das vegetaçao florestal, vamos primeiramente delimitar a geometria do Bénin e da nigéria,

.

Vamos baixar as imagens na pasta mosaico no Google Drive. Usaremos o seguinte código:

//Vamos esportar imagens em TIF  
Export.image.toDrive({  
 image:areaGain,  
 description: 'ParquesGain',  
 folder: 'mosaico',  
 maxPixels:10000000000000,  
 scale: 30,  
 crs:'EPSG:4326',  
 region: Benin\_Nigeria  
   
});  
  
Export.image.toDrive({  
 image:areaLoss,  
 description: 'ParquesLoss',  
 folder: 'mosaico',  
 maxPixels:10000000000000,  
 scale: 30,  
 crs:'EPSG:4326',  
 region: Benin\_Nigeria});

### Analise dos dados raster abaixado do GEE

Primeiro passo vamos instalar e carregar os pacotes que podem processar as dados geospaciais. R também contém muitas funções que permitem processar, visualizar e analisar dados espaciais. Eles estão contidos em vários pacotes (conjuntos de funções), entre outros:GIS - sp, rgdal, rgeos, sf, raster,caret e caretEnsemble estrelas, viridis, rasterVis Geoestatística gstat, geoR, geoRglm. Vamos instalar alguns desses pacotes e carregar los.

# Load packages  
library(sp)  
library(rgdal)

## rgdal: version: 1.4-4, (SVN revision 833)  
## Geospatial Data Abstraction Library extensions to R successfully loaded  
## Loaded GDAL runtime: GDAL 2.2.3, released 2017/11/20  
## Path to GDAL shared files: C:/Program Files/R/R-3.6.1/library/rgdal/gdal  
## GDAL binary built with GEOS: TRUE   
## Loaded PROJ.4 runtime: Rel. 4.9.3, 15 August 2016, [PJ\_VERSION: 493]  
## Path to PROJ.4 shared files: C:/Program Files/R/R-3.6.1/library/rgdal/proj  
## Linking to sp version: 1.3-1

library(raster)  
library(ggplot2)  
library(viridis)

## Loading required package: viridisLite

library(rasterVis)

## Loading required package: lattice

## Loading required package: latticeExtra

## Loading required package: RColorBrewer

##   
## Attaching package: 'latticeExtra'

## The following object is masked from 'package:ggplot2':  
##   
## layer

Depois Disso, vamos definir nosso diretor onde já foram abaixado previamente as imagens .tif do google drive.

setwd("D:/fulbert/02\_Rstudio/Iniciaçao\_r/00\_Script")

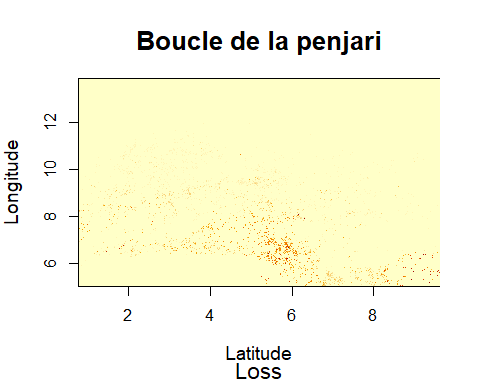
As imagens abaixadas são oito (08) no total. Para cada um, vamos criar um objeto correspondente ao seu nome. Por exemplo. depois Vamos chamar o objeto criado para ver os diferentes componentes dele.

# loss 1  
Parques\_loss1 <- raster('ParquesLoss\_1.tif')  
Parques\_loss1

## class : RasterLayer   
## dimensions : 32768, 32768, 1073741824 (nrow, ncol, ncell)  
## resolution : 0.0002694946, 0.0002694946 (x, y)  
## extent : 0.7766834, 9.607482, 5.050059, 13.88086 (xmin, xmax, ymin, ymax)  
## crs : +proj=longlat +datum=WGS84 +no\_defs +ellps=WGS84 +towgs84=0,0,0   
## source : D:/fulbert/02\_Rstudio/Iniciaçao\_r/00\_Script/ParquesLoss\_1.tif   
## names : ParquesLoss\_1

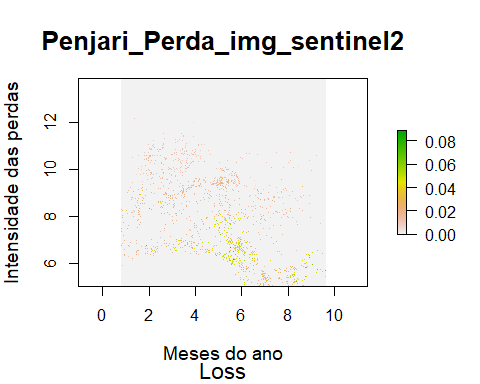
Vamos chamar as imagens:

image(Parques\_loss1,   
 main="Boucle de la penjari",  
 sub="Loss",  
 xlab = "Latitude",  
 ylab = "Longitude",  
 col.main="black", col.lab="black", col.sub="black",  
 cex.main=1.6, cex.lab=1.2, cex.sub=1.4)



Agorar vamos plotar esses imagens para ter uma idea clara da sua representaçao.

plot(Parques\_loss1,  
 main="Penjari\_Perda\_img\_sentinel2",  
 sub="Loss",  
 xlab = "Meses do ano",  
 ylab = "Intensidade das perdas",  
 col.main="black", col.lab="black", col.sub="black",  
 cex.main=1.6, cex.lab=1.2, cex.sub=1.4)



### Estabelecimento e documentação de procedimentos para ingestão e manipulação de dados derivados de sensoriamento remoto em formatos raster e vetorial utilizando-se de rotinas na linguagem de programação R:



**limguagem R**



**software R studio**

Segundo O manual do R1, o R é definido como um ambiente de programação, com um conjunto integrado de ferramentas para manipulação de dados, cálculos e apresentação gráfica **(VENABLES; SMITH, 2010)**. Vale precisar que o R é a linguagem de programação e o Rstudio é o software.

Mas porque usar o R? R além de ser um programa gratuita e poderosa, está se tornando a linguagem padrão em projetos de Data Science. Ele vem se tornando uns dos principais “idiomas” de Cientistas e Analistas de Dados e está liderando a revolução proporcionada pelo Big Data Analytics. A linguagem R é simplesmente a melhor linguagem estatística disponível hoje para análise de Data e se tornando quase um pré-requisito para pós-graduação e consultorias na área de Ecologia. Então nós vamos usar esse poderoso programa para analisar nossos dados abaixado do google Earth engine.

Analise dos dados .csv abaixado do GEE: Primeiro passo vamos instalar e carregar os pacotes que podem processar os nossos dados:

library(ggplot2)  
library(ggthemr) # para definir um tema personalizado  
library(forcats) #para reordenar variáveis categóricas

agora nos vamos abaixar os dados Excel.csv do google drive e carregar o repositório no Rstudio.

setwd("D:/fulbert/02\_Rstudio/Iniciaçao\_r/00\_Script")

vamos nomear os dados com os nome originais dos aquivos

NP\_forest\_gain <- read.csv("NP\_forest\_gain.csv")  
NP\_forest\_loss <- read.csv("NP\_forest\_loss.csv")

vamos Criar coluna identificadora para ganhos e para perdas

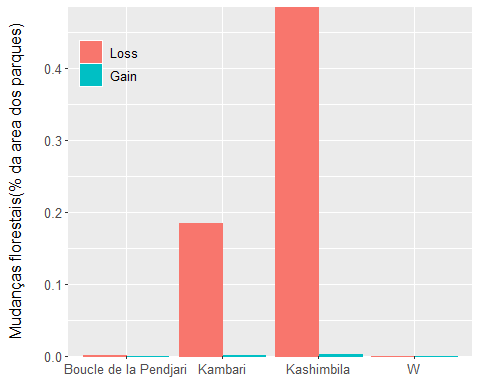
NP\_forest\_gain$type <- "Gain"  
NP\_forest\_loss$type <- "Loss"

vamos juntas os dois objetos ganhos e perdars

forest\_change <- rbind(NP\_forest\_gain, NP\_forest\_loss)

Isso nos permitira de ter uma visão global dos objetos plotados

(forest\_barplot <- ggplot(forest\_change, aes(x = NAME, y = sum/GIS\_AREA,   
 colour = fct\_rev(type),   
 fill = fct\_rev(type))) +  
 geom\_bar(stat = "identity", position = "dodge") +  
 labs(x = NULL, y = "Mudanças florestais(% da area dos parques)\n") +  
 # A expansão da escala remove o espaço vazio abaixo das barras  
 scale\_y\_continuous(expand = c(0, 0)) +  
 theme(text = element\_text(size = 12), # tamanho de fonte varia com o numero escolhido  
 legend.position = c(0.1, 0.85), # muda o posicionamento da legenda  
 legend.title = element\_blank(), # título da legenda  
 legend.background = element\_rect(color = "black",   
 fill = "transparent", # remove o fundo branco por trás da legenda  
 linetype = "blank")))



O codigo em baixo vai nos permitir de salva a plot em .png

ggsave(forest\_barplot, filename = "forestas\_barplot.png",  
 height = 5, width = 7)