

Using NDVI to assess vegetation damage after wildfires

Cato Ceciel Brinkhof

UniBo

30-01-2025

Introduction

Using sentinel 2 data to analyse the effects of two wildfires on vegetation health

- **Location:** Deurnse Peel, Brabant, the Netherlands
- **Habitat type:** swamps, shrubs, forests, and peatlands
- **Fire 1:** 20 april 2020 - 16 june 2020, 710 hectares damaged
- **Fire 2:** 31 august 2022, 41 hectares damaged

Data collection

The data is collected from the Copernicus Dataspace Browser. We use Sentinel-2 data. Throughout this project we use the following packages:

```
13 ## Step 1: Retrieve required packages from library
14 library(terra) #the terra package is for importing tiffs, and to run imageRy
15 library(imageRy) # imageRy is used to visualize satellite data.
16 library(raster) #used for plotting
17 library(viridis) #used to apply a categorized version of the viridis colour scale
18 library(graphics) #used for the captions of the plots
```

Figure: Code for importing packages

Data collection

The `rast()` function, from the `terra` package, is used to import the tiff images we need.

```
# One year before fire 1: 2019-04-18
b02_20190418 <- rast("2019-04-18-00_00_2019-04-18-23_59_Sentinel-2_L1C_B02_(Raw).tiff")
b03_20190418 <- rast("2019-04-18-00_00_2019-04-18-23_59_Sentinel-2_L1C_B03_(Raw).tiff")
b04_20190418 <- rast("2019-04-18-00_00_2019-04-18-23_59_Sentinel-2_L1C_B04_(Raw).tiff")
b08_20190418 <- rast("2019-04-18-00_00_2019-04-18-23_59_Sentinel-2_L1C_B08_(Raw).tiff")

#A few days before fire 1: 2020-09-12
b02_20200417 <- rast("2020-04-17-00_00_2020-04-17-23_59_Sentinel-2_L1C_B02_(Raw).tiff")
b03_20200417 <- rast("2020-04-17-00_00_2020-04-17-23_59_Sentinel-2_L1C_B03_(Raw).tiff")
b04_20200417 <- rast("2020-04-17-00_00_2020-04-17-23_59_Sentinel-2_L1C_B04_(Raw).tiff")
b08_20200417 <- rast("2020-04-17-00_00_2020-04-17-23_59_Sentinel-2_L1C_B08_(Raw).tiff")

#Immediately after fire 1: 2020-06-24
b02_20200624 <- rast("2020-06-24-00_00_2020-06-24-23_59_Sentinel-2_L1C_B02_(Raw).tiff")
b03_20200624 <- rast("2020-06-24-00_00_2020-06-24-23_59_Sentinel-2_L1C_B03_(Raw).tiff")
b04_20200624 <- rast("2020-06-24-00_00_2020-06-24-23_59_Sentinel-2_L1C_B04_(Raw).tiff")
b08_20200624 <- rast("2020-06-24-00_00_2020-06-24-23_59_Sentinel-2_L1C_B08_(Raw).tiff")
```

Figure: Code for data. This step is repeated for nine dates in total: 18-04-2019, 17-04-2020, 24-06-2020, 27-04-2021, 17-04-2022, 25-08-2022, 12-09-2022, 05-04-2023, and 29-04-2024

Data preparation

Now we stack our .tiff files (one file per band per date) into stack per date. This enables us to use `im.plotRGB` (package: `imageRy`) efficiently. We also create a string of captions, corresponding to each stack of bands.

```
sent_20190418 <- c(b02_20190418, b03_20190418, b04_20190418, b08_20190418)
sent_20200417 <- c(b02_20200417, b03_20200417, b04_20200417, b08_20200417)
sent_20200624 <- c(b02_20200624, b03_20200624, b04_20200624, b08_20200624)
sent_20210427 <- c(b02_20210427, b03_20210427, b04_20210427, b08_20210427)
sent_20220417 <- c(b02_20220417, b03_20220417, b04_20220417, b08_20220417)
sent_20220825 <- c(b02_20220825, b03_20220825, b04_20220825, b08_20220825)
sent_20220912 <- c(b02_20220912, b03_20220912, b04_20220912, b08_20220912)
sent_20230405 <- c(b02_20230405, b03_20230405, b04_20230405, b08_20230405)
sent_20240429 <- c(b02_20240429, b03_20240429, b04_20240429, b08_20240429)

#Next we create a list of the images and the corresponding dates. These will be the captions.
sent_list <- list(sent_20190418, sent_20200417, sent_20200624, sent_20210427,
                 sent_20220417, sent_20220825, sent_20220912, sent_20230405, sent_20240429)
captions <- c("18-04-19: 1 year before fire 1", "17-04-20: right before fire 1", "24-06-20: right after fire 1",
              "27-04-2021: 1 year after fire 1", "17-04-22: 2 years after fire 1", "25-08-22: right before fire 2",
              "12-09-22: right after fire 2", "05-04-23: 3 years after fire 1", "29-04-24: 4 years after fire 1")
```

Figure: Code for stacking and listing all our data

Creating a natural colour image

The first thing we do is to create a natural colour image. To do this we use `im.plotRGB`. Some details:

- Since the exact step is repeated nine times, once per date, we make a `for()` loop for the step.
- A natural colour image consists of bands 4, 3, and 2, as red, green, and blue, respectively.
- Given that band 1 is not in our list, band 4 becomes 3, 3 becomes 2, and 2 becomes 1.

Code for natural colour images

```
par(mfrow = c(3, 3), mar = c(5,4,4,4), oma=c(5,10,5,10))  
for (i in seq_along(sent_list)) {  
  im.plotRGB(sent_list[[i]], r = 3, g = 2, b = 1)  
  mtext(captions[i], side = 1, line = 4, cex = 0.5, font = 3, col = "black")  
}
```

Figure: The loop that creates nine natural colour images

Natural colour images



Figure: Natural colour images before- and after fires 1 and 2

NDVI

The natural colour image does not give us that much information. As an alternative, we try the Normalized Difference Vegetation Index (NDVI). With sentinel-2 data, NDVI is given as:

$$NDVI = (B8 - B4)/(B8 + B4) \quad (1)$$

$$NDVI = (NIR - RED)/(NIR + RED) \quad (2)$$

We write a function to do this in R:

NDVI in R

```
#We make a function for calculating the NDVI for every year, as follows:
NDVI <- function(x,y){
  NDVI=((x-y)/(x+y))
  return(NDVI)
}

#Using the function:
NDVI_20190418 <- NDVI(b08_20190418, b04_20190418)
NDVI_20200417 <- NDVI(b08_20200417, b04_20200417)
NDVI_20200624 <- NDVI(b08_20200624, b04_20200624)
NDVI_20210427 <- NDVI(b08_20210427, b04_20210427)
NDVI_20220417 <- NDVI(b08_20220417, b04_20220417)
NDVI_20220825 <- NDVI(b08_20220825, b04_20220825)
NDVI_20220912 <- NDVI(b08_20220912, b04_20220912)
NDVI_20230405 <- NDVI(b08_20230405, b04_20230405)
NDVI_20240429 <- NDVI(b08_20240429, b04_20240429)
```

Figure: Code for calculating NDVI in R

NDVI visualisation

```
ndvi_list <- list(NDVI_20190418, NDVI_20200417, NDVI_20200624, NDVI_20210427,  
                 NDVI_20220417, NDVI_20220825, NDVI_20220912, NDVI_20230405,  
                 NDVI_20240429)  
ndvi_min <- min(sapply(ndvi_list, function(x) min(values(x), na.rm = TRUE)))  
ndvi_max <- max(sapply(ndvi_list, function(x) max(values(x), na.rm = TRUE)))  
  
#Again, we make the plots with a loop. We use ndvi_min and ndvi_max to determine the range.  
dev.off()  
par(mfrow = c(3, 3), mar = c(5, 4, 2, 1))  
for (i in seq_along(ndvi_list)) {  
  plot(ndvi_list[[i]], axes=TRUE, range=c(ndvi_min, ndvi_max))  
  mtext(captions[i], side = 1, line = 4, cex = 0.8, font = 1, col = "black", outer = FALSE)  
}
```

Figure: Code for visualising the NDVI of each date

NDVI visualisation

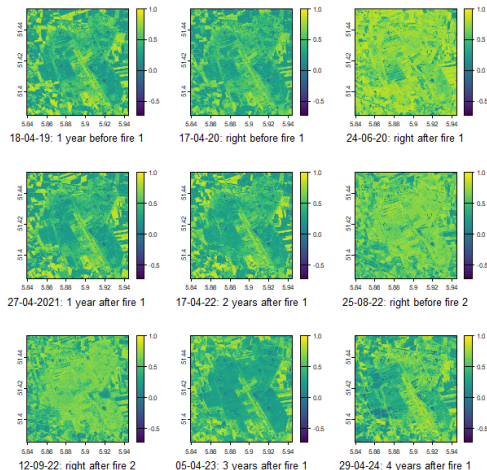


Figure: NDVI for each date

Calculating the changes in NDVI values

Now it is time to compare the before- and afters for each fire. We make five comparisons:

- **Before/after fire 1:** April 2020 versus April 2021
- **Before/after fire 1, no outliers:** Average April 2019/20 versus Average April 2021/22
- **Before/after fire 2:** April 2022 versus April 2023
- **Before/after fire 2, no outliers:** Average April 2021/22 versus Average April 2023/24
- **Before/after both fires, no outliers** Average April 2019/20 versus Average April 2023/24

Code for calculating NDVI changes

```
147 #Now it is time to do the actual analysis
148 #Question 1: Assessing the damage after fire no. 1
149 #Comparison 1: april 2020 versus april 2021
150 change_20200417_20210427 <- NDVI_20210427-NDVI_20200417
151
152 #Avoiding outliers, comparison 1.1: average of april 2019 and 2020 versus april 2021 and april 2022
153 change_avg20192020_avg20212022 <- ((NDVI_20210427+NDVI_20210427)/2)-((NDVI_20190418+NDVI_20200417)/2)
154
155 #Question 2: Assessing the additional damage from fire no. 2
156
157 #Comparison 2.1: august 2022 v september 2022
158 change_20220825_20220912 <- NDVI_20220912-NDVI_20220825
159
160 #Comparison 2.2: april 2022 versus april 2023
161 change_20220417_20230405 <- NDVI_20230405-NDVI_20220417
162
163 #Avoiding outliers, comparison 2.3: average of april 2021 and 2022 versus april 2023 and 2024
164 change_avg20212022_avg20232024 <- ((NDVI_20230405+NDVI_20240429)/2)-((NDVI_20210427+NDVI_20220417)/2)
165
166 #Question 3: To what extent has the vegetation recovered four years later?
167 #Comparison 3
168 change_avg20192020_avg20232024 <- ((NDVI_20230405+NDVI_20240429)/2)-((NDVI_20190418+NDVI_20200417)/2)
169
```

Figure: Code for calculating the difference in NDVI values

Code for plot of all NDVI changes

This is the code for plotting all five images. It follows the same structure as before, when we plotted nine images of individual NDVI values.

```

170 #Plotting everything as we did before
171 #Making a list to allow the loop and to determine the min and max values for the color ramp
172 change_list <- list(change_20200417_20210427, change_avg20192020_avg20212022,
173                    change_20220417_20230405, change_avg20212022_avg20232024,
174                    change_avg20192020_avg20232024)
175 change_min <- min(sapply(change_list, function(x) min(values(x), na.rm = TRUE))))
176 change_max <- max(sapply(change_list, function(x) max(values(x), na.rm = TRUE))))
177
178 #Captions
179 change_captions <- c("NDVI change after fire 1, april 2020 versus april 2021",
180                    "change in average NDVI 2019/20 versus 2021/22",
181                    "NDVI change after fire 2, april 2022 versus april 2023",
182                    "change in average NDVI 2021/22 versus 2023/24",
183                    "change in total average NDVI: 2019/20 versus 2023/24")
184
185 #Putting the plot together
186 dev.off()
187 par(mfrow = c(5, 1), mar = c(5, 4, 2, 1))
188 for (i in seq_along(change_list)) {
189   plot(change_list[[i]], axes=TRUE, range=c(change_min, change_max))
190   mtext(change_captions[i], side = 1, line = 4, cex = 0.8, font = 1,
191         col = "black", outer = FALSE)
192 }
193

```

Figure: Code for plot of all NDVI changes

NDVI changes plot



NDVI change after fire 1, april 2020 versus april 2021



Change in average NDVI 2019/20 versus 2021/22



NDVI change after fire 2, april 2022 versus april 2023



Change in average NDVI 2021/22 versus 2023/24



Change in total average NDVI: 2019/20 versus 2023/24

Figure: Changes in NDVI

Categorical changes

This is a bit difficult to read. As an alternative we make the plots categorical: this is easier for the human eye. The min and max values are artificially symmetrical: this makes it easier to visualize positives and negatives.

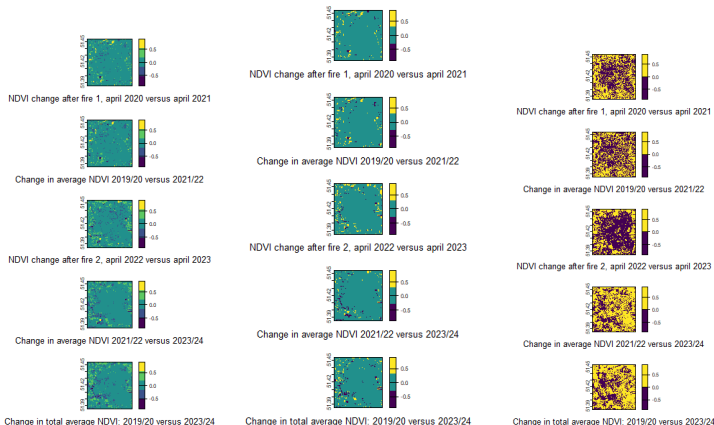
```

8 par(mfrow = c(5, 1), mar = c(5, 4, 2, 1))
9 for (i in seq_along(change_list)) {
10   plot(change_list[[i]], col=viridis(5), axes=TRUE, range=c(-0.9, 0.9))
11   mtext(change_captions[i], side = 1, line = 4, cex = 0.8, font = 1, col = "black", outer = FALSE)
12 }
13
14 #Then another one with only 3 categories, to make it clearer.
15 dev.off()
16 par(mfrow = c(5, 1), mar = c(5, 4, 2, 1))
17 for (i in seq_along(change_list)) {
18   plot(change_list[[i]], col=viridis(3), axes=TRUE, range=c(-0.9, 0.9))
19   mtext(change_captions[i], side = 1, line = 4, cex = 0.8, font = 1, col = "black", outer = FALSE)
20 }
21
22 #Finally, only 2 categories
23 dev.off()
24 par(mfrow = c(5, 1), mar = c(5, 4, 2, 1))
25 for (i in seq_along(change_list)) {
26   plot(change_list[[i]], col=viridis(2), axes=TRUE, range=c(-0.9, 0.9))
27   mtext(change_captions[i], side = 1, line = 4, cex = 0.8, font = 1, col = "black", outer = FALSE)
28 }
29

```

Figure: Code to make the colour scheme categorical: 5, 3, and then 2 categories.

Categorical changes



Counting values

To be certain we also calculate the percentage of negative values for each change tile. This is done as follows:

```
values_change_1 <- values(change_20200417_20210427)
increase_1 <- length(which(values_change_1>0))
decrease_1 <- length(which(values_change_1<=0))
perc_decrease_1 <- (decrease_1/(decrease_1+increase_1))*100

values_change_2 <- values(change_avg20192020_avg20212022)
increase_2 <- length(which(values_change_2>0))
decrease_2 <- length(which(values_change_2<=0))
perc_decrease_2 <- (decrease_2/(decrease_2+increase_2))*100

values_change_3 <- values(change_20220417_20230405)
increase_3 <- length(which(values_change_3>0))
decrease_3 <- length(which(values_change_3<=0))
perc_decrease_3 <- (decrease_3/(decrease_3+increase_3))*100

values_change_4 <- values(change_avg20212022_avg20232024)
increase_4<- length(which(values_change_4>0))
decrease_4<- length(which(values_change_4<=0))
perc_decrease_4 <- (decrease_4/(decrease_4+increase_4))*100

values_change_5 <- values(change_avg20192020_avg20232024)
increase_5<- length(which(values_change_5>0))
decrease_5<- length(which(values_change_5<=0))
perc_decrease_5 <- (decrease_5/(decrease_5+increase_5))*100
```

Figure: Code for calculating percentage showing negative NDVI change

Observed value changes

We get the following results:

- **April 2020 versus April 2021:** 50.5 percent was negative
- **Average April 2019/20 versus April 2021/22:** 53 percent was negative
- **April 2022 versus April 2023:** 66 percent was negative
- **Average April 2021/22 versus April 2023/24:** 39 percent was negative
- **Average April 2019/20 versus April 2023/24:** 38 percent was negative

Two possible conclusions: (1) vegetation recovers quickly (2) NDVI is not suitable for this type of land (e.g. lot of water)

Sources

- Copernicus Sentinel data [2025]. Retrieved from Copernicus Browser [24 01 2025], processed by ESA.
- RStudio version 1.4.1106
- R version 4.4.2