Quality Assessment of MODIS IV data

AJ Perez-Luque (@ajpelu)

2017 March

Filter data

```
library("tidyverse")
library("knitr")
library("binaryLogic")
library("corrplot")
source(paste0(di,"/script/R/getComposite.R"))
```

```
# Read data
rawdata <- read.csv(file=paste(di, "/data_raw/evi/iv_qp_raw_qa_2017_aug.csv", sep= ""), header = TRUE, s
```

Prepare Raw Data

- Raw data come from GEE script (see `/script/GEE/get_iv_modis.qp.js`
- Date:
  - Get date of the image (from hdf title, system.index)
  - Store as date (date format) and create new variable for year
- Select and rename variables of interest

```
raw <- rawdata %>%
  mutate(
    # Date
    date = as.Date(substr(system.index,1,10), format = "%Y_%m_%d"),
    # date2 = as.Date(substr(system.index,1,10), format = "%Y_%m_%d"),
    year = lubridate::year(date),

    # GEE index
    gee_index = stringr::str_replace(
      substr(system.index,22, nchar(as.character(system.index))),
    pattern = '_', "")) %>%
    # test = ifelse(date == date2, 0, 1))
  dplyr::select(doy = DayOfYear, evi = EVI, ndvi = NDVI, summQA = SummaryQA, iv_malla_modi_id,
                pop, date, year, gee_index, lat, long,
                qa_quality, qa_use, qa_aerosol, qa_adj_cloud,
                qa_atmos, qa_mix_cloud, qa_landwater, qa_snow, qa_shadow)
```

Some metadata of the time series

- Temporal range of the time series

```r
# Get temporal range of the data
# Start date
unique(min(as.Date(raw$date)))
```

```
## [1] "2000-02-18"
```

```r
# End date
unique(max(as.Date(raw$date)))
```

```
## [1] "2016-12-18"
```

- Number of images per year

```r
# See n of images per year and per pixel
n_images_pixel <- raw %>%
  mutate(year = lubridate::year(date)) %>%
  group_by(year) %>%
  summarise(n = n(),
            n_pixel = n()/length(unique(iv_malla_modi_id)))

kable(n_images_pixel)
```

year	n	n_pixel
2000	18560	20
2001	21344	23
2002	21344	23
2003	21344	23
2004	21344	23
2005	21344	23
2006	21344	23
2007	21344	23
2008	21344	23
2009	21344	23
2010	21344	23
2011	21344	23
2012	21344	23
2013	21344	23
2014	21344	23
2015	21344	23
2016	21344	23

- Summary table of layer summaryQA

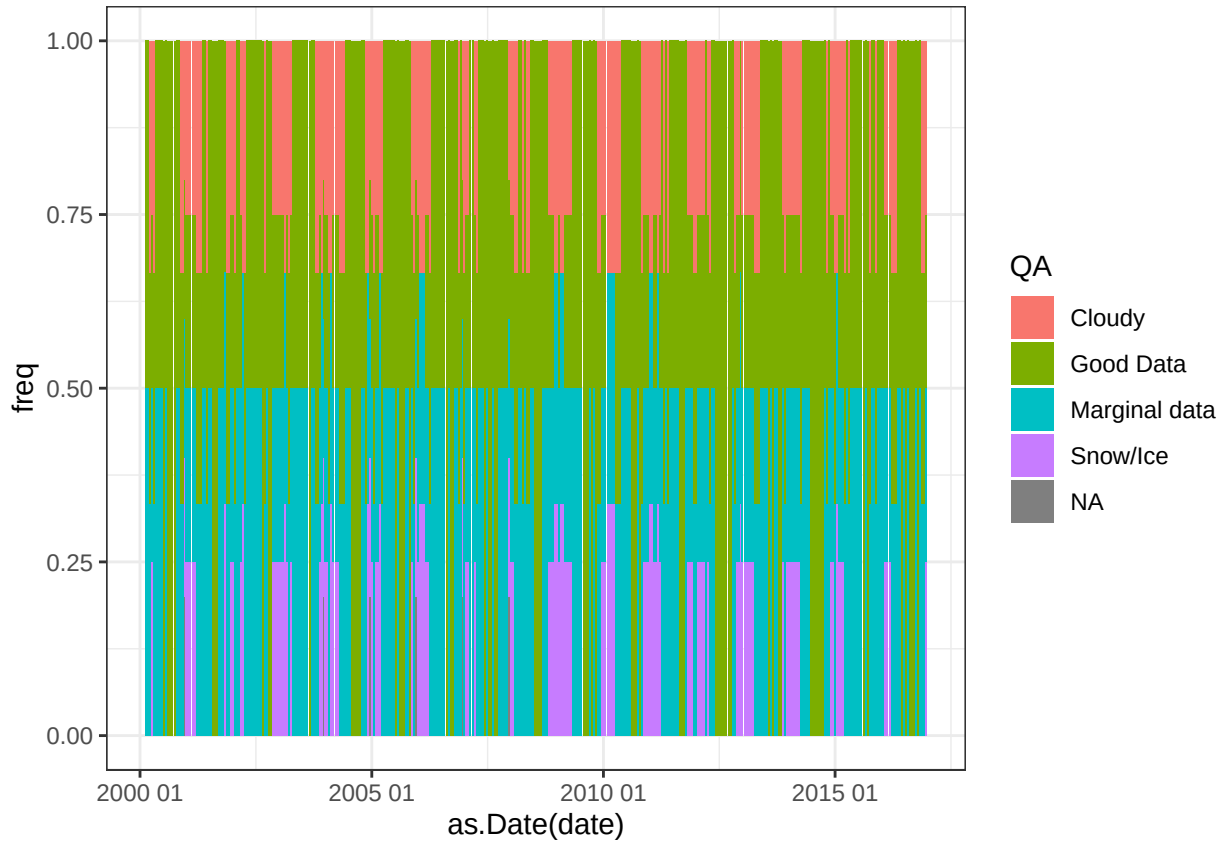
```r
# Table of QA
raw %>% group_by(summQA) %>%
  summarise(npixels = n()) %>%
  mutate(freq = round((npixels / sum(npixels)*100),2),
         QA = plyr::mapvalues(summQA, c(0, 1, 2, 3), c("Good Data", "Marginal data", "Snow/Ice", "Cloud
  kable()
```

summQA	npixels	freq	QA
0	208437	57.89	Good Data
1	116411	32.33	Marginal data
2	9268	2.57	Snow/Ice
3	25504	7.08	Cloudy
NA	444	0.12	NA

QA

- Explore temporal distribution of QA

```
qadate <- raw %>%
  group_by(date, summQA) %>%
  count(date, summQA) %>%
  mutate(freq = round((n / sum(n))*100, 2),
         QA = plyr::mapvalues(summQA, c(0, 1, 2, 3),
                              c("Good Data", "Marginal data", "Snow/Ice", "Cloudy")))
qadate %>%
  ggplot(aes(x=as.Date(date), y=freq, fill=QA)) +
  geom_bar(stat = 'identity', position='fill') +
#  facet_wrap(~QA, ncol=1) +
  theme_bw() +
  scale_x_date(date_labels = "%Y %d")
```



QA detailed

- Only explore the marginal data

```
decodeQA <- function(x, nb){
  bit <- intToBits(x)
  paste(tail(rev(as.integer(bit)), nb), collapse="")}

decodeQAv <- Vectorize(decodeQA)


marginal <- raw %>% filter(summQA == 1)
```

Quality

```
marginal %>%
  mutate(qa_quality_dec = decodeQAv(qa_quality, nb=2)) %>%
  group_by(qa_quality_dec) %>%
  summarise(n = n()) %>%
  mutate(freq = round(n / sum (n)*100,2)) %>%
  kable()
```

qa_quality_dec	n	freq
00	27360	23.5
01	89051	76.5

Quality usefulness

- All data

```
# See https://lpdaac.usgs.gov/sites/default/files/public/product_documentation/mod13_user_guide.pdf
raw %>%
  mutate(qa_use_dec = decodeQAv(qa_use, nb=4)) %>%
  group_by(qa_use_dec) %>%
  summarise(n = n()) %>%
  mutate(freq = round(n / sum (n)*100,2)) %>%
  kable()
```

qa_use_dec	n	freq
0000	191348	53.14
0001	62054	17.23
0010	31643	8.79
0011	27297	7.58
0100	23855	6.63
0101	13003	3.61
0110	6146	1.71
0111	2968	0.82
1000	890	0.25

qa_use_dec	n	freq
1001	264	0.07
1010	140	0.04
1011	8	0.00
1111	448	0.12

- Marginal data

```
marginal %>%
  mutate(qa_use_dec = decodeQAv(qa_use, nb=4)) %>%
  group_by(qa_use_dec) %>%
  summarise(n = n()) %>%
  mutate(freq = round(n / sum (n)*100,2)) %>%
  kable()
```

qa_use_dec	n	freq
0001	45201	38.83
0010	27295	23.45
0011	19753	16.97
0100	14746	12.67
0101	6642	5.71
0110	1891	1.62
0111	708	0.61
1000	53	0.05
1111	122	0.10

Aerosol

```
marginal %>%
  mutate(qa_aerosol_dec = decodeQAv(qa_aerosol, nb=2)) %>%
  group_by(qa_aerosol_dec) %>%
  summarise(n = n()) %>%
  mutate(freq = round(n / sum (n)*100,2)) %>%
  kable()
```

qa_aerosol_dec	n	freq
00	1418	1.22
01	27360	23.50
10	79535	68.32
11	8098	6.96

- Only the 6.96 % of the pixels marked as 'Marginal data' contain high concentration of Aerosols.

Clouds

```
# Adjacents
marginal %>%
  mutate(qa_adj_cloud_dec = decodeQAv(qa_adj_cloud, nb=1)) %>%
  group_by(qa_adj_cloud_dec) %>%
  summarise(n = n()) %>%
  mutate(freq = round(n / sum (n)*100,2)) %>%
  kable()
```

qa_adj_cloud_dec	n	freq
0	81517	70.03
1	34894	29.97

```
# Mixed
marginal %>%
  mutate(qa_mix_cloud_dec = decodeQAv(qa_mix_cloud, nb=1)) %>%
  group_by(qa_mix_cloud_dec) %>%
  summarise(n = n()) %>%
  mutate(freq = round(n / sum (n)*100,2)) %>%
  kable()
```

qa_mix_cloud_dec	n	freq
0	116411	100

- All the pixel marked as 'Marginal data' do not contain mixed cloud.
- We have to consider the Adjacent clouds

Snow

```
marginal %>%
  mutate(qa_snow_dec = decodeQAv(qa_snow, nb=1)) %>%
  group_by(qa_snow_dec) %>%
  summarise(n = n()) %>%
  mutate(freq = round(n / sum (n)*100,2)) %>%
  kable()
```

qa_snow_dec	n	freq
0	116411	100

- All the pixel marked as 'Marginal data' do not contain snow or ice.

Shadow

```
marginal %>%
  mutate(qa_shadow_dec = decodeQAv(qa_shadow, nb=1)) %>%
  group_by(qa_shadow_dec) %>%
  summarise(n = n()) %>%
  mutate(freq = round(n / sum (n)*100,2)) %>%
  kable()
```

qa_shadow_dec	n	freq
0	78388	67.34
1	38023	32.66

- We have to consider the shadow

Filter

We applied customized filter based on Reyes-Díez et al. 2015.

- Creamos una variable nueva, llamada `filtered`. Aquellos pixeles con *Good Data* le asignamos valor `90`. Los *Snow/Ice* y los *Cloudy* le asignamos valor `99`. Para los *Marginal Data* miramos la calidad. En concreto previamente hemos analizado la cantidad de aerosoles, la de nubes y la de sombras (ver mas arriba). La forma de proceder es la siguiente:
  - Si tiene alta concentración de aerosoles, marcamos el pixel con 1.
  - Si tiene nubes adjacentes, marcamos el pixel con 1.
  - Si tiene sombra, marcamos el pixel con 1.
  - Al final, creamos una variable llamada `f_sum` con la suma de esos tres filtros. Si en alguno de los tres filtros se da la condición 1, es decir `f_sum > = 1`, entonces, le asignamos el valor `99` en la variable filtered.

```
rawfilter <- raw %>%
  mutate(f_aerosol = ifelse(decodeQAv(qa_aerosol, nb=2) == '11', 'Aerosol (high)', 'OK'),
         f_cloud_a = ifelse(decodeQAv(qa_adj_cloud, nb=1) == '1', 'Clouds: adjacent', 'OK'),
         f_shadow = ifelse(decodeQAv(qa_shadow, nb=1) == '1', 'Shadow','OK'),
         # as integer
         f_aerosoli = ifelse(f_aerosol == 'OK', 0, 1),
         f_cloud_ai = ifelse(f_cloud_a == 'OK', 0, 2),
         f_shadowi = ifelse(f_shadow == 'OK', 0, 4))

rawfilter <- rawfilter %>%
  mutate(f_sum = f_aerosoli + f_cloud_ai + f_shadowi,
         filtered = ifelse(summQA == 0, 90,
                           ifelse(summQA %in% c(2,3), 99, f_sum)),

         filtered_qa = plyr::mapvalues(filtered,
                                       c(0, 1, 2, 3, 4, 5, 6, 7, 90, 99, NA),
                                       c("Others", "Aerosol",
                                         "Clouds adj", "Aerosol + Clouds adj",
                                         "Shadow", "Aerosol + Shadow",
```

```
                                        "Clouds adj + Shadow",
                                        "Aerosol + Clouds adj + Shadow",
                                        "Good Data", "Snow / Ice or Cloudy", "NA")),
         mes = lubridate::month(date))

rawfilter %>% group_by(filtered_qa) %>%
  summarise(npixels = n()) %>%
  mutate(freq = round((npixels / sum(npixels)*100),2)) %>%
  kable()
```

filtered_qa	npixels	freq
Aerosol	3246	0.90
Aerosol + Clouds adj	2733	0.76
Aerosol + Clouds adj + Shadow	1392	0.39
Aerosol + Shadow	727	0.20
Clouds adj	16134	4.48
Clouds adj + Shadow	14635	4.06
Good Data	208437	57.89
NA	444	0.12
Others	56275	15.63
Shadow	21269	5.91
Snow / Ice or Cloudy	34772	9.66

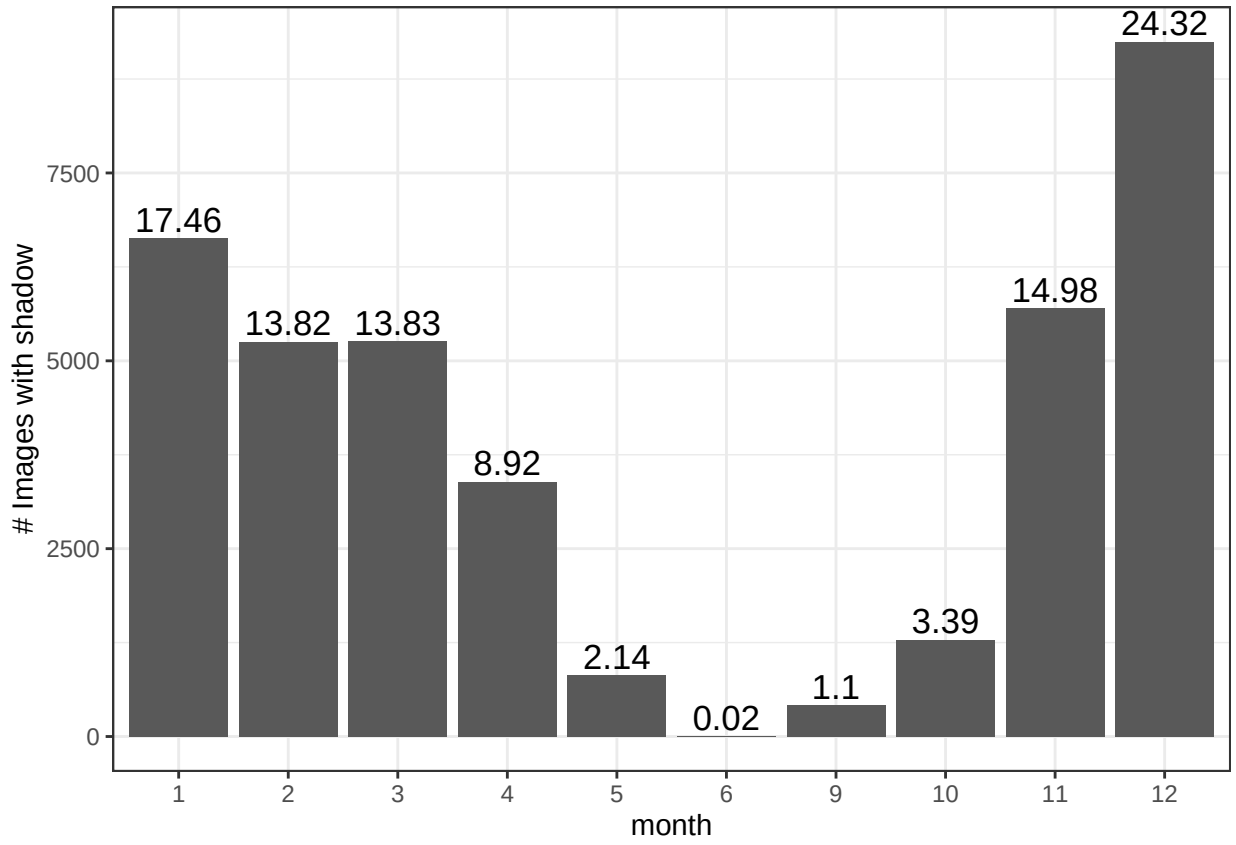
- According to Reyes-Díez et al. (2015) we must consider the shadow in the mountain, but we can discard the filter of adjacent clouds. On the other hand, the use of EVI mean is highly stable under the use of any filter (see Reyes-Díez et al. 2015).

```
rawfilter <- rawfilter %>%
  mutate(filtered_2 = ifelse(filtered %in% c(0,1,2,3), 90, filtered),
         filtered_2_qa = plyr::mapvalues(filtered_2,
                                         c(4, 5, 6, 7, 90, 99, NA),
                                         c("Shadow", "Aerosol + Shadow",
                                           "Clouds adj + Shadow",
                                           "Aerosol + Clouds adj + Shadow",
                                           "Good Data", "Snow / Ice or Cloudy", "NA")))


rawfilter %>% group_by(filtered_2_qa) %>%
  summarise(npixels = n()) %>%
  mutate(freq = round((npixels / sum(npixels)*100),2)) %>%
  kable()
```

filtered_2_qa	npixels	freq
Aerosol + Clouds adj + Shadow	1392	0.39
Aerosol + Shadow	727	0.20
Clouds adj + Shadow	14635	4.06
Good Data	286825	79.66
NA	444	0.12
Shadow	21269	5.91
Snow / Ice or Cloudy	34772	9.66

We have to explore the temporal distribution of the filtered data (especially for Shadow)

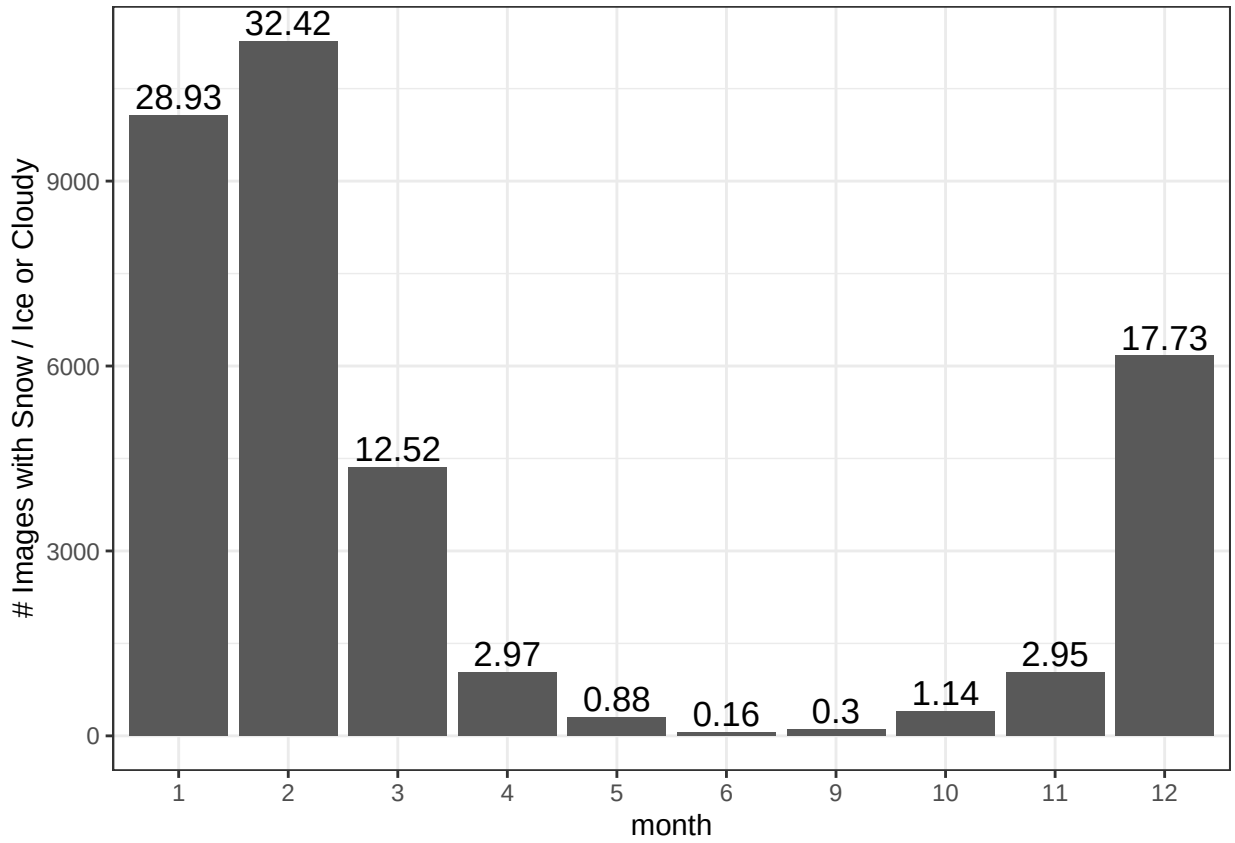
```
sombras <- c("Shadow", "Aerosol + Shadow","Clouds adj + Shadow", "Aerosol + Clouds adj + Shadow")
sombras_n <- c(4, 5, 6, 7)

# Shadow
rawfilter %>% filter(filtered_2 %in% sombras_n) %>% group_by(mes) %>%
  summarise(n=n()) %>%
  mutate(freq = round((n  / sum(n )*100),2)) %>%
  ggplot(aes(x=as.factor(mes), y=n)) +
  geom_bar(stat='identity') +
  geom_text(aes(label=freq, size=3, vjust=-.25)) +
  theme_bw() +
  theme(legend.position = 'none') +
  xlab('month') +
  ylab('# Images with shadow')
```



Vemos que la mayoría de las imagenes con sombra están en los meses de invierno: mes 1, 2, 3, y 12 (69.32 %), esto es: el 69.32 % de las imágenes que están marcadas con sombra en QA Detailed, se distribuyen temporalmente entre los meses 1,2,3 y 12, que coincide con el periodo donde el Quercus pyrenaica no presenta actividad fotosintética (incluir cita de estación de crecimiento, y de cuando tira la hoja), por lo tanto los valores con sombra los debemos eliminar a la hora de calcular la media de EVI, ya que se encuentran la mayoría en los meses de periodo en los que el roble no tiene actividad fotosintética.

Now for Ice / Cloudy

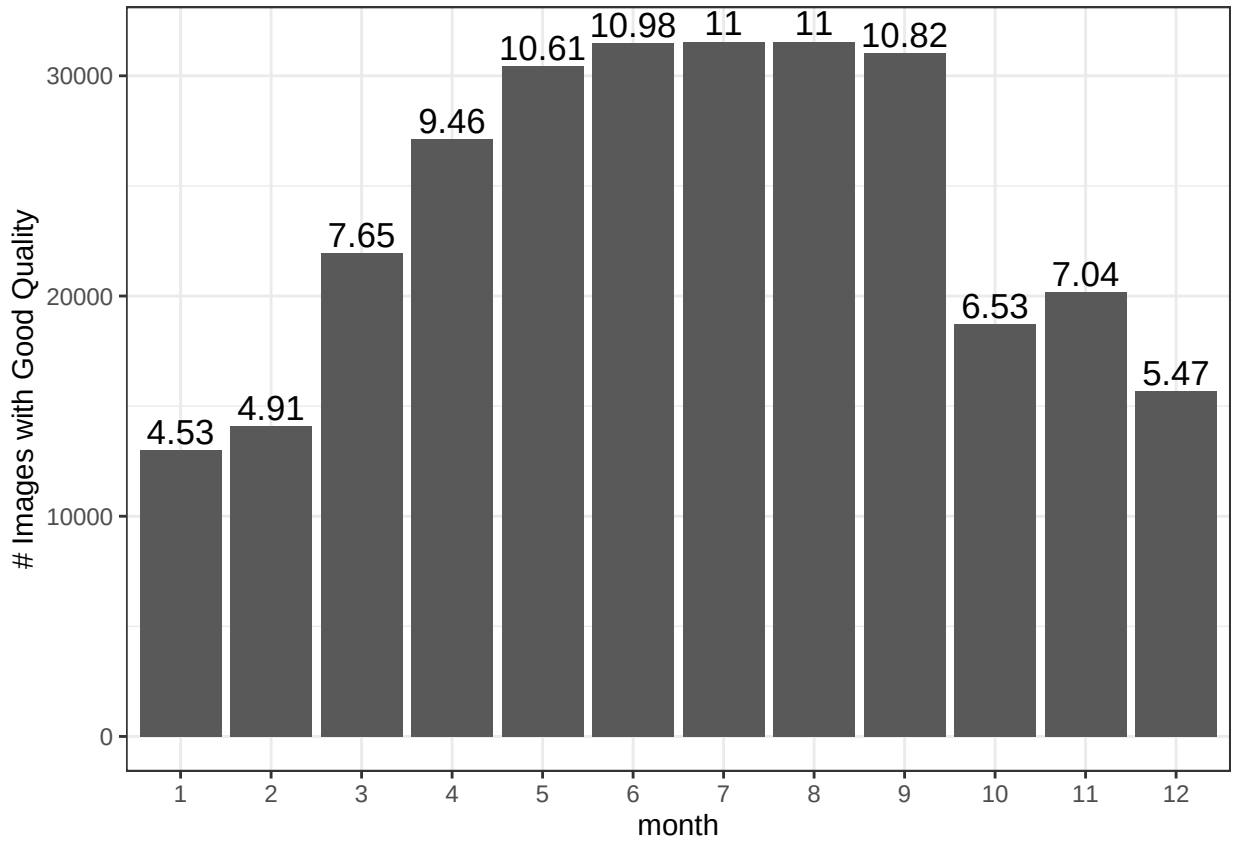
```
# "Snow / Ice or Cloudy" (99)

# Shadow
rawfilter %>% filter(filtered_2 == 99) %>% group_by(mes) %>%
  summarise(n=n()) %>%
  mutate(freq = round((n  / sum(n )*100),2)) %>%
  ggplot(aes(x=as.factor(mes), y=n)) +
  geom_bar(stat='identity') +
  geom_text(aes(label=freq, size=3, vjust=-.25)) +
  theme_bw() +
  theme(legend.position = 'none') +
  xlab('month') +
  ylab('# Images with Snow / Ice or Cloudy')
```



We select only data with high Good Quality, because shadow, ice or cloudy are mainly locate at winter months.

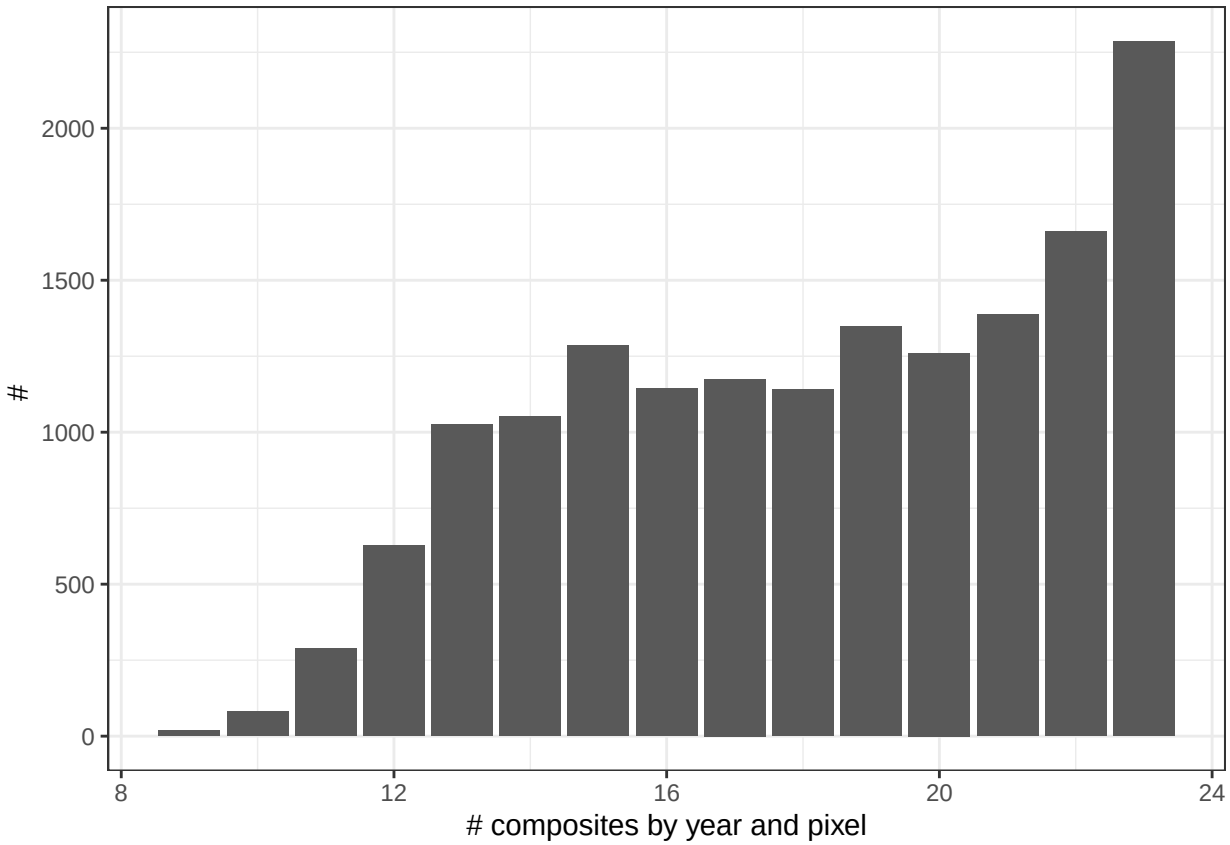
```
df <- rawfilter %>%
  filter(filtered_2 == 90) %>%
  select(doy, evi, ndvi, iv_malla_modi_id, pop, date, year, lat, long, summQA, qa_use, filtered, filtere

df %>% group_by(mes) %>%
  summarise(n=n()) %>%
  mutate(freq = round((n/sum(n )*100),2)) %>%
  ggplot(aes(x=as.factor(mes), y=n)) +
  geom_bar(stat='identity') +
```

```
  geom_text(aes(label=freq, size=3, vjust=-.25)) +
  theme_bw() +
  theme(legend.position = 'none') +
  xlab('month') +
  ylab('# Images with Good Quality')
```




```
ncomposites_year <- df %>% group_by(iv_malla_modi_id, year) %>% summarise(n=n())

ncomposites_year %>% ggplot(aes(x=n)) +
  geom_histogram(stat='count') +
  xlab('# composites by year and pixel') +
  ylab('# ')+
  theme_bw()
```

```
## Warning: Ignoring unknown parameters: binwidth, bins, pad
```



En este último gráfico, exploramos el número de pixeles que tienen todos las imágenes de un año completo. Aunque no se si tiene mucho sentido este gráfico.

We do not apply a filter treshold, because we selected only the Good Data.

Prepare data of processed IV

Get the composite of the images and the season

- See Testa et al. 2014
- Use a custom function

```
# Get leap years
years <- unique(df$year)
ly <- years[lubridate::leap_year(years)]

# Two functions
rd <- df %>%
  mutate(m= lubridate::month(date),
         d= lubridate::day(date)) %>%
  mutate(composite = ifelse(year %in% ly,
                            getComposite_leap(m,d),
                            getComposite_nonleap(m,d))) %>%
  mutate(season = ifelse(composite < 6, 'winter',
```

```
                    ifelse(composite < 12, 'spring',
                     ifelse(composite < 18, 'summer','autumn'))))
```

Scale factor of the NDVI and EVI data

```
# Apply scale factor https://lpdaac.usgs.gov/dataset_discovery/modis/modis_products_table/mod13q1
rd <- rd %>%
  mutate(evi = evi * 0.0001,
         ndvi = ndvi * 0.0001)
```

We created several datasets:

iv_composite

iv by pixel and by composite (output as `./data/evi/iv_composite.csv`). It contains the following fields:

- `iv_malla_modi_id`: the identifier of the modis cell
- `year`
- `evi` or `ndvi`: the value of the EVI (or NDVI) for the composite
- `date`: date of adquisition of the image
- `composite`: number of composite (23 by year)
- `long`: longitude coordinates
- `lat`: latitute coordinates
- `pop`: numeric code of the *Q. pyrenaica* population
- `season`:
  - `0` annual value
  - `1` spring value
  - `2` summer value
  - `3` autumn value
  - `4` winter value
- `seasonF`: the season coded as factor
- `summQA`: the summary Quality Assessment (see MODIS)
- `qa_use`: the usseful index (see QA MODIS)

evi_mean

evi mean by pixel and year (output as `./data/evi/evi_mean.csv`). It contains the following fields:

- `iv_malla_modi_id`: the identifier of the modis cell
- `year`
- `evi`: mean value of evi for pixel i at year j
- `long`: longitude coordinates
- `lat`: latitute coordinates
- `pop`: numeric code of the *Q. pyrenaica* population
- `n_composites`: number of composites used to computed the annual mean value of EVI (according to previously filtering)

evi_seasonal* and *ndvi_seasonal

- annual and seasonal evi by pixel (output as `./data/evi/evi_atributes_all.csv`)
- annual and seasonal ndvi by pixel (output as `./data/evi/ndvi_atributes_all.csv`)

These two dataframes have the following fields:

- `iv_malla_modi_id`: the identifier of the modis cell
- `year`
- `evi` or `ndvi`: the value of the EVI (or NDVI) (cumulative value for each season)
- `season`: the season of cumulative evi:
  - `0` annual value
  - `1` spring value
  - `2` summer value
  - `3` autumn value
  - `4` winter value
- `seasonF`: the season coded as factor
- `long`: longitude coordinates
- `lat`: latitute coordinates
- `pop`: numeric code of the *Q. pyrenaica* population

Create dataframe with composites

```
iv_composite <- rd %>%
  dplyr::select(iv_malla_modi_id, evi, ndvi, pop, date, year, long, lat, composite, seasonF = season, su
```

Create EVI mean dataset

```
evimean <- rd %>%
  group_by(iv_malla_modi_id, year) %>%
  summarise(evi = mean(evi[evi >=0], na.rm=T),
            n_composites = length(year))

# Add coordinates and pob and other useful info
aux_rd <- rd %>% dplyr::select(iv_malla_modi_id, long, lat, pop) %>%
  group_by(iv_malla_modi_id) %>% unique()

# Join dataframes
evimean <- evimean %>% dplyr::inner_join(aux_rd, by="iv_malla_modi_id")
```

Create seasonal dataframes of EVI and NDVI (integrate, EVI and NDVI sum)

- EVI

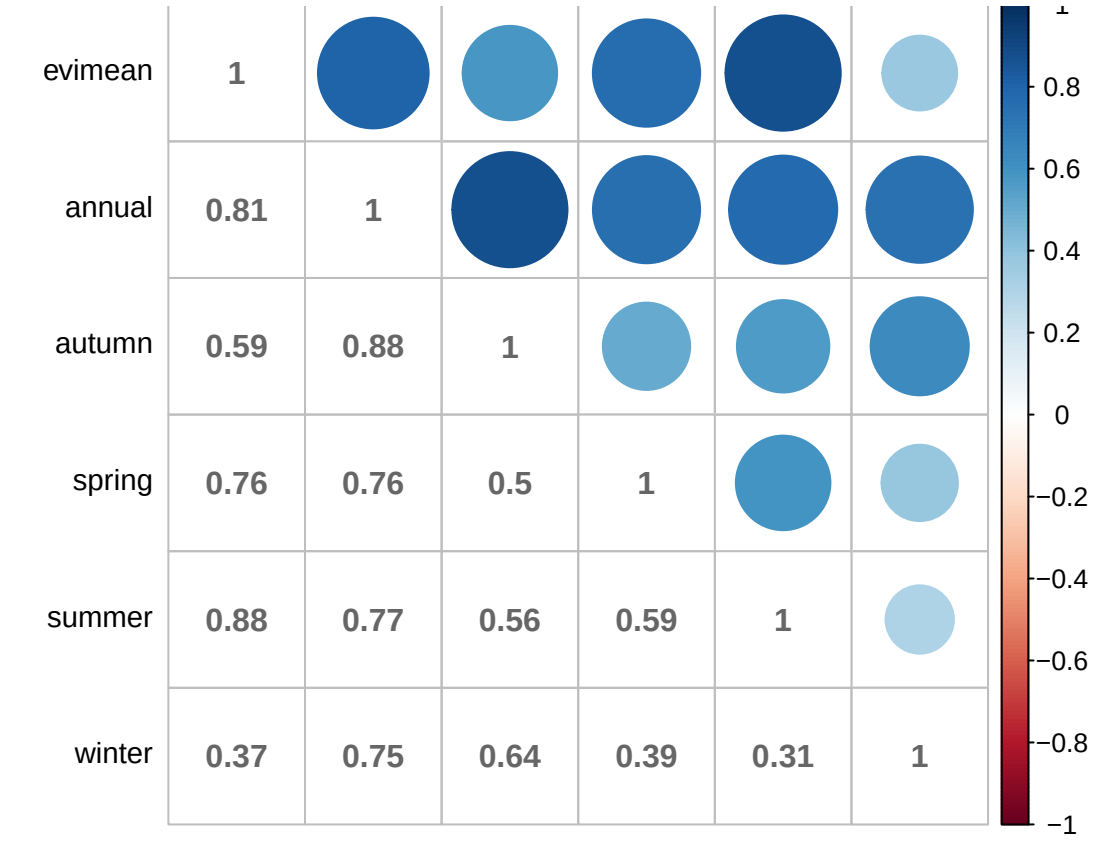
```
# Create annual evi by pixel
evi_annual <- rd %>%
  group_by(iv_malla_modi_id, year) %>%
  summarise(evi = sum(evi[evi >=0],  na.rm=T)) %>%
  mutate(seasonF='annual',
```

```
         season = 0)

# Create seasonal evi by pixel
evi_season <- rd %>%
  group_by(iv_malla_modi_id, year, season) %>%
  summarise(evi = sum(evi[evi >=0], na.rm=T)) %>%
  mutate(seasonF = season) %>%
  mutate(season = ifelse(season == 'autumn', 3,
                  ifelse(season == 'winter', 4,
                   ifelse(season == 'spring', 1,2))))

evidf <- rbind(evi_annual, evi_season)

# Add coordinates and pob
aux_rd <- rd %>% dplyr::select(iv_malla_modi_id, long, lat, pop) %>%
  group_by(iv_malla_modi_id) %>% unique()


# Join dataframes
evidf <- evidf %>% dplyr::inner_join(aux_rd, by="iv_malla_modi_id")
```

Correlaciones entre el EVI medio y los EVI estacionales para los robledales

```
co <- evidf %>% dplyr::select(iv_malla_modi_id, year, evi, seasonF) %>%
  spread(seasonF, evi) %>% as.data.frame()

co2 <- evimean %>%
  dplyr::select(iv_malla_modi_id, year, evimean = evi) %>%
  inner_join(co, by =c("iv_malla_modi_id","year")) %>% as.data.frame()


co3 <- co2 %>% dplyr::select(-iv_malla_modi_id, -year)

# Ver https://cran.r-project.org/web/packages/corrplot/vignettes/corrplot-intro.html
cor.mtest <- function(mat, conf.level = 0.95){
  mat <- as.matrix(mat)
    n <- ncol(mat)
    p.mat <- lowCI.mat <- uppCI.mat <- matrix(NA, n, n)
    diag(p.mat) <- 0
    diag(lowCI.mat) <- diag(uppCI.mat) <- 1
    for(i in 1:(n-1)){
        for(j in (i+1):n){
            tmp <- cor.test(mat[,i], mat[,j], conf.level = conf.level)
            p.mat[i,j] <- p.mat[j,i] <- tmp$p.value
            lowCI.mat[i,j] <- lowCI.mat[j,i] <- tmp$conf.int[1]
            uppCI.mat[i,j] <- uppCI.mat[j,i] <- tmp$conf.int[2]
        }
    }
    return(list(p.mat, lowCI.mat, uppCI.mat))
}
```

```
m <- cor(co3, use='complete.obs')
res1 <- cor.mtest(co3,0.99)


corrplot(m, type='upper',diag=TRUE,
         tl.cex=.9, tl.col='black', tl.pos='lt',
         p.mat=res1[[1]], insig='p-value', sig.level=0.001)

corrplot(m, method= 'number', type='lower',
           diag=TRUE, col='grey40',
           tl.pos='n', cl.pos='n',
           add=TRUE)
```



- NDVI

```
# Create annual ndvi by pixel
ndvi_annual <- rd %>%
  group_by(iv_malla_modi_id, year) %>%
  summarise(ndvi = sum(ndvi[ndvi >=0], na.rm=T)) %>%
  mutate(seasonF='annual',
         season = 0)

# Create seasonal ndvi by pixel
ndvi_season <- rd %>%
```

```
  group_by(iv_malla_modi_id, year, season) %>%
  summarise(ndvi = sum(ndvi[ndvi >=0], na.rm=T)) %>%
  mutate(seasonF = season) %>%
  mutate(season = ifelse(season == 'autumn', 3,
                  ifelse(season == 'winter', 4,
                   ifelse(season == 'spring', 1,2))))

ndvidf <- rbind(ndvi_annual, ndvi_season)

# Join dataframes
ndvidf <- ndvidf %>% dplyr::inner_join(aux_rd, by="iv_malla_modi_id")
```

Export dataframes

```
# Export dataframes
write.csv(evimean, file=paste(di, "/data/evi/evi_mean.csv", sep=""), row.names = FALSE)
write.csv(evidf, file=paste(di, "/data/evi/evi_atributes_all.csv", sep=""), row.names = FALSE)
write.csv(ndvidf, file=paste(di, "/data/evi/ndvi_atributes_all.csv", sep=""), row.names = FALSE)
write.csv(iv_composite, file=paste(di, "/data/evi/iv_composite.csv", sep=""), row.names = FALSE)
```