Yield Data Exploration October 2024

Elizaveta Shcherbinina

2024-10-15

Pre check

Pre check is done before proceeding to Alfred's check. It is made to the data that was collected after running the GEE code and merging it with the original properties of the dataset.

Attention: - 36 observations from `FEB23_440` have `NA` in **number of replicates**. - 3008 observations have `NA` in **yield_SD_control**. - 3007 observations have `NA` in **yield_SD_treatment**. - Points from Hawaii are excluded because there was no map provided for Hawaii, and the calculation resulted in zeroes.

```
library(tidyr)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)

data_20241020 <- read.csv("C:\\Users\\lisa7\\Documents\\surrounding_landscapes\\data\\working data\\2024

x <-data_20241020[which(is.na(data_20241020$ma_id)), ] # 0 obs
x <-data_20241020[which(is.na(data_20241020$measurement_id)), ] # 0 obs
x <-data_20241020[which(is.na(data_20241020$study_id)), ] # 0 obs
x <-data_20241020[which(is.na(data_20241020$control_id)), ] # 0 obs
x <- data_20241020[which(is.na(data_20241020$control_replicates)), ] # 36 obs FEB23_440
x <- data_20241020[which(is.na(data_20241020$treatment_replicates)), ] # same 36 obs FEB23_440
x <-data_20241020[which(is.na(data_20241020$crop_type)), ] # 0 obs
x <-data_20241020[which(is.na(data_20241020$crop_type_grouped_small)), ] # 0 obs
x <-data_20241020[which(is.na(data_20241020$crop_type_grouped_big)), ] # 0 obs
x <-data_20241020[which(is.na(data_20241020$treatment)), ] # 0 obs
x <-data_20241020[which(is.na(data_20241020$yield_control)), ] # 0 obs
x <-data_20241020[which(is.na(data_20241020$yield_treatment)), ] # 0 obs
x <-data_20241020[which(is.na(data_20241020$yield_SD_control)), ] # 3008 obs
```

```r
x <-data_20241020[which(is.na(data_20241020$yield_SD_treatment)), ] # 3007 obs
x <-data_20241020[which(is.na(data_20241020$yield_unit)), ] # 0 obs
x <-data_20241020[which(is.na(data_20241020$longitude_decimal)), ] # 0 obs
x <-data_20241020[which(is.na(data_20241020$latitude_decimal)), ] # 0 obs
x <-data_20241020[which(is.na(data_20241020$country_new)), ] # 0 obs
x <-data_20241020[which(is.na(data_20241020$landcover_map_year)), ] # 0 obs

#
# #DataExplorer::create_report(data_20241020, output_file = "report.pdf",
#                             output_format = "pdf_document")
#
# # library(CoordinateCleaner)
# # flags <- clean_coordinates(data_20241020, lon = "longitude_decimal",
#  lat = "latitude_decimal", species = "measurement_id")
# # summary(flags)
# #
# # flags %>% filter(.sea == "FALSE") # those are the coordinates we added manually,
# # # we can be sure that they just land very close to water
#

# Hawaiian points could not be calculated because there is no map for Hawaii:
hawaii <-
data_20241020 %>%
  filter(longitude_decimal < -150) %>%
  select(measurement_id, latitude_decimal, longitude_decimal) %>%
  pull(measurement_id)

data_20241020 <-
data_20241020 %>%
  filter(!(measurement_id %in% hawaii))


write.csv(data_20241020, "C:\\Users\\lisa7\\Documents\\surrounding_landscapes\\data\\working data\\2024
```

```r
data <- read.csv("C:\\Users\\lisa7\\Documents\\surrounding_landscapes\\data\\working data\\20241026_data

# lets save a data_unchanged dataset in case we need to look up anything after
# transformations :
data_unchanged  <- data

landcover.meta<-read.csv('C:\\Users\\lisa7\\Documents\\surrounding_landscapes\\data\\legend_classcode_la
```

Attention: - New columns added: **SD** for kg/ha. - `measurement_id`, `control_id` (all lowercase). - The order of columns changed.

Structure of the dataset:

```r
str(data)
```

```
## 'data.frame':    849852 obs. of  51 variables:
##  $ ma_id                                : chr  "D352" "D352" "D352" "D352" ...
##  $ measurement_id                       : int  1246 1246 1246 1246 1246 1246 1246 1246 1246 1246 ..
##  $ study_id                             : int  11 11 11 11 11 11 11 11 11 11 ...
```

```
##  $ control_id                                : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ author_year                               : chr  "Allen et al. 2009" "Allen et al. 2009" "Allen et al
##  $ title                                     : chr  "" "" "" "" ...
##  $ study_pubyear                             : int  2009 2009 2009 2009 2009 2009 2009 2009 2009 2009 ..
##  $ harvest_year                              : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ longitude                                 : chr  "104.25" "104.25" "104.25" "104.25" ...
##  $ latitude                                  : chr  "47.76666667" "47.76666667" "47.76666667" "47.766666
##  $ region                                    : chr  "" "" "" "" ...
##  $ country                                   : chr  "" "" "" "" ...
##  $ control_replicates                        : int  3 3 3 3 3 3 3 3 3 3 ...
##  $ treatment_replicates                      : int  3 3 3 3 3 3 3 3 3 3 ...
##  $ crop_type                                 : chr  "Wheat" "Wheat" "Wheat" "Wheat" ...
##  $ treatment                                 : chr  "Erosion simulation" "Erosion simulation" "Erosion si
##  $ yield_control                             : num  1350 1350 1350 1350 1350 1350 1350 1350 1350 1350 ..
##  $ yield_treatment                           : num  1270 1270 1270 1270 1270 1270 1270 1270 1270 1270 ..
##  $ yield_SD_control                          : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ yield_SD_treatment                        : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ yield_unit                                : chr  "kg/ha" "kg/ha" "kg/ha" "kg/ha" ...
##  $ soil_type                                 : chr  "unknow" "unknow" "unknow" "unknow" ...
##  $ temperature                               : chr  "" "" "" "" ...
##  $ precipitation                             : chr  "" "" "" "" ...
##  $ climate_zone                              : chr  "" "" "" "" ...
##  $ LRR                                       : num  -0.0611 -0.0611 -0.0611 -0.0611 -0.0611 ...
##  $ LRR_vi                                    : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ reference                                 : chr  "" "" "" "" ...
##  $ longitude_decimal                         : num  104 104 104 104 104 ...
##  $ latitude_decimal                          : num  47.8 47.8 47.8 47.8 47.8 ...
##  $ country_new                               : chr  "Mongolia" "Mongolia" "Mongolia" "Mongolia" ...
##  $ crop_type_grouped_small                   : chr  "Wheat" "Wheat" "Wheat" "Wheat" ...
##  $ crop_type_grouped_big                     : chr  "Grains" "Grains" "Grains" "Grains" ...
##  $ publication_to_harvest_year_difference: int  NA NA NA NA NA NA NA NA NA NA ...
##  $ harvest_year_by_median                    : int  2002 2002 2002 2002 2002 2002 2002 2002 2002 2002 ..
##  $ landcover_map_year                        : int  2002 2002 2002 2002 2002 2002 2002 2002 2002 2002 ..
##  $ yield_control_kgha                        : num  1350 1350 1350 1350 1350 1350 1350 1350 1350 1350 ..
##  $ yield_treatment_kgha                      : num  1270 1270 1270 1270 1270 1270 1270 1270 1270 1270 ..
##  $ yield_SD_control_kgha                     : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ yield_SD_treatment_kgha                   : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ class                                     : int  10 11 12 20 51 52 61 62 71 72 ...
##  $ area_m2                                   : num  0 28443 0 0 0 ...
##  $ edgelength_m                              : num  0 105695 0 0 0 ...
##  $ buffer_radius_m                           : int  1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 ..
##  $ buffer_area_m2                            : num  3141593 3141593 3141593 3141593 3141593 ...
##  $ proportion                                : num  0 0.00905 0 0 0 ...
##  $ simpsons_index                            : num  0.33 0.33 0.33 0.33 0.33 ...
##  $ species_richness                          : int  36 36 36 36 36 36 36 36 36 36 ...
##  $ simpsons_evenness                         : num  0.0842 0.0842 0.0842 0.0842 0.0842 ...
##  $ perimeter_to_area                         : num  NA 3.72 NA NA NA ...
##  $ shannons_index                            : num  0.535 0.535 0.535 0.535 0.535 ...
```

```r
# to see the number of unique measurements:
x <- data %>% select(measurement_id) %>% unique()
```

The current dataset has 7869 unique measurements.

(I) check and clean the data

(A) check frequency of class entries per unique treatment yield measurement (i.e. No.)

Check whether we have each land-cover class only one time for each unique measurement

```
check<-as.data.frame(table(data$class, data$buffer_radius_m, data$measurement_id))
colnames(check)<- c('class', 'radius', 'number','Freq')

# Good that seems right now!
x <- unique(check$number[which(check$Freq>1)])

length(x)
```

```
## [1] 0
```

```
# clean up
rm(check)
```

Result: All observations are unique and there are no land-cover dublicates

(B) check for missing landscape data

```
unique(data$buffer_radius_m)
```

```
## [1] 1000 2500 5000
```

```
x<-data[which(is.na(data$buffer_radius_m)), ]
print(x)
```

```
##  [1] ma_id
##  [2] measurement_id
##  [3] study_id
##  [4] control_id
##  [5] author_year
##  [6] title
##  [7] study_pubyear
##  [8] harvest_year
##  [9] longitude
## [10] latitude
## [11] region
## [12] country
## [13] control_replicates
## [14] treatment_replicates
## [15] crop_type
## [16] treatment
## [17] yield_control
## [18] yield_treatment
```

```
## [19] yield_SD_control
## [20] yield_SD_treatment
## [21] yield_unit
## [22] soil_type
## [23] temperature
## [24] precipitation
## [25] climate_zone
## [26] LRR
## [27] LRR_vi
## [28] reference
## [29] longitude_decimal
## [30] latitude_decimal
## [31] country_new
## [32] crop_type_grouped_small
## [33] crop_type_grouped_big
## [34] publication_to_harvest_year_difference
## [35] harvest_year_by_median
## [36] landcover_map_year
## [37] yield_control_kgha
## [38] yield_treatment_kgha
## [39] yield_SD_control_kgha
## [40] yield_SD_treatment_kgha
## [41] class
## [42] area_m2
## [43] edgelength_m
## [44] buffer_radius_m
## [45] buffer_area_m2
## [46] proportion
## [47] simpsons_index
## [48] species_richness
## [49] simpsons_evenness
## [50] perimeter_to_area
## [51] shannons_index
## <0 rows> (or 0-length row.names)
```

(C) check the latitude data

```
summary(data$latitude_decimal)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  -36.54   11.97   31.82   23.11   40.27   62.79
```

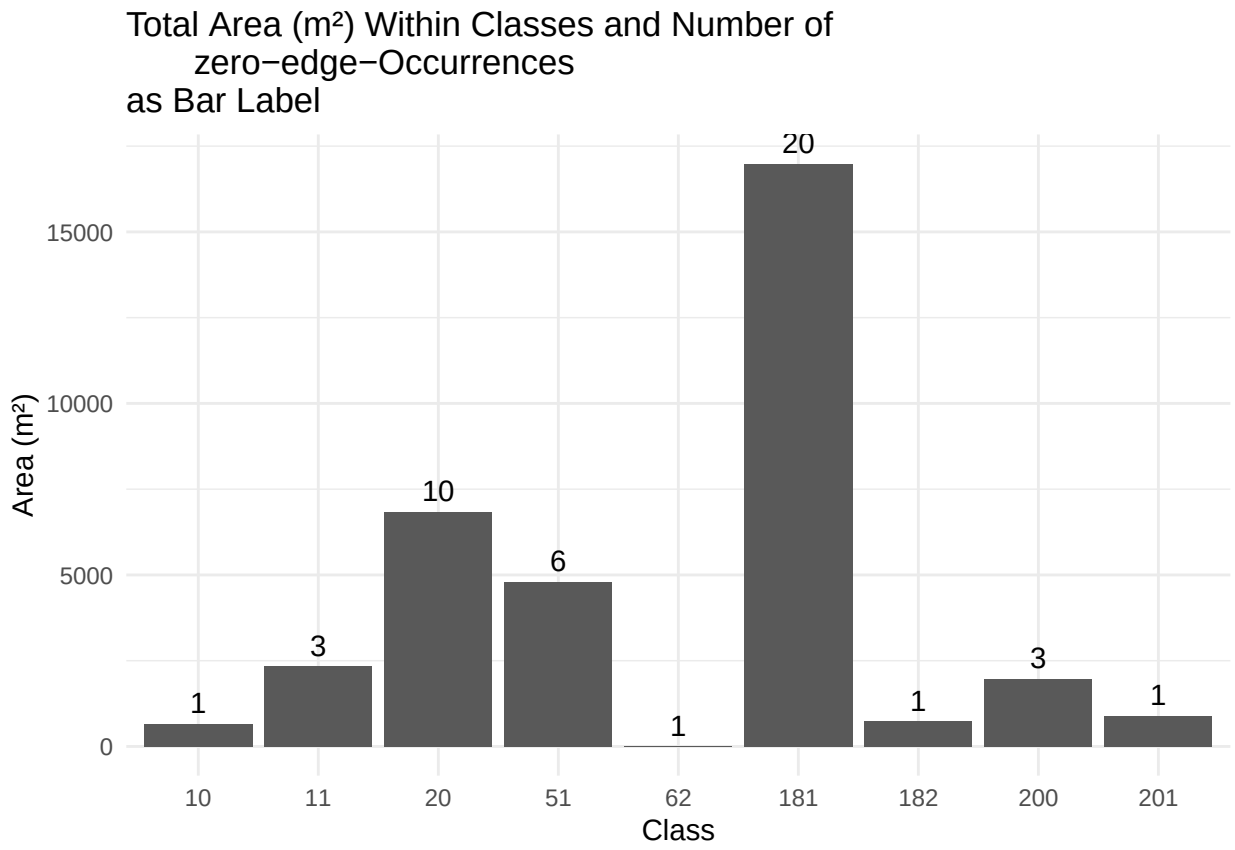
Result: Latitude fits the allowed interval of [-90:+90]

(D) Fix 0 edge issues

There is an issue that some land-cover specifications have 0 edge length. The reason is unknown.

```
data_edgelength_zero <-
data %>%
```

```r
  filter(edgelength_m == 0) %>%
  select(measurement_id, class,buffer_radius_m, area_m2, edgelength_m) %>%
  filter(area_m2 != 0) %>%
  mutate(class = as.factor(class))


## ATTENTION!
## I exclude class 0 because those are NA areas, and no wonder they have no edge.
## Though those are the most cases of edge = 0 and area != 0
data_edgelength_zero %>%
  group_by(class) %>%
  filter(class != 0) %>%
  summarise(sum_area = sum(area_m2), count = n()) %>%
  ggplot(aes(x = factor(class), y = sum_area)) +
  geom_bar(stat = "identity") +
  geom_text(aes(label = count), vjust = -0.5) +
  labs(title = "Total Area (m²) Within Classes and Number of
       zero-edge-Occurrences \nas Bar Label ",
       x = "Class",
       y = "Area (m²)") +
  theme_minimal()
```



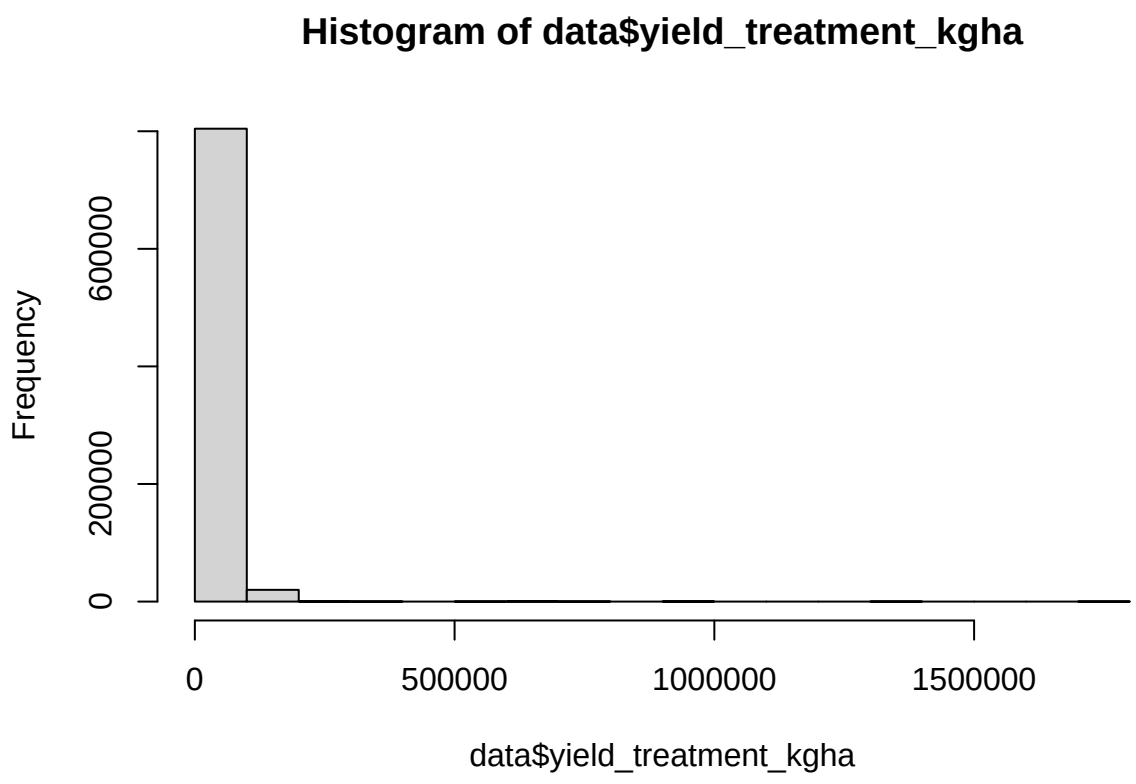
There are in total 56 observations that have an area but dont have an edge.

Since we don't know the reasoin we simply fix it. The fix is that we assume a circular area of that land-cover class and that it is to 100% in the radius.

```
selection<-which(data$edgelength_m==0)
# calculates from the area the radius of a circle with that area and then
#computes from the radius the perimeter
data$edgelength_m[selection]<-(sqrt(data$area_m2[selection])) * 2

rm(selection)

# short check:
x <- data %>%
  filter(edgelength_m == 0) %>%
  filter(area_m2 != 0)

# It worked!
```

(E) check whether yield data is realistic

```
hist(data$yield_treatment_kgha)
```




```
summary(data$yield_treatment_kgha)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.      NA's
##       0.1    2619.0    5720.0   23785.3   27888.9 1783000.0     24192
```

```r
# Alfred uses the following lines to identify the crops that have an
# inappropriately high yield
unique(data$crop_type_grouped_small[which(data$yield_treatment_kgha>10^7)])
```

```
## character(0)
```

```r
unique(data$crop_type_grouped_big[which(data$yield_treatment_kgha>10^7)])
```

```
## character(0)
```

```r
# it is also important to check what kind of units are reported
unique(data$yield_unit[which(data$yield_treatment_kgha>10^7)])
```

```
## character(0)
```

@Alfred's comment: this seems to be a bit odd - a very good corn yield per ha is e.g. 13 tons. So, that fits with the median (6000 kg). However, we have here a lot of yields that are 10000 tons and higher. That cannot be right. We should look into this closer! One possibility could be that this is the whole biomass and not the corn yield, but that would still weird somehow to report this.

@Elizaveta's comment: Since we do not want to dive into primary literature, lets remove the outliers. The highest possible maximum for maize yield in controlled conditions is 83.3 t/ha. https://www-archiv.fdm.uni-hamburg.de/b-online/library/maize/www.ag.iastate.edu/departments/agronomy/yield.html#:~:text=When%20maximizing%20these%20factors%20to,(83.3%20t%2Fha).

83.3 t/ha is 83000 kg/ha. So let's remove any grains observation that surpass 100.000 kg/ha

```r
data <-
data %>%
  mutate(yield_control_kgha = case_when(
    crop_type_grouped_big %in% c("Grains", "Other Grains") &
      yield_control_kgha > 100000 ~ NA_real_,
    TRUE ~ yield_control_kgha
  )) %>%
    mutate(yield_treatment_kgha = case_when(
    crop_type_grouped_big %in% c("Grains", "Other Grains") &
      yield_treatment_kgha > 100000 ~ NA_real_,
    TRUE ~ yield_treatment_kgha
  )) %>%
  filter(!is.na(yield_control_kgha)) %>%
  filter(!is.na(yield_treatment_kgha))

x <- data %>% select(measurement_id) %>% unique()
```

In the number of observations we went down to 7467 unique measurements.

@Alfreds comment: There are some data that show a quite high change in values - so either an increase by more than 150% or a shrinkage of the yield in the treatment to less than a quarter of what the original yield was. This seems extreme - can we trust this? We should probably also double check?

```
# @Elizaveta: I rewrote Alfred's code (the next 2 lines) below

#x<-data$yield_treatment_kgha/ data$yield_control_kgha
#x[which(x> 2.5 | x<0.25)]

data %>%
  mutate(coef = yield_treatment_kgha/ data$yield_control_kgha) %>%
  filter(coef %in% (0.25:2.5)) %>%
  select(-(class:shannons_index)) %>%
  unique() %>%
  select(1:5, yield_control_kgha, yield_treatment_kgha)
```

```
##     ma_id measurement_id study_id control_id                      author_year
## 1    D973           4446       58         37                 Nixon et al. 2003
## 109  D973           4569       22          3 Campanhola Bortoluzzi et al. 2013
## 217  A288           8778      106         34         Thorup-Kristensen et al.
##     yield_control_kgha yield_treatment_kgha
## 1                 8400                10500
## 109               1560                 1950
## 217               3240                 4050
```

There are only 3 unique measurements with a relatively high change in yield values between control and treatment.

(F) Sort out proportion data so that land cover classes always add up to 1

```
id<-paste0(data$ma_id, '_',data$measurement_id, '_',data$buffer_radius_m)
unique_id<-unique(id)

# takes a bit because it is a slow loop... but is okay :)
for(i in 1:length(unique_id)){x<-which(id==unique_id[i])
  data$proportion[x]<-data$proportion[x]/sum(data$proportion[x])}
```

(II) transform the data

(A) Change the format

Rewrite the data in the long format

```
# Alfred's code
# data.mod<- pivot_wider(data[,c(1:49, 51)], names_from = class,
#                        values_from = c(42,43,46))

# and perimeter to area ratio is due to some reasons (?) left out

data.mod <-
  data %>%
  select(-perimeter_to_area) %>%
  pivot_wider(names_from = class,
```

```
              values_from = c(area_m2, edgelength_m, proportion))



x<-as.data.frame(table(data.mod$measurement_id))
table(x$Freq)
```

```
##
##    3
## 7467
```

That worked nicely - each unique measurement has now exactly three entries(for each of the three radii)

```
# second step here - we have still three

data.mod.2<- pivot_wider(data.mod[,-c(42:46)], names_from = buffer_radius_m,
                         values_from = c(42:(ncol(data.mod)-5)))

x<-as.data.frame(table(data.mod.2$measurement_id))
table(x$Freq)
```

```
##
##    1
## 7467
```

that worked nicely again!

(B) check differences between control and treatment yield data

```
x<-data.mod.2[which(data.mod.2$yield_control_kgha == data.mod.2$yield_treatment_kgha),]
#which(data$yield_control_kgha == data$yield_treatment_kgha)

print(x)
```

```
## # A tibble: 55 x 364
##    ma_id measurement_id study_id control_id author_year      title study_pubyear
##    <chr>          <int>    <int>      <int> <chr>            <chr>         <int>
##  1 D352            1298        6          5 "Lamey et al. 2~ ""             2000
##  2 D473            1494       14          1 ""               ""             2014
##  3 D652            2673      216          2 ""               ""             2011
##  4 D921            3213      392          1 "Niu et al. 201~ ""             2017
##  5 D973            4055       60          2 "Pagani and Mal~ ""             2014
##  6 D973            4083       60         30 "Pagani and Mal~ ""             2014
##  7 D973            4084       60         31 "Pagani and Mal~ ""             2014
##  8 D973            4196       12          7 "Ayalew 2011"    ""             2011
##  9 D973            4198       12          9 "Ayalew 2011"    ""             2011
## 10 D973            4258        1         39 "Adeoye and Sin~ ""             1984
## # i 45 more rows
## # i 357 more variables: harvest_year <int>, longitude <chr>, latitude <chr>,
```

```
## #   region <chr>, country <chr>, control_replicates <int>,
## #   treatment_replicates <int>, crop_type <chr>, treatment <chr>,
## #   yield_control <dbl>, yield_treatment <dbl>, yield_SD_control <dbl>,
## #   yield_SD_treatment <dbl>, yield_unit <chr>, soil_type <chr>,
## #   temperature <chr>, precipitation <chr>, climate_zone <chr>, LRR <dbl>, ...
```

The total number of such yield_control = yield_treatmnet observations is 55. Most of the observations however come from FEB23_440 Bender D.A., Morrison W.P., Frisbie R.E. And those indeed are a little bit weird, because the control and treatment are exact the same number across many measurements.

(C) erase double data entries

In this section we remove double data entries and aggregate the dataset via means.

I rewrote Alfred's code in my way, so the first code section (Alfred's) is silenced, and I do the same procedure in the second code section.

```
# here I use an aggregate function to compute average for all data that has the same publication year,

# @Elizaveta: ATTENTION! : STUDY ID IS NOT UNIQUE
# if I do add study ID (different for each meta-analysis), then I get suddenly 9 additional data points
# are twice in our data set and we probably better remove them.


## START ALFRED'S CODE:
#
#data.mod.2$Lat_long<- paste0(data.mod.2$latitude_decimal, data.mod.2$longitude_decimal)
#
# means.check<-aggregate(list(data.mod.2$yield_control_kgha, data.mod.2$yield_treatment_kgha, data.mod.
#                         by = list(data.mod.2$study_pubyear, data.mod.2$treatment, data.mod.2$Lat_long,
#                                   data.mod.2$crop_type),   function(x){mean(x, na.rm = T)})
#
# colnames(means.check)<-c('study_pubyear', 'treatment', 'Lat_long', 'crop_type','yield_control_kgha',
#                          'yield_treatment_kgha', 'measurement_id')
#
# means.check.2<-aggregate(list(data.mod.2$yield_control_kgha, data.mod.2$yield_treatment_kgha, data.mo
#                         by = list(data.mod.2$ma_id, data.mod.2$study_id, data.mod.2$study_pubyear, dat
#                                   data.mod.2$crop_type),   function(x){mean(x, na.rm = T)})
#
# colnames(means.check.2)<-c('ma_id','study_id','study_pubyear', 'treatment', 'Lat_long', 'crop_type','
#                          'yield_treatment_kgha', 'measurement_id')
#
# means.check.2$ID<-paste0(means.check.2$study_pubyear, means.check.2$treatment, means.check.2$Lat_long
# means.check$ID<-  paste0(means.check$study_pubyear, means.check$treatment, means.check$Lat_long,means

## STOP ALFRED'S CODE:
```

```
data.mod.2$Lat_long<-
  paste0(data.mod.2$latitude_decimal, data.mod.2$longitude_decimal)

means.check <-
data.mod.2 %>%
  group_by(ma_id, study_id, study_pubyear, treatment, Lat_long, crop_type) %>%
```

```
  summarize(
    mean_yield_control_kgha = mean(yield_control_kgha, na.rm = TRUE),
    mean_yield_treatment_kgha = mean(yield_treatment_kgha, na.rm = TRUE),
    mean_measurement_id = mean(measurement_id, na.rm = TRUE)
  ) %>%
  ungroup()
```

```
## `summarise()` has grouped output by 'ma_id', 'study_id', 'study_pubyear',
## 'treatment', 'Lat_long'. You can override using the `.groups` argument.
```

```
means.check$ID<-paste0(means.check$study_pubyear, means.check$treatment,
                       means.check$Lat_long,means.check$crop_type)

x<-as.data.frame(table(means.check$ID))
which(x$Freq>1)
```

```
##  [1]   34   84  140  248  261  314  319  334  335  339  412  469  476  610  611
## [16]  615  630  646  668  704  725  734  735  829  877  917  975  976  979  980
## [31]  982  984  990  991 1028 1117 1123 1136 1140 1142 1143 1186 1206 1208 1211
## [46] 1226 1257 1271 1362 1385
```

```
# alfreds comment:
# identify the double entries - where the everything is the same
#('study_pubyear', 'treatment', 'Lat_long', 'crop_type' and there are still two entries)


dublicates <-
means.check %>%
  group_by(ID) %>%
  filter(n() > 1) %>%
  ungroup() %>%
  arrange(ID)

dublicates_ID <-
  dublicates %>%
  pull(ID)


data_unchanged_means.check <-
  data_unchanged %>%
  mutate(Lat_long = paste0(latitude_decimal, longitude_decimal),
         ID = paste0(study_pubyear, treatment, Lat_long, crop_type))

data_unchanged_means.check %>%
  filter(ID %in% dublicates_ID) %>%
  select(-(class:(ncol(data_unchanged_means.check)-1))) %>%
  unique() %>%
  select(ma_id, measurement_id, author_year, landcover_map_year,
         yield_control_kgha, yield_treatment_kgha) %>%
  head(10)
```

```
##     ma_id measurement_id     author_year landcover_map_year yield_control_kgha
```

```
## 1    D352           1363 Sui et al. 2012              2005              6540
## 109  D352           1364 Sui et al. 2012              2005              6104
## 217  D352           1366 Sui et al. 2012              2005              7185
## 325  D352           1368 Sui et al. 2012              2005              8702
## 433  D352           1371 Sui et al. 2012              2005              8702
## 541  D352           1372 Sui et al. 2012              2005              6540
## 649  D352           1373 Sui et al. 2012              2005              6104
## 757  D352           1376 Sui et al. 2012              2005              7185
## 865  D352           1380 Sui et al. 2012              2005              6104
## 973  D352           1381 Sui et al. 2012              2005              7185
##     yield_treatment_kgha
## 1                   5610
## 109                 5411
## 217                 6571
## 325                 8019
## 433                 6518
## 541                 5497
## 649                 5261
## 757                 6296
## 865                 3258
## 973                 3842
```

As you can see, these entries are not duplicates (I think the easiest way is to identify it by the yield_control_kgha, yield_treatment_kgha combination). They indeed share the publication year, location, treatment and crop, but come from different studies. So they do not have to be excluded.

(D) Average replicates that use the same location and the same land-cover map

We need: - (i) The mean of the two yields. - (ii) The combination of the standard deviation. - (iii) The sum of the sample sizes. - (iv) The number of aggregated unique measurement IDs.

Again, I silenced Alfred's code, and did the same operation in the next code section.

```
# Alfred's code:

# means<-aggregate(list(data.mod.2$yield_control_kgha, data.mod.2$yield_treatment_kgha),
#                          by = list(data.mod.2$landcover_map_year, data.mod.2$treatment, data.mod.2$Lat_
#                                    data.mod.2$crop_type),   function(x){mean(x, na.rm = T)})
# colnames(means)<-c('landcover_map_year','treatment', 'Lat_long', 'crop_type', 'yield_control_kgha', '

# @Elizaveta: ATTENTION: I dont get why for Sd treatment yield_treatment is used

# add sds of yield
# sds.yield<-aggregate(list(data.mod.2$yield_SD_control, data.mod.2$yield_treatment),
#                          by = list(data.mod.2$landcover_map_year, data.mod.2$treatment, data.mod.2$Lat_
#                                    data.mod.2$crop_type),   function(x){ sqrt(sum(x^2))})
# colnames(sds.yield)<-c('landcover_map_year','treatment', 'Lat_long', 'crop_type', 'yield_SD_control',

# means$yield_SD_control<- sds.yield$yield_SD_control
# means$yield_SD_treatment<- sds.yield$yield_treatment
#
# # sum sample size of yield
# n.yield<-aggregate(list(data.mod.2$control_replicates, data.mod.2$treatment_replicates),
```

```
#                       by = list(data.mod.2$landcover_map_year, data.mod.2$treatment, data.mod.2$Lat_lo
#                                 data.mod.2$crop_type),   function(x){sum(x)})
# colnames(n.yield)<-c('landcover_map_year','treatment', 'Lat_long', 'crop_type', 'replicates_control_s
#
# means$replicates_control_summed<- n.yield$replicates_control_summed
# means$replicates_treatment_summed<- n.yield$replicates_treatment_summed
#
# # sum sample size of yield
# n.aggregated<-aggregate(list(data.mod.2$control_replicates),
#                     by = list(data.mod.2$landcover_map_year, data.mod.2$treatment, data.mod.2$Lat_long
#                               data.mod.2$crop_type),   function(x){length(x)})
# colnames(n.aggregated)<-c('landcover_map_year','treatment', 'Lat_long', 'crop_type', 'n_aggregated')
#
# means$n_aggregated<- n.aggregated$n_aggregated
```

```
means <-
  data.mod.2 %>%
  group_by(landcover_map_year, treatment, Lat_long, crop_type) %>%
  summarize(
    mean_yield_control_kgha = mean(yield_control_kgha, na.rm = TRUE),
    mean_yield_treatment_kgha = mean(yield_treatment_kgha, na.rm = TRUE),
    yield_SD_control = sqrt(sum(yield_SD_control_kgha^2)),
    yield_SD_treatment = sqrt(sum(yield_SD_treatment_kgha^2)),
    replicates_control_summed = sum(control_replicates),
    replicates_treatment_summed = sum(treatment_replicates),
    n_aggregated = length(control_replicates)
  ) %>%
  ungroup()
```

```
## `summarise()` has grouped output by 'landcover_map_year', 'treatment',
## 'Lat_long'. You can override using the `.groups` argument.
```

(E) Add additional columns to aggregated data

Alfred's comment: add remaining data to the average data. It should be okay to pull in things that have the same latidude-longitude combination and the same land-cover map-year.

```
means<-cbind(means, data.mod.2[match(paste0(means$Lat_long,
                                            means$landcover_map_year, means$treatment,
                                            means$crop_type),
                                     paste0(data.mod.2$Lat_long,
                                            data.mod.2$landcover_map_year,
                                            data.mod.2$treatment,
                                            data.mod.2$crop_type)),
                                c(1:8,11:12,29:37,41:ncol(data.mod.2))])

# Now landcover_map_year and Lat_long are duplicated
# lets remove the duplicates of these columns.
means <- means[, -c(354, 355)]

data.mod.2$reference[data.mod.2$reference=='']<-NA
length(which(is.na(data.mod.2$reference)))/nrow(data.mod.2)
```

```
## [1] 0.6389447
```

```
# compute the log-response ratio
means$logrr.yi <- log10(means$mean_yield_control_kgha/means$mean_yield_treatment_kgha)
```

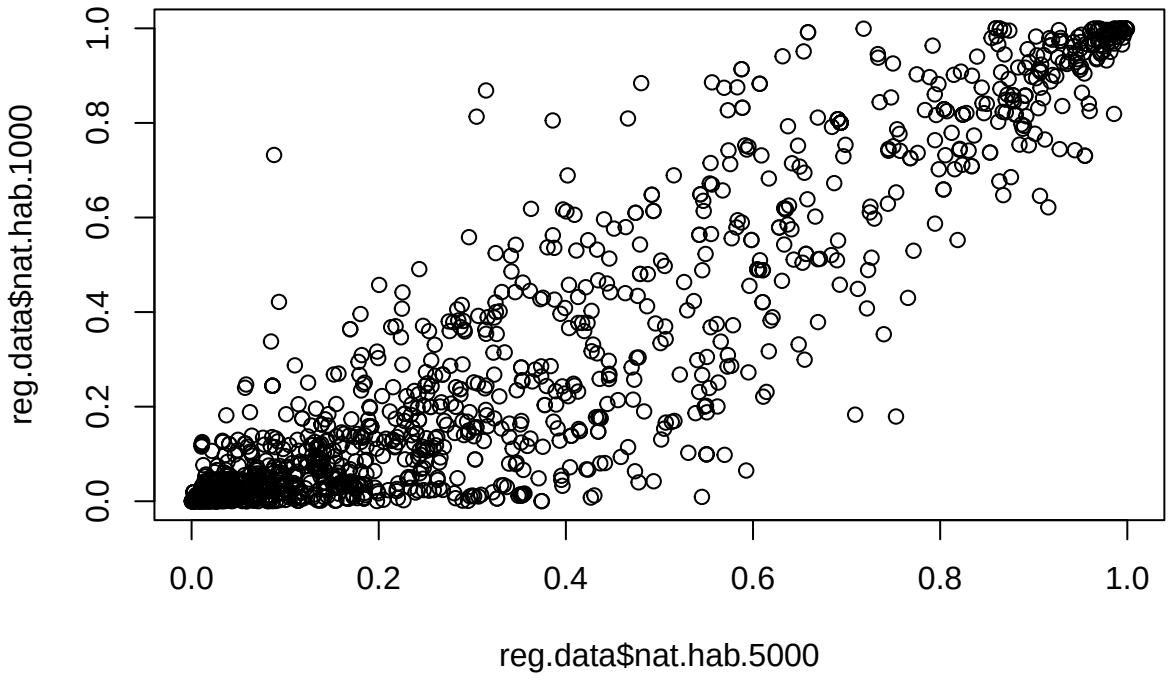
Some references are given in the title or author year columns. Maybe it makes sense to unite them, because the proportion calculated before is not representative.

(III) Prepare predictors

```
# all original properties -> without area, edge length  and proportion:
reg.data<-means[,c(1:28)]

col.names<-colnames(means)
```

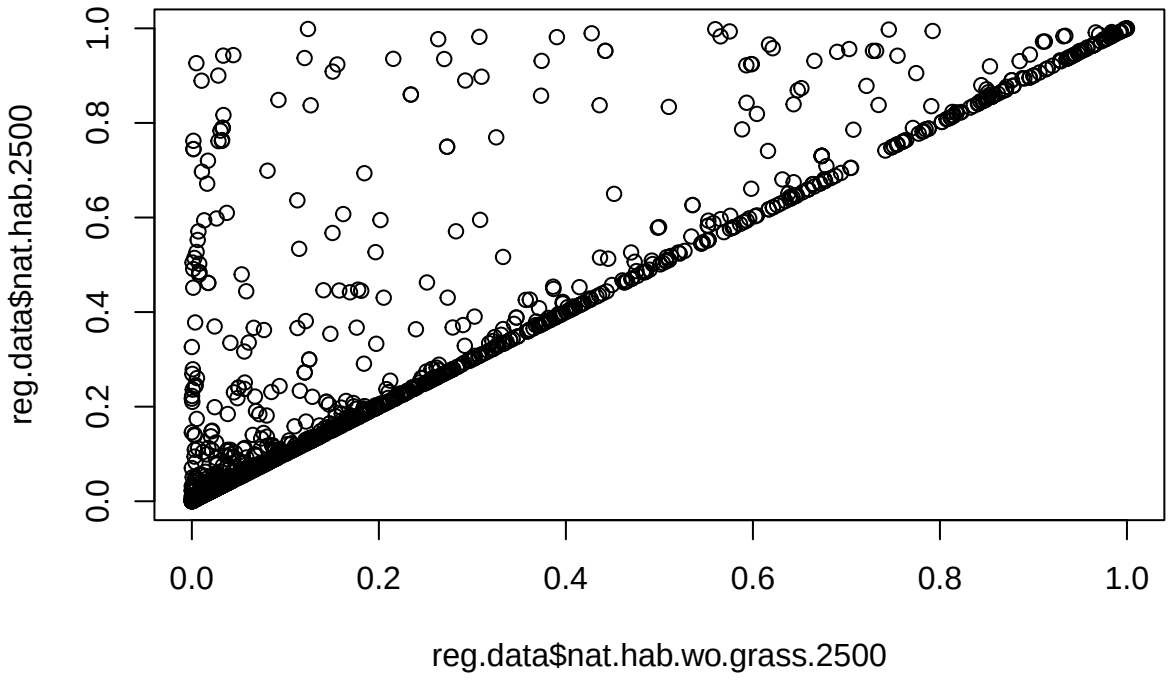
(A) create the semi natural habitat proportion (including grassland)

```
# (i) get the classes which need to be included
nat.hab.class.code<-landcover.meta$Class.Code[-which(
  landcover.meta$Bigger.CLass=='Bare Surfaces'|
  landcover.meta$Class.Description.by.ESA=='Permanent ice and snow'|
  landcover.meta$Bigger.CLass=='Cropland' )]

# (ii) natural habitat for the three different radii: get the column names for the 1000 radius
# create the column names that should be included
nat.hab.col.names<-c(paste0('proportion_',nat.hab.class.code,'_1000'))

# create the nat. habitat variable
reg.data$nat.hab.1000<-rowSums(means[,which(is.element(col.names,nat.hab.col.names))], na.rm = T)

# (iii) now for the other two radii
 nat.hab.col.names<-c(paste0('proportion_',nat.hab.class.code,'_2500'))
 reg.data$nat.hab.2500<-rowSums(means[,which(is.element(col.names,nat.hab.col.names))], na.rm = T)

nat.hab.col.names<-c(paste0('proportion_',nat.hab.class.code,'_5000'))
reg.data$nat.hab.5000<-rowSums(means[,which(is.element(col.names,nat.hab.col.names))], na.rm = T)

plot(reg.data$nat.hab.5000, reg.data$nat.hab.1000) # looks right
```

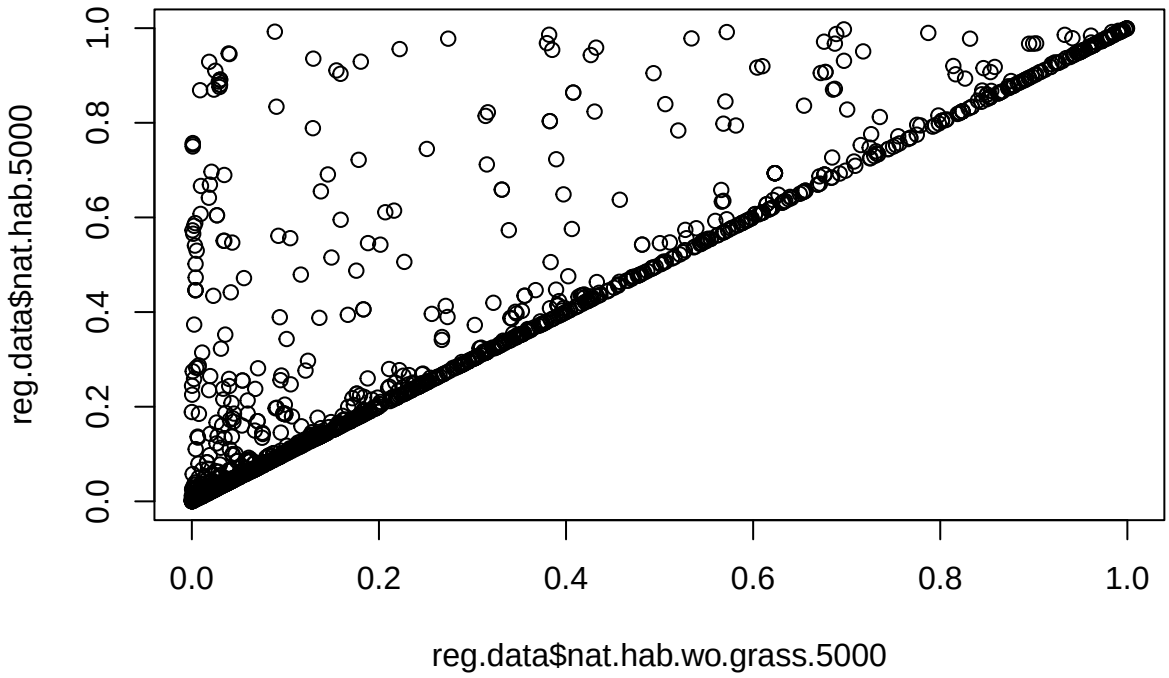


(B) create the semi natural habitat proportion (now without grassland)

```
# (i) get the classes which need to be included
nat.hab.class.code<-landcover.meta$Class.Code[-which(
  landcover.meta$Bigger.CLass=='Bare Surfaces'|
    landcover.meta$Class.Description.by.ESA=='Permanent ice and snow'|
    landcover.meta$Bigger.CLass=='Grassland' |
    landcover.meta$Bigger.CLass=='Cropland' )]

# (ii) natural habitat for the three different radii:
# create the column names that should be included
nat.hab.col.names<-c(paste0('proportion_',nat.hab.class.code,'_1000'))
reg.data$nat.hab.wo.grass.1000<-
  rowSums(means[,which(is.element(col.names,nat.hab.col.names))], na.rm = T)

# now for the other two radii
nat.hab.col.names<-c(paste0('proportion_',nat.hab.class.code,'_2500'))
reg.data$nat.hab.wo.grass.2500<-rowSums(means[,which(is.element(col.names,nat.hab.col.names))], na.rm =

nat.hab.col.names<-c(paste0('proportion_',nat.hab.class.code,'_5000'))
reg.data$nat.hab.wo.grass.5000<-
  rowSums(means[,which(is.element(col.names,nat.hab.col.names))], na.rm = T)

plot(reg.data$nat.hab.wo.grass.2500, reg.data$nat.hab.2500)
```



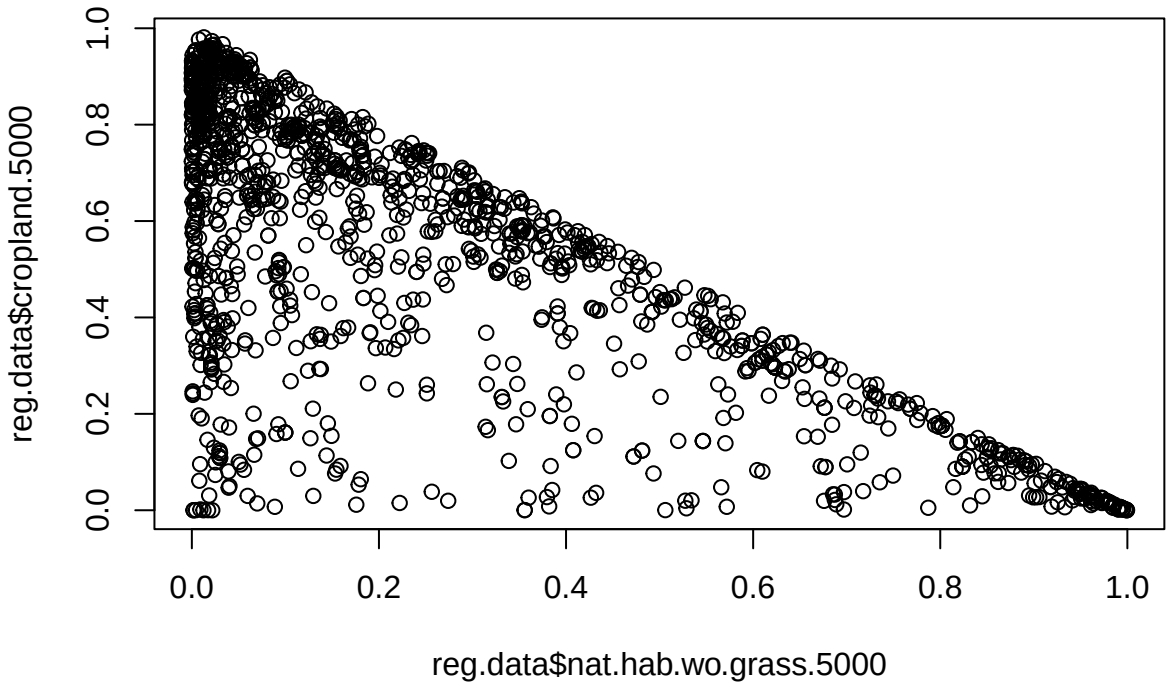

```
plot(reg.data$nat.hab.wo.grass.5000, reg.data$nat.hab.5000)
```



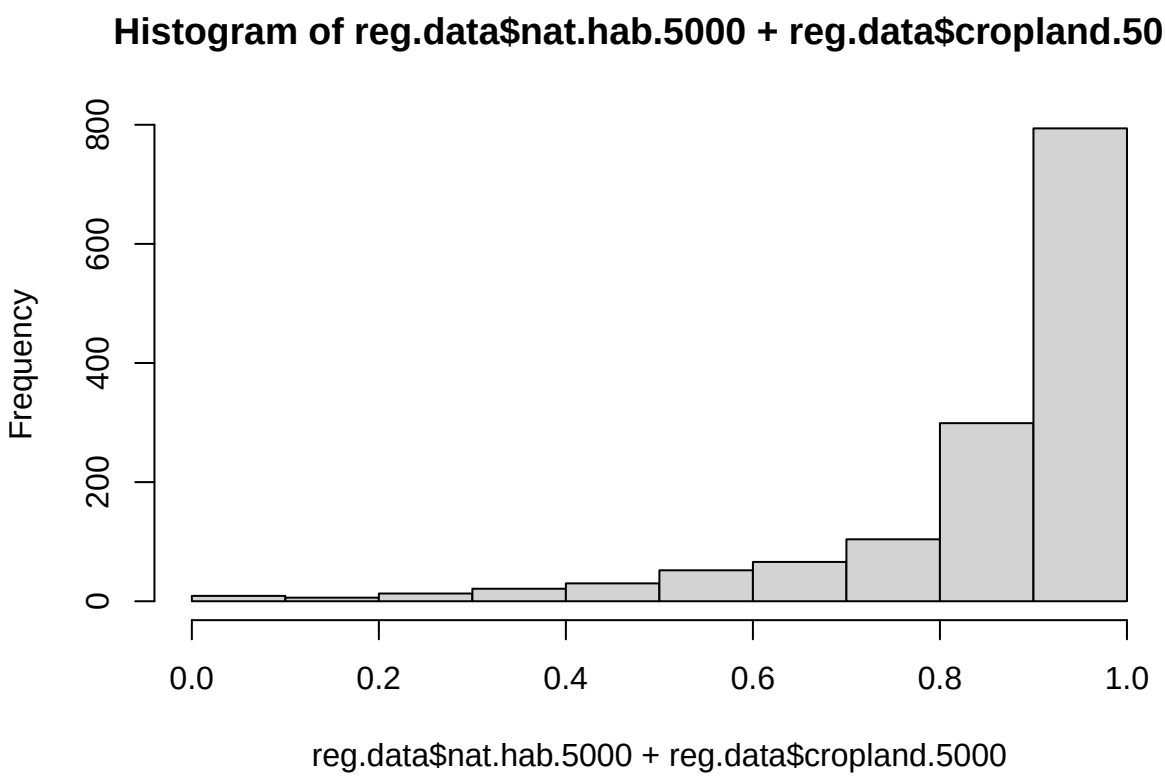
Looks nice and right on the first glance

(C) create the cropland proportion data

```
# (i) get the classes which need to be included and create their col-names
cropland.class.code<-landcover.meta$Class.Code[which(landcover.meta$Bigger.CLass=='Cropland')]
names<-c(paste0('proportion_',cropland.class.code,'_1000'))

# (ii) create the data for the 1km radius
reg.data$cropland.1000<-rowSums(means[,which(is.element(col.names,names))], na.rm = T)

# (iii) repeat that for the 2 and 5km radii
 names<-c(paste0('proportion_',cropland.class.code,'_2500'))
 reg.data$cropland.2500<-rowSums(means[,which(is.element(col.names,names))], na.rm = T)

names<-c(paste0('proportion_',cropland.class.code,'_5000'))
reg.data$cropland.5000<-rowSums(means[,which(is.element(col.names,names))], na.rm = T)

plot(reg.data$nat.hab.wo.grass.5000, reg.data$cropland.5000)
```




```
hist(reg.data$nat.hab.5000 + reg.data$cropland.5000)
```




```
summary(reg.data$nat.hab.5000 + reg.data$cropland.5000)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## 0.002308 0.826028 0.920127 0.858694 0.974408 1.000000
```

```
summary(reg.data$nat.hab.2500 + reg.data$cropland.2500)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## 0.0000331 0.8002804 0.9141443 0.8348084 0.9750768 1.0000000
```

```
summary(reg.data$nat.hab.1000 + reg.data$cropland.1000)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0000  0.7782  0.9225  0.8146  0.9791  1.0000
```

```
# all looks nice and right on the first glance
```

(D) create the cropland area vs perimeter data

```
#colnames(means)

# (i) create the column names
names<-c(paste0('area_m2_',cropland.class.code,'_1000'))
names.per<-c(paste0('edgelength_m_',cropland.class.code,'_1000'))

# (ii) create the variable
reg.data$crop.peri.area.ratio.1000 <-
  rowSums(means[,which(is.element(col.names,names.per))], na.rm = T) /
  rowSums(means[,which(is.element(col.names,names))], na.rm = T) # unit: m/m2

# we have some data points where both area and edgelength of agricultural land is 0
x<-rowSums(means[,which(is.element(col.names,names))], na.rm = T)
y<-rowSums(means[,which(is.element(col.names,names.per))], na.rm = T)

which(x==0); which(y==0)
```

```
##  [1]  225  257  258  324  523  617  632  633  689  716  871 1023 1024 1308 1375
## [16] 1378 1383
```

```
##  [1]  225  257  258  324  523  617  632  633  689  716  871 1023 1024 1308 1375
## [16] 1378 1383
```

```
# for now, we can exclude these points? But maybe we can look into this problem?
# we do the exclusion at the end...
```

The rows that have area = 0 are the same rows as edge = 0.

Lets export these coordinates and check them in gee. * In gee those are normal points in Mongolia, China, Africa. * Maybe it makes more sense to check the composition, the treatment and the crop of the observation.

```
means %>%
  select(measurement_id, crop_type_grouped_small, treatment,
         contains("area")&contains("1000")) %>%
  filter(if_all(all_of(names), ~ . == 0)) %>%
  select(measurement_id)
```

```
##    measurement_id
## 1            8963
## 2            8225
## 3            8244
## 4            9004
## 5            6730
## 6            6435
## 7            8281
## 8            8283
## 9            1743
## 10           8971
## 11           6605
## 12           4861
## 13           4791
## 14           2111
```

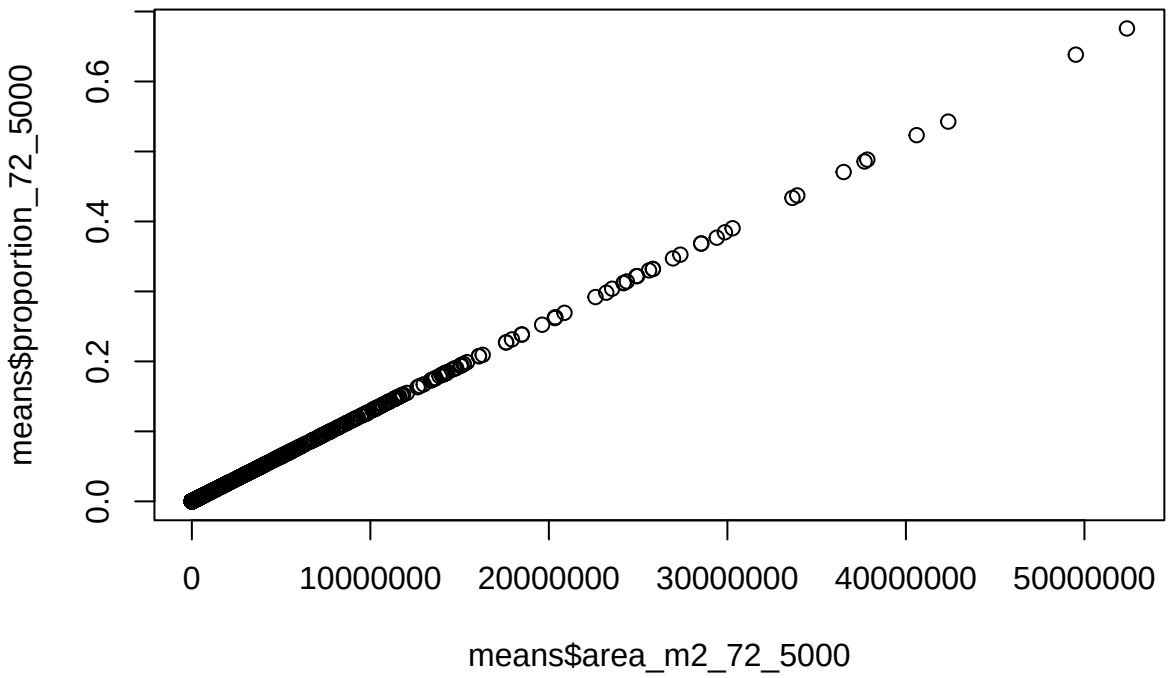
```
## 15           2878
## 16           3070
## 17           3367
```

```
measurement_ids_zeroagriarea <-
  means %>%
  select(measurement_id, crop_type_grouped_small, treatment,
         contains("area")&contains("1000")) %>%
  filter(if_all(all_of(names), ~ . == 0)) %>%
  pull(measurement_id)

long_data <- means %>%
  select(measurement_id, crop_type_grouped_small, treatment, landcover_map_year,
         contains("area") & contains("1000")) %>%
  filter(if_all(starts_with("area"), ~ . == 0)) %>%
  pivot_longer(cols = starts_with("area"),
               names_to = "area_class",
               values_to = "area_m2")
```

long_data has no entries, meaning that the points that have no agricultural area still have other land cover classes, and are not empty entries.

Check whether the proportion data matches with the area data:

```
plot(means$area_m2_72_5000 , means$proportion_72_5000)
```



yes - it does, good nicely!

```
# (iii) repeat that for the 2 and 5km radii
 names<-c(paste0('area_m2_',cropland.class.code,'_2500'))
 names.per<-c(paste0('edgelength_m_',cropland.class.code,'_2500'))
 reg.data$crop.peri.area.ratio.2500 <-
   rowSums(means[,which(is.element(col.names,names.per))], na.rm = T) /
 rowSums(means[,which(is.element(col.names,names))], na.rm = T) # unit: m/m2

# we have some data points where both area and edgelength of agricultural land is 0
# x<-rowSums(means[,which(is.element(col.names,names))], na.rm = T)
# y<-rowSums(means[,which(is.element(col.names,names.per))], na.rm = T)
#
# which(x==0); which(y==0)

means %>%
  select(measurement_id, crop_type_grouped_small, treatment,
         contains("area")&contains("2500")) %>%
  filter(if_all(all_of(names), ~ . == 0)) %>%
  select(measurement_id)
```

```
##   measurement_id
## 1           8281
## 2           8283
## 3           2878
## 4           3070
## 5           3367
```

```
names<-c(paste0('area_m2_',cropland.class.code,'_5000'))
names.per<-c(paste0('edgelength_m_',cropland.class.code,'_5000'))
reg.data$crop.peri.area.ratio.5000 <-
  rowSums(means[,which(is.element(col.names,names.per))], na.rm = T) /
  rowSums(means[,which(is.element(col.names,names))], na.rm = T) # unit: m/m2

 # we have some data points where both area and edgelength of agricultural land is 0
# x<-rowSums(means[,which(is.element(col.names,names))], na.rm = T)
# y<-rowSums(means[,which(is.element(col.names,names.per))], na.rm = T)
#
# which(x==0); which(y==0)

means %>%
  select(measurement_id, crop_type_grouped_small, treatment, contains("area")&contains("5000")) %>%
  filter(if_all(all_of(names), ~ . == 0)) %>%
  select(measurement_id)
```

```
##   measurement_id
## 1           2878
## 2           3070
## 3           3367
```

Measurements 8963, 8225, 8244, 9004, 6730, 6435, 8281, 8283, 1743, 8971, 6605, 4861, 4791, 2111, 2878, 3070, 3367 had problems with the buffer 1000.

Measurements 8281, 8283, 2878, 3070, 3367 had problems with the buffer 2500.

Measurements 2878, 3070, 3367 had problems with the buffer 5000.

Meaning: - (1) The agricultural land doesn't exist for 17 measurements. - (2) `measurement_id` 2878, 3070, and 3367 do not have agricultural land in any buffer. - (3) 8281 and 8283 no agri area for *1000m* and *2500m* buffers. - (4) 8225, 8244, 9004, 6730, 6435, 1743, 8971, 6605, 4861, 4791, and 2111 no agricultural land in *1000m* buffer.

(E) create the agricultural edge-length

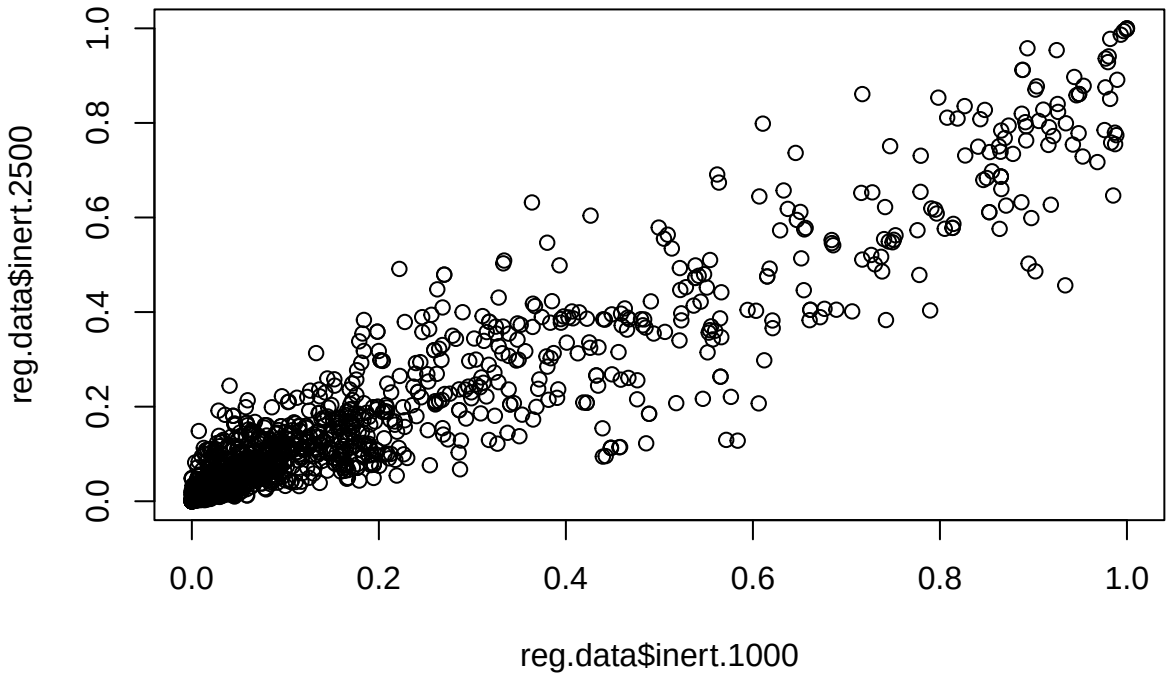
```
names.per<-c(paste0('edgelength_m_',cropland.class.code,'_1000'))
reg.data$crop.edgelength.1000 <-
  rowSums(means[,which(is.element(col.names,names.per))], na.rm = T)

names.per<-c(paste0('edgelength_m_',cropland.class.code,'_2500'))
reg.data$crop.edgelength.2500 <-
  rowSums(means[,which(is.element(col.names,names.per))], na.rm = T)

names.per<-c(paste0('edgelength_m_',cropland.class.code,'_5000'))
reg.data$crop.edgelength.5000 <-
  rowSums(means[,which(is.element(col.names,names.per))], na.rm = T)
```

(F) create the inert land-cover

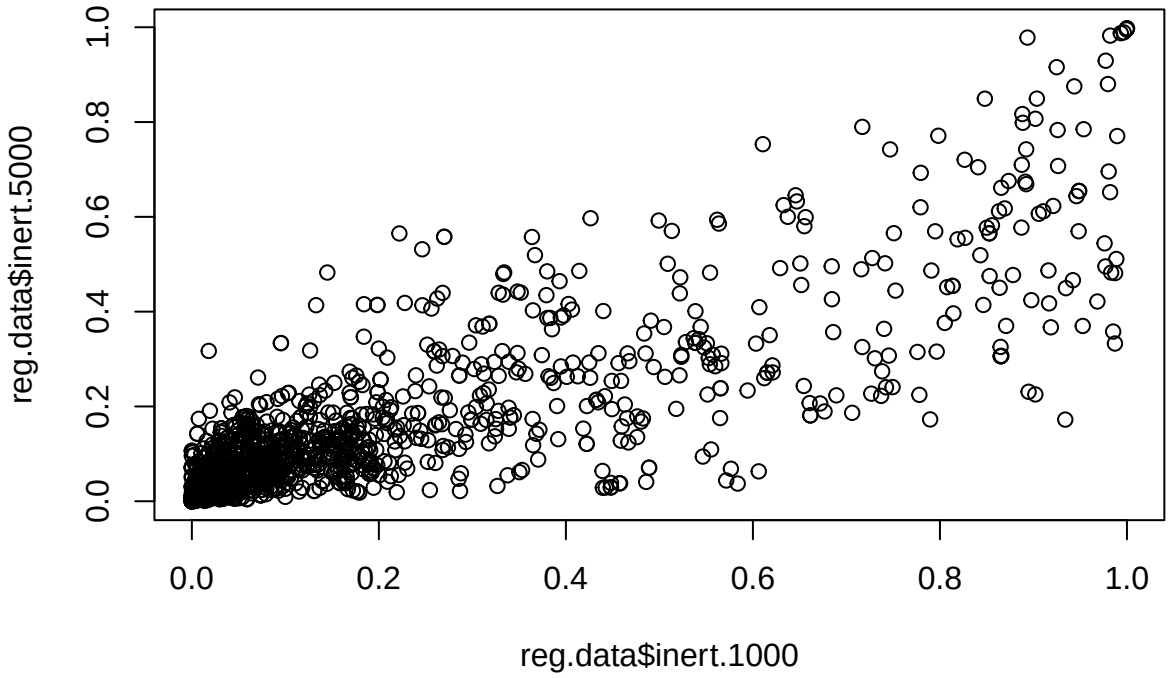
```
# (i) get the classes which need to be included and create their col-names
inert.land.class.code<-landcover.meta$Class.Code[-which(
  landcover.meta$Bigger.CLass=='Bare Surfaces'|
  landcover.meta$Class.Description.by.ESA=='Permanent ice and snow')]
```

Comment to the following code section: The urban areas and other built environments ARE included in the land cover classes. They are called "Impervious surfaces" by teh ESA Classification, have the code 190 and "Bare Surfaces" in Bigger.CLass.

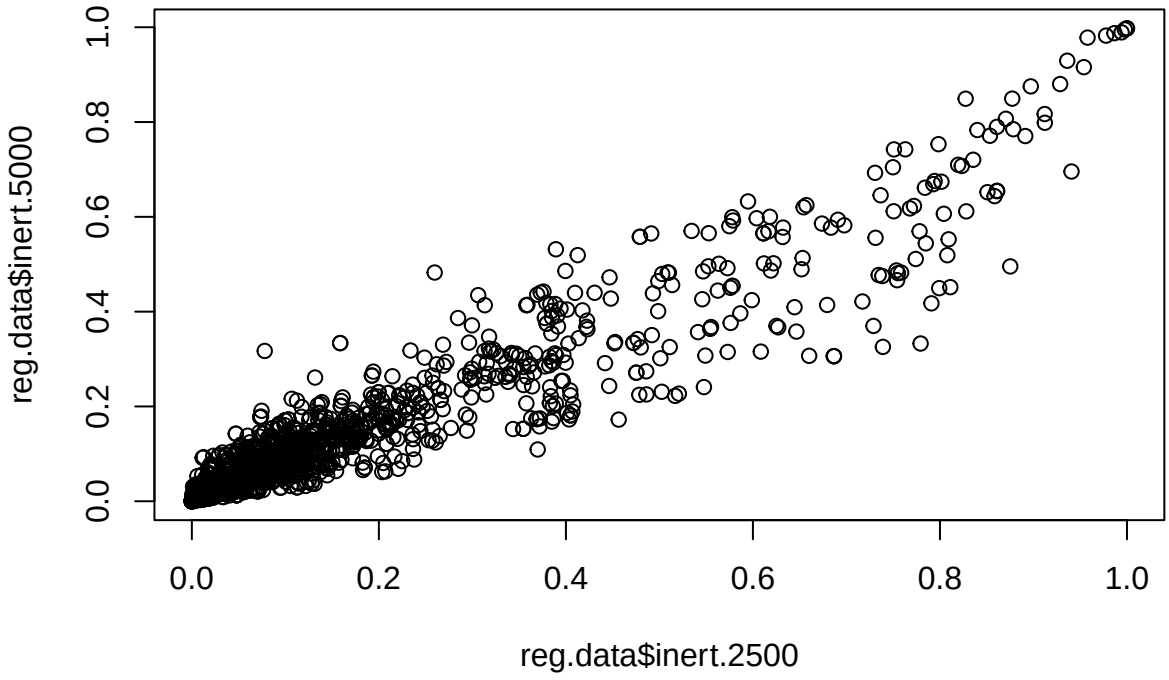
```
# Alfred's comment:
# (ii) create the data for the 1km radius - here we also have urban areas and other built environments,
# not included in our land-cover classes, so we have to use the not inert area and subtract it from 1 t
# right value.
names<-c(paste0('proportion_',inert.land.class.code,'_1000'))
reg.data$inert.1000<- 1-rowSums(means[,which(is.element(col.names,names))], na.rm = T)

# (iii) repeat that for the 2 and 5km radii
names<-c(paste0('proportion_',inert.land.class.code,'_2500'))
reg.data$inert.2500<- 1-rowSums(means[,which(is.element(col.names,names))], na.rm = T)

names<-c(paste0('proportion_',inert.land.class.code,'_5000'))
reg.data$inert.5000<- 1-rowSums(means[,which(is.element(col.names,names))], na.rm = T)

# looks good, but we have some very high inert values...
plot(reg.data$inert.1000, reg.data$inert.2500)
```




```
plot(reg.data$inert.1000, reg.data$inert.5000)
```




```
plot(reg.data$inert.2500, reg.data$inert.5000)
```




```
which(reg.data$inert.1000>0.8) # maybe check these points once more?
```

```
##  [1]   19   22   23   39   40   41   47   48   49   96  103  105  131  167  182
## [16]  207  209  243  244  257  258  285  301  304  328  334  360  362  363  377
## [31]  383  402  453  481  555  556  564  568  570  586  621  632  633  666  667
## [46]  668  689  691  756  773  817  825  867  936 1021 1035 1054 1080 1098 1101
## [61] 1111 1151 1154 1170 1176 1179 1200 1201 1213 1215 1236 1243 1272 1304 1308
## [76] 1341 1342 1343 1346 1375 1378
```

```
names<-c(paste0('proportion_',inert.land.class.code,'_1000'))

means %>%
  select(measurement_id, country_new, crop_type_grouped_small, treatment, contains("proportion_"), conta
  rowwise() %>%
  filter((1 - sum(c_across(names), na.rm = TRUE)) > 0.8) %>%
  select(measurement_id)
```

```
## Warning: There was 1 warning in `filter()`.
## i In argument: `(1 - sum(c_across(names), na.rm = TRUE)) > 0.8`.
## i In row 1.
## Caused by warning:
## ! Using an external vector in selections was deprecated in tidyselect 1.1.0.
## i Please use `all_of()` or `any_of()` instead.
##   # Was:
##   data %>% select(names)
```

```
##
##   # Now:
##   data %>% select(all_of(names))
##
## See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
```

```
## # A tibble: 81 x 1
## # Rowwise:
##    measurement_id
##             <int>
##  1           7871
##  2           1274
##  3           4940
##  4           8422
##  5           8421
##  6           8418
##  7           8321
##  8           8409
##  9           8405
## 10           2857
## # i 71 more rows
```

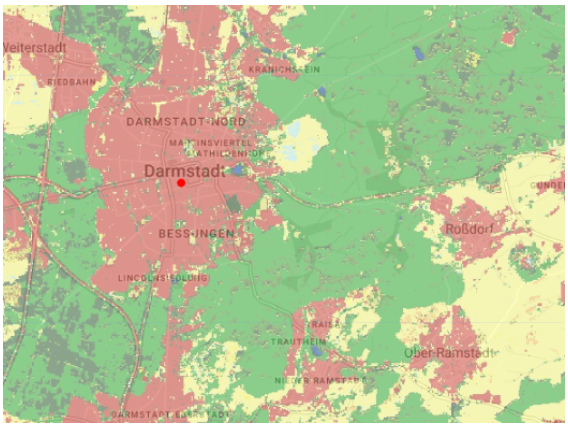
```
measurement_ids_biginertproportion <-
means %>%
  select(measurement_id, country_new, crop_type_grouped_small, treatment, contains("proportion_"), conta
  rowwise() %>%
  filter((1 - sum(c_across(names), na.rm = TRUE)) > 0.8) %>%
  pull(measurement_id)
```

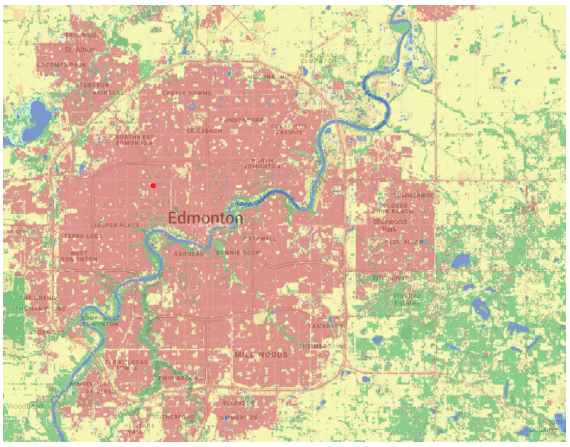
Some points from the previous analysis with 0 in agricultural area are there (9 measurements out of 81):

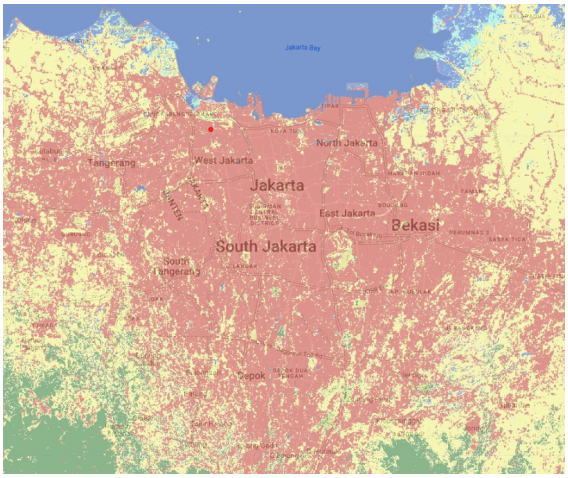
```
# Find common measurement_ids between the two sets
intersect(measurement_ids_biginertproportion, measurement_ids_zeroagriarea)
```

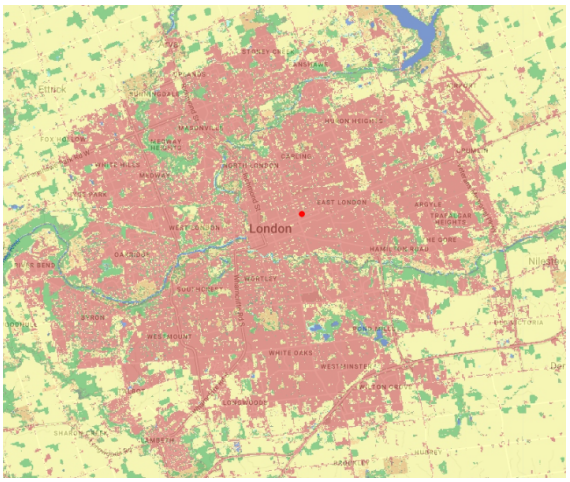
```
## [1] 8225 8244 8281 8283 1743 2111 2878 3070
```

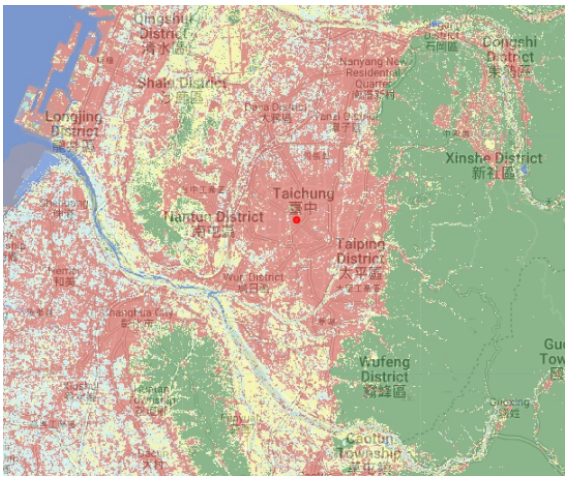
```
x <-
means %>%
  select(measurement_id, country_new, crop_type_grouped_small, treatment, landcover_map_year, contains(
  rowwise() %>%
  filter((1 - sum(c_across(names), na.rm = TRUE)) > 0.8)
```

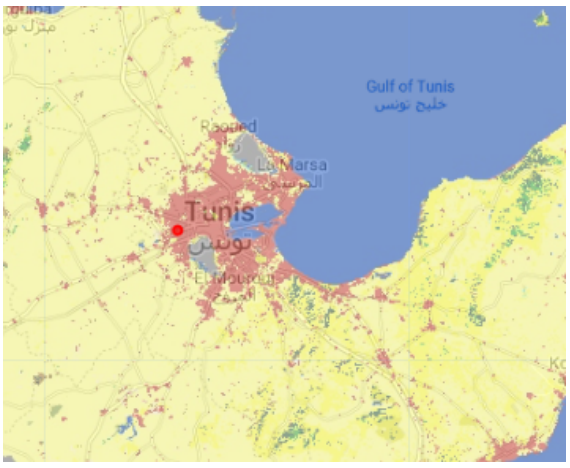

Darmstadt
DARMSTADT-NORD
BESSUNGEN
Roßdorf
Ober-Ramstadt

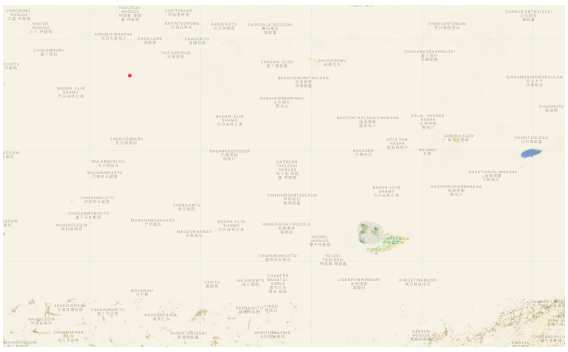

Edmonton


Jakarta Bay
Tangerang
West Jakarta
North Jakarta
Jakarta
East Jakarta
Bekasi
South Jakarta
Depok


London


Qingshui District
Shalu District
Longjing District
Dajia District
Daya District
Dadu District
Houli District
Taichung
Tantzu District
Xinshe District
Taiping District
Wuri District
Wufeng District
Caotun Township


Gulf of Tunis
La Marsa
Tunis
El Mourouj



(H) create the shannon index

```
# (i) define the variables that need to be included
names<-c(paste0('area_m2_',inert.land.class.code,'_1000'))
# inert.land.class.code - contains all non inert land-cover types

# (ii) create the data frame for the analysis
shannon.df<-means[,which(is.element(col.names,names))]
shannon.df$inert<- reg.data$inert.1000

# (iii) replace NAs with Os
for(i in 1:ncol(shannon.df)){shannon.df[which(is.na(shannon.df[,i])) , i]<-0}

# (iv) calculate the shannon diversity
library(vegan) # 'species' need to be the columns
```

```
## Loading required package: permute
```

```
## Loading required package: lattice
```

```
## This is vegan 2.6-6.1
```

```
reg.data$shannon.1000<-diversity(shannon.df, index = "shannon")

# (v) repeat for the other two radii
names<-c(paste0('area_m2_',inert.land.class.code,'_2500'))
shannon.df<-means[,which(is.element(col.names,names))]
shannon.df$inert<- reg.data$inert.2500
for(i in 1:ncol(shannon.df)){shannon.df[which(is.na(shannon.df[,i])) , i]<-0}
reg.data$shannon.2500<-diversity(shannon.df, index = "shannon")

names<-c(paste0('area_m2_',inert.land.class.code,'_5000'))
shannon.df<-means[,which(is.element(col.names,names))]
shannon.df$inert<- reg.data$inert.5000
for(i in 1:ncol(shannon.df)){shannon.df[which(is.na(shannon.df[,i])) , i]<-0}
reg.data$shannon.5000<-diversity(shannon.df, index = "shannon")
```

```r
# Diagnostic check whether we have NAs - we should not have them
which(is.na(reg.data$inert.1000))
```

```
## integer(0)
```

```r
which(is.na(reg.data$cropland.1000))
```

```
## integer(0)
```

```r
which(is.na(reg.data$crop.edgelength.1000))
```

```
## integer(0)
```

```r
which(is.na(reg.data$nat.hab.1000))
```

```
## integer(0)
```

```r
which(is.na(reg.data$nat.hab.wo.grass.1000))
```

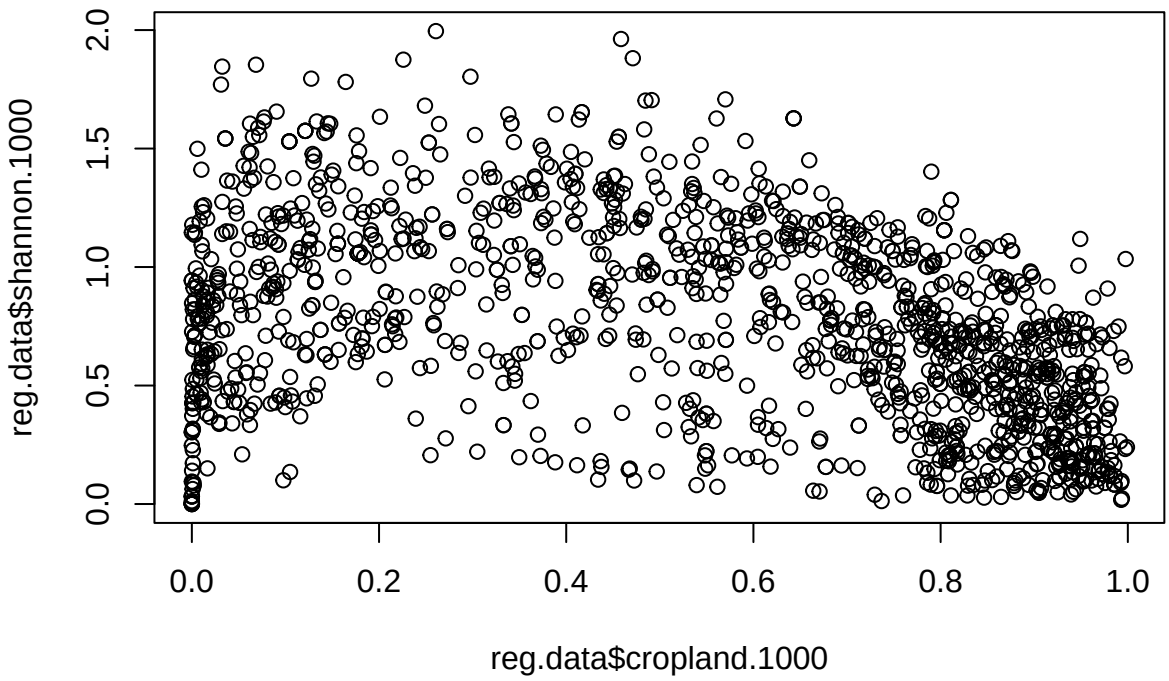
```
## integer(0)
```

```r
# that all looks good
```

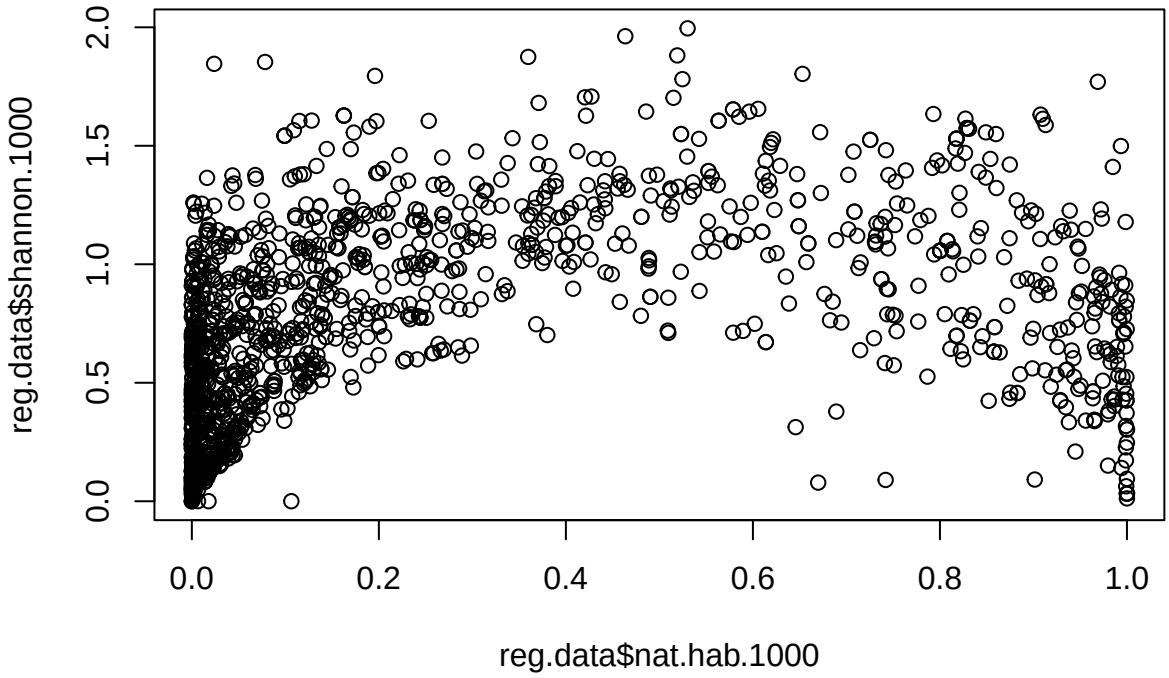
```r
# clean up some objects that are not needed anymore
rm(nat.hab.col.names, nat.hab.class.code, cropland.class.code,names,
   names.per, inert.land.class.code, shannon.df)
```

```r
# save the file
#write.csv(reg.data, file = 'data/20241026_data_processed.csv',row.names = TRUE)
```

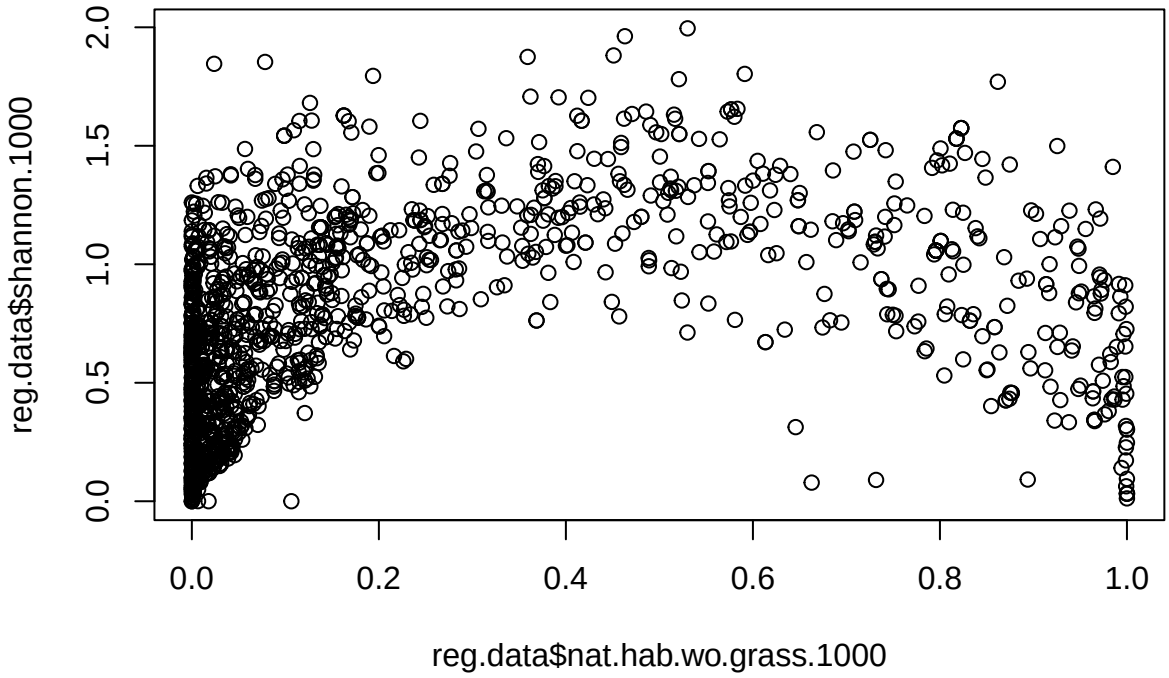
```
plot(reg.data$shannon.1000~reg.data$cropland.1000)
```




```
plot(reg.data$shannon.1000~reg.data$nat.hab.1000)
```




```
plot(reg.data$shannon.1000~reg.data$nat.hab.wo.grass.1000)
```




```
# check how many data points we have per study now...
summary(as.vector(table(paste0(reg.data$study_id,reg.data$ma_id))))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   1.000   1.000   1.808   2.000  57.000
```

Although I am not sure whether it makes sense to count points per study since study_id is not unique across the dataset.

```
write.csv(reg.data, "C:\\Users\\lisa7\\Documents\\surrounding_landscapes\\data\\20241027_data_processed
```