Selection of locations for insect monitoring in oaks

Jens Å

21 February, 2024

Contents

Set up a cache variable for saving intermediate work. Set it to false to use stored intermediates. Set to TRUE to rerun from scratch.

```
cache_var <- FALSE
```

```
con <- NinaR::postgreSQLConnect()
```

Plan

Task, from Rannveig's notes.

Vi skal trekke 150 eiker (100 til overvåkingen, men må ha ekstra for å justere i forhold til logistikk, grunneiertillatelse og i felt feks om noen trær er borte) fra de 600 ARKO-eikene (657 minus «gone» og «not found» i 2019), etter følgende kriterier:

Doblet sannsynlighet for å trekke trær med omkrets over 200 cm.

Andel eiker i vårt utvalg speiler fordelingen blant alle ARKO-eiker; hvis det feks er 20% i Vestland, 30% i Agder, 30% i V/T, 20% i Viken, så skal overvåkingseikene fordeles etter samme andeler.

Maks 2 overvåkingstrær per ARKO-rute.

Da hule eiker i utgangspunktet er relativt jevnt fordelt mellom skog og åpent landskap, regner vi med at dette vil reflekteres i utvalget av overvåkingstrær uten å legge inn noe styrende kriterium for dette. Vi sjekker om vi har tilfredsstillende fordeling av eiker i skog og utenfor skog totalt og i hver region etter å ha trukket et sett overvåkingstrær.

Etter å ha gjort et utvalg sjekker vi fordeling i forhold til ulike parametre (feks hulrom, vedmuld, barktype, treform), særlig hvor mange A-eiker som er representert.

Add by Jens In addition to this, we will only consider squares with a single tree if that square isn't too far away from other squares (other chosen squares?)

Jens interpretation

It's not straightforward to meet all these criteria with a fully automatated procedure, so we'll do it in steps. We need to implement a random draw with a defined total size, which is weighted on tree diameter, allows up to 2 trees per square, only takes squares if they are closer to other squares than a set distance.

Apropos "double probability to draw a tree > 200 cm in circumference". This can be interpreted in several ways. There are slightly more trees above the treshold, so a random sample will produce a higher probability of large trees anyway. We could interpret it as drawing double the amount of large trees (>200cm) than smalller.

After some thinking, I will try this algorithm. It's based on an initial random draw of trees, with later filtering:

1. Order the trees randomly, i.e., draw a random order of all trees, with probablities based on tree diameter.
2. Note the distance for each square to the closest square.
3. Note the (randomly drawn) order of trees within each square.
4. Note the randomly drawn order of squares.
5. Record the total number of trees within each square.
6. Discard trees with order > 2.
7. Discard squares with total number of trees < 2 AND distance_to_nearest_neighbor < distance_limit (Note that this would optimally depend on that the other squares are selected the same year. This might get complex.)

Set up the data

Load the source excel-file

```
loc_raw <- openxlsx::read.xlsx("../rawData/Oak_2017data_2019resurveydata.xlsx") %>%
  as_tibble()

loc_raw
```

```
## # A tibble: 657 x 47
##    row_number RuteID RuteJA TreID Antall Verdi Omkrets Synlig_hul
##         <dbl>  <dbl> <chr>  <chr>  <dbl> <chr> <chr>   <chr>
##  1          1      7 Nei    7_01       1 A     332     JA
##  2          2     23 Nei    23_1       1 C     225     NEI
##  3          3     23 Nei    23_2       1 C     205     NEI
##  4          4     24 Nei    24_10      1 B     300     JA
##  5          5     24 Nei    24_2       1 B     237     JA
##  6          6     24 Nei    24_3       1 C     230     NEI
##  7          7     24 Nei    24_4       1 C     222     NEI
##  8          8     24 Nei    24_5       1 C     216     NEI
##  9          9     24 Nei    24_6       1 B     245     JA
## 10         10     24 Nei    24_7       1 B     320     NEI
## # i 647 more rows
## # i 39 more variables: Hulhet_areal_åpning <chr>, Hulhet_Plassering <chr>,
## #   Vedmuld <chr>, Treform <chr>, Barktype <chr>, Mosedekning <chr>,
## #   Vitalitet <dbl>, Kulturspor <chr>, Omgivelser <chr>, Renskog <chr>,
## #   Mestskog <chr>, Noeskog <chr>, PlasseringAR5 <chr>, Forskrift_gammel <dbl>,
## #   Forskrift <dbl>, Vern <chr>, Gjenvoksing <dbl>, Gjenvoksing2 <dbl>,
## #   Skjøtselsbehov <chr>, UTM32_X_koordinat <dbl>, UTM32_Y_koordinat <dbl>, ...
```

Filter out trees that is gone or not found in 2019.

```
loc <- loc_raw %>%
  filter(
    Gone != 1,
    Not_found != 1
  )

no_cand_trees <- nrow(loc)
```

We then have 600 trees to choose from. A bit more than half has a diameter of more than 200 cm.

```
loc %>%
  group_by(Omkrets > 200) %>%
  summarise(no = n())
```

```
## # A tibble: 3 x 2
##   `Omkrets > 200`    no
##   <lgl>           <int>
## 1 FALSE             269
## 2 TRUE              329
## 3 NA                  2
```

A quick look at the distribution of trees

```
no_rute <- loc %>%
  summarise(no_rute = n_distinct(RuteID)) %>%
  pull()

squares <- loc %>%
  select(RuteID) %>%
  distinct() %>%
  pull()
```

```
loc_with_at_least_two <- loc %>%
  group_by(RuteID) %>%
  summarise(no_trees = n()) %>%
  filter(no_trees > 1) %>%
  summarise(n_distinct(RuteID)) %>%
  pull()
```

We have 107 distinct survey squares (SSB) to choose from. But only 74 survey squares with at least 2 trees (if we want to restrict it to that). The distribution of number of trees within each survey square can be seen in figure 1.

```
loc %>%
  group_by(RuteID) %>%
  summarise(no_trees = n()) %>%
  ggplot() +
  geom_bar(aes(x = no_trees)) +
  geom_vline(aes(xintercept = 2),
    col = "red"
  ) +
  xlab("Antall trær innen én rute") +
  ylab("Antall ruter")
```

Add info on distance to other squares

Add a column with distances to the other squares (if we want to use squares with only 1 tree, if they are close enough to other squares)

Make an SF object (create a geometry).

```
loc_sf <- loc %>%
  st_as_sf(
    coords = c(
      "UTM32_X_koordinat",
      "UTM32_Y_koordinat"
    ),
```

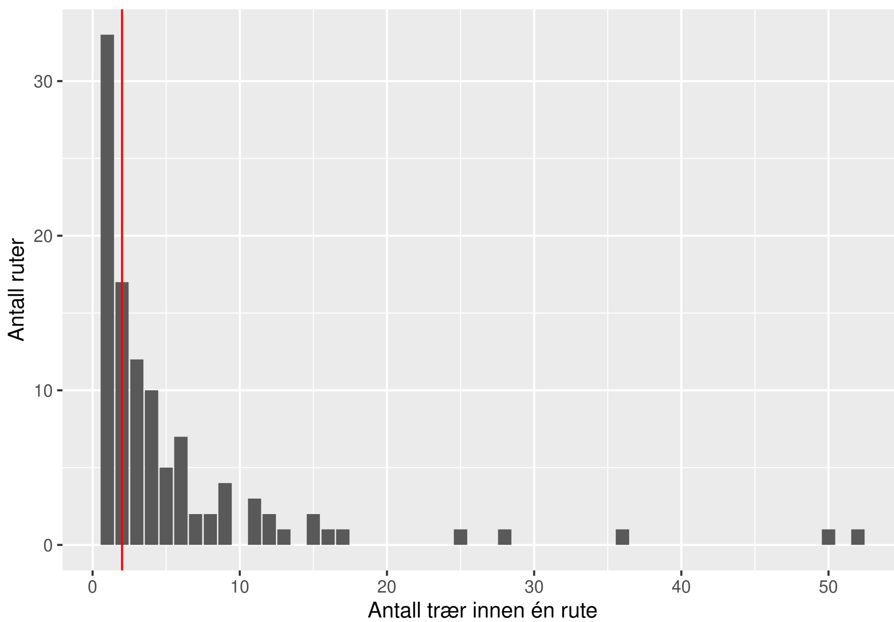


Figure 1: Distribution of number of trees within each survey square. Red vertical line deliniates plots with at least 2 trees.

```
    crs = 25832
  )
```

Get the ssb square geometries (from the gisdata database). Cache this to speed up render times.

```
ssb_500m <- read_sf(
  con,
  Id(
    schema = "ssb_data_utm33n",
    table = "ssb_500m"
  )
) %>%
  st_transform(crs = 25832)

cand_ssb_500m <- ssb_500m %>%
  st_join(loc_sf,
    left = FALSE
  ) %>%
  mutate(ssbid = as.character(ssbid)) %>%
  select(ssbid) %>%
  distinct()

system("mkdir -p out")

save(cand_ssb_500m,
  file = "out/cand_ssb_500m.Rdata"
)
```

```
load(file = "out/cand_ssb_500m.Rdata")
```

Get the distance to the nearest neighbor ssb square.

```
cand_ssb_500m <- cand_ssb_500m %>%
  mutate(nearest_dist = st_distance(., cand_ssb_500m[st_nearest_feature(cand_ssb_500m), ], b

cand_ssb_500m %>%
  st_drop_geometry() %>%
  select(ssbid, nearest_dist) %>%
  arrange(nearest_dist)
```

```
## # A tibble: 105 x 2
##    ssbid          nearest_dist
##    <chr>                   [m]
##  1 20830006464000            0
##  2 20835006463500            0
##  3 20605006483000         499.
```

```
##   4 20610006482000         499.
##   5 21035006539000         499.
##   6 21040006540000         499.
##   7 21720006586000         499.
##   8 21730006586000         499.
##   9 21965006564500        1117.
##  10 21980006563500        1117.
## # i 95 more rows
```

Get some background geometries.

```
regions <- read_sf(
  con,
  Id(
    schema = "insect_survey",
    table = "new_landsdel"
  )
)

south <- regions %>%
  filter(!(fylke %in% c("Trøndelag", "Nordland", "Troms og Finnmark")))
```

Join the locations with the ssbids and distances.

```
loc_sf <- loc_sf %>%
  st_join(cand_ssb_500m,
    left = TRUE
  )
```

```
loc_sf %>%
  select(ssbid, nearest_dist)
```

Add large/small tree category.

```
# loc_sf %>%
#   st_drop_geometry() %>%
#   select(Synlig_hul) %>%
#   distinct()

loc_sf <- loc_sf %>%
  mutate(
    large = Omkrets > 200,
    visible_hole = Synlig_hul == "JA"
  )
```

```
# tmap_mode("view")
tm_shape(south) +
  tm_borders() +
  tm_shape(cand_ssb_500m) +
```

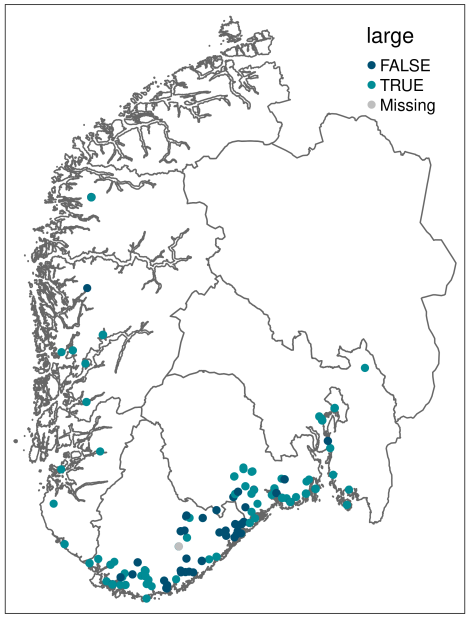


Figure 2: Locations of candidate trees. Size categories are diameter > 200 cm.

```
  tm_borders() +
  tm_shape(loc_sf) +
  tm_dots(
    col = "large",
    size = 0.1,
    palette = ninaPalette()
  )
```

```
## Warning: palette colors names missing for FALSE, TRUE. Therefore, palette color
## names will be ignored
```

Draw random selection of trees

For practical reasons, we're better of not visiting very distant single trees. We risk traveling a long time just to find the single tree is gone. Setting the upper limit to 30 km.

```
largest_dist_to_neighbor_m <- 30000 # as the bird flies (might be longer on roads)
```

Draw random order. This is the "master" random order of trees. We can then go down this order in sequence, with various filters.

```
set.seed(12345) # For reproducibility

tree_sel_random_order <- loc_sf %>%
  filter(!is.na(large)) %>% # Must know the diameter
  mutate(sel_prob = ifelse(large & visible_hole, 2 / 3, 1 / 3)) %>% # double the probabilit
  slice(sample(1:n(), n(), prob = sel_prob)) %>%
  ungroup() %>%
  mutate(rand_selection_order = row_number())
```

Note tree order within squares, total amount of trees within square, and (random) rute order. Order it after square random order and tree random order within squares. This was surprisingly tricky. Needed to make a character factor to be able to preserve the order of appearence with fct_inorder.

```
tree_sel_random_order <- tree_sel_random_order %>%
  group_by(RuteID) %>%
  mutate(
    tree_order_within_square = row_number(),
    no_trees_within_square = n()
  ) %>%
  ungroup() %>%
  mutate(rute_id_rand_order = forcats::fct_inorder(paste0("rute_", RuteID))) %>%
  group_by(rute_id_rand_order) %>%
  mutate(rute_id_order = cur_group_id()) %>%
  arrange(
    rute_id_order,
    tree_order_within_square
  ) %>%
  ungroup() %>%
  mutate(selection_order = row_number())
```

Add a note if single trees are farther away than distance limit. We will avoid these localities.

```
tree_sel_random_order <- tree_sel_random_order %>%
  mutate(
    single_and_lonely = no_trees_within_square < 2 &
      nearest_dist < units::set_units(largest_dist_to_neighbor_m, "m"),
    fylke_navn = "",
    kommune_navn = "",
    kommune_no_2022 = ""
  ) %>%
  select(
    selection_order,
    rand_selection_order,
    rute_id_order,
```

```
    tree_order_within_square,
    single_and_lonely,
    no_trees_within_square,
    RuteID,
    everything()
  )
```

This gives us 30 single trees farther away than 3×10^4 meters to other surveyed squares. This is a glimpse of the dataset.

```
tree_sel_random_order %>%
  filter(single_and_lonely) %>%
  select(
    RuteID,
    rute_id_rand_order,
    rute_id_order,
    tree_order_within_square,
    no_trees_within_square
  )
```

```
## Simple feature collection with 30 features and 5 fields
## Geometry type: POINT
## Dimension:     XY
## Bounding box:  xmin: 304977 ymin: 6442025 xmax: 574419 ymax: 6678309
## Projected CRS: ETRS89 / UTM zone 32N
## # A tibble: 30 x 6
##    RuteID rute_id_rand_order rute_id_order tree_order_within_square
##     <dbl> <fct>                      <int>                    <int>
##  1    338 rute_338                       4                        1
##  2     32 rute_32                        6                        1
##  3    359 rute_359                       9                        1
##  4    269 rute_269                      23                        1
##  5     40 rute_40                       42                        1
##  6    177 rute_177                      46                        1
##  7    159 rute_159                      66                        1
##  8    239 rute_239                      71                        1
##  9     56 rute_56                       73                        1
## 10    372 rute_372                      75                        1
## # i 20 more rows
## # i 2 more variables: no_trees_within_square <int>, geometry <POINT [m]>
```

Save this complete list for QGIS. Also do some PostGIS stuff.

I won't show this, see source file for details.

```
knitr::opts_chunk$set(
  echo = FALSE,
  results = "hide"
)
```

Test results of filtering out a set of trees

We will here test the approach by selecting the first 100 in the random order. Here we consider only the first 2 trees within each square, plus the single trees that are not isolated. We then take the first 100 rows (trees).

This can be replicated in QGIS by filtering on "tree_order_within_square<=2 OR (no_trees_within_square<2 AND single_and_lonely IS FALSE)"

```
tree_sel_test <- tree_sel_random_order %>%
  filter(tree_order_within_square <= 2 |
    (no_trees_within_square < 2 & !single_and_lonely)) %>%
  slice(1:100)
```

Tree diameter

This is the distribution of the tree diameter class (above 200 cm diameter).

```
n_tree_size <- tree_sel_test %>%
  st_drop_geometry() %>%
  group_by(large) %>%
  summarise(no_trees = n())

n_tree_size
```

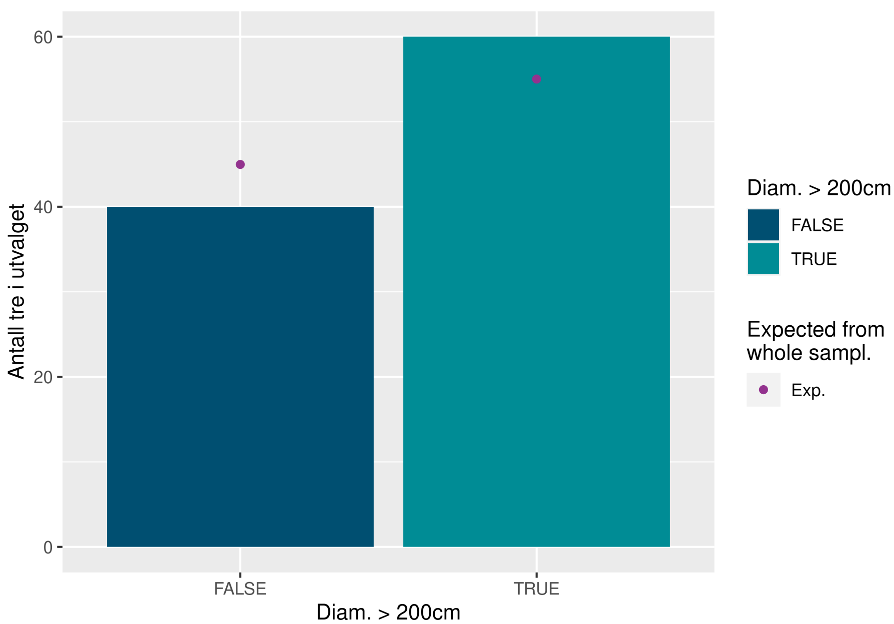
```
## # A tibble: 2 x 2
##   large no_trees
##   <lgl>    <int>
## 1 FALSE       40
## 2 TRUE        60
```

Which can be related to the distribution within the whole candidate set.

```
n_tree_size_exp <- tree_sel_random_order %>%
  st_drop_geometry() %>%
  group_by(large) %>%
  summarise(no_trees = n()) %>%
  ungroup() %>%
  mutate(no_trees_perc = round((no_trees / sum(no_trees) * 100), 2))


n_tree_size_exp
```

```
## # A tibble: 2 x 3
##   large no_trees no_trees_perc
##   <lgl>    <int>         <dbl>
## 1 FALSE      269          45.0
## 2 TRUE       329          55.0
```

```
ggplot(n_tree_size, aes(y = no_trees, x = large)) +
  geom_bar(aes(fill = large),
    stat = "identity"
  ) +
  geom_point(
    aes(
      y = no_trees_perc,
      x = large,
      col = "Exp."
    ),
    data = n_tree_size_exp
  ) +
  scale_fill_nina(name = "Diam. > 200cm") +
  scale_color_nina(
    name = "Expected from\nwhole sampl.",
    palette = "purple-green"
  ) +
  ylab("Antall tre i utvalget") +
  xlab("Diam. > 200cm")
```



Of these first 100 prioritized trees, we have 35 small trees, and 65 small. Pretty close to double the amount of larger trees. The red dots show the expected number of large and small, if we had choosen them randomly. The random selection seems to come close to the population means.

##Isolated trees

```
n_isolated <- tree_sel_test %>%
  st_drop_geometry() %>%
  group_by(no_trees_within_square > 1) %>%
  summarise(no_trees = n())

n_isolated
```

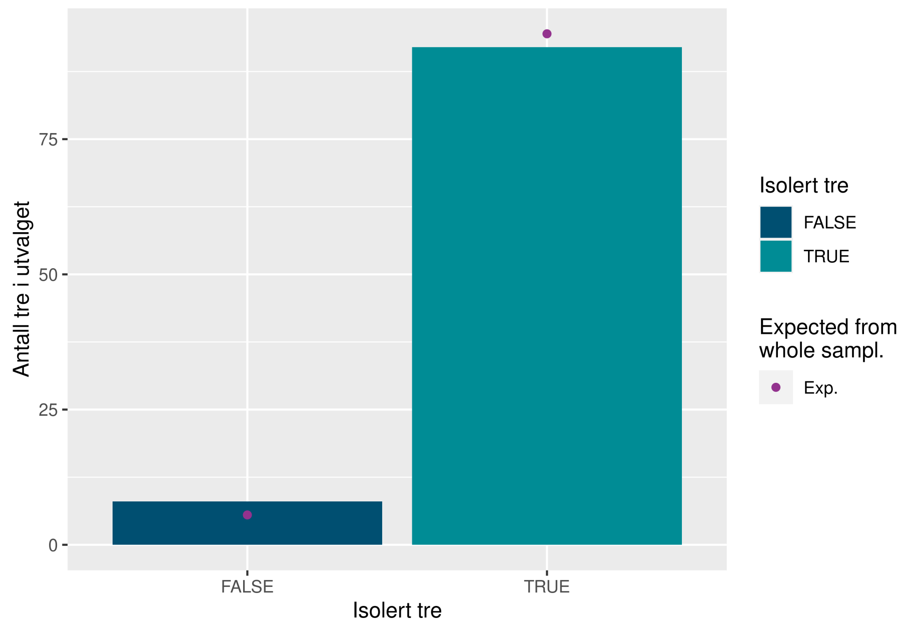
```
## # A tibble: 2 x 2
##   `no_trees_within_square > 1` no_trees
##   <lgl>                           <int>
## 1 FALSE                               8
## 2 TRUE                               92
```

```
n_isolated_exp <- tree_sel_random_order %>%
  st_drop_geometry() %>%
  group_by(no_trees_within_square > 1) %>%
  summarise(no_trees = n()) %>%
  ungroup() %>%
  mutate(no_trees_perc = round((no_trees / sum(no_trees) * 100), 2))


n_isolated_exp
```

```
## # A tibble: 2 x 3
##   `no_trees_within_square > 1` no_trees no_trees_perc
##   <lgl>                           <int>         <dbl>
## 1 FALSE                              33          5.52
## 2 TRUE                              565         94.5
```

```
ggplot(n_isolated, aes(y = no_trees, x = `no_trees_within_square > 1`)) +
  geom_bar(aes(fill = `no_trees_within_square > 1`),
    stat = "identity"
  ) +
  geom_point(
    aes(
      y = no_trees_perc,
      x = `no_trees_within_square > 1`,
      col = "Exp."
    ),
    data = n_isolated_exp
  ) +
```

```
  scale_fill_nina(name = "Isolert tre") +
  scale_color_nina(
    name = "Expected from\nwhole sampl.",
    palette = "purple-green"
  ) +
  ylab("Antall tre i utvalget") +
  xlab("Isolert tre")
```



Value categories

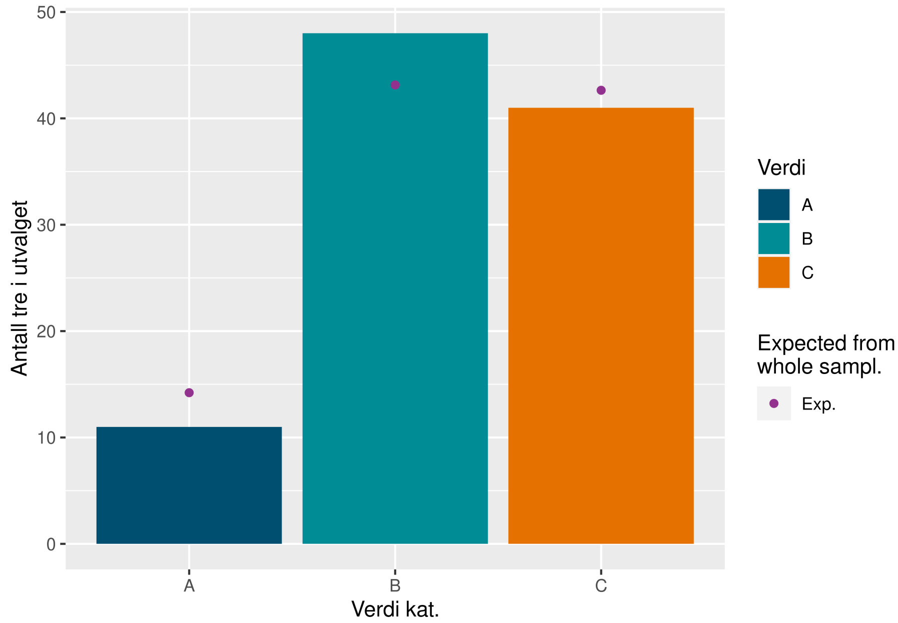
```
n_verdi <- tree_sel_test %>%
  st_drop_geometry() %>%
  group_by(Verdi) %>%
  summarise(no_trees = n())

n_verdi
```

```
## # A tibble: 3 x 2
##   Verdi no_trees
##   <chr>    <int>
## 1 A           11
## 2 B           48
## 3 C           41
```

```r
n_verdi_exp <- tree_sel_random_order %>%
  st_drop_geometry() %>%
  group_by(Verdi) %>%
  summarise(no_trees = n()) %>%
  ungroup() %>%
  mutate(no_trees_perc = round((no_trees / sum(no_trees) * 100), 2))


n_verdi_exp
```

```
## # A tibble: 3 x 3
##   Verdi no_trees no_trees_perc
##   <chr>    <int>         <dbl>
## 1 A           85          14.2
## 2 B          258          43.1
## 3 C          255          42.6
```

```r
ggplot(n_verdi, aes(y = no_trees, x = Verdi)) +
  geom_bar(aes(fill = Verdi),
    stat = "identity"
  ) +
  geom_point(
    aes(
      y = no_trees_perc,
      x = Verdi,
      col = "Exp."
    ),
    data = n_verdi_exp
  ) +
  scale_fill_nina(name = "Verdi") +
  scale_color_nina(
    name = "Expected from\nwhole sampl.",
    palette = "purple-green"
  ) +
  ylab("Antall tre i utvalget") +
  xlab("Verdi kat.")
```



Location, land-types

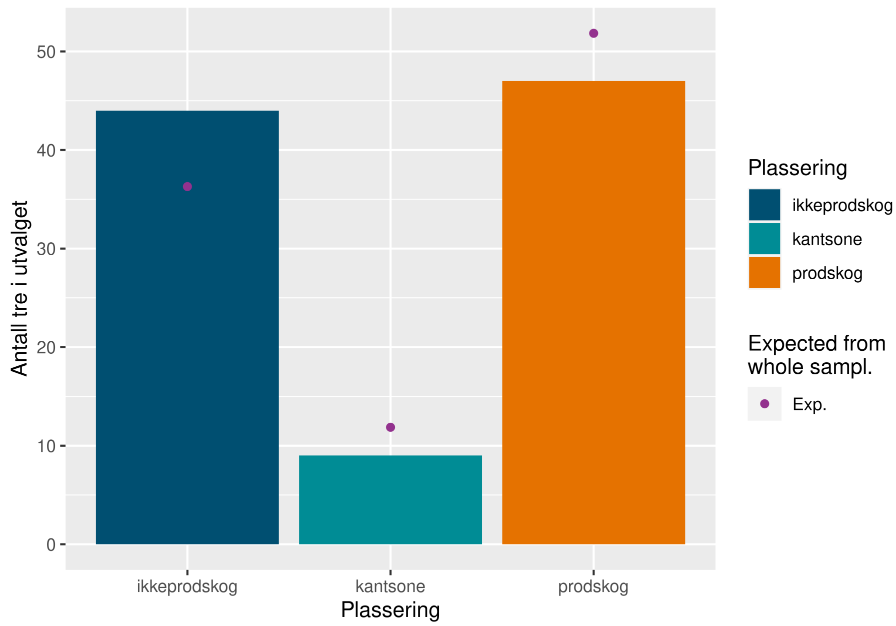
```
n_plass <- tree_sel_test %>%
  st_drop_geometry() %>%
  group_by(PlasseringAR5) %>%
  summarise(no_trees = n())

n_plass
```

```
## # A tibble: 3 x 2
##   PlasseringAR5 no_trees
##   <chr>            <int>
## 1 ikkeprodskog        44
## 2 kantsone             9
## 3 prodskog            47
```

```
n_plass_exp <- tree_sel_random_order %>%
  st_drop_geometry() %>%
  group_by(PlasseringAR5) %>%
  summarise(no_trees = n()) %>%
  ungroup() %>%
  mutate(no_trees_perc = round((no_trees / sum(no_trees) * 100), 2))


n_plass_exp
```

```
## # A tibble: 3 x 3
##   PlasseringAR5 no_trees no_trees_perc
##   <chr>            <int>         <dbl>
## 1 ikkeprodskog       217          36.3
## 2 kantsone            71          11.9
## 3 prodskog           310          51.8
```

```
ggplot(n_plass, aes(y = no_trees, x = PlasseringAR5)) +
  geom_bar(aes(fill = PlasseringAR5),
    stat = "identity"
  ) +
  geom_point(
    aes(
      y = no_trees_perc,
      x = PlasseringAR5,
      col = "Exp."
    ),
    data = n_plass_exp
  ) +
  scale_fill_nina(name = "Plassering") +
  scale_color_nina(
    name = "Expected from\nwhole sampl.",
    palette = "purple-green"
  ) +
  ylab("Antall tre i utvalget") +
  xlab("Plassering")
```



Visible holes

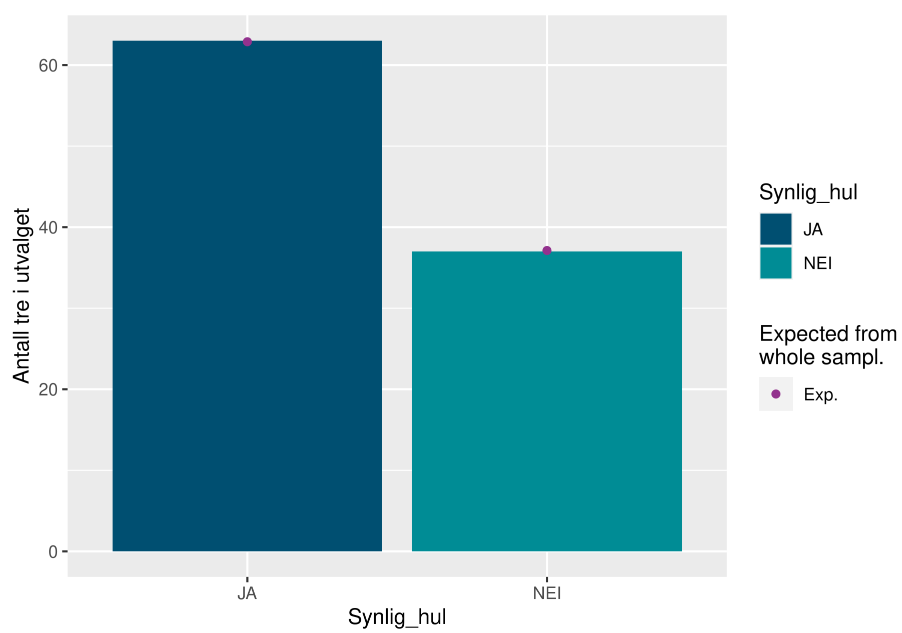
```
n_hole <- tree_sel_test %>%
  st_drop_geometry() %>%
  group_by(Synlig_hul) %>%
  summarise(no_trees = n())

n_hole
```

```
## # A tibble: 2 x 2
##   Synlig_hul no_trees
##   <chr>         <int>
## 1 JA               63
## 2 NEI              37
```

```
n_hole_exp <- tree_sel_random_order %>%
  st_drop_geometry() %>%
  group_by(Synlig_hul) %>%
  summarise(no_trees = n()) %>%
  ungroup() %>%
  mutate(no_trees_perc = round((no_trees / sum(no_trees) * 100), 2))


n_hole_exp
```

```
## # A tibble: 2 x 3
##   Synlig_hul no_trees no_trees_perc
##   <chr>         <int>         <dbl>
## 1 JA              376          62.9
## 2 NEI             222          37.1
```

```
ggplot(n_hole, aes(y = no_trees, x = Synlig_hul)) +
  geom_bar(aes(fill = Synlig_hul),
    stat = "identity"
  ) +
  geom_point(
    aes(
      y = no_trees_perc,
      x = Synlig_hul,
      col = "Exp."
    ),
    data = n_hole_exp
  ) +
  scale_fill_nina(name = "Synlig_hul") +
  scale_color_nina(
    name = "Expected from\nwhole sampl.",
    palette = "purple-green"
  ) +
  ylab("Antall tre i utvalget") +
  xlab("Synlig_hul")
```



Counties / Fylke

I have added the county information in the database. Will fetch the data back.

```
fylke_test_raw <- tbl(
  con,
  Id(
    schema = "hule_eiker_insekt",
    table = "oak_sel_random_order"
  )
) %>%
  collect()

fylke_test <- fylke_test_raw %>%
  filter(tree_order_within_square <= 2 |
    (no_trees_within_square < 2 & !single_and_lonely))

fylke_test_100 <- fylke_test %>%
  slice(1:100)
```

```
n_fylke <- fylke_test_100 %>%
  st_drop_geometry() %>%
  group_by(fylke_navn) %>%
  summarise(no_trees = n())

n_fylke
```

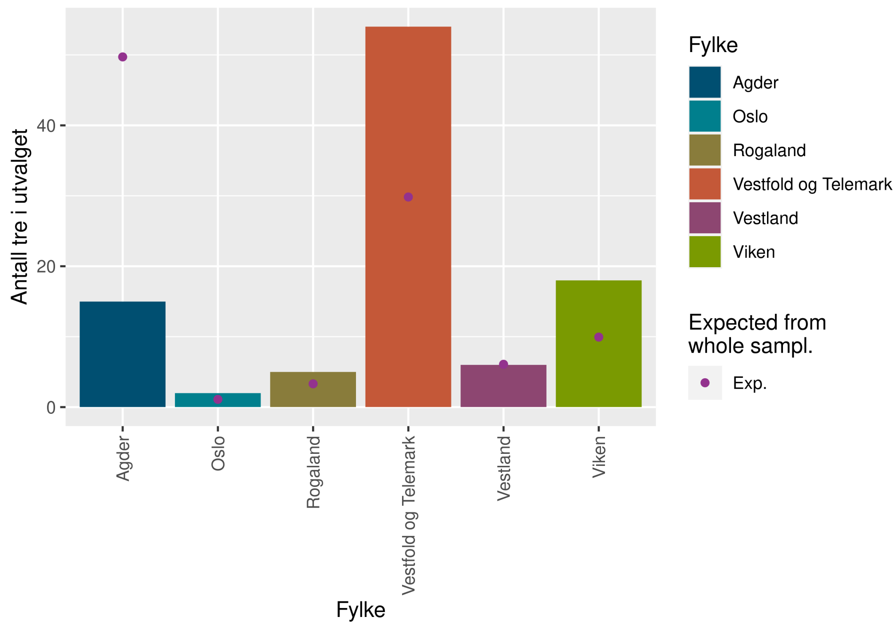
```
## # A tibble: 6 x 2
##   fylke_navn           no_trees
##   <chr>                   <int>
## 1 Agder                      15
## 2 Oslo                        2
## 3 Rogaland                    5
## 4 Vestfold og Telemark       54
## 5 Vestland                    6
## 6 Viken                      18
```

```
n_fylke_exp <- fylke_test %>%
  st_drop_geometry() %>%
  group_by(fylke_navn) %>%
  summarise(no_trees = n()) %>%
  ungroup() %>%
  mutate(no_trees_perc = round((no_trees / sum(no_trees) * 100), 2))


n_fylke_exp
```

```
## # A tibble: 6 x 3
```

```
##   fylke_navn           no_trees no_trees_perc
##   <chr>                   <int>         <dbl>
## 1 Agder                      90         49.7
## 2 Oslo                        2          1.1
## 3 Rogaland                    6          3.31
## 4 Vestfold og Telemark       54         29.8
## 5 Vestland                   11          6.08
## 6 Viken                      18          9.94
```

```
ggplot(n_fylke, aes(y = no_trees, x = fylke_navn)) +
  geom_bar(aes(fill = fylke_navn),
    stat = "identity"
  ) +
  geom_point(
    aes(
      y = no_trees_perc,
      x = fylke_navn,
      col = "Exp."
    ),
    data = n_fylke_exp
  ) +
  scale_fill_nina(name = "Fylke") +
  scale_color_nina(
    name = "Expected from\nwhole sampl.",
    palette = "purple-green"
  ) +
  ylab("Antall tre i utvalget") +
  xlab("Fylke") +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust = 1))
```



Fetch the ssb info for the selection.

We have the SSB 500x500 grid geometries in the database. We join this to the selection set, and create a view that we can use for plotting maps.

```
shortlist_ssb_500m <- cand_ssb_500m %>%
  filter(ssbid %in% tree_sel_test$ssbid)
```

```
dbSendStatement(
  my_con,
  "
      DROP TABLE IF EXISTS hule_eiker_insekt.selection_ssb ;
      "
)


dbSendStatement(
  my_con,
  "
      CREATE TABLE hule_eiker_insekt.selection_ssb as

      SELECT distinct on(ssbid) s.ssbid, s.geom
      FROM hule_eiker_insekt.oak_sel_random_order o,
      ssb_data_utm33n.ssb_500m s
      WHERE o.ssbid::bigint = s.ssbid;
```

```
"
)


dbSendStatement(
  my_con,
  "
                ALTER TABLE hule_eiker_insekt.selection_ssb ADD PRIMARY KEY(ssbid);
                "
)

dbSendStatement(
  my_con,
  "
                CREATE INDEX ON  hule_eiker_insekt.selection_ssb USING Gist(geom);
                "
)
```

```
dbSendStatement(
  my_con,
  "
                -- View: hule_eiker_insekt.squares_for_map

-- DROP VIEW hule_eiker_insekt.squares_for_map;

CREATE OR REPLACE VIEW hule_eiker_insekt.squares_for_map
 AS
 SELECT DISTINCT ON (s.ssbid, foo.matrikkel_no) row_number() OVER (ORDER BY s.ssbid) AS id,
    foo.\"Områdenavn\" AS omrade,
    foo.\"RuteID\" AS rute_id,
    s.ssbid,
    foo.matrikkel_no,
    foo.grunneier,
    foo.grunneier_telefon,
    foo.kommentar,
    foo.kommune_navn,
    foo.fylke_navn,
    foo.no_trees_within_square,
    round(st_x(st_transform(st_centroid(s.geom), 4326))::numeric, 6) AS lon_senterpkt,
    round(st_y(st_transform(st_centroid(s.geom), 4326))::numeric, 6) AS lat_senterpkt,
    s.geom
   FROM ( SELECT sel.selection_order,
            sel.rand_selection_order,
            sel.rute_id_order,
            sel.tree_order_within_square,
```

```
            sel.single_and_lonely,
            sel.no_trees_within_square,
            sel.\"RuteID\",
            sel.row_number,
            sel.\"RuteJA\",
            sel.\"TreID\",
            sel.\"Antall\",
            sel.\"Verdi\",
            sel.\"Omkrets\",
            sel.\"Synlig_hul\",
            sel.\"Hulhet_areal_åpning\",
            sel.\"Hulhet_Plassering\",
            sel.\"Vedmuld\",
            sel.\"Treform\",
            sel.\"Barktype\",
            sel.\"Mosedekning\",
            sel.\"Vitalitet\",
            sel.\"Kulturspor\",
            sel.\"Omgivelser\",
            sel.\"Renskog\",
            sel.\"Mestskog\",
            sel.\"Noeskog\",
            sel.\"PlasseringAR5\",
            sel.\"Forskrift_gammel\",
            sel.\"Forskrift\",
            sel.\"Vern\",
            sel.\"Gjenvoksing\",
            sel.\"Gjenvoksing2\",
            sel.\"Skjøtselsbehov\",
            sel.\"Kommune\",
            sel.\"Områdenavn\",
            sel.\"Nøyaktighetsklasse\",
            sel.\"Utvalgt.Natur.type\",
            sel.\"Eikeart\",
            sel.\"Renskog3\",
            sel.\"Mestskog4\",
            sel.\"Noeskog5\",
            sel.\"Renskog6\",
            sel.\"Mestskog7\",
            sel.\"Noeskog8\",
            sel.\"Gone\",
            sel.\"Not_found\",
            sel.\"Ny_vitalitet\",
            sel.\"Ny_gjenvoksing\",
            sel.\"Ny_gjenvoksing2\",
```

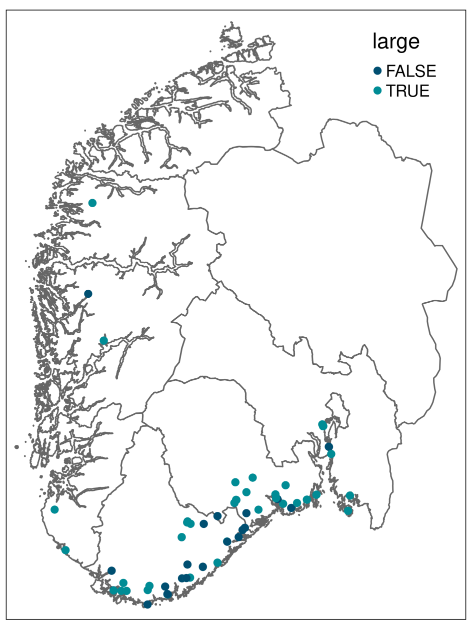
```
            sel.\"Change_vitalitet\",
            sel.\"Change_gjenvoksing\",
            sel.ssbid,
            sel.nearest_dist,
            sel.large,
            sel.sel_prob,
            sel.rute_id_rand_order,
            sel.fylke_navn,
            sel.kommune_navn,
            sel.kommune_no_2022,
            sel.geom,
            sel.geom_25833,
            sel.east_west,
            sel.lon_lat_25833,
            sel.matrikkel_no,
            sel.grunneier,
            sel.grunneier_telefon,
            sel.grunneier_epost,
            sel.grunneier_adresse,
            sel.kommentar
           FROM hule_eiker_insekt.oak_sel_random_order sel
          WHERE sel.tree_order_within_square <= 2 OR sel.no_trees_within_square < 2 AND sel.
     LEFT JOIN hule_eiker_insekt.selection_ssb s ON foo.ssbid::bigint = s.ssbid;
     "
)

dbSendStatement(
  my_con,
  "
ALTER TABLE hule_eiker_insekt.squares_for_map
    OWNER TO \"jens.astrom\";
    "
)


dbSendStatement(
  my_con,
  "
GRANT SELECT ON TABLE hule_eiker_insekt.squares_for_map TO gisuser;
"
)

dbSendStatement(
  my_con,
  "
```

```
GRANT ALL ON TABLE hule_eiker_insekt.squares_for_map TO \"jens.astrom\";
                "
)
```

Take a look at the selection

```
# tmap_mode("view")
tm_shape(south) +
  tm_borders() +
  tm_shape(shortlist_ssb_500m) +
  tm_borders() +
  tm_shape(tree_sel_test) +
  tm_dots(
    col = "large",
    size = 0.1,
    palette = ninaPalette()
  )
```

```
## Warning: palette colors names missing for FALSE, TRUE. Therefore, palette color
## names will be ignored
```



Instruction for QGIS

This instructions for some manual work in QGIS, going through the proposed selection, calling owners for permission, selecting the final set of trees.

Project "hule_eiker" at P:\153018_overvaking_av_insekter_i_hule_eiker\GIS.

Use the layer oak_sel_random_order.

Some new columns: 'selection_order': Use this order to select trees. This is a random order we can follow. 'rand_selection_order': the original random order (not sorted on RuteID, for documentation) 'rute_id_order': the order the squares appeared in the random sample 'tree_order_within_square': the (random) order of trees within each square. Use tree 1 & 2, but if needed higher numbers if we don't find tree no 1 and 2. 'single_and_lonely': Is the tree alone in its square and is the square > 30 000 km from the nearest square?

I have filtered the entire selection to only look at 2 trees or 1 tree if they are not isolated.

Filter = tree_order_within_square<=2 OR (no_trees_within_square<2 AND single_and_lonely IS FALSE)

If these trees are not enough, we can remove or change the 'tree_order_within_square<=2' to show more trees within each square.

Proposed work within qgis:

1. Sort the table on selection order. Start with tree 1 (selection_order = 1), show the info with the "i" button in QGIS.
2. Use the matrikkel_no with https://matrikkeldata.nina.no/ to get the owner.
3. Find the contact info for the owner.

Look at the final selection

The final selection was made in QGIS, and phoning land owners. The final set of used localities are stored in an excel-file. We here fetch this selection and reproduce some plots for the report.

Get the list of selected trees

```
hule_eiker_2023_raw <- openxlsx::read.xlsx("../rawData/Feltark utfylt - insekter i hule eike
  sheet = 2
)

hule_eiker_2023 <- hule_eiker_2023_raw %>%
  select(
    RuteID,
    TreID,
    Område
  ) %>%
  as_tibble()
```

```
fylke_navn <- fylke_test %>%
  select(
    RuteID,
    fylke_navn
  ) %>%
  distinct()
```

Fix some upper cases, and join fylke info.

```
tree_sel_random_order <- tree_sel_random_order %>%
  mutate(PlasseringAR5 = stringr::str_to_sentence(PlasseringAR5)) %>%
  left_join(fylke_navn,
    by = c("RuteID" = "RuteID")
  )
```

```
hule_eiker_2023_append <- hule_eiker_2023 %>%
  left_join(tree_sel_random_order,
    by = c(
      "RuteID" = "RuteID",
      "TreID" = "TreID"
    )
  ) %>%
  left_join(fylke_navn,
    by = c("RuteID" = "RuteID")
  )
```

Location, land-types

```
n_plass <- hule_eiker_2023_append %>%
  st_drop_geometry() %>%
  group_by(PlasseringAR5) %>%
  summarise(no_trees = n())

n_plass
```

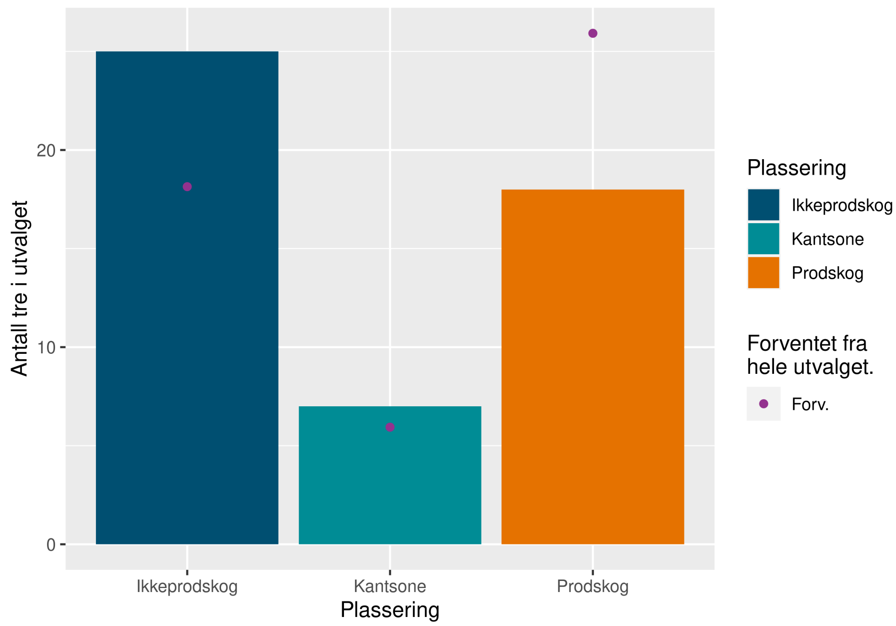
```
## # A tibble: 3 x 2
##   PlasseringAR5 no_trees
##   <chr>            <int>
## 1 Ikkeprodskog        25
## 2 Kantsone             7
## 3 Prodskog            18
```

```
n_plass_exp <- tree_sel_random_order %>%
  st_drop_geometry() %>%
  group_by(PlasseringAR5) %>%
  summarise(no_trees = n()) %>%
  ungroup() %>%
```

```
  mutate(no_trees_perc = round((no_trees / sum(no_trees) * 50), 2))


n_plass_exp
```

```
## # A tibble: 3 x 3
##   PlasseringAR5 no_trees no_trees_perc
##   <chr>            <int>         <dbl>
## 1 Ikkeprodskog       217         18.1
## 2 Kantsone            71          5.94
## 3 Prodskog           310         25.9
```

```
ggplot(n_plass, aes(y = no_trees, x = PlasseringAR5)) +
  geom_bar(aes(fill = PlasseringAR5),
    stat = "identity"
  ) +
  geom_point(
    aes(
      y = no_trees_perc,
      x = PlasseringAR5,
      col = "Forv."
    ),
    data = n_plass_exp
  ) +
  scale_fill_nina(name = "Plassering") +
  scale_color_nina(
    name = "Forventet fra\nhele utvalget.",
    palette = "purple-green"
  ) +
  ylab("Antall tre i utvalget") +
  xlab("Plassering")
```



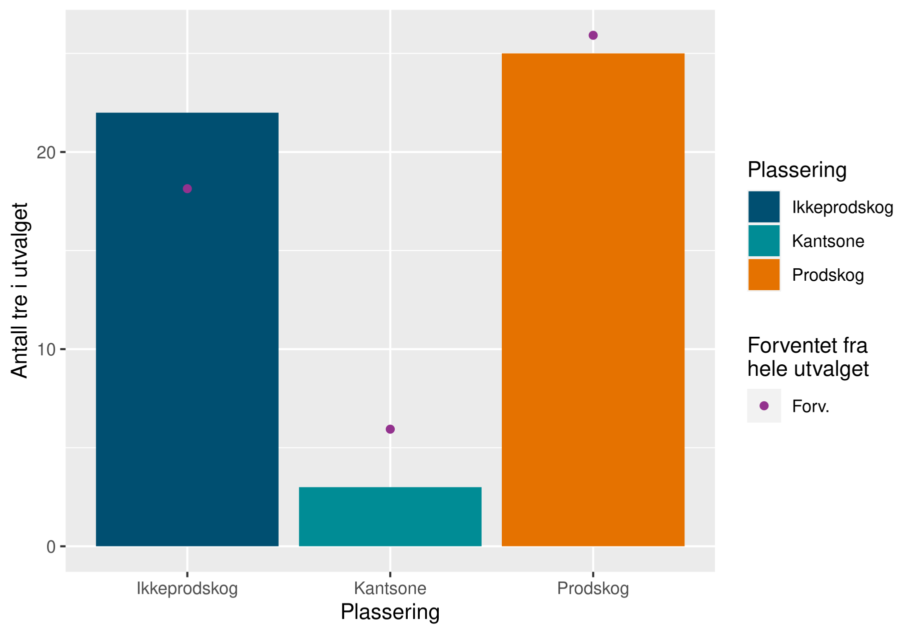
Do the same fig for the top 50 trees in the Western region (that we are expecting to choose)

```
n_plass_west <- fylke_test %>%
  st_drop_geometry() %>%
  mutate(PlasseringAR5 = stringr::str_to_sentence(PlasseringAR5)) %>%
  filter(fylke_navn %in% c("Vestland", "Agder", "Rogaland")) %>%
  arrange(rand_selection_order) %>%
  slice(1:50) %>%
  group_by(PlasseringAR5) %>%
  summarise(no_trees = n())

n_plass_west
```

```
## # A tibble: 3 x 2
##   PlasseringAR5 no_trees
##   <chr>            <int>
## 1 Ikkeprodskog        22
## 2 Kantsone             3
## 3 Prodskog            25
```

```
ggplot(n_plass_west, aes(y = no_trees, x = PlasseringAR5)) +
  geom_bar(aes(fill = PlasseringAR5),
    stat = "identity"
  ) +
  geom_point(
```

```
    aes(
      y = no_trees_perc,
      x = PlasseringAR5,
      col = "Forv."
    ),
    data = n_plass_exp
  ) +
  scale_fill_nina(name = "Plassering") +
  scale_color_nina(
    name = "Forventet fra\nhele utvalget",
    palette = "purple-green"
  ) +
  ylab("Antall tre i utvalget") +
  xlab("Plassering")
```



Location, counties

```
n_fylke_ost <- hule_eiker_2023_append %>%
  st_drop_geometry() %>%
  group_by(fylke_navn.y) %>%
  summarise(no_trees = n())

n_fylke_ost
```

```
## # A tibble: 2 x 2
##   fylke_navn.y         no_trees
##   <chr>                   <int>
## 1 Vestfold og Telemark       38
## 2 Viken                      12
```

```
n_fylke_exp_ost <- fylke_test %>%
  st_drop_geometry() %>%
  filter(fylke_navn %in% c("Vestfold og Telemark", "Viken", "Oslo")) %>%
  group_by(fylke_navn) %>%
  summarise(no_trees = n()) %>%
  ungroup() %>%
  mutate(no_trees_perc = round((no_trees / sum(no_trees) * 50), 2))


n_fylke_exp_ost
```

```
## # A tibble: 3 x 3
##   fylke_navn           no_trees no_trees_perc
##   <chr>                   <int>         <dbl>
## 1 Oslo                        2          1.35
## 2 Vestfold og Telemark       54         36.5
## 3 Viken                      18         12.2
```

```
ggplot(n_fylke_ost, aes(y = no_trees, x = fylke_navn.y)) +
  geom_bar(aes(fill = fylke_navn.y),
    stat = "identity"
  ) +
  geom_point(
    aes(
      y = no_trees_perc,
      x = fylke_navn,
      col = "Forv."
    ),
    data = n_fylke_exp_ost
  ) +
  scale_fill_nina(name = "Fylke") +
  scale_color_nina(
    name = "Forventet fra\nhele utvalget",
    palette = "purple-green"
  ) +
  ylab("Antall tre i utvalget") +
  xlab("Fylke") +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust = 1))
```

Do the same fig for the top 50 trees in the Western region (that we are expecting to choose)

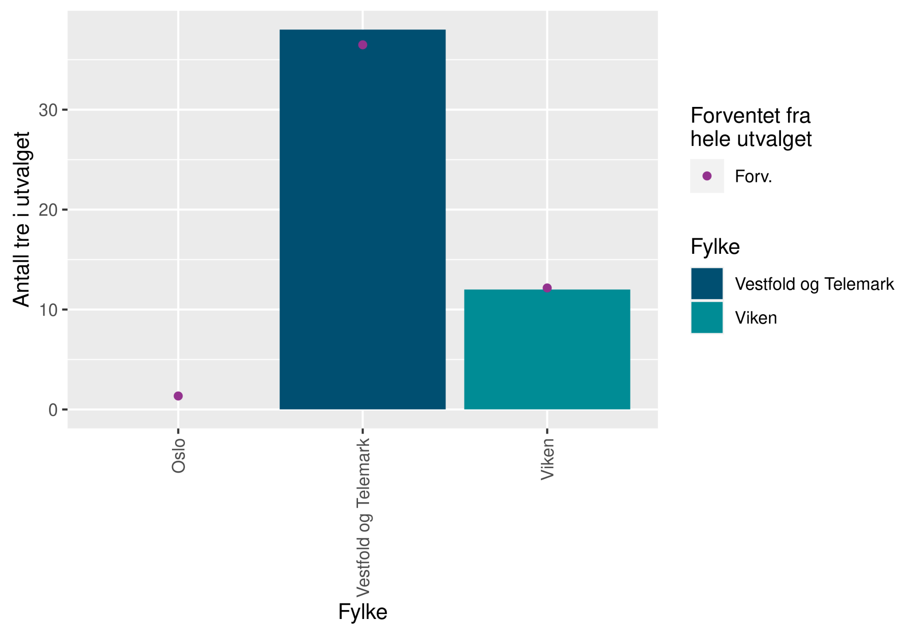


Figure 3: Fordeling av plassering i fylker, i det forventede utvalget i region Vest

```
n_fylke_west <- fylke_test %>%
  st_drop_geometry() %>%
  mutate(PlasseringAR5 = stringr::str_to_sentence(PlasseringAR5)) %>%
  filter(fylke_navn %in% c("Vestland", "Agder", "Rogaland")) %>%
  arrange(rand_selection_order) %>%
  slice(1:50) %>%
  group_by(fylke_navn) %>%
  summarise(no_trees = n())

n_fylke_west
```

```
## # A tibble: 3 x 2
##   fylke_navn no_trees
##   <chr>         <int>
## 1 Agder            39
## 2 Rogaland          4
## 3 Vestland          7
```

```
n_fylke_exp_west <- fylke_test %>%
  st_drop_geometry() %>%
  filter(fylke_navn %in% c("Vestland", "Agder", "Rogaland")) %>%
  group_by(fylke_navn) %>%
  summarise(no_trees = n()) %>%
  ungroup() %>%
  mutate(no_trees_perc = round((no_trees / sum(no_trees) * 50), 2))


n_fylke_exp_west
```

```
## # A tibble: 3 x 3
##   fylke_navn no_trees no_trees_perc
##   <chr>         <int>         <dbl>
## 1 Agder            90         42.1
## 2 Rogaland          6          2.8
## 3 Vestland         11          5.14
```

```
ggplot(n_fylke_west, aes(y = no_trees, x = fylke_navn)) +
  geom_bar(aes(fill = fylke_navn),
    stat = "identity"
  ) +
  geom_point(
    aes(
      y = no_trees_perc,
      x = fylke_navn,
      col = "Forv."
    ),
```

```
    data = n_fylke_exp_west
  ) +
  scale_fill_nina(name = "Fylke") +
  scale_color_nina(
    name = "Forventet fra\nhele utvalget",
    palette = "purple-green"
  ) +
  ylab("Antall tre i utvalget") +
  xlab("Fylke") +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust = 1))
```