SRFN figures

Marissa Dyck

2025-1-10

Contents

The first two chunks of this r markdown file after the r setup allow for plot zooming, but it also means that the html file must be opened in a browser to view the document properly. When it knits in RStudio the preview will appear empty but the html when opened in a browser will have all the info and you can click on each plot to Zoom in on it.

Before you begin

Notes

A few notes about this script.

If you are running this make sure you download the whole SRFN (GitHub repository)[https://github.com/marissadyck/SRFN_ACME_Camera_Project] from my GitHub. This will ensure you have all the files, data, and proper folder structure you will need to run this code and associated analyses.

Also make sure you open RStudio through the R project (ARFN_ACME_Camera_Project.Rproj) this will automatically set your working directory to the correct place (wherever you saved the repository and it's files) and ensure you don't have to change the file paths for some of the data.

Lastly, if you are looking to adapt this code for a future year of data, you will want to ensure you have run all the code through 3_ACME_SRFN_analysis.Rmd with your data as there is much data formatting, cleaning, and restructuring that has to be done before this code will work. *Helpful note: The files are numbered in the order they are used to prep for this analysis.*

If you have question please email the most recent author, currently

Marissa A. Dyck
Postdoctoral research fellow
University of Victoria
School of Environmental Studies
Email: marissadyck17@gmail.com

(*update/add authors as needed*)

R and RStudio

Before starting you should ensure you have the latest version of R and RStudio downloaded. This code was generated under R version 4.2.3 and with RStudio version 2024.04.2+764.

You can download R and RStudio HERE

R markdown

This script is written in R markdown and thus uses a mix of coding markup languages and R. If you are planning to run this script with new data or make any modifications you will want to be familiar with some

basics of R markdown.

Below is an R markdown cheatsheet to help you get started,
R markdown cheatsheet

Install packages

If you don't already have the following packages installed, use the code below to install them. *NOTE this will not run automatically as eval=FALSE is included in the chunk setup (i.e. I don't want it to run every time I run this code since I have the packages installed).

```
install.packages('tidyverse')
install.packages('ggpubr')
install.packages('corrplot')
install.packages('Hmisc')
install.packages('glmmTMB')
install.packages('MuMIn')
install.packages('TMB', type = 'source')
install.packages('rphylopic')
install.packages('broom')
install.packages('ggeffects')
```

Load libraries

Then load the packages to your library so they are usable for this session.

```
library(tidyverse) # data tidying, visualization, and much more; this will load all tidyverse packages,
library(ggpubr) # make modifications to plot for publication (arrange plots)
```

```
## Warning: package 'ggpubr' was built under R version 4.5.1
```

```
library(PerformanceAnalytics)    # Used to generate a correlation plot
library(Hmisc) # used to generate histograms for all variables in data frame
library(glmmTMB)      # Constructing GLMMs
library(MuMIn) # for model selection
library(rphylopic) # add animal silhouettes to graphs
library(broom) # extracting odds ratios in a tidy format
library(ggeffects) # for extracting predicted probabilities from glms for plotting
```

Detection and Naive Occupancy

Load independent detection data

Read in saved and cleaned **independent** detection data for the project e.g., what was generated in the 1_ACME_SRFN_camera_script_2024-12-10.Rmd.

```
# detection data
# read in saved and cleaned detection data generated from the 1_ACME_SRFN_camera_script.Rmd
detections <- read_csv('data/processed/srfn_ind_det.csv') %>%

      # ensure the array, site, species, and event_id read in as factors
      mutate_if(is.character,
                as.factor)
```

```
## Rows: 8428 Columns: 9
## -- Column specification --------------------------------------------------------
## Delimiter: ","
```

```
## chr  (4): array, site, species, event_id
## dbl  (4): site_number, month, year, timediff
## dttm (1): datetime
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
# look at data structure
str(detections)
```

```
## tibble [8,428 x 9] (S3: tbl_df/tbl/data.frame)
##  $ array      : Factor w/ 8 levels "LUAG","LUBF",..: 7 7 7 7 7 7 7 7 7 7 ...
##  $ site_number: num [1:8428] 100 100 100 100 100 100 100 100 100 100 ...
##  $ site       : Factor w/ 63 levels "LUAG_119","LUAG_124",..: 59 59 59 59 59 59 59 59 59 59 ...
##  $ species    : Factor w/ 30 levels "ATVer","Black bear",..: 30 30 30 30 14 14 14 14 14 14 ...
##  $ datetime   : POSIXct[1:8428], format: "2022-05-23 03:06:03" "2022-11-20 12:28:11" ...
##  $ month      : num [1:8428] 5 11 12 12 5 5 9 9 10 11 ...
##  $ year       : num [1:8428] 2022 2022 2022 2022 2022 ...
##  $ timediff   : num [1:8428] NA 261202 27346 4188 NA ...
##  $ event_id   : Factor w/ 8428 levels "E0","E1","E10",..: 1 2 1113 2224 3335 4446 5557 6668 7779 8318
```

Data checks

```
# check for NAs introduced during data merge
summary(detections)
```

```
##      array       site_number           site                   species
##  LUBF   :4125   Min.   :  1.00   LUBF_132: 801   White-tailed deer:5497
##  LUS    :1721   1st Qu.: 44.00   LUC_141 : 422   Moose             : 790
##  LUC    :1644   Median : 88.00   LUBF_83 : 413   Snowshoe hare     : 478
##  LUAG   : 558   Mean   : 86.42   LUBF_44 : 392   Coyote            : 322
##  LUUK   : 179   3rd Qu.:130.00   LUS_39  : 365   Black bear        : 298
##  LUW    : 163   Max.   :166.00   LUS_56  : 362   Unknown           : 211
##  (Other):  38                    (Other) :5673   (Other)           : 832
##     datetime                      month            year
##  Min.   :2022-04-02 06:57:12   Min.   : 1.000   Min.   :2022
##  1st Qu.:2022-08-04 12:00:00   1st Qu.: 5.000   1st Qu.:2022
##  Median :2022-12-17 17:47:40   Median : 7.000   Median :2022
##  Mean   :2023-01-05 08:00:08   Mean   : 6.879   Mean   :2022
##  3rd Qu.:2023-06-07 02:42:41   3rd Qu.: 9.000   3rd Qu.:2023
##  Max.   :2024-01-05 07:01:03   Max.   :12.000   Max.   :2024
##
##     timediff            event_id
##  Min.   :    30.02   E0     :   1
##  1st Qu.:   396.22   E1     :   1
##  Median :  1430.00   E10    :   1
##  Mean   : 13199.47   E100   :   1
##  3rd Qu.:  6210.98   E1000  :   1
##  Max.   :725178.60   E1001  :   1
##  NA's   :463         (Other):8422
```

THe only NAs are in the timediff column which is what we expect since any of the first observations won't have a a value for timediff. If you are confused by this re-visit the 1_ACME_camera_script.

Summary

Some summaries of the merged detection data

```
# create a list of focal species for filtering the data/plots
 srfn_focal_species <- c('White-tailed deer',
              'Black bear',
              'Cougar',
              'Coyote',
              'Elk',
              'Fisher',
              'Grey wolf',
              'Grizzly bear',
              'Lynx',
              'Marten',
              'Moose',
              'Mule deer',
              'Red fox',
              'Snowshoe hare')

detections <- detections %>%

  filter(species %in% srfn_focal_species)
```

```
detections %>%

  # group by array and species
  group_by(species) %>%
  summarise(n = n()) %>%

  # sort from greatest to least
  arrange(desc(n)) %>%

  # have R print everything
  print(n = nrow(.))
```

```
## # A tibble: 14 x 2
##    species              n
##    <fct>            <int>
##  1 White-tailed deer  5497
##  2 Moose               790
##  3 Snowshoe hare       478
##  4 Coyote              322
##  5 Black bear          298
##  6 Grey wolf           144
##  7 Elk                 134
##  8 Lynx                 94
##  9 Mule deer            57
## 10 Grizzly bear         19
## 11 Red fox              19
## 12 Fisher               17
## 13 Marten                9
## 14 Cougar                6
```

Subset data

We will also want to subset the data by landscape unit (LU) and generate a new data frame for each LU to use for plotting

I'm not great at writing loops, so let's see how this shit goes... probably bad but who knows

```
array_frames <- list()

for (i in unique(detections$array)){

   #Subset data based on radius
  df <- detections %>%
    filter(array == i)

 # list of dataframes
  array_frames <- c(array_frames, list(df))


}

# inspect one data frame
print(array_frames[[1]])
```

```
## # A tibble: 172 x 9
##    array site_number site     species   datetime            month  year timediff
##    <fct>       <dbl> <fct>    <fct>     <dttm>              <dbl> <dbl>    <dbl>
##  1 LUUK          100 LUUK_100 White-ta~ 2022-05-23 03:06:03     5  2022      NA
##  2 LUUK          100 LUUK_100 White-ta~ 2022-11-20 12:28:11    11  2022  261202.
##  3 LUUK          100 LUUK_100 White-ta~ 2022-12-09 12:14:00    12  2022   27346.
##  4 LUUK          100 LUUK_100 White-ta~ 2022-12-12 10:05:37    12  2022    4188.
##  5 LUUK          100 LUUK_100 Moose     2022-05-30 11:49:21     5  2022      NA
##  6 LUUK          100 LUUK_100 Moose     2022-05-31 15:30:36     5  2022    1661.
##  7 LUUK          100 LUUK_100 Moose     2022-09-02 16:18:21     9  2022  135400.
##  8 LUUK          100 LUUK_100 Moose     2022-09-18 08:26:00     9  2022   22544.
##  9 LUUK          100 LUUK_100 Moose     2022-10-03 13:48:35    10  2022   21922.
## 10 LUUK          100 LUUK_100 Moose     2022-11-10 10:34:25    11  2022   54526.
## # i 162 more rows
## # i 1 more variable: event_id <fct>
```

... I think this worked

Now let's change names of list items using purrr, couldn't figure out how to name them in the loop, you don't necessarily need to do this because we change the names in the next section, but I like having things named

```
array_frames <- array_frames %>%

  purrr::set_names('agriculture',
                   'broadleaf',
                   'coniferous',
                   'developed',
                   'grassland',
                   'shrubland',
                   'unknown',
                   'water')

# inspect each data frame
```

```r
head(array_frames$broadleaf)
```

```
## # A tibble: 6 x 9
##   array site_number site     species    datetime            month  year timediff
##   <fct>       <dbl> <fct>    <fct>      <dttm>              <dbl> <dbl>    <dbl>
## 1 LUBF          104 LUBF_104 White-tai~ 2022-05-10 16:35:41     5  2022     NA
## 2 LUBF          104 LUBF_104 White-tai~ 2022-05-10 17:12:09     5  2022     36.4
## 3 LUBF          104 LUBF_104 White-tai~ 2022-06-10 09:34:15     6  2022  44182.
## 4 LUBF          104 LUBF_104 White-tai~ 2022-06-11 17:21:59     6  2022   1908.
## 5 LUBF          104 LUBF_104 White-tai~ 2022-06-16 09:13:57     6  2022   6712.
## 6 LUBF          104 LUBF_104 White-tai~ 2022-06-16 19:02:42     6  2022    589.
## # i 1 more variable: event_id <fct>
```

Detection plots

Detection data

Now we can apply the same data formatting for each LUs' data frame using purrr.

We want to count the number of independent detections per species per LU to use in the detection plots

```r
# apply the same formatting to each LU data frame using purrr map
detection_data <- array_frames %>%

  purrr::map(
    ~.x %>%

      # group by species
      group_by(species) %>%

      # calculate a column with unique accounts of each species
      mutate(count = n_distinct(event_id)) %>%

      # keep just the columns we need
      select(species, count) %>%

      # keep only unique (distinct) rows so we should be left with one row per species, this helps with
      distinct()) %>%

  # set names of list objects
  purrr::set_names(~ paste('Detections', names(array_frames)))
```

Detection plots

Now to graph independent detections for each LU using purrr, this avoids a TON of code repetition needed to plot each one individually

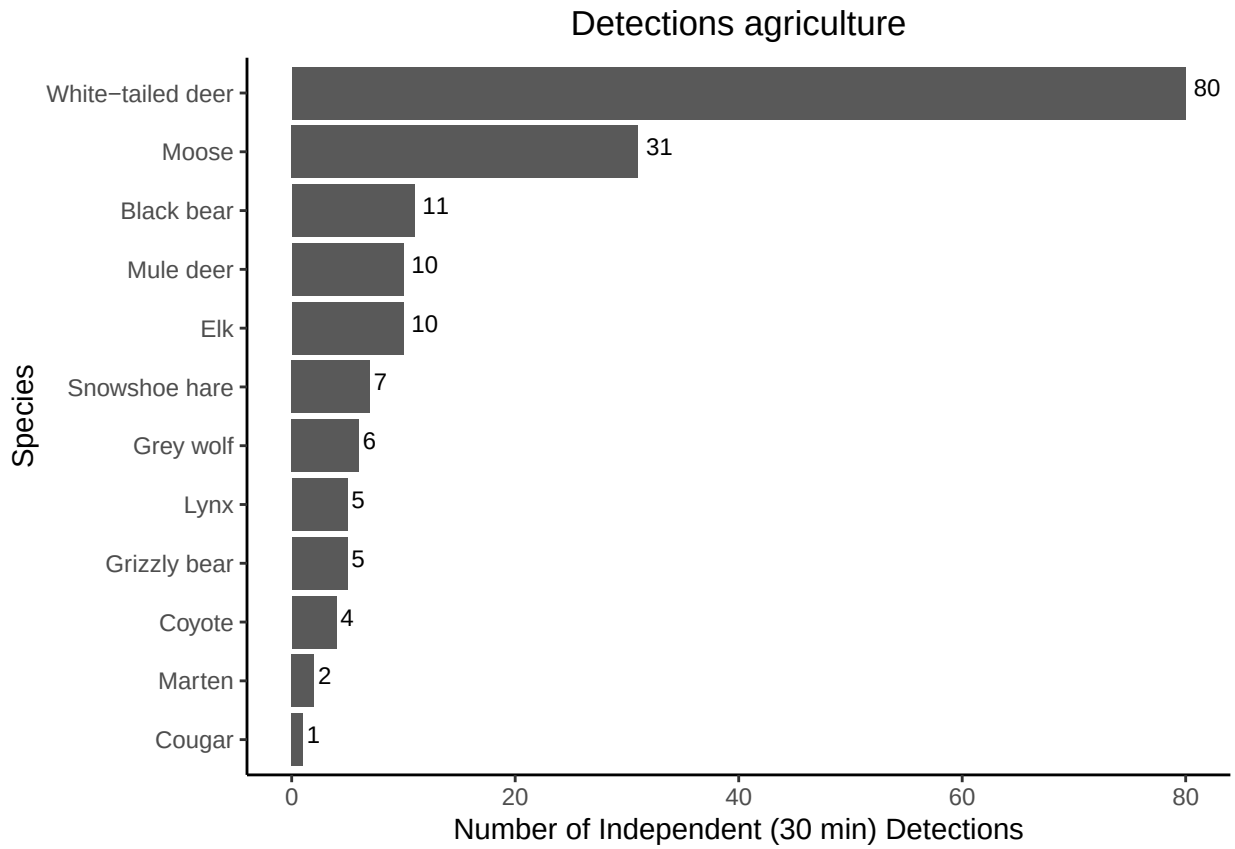
We use `purrr::imap()` instead of `purrr::map()` because imap maintains the variable names in our list (e.g. Detections LU01, Detections LU13, etc.) which we can then use to title each plot.

Within `purrr::imap()` we just paste the code we would use for a single ggplot since all the graphical elements (except the title which we change with the file name [.y]) are the same

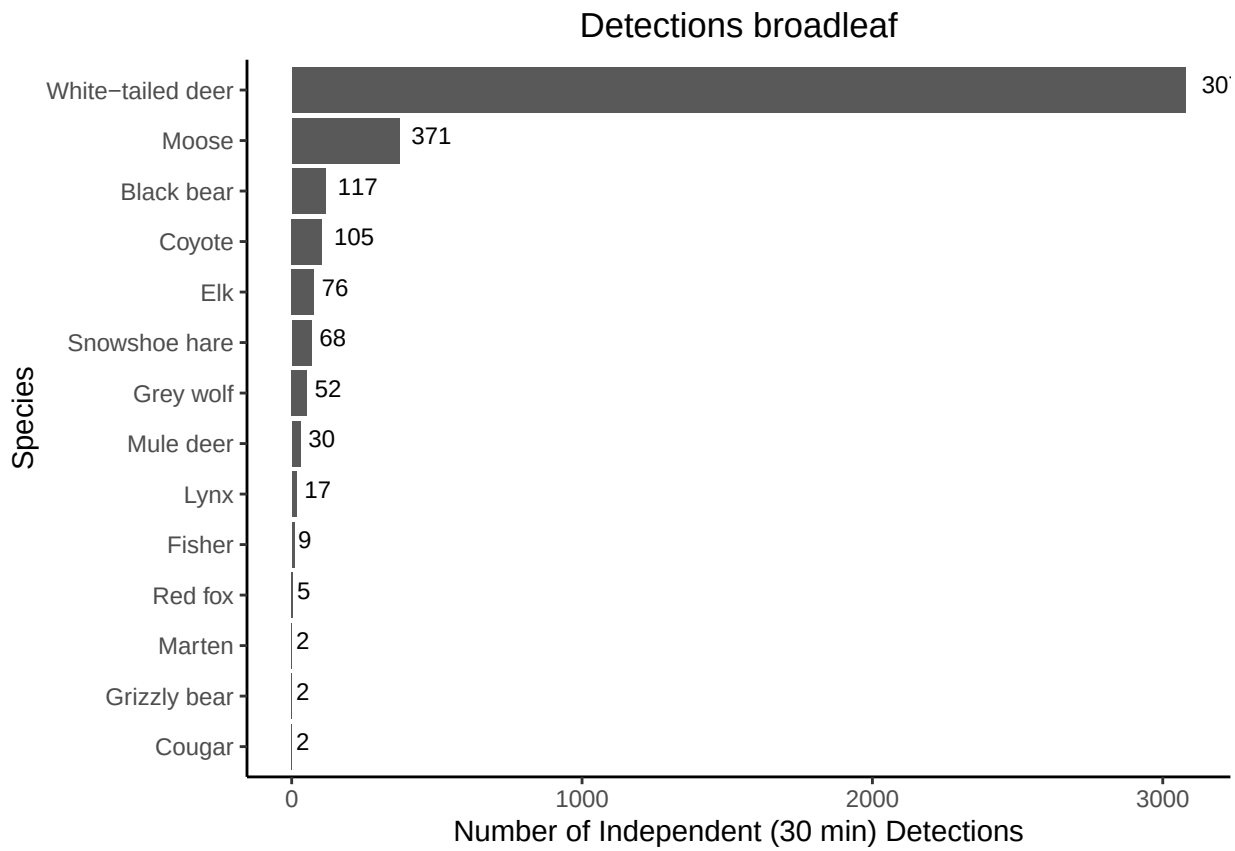
```r
# create object detection plots which uses the detection_data list (w/ all 4 LUs)
detection_plots <- detection_data %>%
```

```r
  # use imap instead of map as it allows us to use .y to paste the list element names as the plot title
  purrr::imap(
    ~.x %>%

      # now just copy and paste the ggplot code for the detection graphs
      ggplot(.,
             aes(x = reorder(species, count), y = count)) +

      # plot as bar graph using geom_col so we don't have to provide a y aesthetic
      geom_col() +

      # switch the x and y axis
      coord_flip() +

      # add the number of detections at the end of each bar
      geom_text(aes(label = count),
                color = "black",
                size = 3,
                hjust = -0.3,
                vjust = 0.2) +

      # label x and y axis with informative titles
      labs(x = 'Species',
           y = 'Number of Independent (30 min) Detections') +

      # add title to plot with LU name the .y will take the name of whatever you named each list elemen
      ggtitle(.y) +

      # set the theme
      theme_classic() +
      theme(plot.title = element_text(hjust = 0.5)))

# view plots, this will print each in it's own window so you have to scroll back in the plot viewer pan
detection_plots
```

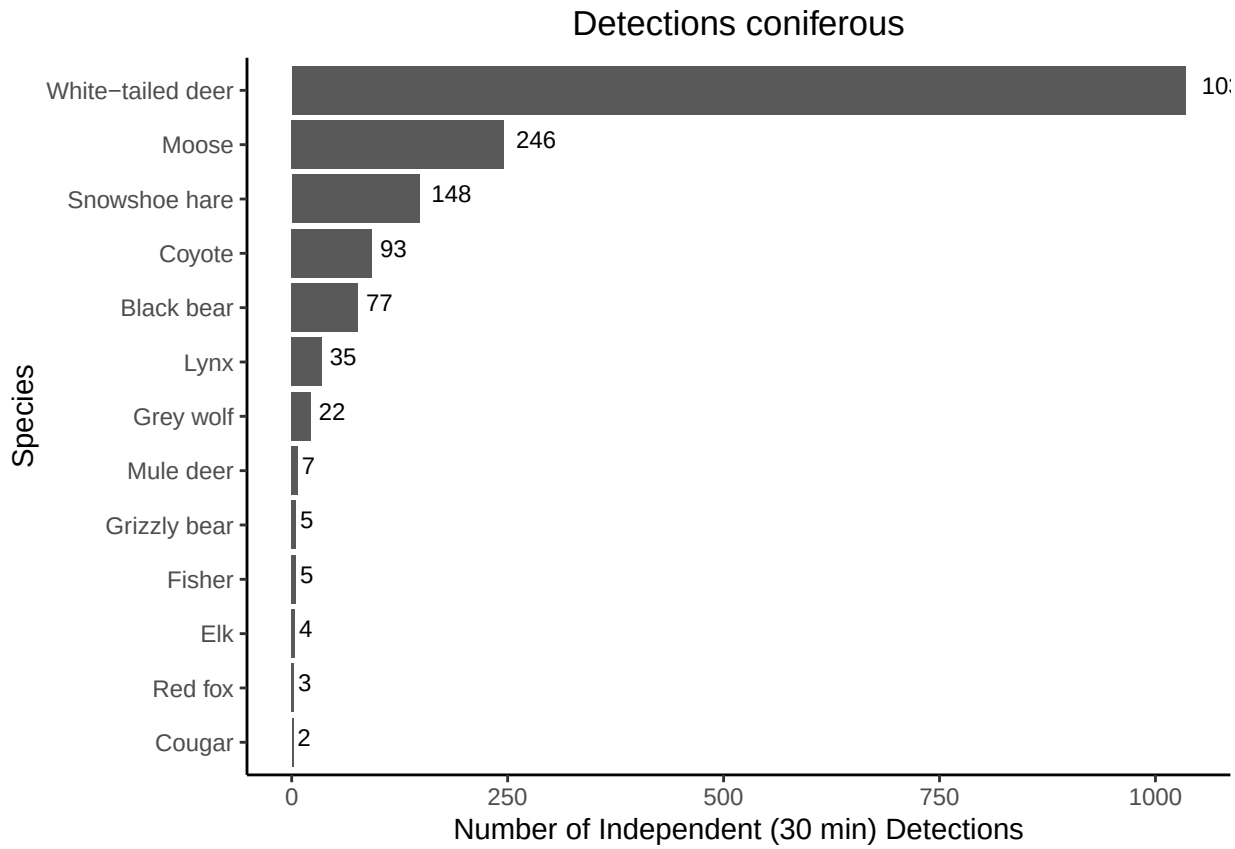
```
## $`Detections agriculture`
```



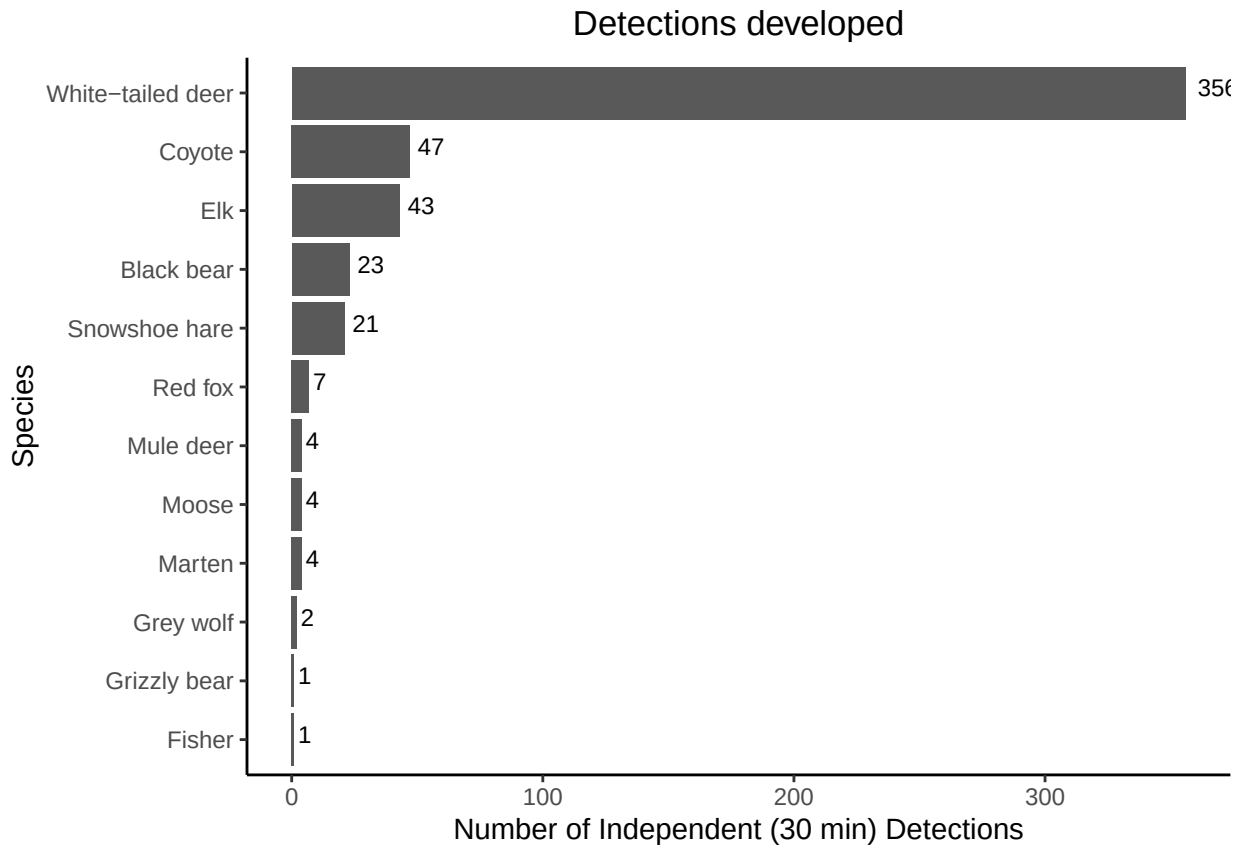

```
##
## $‘Detections broadleaf‘
```



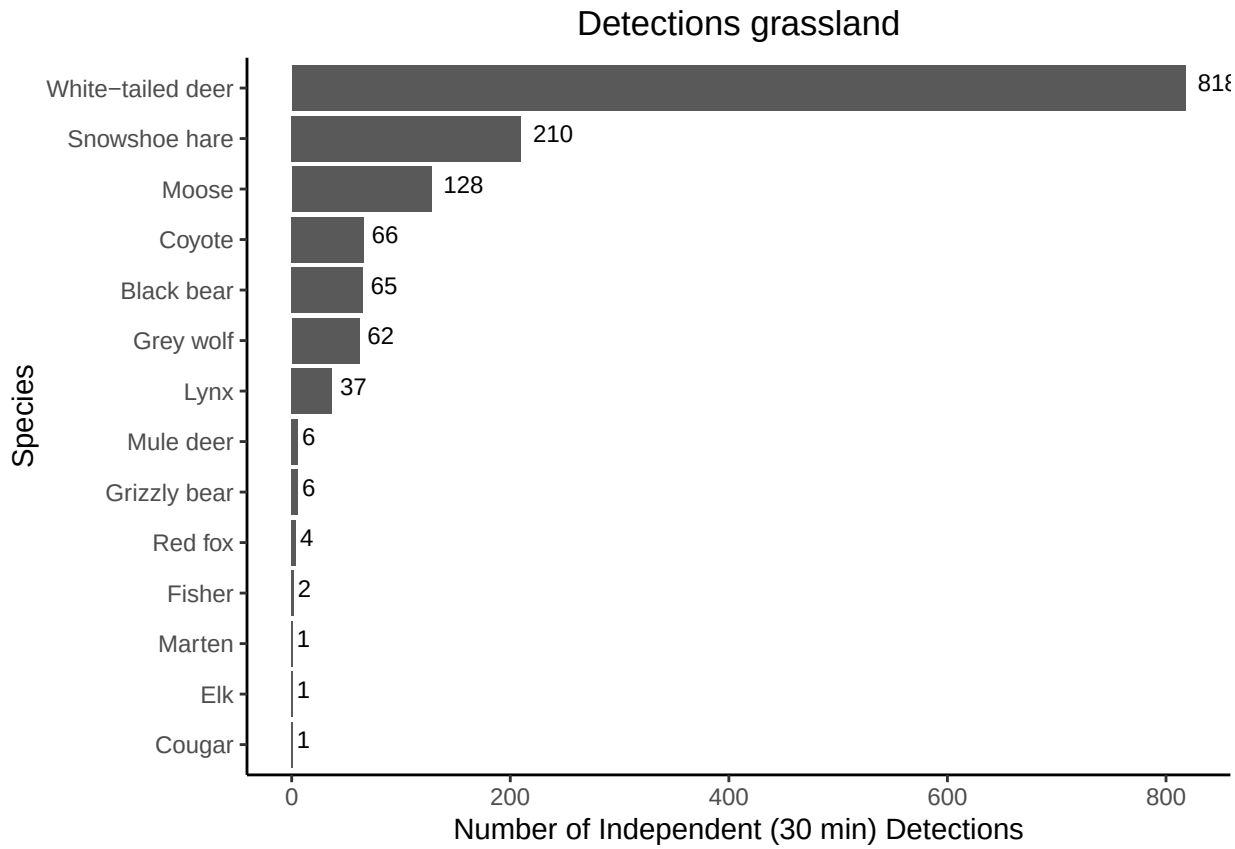

```
##
## $‘Detections coniferous‘
```



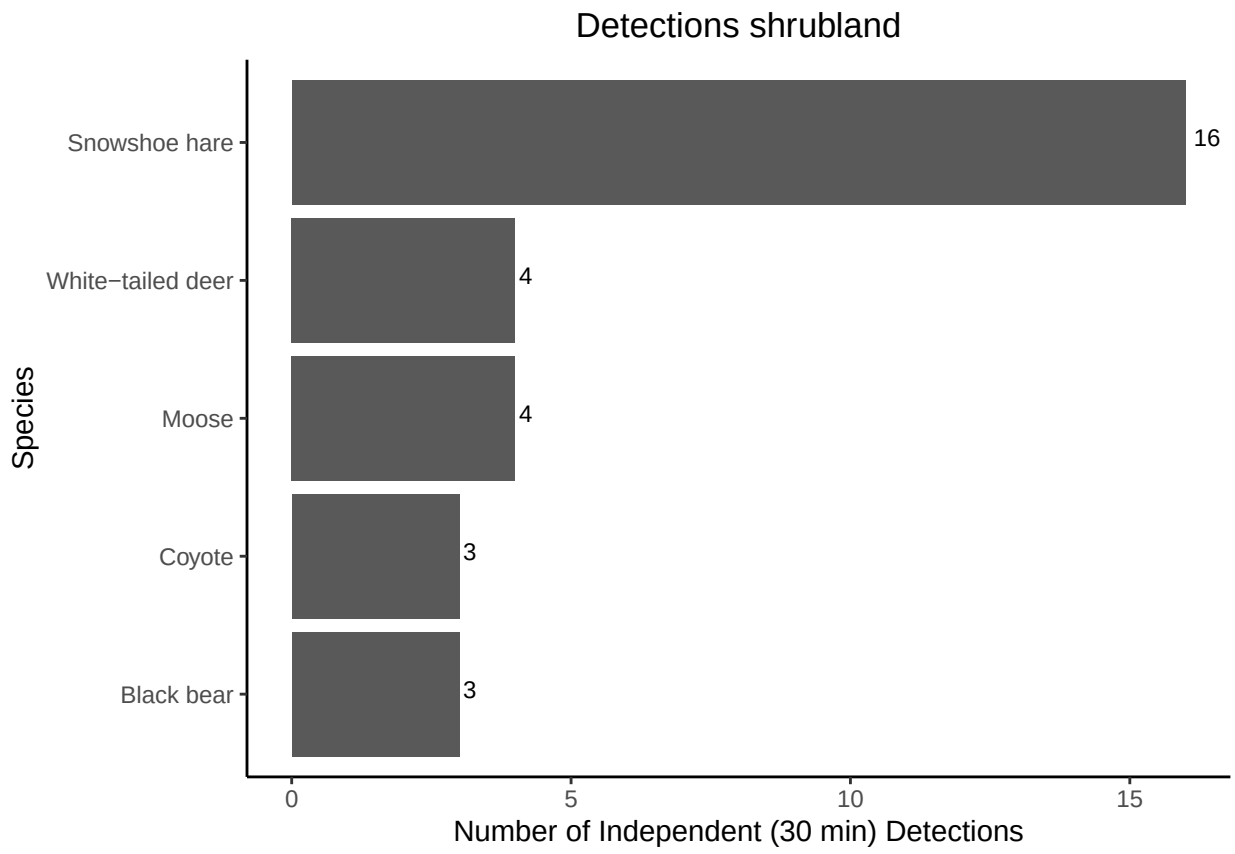

```
##
## $`Detections developed`
```



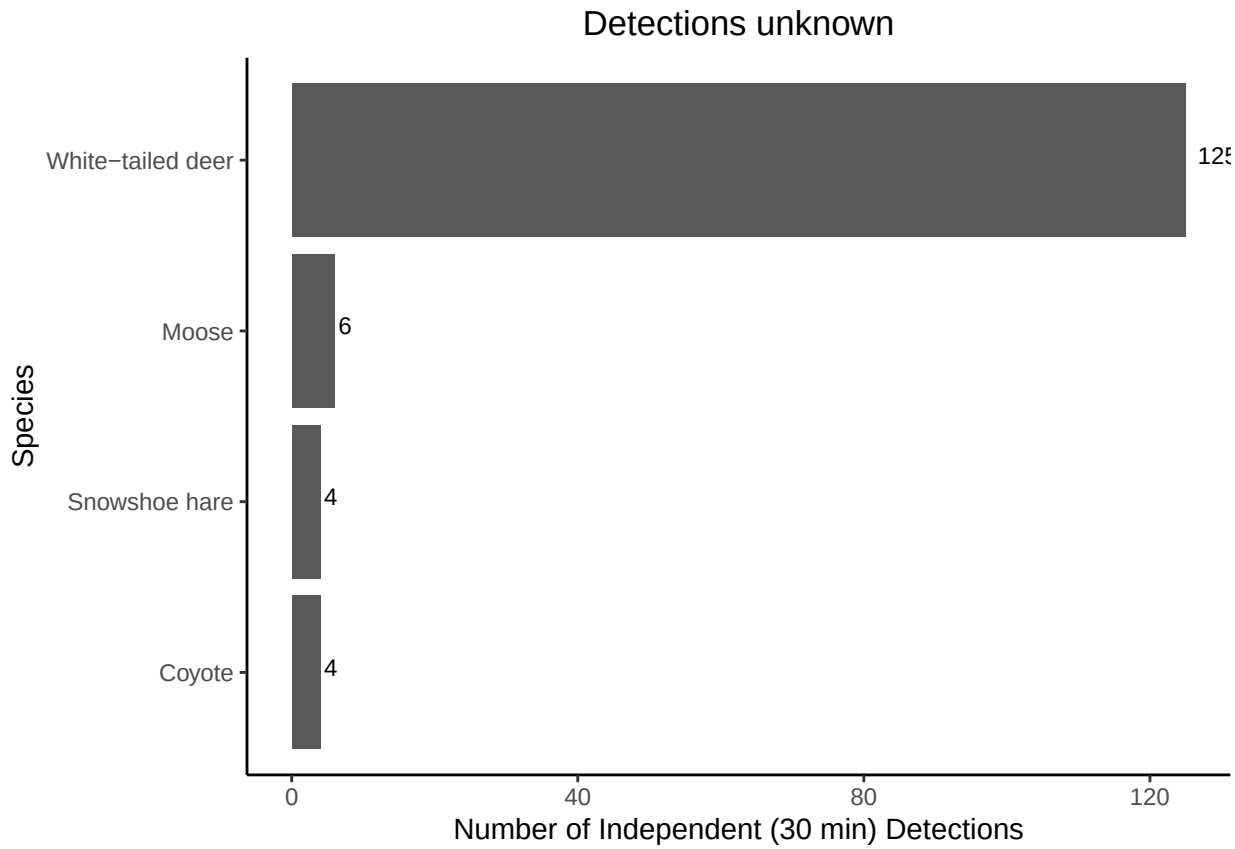

```
##
## $`Detections grassland`
```



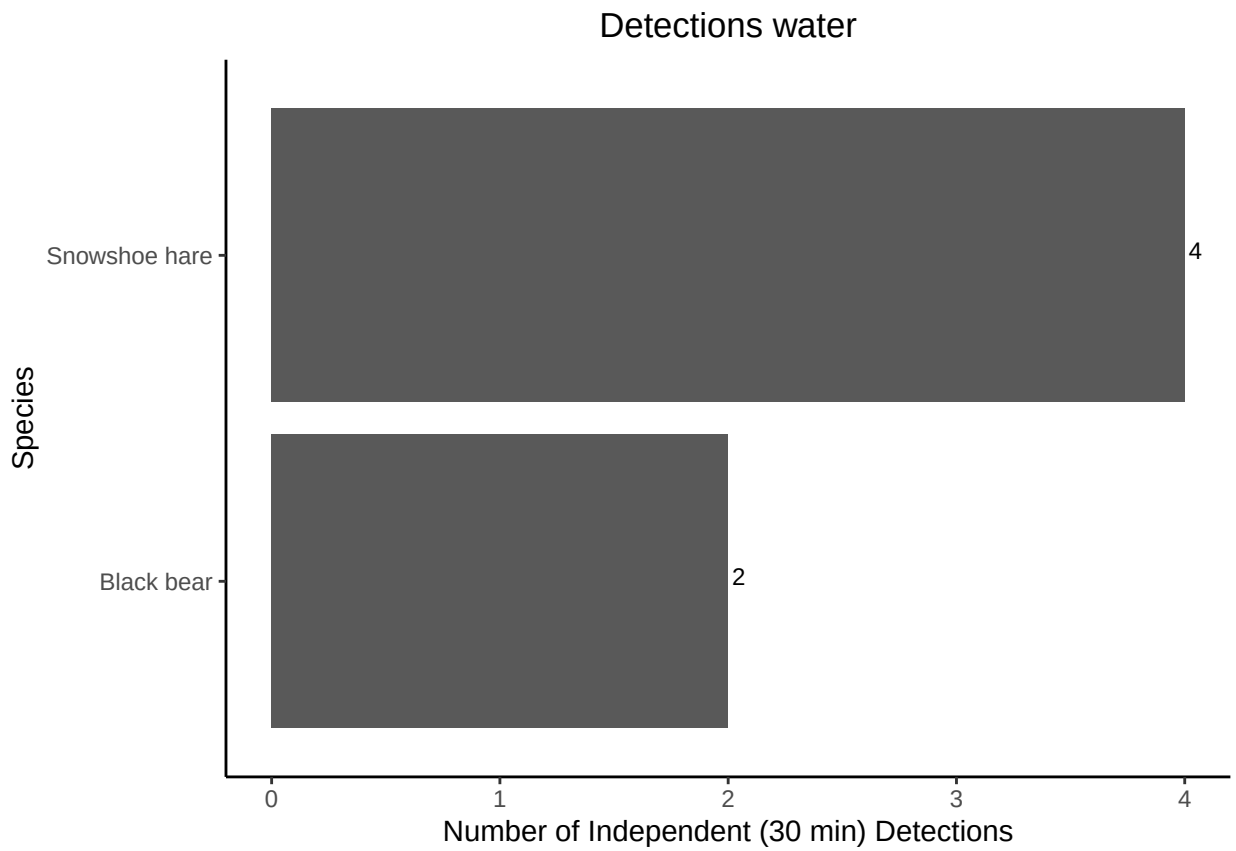

```
##
## $`Detections shrubland`
```




```
##
## $`Detections unknown`
```




```
##
## $`Detections water`
```



Save detection plots

Now we want to save these plots in case we need each individual one (we will combine the detection and naive occ plots into a single figure for each LU later and use those for the OSM report, but we may want these standalone plots later so let's save them while they are here).

We can save all the plots from the purrr iteration above using `purrr::imap`. imap is used instead of map because it allows us to retain the list object names (plot names) to paste as the file name with the .y command.

> IMPORTANT if you are using this code for a future github repo, DO NOT use .tiff as the file extension. This will cause issues when trying to push any changes to the github repo as the files are too large to meet githubs requirements

```
# save plots only use if needed
purrr::imap(
  detection_plots,
  ~ggsave(.x,
            file = paste0("figures/SRFN_",
                          .y,
                          '.jpg'), # avoid using .tiff extension in the github repo, those files are t
          dpi = 600,
          width = 11,
          height = 9,
          units = 'in'))
```

Naive occupancy

Data

We also need to alter the detection data a bit to use for naive occupancy plots.

We will use the individual LU detection data like we did before and use `purrr::map()` to apply the dame data formatting to all 4 data frames.

Here we want to calculate the total number of sites in each LU, the number of sites each species was detected at in each LU and then use both those numbers to calculate naive occupancy for each species in each LU

```
# First we need to alter the data frame a bit for these plots, let's create a data frame for each LU (I


# apply the same formatting to each data frame using purrr
occupancy_data <- array_frames %>%

  purrr::map(
    ~.x %>%

      # calculate the total number of sites for each LU
      mutate(total_sites = n_distinct(site)) %>%

      # group by species to calculate the number of sites each spp occurred at
      group_by(species) %>%

      # add columns to count the number of sites each spp occurred at and then the naive occupancy
  reframe(count = n_distinct(site),
          naive_occ = count/total_sites,
          ind_det = n_distinct(event_id)) %>%

    # keep just the columns we need
  select(species, naive_occ, ind_det) %>%

    # keep only unique (distinct) rows so we should be left with one row per species, this helps with p
  distinct()) %>%

  purrr::set_names(~ paste('Occupancy', names(array_frames)))
```

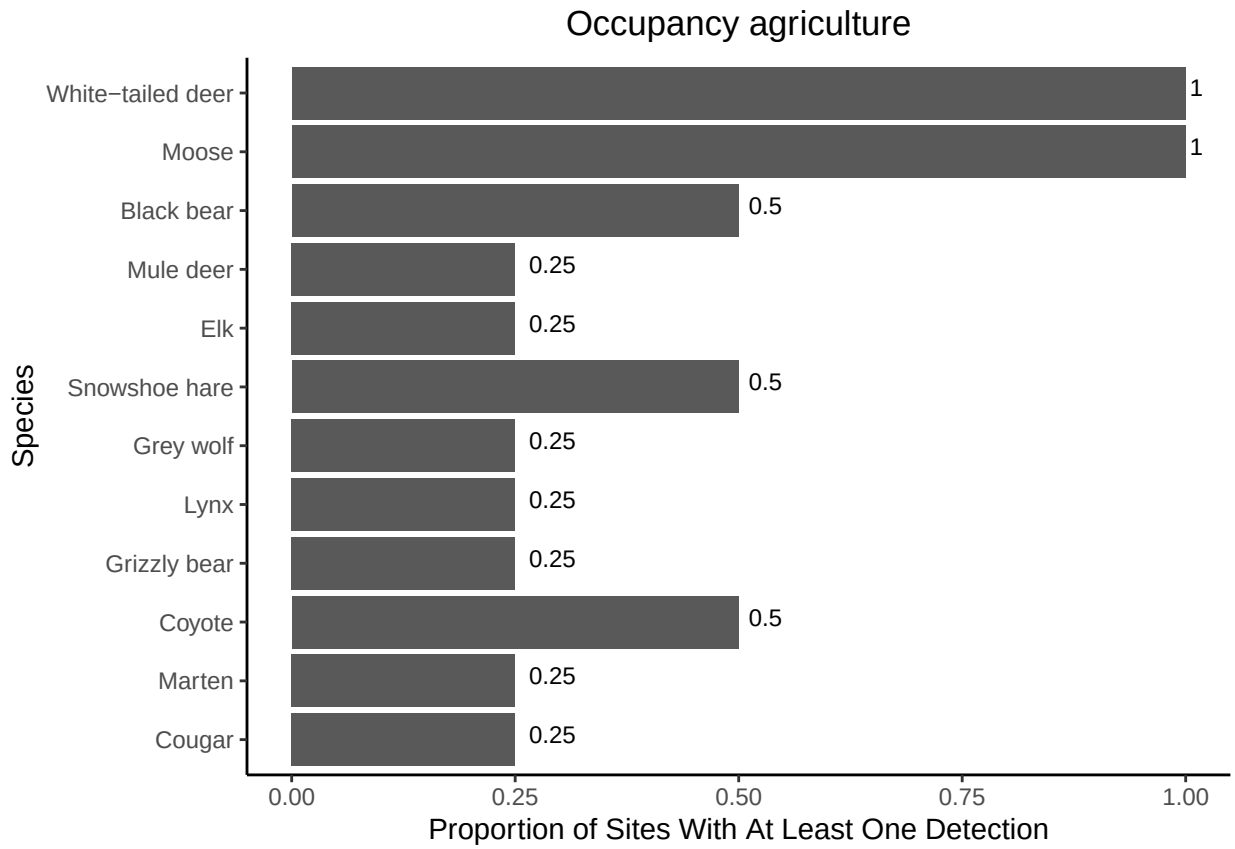
Occupancy plots

Now we can graph naive occupancy for each LU using purrr, and as with the detection plots this saves a massive amount of coding using purrr to run an iteration on the data files and produce four plots at once instead of copying and pasting code for each individually

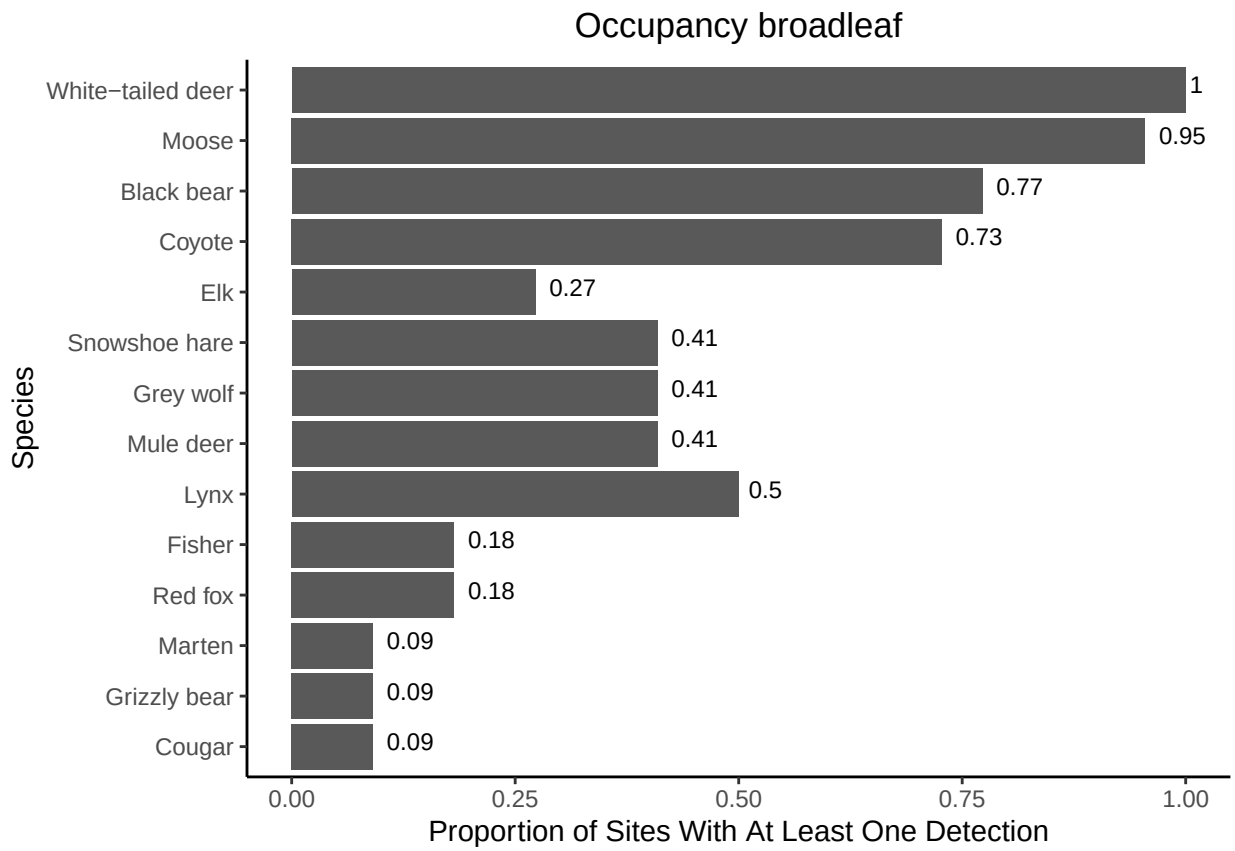
```
# create object occupancy_plots which uses the occupancy_data list (w/ all 4 LUs)
occupancy_plots <- occupancy_data %>%

  # use imap instead of map as it allows us to use .y to paste the list element names as the plot title
  purrr::imap(
    ~.x %>%

      # now just copy and paste the ggplot code for the occupancy graphs
      ggplot(.,
             aes(x = fct_reorder(species,
```

```
                                ind_det), # this reorders the species so they match the order of the d
                y = naive_occ)) +

      # plot as bars using geom_col() which uses stat = 'identity', instead of geom_bar() which will co
      geom_col() +

      # flip x and y axis
      coord_flip() +

      # add text to end of bars that provides naive occ value
      geom_text(aes(label = round(naive_occ, 2)),
                size = 3,
                hjust = -0.3,
                vjust = 0.2) +

      # relabel x and y axis and title
      labs(x = 'Species',
           y = 'Proportion of Sites With At Least One Detection') +

      # set plot title using .y (name of list object)
      ggtitle(.y) +

      # set. theme elements
      theme_classic()+
      theme(plot.title = element_text(hjust = 0.5)))

# view plots
occupancy_plots
```

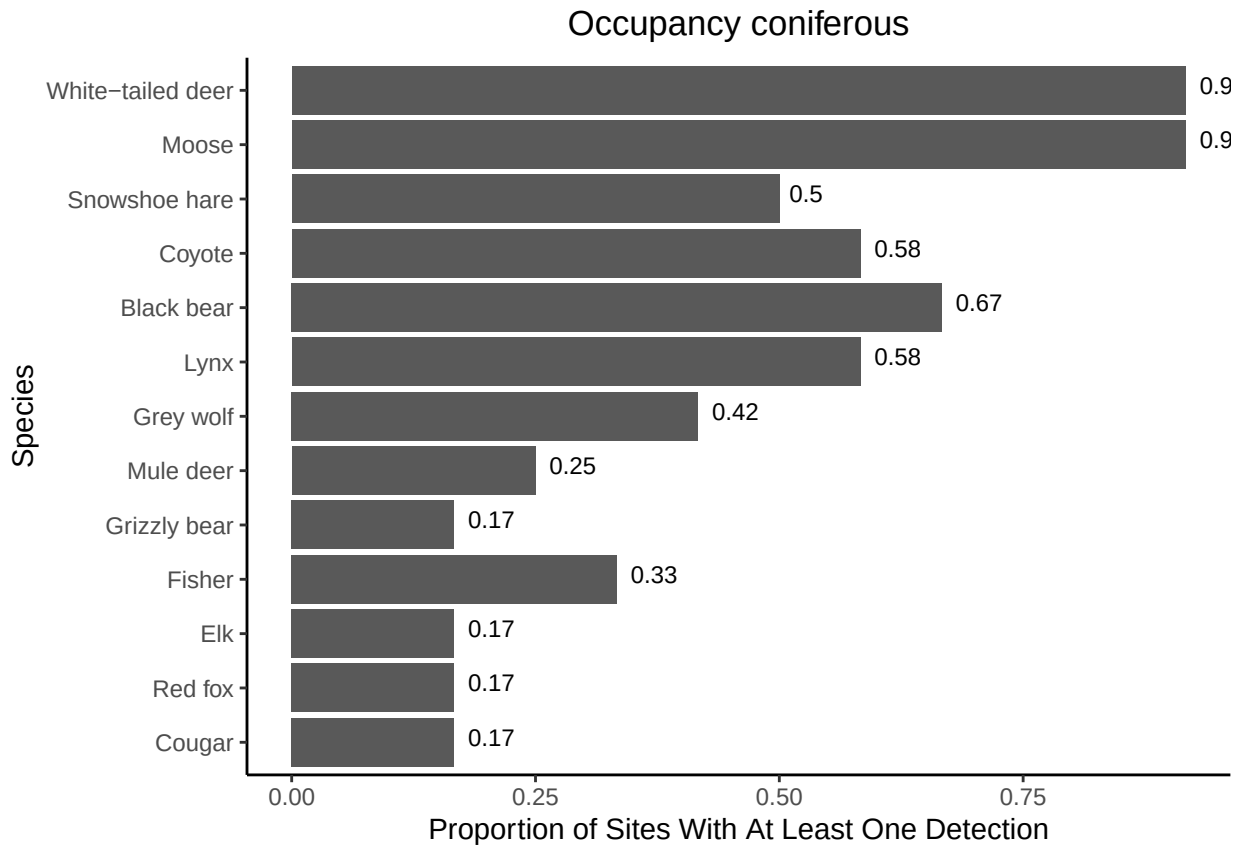
```
## $`Occupancy agriculture`
```




```
##
## $‘Occupancy broadleaf‘
```

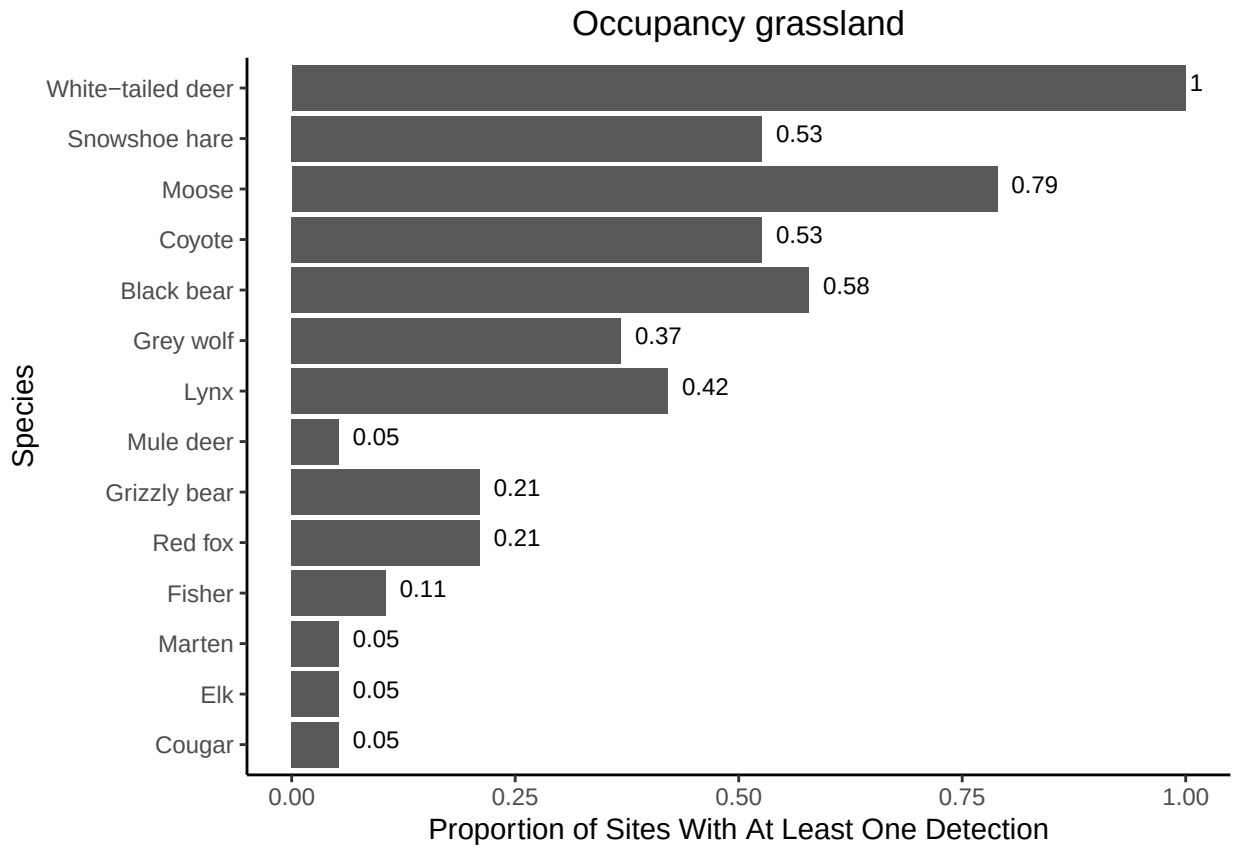



```
##
## $`Occupancy coniferous`
```

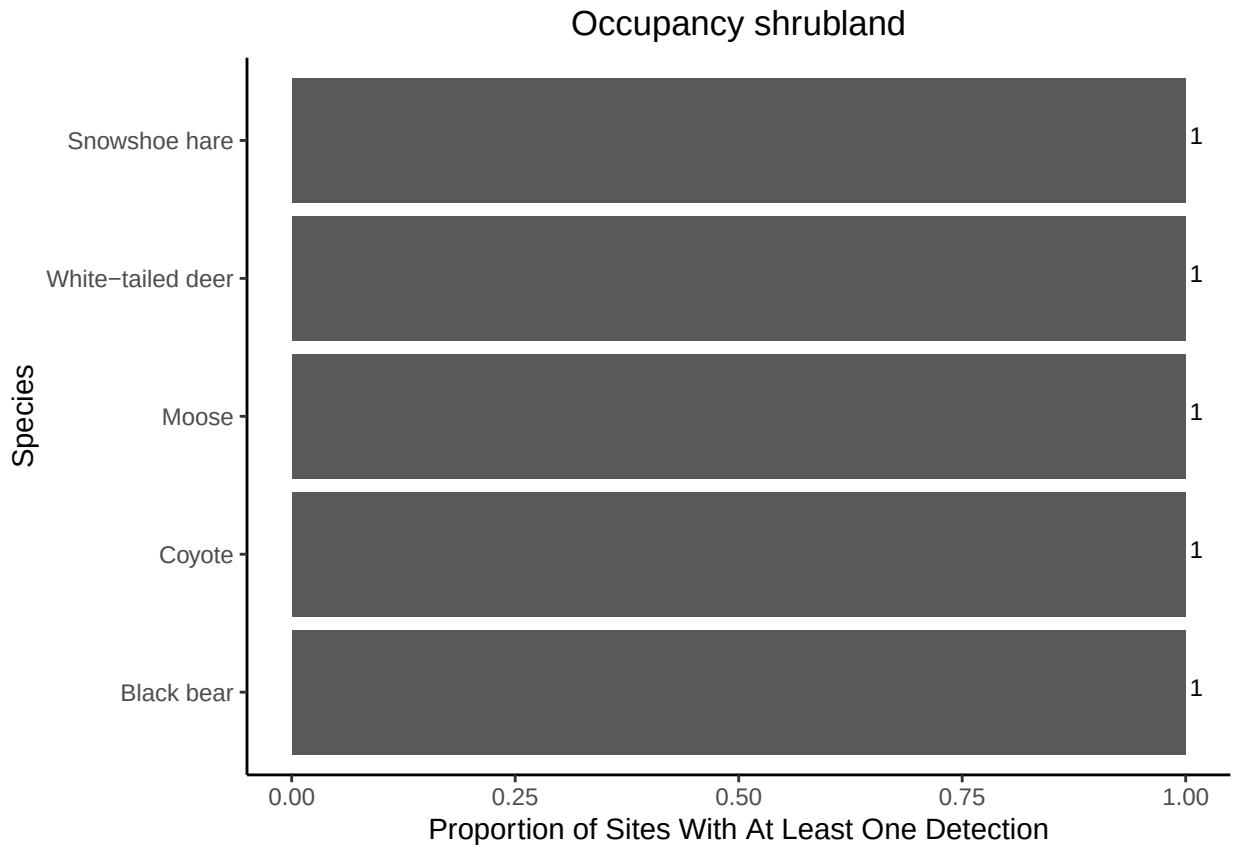



```
##
## $`Occupancy developed`
```

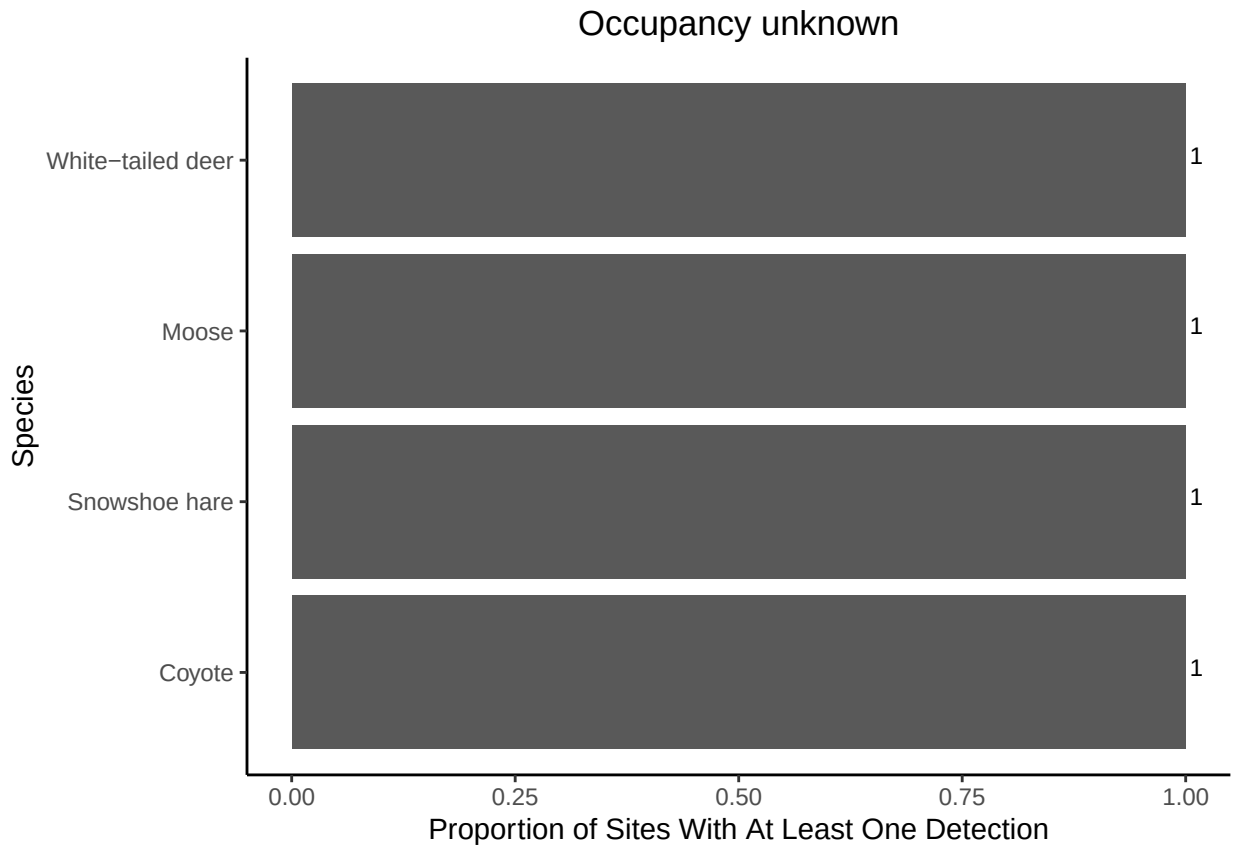

```
##
## $`Occupancy grassland`
```



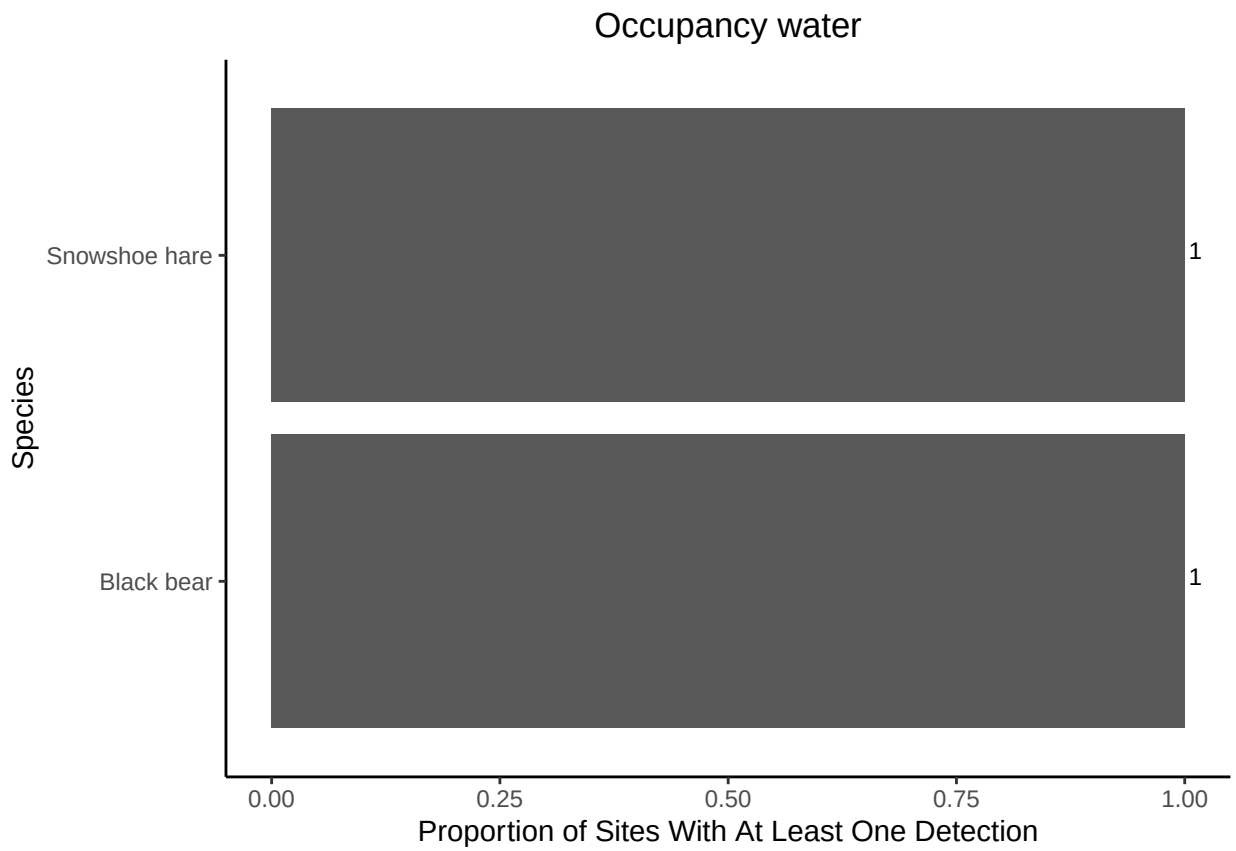

```
##
## $`Occupancy shrubland`
```




```
##
## $`Occupancy unknown`
```




```
##
## $`Occupancy water`
```



Save occupancy plots

As with the detection plots, we might want these individual plots later for something so we can use `purrr::imap()` to save them to the figures folder

Again avoid using the .tiff extension in github

```
# save plots
purrr::imap(
  occupancy_plots,
  ~ggsave(.x,
          file = paste0("figures/SRFN_",
                        .y,
                        '.jpg'), # avoid using .tiff extension in the github repo, those files are too
          dpi = 600,
          width = 11,
          height = 9,
          units = 'in'))
```

Full study occupancy

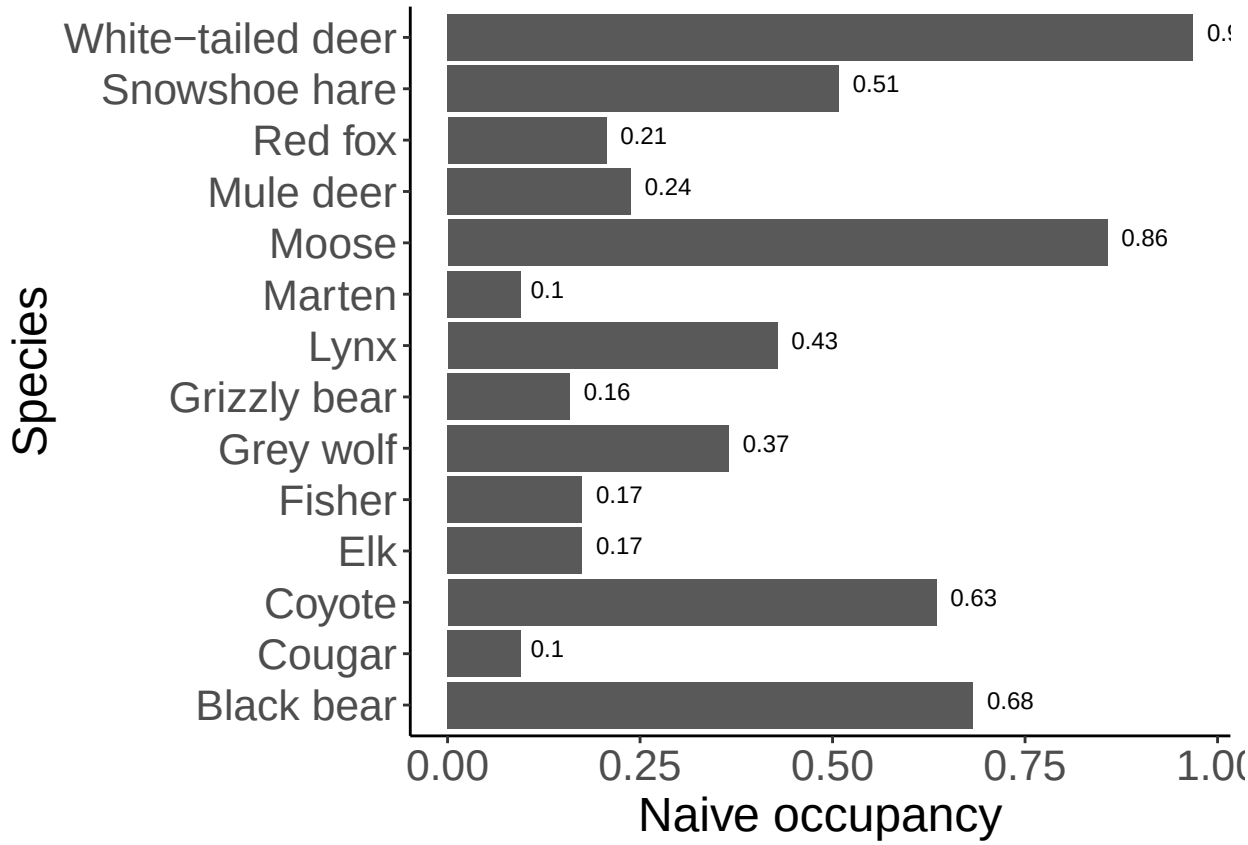
For the purposes of this project since the LUs represent different landscape types within the same study area (a bit different from LUs in the OSR project which this script was originally drafted for), let's also add a plot with naive occupancy for all LUs

```
naive_occ <- detections %>%
```

```
      # calculate the total number of sites
      mutate(total_sites = n_distinct(site)) %>%

  # Convert species to a factor with alphabetical levels
  mutate(species = factor(as.character(species),
                          levels = sort(unique(as.character(species))))) %>%

      # group by species to calculate the number of sites each spp occurred at
      group_by(species) %>%

      # add columns to count the number of sites each spp occurred at and then the naive occupancy
  reframe(count = n_distinct(site),
          naive_occ = count/total_sites,
          ind_det = n_distinct(event_id)) %>%

    # keep just the columns we need
  select(species, naive_occ, ind_det) %>%

    # keep only unique (distinct) rows so we should be left with one row per species, this helps with p
  distinct()
```

Now plot

```
naive_occ_plot <-  ggplot(data = naive_occ,
             aes(x = species,
                 y = naive_occ)) +

      # plot as bars using geom_col() which uses stat = 'identity', instead of geom_bar() which will co
      geom_col() +

      # flip x and y axis
      coord_flip() +

      # add text to end of bars that provides naive occ value
      geom_text(aes(label = round(naive_occ, 2)),
                size = 3,
                hjust = -0.3,
                vjust = 0.2) +

      # relabel x and y axis and title
      labs(x = 'Species',
           y = 'Naive occupancy') +

      # set. theme elements
      theme_classic()+
       theme(axis.text = element_text(size = 16),
        axis.title = element_text(size = 18))

naive_occ_plot
```



And lastly save this figure

```
ggsave('figures/SRFN_naive_occupancy.jpg',
       naive_occ_plot,
       dpi = 600,
       width = 11,
       height = 9,
       units = 'in')
```

Site data

I also want to make a histogram of how many sites there were in each habitat category, will do that with the code below

Import data

First we need to import the deployment fixed data from script #1

```
deploy_fixed <- read_csv('data/processed/srfn_deploy_fixed.csv')
```

```
## Rows: 64 Columns: 6
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (3): project_id, site, array
## dbl  (1): site_number
## date (2): start_date, end_date
##
```

```
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
str(deploy_fixed)
```

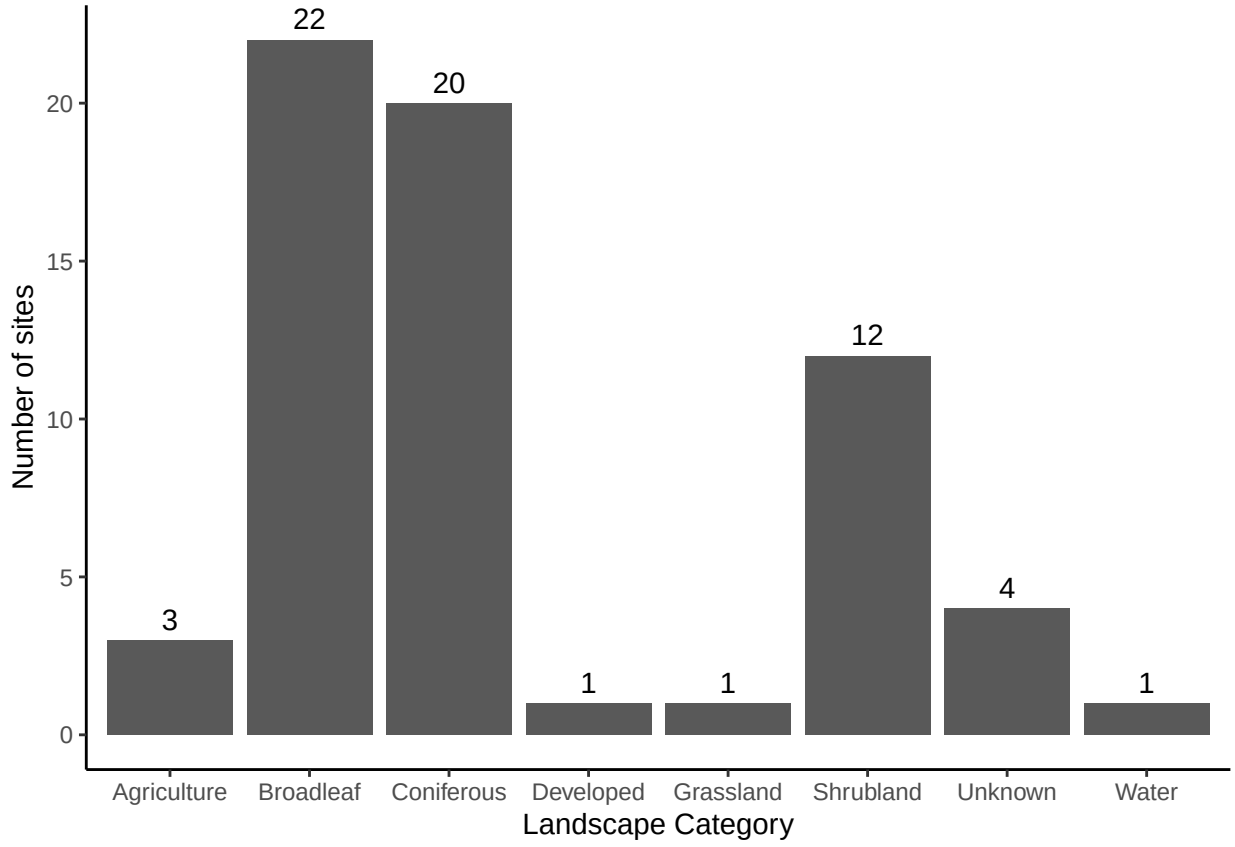
```
## spc_tbl_ [64 x 6] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ project_id : chr [1:64] "SRFN" "SRFN" "SRFN" "SRFN" ...
##  $ site_number: num [1:64] 1 2 4 6 10 12 13 17 18 21 ...
##  $ site       : chr [1:64] "LUD_1" "LUC_2" "LUC_4" "LUS_6" ...
##  $ array      : chr [1:64] "LUD" "LUC" "LUC" "LUS" ...
##  $ start_date : Date[1:64], format: "2022-04-06" "2022-04-07" ...
##  $ end_date   : Date[1:64], format: "2023-10-03" "2023-10-04" ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   project_id = col_character(),
##   ..   site_number = col_double(),
##   ..   site = col_character(),
##   ..   array = col_character(),
##   ..   start_date = col_date(format = ""),
##   ..   end_date = col_date(format = "")
##   .. )
##  - attr(*, "problems")=<externalptr>
```

Plot

Now we can have tidyverse functions count the number of different array entries (landcover types) and pipe this to ggplot to make a nice histogram

```
# start with data
sites_plot <- deploy_fixed %>%

  # add counts
  count(array) %>%

# pipe to ggplot to create histogram of number of sites
ggplot(aes(x = array,
           y = n)) +

  # add bars for each category
   geom_bar(stat = 'identity',
            linewidth = 1) +

  # add number of sites (count) above each bar
  geom_text(aes(label = n),
            vjust = -0.5) +

  # adjust axis labels
  labs(x = 'Landscape Category',
       y = 'Number of sites') +

  # manually change axis labels for x-axis
  scale_x_discrete(labels = c('Agriculture',
                              'Broadleaf',
                              'Coniferous',
                              'Developed',
```

```
                              'Grassland',
                              'Shrubland',
                              'Unknown',
                              'Water')) +
  # adjust theme
  theme_classic()

sites_plot
```



Save plot

Use ggsave to save to figures folder

```
ggsave('figures/srfn_sites_histogram.jpg',
       sites_plot,
       dpi = 600,
       width = 11,
       height = 9,
       units = 'in')
```

GLM Plots

First we need to provide the final data we used for the models and the top model for each species needs to be defined and fit again in this script

Read in data

```
prop_det_data <- readRDS('data/processed/prop_det_data.rds')
```

Define and fit top models for each species

```
# black bear
# linear disturbance (transportation + linear energy development)
bbear_linear <- glm(cbind(black_bear, absent_black_bear) ~
                      scale(roads) +
                      # pipeline + can't include correlated w/ roads 0.62
                      scale(seismic_lines),
                    data = prop_det_data$`250 meter buffer`,
                    family = 'binomial')


# coyote
# overall human disturbance (can't include pipeline or wells which are correlated with roads)
coyote_disturb <- glm(cbind(coyote, absent_coyote) ~
                       scale(harvest_2000) +
                       scale(harvest_pre2000) +
                       scale(roads) +
                       scale(seismic_lines),
                     data = prop_det_data$`5000 meter buffer`,
                     family = 'binomial')

# grey wolf
# linear + natural (have to pix max of 5 variables) based on the detections per habitat type I chose sh
grey_wolf_linear_nat <- glm(cbind(grey_wolf, absent_grey_wolf) ~
                              scale(roads) +
                              scale(seismic_lines) +
                              scale(lc_mixed)  +
                              scale(lc_shrub),
                    data = prop_det_data$`5000 meter buffer`,
                    family = 'binomial')

# Lynx
# polygonal disturbance (harvest + polygonal energy development + agriculture)
lynx_poly <- glm(cbind(lynx, absent_lynx) ~
                   scale(harvest_2000) +
                   scale(harvest_pre2000) +
                   scale(wells),
                 data = prop_det_data$`500 meter buffer`,
                 family = 'binomial')

# Moose
# linear disturbance (transportation + linear energy development)
moose_linear <- glm(cbind(moose, absent_moose) ~
                      scale(roads) +
                      # pipeline + can't include correlated w/ roads 0.62
                      scale(seismic_lines),
                    data = prop_det_data$`500 meter buffer`,
                    family = 'binomial')
```

```
# snowshoe hare
# overall human disturbance (limit to 5 vars)
snowshoe_hare_disturb <- glm(cbind(snowshoe_hare, absent_snowshoe_hare) ~
                      scale(harvest_2000) +
                     scale(harvest_pre2000) +
                       # scale(roads) + correlated w/ wells 0.80
                       # pipeline + can't include correlated w/ wells 0.91
                       scale(seismic_lines) +
                       scale(wells),
                     data = prop_det_data$`4000 meter buffer`,
                     family = 'binomial')

# white-tailed deer
# Natural heterogeneity (checked how taking out broadleaf or grassland affected model results since thi
w_deer_nat <- glm(cbind(`white-tailed_deer`, `absent_white-tailed_deer`) ~
                       scale(lc_broadleaf) +
                      scale(lc_grassland)  +
                       scale(lc_mixed)  +
                       scale(lc_shrub),
                     data = prop_det_data$`4250 meter buffer`,
                     family = 'binomial')
```

Odds plots

Black bear

Let's extract the odds ratios for the top black bear model so we can plot them later.

```
bbear_model_odds <-

  # extract the coefficients and upper and lower CI
  tidy(bbear_linear, conf.int = TRUE) %>%

  # convert the coefficients into odds ratios
  mutate(estimate = exp(estimate),
         conf.low = exp(conf.low),
         conf.high = exp(conf.high)) %>%

  # rename columns
  rename('lower' = conf.low,
         'upper' = conf.high) %>%

  # Remove intercept for plotting
filter(term != '(Intercept)')
```

First let's get a silhouette for this graphy from phylopic

```
black_bear_img <- get_phylopic(get_uuid(name = 'Ursus americanus'))
```

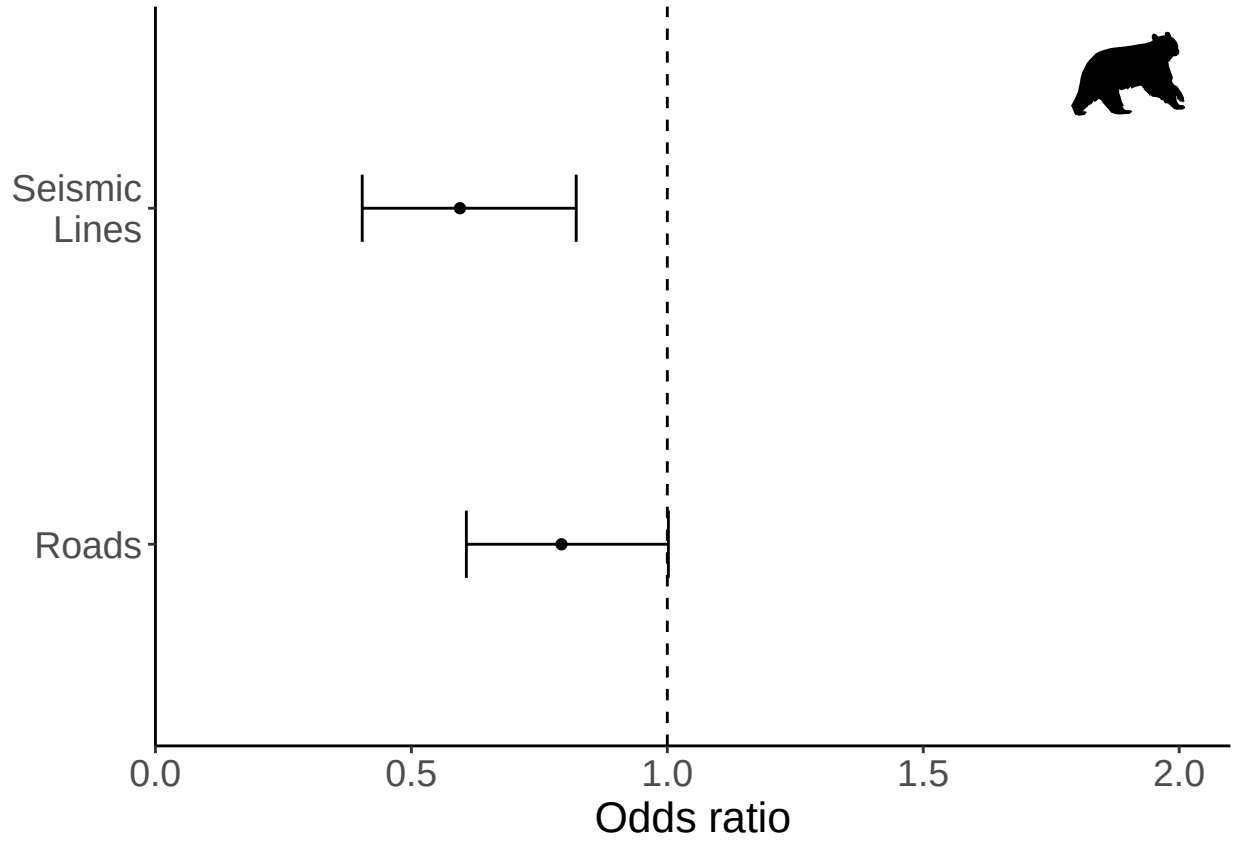
Now let's use ggplot to plot the odds ratios for each feature in the top model

```
# name to save plot later
bbear_odds_plot <-
# provide data and mapping aesthetics
ggplot(bbear_model_odds, aes(x = term,
```

```
                        y = estimate)) +

  geom_point() +

   geom_errorbar(aes(ymin = lower,
                     ymax = upper),
                 width = 0.2,
                 linewidth = 0.5,
                 position = position_dodge(width = 0.9)) +

   geom_hline(yintercept = 1, linetype = "dashed") +

  labs(y = 'Odds ratio') +

  scale_x_discrete(labels = ~ str_wrap(as.character(c('Roads',
                                                      'Seismic Lines')),
                                       9)) +

  # scale_x_discrete(labels = c('Roads',
  #                             'Seismic Lines')) +

  scale_y_continuous(expand = expansion(mult = c(0, 0.05)),
                     limits = c(0, 2)) +

  coord_flip() +

  add_phylopic(black_bear_img,
               x = 2.4,
               y = 1.9,
               ysize = 0.25) +

  theme_classic() +

  theme(axis.title.y = element_blank(),
        axis.title = element_text(size = 16),
        axis.text = element_text(size = 14))
```

```
## Warning: The ‘ysize‘ argument of ‘add_phylopic()‘ is deprecated as of rphylopic 1.5.0.
## i Please use the ‘height‘ argument instead.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```

```
bbear_odds_plot
```



Save plot

```
ggsave('figures/odds_plot_blackbear.jpg',
       bbear_odds_plot,
       width = 10,
       height = 8,
       units = 'in',
       dpi = 600)
```

Now we will do the same for each remaining species

Coyote

First we need to extract odds ratios from the model

```
coyote_model_odds <-

  # extract the coefficients and upper and lower CI
  tidy(coyote_disturb, conf.int = TRUE) %>%

  # convert the coefficients into odds ratios
  mutate(estimate = exp(estimate),
         conf.low = exp(conf.low),
         conf.high = exp(conf.high)) %>%

  # rename columns
  rename('lower' = conf.low,
         'upper' = conf.high) %>%
```

```
  # Remove intercept for plotting
filter(term != '(Intercept)')
```

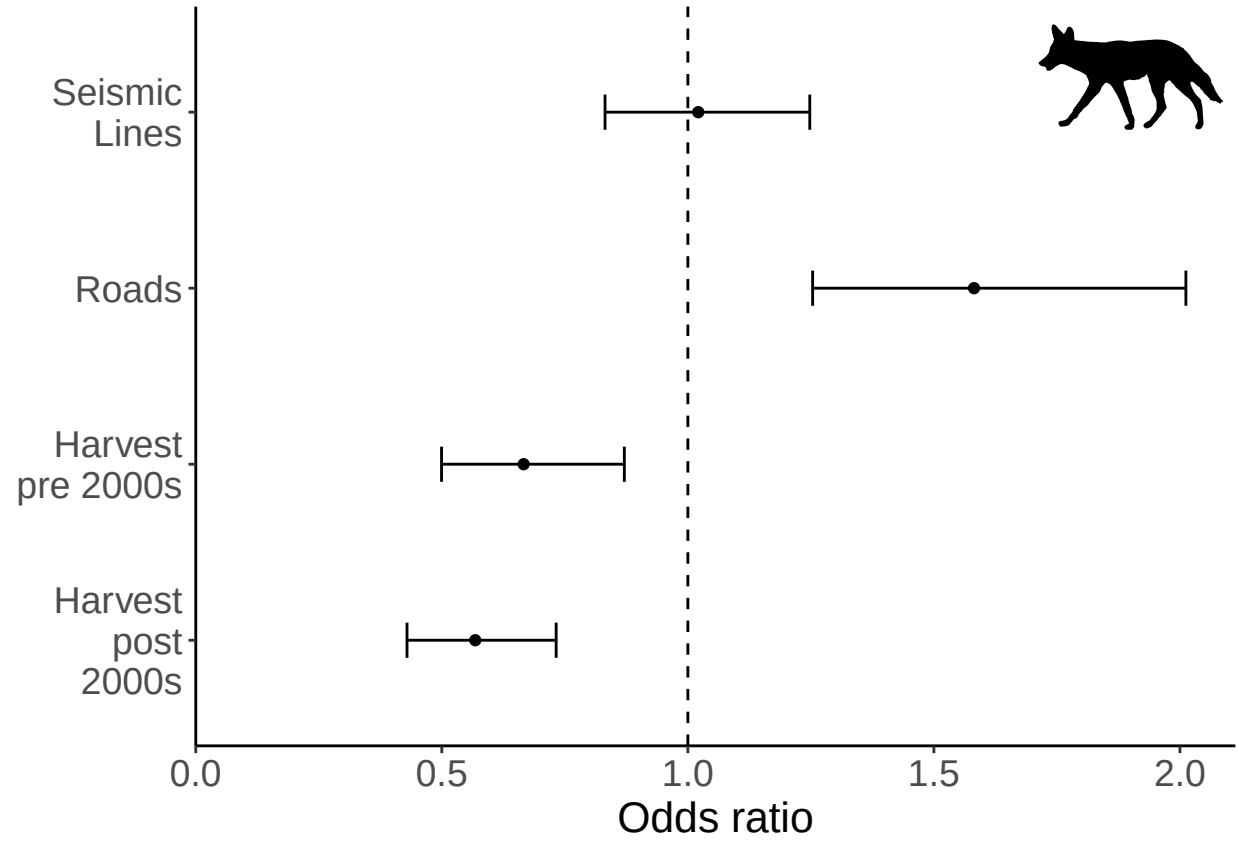
Get coyote silhouette

```
# this time I copied the specific phylopic uuid I want from the website in the image url
coyote_img <- get_phylopic('d451e353-585a-4543-84e7-7ef2f90aa407')
```

Plot odds

```
# name plot
coyote_odds_plot <-

  # provide data and mapping aesthetics
  ggplot(coyote_model_odds, aes(x = term,
                                y = estimate)) +

  geom_point() +

  geom_errorbar(aes(ymin = lower,
                    ymax = upper),
                width = 0.2,
                linewidth = 0.5,
                position = position_dodge(width = 0.9)) +

  geom_hline(yintercept = 1, linetype = "dashed") +

  labs(y = 'Odds ratio') +

  scale_x_discrete(labels = ~ str_wrap(as.character(c('Harvest post 2000s',
                               'Harvest pre 2000s',
                               'Roads',
                               'Seismic Lines')),
                                       9)) +

  # scale_x_discrete(labels = c('Harvest post 2000s',
  #                             'Harvest pre 2000s',
  #                             'Roads',
  #                             'Seismic Lines')) +

  scale_y_continuous(expand = expansion(mult = c(0, 0.05)),
                     limits = c(0, NA)) +

  coord_flip() +

  add_phylopic(coyote_img,
               x = 4.2,
               y = 1.9,
               ysize = 0.6) +

  theme_classic() +

  theme(axis.title.y = element_blank(),
        axis.title = element_text(size = 16),
        axis.text = element_text(size = 14))
```

```
coyote_odds_plot
```



Save plot

```
ggsave('figures/odds_plot_coyote.jpg',
       coyote_odds_plot,
       width = 10,
       height = 8,
       units = 'in',
       dpi = 600)
```

Grey wolf

```
grey_wolf_model_odds <-

  # extract the coefficients and upper and lower CI
  tidy(grey_wolf_linear_nat, conf.int = TRUE) %>%

  # convert the coefficients into odds ratios
  mutate(estimate = exp(estimate),
         conf.low = exp(conf.low),
         conf.high = exp(conf.high)) %>%

  # rename columns
  rename('lower' = conf.low,
```

```
         'upper' = conf.high) %>%

  # Remove intercept for plotting
filter(term != '(Intercept)')
```

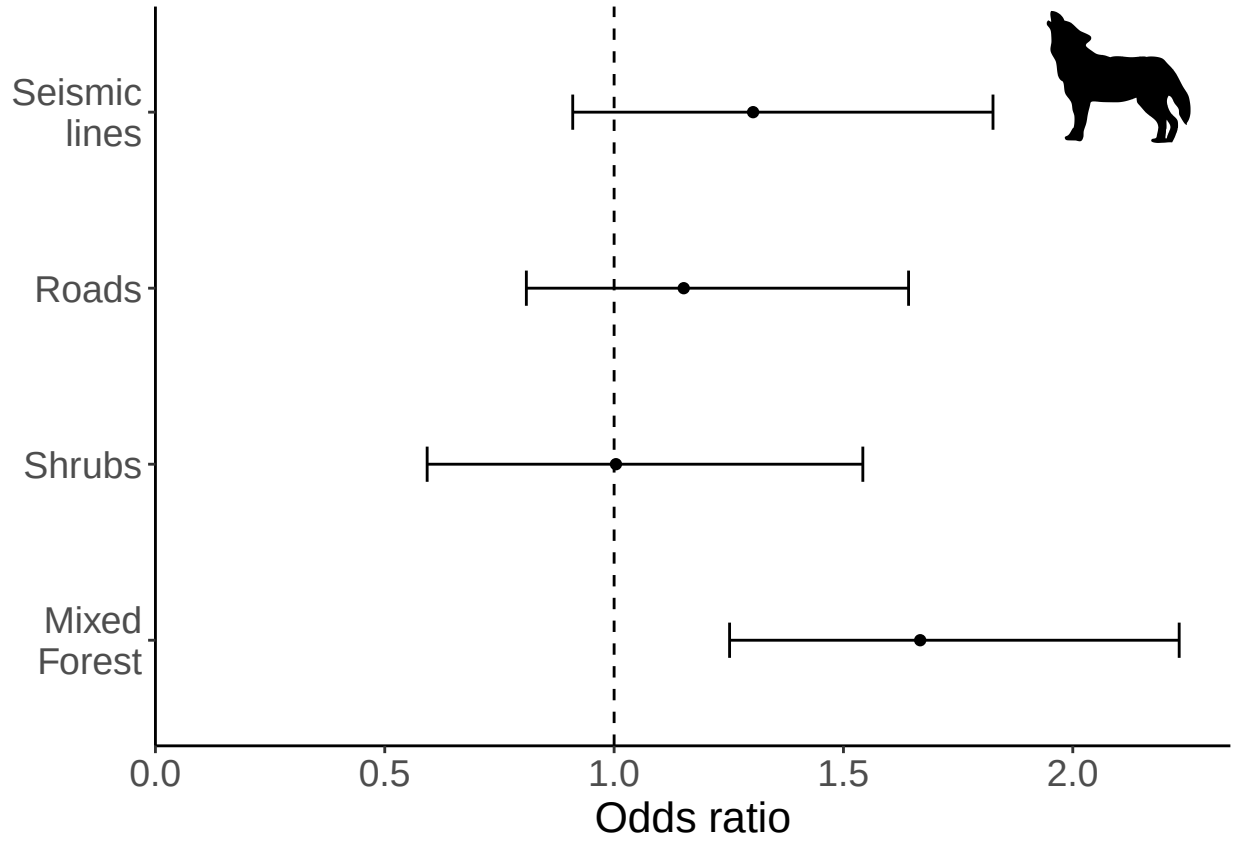
Get wolf silhouette

```
wolf_img <- get_phylopic('e4e306cd-73b6-4ca3-a08c-753a856f7f12')
```

Plot odss

```
# name to save plot later
grey_wolf_odds_plot <-
  # provide data and mapping aesthetics
  ggplot(grey_wolf_model_odds, aes(x = term,
                                   y = estimate)) +

  geom_point() +

  geom_errorbar(aes(ymin = lower,
                    ymax = upper),
                width = 0.2,
                linewidth = 0.5,
                position = position_dodge(width = 0.9)) +

  geom_hline(yintercept = 1, linetype = "dashed") +

  labs(y = 'Odds ratio') +

  scale_x_discrete(labels = ~ str_wrap(as.character(c('Mixed Forest',
                               'Shrubs',
                               'Roads',
                               'Seismic lines')),
                                       9)) +

  # scale_x_discrete(labels = c('Mixed Forest',
  #                             'Shrubs',
  #                             'Roads',
  #                             'Seismic lines')) +

  scale_y_continuous(expand = expansion(mult = c(0, 0.05)),
                     limits = c(0, NA)) +

  coord_flip() +

  add_phylopic(wolf_img,
               x = 4.2,
               y = 2.1,
               ysize = 0.75) +

  theme_classic() +

  theme(axis.title.y = element_blank(),
        axis.title = element_text(size = 16),
        axis.text = element_text(size = 14))
```

```
grey_wolf_odds_plot
```



Save plot

```
ggsave('figures/odds_plot_greywolf.jpg',
       grey_wolf_odds_plot,
       width = 10,
       height = 8,
       units = 'in',
       dpi = 600)
```

Lynx

Calculate odds

```
lynx_model_odds <-

  # extract the coefficients and upper and lower CI
  tidy(lynx_poly, conf.int = TRUE) %>%

  # convert the coefficients into odds ratios
  mutate(estimate = exp(estimate),
         conf.low = exp(conf.low),
         conf.high = exp(conf.high)) %>%

  # rename columns
```

```
  rename('lower' = conf.low,
         'upper' = conf.high) %>%

  # Remove intercept for plotting
filter(term != '(Intercept)')
```

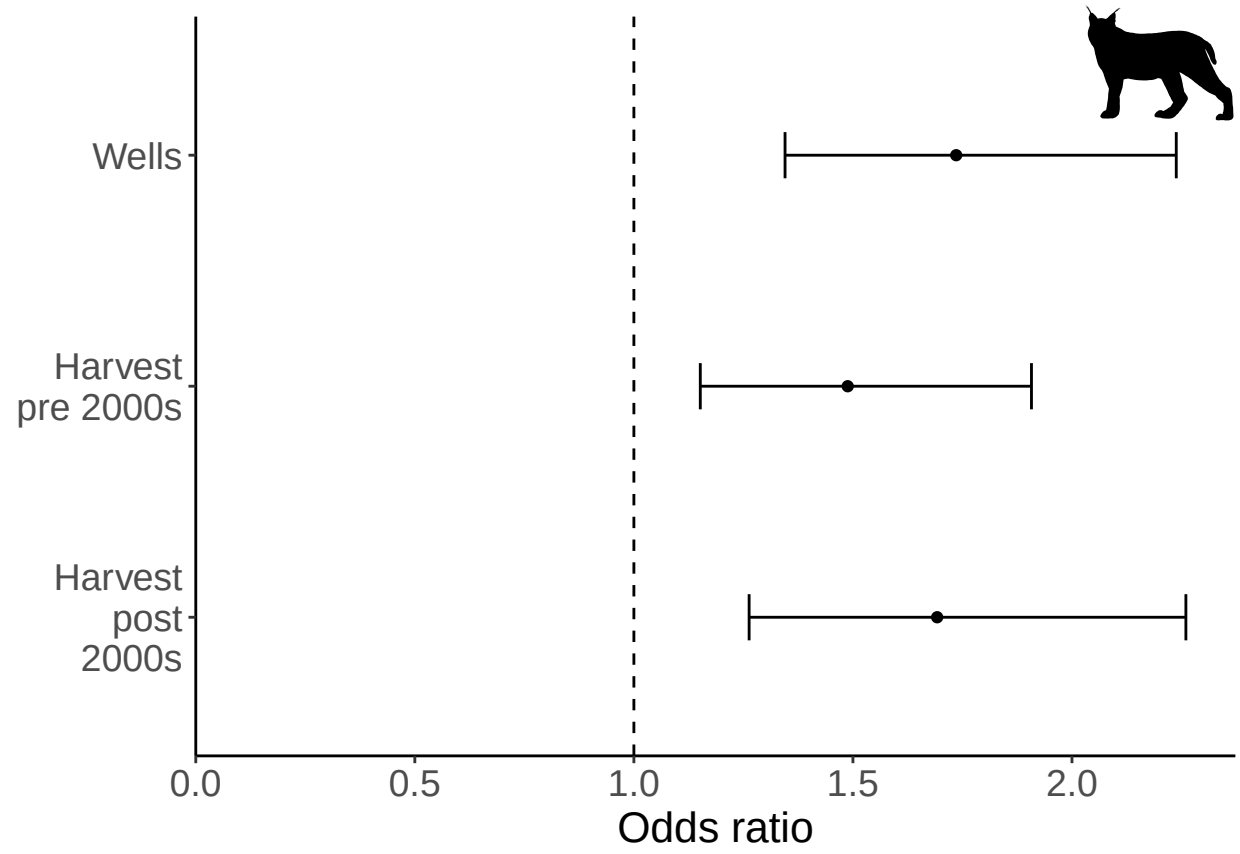
Get lynx silhouette

```
lynx_img <- get_phylopic('24f763a3-accf-44c9-9a08-71e9834047b7')
```

Plot odds

```
# name to save plot later
lynx_odds_plot <-
# provide data and mapping aesthetics
ggplot(lynx_model_odds, aes(x = term,
                            y = estimate)) +

  geom_point() +

   geom_errorbar(aes(ymin = lower,
                     ymax = upper),
                 width = 0.2,
                 linewidth = 0.5,
                 position = position_dodge(width = 0.9)) +

   geom_hline(yintercept = 1, linetype = "dashed") +

  labs(y = 'Odds ratio') +

    scale_x_discrete(labels = ~ str_wrap(as.character(c('Harvest post 2000s',
                                'Harvest pre 2000s',
                                'Wells')),
                                        9)) +

  # scale_x_discrete(labels = c('Harvest ppost 2000s',
  #                             'Harvest pre 2000s',
  #                             'Wells')) +

  scale_y_continuous(expand = expansion(mult = c(0, 0.05)),
                     limits = c(0, NA)) +

  coord_flip() +

  add_phylopic(lynx_img,
               x = 3.4,
               y = 2.2,
               ysize = 0.5) +

  theme_classic() +

  theme(axis.title.y = element_blank(),
        axis.title = element_text(size = 16),
        axis.text = element_text(size = 14))
```

```
lynx_odds_plot
```



Save plot

```
ggsave('figures/odds_plot_lynx.jpg',
       lynx_odds_plot,
       width = 10,
       height = 8,
       units = 'in',
       dpi = 600)
```

Moose

Calculate odds

```
moose_model_odds <-

  # extract the coefficients and upper and lower CI
  tidy(moose_linear, conf.int = TRUE) %>%

  # convert the coefficients into odds ratios
  mutate(estimate = exp(estimate),
         conf.low = exp(conf.low),
         conf.high = exp(conf.high)) %>%

  # rename columns
  rename('lower' = conf.low,
```

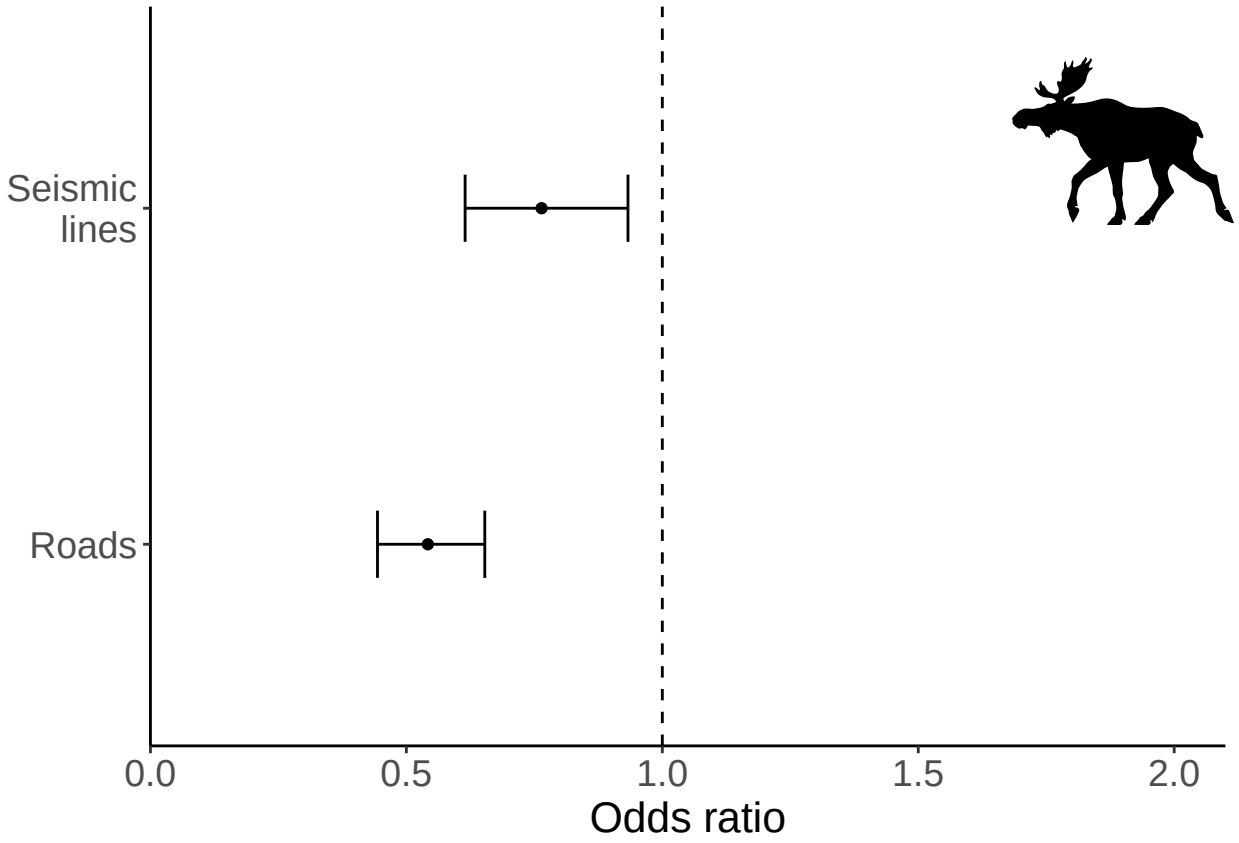
```
         'upper' = conf.high) %>%

  # Remove intercept for plotting
filter(term != '(Intercept)')
```

Get silhouette for plots

```
moose_img <- get_phylopic('74eab34a-498c-4614-aece-f02361874f79')
```

Plot odds

```
# name to save plot later
moose_odds_plot <-
# provide data and mapping aesthetics
ggplot(moose_model_odds, aes(x = term,
                             y = estimate)) +

  geom_point() +

   geom_errorbar(aes(ymin = lower,
                     ymax = upper),
                 width = 0.2,
                 linewidth = 0.5,
                 position = position_dodge(width = 0.9)) +

   geom_hline(yintercept = 1, linetype = "dashed") +

  labs(y = 'Odds ratio') +

    scale_x_discrete(labels = ~ str_wrap(as.character(c('Roads',
                               'Seismic lines')),
                                         9)) +

  # scale_x_discrete(labels = c('Roads',
  #                             'Seismic lines')) +

  scale_y_continuous(expand = expansion(mult = c(0, 0.05)),
                     limits = c(0, 2)) +

  coord_flip() +

  add_phylopic(moose_img,
               x = 2.2,
               y = 1.9,
               ysize = 0.5) +

  theme_classic() +

  theme(axis.title.y = element_blank(),
        axis.title = element_text(size = 16),
        axis.text = element_text(size = 14))

moose_odds_plot
```



Save plot

```
ggsave('figures/odds_plot_moose.jpg',
       moose_odds_plot,
        width = 10,
        height = 8,
        units = 'in',
        dpi = 600)
```

Snowshoe hare

Calculate odds

```
hare_model_odds <-

  # extract the coefficients and upper and lower CI
  tidy(snowshoe_hare_disturb, conf.int = TRUE) %>%

  # convert the coefficients into odds ratios
  mutate(estimate = exp(estimate),
         conf.low = exp(conf.low),
         conf.high = exp(conf.high)) %>%

  # rename columns
  rename('lower' = conf.low,
         'upper' = conf.high) %>%

  # Remove intercept for plotting
```

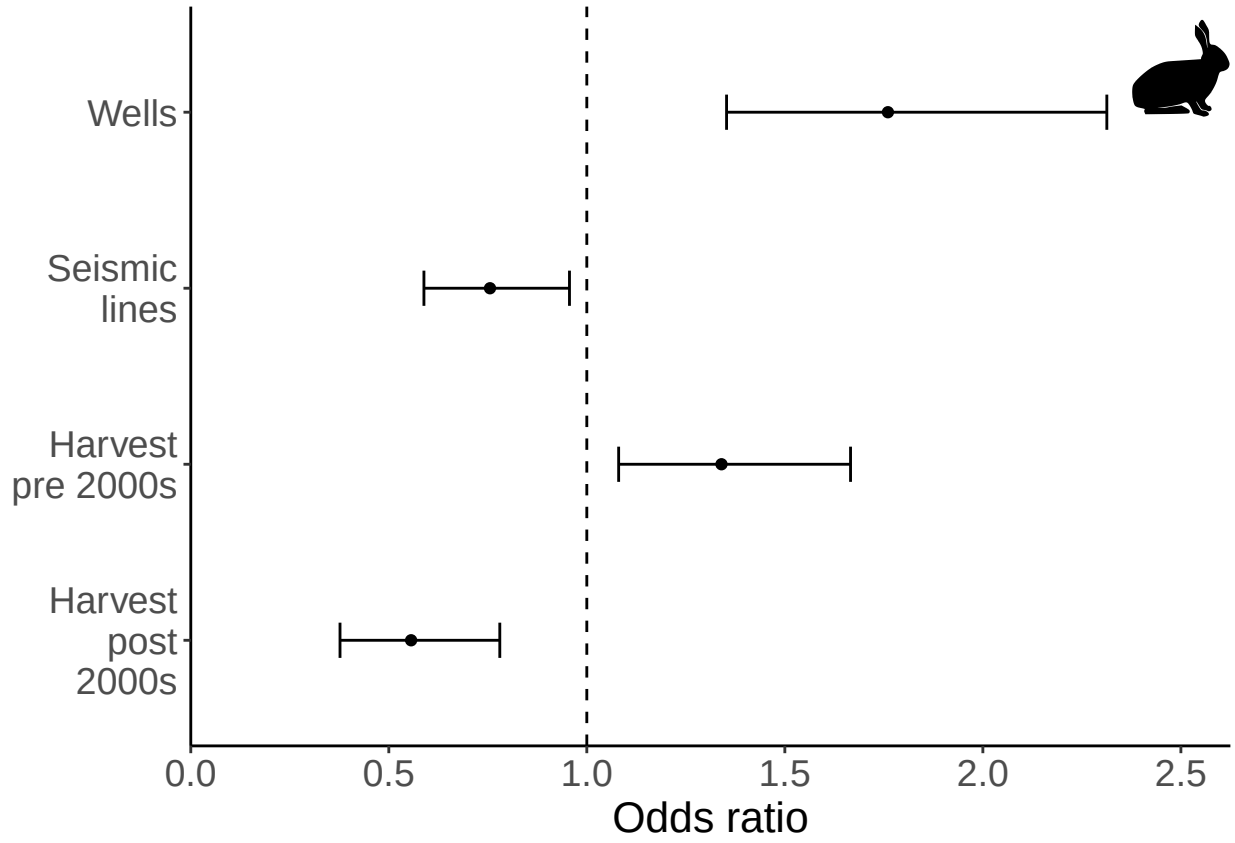
```
filter(term != '(Intercept)')
```

Get silhouette for plots

```
hare_img <- get_phylopic(get_uuid(name = 'Lepus americanus'))
```

Plot odds

```
# name to save plot later
hare_odds_plot <-
# provide data and mapping aesthetics
ggplot(hare_model_odds, aes(x = term,
                            y = estimate)) +

  geom_point() +

   geom_errorbar(aes(ymin = lower,
                     ymax = upper),
                 width = 0.2,
                 linewidth = 0.5,
                 position = position_dodge(width = 0.9)) +

   geom_hline(yintercept = 1, linetype = "dashed") +

  labs(y = 'Odds ratio') +

      scale_x_discrete(labels = ~ str_wrap(as.character(c('Harvest post 2000s',
                                'Harvest pre 2000s',
                                'Seismic lines',
                                'Wells')),
                                        9)) +

  # scale_x_discrete(labels = c('Harvest post 2000s',
  #                             'Harvest pre 2000s',
  #                             'Seismic lines',
  #                             'Wells')) +

  scale_y_continuous(expand = expansion(mult = c(0, 0.05)),
                     limits = c(0, NA)) +

  coord_flip() +

  add_phylopic(hare_img,
               x = 4.25,
               y = 2.5,
               ysize = 0.55) +

  theme_classic() +

  theme(axis.title.y = element_blank(),
        axis.title = element_text(size = 16),
        axis.text = element_text(size = 14))

hare_odds_plot
```



Save plot

```
ggsave('figures/odds_plot_snowshoehare.jpg',
       hare_odds_plot,
       width = 10,
       height = 8,
       units = 'in',
       dpi = 600)
```

White-tailed deer

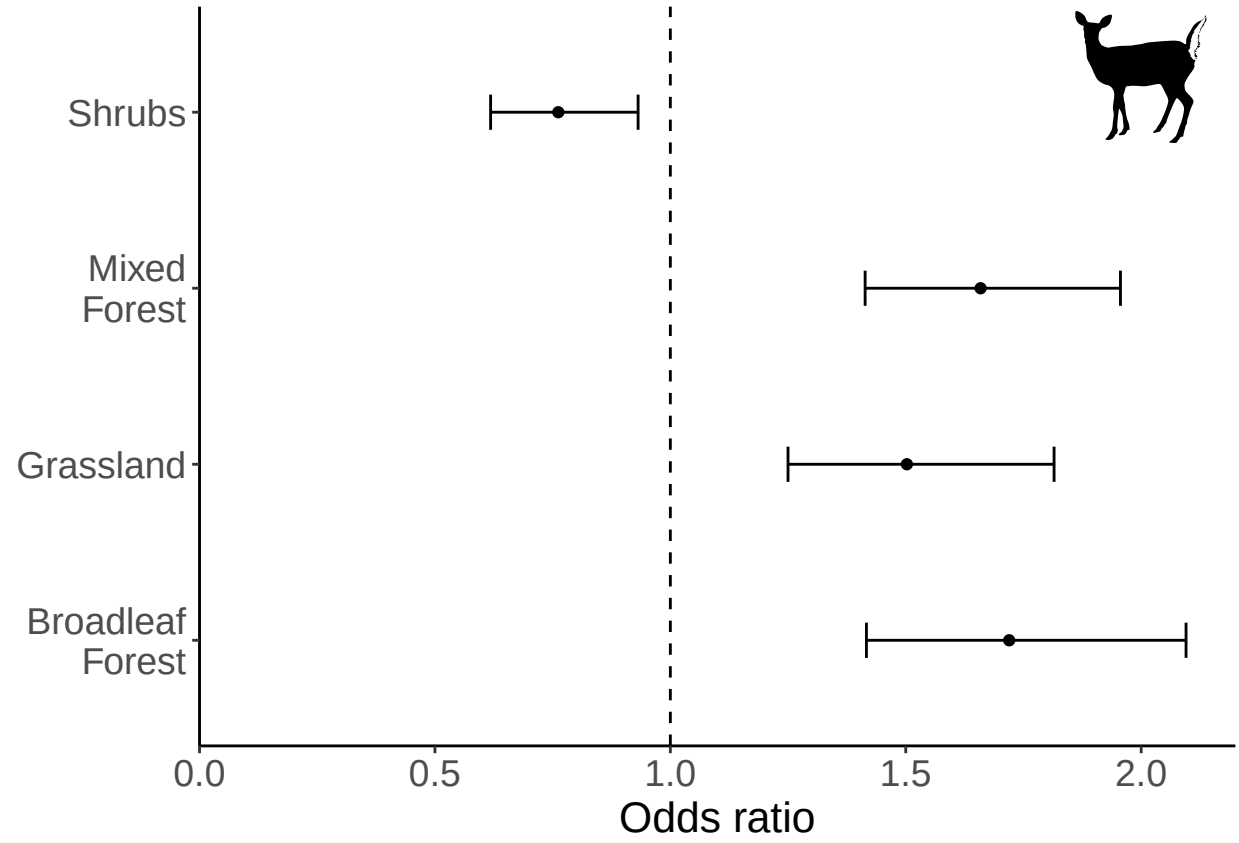
```
w_deer_model_odds <-

  # extract the coefficients and upper and lower CI
  tidy(w_deer_nat, conf.int = TRUE) %>%

  # convert the coefficients into odds ratios
  mutate(estimate = exp(estimate),
         conf.low = exp(conf.low),
         conf.high = exp(conf.high)) %>%

  # rename columns
  rename('lower' = conf.low,
         'upper' = conf.high) %>%

  # Remove intercept for plotting
filter(term != '(Intercept)')
```

Get silhouette for plots

```
w_deer_img <- get_phylopic('6038e80c-398d-47b2-9a69-2b9edf436f64')
```

Plot odds

```
# name to save plot later
w_deer_odds_plot <-
# provide data and mapping aesthetics
ggplot(w_deer_model_odds, aes(x = term,
                              y = estimate)) +

  geom_point() +

   geom_errorbar(aes(ymin = lower,
                     ymax = upper),
                 width = 0.2,
                 linewidth = 0.5,
                 position = position_dodge(width = 0.9)) +

   geom_hline(yintercept = 1, linetype = "dashed") +

  labs(y = 'Odds ratio') +

    scale_x_discrete(labels = ~ str_wrap(as.character(c('Broadleaf Forest',
                                'Grassland',
                                'Mixed Forest',
                                'Shrubs')),
                                        9)) +

  # scale_x_discrete(labels = c('Broadleaf Forest',
  #                             'Grassland',
  #                             'Mixed Forest',
  #                             'Shrubs')) +

  scale_y_continuous(expand = expansion(mult = c(0, 0.05)),
                     limits = c(0, NA)) +

  coord_flip() +

  add_phylopic(w_deer_img,
               x = 4.2,
               y = 2,
               ysize = 0.75) +

  theme_classic() +

  theme(axis.title.y = element_blank(),
        axis.title = element_text(size = 16),
        axis.text = element_text(size = 14))

w_deer_odds_plot
```



Save plot

```
ggsave('figures/odds_plot_whitetaileddeer.jpg',
       w_deer_odds_plot,
       width = 10,
       height = 8,
       units = 'in',
       dpi = 600)
```

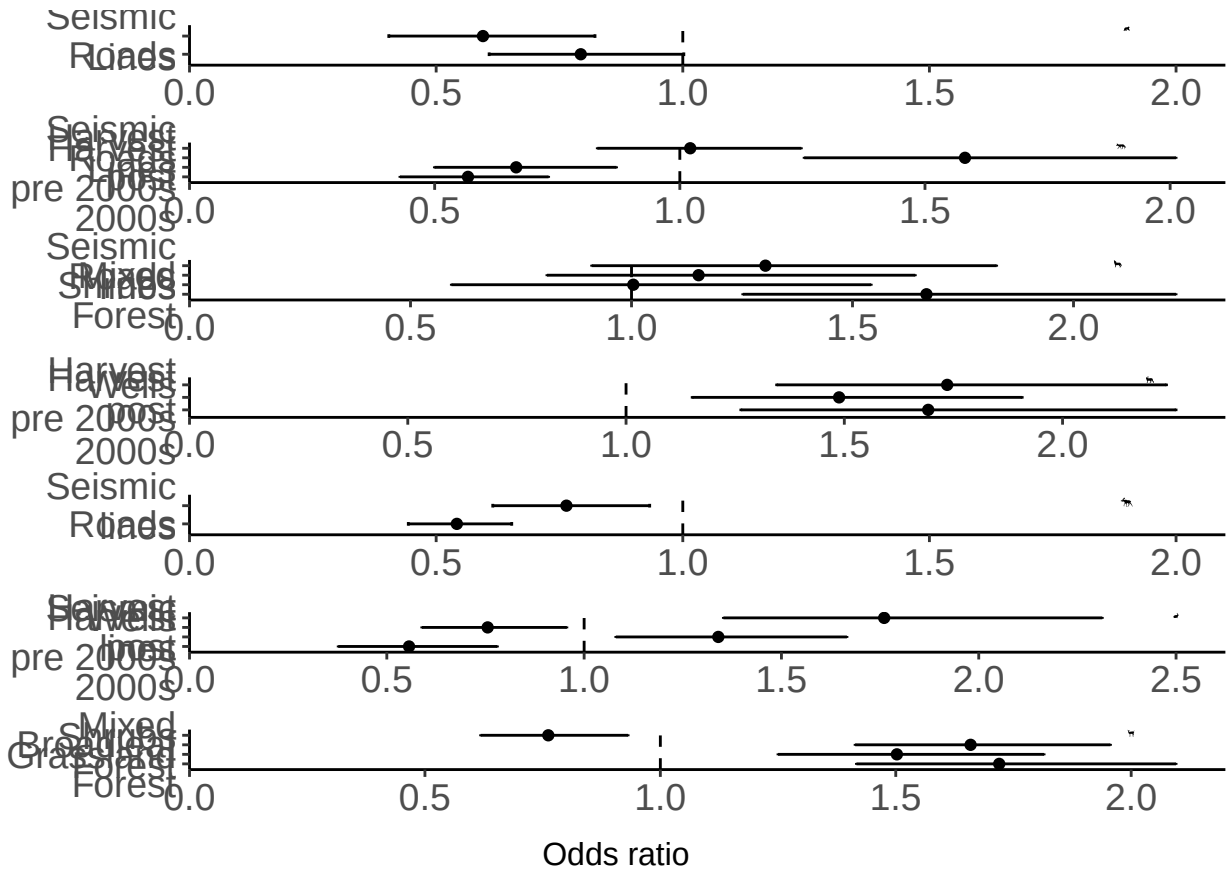
Combined odds plot

First remove the axis from the plots

```
# List of plots
odds_plots <- list (bbear_odds_plot,
          coyote_odds_plot,
          grey_wolf_odds_plot,
          lynx_odds_plot,
          moose_odds_plot,
          hare_odds_plot,
          w_deer_odds_plot)

# Remove the x-axis from each plot using purrr::map
odds_plots_no_xaxis <- odds_plots %>%

  map( ~ .x
       + theme(axis.title.x = element_blank()))
```

Then arrange as one plot

```
figure_2 <-
  ggarrange(plotlist = odds_plots_no_xaxis,

            ncol = 1,
            nrow = 7,
            align = 'hv')


figure_2 <-  annotate_figure(figure_2,
                             bottom = text_grob('Odds ratio'))

figure_2
```



Save combined plot

```
ggsave('figures/publication_figures/figure_2_odds.jpg',
       figure_2,
       width = 10,
       height = 20,
       units = 'in',
       dpi = 600)
```

Predictive plots

Black bear

Data

First we need to extract predicted probabilities of species occurrence for each covariate of interest in the best fit model, and generate a tibble of that data that we can plot to interpret how each variable influences species occurrence

The handy ggpredict function will do this for us, I'm combining it with purrr to iterate the function over all fixed effects so I don't have to copy and paste code for every variable in the model

```
# supply vector of fixed effects as they appear in model
bbear_fes <- c('roads',
               'seismic_lines')

# Use purrr to iterate ggpredict and rename the list elements
bbear_predicted_data <- map(set_names(bbear_fes), ~ {
  ggpredict(bbear_linear, terms = paste0(.x, "[all]"))
})

# I viewed the full data from my environemnt but here's a printout of one variable
head(bbear_predicted_data$roads)
```

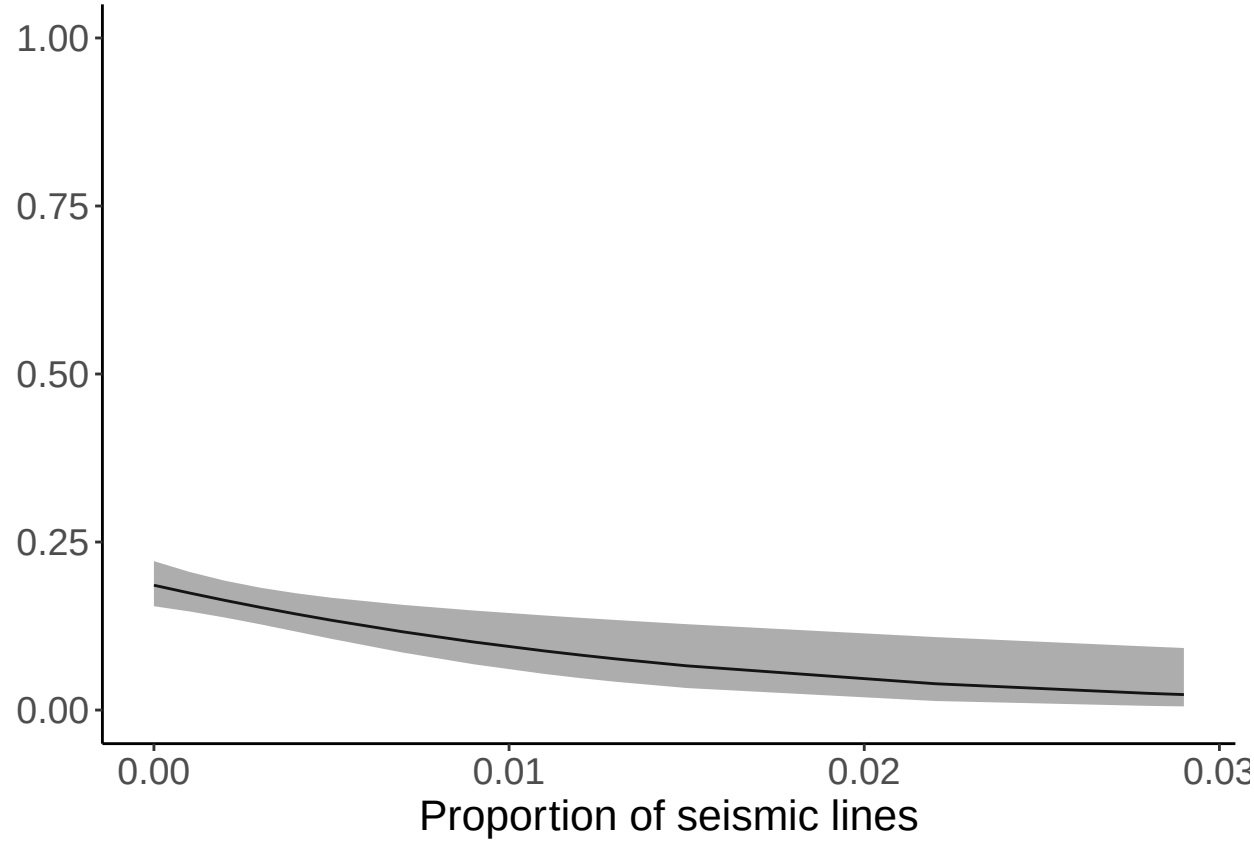
```
## # Predicted probabilities of cbind(black_bear, absent_black_bear)
##
##    roads | Predicted |     95% CI
## ---------------------------------
##     0.00 |      0.17 | 0.14, 0.20
## 1.00e-03 |      0.17 | 0.14, 0.20
## 2.00e-03 |      0.16 | 0.14, 0.20
## 4.00e-03 |      0.16 | 0.13, 0.19
## 5.00e-03 |      0.16 | 0.13, 0.19
## 7.00e-03 |      0.15 | 0.13, 0.18
##
## Adjusted for:
## * seismic_lines = 0.00
```

And then we plot with ggplot, starting with seismic lines

```
# and now plot with ggplot

# name plot and assign to environment
bbear_seismic_plot <-

  # provide data from ggpredict with x and y
  ggplot(bbear_predicted_data$seismic_lines,
         aes(x = x,
             y = predicted)) +

  # plot relationship with line
  geom_line() +

  # plot confidence with geom ribbon, must supply new aesthetics
  geom_ribbon(aes(ymin = conf.low,
                  ymax = conf.high),
```

```
              alpha = 0.4) +

  # set the axis limits so all plots go from 0-1
  scale_y_continuous(limits = c(0, 1)) +

  # label axis
  labs(x = 'Proportion of seismic lines',
       y = 'Predicted probabiity of occurrence') +

  # specify overall theme
  theme_classic() +

  # change font size
  theme(axis.title = element_text(size = 16),
        axis.text = element_text(size = 14),
        axis.title.y = element_blank())

   # # add silhouette
   # add_phylopic(uuid = get_uuid(name = 'Ursus americanus'),
   #              x = 0.028,
   #              y = 0.9,
   #              ysize = 0.2)
```

```
bbear_seismic_plot
```

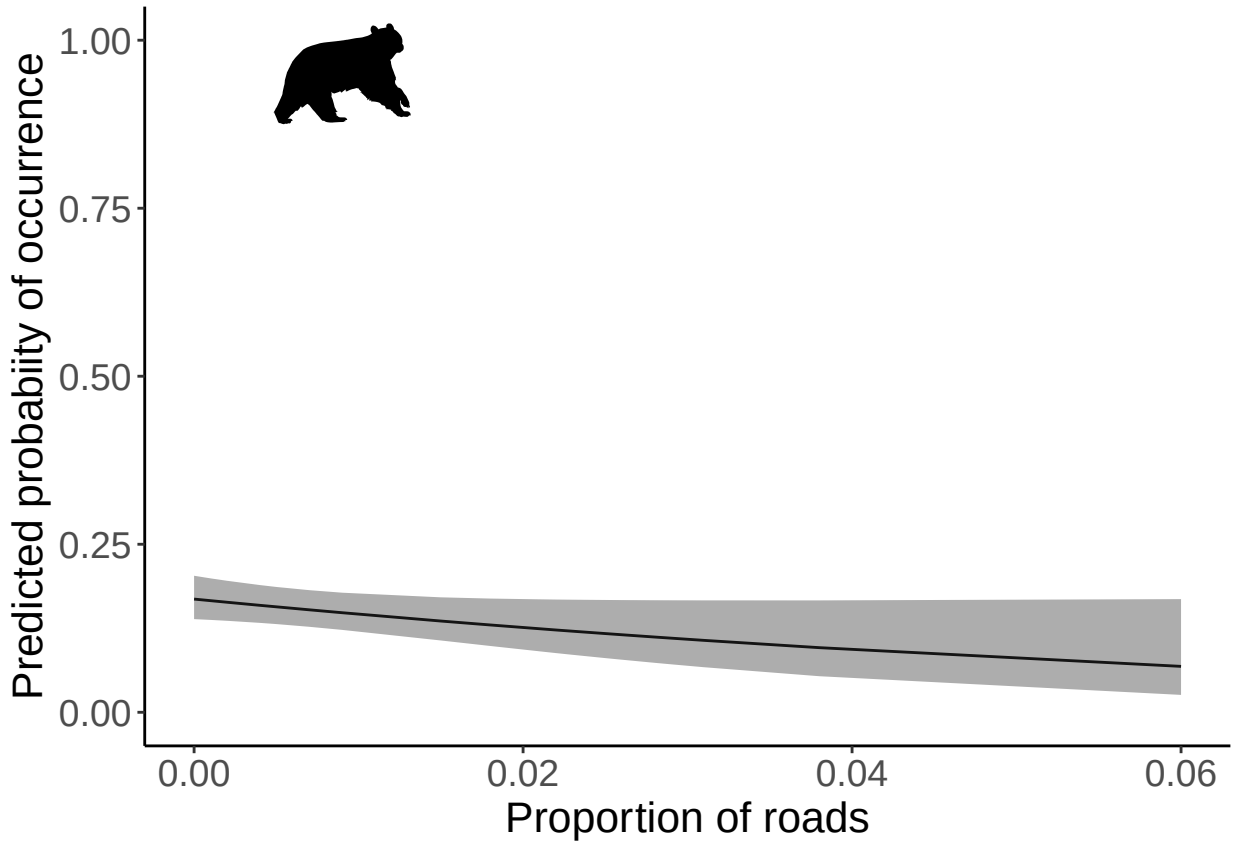


Save plot

```r
ggsave('figures/black_bear_seismic_lines_plot.jpg',
       bbear_seismic_plot,
       width = 14,
       height = 10,
       units = 'in',
       dpi = 600)
```

And now repeat for roads, I didn't bother creating a function or iteration for this because given the labs need to have nice names and the sillhouettes may need to be placed in different spots given the x axis for each variable this was simmpler for now

```r
# name plot and assign to environment
bbear_rds_plot <-

  # provide data from ggpredict with x and y
  ggplot(bbear_predicted_data$roads,
         aes(x = x,
             y = predicted)) +

  # plot relationship with line
  geom_line() +

  # plot confidence with geom ribbon, must supply new aesthetics
  geom_ribbon(aes(ymin = conf.low,
                  ymax = conf.high),
              alpha = 0.4) +

  # set the axis limits so all plots go from 0-1
  scale_y_continuous(limits = c(0, 1)) +

  # label axis
  labs(x = 'Proportion of roads',
       y = 'Predicted probabiity of occurrence') +

  # specify overall theme
  theme_classic() +

  # change font size
  theme(axis.title = element_text(size = 16),
        axis.text = element_text(size = 14)) +

   # add silhouette
   add_phylopic(uuid = get_uuid(name = 'Ursus americanus'),
                x = 0.009,
                y = 0.95,
                ysize = 0.15)

bbear_rds_plot
```



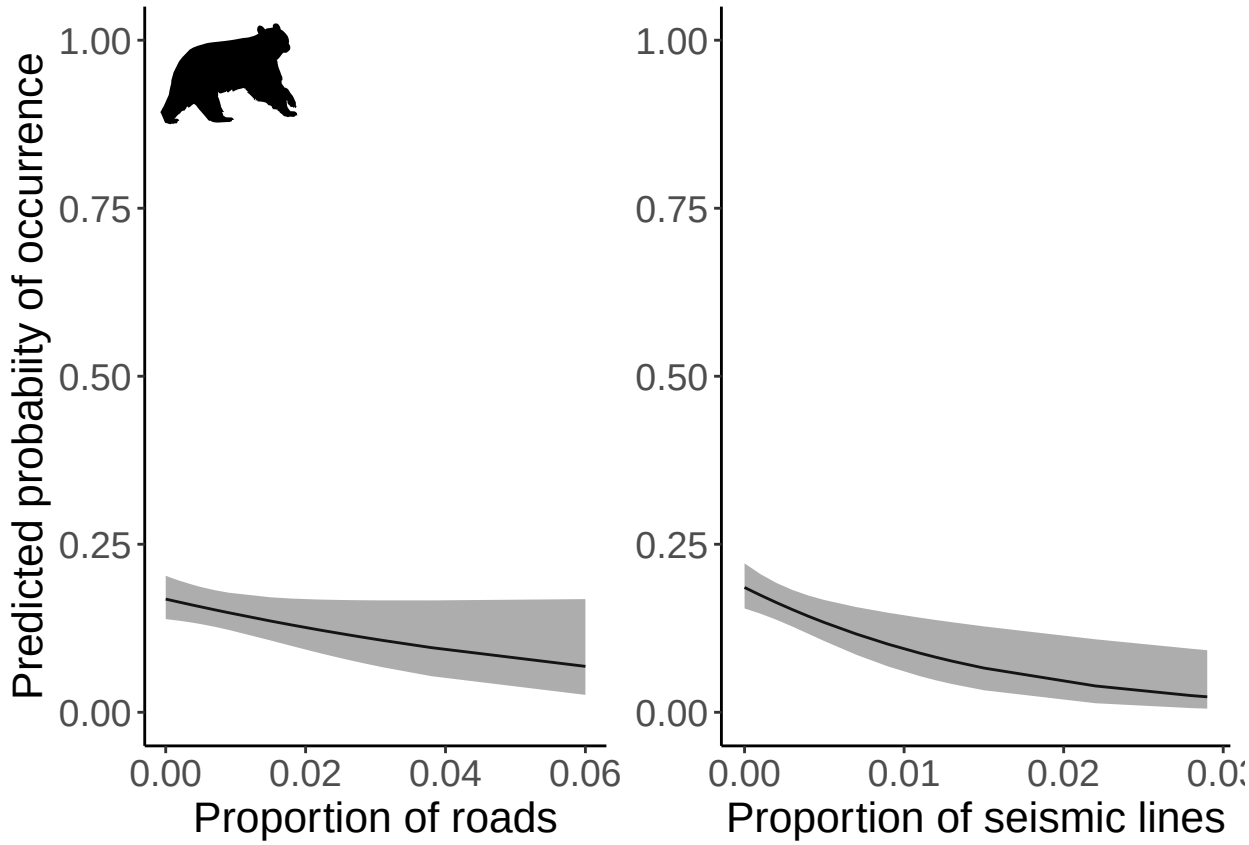
Save plot

```
ggsave('figures/black_bear_roads_plot.jpg',
       bbear_rds_plot,
       width = 14,
       height = 10,
       units = 'in',
       dpi = 600)
```

Join plots

We want multiple panels in the same figure for each species. let's join all the predictive plots for black bears into one figure

```
# join plots
bbear_plot <- ggarrange(bbear_rds_plot,
                        bbear_seismic_plot)

# view
bbear_plot
```



Save final figure

```
ggsave('figures/publication_figures/bbear_plot.jpg',
       bbear_plot,
        width = 10,
       height = 8,
       units = 'in',
       dpi = 600)
```

Let's repeat this process for each species that we have enough data for.

Coyote

Data

First extract predicted data for each covariate in model using purrr below

```
# supply vector of fixed effects as they appear in model
coyote_fes <- c('roads',
                'seismic_lines',
                'harvest_2000',
                'harvest_pre2000')

# Use purrr to iterate ggpredict and rename the list elements
coyote_predicted_data <- map(set_names(coyote_fes), ~ {
  ggpredict(coyote_disturb, terms = paste0(.x, "[all]"))
})
```

```r
# I viewed the full data from my environemnt but here's a printout of one variable
head(coyote_predicted_data$roads)
```

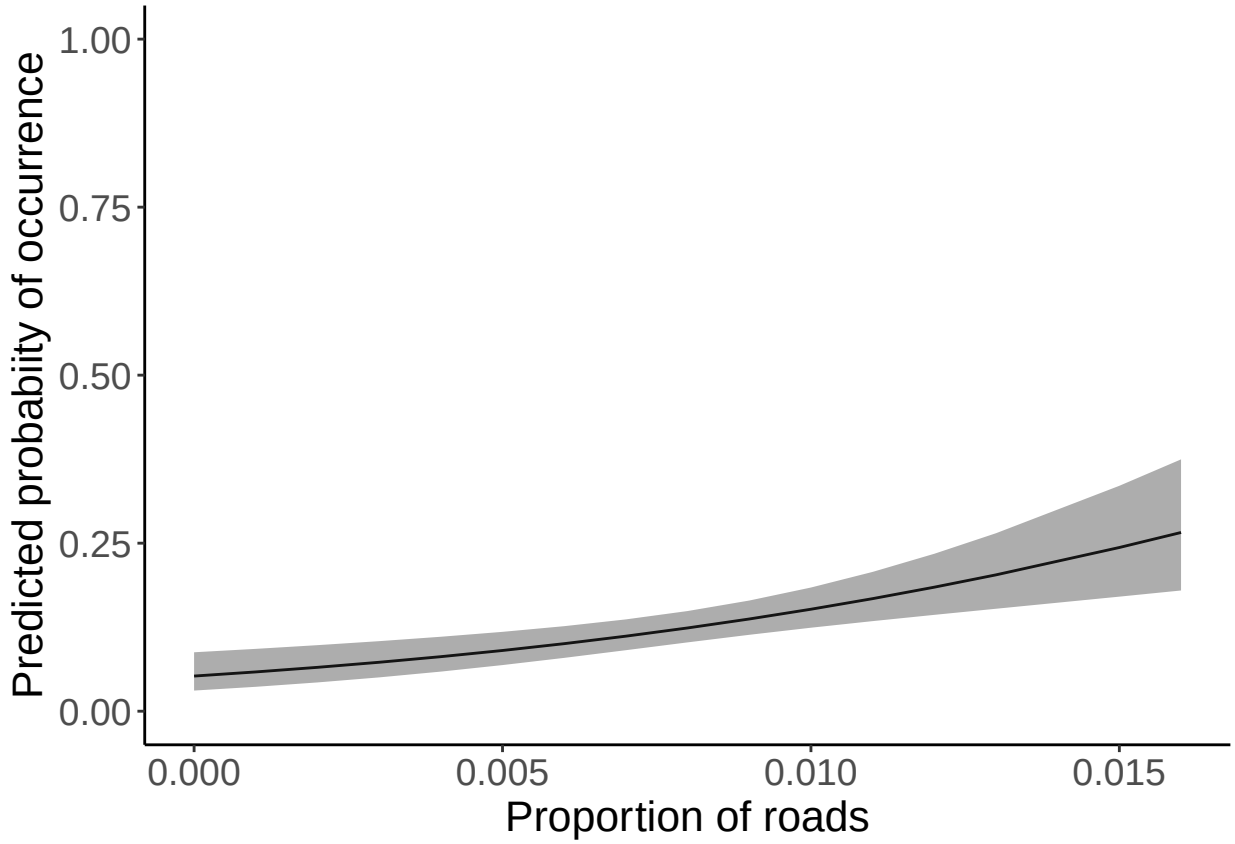
```
## # Predicted probabilities of cbind(coyote, absent_coyote)
##
##    roads | Predicted |     95% CI
## --------------------------------
##     0.00 |      0.05 | 0.03, 0.09
## 1.00e-03 |      0.06 | 0.04, 0.09
## 2.00e-03 |      0.07 | 0.04, 0.10
## 3.00e-03 |      0.07 | 0.05, 0.10
## 4.00e-03 |      0.08 | 0.06, 0.11
## 5.00e-03 |      0.09 | 0.07, 0.12
##
## Adjusted for:
## *    harvest_2000 = 0.15
## * harvest_pre2000 = 0.09
## *   seismic_lines = 0.00
```

Now plot each one individually

Roads

```r
# name plot and assign to environment
coyote_rds_plot <-

  # provide data from ggpredict with x and y
  ggplot(coyote_predicted_data$roads,
         aes(x = x,
             y = predicted)) +

  # plot relationship with line
  geom_line() +

  # plot confidence with geom ribbon, must supply new aesthetics
  geom_ribbon(aes(ymin = conf.low,
                  ymax = conf.high),
              alpha = 0.4) +

  # set the axis limits so all plots go from 0-1
  scale_y_continuous(limits = c(0, 1)) +

  # label axis
  labs(x = 'Proportion of roads',
       y = 'Predicted probabiity of occurrence') +

  # specify overall theme
  theme_classic() +

  # change font size
  theme(axis.title = element_text(size = 16),
        axis.text = element_text(size = 14))

   # # add silhouette
```

```
  # add_phylopic(uuid = 'd451e353-585a-4543-84e7-7ef2f90aa407',
  #              x = 0.015,
  #              y = 0.9,
  #              ysize = 0.2)

coyote_rds_plot
```



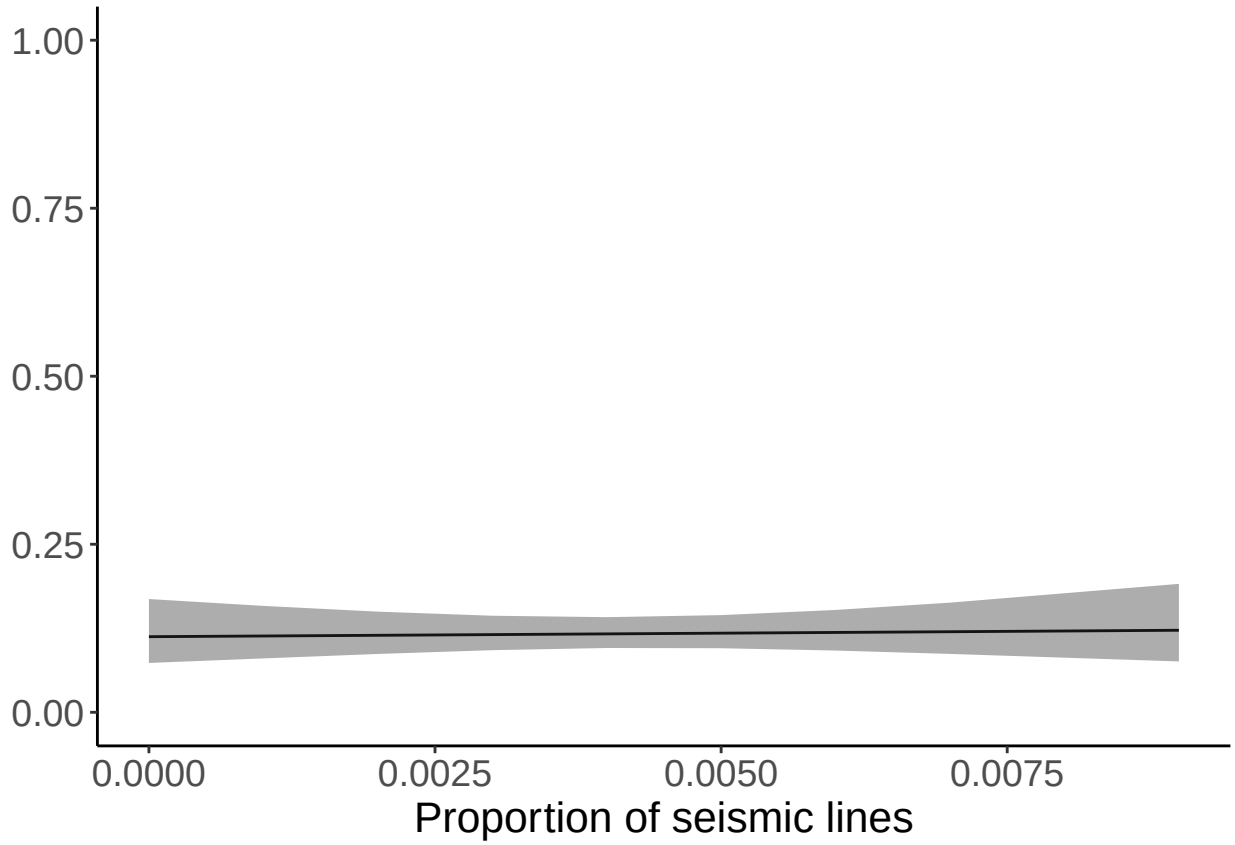
Save plot

```
ggsave('figures/coyote_roads_plot.jpg',
       coyote_rds_plot,
       width = 14,
       height = 10,
       units = 'in',
       dpi = 600)
```

Seismic lines

```
# name plot and assign to environment
coyote_seismic_plot <-

  # provide data from ggpredict with x and y
  ggplot(coyote_predicted_data$seismic_lines,
         aes(x = x,
             y = predicted)) +
```

```
  # plot relationship with line
  geom_line() +

  # plot confidence with geom ribbon, must supply new aesthetics
  geom_ribbon(aes(ymin = conf.low,
                  ymax = conf.high),
              alpha = 0.4) +

  # set the axis limits so all plots go from 0-1
  scale_y_continuous(limits = c(0, 1)) +

  # label axis
  labs(x = 'Proportion of seismic lines',
       y = 'Predicted probabiity of occurrence') +

  # specify overall theme
  theme_classic() +

  # change font size
  theme(axis.title = element_text(size = 16),
        axis.text = element_text(size = 14),
        axis.title.y = element_blank())

   # # add silhouette
   # add_phylopic(uuid = 'd451e353-585a-4543-84e7-7ef2f90aa407',
   #              x = 0.008,
   #              y = 0.9,
   #              ysize = 0.2)

coyote_seismic_plot
```



Save plot

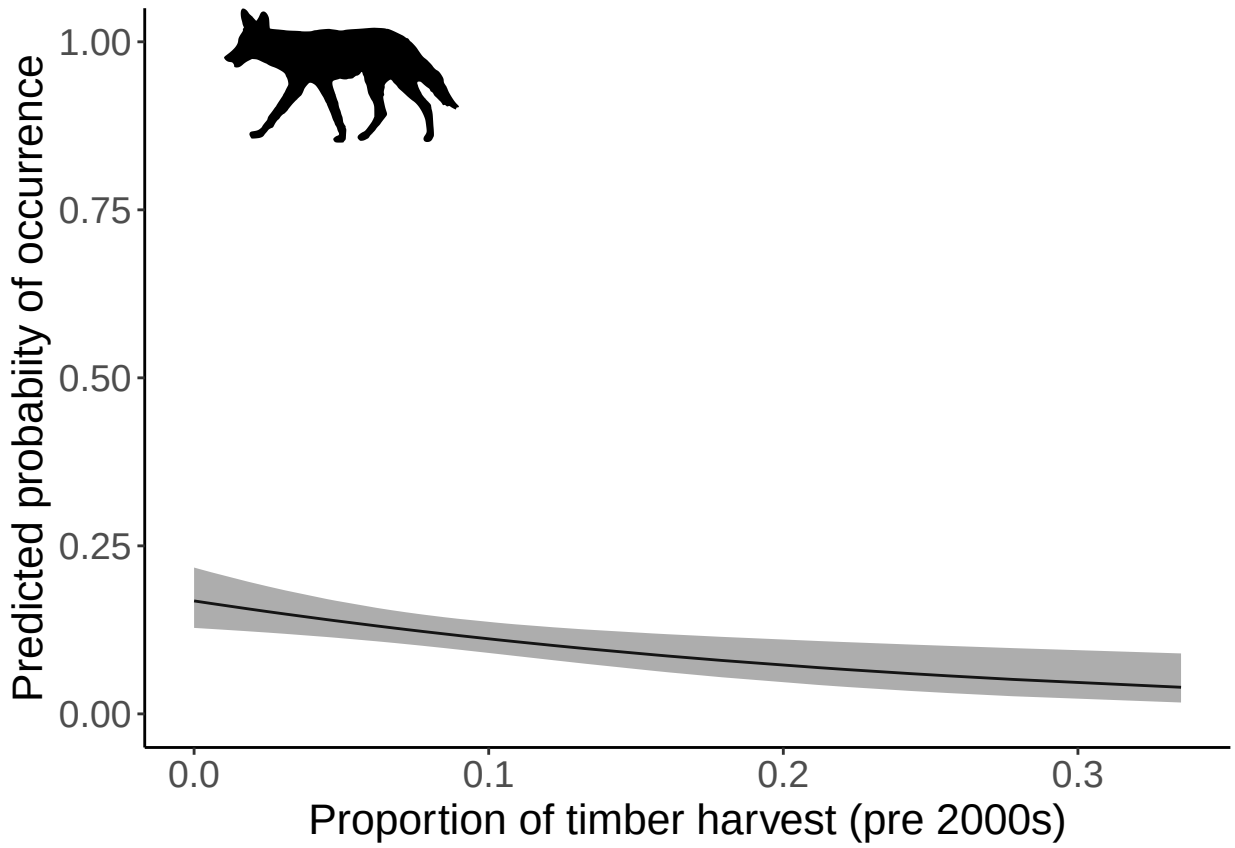
```
ggsave('figures/coyote_seismic_lines_plot.jpg',
       coyote_seismic_plot,
       width = 14,
       height = 10,
       units = 'in',
       dpi = 600)
```

Harvest pre 2000s

```
# name plot and assign to environment
coyote_harvest_pre_plot <-

  # provide data from ggpredict with x and y
  ggplot(coyote_predicted_data$harvest_pre2000,
         aes(x = x,
             y = predicted)) +

  # plot relationship with line
  geom_line() +

  # plot confidence with geom ribbon, must supply new aesthetics
  geom_ribbon(aes(ymin = conf.low,
                  ymax = conf.high),
              alpha = 0.4) +
```

```
  # set the axis limits so all plots go from 0-1
  scale_y_continuous(limits = c(0, 1)) +

  # label axis
  labs(x = 'Proportion of timber harvest (pre 2000s)',
       y = 'Predicted probabiity of occurrence') +

  # specify overall theme
  theme_classic() +

  # change font size
  theme(axis.title = element_text(size = 16),
        axis.text = element_text(size = 14)) +

   # add silhouette
   add_phylopic(uuid = 'd451e353-585a-4543-84e7-7ef2f90aa407',
                x = 0.05,
                y = 0.95,
                ysize = 0.2)
```

```
coyote_harvest_pre_plot
```



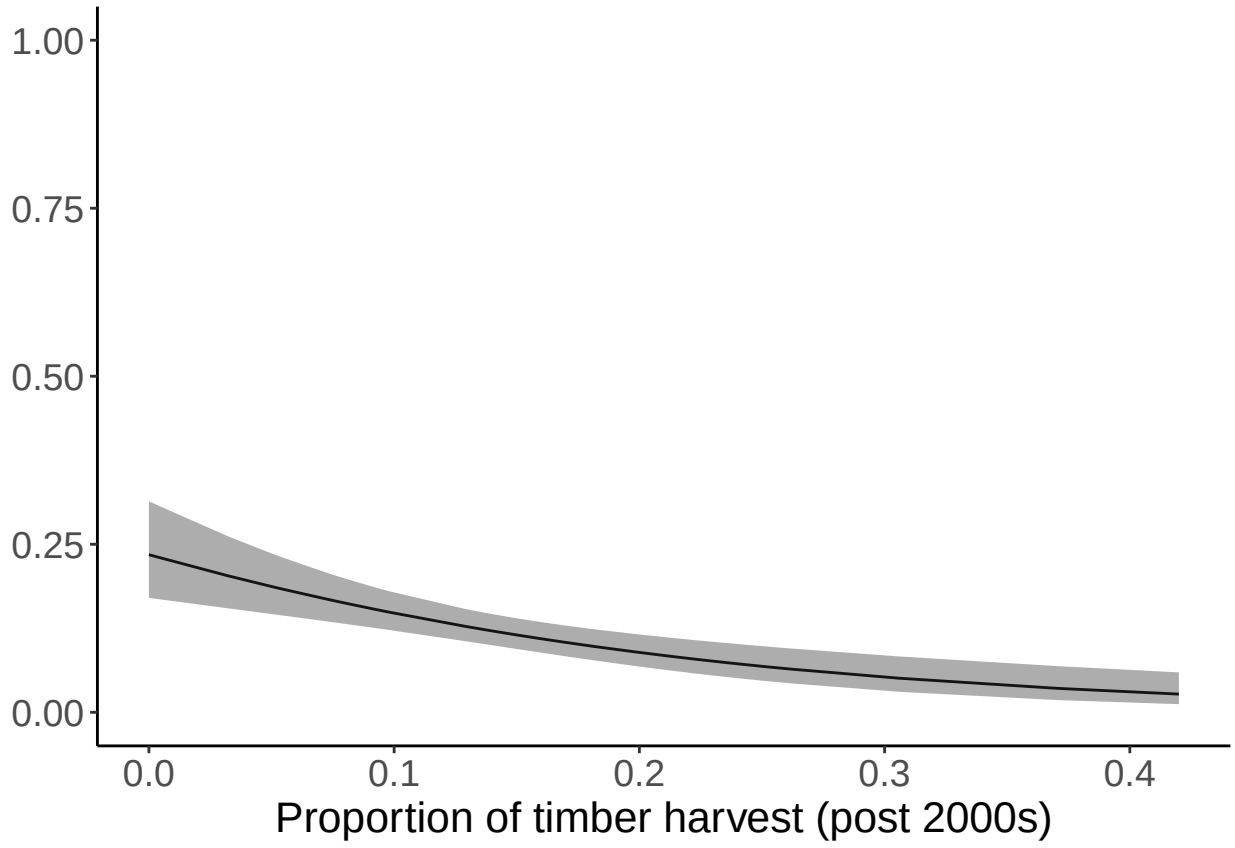
Save plot

```
ggsave('figures/coyote_harvest_pre2000_plot.jpg',
       coyote_harvest_pre_plot,
```

```
       width = 14,
       height = 10,
       units = 'in',
       dpi = 600)
```

Harvest post 2000

```
# name plot and assign to environment
coyote_harvest_post_plot <-

  # provide data from ggpredict with x and y
  ggplot(coyote_predicted_data$harvest_2000,
         aes(x = x,
             y = predicted)) +

  # plot relationship with line
  geom_line() +

  # plot confidence with geom ribbon, must supply new aesthetics
  geom_ribbon(aes(ymin = conf.low,
                  ymax = conf.high),
              alpha = 0.4) +

  # set the axis limits so all plots go from 0-1
  scale_y_continuous(limits = c(0, 1)) +

  # label axis
  labs(x = 'Proportion of timber harvest (post 2000s)',
       y = 'Predicted probabiity of occurrence') +

  # specify overall theme
  theme_classic() +

  # change font size
  theme(axis.title = element_text(size = 16),
        axis.text = element_text(size = 14),
        axis.title.y = element_blank())

   # # add silhouette
   # add_phylopic(uuid = 'd451e353-585a-4543-84e7-7ef2f90aa407',
   #              x = 0.38,
   #              y = 0.9,
   #              ysize = 0.2)

coyote_harvest_post_plot
```



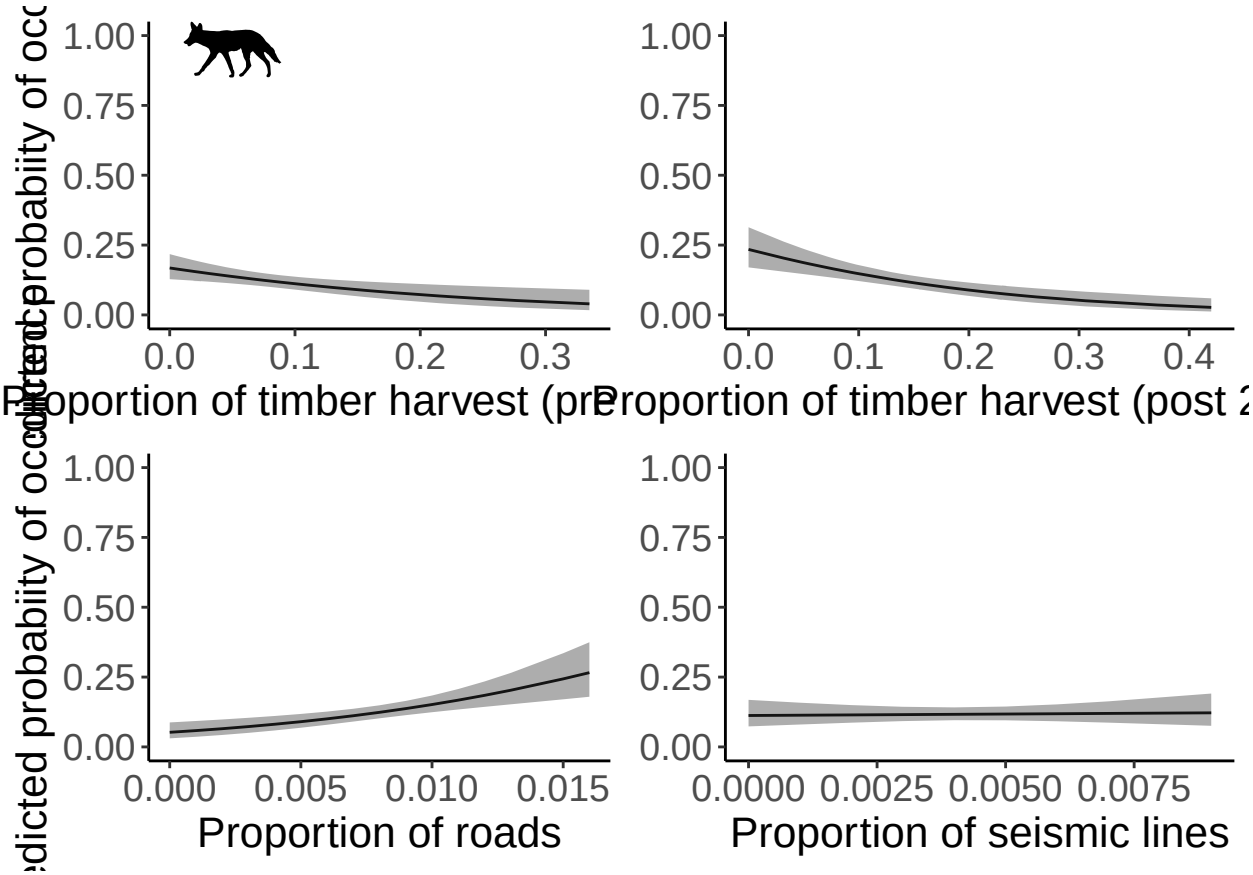
Save plot

```
ggsave('figures/coyote_harvest_post2000_plot.jpg',
       coyote_harvest_post_plot,
        width = 14,
        height = 10,
        units = 'in',
        dpi = 600)
```

Join plots

We want multiple panels in the same figure for each species. let's join all the predictive plots for black bears into one figure

```
# join plots
coyote_plot <- ggarrange(coyote_harvest_pre_plot,
                         coyote_harvest_post_plot,
                         coyote_rds_plot,
                         coyote_seismic_plot,

                         nrow = 2,
                         ncol = 2)

# view
coyote_plot
```



Save final figure

```
ggsave('figures/publication_figures/coyote_plot.jpg',
     coyote_plot,
        width = 10,
       height = 8,
       units = 'in',
       dpi = 600)
```

Grey wolf

Data

Use code from above to get pred data for each variable

```
# supply vector of fixed effects as they appear in model
wolf_fes <- c('roads',
              'seismic_lines',
              'lc_shrub',
              'lc_mixed')

# Use purrr to iterate ggpredict and rename the list elements
wolf_predicted_data <- map(set_names(wolf_fes), ~ {
  ggpredict(grey_wolf_linear_nat, terms = paste0(.x, "[all]"))
})
```

```r
# I viewed the full data from my environment but here's a printout of one variable
head(wolf_predicted_data$roads)
```

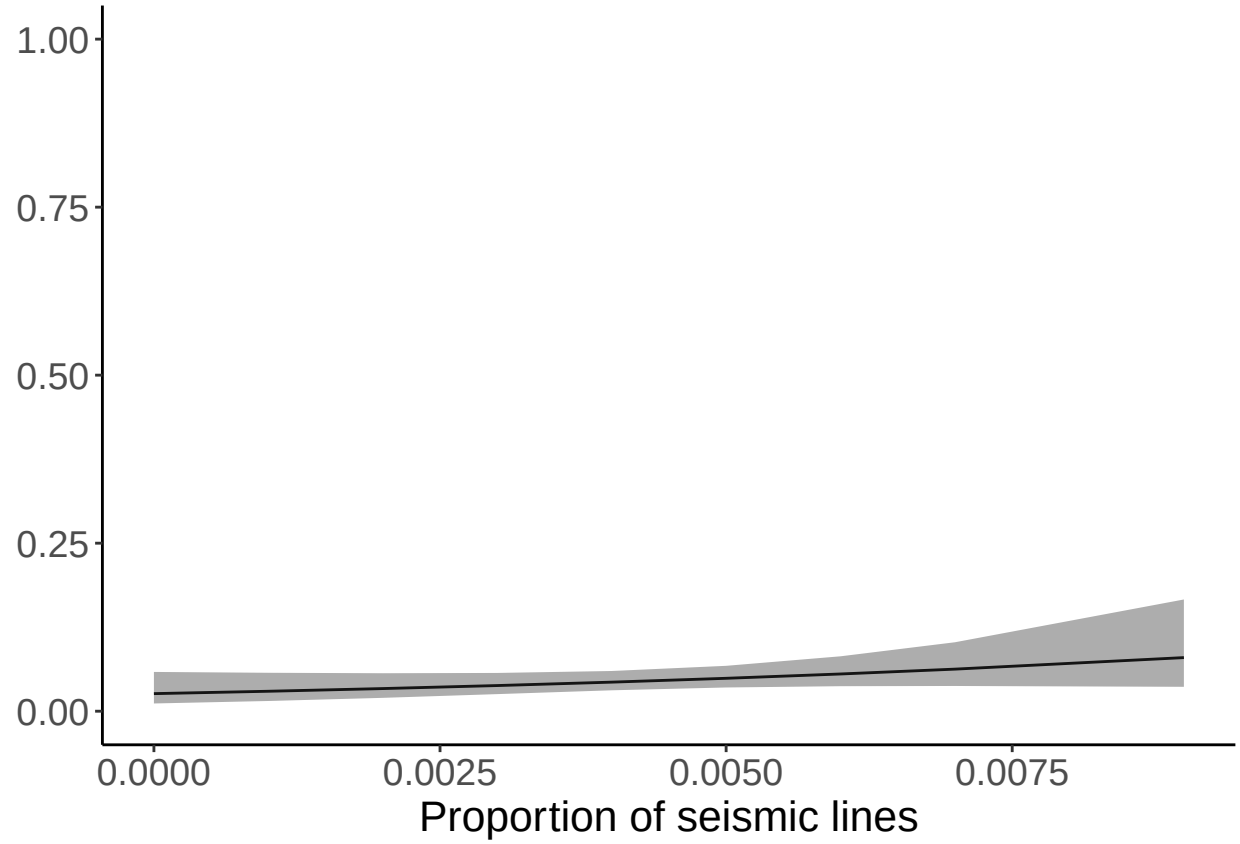
```
## # Predicted probabilities of cbind(grey_wolf, absent_grey_wolf)
##
##    roads | Predicted |     95% CI
## --------------------------------
##     0.00 |      0.03 | 0.02, 0.07
## 1.00e-03 |      0.03 | 0.02, 0.07
## 2.00e-03 |      0.04 | 0.02, 0.07
## 3.00e-03 |      0.04 | 0.02, 0.06
## 4.00e-03 |      0.04 | 0.02, 0.06
## 5.00e-03 |      0.04 | 0.03, 0.06
##
## Adjusted for:
## * seismic_lines = 0.00
## *      lc_mixed = 0.04
## *      lc_shrub = 0.16
```

Seismic lines

Now plot results for each fixed effect starting with seismic lines

```r
# name plot and assign to environment
wolf_seismic_plot <-

  # provide data from ggpredict with x and y
  ggplot(wolf_predicted_data$seismic_lines,
         aes(x = x,
             y = predicted)) +

  # plot relationship with line
  geom_line() +

  # plot confidence with geom ribbon, must supply new aesthetics
  geom_ribbon(aes(ymin = conf.low,
                  ymax = conf.high),
              alpha = 0.4) +

  # set the axis limits so all plots go from 0-1
  scale_y_continuous(limits = c(0, 1)) +

  # label axis
  labs(x = 'Proportion of seismic lines',
       y = 'Predicted probabiity of occurrence') +

  # specify overall theme
  theme_classic() +

  # change font size
  theme(axis.title = element_text(size = 16),
        axis.text = element_text(size = 14),
        axis.title.y = element_blank())

   # # add silhouette
```

```
  # add_phylopic(uuid = 'e4e306cd-73b6-4ca3-a08c-753a856f7f12',
  #              x = 0.0085,
  #              y = 0.9,
  #              ysize = 0.2)

wolf_seismic_plot
```



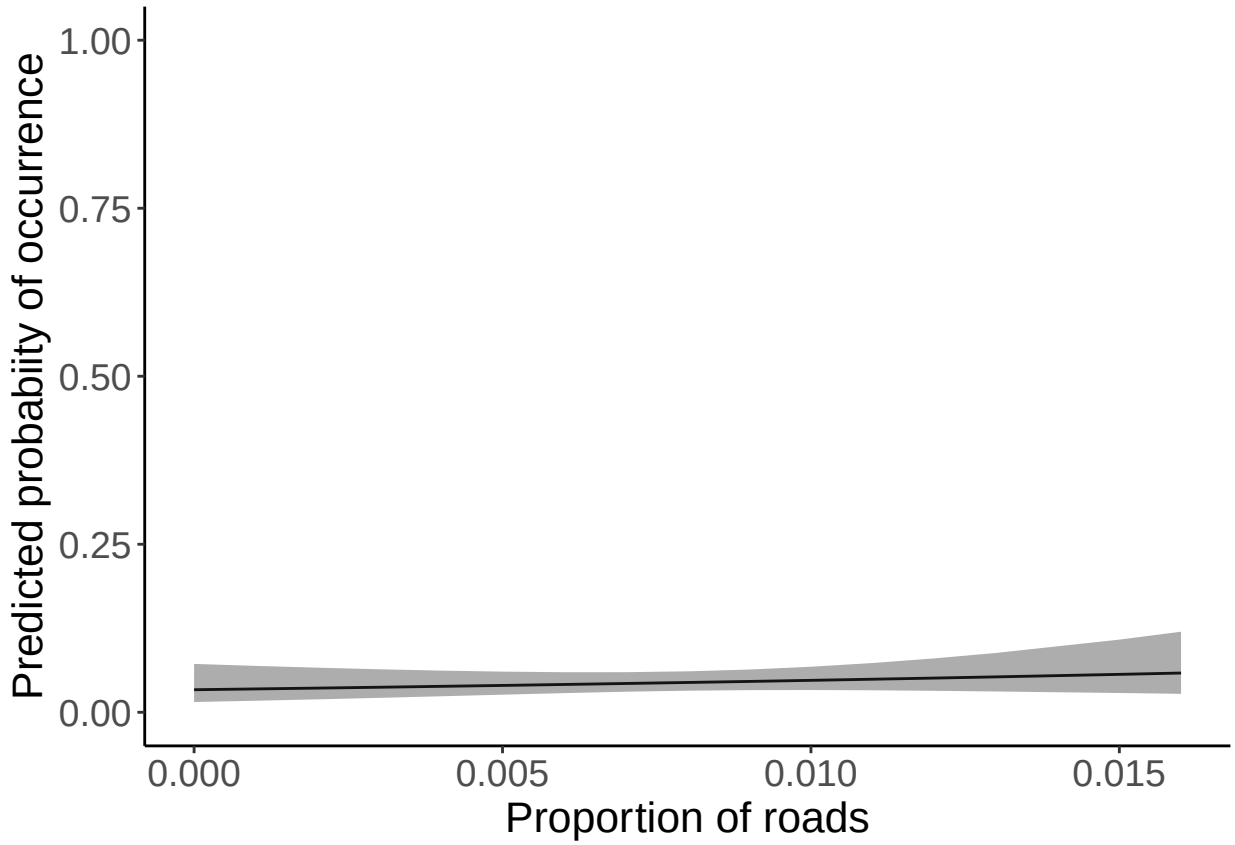
Save plot

```
ggsave('figures/grey_wolf_seismic_lines_plot.jpg',
       wolf_seismic_plot,
       width = 14,
       height = 10,
       units = 'in',
       dpi = 600)
```

Roads

```
# name plot and assign to environment
wolf_rds_plot <-

  # provide data from ggpredict with x and y
  ggplot(wolf_predicted_data$roads,
         aes(x = x,
             y = predicted)) +

  # plot relationship with line
```

```
  geom_line() +

  # plot confidence with geom ribbon, must supply new aesthetics
  geom_ribbon(aes(ymin = conf.low,
                  ymax = conf.high),
              alpha = 0.4) +

  # set the axis limits so all plots go from 0-1
  scale_y_continuous(limits = c(0, 1)) +

  # label axis
  labs(x = 'Proportion of roads',
       y = 'Predicted probabiity of occurrence') +

  # specify overall theme
  theme_classic() +

  # change font size
  theme(axis.title = element_text(size = 16),
        axis.text = element_text(size = 14))

   # # add silhouette
   # add_phylopic(uuid = 'e4e306cd-73b6-4ca3-a08c-753a856f7f12',
   #              x = 0.015,
   #              y = 0.9,
   #              ysize = 0.2)

wolf_rds_plot
```



Save plot

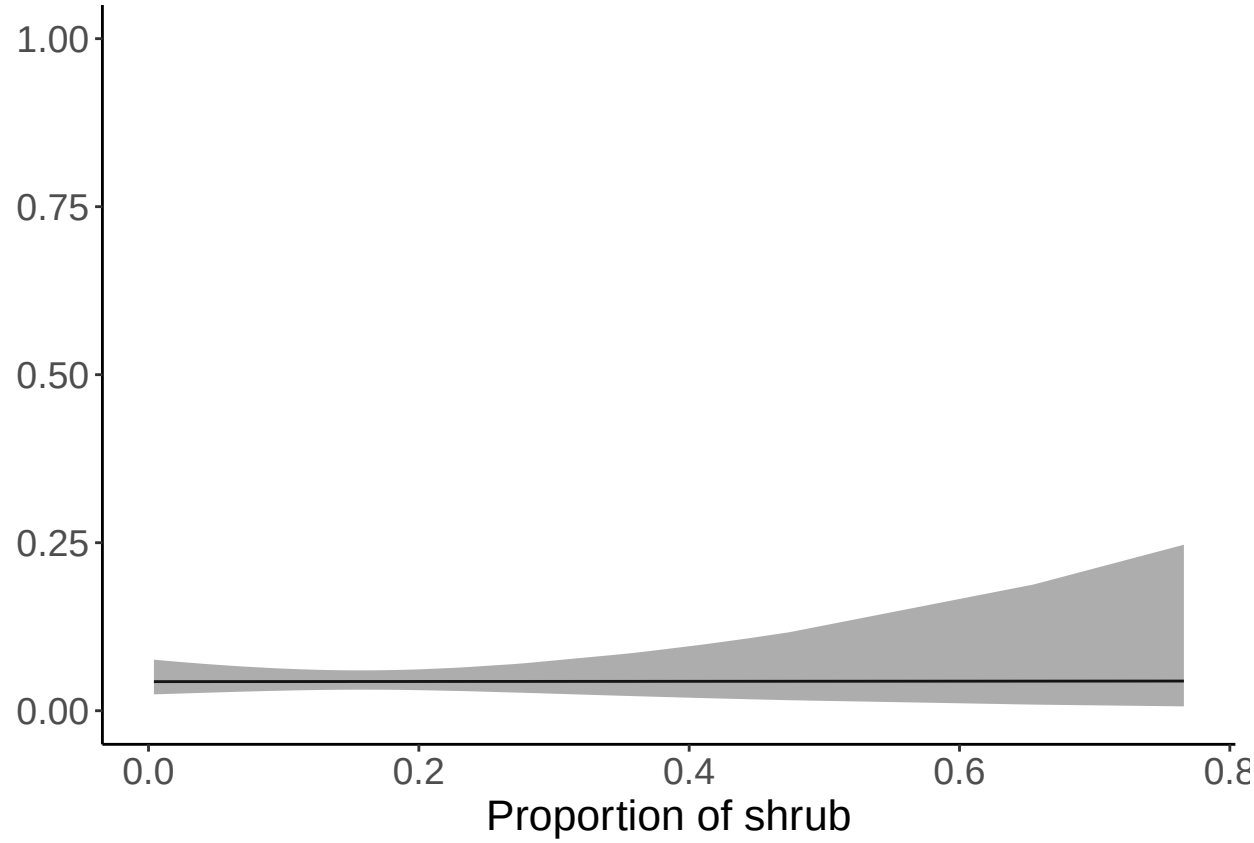
```
ggsave('figures/grey_wolf_roads_plot.jpg',
       wolf_rds_plot,
       width = 14,
       height = 10,
       units = 'in',
       dpi = 600)
```

Shrubs

```
# name plot and assign to environment
wolf_shrub_plot <-

  # provide data from ggpredict with x and y
  ggplot(wolf_predicted_data$lc_shrub,
         aes(x = x,
             y = predicted)) +

  # plot relationship with line
  geom_line() +

  # plot confidence with geom ribbon, must supply new aesthetics
  geom_ribbon(aes(ymin = conf.low,
                  ymax = conf.high),
              alpha = 0.4) +
```

```
  # set the axis limits so all plots go from 0-1
  scale_y_continuous(limits = c(0, 1)) +

  # label axis
  labs(x = 'Proportion of shrub',
       y = 'Predicted probabiity of occurrence') +

  # specify overall theme
  theme_classic() +

  # change font size
  theme(axis.title = element_text(size = 16),
        axis.text = element_text(size = 14),
        axis.title.y = element_blank())

   # add silhouette
   # add_phylopic(uuid = 'e4e306cd-73b6-4ca3-a08c-753a856f7f12',
   #              x = 0.7,
   #              y = 0.9,
   #              ysize = 0.2)
```

```
wolf_shrub_plot
```

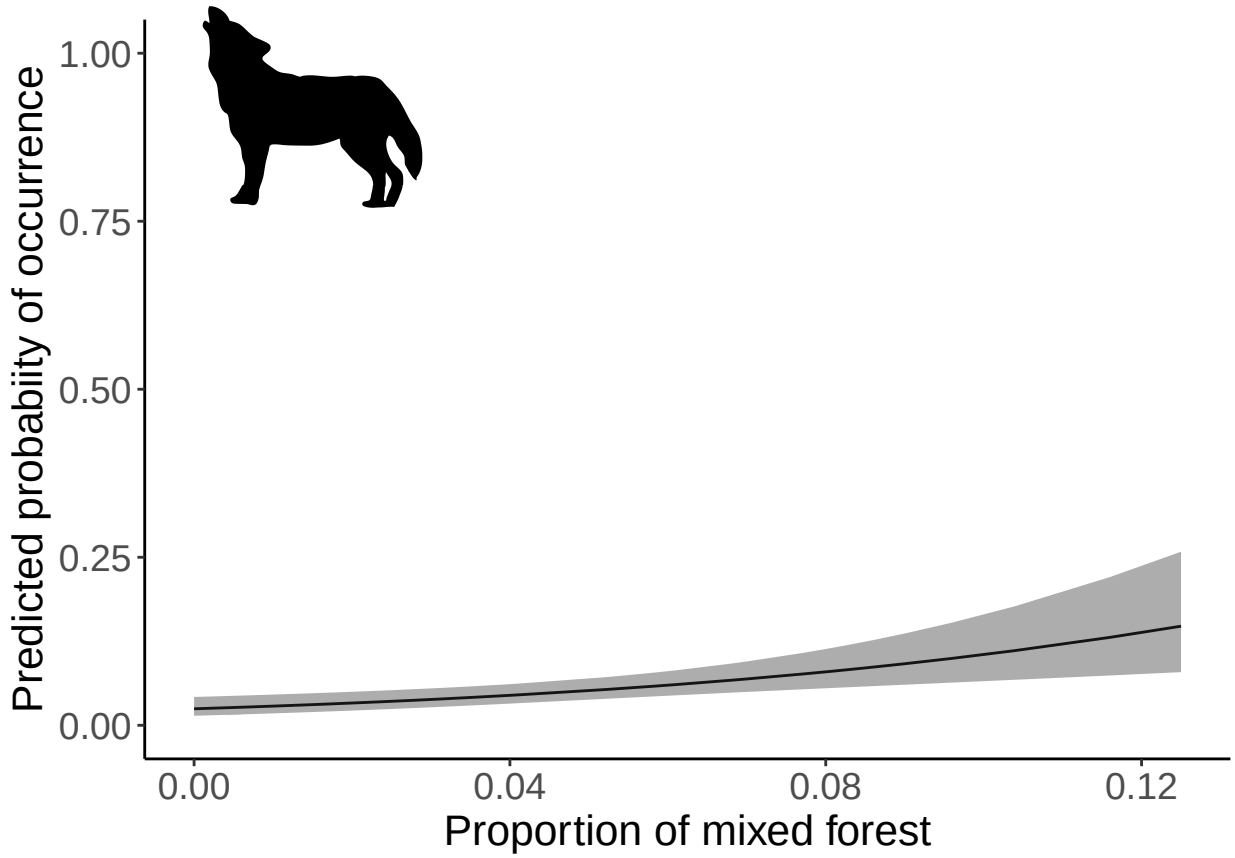


Save plot

```r
ggsave('figures/grey_wolf_shrub_plot.jpg',
       wolf_shrub_plot,
       width = 14,
       height = 10,
       units = 'in',
       dpi = 600)
```

Mixed forest

```r
# name plot and assign to environment
wolf_mixed_plot <-

  # provide data from ggpredict with x and y
  ggplot(wolf_predicted_data$lc_mixed,
         aes(x = x,
             y = predicted)) +

  # plot relationship with line
  geom_line() +

  # plot confidence with geom ribbon, must supply new aesthetics
  geom_ribbon(aes(ymin = conf.low,
                  ymax = conf.high),
              alpha = 0.4) +

  # set the axis limits so all plots go from 0-1
  scale_y_continuous(limits = c(0, 1)) +

  # label axis
  labs(x = 'Proportion of mixed forest',
       y = 'Predicted probabiity of occurrence') +

  # specify overall theme
  theme_classic() +

  # change font size
  theme(axis.title = element_text(size = 16),
        axis.text = element_text(size = 14)) +

   # add silhouette
   add_phylopic(uuid = 'e4e306cd-73b6-4ca3-a08c-753a856f7f12',
                x = 0.015,
                y = 0.92,
                ysize = 0.3)

wolf_mixed_plot
```

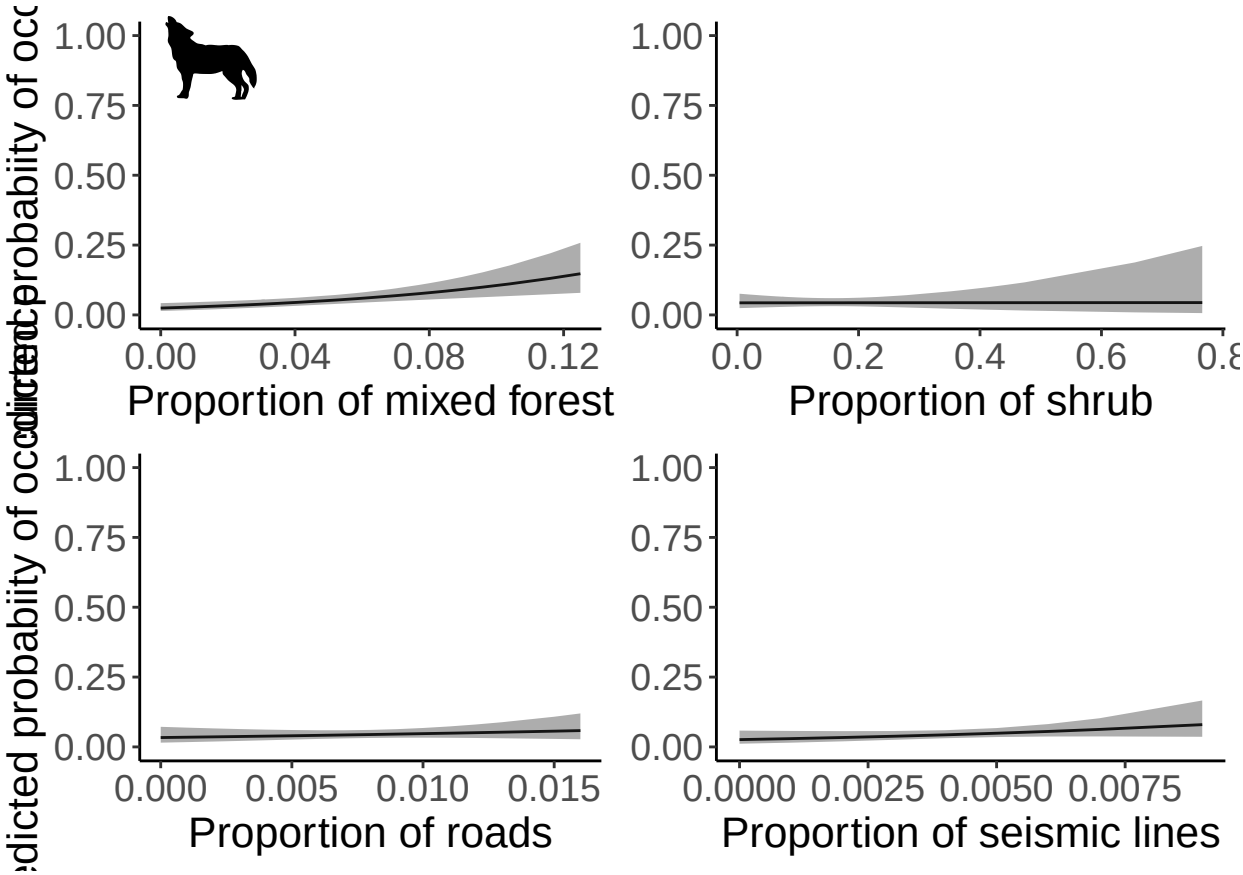


Save plot

```
ggsave('figures/grey_wolf_mixed_plot.jpg',
       wolf_mixed_plot,
       width = 14,
       height = 10,
       units = 'in',
       dpi = 600)
```

Join plots

```
wolf_plot <- ggarrange(wolf_mixed_plot,
                       wolf_shrub_plot,
                       wolf_rds_plot,
                       wolf_seismic_plot,

                       nrow = 2,
                       ncol = 2)

wolf_plot
```



Save final figure

```
ggsave('figures/publication_figures/wolf_plot.jpg',
       wolf_plot,
       width = 10,
       height = 8,
       units = 'in',
       dpi = 600)
```

Lynx

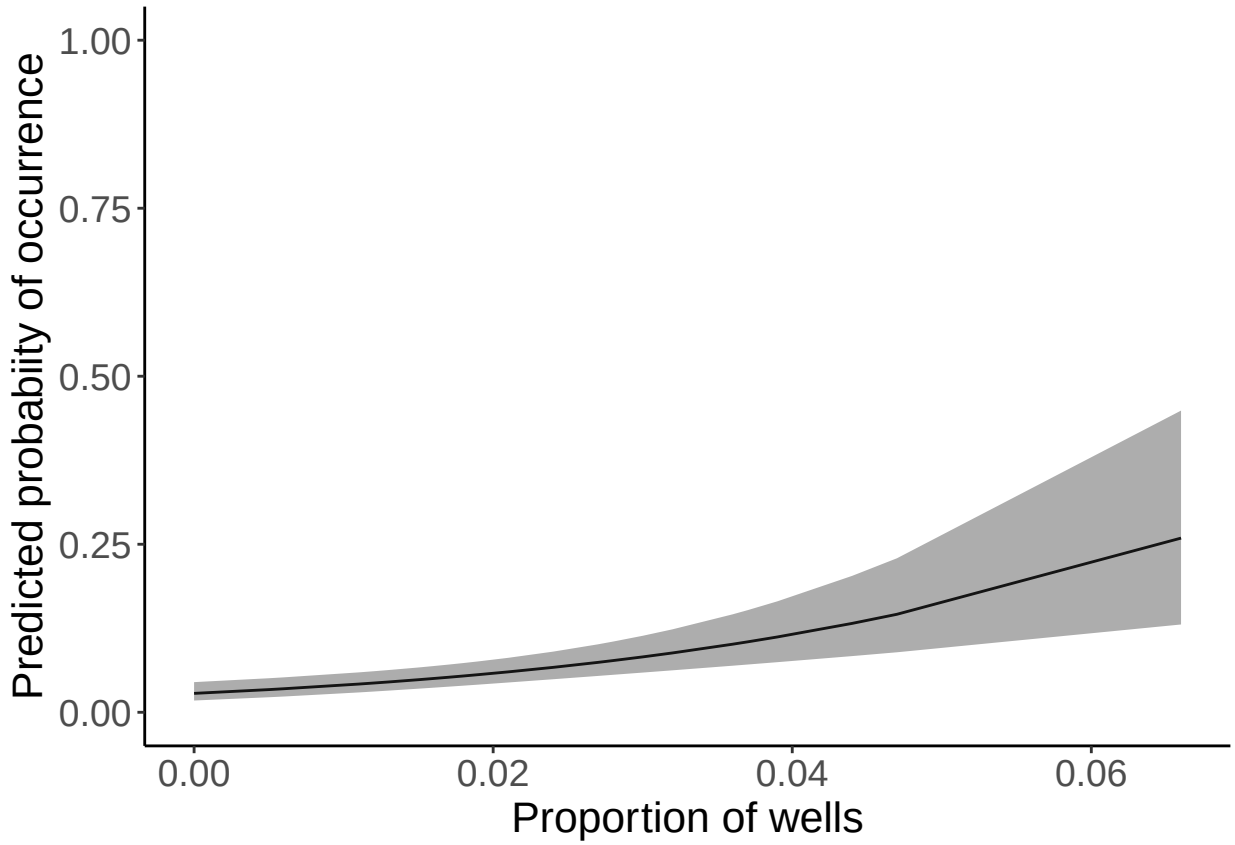
data

```
# supply vector of fixed effects as they appear in model
lynx_fes <- c('wells',
              'harvest_pre2000',
               'harvest_2000')

# Use purrr to iterate ggpredict and rename the list elements
lynx_predicted_data <- map(set_names(lynx_fes), ~ {
  ggpredict(lynx_poly, terms = paste0(.x, "[all]"))
})

# I viewed the full data from my environemnt but here's a printout of one variable
head(lynx_predicted_data$wells)
```

```
## # Predicted probabilities of cbind(lynx, absent_lynx)
##
##    wells | Predicted |     95% CI
## ---------------------------------
##     0.00 |      0.03 | 0.02, 0.04
## 5.00e-03 |      0.03 | 0.02, 0.05
## 6.00e-03 |      0.04 | 0.02, 0.05
##     0.01 |      0.04 | 0.03, 0.06
##     0.01 |      0.04 | 0.03, 0.06
##     0.01 |      0.05 | 0.03, 0.06
##
## Adjusted for:
## *    harvest_2000 = 0.12
## * harvest_pre2000 = 0.12
```

Wells

```
# name plot and assign to environment
lynx_well_plot <-

  # provide data from ggpredict with x and y
  ggplot(lynx_predicted_data$wells,
         aes(x = x,
             y = predicted)) +

  # plot relationship with line
  geom_line() +

  # plot confidence with geom ribbon, must supply new aesthetics
  geom_ribbon(aes(ymin = conf.low,
                  ymax = conf.high),
              alpha = 0.4) +

  # set the axis limits so all plots go from 0-1
  scale_y_continuous(limits = c(0, 1)) +

  # label axis
  labs(x = 'Proportion of wells',
       y = 'Predicted probabiity of occurrence') +

  # specify overall theme
  theme_classic() +

  # change font size
  theme(axis.title = element_text(size = 16),
        axis.text = element_text(size = 14))

   # # add silhouette
   # add_phylopic(uuid = '24f763a3-accf-44c9-9a08-71e9834047b7',
   #              x = 0.06,
   #              y = 0.9,
   #              ysize = 0.2)

lynx_well_plot
```



Save plot

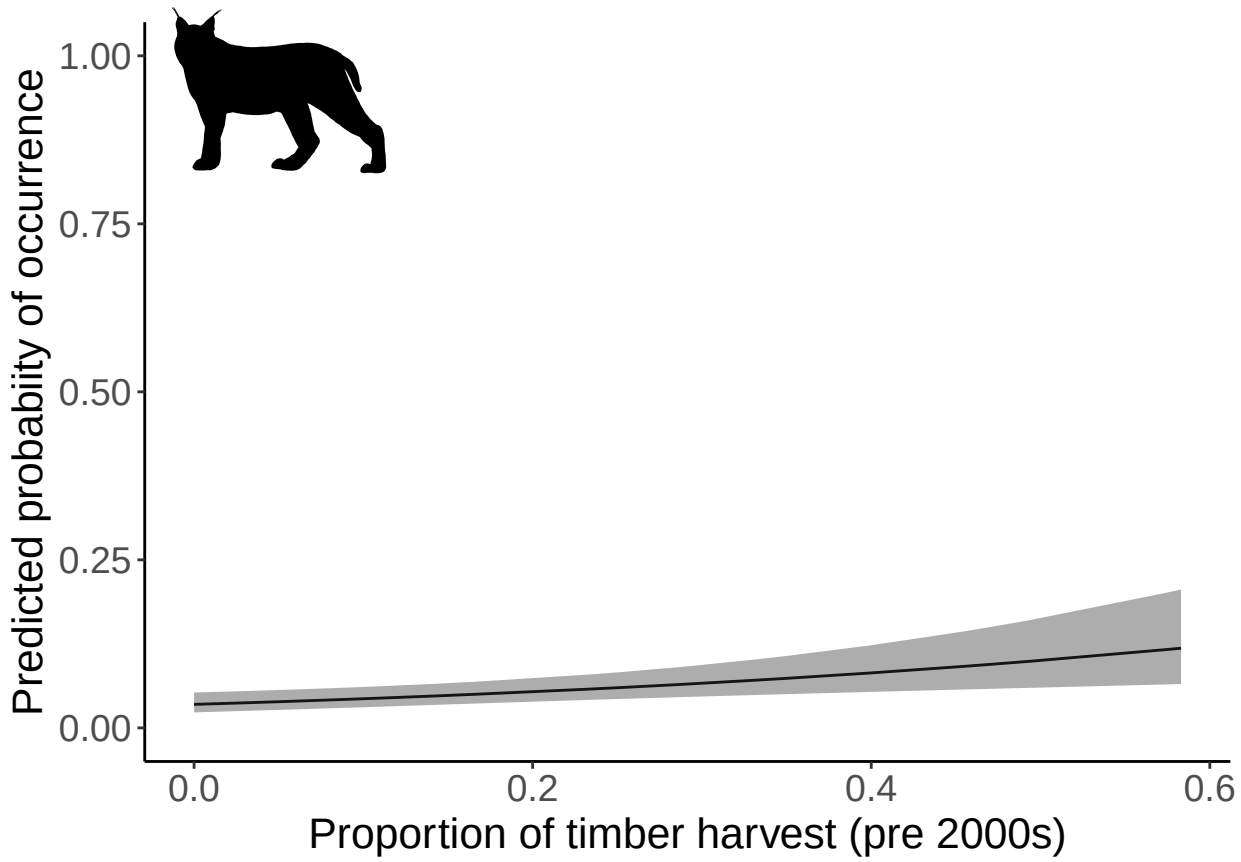
```
ggsave('figures/lynx_well_plot.jpg',
       lynx_well_plot,
        width = 14,
        height = 10,
        units = 'in',
        dpi = 600)
```

Harvest pre 2000s

```
# name plot and assign to environment
lynx_harvest_pre_plot <-

  # provide data from ggpredict with x and y
  ggplot(lynx_predicted_data$harvest_pre2000,
         aes(x = x,
             y = predicted)) +

  # plot relationship with line
  geom_line() +

  # plot confidence with geom ribbon, must supply new aesthetics
  geom_ribbon(aes(ymin = conf.low,
                  ymax = conf.high),
              alpha = 0.4) +
```

```
  # set the axis limits so all plots go from 0-1
  scale_y_continuous(limits = c(0, 1)) +

  # label axis
  labs(x = 'Proportion of timber harvest (pre 2000s)',
       y = 'Predicted probabiity of occurrence') +

  # specify overall theme
  theme_classic() +

  # change font size
  theme(axis.title = element_text(size = 16),
        axis.text = element_text(size = 14)) +

   # add silhouette
   add_phylopic(uuid = '24f763a3-accf-44c9-9a08-71e9834047b7',
                x = 0.05,
                y = 0.95,
                ysize = 0.25)
```

```
lynx_harvest_pre_plot
```



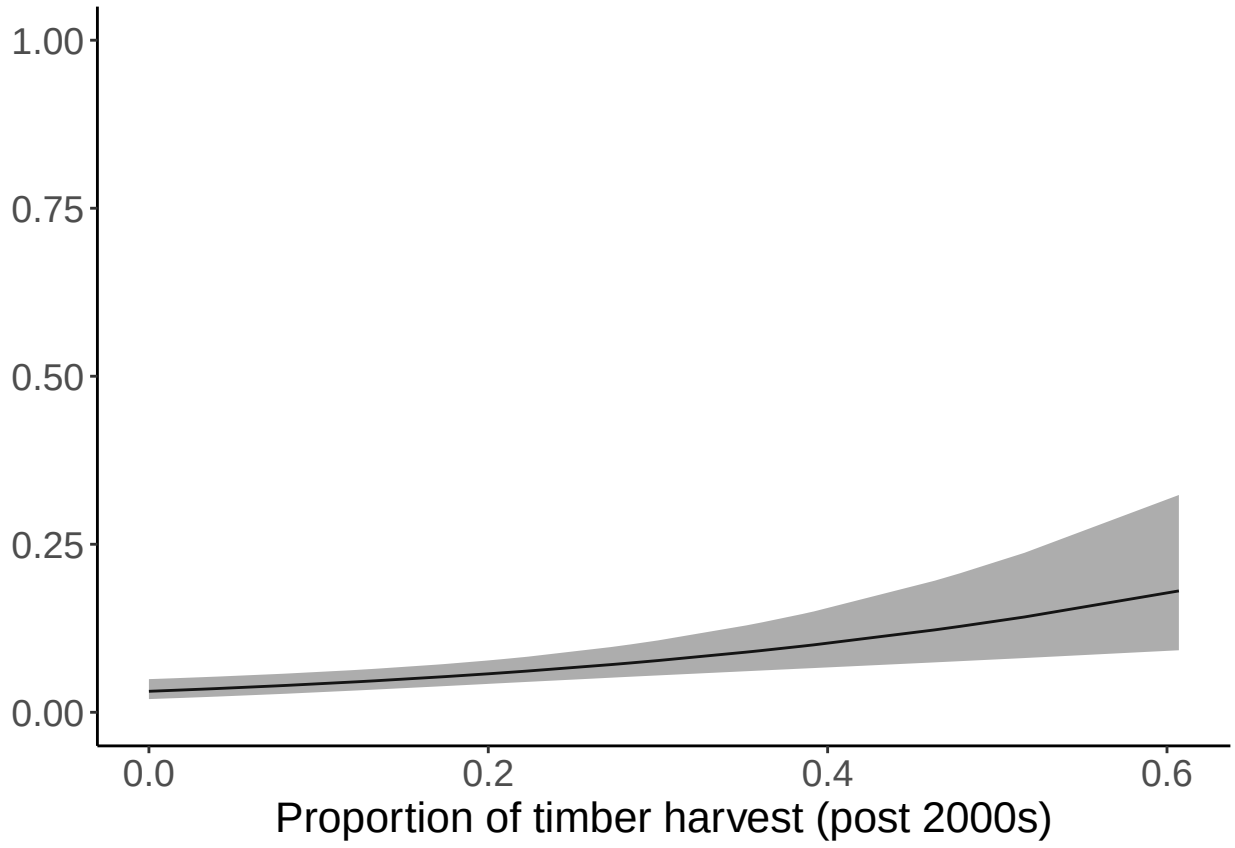
Save plot

```
ggsave('figures/lynx_harvest_pre2000_plot.jpg',
       lynx_harvest_pre_plot,
        width = 14,
```

```
       height = 10,
       units = 'in',
       dpi = 600)
```

Harvest post 2000s

```
# name plot and assign to environment
lynx_harvest_post_plot <-

  # provide data from ggpredict with x and y
  ggplot(lynx_predicted_data$harvest_2000,
         aes(x = x,
             y = predicted)) +

  # plot relationship with line
  geom_line() +

  # plot confidence with geom ribbon, must supply new aesthetics
  geom_ribbon(aes(ymin = conf.low,
                  ymax = conf.high),
              alpha = 0.4) +

  # set the axis limits so all plots go from 0-1
  scale_y_continuous(limits = c(0, 1)) +

  # label axis
  labs(x = 'Proportion of timber harvest (post 2000s)',
       y = 'Predicted probabiity of occurrence') +

  # specify overall theme
  theme_classic() +

  # change font size
  theme(axis.title = element_text(size = 16),
        axis.text = element_text(size = 14),
        axis.title.y = element_blank())

   # # add silhouette
   # add_phylopic(uuid = '24f763a3-accf-44c9-9a08-71e98340476b7',
   #              x = 0.55,
   #              y = 0.9,
   #              ysize = 0.2)

lynx_harvest_post_plot
```

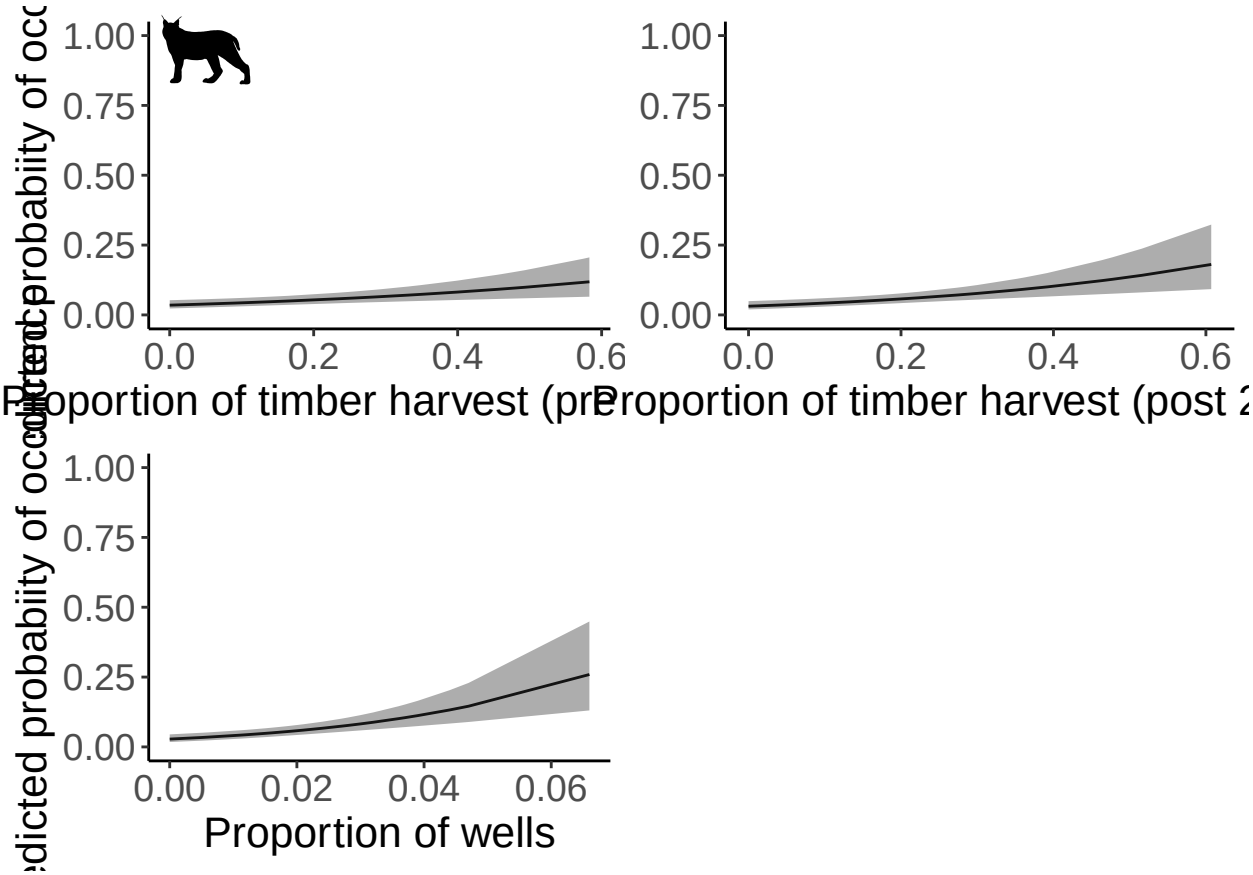


Save plot

```
ggsave('figures/lynx_harvest_post2000_plot.jpg',
       lynx_harvest_post_plot,
        width = 14,
        height = 10,
        units = 'in',
        dpi = 600)
```

Join plots

```
lynx_plot <- ggarrange(lynx_harvest_pre_plot,
                       lynx_harvest_post_plot,
                       lynx_well_plot,

                       nrow = 2,
                       ncol = 2)

lynx_plot
```



Save final figure

```
ggsave('figures/publication_figures/lynx_plot.jpg',
       lynx_plot,
       width = 14,
       height = 10,
       units = 'in',
       dpi = 600)
```

Moose

Data

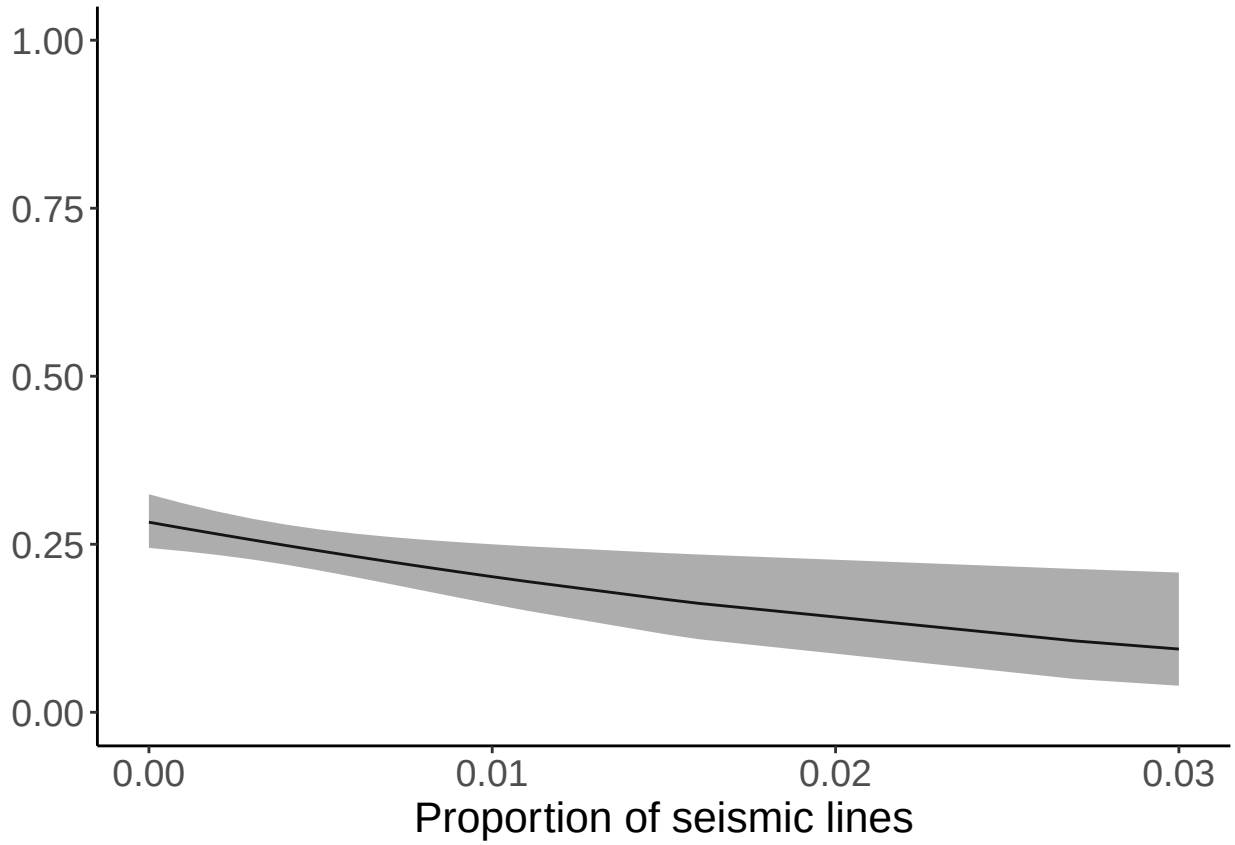
```
# supply vector of fixed effects as they appear in model
moose_fes <- c('roads',
               'seismic_lines')

# Use purrr to iterate ggpredict and rename the list elements
moose_predicted_data <- map(set_names(moose_fes), ~ {
  ggpredict(moose_linear, terms = paste0(.x, "[all]"))
})

# I viewed the full data from my environemnt but here's a printout of one variable
head(moose_predicted_data$roads)
```

```
## # Predicted probabilities of cbind(moose, absent_moose)
```

```
##
##    roads | Predicted |     95% CI
## --------------------------------
##     0.00 |      0.35 | 0.31, 0.40
## 2.00e-03 |      0.32 | 0.29, 0.36
## 4.00e-03 |      0.30 | 0.26, 0.33
## 5.00e-03 |      0.28 | 0.25, 0.31
## 6.00e-03 |      0.27 | 0.24, 0.30
## 7.00e-03 |      0.26 | 0.23, 0.29
##
## Adjusted for:
## * seismic_lines = 0.00
```

Seismic lines

```
# name plot and assign to environment
moose_seismic_plot <-

  # provide data from ggpredict with x and y
  ggplot(moose_predicted_data$seismic_lines,
         aes(x = x,
             y = predicted)) +

  # plot relationship with line
  geom_line() +

  # plot confidence with geom ribbon, must supply new aesthetics
  geom_ribbon(aes(ymin = conf.low,
                  ymax = conf.high),
              alpha = 0.4) +

  # set the axis limits so all plots go from 0-1
  scale_y_continuous(limits = c(0, 1)) +

  # label axis
  labs(x = 'Proportion of seismic lines',
       y = 'Predicted probabiity of occurrence') +

  # specify overall theme
  theme_classic() +

  # change font size
  theme(axis.title = element_text(size = 16),
        axis.text = element_text(size = 14),
        axis.title.y = element_blank())

   # # add silhouette
   # add_phylopic(uuid = '74eab34a-498c-4614-aece-f02361874f79',
   #              x = 0.028,
   #              y = 0.9,
   #              ysize = 0.2)

moose_seismic_plot
```



Save plot

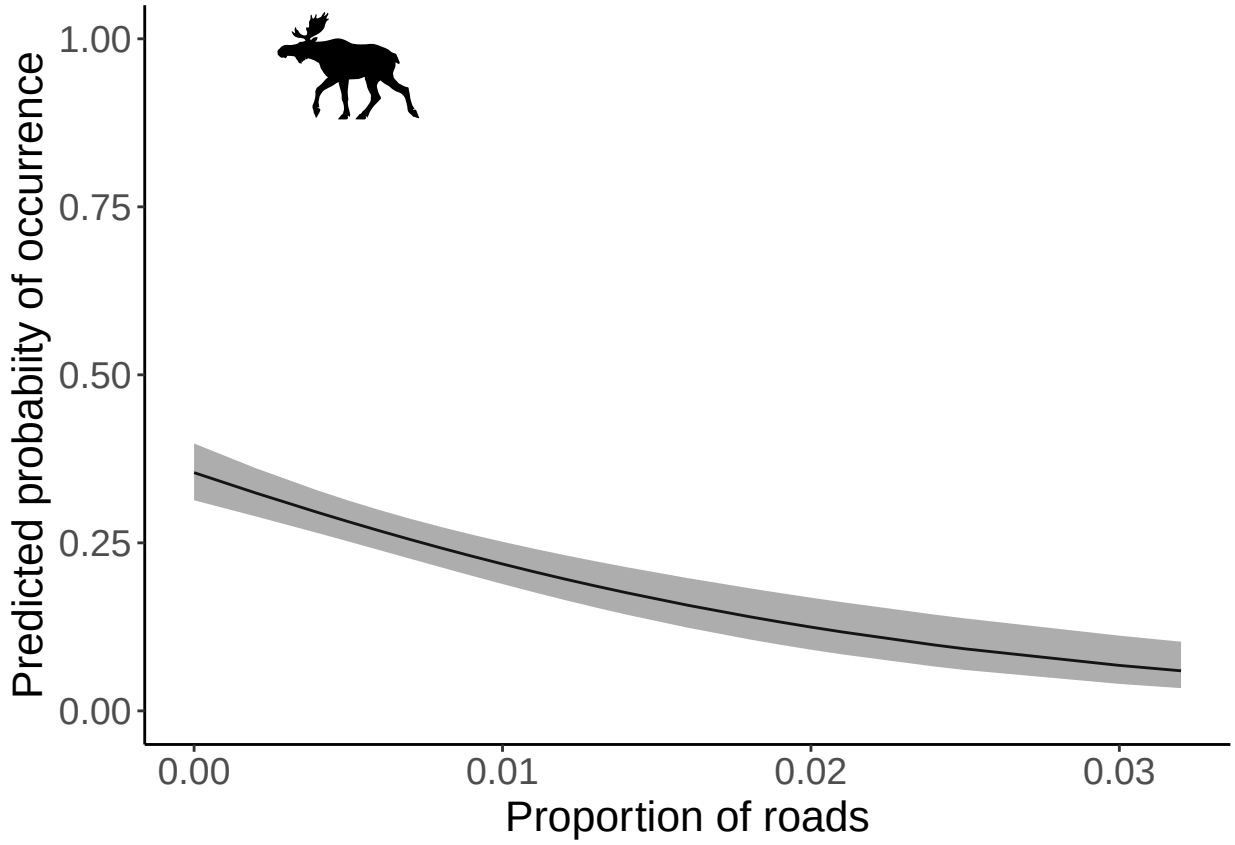
```
ggsave('figures/moose_seismic_lines_plot.jpg',
      moose_seismic_plot,
       width = 14,
       height = 10,
       units = 'in',
       dpi = 600)
```

Roads

```
# name plot and assign to environment
moose_rds_plot <-

  # provide data from ggpredict with x and y
  ggplot(moose_predicted_data$roads,
         aes(x = x,
             y = predicted)) +

  # plot relationship with line
  geom_line() +

  # plot confidence with geom ribbon, must supply new aesthetics
  geom_ribbon(aes(ymin = conf.low,
                  ymax = conf.high),
              alpha = 0.4) +
```

```
  # set the axis limits so all plots go from 0-1
  scale_y_continuous(limits = c(0, 1)) +

  # label axis
  labs(x = 'Proportion of roads',
       y = 'Predicted probabiity of occurrence') +

  # specify overall theme
  theme_classic() +

  # change font size
  theme(axis.title = element_text(size = 16),
        axis.text = element_text(size = 14)) +

   # add silhouette
   add_phylopic(uuid = '74eab34a-498c-4614-aece-f02361874f79',
                x = 0.005,
                y = 0.96,
                ysize = 0.16)
```

```
moose_rds_plot
```



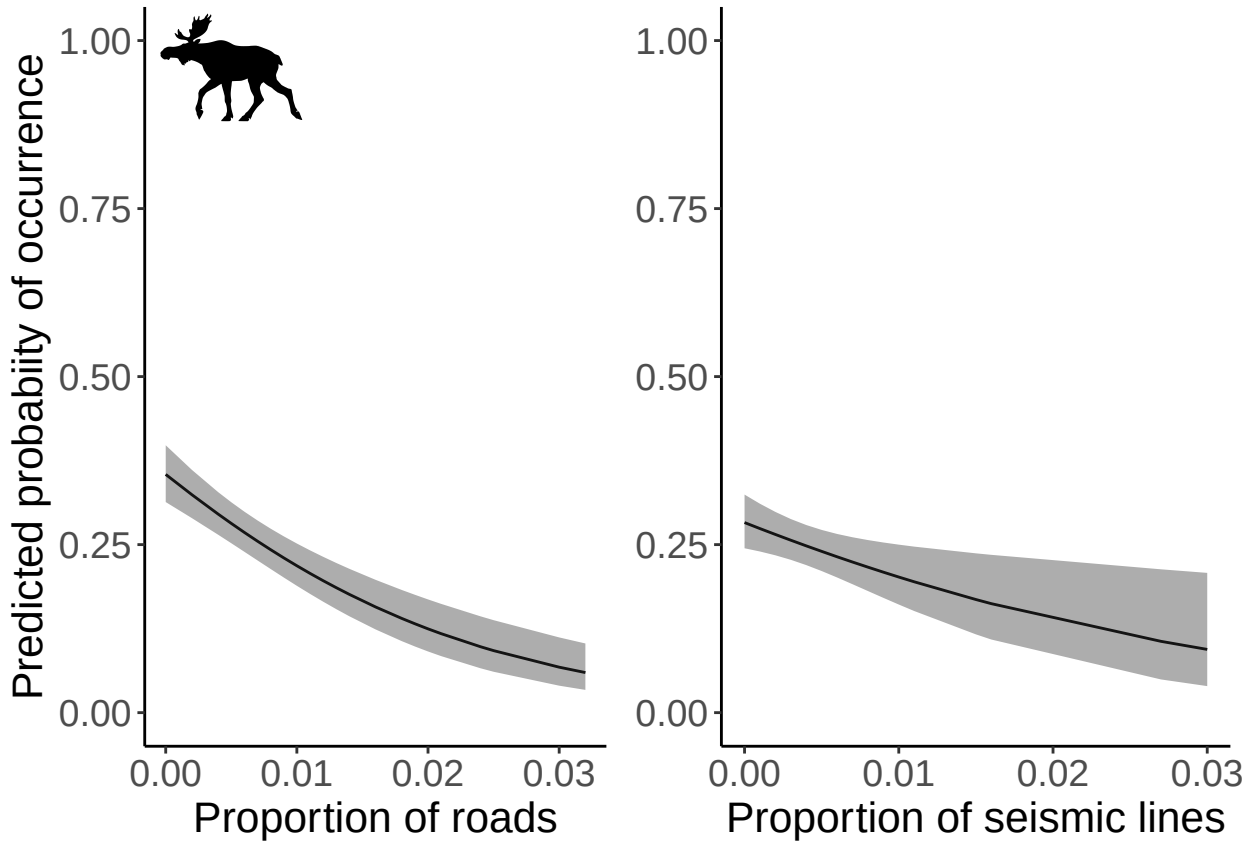
Save plot

```
ggsave('figures/moose_roads_plot.jpg',
       moose_rds_plot,
```

```
       width = 14,
       height = 10,
       units = 'in',
       dpi = 600)
```

Join plots

```
moose_plot <- ggarrange(moose_rds_plot,
                        moose_seismic_plot,

                        nrow = 1,
                        ncol = 2)

moose_plot
```



Save final figure

```
ggsave('figures/publication_figures/moose_plot.jpg',
       moose_plot,
       width = 14,
       height = 10,
       units = 'in',
       dpi = 600)
```

Snowshoe hare

Data

```
# supply vector of fixed effects as they appear in model
hare_fes <- c('wells',
              'seismic_lines',
              'harvest_2000',
              'harvest_pre2000')

# Use purrr to iterate ggpredict and rename the list elements
hare_predicted_data <- map(set_names(hare_fes), ~ {
  ggpredict(snowshoe_hare_disturb, terms = paste0(.x, "[all]"))
})

# I viewed the full data from my environment but here's a printout of one variable
head(hare_predicted_data$wells)
```

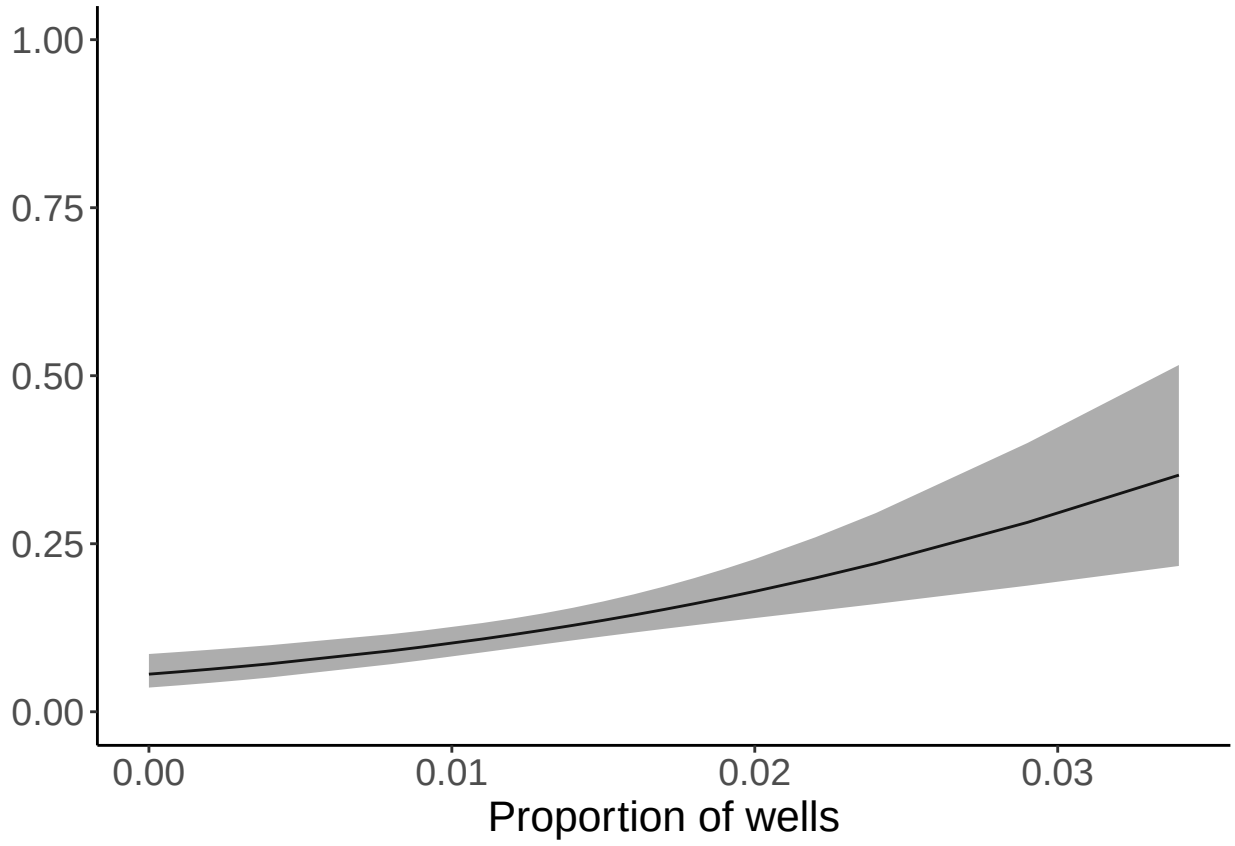
```
## # Predicted probabilities of cbind(snowshoe_hare, absent_snowshoe_hare)
##
##    wells | Predicted |     95% CI
## --------------------------------
##     0.00 |      0.06 | 0.04, 0.09
## 1.00e-03 |      0.06 | 0.04, 0.09
## 2.00e-03 |      0.06 | 0.04, 0.09
## 3.00e-03 |      0.07 | 0.05, 0.10
## 4.00e-03 |      0.07 | 0.05, 0.10
## 8.00e-03 |      0.09 | 0.07, 0.12
##
## Adjusted for:
## *    harvest_2000 = 0.14
## * harvest_pre2000 = 0.09
## *   seismic_lines = 0.00
```

Wells

```
# and now plot with ggplot

# name plot and assign to environment
hare_well_plot <-

  # provide data from ggpredict with x and y
  ggplot(hare_predicted_data$wells,
         aes(x = x,
             y = predicted)) +

  # plot relationship with line
  geom_line() +

  # plot confidence with geom ribbon, must supply new aesthetics
  geom_ribbon(aes(ymin = conf.low,
                  ymax = conf.high),
              alpha = 0.4) +
```

```
  # set the axis limits so all plots go from 0-1
  scale_y_continuous(limits = c(0, 1)) +

  # label axis
  labs(x = 'Proportion of wells',
       y = 'Predicted probabiity of occurrence') +

  # specify overall theme
  theme_classic() +

  # change font size
  theme(axis.title = element_text(size = 16),
        axis.text = element_text(size = 14),
        axis.title.y = element_blank())

   # # add silhouette
   # add_phylopic(uuid =  get_uuid(name = 'Lepus americanus'),
   #              x = 0.03,
   #              y = 0.9,
   #              ysize = 0.2)
```

```
hare_well_plot
```



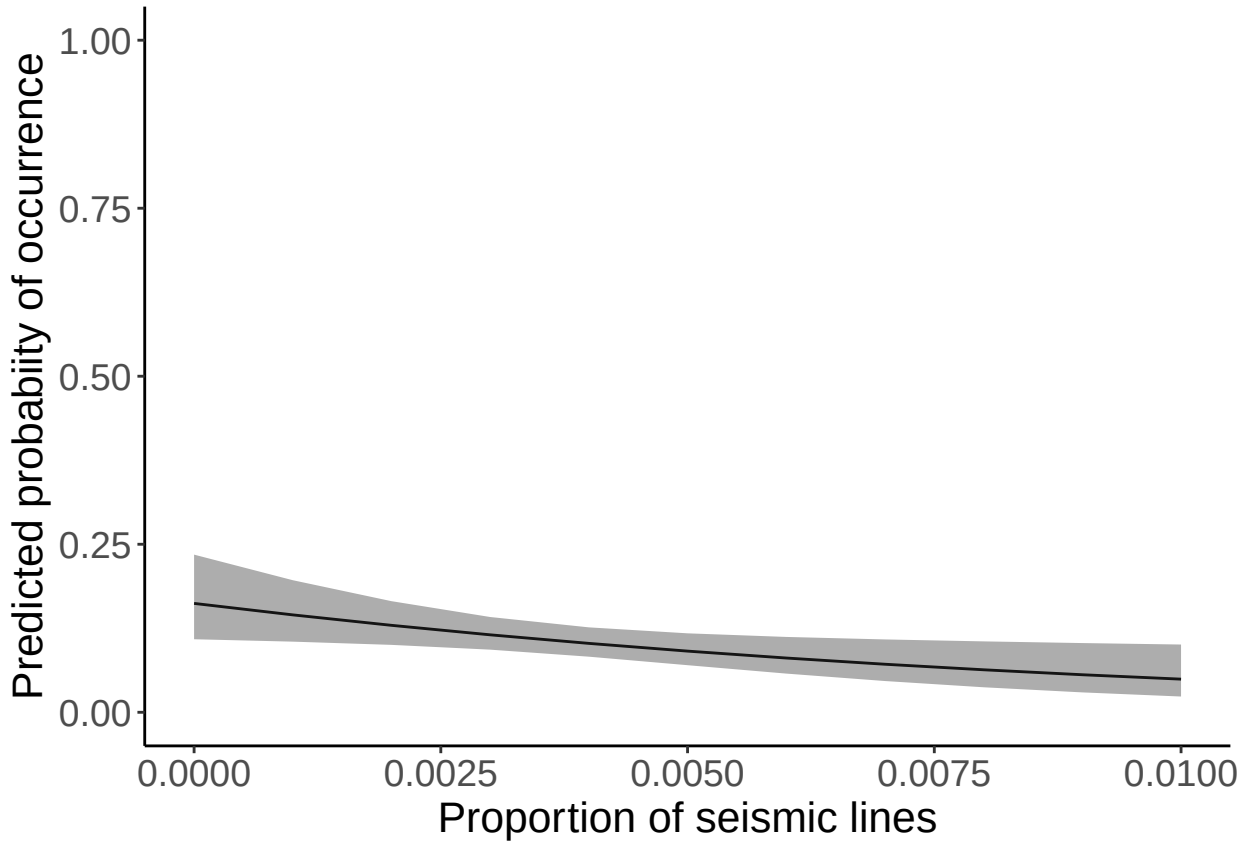
Save plot

```
ggsave('figures/snowshoehare_wells_plot.jpg',
       hare_well_plot,
```

```
       width = 14,
       height = 10,
       units = 'in',
       dpi = 600)
```

Seismic lines

```
# name plot and assign to environment
hare_seismic_plot <-

  # provide data from ggpredict with x and y
  ggplot(hare_predicted_data$seismic_lines,
         aes(x = x,
             y = predicted)) +

  # plot relationship with line
  geom_line() +

  # plot confidence with geom ribbon, must supply new aesthetics
  geom_ribbon(aes(ymin = conf.low,
                  ymax = conf.high),
              alpha = 0.4) +

  # set the axis limits so all plots go from 0-1
  scale_y_continuous(limits = c(0, 1)) +

  # label axis
  labs(x = 'Proportion of seismic lines',
       y = 'Predicted probabiity of occurrence') +

  # specify overall theme
  theme_classic() +

  # change font size
  theme(axis.title = element_text(size = 16),
        axis.text = element_text(size = 14))

   # # add silhouette
   # add_phylopic(uuid =  get_uuid(name = 'Lepus americanus'),
   #              x = 0.009,
   #              y = 0.9,
   #              ysize = 0.2)

hare_seismic_plot
```



Save plot

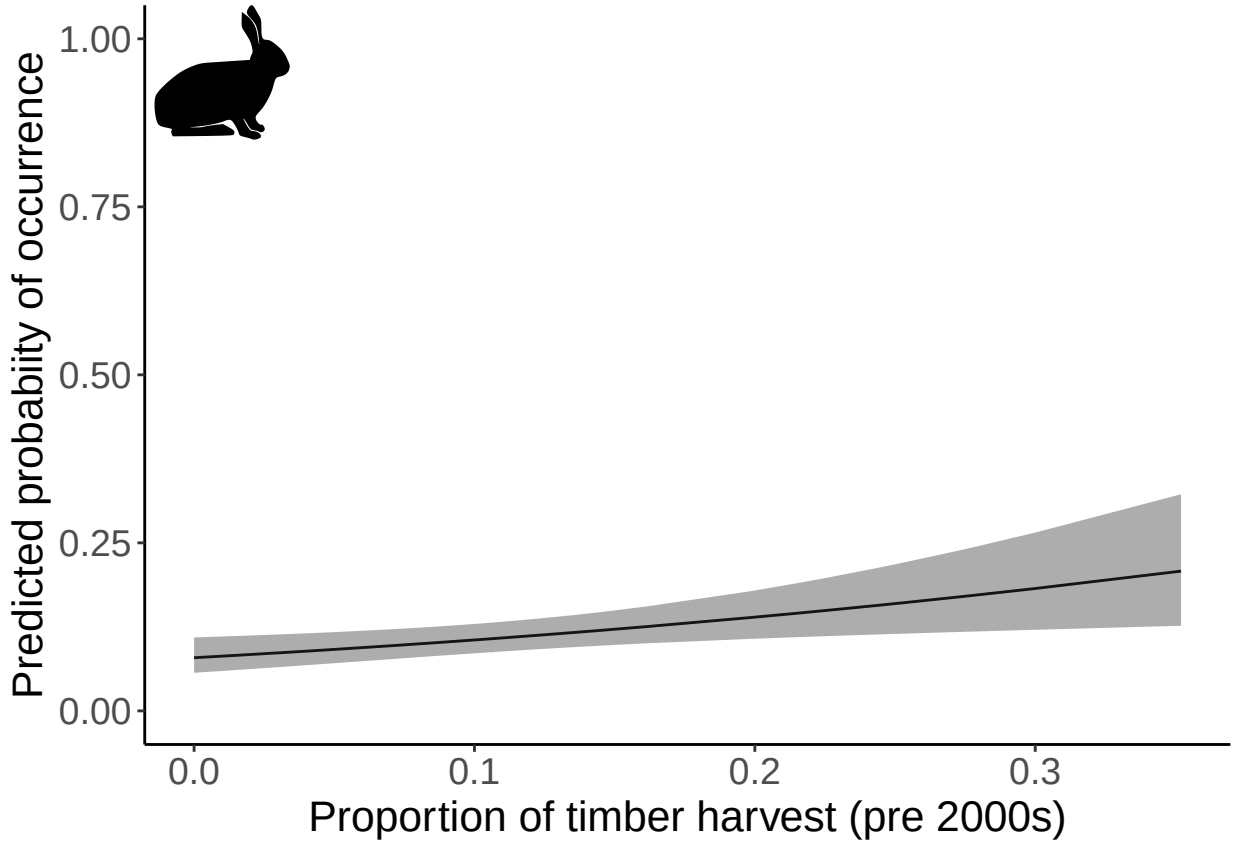
```
ggsave('figures/snowshoehare_seismic_plot.jpg',
       hare_seismic_plot,
       width = 14,
       height = 10,
       units = 'in',
       dpi = 600)
```

Harvest pre 2000s

```
# name plot and assign to environment
hare_harvest_pre_plot <-

  # provide data from ggpredict with x and y
  ggplot(hare_predicted_data$harvest_pre2000,
         aes(x = x,
             y = predicted)) +

  # plot relationship with line
  geom_line() +

  # plot confidence with geom ribbon, must supply new aesthetics
  geom_ribbon(aes(ymin = conf.low,
                  ymax = conf.high),
              alpha = 0.4) +
```

```
  # set the axis limits so all plots go from 0-1
  scale_y_continuous(limits = c(0, 1)) +

  # label axis
  labs(x = 'Proportion of timber harvest (pre 2000s)',
       y = 'Predicted probabiity of occurrence') +

  # specify overall theme
  theme_classic() +

  # change font size
  theme(axis.title = element_text(size = 16),
        axis.text = element_text(size = 14)) +

   # add silhouette
   add_phylopic(uuid =  get_uuid(name = 'Lepus americanus'),
                x = 0.01,
                y = 0.95,
                ysize = 0.2)
```

```
hare_harvest_pre_plot
```



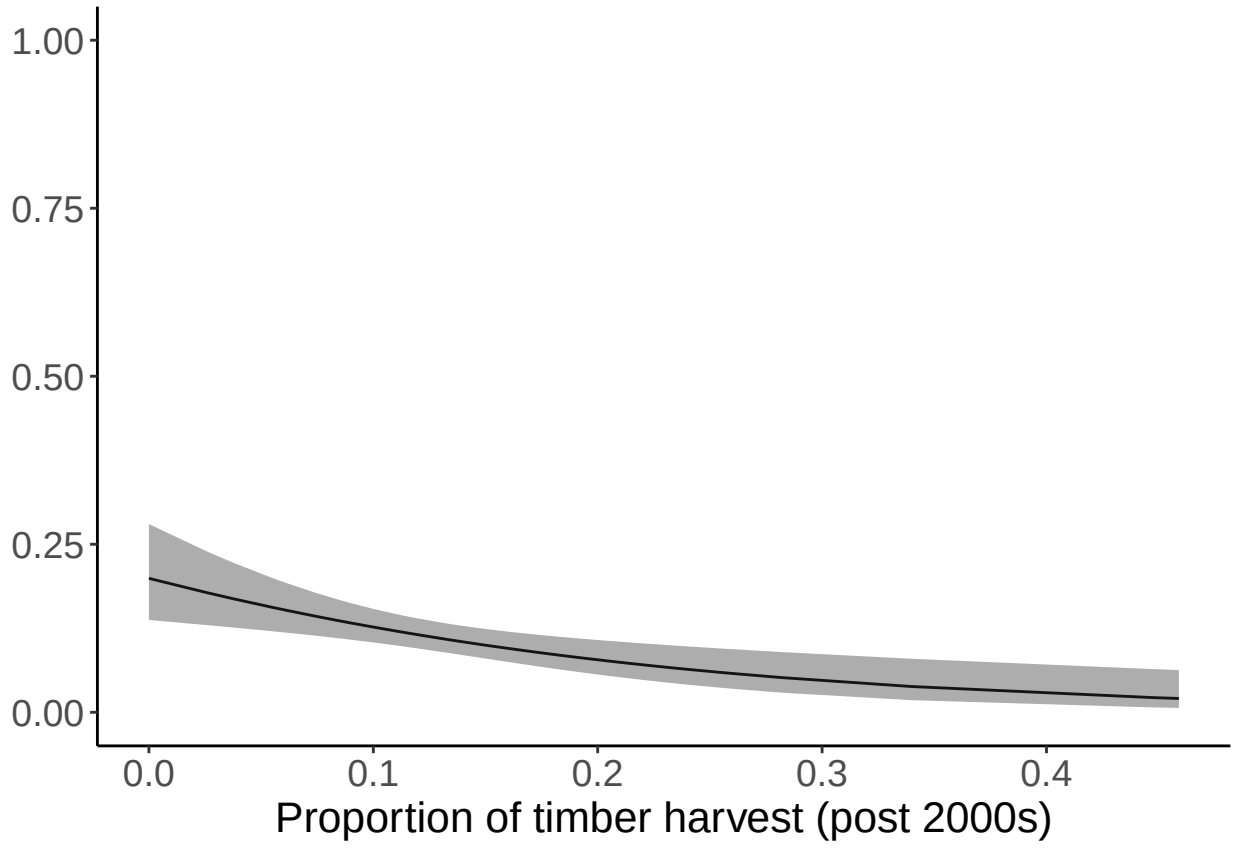
Save plot

```
ggsave('figures/snowshoehare_harvest_pre200s_plot.jpg',
       hare_harvest_pre_plot,
```

```
       width = 14,
       height = 10,
       units = 'in',
       dpi = 600)
```

Harvest post 2000s

```
# name plot and assign to environment
hare_harvest_post_plot <-

  # provide data from ggpredict with x and y
  ggplot(hare_predicted_data$harvest_2000,
         aes(x = x,
             y = predicted)) +

  # plot relationship with line
  geom_line() +

  # plot confidence with geom ribbon, must supply new aesthetics
  geom_ribbon(aes(ymin = conf.low,
                  ymax = conf.high),
              alpha = 0.4) +

  # set the axis limits so all plots go from 0-1
  scale_y_continuous(limits = c(0, 1)) +

  # label axis
  labs(x = 'Proportion of timber harvest (post 2000s)',
       y = 'Predicted probabiity of occurrence') +

  # specify overall theme
  theme_classic() +

  # change font size
  theme(axis.title = element_text(size = 16),
        axis.text = element_text(size = 14),
        axis.title.y = element_blank())

   # # add silhouette
   # add_phylopic(uuid =  get_uuid(name = 'Lepus americanus'),
   #              x = 0.43,
   #              y = 0.9,
   #              ysize = 0.2)

hare_harvest_post_plot
```

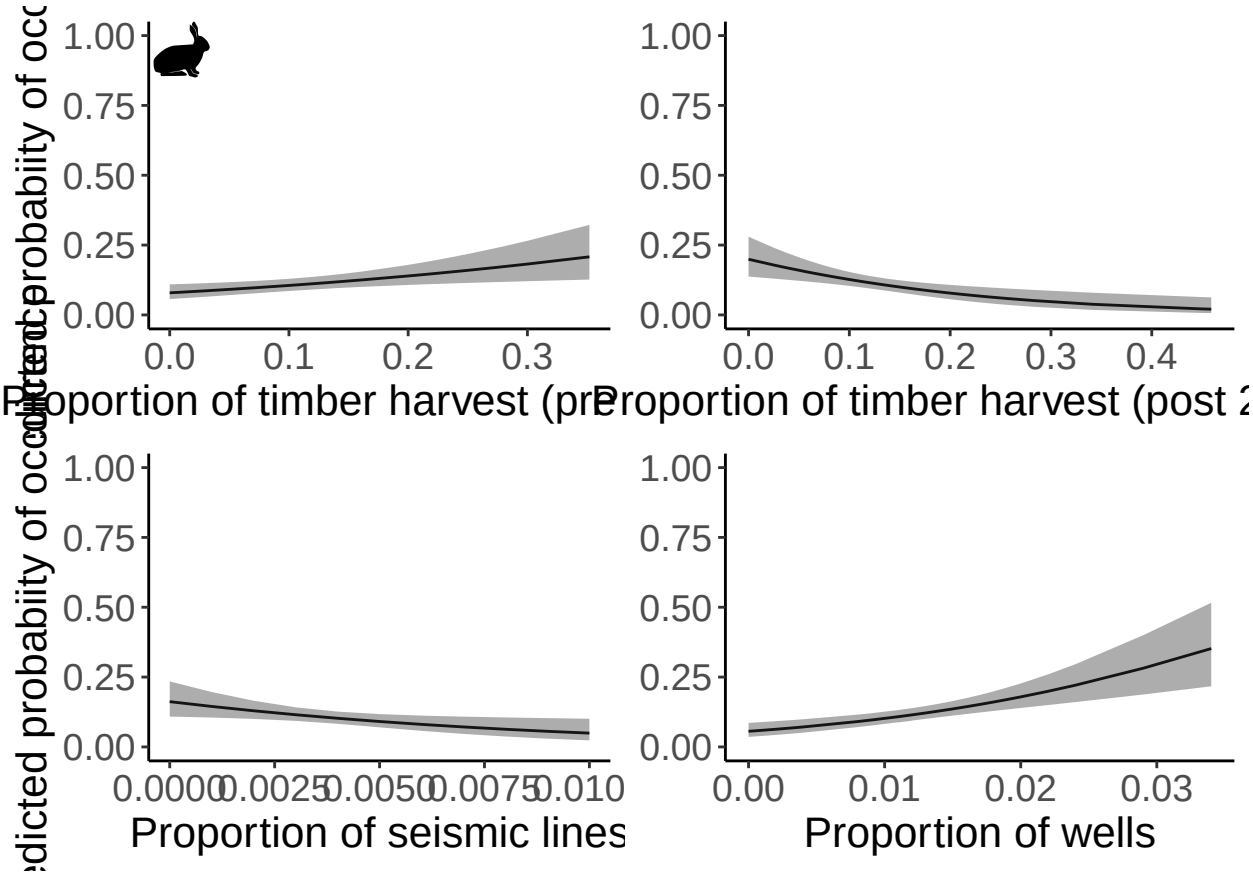


Save plot

```
ggsave('figures/snowshoehare_harvest_post200s_plot.jpg',
       hare_harvest_post_plot,
       width = 14,
       height = 10,
       units = 'in',
       dpi = 600)
```

Join plots

```
hare_plot <- ggarrange(hare_harvest_pre_plot,
                       hare_harvest_post_plot,
                       hare_seismic_plot,
                       hare_well_plot,

                       nrow = 2,
                       ncol = 2)

hare_plot
```



Save final figure

```
ggsave('figures/publication_figures/snowshoe_hare_plot.jpg',
       hare_plot,
       width = 14,
       height = 10,
       units = 'in',
       dpi = 600)
```

White-tailed deer

Data

```
# supply vector of fixed effects as they appear in model
w_deer_fes <- c('lc_broadleaf',
                'lc_grassland',
                'lc_mixed',
                'lc_shrub')

# Use purrr to iterate ggpredict and rename the list elements
w_deer_predicted_data <- map(set_names(w_deer_fes), ~ {
  ggpredict(w_deer_nat, terms = paste0(.x, "[all]"))
})

# I viewed the full data from my environemnt but here's a printout of one variable
```

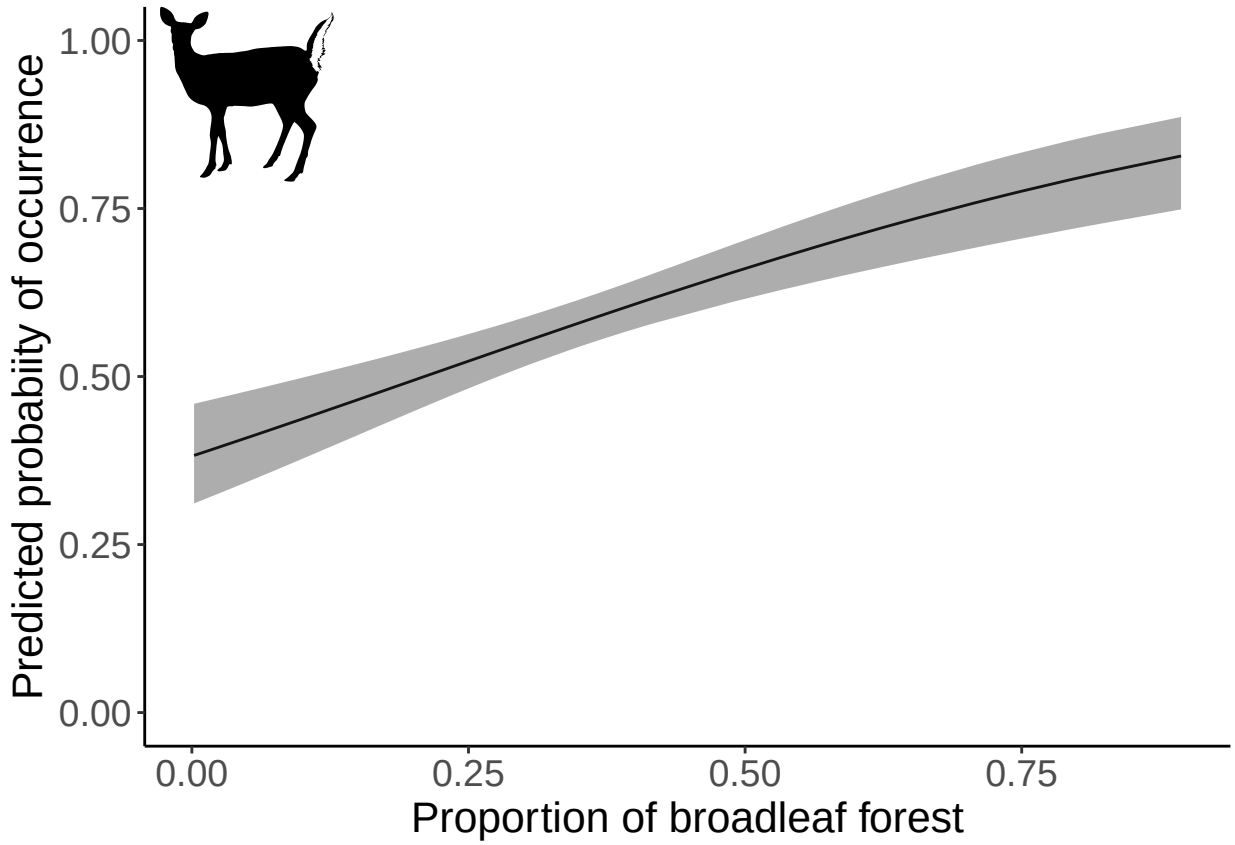
```
head(w_deer_predicted_data$lc_broadleaf)
```

```
## # Predicted probabilities of cbind(`white-tailed_deer`, `absent_white-tailed_deer`)
##
## lc_broadleaf | Predicted |     95% CI
## -------------------------------------
##     2.00e-03 |      0.38 | 0.31, 0.46
##         0.01 |      0.39 | 0.32, 0.46
##         0.02 |      0.39 | 0.33, 0.47
##         0.06 |      0.41 | 0.35, 0.48
##         0.06 |      0.41 | 0.35, 0.48
##         0.07 |      0.42 | 0.36, 0.49
##
## Adjusted for:
## * lc_grassland = 0.04
## *     lc_mixed = 0.04
## *     lc_shrub = 0.16
```

Broadleaf forest

```
w_deer_broadleaf_plot <-

  # provide data from ggpredict with x and y
  ggplot(w_deer_predicted_data$lc_broadleaf,
         aes(x = x,
             y = predicted)) +

  # plot relationship with line
  geom_line() +

  # plot confidence with geom ribbon, must supply new aesthetics
  geom_ribbon(aes(ymin = conf.low,
                  ymax = conf.high),
              alpha = 0.4) +

  # set the axis limits so all plots go from 0-1
  scale_y_continuous(limits = c(0, 1)) +

  # label axis
  labs(x = 'Proportion of broadleaf forest',
       y = 'Predicted probabiity of occurrence') +

  # specify overall theme
  theme_classic() +

  # change font size
  theme(axis.title = element_text(size = 16),
        axis.text = element_text(size = 14)) +

   # add silhouette
   add_phylopic(uuid = '6038e80c-398d-47b2-9a69-2b9edf436f64',
                x = 0.05,
                y = 0.92,
```

```
              ysize = 0.26)

w_deer_broadleaf_plot
```



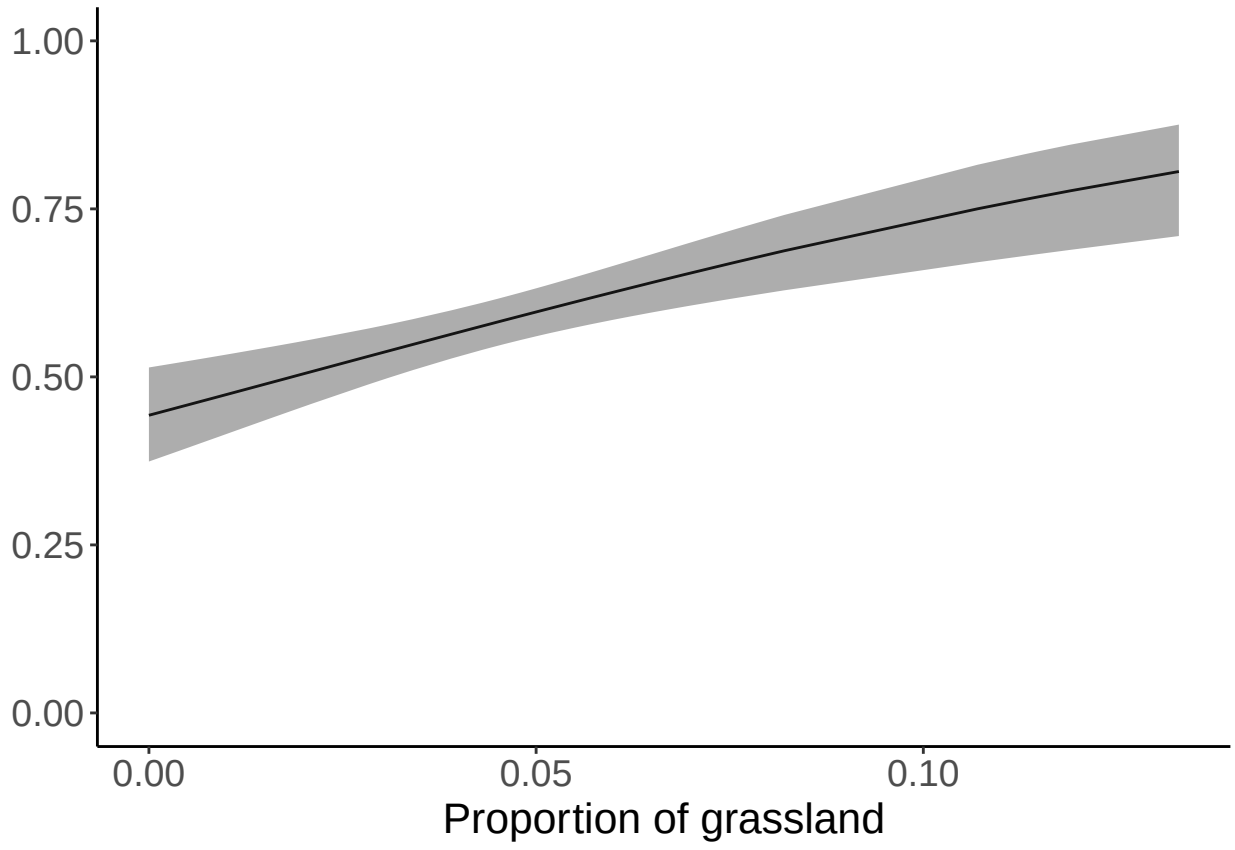
Save plot

```
ggsave('figures/whitetaileddeer_broadleaf_plot.jpg',
       w_deer_broadleaf_plot,
       width = 14,
       height = 10,
       units = 'in',
       dpi = 600)
```

Grassland

```
w_deer_grass_plot <-

  # provide data from ggpredict with x and y
  ggplot(w_deer_predicted_data$lc_grassland,
         aes(x = x,
             y = predicted)) +

  # plot relationship with line
  geom_line() +

  # plot confidence with geom ribbon, must supply new aesthetics
  geom_ribbon(aes(ymin = conf.low,
```

```
                ymax = conf.high),
            alpha = 0.4) +

  # set the axis limits so all plots go from 0-1
  scale_y_continuous(limits = c(0, 1)) +

  # label axis
  labs(x = 'Proportion of grassland',
       y = 'Predicted probabiity of occurrence') +

  # specify overall theme
  theme_classic() +

  # change font size
  theme(axis.title = element_text(size = 16),
        axis.text = element_text(size = 14),
        axis.title.y = element_blank())

   # # add silhouette
   # add_phylopic(uuid = '6038e80c-398d-47b2-9a69-2b9edf436f64',
   #              x = 0.12,
   #              y = 0.95,
   #              ysize = 0.2)

w_deer_grass_plot
```

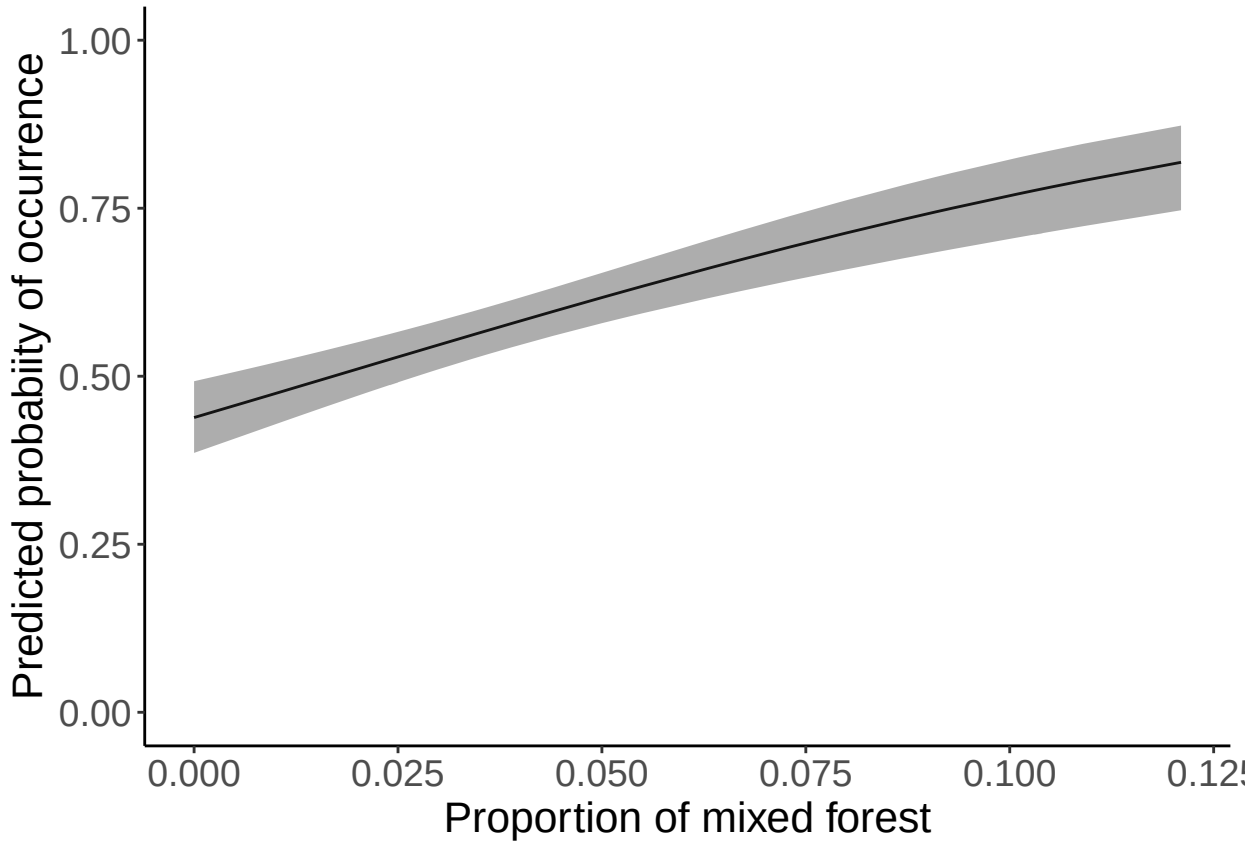


Save plot

```
ggsave('figures/whitetaileddeer_grassland_plot.jpg',
       w_deer_grass_plot,
       width = 14,
       height = 10,
       units = 'in',
       dpi = 600)
```

Mixed forest

```
w_deer_mixed_plot <-

  # provide data from ggpredict with x and y
  ggplot(w_deer_predicted_data$lc_mixed,
         aes(x = x,
             y = predicted)) +

  # plot relationship with line
  geom_line() +

  # plot confidence with geom ribbon, must supply new aesthetics
  geom_ribbon(aes(ymin = conf.low,
                  ymax = conf.high),
              alpha = 0.4) +

  # set the axis limits so all plots go from 0-1
  scale_y_continuous(limits = c(0, 1)) +

  # label axis
  labs(x = 'Proportion of mixed forest',
       y = 'Predicted probabiity of occurrence') +

  # specify overall theme
  theme_classic() +

  # change font size
  theme(axis.title = element_text(size = 16),
        axis.text = element_text(size = 14))

   # # add silhouette
   # add_phylopic(uuid = '6038e80c-398d-47b2-9a69-2b9edf436f64',
   #              x = 0.125,
   #              y = 0.95,
   #              ysize = 0.2)

w_deer_mixed_plot
```



Save plot

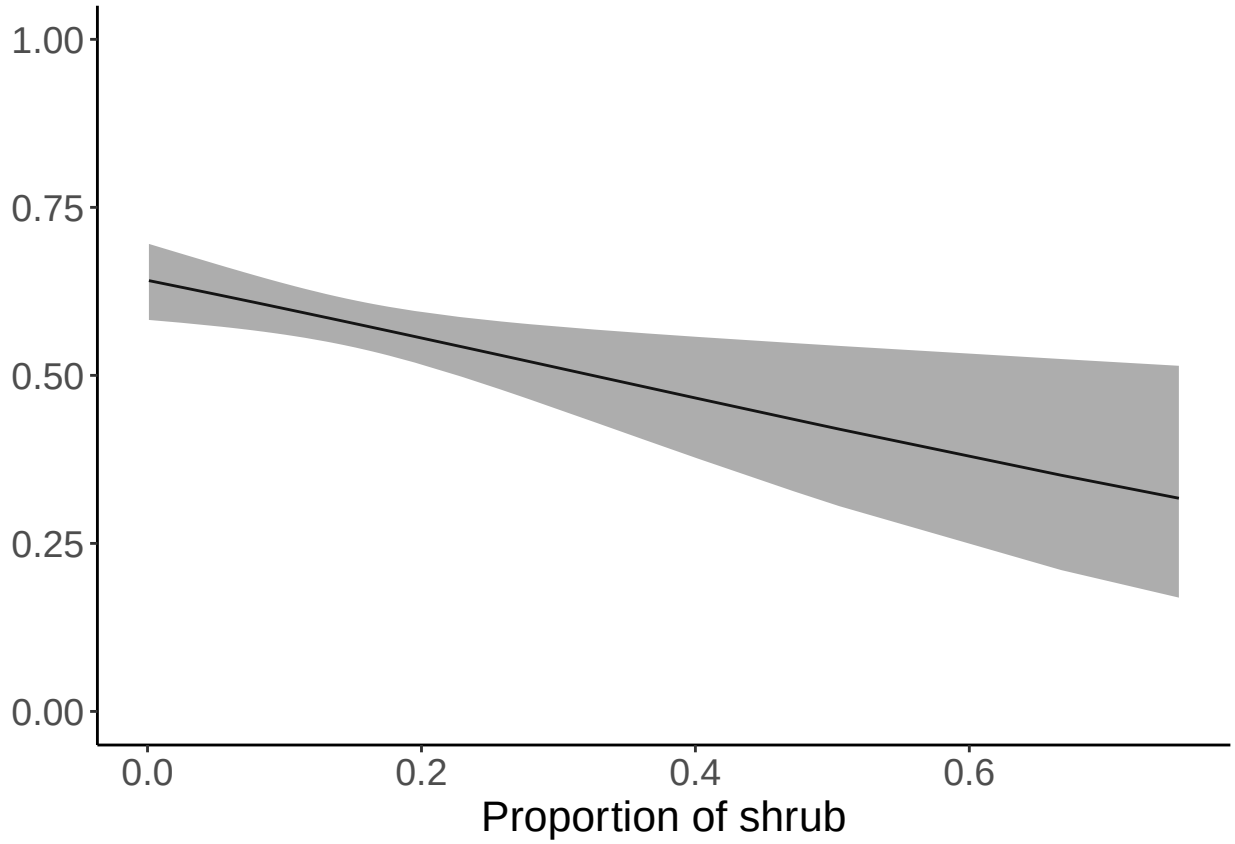
```
ggsave('figures/whitetaileddeer_mixed_plot.jpg',
       w_deer_mixed_plot,
       width = 14,
       height = 10,
       units = 'in',
       dpi = 600)
```

Shrubs

```
w_deer_shrub_plot <-

  # provide data from ggpredict with x and y
  ggplot(w_deer_predicted_data$lc_shrub,
         aes(x = x,
             y = predicted)) +

  # plot relationship with line
  geom_line() +

  # plot confidence with geom ribbon, must supply new aesthetics
  geom_ribbon(aes(ymin = conf.low,
                  ymax = conf.high),
              alpha = 0.4) +

  # set the axis limits so all plots go from 0-1
```

```
  scale_y_continuous(limits = c(0, 1)) +

  # label axis
  labs(x = 'Proportion of shrub',
       y = 'Predicted probabiity of occurrence') +

  # specify overall theme
  theme_classic() +

  # change font size
  theme(axis.title = element_text(size = 16),
        axis.text = element_text(size = 14),
        axis.title.y = element_blank())

   # # add silhouette
   # add_phylopic(uuid = '6038e80c-398d-47b2-9a69-2b9edf436f64',
   #              x = 0.7,
   #              y = 0.95,
   #              ysize = 0.2)
```

```
w_deer_shrub_plot
```



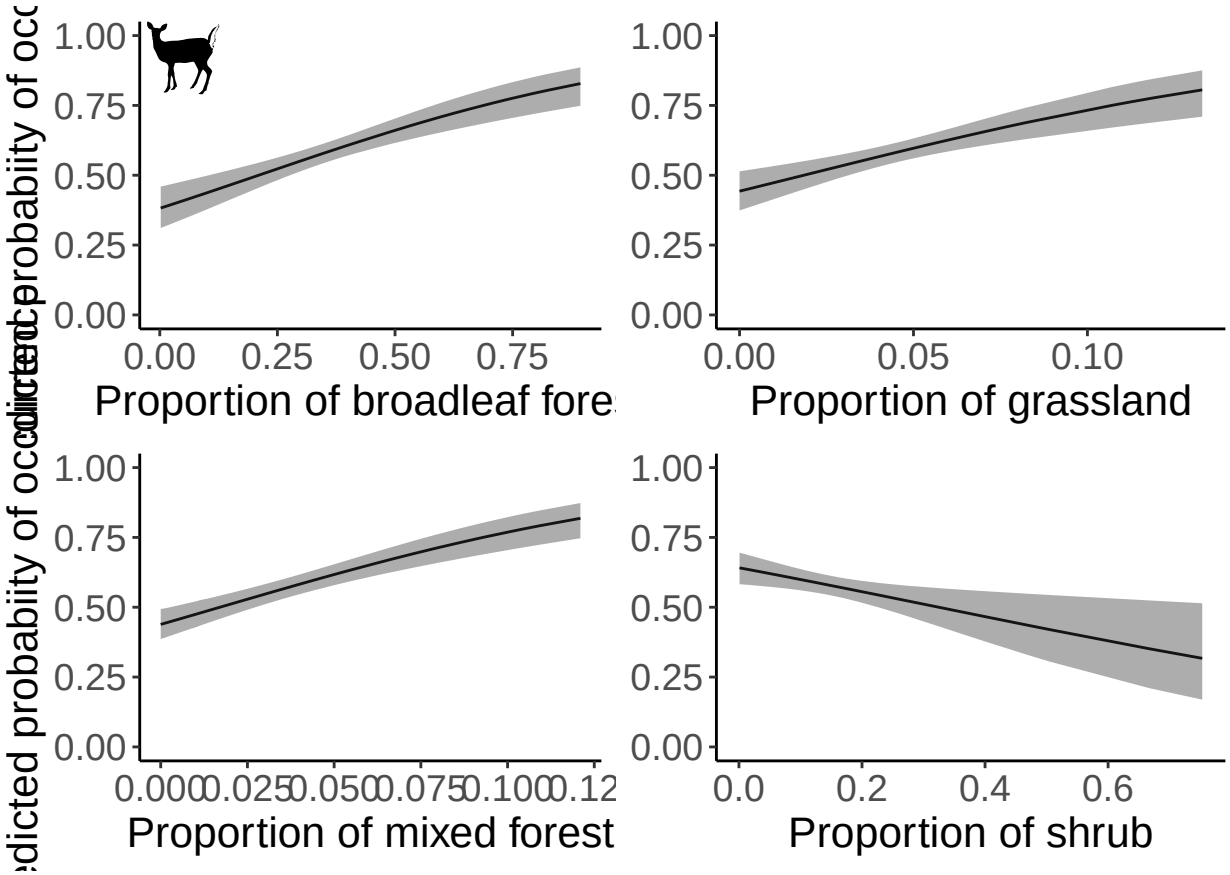
Save plot

```
ggsave('figures/whitetaileddeer_shrub_plot.jpg',
       w_deer_shrub_plot,
```

```
       width = 14,
       height = 10,
       units = 'in',
       dpi = 600)
```

Join plots

```
deer_plot <- ggarrange(w_deer_broadleaf_plot,
                       w_deer_grass_plot,
                       w_deer_mixed_plot,
                       w_deer_shrub_plot,

                       nrow = 2,
                       ncol = 2)

# view plot
deer_plot
```



Save final figure

```
ggsave('figures/publication_figures/deer_plot.jpg',
       deer_plot,
       width = 14,
       height = 10,
       units = 'in',
```

```
       dpi = 600)
```