Model v1.1 Sensitivity analysis - Full Factorial

Eduardo Zanette

05/08/2022

Abstract

This document was produced for reporting the sensitivity analysis (Full factorial design) of the BLT Model with the nlrx package. The main aim was to see if we could drop an implementation (which we called 'phenology' so far, but it is now called 'feeding bout' for sake of clarity). We also aimed at testing the effect of each of the four parameters below. For this, we ran the model with the same parameters with phenology "on" and "off". We used the Guareí environment with resources from July.

- Parameters related to memory:

1. "step_revisit" = list(min=1, max = 100, step = 10, qfun="qunif")
2. "visual" = list(min=1, max = 3, step = 1, qfun="qunif")

- Parameters related to movement:

4. "p-foraging-while-traveling" = list(min = 0.1, max = 0.6, step = 0.1, qfun="qunif")
5. "duration" = list(min=1, max = 6, step = 2, qfun="qunif")

Contents

nlrx script available in "Model_v1.1_FullFact_large.R" and "Model_v1.1_FullFact_small.R" in this link

The *five* conclusions are:

1. Feeding bout parameterization does not seem to be needed for Guareí.
2. Without the feeding bout parametrization, agents usually hit higher levels of energy. This might be the reason why DPL is lightly higher in these runs.
3. But runs with parameterized feeding bout ("on") varied more in output (home range and daily path length)
4. Parameters (inputs) that showed consistent trend on variables (outputs):

- step revisit on HR & DPL (direct but weak relationship)
- p foraging on DPL (inverse strong relationship)
- duration on DPL (inverse strong relationship)

5. Simulated home ranges have a mean of 20 hectares. I still haven't checked monthly empirical home range sizes of Guareí, but the size for all the fours months is ~50 ha.

Data

Read and clean

```r
# Read data from experiments
nl_off_s <- readRDS(here("Model_analysis/Sensitivity-analysis/temp/Model_v1.1_Sensitivit
data_off_s <- unnest_simoutput(nl_off_s) ; rm(nl_off_s)

data_off_s <- data_off_s %>%
  mutate(feedingbout = "Feeding bout off")

nl_on_s <- readRDS(here("Model_analysis/Sensitivity-analysis/temp/Model_v1.1_Sensitivity
data_on_s <- unnest_simoutput(nl_on_s) ; rm(nl_on_s)

data_on_s <- data_on_s %>%
  dplyr::select(-`phenology-on?`) %>%
  mutate(feedingbout = "Feeding bout on")

# Bind all together
db <- bind_rows(data_on_s, data_off_s)
rm(data_on_s); rm(data_off_s)

# Last cleaning:
 # Remove - and space characters:
db <- db %>% dplyr::rename_all(~str_replace_all(., c("-" = "_",
                                                     "\\." = "_",
                                                     " " = ""))) %>%
  rename("p_foraging" = "p_foraging_while_traveling",
         x = "x_UTM",
         y = "y_UTM") %>%
  dplyr::select(c(-agent, -breed))

# names(db)
unique(db$random_seed) # 10 seeds, 5 seeds for on and 5 seeds for off
```

```
##  [1] -1009315994  -588459717  -885863143   621018773 -1580201419 -1443516878
##  [7] -1375739275  1005320067  -848484702 -1154381652
```

```r
# Group and define unique id
db <- db %>% group_by(random_seed, siminputrow, feedingbout,
                      step_forget, visual, p_foraging,
                      duration, day) %>%
  mutate(id = paste0("GuaSim_", cur_group_id())) # This creates id groups = day for eac
```

Check parameter variation (input)

Table 1: Input parameters for Full Factorial Sensitivity

Parameter	Values
step_revisit	1.00, 31.00, 61.00, 91.00, 121.00, 151.00
p_foraging	0.10, 0.30, 0.50
duration	1.00, 3.00, 5.00
visual	1.00, 2.00, 3.00

Calculate DPL

```
# Transform dist_traveled from patch to meters (script Scale_Netlogo.R)
db <- db %>%
  mutate(dist_traveled = case_when(dist_traveled == 1 ~ 28.28803,
                                   dist_traveled == 0.7 ~ 19.80162,
                                   dist_traveled == 0.0 ~ 0))
# Sum up DPL
db <- db %>%
  group_by(id) %>%
  mutate(DPL = sum(dist_traveled)) %>%
  ungroup() %>% as_tibble() # the tbl has to be ungrouped for the amt pckage nest() fun
```

Calculate HR

```
# amt vignette: https://cran.r-project.org/web/packages/amt/vignettes/p1_getting_start


# For a specific group:
a1 <- db %>% filter(id == "GuaSim_19") %>%
  make_track(.x=x, .y=y, id = id, crs = our_crs)

a1
kde1 <- a1 %>%
  hr_kde(levels = c(0.3, 0.95))
```

```r
# kde1$h
# kde1$estimator
# kde1$ud
# kde1$h

plot(kde1)
amt::hr_isopleths(kde1)

hr_area(kde1) %>%
  mutate(area_ha = area / 10000)


# For all groups (require dplyr::nest())
db_nest <- db %>%
  make_track(.x=x, .y=y, id = id, crs = our_crs) %>%
  nest(data = -c(id)) # group only by id or use: nest(data = c(x_, y_, t_, var1))

  # example run:
db_nest$data[[1]]

  # calculate HR metrics for every list (=id = run) using map()
db_nest <- db_nest %>%
  mutate(
    KDE95 = map(data, hr_kde),
    KDE50 = map(data, hr_kde)
  )

db_nest <- db_nest %>%
  select(-data) %>% # drop all track data, we don't need it anymore (it was used to ca
  pivot_longer(KDE95:KDE50, names_to = "KDE_value", values_to = "hr")

db_nest <- db_nest %>%
  mutate(hr_area = map(hr, hr_area)) %>%
  unnest(cols = hr_area)

# db_nest %>%
  # select(-what) %>%
  # select(-hr)

# Deu ruim no KD_50 (ficou igual ao 95)
db_nest <- db_nest %>% filter(KDE_value == "KDE95")

db_nest <- db_nest %>%
  dplyr::select(-c(3, 4)) # does not work by collumn name
```

```r
# Merge HR to db and save
db <- left_join(db, db_nest)
db <- db %>%
  mutate(hr_area_ha = area / 10000)

# I only calculated KDE95, so I'll simplify the collums
db <- db %>%
  select(-c(KDE_value, area)) %>%
  rename(KDE95 = hr_area_ha)

# Save data imediatly because it is very memory consuming:
# db %>% write.csv(file=here("Model_analysis", "Sensitivity-analysis",
# "temp", "HR_values_FF_2022-08-06d.csv"))
```

As I didn't use href values in the function hd_kde(), the home range values might be smaller. I ran analysis with h=30 and h=60 before this script was lost with a crash on August 03rd 2022, but the results didn't change too much. So the reason why the home range is so small for runs (compare with runs with speed_val) might be because there's something wrong in the model or because in the speedval runs I was using the home range of all months for validation.

Load data

The functions to estimate HR take to much time, so we will read the saved .csv (chunks above are with eval=F)

```r
db <- read_csv(here("Model_analysis", "Sensitivity-analysis",
                    "temp", "HR_values_FF_2022-08-06d.csv")) %>%
  select(-1)
```

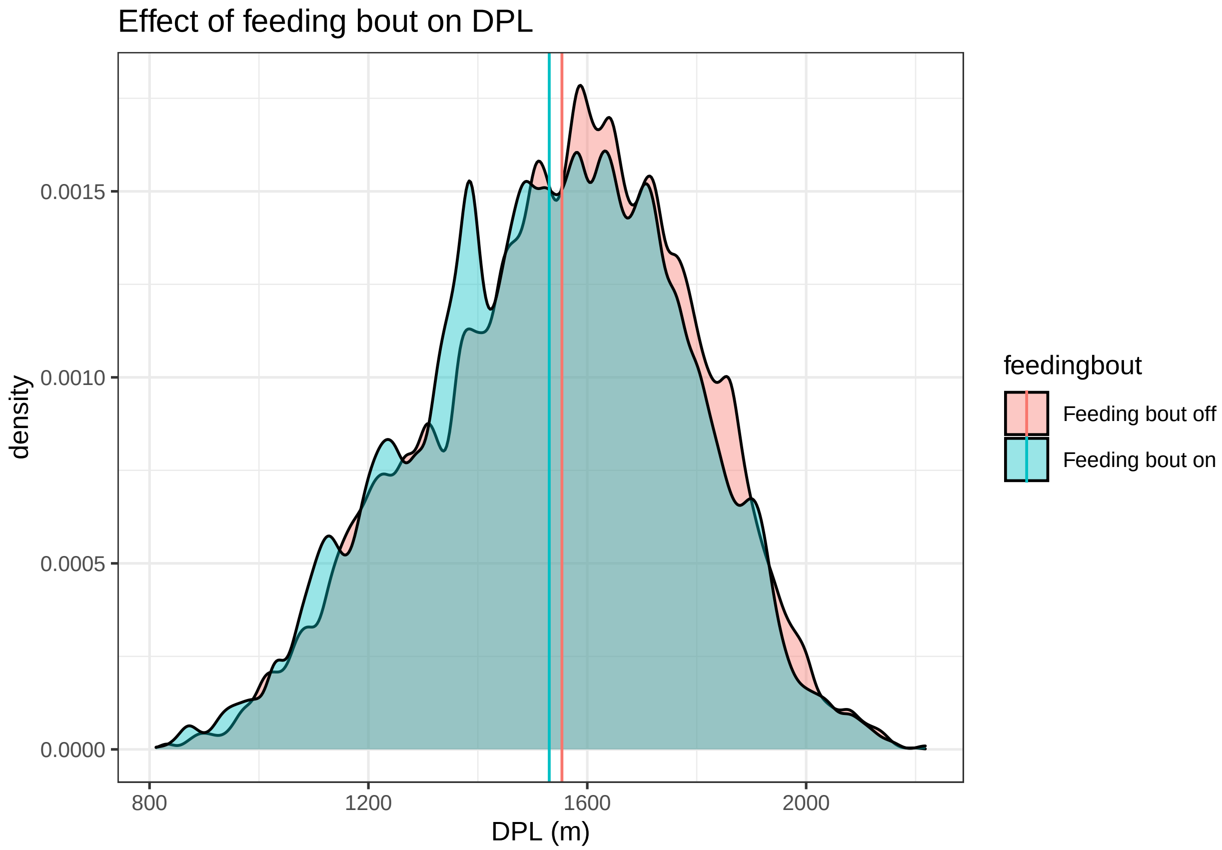
```
## New names:
## Rows: 1710727 Columns: 21
## -- Column specification
## -------------------------------------------------------- Delimiter: "," chr
## (4): behavior, travel_mode, feedingbout, id dbl (17): ...1, [runnumber],
## step_forget, visual, p_foraging, duration, rand...
## i Use `spec()` to retrieve the full column specification for this data. i
## Specify the column types or set `show_col_types = FALSE` to quiet this message.
## * `` -> `...1`
```

```r
# Transform params into factors
  ### if you don't make it a factor, ggplot does not plot side by side
# params <- c("step_forget", "visual", "p_foraging", "duration")
db <- db %>%
  mutate_if(is.character, as.factor) %>%
  mutate(across(2:5, as.factor))
# str(db)
```

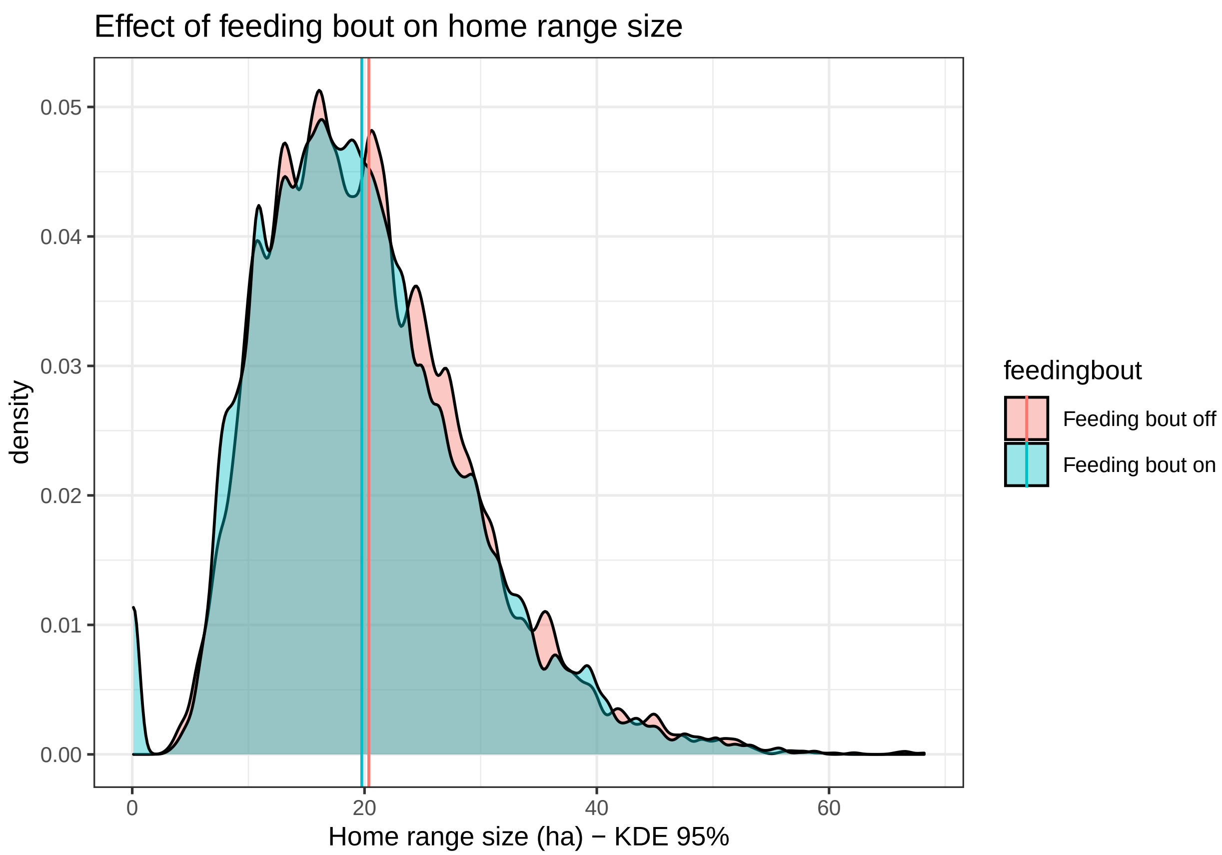
Effect of feeding bout

Apparently smaller changes in DPL (~60 m longer when feeding bout is off) and no changes in Home range

```r
# DPL
DPL_means_df <- db %>%
  group_by(feedingbout) %>%
  summarise(meanDPL = mean(DPL))

db %>%
  ggplot(aes(x=DPL, fill = feedingbout)) +
  geom_density(alpha=0.4) +
  xlab("DPL (m)") +
  ggtitle("Effect of feeding bout on DPL") +
  geom_vline(data=DPL_means_df, aes(xintercept = meanDPL,
                                    color = feedingbout))
```



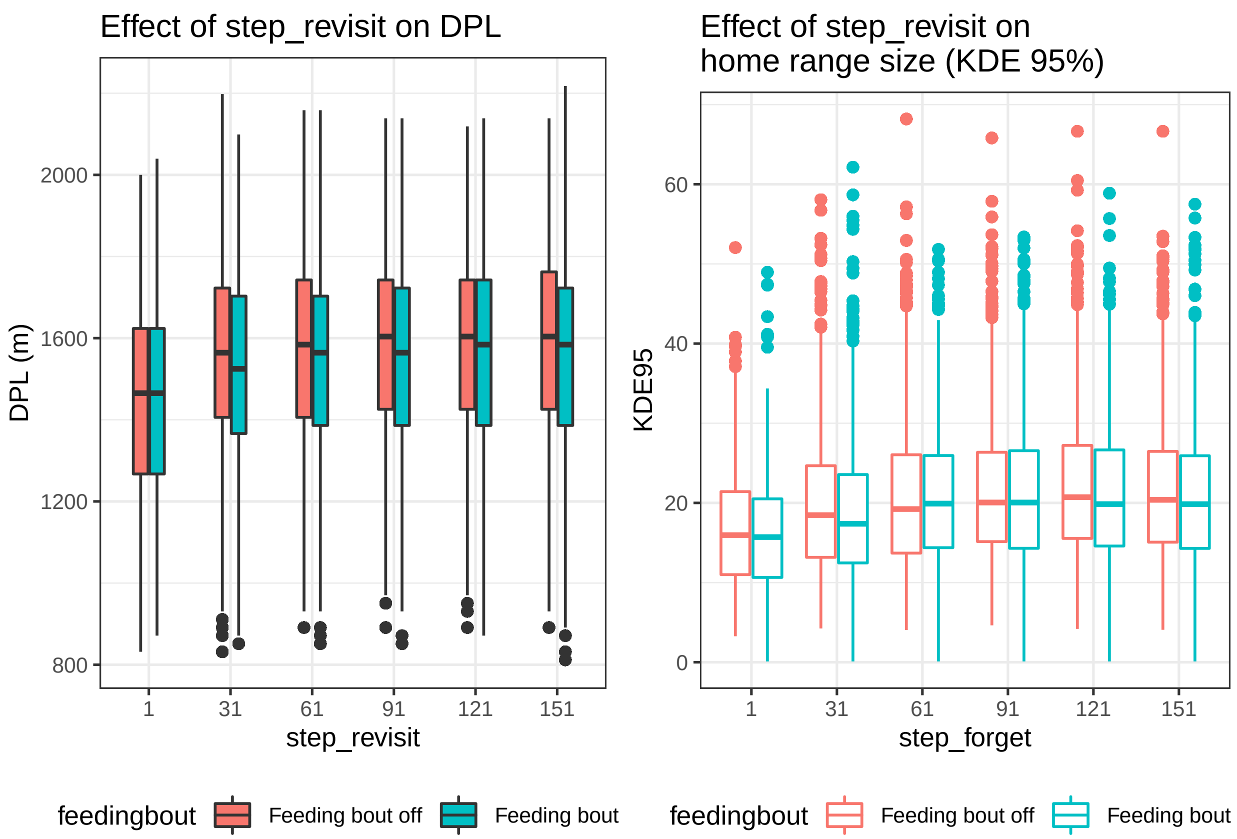

```
# Home range
HR_means_df <- db %>%
  group_by(feedingbout) %>%
  summarise(meanHR = mean(KDE95))

db %>%
  ggplot(aes(x=KDE95, fill = feedingbout)) +
  geom_density(alpha=0.4) +
  xlab("Home range size (ha) - KDE 95%") +
  ggtitle("Effect of feeding bout on home range size") +
  geom_vline(data=HR_means_df, aes(xintercept = meanHR,
                                    color = feedingbout))
```



Effect of step_revisit

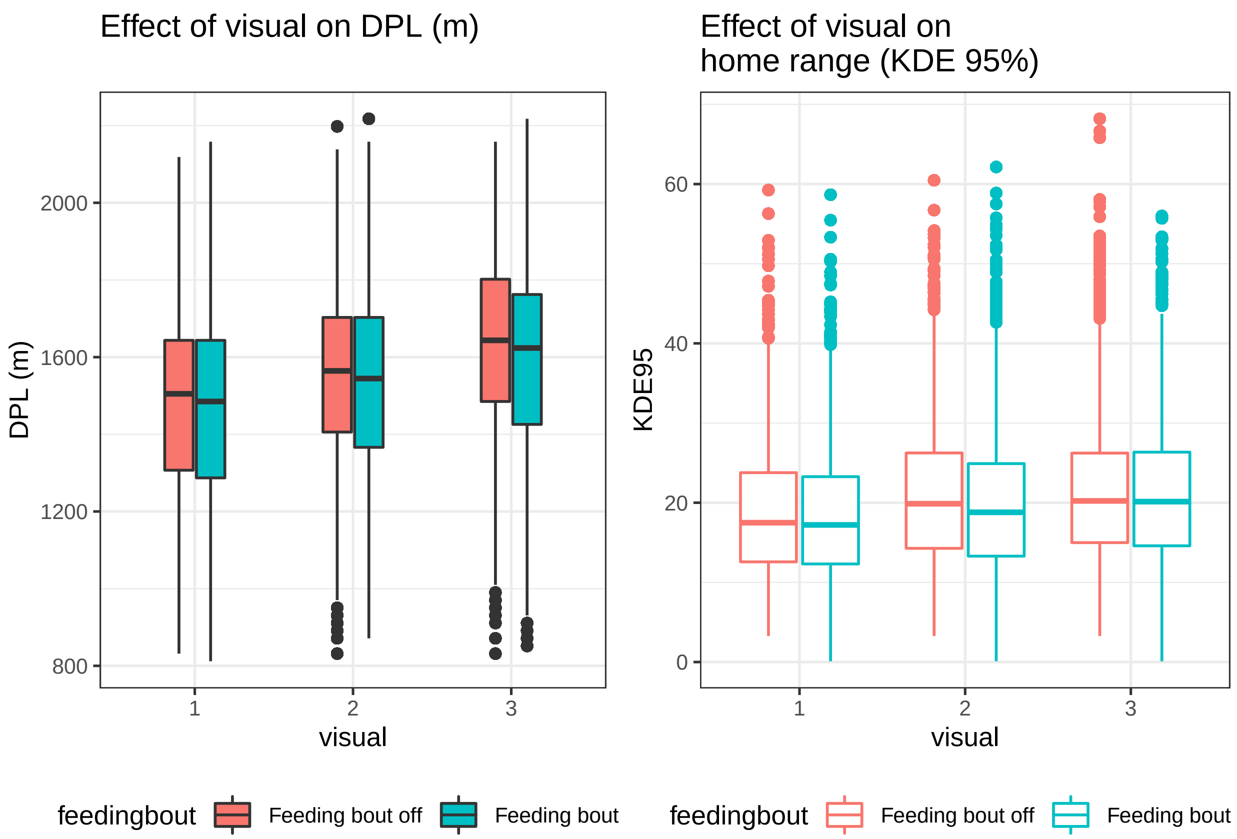
```
# db %>%
#   group_by(feedingbout, step_forget, random_seed) %>%
#   dplyr::summarise(nruns = n())

# db$siminputrow %>% summary() #number of runs


# DPL
p1 <- db %>%
  ggplot(aes(x=step_forget, y=DPL, fill = feedingbout)) +
  ylab("DPL (m)") +
  geom_boxplot(width = 0.4) +
  ggtitle("Effect of step_revisit on DPL") +
  xlab("step_revisit") +
  theme(legend.position = "bottom")
  # facet_wrap(~feedingbout)
```

```
# Home range
p2 <- db %>%
  ggplot(aes(x=step_forget, y=KDE95, color = feedingbout)) +
  # facet_wrap(~feedingbout) +
  geom_boxplot() +
  ggtitle("Effect of step_revisit on\nhome range size (KDE 95%)") +
  theme(legend.position = "bottom")


gridExtra::grid.arrange(arrangeGrob(p1, p2,
                                    ncol = 2, nrow = 1))
```



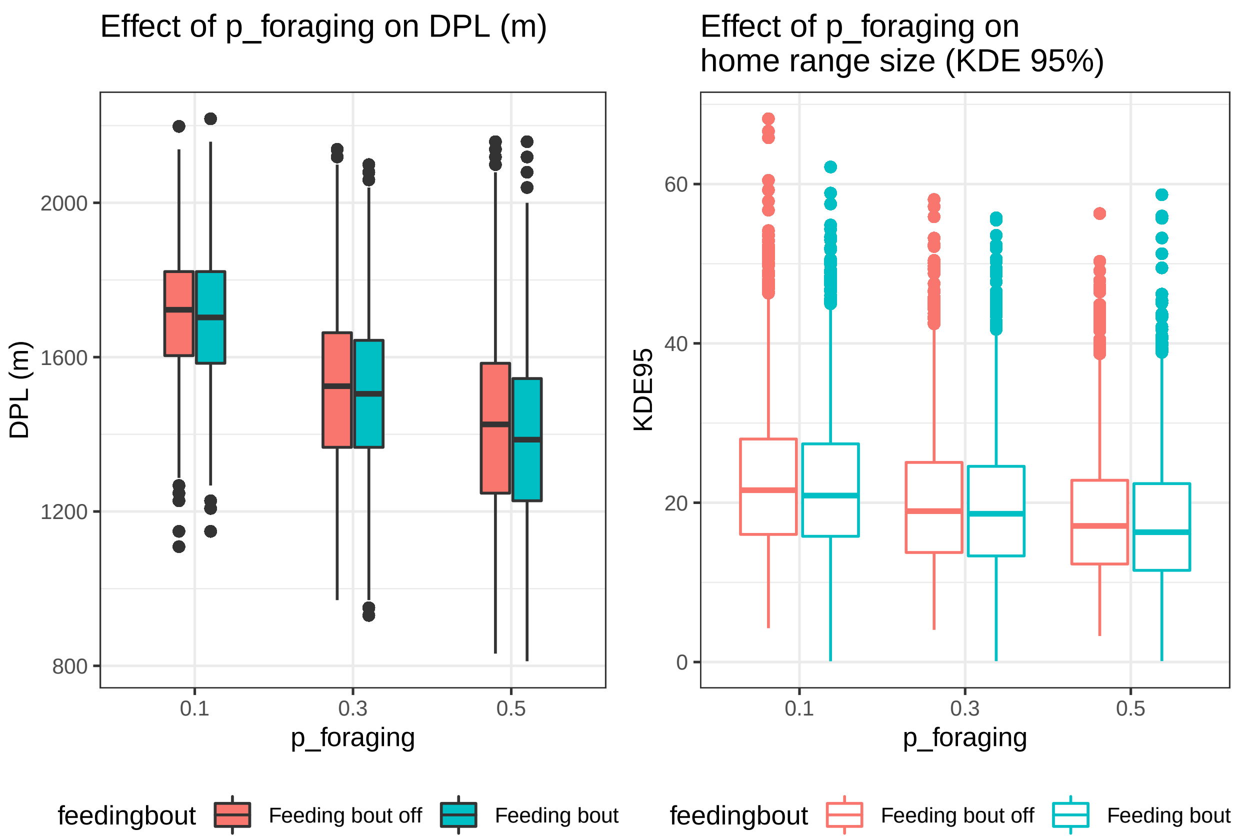
Effect of visual

```r
# DPL
p3 <- db %>%
  ggplot(aes(x=visual, y=DPL, fill = feedingbout)) +
  ylab("DPL (m)") +
  geom_boxplot(width = 0.4) +
  ggtitle("Effect of visual on DPL (m)\n") +
  theme(legend.position = "bottom")
  # theme(legend.position = "none")

  # facet_wrap(~feedingbout)


# Home range
p4 <- db %>%
  ggplot(aes(x=visual, y=KDE95, color = feedingbout)) +
  # facet_wrap(~feedingbout) +
  geom_boxplot() +
  ggtitle("Effect of visual on\nhome range (KDE 95%)") +
  theme(legend.position = "bottom")

# p3; p4

gridExtra::grid.arrange(arrangeGrob(p3, p4,
                                    ncol = 2, nrow = 1))
```


Effect of visual on DPL (m)
DPL (m)
2000
1600
1200
800
1
2
3
visual
feedingbout
Feeding bout off
Feeding bout
Effect of visual on
home range (KDE 95%)
KDE95
60
40
20
0
1
2
3
visual
feedingbout
Feeding bout off
Feeding bout c

Effect of p_foraging

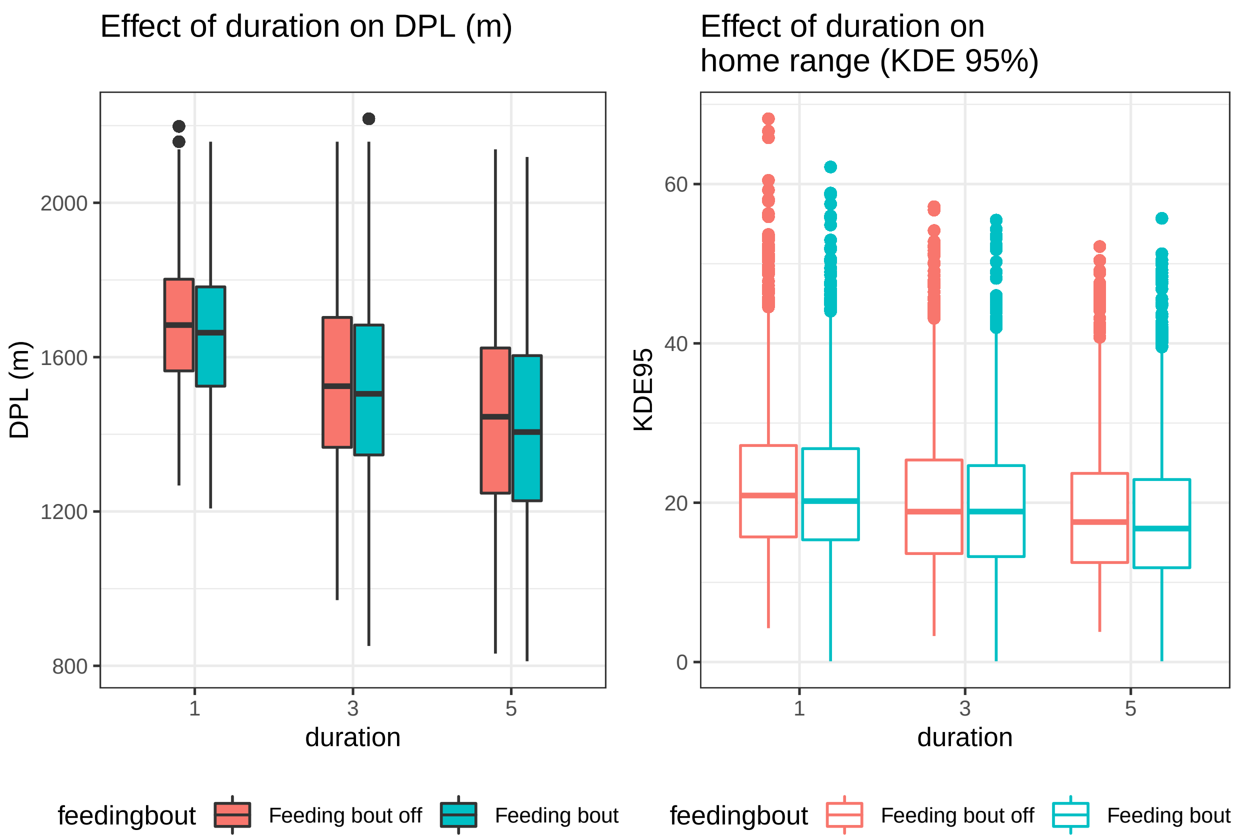
```r
# DPL
p5 <- db %>%
  ggplot(aes(x=p_foraging, y=DPL, fill = feedingbout)) +
  ylab("DPL (m)") +
  geom_boxplot(width = 0.4) +
  ggtitle("Effect of p_foraging on DPL (m)\n") +
  theme(legend.position = "bottom")

  # facet_wrap(~feedingbout)


# Home range
p6 <- db %>%
  ggplot(aes(x=p_foraging, y=KDE95, color = feedingbout)) +
  # facet_wrap(~feedingbout) +
  geom_boxplot() +
  ggtitle("Effect of p_foraging on\nhome range size (KDE 95%)") +
  theme(legend.position = "bottom")



gridExtra::grid.arrange(arrangeGrob(p5, p6,
                                    ncol = 2, nrow = 1))
```



Effect of p_foraging on DPL (m)
DPL (m)
2000
1600
1200
800
0.1
0.3
0.5
p_foraging
Effect of p_foraging on
home range size (KDE 95%)
KDE95
60
40
20
0
0.1
0.3
0.5
p_foraging
feedingbout
Feeding bout off
Feeding bout
feedingbout
Feeding bout off
Feeding bout c

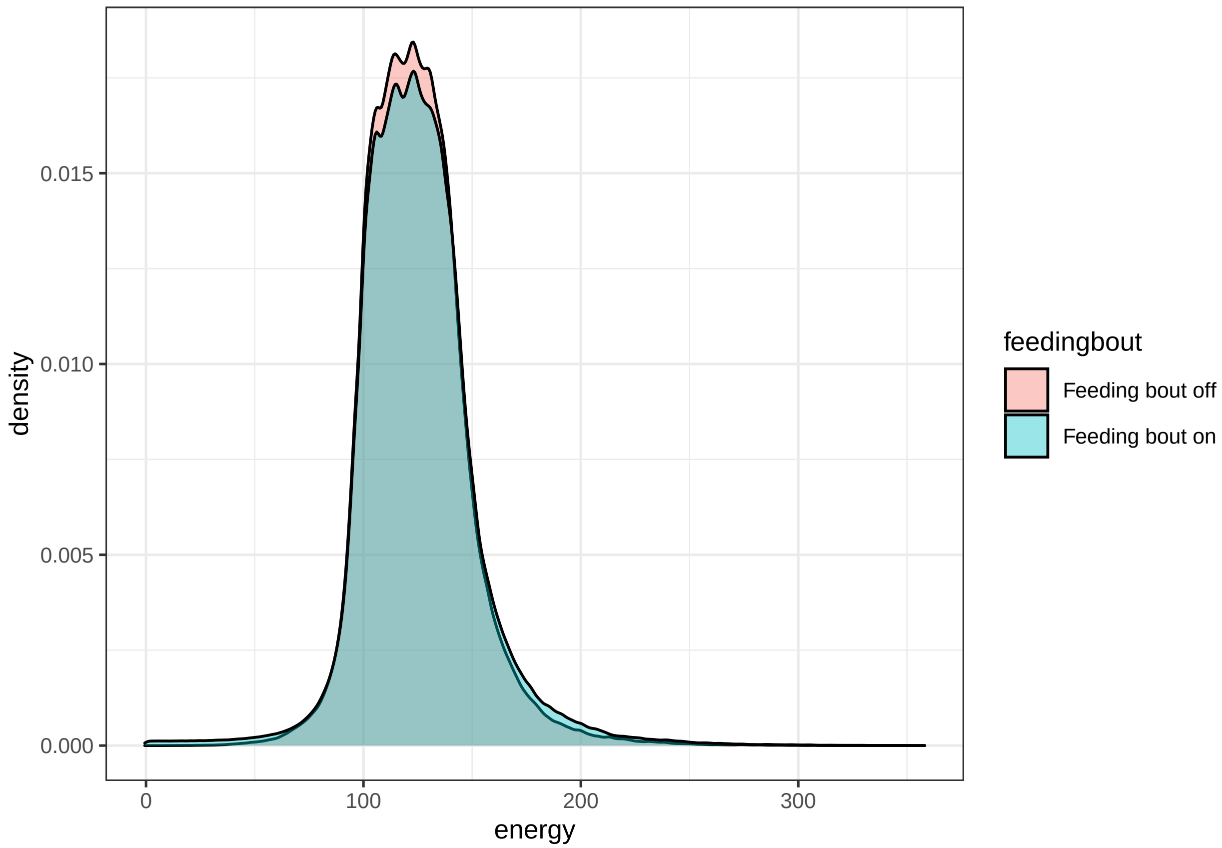
Effect of duration

```
# DPL
p7 <- db %>%
  ggplot(aes(x=duration, y=DPL, fill = feedingbout)) +
  geom_boxplot(width = 0.4) +
  ylab("DPL (m)") +
  ggtitle("Effect of duration on DPL (m)\n") +
  theme(legend.position = "bottom")

  # facet_wrap(~feedingbout)


# Home range
p8 <- db %>%
  ggplot(aes(x=duration, y=KDE95, color = feedingbout)) +
  # facet_wrap(~feedingbout) +
  geom_boxplot() +
  ggtitle("Effect of duration on\nhome range (KDE 95%)") +
  theme(legend.position = "bottom")


gridExtra::grid.arrange(arrangeGrob(p7, p8,
                                    ncol = 2, nrow = 1))
```


Effect of duration on DPL (m)
DPL (m)
2000
1600
1200
800
1
3
5
duration
feedingbout
Feeding bout off
Feeding bout
Effect of duration on
home range (KDE 95%)
KDE95
60
40
20
0
1
3
5
duration
feedingbout
Feeding bout off

Effect on survival

```r
# Some runs finished with low energy (probably the agents died)
db %>%
  ggplot(aes(x=energy, fill = feedingbout)) +
  geom_density(alpha=0.4)
```




```r
set.seed(173)
db %>% #group_by(siminputrow) %>%
  filter(energy < 5) %>%
  sample_n(10) %>%
  flextable()
```

[runnumber]	step_forget	visual	p_foraging	duration	random_seed	[step]	timestep	day	
1	121	3	0.1	1	-1,580,201,419	68	68	1	78

[runnumber]	step_forget	visual	p_foraging	duration	random_seed	[step]	timestep	day	
1	121	2	0.3	5	-1,580,201,419	70	70	1	78
1	151	2	0.3	3	-1,580,201,419	69	69	1	78
1	61	1	0.5	3	-1,580,201,419	70	70	1	78
1	91	3	0.1	3	-1,580,201,419	68	68	1	78
1	31	1	0.1	5	-1,580,201,419	67	67	1	78
1	31	2	0.1	5	-1,580,201,419	67	67	1	78
1	91	2	0.3	3	-1,580,201,419	67	67	1	78
1	1	3	0.3	3	-1,580,201,419	69	69	1	78
1	151	1	0.3	5	-1,580,201,419	68	68	1	78

Most of the agents that died are within visual >= 2, but not necessarily. This means these values are too high If monkeys were restricted and without resources, they'd probably come back to the previous one up to total fruit depletion. This is process of prefering resources not recently visited does not include a mechanism of ignoring it in case the energy is low. I wonder if making agents dead really makes sense. Maybe restoring the seed_pot_list to its original state would be the most relistic way.

Another thing to notice: energy usually goes to 0 when the agents are in the same point (look to x and y). I'll still investigate this.

Further analysis (subset)

Up to now I have plotted all the simulations in relation to one unique variable. Now, we will plot runs while keeping the other variables constant

```
theme_set(theme_bw(base_size = 5))
```

Step revisit with data subset

```
# Subset
db_ss <- db %>%
  filter(p_foraging == val_p_foraging[1] &
           # step_forget == val_step_forget[1] &
           duration == val_duration[1] &
           visual == val_visual[1])

p1 <- db_ss %>%
  ggplot(aes(x=step_forget, y=DPL, fill = feedingbout)) +
  # ylab("DPL (m)") +
  geom_boxplot(width = 0.4) +
  xlab("step_revisit") +
  theme(legend.position = "none")

# Home range
p2 <- db_ss %>%
  ggplot(aes(x=step_forget, y=KDE95, color = feedingbout)) +
  # facet_wrap(~feedingbout) +
  geom_boxplot()  +
  xlab("step_revisit") +
  theme(legend.position = "none")#+
  # ggtitle("Home range (KDE 95%) by step_revisit (data subset)")

# p1; p2
```

Visual with data subset

```
# Subset
db_ss <- db %>%
  filter(p_foraging == val_p_foraging[1] &
           step_forget == val_step_forget[1] &
           duration == val_duration[1] #&
```

```
          # visual == val_visual[1]
        )

p3 <- db_ss %>%
  ggplot(aes(x=visual, y=DPL, fill = feedingbout)) +
  ylab("DPL (m)") +
  geom_boxplot(width = 0.4) +
  theme(legend.position = "bottom")

# Home range
p4 <- db_ss %>%
  ggplot(aes(x=visual, y=KDE95, color = feedingbout)) +
  # facet_wrap(~feedingbout) +
  geom_boxplot() +
  theme(legend.position = "bottom") #+
  # ggtitle("Home range (KDE 95%) by visual (data subset)")
```

p_foraging with data subset

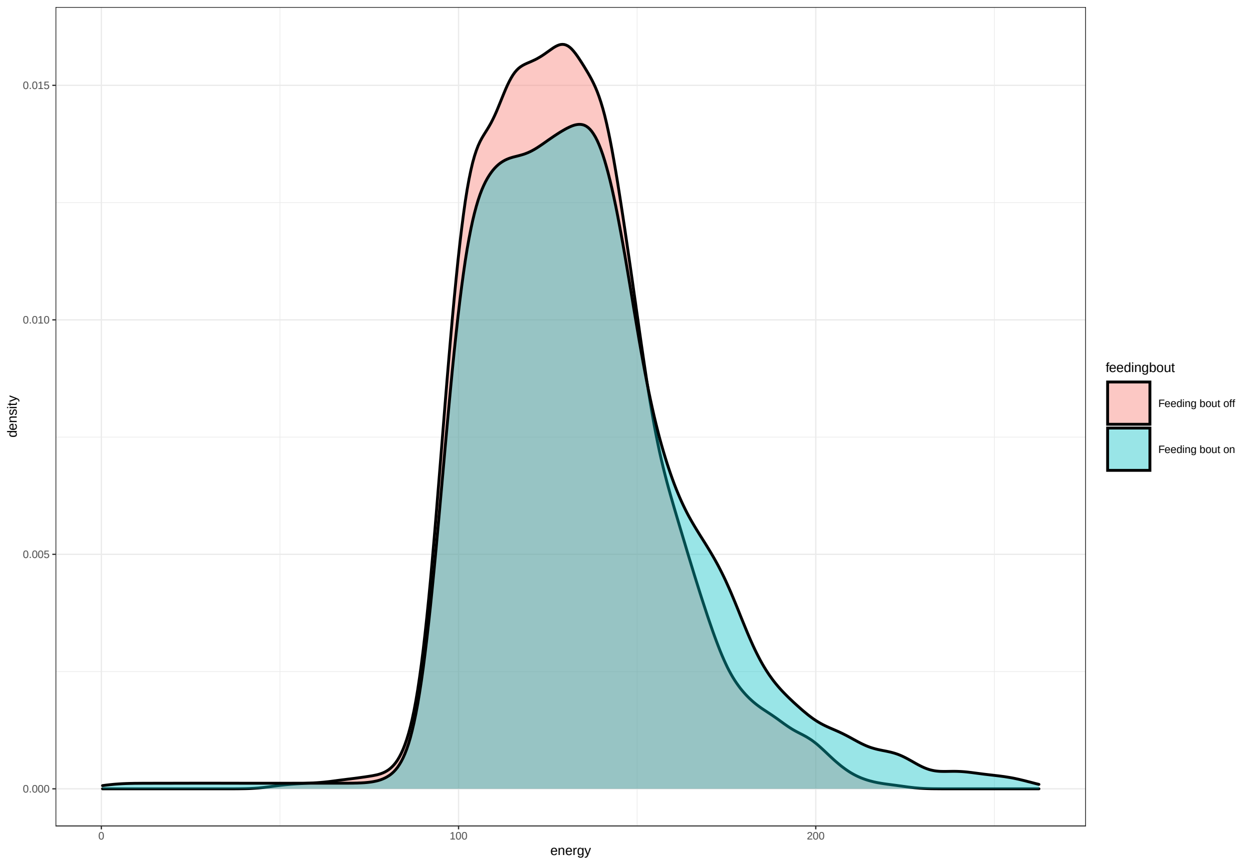
```
# Subset
db_ss <- db %>%
  filter(#p_foraging == val_p_foraging[1] &
           step_forget == val_step_forget[1] &
           duration == val_duration[1] &
           visual == val_visual[1]
        )

p5 <- db_ss %>%
  ggplot(aes(x=p_foraging, y=DPL, fill = feedingbout)) +
  ylab("DPL (m)") +
  geom_boxplot(width = 0.4) +
  theme(legend.position = "none")


# Home range
p6 <- db_ss %>%
  ggplot(aes(x=p_foraging, y=KDE95, color = feedingbout)) +
  # facet_wrap(~feedingbout) +
  geom_boxplot() +
  theme(legend.position = "none")#+
  # ggtitle("Home range (KDE 95%) by p_foraging (data subset)")
```

Duration with data subset

```
# Subset
db_ss <- db %>%
  filter(p_foraging == val_p_foraging[1] &
           step_forget == val_step_forget[1] &
           # duration == val_duration[1] &
           visual == val_visual[1]
         )

p7 <- db_ss %>%
  ggplot(aes(x=duration, y=DPL, fill = feedingbout)) +
  ylab("DPL (m)") +
  geom_boxplot(width = 0.4) +
  theme(legend.position = "bottom")

# Home range
p8 <- db_ss %>%
  ggplot(aes(x=duration, y=KDE95, color = feedingbout)) +
  # facet_wrap(~feedingbout) +
  geom_boxplot() +
  theme(legend.position = "bottom") #+
  # ggtitle("Home range (KDE 95%) by duration (data subset)")
```

Energy with data subset

```
# Let's check the effect with less noise of parameter variation:
db_ss <- db %>%
  filter(p_foraging == val_p_foraging[1] &
           step_forget == val_step_forget[1] &
           duration == val_duration[1] &
           visual == val_visual[1]
         )

db_ss %>%
  ggplot(aes(x=energy, fill = feedingbout)) +
  geom_density(alpha=0.4) +
  theme_set(theme_bw(base_size = 10))
```

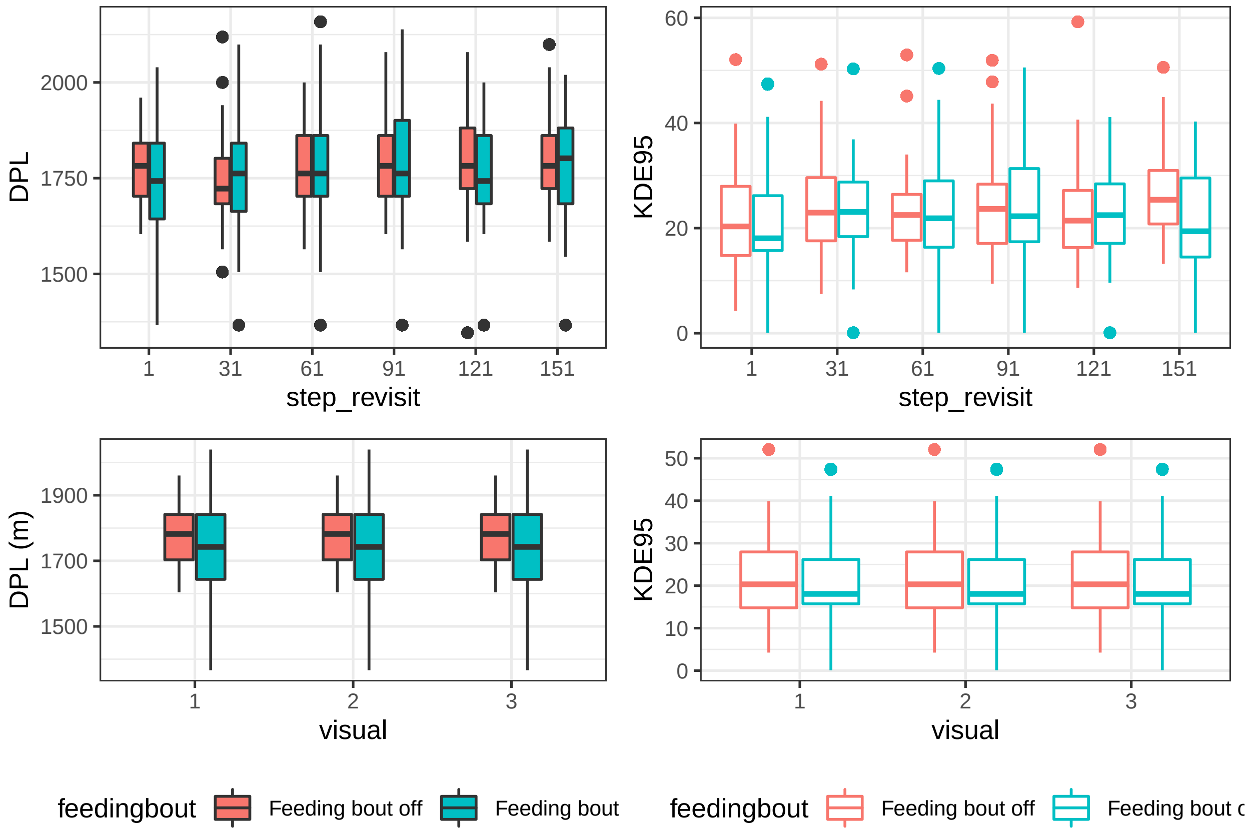



```r
# db_ss %>% group_by(siminputrow) %>%
#   filter(energy < 1) %>%
#   flextable()
```

It seems that when feeding bout is on, agents slightly tend to keep higher levels of energy, but I would not say this effect is big. Without the feeding bout parametrization, agents usually hit higher levels of energy. This might be the reason why DPL is higher in these situations.

```r
# Remove axis titles from all plots
# p = list(p1, p2, p3, p4) %>% map(~.x + labs(x=NULL, y=NULL)) %>% map(~.x + ggtitle(N
# q = list(p5, p6, p7, p8) %>% map(~.x + labs(x=NULL, y=NULL)) %>% map(~.x + ggtitle(N

# Same yaxis every grob https://community.rstudio.com/t/common-axis-title-in-grid-arra
gridExtra::grid.arrange(arrangeGrob(p1, p2, p3, p4,
                                    ncol = 2, nrow = 2) #,
                        # arrangeGrob(p9)
                        )
```

Plotting



DPL
step_revisit
KDE95
step_revisit
DPL (m)
visual
KDE95
visual
feedingbout
Feeding bout off
Feeding bout
feedingbout
Feeding bout off

```r
gridExtra::grid.arrange(arrangeGrob(p5, p6, p7, p8,
                                    ncol = 2, nrow = 2))
```