How resource abundance and stochasticity affect animals' space-use requirements

Appendix B: Simulations

1 Overview

This appendix illustrates all the steps necessary to produce the simulation figures in the main manuscript (figs. 3 and 4). To achieve full transparency while minimizing computational times, the code illustrated in this pdf is not executed during the knitting of the document. Instead, the R Markdown document (`writing/appendix-2-simulations.Rmd`) contains code chunks that import the `RDS` files saved by the scripts used during the analysis via `R` code that is not printed in the pdf file. Although one can replicate the analyses by running the code in this pdf, we suggest only using this document for illustrative purposes and as a general guide. We suggest using the `R` scripts to replicate the simulations, instead.

2 Simulating the movement tracks

To reduce sampling variance between simulations at each time point in each panel of fig. 3 in the main text, we used the same set of 200 simulated tracks, which we generate in the `analysis/simulations/tracks.R` script. Most intermediate and diagnostic checks are not included in this document for the sake of brevity and simplicity, but their outputs and conclusions are listed in this section. In this script, we use the `ctmm` package (version 1.1.0, Fleming and Calabrese 2021) for movement modeling; the `terra` package (version 1.7-39, Hijmans 2023) to work with the simulated resource rasters; the `dplyr` (version 1.0.10, Wickham et al. 2022), `purrr` (version 0.3.5, Henry and Wickham 2022), and `tidyr` (version 1.2.1, Wickham and Girlich 2022) packages for data wrangling; and the the `ggplot2` (version 3.4.0, Wickham 2016) and `cowplot` (version 1.1.1, Wilke 2020) packages for plotting.

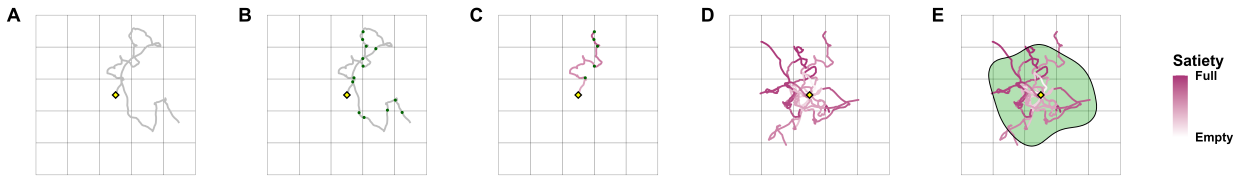


Figure B1: Overview of how animals' space-use requirements were simulated. (a.) Tracks were simulated using an Integrated Ornstein-Uhlenbeck model (IOU model, an infinitely diffusive and continous-velocity movement model), starting from the point $\langle 0, 0 \rangle$ (black and yellow square). (b.) Each time the track crossed into a new cell (green dots), the animal collected a random amount of resources that followed a Gamma distribution with common mean $\mu(t)$ and variance $\sigma^2(t)$. (c.) Each time the animal collected more resources, its satiety (purple) increased. Once the animal collected sufficient resources, the animal stopped exploring (i.e., the track was truncated) and was allowed to return to $\langle 0, 0 \rangle$ over the same amount of time needed to reach satiety. (D) The process was repeated 200 times (13 tracks pictured in this panel). (E) The final set of (truncated) tracks was then modeled using Ornstein-Uhlenbeck Foraging models to estimate the 95% home range estimates using Autocorrelated Kernel Density Estimates.

```r
# NOTE: change working directory to "hr-resource-stoch" or modify paths

# attach all necessary packages
library('ctmm')    # for generating movement models and movement modeling
library('terra')   # for working with raster data
library('dplyr')   # for data wrangling
library('purrr')   # for functional programming
library('tidyr')   # for data wrangling (e.g., nested tibbles)
library('mgcv')    # for empirical Bayes GAMs
library('ggplot2') # for fancy plots
library('cowplot') # for fancy multi-panel plots
library('dagitty') # for directed acyclical graphs
library('ggdag')   # for directed acyclical graphs
theme_set(theme_bw()) # change default theme

# source custom functions
source('functions/rgamma2.R') # rgamma() parameterized by mean and variance
source('analysis/mean-variance-trends-panel-data.R') # mu and sigma2
source('analysis/simulations/movement-model.R') # for consistency
source('functions/get_hr.R') # for extracting gaussian home range
source('functions/label_visits.R') # decides when animal encounters food
source('analysis/figures/default-figure-styling.R') # for color palette

DELTA_T <- 60 # sampling interval in seconds
SAMPLES <- seq(0, 60 * 60 * 12, by = DELTA_T) # 12 hours by DELTA_T seconds

# projected raster of resources
PROJECTION <- '+proj=aeqd +lon_0=0 +lat_0=0 +datum=WGS84'
HABITAT <- matrix(data = 1, nrow = 500, ncol = 500) %>%
  raster(xmx = 1e3, xmn = -1e3, ymx = 1e3, ymn = -1e3, crs = PROJECTION)

# infinitely diffusive movement model
model <- ctmm(tau = c(Inf, 1e3), sigma = 0.1, mu = c(0, 0))

N_DAYS <- 2^10 # number of "days" (i.e., tracks with different seeds)

# extracts tracks from a ctmm movement model for given sample times
get_tracks <- function(day, times = SAMPLES) {
  simulate(model, # ctmm movement model
           t = times, # sampling times in seconds
           seed = day, # for a consistent track each day
           complete = TRUE, # add lat, long, and timestamp to telemetry
           crs = PROJECTION) # CRS projection string
}
```

```r
# generate simulated tracks (will be truncated at satiety later)
tels <- tibble(day = 1:N_DAYS, # a simulation for each day
               tel = map(.x = day, # set a seed for consistent results
                         .f = get_tracks)) # function to generate tracks
tels
```

```
Warning: package 'ggdag' was built under R version 4.3.2
```

```
# A tibble: 1,024 x 2
     day tel
   <int> <list>
 1     1 <telemtry[,8]>
 2     2 <telemtry[,8]>
 3     3 <telemtry[,8]>
 4     4 <telemtry[,8]>
 5     5 <telemtry[,8]>
 6     6 <telemtry[,8]>
 7     7 <telemtry[,8]>
 8     8 <telemtry[,8]>
 9     9 <telemtry[,8]>
10    10 <telemtry[,8]>
# i 1,014 more rows
```

```r
# find patch visits and calories consumed from the tracks
tracks <- transmute(tels, # drop tel column
                    day, # keep day column
                    track = map(.x = tel, # add a column of full tracks
                                .f = \(x) {
                                  label_visits(.tel = x, .habitat = HABITAT)
                                }))
tracks <- tidyr::unnest(tracks, track) # make a single, large tibble
tracks
```

```
# A tibble: 738,304 x 11
     day     t       x      y        vx      vy    longitude    latitude
   <int> <dbl>   <dbl>  <dbl>     <dbl>   <dbl>        <dbl>       <dbl>
 1     1     0  0      0       0        0        0           0
 2     1    60  0.0363 0.0593 -0.000529 0.00271  0.000000326 0.000000536
 3     1   120  0.0722 0.395   0.000595 0.00755  0.000000648 0.00000358
 4     1   180  0.0276 0.967  -0.00517  0.0102   0.000000248 0.00000875
```

```
 5     1   240 -0.273  1.66   -0.00323  0.0130  -0.00000245  0.0000150
 6     1   300 -0.351  2.42    0.000977 0.00894 -0.00000315  0.0000219
 7     1   360 -0.221  2.92    0.00258  0.00534 -0.00000199  0.0000264
 8     1   420 -0.126  3.39    0.00182  0.00871 -0.00000113  0.0000307
 9     1   480  0.0258 3.74    0.00272  0.00354  0.000000231 0.0000338
10     1   540  0.138  4.07    0.00133  0.00783  0.00000124  0.0000368
# i 738,294 more rows
# i 3 more variables: timestamp <dttm>, cell_id <dbl>, new_cell <lgl>
```

After generating the tracks, we performed the following tests to ensure the number and length of the tracks were large enough for results to be stable. For the sake of conciseness, the code for each of the checks is not presented in this appendix, but it is available in the `R` scripts referenced in each section.

2.1 Checking whether adding return trips is necessary

Script: `analysis/figures/return-sensitivity.R`

Adding return trips to $\langle 0, 0\rangle$ after an animal reached satiety doubled computational times without appreciable improvements on the home range estimates (including the 95% confidence intervals of the estimates).

2.2 Checking whether the sampling interval is sufficiently small

Script: `analysis/figures/delta-t-sensitivity.R`

Using three tracks generated with three arbitrary seeds (1, 2, and 3), we explored the effects of sampling interval (Δt) on the number of encounters with food (i.e., movements to new cells) detected. From each of the four checks, we created an exploratory plot that we present in figure B2.

Exploratory plot B2a. The amount of time between encounters ranged from 1 second (the minimum sampling interval) to 24 minutes and 10 seconds. Approximately 93% of the

encounters (500/536, excluding the first 3 events of each track) occurred with 30 or more seconds between events.

Exploratory plot B2b. Halving the sampling interval had little to no effect on the total number of encounters for $\Delta t \lessapprox 60\,s$.

Exploratory plot B2c. A sampling interval of $\Delta t = 30$ seconds was small enough to capture fine-scale movement in the tracks but large enough to avoid excessive amounts of data and an inflated amount of encounters when an animal was near cell boundaries.

Exploratory plot B2d. The three tracks used for these exploratory plots are sufficiently different that we considered them to be a representative sample of movement tracks simulated by the OUF model. All three tracks have a reasonable amount of both tortuous and directed movement.

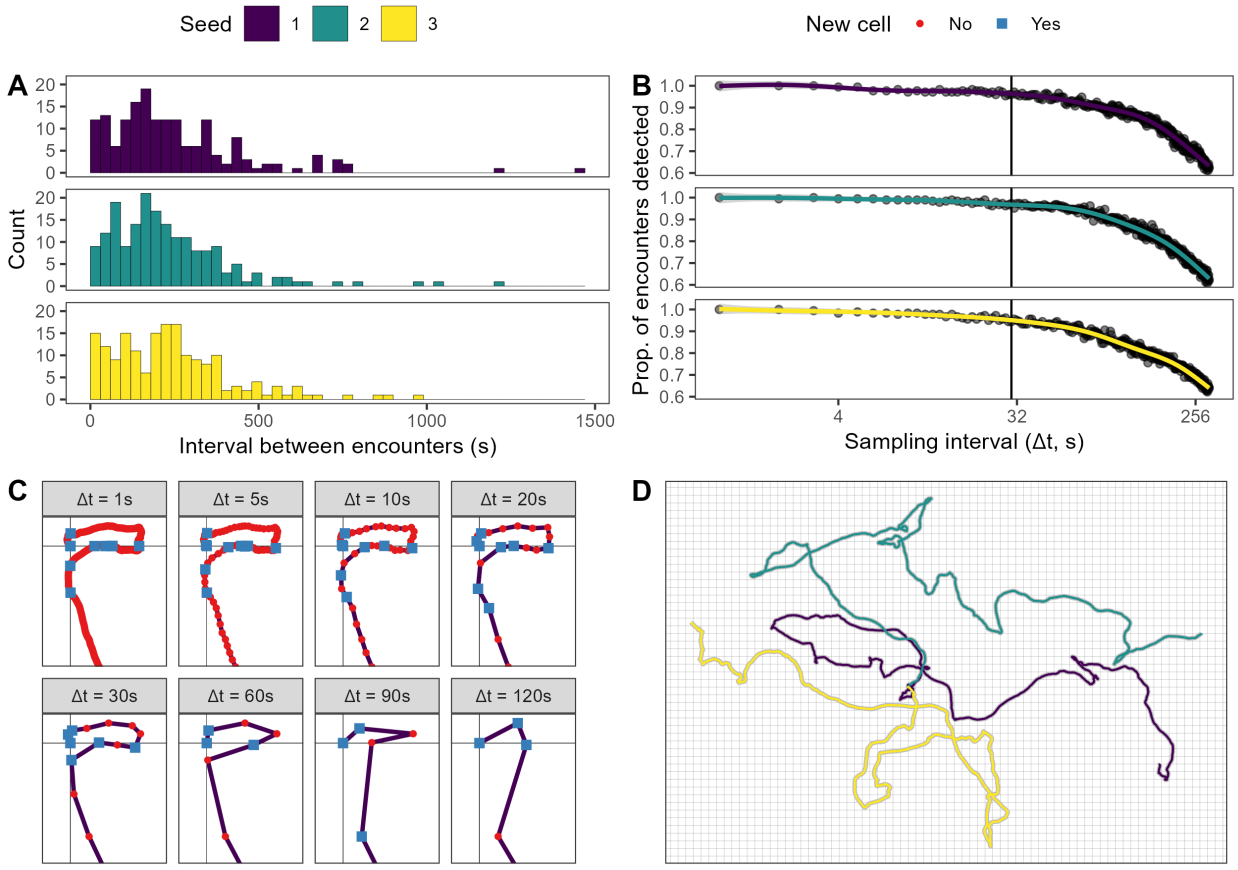


Figure B2: Exploratory plots used to decide an appropriate sampling interval. (A) Histograms of the number of encounters as a function of the interval between encounters, with a binwith of 30 seconds. Although some encounters occur with less than 30 seconds between them, approximately 93% of them occur at least 30 seconds apart. (B) Number of detected encounters as a function of sampling interval. The colored lines indicate the estimated relationship based on a Generalized Additive Model fit using the `geom_smooth` function from the `ggplot2` package. Although the number of encounters detected decreases as sampling interval doubles, the loss at $\Delta = 30$s is negligible. (C) Beginning of the track generated with seed "1" (purple line) for different sampling intervals. Red dots indicate locations where the animal remained in the same cell, while the blue squares indicate when an animal was in a new cell and thus encountered food. While the number of encounters detected decreases as the sampling interval increases, most of the encounters lost at $\Delta t = 30$s occured because the animal remained almost adjacent to the borders between cells. Additionally, the track at $\Delta t = 30$s is still sufficiently tortuous to represent realistic animal movement. (D) The three tracks used in these tests over the raster used for determining when the animals encountered food.

2.3 Checking how many tracks were necessary

Script: `analysis/simulations/2-hr-mean-variance-simulations-days.R`

The end of this script generates figures that show the mean time spent searching for resources before reaching satiety. This appendix includes the two figures that show the absolute and relative differences in time required to reach satiety as a function of the number of tracks. The estimated times for each number of replicates can be found in the `figures/5-by-5-sensitivity-analysis/` folder.

```
all <-
  tibble(d = map_dfr(
    list.files('simulations', pattern = '*-replicates.rds',
               full.names = TRUE),
    function(fn) {
      readRDS(fn) %>%
        mutate(mean = paste(mean, 'mean'),
               variance = paste(variance, 'var'),
               replicates = max(day)) %>%
        group_by(mean, variance, animal, replicates) %>%
        summarize(est = mean(t_expl),
                  .groups = 'drop')
    })) %>%
  unnest(d) %>%
  mutate(est = est / (1 %#% 'hours')) %>%
  pivot_wider(names_from = replicates, values_from = est) %>%
  pivot_longer(cols = as.character(2^(4:9)), values_to = 'est',
               names_to = 'replicates') %>%
  mutate(diff_est = est - `1024`,
         rel_est = est / `1024`,
         replicates = factor(replicates, levels = 2^(4:9)))

ggplot(all) +
  facet_wrap(~ replicates) +
  geom_line(aes(animal, diff_est, group = paste(mean, variance)),
            alpha = 0.1) +
  geom_hline(yintercept = 0, color = 'red') +
  labs(x = 'Time',
       y = 'Difference in average daily exploration time (hours)')
```

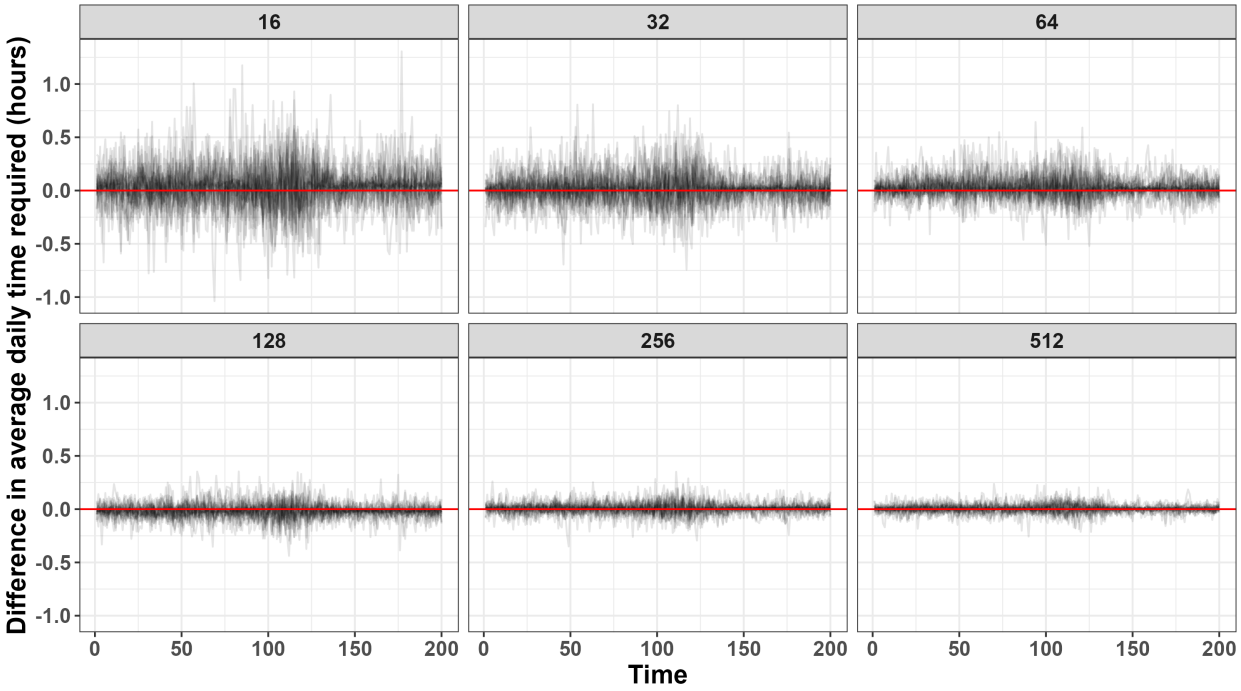


Figure B3: Difference in average time required to reach satiety for each scenario in figure 1 (indicated by each line) and different amounts of tracks (indicated at the top of each facet). The differences are relative to simulations with $2^{10} = 1024$ tracks for each time point on the x axis (and each scenario in figure 1). The amount of time stabilizes for $\gtrapprox 100$ tracks, which suggests that the home range estimates should also be stable with more than 100 tracks.

```
ggplot(all) +
  facet_wrap(~ replicates) +
  geom_line(aes(animal, rel_est, group = paste(mean, variance)),
            alpha = 0.1) +
  geom_hline(yintercept = 1, color = 'darkorange') +
  scale_y_continuous(
    trans = 'log2', breaks = 2^seq(-1, 1, by = 0.5),
    labels = parse(text = paste0('2^', seq(-1, 1, by = 0.5))))+
  labs(x = 'Time',
       y = expression(paste('Relative difference from estimates with',
                            '1024 replicates'~(log[2]~scale))))
```

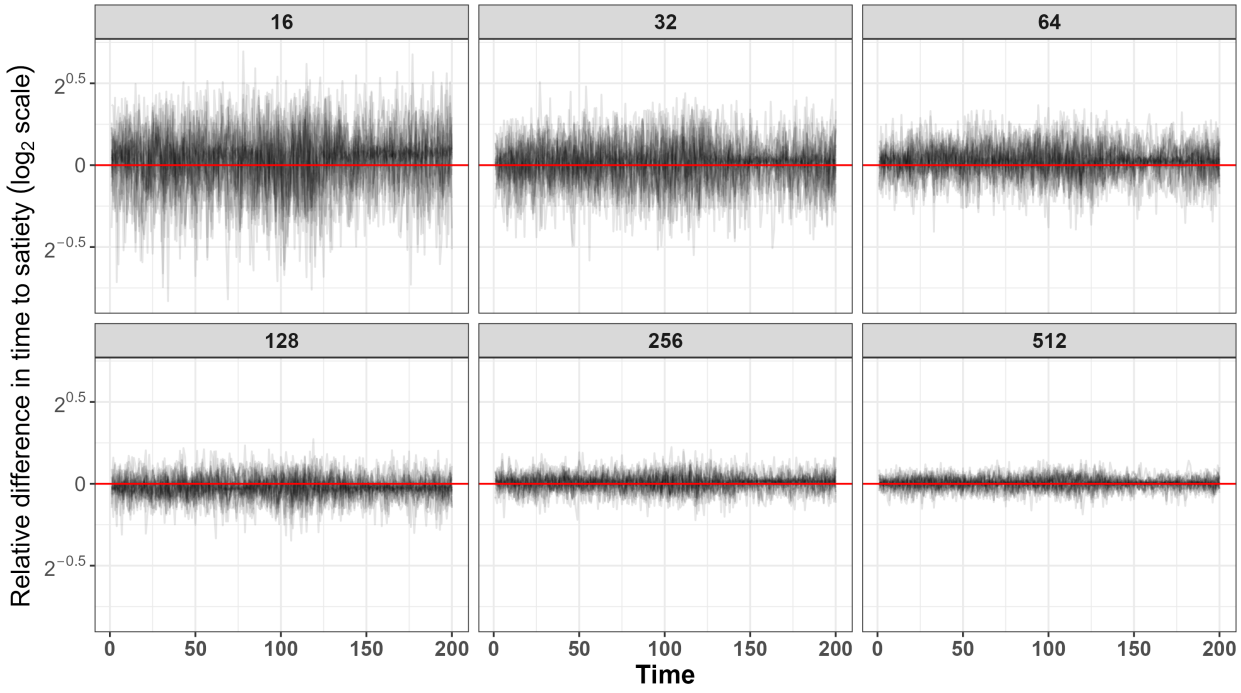


Figure B4: Ratios of average time required to reach satiety for each scenario in figure 1 (indicated by each line) and different amounts of tracks (indicated at the top of each facet). The ratios are relative to simulations with $2^{10} = 1024$ tracks for each time point on the x axis (and each scenario in fig. 1). The amount of time stabilizes for $\gtrapprox 100$ tracks, which suggests that the home range estimates should also be stable with more than 100 tracks.

Script: `analysis/hr-simulation-extreme-scenarios.R`

In this script, we check how many tracks are necessary to produce stable and accurate estimates of the space-use requirements. We do so by estimating the home ranges of animals for varying numbers of track in the best-case scenario (highest E(R) and lowest Var(R)) and worst-case scenario (lowest E(R) and highest Var(R)).

```
set.seed(1) # for consistent results
tels <- readRDS('simulations/tracks.rds') # list of telemetry tracks
tracks <- readRDS('simulations/labelled-tracks.rds') # tibble of tracks
MAX_T <- max(tracks$t) # maximum amount of exploration time

WORST <- filter(d55, mu == min(mu)) %>% # lowest mean resources
  filter(sigma2 == max(sigma2)) %>% # with highest variance
  slice(1) # take the first row only
BEST <- filter(d55, mu == max(mu)) %>% # highest mean resources
  filter(sigma2 == min(sigma2)) %>% # with lowest variance
  slice(1) # take the first row only
```

```r
days <-
  transmute(bind_rows(WORST, BEST),
            animal,
            mu,
            sigma2,
            d = list(tracks),
            scenario = c('Worst case', 'Best case')) %>%
  unnest(d) %>% # unnest the datasets so we have a single, large tibble
  select(-timestamp) %>%
  # generate the food for each row from a gamma distribution
  mutate(food = rgamma2(mu = mu, sigma2 = sigma2, N = n()),
         # the animal finds food if it visits a new cell, otherwise not
         food = if_else(new_cell, food, 0)) %>%
  # end the movement once the animal has reached satiety
  group_by(day, animal, scenario) %>%
  # calculate the total visits, total calories, and if animal is full
  mutate(satiety = cumsum(food), # for diagnostics if animals aren't full
         full = satiety >= REQUIRED) %>% # did the animal reach its needs?
  filter(cumsum(full) <= 1) %>% # full only once
  ungroup()

# single estimates that eventually converge to the asymptote ----
days_summarized <-
  days %>%
  # find how long it took to reach satiety
  group_by(scenario, day) %>%
  nest(tel_day = -c(scenario, day)) %>%
  mutate(t_expl = map_dbl(tel_day, \(d) max(d$t))) %>%
  # add days sequentially
  group_by(scenario) %>%
  mutate(t_start = lag(2 * t_expl), # add the return time before next "day"
         t_start = if_else(is.na(t_start), 0, t_start), # start at 0
         t_start = cumsum(t_start), # make start times comsecutive
         tel_day = map2(day, t_expl,
                        \(i, te) tels$tel[[i]] %>% # extract day's tel
                          data.frame() %>% # for filtering
                          filter(t <= te))) %>% # end tracks at satiety
  unnest(tel_day) %>% # make one big dataset
  mutate(t = t + t_start, # make times consecutive
         individual.local.identifier = scenario, # ctmm identifier
         timestamp = as.POSIXct(t, origin = '2000-01-01')) %>% # new times
  ungroup() # remove grouping by scenario
```

```r
# estimate saturation curve of home range size over number of days
saturation_days <-
  expand_grid(n_days = (2^seq(1, log2(1e3), by = 0.2)) %>%
                round() %>%
                unique(),
              case = unique(days_summarized$scenario)) %>%
  mutate(data = map2(n_days, case,
                     \(.n, .case) filter(days_summarized,
                                         day <= .n,
                                         scenario == .case)),
         tel = map(data, as.telemetry), # convert to telemetry for modeling
         theta = map(tel, \(x) ctmm.guess(data = x, interactive = FALSE)),
         m = map(1:n(), \(i) {
           cat('Fitting model', i, '\n')
           ctmm.fit(tel[[i]], theta[[i]])
         }), # fit movement model
         sigma = map_dbl(m, \(.m) ctmm:::area.covm(.m$sigma)), # var(pos)
         hr = get_hr(.sigma = sigma, quantile = 0.95)) # Gaussian HR

saturation_days %>%
  select(case, n_days, sigma, hr) %>%
  readr::write_csv('simulations/hr-saturation-days.csv')

ggplot(saturation_days, aes(n_days, hr)) +
  facet_wrap(~ case, nrow = 1) +
  geom_vline(xintercept = 100, color = 'darkorange') +
  geom_smooth(method = 'gam', color = 'black',
              formula = y ~ s(x, bs = 'cs', k = 10),
              method.args = list(family = Gamma(link = 'log'))) +
  geom_point(alpha = 0.3) +
  scale_x_continuous(expression(Number~of~days~sampled~(log[2]~scale)),
                     trans = 'log2', breaks = c(2, 16, 128, 1024),
                     limits = c(2, 1100)) +
  scale_y_continuous(expression(atop(Estimated~'space-use',
                                     requirements~(log[2]~scale))),
                     trans = 'log2')
```

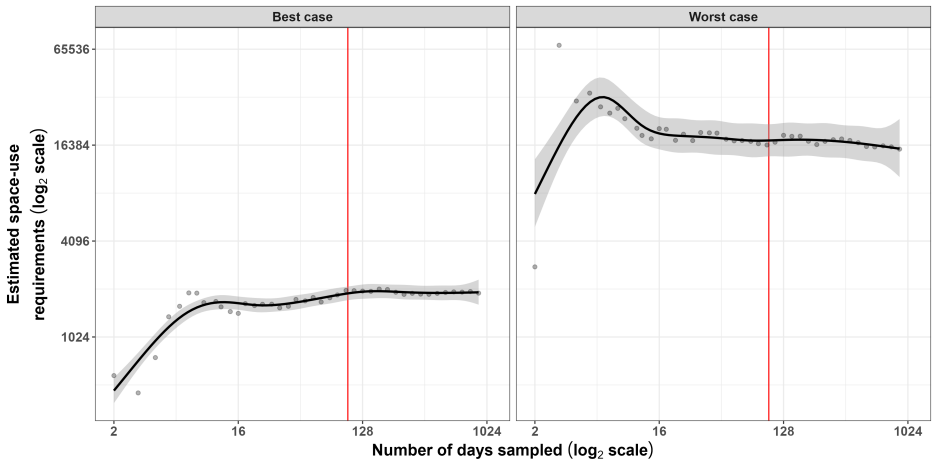


Figure B5: Estimated space-use requirements as a function of the number of days sampled for an animal in a habitat with the highest E(R) and lowest Var(R) (left) and an animal with the lowest E(R) and highest Var(R) (right). In both cases, 100 days are sufficient to produce stable estimates of space-use requirements.

3 Main scripts (to be run in the following order)

1. `analysis/simulations/hr-mean-variance-simulations-days.R`
2. `analysis/simulations/hr-mean-variance-simulations-days-summarized.R`
3. `analysis/simulations/hr-mean-variance-simulations-modeling.R`
4. `analysis/simulations/hr-mean-variance-simulations-hrs.R`
5. `analysis/simulations/modeling-R-and-hr.R`

4 Modeling the results from the simulations

This final section illustrates the location-scale GAM used to estimate the effects of resource abundance and stochasticity on the simulated space-use requirements. Although the model was fit to both the 95% and core (50%) home range estimates, the results were presented only for the 95% quantile for simplicity. The home range was caulculated using a Gaussian approximation using the simulated organism's positional variance:

$$\widehat{h_q} = -2\log(1-q)\pi\sigma^2_{pos},$$

where $widehath_q$ is the estimated home range for the q^{th} quantile and σ^2_{pos} is the positional variance. Consequently, the estimated core home range estimate is simply a multiple of the 95% quantile:

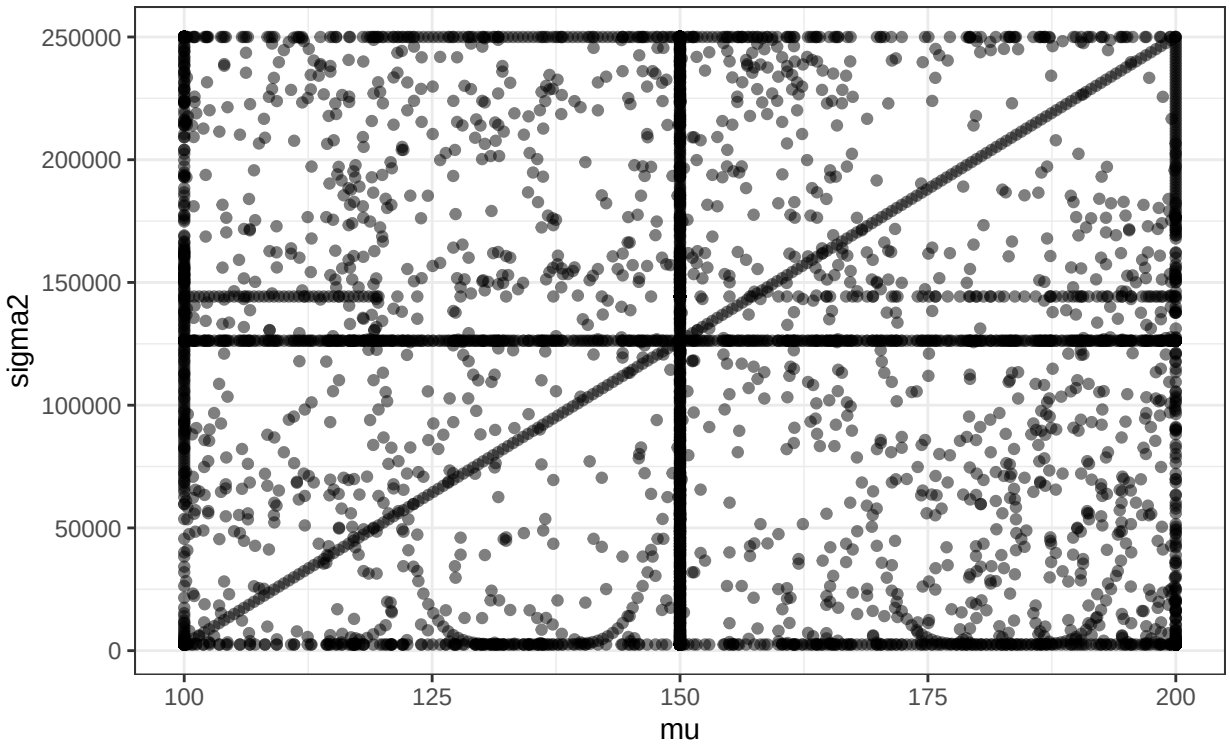
$$\widehat{h_{0.95}}/\widehat{h_{0.5}} = \frac{-2\log(1-0.95)\pi\sigma^2_{pos}}{-2\log(1-0.5)\pi\sigma^2_{pos}} = \frac{\log(1-0.95)}{\log(1-0.5)} = \frac{\log(0.05)}{\log(0.5)} = \log_{0.5}(0.05) \approx 4.322,$$

and therefore $\mathrm{E}(R)$ and $\mathrm{Var}(R)$ would have similar effects (but about 4.322 times weaker) on the core home range than they do on the 95% home range. Note that we used this approximation because, unlike real organisms, the simulated organism did not use the available space selectively (e.g., spending more time near cell boundaries), so the Gaussian approximation is appropriate. When estimating space-use requirements based on empirical data, we suggest estimating utilization distributions using methods such as Autocorrelated Kernel Density Estimation (see Appendix C on empirical modeling and Noonan et al. 2019; Alston et al. 2022; Silva et al. 2022).

```
# switch data to long format with a column of quantile
sims <- readRDS('H:/GitHub/hr-resource-stoch/simulations/days-hrs.rds') %>%
  tibble() %>%
  pivot_longer(c(hr_50, hr_95), names_to = 'quantile', values_to = 'hr') %>%
  mutate(quantile = factor(quantile)) # necessary for GAMs
```

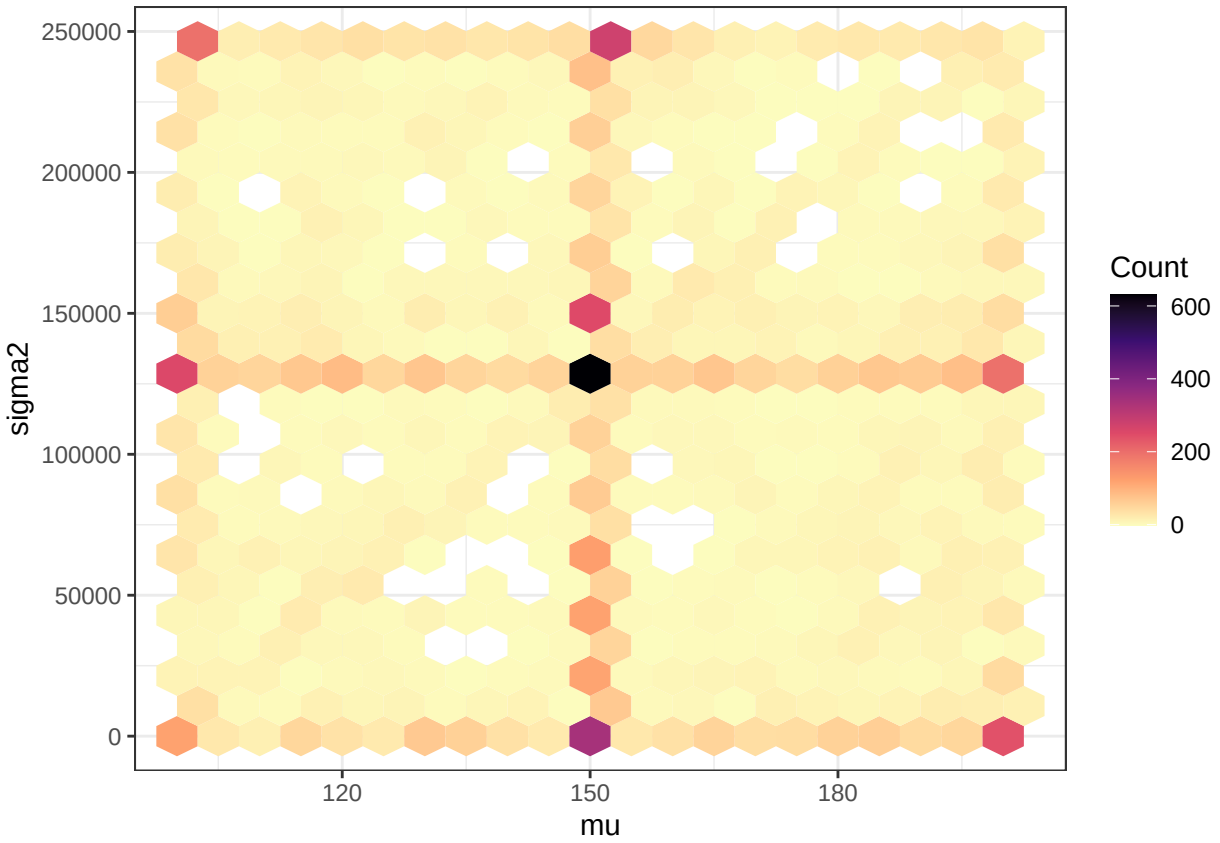
The sampling of the data is not uniform:

```
ggplot(sims, aes(mu, sigma2)) +
  geom_point(alpha = 0.3) +
  theme_bw() # Rmd only keeps theme_bw as default for a single plot
```



But it is not particularly problematic:

```
ggplot(sims, aes(mu, sigma2)) +
  geom_hex(bins = 20) +
  scale_fill_viridis_c('Count', limits = c(0, NA), option = 'A',
                       direction = -1) +
  theme_bw()
```



The abundance of data for the average E(R) and Var(R) did not create biases in the models, since removing such values had no appreciable effect on the model estimates (not shown). To estimate the effects of resource abundance and stochasticity on the simulated space-use requirements, we fit 3 location-scale GAMs (GAMLSs) with a location-scale gamma distribution (since home ranges are strictly positive) of increasing complexity. We hypothesized that the effect of E(R) would depend on Var(R) (and the effect of Var(R) would depend on E(R)), since we expected organisms to respond to changes in $\mu(t)$ less in predictable environments than in stochastic ones. The three models are detailed below and compared using the Akaike Information Criterion and the Bayesian Information Criterion. Both information criteria demonstrate that the third model, which accounts for the interaction effects of E(R) and Var(R), is the best model. Each of the three models are based on the directed acyclical graph (DAG) shown in figure B6. Within a Bayesian framework, DAGs can be used to infer causality within statistical models (as the latter alone does not allow one to infer

causality, see McElreath 2016). The DAG we present in this appendix can be interpreted as follows: E(R) and Var(R) are the controlled variables that determine both the amount of resources in a given encounter with resources, R, and, consequently, an organism's space-use requirements, H. Additionally, as indicated by the arrow going from E(R) to Var(R), the effect of Var(R) depends of the value of E(R) (and vice-versa, but arrows in DAGs have to be unidirectional).

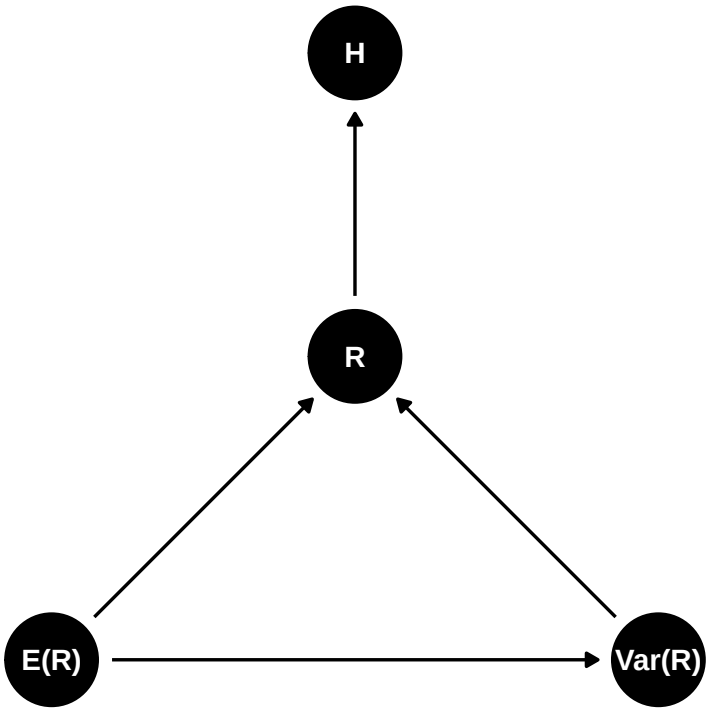
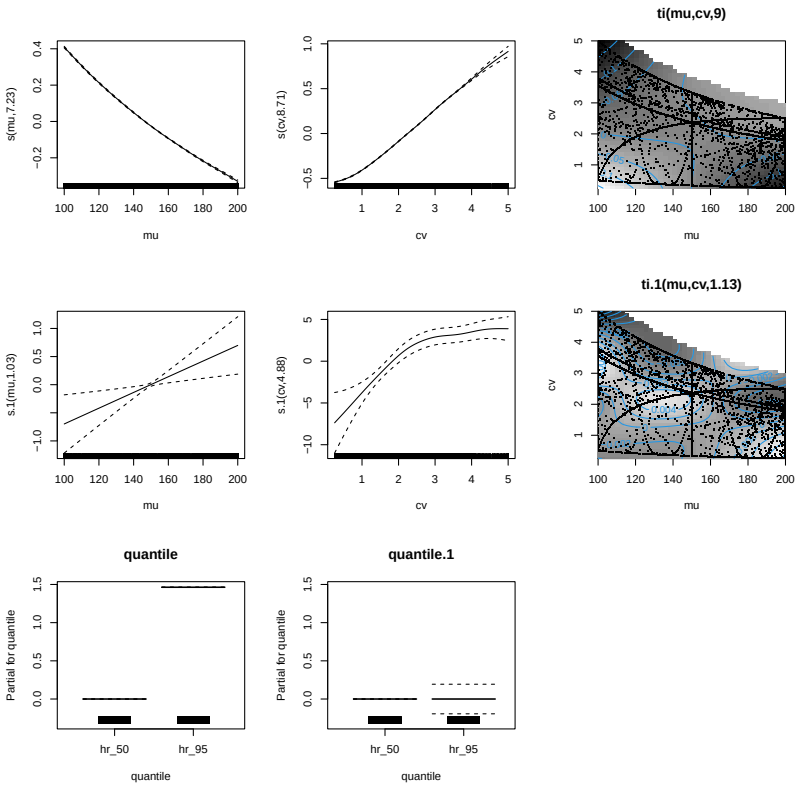


Figure B6: Directed Acyclical Graph assumed for inferring the causal effects of E(R) and Var(R) on H.

```
# marginal effects of mu and sigma only (linear on the log link scale)
m_1 <- gam(
  list(
    # predictor for the mean
    hr ~ quantile + mu + sigma2,
    # predictor for the scale (variance = scale * mean)
    ~ quantile + mu + sigma2),
  family = gammals(),
  data = sims,
  method = 'REML')
```

```r
# marginal effects of mu and sigma only (nonlinear on the log link scale)
m_2 <- gam(
  list(
    hr ~ quantile + s(mu) + s(sigma2),
    ~ quantile + s(mu) + s(sigma2)),
  family = gammals(),
  data = sims,
  method = 'REML')

# marginal and interaction effects (nonlinear on the log link scale)
m_3 <- gam(
  list(
    hr ~ quantile + s(mu, k = 5) + s(sigma2, k = 5) + ti(mu, sigma2, k = 5),
    ~ quantile + s(mu, k = 5) + s(sigma2, k = 5) + ti(mu, sigma2, k = 5)),
  family = gammals(),
  data = sims,
  method = 'REML')

# check if CV = sigma/mu is better than sigma
m_4 <- gam(
  list(
    hr ~ quantile + s(mu) + s(cv) + ti(mu, cv, k = 5),
    ~ quantile + s(mu) + s(cv) + ti(mu, cv, k = 5)),
  family = gammals(),
  data = mutate(sims, cv = sqrt(sigma2) / mu),
  method = 'REML')

# ti term of the scale parameter looks over-fit and too uncertain
plot(m_4, all.terms = TRUE, pages = 1, scheme = 3, scale = 0)
```


s(mu,7.23)
mu
s(cv,8.71)
cv
ti(mu,cv,9)
cv
mu
s.1(mu,1.03)
mu
s.1(cv,4.88)
cv
ti.1(mu,cv,1.13)
cv
mu
quantile
Partial for quantile
hr_50
hr_95
quantile
quantile.1
Partial for quantile
hr_50
hr_95
quantile

```r
AIC(m_1, m_2, m_3, m_4) # not much of a difference between m_3 and m_4
```

```
          df      AIC
m_1  8.00000 133093.9
m_2 34.34593 120810.8
m_3 34.00258 115836.6
m_4 39.93244 115818.5
```

```r
BIC(m_1, m_2, m_3, m_4) # not much of a difference between m_3 and m_4
```

```
          df      BIC
m_1  8.00000 133151.6
m_2 34.34593 121058.5
m_3 34.00258 116081.8
m_4 39.93244 116106.4
```

```r
# the difference in df (and significance) is in the ti.1() term
summary(m_3)
```

```
Family: gammals
Link function: identity log

Formula:
hr ~ quantile + s(mu, k = 5) + s(sigma2, k = 5) + ti(mu, sigma2,
    k = 5)
~quantile + s(mu, k = 5) + s(sigma2, k = 5) + ti(mu, sigma2,
    k = 5)

Parametric coefficients:
                  Estimate Std. Error   z value Pr(>|z|)
(Intercept)      7.097e+00  4.949e-04 14340.603  < 2e-16 ***
quantilehr_95    1.464e+00  6.848e-04  2137.506  < 2e-16 ***
(Intercept).1   -2.875e+00  3.636e-01    -7.907 2.63e-15 ***
quantilehr_95.1 -1.117e-13  9.673e-02     0.000        1
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Approximate significance of smooth terms:
                   edf Ref.df    Chi.sq  p-value
s(mu)            3.990  4.000 1.074e+06  < 2e-16 ***
s(sigma2)        3.995  4.000 7.752e+05  < 2e-16 ***
ti(mu,sigma2)   14.325 15.662 8.625e+03  < 2e-16 ***
s.1(mu)          1.309  1.542 2.753e+01 3.09e-06 ***
s.1(sigma2)      3.404  3.739 1.166e+02  < 2e-16 ***
ti.1(mu,sigma2)  1.032  1.060 9.571e+00  0.00219 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Deviance explained = 99.9%
-REML =  57981  Scale est. = 1         n = 10000
```

```r
summary(m_4)
```

```
Family: gammals
Link function: identity log

Formula:
hr ~ quantile + s(mu) + s(cv) + ti(mu, cv, k = 5)
~quantile + s(mu) + s(cv) + ti(mu, cv, k = 5)

Parametric coefficients:
                  Estimate Std. Error z value Pr(>|z|)
(Intercept)      7.122e+00  1.762e-03 4040.86  < 2e-16 ***
quantilehr_95    1.464e+00  6.837e-04 2140.92  < 2e-16 ***
(Intercept).1   -3.037e+00  4.658e-01   -6.52 7.03e-11 ***
quantilehr_95.1  1.324e-14  9.687e-02    0.00        1
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Approximate significance of smooth terms:
              edf Ref.df    Chi.sq p-value
s(mu)       7.228  8.215 4.445e+04 < 2e-16 ***
s(cv)       8.710  8.965 3.578e+05 < 2e-16 ***
ti(mu,cv)   9.005 10.649 1.148e+04 < 2e-16 ***
s.1(mu)     1.027  1.053 7.681e+00 0.00752 **
s.1(cv)     4.884  5.817 1.082e+02 < 2e-16 ***
ti.1(mu,cv) 1.126  1.235 1.000e-03 0.99757
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Deviance explained = 99.9%
-REML =  57967  Scale est. = 1         n = 10000
```

```
m <- m_3 # select the best model
```

Fig. B7 shows each of the terms from model 3. The estimated effect of `quantile` (see the output of `summary()` above) supports the relationship between the gaussian HR estimates detailed above, since $\exp(1.464) = 4.323218 \approx \log_{0.5}(0.05)$, but there is no differences in the scale parameter between the two quantiles. Additionally, the interaction term of $\mu(t)$ and $\sigma^2(t)$ on the mean HR indicates that the effect of $\sigma^2(t)$ is stronger at lower values of $\mu(t)$ (e.g., $\mu(t) = 100$), since the estimated effect is negative for low values of $\sigma^2(t)$ and positive for high values of $\sigma^2(t)$, which accentuates the effect of $\sigma^2(t)$ on the mean home range. The opposite is true for high values of $\mu(t)$, where the effect of $\sigma^2(t)$ becomes flatter, and thus weaker. Similar but opposite conclusions can be drawn from the interaction term for the scale parameter, since the effect of $\sigma^2(t)$ on the scale parameter is weaker when $\mu(t)$ is low and stronger when $\mu(t)$ is high.

Finally, we can use the model to recreate fig. 4 from the main text and visualize the marginal effects of $\mu(t)$ and $\sigma^2(t)$ on the 95% home range.

```
# effect of mu
newd_mu <-
  expand_grid(mu = seq(min(sims$mu), max(sims$mu), length.out = 400),
              sigma2 = mean(sims$sigma2),
              quantile = unique(sims$quantile))

preds_mu <- bind_cols( # bind new data and predictions
  newd_mu,
  predict(m, newdata = newd_mu, type = 'link', se.fit = TRUE) %>%
    data.frame() %>% # convert list to data frame
    # 95% CIs assuming Gaussian credible intervals on the link scale
    transmute(hr = exp(fit.1),
              hr_lwr = exp(fit.1 - se.fit.1 * 1.96),
              hr_upr = exp(fit.1 + se.fit.1 * 1.96)))
```

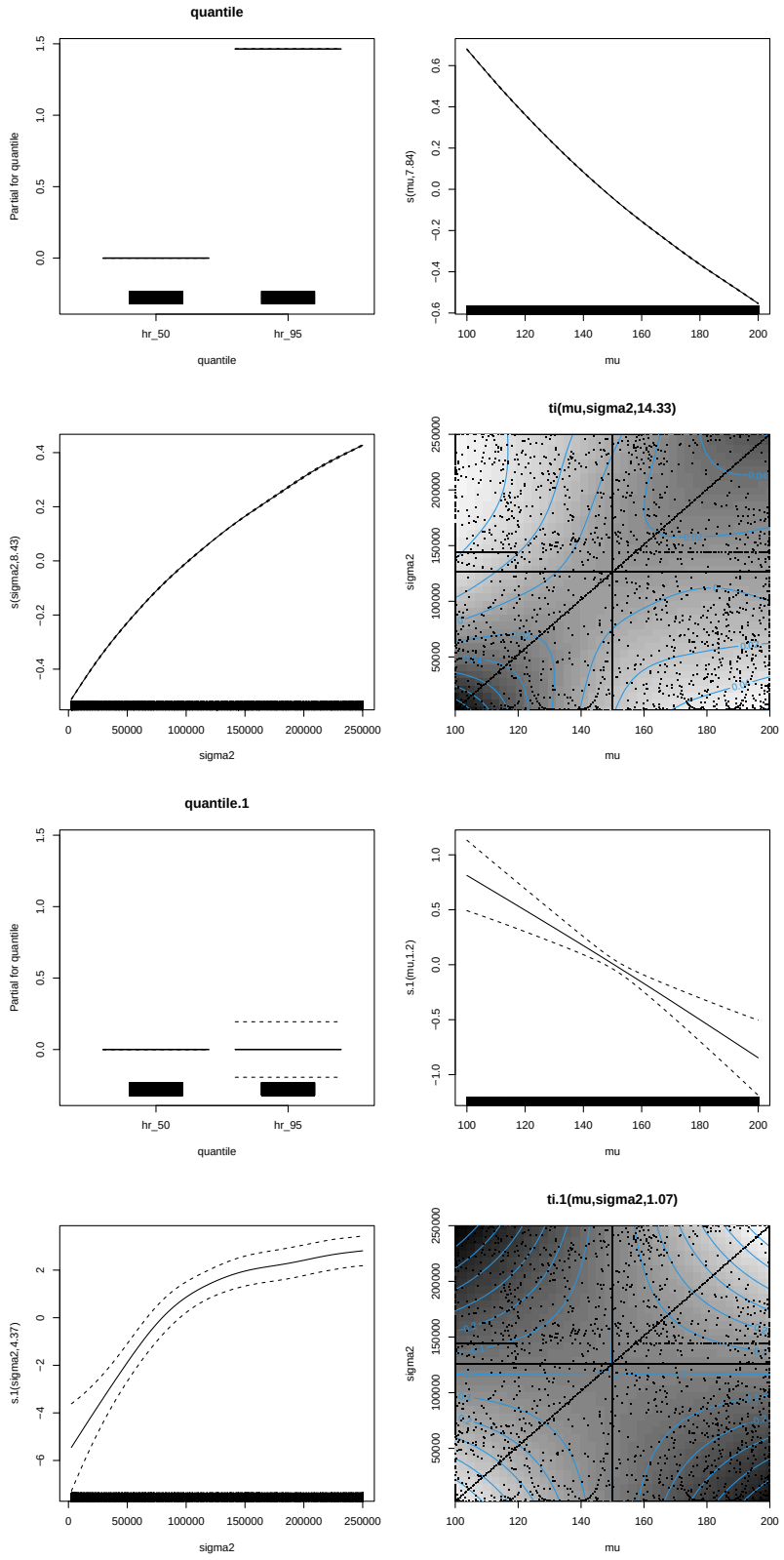
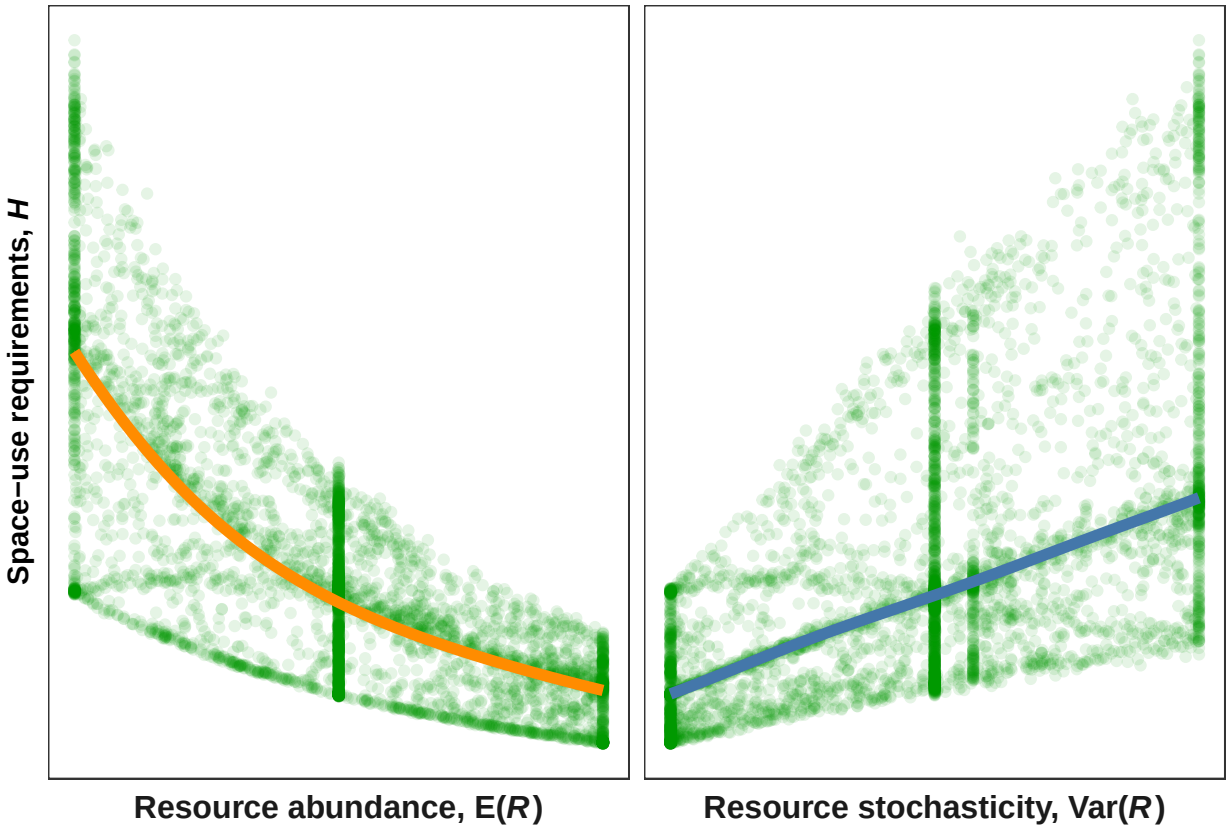


Figure B7: Marginal and interaction effects of each of the terms present in the final GAMLS (on the log link scales). The dashed lines indicate the 95% Bayesian credible intervals. The data points are included as rug plots in all marginal terms and as points in the interaction terms.

```r
# effect of sigma2
newd_sigma2 <-
  expand_grid(mu = mean(sims$mu),
              sigma2 = seq(min(sims$sigma2), max(sims$sigma2),
                           length.out = 400),
              quantile = unique(sims$quantile))

preds_sigma2 <- bind_cols(
  newd_sigma2,
  predict(m, newdata = newd_sigma2, type = 'link', se.fit = TRUE) %>%
    data.frame() %>%
    transmute(hr = exp(fit.1),
              hr_lwr = exp(fit.1 - se.fit.1 * 1.96),
              hr_upr = exp(fit.1 + se.fit.1 * 1.96)))

# final figure (Rmd throws and error with label_bquote)
e_r <- 'paste(bold("Resource abundance, E("), bolditalic("R"), bold(")"))'
v_r <-
  'paste(bold("Resource stochasticity, Var("), bolditalic("R"), bold(")"))'

preds <- bind_rows(mutate(preds_mu, x = e_r) %>%
                     rename(value = mu) %>%
                     select(-sigma2),
                   mutate(preds_sigma2, x = v_r) %>%
                     rename(value = sigma2) %>%
                     select(-mu))

sims_l <- sims %>%
  mutate(e = hr - predict(m, newdata = sims, type = 'response')[, 1]) %>%
  pivot_longer(c(mu, sigma2)) %>%
  mutate(x = if_else(name == 'mu', e_r, v_r))
```

```
ggplot() +
  facet_grid(. ~ x, scales = 'free', switch = 'x',
             labeller = label_parsed) + # label below x axis
  geom_point(aes(value, hr), filter(sims_l, quantile == 'hr_95'),
             alpha = 0.1, color = pal[3]) +
  # 95% CIs are present but hidden under the estimates
  geom_ribbon(aes(value, ymin = hr_lwr, ymax = hr_upr, fill = x),
              filter(preds, quantile == 'hr_95'), alpha = 0.5) +
  geom_line(aes(value, hr, color = x),
            filter(preds, quantile == 'hr_95'), linewidth = 2) +
  scale_color_manual(values = pal) +
  scale_x_continuous(NULL, breaks = NULL) +
  scale_y_continuous(bquote(paste(bold('Space-use requirements, '),
                                  bolditalic('H'))), breaks = NULL) +
  theme_bw() +
  # to make facet strip look like the x axis
  theme(legend.position = 'none',
        strip.background = element_blank(),
        strip.text = element_text(size = 12))
```



References

Alston, J. M., C. H. Fleming, R. Kays, J. P. Streicher, C. T. Downs, T. Ramesh, B. Reineking, et al. 2022. Mitigating pseudoreplication and bias in resource selection functions with autocorrelation-informed weighting. Methods in Ecology and Evolution 2041–210X.14025.

Fleming, C. H., and J. M. Calabrese. 2021. Ctmm: Continuous-time movement modeling.

Henry, L., and H. Wickham. 2022. Purrr: Functional programming tools.

Hijmans, R. J. 2023. Terra: Spatial data analysis.

McElreath, R. 2016. Statistical rethinking: A bayesian course with examples in r and stan. Chapman & hall/CRC texts in statistical science series. CRC Press/Taylor & Francis Group, Boca Raton.

Noonan, M. J., M. A. Tucker, C. H. Fleming, T. S. Akre, S. C. Alberts, A. H. Ali, J. Altmann, et al. 2019. A comprehensive analysis of autocorrelation and bias in home range estimation. Ecological Monographs 89:e01344.

Silva, I., C. H. Fleming, M. J. Noonan, J. Alston, C. Folta, W. F. Fagan, and J. M. Calabrese. 2022. Autocorrelation-informed home range estimation: A review and practical guide. Methods in Ecology and Evolution 13:534–544.

Wickham, H. 2016. ggplot2: Elegant graphics for data analysis. Springer-Verlag New York.

Wickham, H., R. François, L. Henry, and K. Müller. 2022. Dplyr: A grammar of data manipulation.

Wickham, H., and M. Girlich. 2022. Tidyr: Tidy messy data.

Wilke, C. O. 2020. Cowplot: Streamlined plot theme and plot annotations for 'ggplot2'.