sensitivity

Getting dependencies

```
rm(list=ls())
knitr::purl(input="loon3stages.qmd", output="matrixFunctions.R",
 ↪  documentation=0)
source("matrixFunctions.R")

library(ggpubr)
library(reshape2)
library(ggplot2)
library(tidyverse)
```

Hypothetical parameters

```
# Define hypothetical transition parameters
hyp_p3 <- list(
  b_a=0.8,   # adult pairing propensity
  b_y=0.8, # young adult pairing propensity
  m=0.48, # chick production.
  r=0.5,   #sex ratio
  sigma_j = 0.45^(12/34), #juv survival
  sigma_a=0.92,  # Adult survival
  sigma_y=0.92 # young adult survival
)

hyp_p2 <- list(
  b=0.8,   # adult pairing propensity
```

```
  m=0.48, # chick production.
  r=0.5,   #sex ratio
  sigma_j = 0.45^(12/34), #juv survival
  Pa=0.92  # Adult survival
)
```

Comparing lambdas

Here I compare lambdas of different matrices. I need to confirm all matrices have the same lambdas when all parameters are the same. This means all matrices are collapsing correctly.

```
# construct matrix
lam2 <- constructMatrix2(hyp_p2)$lam2
lam3 <- constructMatrix3(hyp_p3)$lam3
lam4 <- constructMatrix4(hyp_p2)$lam4
lam7 <- constructMatrix7(hyp_p3)$lam7


tibble(lam2,lam4,lam7,lam3) %>%
  knitr::kable()
```

lam2	lam4	lam7	lam3
0.9974851	0.9974851	0.9974851	0.9974851

Functions

3 to 2 matrix parameter collapse function

When given a parameter list for 3 stage matrix, this function can calculate and return a list of parameters for the 2 stage matrix.

```
param_collapse <- function(param_list){
  SSD3<-eigen(constructMatrix3(param_list)$three_stage_matrix)$vectors[,1]/e
 ↪  igen(constructMatrix3(param_list)$three_stage_matrix)$vectors[2,1]
  N0_3<-Re(SSD3)/sum(Re(SSD3)) * 500
  p2 <- list(
  b=(param_list$b_a * N0_3[3] + param_list$b_y * N0_3[2]) / (N0_3[3] +
   ↪  N0_3[2]),   # adult pairing propensity
```

```
  m=param_list$m, # chick production.
  r=param_list$r,   #sex ratio
  sigma_j = param_list$sigma_j, #juv survival
  Pa=(param_list$sigma_a * N0_3[3] + param_list$sigma_y * N0_3[2]) / (N0_3[3]
   ↪  + N0_3[2])  # Adult survival
)
  return(p2)
}
```

test if lambda is the same after collapsing

```
b_test_p3 <- list(
  b_a=0.6,   # adult pairing propensity
  b_y=0.7, # young adult pairing propensity
  m=0.48, # chick production.
  r=0.5,   #sex ratio
  sigma_j = 0.45^(12/34), #juv survival
  sigma_a=0.92,  # Adult survival
  sigma_y=0.92 # young adult survival
)

b_test_p2 <- param_collapse(b_test_p3)

constructMatrix2(b_test_p2)$lam2 == constructMatrix3(b_test_p3)$lam3
```

```
[1] TRUE
```

```
p_test_p3 <- list(
  b_a=0.8,   # adult pairing propensity
  b_y=0.8, # young adult pairing propensity
  m=0.48, # chick production.
  r=0.5,   #sex ratio
  sigma_j = 0.45^(12/34), #juv survival
  sigma_a=0.93,  # Adult survival
  sigma_y=0.7 # young adult survival
)

p_test_p2 <- param_collapse(p_test_p3)

constructMatrix2(p_test_p2)$lam2 == constructMatrix3(p_test_p3)$lam3
```

```
[1] FALSE
```

Plotting lamdas

```
hyp_p3 <- list(
  b_a=0.8,   # adult pairing propensity
  b_y=0.8, # young adult pairing propensity
  m=0.48, # chick production.
  r=0.5,   #sex ratio
  sigma_j = 0.45^(12/34), #juv survival
  sigma_a=0.9,  # Adult survival
  sigma_y=0.9 # young adult survival
)

hyp_p2 <- list(
  b=0.8,   # adult pairing propensity
  m=0.48, # chick production.
  r=0.5,   #sex ratio
  sigma_j = 0.45^(12/34), #juv survival
  Pa=0.9  # Adult survival
)
```

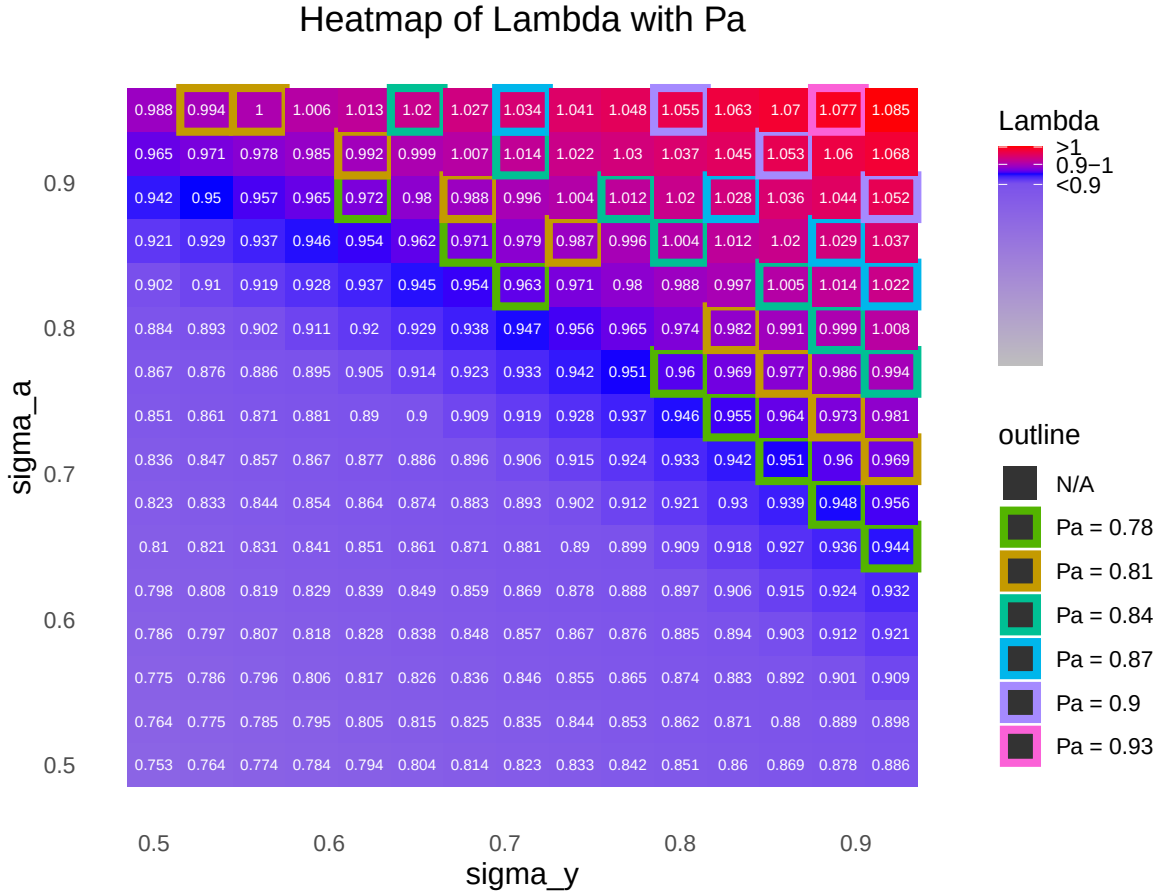
heat map

for sigma_a and sigma_y and Pa

```
# Create a grid of sigma_y and sigma_a values
sigma_y_values <- seq(0.5, 0.94, by = 0.03)
sigma_a_values <- seq(0.5, 0.97, by = 0.03)

# Initialize a data frame to store results
heatmap_data <- data.frame(sigma_a = numeric(0), sigma_y = numeric(0), lam3 =
 ↪  numeric(0), Pa = numeric(0))

# Iterate through all combinations of sigma_y and sigma_a
for (sigma_y in sigma_y_values) {
  for (sigma_a in sigma_a_values) {
    # Update the parameter list
    param_list <- list(
      sigma_y = sigma_y,
      sigma_a = sigma_a,
      b_a = 0.6, # example value
```

```
      b_y = 0.6, # example value
      m = 1.5,   # example value
      r = 0.5,   # example value
      sigma_j = 0.45^(12/34) # example value
    )

    # Calculate lam3 and Pa
    construct_result <- constructMatrix3(param_list)
    p_collapsed <- param_collapse(param_list)

    # Append the result to the data frame
    heatmap_data <- rbind(heatmap_data, data.frame(
      sigma_a = sigma_a,
      sigma_y = sigma_y,
      lam3 = construct_result$lam3,
      Pa = p_collapsed$Pa
    ))
  }
}

interest_Pa <- seq(0.78, 0.94, by = 0.03)

# Compare each Pa with all others and add an outline flag
heatmap_data <- heatmap_data %>%
  mutate(outline = "N/A") %>%  # Set default outline as "N/A"
  mutate(outline = sapply(Pa, function(current_Pa) {
    # Check if current_Pa is within 0.005 of any value in interest_Pa
    matched_Pa <- sapply(interest_Pa, function(target_Pa) {abs(current_Pa -
 ↪  target_Pa) <= 0.005})

    # If matched, return the Pa value; otherwise, return "N/A"
    if (any(matched_Pa)) {paste("Pa =", interest_Pa[which(matched_Pa)])}
     ↪  else {"N/A"}
  }))

# Generate a color palette for each Pa group
n_groups <- length(unique(heatmap_data$outline))
group_colors <- scales::hue_pal()(n_groups)

group_colors[1] <- "transparent"

# Plot the heatmap
```

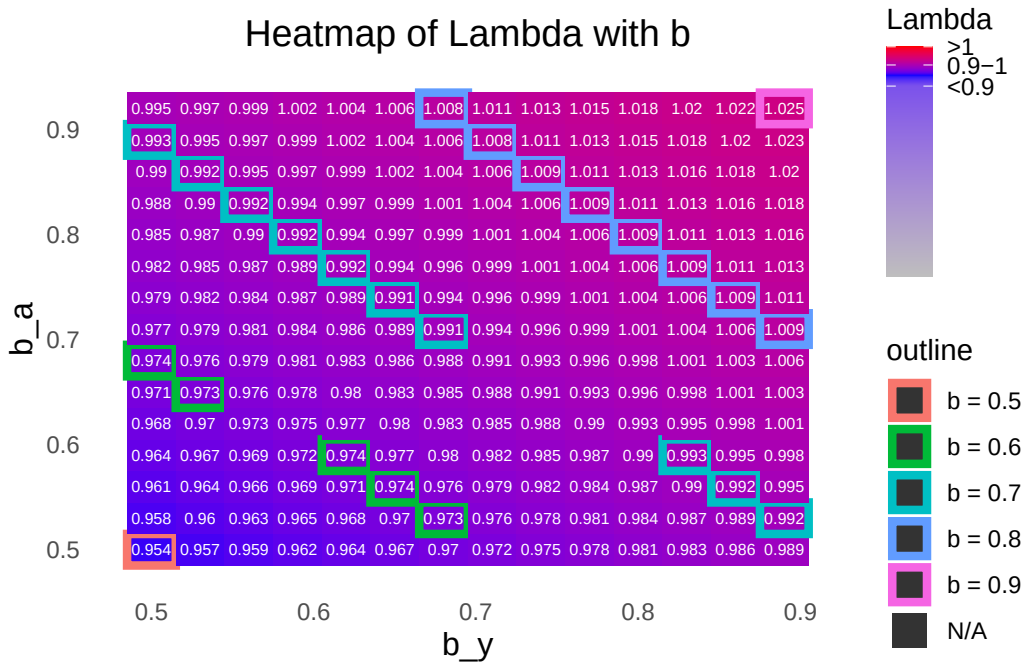
```r
ggplot(heatmap_data, aes(x = sigma_y, y = sigma_a, fill = lam3)) +
  geom_tile(aes(color = outline), size = 1.5) +  # Use Pa_group for the color
   ↪  outline
  geom_text(aes(label = round(lam3, digits = 3)), color = "white", size = 2)
   ↪  +
  scale_color_manual(values = setNames(group_colors,
   ↪  unique(heatmap_data$outline))) +  # Apply color palette
  scale_fill_gradientn(
    colors = c("gray", "blue", "red"),
    values = scales::rescale(c(0, 0.9, 1, 1.086)),
    limits = c(0, 1.086),
    breaks = c(0.9, 1, 1.086),
    labels = c("<0.9", "0.9-1", ">1"),
    name = "Lambda"
  ) +
  labs(x = "sigma_y", y = "sigma_a", title = "Heatmap of Lambda with Pa") +
  theme_minimal() +
  theme(
    panel.grid = element_blank(),
    axis.title = element_text(size = 12),
    plot.title = element_text(hjust = 0.5, size = 14)
  )
```




```r
b_y_values <- seq(0.5, 0.9, by = 0.03)
b_a_values <- seq(0.5, 0.93, by = 0.03)

# Initialize a data frame to store results
heatmap_data <- data.frame(b_a = numeric(0), b_y = numeric(0), lam3 =
 ↪  numeric(0), b = numeric(0))

# Iterate through all combinations of sigma_y and sigma_a
for (b_y in b_y_values) {
  for (b_a in b_a_values) {
    # Update the parameter list
    param_list <- list(
      sigma_y = 0.8,
      sigma_a = 0.8,
      b_a = b_a, # example value
      b_y = b_y, # example value
```

```
      m = 1.5,   # example value
      r = 0.5,   # example value
      sigma_j = 0.45^(12/34) # example value
    )

    # Calculate lam3 and Pa
    construct_result <- constructMatrix3(param_list)
    b_collapsed <- param_collapse(param_list)

    # Append the result to the data frame
    heatmap_data <- rbind(heatmap_data, data.frame(
      b_a = b_a,
      b_y = b_y,
      lam3 = construct_result$lam3,
      b = b_collapsed$b
    ))
  }
}

interest_b <- seq(0.5, 0.9, by = 0.1)

# Compare each Pa with all others and add an outline flag
heatmap_data <- heatmap_data %>%
  mutate(outline = "N/A") %>%  # Set default outline as "N/A"
  mutate(outline = sapply(b, function(current_b) {
    # Check if current_Pa is within 0.005 of any value in interest_Pa
    matched_b <- sapply(interest_b, function(target_b) {abs(current_b -
 ↪  target_b) <= 0.005})

    # If matched, return the Pa value; otherwise, return "N/A"
    if (any(matched_b)) {paste("b =", interest_b[which(matched_b)])} else
     ↪  {"N/A"}
  }))

# Generate a color palette for each Pa group
n_groups <- length(unique(heatmap_data$outline))
group_colors <- setNames(scales::hue_pal()(n_groups),
 ↪  unique(heatmap_data$outline))

group_colors["N/A"] <- "transparent"

# Plot the heatmap
```

```r
ggplot(heatmap_data, aes(x = b_y, y = b_a, fill = lam3)) +
  geom_tile(aes(color = outline), size = 1.5) +  # Use Pa_group for the color
   ↪  outline
  geom_text(aes(label = round(lam3, digits = 3)), color = "white", size = 2)
   ↪  +
  scale_color_manual(values = setNames(group_colors,
   ↪  unique(heatmap_data$outline))) +  # Apply color palette
  scale_fill_gradientn(
    colors = c("gray", "blue", "red"),
    values = scales::rescale(c(0, 0.9, 1, 1.086)),
    limits = c(0, 1.086),
    breaks = c(0.9, 1, 1.086),
    labels = c("<0.9", "0.9-1", ">1"),
    name = "Lambda"
  ) +
  labs(x = "b_y", y = "b_a", title = "Heatmap of Lambda with b") +
  theme_minimal() +
  theme(
    panel.grid = element_blank(),
    axis.title = element_text(size = 12),
    plot.title = element_text(hjust = 0.5, size = 14)
  )
```



different pairing propensity

```
b_value <- seq(0, 0.9, by = 0.02)


# Initialize a data frame to store results
all_b_lam3 <- data.frame(b = numeric(0), lam3 = numeric(0))

# Iterate through paired combinations
for (i in 1:length(b_value)) {
  b <- b_value[i]
  hyp_p3$b_a <- b
  hyp_p3$b_y <- b

  # Run constructMatrix3 and calculate lam3
  lam3 <- constructMatrix3(hyp_p3)$lam3

  # Store the current lam3 along with b
  all_b_lam3 <- rbind(all_b_lam3, data.frame(b = b, lam3 = lam3))}

# Calculate log sensitivity: (log difference of lam3 over log difference of
 ↪  b)
all_b_lam3$log_sensitivity <- c(NA, diff(log(all_b_lam3$lam3)) /
 ↪  diff(log(all_b_lam3$b)))

# Calculate log elasticity: (b / lam3) * log_sensitivity
all_b_lam3$log_elasticity <- c(NA, (all_b_lam3$b[-1] / all_b_lam3$lam3[-1]) *
 ↪  diff(log(all_b_lam3$lam3)) / diff(log(all_b_lam3$b)))

b_sensitivity_plot <- ggplot(all_b_lam3, aes(x = b, y = log_sensitivity)) +
  geom_line(color = "red") +
  labs(x = "breeding propensity",
       y = "absolute sensitivity (dLam3/db)") +
  scale_x_continuous(limits = c(0, 0.9))

b_elasticity_plot <- ggplot(all_b_lam3, aes(x = b, y = log_elasticity)) +
  geom_line(color = "blue") +
  labs(x = "breeding propensity",
       y = "Elasticity") +
  scale_x_continuous(limits = c(0, 0.9))
```

```r
b_lam_line_plot <- ggplot(all_b_lam3, aes(x = b, y = lam3)) +
  geom_line()+
  labs(x = "breeding propensity",
       y = "Lambda")+
  scale_x_continuous(limits = c(0, 0.9))+
  scale_y_continuous(limits = c(0.8, 1.05))

params <- ggplot() +
    annotate("text", x = 0.5, y = 0.5,
             label = paste("m =", hyp_p3$m, "\n",
                           "r =",hyp_p3$r,"\n",
                           "sigma_j =", hyp_p3$sigma_j,"\n",
                           "sigma_a =",hyp_p3$sigma_a,"\n",
                           "sigma_y =",hyp_p3$sigma_y,"\n",
                           "b 0-0.9"
                           ),
             size = 4, hjust = 0.5, vjust = 0.5) +
    theme_void()

b_lam_annotated <- ggarrange(b_lam_line_plot,b_sensitivity_plot,
 ↪  b_elasticity_plot, params, ncol = 4, nrow = 1)
```

different chick production

```r
m_value <- seq(0.3, 0.6, by = 0.02)


# Initialize a data frame to store results
all_m_lam3 <- data.frame(m = numeric(0), lam3 = numeric(0))

# Iterate through paired combinations
for (i in 1:length(m_value)) {
  m <- m_value[i]
  hyp_p3$m <- m

  # Run constructMatrix3 and calculate lam3
  lam3 <- constructMatrix3(hyp_p3)$lam3

  # Store the current lam3 along with m
  all_m_lam3 <- rbind(all_m_lam3, data.frame(m = m, lam3 = lam3))}
```

```r
# Calculate log sensitivity: (log difference of lam3 over log difference of
 ↪  m)
all_m_lam3$log_sensitivity <- c(NA, diff(log(all_m_lam3$lam3)) /
 ↪  diff(log(all_m_lam3$m)))

# Calculate log elasticity: (m / lam3) * log_sensitivity
all_m_lam3$log_elasticity <- c(NA, (all_m_lam3$m[-1] / all_m_lam3$lam3[-1]) *
 ↪  diff(log(all_m_lam3$lam3)) / diff(log(all_m_lam3$m)))

m_sensitivity_plot <- ggplot(all_m_lam3, aes(x = m, y = log_sensitivity)) +
  geom_line(color = "red") +
  labs(x = "chick production",
       y = "absolute sensitivity (dLam3/dm)") +
  scale_x_continuous(limits = c(0, 0.9))

m_elasticity_plot <- ggplot(all_m_lam3, aes(x = m, y = log_elasticity)) +
  geom_line(color = "blue") +
  labs(x = "chick production",
       y = "Elasticity") +
  scale_x_continuous(limits = c(0, 0.9))

m_lam_line_plot <- ggplot(all_m_lam3, aes(x = m, y = lam3)) +
  geom_line()+
  labs(x = "chick production",
        y = "Lambda")+
  scale_x_continuous(limits = c(0.3, 0.6))+
  scale_y_continuous(limits = c(0.8, 1.05))

params <- ggplot() +
    annotate("text", x = 0.5, y = 0.5,
             label = paste("b_y =", hyp_p3$b_y, "\n",
                           "b_a =", hyp_p3$b_a, "\n",
                           "r =",hyp_p3$r,"\n",
                           "sigma_j =", hyp_p3$sigma_j,"\n",
                           "sigma_a =",hyp_p3$sigma_a,"\n",
                           "sigma_y =",hyp_p3$sigma_y,"\n",
                           "m 0.3-0.6"
                           ),
             size = 4, hjust = 0.5, vjust = 0.5) +
    theme_void()
```

```r
m_lam_annotated <-
 ↪  ggarrange(m_lam_line_plot,m_sensitivity_plot,m_elasticity_plot,params,
 ↪  ncol = 4, nrow = 1)
```

different sex-ratio

```r
r_value <- seq(0.3, 0.6, by = 0.02)


# Initialize a data frame to store results
all_r_lam3 <- data.frame(r = numeric(0), lam3 = numeric(0))

# Iterate through paired combinations
for (i in 1:length(r_value)) {
  r <- r_value[i]
  hyp_p3$r <- r

  # Run constructMatrix3 and calculate lam3
  lam3 <- constructMatrix3(hyp_p3)$lam3

  # Store the current lam3 along with r
  all_r_lam3 <- rbind(all_r_lam3, data.frame(r = r, lam3 = lam3))}

# Calculate log sensitivity: (log difference of lam3 over log difference of
 ↪  m)
all_r_lam3$log_sensitivity <- c(NA, diff(log(all_r_lam3$lam3)) /
 ↪  diff(log(all_r_lam3$r)))

# Calculate log elasticity: (m / lam3) * log_sensitivity
all_r_lam3$log_elasticity <- c(NA, (all_r_lam3$r[-1] / all_r_lam3$lam3[-1]) *
 ↪  diff(log(all_r_lam3$lam3)) / diff(log(all_r_lam3$r)))

r_sensitivity_plot <- ggplot(all_r_lam3, aes(x = r, y = log_sensitivity)) +
  geom_line(color = "red") +
  labs(x = "sex ratio",
       y = "absolute sensitivity (dLam3/dm)") +
  scale_x_continuous(limits = c(0, 0.9))

r_elasticity_plot <- ggplot(all_r_lam3, aes(x = r, y = log_elasticity)) +
  geom_line(color = "blue") +
```

```
  labs(x = "sex ratio",
       y = "Elasticity") +
  scale_x_continuous(limits = c(0, 0.9))

r_lam_line_plot <- ggplot(all_r_lam3, aes(x = r, y = lam3)) +
  geom_line()+
  labs(x = "sex ratio",
        y = "Lambda")+
  scale_x_continuous(limits = c(0.3, 0.6))+
  scale_y_continuous(limits = c(0.8, 1.05))
params <- ggplot() +
    annotate("text", x = 0.5, y = 0.5,
             label = paste("b_y =", hyp_p3$b_y, "\n",
                           "b_a =", hyp_p3$b_a, "\n",
                           "m =",hyp_p3$m,"\n",
                           "sigma_j =", hyp_p3$sigma_j,"\n",
                           "sigma_a =",hyp_p3$sigma_a,"\n",
                           "sigma_y =",hyp_p3$sigma_y,"\n",
                           "r 0.3-0.6"
                           ),
             size = 4, hjust = 0.5, vjust = 0.5) +
    theme_void()

r_lam_annotated <- ggarrange(r_lam_line_plot,r_sensitivity_plot,
 ↪  r_elasticity_plot, params, ncol = 4, nrow = 1)
```

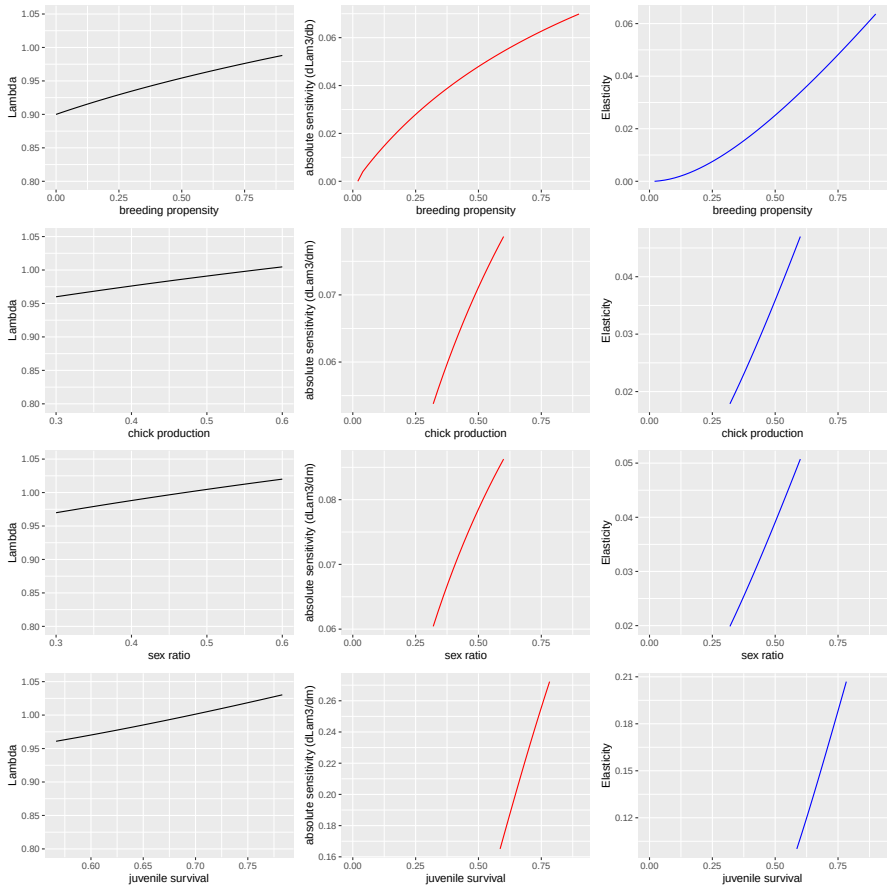
different juvenile survival

```
s <- seq(0.2, 0.5, by = 0.02)
sigma_j_value <- s^(12/34)


# Initialize a data frame to store results
all_sigma_j_lam3 <- data.frame(sigma_j = numeric(0), lam3 = numeric(0))

# Iterate through paired combinations
for (i in 1:length(sigma_j_value)) {
  sigma_j <- sigma_j_value[i]
  hyp_p3$sigma_j <- sigma_j
```

```
  # Run constructMatrix3 and calculate lam3
  lam3 <- constructMatrix3(hyp_p3)$lam3

  # Store the current lam3 along with r
  all_sigma_j_lam3 <- rbind(all_sigma_j_lam3, data.frame(sigma_j = sigma_j,
 ↪  lam3 = lam3))}

# Calculate log sensitivity: (log difference of lam3 over log difference of
 ↪  m)
all_sigma_j_lam3$log_sensitivity <- c(NA, diff(log(all_sigma_j_lam3$lam3)) /
 ↪  diff(log(all_sigma_j_lam3$sigma_j)))

# Calculate log elasticity: (m / lam3) * log_sensitivity
all_sigma_j_lam3$log_elasticity <- c(NA, (all_sigma_j_lam3$sigma_j[-1] /
 ↪  all_sigma_j_lam3$lam3[-1]) * diff(log(all_sigma_j_lam3$lam3)) /
 ↪  diff(log(all_sigma_j_lam3$sigma_j)))

sigma_j_sensitivity_plot <- ggplot(all_sigma_j_lam3, aes(x = sigma_j, y =
 ↪  log_sensitivity)) +
  geom_line(color = "red") +
  labs(x = "juvenile survival",
       y = "absolute sensitivity (dLam3/dm)") +
  scale_x_continuous(limits = c(0, 0.9))

sigma_j_elasticity_plot <- ggplot(all_sigma_j_lam3, aes(x = sigma_j, y =
 ↪  log_elasticity)) +
  geom_line(color = "blue") +
  labs(x = "juvenile survival",
       y = "Elasticity") +
  scale_x_continuous(limits = c(0, 0.9))

sigma_j_lam_line_plot <- ggplot(all_sigma_j_lam3, aes(x = sigma_j, y = lam3))
 ↪  +
  geom_line()+
  labs(x = "juvenile survival",
        y = "Lambda")+
  scale_x_continuous(limits = c(min(sigma_j_value), max(sigma_j_value)))+
  scale_y_continuous(limits = c(0.8, 1.05))

params <- ggplot() +
    annotate("text", x = 0.5, y = 0.5,
             label = paste("b_y =", hyp_p3$b_y, "\n",
```

```
                          "b_a =", hyp_p3$b_a, "\n",
                          "m =",hyp_p3$m,"\n",
                          "r =", hyp_p3$r,"\n",
                          "sigma_a =",hyp_p3$sigma_a,"\n",
                          "sigma_y =",hyp_p3$sigma_y,"\n",
                          "sigma_j 0.57-0.78"
                          ),
             size = 4, hjust = 0.5, vjust = 0.5) +
    theme_void()

sigma_j_lam_annotated <- ggarrange(sigma_j_lam_line_plot,sigma_j_sensitivity
 ↪  _plot,sigma_j_elasticity_plot, params, ncol = 4, nrow = 1)
```

```
ggarrange(b_lam_annotated,
          m_lam_annotated,
          r_lam_annotated,
          sigma_j_lam_annotated,
          ncol = 1, nrow = 4)
```


Lambda
breeding propensity
absolute sensitivity (dLam3/db)
breeding propensity
Elasticity
breeding propensity
m = 0.48
r = 0.5
sigma_j = 0.754404754197739
sigma_a = 0.9
sigma_y = 0.9
b 0−0.9
Lambda
chick production
absolute sensitivity (dLam3/dm)
chick production
Elasticity
chick production
b_y = 0.9
b_a = 0.9
r = 0.5
sigma_j = 0.754404754197739
sigma_a = 0.9
sigma_y = 0.9
m 0.3−0.6
Lambda
sex ratio
absolute sensitivity (dLam3/dm)
sex ratio
Elasticity
sex ratio
b_y = 0.9
b_a = 0.9
m = 0.6
sigma_j = 0.754404754197739
sigma_a = 0.9
sigma_y = 0.9
r 0.3−0.6
Lambda
juvenile survival
absolute sensitivity (dLam3/dm)
juvenile survival
Elasticity
juvenile survival
b_y = 0.9
b_a = 0.9
m = 0.6
r = 0.6
sigma_a = 0.9
sigma_y = 0.9
sigma_j 0.57−0.78