

# R code PA-CR model

Marwan Naciri

2025-07-10

This code generates the results and the main figures presented in the manuscript “Offspring number, size, and survival: state-dependent optimization of litter size in a long-lived capital breeder” *Ecosphere*

Marwan Naciri, Jon Aars, Magnus Andersen, Marie-Anne Blanchet, Andrew E. Derocher, Marlène Gamelon, Øystein Wiig, and Sarah Cubaynes

## A. Run the model

### 1. Build the model

```
PA_CR <- nimbleCode({
  # ++++++ priors ++++++
  # Length
  for (i in 1:n_coef_mL) {
    alpha[i] ~ dnorm(0, sd = 5)
  }
  sigma_tau ~ dunif(0, 10)

  # Cohort random effect
  for (i in 1:n_cohort) {
    zeta[i] ~ dnorm(0, sd = sigma_zeta)
  }
  sigma_zeta ~ dunif(0, 10)

  # Litter size
  for (i in 1:n_coef_LS) {
    beta[i] ~ dnorm(0, sd = 1.5)
  }

  # Girth
  for (i in 1:n_coef_mG) {
    gamma[i] ~ dnorm(0, sd = 5)
  }
  sigma_xi ~ dunif(0, 10)

  # Cub mass
  for (i in 1:n_coef_M) {
    delta[i] ~ dnorm(0, sd = 5)
  }
  sigma_epsilon ~ dunif(0, 10)

  # Yearly random effects
```

```

for (i in 1:n_year) {
  eta[i] ~ dnorm(0, sd = sigma_eta)
  rho[i] ~ dnorm(0, sd = sigma_rho)
  nu[i] ~ dnorm(0, sd = sigma_nu)
}
sigma_eta ~ dunif(0, 10)
sigma_rho ~ dunif(0, 10)
sigma_nu ~ dunif(0, 10)

# Recapture
for (i in 1:n_coef_p) {
  lambda[i] ~ dnorm(0, sd = 1.5)
}

# Survival
for (i in 1:n_coef_phi) {
  theta[i] ~ dnorm(0, sd = 1.5)
}

# Yearly random effects on recapture
for (t in 1:(n_year-1)) {
  omega[t] ~ dnorm(0, sd = sigma_omega)
}
sigma_omega ~ dunif(0, 10)

# ***** model & likelihood *****

# length, girth and litter size
for (k in 1:n_litters) {
  # Length
  mu_mL[k] <- alpha[1] * mAgeY[row_litter[k], col_litter[k]] +
    alpha[2] * mAgeM[row_litter[k], col_litter[k]] +
    alpha[3] * mAge0[row_litter[k], col_litter[k]] +
    alpha[4] * mSpUnk[row_litter[k]] +
    alpha[5] * mSpPel[row_litter[k]] +
    zeta[cohort[k]]
  mL[row_litter[k], col_litter[k]] ~ dnorm(mu_mL[k], sd = sigma_tau)

  # Litter size
  log(q[k, 1]) <- 0

  log(q[k, 2]) <- beta[1] * mAgeY[row_litter[k], col_litter[k]] +
    beta[2] * mAgeM[row_litter[k], col_litter[k]] +
    beta[3] * mAge0[row_litter[k], col_litter[k]] +
    beta[4] * mL[row_litter[k], col_litter[k]] +
    beta[5] * mL[row_litter[k], col_litter[k]]^2 +
    beta[6] * mAgeY[row_litter[k], col_litter[k]] * DOY[row_litter[k], col_litter[k]] +
    beta[7] * mSpUnk[row_litter[k]] +
    beta[8] * mSpPel[row_litter[k]] +
    eta[col_litter[k]]

  log(q[k, 3]) <- beta[9] * mAgeY[row_litter[k], col_litter[k]] +

```

```

    beta[10] * mAgeM[row_litter[k], col_litter[k]] +
    beta[11] * mAge0[row_litter[k], col_litter[k]] +
    beta[12] * mL[row_litter[k], col_litter[k]]

LS[row_litter[k], col_litter[k]] ~ dcat(s[k, 1:3])
for (i in 1:3) {
  s[k, i] <- q[k, i]/sum(q[k, 1:3])
}

LS1[row_litter[k], col_litter[k]] <- 0.5 * (LS[row_litter[k], col_litter[k]] - 2)*(LS[row_litter[k],
LS2[row_litter[k], col_litter[k]] <- -1 * (LS[row_litter[k], col_litter[k]] - 1)*(LS[row_litter[k],
LS3[row_litter[k], col_litter[k]] <- 0.5 * (LS[row_litter[k], col_litter[k]] - 1)*(LS[row_litter[k],

# Girth
mu_mG[k] <- gamma[1] * mAgeY[row_litter[k], col_litter[k]] +
  gamma[2] * mAgeM[row_litter[k], col_litter[k]] +
  gamma[3] * mAge0[row_litter[k], col_litter[k]] +
  gamma[4] * mL[row_litter[k], col_litter[k]] +
  gamma[5] * DOY[row_litter[k], col_litter[k]] +
  gamma[6] * (-1) * (LS[row_litter[k], col_litter[k]] - 1)*(LS[row_litter[k], col_litter[k]] - 3) +
  gamma[7] * 0.5 * (LS[row_litter[k], col_litter[k]] - 1)*(LS[row_litter[k], col_litter[k]] - 2) +
  gamma[8] * mSpUnk[row_litter[k]] +
  gamma[9] * mSpPel[row_litter[k]] +
  rho[col_litter[k]]
mG[row_litter[k], col_litter[k]] ~ dnorm(mu_mG[k], sd = sigma_xi)
}

# Cub mass
for (k in 1:n_cubs) {

  mu_M[k] <- delta[1] * LS1[mother[k], col_cub[k]] * mAgeY[mother[k], col_cub[k]] +
    delta[2] * LS1[mother[k], col_cub[k]] * mAgeM[mother[k], col_cub[k]] +
    delta[3] * LS1[mother[k], col_cub[k]] * mAge0[mother[k], col_cub[k]] +

    delta[4] * LS1[mother[k], col_cub[k]] * mL[mother[k], col_cub[k]] +
    delta[5] * LS1[mother[k], col_cub[k]] * mG[mother[k], col_cub[k]] +
    delta[6] * LS1[mother[k], col_cub[k]] * DOY[mother[k], col_cub[k]] +
    delta[7] * LS1[mother[k], col_cub[k]] * Sex[k] +
    delta[8] * LS1[mother[k], col_cub[k]] * mSpUnk[mother[k]] +
    delta[9] * LS1[mother[k], col_cub[k]] * mSpPel[mother[k]] +

    delta[10] * LS2[mother[k], col_cub[k]] * mAgeY[mother[k], col_cub[k]] +
    delta[11] * LS2[mother[k], col_cub[k]] * mAgeM[mother[k], col_cub[k]] +
    delta[12] * LS2[mother[k], col_cub[k]] * mAge0[mother[k], col_cub[k]] +

    delta[13] * LS3[mother[k], col_cub[k]] * mAgeY[mother[k], col_cub[k]] +
    delta[14] * LS3[mother[k], col_cub[k]] * mAgeM[mother[k], col_cub[k]] +
    delta[15] * LS3[mother[k], col_cub[k]] * mAge0[mother[k], col_cub[k]] +

    delta[16] * (LS2[mother[k], col_cub[k]] + LS3[mother[k], col_cub[k]]) * mL[mother[k], col_cub[k]] +
    delta[17] * (LS2[mother[k], col_cub[k]] + LS3[mother[k], col_cub[k]]) * mG[mother[k], col_cub[k]] +
    delta[18] * (LS2[mother[k], col_cub[k]] + LS3[mother[k], col_cub[k]]) * DOY[mother[k], col_cub[k]] +
    delta[19] * (LS2[mother[k], col_cub[k]] + LS3[mother[k], col_cub[k]]) * Sex[k] +

```

```

delta[20] * (LS2[mother[k], col_cub[k]] + LS3[mother[k], col_cub[k]]) * mSpUnk[mother[k]] +
delta[21] * (LS2[mother[k], col_cub[k]] + LS3[mother[k], col_cub[k]]) * mSpPel[mother[k]] +

nu[col_cub[k]]

M[k] ~ dnorm(mu_M[k], sd = sigma_epsilon)
}

for (i in 1:n_ind) {
  for (t in f[i]:(n_year-1)) {
    # Survival
    logit(phi[i, t]) <-
      theta[1] * Age0[i, t] * mAgeY[mother[i], t] +
      theta[2] * Age0[i, t] * mAgeM[mother[i], t] +
      theta[3] * Age0[i, t] * mAge0[mother[i], t] +

      theta[4] * Age0[i, t] * M[i] +
      theta[5] * Age0[i, t] * (-1 * (LS[mother[i], t] - 1) * (LS[mother[i], t] - 3) + # LS2
      0.5 * (LS[mother[i], t] - 1) * (LS[mother[i], t] - 2)) + # LS3
      theta[6] * Age0[i, t] * mL[mother[i], t] +
      theta[7] * Age0[i, t] * mG[mother[i], t] +
      theta[8] * Age0[i, t] * DOY[mother[i], t] +
      theta[9] * Age0[i, t] * Sex[i] +
      theta[10] * Age0[i, t] * mSpUnk[mother[i]] +
      theta[11] * Age0[i, t] * mSpPel[mother[i]] +

      theta[12] * Age1[i, t] +
      theta[13] * (Age2[i, t] + Age3_4[i, t]) +
      theta[14] * Age5_19[i, t] +
      theta[15] * Age20[i, t] +

      theta[16] * (1 - Age0[i, t]) * Sex[i] + # Effect of sex = male f
      theta[17] * ((1 - Age0[i, t]) * (1 - Sex[i]) * mSpUnk[i] + # Effect of space use = NA
      Age1[i, t] * Sex[i] * mSpUnk[mother[i]]) + # Effect of maternal space use
      theta[18] * ((1 - Age0[i, t]) * (1 - Sex[i]) * mSpPel[i] + # Effect of space use = pelagic
      Age1[i, t] * Sex[i] * mSpPel[mother[i]]) + # Effect of maternal space use

    # Recapture
    logit(p[i, t]) <- lambda[1] +
      lambda[2] * (mSpUnk[i] * (1 - Sex[i]) + # all N
      mSpUnk[mother[i]] * Sex[i] * (Age1[i, t+1] + Age2[i, t+1])) + # all male dependence
      lambda[3] * (mSpPel[i] * (1 - Sex[i]) + # all p
      mSpPel[mother[i]] * Sex[i] * (Age1[i, t+1] + Age2[i, t+1])) + # all male dependence
      lambda[4] * Sex[i] * (Age3_4[i, t+1] + Age5_19[i, t+1] + Age20[i, t+1]) +
      omega[t]

    S[1, 1, i, t] <- phi[i, t]
    S[1, 2, i, t] <- 1 - phi[i, t]
    S[2, 1, i, t] <- 0
    S[2, 2, i, t] <- 1

    E[1, 1, i, t] <- p[i, t]
    E[1, 2, i, t] <- 1 - p[i, t]

```

```

      E[2, 1, i, t] <- 0
      E[2, 2, i, t] <- 1
    }
  }

  for(i in 1:n_ind){
    chi[i, f[i], 1] <- 1
    chi[i, f[i], 2] <- 0

    for(t in f[i]:(n_year-1)){
      chi[i, t+1, 1] <- inprod(chi[i, t, 1:2], S[1:2, 1, i, t]) * E[1, CH[i, t+1], i, t]
      chi[i, t+1, 2] <- inprod(chi[i, t, 1:2], S[1:2, 2, i, t]) * E[2, CH[i, t+1], i, t]
    }

    lik[i] <- sum(chi[i, n_year, 1:2])
    ones[i] ~ dbin(lik[i], freq[i])
  }
})

```

## 2. Process the data

## 3. Run the model

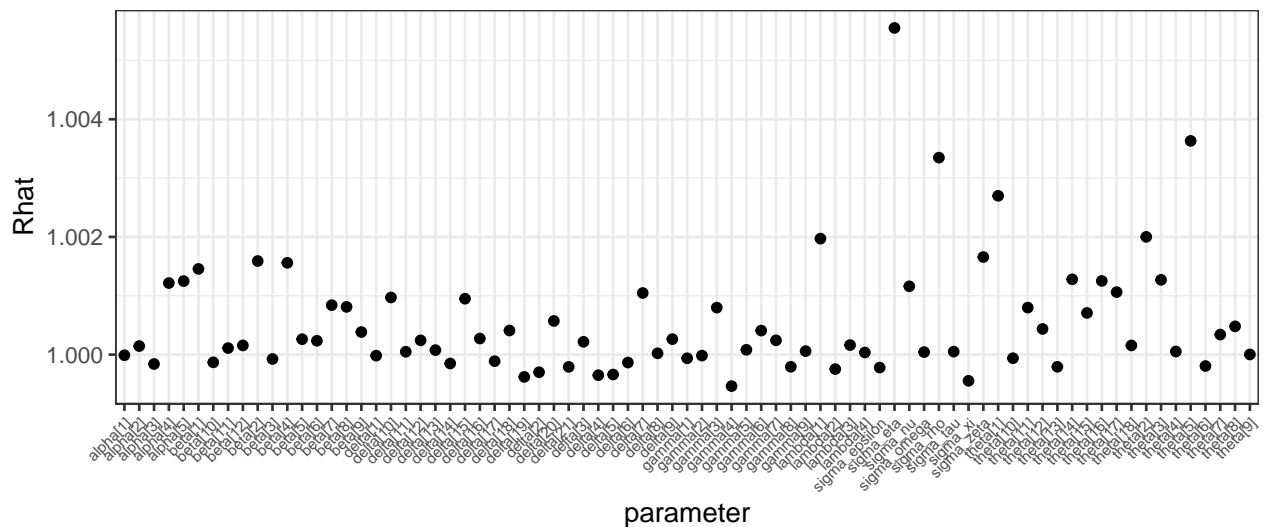
The model takes >5 hours to run on a cluster. The output of the model (fit\_PA\_CR) is provided in the repository as a .RData file.

# B. Inspect the results

## 1. Diagnostic plots

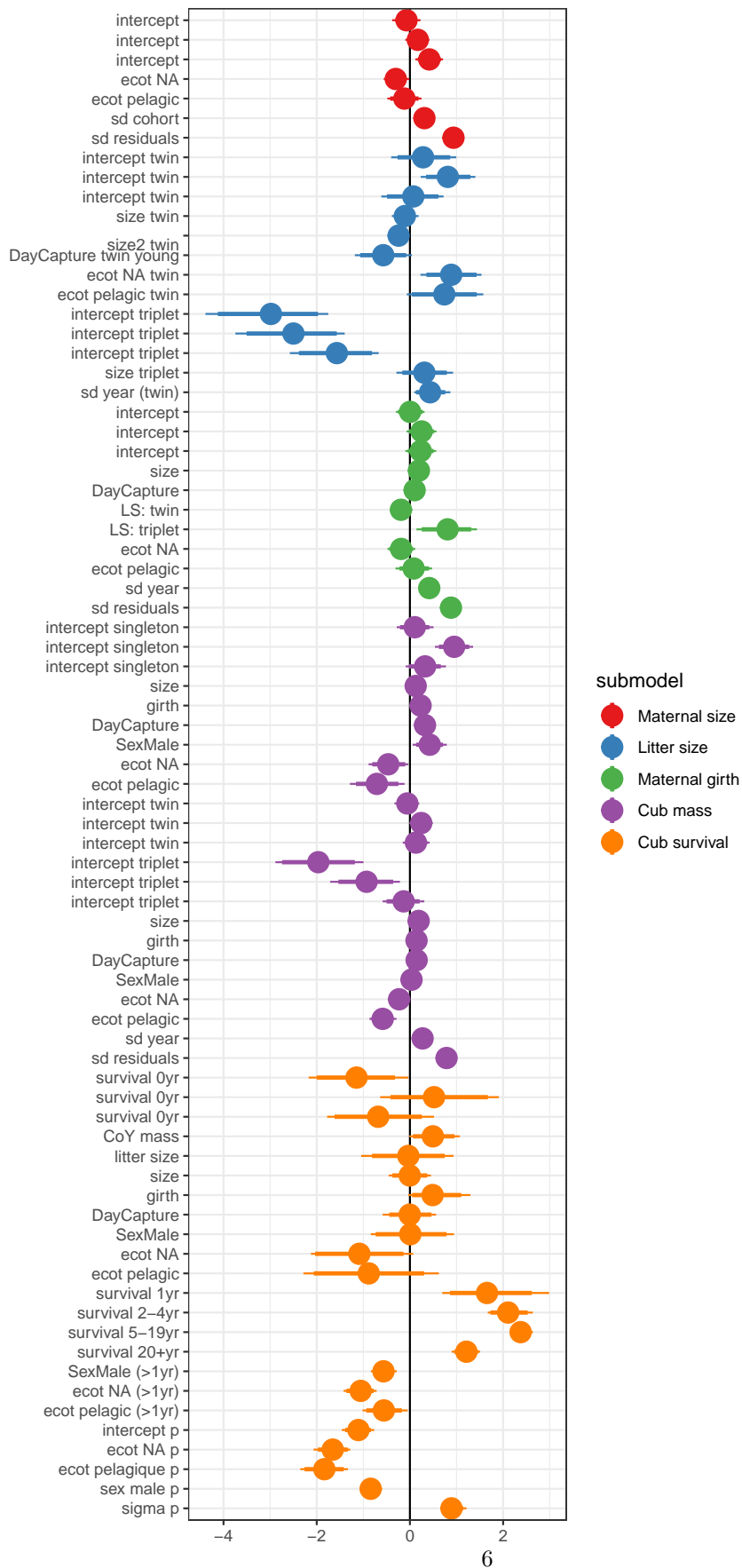
## 2. Rhat

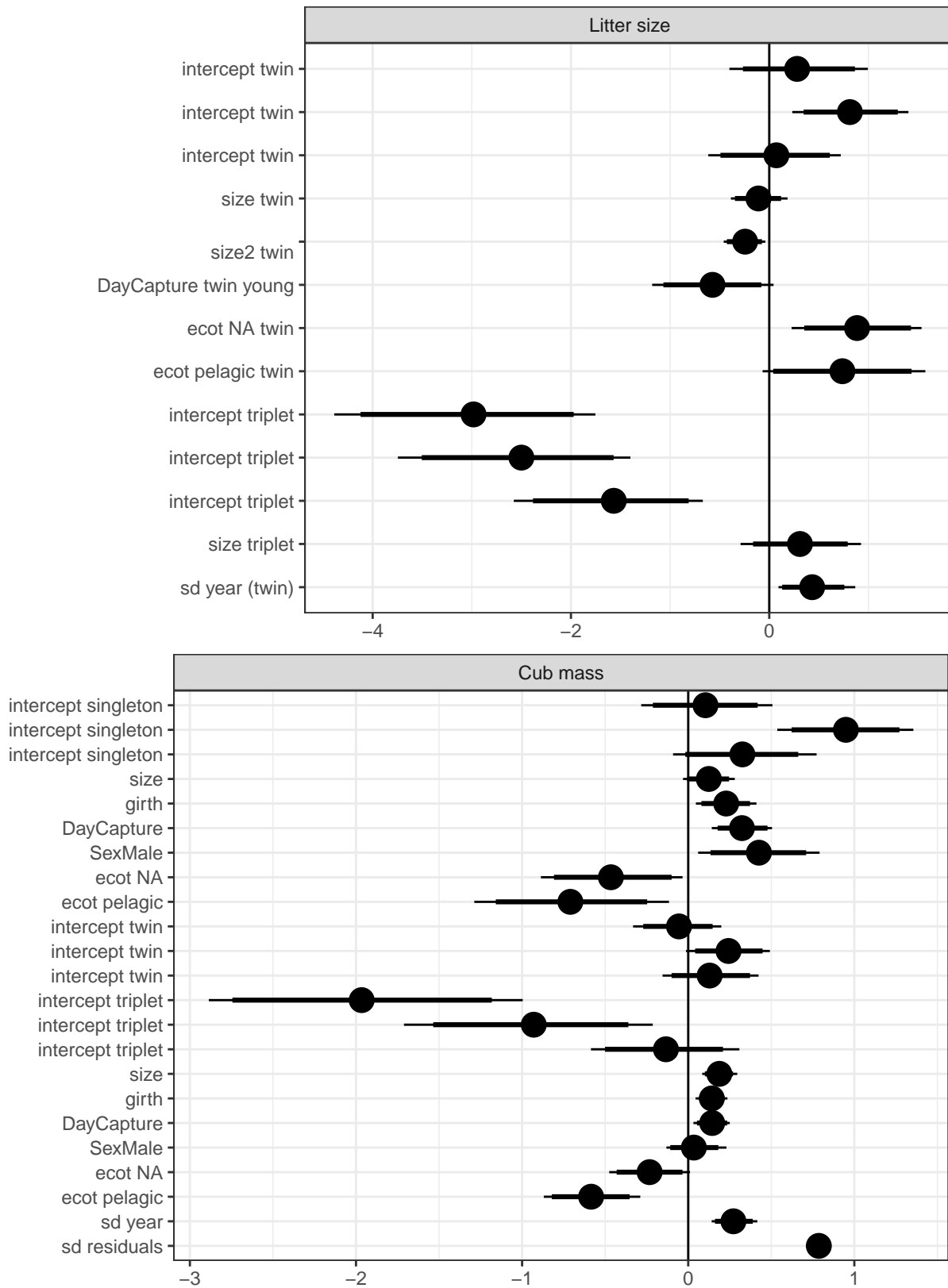
Calculate the Rhat value of each parameter, to make sure the chains have converged (Rhat < 1.1)

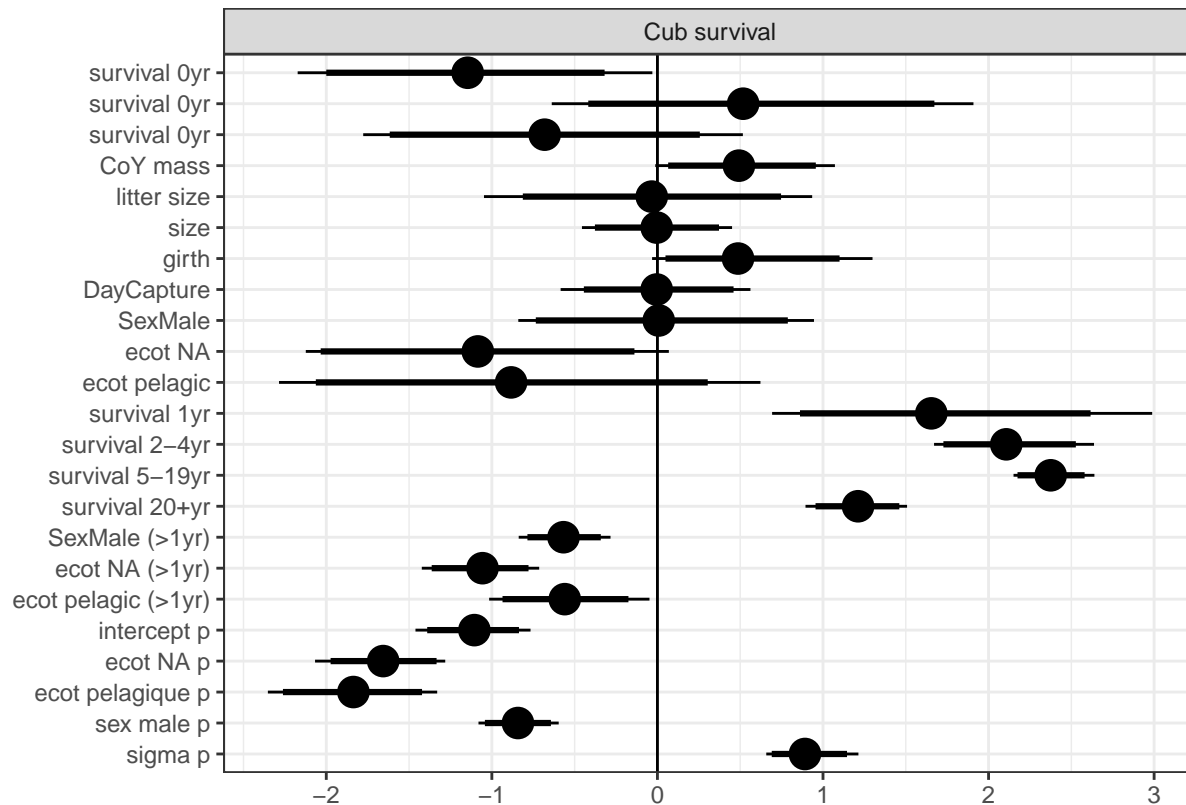


All Rhat are < 1.1

### 3. Caterpillar plots







## C. Compute residuals

The residuals computed here will be used to plot the observations corrected for variables other than that on the x axis for the figures.

a. Cub mass for Fig. 2A

b. Cub mass for Fig. 2B

c. Cub mass for Fig. 2C

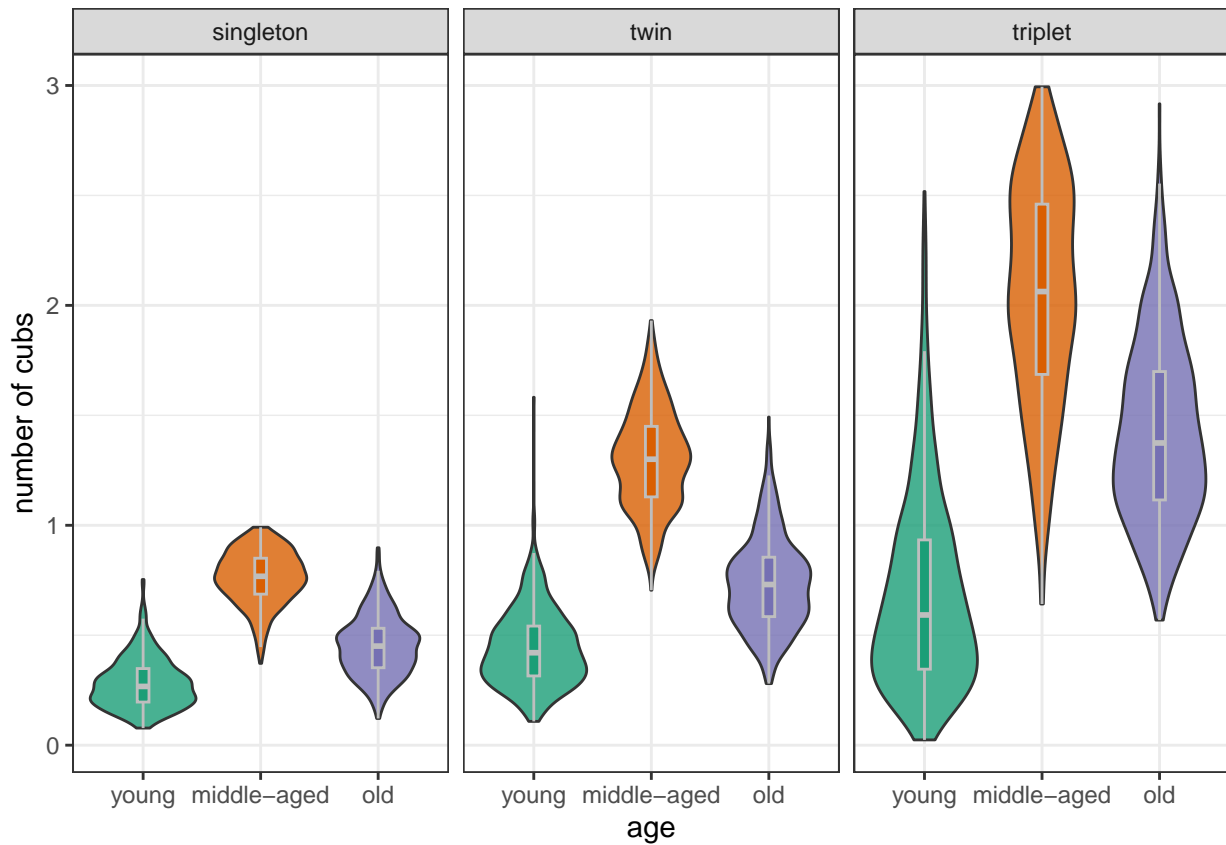
## D. Predict number of surviving cubs

Let's predict the number of surviving cubs by litter size and maternal age

We need to account for the fact that females who produce singleton/twin/triplet litters are not the same size on average. Let's compute the size of females corrected for the cohort random effect and the effect of the space-use strategy:

```
## 'summarise()' has grouped output by 'iteration'. You can override using the
## '.groups' argument.
## 'summarise()' has grouped output by 'iteration'. You can override using the
## '.groups' argument.
## 'summarise()' has grouped output by 'iteration'. You can override using the
## '.groups' argument.
```





## E. Compute probabilities of direction (pd)

### 1. Pd and effect sizes

#### a. Maternal length

```
## [1] 178 214
## [1] 1.598916
##      2.5%      97.5%
## -0.1957341  3.3576141
## [1] 0.9593333
## [1] 1.534974
##      2.5%      97.5%
## -0.5585596  3.6183635
## [1] 0.9257778
## [1] 3.13389
##      2.5%      97.5%
## 0.874069 5.344517
## [1] 0.9966667
## [1] -0.3062836
##      2.5%      97.5%
```

```
## -0.58484200 -0.02190895
## [1] 0.9828889
## [1] -0.1228604
##      2.5%      97.5%
## -0.4915318  0.2399341
## [1] 0.7393333
## [1] 0.1063801
##      2.5%      97.5%
## 0.01520452 0.26450646
```

#### **b. Litter size**

```
## [1] 1.685039
## [1] 0.5290126
## [1] 0.1181196
##      2.5%      97.5%
## -0.0329391  0.2706840
## [1] 0.9348889
## [1] 0.2001149
##      2.5%      97.5%
## 0.02470783 0.36702402
## [1] 0.9882222
## [1] -0.06871656
##      2.5%      97.5%
## -1.610856e-01 -1.350053e-05
## [1] 0.9748889
## [1] 0.0819953
##      2.5%      97.5%
## -0.1057886  0.2676468
## [1] 0.8068889
## [1] -0.07026202
##      2.5%      97.5%
## -0.163690840 0.001204978
## [1] 0.9733333
## [1] -0.2393557
##      2.5%      97.5%
## -0.45221461 -0.03318619
## [1] 0.9884444
## [1] 0.9715556
## [1] 0.9968889
```

```
## [1] 0.9635556
## [1] 0.2275493
##      2.5%      97.5%
## 0.008774673 0.737457806
```

#### c. Maternal girth

```
## [1] 94 135
## [1] 1.822906
##      2.5%      97.5%
## -0.1261582 3.7908947
## [1] 0.9662222
## [1] 0.2249699
##      2.5%      97.5%
## -2.099811 2.532180
## [1] 0.5797778
## [1] 1.597936
##      2.5%      97.5%
## -0.8217357 3.9966875
## [1] 0.9022222
## [1] 1.393393
##      2.5%      97.5%
## -0.4038563 3.1819140
## [1] 0.9366667
## [1] -5.798155
##      2.5%      97.5%
## -10.6329135 -0.9102755
## [1] 0.9902222
## [1] 0.9991111
## [1] 0.9304444
## [1] 0.876
## [1] 0.66
## [1] 0.183741
##      2.5%      97.5%
## 0.0589552 0.3937363
```

#### d. Cub mass

```
## [1] 3.00 31.25
## [1] 1.198992
##      2.5%      97.5%
## 1.096946 1.309545
```

```

## [1] 1
## [1] 1.463102
##      2.5%      97.5%
## 1.270121 1.680392
## [1] 1
## [1] 1.220944
##      2.5%      97.5%
## 1.076283 1.386007
## [1] 0.9993333
## [1] 1.312661
##      2.5%      97.5%
## 1.146553 1.497552
## [1] 1
## [1] 1.228099
##      2.5%      97.5%
## 1.057121 1.416323
## [1] 0.9964444
## [1] 1.072236
##      2.5%      97.5%
## 0.9242601 1.2358751
## [1] 0.812
## [1] 1.102309
##      2.5%      97.5%
## 1.028914 1.179866
## [1] 0.9971111
## [1] 1.038721
##      2.5%      97.5%
## 0.9548766 1.1258012
## [1] 0.8042222
## [1] 1.062718
##      2.5%      97.5%
## 0.9673298 1.1656765
## [1] 0.8955556
## [1] 0.1994588
##      2.5%      97.5%
## 0.03175342 0.37281896
## [1] 0.9895556
## [1] 0.1910185

```

```

##          2.5%          97.5%
## -0.0006108203  0.3885421931
## [1] 0.9744444
## [1] 0.008440393
##          2.5%          97.5%
## -0.1753051  0.1970066
## [1] 0.5386667
## [1] 0.9351111
## [1] 0.9995556
## [1] 0.9917778
## [1] 0.998
## [1] 0.9997778
## [1] 0.9966667
## [1] 1.147383
##          2.5%          97.5%
## 1.024216 1.282927
## [1] 1
## [1] 0.9873333
## [1] 0.9708889
## [1] 0.9924444
## [1] 1
## [1] 0.9917778
## [1] 0.6566667
## [1] 0.07836786
##          2.5%          97.5%
## 0.01959861 0.17793266

```

#### e. Litter mass

```

## singleton      twin    triplet
## 15.27660 25.49816 31.43350
## [1] 1.671562
##          2.5%          97.5%
## 1.527249 1.823244
## [1] 1
## [1] 1.233621
##          2.5%          97.5%
## 1.082246 1.393686
## [1] 0.9993333

```

#### **f. Cub survival**

```
## [1] 0.9666667
## [1] 0.3651858
##      2.5%      97.5%
## 0.1070290 0.6157603
## [1] 0.9966667
## [1] 0.2731349
##      2.5%      97.5%
## -0.0009016138 0.5448867798
## [1] 0.9733333
## [1] -0.09205086
##      2.5%      97.5%
## -0.3391568 0.1318436
## [1] 0.7783333
## [1] 0.54
## [1] -0.04333147
## [1] 0.505
## [1] -0.004916966
## [1] 0.9633333
## [1] 0.5033333
## [1] 0.5116667
## [1] 0.9566667
## [1] 0.8816667
```

#### **g. Recapture probability**

```
## [1] 0.2497323
##      2.5%      97.5%
## 0.1886972 0.3187844
## [1] 0.06095235
##      2.5%      97.5%
## 0.03825788 0.09017166
## [1] 0.05131676
##      2.5%      97.5%
## 0.02832574 0.08375710
## [1] 0.1262124
##      2.5%      97.5%
## 0.09053871 0.16843799
## [1] 0.8926378
```

```
##      2.5%      97.5%
## 0.6492742 1.2203507
```

#### **h. Productivity by litter size**

```
## [1] 0.1594975
##      2.5%      97.5%
## -0.1042502 0.4668759
## [1] 0.8733333
## [1] 0.5401565
##      2.5%      97.5%
## 0.2262592 0.8736775
## [1] 0.9983333
## [1] 0.2818197
##      2.5%      97.5%
## -0.04483155 0.65873897
## [1] 0.9466667
## [1] -0.2499186
##      2.5%      97.5%
## -1.2074287 0.3266533
## [1] 0.7133333
## [1] -0.7425002
##      2.5%      97.5%
## -1.32186771 0.07086852
## [1] 0.97
## [1] -0.6865163
##      2.5%      97.5%
## -1.3766590 -0.2153653
## [1] 1
```

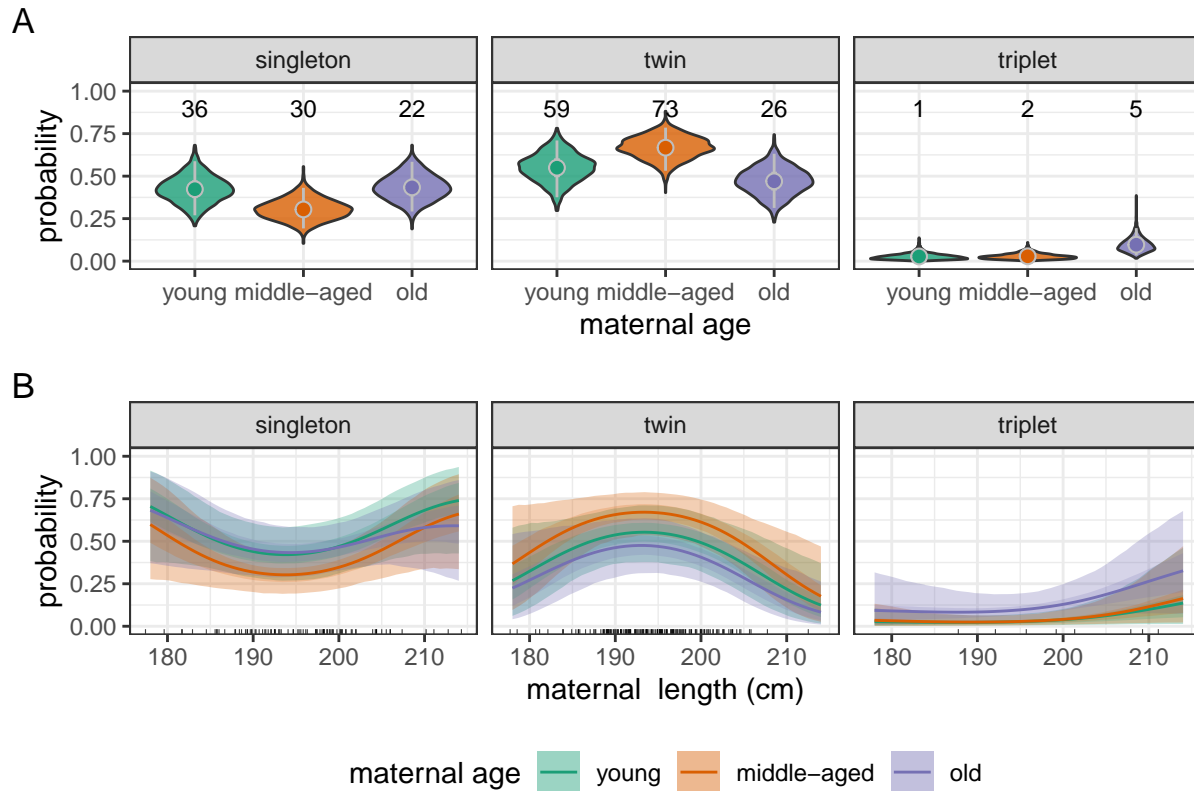
## **2. Variance explained by model**

```
## [1] 0.1232103
## [1] 0.2195112
## [1] 0.379511
```

## **F. Plot main figures**

### **1. Figure 2: Effects on litter size**

```
## 'summarise()' has grouped output by 'var', 'litter_size'. You can override
## using the '.groups' argument.
```



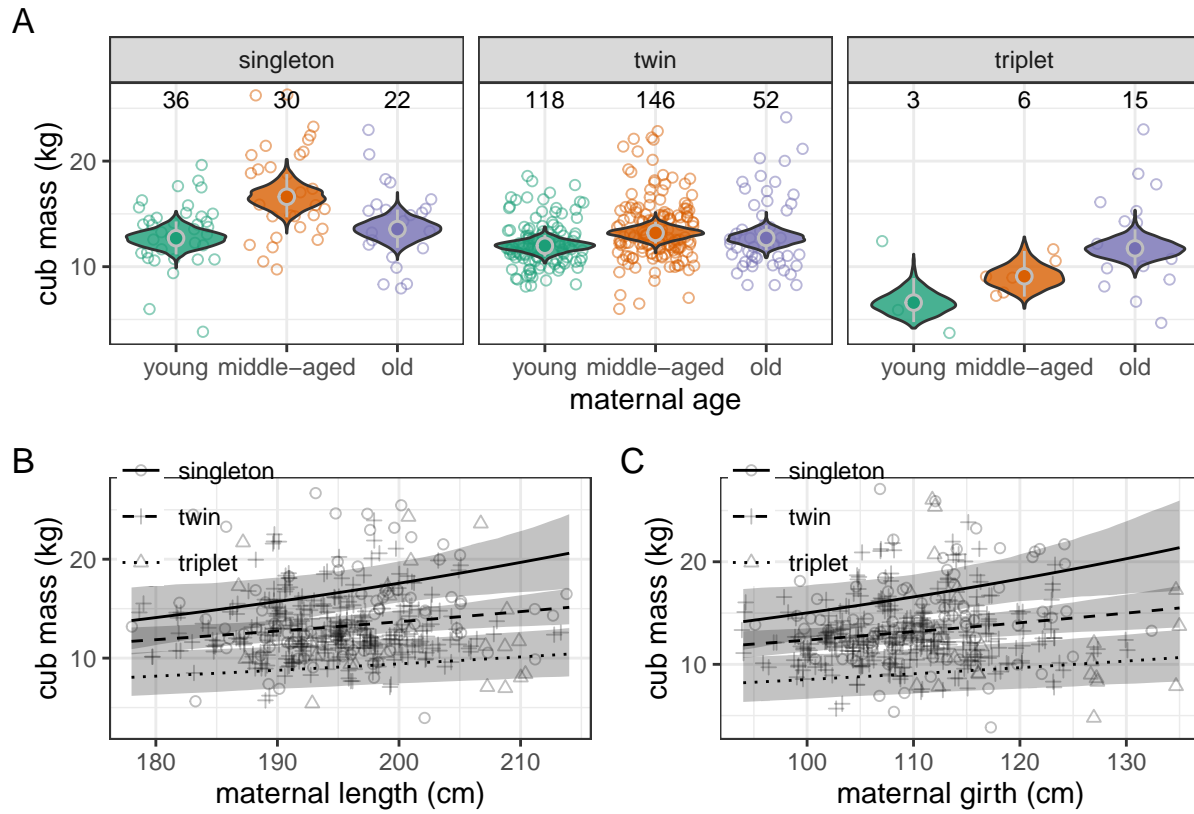
## 2. Figure 3: Determinants of cub mass

```
## 'summarise()' has grouped output by 'var'. You can override using the '.groups'
## argument.

## Warning: A numeric 'legend.position' argument in 'theme()' was deprecated in ggplot2
## 3.5.0.
## i Please use the 'legend.position.inside' argument of 'theme()' instead.
## This warning is displayed once every 8 hours.
## Call 'lifecycle::last_lifecycle_warnings()' to see where this warning was
## generated.

## 'summarise()' has grouped output by 'var'. You can override using the '.groups'
## argument.
```

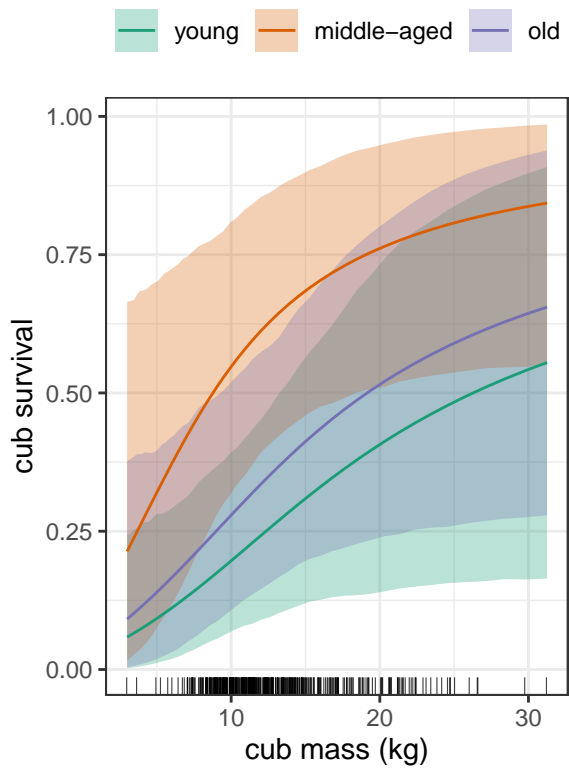




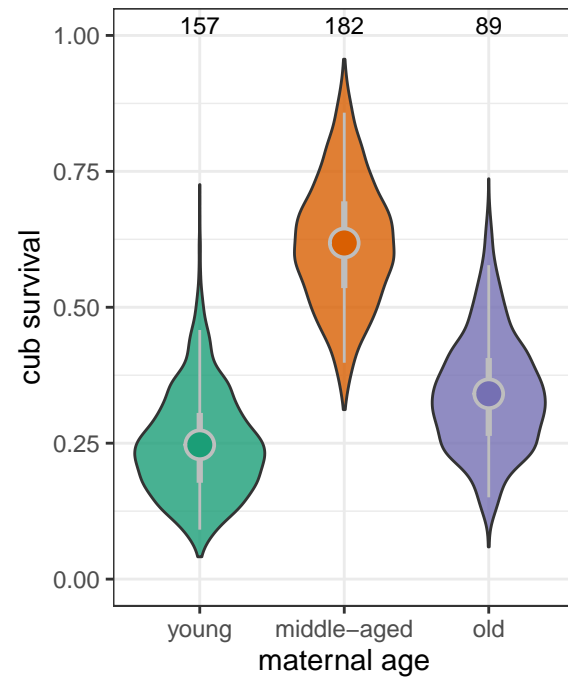
### 3. Figure 4: Determinants of cub survival

## 'summarise()' has grouped output by 'var'. You can override using the '.groups' argument.

A



B



4. Figure 6: Productivity by litter size

