Anolis trade-offs

Lailvaux, Cespedes, Houslay

10/12/2018

Overview

Data to test trade-offs among functional / performance traits in Anolis lizards.

Each lizard underwent up to 2 observation sessions, each of which includes up to 5 repeats of bite force and sprint speed, and 1 of endurance.

Our random effects levels will therefore be an individual-level effect, a within-individual among-session effect (for bite force and sprint speed, but not endurance), as well as residual variation (within-individual, within-session).

For multivariate models, we should be able to model correlations at the among-individual level and the within-individual among-session level. We do not model residual covariances as bite force and sprint speed are not measured at the same time for each trial within an assay.

Libraries and matrix functions

Load libraries:

```
library(asreml)
library(lme4)
library(tidyverse)
library(broom)
library(car)
library(RLRsim)
library(stringr)
```

Load a couple of matrix functions (first converts a vector to a matrix, given the upper triangle and the number of traits; second pulls out the id-related variance components from an asreml model and uses the previous function to return a matrix):

```
vecToMat <- function(X, n) {
  S <- diag(n)
  S[upper.tri(S, diag=TRUE)] <- X
  S <- S + t(S) - diag(diag(S))
  return(S)
}

getIMat <- function(asr_model, n) {

  # Extract variance components
  model_df <- data_frame(Var = row.names(summary(asr_model)$varcomp),
                         Num = summary(asr_model)$varcomp$component)

  model_df <- model_df %>%
    filter(substring(Var, 1, 9) == "trait:id!")

  return(vecToMat(model_df$Num, n))  ## Second value is number of traits
```

```
}
```

Create a quick function for standard error of the mean:

```
std_err <- function(x){
  x <- x[!is.na(x)]
  return(sd(x)/sqrt(length(x)))
}
```

Data

Load and take a quick look:

```
df_an_full <- read_csv("Data/Anolis mmm all data_corrected.csv")
```

```
## Parsed with column specification:
## cols(
##   id = col_character(),
##   assay = col_character(),
##   sex = col_character(),
##   trial = col_character(),
##   svl = col_double(),
##   mass = col_double(),
##   bite = col_double(),
##   sprint = col_double(),
##   endurance = col_double()
## )
```

```
glimpse(df_an_full)
```

```
## Observations: 1,210
## Variables: 9
## $ id        <chr> "L1", "L1", "L1", "L1", "L1", "L1", "L1", "L1", "L1"...
## $ assay     <chr> "a", "a", "a", "a", "a", "b", "b", "b", "b", "b", "a...
## $ sex       <chr> "m", "m", "m", "m", "m", "m", "m", "m", "m", "m", "m...
## $ trial     <chr> "a1", "a2", "a3", "a4", "a5", "b1", "b2", "b3", "b4"...
## $ svl       <dbl> 70.15, 70.15, 70.15, 70.15, 70.15, 70.15, 70.15, 70....
## $ mass      <dbl> 6.77, 6.77, 6.77, 6.77, 6.77, 6.77, 6.77, 6.77, 6.77...
## $ bite      <dbl> 12.80, 19.36, 11.55, 16.71, 16.78, 12.65, 12.01, 13....
## $ sprint    <dbl> 0.5827506, 0.4051864, 0.6203474, 0.3429355, NA, 0.53...
## $ endurance <dbl> 177.480, NA, NA, NA, NA, 288.104, NA, NA, NA, NA, 15...
```

A bit of wrangling required – get numeric versions of assay (0,1) and trial (-2:2).

Create a new variable called 'id_obs', which combines individual id with assay, enabling us to group repeated measures for an individual within a session (or assay) This means there are 3 variance levels to fit in each model (for bite and sprint, although not endurance):

- id = among-individual
- id_obs = within-individual, among-assay
- residual = within-individual, within-assay

```
df_an_full <- df_an_full %>%
  mutate(assay_num = ifelse(assay == "a", 0, 1),
         trial_num = as.numeric(str_sub(trial,2,2)) - 3) %>%
  unite(id_obs,
```

```
        id, assay,
        remove = FALSE) %>%
  arrange(sex, id)
```

We also create subsets for males and females for later use:

```
df_an_m <- df_an_full %>%
  filter(sex == "m")

df_an_f <- df_an_full %>%
  filter(sex == "f")
```

Analysis

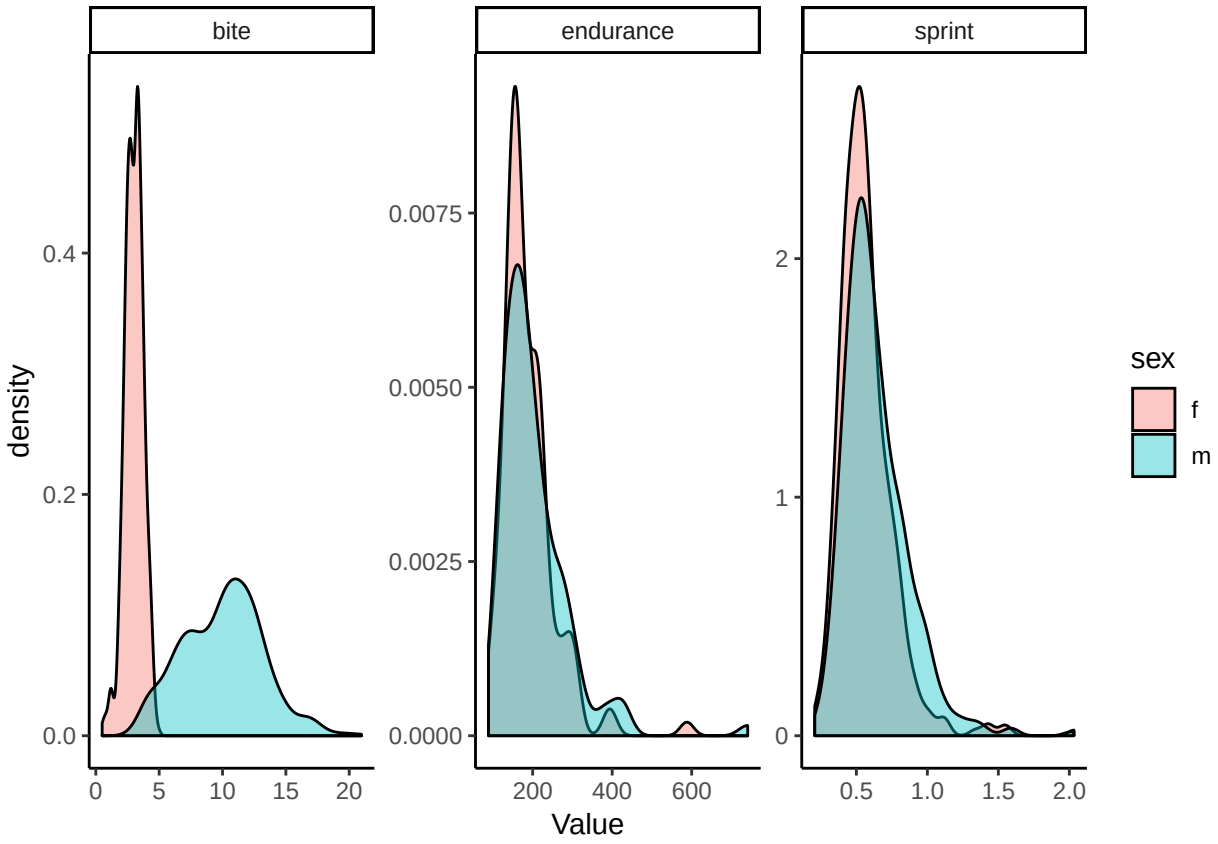
Univariate

Use univariate models to check model assumptions and test for significance of among-individual variation.

Note that the distribution of bite force has sex-specific mean *and* variance, so modelling these separately for each sex may prove to be a better option.

```
df_an_full %>%
  select(sex, bite:endurance) %>%
  gather(Trait, Value,
         bite:endurance) %>%
  ggplot(., aes(x = Value, fill = sex)) +
  geom_density(alpha = 0.4) +
  facet_wrap(~Trait, scales = "free") +
  theme_classic()
```

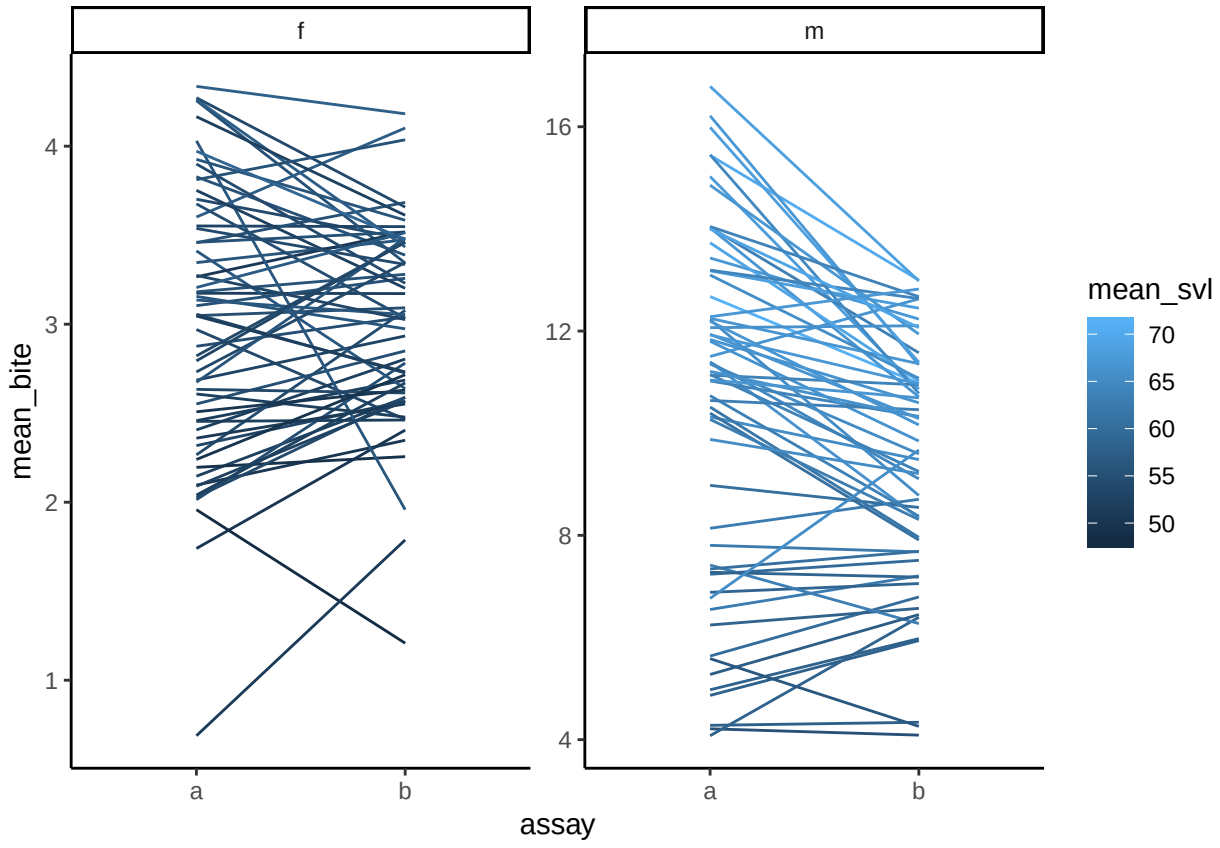
```
## Warning: Removed 1339 rows containing non-finite values (stat_density).
```



Plots

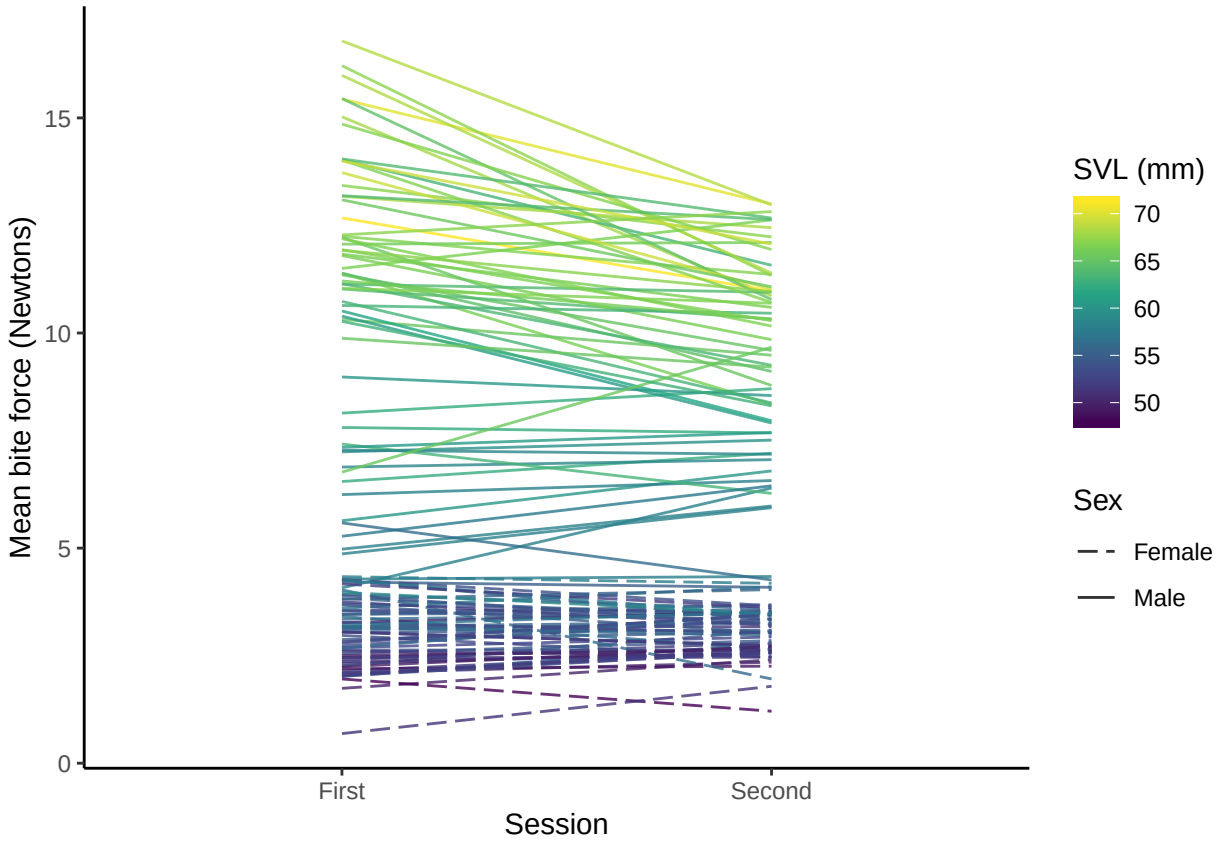
Plots of bite force show that there is a size x assay interaction in male lizards, where larger males tend to bite harder in the first assay relative to the second:

```
df_an_full %>%
  group_by(id, sex, assay) %>%
  summarise(mean_bite = mean(bite, na.rm = TRUE),
            sd_bite = sd(bite, na.rm = TRUE),
            mean_svl = mean(svl, na.rm = TRUE)) %>%
  ggplot(., aes(x = assay, y = mean_bite)) +
  geom_line(aes(group = id,
                colour = mean_svl)) +
  facet_wrap(~ sex, scales = "free") +
  theme_classic()
```



Create plots of bite force plasticity for manuscript

```
df_an_full %>%
  group_by(id, sex, assay) %>%
  summarise(mean_bite = mean(bite, na.rm = TRUE),
            sd_bite = sd(bite, na.rm = TRUE),
            mean_svl = mean(svl, na.rm = TRUE)) %>%
  ggplot(., aes(x = assay, y = mean_bite)) +
  geom_line(aes(group = id,
                colour = mean_svl,
                linetype = sex),
            alpha = 0.8) +
  scale_linetype_manual(values = c("longdash", "solid"),
                        labels = c("Female", "Male"),
                        name = "Sex") +
  scale_colour_viridis_c(name = "SVL (mm)") +
  scale_x_discrete(breaks = c("a","b"),
                   labels = c("First","Second")) +
  labs(x = "Session",
       y = "Mean bite force (Newtons)") +
  theme_classic()
```




```
ggsave("bite-sessions.png", width = 7, height = 7)
```

```
pdf(file = "bite-sessions.pdf", width = 7, height = 7)
df_an_full %>%
  group_by(id, sex, assay) %>%
  summarise(mean_bite = mean(bite, na.rm = TRUE),
            sd_bite = sd(bite, na.rm = TRUE),
            mean_svl = mean(svl, na.rm = TRUE)) %>%
  ggplot(., aes(x = assay, y = mean_bite)) +
  geom_line(aes(group = id,
                colour = mean_svl,
                linetype = sex),
            alpha = 0.8) +
  scale_linetype_manual(values = c("longdash", "solid"),
                        labels = c("Female", "Male"),
                        name = "Sex") +
  scale_colour_viridis_c(name = "SVL (mm)") +
  scale_x_discrete(breaks = c("a","b"),
                   labels = c("First","Second")) +
  labs(x = "Session",
       y = "Mean bite force (Newtons)") +
  theme_classic()
dev.off()
```

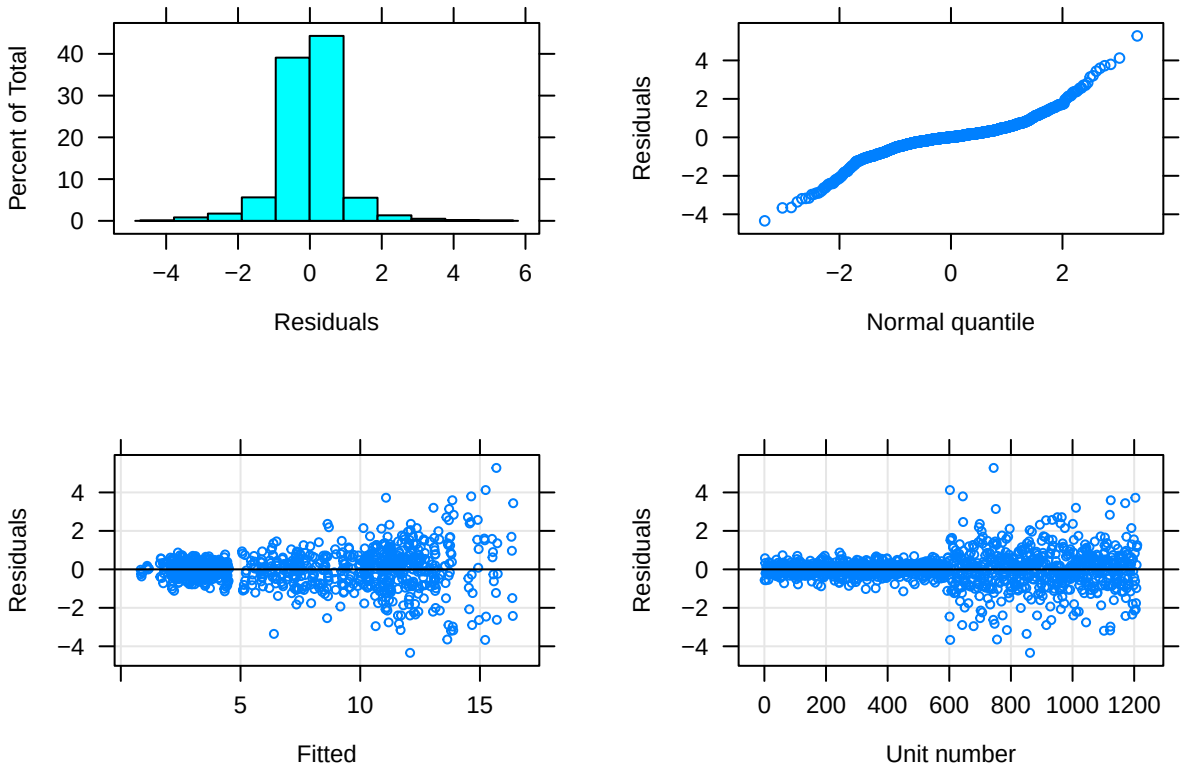
Bite force

Fitting sex-specific variances (at all levels) is a better fit to the data than just sex-specific means:

Clearly there is a pattern in the residuals that needs to be dealt with (the above plots of distributions indicate this is due to sex-specific variation in bite force):

```
asr_bite_0 <- asreml(bite ~ sex + scale(svl) + assay_num
                            + trial_num,
                     random =~ id + id_obs,
                     rcov =~ units,
                     maxiter = 200,
                     data = df_an_full)

plot(asr_bite_0)
```




```
asr_bite_1 <- asreml(bite ~ sex * scale(svl) * assay_num
                            + trial_num,
                     random =~ id + id_obs,
                     rcov =~ units,
                     maxiter = 200,
                     data = df_an_full)
```

```
asr_bite_sexspec <- asreml(bite ~ sex * scale(svl) * assay_num
                            + trial_num,
                           random =~ idh(sex):id + idh(sex):id_obs,
                           rcov =~ at(sex):units,
                           maxiter = 200,
                           data = df_an_full)
```

LRT on 3 df shows this is a much better fit to the data, with males having greater variance at each level (among-individual, within-individual among-assay, within-individual within-assay):

```
summary(asr_bite_1)$varcomp
```

```
##                         gamma component  std.error   z.ratio constraint
## id!id.var           0.8321167 0.6996674 0.13637177  5.130588   Positive
## id_obs!id_obs.var   0.5107499 0.4294530 0.07876377  5.452418   Positive
## R!variance          1.0000000 0.8408284 0.03841722 21.886757   Positive
```

```
summary(asr_bite_sexspec)$varcomp
```

```
##                         gamma  component   std.error   z.ratio constraint
## sex:id!sex.f      0.14097359 0.14097359 0.041387127  3.406218   Positive
## sex:id!sex.m      1.24909696 1.24909696 0.340678118  3.666502   Positive
## sex:id_obs!sex.f  0.12095257 0.12095257 0.026202047  4.616150   Positive
## sex:id_obs!sex.m  0.73264059 0.73264059 0.194670128  3.763498   Positive
## sex_f!variance    0.09946052 0.09946052 0.006446167 15.429405   Positive
## sex_m!variance    1.57297460 1.57297460 0.101320474 15.524746   Positive
```

```
pchisq(2*(asr_bite_sexspec$loglik - asr_bite_1$loglik), 3, lower.tail = FALSE)
```

```
## [1] 8.161275e-178
```

```
2*(asr_bite_sexspec$loglik - asr_bite_1$loglik)
```

```
## [1] 821.7838
```

Fixed effects also show that there is a significant 3-way interaction between sex, body size and assay. As in the plot above, heavier males bite harder, but this effect is reduced in the second assay:

```
summary(asr_bite_sexspec, all = TRUE)$coef.fixed
```

```
##                                solution   std error    z ratio
## sex_f:scale(svl):assay_num   0.00000000          NA         NA
## sex_m:scale(svl):assay_num -1.05716224 0.352998014 -2.9948107
## scale(svl):assay_num       -0.34694598 0.183363323 -1.8921231
## sex_f:assay_num             0.00000000          NA         NA
## sex_m:assay_num             0.35313638 0.361103145  0.9779377
## sex_f:scale(svl)            0.00000000          NA         NA
## sex_m:scale(svl)            3.40106349 0.365160549  9.3138854
## trial_num                  -0.01612074 0.008873125 -1.8168054
## assay_num                  -0.28686591 0.173044300 -1.6577599
## scale(svl)                  1.25773589 0.183340231  6.8601194
## sex_f                       0.00000000          NA         NA
## sex_m                       2.53936104 0.374257517  6.7850636
## (Intercept)                 4.09459643 0.173056559 23.6604522
```

```
asr_bite_sexspec_wald <- wald(asr_bite_sexspec, denDF = "numeric", ssType = "conditional")
```

```
asr_bite_sexspec_wald
```

```
## $Wald
##                            Df denDF    F.inc     F.con Margin           Pr
## (Intercept)                 1  71.9 4523.000 913.40000        1.328730e-42
## sex                         1 116.8 1514.000 175.40000      A 5.195321e-25
## scale(svl)                  1  92.5  169.200 169.30000      A 1.291001e-22
## assay_num                   1  73.6    3.677   3.67900      A 5.898511e-02
## trial_num                   1 537.1    3.365   3.30100      C 6.980410e-02
## sex:scale(svl)              1  93.7   80.700  80.75000      B 2.697356e-14
## sex:assay_num               1 116.5   33.350   0.02909      B 8.648660e-01
```

```
## scale(svl):assay_num      1  96.0   16.280  16.28000      B 1.097499e-04
## sex:scale(svl):assay_num  1  96.7    8.969   8.96900      C 3.487631e-03
##
## $stratumVariances
## NULL
```

Given the differences in variances across males and females, we use sex-specific models to test the significance of among-indiviual variance in bite force:

Males

```
asr_bite_m <- asreml(bite ~ scale(svl) * assay_num
                           + trial_num,
                     random =~ id + id_obs,
                     rcov =~ units,
                     maxiter = 200,
                     data = df_an_m)

asr_bite_m_testid <- asreml(bite ~ scale(svl) * assay_num
                           + trial_num,
                     random =~ id_obs,
                     rcov =~ units,
                     maxiter = 200,
                     data = df_an_m)
```

```
summary(asr_bite_m)$varcomp
```

```
##                       gamma component std.error   z.ratio constraint
## id!id.var         0.7926628 1.2492239 0.3407045  3.666591   Positive
## id_obs!id_obs.var 0.4644991 0.7320432 0.1946790  3.760257   Positive
## R!variance        1.0000000 1.5759841 0.1016132 15.509634   Positive
```

```
0.5*pchisq(2*(asr_bite_m$loglik - asr_bite_m_testid$loglik), 1, lower.tail = FALSE)
```

```
## [1] 2.800069e-06
```

```
2*(asr_bite_m$loglik - asr_bite_m_testid$loglik)
```

```
## [1] 20.62022
```

Females

```
asr_bite_f <- asreml(bite ~ scale(svl) * assay_num
                           + trial_num,
                     random =~ id + id_obs,
                     rcov =~ units,
                     maxiter = 200,
                     data = df_an_f)

asr_bite_f_testid <- asreml(bite ~ scale(svl) * assay_num
                           + trial_num,
                     random =~ id_obs,
                     rcov =~ units,
                     maxiter = 200,
                     data = df_an_f)
```

```
summary(asr_bite_f)$varcomp
```

```
##                       gamma  component   std.error   z.ratio constraint
## id!id.var          1.417191 0.14097202 0.041386961  3.406194   Positive
## id_obs!id_obs.var 1.215920 0.12095101 0.026202232  4.616057   Positive
## R!variance         1.000000 0.09947284 0.006447365 15.428449   Positive
```

```
0.5*pchisq(2*(asr_bite_f$loglik - asr_bite_f_testid$loglik), 1, lower.tail = FALSE)
```

```
## [1] 2.193153e-05
```

```
2*(asr_bite_f$loglik - asr_bite_f_testid$loglik)
```

```
## [1] 16.69645
```

Results

Males show higher bite force than females, and among males there is a strong effect of size: larger males bite harder, although this effect decreases from the first assay to the second.

Males and females show significant differences in bite force variance. We find significant among-individual variance in bite force for both males and females.

Short-term repeatability is the ratio of individual plus individual-assay variance to total phenotypic variance (after adjusting for fixed effects):

Female

```
nadiv:::pin(asr_bite_sexspec, short_R_bite_f ~ (V1+V3)/(V1+V3+V5))
```

```
##                  Estimate         SE
## short_R_bite_f 0.7247809 0.03153259
```

Male

```
nadiv:::pin(asr_bite_sexspec, short_R_bite_m ~ (V2+V4)/(V2+V4+V6))
```

```
##                  Estimate         SE
## short_R_bite_m 0.5574959 0.04248538
```

Long-term repeatability is the ratio of individual variance to total phenotypic variance (after adjusting for fixed effects):

Female

```
nadiv:::pin(asr_bite_sexspec, long_R_bite_f ~ (V1)/(V1+V3+V5))
```

```
##                 Estimate         SE
## long_R_bite_f 0.3900907 0.08322735
```

Male

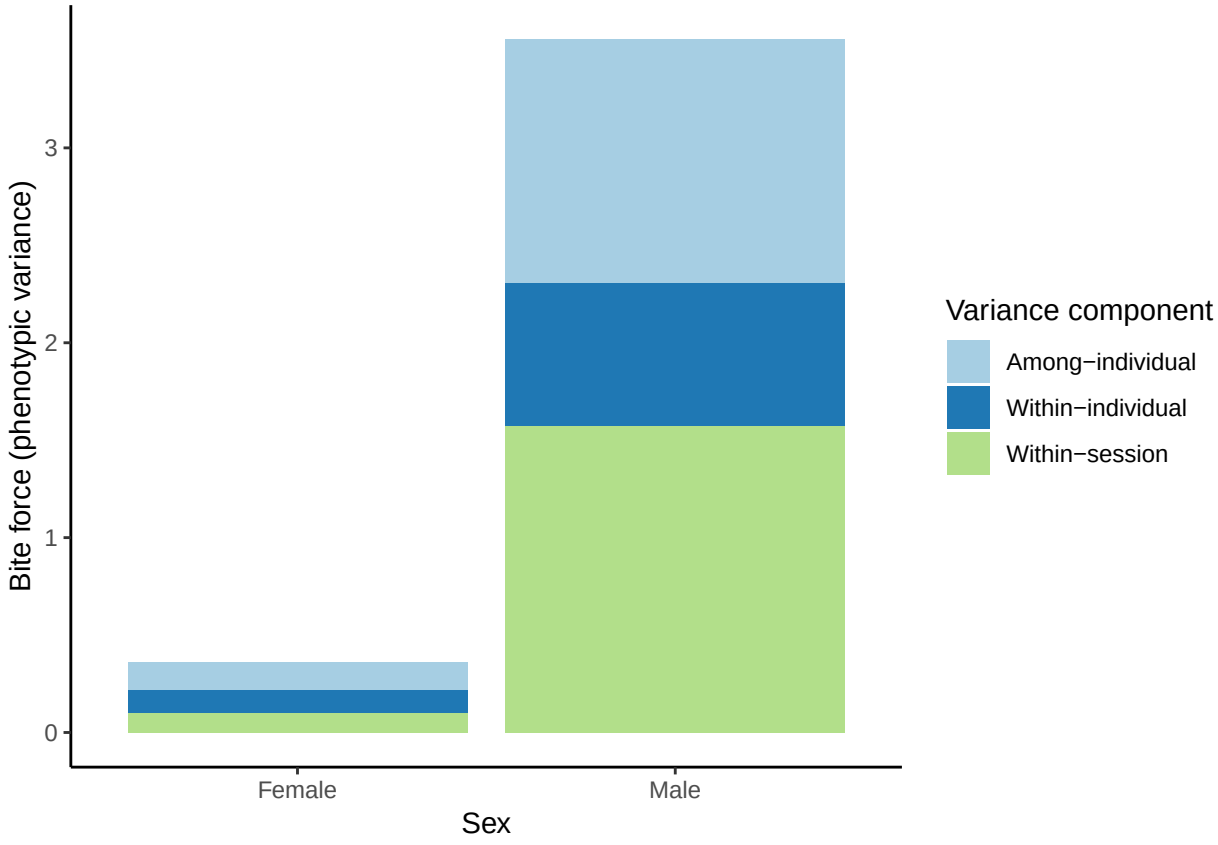
```
nadiv:::pin(asr_bite_sexspec, long_R_bite_m ~ (V2)/(V2+V4+V6))
```

```
##                 Estimate         SE
## long_R_bite_m 0.3513919 0.07007399
```

Save figures of variance component proportions, for males and females separately:

```
data_frame(Sex = rep(c("Female","Male"),times = 3),
           Component = rep(c("Among-individual","Within-individual","Within-session"),each=2),
           Variance = summary(asr_bite_sexspec)$varcomp[,"component"]) %>%
```

```
  ggplot(., aes(x = Sex, y = Variance, fill = Component)) +
  geom_bar(stat = "identity") +
  ylab("Bite force (phenotypic variance)") +
  scale_fill_manual(values = c("#a6cee3",
                               "#1f78b4",
                               "#b2df8a"),
                    name = "Variance component") +
  #scale_fill_viridis_d() +
  theme_classic()
```




```
ggsave("varcomp.png")
```

```
## Saving 6.5 x 4.5 in image
```

```
pdf(file = "varcomp.pdf", width = 7, height = 7)
data_frame(Sex = rep(c("Female","Male"),times = 3),
           Component = rep(c("Among-individual","Within-individual","Within-session"),each=2),
           Variance = summary(asr_bite_sexspec)$varcomp[,"component"]) %>%
  ggplot(., aes(x = Sex, y = Variance, fill = Component)) +
  geom_bar(stat = "identity") +
  ylab("Bite force (phenotypic variance)") +
  scale_fill_manual(values = c("#a6cee3",
                               "#1f78b4",
                               "#b2df8a"),
                    name = "Variance component") +
  #scale_fill_viridis_d() +
  theme_classic()
dev.off()
```

Sprint

Given the above, first just check that males and females do not have different variances (although unlikely based on density plots of the raw data):

```
df_an_full %>%
  group_by(sex) %>%
  summarise(mean_sprint = mean(sprint, na.rm = TRUE),
            se_sprint = std_err(sprint))
```

```
## # A tibble: 2 x 3
##   sex   mean_sprint se_sprint
##   <chr>       <dbl>     <dbl>
## 1 f           0.570   0.00998
## 2 m           0.636   0.0106
```

```
asr_sprint_3 <- asreml(log(sprint) ~ sex + scale(svl) + assay_num
                            + trial_num,
                            random =~ id + id_obs,
                            rcov =~ units,
                            maxiter = 200,
                            data = df_an_full)

summary(asr_sprint_3, all = TRUE)$coef.fixed
```

```
asr_sprint_3_wald <- wald(asr_sprint_3, denDF = "numeric", ssType = "conditional")
```

```
asr_sprint_3_wald
```

```
## $Wald
##             Df denDF    F.inc    F.con Margin           Pr
## (Intercept)  1 115.4 959.3000 476.8000        8.566345e-43
## sex          1 114.5   6.8240   0.7726      A 3.812446e-01
## scale(svl)   1 117.5   0.3354   0.1781      A 6.737950e-01
## assay_num    1 117.0  35.5300  36.3700      A 1.956137e-08
## trial_num    1 656.1   1.1170   1.1170      A 2.908760e-01
##
## $stratumVariances
##                    df   Variance       id   id_obs R!variance
## id          119.0063 0.26281565 6.374933 3.297897          1
## id_obs      115.6933 0.14087152 0.000000 3.309809          1
## R!variance 609.3003 0.05332911 0.000000 0.000000          1
```

```
asr_sprint_sexspec <- asreml(log(sprint) ~ sex + scale(svl) + assay_num
                            + trial_num,
                            random =~ idh(sex):id + idh(sex):id_obs,
                            rcov =~ at(sex):units,
                            maxiter = 200,
                            data = df_an_full)
```

LRT on 3 df shows this is not a better fit to the data:

```
summary(asr_sprint_3)$varcomp
```

```
##                         gamma  component   std.error   z.ratio constraint
## id!id.var           0.3596181 0.01917811 0.006078179  3.155240   Positive
## id_obs!id_obs.var 0.4959652 0.02644938 0.005671672  4.663419   Positive
```

```
## R!variance        1.0000000 0.05332911 0.003055369 17.454230   Positive
```

```
summary(asr_sprint_sexspec)$varcomp
```

```
##                       gamma  component   std.error   z.ratio constraint
## sex:id!sex.f     0.01936997 0.01936997 0.008634056  2.243438   Positive
## sex:id!sex.m     0.01905365 0.01905365 0.008709547  2.187674   Positive
## sex:id_obs!sex.f 0.02459368 0.02459368 0.008032773  3.061668   Positive
## sex:id_obs!sex.m 0.02819408 0.02819408 0.008364042  3.370867   Positive
## sex_f!variance   0.05351557 0.05351557 0.004485555 11.930647   Positive
## sex_m!variance   0.05316585 0.05316585 0.004174189 12.736807   Positive
```

```
pchisq(2*(asr_sprint_sexspec$loglik - asr_sprint_3$loglik), 3, lower.tail = FALSE)
```

```
## [1] 0.9915409
```

```
2*(asr_sprint_sexspec$loglik - asr_sprint_3$loglik)
```

```
## [1] 0.1024572
```

```
asr_sprint_3_testid <- asreml(log(sprint) ~ sex + scale(svl) + assay_num
                              + trial_num,
                              random =~ id_obs,
                              rcov =~ units,
                              maxiter = 200,
                              data = df_an_full)
```

```
summary(asr_sprint_3)$varcomp
```

```
##                       gamma  component   std.error   z.ratio constraint
## id!id.var         0.3596181 0.01917811 0.006078179  3.155240   Positive
## id_obs!id_obs.var 0.4959652 0.02644938 0.005671672  4.663419   Positive
## R!variance        1.0000000 0.05332911 0.003055369 17.454230   Positive
```

```
0.5*pchisq(2*(asr_sprint_3$loglik - asr_sprint_3_testid$loglik), 1, lower.tail = FALSE)
```

```
## [1] 0.0004186532
```

```
2*(asr_sprint_3$loglik - asr_sprint_3_testid$loglik)
```

```
## [1] 11.15666
```

Short-term repeatability

```
nadiv:::pin(asr_sprint_3, short_R ~ (V1+V2)/(V1+V2+V3))
```

```
##          Estimate         SE
## short_R 0.4610859 0.03788123
```

Long-term repeatability

```
nadiv:::pin(asr_sprint_3, long_R ~ (V1)/(V1+V2+V3))
```

```
##         Estimate         SE
## long_R 0.1938033 0.05573822
```

Endurance

As above, first just check that males and females do not have different variances (although unlikely based on density plots of the raw data):

```
df_an_full %>%
  group_by(sex) %>%
  summarise(mean_endurance = mean(endurance, na.rm = TRUE),
            se_endurance = std_err(endurance))
```

```
## # A tibble: 2 x 3
##   sex   mean_endurance se_endurance
##   <chr>          <dbl>        <dbl>
## 1 f               188.         6.16
## 2 m               203.         7.82
```

```
asr_end_3 <- asreml(log(endurance) ~ sex + scale(svl) + assay_num,
                          random =~ id,
                          rcov =~ units,
                          maxiter = 200,
                          data = df_an_full)

summary(asr_end_3, all = TRUE)$coef.fixed
```

```
asr_end_3_wald <- wald(asr_end_3, denDF = "numeric", ssType = "conditional")
```

```
asr_end_3_wald
```

```
## $Wald
##              Df denDF     F.inc     F.con Margin            Pr
## (Intercept)   1 117.0 62390.000 33500.000        1.055044e-145
## sex           1 116.7     1.957     9.556      A  2.493357e-03
## scale(svl)    1 117.3    19.540    19.630      A  2.122746e-05
## assay_num     1 119.0     3.554     3.554      A  6.183129e-02
##
## $stratumVariances
##                   df   Variance       id R!variance
## id         118.5208 0.10502539 1.981415          1
## R!variance 118.4792 0.09421128 0.000000          1
```

```
asr_end_sexspec <- asreml(log(endurance) ~ sex + scale(svl) + assay_num
                            + trial_num,
                            random =~ idh(sex):id,
                            rcov =~ units,
                            maxiter = 200,
                            data = df_an_full)
```

LRT on 3 df shows this is not a better fit to the data:

```
summary(asr_end_3)$varcomp
```

```
##                 gamma   component   std.error   z.ratio constraint
## id!id.var  0.05793119 0.005457771 0.009250589 0.5899918   Positive
## R!variance 1.00000000 0.094211283 0.012240433 7.6967278   Positive
```

```
summary(asr_end_sexspec)$varcomp
```

```
##                     gamma    component    std.error  z.ratio constraint
## sex:id!sex.f 4.677277e-08 4.307050e-09 4.569749e-10 9.425136   Boundary
## sex:id!sex.m 1.634694e-01 1.505300e-02 1.223735e-02 1.230087   Positive
## R!variance   1.000000e+00 9.208456e-02 9.770105e-03 9.425136   Positive
```

```
0.5*pchisq(2*(asr_end_sexspec$loglik - asr_end_3$loglik), 1, lower.tail = FALSE)
```

```
## [1] 0.1085942
```

```
2*(asr_end_sexspec$loglik - asr_end_3$loglik)
```

```
## [1] 1.522852
```

```
asr_end_3_testid <- asreml(log(endurance) ~ sex + scale(svl) + assay_num,
                           rcov =~ units,
                           maxiter = 200,
                           data = df_an_full)
```

```
0.5*pchisq(2*(asr_end_3$loglik - asr_end_3_testid$loglik), 1, lower.tail = FALSE)
```

```
## [1] 0.278295
```

```
2*(asr_end_3$loglik - asr_end_3_testid$loglik)
```

```
## [1] 0.3456431
```

Multivariate

Do logging and scaling in data frame so that sex-specific models are comparable:

```
df_an_full_sc <- df_an_full %>%
  mutate(bite = bite/sd(bite, na.rm=TRUE),
         sprint = log(sprint)/sd(log(sprint), na.rm=TRUE),
         endurance = log(endurance)/sd(log(endurance), na.rm=TRUE))

df_an_sc_m <- df_an_full_sc %>% filter(sex == "m")
df_an_sc_f <- df_an_full_sc %>% filter(sex == "f")
```

Set up some (co)variance / correlation structures for the multivariate models.

First one is for among-individual level, as we don't have any among-individual variance in endurance. We therefore set that to (very close to) zero, and any correlations involving endurance at that level are set to 0.

Second is for the residual (within-individual within-assay) variance, where we estimate this for bite and sprint only (fixing endurance to very close to 0).

```
init_I_cor <- c(0.1,
                0,0,
                1,1,1e-8)

names(init_I_cor) <- c("U","F","F",
                       "P","P","F")



init_R_idh <- c(1,1,1e-8)

names(init_R_idh) <- c("P","P","F")
```

Female

Full model:

```
asr_mv_bse_f_1 <- asreml(cbind(bite, sprint, endurance) ~ trait +
                           trait:(scale(svl) + assay_num) +
                           at(trait,1:2):(trial_num),
                         random =~ corgh(trait, init=init_I_cor):id +
                           us(trait):id_obs,
                         rcov =~ units:idh(trait, init = init_R_idh),
                         data = df_an_sc_f)
```

```
summary(asr_mv_bse_f_1)$varcomp
```

```
##                                                 gamma    component
## trait:id!trait.sprint:!trait.bite.cor      -0.463424520 -0.463424520
## trait:id!trait.endurance:!trait.bite.cor    0.000000000  0.000000000
## trait:id!trait.endurance:!trait.sprint.cor  0.000000000  0.000000000
## trait:id!trait.bite                         0.007975970  0.007975970
## trait:id!trait.sprint                       0.172912109  0.172912109
## trait:id!trait.endurance                    0.000000010  0.000000010
## trait:id_obs!trait.bite:bite                0.007074624  0.007074624
## trait:id_obs!trait.sprint:bite             -0.002657024 -0.002657024
## trait:id_obs!trait.sprint:sprint            0.221977338  0.221977338
## trait:id_obs!trait.endurance:bite           0.017198452  0.017198452
## trait:id_obs!trait.endurance:sprint        -0.083371741 -0.083371741
## trait:id_obs!trait.endurance:endurance      0.829327437  0.829327437
## R!variance                                  1.000000000  1.000000000
## R!trait.bite                                0.005655357  0.005655357
## R!trait.sprint                              0.506795179  0.506795179
## R!trait.endurance                           0.000000010  0.000000010
##                                               std.error    z.ratio
## trait:id!trait.sprint:!trait.bite.cor      0.2376868440 -1.9497273
## trait:id!trait.endurance:!trait.bite.cor             NA         NA
## trait:id!trait.endurance:!trait.sprint.cor           NA         NA
## trait:id!trait.bite                        0.0023260615  3.4289593
## trait:id!trait.sprint                      0.0776712391  2.2262051
## trait:id!trait.endurance                             NA         NA
## trait:id_obs!trait.bite:bite               0.0014881873  4.7538529
## trait:id_obs!trait.sprint:bite             0.0073045424 -0.3637495
## trait:id_obs!trait.sprint:sprint           0.0735554352  3.0178237
## trait:id_obs!trait.endurance:bite          0.0093918365  1.8312128
## trait:id_obs!trait.endurance:sprint        0.0605036334 -1.3779626
## trait:id_obs!trait.endurance:endurance     0.1088889266  7.6162697
## R!variance                                           NA         NA
## R!trait.bite                               0.0003665533 15.4284702
## R!trait.sprint                             0.0424749980 11.9316116
## R!trait.endurance                                    NA         NA
##                                             constraint
## trait:id!trait.sprint:!trait.bite.cor      Unconstrained
## trait:id!trait.endurance:!trait.bite.cor           Fixed
## trait:id!trait.endurance:!trait.sprint.cor         Fixed
## trait:id!trait.bite                         Positive
## trait:id!trait.sprint                       Positive
## trait:id!trait.endurance                       Fixed
## trait:id_obs!trait.bite:bite                Positive
## trait:id_obs!trait.sprint:bite              Positive
## trait:id_obs!trait.sprint:sprint            Positive
```

```
## trait:id_obs!trait.endurance:bite               Positive
## trait:id_obs!trait.endurance:sprint             Positive
## trait:id_obs!trait.endurance:endurance          Positive
## R!variance                                         Fixed
## R!trait.bite                                    Positive
## R!trait.sprint                                  Positive
## R!trait.endurance                                  Fixed
```

```
nadiv:::pin(asr_mv_bse_f_1, r ~ V8/(sqrt(V7)*sqrt(V9)))
```

```
##      Estimate        SE
## r -0.06704856 0.1836386
```

```
nadiv:::pin(asr_mv_bse_f_1, r ~ V10/(sqrt(V7)*sqrt(V12)))
```

```
##    Estimate        SE
## r 0.2245302 0.1167944
```

```
nadiv:::pin(asr_mv_bse_f_1, r ~ V11/(sqrt(V12)*sqrt(V9)))
```

```
##     Estimate        SE
## r -0.1943129 0.1351551
```

Bite and sprint only for testing hypotheses / looking at covariances:

```
asr_mv_bs_f_1 <- asreml(cbind(bite, sprint) ~ trait +
                          trait:(scale(svl) + assay_num) +
                          at(trait,1:2):(trial_num),
                        random =~ us(trait):id +
                          us(trait):id_obs,
                        rcov =~ units:idh(trait),
                        data = df_an_sc_f)
```

```
summary(asr_mv_bs_f_1)$varcomp
```

```
##                                        gamma    component    std.error
## trait:id!trait.bite:bite           0.007839918  0.007839918 0.0023613468
## trait:id!trait.sprint:bite        -0.019476098 -0.019476098 0.0099477615
## trait:id!trait.sprint:sprint       0.196506998  0.196506998 0.0802755894
## trait:id_obs!trait.bite:bite       0.007226165  0.007226165 0.0015415726
## trait:id_obs!trait.sprint:bite    -0.001039942 -0.001039942 0.0072168090
## trait:id_obs!trait.sprint:sprint   0.205563767  0.205563767 0.0696987642
## R!variance                         1.000000000  1.000000000           NA
## R!trait.bite                       0.005655438  0.005655438 0.0003665611
## R!trait.sprint                     0.506512315  0.506512315 0.0424384222
##                                     z.ratio constraint
## trait:id!trait.bite:bite           3.320105   Positive
## trait:id!trait.sprint:bite        -1.957837   Positive
## trait:id!trait.sprint:sprint       2.447905   Positive
## trait:id_obs!trait.bite:bite       4.687528   Positive
## trait:id_obs!trait.sprint:bite    -0.144100   Positive
## trait:id_obs!trait.sprint:sprint   2.949317   Positive
## R!variance                               NA      Fixed
## R!trait.bite                      15.428365   Positive
## R!trait.sprint                    11.935230   Positive
```

Male

```
asr_mv_bse_m_1 <- asreml(cbind(bite, sprint, endurance) ~ trait +
                           at(trait,1):(scale(svl) * assay_num) +
                           at(trait,2:3):(scale(svl) + assay_num) +
                           trait:trial_num,
                         random =~ corgh(trait, init=init_I_cor):id +
                           us(trait):id_obs,
                         rcov =~ units:idh(trait, init = init_R_idh),
                         data = df_an_sc_m,
                         maxiter = 200)
```

```
summary(asr_mv_bse_m_1)$varcomp
```

```
##                                                 gamma    component
## trait:id!trait.sprint:!trait.bite.cor      -0.061908055 -0.061908055
## trait:id!trait.endurance:!trait.bite.cor    0.000000000  0.000000000
## trait:id!trait.endurance:!trait.sprint.cor  0.000000000  0.000000000
## trait:id!trait.bite                         0.071274315  0.071274315
## trait:id!trait.sprint                       0.182562494  0.182562494
## trait:id!trait.endurance                    0.000000010  0.000000010
## trait:id_obs!trait.bite:bite                0.041585255  0.041585255
## trait:id_obs!trait.sprint:bite             -0.009107093 -0.009107093
## trait:id_obs!trait.sprint:sprint            0.244261274  0.244261274
## trait:id_obs!trait.endurance:bite           0.029035441  0.029035441
## trait:id_obs!trait.endurance:sprint        -0.054517338 -0.054517338
## trait:id_obs!trait.endurance:endurance      1.006787663  1.006787663
## R!variance                                  1.000000000  1.000000000
## R!trait.bite                                0.089595623  0.089595623
## R!trait.sprint                              0.504995509  0.504995509
## R!trait.endurance                           0.000000010  0.000000010
##                                              std.error    z.ratio
## trait:id!trait.sprint:!trait.bite.cor      0.24084416 -0.2570461
## trait:id!trait.endurance:!trait.bite.cor           NA         NA
## trait:id!trait.endurance:!trait.sprint.cor         NA         NA
## trait:id!trait.bite                        0.01931506  3.6900896
## trait:id!trait.sprint                      0.07990563  2.2847263
## trait:id!trait.endurance                           NA         NA
## trait:id_obs!trait.bite:bite               0.01102136  3.7731501
## trait:id_obs!trait.sprint:bite             0.02025641 -0.4495907
## trait:id_obs!trait.sprint:sprint           0.07471519  3.2692320
## trait:id_obs!trait.endurance:bite          0.02874448  1.0101222
## trait:id_obs!trait.endurance:sprint        0.06790594 -0.8028361
## trait:id_obs!trait.endurance:endurance     0.13052019  7.7136543
## R!variance                                         NA         NA
## R!trait.bite                               0.00577664 15.5099902
## R!trait.sprint                             0.03967145 12.7294445
## R!trait.endurance                                  NA         NA
##                                            constraint
## trait:id!trait.sprint:!trait.bite.cor      Unconstrained
## trait:id!trait.endurance:!trait.bite.cor           Fixed
## trait:id!trait.endurance:!trait.sprint.cor         Fixed
## trait:id!trait.bite                             Positive
## trait:id!trait.sprint                           Positive
```

```
## trait:id!trait.endurance                            Fixed
## trait:id_obs!trait.bite:bite                     Positive
## trait:id_obs!trait.sprint:bite                   Positive
## trait:id_obs!trait.sprint:sprint                 Positive
## trait:id_obs!trait.endurance:bite                Positive
## trait:id_obs!trait.endurance:sprint              Positive
## trait:id_obs!trait.endurance:endurance           Positive
## R!variance                                          Fixed
## R!trait.bite                                     Positive
## R!trait.sprint                                   Positive
## R!trait.endurance                                   Fixed
```

```
nadiv:::pin(asr_mv_bse_m_1, r ~ V8/(sqrt(V7)*sqrt(V9)))
```

```
##      Estimate        SE
## r -0.09036137 0.1993679
```

```
nadiv:::pin(asr_mv_bse_m_1, r ~ V10/(sqrt(V7)*sqrt(V12)))
```

```
##    Estimate        SE
## r 0.1419024 0.1381426
```

```
nadiv:::pin(asr_mv_bse_m_1, r ~ V11/(sqrt(V12)*sqrt(V9)))
```

```
##     Estimate        SE
## r -0.1099356 0.1341958
```

Bite and sprint only for testing hypotheses / looking at covariances:

```
asr_mv_bs_m_1 <- asreml(cbind(bite, sprint) ~ trait +
                          at(trait,1):(scale(svl) * assay_num) +
                          at(trait,2):(scale(svl) + assay_num) +
                          trait:trial_num,
                        random =~ us(trait):id +
                          us(trait):id_obs,
                        rcov =~ units:idh(trait),
                        data = df_an_sc_m)
```

```
summary(asr_mv_bs_m_1)$varcomp
```

```
##                                       gamma    component   std.error
## trait:id!trait.bite:bite          0.071027874  0.071027874 0.019371833
## trait:id!trait.sprint:bite       -0.010815447 -0.010815447 0.027924755
## trait:id!trait.sprint:sprint      0.199051567  0.199051567 0.080924169
## trait:id_obs!trait.bite:bite      0.041625289  0.041625289 0.011069371
## trait:id_obs!trait.sprint:bite   -0.006258249 -0.006258249 0.019854151
## trait:id_obs!trait.sprint:sprint  0.232331062  0.232331062 0.071760253
## R!variance                        1.000000000  1.000000000          NA
## R!trait.bite                      0.089598305  0.089598305 0.005776894
## R!trait.sprint                    0.505168884  0.505168884 0.039690534
##                                     z.ratio constraint
## trait:id!trait.bite:bite          3.6665541   Positive
## trait:id!trait.sprint:bite       -0.3873068   Positive
## trait:id!trait.sprint:sprint      2.4597295   Positive
## trait:id_obs!trait.bite:bite      3.7604024   Positive
## trait:id_obs!trait.sprint:bite   -0.3152111   Positive
## trait:id_obs!trait.sprint:sprint  3.2376009   Positive
## R!variance                             NA      Fixed
```

```
## R!trait.bite                    15.5097722   Positive
## R!trait.sprint                  12.7276919   Positive
```

Bivariate models for hypothesis testing

We create a correlation structure for testing bivariate relationships, where we estimate both variances but set the correlation to 0:

```
init_biv0 <- c(0,
               1,1)
names(init_biv0) <- c("F","P","P")
```

Bite force / sprint

Male

```
asr_mv_bs_m_testcor <- asreml(cbind(bite, sprint) ~ trait +
                            at(trait,1):(scale(svl) * assay_num) +
                            at(trait,2):(scale(svl) + assay_num) +
                            trait:trial_num,
                          random =~ corgh(trait, init=init_biv0):id +
                            us(trait):id_obs,
                          rcov =~ units:idh(trait),
                          data = df_an_sc_m)
```

```
pchisq(2*(asr_mv_bs_m_1$loglik - asr_mv_bs_m_testcor$loglik), 1, lower.tail = FALSE)
```

```
## [1] 0.699224
```

Female

```
asr_mv_bs_f_testcor <- asreml(cbind(bite, sprint) ~ trait +
                            trait:(scale(svl) + assay_num) +
                            at(trait,1:2):(trial_num),
                          random =~ corgh(trait, init=init_biv0):id +
                            us(trait):id_obs,
                          rcov =~ units:idh(trait),
                          data = df_an_sc_f)
```

```
pchisq(2*(asr_mv_bs_f_1$loglik - asr_mv_bs_f_testcor$loglik), 1, lower.tail = FALSE)
```

```
## [1] 0.03822563
```

Sprint / endurance

This residual structure means we estimate only the first variance, the other is set to almost zero, and there is no covariance:

```
init_1re <- c(0.2,1e-8)
names(init_1re) <- c("P","F")
```

```
asr_mv_se_f_1 <- asreml(cbind(sprint, endurance) ~ trait +
                            trait:(scale(svl) + assay_num) +
                            at(trait,1):(trial_num),
                          random =~ idh(trait, init=init_1re):id +
                            us(trait):id_obs,
```

```
                      rcov =~ units:idh(trait, init=init_1re),
                      data = df_an_sc_f,
                      maxiter = 200)

asr_mv_se_f_testidobs <- asreml(cbind(sprint, endurance) ~ trait +
                           trait:(scale(svl) + assay_num) +
                           at(trait,1):(trial_num),
                      random =~ idh(trait, init=init_1re):id +
                        idh(trait):id_obs,
                      rcov =~ units:idh(trait, init=init_1re),
                      data = df_an_sc_f,
                      maxiter = 200)
```

```
pchisq(2*(asr_mv_se_f_1$loglik - asr_mv_se_f_testidobs$loglik), 1, lower.tail = FALSE)
```

```
## [1] 0.2030372
```

```
asr_mv_se_m_1 <- asreml(cbind(sprint, endurance) ~ trait +
                           trait:(scale(svl) * assay_num) +
                           at(trait,1):(trial_num),
                      random =~ idh(trait, init=init_1re):id +
                        us(trait):id_obs,
                      rcov =~ units:idh(trait, init=init_1re),
                      data = df_an_sc_m,
                      maxiter = 200)

asr_mv_se_m_testidobs <- asreml(cbind(sprint, endurance) ~ trait +
                           trait:(scale(svl) * assay_num) +
                           at(trait,1):(trial_num),
                      random =~ idh(trait, init=init_1re):id +
                        idh(trait):id_obs,
                      rcov =~ units:idh(trait, init=init_1re),
                      data = df_an_sc_m,
                      maxiter = 200)
```

```
pchisq(2*(asr_mv_se_m_1$loglik - asr_mv_se_m_testidobs$loglik), 1, lower.tail = FALSE)
```

```
## [1] 0.3430199
```

Bite force / endurance

```
asr_mv_be_f_1 <- asreml(cbind(bite, endurance) ~ trait +
                           at(trait,1):(scale(svl) * assay_num + trial_num) +
                           at(trait,2):(scale(svl) + assay_num),
                      random =~ idh(trait, init=init_1re):id +
                        us(trait):id_obs,
                      rcov =~ units:idh(trait, init=init_1re),
                      data = df_an_sc_f,
                      maxiter = 200)

asr_mv_be_f_testidobs <- asreml(cbind(bite, endurance) ~ trait +
                           at(trait,1):(scale(svl) * assay_num + trial_num) +
                           at(trait,2):(scale(svl) + assay_num),
                      random =~ idh(trait, init=init_1re):id +
                        idh(trait):id_obs,
```

```
                      rcov =~ units:idh(trait, init=init_1re),
                      data = df_an_sc_f,
                      maxiter = 200)
```

```
pchisq(2*(asr_mv_be_f_1$loglik - asr_mv_be_f_testidobs$loglik), 1, lower.tail = FALSE)
```

```
## [1] 0.06140437
```

Plots

Bivariate model plots

Female: bite v sprint

```
df_bs_f_coefs <- data_frame(Trait =attr(asr_mv_bs_f_1$coefficients$random, "names"),
                            Value = asr_mv_bs_f_1$coefficients$random) %>%
  separate(Trait,c("Trait","ID"), sep = ":") %>%
  filter(str_sub(ID,4,6) != "obs") %>%
  spread(Trait, Value)

df_bs_f_SE <- data_frame(Trait =attr(summary(asr_mv_bs_f_1,all=TRUE)$coef.random, "dimnames")[[1]],
                         SE =summary(asr_mv_bs_f_1,all=TRUE)$coef.random[,"std error"]) %>%
  separate(Trait, c("Trait","ID"), sep = ":") %>%
  filter(str_sub(ID,4,6) != "obs") %>%
  spread(Trait, SE) %>%
  rename(SE_bite = trait_bite,
         SE_sprint = trait_sprint)

df_bs_f_coefs <- left_join(df_bs_f_coefs,
                           df_bs_f_SE)

bs_f_slope <- as.numeric(nadiv:::pin(asr_mv_bs_f_1,
                                     slope ~ (V2/V1))$Estimate)
```

```
gg_bs_f <- ggplot(df_bs_f_coefs, aes(x = trait_bite, y = trait_sprint, group = ID)) +
  geom_errorbarh(aes(xmin = trait_bite - SE_bite,
                     xmax = trait_bite + SE_bite),
                 alpha = 0.1) +
  geom_pointrange(aes(ymin = trait_sprint - SE_sprint,
                      ymax = trait_sprint + SE_sprint),
                  fatten = 1,
                  alpha = 0.1) +
  geom_point(colour = "grey30", alpha = 0.8) +
  geom_abline(intercept = 0, slope = bs_f_slope) +
  #xlim(-1.55,1.25) +
  labs(x = "Bite force (BLUP +/- SE)",
       y = "") +
  theme_classic()
```

Male: bite v sprint

```
df_bs_m_coefs <- data_frame(Trait =attr(asr_mv_bs_m_1$coefficients$random, "names"),
                            Value = asr_mv_bs_m_1$coefficients$random) %>%
  separate(Trait,c("Trait","ID"), sep = ":") %>%
  filter(str_sub(ID,4,6) != "obs") %>%
  spread(Trait, Value)

df_bs_m_SE <- data_frame(Trait =attr(summary(asr_mv_bs_m_1,all=TRUE)$coef.random, "dimnames")[[1]],
                         SE =summary(asr_mv_bs_m_1,all=TRUE)$coef.random[,"std error"]) %>%
  separate(Trait, c("Trait","ID"), sep = ":") %>%
  filter(str_sub(ID,4,6) != "obs") %>%
  spread(Trait, SE) %>%
  rename(SE_bite = trait_bite,
         SE_sprint = trait_sprint)

df_bs_m_coefs <- left_join(df_bs_m_coefs,
                           df_bs_m_SE)
```

```
## Joining, by = "ID"
```

```
bs_m_slope <- as.numeric(nadiv:::pin(asr_mv_bs_m_1,
                                     slope ~ (V2/V1))$Estimate)
```

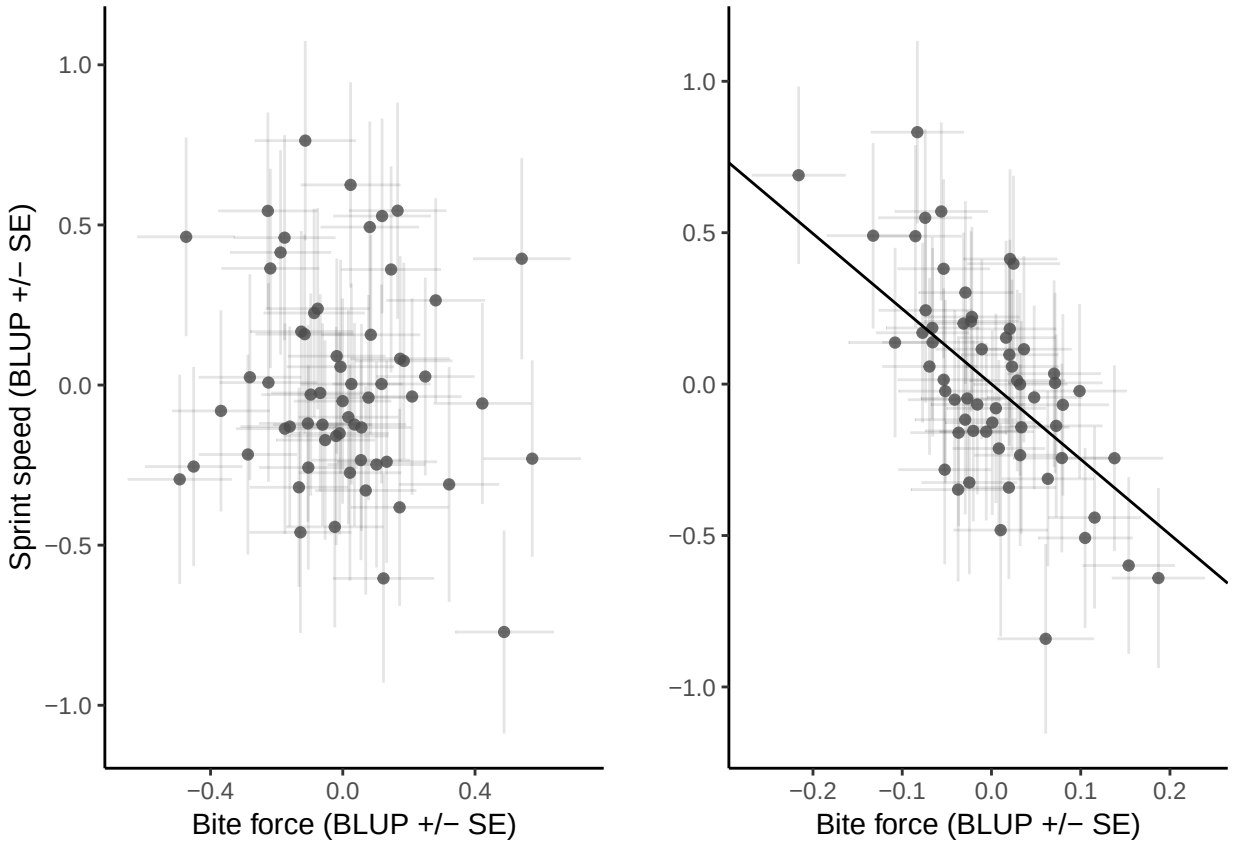
```
gg_bs_m <- ggplot(df_bs_m_coefs, aes(x = trait_bite, y = trait_sprint, group = ID)) +
  geom_errorbarh(aes(xmin = trait_bite - SE_bite,
                     xmax = trait_bite + SE_bite),
                 alpha = 0.1) +
  geom_pointrange(aes(ymin = trait_sprint - SE_sprint,
                      ymax = trait_sprint + SE_sprint),
                  fatten = 1,
                  alpha = 0.1) +
  geom_point(colour = "grey30", alpha = 0.8) +
  #geom_abline(intercept = 0, slope = bs_m_slope) +
  #xlim(-1.55,1.25) +
  labs(x = "Bite force (BLUP +/- SE)",
       y = "Sprint speed (BLUP +/- SE)") +
  theme_classic()
```

```
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
gg_grid <- grid.arrange(gg_bs_m,
                        gg_bs_f,
                        ncol = 2)
```




```
gg_grid
```

```
## TableGrob (1 x 2) "arrange": 2 grobs
##   z     cells    name           grob
## 1 1 (1-1,1-1) arrange gtable[layout]
## 2 2 (1-1,2-2) arrange gtable[layout]
```

Save individual-level plots

```
library(gridExtra)

gg_grid <- grid.arrange(gg_bs_m,
                        gg_bs_f,
                        ncol = 2)


gg_grid

ggsave("BLUP_panels.png", gg_grid, height = 4, width = 9)


pdf(file = "BLUP_panels.pdf", width = 13, height = 6)
grid.arrange(gg_bs_m,
             gg_bs_f,
             ncol = 2)
dev.off()
```