Lathyrus ms2: selection on reaction norms for flowering time

Models with all data performed with MCMCglmm and brms

Alicia Valdés

Contents

Univariate models **3**

MCMCglmm . . . 3

FFD with random effects of year and individual-intercept . . . 3

Random regression for FFD, including random effects of individual slopes and covariance between intercept and slope . . . 4

Extract BLUPs from this model . . . 5

brms . . . 7

FFD with random effects of year and individual-intercept . . . 7

Random regression for FFD, including random effects of individual slopes and covariance between intercept and slope . . . 8

Extract BLUPs from this model . . . 9

Compare results of MCMCglmm and brms . . . 10

FFD with random effects of year and individual-intercept . . . 11

Random regression for FFD, including random effects of individual slopes and covariance between intercept and slope . . . 11

Bivariate models **12**

MCMCglmm . . . 12

Mean fitness per flowering event . . . 15

With no condition variable . . . 15

With shoot volume . . . 21

Mean fitness per year of study . . . 27

With no condition variable . . . 27

With shoot volume . . . 32

brms . . . 38

Mean fitness per flowering event . . . 38

With no condition variable . . . 38

With shoot volume . . . 42

Mean fitness per year of study 44
With no condition variable 44
With shoot volume 45
Compare results of MCMCglmm and brms 47
Mean fitness per flowering event, no condition variable 47

Read data

```
datadef<-read.csv(
  "data/datadef.csv")
head(datadef)
```

```
##   year id_nr    id fcode      FFD n_fl n_fr totseed intactseed shoot_vol period
## 1 1989     1 old_1     1       NA    6    3       8          6 1418.6000    old
## 2 1990     1 old_1     0       NA    0    0       0          0  523.2000    old
## 3 1991     1 old_1     1 59.91181   23    3      12         12 1915.4000    old
## 4 1992     1 old_1     1 55.66944   19    2       6          1 1460.1917    old
## 5 1993     1 old_1     1       NA   NA    0       0          0  879.6493    old
## 6 1994     1 old_1     1 59.18403   14    1       3          3 1338.6727    old
##   n_years_fl_fitness n_years_study   mean_4       cmean_4
## 1                  5             8 5.236667 -0.228207783
## 2                  5             8 7.195000  1.730125551
## 3                  5             8 5.245000 -0.219874449
## 4                  5             8 3.828333 -1.636541116
## 5                  5             8 5.461667 -0.003207783
## 6                  5             8 6.418333  0.953458884
```

Number of individuals in each period:

```
length(with(subset(datadef,period=="old"),unique(id)))
```

```
## [1] 607
```

```
length(with(subset(datadef,period=="new"),unique(id)))
```

```
## [1] 230
```

Number of observations in each period:

```
nrow(subset(datadef,period=="old"))
```

```
## [1] 4606
```

```
nrow(subset(datadef,period=="new"))
```

```
## [1] 2231
```

Number of cases with FFD in each period:

```
nrow(subset(datadef,period=="old"&!is.na(FFD)))
```

```
## [1] 1467
```

```
nrow(subset(datadef,period=="new"&!is.na(FFD)))
```

```
## [1] 1011
```

Univariate models

MCMCglmm

Code based on Arnold et al. 2019 Phil. Trans. R. Soc. B.

```
# Scaling factor for MCMCglmm iterations
sc <- 1000 # Increase this parameter for longer runs

priorUV2 <- list(G = list(G1 = list(V = diag(1), nu = 1),
                          # for random effect of year
                          G2 = list(V = diag(1), nu = 1)),
                 # for random effect of id
                 R = list(R1 = list(V = diag(1), nu = 2)))
priorUV2_RR <- list(G = list(G1 = list(V = diag(1), nu = 1),
                             # other random effect (YEAR)
                             G2 = list(V = diag(2), nu = 1)),
                    # ^ 2x2 var-covar matrix for var in slopes + intercepts
                    R = list(R1 = list(V = diag(1), nu = 2)))
```

FFD with random effects of year and individual-intercept

```
univar.FFD.all <- MCMCglmm(FFD ~ cmean_4,
                     random = ~year + id,
                     rcov = ~units,
                     data = datadef,prior = priorUV2,
                     family = "gaussian",nitt = 2100 * sc,
                     thin = sc, burnin = 100 * sc, verbose = F)
# nitt = burnin + thin*(n samples to keep)
# Aim to store 2000 iterations
save(univar.FFD.all,
     file="output/univar.FFD.all.RData")
```

```
summary(univar.FFD.all)
```

```
##
##  Iterations = 100001:2099001
##  Thinning interval  = 1000
##  Sample size  = 2000
```

```
##
##  DIC: 14656.08
##
##  G-structure:  ~year
##
##      post.mean l-95% CI u-95% CI eff.samp
## year     25.51    11.85    42.32     2000
##
##                ~id
##
##    post.mean l-95% CI u-95% CI eff.samp
## id     2.531    1.657    3.407     2000
##
##  R-structure:  ~units
##
##       post.mean l-95% CI u-95% CI eff.samp
## units     19.79    18.57    21.03     1988
##
##  Location effects: FFD ~ cmean_4
##
##             post.mean l-95% CI u-95% CI eff.samp  pMCMC
## (Intercept)   58.5830  56.4189  60.5194     2000 <5e-04 ***
## cmean_4       -2.4144  -4.0448  -0.8616     2183  0.009 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Random regression for FFD, including random effects of individual slopes and covariance between intercept and slope

```
univar.FFD_RR.all <- MCMCglmm(FFD ~ cmean_4,
                        random = ~year + us(1 + cmean_4):id,
                        rcov = ~units,
                        data = datadef,prior = priorUV2_RR,
                        family = "gaussian",nitt = 2100*sc,
                        thin = sc, burnin = 100*sc, verbose = F,pr=T)
# pr= T saves the posterior distribution of the individual random effects
# (analagous to the BLUP from the LMM)
save(univar.FFD_RR.all,
     file="output/univar.FFD_RR.all.RData")
```

```
summary(univar.FFD_RR.all)
```

```
##
##  Iterations = 100001:2099001
##  Thinning interval  = 1000
##  Sample size  = 2000
##
##  DIC: 14594.27
##
##  G-structure:  ~year
##
```

```
##      post.mean l-95% CI u-95% CI eff.samp
## year     25.79    12.17    44.08     1689
##
##                ~us(1 + cmean_4):id
##
##                             post.mean l-95% CI u-95% CI eff.samp
## (Intercept):(Intercept).id    2.4433   1.5595    3.393     2318
## cmean_4:(Intercept).id        0.9493   0.5518    1.382     2000
## (Intercept):cmean_4.id        0.9493   0.5518    1.382     2000
## cmean_4:cmean_4.id            0.7215   0.3581    1.109     2000
##
##  R-structure:  ~units
##
##       post.mean l-95% CI u-95% CI eff.samp
## units     18.95    17.69    20.27     2000
##
##  Location effects: FFD ~ cmean_4
##
##             post.mean l-95% CI u-95% CI eff.samp  pMCMC
## (Intercept)   58.5530  56.4273  60.8180     2000 <5e-04 ***
## cmean_4       -2.3508  -3.8980  -0.7076     2000  0.002 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Extract BLUPs from this model Code adapted from Houslay & Wilson 2017 Behav. Ecol. Code for graphs based on Arnold et al. 2019 Phil. Trans. R. Soc. B.

```
BLUPs_MCMC.all <- tibble(Trait = attr(colMeans(univar.FFD_RR.all$Sol), "names"),
                         Value = colMeans(univar.FFD_RR.all$Sol)) %>%
  filter(grepl("id", Trait))%>% # Select only id intercepts and slopes
  mutate(type=ifelse(grepl("Intercept",Trait),"intercept","slope"))%>%
  mutate(id=sub(".*id.", "", Trait))%>%
  select(-Trait)%>%
  spread(., type, Value) # Convert from long to wide
with(BLUPs_MCMC.all,cor(intercept,slope)) # highly correlated!
```

```
## [1] 0.928645
```

Correlation among intercepts and slopes

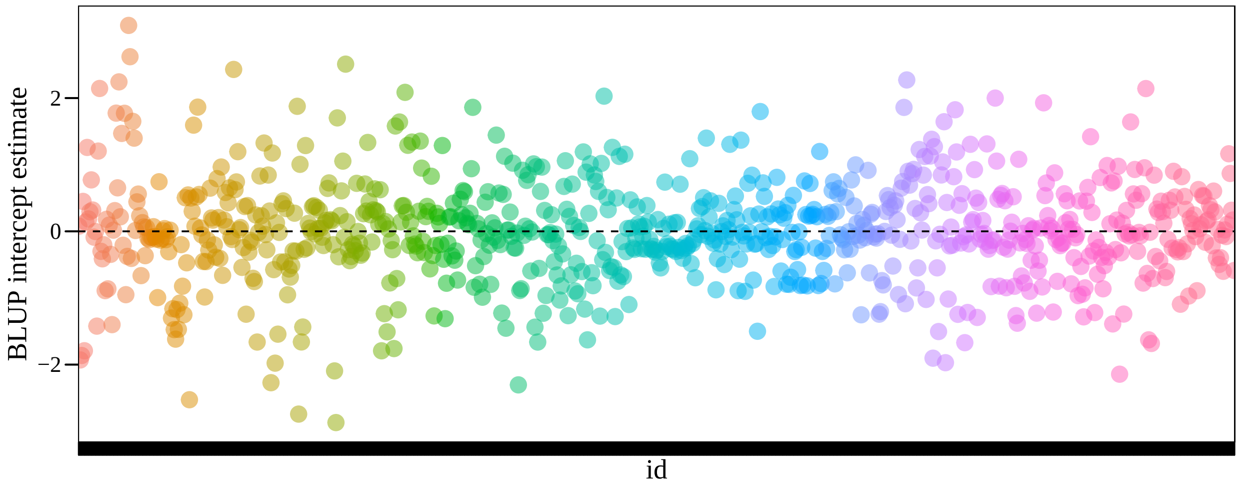
```
univar.FFD_RR.all_intslope <-
  univar.FFD_RR.all$VCV[,"cmean_4:(Intercept).id"]/
(sqrt(univar.FFD_RR.all$VCV[,"(Intercept):(Intercept).id"])*
sqrt(univar.FFD_RR.all$VCV[,"cmean_4:cmean_4.id"]))
posterior.mode(univar.FFD_RR.all_intslope)
```

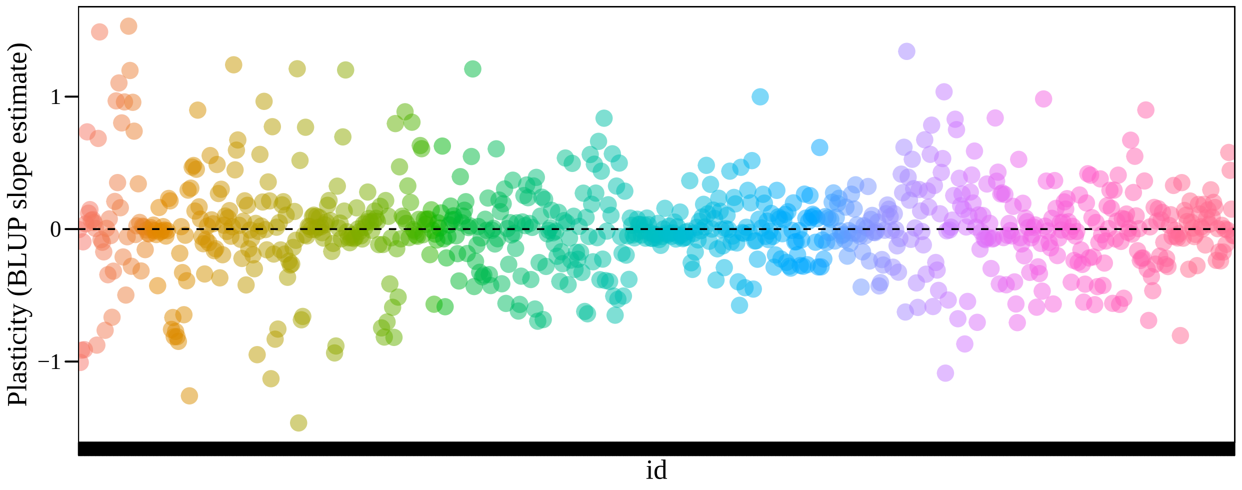
```
##      var1
## 0.7875152
```

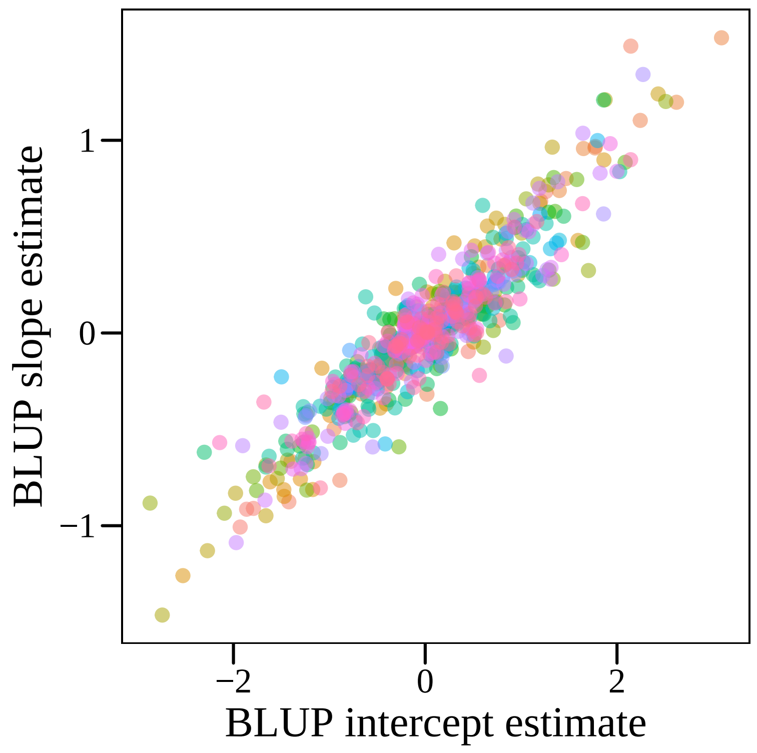
```
HPDinterval(univar.FFD_RR.all_intslope)
```

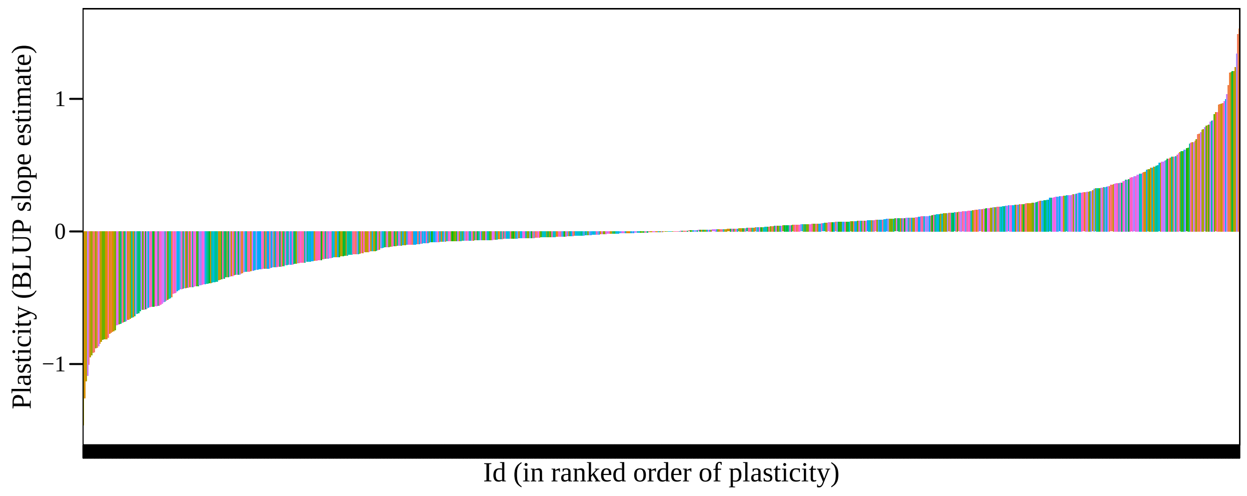
```
##           lower     upper
## var1 0.4991695 0.9065499
## attr(,"Probability")
## [1] 0.95
```

Plots with BLUPs









brms

FFD with random effects of year and individual-intercept

```r
my.cores <- detectCores()
```

```r
univar.FFD.all.brm<-brm(formula=FFD~cmean_4+(1|year)+(1|id),data=datadef,
                 warmup = 1000,iter = 4000,thin=2,chains=4,
                 inits = "random",seed = 12345,cores = my.cores)
# Total of 6000 post-warmup samples
```

```r
save(univar.FFD.all.brm,
     file="output/univar.FFD.all.brm.RData")
```

```r
summary(univar.FFD.all.brm)
```

```
##  Family: gaussian
##   Links: mu = identity; sigma = identity
## Formula: FFD ~ cmean_4 + (1 | year) + (1 | id)
##    Data: datadef (Number of observations: 2478)
## Samples: 4 chains, each with iter = 4000; warmup = 1000; thin = 2;
##          total post-warmup samples = 6000
##
## Group-Level Effects:
## ~id (Number of levels: 837)
##               Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sd(Intercept)     1.60      0.14     1.32     1.88 1.00     3477     4522
##
## ~year (Number of levels: 22)
##               Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sd(Intercept)     5.13      0.85     3.76     7.01 1.00     2252     3898
##
## Population-Level Effects:
##           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept    58.56      1.11    56.44    60.77 1.00     1559     2547
## cmean_4      -2.43      0.81    -4.00    -0.86 1.00     1964     3541
##
## Family Specific Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sigma     4.45      0.07     4.31     4.59 1.00     4320     5245
##
## Samples were drawn using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

Random regression for FFD, including random effects of individual slopes and covariance between intercept and slope

```r
univar.FFD_RR.all.brm<-brm(formula=FFD~cmean_4+(1|year)+(cmean_4|id),data=datadef,
                warmup = 1000,iter = 4000,thin=2,chains=4,
                inits = "random",seed = 12345,cores = my.cores)
# Total of 6000 post-warmup samples
save(univar.FFD_RR.all.brm,
     file="output/univar.FFD_RR.all.brm.RData")
```

```r
summary(univar.FFD_RR.all.brm)
```

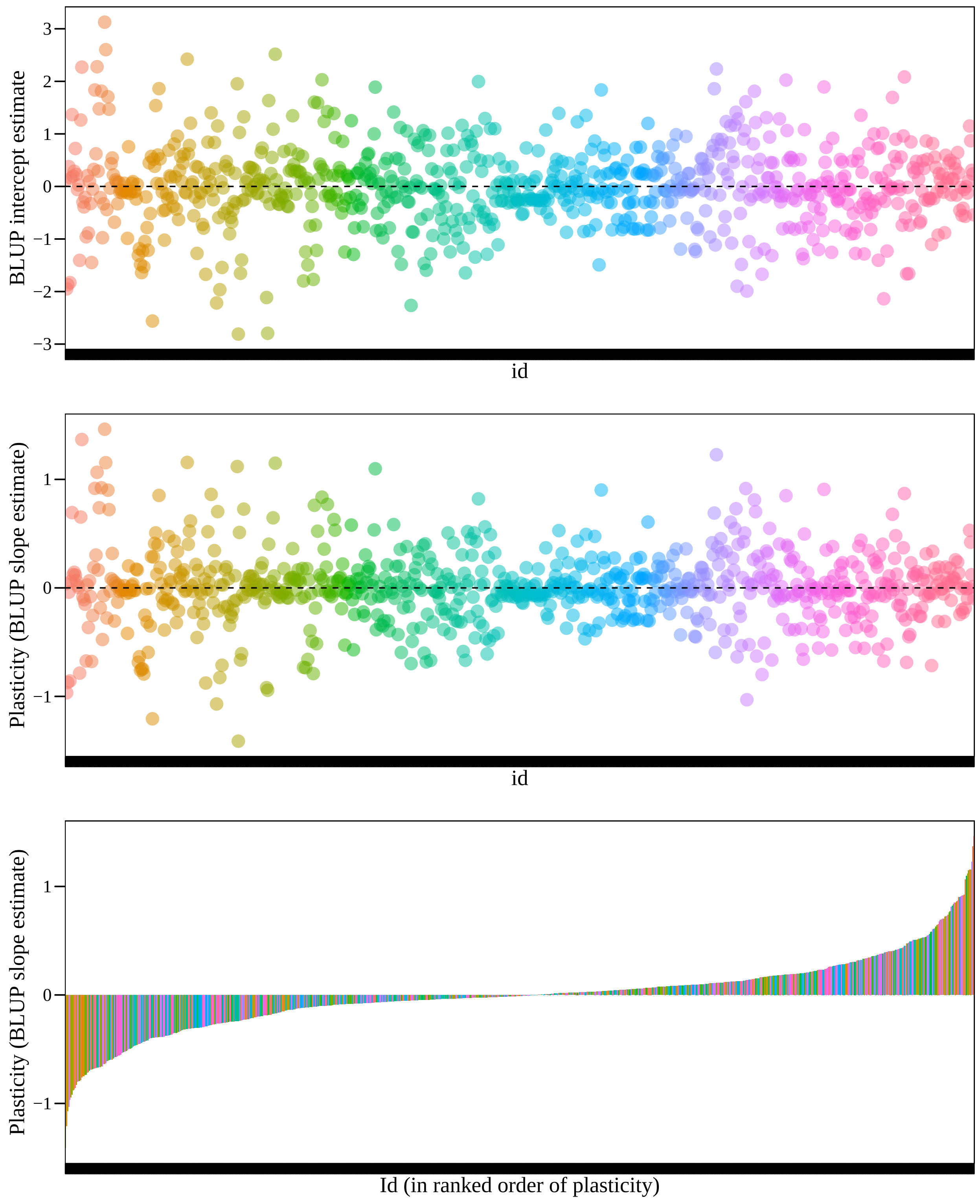
```
##  Family: gaussian
##   Links: mu = identity; sigma = identity
## Formula: FFD ~ cmean_4 + (1 | year) + (cmean_4 | id)
##    Data: datadef (Number of observations: 2478)
## Samples: 4 chains, each with iter = 4000; warmup = 1000; thin = 2;
##          total post-warmup samples = 6000
##
## Group-Level Effects:
## ~id (Number of levels: 837)
##                        Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS
## sd(Intercept)              1.55      0.15     1.25     1.85 1.00     3313
## sd(cmean_4)                0.78      0.13     0.53     1.04 1.00     2604
## cor(Intercept,cmean_4)     0.80      0.13     0.50     0.99 1.00     1696
##                        Tail_ESS
## sd(Intercept)              4966
## sd(cmean_4)                4060
## cor(Intercept,cmean_4)     3395
##
## ~year (Number of levels: 22)
##               Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sd(Intercept)     5.14      0.84     3.81     7.07 1.00     2755     3878
##
## Population-Level Effects:
##           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept    58.62      1.12    56.45    60.90 1.00     1339     2675
## cmean_4      -2.38      0.83    -4.04    -0.73 1.00     2002     3361
##
## Family Specific Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sigma     4.37      0.07     4.22     4.52 1.00     3324     4496
##
## Samples were drawn using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

Extract BLUPs from this model This code needs to be checked - not sure that this is the correct way to extract BLUPs!

```r
BLUPs_MCMC.all.brms  <- cbind(as.factor(c(1:837)),
                              as.data.frame(ranef(univar.FFD_RR.all.brm)$id)
                              [c(1,5)])
colnames(BLUPs_MCMC.all.brms) <- c("id", "intercept", "slope")
with(BLUPs_MCMC.all.brms,cor(intercept,slope)) # highly correlated!
```

```
## [1] 0.961559
```

Plots with BLUPs



Compare results of MCMCglmm and brms

Fixed effects can be compared directly between MCMCglmm and brms outputs.

From Pieter's email: The group-level (random) effects reported by brms are standard deviations and correlations rather than the variances and covariances that mcmcglmm outputs. We need to square all the mcmc samples from the posterior then take their mean to get true estimates of the variance from brms (then we can compare them).

The code for these comparisons needs to be checked!

FFD with random effects of year and individual-intercept

```
kable(rbind(MCMCglmm=summary(univar.FFD.all)$solutions[,1],
      brms=summary(univar.FFD.all.brm)$fixed[,1])) # Comparison fixed effects
```

	(Intercept)	cmean_4
MCMCglmm	58.58298	-2.414362
brms	58.56082	-2.430598

```
univar.FFD.all.brm_asmcmc <- as.mcmc(univar.FFD.all.brm, combine_chains = TRUE)
#head(univar.FFD.all.brm_asmcmc) # check which column the parameters are in

univar.FFD.all.brm_year <- (univar.FFD.all.brm_asmcmc[,4]^2)
# sd_year__Intercept^2
univar.FFD.all.brm_id_intercept <- (univar.FFD.all.brm_asmcmc[,3]^2)
# sd_id__Intercept^2 (individual intercept)
univar.FFD.all.brm_resid <- (univar.FFD.all.brm_asmcmc[,5]^2)
# sigma^2 (residual)

kable(rbind(MCMCglmm=summary(univar.FFD.all$VCV)$statistics[,1],
      brms=as.vector(cbind(mean(univar.FFD.all.brm_year),
                          mean(univar.FFD.all.brm_id_intercept),
            mean(univar.FFD.all.brm_resid))))) # Comparison random effects
```

	year	id	units
MCMCglmm	25.50599	2.531302	19.78506
brms	27.08841	2.582339	19.79277

```
# Calculate 95% CI interval of the converted values
# round(HPDinterval(univar.FFD.all.brm_id_intercept), 2)
# round(HPDinterval(univar.FFD.all.brm_year), 2)
# round(HPDinterval(univar.FFD.all.brm_resid), 2)
```

Quite similar values.

Random regression for FFD, including random effects of individual slopes and covariance between intercept and slope

```
kable(rbind(MCMCglmm=summary(univar.FFD_RR.all)$solutions[,1],
      brms=summary(univar.FFD_RR.all.brm)$fixed[,1])) # Comparison fixed effects
```

	(Intercept)	cmean_4
MCMCglmm	58.55304	-2.350803
brms	58.62167	-2.376662

```
univar.FFD_RR.all.brm_asmcmc <- as.mcmc(univar.FFD_RR.all.brm,
                                        combine_chains = TRUE)
#head(univar.FFD_RR.all.brm_asmcmc) # check which column the parameters are in

univar.FFD_RR.all.brm_year <- (univar.FFD_RR.all.brm_asmcmc[,5]^2)
# sd_year__Intercept^2
univar.FFD_RR.all.brm_id_intercept <- (univar.FFD_RR.all.brm_asmcmc[,3]^2)
# sd_id__Intercept^2 (individual intercept)
univar.FFD_RR.all.brm_year_cor_id_intercept_cmean_4<-(univar.FFD_RR.all.brm_asmcmc[,6]^2)
#cor_id__Intercept__cmean_4^2 (corr intercept-slope)
univar.FFD_RR.all.brm_year_sd_id_cmean_4<-(univar.FFD_RR.all.brm_asmcmc[,4]^2)
#sd_id__cmean_4^2 (individual slope)
univar.FFD_RR.all.brm_resid <- (univar.FFD_RR.all.brm_asmcmc[,7]^2)
# sigma^2 (residual)

kable(cbind(MCMCglmm=summary(univar.FFD_RR.all$VCV)$statistics[,1],
      brms=as.vector(cbind(mean(univar.FFD_RR.all.brm_year),
                           mean(univar.FFD_RR.all.brm_id_intercept),
                           mean(univar.FFD_RR.all.brm_year_cor_id_intercept_cmean_4),
                           mean(univar.FFD_RR.all.brm_year_cor_id_intercept_cmean_4),
                           mean(univar.FFD_RR.all.brm_year_sd_id_cmean_4),
                           mean(univar.FFD_RR.all.brm_resid)))))
```

	MCMCglmm	brms
year	25.7948433	27.1587835
(Intercept):(Intercept).id	2.4432653	2.4336152
cmean_4:(Intercept).id	0.9492754	0.6554499
(Intercept):cmean_4.id	0.9492754	0.6554499
cmean_4:cmean_4.id	0.7214901	0.6287895
units	18.9507223	19.0848598

```
# Comparison random effects
```

Bivariate models

MCMCglmm

Code based on Arnold et al. 2019 Phil. Trans. R. Soc. B.

Fitting bivariate models of fitness and FFD, with random regressions for individuals, using a Poisson distribution for fitness (and using absolute fitness). Using mean April temperature. Using either mean fitness per year of study (dividing sum of fitness by the number of years that each plant was included in the study) or mean fitness per flowering event (dividing sum of fitness by the number of years that each plant flowered and which had fitness information available). Including / not including mean shoot volume over all years with available data (with an effect on fitness) as a condition variable.

Data preparation

```
datadef_total<-datadef %>%
  group_by(id)%>%
  summarise(mean_fitness_study=sum(intactseed,na.rm=T)/mean(n_years_study),
            mean_fitness_fl=sum(intactseed,na.rm=T)/mean(n_years_fl_fitness))%>%
   arrange(.,id) # Order by id

with(datadef_total,cor(mean_fitness_study,mean_fitness_fl))  # Highly corr (0.87)
```

```
## [1] 0.8660685
```

```
# Calculate mean shoot volume for each id using values of shoot volume for all ids/years
# (including flowering and non-flowering years)

shoot_vol_all_means<-datadef[c(1,3,10)]%>%
  group_by(id)%>%
  summarise(shoot_vol_mean=mean(shoot_vol,na.rm=T)) # Mean of all available values

# Join shoot volume data
datadef_total<-datadef_total%>%left_join(shoot_vol_all_means)%>%
  left_join(unique(datadef[c(2,3,11)]))
head(datadef_total)
```

```
## # A tibble: 6 x 6
##   id      mean_fitness_study mean_fitness_fl shoot_vol_mean id_nr period
##   <chr>                <dbl>           <dbl>          <dbl> <int> <chr>
## 1 new_1                 0               0             1830.     1 new
## 2 new_10               14.3            15.7           9794.    10 new
## 3 new_100               3.89            5.83          1959.   100 new
## 4 new_101               2.25            3.00          1195.   101 new
## 5 new_102               5.61            6.73          3269.   102 new
## 6 new_103               3.60            4.32          1694.   103 new
```

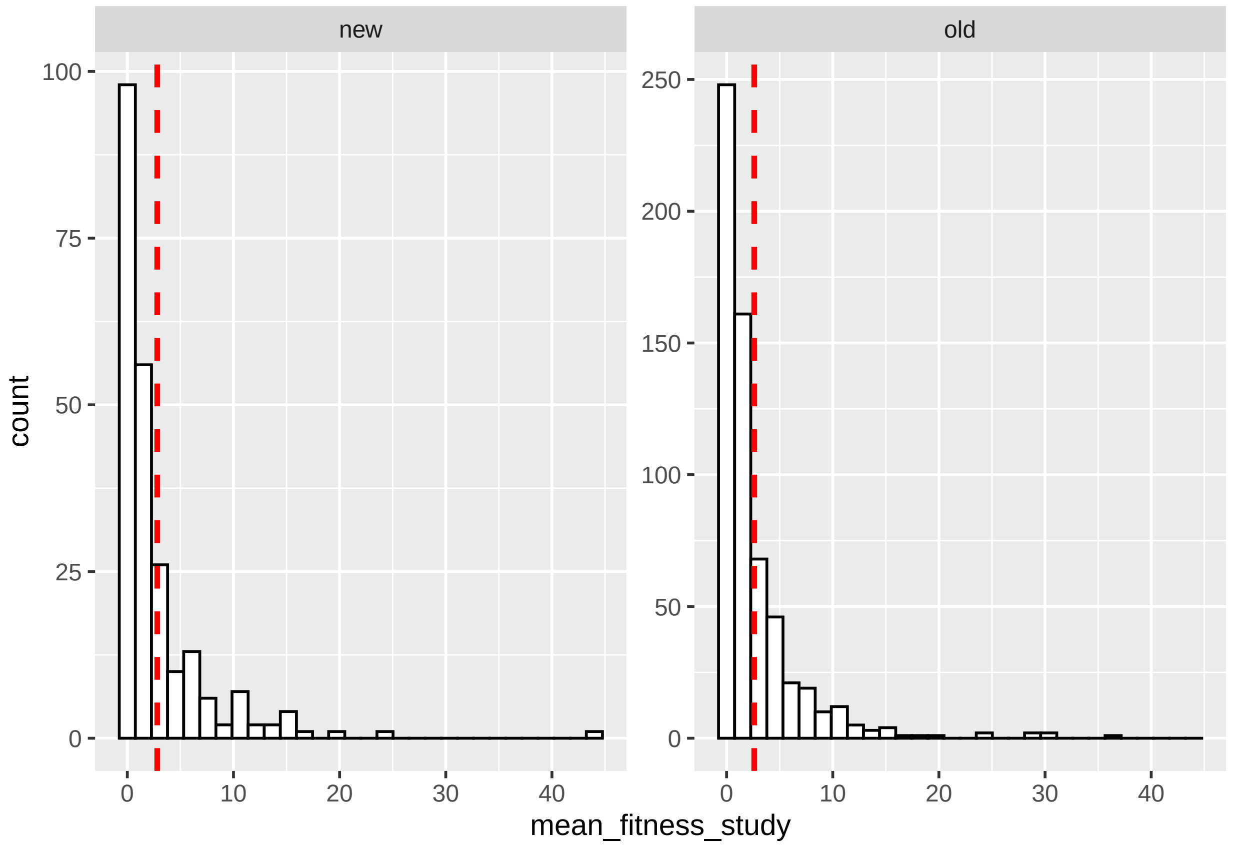
```
nrow(subset(datadef_total,is.na(shoot_vol_mean)))
```

```
## [1] 46
```

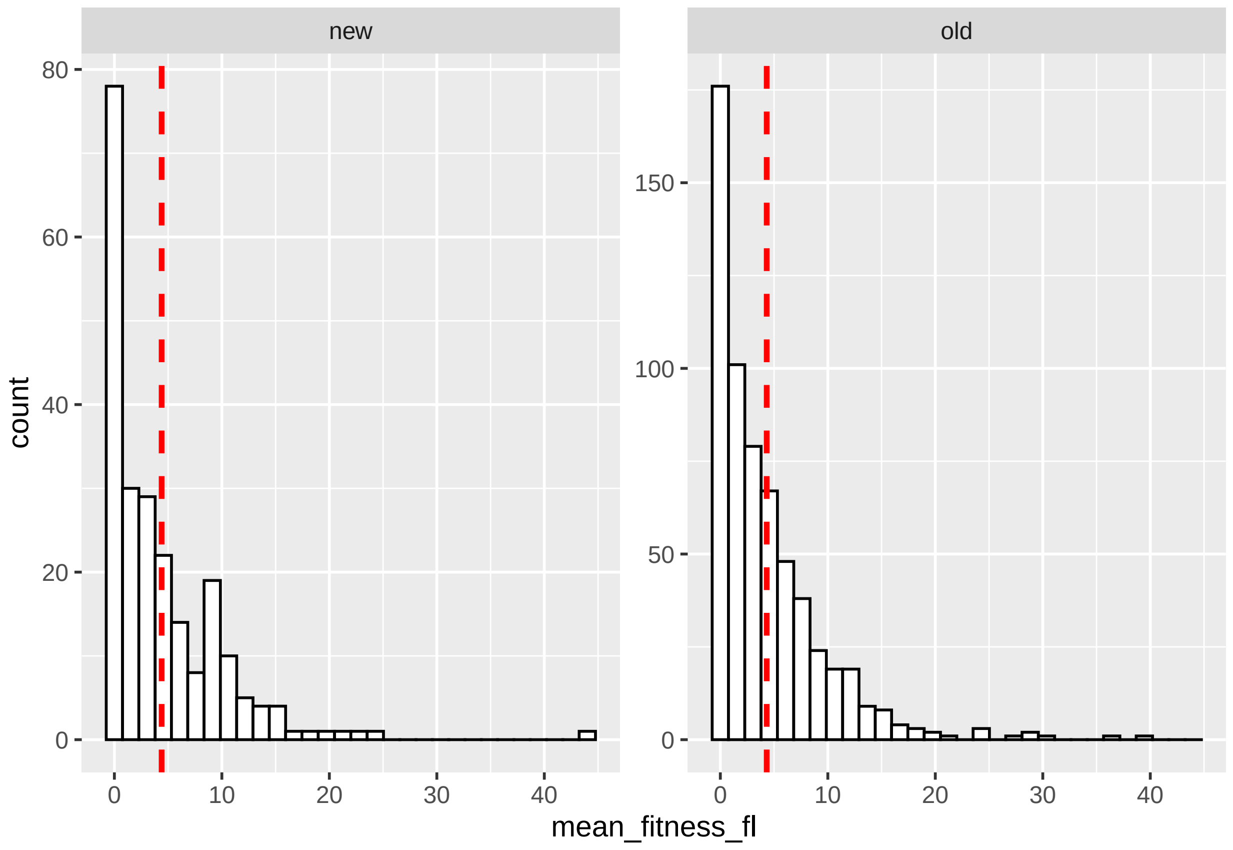
```
# 46 ids with no info on shoot volume
```

Compare distributions of mean fitness per year of study and mean fitness per flowering event between old and new periods:

```r
ggplot(datadef_total,aes(x=mean_fitness_study))+
  geom_histogram(colour="black",fill="white",position="dodge")+
  facet_wrap(~period,scales="free_y")+
  geom_vline(data=plyr::ddply(datadef_total,"period",summarise,
                              mean_fitness_study.mean=mean(mean_fitness_study)),
             aes(xintercept=mean_fitness_study.mean),
             linetype="dashed", size=1, colour="red")
```



```r
ggplot(datadef_total,aes(x=mean_fitness_fl))+
  geom_histogram(colour="black",fill="white",position="dodge")+
  facet_wrap(~period,scales="free_y")+
  geom_vline(data=plyr::ddply(datadef_total,"period",summarise,
                              mean_fitness_fl.mean=mean(mean_fitness_fl)),
             aes(xintercept=mean_fitness_fl.mean),
             linetype="dashed", size=1, colour="red")
```



Distributions and means of the two mean fitness measures are similar among the two periods.

Mean fitness per flowering event

With no condition variable Stack data:

```
# Create a single data-set "datadef.stack1", with single column at start
# to index observations
datadef.stack1 <- c()
datadef.stack1$Obs <- 1:(837 + 2478)
datadef.stack1$id <- c(as.character(datadef_total$id),
                       as.character(subset(datadef,!is.na(FFD))$id))
# ids in alphabetical order

# Add first_yr to total data +
# Year column is only relevant for FFD, but is set to first_yr for fitness values
datadef_total$first_yr<-ifelse(grepl("old",as.character(datadef_total$id)),1987,2006)
datadef.stack1$year <- c(datadef_total$first_yr,
                         subset(datadef,!is.na(FFD))$year)

# Temperature column is only relevant for FFD, but is set to 0 for fitness values
datadef.stack1$temp <- c(rep(0, 837), subset(datadef,!is.na(FFD))$cmean_4)

# Create single column with first fitness values (ABSOLUTE VALUES), then FFD values:
datadef.stack1$fitness.FFD.stack <- c(round(datadef_total$mean_fitness_fl),
                                      subset(datadef,!is.na(FFD))$FFD)
```

```r
# Create 3 index columns needed for MCMCglmm
datadef.stack1$traits <- as.factor(c(rep("fitness", 837), rep("FFD", 2478)))
datadef.stack1$variable <- as.factor(datadef.stack1$traits)
# Fitness will be modelled with an overdispersed Poisson
# FFD will be modelled with a Gaussian distribution
# Specify this with the column 'family':
datadef.stack1$family <- c(rep("poisson", 837), rep("gaussian", 2478))
datadef.stack1 <- data.frame(datadef.stack1)

datadef.stack1$id <- as.factor(datadef.stack1$id)
datadef.stack1$year <- as.factor(datadef.stack1$year)
head(datadef.stack1)
```

```
##   Obs      id year temp fitness.FFD.stack  traits variable  family
## 1   1   new_1 2006    0                 0 fitness  fitness poisson
## 2   2  new_10 2006    0                16 fitness  fitness poisson
## 3   3 new_100 2006    0                 6 fitness  fitness poisson
## 4   4 new_101 2006    0                 3 fitness  fitness poisson
## 5   5 new_102 2006    0                 7 fitness  fitness poisson
## 6   6 new_103 2006    0                 4 fitness  fitness poisson
```

```r
# Scaling factor for MCMCglmm iterations
sc <- 1000 # Increase this parameter for longer runs

priorBiv <- list(G = list(G1 = list(V = diag(1), nu = 1)),
                    # ^ random effect for year (fitted for FFD only)
                    R = list(R1 = list(V = diag(3), nu = 3, covu = TRUE),
                             # ^ 3-way var-cov matrix of (id + temp:id) for FFD,
                             # residual for fitness
                             R2 = list(V = diag(1), nu = 1))) # residual for FFD
```

```r
bivar1.all <- MCMCglmm(fitness.FFD.stack ~ variable - 1 +
                         # ^ means for each variable
                         # (and no overall mean (hence "-1"))
                         at.level(variable, "FFD"):temp,
                       # single fixed effect of temp
                       random = ~us(at.level(variable, "FFD")):year +
                         us(at.level(variable, "FFD") +
                              at.level(variable,"FFD"):temp):id,
                       # ^ random intercepts for individual,
                       # and random slopes for temp|id
                       rcov = ~us(at.level(variable, "fitness")):id +
                         # ^ variance between indivduals in fitness
                         # (which is residual variance)
                         us(at.level(variable, "FFD")):Obs,
                         # ^ residual variance within indivduials between years
                       # (labelled by 'Obs')
                       data = datadef.stack1,
                       prior = priorBiv,
                       family = NULL, # specified already in the data-set
                       nitt = 2100 * sc, thin = sc, burnin = 100 * sc,
                       verbose = F,singular.ok = T)
```

```r
# nitt = burnin + thin*(n samples to keep)
# Aim to store 2000 iterations
save(bivar1.all,file="output/bivar1.all.RData")
```

```r
kable(summary(bivar1.all)$solutions,digits=c(3,3,3,0,3),caption="Fixed effects")
```

Table 5: Fixed effects

	post.mean	l-95% CI	u-95% CI	eff.samp	pMCMC
variableFFD	58.808	56.485	60.943	2000	0.000
variablefitness	0.835	0.727	0.943	2000	0.000
at.level(variable, “FFD”):temp	-2.350	-3.863	-0.758	2000	0.006

```r
kable(summary(bivar1.all)$Gcovariances,digits=c(3,3,3,0),caption="Random effects")
```

Table 6: Random effects

	post.mean	l-95% CI	u-95% CI	eff.samp
at.level(variable, “FFD”):at.level(variable, “FFD”).year	25.195	12.16	43.439	2000

```r
kable(summary(bivar1.all)$Rcovariances,digits=c(3,3,3,0),caption="Random effects")
```

Table 7: Random effects

	post.mean	l-95% CI	u-95% CI	eff.samp
at.level(variable, “FFD”).id:at.level(variable, “FFD”).id	2.834	1.947	3.681	2000
at.level(variable, “FFD”):temp.id:at.level(variable, “FFD”).id	0.824	0.415	1.280	1836
at.level(variable, “fitness”).id:at.level(variable, “FFD”).id	-1.220	-1.602	-0.887	2000
at.level(variable, “FFD”).id:at.level(variable, “FFD”):temp.id	0.824	0.415	1.280	1836
at.level(variable, “FFD”):temp.id:at.level(variable, “FFD”):temp.id	0.826	0.418	1.201	2000
at.level(variable, “fitness”).id:at.level(variable, “FFD”):temp.id	-0.084	-0.369	0.215	2278
at.level(variable, “FFD”).id:at.level(variable, “fitness”).id	-1.220	-1.602	-0.887	2000
at.level(variable, “FFD”):temp.id:at.level(variable, “fitness”).id	-0.084	-0.369	0.215	2278
at.level(variable, “fitness”).id:at.level(variable, “fitness”).id	1.508	1.268	1.756	2183
at.level(variable, “FFD”):at.level(variable, “FFD”).Obs	18.625	17.462	19.862	2245

```r
kable(diag(autocorr(bivar1.all$Sol)[2, , ]),caption="Autocorrelation")
```

Table 8: Autocorrelation

	x
variableFFD	0.0006864
variablefitness	-0.0103182
at.level(variable, “FFD”):temp	0.0285160

```
kable(diag(autocorr(bivar1.all$VCV)[2, , ]),caption="Autocorrelation")
```

Table 9: Autocorrelation

	x
at.level(variable, “FFD”):at.level(variable, “FFD”).year	-0.0029005
at.level(variable, “FFD”).id:at.level(variable, “FFD”).id	0.0077954
at.level(variable, “FFD”):temp.id:at.level(variable, “FFD”).id	0.0246757
at.level(variable, “fitness”).id:at.level(variable, “FFD”).id	-0.0257555
at.level(variable, “FFD”).id:at.level(variable, “FFD”):temp.id	0.0246757
at.level(variable, “FFD”):temp.id:at.level(variable, “FFD”):temp.id	0.0094623
at.level(variable, “fitness”).id:at.level(variable, “FFD”):temp.id	-0.0327239
at.level(variable, “FFD”).id:at.level(variable, “fitness”).id	-0.0257555
at.level(variable, “FFD”):temp.id:at.level(variable, “fitness”).id	-0.0327239
at.level(variable, “fitness”).id:at.level(variable, “fitness”).id	-0.0439917
at.level(variable, “FFD”):at.level(variable, “FFD”).Obs	0.0132773

For interpretation of covariances, we convert them to correlations using the formula for a correlation with the posterior distributions of our (co)variance components. This gives us a distribution of correlation values that we can use to calculate estimates and 95% credible intervals (code adapted from Houslay & Wilson 2017 Behav. Ecol.).

Among-individual correlation between intercepts and slopes for FFD, between FFD and fitness and between fitness and variation in slopes for FFD:

```
cor_bivar1.all_intslope <-
  bivar1.all$VCV[,"at.level(variable, \"FFD\"):temp.id:at.level(variable, \"FFD\").id"]/
(sqrt(bivar1.all$VCV[,"at.level(variable, \"FFD\").id:at.level(variable, \"FFD\").id"])*
sqrt(bivar1.all$VCV[,"at.level(variable, \"FFD\"):temp.id:at.level(variable, \"FFD\"):temp.id"]))
posterior.mode(cor_bivar1.all_intslope)
```

```
##      var1
## 0.5416883
```

```
HPDinterval(cor_bivar1.all_intslope)
```

```
##          lower     upper
## var1 0.3184419 0.7679854
## attr(,"Probability")
## [1] 0.95
```

```
cor_bivar1.all_intfit <-
  bivar1.all$VCV[,"at.level(variable, \"fitness\").id:at.level(variable, \"FFD\").id"]/
  (sqrt(bivar1.all$VCV[,"at.level(variable, \"fitness\").id:at.level(variable, \"fitness\").id"])*
     sqrt(bivar1.all$VCV[,"at.level(variable, \"FFD\").id:at.level(variable, \"FFD\").id"]))
posterior.mode(cor_bivar1.all_intfit)
```

```
##       var1
## -0.5775376
```

```
HPDinterval(cor_bivar1.all_intfit)
```

```
##           lower      upper
## var1 -0.7293062 -0.4519528
## attr(,"Probability")
## [1] 0.95
```

```
cor_bivar1.all_slopefit <-
  bivar1.all$VCV[,"at.level(variable, \"fitness\").id:at.level(variable, \"FFD\"):temp.id"]/
  (sqrt(bivar1.all$VCV[,"at.level(variable, \"fitness\").id:at.level(variable, \"fitness\").id"])*
     sqrt(bivar1.all$VCV[,"at.level(variable, \"FFD\"):temp.id:at.level(variable, \"FFD\"):temp.id"]))
posterior.mode(cor_bivar1.all_slopefit)
```

```
##        var1
## -0.07115746
```

```
HPDinterval(cor_bivar1.all_slopefit)
```

```
##           lower     upper
## var1 -0.3437299 0.1796289
## attr(,"Probability")
## [1] 0.95
```

Correlation between intercepts and slopes for FFD is smaller than in univariate models. Why?

Intercepts and slopes of RNs are positively correlated: Plants that flower earlier on average are also more responsive to temperature. Fitness is negatively correlated with the intercept, but not with the slope of the RN: individuals that flower earlier on average have higher fitness, but responsiveness to temperature does not seem to affect fitness.

Extract selection coefficients Selection differentials or gradients should be calculated using relative fitness, and models are typically fitted assuming Gaussian errors. However, where the fitness measure follows a non-Gaussian distribution, as is typically the case with skewed distributions of fitness, a GLMM of absolute fitness will be preferable. The resulting covariances returned by the model will then be between the trait on the data scale and fitness on a 'latent' (link-function) scale. These estimates need to be transformed if data-scale estimates of selection are required. However, in the case of a GLMM with a log-link function (e.g. Poisson here), it is possible to exploit the fact that the latent-scale covariance with absolute fitness is equivalent to the data-scale covariance of relative fitness: consequently, and conveniently, the covariance components of the var-covar matrix on the latent scale can simply be treated as selection differentials S. By extension, estimates of selection gradients will also provide data-scale selection gradients.

```r
# Extract 3x3 matrix of variance-covariance values for intercepts and slopes
# of temp, and fitness
# These are in the 2nd-10th columns of model output
P.bivar1.all <- bivar1.all$VCV[,2:10]
P.bivar1.all.mode <- matrix(1:9, nrow = 3)
for (k in 1:9) P.bivar1.all.mode[k] <- posterior.mode(P.bivar1.all[,k])
P.bivar1.all.mode
```

```
##            [,1]       [,2]       [,3]
## [1,]  2.9013938  0.8178953 -1.1958737
## [2,]  0.8178953  0.8111153 -0.1431381
## [3,] -1.1958737 -0.1431381  1.5023975
```

```r
# Extract selection *differentials* (i.e. covariances) for intercept and slope:
# and calculate posterior mode and credible intervals for each
S.bivar1.all <- bivar1.all$VCV[, c(4,7)]
S.bivar1.all <- P.bivar1.all[, c(3,6)] # This is exactly the same as above
colnames(S.bivar1.all) <- c("S_intercepts", "S_slopes")
S.bivar1.all.mode <- P.bivar1.all.mode[1:2, 3]
S.bivar1.all.mode
```

```
## [1] -1.1958737 -0.1431381
```

```r
posterior.mode(mcmc(S.bivar1.all)) # This is exactly the same as above
```

```
## S_intercepts     S_slopes
##   -1.1958737   -0.1431381
```

```r
HPDinterval(mcmc(S.bivar1.all))
```

```
##                   lower      upper
## S_intercepts -1.6019519 -0.8865643
## S_slopes     -0.3693018  0.2146900
## attr(,"Probability")
## [1] 0.95
```

```r
# Estimate selection gradients for intercept and slope (beta = S / P)
# on each sample of posterior and extract their mode
n <- length(bivar1.all$VCV[,2])   # sample size
beta_post_bivar1.all <- matrix(NA, n ,2)

for (i in 1:n) {
  P3 <- matrix(rep(NA, 9), nrow = 3)
  # 3x3 matrix of var-cov for individual X.int, X.slope and fitness
  for (k in 1:9) {P3[k] <- P.bivar1.all[i, k] }
  P2 <- P3[1:2, 1:2]   # 2x2 matrix of just trait intercept & slope var-cov
  S <- P3[1:2, 3]   # selection differentials on traits (last column of P3)
  beta_post_bivar1.all[i,] <- solve(P2) %*% S   # selection gradients beta = P^-1 * S
}

# Finally, extract and plot the selection gradients posterior modes
# and 95% credible intervals for both selection on intercepts (trait value)
# and slopes (trait plasticity).

colnames(beta_post_bivar1.all) <- c("beta_intercepts", "beta_slopes")
posterior.mode(mcmc(beta_post_bivar1.all))
```

```
## beta_intercepts     beta_slopes
##      -0.6068313       0.4106121
```

```r
HPDinterval(mcmc(beta_post_bivar1.all))
```

```
##                       lower      upper
## beta_intercepts -0.78547060 -0.3742286
## beta_slopes      0.09973919  0.9192627
## attr(,"Probability")
## [1] 0.95
```

The selection differentials are "significant" for RN intercept (negative), but not for RN slope. The selection gradients are significant for both RN intercept (negative) and slope (positive). This means that there is significant total and direct selection on the intercept of the RN, selecting for an earlier flowering time on average. Not sure how to interpret the selection on the slope though. The selection differential is not significant, meaning that there is no total selection on the slope, but the selection gradient is significant and positive. I guess this means that, after correcting for the covariance between intercepts and slopes, there is significant selection on the slope. And the selection gradient for the slope being positive means that there is selection for more positive slopes (i.e. less negative = individuals less responsive to temperature, because the relationship among FFD and temperature is negative: earlier flowering (lower FFD) with higher temperatures). Am I interpreting this correctly?

With shoot volume Stack data:

```r
# Create a single data-set "datadef.stack2", with single column at start
# to index observations
datadef.stack2 <- c()
nrow(subset(datadef_total,!is.na(shoot_vol_mean))) # 791 ids with info on shoot_vol
```

```
## [1] 791
```

```r
nrow(subset(datadef,!is.na(FFD)&
              id %in% subset(datadef_total,!is.na(shoot_vol_mean))$id))
```

```
## [1] 2432
```

```r
# 2432 cases with info on FFD and fitness
# corresponding to the 791 ids with info on shoot_vol

# Check that those cases correspond to those 791 individuals
length(unique(subset(datadef,!is.na(FFD)&
         id %in% subset(datadef_total,!is.na(shoot_vol_mean))$id)$id))
```

```
## [1] 791
```

```r
datadef.stack2$Obs <- 1:(791 + 2432)
datadef.stack2$id <- c(as.character(subset(datadef_total,!is.na(shoot_vol_mean))$id),
                       as.character(subset(datadef,!is.na(FFD)&
                             id %in% subset(datadef_total,!is.na(shoot_vol_mean))$id)$id))
# ids in alphabetical order

# Year column is only relevant for FFD, but is set to first_yr for fitness values
datadef.stack2$year <- c(subset(datadef_total,!is.na(shoot_vol_mean))$first_yr,
                          subset(datadef,!is.na(FFD)&
                                   id %in%subset(datadef_total,
                                                 !is.na(shoot_vol_mean))$id)$year)

# Temperature column is only relevant for FFD, but is set to 0 for fitness values
datadef.stack2$temp <- c(rep(0, 791),
                          subset(datadef,!is.na(FFD)&
                                   id %in%subset(datadef_total,
                                                 !is.na(shoot_vol_mean))$id)$cmean_4)

# Shoot volume column is only relevant for fitness, but is set to 0 for FFD values
# Using sqrt of mean shoot volume over all years when available, centered
datadef_total<-datadef_total%>%
  mutate(shoot_vol_mean_sqrt=sqrt(shoot_vol_mean),
         cn_shoot_vol_mean_sqrt=scale(shoot_vol_mean_sqrt,center=T,scale=F))
datadef.stack2$cn_shoot_vol <- c(subset(datadef_total,
                                         !is.na(shoot_vol_mean))$cn_shoot_vol_mean_sqrt,
                                  rep(0, 2432))

# Create single column with first fitness values (ABSOLUTE VALUES), then FFD values:
datadef.stack2$fitness.FFD.stack <- c(round(subset(datadef_total,
                                                   !is.na(shoot_vol_mean))$mean_fitness_fl),
                                      subset(datadef,!is.na(FFD)&
                                               id %in%  subset(datadef_total,
                                                               !is.na(shoot_vol_mean))$id)$FFD)

# Create 3 index columns needed for MCMCglmm
datadef.stack2$traits <- as.factor(c(rep("fitness", 791), rep("FFD", 2432)))
datadef.stack2$variable <- as.factor(datadef.stack2$traits)
# Fitness will be modelled with an overdispersed Poisson
# FFD will be modelled with a Gaussian distribution
# Specify this with the column 'family':
datadef.stack2$family <- c(rep("poisson", 791), rep("gaussian", 2432))
datadef.stack2 <- data.frame(datadef.stack2)

datadef.stack2$id <- as.factor(datadef.stack2$id)
datadef.stack2$year <- as.factor(datadef.stack2$year)
head(datadef.stack2)
```

```
##   Obs      id year temp cn_shoot_vol fitness.FFD.stack  traits variable  family
## 1   1   new_1 2006    0    13.199815                 0 fitness  fitness poisson
## 2   2  new_10 2006    0    69.379629                16 fitness  fitness poisson
## 3   3 new_100 2006    0    14.672097                 6 fitness  fitness poisson
## 4   4 new_101 2006    0     4.988883                 3 fitness  fitness poisson
## 5   5 new_102 2006    0    27.594602                 7 fitness  fitness poisson
## 6   6 new_103 2006    0    11.575830              4 fitness  fitness poisson
```

```r
bivar2.all <- MCMCglmm(fitness.FFD.stack ~ variable - 1 +
                         # ^ means for each variable
                         # (and no overall mean (hence "-1"))
                         at.level(variable, "FFD"):temp +
                       # single fixed effect of temp
                         at.level(variable,"fitness"):cn_shoot_vol,
                       random = ~us(at.level(variable, "FFD")):year +
                         us(at.level(variable, "FFD") +
                              at.level(variable,"FFD"):temp):id,
                       # ^ random intercepts for individual,
                       # and random slopes for temp|id
                       rcov = ~us(at.level(variable, "fitness")):id +
                         # ^ variance between indivduials in fitness
                         # (which is residual variance)
                         us(at.level(variable, "FFD")):Obs,
                         # ^ residual variance within indivduials between years
                       # (labelled by 'Obs')
                       data = datadef.stack2,
                       prior = priorBiv,
                       family = NULL, # specified already in the data-set
                       nitt = 2100 * sc, thin = sc, burnin = 100 * sc,
                       verbose = F,singular.ok = T)
# nitt = burnin + thin*(n samples to keep)
# Aim to store 2000 iterations
save(bivar2.all,file="output/bivar2.all.RData")
```

```r
kable(summary(bivar2.all)$solutions,digits=c(3,3,3,0,3),caption="Fixed effects")
```

Table 10: Fixed effects

	post.mean	l-95% CI	u-95% CI	eff.samp	pMCMC
variableFFD	58.614	56.417	60.958	2151	0.000
variablefitness	0.880	0.777	0.972	2000	0.000
at.level(variable, “FFD”):temp	-2.414	-4.100	-0.771	2000	0.007
at.level(variable, “fitness”):cn_shoot_vol	0.036	0.029	0.043	2000	0.000

```r
kable(summary(bivar2.all)$Gcovariances,digits=c(3,3,3,0),caption="Random effects")
```

Table 11: Random effects

	post.mean	l-95% CI	u-95% CI	eff.samp
at.level(variable, “FFD”):at.level(variable, “FFD”).year	26.29	12.817	44.163	2000

```r
kable(summary(bivar2.all)$Rcovariances,digits=c(3,3,3,0),caption="Random effects")
```

Table 12: Random effects

	post.mean	l-95% CI	u-95% CI	eff.samp
at.level(variable, “FFD”).id:at.level(variable, “FFD”).id	2.925	1.954	3.916	2000
at.level(variable, “FFD”):temp.id:at.level(variable, “FFD”).id	0.764	0.318	1.224	2000
at.level(variable, “fitness”).id:at.level(variable, “FFD”).id	-0.944	-1.279	-0.588	2000
at.level(variable, “FFD”).id:at.level(variable, “FFD”):temp.id	0.764	0.318	1.224	2000
at.level(variable, “FFD”):temp.id:at.level(variable, “FFD”):temp.id	0.865	0.491	1.267	2268
at.level(variable, “fitness”).id:at.level(variable, “FFD”):temp.id	0.129	-0.135	0.416	2000
at.level(variable, “FFD”).id:at.level(variable, “fitness”).id	-0.944	-1.279	-0.588	2000
at.level(variable, “FFD”):temp.id:at.level(variable, “fitness”).id	0.129	-0.135	0.416	2000
at.level(variable, “fitness”).id:at.level(variable, “fitness”).id	1.189	0.997	1.393	1782
at.level(variable, “FFD”):at.level(variable, “FFD”).Obs	18.592	17.465	19.898	2000

```r
kable(diag(autocorr(bivar2.all$Sol)[2, , ]),caption="Autocorrelation")
```

Table 13: Autocorrelation

	x
variableFFD	-0.0366703
variablefitness	0.0173347
at.level(variable, “FFD”):temp	-0.0016296
at.level(variable, “fitness”):cn_shoot_vol	-0.0190020

```r
kable(diag(autocorr(bivar2.all$VCV)[2, , ]),caption="Autocorrelation")
```

Table 14: Autocorrelation

	x
at.level(variable, “FFD”):at.level(variable, “FFD”).year	-0.0063735
at.level(variable, “FFD”).id:at.level(variable, “FFD”).id	-0.0009539
at.level(variable, “FFD”):temp.id:at.level(variable, “FFD”).id	-0.0139485
at.level(variable, “fitness”).id:at.level(variable, “FFD”).id	0.0000299
at.level(variable, “FFD”).id:at.level(variable, “FFD”):temp.id	-0.0139485
at.level(variable, “FFD”):temp.id:at.level(variable, “FFD”):temp.id	-0.0299062
at.level(variable, “fitness”).id:at.level(variable, “FFD”):temp.id	0.0206435
at.level(variable, “FFD”).id:at.level(variable, “fitness”).id	0.0000299
at.level(variable, “FFD”):temp.id:at.level(variable, “fitness”).id	0.0206435
at.level(variable, “fitness”).id:at.level(variable, “fitness”).id	0.0573036
at.level(variable, “FFD”):at.level(variable, “FFD”).Obs	0.0136870

Among-individual correlation between intercepts and slopes for FFD, between FFD and fitness and between fitness and variation in slopes for FFD:

```r
cor_bivar2.all_intslope <-
  bivar2.all$VCV[,"at.level(variable, \"FFD\"):temp.id:at.level(variable, \"FFD\").id"]/
(sqrt(bivar2.all$VCV[,"at.level(variable, \"FFD\").id:at.level(variable, \"FFD\").id"])*
sqrt(bivar2.all$VCV[,"at.level(variable, \"FFD\"):temp.id:at.level(variable, \"FFD\"):temp.id"]))
posterior.mode(cor_bivar2.all_intslope)
```

```
##      var1
## 0.5083269
```

```r
HPDinterval(cor_bivar2.all_intslope)
```

```
##          lower     upper
## var1 0.2392467 0.7087073
## attr(,"Probability")
## [1] 0.95
```

```r
cor_bivar2.all_intfit <-
  bivar2.all$VCV[,"at.level(variable, \"fitness\").id:at.level(variable, \"FFD\").id"]/
  (sqrt(bivar2.all$VCV[,"at.level(variable, \"fitness\").id:at.level(variable, \"fitness\").id"])*
     sqrt(bivar2.all$VCV[,"at.level(variable, \"FFD\").id:at.level(variable, \"FFD\").id"]))
posterior.mode(cor_bivar2.all_intfit)
```

```
##       var1
## -0.5107926
```

```r
HPDinterval(cor_bivar2.all_intfit)
```

```
##           lower     upper
## var1 -0.6549624 -0.352541
## attr(,"Probability")
## [1] 0.95
```

```r
cor_bivar2.all_slopefit <-
  bivar2.all$VCV[,"at.level(variable, \"fitness\").id:at.level(variable, \"FFD\"):temp.id"]/
  (sqrt(bivar2.all$VCV[,"at.level(variable, \"fitness\").id:at.level(variable, \"fitness\").id"])*
     sqrt(bivar2.all$VCV[,"at.level(variable, \"FFD\"):temp.id:at.level(variable, \"FFD\"):temp.id"]))
posterior.mode(cor_bivar2.all_slopefit)
```

```
##      var1
## 0.1380331
```

```r
HPDinterval(cor_bivar2.all_slopefit)
```

```
##           lower     upper
## var1 -0.1405281 0.3977752
## attr(,"Probability")
## [1] 0.95
```

Similar results as in model without shoot volume.

```
# Extract 3x3 matrix of variance-covariance values for intercepts and slopes
# of temp, and fitness
# These are in the 2nd-10th columns of model output
P.bivar2.all<- bivar2.all$VCV[,2:10]
P.bivar2.all.mode <- matrix(1:9, nrow = 3)
for (k in 1:9) P.bivar2.all.mode[k] <- posterior.mode(P.bivar2.all
                                                       [,k])
P.bivar2.all.mode
```

Extract selection coefficients

```
##            [,1]      [,2]       [,3]
## [1,]  2.8700716 0.7356946 -0.8200934
## [2,]  0.7356946 0.8154952  0.1194087
## [3,] -0.8200934 0.1194087  1.1438249
```

```
# Extract selection *differentials* (i.e. covariances) for intercept and slope:
# and calculate posterior mode and credible intervals for each
S.bivar2.all <- bivar2.all$VCV[, c(4,7)]
S.bivar2.all <- P.bivar2.all[, c(3,6)] # This is exactly the same as above
colnames(S.bivar2.all) <- c("S_intercepts", "S_slopes")
S.bivar2.all.mode <- P.bivar2.all.mode[1:2, 3]
S.bivar2.all.mode
```

```
## [1] -0.8200934  0.1194087
```

```
posterior.mode(mcmc(S.bivar2.all)) # This is exactly the same as above
```

```
## S_intercepts     S_slopes
##   -0.8200934    0.1194087
```

```
HPDinterval(mcmc(S.bivar2.all))
```

```
##                   lower      upper
## S_intercepts -1.2787717 -0.5880987
## S_slopes     -0.1345258  0.4158604
## attr(,"Probability")
## [1] 0.95
```

```
# Estimate selection gradients for intercept and slope (beta = S / P)
# on each sample of posterior and extract their mode
n <- length(bivar2.all$VCV[,2])   # sample size
beta_post_bivar2.all <- matrix(NA, n ,2)

for (i in 1:n) {
  P3 <- matrix(rep(NA, 9), nrow = 3)
  # 3x3 matrix of var-cov for individual X.int, X.slope and fitness
  for (k in 1:9) {P3[k] <- P.bivar2.all[i, k] }
  P2 <- P3[1:2, 1:2]    # 2x2 matrix of just trait intercept & slope var-cov
  S <- P3[1:2, 3]    # selection differentials on traits (last column of P3)
  beta_post_bivar2.all[i,] <- solve(P2) %*% S    # selection gradients beta = P^-1 * S
}

# Finally, extract and plot the selection gradients posterior modes
# and 95% credible intervals for both selection on intercepts (trait value)
# and slopes (trait plasticity).

colnames(beta_post_bivar2.all) <- c("beta_intercepts", "beta_slopes")
posterior.mode(mcmc(beta_post_bivar2.all))
```

```
## beta_intercepts     beta_slopes
##      -0.4625847       0.6062740
```

```
HPDinterval(mcmc(beta_post_bivar2.all))
```

```
##                      lower      upper
## beta_intercepts -0.6661626 -0.3341345
## beta_slopes      0.2621512  0.9523805
## attr(,"Probability")
## [1] 0.95
```

Selection coefficients give similar results as in model without shoot volume.

Mean fitness per year of study

With no condition variable Stack data:

```
# Create a single data-set "datadef.stack3", with single column at start
# to index observations
datadef.stack3 <- c()
datadef.stack3$Obs <- 1:(837 + 2478)
datadef.stack3$id <- c(as.character(datadef_total$id),
                      as.character(subset(datadef,!is.na(FFD))$id))
# ids in alphabetical order

# Year column is only relevant for FFD, but is set to first_yr for fitness values
datadef.stack3$year <- c(datadef_total$first_yr,
                         subset(datadef,!is.na(FFD))$year)

# Temperature column is only relevant for FFD, but is set to 0 for fitness values
datadef.stack3$temp <- c(rep(0, 837), subset(datadef,!is.na(FFD))$cmean_4)

# Create single column with first fitness values (ABSOLUTE VALUES), then FFD values:
datadef.stack3$fitness.FFD.stack <- c(round(datadef_total$mean_fitness_study),
                                      subset(datadef,!is.na(FFD))$FFD)

# Create 3 index columns needed for MCMCglmm
datadef.stack3$traits <- as.factor(c(rep("fitness", 837), rep("FFD", 2478)))
datadef.stack3$variable <- as.factor(datadef.stack3$traits)
# Fitness will be modelled with an overdispersed Poisson
```

```r
# FFD will be modelled with a Gaussian distribution
# Specify this with the column 'family':
datadef.stack3$family <- c(rep("poisson", 837), rep("gaussian", 2478))
datadef.stack3 <- data.frame(datadef.stack3)

datadef.stack3$id <- as.factor(datadef.stack3$id)
datadef.stack3$year <- as.factor(datadef.stack3$year)
head(datadef.stack3)
```

```
##   Obs      id year temp fitness.FFD.stack  traits variable  family
## 1   1   new_1 2006    0                 0 fitness  fitness poisson
## 2   2  new_10 2006    0                14 fitness  fitness poisson
## 3   3 new_100 2006    0                 4 fitness  fitness poisson
## 4   4 new_101 2006    0                 2 fitness  fitness poisson
## 5   5 new_102 2006    0                 6 fitness  fitness poisson
## 6   6 new_103 2006    0                 4 fitness  fitness poisson
```

```r
bivar3.all <- MCMCglmm(fitness.FFD.stack ~ variable - 1 +
                         # ^ means for each variable
                         # (and no overall mean (hence "-1"))
                         at.level(variable, "FFD"):temp,
                       # single fixed effect of temp
                       random = ~us(at.level(variable, "FFD")):year +
                         us(at.level(variable, "FFD") +
                              at.level(variable,"FFD"):temp):id,
                       # ^ random intercepts for individual,
                       # and random slopes for temp|id
                       rcov = ~us(at.level(variable, "fitness")):id +
                         # ^ variance between indivduials in fitness
                         # (which is residual variance)
                         us(at.level(variable, "FFD")):Obs,
                         # ^ residual variance within indivduials between years
                       # (labelled by 'Obs')
                       data = datadef.stack3,
                       prior = priorBiv,
                       family = NULL, # specified already in the data-set
                       nitt = 2100 * sc, thin = sc, burnin = 100 * sc,
                       verbose = F,singular.ok = T)
# nitt = burnin + thin*(n samples to keep)
# Aim to store 2000 iterations
save(bivar3.all,file="output/bivar3.all.RData")
```

```r
kable(summary(bivar3.all)$solutions,digits=c(3,3,3,0,3),caption="Fixed effects")
```

Table 15: Fixed effects

	post.mean	l-95% CI	u-95% CI	eff.samp	pMCMC
variableFFD	58.814	56.673	60.933	2197	0.000
variablefitness	0.209	0.098	0.336	2000	0.000
at.level(variable, “FFD”):temp	-2.282	-3.947	-0.705	2000	0.009

```
kable(summary(bivar3.all)$Gcovariances,digits=c(3,3,3,0),caption="Random effects")
```

Table 16: Random effects

	post.mean	l-95% CI	u-95% CI	eff.samp
at.level(variable, “FFD”):at.level(variable, “FFD”).year	25.72	11.503	42.716	2000

```
kable(summary(bivar3.all)$Rcovariances,digits=c(3,3,3,0),caption="Random effects")
```

Table 17: Random effects

	post.mean	l-95% CI	u-95% CI	eff.samp
at.level(variable, “FFD”).id:at.level(variable, “FFD”).id	2.759	1.862	3.667	2000
at.level(variable, “FFD”):temp.id:at.level(variable, “FFD”).id	0.879	0.451	1.313	2000
at.level(variable, “fitness”).id:at.level(variable, “FFD”).id	-1.371	-1.752	-1.007	2000
at.level(variable, “FFD”).id:at.level(variable, “FFD”):temp.id	0.879	0.451	1.313	2000
at.level(variable, “FFD”):temp.id:at.level(variable, “FFD”):temp.id	0.808	0.441	1.192	2000
at.level(variable, “fitness”).id:at.level(variable, “FFD”):temp.id	-0.272	-0.581	0.020	2000
at.level(variable, “FFD”).id:at.level(variable, “fitness”).id	-1.371	-1.752	-1.007	2000
at.level(variable, “FFD”):temp.id:at.level(variable, “fitness”).id	-0.272	-0.581	0.020	2000
at.level(variable, “fitness”).id:at.level(variable, “fitness”).id	1.649	1.384	1.928	2000
at.level(variable, “FFD”):at.level(variable, “FFD”).Obs	18.698	17.525	19.985	1653

```
kable(diag(autocorr(bivar3.all$Sol)[2, , ]),caption="Autocorrelation")
```

Table 18: Autocorrelation

	x
variableFFD	0.0024323
variablefitness	-0.0189290
at.level(variable, “FFD”):temp	0.0025070

```
kable(diag(autocorr(bivar3.all$VCV)[2, , ]),caption="Autocorrelation")
```

Table 19: Autocorrelation

	x
at.level(variable, “FFD”):at.level(variable, “FFD”).year	0.0087356
at.level(variable, “FFD”).id:at.level(variable, “FFD”).id	-0.0112095
at.level(variable, “FFD”):temp.id:at.level(variable, “FFD”).id	0.0053146

	x
at.level(variable, “fitness”).id:at.level(variable, “FFD”).id	0.0138760
at.level(variable, “FFD”).id:at.level(variable, “FFD”):temp.id	0.0053146
at.level(variable, “FFD”):temp.id:at.level(variable, “FFD”):temp.id	-0.0012240
at.level(variable, “fitness”).id:at.level(variable, “FFD”):temp.id	0.0052883
at.level(variable, “FFD”).id:at.level(variable, “fitness”).id	0.0138760
at.level(variable, “FFD”):temp.id:at.level(variable, “fitness”).id	0.0052883
at.level(variable, “fitness”).id:at.level(variable, “fitness”).id	0.0061645
at.level(variable, “FFD”):at.level(variable, “FFD”).Obs	0.0205247

Among-individual correlation between intercepts and slopes for FFD, between FFD and fitness and between fitness and variation in slopes for FFD:

```
cor_bivar3.all_intslope <-
  bivar3.all$VCV[,"at.level(variable, \"FFD\"):temp.id:at.level(variable, \"FFD\").id"]/
(sqrt(bivar3.all$VCV[,"at.level(variable, \"FFD\").id:at.level(variable, \"FFD\").id"])*
sqrt(bivar3.all$VCV[,"at.level(variable, \"FFD\"):temp.id:at.level(variable, \"FFD\"):temp.id"]))
posterior.mode(cor_bivar3.all_intslope)
```

```
##      var1
## 0.6449642
```

```
HPDinterval(cor_bivar3.all_intslope)
```

```
##          lower     upper
## var1 0.3769574 0.7925492
## attr(,"Probability")
## [1] 0.95
```

```
cor_bivar3.all_intfit <-
  bivar3.all$VCV[,"at.level(variable, \"fitness\").id:at.level(variable, \"FFD\").id"]/
  (sqrt(bivar3.all$VCV[,"at.level(variable, \"fitness\").id:at.level(variable, \"fitness\").id"])*
     sqrt(bivar3.all$VCV[,"at.level(variable, \"FFD\").id:at.level(variable, \"FFD\").id"]))
posterior.mode(cor_bivar3.all_intfit)
```

```
##       var1
## -0.6494609
```

```
HPDinterval(cor_bivar3.all_intfit)
```

```
##           lower      upper
## var1 -0.7796626 -0.5143761
## attr(,"Probability")
## [1] 0.95
```

```
cor_bivar3.all_slopefit <-
  bivar3.all$VCV[,"at.level(variable, \"fitness\").id:at.level(variable, \"FFD\"):temp.id"]/
  (sqrt(bivar3.all$VCV[,"at.level(variable, \"fitness\").id:at.level(variable, \"fitness\").id"])*
     sqrt(bivar3.all$VCV[,"at.level(variable, \"FFD\"):temp.id:at.level(variable, \"FFD\"):temp.id"]))
posterior.mode(cor_bivar3.all_slopefit)
```

```
##       var1
## -0.2222562
```

```
HPDinterval(cor_bivar3.all_slopefit)
```

```
##           lower      upper
## var1 -0.4972722 0.01544238
## attr(,"Probability")
## [1] 0.95
```

Similar results as in previous models.

```


# Extract 3x3 matrix of variance-covariance values for intercepts and slopes
# of temp, and fitness
# These are in the 2nd-10th columns of model output
P.bivar3.all<- bivar3.all$VCV[,2:10]
P.bivar3.all.mode <- matrix(1:9, nrow = 3)
for (k in 1:9) P.bivar3.all.mode[k] <- posterior.mode(P.bivar3.all
                                                    [,k])
P.bivar3.all.mode
```

Extract selection coefficients

```
##            [,1]       [,2]       [,3]
## [1,]  2.5850376  0.8635925 -1.3272087
## [2,]  0.8635925  0.7054789 -0.3118816
## [3,] -1.3272087 -0.3118816  1.6130686
```

```
# Extract selection *differentials* (i.e. covariances) for intercept and slope:
# and calculate posterior mode and credible intervals for each
S.bivar3.all <- bivar3.all$VCV[, c(4,7)]
S.bivar3.all <- P.bivar3.all[, c(3,6)] # This is exactly the same as above
colnames(S.bivar3.all) <- c("S_intercepts", "S_slopes")
S.bivar3.all.mode <- P.bivar3.all.mode[1:2, 3]
S.bivar3.all.mode
```

```
## [1] -1.3272087 -0.3118816
```

```
posterior.mode(mcmc(S.bivar3.all)) # This is exactly the same as above
```

```
## S_intercepts     S_slopes
##   -1.3272087   -0.3118816
```

```
HPDinterval(mcmc(S.bivar3.all))
```

```
##                  lower       upper
## S_intercepts -1.751607 -1.00729831
## S_slopes     -0.581332  0.02043382
## attr(,"Probability")
## [1] 0.95
```

```
# Estimate selection gradients for intercept and slope (beta = S / P)
# on each sample of posterior and extract their mode
n <- length(bivar3.all$VCV[,2])   # sample size
beta_post_bivar3.all <- matrix(NA, n ,2)

for (i in 1:n) {
  P3 <- matrix(rep(NA, 9), nrow = 3)
  # 3x3 matrix of var-cov for individual X.int, X.slope and fitness
  for (k in 1:9) {P3[k] <- P.bivar3.all[i, k] }
  P2 <- P3[1:2, 1:2]   # 2x2 matrix of just trait intercept & slope var-cov
  S <- P3[1:2, 3]   # selection differentials on traits (last column of P3)
  beta_post_bivar3.all[i,] <- solve(P2) %*% S   # selection gradients beta = P^-1 * S
}

# Finally, extract and plot the selection gradients posterior modes
# and 95% credible intervals for both selection on intercepts (trait value)
# and slopes (trait plasticity).

colnames(beta_post_bivar3.all) <- c("beta_intercepts", "beta_slopes")
posterior.mode(mcmc(beta_post_bivar3.all))
```

```
## beta_intercepts     beta_slopes
##      -0.5735520       0.3706379
```

```
HPDinterval(mcmc(beta_post_bivar3.all))
```

```
##                      lower      upper
## beta_intercepts -0.8527109 -0.3775726
## beta_slopes     -0.1279825  0.8710638
## attr(,"Probability")
## [1] 0.95
```

The selection differentials are “significant” for RN intercept (negative) but not for slope. The selection gradients are significant for RN intercept (negative) but not for RN slope. This means that there is significant total and direct selection on the intercept of the RN.

With shoot volume Stack data:

```
# Create a single data-set "datadef.stack4", with single column at start
# to index observations
datadef.stack4 <- c()
nrow(subset(datadef_total,!is.na(shoot_vol_mean))) # 791 ids with info on shoot_vol
```

```
## [1] 791
```

```
nrow(subset(datadef,!is.na(FFD)&
              id %in% subset(datadef_total,!is.na(shoot_vol_mean))$id))
```

```
## [1] 2432
```

```r
# 2432 cases with info on FFD and fitness
# corresponding to the 791 ids with info on shoot_vol

# Check that those cases correspond to those 791 individuals
length(unique(subset(datadef,!is.na(FFD)&
         id %in% subset(datadef_total,!is.na(shoot_vol_mean))$id)$id))
```

```
## [1] 791
```

```r
datadef.stack4$Obs <- 1:(791 + 2432)
datadef.stack4$id <- c(as.character(subset(datadef_total,!is.na(shoot_vol_mean))$id),
                       as.character(subset(datadef,!is.na(FFD)&
                             id %in% subset(datadef_total,!is.na(shoot_vol_mean))$id)$id))
# ids in alphabetical order

# Year column is only relevant for FFD, but is set to first_yr for fitness values
datadef.stack4$year <- c(subset(datadef_total,!is.na(shoot_vol_mean))$first_yr,
                         subset(datadef,!is.na(FFD)&
                                  id %in%subset(datadef_total,
                                                !is.na(shoot_vol_mean))$id)$year)

# Temperature column is only relevant for FFD, but is set to 0 for fitness values
datadef.stack4$temp <- c(rep(0, 791),
                         subset(datadef,!is.na(FFD)&
                                  id %in%subset(datadef_total,
                                                !is.na(shoot_vol_mean))$id)$cmean_4)

# Shoot volume column is only relevant for fitness, but is set to 0 for FFD values
# Using sqrt of mean shoot volume over all years when available, centered
datadef.stack4$cn_shoot_vol <- c(subset(datadef_total,
                                        !is.na(shoot_vol_mean))$cn_shoot_vol_mean_sqrt,
                                 rep(0, 2432))

# Create single column with first fitness values (ABSOLUTE VALUES), then FFD values:
datadef.stack4$fitness.FFD.stack <- c(round(subset(datadef_total,
                                                   !is.na(shoot_vol_mean))$mean_fitness_study),
                                      subset(datadef,!is.na(FFD)&
                                               id %in%  subset(datadef_total,
                                                               !is.na(shoot_vol_mean))$id)$FFD)

# Create 3 index columns needed for MCMCglmm
datadef.stack4$traits <- as.factor(c(rep("fitness", 791), rep("FFD", 2432)))
datadef.stack4$variable <- as.factor(datadef.stack4$traits)
# Fitness will be modelled with an overdispersed Poisson
# FFD will be modelled with a Gaussian distribution
# Specify this with the column 'family':
datadef.stack4$family <- c(rep("poisson", 791), rep("gaussian", 2432))
datadef.stack4 <- data.frame(datadef.stack4)

datadef.stack4$id <- as.factor(datadef.stack4$id)
datadef.stack4$year <- as.factor(datadef.stack4$year)
head(datadef.stack4)
```

```
##   Obs      id year temp cn_shoot_vol fitness.FFD.stack  traits variable  family
## 1   1   new_1 2006    0    13.199815                 0 fitness  fitness poisson
## 2   2  new_10 2006    0    69.379629                14 fitness  fitness poisson
## 3   3 new_100 2006    0    14.672097                 4 fitness  fitness poisson
## 4   4 new_101 2006    0     4.988883                 2 fitness  fitness poisson
## 5   5 new_102 2006    0    27.594602                 6 fitness  fitness poisson
## 6   6 new_103 2006    0    11.575830                 4 fitness  fitness poisson
```

```
bivar4.all <- MCMCglmm(fitness.FFD.stack ~ variable - 1 +
                         # ^ means for each variable
                         # (and no overall mean (hence "-1"))
                         at.level(variable, "FFD"):temp +
                       # single fixed effect of temp
                         at.level(variable,"fitness"):cn_shoot_vol,
                       random = ~us(at.level(variable, "FFD")):year +
                         us(at.level(variable, "FFD") +
                              at.level(variable,"FFD"):temp):id,
                       # ^ random intercepts for individual,
                       # and random slopes for temp|id
                       rcov = ~us(at.level(variable, "fitness")):id +
                         # ^ variance between indivduals in fitness
                         # (which is residual variance)
                         us(at.level(variable, "FFD")):Obs,
                         # ^ residual variance within indivduials between years
                       # (labelled by 'Obs')
                       data = datadef.stack4,
                       prior = priorBiv,
                       family = NULL, # specified already in the data-set
                       nitt = 2100 * sc, thin = sc, burnin = 100 * sc,
                       verbose = F,singular.ok = T)
# nitt = burnin + thin*(n samples to keep)
# Aim to store 2000 iterations
save(bivar4.all,file="output/bivar4.all.RData")
```

```
kable(summary(bivar4.all)$solutions,digits=c(3,3,3,0,3),caption="Fixed effects")
```

Table 20: Fixed effects

	post.mean	l-95% CI	u-95% CI	eff.samp	pMCMC
variableFFD	58.587	56.252	60.571	2000	0.00
variablefitness	0.224	0.097	0.321	2206	0.00
at.level(variable, "FFD"):temp	-2.410	-3.947	-0.781	2000	0.01
at.level(variable, "fitness"):cn_shoot_vol	0.050	0.043	0.058	2000	0.00

```
kable(summary(bivar4.all)$Gcovariances,digits=c(3,3,3,0),caption="Random effects")
```

Table 21: Random effects

	post.mean	l-95% CI	u-95% CI	eff.samp
at.level(variable, “FFD”):at.level(variable, “FFD”).year	26.492	12.394	44.92	1841

```
kable(summary(bivar4.all)$Rcovariances,digits=c(3,3,3,0),caption="Random effects")
```

Table 22: Random effects

	post.mean	l-95% CI	u-95% CI	eff.samp
at.level(variable, “FFD”).id:at.level(variable, “FFD”).id	2.884	2.046	3.936	2000
at.level(variable, “FFD”):temp.id:at.level(variable, “FFD”).id	0.836	0.412	1.289	2000
at.level(variable, “fitness”).id:at.level(variable, “FFD”).id	-0.953	-1.284	-0.592	2000
at.level(variable, “FFD”).id:at.level(variable, “FFD”):temp.id	0.836	0.412	1.289	2000
at.level(variable, “FFD”):temp.id:at.level(variable, “FFD”):temp.id	0.846	0.467	1.249	2000
at.level(variable, “fitness”).id:at.level(variable, “FFD”):temp.id	0.011	-0.283	0.287	1846
at.level(variable, “FFD”).id:at.level(variable, “fitness”).id	-0.953	-1.284	-0.592	2000
at.level(variable, “FFD”):temp.id:at.level(variable, “fitness”).id	0.011	-0.283	0.287	1846
at.level(variable, “fitness”).id:at.level(variable, “fitness”).id	1.125	0.922	1.318	2251
at.level(variable, “FFD”):at.level(variable, “FFD”).Obs	18.677	17.381	19.774	2000

```
kable(diag(autocorr(bivar4.all$Sol)[2, , ]),caption="Autocorrelation")
```

Table 23: Autocorrelation

	x
variableFFD	-0.0173294
variablefitness	-0.0491684
at.level(variable, “FFD”):temp	-0.0119735
at.level(variable, “fitness”):cn_shoot_vol	-0.0174937

```
kable(diag(autocorr(bivar4.all$VCV)[2, , ]),caption="Autocorrelation")
```

Table 24: Autocorrelation

	x
at.level(variable, “FFD”):at.level(variable, “FFD”).year	0.0412116
at.level(variable, “FFD”).id:at.level(variable, “FFD”).id	-0.0102260
at.level(variable, “FFD”):temp.id:at.level(variable, “FFD”).id	-0.0252653
at.level(variable, “fitness”).id:at.level(variable, “FFD”).id	-0.0121487
at.level(variable, “FFD”).id:at.level(variable, “FFD”):temp.id	-0.0252653

	x
at.level(variable, “FFD”):temp.id:at.level(variable, “FFD”):temp.id	0.0023645
at.level(variable, “fitness”).id:at.level(variable, “FFD”):temp.id	0.0401482
at.level(variable, “FFD”).id:at.level(variable, “fitness”).id	-0.0121487
at.level(variable, “FFD”):temp.id:at.level(variable, “fitness”).id	0.0401482
at.level(variable, “fitness”).id:at.level(variable, “fitness”).id	-0.0151167
at.level(variable, “FFD”):at.level(variable, “FFD”).Obs	0.0020280

Among-individual correlation between intercepts and slopes for FFD, between FFD and fitness and between fitness and variation in slopes for FFD:

```
cor_bivar4.all_intslope <-
  bivar4.all$VCV[,"at.level(variable, \"FFD\"):temp.id:at.level(variable, \"FFD\").id"]/
(sqrt(bivar4.all$VCV[,"at.level(variable, \"FFD\").id:at.level(variable, \"FFD\").id"])*
sqrt(bivar4.all$VCV[,"at.level(variable, \"FFD\"):temp.id:at.level(variable, \"FFD\"):temp.id"]))
posterior.mode(cor_bivar4.all_intslope)
```

```
##      var1
## 0.5879959
```

```
HPDinterval(cor_bivar4.all_intslope)
```

```
##         lower     upper
## var1 0.314819 0.7674397
## attr(,"Probability")
## [1] 0.95
```

```
cor_bivar4.all_intfit <-
  bivar4.all$VCV[,"at.level(variable, \"fitness\").id:at.level(variable, \"FFD\").id"]/
  (sqrt(bivar4.all$VCV[,"at.level(variable, \"fitness\").id:at.level(variable, \"fitness\").id"])*
     sqrt(bivar4.all$VCV[,"at.level(variable, \"FFD\").id:at.level(variable, \"FFD\").id"]))
posterior.mode(cor_bivar4.all_intfit)
```

```
##       var1
## -0.5718007
```

```
HPDinterval(cor_bivar4.all_intfit)
```

```
##           lower    upper
## var1 -0.6857635 -0.37053
## attr(,"Probability")
## [1] 0.95
```

```
cor_bivar4.all_slopefit <-
  bivar4.all$VCV[,"at.level(variable, \"fitness\").id:at.level(variable, \"FFD\"):temp.id"]/
  (sqrt(bivar4.all$VCV[,"at.level(variable, \"fitness\").id:at.level(variable, \"fitness\").id"])*
     sqrt(bivar4.all$VCV[,"at.level(variable, \"FFD\"):temp.id:at.level(variable, \"FFD\"):temp.id"]))
posterior.mode(cor_bivar4.all_slopefit)
```

```
##        var1
## 0.002384568
```

```
HPDinterval(cor_bivar4.all_slopefit)
```

```
##           lower     upper
## var1 -0.2698282 0.3046075
## attr(,"Probability")
## [1] 0.95
```

Similar results as in previous models.

```
# Extract 3x3 matrix of variance-covariance values for intercepts and slopes
# of temp, and fitness
# These are in the 2nd-10th columns of model output
P.bivar4.all<- bivar4.all$VCV[,2:10]
P.bivar4.all.mode <- matrix(1:9, nrow = 3)
for (k in 1:9) P.bivar4.all.mode[k] <- posterior.mode(P.bivar4.all
                                                     [,k])
P.bivar4.all.mode
```

Extract selection coefficients

```
##            [,1]       [,2]        [,3]
## [1,]  2.7271270 0.85905053 -1.04103710
## [2,]  0.8590505 0.74051288  0.04319153
## [3,] -1.0410371 0.04319153  1.12528101
```

```
# Extract selection *differentials* (i.e. covariances) for intercept and slope:
# and calculate posterior mode and credible intervals for each
S.bivar4.all <- bivar4.all$VCV[, c(4,7)]
S.bivar4.all <- P.bivar4.all[, c(3,6)] # This is exactly the same as above
colnames(S.bivar4.all) <- c("S_intercepts", "S_slopes")
S.bivar4.all.mode <- P.bivar4.all.mode[1:2, 3]
S.bivar4.all.mode
```

```
## [1] -1.04103710  0.04319153
```

```
posterior.mode(mcmc(S.bivar4.all)) # This is exactly the same as above
```

```
## S_intercepts     S_slopes
##  -1.04103710   0.04319153
```

```
HPDinterval(mcmc(S.bivar4.all))
```

```
##                   lower      upper
## S_intercepts -1.2836473 -0.5918830
## S_slopes     -0.2826174  0.2874349
## attr(,"Probability")
## [1] 0.95
```

```
# Estimate selection gradients for intercept and slope (beta = S / P)
# on each sample of posterior and extract their mode
n <- length(bivar4.all$VCV[,2])   # sample size
beta_post_bivar4.all <- matrix(NA, n ,2)

for (i in 1:n) {
  P3 <- matrix(rep(NA, 9), nrow = 3)
  # 3x3 matrix of var-cov for individual X.int, X.slope and fitness
  for (k in 1:9) {P3[k] <- P.bivar4.all[i, k] }
  P2 <- P3[1:2, 1:2]   # 2x2 matrix of just trait intercept & slope var-cov
  S <- P3[1:2, 3]   # selection differentials on traits (last column of P3)
  beta_post_bivar4.all[i,] <- solve(P2) %*% S   # selection gradients beta = P^-1 * S
}

# Finally, extract and plot the selection gradients posterior modes
# and 95% credible intervals for both selection on intercepts (trait value)
# and slopes (trait plasticity).

colnames(beta_post_bivar4.all) <- c("beta_intercepts", "beta_slopes")
posterior.mode(mcmc(beta_post_bivar4.all))
```

```
## beta_intercepts     beta_slopes
##      -0.4619623       0.4791965
```

```
HPDinterval(mcmc(beta_post_bivar4.all))
```

```
##                      lower      upper
## beta_intercepts -0.6555253 -0.3008405
## beta_slopes      0.1124081  0.8388997
## attr(,"Probability")
## [1] 0.95
```

The selection differentials are "significant" for RN intercept (negative), but not for RN slope. The selection gradients are significant for both RN intercept (negative) and slope (positive).

brms

Mean fitness per flowering event

With no condition variable I tried to use the ID-syntax to specify fitness to be correlated with the intercept and slope of FFD on temperature - check that this is correctly done!

Regarding distributions, I tried Poisson distribution for fitness, but not sure how eventual overdispersion is handled. I also tried adding an observation-level random effect, and using a negative binomial distribution. Results seem quite similar.

```
datadef<-left_join(datadef,datadef_total[c(1:3,9)]) # Add info on mean fitness and mean shoot volume
bf_fitness <- bf(round(mean_fitness_fl) ~  (1|ID1|id)) # Set up model formula
bf_FFD <- bf(FFD ~ cmean_4 + (cmean_4|ID1|id) + (1|year)) # Set up model formula
```

```
# Specifying group-level effects of the same grouping factor (id here) to be correlated across formulas
# Expand the | operator into |<ID>|, where <ID> can be any value (ID1 here)
# Group-level terms with the same ID1 will be modeled as correlated if they share same grouping factor(
```

```
bivar1.all.brm.pois<-brm(bf_fitness + bf_FFD, family = c(poisson, gaussian), data = datadef,
                 warmup = 1000,iter = 4000,chains=4,thin=2,
                 inits = "random",seed = 12345,cores = my.cores)
# Total of 6000 post-warmup samples
save(bivar1.all.brm.pois,
     file="output/bivar1.all.brm.pois.RData")
```

```
summary(bivar1.all.brm.pois)
```

Poisson distribution

```
##  Family: MV(poisson, gaussian)
##   Links: mu = log
##          mu = identity; sigma = identity
## Formula: round(mean_fitness_fl) ~ (1 | ID1 | id)
##          FFD ~ cmean_4 + (cmean_4 | ID1 | id) + (1 | year)
##    Data: datadef (Number of observations: 2478)
## Samples: 4 chains, each with iter = 4000; warmup = 1000; thin = 2;
##          total post-warmup samples = 6000
##
## Group-Level Effects:
## ~id (Number of levels: 837)
##                                                     Estimate Est.Error l-95% CI
## sd(roundmeanfitnessfl_Intercept)                        1.38      0.05     1.28
## sd(FFD_Intercept)                                       1.64      0.14     1.36
## sd(FFD_cmean_4)                                         0.79      0.14     0.53
## cor(roundmeanfitnessfl_Intercept,FFD_Intercept)        -0.53      0.07    -0.66
## cor(roundmeanfitnessfl_Intercept,FFD_cmean_4)          -0.04      0.13    -0.31
## cor(FFD_Intercept,FFD_cmean_4)                          0.69      0.13     0.41
##                                                     u-95% CI Rhat Bulk_ESS Tail_ESS
## sd(roundmeanfitnessfl_Intercept)                        1.48 1.01     1086     2302
## sd(FFD_Intercept)                                       1.92 1.00     3605     4984
## sd(FFD_cmean_4)                                         1.06 1.00     2369     4061
## cor(roundmeanfitnessfl_Intercept,FFD_Intercept)        -0.38 1.00     4199     4984
## cor(roundmeanfitnessfl_Intercept,FFD_cmean_4)           0.22 1.00     4979     5338
## cor(FFD_Intercept,FFD_cmean_4)                          0.91 1.00     1733     3676
##
## ~year (Number of levels: 22)
##                     Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sd(FFD_Intercept)       5.11      0.86     3.76     7.11 1.00     2625     4238
##
## Population-Level Effects:
##                               Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS
## roundmeanfitnessfl_Intercept      0.76      0.06     0.65     0.87 1.00      959
## FFD_Intercept                    58.68      1.08    56.59    60.82 1.00     1525
## FFD_cmean_4                     -2.33      0.81    -3.94    -0.69 1.00     1997
##                              Tail_ESS
## roundmeanfitnessfl_Intercept     1969
## FFD_Intercept                    2549
## FFD_cmean_4                      3518
##
## Family Specific Parameters:
##           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sigma_FFD     4.35      0.07     4.21     4.49 1.00     3275     4014
##
## Samples were drawn using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

How to extract selection coefficients from models fitted with brms?

```
datadef$OLRE<-seq_len(nrow(datadef))
bf_fitness_OLRE <- bf(round(mean_fitness_fl) ~  (1|ID1|id) + (1|OLRE))
bf_FFD_OLRE <- bf(FFD ~ cmean_4 + (cmean_4|ID1|id) + (1|year) + (1|OLRE))
```

```
bivar1.all.brm.OLRE<-brm(bf_fitness_OLRE + bf_FFD_OLRE, family = c(poisson, gaussian),
                         data = datadef,warmup = 1000,iter = 4000,chains=4,thin=2,
                         inits = "random",seed = 12345,cores = my.cores)
# Model gives some warnings - not sure what the problems are!
save(bivar1.all.brm.OLRE,
     file="output/bivar1.all.brm.OLRE.RData")
```

```
summary(bivar1.all.brm.OLRE)
```

Poisson distribution, observation-level random effect

```
##  Family: MV(poisson, gaussian)
##   Links: mu = log
##          mu = identity; sigma = identity
## Formula: round(mean_fitness_fl) ~ (1 | ID1 | id) + (1 | OLRE)
##          FFD ~ cmean_4 + (cmean_4 | ID1 | id) + (1 | year) + (1 | OLRE)
##    Data: datadef (Number of observations: 2478)
## Samples: 4 chains, each with iter = 4000; warmup = 1000; thin = 2;
##          total post-warmup samples = 6000
##
## Group-Level Effects:
## ~id (Number of levels: 837)
##                                                  Estimate Est.Error l-95% CI
## sd(roundmeanfitnessfl_Intercept)                     1.38      0.05     1.27
## sd(FFD_Intercept)                                    1.64      0.14     1.35
## sd(FFD_cmean_4)                                      0.78      0.13     0.53
## cor(roundmeanfitnessfl_Intercept,FFD_Intercept)     -0.52      0.07    -0.66
## cor(roundmeanfitnessfl_Intercept,FFD_cmean_4)      -0.04      0.14    -0.31
## cor(FFD_Intercept,FFD_cmean_4)                      0.70      0.13     0.42
##                                                  u-95% CI Rhat Bulk_ESS Tail_ESS
## sd(roundmeanfitnessfl_Intercept)                     1.48 1.00     1028     1963
## sd(FFD_Intercept)                                    1.91 1.00     2146     3348
## sd(FFD_cmean_4)                                      1.05 1.00     1119     2805
## cor(roundmeanfitnessfl_Intercept,FFD_Intercept)     -0.38 1.00     2916     4366
## cor(roundmeanfitnessfl_Intercept,FFD_cmean_4)        0.22 1.00     2632     4601
## cor(FFD_Intercept,FFD_cmean_4)                       0.92 1.00      794      729
##
## ~OLRE (Number of levels: 2478)
##                                   Estimate Est.Error l-95% CI u-95% CI Rhat
## sd(roundmeanfitnessfl_Intercept)      0.01      0.01     0.00     0.02 1.00
## sd(FFD_Intercept)                     2.46      1.34     0.12     4.27 1.74
##                                   Bulk_ESS Tail_ESS
## sd(roundmeanfitnessfl_Intercept)      4697     4476
## sd(FFD_Intercept)                        6       15
##
## ~year (Number of levels: 22)
##                   Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sd(FFD_Intercept)     5.09      0.86     3.75     7.04 1.00     2304     3744
##
## Population-Level Effects:
##                               Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS
## roundmeanfitnessfl_Intercept      0.76      0.06     0.65     0.87 1.01     1012
## FFD_Intercept                    58.75      1.12    56.47    60.97 1.00     1375
## FFD_cmean_4                      -2.33      0.83    -3.99    -0.69 1.00     2452
##                               Tail_ESS
## roundmeanfitnessfl_Intercept      2536
## FFD_Intercept                     3158
## FFD_cmean_4                       4022
##
## Family Specific Parameters:
##           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sigma_FFD     3.14      1.09     1.03     4.41 1.76        6       13
##
## Samples were drawn using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

```
bivar1.all.brm.nb<-brm(bf_fitness + bf_FFD, family = c(negbinomial, gaussian),
                       data = datadef,warmup = 1000,iter = 4000,chains=4,thin=2,
                         inits = "random",seed = 12345,cores = my.cores)
# No warnings, so maybe the way to go to account for overdispersion?
save(bivar1.all.brm.nb,
     file="output/bivar1.all.brm.nb.RData")
```

```
summary(bivar1.all.brm.nb)
```

Negative binomial distribution

```
##  Family: MV(negbinomial, gaussian)
##   Links: mu = log; shape = identity
##          mu = identity; sigma = identity
## Formula: round(mean_fitness_fl) ~ (1 | ID1 | id)
##          FFD ~ cmean_4 + (cmean_4 | ID1 | id) + (1 | year)
##    Data: datadef (Number of observations: 2478)
## Samples: 4 chains, each with iter = 4000; warmup = 1000; thin = 2;
##          total post-warmup samples = 6000
##
## Group-Level Effects:
## ~id (Number of levels: 837)
##                                               Estimate Est.Error l-95% CI
## sd(roundmeanfitnessfl_Intercept)                  1.37      0.05     1.28
## sd(FFD_Intercept)                                 1.64      0.15     1.36
## sd(FFD_cmean_4)                                   0.78      0.13     0.53
## cor(roundmeanfitnessfl_Intercept,FFD_Intercept)    -0.52      0.07    -0.66
## cor(roundmeanfitnessfl_Intercept,FFD_cmean_4)      -0.04      0.13    -0.31
## cor(FFD_Intercept,FFD_cmean_4)                    0.70      0.13     0.41
##                                               u-95% CI Rhat Bulk_ESS Tail_ESS
## sd(roundmeanfitnessfl_Intercept)                  1.48 1.00      940     2293
## sd(FFD_Intercept)                                 1.93 1.00     3306     5086
## sd(FFD_cmean_4)                                   1.05 1.00     2490     4202
## cor(roundmeanfitnessfl_Intercept,FFD_Intercept)    -0.38 1.00     4149     5242
## cor(roundmeanfitnessfl_Intercept,FFD_cmean_4)       0.22 1.00     5082     5502
## cor(FFD_Intercept,FFD_cmean_4)                    0.92 1.00     1930     4111
##
## ~year (Number of levels: 22)
##                   Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sd(FFD_Intercept)     5.08      0.84     3.74     7.00 1.00     2512     3598
##
## Population-Level Effects:
##                              Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS
## roundmeanfitnessfl_Intercept     0.76      0.06     0.65     0.87 1.00      823
## FFD_Intercept                   58.74      1.10    56.56    60.89 1.00     1495
## FFD_cmean_4                     -2.33      0.83    -3.98    -0.72 1.00     1420
##                              Tail_ESS
## roundmeanfitnessfl_Intercept     2116
## FFD_Intercept                    2661
## FFD_cmean_4                      2799
##
## Family Specific Parameters:
##                          Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS
## shape_roundmeanfitnessfl   669.89    185.35   378.93  1107.59 1.00     5266
## sigma_FFD                    4.35      0.07     4.21     4.49 1.00     3052
##                          Tail_ESS
## shape_roundmeanfitnessfl     5010
## sigma_FFD                    4355
##
## Samples were drawn using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

With shoot volume Tried only with negative binomial distribution so far.

```r
bf_fitness_shoot <- bf(round(mean_fitness_fl) ~  cn_shoot_vol_mean_sqrt +
                         (1|ID1|id)) # Set up model formula
```

```r
bivar2.all.brm.nb<-brm(bf_fitness_shoot + bf_FFD,
                       family = c(negbinomial, gaussian),
                       data = datadef,warmup = 1000,iter = 4000,chains=4,thin=2,
                         inits = "random",seed = 12345,cores = my.cores)
save(bivar2.all.brm.nb,
     file="output/bivar2.all.brm.nb.RData")
```

```r
summary(bivar2.all.brm.nb)
```

```
##  Family: MV(negbinomial, gaussian)
##   Links: mu = log; shape = identity
##          mu = identity; sigma = identity
## Formula: round(mean_fitness_fl) ~ cn_shoot_vol_mean_sqrt + (1 | ID1 | id)
##          FFD ~ cmean_4 + (cmean_4 | ID1 | id) + (1 | year)
##    Data: datadef (Number of observations: 2432)
## Samples: 4 chains, each with iter = 4000; warmup = 1000; thin = 2;
##          total post-warmup samples = 6000
##
## Group-Level Effects:
## ~id (Number of levels: 791)
##                                                   Estimate Est.Error l-95% CI
## sd(roundmeanfitnessfl_Intercept)                      1.24      0.05     1.15
## sd(FFD_Intercept)                                     1.66      0.15     1.37
## sd(FFD_cmean_4)                                       0.80      0.13     0.55
## cor(roundmeanfitnessfl_Intercept,FFD_Intercept)      -0.45      0.07    -0.59
## cor(roundmeanfitnessfl_Intercept,FFD_cmean_4)         0.16      0.14    -0.11
## cor(FFD_Intercept,FFD_cmean_4)                        0.64      0.14     0.35
##                                                   u-95% CI Rhat Bulk_ESS Tail_ESS
## sd(roundmeanfitnessfl_Intercept)                      1.34 1.00     1200     2558
## sd(FFD_Intercept)                                     1.95 1.00     3655     5039
## sd(FFD_cmean_4)                                       1.06 1.00     2799     4065
## cor(roundmeanfitnessfl_Intercept,FFD_Intercept)      -0.30 1.00     3749     4662
## cor(roundmeanfitnessfl_Intercept,FFD_cmean_4)         0.42 1.00     4236     5011
## cor(FFD_Intercept,FFD_cmean_4)                        0.87 1.00     2350     4213
##
## ~year (Number of levels: 22)
##                   Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sd(FFD_Intercept)     5.22      0.88     3.83     7.23 1.00     2190     3826
##
## Population-Level Effects:
##                                        Estimate Est.Error l-95% CI u-95% CI
## roundmeanfitnessfl_Intercept               0.79      0.05     0.69     0.89
## FFD_Intercept                             58.56      1.11    56.43    60.82
## roundmeanfitnessfl_cn_shoot_vol_mean_sqrt     0.04      0.00     0.03     0.05
## FFD_cmean_4                               -2.38      0.86    -4.10    -0.70
##                                        Rhat Bulk_ESS Tail_ESS
## roundmeanfitnessfl_Intercept           1.00     1192     2815
## FFD_Intercept                          1.00     1324     2051
## roundmeanfitnessfl_cn_shoot_vol_mean_sqrt 1.00      693     1538
## FFD_cmean_4                            1.00     1932     3225
```

```
##
## Family Specific Parameters:
##                          Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS
## shape_roundmeanfitnessfl   668.32    184.07   377.59  1087.96 1.00     4883
## sigma_FFD                    4.35      0.07     4.21     4.50 1.00     3594
##                          Tail_ESS
## shape_roundmeanfitnessfl     4626
## sigma_FFD                    4392
##
## Samples were drawn using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

Mean fitness per year of study

With no condition variable Tried only with negative binomial distribution so far.

```
bf_fitness_study <- bf(round(mean_fitness_study) ~  (1|ID1|id))
# Set up model formula
```

```
bivar3.all.brm.nb<-brm(bf_fitness_study + bf_FFD,
                       family = c(negbinomial, gaussian),
                       data = datadef,warmup = 1000,iter = 4000,chains=4,thin=2,
                         inits = "random",seed = 12345,cores = my.cores)
save(bivar3.all.brm.nb,
     file="output/bivar3.all.brm.nb.RData")
```

```
summary(bivar3.all.brm.nb)
```

```
##  Family: MV(negbinomial, gaussian)
##   Links: mu = log; shape = identity
##          mu = identity; sigma = identity
## Formula: round(mean_fitness_study) ~ (1 | ID1 | id)
##          FFD ~ cmean_4 + (cmean_4 | ID1 | id) + (1 | year)
##    Data: datadef (Number of observations: 2478)
## Samples: 4 chains, each with iter = 4000; warmup = 1000; thin = 2;
##          total post-warmup samples = 6000
##
## Group-Level Effects:
## ~id (Number of levels: 837)
##                                                   Estimate Est.Error l-95% CI
## sd(roundmeanfitnessstudy_Intercept)                   1.43      0.06     1.33
## sd(FFD_Intercept)                                     1.61      0.15     1.32
## sd(FFD_cmean_4)                                       0.78      0.13     0.52
## cor(roundmeanfitnessstudy_Intercept,FFD_Intercept)   -0.57      0.07    -0.71
## cor(roundmeanfitnessstudy_Intercept,FFD_cmean_4)     -0.18      0.13    -0.44
## cor(FFD_Intercept,FFD_cmean_4)                        0.74      0.13     0.46
##                                                   u-95% CI Rhat Bulk_ESS
## sd(roundmeanfitnessstudy_Intercept)                   1.55 1.00     1027
## sd(FFD_Intercept)                                     1.90 1.00     3429
## sd(FFD_cmean_4)                                       1.05 1.00     2965
## cor(roundmeanfitnessstudy_Intercept,FFD_Intercept)   -0.42 1.00     4118
## cor(roundmeanfitnessstudy_Intercept,FFD_cmean_4)       0.08 1.00     4622
## cor(FFD_Intercept,FFD_cmean_4)                          0.95 1.00     1968
##                                                     Tail_ESS
## sd(roundmeanfitnessstudy_Intercept)                     2084
## sd(FFD_Intercept)                                       4665
## sd(FFD_cmean_4)                                         4044
## cor(roundmeanfitnessstudy_Intercept,FFD_Intercept)      5066
## cor(roundmeanfitnessstudy_Intercept,FFD_cmean_4)        4871
## cor(FFD_Intercept,FFD_cmean_4)                          3762
##
## ~year (Number of levels: 22)
##                   Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sd(FFD_Intercept)     5.16      0.86     3.76     7.12 1.00     3019     4134
##
## Population-Level Effects:
##                                Estimate Est.Error l-95% CI u-95% CI Rhat
## roundmeanfitnessstudy_Intercept     0.13      0.06     0.00     0.25 1.00
## FFD_Intercept                      58.81      1.11    56.63    61.04 1.00
## FFD_cmean_4                        -2.33      0.83    -4.00    -0.71 1.00
##                                Bulk_ESS Tail_ESS
## roundmeanfitnessstudy_Intercept     1399     2985
## FFD_Intercept                       1732     2935
## FFD_cmean_4                         2238     3113
##
## Family Specific Parameters:
##                            Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS
## shape_roundmeanfitnessstudy   554.19    169.59   295.25   954.46 1.00     4679
## sigma_FFD                       4.36      0.07     4.22     4.50 1.00     3075
##                            Tail_ESS
## shape_roundmeanfitnessstudy     4569
## sigma_FFD                       3903
##
## Samples were drawn using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

With shoot volume Tried only with negative binomial distribution so far.

```
bf_fitness_study_shoot <- bf(round(mean_fitness_study) ~
                               cn_shoot_vol_mean_sqrt +
                          (1|ID1|id)) # Set up model formula
```

```
bivar4.all.brm.nb<-brm(bf_fitness_study_shoot + bf_FFD,
                       family = c(negbinomial, gaussian),
                       data = datadef,warmup = 1000,iter = 4000,chains=4,thin=2,
                         inits = "random",seed = 12345,cores = my.cores)
save(bivar4.all.brm.nb,
     file="output/bivar4.all.brm.nb.RData")
```

```
summary(bivar4.all.brm.nb)
```

```
##  Family: MV(negbinomial, gaussian)
##   Links: mu = log; shape = identity
##          mu = identity; sigma = identity
## Formula: round(mean_fitness_study) ~ cn_shoot_vol_mean_sqrt + (1 | ID1 | id)
##          FFD ~ cmean_4 + (cmean_4 | ID1 | id) + (1 | year)
##    Data: datadef (Number of observations: 2432)
## Samples: 4 chains, each with iter = 4000; warmup = 1000; thin = 2;
##          total post-warmup samples = 6000
##
## Group-Level Effects:
## ~id (Number of levels: 791)
##                                                  Estimate Est.Error l-95% CI
## sd(roundmeanfitnessstudy_Intercept)                  1.20      0.05     1.10
## sd(FFD_Intercept)                                    1.65      0.15     1.36
## sd(FFD_cmean_4)                                      0.80      0.13     0.56
## cor(roundmeanfitnessstudy_Intercept,FFD_Intercept)   -0.47      0.08    -0.61
## cor(roundmeanfitnessstudy_Intercept,FFD_cmean_4)      0.06      0.14    -0.22
## cor(FFD_Intercept,FFD_cmean_4)                        0.69      0.13     0.40
##                                                  u-95% CI Rhat Bulk_ESS
## sd(roundmeanfitnessstudy_Intercept)                  1.30 1.00     1615
## sd(FFD_Intercept)                                    1.94 1.00     3474
## sd(FFD_cmean_4)                                      1.06 1.00     2950
## cor(roundmeanfitnessstudy_Intercept,FFD_Intercept)   -0.31 1.00     3630
## cor(roundmeanfitnessstudy_Intercept,FFD_cmean_4)      0.33 1.00     4339
## cor(FFD_Intercept,FFD_cmean_4)                        0.91 1.00     2290
##                                                  Tail_ESS
## sd(roundmeanfitnessstudy_Intercept)                  2946
## sd(FFD_Intercept)                                    4455
## sd(FFD_cmean_4)                                      4170
## cor(roundmeanfitnessstudy_Intercept,FFD_Intercept)    4691
## cor(roundmeanfitnessstudy_Intercept,FFD_cmean_4)      4711
## cor(FFD_Intercept,FFD_cmean_4)                        3981
##
## ~year (Number of levels: 22)
##                   Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sd(FFD_Intercept)     5.25      0.86     3.87     7.23 1.00     2672     3580
##
## Population-Level Effects:
##                                            Estimate Est.Error l-95% CI
## roundmeanfitnessstudy_Intercept                0.13      0.06     0.02
## FFD_Intercept                                 58.63      1.11    56.48
## roundmeanfitnessstudy_cn_shoot_vol_mean_sqrt     0.05      0.00     0.05
## FFD_cmean_4                                   -2.42      0.85    -4.10
##                                            u-95% CI Rhat Bulk_ESS Tail_ESS
## roundmeanfitnessstudy_Intercept                0.24 1.00     2144     3687
## FFD_Intercept                                 60.84 1.01     1426     2668
## roundmeanfitnessstudy_cn_shoot_vol_mean_sqrt     0.06 1.00     1068     1986
## FFD_cmean_4                                   -0.74 1.00     1972     3276
##
## Family Specific Parameters:
##                           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS
## shape_roundmeanfitnessstudy   558.16    173.14   295.95   966.96 1.00     4631
## sigma_FFD                       4.36      0.07     4.21     4.50 1.00     3388
##                           Tail_ESS
## shape_roundmeanfitnessstudy     3998
## sigma_FFD                          4309
##
## Samples were drawn using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

Compare results of MCMCglmm and brms

I am not sure the code for this comparison is correct, the code needs checking! Therefore, I only performed the comparison for models with mean fitness per flowering event and no condition variable so far.

Mean fitness per flowering event, no condition variable

```r
kable(data.frame(summary(bivar1.all)$solutions)[1]) # Fixed effects MCMCglmm
```

	post.mean
variableFFD	58.8080138
variablefitness	0.8349578
at.level(variable, "FFD"):temp	-2.3498970

```r
kable(data.frame(summary(bivar1.all.brm.nb)$fixed)[1]) # Fixed effects brms
```

	Estimate
roundmeanfitnessfl_Intercept	0.7587717
FFD_Intercept	58.7411529
FFD_cmean_4	-2.3315906

```r
bivar1.all.brm.nb_asmcmc <- as.mcmc(bivar1.all.brm.nb, combine_chains = TRUE)
#head(bivar1.all.brm.nb_asmcmc) # check which column the parameters are in

bivar1.all.brm.nb_year <- (bivar1.all.brm.nb_asmcmc[,7]^2)
# sd_year__Intercept^2

bivar1.all.brm.nb_id_intercept_FFD <- (bivar1.all.brm.nb_asmcmc[,5]^2)
# sd(FFD_Intercept)^2 (individual intercept for FFD)

bivar1.all.brm.nb_cor_intercept_slope<-(bivar1.all.brm.nb_asmcmc[,10]^2)
# cor(FFD_Intercept,FFD_cmean_4)^2 (corr intercept for FFD - slope for FFD)

bivar1.all.brm.nb_cor_intercept_fitness<-(bivar1.all.brm.nb_asmcmc[,8]^2)
# cor(roundmeanfitnessfl_Intercept,FFD_Intercept)^2 (corr intercept FFD - fitness)

# cor(FFD_Intercept,FFD_cmean_4)^2 (corr intercept for FFD - slope for FFD) (rep)

bivar1.all.brm.nb_id_slope_FFD <- (bivar1.all.brm.nb_asmcmc[,6]^2)
# sd(FFD_cmean_4)^2 (individual slope for FFD)
```

```r
bivar1.all.brm.nb_cor_slope_fitness<-(bivar1.all.brm.nb_asmcmc[,9]^2)
# cor(roundmeanfitnessfl_Intercept,FFD_cmean_4)^2 (corr slope for FFD - fitness)

# cor(roundmeanfitnessfl_Intercept,FFD_Intercept)^2 (corr intercept for FFD - fitness) (rep)

# cor(roundmeanfitnessfl_Intercept,FFD_cmean_4)^2 (corr slope for FFD - fitness) (rep)

bivar1.all.brm.nb_intercept_fitness<-(bivar1.all.brm.nb_asmcmc[,4]^2)
# sd(roundmeanfitnessfl_Intercept)^2 (intercept for fitness)

bivar1.all.brm.nb_resid<-(bivar1.all.brm.nb_asmcmc[,12]^2)
# sigma_FFD^2 (residual)

compar_bivar1<-cbind(MCMCglmm=summary(bivar1.all$VCV)$statistics[,1],
      brms=as.vector(cbind(mean(bivar1.all.brm.nb_year),
                           mean(bivar1.all.brm.nb_id_intercept_FFD),
                           mean(bivar1.all.brm.nb_cor_intercept_slope),
                           -mean(bivar1.all.brm.nb_cor_intercept_fitness),
                           mean(bivar1.all.brm.nb_cor_intercept_slope),
                           mean(bivar1.all.brm.nb_id_slope_FFD),
                           -mean(bivar1.all.brm.nb_cor_slope_fitness),
                           -mean(bivar1.all.brm.nb_cor_intercept_fitness),
                           -mean(bivar1.all.brm.nb_cor_slope_fitness),
                           mean(bivar1.all.brm.nb_intercept_fitness),
                           mean(bivar1.all.brm.nb_resid))))
# Comparison random effects
compar_bivar1<-compar_bivar1[c(1:4,6:7,10:11),]
row.names(compar_bivar1)<-c("year_FFD",
                          "id_var_intercept_FFD",
                          "id_covar_intercept_slope_FFD",
                          "id_covar_fitness_intercept_FFD",
                          "id_var_slope_FFD",
                          "id_covar_fitness_slope_FFD",
                          "id_var_intercept_fitness",
                          "residual")
kable(compar_bivar1)
```

	MCMCglmm	brms
year_FFD	25.1950302	26.4739493
id_var_intercept_FFD	2.8337877	2.7146575
id_covar_intercept_slope_FFD	0.8240487	0.5029320
id_covar_fitness_intercept_FFD	-1.2196784	-0.2799562
id_var_slope_FFD	0.8262552	0.6338288
id_covar_fitness_slope_FFD	-0.0839816	-0.0196125
id_var_intercept_fitness	1.5083954	1.8869400
residual	18.6252929	18.9166192