Missing values

Facundo Muñoz

2017-01-10 breedR version: 0.11.5

Contents

The handling of missing values (i.e. `NA`) depends on *where* they are.

Missing response

It is perfectly valid to have missing vaules in the dependent variable. There is no need of removing those individuals from the dataset. Furthermore, including them will yield predictions for their phenotype, based on the predictive variables.

```r
library(breedR)

N <- 1e3
x <- rep(1:4, each = N/4)
dat <- data.frame(y = x + rnorm(N),
                  x = factor(letters[x]))
dat$y[1] <- NA
head(dat)
```

```
##            y x
## 1         NA a
## 2 0.69274277 a
## 3 0.09790199 a
## 4 1.62706874 a
## 5 2.12035503 a
## 6 3.12721355 a
```

```r
res <- remlf90(y ~ x, data = dat)

## The predicted phenotype for y[1] is the estimated effect
## of the corresponding level of x
fitted(res)[1] == fixef(res)$x['a', 'value']
```

```
##    1
## TRUE
```

Missing value for a fixed effect

This is not allowed, as it would yield an underdetermined system of equations. `breedR` issues a warning if missing values are detected.

```r
N <- 1e3
x <- rep(1:4, each = N/4)
dat <- data.frame(y = x + rnorm(N),
                  x = factor(letters[x]))
dat$x[c(1, 3, 5)] <- NA
head(dat)
```

```
##           y    x
## 1 0.3211924 <NA>
## 2 1.5743127    a
## 3 0.2954855 <NA>
## 4 0.4660159    a
## 5 1.7743846 <NA>
## 6 0.5243786    a
```

```
res <- remlf90(y ~ x, data = dat)
```

```
## Error in progsf90(mf, weights = weights, effects, opt = union("sol se", :
## Missing values in covariates are not allowed
## check individuals: 1, 3, 5
```

Idem for a regression variable.

```
N <- 1e3
x <- runif(N)
dat <- data.frame(y = 1 + 2*x + rnorm(N),
                  x = x)
dat$x[c(1, 3, 5)] <- NA
head(dat)
```

```
##            y         x
## 1  3.0572315        NA
## 2  2.8322030 0.5910449
## 3  1.8304948        NA
## 4  1.4058794 0.8450186
## 5 -0.9665408        NA
## 6  4.8346455 0.8146597
```

```
res <- remlf90(y ~ x, data = dat)
```

```
## Error in progsf90(mf, weights = weights, effects, opt = union("sol se", :
## Missing values in covariates are not allowed
## check individuals: 1, 3, 5
```

Missing value for a random effect

These **are** allowed. The incidence matrix will have a row of zeros for the corresponding individual.

```
N <- 1e3
N.blk <- 20
blk.effects <- rnorm(N.blk, sd = 2)
blk.idx <- sample(seq_len(N.blk), N, replace = TRUE)
dat <- data.frame(y = 1 + blk.effects[blk.idx] + rnorm(N),
                  blk = factor(blk.idx))
dat$blk[1] <- NA
head(dat)
```

```
##            y  blk
## 1  4.0317116 <NA>
## 2  0.8275709    1
## 3  1.4229472    6
## 4  1.2536024    8
## 5  1.7449430    6
## 6 -2.4041627    3
```

```
res <- remlf90(y ~ 1, random = ~ blk, data = dat)

sum(model.matrix(res)$blk[1,])
```

```
## [1] 0
```

As a consequence, the predicted phenotype will be based on the remaining available effects. In this case, the global mean.

```
fitted(res)[1] == fixef(res)$Intercept[1, 'value']
```

```
##    1
## TRUE
```

The spatial block effect is another way of writing the previous experiment. So it works in the same way.

```
coord <- expand.grid(row = 1:20, col = 1:50)
res <- remlf90(y ~ 1,
               spatial = list(model = 'blocks',
                              coord = coord,
                              id    = 'blk'),
               data = dat)

c(sum(model.matrix(res)$spatial[1,]) == 0,
fitted(res)[1] == fixef(res)$Intercept[1, 'value'])
```

```
##         1
## TRUE TRUE
```

However, the empirical residuals of the individuals with missing values of the random effects will have an increased variance. We can show that by replicating the previous experiment and computing the variance of the residual for the first observation.

```
resid_sample <- replicate(1e3, sample_first_residual())
var(resid_sample)
```

```
## [1] 3.40601
```

This can be important when fitting several random effects. See below.

Missing values in genetic effects

For an additive genetic effect, the relationship between individuals is given in the pedigree. It is legitimate not knowing the relatives for some individual. This is what happens with founders, for example.

Use `NA` for unknown relatives. If both are unknown (e.g. founders), the genetic effect (Breeding Value) will be predicted based on its phenotype, the other effects, and the estimated heritability.

```
dat <- breedR.sample.phenotype(
  fixed = c(mu = 10, x = 2),
  genetic = list(model     = 'add_animal',
                 Nparents = c(10, 10),
                 sigma2_a = 2,
                 check.factorial = FALSE),
  N = 1e3)
head(dat)
```

```
##   self sire dam X.mu       X.x          BV      resid phenotype
## 1    1   NA  NA    1 0.4231002  1.57893022 -0.2718524 12.153278
## 2    2   NA  NA    1 0.8969499 -0.05520568 -1.8363034  9.902391
## 3    3   NA  NA    1 0.3827669  0.03921312 -0.9225210  9.882226
## 4    4   NA  NA    1 0.2923960  1.70309765 -1.4938989 10.793991
## 5    5   NA  NA    1 0.3696142 -2.72022917 -0.4718799  7.547119
## 6    6   NA  NA    1 0.4158540 -0.45599038  0.1535756 10.529293
```

```
res <- remlf90(phenotype ~ 1 + X.x,
               genetic = list(model = 'add_animal',
                              pedigree = dat[, 1:3],
                              id    = 'self'),
               data = dat)

str(ranef(res)$genetic)
```

```
##  atomic [1:1020] 1.433 -0.397 0.273 1.739 -2.741 ...
##  - attr(*, "se")= num [1:1020] 0.464 0.456 0.469 0.447 0.456 ...
```

Important issue Having random effects with missing values in **combination** with genetic models, can yield spurious predictions of Breeding Values. This is due to the higher variability of the residual term, for the individuals with missing values in random effects.

Missing values in coordinates of spatial effects

Are allowed. Just like in any other random effect. For those cases, the spatial component will not participate in the prediction.

```
dat <- breedR.sample.phenotype(
  fixed = c(mu = 10, x = 2),
  spatial = list(model     = 'AR',
                 grid.size = c(10, 5),
                 rho       = c(.2, .8),
                 sigma2_s  = 1)
)
dat$Var1[1] <- NA
head(dat)
```

```
##   X.mu       X.x Var1 Var2    spatial       resid phenotype
## 1    1 0.6082442   NA    3  0.1186114  0.28370787 11.618808
## 2    1 0.3189472    2    1 -0.2659814  0.22200293 10.593916
## 3    1 0.5657411    6    5 -0.3708866  1.00097936 11.761575
## 4    1 0.6163408    9    1 -0.1719042  1.51265041 12.573428
## 5    1 0.4638350    6    3 -0.9863858 -0.30259924  9.638685
## 6    1 0.6305551    8    2  0.9520079 -0.02759329 12.185525
```

```
res <- remlf90(phenotype ~ 1 + X.x,
               spatial = list(model = 'AR',
                              coord = dat[, c('Var1', 'Var2')],
                              rho   = c(0.2, 0.8)),
               data = dat)

sum(model.matrix(res)$spatial[1,])
```

```
## [1] 0
```