Model Customisation in missingHE

For each of the three types of models that can be fitted using `missingHE`, namely **selection**, **pattern mixture**, and **hurdle** models, the package provides a series of customisation options to allow a flexible specification of the models in terms of modelling assumptions and prior choices. These can be extremely useful for handling the typical features of trial-based CEA data, such as non-normality, clustering, and type of missingness mechanism. This tutorial shows how it is possible to customise different aspects of the models using the arguments of each type of function in the package. Throughout, we will use the built-in dataset called `MenSS` as a toy example, which is directly available when installing and loading `missingHE` in your `R` workspace. See the vignette called *Introduction to missingHE* for an introductory tutorial of each function in `missingHE` and a general presentation of the data from the `MenSS` dataset.

If you would like to have more information on the package, or would like to point out potential issues in the current version, feel free to contact the maintainer at ucakgab@ucl.ac.uk. Suggestions on how to improve the package are also very welcome.

Changing the distributional assumptions

A general concern when analysing trial-based CEA data is that, in many cases, both effectiveness and costs are characterised by highly skewed distributions, which may cause standard normality modelling assumptions to be difficult to justify, especially for small samples. `missingHE` allows to chose among a range of parametric distributions for modelling both outcome variables, which were selected based on the most common choices in standard practice and the literature.

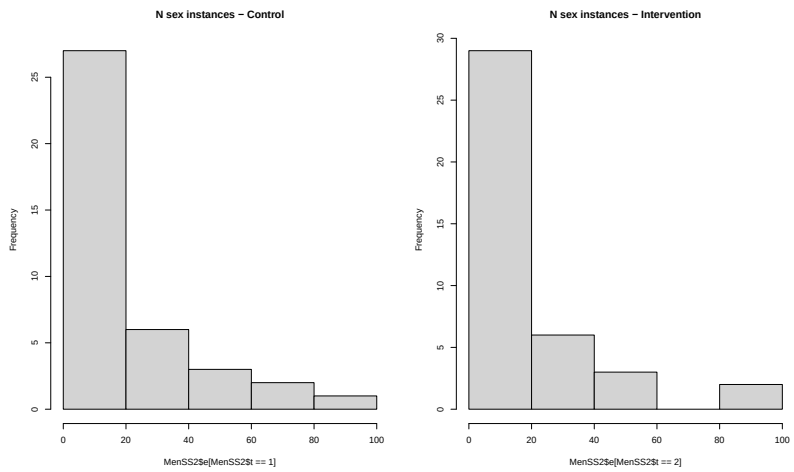
In each model, the specific type of distributions for the effectiveness (e) and cost (c) outcome can be selected by setting the arguments `dist_e` and `dist_c` to specific character names. Available choices include: Normal (`"norm"`) and Beta (`"beta"`) distributions for e and Normal (`"norm"`) and Gamma (`"gamma"`) for c. Distributions for modelling both discrete and binary effectiveness variables are also available, such as Poisson (`"pois"`) and Bernoulli (`bern`) distributions. The full list of available distributions for each type of outcome can be seen by using the `help` function on each function of the package.

In the `MenSS` dataset the default effectiveness variables are the QALYs. However, in general, other types of effectiveness measures may be of interest in the economic analysis (e.g. the primary outcome from a trial). In our dataset we have the number of instances of unprotected sex at 12 months follow-up, denoted as `sex_inst`, which could be used in the CEA instead of QALYs. Thus, we create a second dataset called `MenSS2`, where we assign the name e to the variable `sex_inst`, which allows `missingHE` to identify this variable as the main effectiveness variable for the analysis. This can be done by typing

```
> MenSS2 <- MenSS
> MenSS2$e <- MenSS$sex_inst
>
> #first 10 entries of e
> head(MenSS2$e, n = 10)
 [1] NA 50  3  0 99 NA 20 NA NA NA
```

The new effectiveness outcome is now a discrete variable and therefore the use of discrete distribtuions is likely to be more appropriate for modelling purposes compared to standard normality assumptions. We can check the empirical histograms of e by treatment group by typing

```
> par(mfrow=c(1,2))
> hist(MenSS2$e[MenSS2$t==1], main = "N sex instances - Control")
> hist(MenSS2$e[MenSS2$t==2], main = "N sex instances - Intervention")
```



We can also see that the proportion of missing values in e is considerably large in both treatment groups.

```
> #proportions of missing values in the control group
> sum(is.na(MenSS2$e[MenSS$t==1])) / length(MenSS2$e[MenSS$t==1])
[1] 0.48

>
> #proportions of missing values in the intervention group
> sum(is.na(MenSS2$e[MenSS$t==2])) / length(MenSS2$e[MenSS$t==2])
[1] 0.5238095
```

As an example, we fit a selection model assuming Poisson distributions to handle the discrete nature of e, and we choose Gamma distributions to capture the skewness in the costs. We note that, in case some of individuals have costs that are equal to zero (as in the `MenSS` dataset), standard parametric distributions with a positive support are not typically defined at 0 (e.g. the Gamma distributions), making their implementation impossible. Thus, in these cases, it is necessary to use a trick to modify the boundary values before fitting the model. A common approach is to add a small constant to the cost variables. These can be done by typing

```
>  MenSS2$c <- MenSS2$c + 0.01
```

We note that, although simple, this strategy has the potential drawback that results may be affected by the choice of the constant added and therefore sensitivity analysis to the value used is typically recommended. `missingHE` provides an alternative way to deal with this issue by means of fitting a two-part regression or *hurdle* model which does not require the use of any constant. For more information on hurdle models, type `help(hurdle)`.

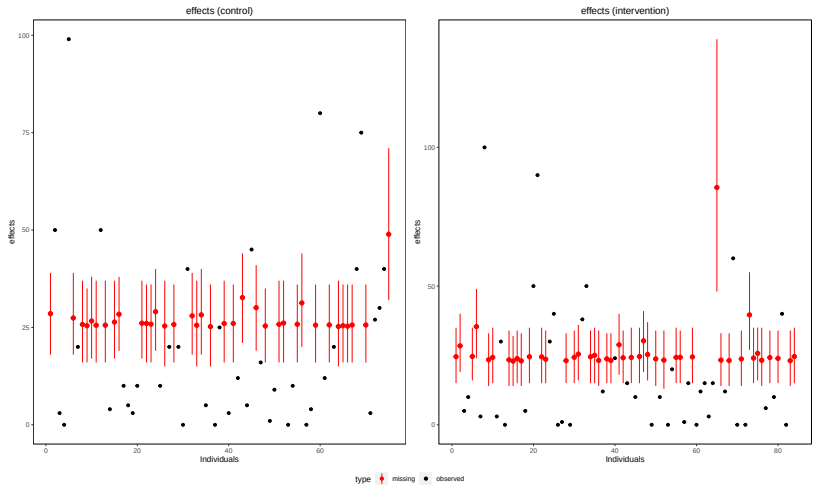
We are now ready to fit our selection model to the `MenSS2` dataset using the following command

```
> PG.sel=selection(data = MenSS2, model.eff = e ~ sex_inst.0, model.cost = c ~ 1,
+   model.me = me ~ age + ethnicity + employment,
+   model.mc = mc ~ age + ethnicity + employment, type = "MAR",
+   n.iter = 1000, dist_e = "pois", dist_c = "gamma")
```

The arguments `dist_e = "pois"` and `dist_c = "gamma"` specify the distributions assumed for the outcome variables and, in the model of e, we also adjust for the baseline outcome values (`sex_inst.0`). According to the type of distributions chosen, `missingHE` automatically models the dependence between covariates and the mean outcome on a specific scale to reduce the chance of incurring into numerical problems due to the constraints of the distributions. For example, for both Poisson and Gamma distributions means are modelled on the log scale, while for Beta and Bernoulli distirbutions they are modelled on the logit scale. To see the modelling scale used by `missingHE` according to the type of distribution selected, use the `help` command on each function of the package.

The model assumes MAR conditional on `age`, `ethnicity` and `employment` as auxiliary variables for predicting missingness in both outcomes. We can look at how the model generate imputations for the outcomes by treatment group using the generic function `plot`. For example, we can look at how the missing e are imputed by typing

```
> plot(PG.sel, outcome = "effects")
```



Summary results of our model from a statistical perspective can be inspected using the command `coef`, which extracts the estimates of the mean regression coefficients for e and c by treatment group. By default, the lower and upper bounds provide the 95% credibile intervals for each estimate (based on the 0.025 and 0.975 quantiles of the posterior distribution). However, it is possible to modify these values using the argument `prob` to change the level of the intervals to match the one desired. For example, if we want 90% intervals, we can type

```
> coef(PG.sel, prob = c(0.05, 0.95))
$Comparator
$Comparator$Effects
             mean    sd lower upper
(Intercept) 3.235 0.046 3.163 3.313
sex_inst.0  0.004 0.001 0.002 0.006

$Comparator$Costs
             mean   sd lower upper
(Intercept) 5.518 0.35 4.973 6.111


$Reference
$Reference$Effects
             mean    sd lower upper
(Intercept) 3.140 0.056 3.051 3.232
sex_inst.0  0.011 0.002 0.007 0.014

$Reference$Costs
             mean    sd lower upper
(Intercept) 5.469 0.397 4.836 6.134
```

The entire posterior distribution for each parameter of the model can also be extracted from the output of the model by typing `PG.sel$model_output`, which returns a list object containing the posterior estimates for each model parameter. An overall summary of the economic analysis based on the model estimates can be obtained using the `summary` command

```
> summary(PG.sel)

 Cost-effectiveness analysis summary

 Comparator intervention: intervention 1
 Reference intervention: intervention 2

 Parameter estimates under MAR assumption

 Comparator intervention
                         mean     sd     LB      UB
mean effects (t = 1)  27.245  1.014 25.681  29.083
mean costs (t = 1)   264.791 95.429 144.42 450.583

 Reference intervention
                        mean      sd      LB      UB
mean effects (t = 2) 25.896   1.082  24.158   27.76
mean costs (t = 2)   256.95 106.687 125.994 461.364

 Incremental results
                mean      sd       LB      UB
delta effects -1.349   1.466     -3.7   1.091
delta costs   -7.841 139.493 -245.912 215.675
ICER           5.812
```

which shows summary statistics for the mean effectiveness and costs in each treatment group, for the mean differentials and the estimate of the ICER.

Including random effects terms

For each type of model, `missingHE` allows the inclusion of random effects terms to handle clustered data. To be precise, the term *random effects* does not have much meaning within a Bayesian approach since all parameters are in fact random variables. However, this terminology is quite useful to explain the structure of the model.

We show how random effects can be added to the model of e and c within a pattern mixture model fitted to the `MenSS2` dataset using the function `pattern`. The clustering variable over which the random effects are specified is the factor variable `site`, representing the centres at which data were collected in the trial. Using the same distributional assumptions of the selection model, we fit the pattern mixture model by typing

```
> PG.pat=pattern(data = MenSS2, model.eff = e ~ sex_inst.0 + (1 + sex_inst.0| site),
+                model.cost = c ~ 1 + (1 | site), type = "MAR", restriction = "AC",
+                n.iter = 1000, Delta_e = 0, Delta_c = 0, dist_e = "pois", dist_c = "gamma")
```

The function fits a random intercept and slope model for e, as indicated by the notation `(1 + sex_inst.0 | site)`, and a random intercept only model for c, as indicated by the notation `(1 | site)`. In both models, `site` is the clustering variable over which the random coefficients are estimated. `missingHE` allows the user to choose among different clustering variables for the model of e and c if these are available in the dataset. It is also possible to specify random slope only models, for example in the model of e by using the notation `(0 + sex_inst.0 | site)` where `0` indicates the removal of the random intercept. The same notation can be applied when using the `selection` and `hurdle` functions inside `missingHE`, with the addition that for these models random effects can also be specified for the missingness and structural value mechanisms. Use the `help` command to obtain more information on how random effects can be specified for each type of model.

Coefficient estimates for the random effects can be extracted using the `coef` function and setting the argument `random = TRUE` (if set to `FALSE` then the fixed effects estimates are displayed).

```
> coef(PG.pat, random = TRUE)
$Comparator
$Comparator$Effects
                mean     sd   lower  upper
(Intercept) 1  7.379 28.796 -35.742 98.509
sex_inst.0 1  -0.016 25.008 -26.716 26.969
(Intercept) 2  7.840 28.800 -35.229 98.842
sex_inst.0 2   0.008 25.008 -26.706 26.982
(Intercept) 3  5.342 28.780 -38.049 96.270
sex_inst.0 3   0.067 25.009 -26.649 27.048

$Comparator$Costs
               mean    sd lower upper
(Intercept) 1 4.550 0.811 2.662 5.640
(Intercept) 2 4.511 0.807 2.593 5.595
(Intercept) 3 4.564 0.796 2.690 5.652


$Reference
$Reference$Effects
                 mean     sd   lower   upper
(Intercept) 1 -22.991 30.736 -72.593  40.957
sex_inst.0 1  -21.036  4.710 -29.177 -15.624
(Intercept) 2 -23.047 30.739 -72.732  40.860
sex_inst.0 2  -21.062  4.711 -29.205 -15.649
(Intercept) 3 -21.754 30.738 -71.557  42.282
sex_inst.0 3  -21.109  4.710 -29.252 -15.697

$Reference$Costs
               mean    sd lower  upper
(Intercept) 1 8.046 1.585 5.599 11.086
(Intercept) 2 7.853 1.600 5.433 10.909
(Intercept) 3 8.117 1.579 5.619 11.100
```

For both e and c models, summary statistics for the random coefficient estimates are displayed for each treatment group and each of the 3 clusters in `site`.

Changing the priors

By default, all models in `missingHE` are fitted using vague prior distributions so that posterior results are essentially derived based on the information from the observed data alone. This ensures a rough approximation to results obtained under a frequentist approach based on the same type of models.

However, in some cases, it may be reasonable to use more informative priors to ensure a better stability of the posterior estimates by restricting the range over which estimates can be obtained. For example if, based on previous evidence or knowledge, the range over which a specific parameter is likely to vary is known, then priors can be specified so to give less weight to values outside that range when deriving the posterior estimates. However, unless the user is familiar with the choice of informative priors, it is generally recommended not to change the default priors of `missingHE` as the unintended use of informative priors may substantially drive posterior estimates and lead to incorrect results.

For each type of model in `missingHE`, priors can be modified using the argument `prior`, which allows to change the hyperprior values for each model parameter. The interpretation of the prior values change according to the type of parameter and model considered. For example, we can fit a hurdle model using `hurdle` to the `MenSS2` dataset using more informative priors on some parameters.

Prior values can be modified by first creating a list object which, for example, we call `my.prior`. Within this list, we create a number of elements (vectors of length two) which should be assigned specific names based on the type of parameters which priors we want to change.

```
> my.prior <- list(
+   "alpha0.prior" = c(0 , 0.0000001),
+   "alpha.prior" = c(0, 0.0000001),
+   "beta0.prior" = c(0, 0.0000001),
+   "gamma0.prior.c"= c(0, 1),
+   "gamma.prior.c" = c(0, 0.01),
+   "mu.b0.prior" = c(0, 0.001),
+   "mu.g0.prior.c"= c(0, 0.001),
+   "s.b0.prior" = c(0, 100),
+   "s.g0.prior.c"= c(0, 100),
+   "sigma.prior.c" = c(0, 10000)
+ )
```

The names above have the following interpretations in terms of the model parameters:

- `"alpha0.prior"` is the intercept of the model of e. The first and second elements inside the vector for this parameter are the mean and precision (inverse of variance) that should be used for the normal prior given to this parameter by `missingHE`.
- `"alpha.prior"` are the regression coefficients (exclusing the intercept) of the model of e. The first and second elements inside the vector for this parameter are the mean and precision (inverse of variance) that should be used for the normal priors given to each coefficient by `missingHE`.
- `"beta0.prior"` is the intercept of the model of c. The first and second elements inside the vector for this parameter are the mean and precision (inverse of variance) that should be used for the normal prior given to this parameter by `missingHE`.
- `"gamma0.prior.c"` is the intercept of the model of sc. The first and second elements inside the vector for this parameter are the mean and precision (inverse of variance) that should be used for the logistic prior given to this parameter by `missingHE`.
- `"gamma.prior.c"` are the regression coefficients (exclusing the intercept) of the model of sc. The first and second elements inside the vector for this parameter are the mean and precision (inverse of variance) that should be used for the normal priors given to each coefficient by `missingHE`.
- `"mu.b0.prior"` is the mean of the random intercept of the model of c. The first and second elements inside the vector for this parameter are the mean and precision (inverse of variance) that should be used for the normal prior given to this parameter by `missingHE`.
- `"s.b0.prior"` is the standard deviation of the random intercept of the model of c. The first and second elements inside the vector for this parameter are the lower and upper bounds that should be used for the uniform prior given to this parameter by `missingHE`.
- `"mu.g0.prior"` is the mean of the random intercept of the model of sc. The first and second elements inside the vector for this parameter are the mean and precision (inverse of variance) that should be used for the normal prior given to this parameter by `missingHE`.
- `"s.g0.prior"` is the standard deviation of the random intercept of the model of sc. The first and second elements inside the vector for this parameter are the lower and upper bounds that should be used for the uniform prior given to this parameter by `missingHE`.
- `"sigma.prior.c"` is the standard deviation of the model of c. The first and second elements inside the vector for this parameter are the lower and upper bounds that should be used for the uniform prior given to this parameter by `missingHE`.

The values shown above are the default values set in `missingHE` for each of these parameters. It is possible to change the priors by providing different values, for example by increasing the precision for some of the coefficient estimates or descreasing the upper bound for standard deviation parameters. Different names should be used to indicate for which parameter the prior should be modified, keeping in mind that the full list of names that can be used varies depending on the type of models and modelling assumptions specified. The full list of parameter names for each type of model can be assessed using the `help`command on each function of `missingHE`.

We can now fit the hurdle model using our priors by typing

```
> #remove added constant from costs
> MenSS2$c <- MenSS2$c - 0.01
>
> PG.hur=hurdle(data = MenSS2, model.eff = e ~ sex_inst.0, model.cost = c ~ 1 + (1 | site),
+   model.se = se ~ 1, model.sc = sc ~ age + (1 | site), type = "SAR", se = NULL, sc = 0,
+   n.iter = 1000, dist_e = "pois", dist_c = "gamma", prior = my.prior)
```

Notice that, before fitting the model, we have removed the constant that we previously added to the costs since hurdle models can handle zero costs by specifying the structural value using the argument `sc = 0` inside `hurdle`. In this case, we do not require handling any structural value in e and we pass this information to the function by setting `se = NULL`. Finally, we can inspect the statistical results from the model by typing

```
> coef(PG.hur, random = FALSE)
$Comparator
$Comparator$Effects
             mean    sd lower upper
(Intercept) 3.253 0.048 3.162 3.351
sex_inst.0  0.004 0.001 0.002 0.006

$Comparator$Costs
            mean   sd lower upper
(Intercept) 0.33 1.97 -2.89 2.864


$Reference
$Reference$Effects
             mean    sd lower upper
(Intercept) 3.192 0.057 3.084 3.304
sex_inst.0  0.009 0.002 0.005 0.014

$Reference$Costs
             mean    sd  lower upper
(Intercept) -0.13 2.797 -5.056 4.935
```

and

```
> coef(PG.hur, random = TRUE)
$Comparator
$Comparator$Effects
NULL

$Comparator$Costs
               mean    sd lower upper
(Intercept) 1 5.296 1.972 2.633 8.529
(Intercept) 2 5.514 1.918 2.989 8.652
(Intercept) 3 5.376 1.947 2.746 8.540
```

```
$Reference
$Reference$Effects
NULL

$Reference$Costs
               mean    sd lower  upper
(Intercept) 1 5.658 2.808 0.503 10.815
(Intercept) 2 5.852 2.827 0.672 10.950
(Intercept) 3 5.598 2.811 0.506 10.857
```