Discrete Variables Copula

Robin J. Evans

2024-03-01

Incorporating discrete variables

We can also fit models in which some of the covariates and/or outcomes are binary. For example, suppose we assume that

$$Z_1 \sim N(0, \sigma_1^2) \qquad\qquad Z_2 \sim \text{Bernoulli}(p_2) \qquad\qquad Y \mid do(X = x) \sim N(0, \sigma_y^2)$$

and $X \mid Z_1 = z_1, Z_2 = z_2 \sim \text{Bernoulli}(q)$, and we assume a Gaussian copula over Z_1, Z_2, Y with correlation R, where $\rho_{Z_1 Z_2} = 0.5$, $\rho_{Z_1 Y} = 0.3$ and $\rho_{Z_2 Y} = 0.4$.

Then we can set up this model in the usual way:

```
forms <- list(list(Z1 ~ 1, Z2 ~ 1), X ~ Z1*Z2, Y ~ X, ~ 1)
fams <- list(c(1,5), 5, 1, 1)
pars <- list(Z1 = list(beta=0, phi=1),
             Z2 = list(beta=0),
             X = list(beta=c(-0.3,0.1,0.2,0)),
             Y = list(beta=c(-0.25, 0.5), phi=0.5),
             cop = list(beta=matrix(c(0.5,0.3,0.4), nrow=1)))
```

Now call `rfrugalParam` as usual.

```
set.seed(1234)
n <- 1e4
dat <- rfrugalParam(n, formulas = forms, family = fams, pars = pars)
```

Then we can check that the distributions follow the specification that we asked for.

```
ks.test(dat$Z1, pnorm)
```

```
##
##  Asymptotic one-sample Kolmogorov-Smirnov test
##
## data:  dat$Z1
## D = 0.01, p-value = 0.2
## alternative hypothesis: two-sided
```

```
binom.test(sum(dat$Z2), n=n, p = 0.5)
```

```
##
##  Exact binomial test
##
## data:  sum(dat$Z2) and n
## number of successes = 5099, number of trials = 10000, p-value = 0.05
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
##  0.50 0.52
## sample estimates:
## probability of success
##                   0.51
```

```r
glmX <- glm(X ~ Z1*Z2, family=binomial, data=dat)
summary(glmX)$coefficients
```

```
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.30184     0.0289 -10.433 1.75e-25
## Z1           0.08907     0.0293   3.040 2.37e-03
## Z2           0.15937     0.0403   3.951 7.79e-05
## Z1:Z2        0.00486     0.0409   0.119 9.05e-01
```

These are all consistent with the values we provided. We can finally check the outcome model.

```r
ps <- predict(glmX, type="response")
wt <- dat$X/ps + (1-dat$X)/(1-ps)
glmY <- svyglm(Y ~ X, design = svydesign(~1, weights=wt, data=dat))
summary(glmY)$coefficients
```

```
##             Estimate Std. Error t value  Pr(>|t|)
## (Intercept)   -0.246    0.00946   -26.0 9.70e-145
## X              0.494    0.01398    35.3 1.82e-257
```

Categorical and Ordinal Variables

We can also extend this to variables that are categorical or ordered categorical. These options correspond to the family indicators 11 and 10 respectively, or they can be accessed by passing the strings `"categorical"` and `"ordinal"` to the `family` argument. Both types of variable are stored as a factor, with labels for the objects `1`, `2`, up to ℓ, where ℓ is the number of levels.

These methods respectively use a contrast with the baseline level, or a contrast with the category immediately below the current one in the ordering. For example, if we have a three variable categorical variable using the formula `Z ~ A` then the parameters are of the form:

$$\log \frac{P(Z=i)}{P(Z=1)} = \beta_{0i} + \beta_{ai} a, \qquad\qquad i = 2, \dots, \ell.$$

For a categorical variable the contrasts become $\log\{P(Z=i)/P(Z=i-1)\}$ for the same range of i.

The regression parameters can be passed either as a vector or a matrix, but in either case they must be ordered so that all the coefficients for one contrast precede all those for another. For example, if we want we could put `Z = list(beta = c(0.5,0.1,-0.5,0.4), nlevel = 3)` in the `pars` argument to require that

$$\log \frac{P(Z=2)}{P(Z=1)} = 0.5 + 0.1a \qquad\qquad \log \frac{P(Z=3)}{P(Z=1)} = -0.5 + 0.4a$$

for an unordered categorical variable, and the same for an ordinal variable but replacing the second quantity with $\log\{P(Z=3)/P(Z=2)\}$. Equivalently, we could put `Z = list(beta = matrix(c(0.5,0.1,-0.5,0.4), nrow=2), nlevel = 3)`, and the same model would be fitted.

```r
forms <- list(list(Z0 ~ A0, Z1 ~ A0),
              list(A0 ~ 1, A1 ~ A0*Z0),
              Y ~ A0*A1,
              ~ A0)
fams <- list(c("categorical","categorical"),c(5,5),1,1)
pars <- list(A0 = list(beta = 0),
             Z0 = list(beta = c(0.3,-0.2,0.4,0.1), nlevel=3),
             Z1 = list(beta = c(0.3,-0.2,0.4,0.1), nlevel=3),
             A1 = list(beta = 2*c(-0.3,0.4,0.3,0.5,0.3,0.5)),
             Y = list(beta = c(-0.5,0.2,0.3,0), phi=1),
             cop = list(beta=c(2,0.5)))
```

In this case we set both covariate variables to be categorical.

```
set.seed(123)
n <- 5e4
dat <- rfrugalParam(n, formulas = forms, pars=pars, family = fams)
```

Now we can check that the parameters being simulated match those input.

```
glm(I(Z0==2) ~ A0, family=binomial, data=dat[dat$Z0 != 3,])$coefficients
```

```
## (Intercept)          A0
##       0.307      -0.185
```

```
glm(I(Z0==3) ~ A0, family=binomial, data=dat[dat$Z0 != 2,])$coefficients
```

```
## (Intercept)          A0
##      0.4175      0.0963
```

```
glm(I(Z1==2) ~ A0, family=binomial, data=dat[dat$Z1 != 3,])$coefficients
```

```
## (Intercept)          A0
##       0.304      -0.210
```

```
glm(I(Z1==3) ~ A0, family=binomial, data=dat[dat$Z1 != 2,])$coefficients
```

```
## (Intercept)          A0
##      0.4307      0.0643
```

Finally, we can check that the outcome model works as it should.

```
ps <- predict(glm(A1 ~ A0*Z0, family=binomial, data=dat), type="response")
wt <- dat$A1/ps + (1-dat$A1)/(1-ps)
summary(svyglm(Y ~ A0*A1, design = svydesign(~1,data=dat,weights=wt)))$coef
```

```
##             Estimate Std. Error t value  Pr(>|t|)
## (Intercept)  -0.4884     0.0091   -53.6  0.00e+00
## A0            0.2201     0.0186    11.8  2.91e-32
## A1            0.3018     0.0129    23.4 6.22e-121
## A0:A1        -0.0374     0.0220    -1.7  8.89e-02
```

Comparing with a naïve (unweighted) estimate.

```
summary(svyglm(Y ~ A0*A1, design = svydesign(~1,data=dat,weights=~1)))$coef
```

```
##             Estimate Std. Error t value  Pr(>|t|)
## (Intercept)   -0.578    0.00894  -64.63  0.00e+00
## A0             0.048    0.01621    2.96  3.08e-03
## A1             0.480    0.01252   38.34 4.67e-317
## A0:A1          0.118    0.01978    5.96  2.59e-09
```