Introduction

July 31, 2018

1 Introduction

The package is well suited to estimate generalized linear models with a linear predictor of the following form:

$$\boldsymbol{\eta} = \mathbf{Z}\boldsymbol{\gamma} = \mathbf{D}\boldsymbol{\alpha} + \mathbf{X}\boldsymbol{\beta} = \sum_{k=1}^{K} \mathbf{D}_k \boldsymbol{\alpha}_k + \mathbf{X}\boldsymbol{\beta} \, ,$$

where the matrix **D** arises from dummy encoding of K high-dimensional categorical variables and **X** contains the variables of interst. We refer to $\boldsymbol{\beta}$ as the structural parameters whereas $\boldsymbol{\alpha}$ are the so called fixed effects.

Brute force estimation of this kind of models is often restricted to computational limitations. We tackle this problem by providing a fast and memory efficient algorithm based on the combination of the Frisch-Waugh-Lovell theorem and the method of alternating projections (Stammann 2018). We restrict ourselves to non-linear models since Gaure (2013) already offers a great package for linear models.

2 Exactness

To show how to use the package we start with generating an artificial data set of a two-way fixed effects logit model. The data generating process is as follows:

$$y_{it} = 1[\mathbf{x}_{it}' \boldsymbol{\beta} + \alpha_i + \gamma_t + \epsilon_{it} > 0] \, ,$$

where $\mathbf{x}_{it}$ is generated as iid. $\mathcal{N} \sim (0_3, \mathbf{I}_3)$ and ϵ_{it} is an iid. logistic error term with location zero and scale one. α_i and γ_t are generated as iid. standard normal and $\boldsymbol{\beta} = [1, -1, 1]'$. The function `simGLM(n, t, seed, "logit")` constructs an artificial data set for arbitrary n and t.

To compare `feglm()` to the standard `glm()` we generate a data set with $n = 200$ and $t = 100$ and extract the estimates of the structural parameters. For `feglm()` this can be done using the generic function `coef()`.

```
# Generate artificial data set
n <- 200L
t <- 100L
data <- simGLM(n, t, 1805L, "logit")

# Use 'feglm()' and 'glm()'
mod.alpaca <- feglm(y ~ x1 + x2 + x3 | i + t, data, family = binomial())
mod.glm <- glm(y ~ x1 + x2 + x3 + factor(i) + factor(t) + 0, data, family = binomial())

# Compare estimates of structural parameters
beta.mat <- cbind(coef(mod.alpaca), coef(mod.glm)[seq(3L)])
colnames(beta.mat) <- c("feglm", "glm")
beta.mat
```

```
        feglm        glm
x1  0.9936333  0.9936333
x2 -1.0028880 -1.0028880
x3  1.0224925  1.0224925

# Show 'summary()'
summary(mod.alpaca)

binomial

y ~ x1 + x2 + x3 | i + t

l= [200, 100], n= 20000, deviance= 16876.61

Structural parameter(s):

   Estimate Std. error z value Pr(> |z|)
x1  0.99363    0.02229   44.58    <2e-16 ***
x2 -1.00289    0.02248  -44.62    <2e-16 ***
x3  1.02249    0.02263   45.19    <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Other generic functions already implemented are: `fitted()`, `predict()`, `summary()`, and `vcov()`. The standard errors are based on the inverse of the negative Hessian after convergence. The estimates of `feglm()` and `glm()` are identical (Stammann 2018).

Now we want to have a look at the estimates of the fixed effects. By default `glm()` drops the first level of the second category due to perfect collinearity. Thus this level becomes our reference group which means that all remaining coefficients have to be interpreted as difference to the reference group.

The function `getFEs()` computes the estimated fixed effects from an object returned by `feglm()`. However the underlying routine does not drop any level while solving the system of equations. Thus an estimable function has to be applied to our solution to get meaningful estimates of the fixed effects. See Gaure (2013) for an extensive treatment of this issue.

In order to reproduce the coefficients of `glm()` we substract the coefficient of the reference group from all remaining coefficients of the second category and add it to all coefficents of the first category.

```
# Rank deficient solution
alpha.rd <- getFEs(mod.alpaca)

# Apply estimable function. Use First level of second fixed effects category
# as reference
alpha <- alpha.rd[-(n + 1L)]
alpha.ref <- alpha.rd[n + 1L]
alpha <- c(alpha[seq(n)] + alpha.ref, alpha[seq(n + 1L, n + t - 1L)] - alpha.ref)

# Compare estimates of fixed effects
alpha.mat <- cbind(alpha, coef(mod.glm)[-seq(3L)])
colnames(alpha.mat) <- c("feglm", "glm")
head(alpha.mat)

       feglm        glm
1 -0.9056083 -0.9056083
2 -1.4685214 -1.4685214
3 -1.3228554 -1.3228555
```

```
4 -1.3187228 -1.3187228
5  1.7649990  1.7649990
6 -1.3210217 -1.3210217
```

```
tail(alpha.mat)
```

```
         feglm        glm
95   0.3341468  0.3341468
96   0.2056819  0.2056818
97  -0.2959511 -0.2959511
98   1.7303832  1.7303832
99   0.2154531  0.2154531
100  0.9322536  0.9322536
```

3 Computation Time

Taking the simulated data set from the previous section and measuring the computation time shows the advantage of `feglm()` over `glm()` even in quite small samples.

```
sec.alpaca <- system.time(mod.alpaca <- feglm(y ~ x1 + x2 + x3 | i + t, data,
    family = binomial()))[[3L]]
sec.glm <- system.time(mod.glm <- glm(y ~ x1 + x2 + x3 + factor(i) + factor(t) +
    0, data, family = binomial()))[[3L]]
c(alpaca = sec.alpaca, glm = sec.glm)
```

```
alpaca    glm
 0.056  3.179
```

References

Gaure, S. (2013), ‘lfe: Linear group fixed effects’, *The R Journal* **5**(2), 104 – 117.

Stammann, A. (2018), ‘Fast and feasible estimation of generalized linear models with high-dimensional k-way fixed effects’, *ArXiv e-prints* .