FACULTY OF LAW, ECONOMICS AND FINANCE

CREA
Discussion
Paper
2018-13

Management

Center for Research in Economics and Management
University of Luxembourg

Efficient estimation of maximum likelihood models with multiple fixed-effects: the R package FENmlm

available online : http://wwwfr.uni.lu/recherche/fdef/crea/publications2/discussion_papers

Laurent Bergé, CREA, University of Luxembourg,

July, 2018

For editorial correspondence, please contact: crea@uni.lu
University of Luxembourg
Faculty of Law, Economics and Finance
162A, avenue de la Faïencerie
L-1511 Luxembourg

Efficient estimation of maximum likelihood models with multiple fixed-effects: the R package FENmlm

Laurent R. Bergé*

CREA, Université du Luxembourg

July 6, 2018

Abstract

Fixed-effect models are widely used econometric methods. This paper presents the R package FENmlm which is devoted to the estimation of maximum likelihood (ML) models with any number of fixed-effects. The core of the algorithm, detailed in the paper, is based on a general framework to estimate any ML model with multiple fixed-effects. It also integrates a fixed-point acceleration method to hasten the convergence of the fixed-effect coefficients. The R function offers the user a simple way to estimate any of four different maximum likelihood models: Poisson, Negative Binomial, Gaussian and Logit. Illustrations with real data detail the estimation process as well as the clustering of standard-errors and the various tools to export and manage results from multiple estimations. Simulations show that the algorithm outperforms existing methods in terms of computing time (often by orders of magnitude) or, in the Gaussian case, is on a par with the most efficient ones. Most interestingly, apart from the Gaussian case, the algorithm is revealed to be the only able to estimate models with many fixed-effects on a simple laptop. FENmlm is a free software and distributed under the general public license, as part of the R software project.

Keywords: fixed-effects, R package, maximum likelihood estimation

* *e-mail*: laurent.berge@uni.lu

1 Introduction

Fixed-effects estimations are widely used econometric methods (see e.g., Allison, 2009 or Wooldridge, 2010). They provide a simple way to control for unobserved heterogeneity and are thus popular in many fields and in particular in the social sciences (Einav and Levin, 2014). Recent years have witnessed a vast increase in data availability while at the same time no commensurate increase in – often costly – computational power. This implies that econometric estimations for ever larger samples take ever more time and resources to run. Thus efficient algorithms are needed to alleviate this issue.

This paper presents the *R* package **FENmlm** which offers a method to efficiently estimate maximum likelihood (ML) models with *any number* of fixed-effects. The core of the algorithm is based on a general framework to estimate any ML model with multiple fixed-effects. Contrary to methods that treat the fixed-effects as simple coefficients and apply some "tricks" to facilitate their estimation (e.g., Greene, 2004), here the algorithm is solely based on the concentrated likelihood. Thus the optimization is done only on the coefficients of interest, while the fixed-effects are dealt with separately in the concentrated likelihood. Although the fixed-effects have no analytical solution, the algorithm integrates a fixed-point acceleration method to hasten the convergence to their solution (see Ramière and Helfer, 2015). Finally, all the moments of the concentrated likelihood are provided, important to obtain the asymptotic standard-errors of the coefficients of interest.

A gravity model example using trade data is used to illustrate the package which integrates four likelihood models: Poisson, Negative Binomial, Gaussian and Logit likelihood. It shows the estimation process and details how the user can easily cluster the standard-errors (Cameron et al., 2011), obtain the fixed-effects, and visualize and export (to Latex) the results of multiple estimations. In another example, it is shown how to estimate models with non-linear in parameters right hand sides.

Some methods for estimating fixed-effects maximum likelihood models already exist. In the Poisson case, there is the function `xtpoisson` in *Stata* and the *R* package **glmmML** (Broström and Holmberg, 2011; Broström, 2018). Both support only one set of fixed-effects. For the two fixed-effects case, there is the *Stata* module **poi2hdfe** (Guimaraes and Portugal, 2010). The *R* package **glmmML** has also implemented a function that deals with one fixed-effect in the Logit case. For the Negative Binomial, no method exists neither in *Stata*[1] nor in *R*. Finally, differently from the other models, there exists methods dealing with multiple fixed-effects in the Gaussian case, because the estimator is equivalent to the one of the ordinary least squares (OLS). In OLS, fixed-effects are dealt with by adding a first step in which all variables are projected alternately onto the fixed-effects subspaces. The Stata module

[1]Note that in Stata the function xtnbreg is not a fixed-effects method.

a2reg (Ouazad, 2008) deals with two fixed-effects. In the more general case of any number of fixed-effects, a method has been implemented in the *R* package **lfe** (Gaure, 2013b,a) and in the *Stata* module **reghdfe** (Correia, 2016). Now the package **FENmlm** integrates all the models into one function and extends the available range of fixed-effects for these models.

In a simulation exercise, the method reveals to be faster than all existing methods, even in the simple case of one fixed-effect. Depending on the likelihood model and the number of fixed-effects, the method is shown to be: faster, orders of magnitude faster or even the only one able to cope with the estimation using a simple laptop.[2] Only in the Gaussian case the performance of other existing methods are comparable to the one of **FENmlm**, and so for any number of fixed-effects.

This article however does not deal with the question of when the user should apply fixed-effects models (see e.g., Clark and Linzer, 2015). In particular, the Logit fixed-effect estimator, contrary to the three other likelihoods,[3] is known to suffer from the incidental parameters problem (Neyman and Scott, 1948; Lancaster, 2000). This problems leads to biased estimators (i.e., that deviate from their "true" value) for short panels. Appendix A offers an illustration of the incidental parameter problem in a Monte Carlo exercise. In this experiment, the Negative Binomial model seems not to suffer from this problem, in accordance with the article of Allison and Waterman (2002).

The next section details the algorithm implemented in the package. Section 3 describes the main functions and their arguments while Section 4 provides a full example of the functionalities of the package. The function is then compared to other methods in simulations reported in Section 5. Last section concludes.

2 Maximum likelihood estimation with multiple fixed-effects

2.1 Notations and problem formulation

Trade example. Consider the typical example of trade data where we want to explain trade flows between countries using fixed-effects (Eaton and Kortum, 2002). In this context, there are countries at the *origin* of the flow (the exporters), and countries at the *destination* of the flow (the importers). Further, say we have a balanced panel of n^{county} countries and

[2]The experiments were conducted using a laptop with 8GB of RAM and an Intel i7-6700HQ @2.6Ghz processor.

[3]Note that these three models do not suffer from the incidental parameter problem only for the parameters of interest. For example, in the Gaussian case the estimate of the parameter σ is also known to suffer from the incidental parameter problem, but not the other coefficients of the estimation, which are the coefficients of interest.

n^{year} years. Then we are interested in models for which the density of the random variable y_{ijt} is defined as:

$$d\left(y_{ijt}|X\right) = \mathcal{L}\left(y_{ijt}, \alpha_i^{origin} + \alpha_j^{destination} + \alpha_t^{year} + \beta' x_{ijt}, \theta\right), \tag{1}$$

where X represents the $N \times K$ matrix of all explanatory variables, x_{ijt} is one row-vector of this matrix, β is the vector of parameters of interest and θ represents some ancillary parameters as for example the over-dispersion parameter in the Negative Binomial model. The subscripts i, j and t refer to the country of origin, the country of destination and the year, respectively. The previous equation contains three sets of fixed-effects denoted by the vectors α^{origin}, $\alpha^{destination}$ and α^{year}.

This section illustrates in details a method to efficiently estimate the parameters β in the presence of fixed-effects. Not only the method applies to Equation (1) which has three sets of fixed-effects, but it is developed for models with any number of fixed-effects that enter the density linearly.

Notations. Since we are interested in dealing with any number of fixed-effects, we introduce new notations which will necessarily be a tad cumbersome to deal with the generality. We define the notations while applying them to the previous example.

Consider N observations denoted by the letter o. There are C different *clusters*, a cluster being defined as a fixed-effects *dimension*. In the previous example we have $C = 3$, meaning there are three clusters: the country of origin fixed-effects is the first cluster, the country of destination fixed-effects is the second while the year fixed-effects is the last one.

Each cluster c represents a partition of the data and is composed of n_c different categories. In the previous example, the first cluster is the country of origin fixed-effects, and the dimension of this first cluster is the number of countries: $n_1 = n^{country}$; this is similar for the second cluster $n_2 = n^{country}$; for the third cluster, the year fixed-effects, we have $n_3 = n^{year}$.

The vector γ^c represents the fixed-effects coefficients of cluster c and is of length n_c. The vectors γ^1, γ^2 and γ^3 are equivalent to the fixed-effects vectors of the previous example: α^{origin}, $\alpha^{destination}$ and α^{year}.

An important element is the N-vector $\pi^c \in \{1, \ldots, n_c\}^N$ which gives the category of cluster c to which each observation belongs. That is, if $\pi_o^c = v$ this means that observation o is of category v in cluster c. Keeping the previous example, looking at the first cluster, if the country of origin number 2 appears in observations 3, 5, 7, then $\pi_3^1 = \pi_5^1 = \pi_7^1 = 2$. To continue with the example, looking at the third cluster, if observations 4, 5 and 7 are of year 1, 2 and 3; then we have $\pi_4^3 = 1$, $\pi_5^3 = 2$ and $\pi_7^3 = 3$.

Finally $\delta_v^c \equiv \{o|\pi_o^c = v\}$ is the set of observations which are of category v in cluster c, this notation will be useful later on.

With these new notations at hand, we can rewrite the Equation (1):

$$d\left(y_o|X\right) = \mathcal{L}\left(y_o, \beta' x_o + \gamma^1_{\pi^1_o} + \gamma^2_{\pi^2_o} + \gamma^2_{\pi^2_o}, \theta\right),$$

where $\gamma^1_{\pi^1_o}$ replaces α_i^{origin}, with $\pi^1_o = i$ and $\gamma^1 = \alpha^{origin}$; the value $\gamma^2_{\pi^2_o}$ replaces $\alpha_j^{destination}$, with $\pi^2_o = j$ and where $\gamma^2 = \alpha^{destination}$, etc. Note that the two equations are exactly identical.

The general problem. In what follows we drop the ancillary parameters to treat the case of only likelihood functions without such parameters. I treat likelihood functions with an ancillary parameter in Section 2.6.

Although more complex, these notations prove convenient to deal with any number of clusters. The general set of models we deal with is then defined by the following density:

$$d\left(y_o|X\right) = \mathcal{L}\left(y_o, \beta' x_o + \sum_{c=1}^{C} \gamma^c_{\pi^c_o}\right),$$

with $\mathcal{L}$ a likelihood function.

In this context, the econometric problem of the maximum likelihood estimation pertains to finding the set of coefficients that maximizes the log-likelihood function:

$$\left(\hat{\beta}, \hat{\gamma}^1_1, \ldots \hat{\gamma}^C_{n_C}\right) \equiv \arg \max_{\beta, \gamma^1_1, \ldots \gamma^C_{n_C}} \sum_{o=1}^{N} \ln \mathcal{L}\left(y_o, \beta' x_o + \sum_{c=1}^{C} \gamma^c_{\pi^c_o}\right) \tag{2}$$

The naive way to obtain the estimate of $\hat{\beta}$ given the fixed-effects setup is simply to maximize the likelihood over the $K + \sum_c n_c$ parameters.[4] In practice however, the dimension of $\sum_c n_c$ is often very large, making such direct estimation at least very slow, at worst just impossible (e.g., due to memory limitations). Think for instance to the case of individual micro data where there can be thousands/millions of fixed-effects.

But the fixed-effects have a very specific structure: each cluster represents a partition of the data. One can then leverage this specific property to simplify the problem at hand. In particular, given that $K \ll \sum_c n_c$, finding a way to optimize only over the K parameters would greatly simplify the problem. This is what does the method I introduce here which is based on the concentrated likelihood.

Concentrated likelihood. The concentrated likelihood is defined as:

$$g(\beta) \equiv \sum_{o=1}^{N} \ell\left(y_o, \beta' x_o + \sum_{c=1}^{C} \tilde{\gamma}^c_{\pi^c_o}(\beta)\right),$$

[4]Note that the exact number of parameters to estimate is $K + \sum_c n_c$ minus the number of parameters restrictions to avoid collinearity (there are at least $C - 1$ such restrictions).

with ℓ the log-likelihood ($\ln \mathcal{L}$) and where $\left(\tilde{\gamma}_1^1(\beta), \ldots, \tilde{\gamma}_{n_C}^C(\beta)\right)$ is *one* set of fixed-effects coefficients that maximize the likelihood given β:

$$\left(\tilde{\gamma}_1^1(\beta), \ldots, \tilde{\gamma}_{n_C}^C(\beta)\right) \equiv \arg \max_{\gamma_1^1, \ldots, \gamma_{n_C}^C} \sum_{o=1}^{N} \ell\left(y_o, \beta' x_o + \sum_{c=1}^{C} \gamma_{\pi_o^c}^c\right).$$

Using the concentrated likelihood, we then only need to maximize over β. This method is more efficient only if the problem involved in the concentration of the likelihood is "more simple" than the optimization of the full likelihood, which is the case thanks to the particular structure of the clusters.

In what follows, I describe how to obtain the set of fixed-effects $\tilde{\gamma}_v^c(\beta)$, the gradient and the Hessian of the concentrated likelihood $g(\beta)$.

2.2 The fixed-effects of the concentrated likelihood

Unfortunately, when there is more than one cluster, concentrating the likelihood over the fixed-effects gives no closed-form solution. It is yet possible to obtain the set of fixed-effects of the clusters through a fixed-point algorithm.

The fixed-effects of $g(\beta)$ are defined as maximizing the likelihood given β, this means that they jointly respect the following first-order condition:

$$\begin{aligned} \sum_{o \in \delta_w^c} \frac{\partial \ell\left(y_o, \beta' x_o + \sum_{c' \neq c} \tilde{\gamma}_{\pi_o^{c'}}^{c'}(\beta) + \tilde{\gamma}_w^c(\beta)\right)}{\partial \gamma_w^c(\beta)} &= 0 \\ \Leftrightarrow \sum_{o \in \delta_w^c} \ell'\left(y_o, \beta' x_o + \sum_{c' \neq c} \tilde{\gamma}_{\pi_o^{c'}}^{c'}(\beta) + \tilde{\gamma}_w^c(\beta)\right) &= 0, \forall c, w, \end{aligned} \tag{3}$$

where ℓ' is the first derivative of the log-likelihood function ℓ with respect to its second argument, and δ_w^c is the set of observations of category w in cluster c. The set of all fixed-effects of all clusters is then the solution of a system of $\sum_{c=1}^{C} n_c$ equations with the same number of unknowns.

Note that although there is an infinity of cluster coefficients that solve the system of equation defined by Equation (3),[5] the sum of the fixed-effects over all clusters for each observation is unique. Let $S(\beta)$ be the vector of the sum of fixed-effects, such that $S_o(\beta) \equiv \sum_{c=1}^{C} \tilde{\gamma}_{\pi_o^c}^c(\beta)$, then the previous system can be rewritten as

$$\sum_{o \in \delta_w^c} \ell'\left(y_o, \beta' x_o + S_o(\beta)\right) = 0, \forall c, w. \tag{4}$$

[5]For instance consider the case of two clusters and a set of cluster coefficients γ^1 and γ^2 that solves Equation (3). Then you can define $\gamma_v^{1,bis} \equiv \gamma_v^1 + 10, \forall v$ and $\gamma_v^{2,bis} \equiv \gamma_v^2 - 10, \forall v$, and this new set of cluster coefficients $\gamma^{1,bis}$ and $\gamma^{2,bis}$ also solves Equation (3).

Algorithm 1 Obtaining the cluster coefficients for any number of clusters.

1. Initialize the sum of all cluster coefficients for each observation to 0: $S_o^0 = 0, \forall o$.
2. Loop until convergence:
   - For each cluster c:
     - For each category w of cluster c
       * Compute the optimal increment z_w^c such that the first order condition holds:
       $$\sum_{o \in \delta_w^c} \ell' \left(y_o, \beta' x_o + S_o^t + z_w^c \right) = 0$$
       The solution of this equation for each likelihood model implemented in the package are given in Appendix B.
       * Add the obtained increment to S_o: $S_o^{t+1} \leftarrow S_o^t + z_w^c, \forall o \in \delta_w^c$
3. At convergence, the vector S^t is a quantity such that Equation (3) holds for any c and w.

When there is only one cluster, getting the fixed-effects is straightforward since the value of each of them does not depend on the value of the coefficients for other clusters. The Poisson and Gaussian likelihood functions have a closed-form solution for Equation (3), while for the Negative Binomial and the Logit models the cluster coefficients are obtained using a Newton-Raphson algorithm combined with dichotomy. The solution of Equation (3) for one cluster and for each likelihood model implemented in the package are given in Appendix B.

When there are two or more clusters, we can use an iterative fixed-point algorithm to find out the solution to Equation (4). Algorithm 1 illustrates the process. At each step, the algorithm computes the first order condition taking the sum of all fixed-effects, S, as a given. Then it adds the obtained coefficients to S, and repeat the process until there is no more increment. This algorithm is convergent because each iteration is equivalent to increasing the overall log-likelihood with respect to each cluster, up to its maximum.

Although convergent, in some cases the speed of convergence of this algorithm can be slow. When observations are evenly distributed across the clusters, then convergence speed is fast. It becomes, however, a problem when the clusters are strongly unbalanced across observations, becoming dire when the cluster partitions are very close to each other.[6] In such cases, we can apply well functioning fixed-point algorithms to increase convergence

[6] Gaure (2013b) aptly describes, although in the linear context, the difference in convergence rate in the case of two clusters using the analogy of geometry: "Consider e.g., two intersecting lines in [an] Euclidean plane. To find the intersection, we can start at a point anywhere in the plane, project it orthogonally onto one of the lines, project it again onto the other line, then back to the first line, and so on. We zigzag towards the intersection. Clearly, if the lines are orthogonal, we will reach the intersection in just two steps. If the lines intersect at an acute angle, more steps will be needed to get close."

speed (see Ramière and Helfer, 2015, for a review of different methods). In practice, we implemented the Irons and Tuck iteration (Irons and Tuck, 1969).

I now describe how this acceleration process was implemented. I illustrate it with an example requiring new notations. So readers not interested in these details may skip to the next subsection.

Acceleration algorithm to obtain the fixed-effects. To illustrate the Irons and Tuck iteration, let us take the example of three clusters. We define the vectors a, b and c as the vectors of fixed-effects of the three clusters (i.e., referring to γ^1, γ^2 and γ^3).

The first order condition for the first cluster can be written as:

$$\sum_{o \in \delta_w^1} \ell' \left(y_o, \beta' x_o + b_{\pi_o^2} + c_{\pi_o^3} + a_w \right) = 0, \forall w. \tag{5}$$

Solving this equation for all a_w gives us the vector a that maximizes the likelihood given all other parameters.

Let us define the function $f^1 : \mathbb{R}^{n_2} \times \mathbb{R}^{n_3} \longrightarrow \mathbb{R}^{n_1}$ such that $a = f^1(b, c)$ is the vector solution of Equation (5) given the values of the other fixed-effects b and c. Similarly we can define the functions $f^2(a, c)$ and $f^3(a, b)$ that give the solution of the first order conditions for the second and third cluster taking the value of other fixed-effects as given. To find out the set of vectors (a^*, b^*, c^*) that *jointly* solves the first order conditions, we can apply the following fixed-point algorithm:

$$\begin{aligned} a^{t+1} &= f^1\left(b^t, c^t\right) \\ b^{t+1} &= f^2\left(a^{t+1}, c^t\right) \\ c^{t+1} &= f^3\left(a^{t+1}, b^{t+1}\right), \end{aligned}$$

given one set of starting values $\left(a^0, b^0, c^0\right)$.

Clearly we can rewrite it simply as a function of b^t and c^t since the value of a^{t+1} is deduced from these two vectors. So in fact we only have:

$$\begin{aligned} b^{t+1} &= f^2\left(f^1\left(b^t, c^t\right), c^t\right) \\ c^{t+1} &= f^3\left(f^1\left(b^t, c^t\right), f^2\left(f^1\left(b^t, c^t\right), c^t\right)\right). \end{aligned}$$

By denoting

$$F\begin{pmatrix} b^t \\ c^t \end{pmatrix} \equiv \begin{pmatrix} f^2\left(f^1\left(b^t, c^t\right), c^t\right) \\ f^3\left(f^1\left(b^t, c^t\right), f^2\left(f^1\left(b^t, c^t\right), c^t\right)\right) \end{pmatrix}$$

and

$$Z^t \equiv \begin{pmatrix} b^t \\ c^t \end{pmatrix},$$

we can now rewrite the fixed-point algorithm described in Algorithm (1), which is *exactly* identical to:

$$Z^{t+1} = F\left(Z^t\right).$$

As mentioned before, this algorithm can be slow depending on the structure of the clusters. To accelerate the convergence, we use instead the Irons and Tuck iteration (Ramière and Helfer, 2015) which is defined as:

$$Z^{t+1} = F\left(F\left(Z^t\right)\right) - \frac{\Delta F\left(Z^t\right) \cdot \Delta^2 Z^t}{\left\|\Delta^2 Z^t\right\|^2} \Delta F\left(Z^t\right),$$

where $\Delta Z^t = F\left(Z^t\right) - Z^t$, $\Delta F\left(Z^t\right) = F\left(F\left(Z^t\right)\right) - F\left(Z^t\right)$, and $\Delta^2 Z^t = \Delta F\left(Z^t\right) - \Delta Z^t$.

Although I illustrated the method with a 3 clusters example, it can be straightforwardly extended to the case of any number of clusters.

2.3 The gradient of the concentrated likelihood

To optimize $g(\beta)$ one key element of the optimization process is the gradient. Once the optimal fixed-effects coefficients are obtained (see previous section), the gradient of function g is straightforward. Indeed, the derivative with respect to the k^{th} coefficient is:

$$\begin{aligned} \frac{\partial g(\beta)}{\partial \beta_k} &= \sum_o \frac{\partial \ell\left(y_o, \beta' x_o + \sum_c \tilde{\gamma}^c_{\pi^c_o}(\beta)\right)}{\partial \beta_k} \\ &= \sum_o \left[x_{ok} + \sum_c \frac{\partial \tilde{\gamma}^c_{\pi^c_o}(\beta)}{\partial \beta_k}\right] \ell'\left(y_o, \beta' x_o + S_o(\beta)\right) \\ &= \sum_o x_{ok} \times \ell'\left(y_o, \beta' x_o + S_o(\beta)\right), \end{aligned} \tag{6}$$

where the last equality is obtained by using the identity of Equation (4) and noting that, for each cluster c:

$$\begin{aligned} \sum_o \left\{ \frac{\partial \tilde{\gamma}^c_{\pi^c_o}(\beta)}{\partial \beta_k} \ell'\left(y_o, \beta' x_o + S_o(\beta)\right) \right\} &= \sum_{v=1}^{n_c} \sum_{o \in \delta^c_v} \frac{\partial \tilde{\gamma}^c_v(\beta)}{\partial \beta_k} \ell'\left(y_o, \beta' x_o + S_o(\beta)\right) \\ &= \sum_{v=1}^{n_c} \frac{\partial \tilde{\gamma}^c_v(\beta)}{\partial \beta_k} \sum_{o \in \delta^c_v} \ell'\left(y_o, \beta' x_o + S_o(\beta)\right) \\ &= 0. \end{aligned}$$

Thus the gradient of g can readily be computed since it depends only on known quantities.

2.4 The Hessian of the concentrated likelihood

After simplification, the elements of the Hessian can be written as:[7]

$$\frac{\partial^2 g(\beta)}{\partial\beta_k\partial\beta_l} = \sum_o \left[x_{ok} + \frac{\partial S_o(\beta)}{\partial\beta_k}\right]\left[x_{ol} + \frac{\partial S_o(\beta)}{\partial\beta_l}\right]\ell''\left(y_o, \beta' x_o + S_o(\beta)\right). \tag{7}$$

This expression is problematic because it uses quantities for which we have no direct expression, namely $\partial S_o(\beta)/\partial\beta_k$. The next steps consist in finding this value.

Derivatives of the fixed-effects of the concentrated likelihood. First, note that Equation (4) holds by the definition of the concentrated likelihood, whatever the value of β. This means that the quantity $\sum_{o\in\delta_w^c}\ell'\left(y_o, \beta' x_o + S_o(\beta)\right), \forall c, w$ is constant with respect to β. Which leads to:

$$\begin{aligned}
\frac{\partial}{\partial\beta_k}\left(\sum_{o\in\delta_w^c}\ell'\left(y_o, \beta' x_o + S_o(\beta)\right)\right) &= 0 \\
\Leftrightarrow \sum_{o\in\delta_w^c}\left[x_{ok} + \frac{\partial S_o(\beta)}{\partial\beta_k}\right]\ell''\left(y_o, \beta' x_o + S_o(\beta)\right) &= 0 \\
\Leftrightarrow \sum_{o\in\delta_w^c}\left[x_{ok} + \sum_{c'\neq c}\frac{\partial\tilde{\gamma}^{c'}_{\pi_o^{c'}}(\beta)}{\partial\beta_k} + \frac{\partial\tilde{\gamma}_w^c(\beta)}{\partial\beta_k}\right]\ell''\left(y_o, \beta' x_o + S_o(\beta)\right) &= 0 \\
\Leftrightarrow \frac{\partial\tilde{\gamma}_w^c(\beta)}{\partial\beta_k} = -\frac{\sum_{o\in\delta_w^c}\left[x_{ok} + \sum_{c'\neq c}\frac{\partial\tilde{\gamma}^{c'}_{\pi_o^{c'}}(\beta)}{\partial\beta_k}\right]\ell''\left(y_o, \beta' x_o + S_o(\beta)\right)}{\sum_{o\in\delta_m^c}\ell''\left(y_o, \beta' x_o + S_o(\beta)\right)}&, \forall c, m.
\end{aligned} \tag{8}$$

The derivatives of the fixed-effects of the concentrated likelihood, $\partial\tilde{\gamma}_w^c(\beta)/\partial\beta_k$, can then be obtained as the solution of a system of $\sum_{c=1}^{c} n_c$ *linear* equations.

It is clear that when there is only one cluster, the derivatives can be obtained directly since their values do not depend on the value of the derivatives of other clusters. When there are two or more clusters however, we need to solve this system of linear equations. To solve it I simply use a fixed-point algorithm described in Algorithm (2) to find out $\partial S(\beta)/\partial\beta_k$, the partial derivative of the sum of the fixed-effects.

Now that we have a numerical approximation of $\partial S(\beta)/\partial\beta_k$, the Hessian can be computed according to Equation (7).

[7] Note that based on Equation (4) and after some simple calculus, $\sum_o\left(\partial^2 S_o(\beta)/\partial\beta_k/\partial\beta_l\right)\ell'\left(y_o, \beta' x_o + S_o(\beta)\right) = 0$.

Algorithm 2 Obtaining $\partial S_o(\beta)/\partial\beta_k$, the partial derivative of the sum of the fixed-effects with respect to β_k.

1. Initialize the vector of partial derivatives to 0: ${S'_{ok}}^0 = 0, \forall o$.
2. Loop until convergence:
   - For each cluster c:
     - For each category w of cluster c
       * Compute z_w^c that solves Equation (8):

$$z_w^c = \frac{-1}{\sum_{o\in\delta_w^c} \ell''\left(y_o, \beta' x_o + S_o(\beta)\right)} \times \sum_{o\in\delta_w^c} \left[x_{ok} + {S'_{ok}}^t\right] \ell''\left(y_o, \beta' x_o + S_o(\beta)\right)$$

       * Add the increments to the sum of derivatives S': ${S'_{ok}}^{t+1} \leftarrow {S'_{ok}}^t + z_w^c$, $\forall o \in \delta_w^c$.
3. At convergence, the vector ${S'_{ok}}^t$ is a quantity such that Equation (8) holds for any c and w.

2.5 Summing up the optimization process

The solution of the fixed-effects maximum likelihood estimation is then the result of the optimization over the K parameters of the concentrated likelihood function $g(\beta)$. At each step of the optimization process:

1. the optimal fixed-effects coefficients of the concentrated likelihood are obtained according to Algorithm 1,
2. the gradient is computed along Equation (6),
3. the sum of the partial derivatives of the fixed-effects are computed according to Algorithm 2,
4. the hessian is computed according to Equation (7).

By construction, the speed of this algorithm is then without comparison to methods treating the fixed-effects as variables and optimizing over the $K + \sum_c n_c$ parameters.

2.6 Specific cases

Here I detail two specific cases: 1) when likelihood models have an ancillary parameter (next subsection), and 2) when we assume non-linear in parameters functions.

2.6.1 Likelihood models with an ancillary parameter

So far, we have dealt only with likelihood models without ancillary parameter (such as the Gaussian, the Poisson or the Logit models). In this section I describe the additional specifics of likelihood models with one ancillary parameter.

First, the ML problem is now written as:

$$\left(\hat{\beta}, \hat{\gamma}_1^1, \dots \hat{\gamma}_{n_C}^C, \hat{\theta}\right) \equiv \arg \max_{\beta, \gamma_1^1, \dots \gamma_{n_C}^C} \sum_{o=1}^{N} \ln \mathcal{L}\left(y_o, \beta' x_o + \sum_{c=1}^{C} \gamma_{\pi_o^c}^c, \theta\right).$$

Meaning that 1) we have the additional parameters θ to estimate, and 2) the likelihood function now has three arguments (instead of two previously).

We do similarly as before, with a concentrated likelihood approach:

$$g(\beta, \theta) \equiv \sum_{o=1}^{N} \ell\left(y_o, \beta' x_o + \sum_{c=1}^{C} \tilde{\gamma}_{\pi_o^c}^c(\beta, \theta), \theta\right),$$

where the concentrated likelihood is now also a function of θ, the ancillary parameter.

Notations of the derivatives of the log-likelihood. Contrary to the case without ancillary parameter where the derivation of the log-likelihood was unequivocal, now the log-likelihood has three parameters and we are going to use derivatives with respect to the second and/or third argument. Thus we use subscripts to explicit the argument over which we derive it, so that e.g., ℓ_2' is the first derivative with respect to the second argument and $\ell_{2,3}''$ is the cross derivative between the second and third arguments, etc...

Fixed-effects of the concentrated likelihood. First we need to obtain the concentrated likelihood parameters $\left(\tilde{\gamma}_1^1(\beta, \theta), \dots, \tilde{\gamma}_{n_C}^C(\beta, \theta)\right)$ which are *one* set of fixed-effects that jointly solve the following system of first order conditions:

$$\Leftrightarrow \sum_{o \in \delta_w^c} \ell_2'\left(y_0, \beta' x_o + \sum_{c' \neq c} \tilde{\gamma}_{\pi_o^{c'}}^{c'}(\beta, \theta) + \tilde{\gamma}_w^c(\beta, \theta), \theta\right) = 0, \forall c, w. \tag{9}$$

This problem is identical to the case without θ, so we obtain this set of fixed-effects, $\tilde{\gamma}_w^c(\beta, \theta)$, $\forall c, w$, exactly as before.

Gradient. The gradient with respect to β is identical as before because the fixed-effects only appear in the second argument of the log-likelihood. Using the identity of Equation (9),

the gradient with respect to θ is:

$$\frac{\partial g(\beta,\theta)}{\partial \theta} = \sum_o \ell_3'\left(y_o, \beta' x_o + S_o(\beta,\theta), \theta\right),$$

with $S_o(\beta,\theta)$ the sum of all fixed-effects of the concentrated likelihood for observation o (i.e., $S_o(\beta,\theta) \equiv \sum_{c=1}^{C} \tilde{\gamma}_{\pi_o^c}^c(\beta,\theta)$).

Hessian. The main differences concern the Hessian. Let us first look at the cross derivative. After simplifications and using again the identity of Equation (9), the cross-derivatives can be written as:

$$\begin{aligned}\frac{\partial g(\beta,\theta)}{\partial \beta_k \partial \theta} = \sum_o \left[x_{ok} + \frac{\partial S_o(\beta,\theta)}{\partial \beta_k}\right] \quad \times \quad & \left\{\frac{\partial S_o(\beta,\theta)}{\partial \theta} \times \ell_{2,2}''\left(y_o, \beta' x_o + S_o(\beta,\theta), \theta\right)\right. \\ & \left. + \ell_{2,3}''\left(y_o, \beta' x_o + S_o(\beta,\theta), \theta\right)\right\}.\end{aligned}$$

After simplifications, the second derivative with respect to θ is:

$$\begin{aligned}\frac{\partial^2 g(\beta,\theta)}{\partial^2 \theta} \quad = \quad & \sum_o \left[\frac{\partial S_o(\beta,\theta)}{\partial \theta}\right]^2 \ell_{2,2}''\left(y_o, \beta' x_o + S_o(\beta,\theta), \theta\right) \\ & + 2 \times \sum_o \frac{\partial S_o(\beta,\theta)}{\partial \theta} \times \ell_{2,3}''\left(y_o, \beta' x_o + S_o(\beta,\theta), \theta\right) \\ & + \sum_o \ell_{3,3}''\left(y_o, \beta' x_o + S_o(\beta,\theta), \theta\right).\end{aligned}$$

Both expressions require to know the derivative of the sums of the fixed-effects with respect to θ: $\partial S_o(\beta,\theta)/\partial\theta$. We detail how it can be obtained below.

The derivative of the sum of the fixed-effects. Using the same argument as in the case without ancillary parameter, we have that Equation (9) is constant with respect to θ. This leads to:

$$\frac{\partial \tilde{\gamma}_w^c(\beta,\theta)}{\partial \theta} = -\frac{\sum_{o \in \delta_w^c}\left[\ell_{2,3}''(y_o, \beta' x_o + S_o(\beta,\theta), \theta) + \sum_{c' \neq c} \frac{\partial \tilde{\gamma}_{\pi_o^{c'}}^{c'}(\beta,\theta)}{\partial \theta} \ell_{2,2}''(y_o, \beta' x_o + S_o(\beta,\theta), \theta)\right]}{\sum_{o \in \delta_w^c} \ell_{2,2}''(y_o, \beta' x_o + S_o(\beta,\theta), \theta)}, \forall c, w.$$

Thus the set of derivatives $\partial \tilde{\gamma}_w^c(\beta,\theta)/\partial\theta$ is the solution of a linear system of equations. We can solve this system with a fixed-point algorithm, similarly to Algorithm 2.

Summing up the optimization. To obtain the ML estimate for likelihood models with an ancillary parameter, we then just have to apply a maximization algorithm on the concen-

trated likelihood function $g(\beta,\theta)$. This function has only $K+1$ parameters and its essential components were described above.

2.6.2 Non-linear in parameters functions

So far we assumed that the set of parameters of interest, β, were linearly associated to the variables X. Now let us extend to the case of non-linear in parameters functions. Consider the function $f(x_o,\beta)$ that links the individual variables $x_o \in \mathbb{R}^{K_1}$ to the vector of parameters $\beta \in \mathbb{R}^{K_2}$, and is twice differentiable. For the example, in the previous linear case we had $f(x_o,\beta) = \beta' x_o = \sum_k x_{ok}\beta_k$.

Now the ML problem writes:

$$\left(\hat{\beta},\hat{\gamma}_1^1,\ldots\hat{\gamma}_{n_C}^C\right) \equiv \arg\max_{\beta,\gamma_1^1,\ldots\gamma_{n_C}^C} \sum_{o=1}^{N} \ln\mathcal{L}\left(y_o, f(x_o,\beta) + \sum_{c=1}^{C}\gamma_{\pi_o^c}^c\right).$$

The process to obtain the ML estimates is exactly similar to the linear case: we use the concentrated likelihood approach. The difference with the linear case are very limited: we only have to write explicitly the derivatives of $f(x_o,\beta)$.

So the gradient becomes:

$$\frac{\partial g(\beta)}{\partial \beta_k} = \sum_o \frac{\partial f(x_o,\beta)}{\partial \beta_k} \times \ell'\left(y_o, f(x_o,\beta) + S_o(\beta)\right).$$

And the Hessian becomes:

$$\begin{aligned}\frac{\partial^2 g(\beta)}{\partial\beta_k\partial\beta_l} &= \sum_o \frac{\partial^2 f(x_o,\beta)}{\partial\beta_k\partial\beta_l} \times \ell'\left(y_o, f(x_o,\beta) + S_o(\beta)\right) \\ &\quad + \sum_o \left[\frac{\partial f(x_o,\beta)}{\partial\beta_k} + \frac{\partial S_o(\beta)}{\partial\beta_k}\right]\left[\frac{\partial f(x_o,\beta)}{\partial\beta_l} + \frac{\partial S_o(\beta)}{\partial\beta_l}\right]\ell''\left(y_o, f(x_o,\beta) + S_o(\beta)\right).\end{aligned}$$

The optimization process is identical.

2.7 Inferences and clustered variance-covariance estimators

From the previous optimization process we obtain the set of coefficients $\hat{\beta}$ that optimize the log-likelihood in the presence of fixed-effects. From asymptotic theory (see e.g., Wooldridge, 2010, chapter 12), these coefficient are normally distributed and their variance-covariance estimator, $VCE\left(\hat{\beta}\right)$, is defined as:

$$VCE\left(\hat{\beta}\right) \equiv \left(-H\left(\hat{\beta}\right)\right)^{-1}, \tag{10}$$

where H is the Hessian of function g. However, in the case of models where the data is partitioned into clusters, the variance of the parameters from $VCE\left(\hat{\beta}\right)$ may be underesti-

mated due to intra-cluster correlation (see e.g., Wooldridge, 2003; Hansen, 2007; Cameron and Miller, 2015). One way to circumvent this problem is to compute the one-way (or multi-way) clustered variance-covariance estimator (VCE).

One-way clustering. Assume that the VCE is clustered along cluster c_1. Let b_{ik} be the score of each observation, defined as:

$$b_{ok} \equiv \frac{\partial \ell\left(y_o, \beta' x_o + S_o\left(\beta\right)\right)}{\partial \beta_k}.$$

Computing b_{ik} requires to know the value of $\partial S_o\left(\hat{\beta}\right)/\partial \beta_k$ that is computed along Algorithm 2. Now let h_{c_1} be the $n_{c_1} \times K$ matrix defined as:

$$h_c = \begin{pmatrix} \sum_{o \in \delta_1^c} b_{o1} & \cdots & \sum_{o \in \delta_1^c} b_{oK} \\ & \ddots & \\ \vdots & \sum_{o \in \delta_w^c} b_{ok} & \vdots \\ & & \ddots \\ \sum_{o \in \delta_{n_{c_1}}^c} b_{o1} & \cdots & \sum_{o \in \delta_{n_{c_1}}^c} b_{oK} \end{pmatrix}$$

Then the one-way clustered VCE along cluster c_1 is equal to (Cameron et al., 2011):

$$VCE_{c_1} = V \cdot \left(h_{c_1}' h_{c_1}\right) \cdot V, \tag{11}$$

with $V \equiv VCE\left(\hat{\beta}\right)$.

Of course, this idea can be extended to a multi-way clustering of any order (Cameron et al., 2011). The method to compute the clustered VCEs are directly implemented in the package, up to four-way clustering.

3 Estimation and inference routines

This section first introduces the main inputs and outputs of the estimation procedures. Then it briefly describes some functions to visualize/export the results.

3.1 Estimation inputs and outputs

This package contains one main function: `femlm` which estimates maximum likelihood models with fixed-effects. The routine `femlm` has the following main arguments:

- `fml`: This formula contains the relation to be estimated. It should be linear in parameters (e.g., `fml = z ~ log(y)+log(x)`). To insert clusters, either add them in the

formula with a pipe (e.g., `fml = z ~ log(y)+log(x)|cluster_1` will estimate a model with fixed-effects according to the variable `cluster_1`) either use the other argument `cluster`.

- `data`: A data.frame containing the necessary variables contained in the formula. Note that to avoid unintentional mistakes there is no automatic treatment of `NA`s in the data. Thus, the variables from the formula should be clean of `NA` values.

- `family`: Four different maximum likelihood models have been implemented. They are: the Poisson model (the default), the Negative Binomial model, the Logit model and the Gaussian model.

  - `cluster`: This argument is a vector that gives the name of the variables to be treated as clusters. Note that if this argument is missing and the formula does not contain the information on clusters, then the estimation will be a standard ML estimation without fixed-effects.

- `NL.fml`: A formula containing the non-linear in parameters part of the right hand side, if any. In this formula, the coefficients to be estimated must appear explicitly. For example `NL.fml = ~ log(a*x+b*y)` where `a` and `b` are the coefficients to be estimated. This formula is exactly evaluated such that no constant is included by default. Note that although one can include linear in parameters terms in `NL.fml`, it is more efficient to include them directly in the `fml` argument.[8]

- `start`: Only used in the presence of a non-linear in parameters formula (i.e., `NL.fml`). This is a list of the starting values of the coefficients contained in the argument `NL.fml`.

Finally, other arguments can be passed to the function `femlm` that pertains to the optimization process, like the bounds of the non-linear coefficients if box constrained optimization is to be performed, or a user defined gradient of the non-linear function.

Outputs. The output is a `femlm` object that contains the following main elements:

- `coef`: The estimated coefficients $\hat{\beta}$.

- `coeftable`: The table with the estimated coefficients $\hat{\beta}$, as well as their standard-errors, associated z statistics and p values.

- `cov.unscaled`: The VCE obtained as described in Equation (10).

- `loglik`: The log-likelihood.

[8] For example, `fml=z~v1` with `NL.fml=~log(a*x+b*y)` is equivalent to estimating `fml=z~0` with `NL.fml=~constant+log(a*x+b*y)+coef_v1*v1`; but the former will be faster.

- `expected.predictor`: This value is the conditional expectation of the dependent variable: $E\left(Y|X,\hat{\beta}\right)$.

- `sq.cor`: This is the squared correlation between the dependent variable and the expected predictor. It is reported to give an idea of the goodness of fit for ML models, and relates to the idea of R^2 in the OLS case (Wooldridge, 2010).

- `sumFE`: The sum of the fixed-effects for each observation, this is equivalent to the variable $S_o\left(\hat{\beta}\right)$ defined in Section 2.

Note that the fixed-effects (the values $\tilde{\gamma}_w^c\left(\hat{\beta}\right)$) are not reported directly in the output but can be obtained using the function `getFE` that takes only a `femlm` object as argument. One `print` method is associated to it to obtain a quick overview of the fixed-effects, and a `plot` method depicts the most notable fixed-effects of each cluster.

3.2 Other facilities

From a practical viewpoint, ease of use is critical, thus this package has built-in methods to compute the standard-errors of the coefficients (`summary.femlm`), and to visualize on the *R*-console (`res2table`) as well as to export in Latex (`res2tex`) the results of multiple estimations.

The method `summary.femlm` computes the standard-errors of the coefficients according to the choice of the user. Its main augments are:

- `se`: The standard-errors can be computed in four different ways. First is the standard way described in Equation (10), which is the default. Second, the White corrected standard-errors can be computed (White, 1980). Finally, and more specifically for fixed-effect models, up to four-way clustered standard-errors are available (see Section 2.7 for details). Note that if the estimation was performed with several clusters, only the first k clusters are used to perform the k-way clustering. The clustering can also be done using user-specified clusters given in the argument `clusters`.

- `cluster`: A list of variables used to cluster the VCE. This is only used if multi-way clustering is to be performed.

- `dof_correction`: Whether finite sample corrections are to be applied to the VCE (this is not the case by default). If so, the VCE is multiplied by a factor $(N-1)/(N-K)$.

This function returns a `femlm` object with following additional elements:

- `cov.scaled`: The VCE computed according to the choice of the user.

- `coeftable`: This is the coefficient table where the standard-error, z-statistic and p-value of the coefficients are computed according to `cov.scaled`.

This package also includes the function `res2table` to display the results (i.e. coefficients, standard-errors, goodness of fit) of multiple estimations in the R-console and the function `res2tex` to export these results into a high quality Latex table. The main arguments of these two functions are:

- `...`: This should contain any number of `femlm` objects for which the results are to be displayed. Note that this argument also accepts lists of `femlm` objects.
- `se`: This argument is similar to the one in `summary.femlm` and states how the standard-errors should be computed for all models in the table.

On top of these arguments, both functions contain many display options.

4 Practical examples in *R*

In this section, we illustrate the function `femlm` using two examples in which fixed-effect ML is required. The first one consists in estimating a gravity model using data from international trade in Europe for which there are many clusters to include in the model (like country of destination, of origin, sector, etc...). The second example illustrates a non-linear in parameters estimation in which we correlate the production of inventors to their – non-linear – network centrality while using fixed-effects.

4.1 Application to a gravity model in international trade

The first illustration deals with international trade, which is a setup that often requires performing estimations with many fixed-effects. We estimate a very simple gravity model in which we are interested in finding out the negative effect of geographic distance on trade. The sample data consists of European trade extracted from Eurostat.[9] The data set focuses on the levels of bilateral importations between EU15 countries from 2007 and 2016. The data is further broken down according to 97 product categories. After cleaning for missing values, the data set consists of 189,712 observations. A sample of the data is given in Table 1. As we can see, the data is naturally partitioned into four clusters: the country of origin and destination of the trade flow, the product category and the year.

The dependent variable of the estimation will be the level of trade between two countries while the independent variable is the geographic distance between the two countries. To

[9]The exact table name is "DS-016894" and can be found on Eurostat's website: http://epp.eurostat.ec.europa.eu/newxtweb/ (the data was extracted in July 2017). A small sample based on this data source is included in the package.

Table 1: European bilateral trade data.

Country of Origin	Country of Destination	Product Category	Year	Distance (km)	Trade in Euros
Belgium	Austria	1	2007	770.06	210,000
Belgium	Austria	2	2007	770.06	6,303,716
Belgium	Austria	3	2007	770.06	4,235,115
⋮	⋮	⋮	⋮	⋮	⋮
Portugal	Sweden	94	2016	3057.36	13,637,252
Portugal	Sweden	95	2016	3057.36	77,236
Portugal	Sweden	96	2016	3057.36	2,868,293

Source: Eurostat.

obtain the elasticity of geographic distance net of the effects of the four clusters, we estimate the following model:

$$E\left(Trade_{i,j,p,t}\right) = \gamma_i^{Exporter} \times \gamma_j^{Importer} \times \gamma_p^{Product} \times \gamma_t^{Year} \times Distance_{ij}^{\beta},$$

where the subscripts i, j, p and t stand respectively for the exporting country, the importing country, the type of product and the year, and the γ_v^c are fixed-effects for these groups. Here β is the elasticity of interest.

The estimation of this model in R using a Poisson likelihood is as follows:

```
R> gravity_results <- femlm(Euros ~ log(dist_km), base_trade, family = "poisson",
+                           cluster = c("Origin", "Destination", "Product", "Year"))
R> print(gravity_results)
ML estimation, family = Poisson
Observations: 189,712
Cluster sizes: Destination: 15, Origin: 15, Product: 97, Year: 10
Standard errors type: Standard
Estimate Std. Error z value Pr(>|z|)
log(dist_km) -1.1766e+00 5.8969e-07 -1995198 < 2.2e-16 ***
---
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
# Evaluations: 4
Log-likelihood: -7.033212e+12
BIC: 1.406642e+13
Pseudo-R2: 0.8439591
Squared Cor.: 0.6211809
```

```
Convergence state: relative convergence (4)
```

As we can see, the `print` method already displays the main information regarding the estimation. The elasticity of distance is equal to -1.17 with a standard-error close to 0. However, because the data is partitioned into different clusters, treating each observation as independent from the others leads to underestimate the variance of the coefficient (see e.g., Cameron et al., 2011). To cope with this issue, we can use clustered standard-errors. Here the natural clusters are i) the country of origin and ii) the country of destination. Clustering the standard-errors with respect to the first two clusters is easily done using the method `summary` in combination with `se = "twoway"`:

```
R> summary(gravity_results, se = "twoway")
ML estimation, family = Poisson
Observations: 189,712
Cluster sizes: Destination: 15, Origin: 15, Product: 97, Year: 10
Standard-errors type: Two-way
Estimate Std. Error z value Pr(>|z|)
log(dist_km) -1.176552 0.082864 -14.199 < 2.2e-16 ***
---
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
# Evaluations: 4
Log-likelihood: -7.033212e+12
BIC: 1.406642e+13
Pseudo-R2: 0.8439591
Squared Cor.: 0.6211809
Convergence state: relative convergence (4)
```

We obtain the same information but with the corrected standard-error. As expected its value is higher than if the observations were considered as independent.

Displaying multiple estimations. Now I illustrate this package's facilities to display multiple outputs. Assume that we want to see the evolution of the coefficient of distance with respect to the progressive inclusion of cluster variables, and then export these results into a Latex table.

First, we use a loop to estimate models including more and more clusters:

```
R> gravity_subcluster = list()
R> all_clusters = c("Year", "Destination", "Origin", "Product")
R> for(i in 0:4){
```

```
R> gravity_subcluster[[i+1]] = femlm(Euros ~ log(dist_km), base_trade,
+                                    family = "poisson",
+                                    cluster = all_clusters[0:i])
R> }
```

Then, all the results are exported at once using the function `res2tex` with some options:

```
R> dict_gravity = c(Euros = "Trade Value in \\euro{}",
+                   "log(dist_km)" = "Distance in km (ln)",
+                   Destination = "Country of Destination",
+                   Origin = "Country of Origin", Product = "Product Category")
R> res2tex(gravity_subcluster, se = "twoway", dict = dict_gravity,
+          title = "Estimation of gravity models with different clusters.",
+          yesNoCluster = c("$\\checkmark$", ""),
+          cluster = base_trade[, c("Destination", "Origin")])
```

The results of these estimations are reported in Table 2, which is the raw output of `res2tex`. Note that all models of the table report standard errors clustered with respect to the country of origin and destination, thanks to the argument `se="twoway"` in combination with the `cluster` option. We relabel the variables by using the option `dict` and giving it the named vector `dict_gravity`. Finally, the option `yesNoCluster` is used to customize the symbols representing the presence of clusters (the default being simply `c("Yes", "No")`).

Now say we want to see the influence of the likelihood function on the estimated elasticity of distance. We use the argument `family` to estimate the model with the Gaussian and the Negative Binomial likelihoods. We compare the elasticities using the function `res2table` which displays the results on the *R* console.

```
R> all_clusters = c("Destination", "Origin", "Product", "Year")
R> gravity_gaussian <- femlm(log(Euros + 1) ~ log(dist_km), base_trade,
+                            family = "gaussian", cluster = all_clusters)
R> gravity_negbin <- femlm(Euros ~ log(dist_km), base_trade,
+                          family = "negbin", cluster = all_clusters)
R> res2table(gravity_result, gravity_gaussian, gravity_negbin, se = "twoway",
+            subtitles = c("Poisson", "Gaussian", "Negative Binomial"))
Poisson Gaussian Negative Binomial
log(dist_km) -1.1766*** (0.0829) -1.8168*** (0.1334) -1.4212*** (0.1225)
Dispersion Parameter 0.5066*** (0.0469)
Clusters: ------------------ ------------------ ------------------
Destination YES YES YES
```

Table 2: Estimation of gravity models with different clusters.

Dependent Variable:	Trade Value in €				
Model:	(1)	(2)	(3)	(4)	(5)
Variables					
(Intercept)	24.8779***				
	(0.9881)				
Distance in km (ln)	-0.9386***	-0.9386***	-1.0085***	-1.1616***	-1.1766***
	(0.1397)	(0.1397)	(0.1773)	(0.0835)	(0.0829)
Fixed-Effects					
Year		✓	✓	✓	✓
Country of Destination			✓	✓	✓
Country of Origin				✓	✓
Product Category					✓
Fit statistics					
Observations	189,712	189,712	189,712	189,712	189,712
Adj-pseudo R^2	0.10464	0.10575	0.22018	0.38259	0.84396
Log-Likelihood	-4.036×10^{13}	-4.031×10^{13}	-3.515×10^{13}	-2.783×10^{13}	-7.033×10^{12}
BIC	8.071×10^{13}	8.061×10^{13}	7.03×10^{13}	5.566×10^{13}	1.407×10^{13}

*Two-way clustered standard-errors in parenthesis. Signif Codes: ***: 0.01, **: 0.05, *: 0.1*

Notes: This table is the unaltered output of the function `res2tex`.

```
Origin YES YES YES
Product YES YES YES
Year YES YES YES
------------ ------------------- ------------------- -------------------
N 189,712 189,712 189,712
Squared-Correlation 0.621 0.714 0.526
Adj-pseudo R2 0.844 0.2366 0.0395
Log-Likelihood -7.033e+12 -383,002.99 -3,228,806.24
BIC 1.407e+13 767,646.68 6,459,265.34
```

According to the three models, the elasticity of geographic distance on trade ranges from -1.17 to -1.81.

Fixed-effects. To obtain the fixed-effects of the estimation, the function `getFE` must be performed on the results. The `print` method helps to have a quick overview:

```
R> fixedEffects <- getFE(gravity_result)
R> fixedEffects
Fixed-effects coefficients.
Destination Origin Product Year
Number of fixed-effects 15 15 97 10
Number of references 0 1 1 1
Mean 24.7 -0.378 0.0337 -0.0115
Variance 1.15 1.35 2.83 0.00652
COEFFICIENTS:
Destination: AT BE DE DK ES
24.31 24.79 26.16 23.79 25.64 ... 10 remaining
-----
Origin: AT BE DE DK ES
-0.8429 0 1.382 -1.108 0.4932 ... 10 remaining
-----
Product: 1 2 3 4 5
0 1.412 0.6547 1.446 -1.52 ... 92 remaining
-----
Year: 2007 2008 2009 2010 2011
0 -0.001208 -0.2125 -0.07906 0.01385 ... 5 remaining
R> plot(fixedEffects)
```

We can see that the fixed-effects are balanced across clusters. Indeed, apart from the first cluster, only one fixed-effect per cluster needs to be set as reference (i.e., fixed to 0) to avoid

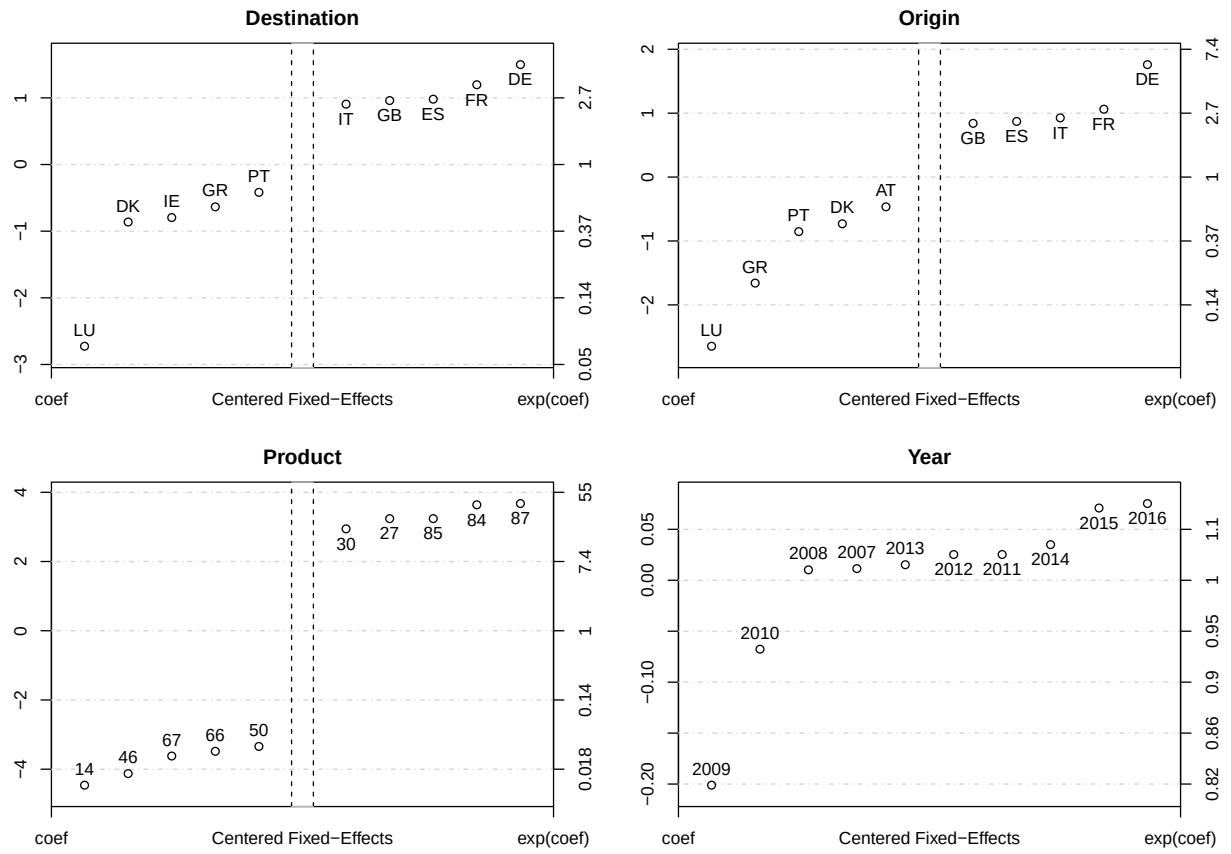

Figure 1: Most notable fixed-effects for each cluster of the gravity estimation. This is the result of the `plot` method applied to the fixed-effects.

collinearity across the fixed-effects of the different clusters. This ensures that the fixed-effects coefficients can be compared *within* cluster. Had there be strictly more than one reference per cluster, their interpretation would have been not possible at all. If this was the case though, a warning message would have been prompted. Note that the mean values are meaningless *per se*, but give a reference points to which compare the fixed-effects within a cluster.

Finally, the `plot` method helps to distinguish the most notable fixed-effects. The results of the plot are reported in Figure 1. For each cluster, the fixed-effects are first centered, then sorted, and finally the most notable (i.e., highest and lowest) are reported. The exponential of the coefficient is reported in the right hand side to simplify the interpretation for models with log-link (as the Poisson model). As we can see from the country of destination cluster, trade involving France (FR), Spain (ES) and Germany (DE) as destination countries is roughly 2.7 times higher than the EU15 average. Further, the highest heterogeneity come from the product category, where trade in product 87 (*vehicles*) is roughly 55 times the average while product 14 (*vegetable plaiting materials*) represents a negligible fraction of the average.

Note however that the interpretation of the fixed-effects must be taken with extra care. In particular, here the fixed-effects can be interpreted only because they are perfectly balanced.

4.2 Non-linear in parameters application

Now I provide an example of non-linear in parameters estimation with linear fixed-effects. This is an overly simple "toy" example based on a variation of Bergé et al. (2018) who investigate the link between the position of inventors within a network and their productivity. The context is that of inventors who collaborate to produce patents, the set of collaborations between them forming a network.

They use the following definition of network centrality:

$$c_{it}(\lambda, \alpha, \beta) = 1 + \lambda \sum_j g_{ij}^t \frac{c_{jt}^\alpha(\lambda, \alpha, \beta)}{d_{jt}^\beta}, \tag{12}$$

with c_{it} the centrality of individual i in year t, and λ, α and β are parameters of this centrality which reflect the idea of connectivity, complementarity and rivalry, respectively (see Bergé et al., 2018, for more details). The matrix g^t is the network matrix of year t such that $g_{ij}^t = 1$ is inventors i and j have collaborated in year t, and 0 otherwise. Finally $d_{jt} \equiv \sum_i g_{ij}$ is the degree of inventor i, i.e., the number of collaborators she has.

We here want to estimate the following relation:

$$Cites_{it} = \gamma_i^{inventor} \gamma_t^{year} c_{it}(\lambda, \alpha, \beta), \tag{13}$$

where $Cites_{it}$ is the number of citations received by the patents produced by inventor i stemming from other patents in a 5 years forward window – this is a proxy for quality (see Criscuolo and Verspagen, 2008, for a discussion of such measures).

The coefficients of interests are the ones of the network centrality. However, as it can be clearly seen in Equation (12), the network centrality parameters are not linearly associated to an exogenous variable: this is a non-linear relationship. However non-linear, the centrality is twice differentiable in its parameters for a broad range of their values.

To estimate Equation (13), we need to make use of the argument `NL.fml` to add this non-linear term, as follows:

```
R> res_cent = femlm(cites ~ 1, base,
+                   NL.fml = ~ log(centrality(lambda, alpha, beta)),
+                   start = list(lambda = 1, alpha = 0.2, beta = 0.5),
+                   lower = list(lambda = 0, alpha = 0, beta = 0),
+                   upper = list(alpha = 0.99),
+                   cluster = c("inventor", "year"))
```

The function `centrality` computes the centrality of all inventor-year and returns a vector of the same length and order as of the data set `base`. Details on the data and on the centrality code are given in Appendix C.

We add the argument `start` to give starting values to the parameters. Further, because the centrality is *not* defined for any real value of the parameters (for instance λ cannot be negative when $\alpha > 0$, and the centrality is not defined when $\alpha > 1$), we add restrictions on the values the parameters can take in the estimation process by using the arguments `lower` and `upper`. Finally we insert the fixed-effects as before with the argument `cluster`.

Finally, we use the function `summary` and cluster the standard-errors with respect to the inventors:

```
R> summary(res_cent, "cluster")
Non-linear ML estimation, family = Poisson
Observations: 6,415
Cluster sizes: inventor: 1423, year: 19
Standard-errors type: Clustered
Estimate Std. Error z value Pr(>|z|)
lambda 0.755431 0.196811 3.8384 0.000124 ***
alpha 0.653265 0.220619 2.9610 0.003066 **
beta 0.713139 0.171425 4.1601 3.2e-05 ***
---
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
# Evaluations: 5
Log-likelihood: -13,502.18
BIC: 39,663.03
Pseudo-R2: 0.2778562
Squared Cor.: 0.3845669
Convergence state: relative convergence (4)
```

We can see that the method finds the three centrality parameters that maximize the likelihood when controlling for individual heterogeneity via the fixed-effects.

5 Benchmarking

We now compare the performance of the *R* function `femlm` to other functions estimating maximum likelihood models in both *R* and *Stata*.

5.1 The setup

We first describe the generation of the data before detailing the function to which `femlm` will be compared.

Data generation. We generate the outcome variable y according to a negative binomial law with presence of over-dispersion:[10]

$$y_o \sim Negative\,Binomial\left(\exp\left(\mu_o\right), \theta = 0.5\right)$$

with $\mu_o = x_o + 0.05 \times x_o^2 + \sum_{c=1}^{3} \gamma_{\pi_o^c}^c$ where the variable x_o and the cluster coefficients γ_v^c were generated according to a Gaussian density of mean 0 and standard-deviation 1. Ten data sets are generated for a varying number of observations n and a varying number of clusters. The number of observations ranges from 10^3 to 10^6. We consider clusters sizes (i.e., number of fixed-effects per cluster) equal to $n/50$, $\sqrt{n}$ and $\sqrt[3]{n}$, respectively for the first, second and third cluster. The cluster to which belongs each observation is drawn randomly.

Empirical models. We use the same data to estimate the four different likelihood models. For notational simplicity, consider that the three clusters are perfectly balanced meaning that each observation is evenly split across the three clusters (indexed by the subscripts i, j and k).

The chosen empirical specification for the Poisson and Negative Binomial is:

$$E\left(y_{ijk}\right) = \exp\left(\alpha_i + \beta_j + \gamma_k + \delta\, x_{ijk}\right),$$

for the Gaussian case is:

$$E\left(\log\left(y_{ijk} + 1\right)\right) = \alpha_i + \beta_j + \gamma_k + \delta\, x_{ijk},$$

and for the Logit case is:

$$E\left(\mathbf{1}\left\{y_{ijk} > 0\right\}\right) = Logit\left(\alpha_i + \beta_j + \gamma_k + \delta\, x_{ijk}\right).$$

[10] The variance is here equal to $E\left(V\left(y_o\right)\right) = \exp\left(\mu_o\right) + \exp\left(\mu_o\right)^2 / 0.5$.

Where α_i, β_j and γ_k are the cluster coefficients of the first, second and third cluster. The term in x^2 used to generate the data is excluded to keep some noise.

Benchmark functions. The function `femlm` is compared to the following functions:

Poisson: the `xtpoisson` and `poi2hdfe` functions from *Stata*, the `glmmboot` function in *R* from the package **glmmML**. The `poi2hdfe` function is used only for the two clusters case since it has not been developed to handle more clusters. When there is more than one cluster, the other clusters are added as dummies for the functions `xtpoisson` and `glmmboot`.

Negative Binomial: the `nbreg` function in *Stata* and the `glm.nb` function in *R* from the package **MASS** (Venables and Ripley, 2002) where the fixed-effects are added manually as dummies.

Logit: the `logit` function from *Stata*, the `glmmboot` function from *R*. Fixed-effects are added manually for the `logit` function, they are added for more than one clusters for the `glmmboot` function.

Gaussian: the Gaussian ML estimator is identical to the OLS one, so I use the `reghdfe` and `a2reg` functions from *Stata*, and the `felm` function in `R` from the package `lfe`. These functions are devised to handle multiple clusters.

5.2 Results

Each data set, for a given number of observations and clusters, is generated 10 times, then we estimate the models with the different functions and finally the average computing time over the ten replications.

These experiments are performed on a laptop with 8GB of RAM and an Intel i7-6700HQ @2.6Ghz processor. The software used are: *R* version 3.4.3, package **FENmlm** version 2.1.0, package **glmmML** version 1.0.2, package **MASS** version 7.3, package **lfe** version 2.5; *Stata* 15 SE with all external modules (**poi2hdfe**, **a2reg** and **reghdfe**) downloaded in February 2018.

The results are reported in Figure 2.

Poisson. As we can see, even with only one set of fixed-effects, the function `femlm` is several times faster than the other methods. The gap increases starkly with the number of clusters. At two and three clusters, regular methods are unable to estimate the fixed-effects model with just 1 million observations, while it takes only around 10s to the `femlm` function. Even the function `poi2hdfe` which has been developed for high dimensional fixed-effects is more

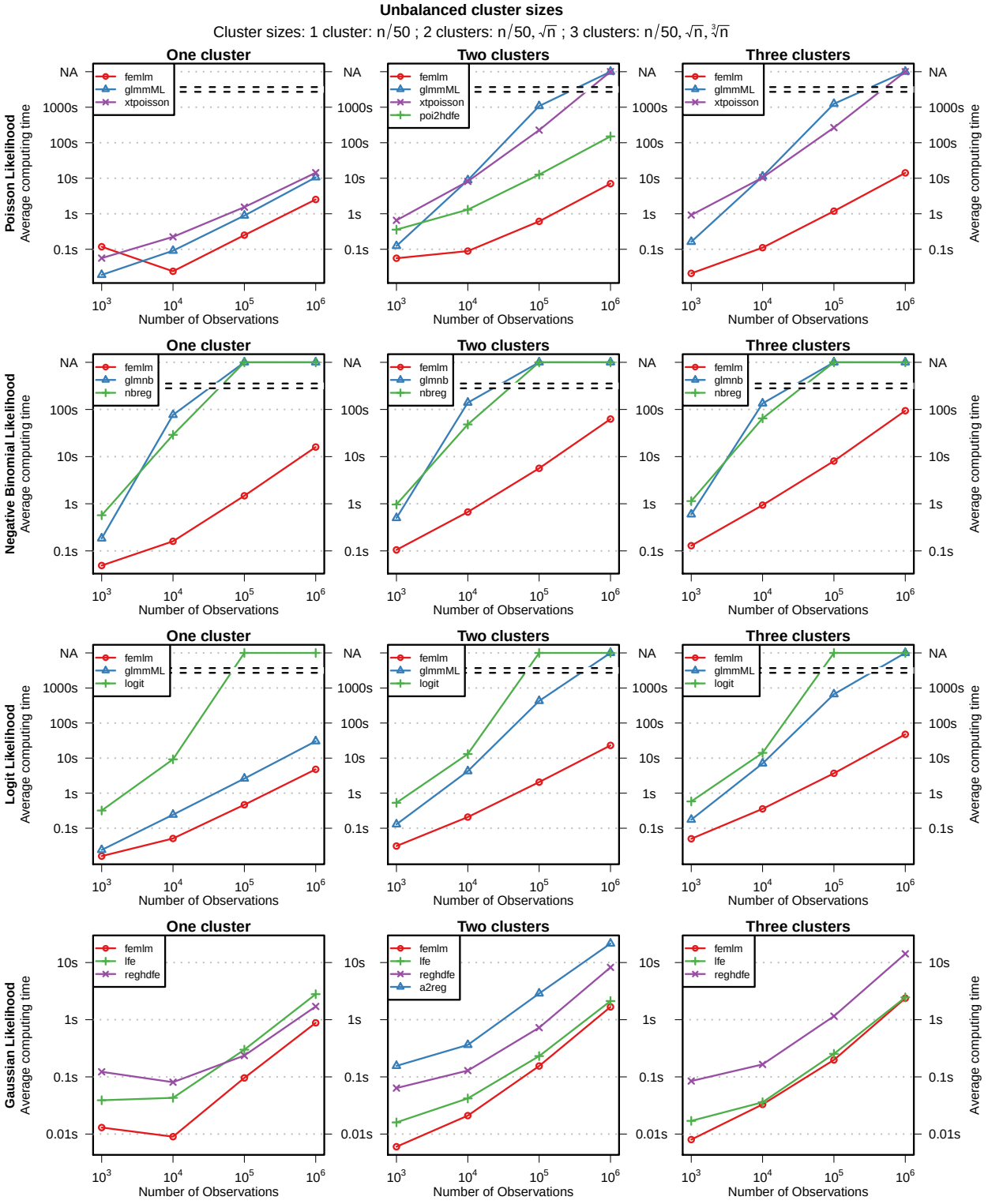

Figure 2: Average computing time on 10 data sets replications – varying number of clusters. Notes: NA (not available) values are reported when: the memory limit was reached or the computing time was in excess of 3600s (i.e., the process was stopped after 3600s which is then a lower bound for the computing time).

than an order of magnitude slower than `femlm`, and simply cannot be used to estimate models with three or more clusters.

Negative Binomial. As opposed to `femlm`, the functions `glm.nb` and `nbreg` have not been specifically devised to cope with fixed-effects, which explains why the difference is so important. For any number of clusters, with data sets as small as of 10,000 observations, the function `femlm` is already more than 100 times faster than the two other methods. For data sets of 100,000 or more observations, the two other methods simply cannot be estimated anymore due to the too large number of variables. This is in contrast to `femlm` which can cope with such data sizes in reasonable amounts of time (less than 100s for 3 clusters and one million observations).

Logit. We can see that the function `femlm` is faster than the two other methods, even `glmmboot` which has been developed to cope with fixed-effects. The difference increases with the number of clusters, to reach several orders of magnitude.

Gaussian. For all cluster cases, and for all data set sizes, the function `femlm` performs faster or equivalently to the most efficient algorithms.

Summary. Overall, these comparisons show that the function `femlm` is faster than all, and even outperforms most, existing methods that deal with fixed-effects. In particular, the function `femlm` is the only one possible to cope with fixed-effects in the Negative Binomial case, and this is also true for Poisson models with more than three clusters.

6 Summary and concluding remarks

This article has presented an algorithm to efficiently estimate maximum likelihood models with any number of fixed effects which has been implemented in the *R* package **FENmlm**. This package integrates four likelihood models (Poisson, Negative Binomial, Gaussian and Logit) and offers the user various possibilities, in particular: to easily cluster the standard-errors, and to exports the results of multiple estimations into customized Latex tables.

Simulations reveal that the method is consistently faster than other existing methods. This is particularly true for the three Poisson, Negative Binomial and Logit likelihoods where the proposed method is often the only one able to cope with estimations with multiple fixed-effects. Regarding the Gaussian case, the method is comparable to existing methods (although slightly faster in these simulations).

As limitations, the present method is constrained to fixed-effects that enter the likelihood function linearly. For instance, it does not cover conditional Negative Binomial models

which is a variation of fixed-effect models (where the over-dispersion parameters end up to be cluster-specific). Further, this package performs fixed-effects estimations, but sometimes such a setup is not the most appropriate to handle the research problem at hand. On this issue, see for instance Clark and Linzer (2015).

The *R* package **FENmlm** is available from the Comprehensive *R* Archive Network at https://CRAN.R-project.org/package=FENmlm.

References

Allison, P. D., 2009. ***Fixed effects regression models***. Vol. 160 of Quantitative Applications in the Social Sciences. SAGE publications.

Allison, P. D., Waterman, R. P., 2002. Fixed-effects negative binomial regression models. ***Sociological Methodology*** 32(1): 247–265.

Bergé, L. R., Carayol, N., Roux, P., 2018. How do inventor networks affect urban invention? ***Regional Science and Urban Economics***In Press.

Broström, G., 2018. glmmML: Generalized Linear Models with Clustering. R package version 1.0.2.
URL https://CRAN.R-project.org/package=glmmML

Broström, G., Holmberg, H., 2011. Generalized linear models with clustered data: Fixed and random effects models. ***Computational Statistics & Data Analysis*** 55(12): 3123–3134.

Cameron, A. C., Gelbach, J. B., Miller, D. L., 2011. Robust inference with multiway clustering. ***Journal of Business & Economic Statistics*** 29(2): 138–249.

Cameron, A. C., Miller, D. L., 2015. A practitioner's guide to cluster-robust inference. ***Journal of Human Resources*** 50(2): 317–372.

Clark, T. S., Linzer, D. A., 2015. Should I use fixed or random effects? ***Political Science Research and Methods*** 3(2): 399–408.

Correia, S., 2016. A feasible estimator for linear models with multi-way fixed effects. ***Duke University Preliminary Version. URL: www. scorreia. com/research/hdfe. pdf***.

Criscuolo, P., Verspagen, B., 2008. Does it matter where patent citations come from? inventor vs. examiner citations in European patents. ***Research Policy*** 37(10): 1892–1908.

Eaton, J., Kortum, S., 2002. Technology, geography, and trade. ***Econometrica*** 70(5): 1741–1779.

Einav, L., Levin, J., 2014. The data revolution and economic analysis. ***Innovation Policy and the Economy*** 14(1): 1–24.

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

Gaure, S., 2013b. OLS with multiple high dimensional category variables. ***Computational Statistics & Data Analysis*** 66: 8–18.

Greene, W., 2004. The behaviour of the maximum likelihood estimator of limited dependent variable models in the presence of fixed effects. ***The Econometrics Journal*** 7(1): 98–119.

Guimaraes, P., Portugal, P., 2010. A simple feasible procedure to fit models with high-dimensional fixed-effects. ***Stata Journal*** 10(4): 628.

Hansen, C. B., 2007. Asymptotic properties of a robust variance matrix estimator for panel data when T is large. ***Journal of Econometrics*** 141(2): 597–620.

Irons, B. M., Tuck, R. C., 1969. A version of the Aitken accelerator for computer iteration. ***International Journal for Numerical Methods in Engineering*** 1(3): 275–277.

Lancaster, T., 2000. The incidental parameter problem since 1948. ***Journal of Econometrics*** 95(2): 391–413.

Neyman, J., Scott, E. L., 1948. Consistent estimates based on partially consistent observations. ***Econometrica*** 16(1): 1–32.

Ouazad, A., 2008. A2REG: Stata module to estimate models with two fixed effects. ***Boston College Department of Economics***.

Ramière, I., Helfer, T., 2015. Iterative residual-based vector methods to accelerate fixed point iterations. ***Computers & Mathematics with Applications*** 70(9): 2210–2226.

Venables, W. N., Ripley, B. D., 2002. ***Modern applied statistics with S***, 4th Edition. Springer, New York, iSBN 0-387-95457-0.
URL http://www.stats.ox.ac.uk/pub/MASS4

White, H., 1980. A heteroskedasticity-consistent covariance matrix estimator and a direct test for heteroskedasticity. ***Econometrica*** 48(4): 817–838.

Wooldridge, J. M., 2003. Cluster-sample methods in applied econometrics. ***The American Economic Review*** 93(2): 133–138.

Wooldridge, J. M., 2010. ***Econometric analysis of cross section and panel data***, 2nd Edition. The MIT Press, Cambridge, Massachussets, London, England.

A Illustration of the incidental parameter problem

To inform on the extent of the incidental parameter problem, we perform a Monte Carlo simulation for the four likelihood models. We look at the bias of estimates of one coefficient in a simple panel setup, where the data was generated with either individual fixed-effects, or individual *and* time fixed-effects.

The data generation processes for the Gaussian, Poisson, Negative Binomial and Logit cases are as follows:

$$y_{it} = 2 \times x_{it} + \mu_{it} + \epsilon_{it},\, \epsilon_{it} \sim \mathcal{N}(0,1)$$

$$y_{it} \sim \text{Poisson}\left(\exp\left(2 \times x_{it} + \mu_{it}\right)\right)$$

$$y_{it} \sim \text{Negative Binomial}\left(\exp\left(2 \times x_{it} + \mu_{it}\right), \theta = 0.5\right)$$

$$y_{it} = \mathbf{1}\left\{2 \times x_{it} + \mu_{it} + \varepsilon_{it}\right\},\, \varepsilon_{it} \sim Logistic(0,1)$$

with $\mu_{it} \equiv \gamma_i^{individual}$ for the generation with individual fixed-effects only, and $\mu_{it} \equiv \gamma_i^{individual} + \gamma_t^{time}$ for the generation with both fixed-effects. Finally all variables x_{it}, $\gamma_i^{individual}$ and γ_t^{time} are generated according to independent draws from a normal law of mean 0 and standard-deviation 1.

We generate data sets of 10,000 observations, with 2,000 individual fixed-effects and 50 time fixed-effects, both randomly assigned to observations. As such the panel is "short" since there is on average 5 observations per individual fixed-effect.

Once the data is generated, we estimate the coefficient associated to x, with the appropriate likelihood model. The distribution of the estimates for 100 replications are reported in Figure (3).

As we can see, only the Logit likelihood function suffers – rather heavily – from the incidental parameter problem (IPP). The estimates are systematically higher than the coefficient used to generate the data. Meaning that even though the package can estimate efficiently Logit with fixed-effects, the user should carefully decide when such kind of model is appropriate. Further, we can see that including time fixed-effects only worsens this problem.

On the other hand the three other likelihood functions do not suffer from the IPP, as confirmed by the simulation, where we find that the estimates hover closely around the true coefficient. The presence of time fixed-effects does not change the pattern.

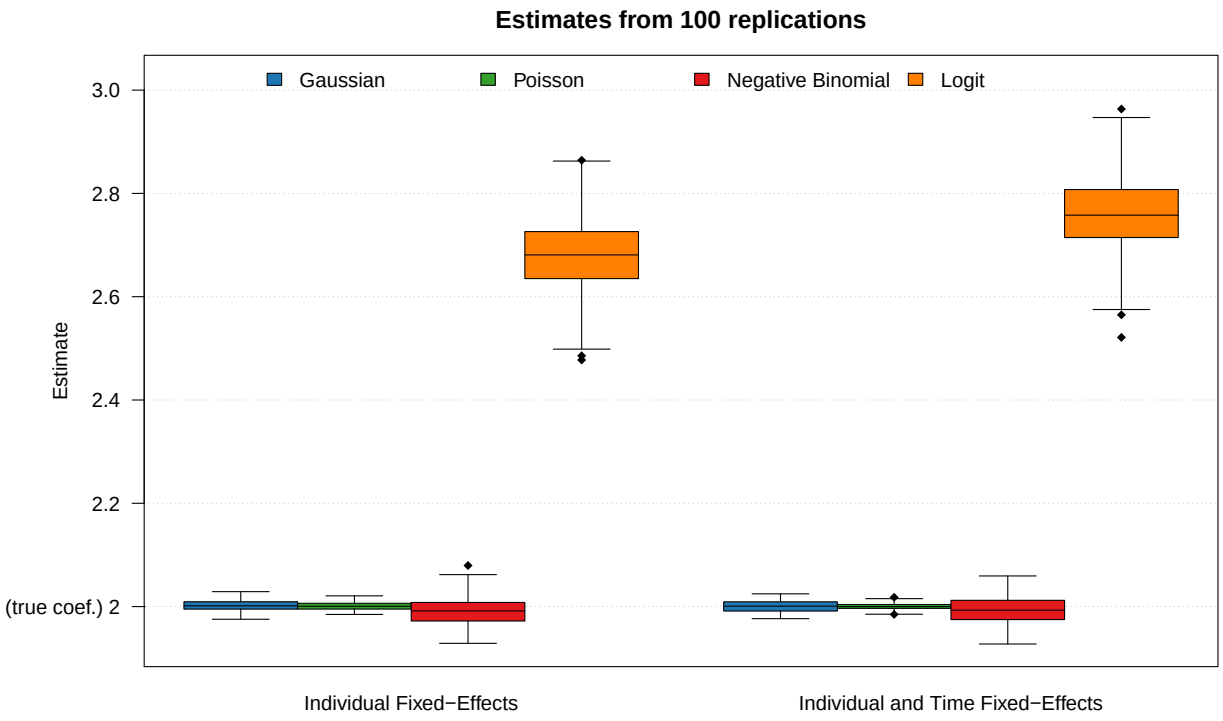

Figure 3: Monte Carlo experiment related to the incidental parameters problem.

B Obtaining the fixed-effects when there is only one cluster

In this section, we give the exact formulas and methods to obtain the fixed-effects when there is only one cluster. These values are in fact equal to the optimal increments used in Section 2 in Algorithm 1 to find the fixed-effects for any number of clusters.

Gaussian. The value of the cluster coefficients has the following closed-form:

$$\gamma_w = \frac{1}{\eta_w} \sum_{o \in \delta_w} \left(y_o - \beta' x_o \right),$$

with $\eta_w = \#\{\delta_w\}$ is the number of observations of the category w.

Poisson. The value of the cluster coefficients also has the following closed-form:

$$\gamma_w = \ln \left(\frac{\sum_{o \in \delta_w} y_o}{\sum_{o \in \delta_w} \exp\left(\beta' x_o\right)} \right).$$

Negative Binomial. Applying the first order condition of Equation (3) to the Negative Binomial log-likelihood, we have:

$$h_w(\gamma_w) \equiv \sum_{o \in \delta_w} y_o - \sum_{o \in \delta_w} (\theta + y_o) \frac{\exp(\gamma_w + \beta' x_o)}{\theta + \exp(\gamma_w + \beta' x_o)} = 0, \tag{14}$$

with θ the over-dispersion parameter.

Here the aim is to find γ_w^* such that $h_w(\gamma_w^*) = 0$. Unfortunately, there is no closed-form solution. To get the cluster coefficient γ_w^*, we apply a Newton-Raphson algorithm combined to dichotomy to ensure convergence in all circumstances. We now describe the initial bounds that we use to apply the dichotomy.

Define $\overline{f_w(\beta)} = \max\{\beta' x_o | o \in \delta_w\}$ and let $\underline{\gamma_w}$ be such that:

$$\sum_{o \in \delta_w} y_o - \sum_{o \in \delta_w} (\theta + y_o) \frac{\exp\left(\underline{\gamma_w} + \overline{f_w(\beta)}\right)}{\theta + \exp\left(\underline{\gamma_w} + \overline{f_w(\beta)}\right)} = 0. \tag{15}$$

Since the function $\exp(x) / (\theta + \exp(x))$ is increasing, we have

$$\sum_{o \in \delta_w} (\theta + y_o) \frac{\exp\left(\underline{\gamma_w} + \beta' x_o\right)}{\theta + \exp\left(\underline{\gamma_w} + \beta' x_o\right)} \leq \sum_{i \in \delta_m} (\theta + y_i) \frac{\exp\left(\underline{\gamma_m} + \overline{f_w(\beta)}\right)}{\theta + \exp\left(\underline{\gamma_w} + \overline{f_w(\beta)}\right)},$$

leading to

$$h_w\left(\underline{\gamma_w}\right) \geq 0.$$

Fortunately, $\underline{\gamma_w}$ has a closed-form. After manipulation and simplifications of Equation 15, we obtain:

$$\underline{\gamma_w} = \log\left(\sum_{o \in \delta_w} y_o\right) - \log(\eta_w) - \overline{f_w(\beta)},$$

where $\eta_w = \#\{\delta_w\}$ is the number of observations of the category w. Since the function h_w is strictly decreasing, $\underline{\gamma_w}$ is then a lower bound of γ_w^*. Applying a similar reasoning, we find the upper bound:

$$\overline{\gamma_w} = \log\left(\sum_{o \in \delta_w} y_o\right) - \log(\eta_w) - \underline{f_w(\beta)},$$

with $\underline{f_w(\beta)} = \min\{\beta' x_o | o \in \delta_w\}$ and $h_w(\overline{\gamma_w}) \leq 0$. We then know that $\gamma_w^* \in \left[\underline{\gamma_w}, \overline{\gamma_w}\right]$ and can apply dichotomy.

Logit. Applying the first order condition of Equation (3) to the Logit log-likelihood, we have:

$$h_w(\gamma_w) \equiv \eta_w^1 - \sum_{o \in \delta_w} \frac{\exp(\gamma_w + \beta' x_o)}{1 + \exp(\gamma_w + \beta' x_o)} = 0,$$

where $\eta_w^1 = \sum_{o \in w} y_o$ is the number of observations taking value 1. We need to find γ_w^* such that $h_w(\gamma_w^*) = 0$ but there is no closed-form solution to this problem.

Applying the exact same reasoning as for the Negative Binomial likelihood, we find the following lower and upper bounds:

$$\underline{\gamma_w} = \log\left(\eta_w^1\right) - \log\left(\eta_w^0\right) - \overline{f_w(\beta)},$$
$$\overline{\gamma_w} = \log\left(\eta_w^1\right) - \log\left(\eta_w^0\right) - \underline{f_w(\beta)},$$

with $\eta_w^0 = \eta_w - \eta_w^1$ is the number of observations taking value 0. Using these bounds we know that $\gamma_w^* \in \left[\underline{\gamma_w}, \overline{\gamma_w}\right]$, we then apply a Newton-Raphson algorithm combined to dichotomy.

C Details on the data and code used in the non-linear application

Data. I use data on granted patents issued by the European Patent Office from 1985 to 2003. I select only inventors having at least one patent in the field of "organic chemistry" and whose residence is in the French region of "Ile-de-France". In total the data consists of 2,621 unique patents and 1,633 inventors having at least one citation over the period.

Code. The centrality defined by Equation (12) can be computed with the following code:

```
centrality = function(lambda, alpha, beta){
# Global variables:
# - G: the network matrix
# - base: the data we use for the estimation
n = nrow(G)
# Computation of the degree (minimum is set to 1)
degree = pmax(rowSums(G), 1)
# we divide gij by the degree at power beta
G_d = t(G/(degree^beta))
# We compute the centrality iteratively
C = C_old = rep(1, n)
maxDiff = Inf # stopping criterion
while(maxDiff > 1e-5){
C = 1 + lambda * G_d %*% C_old^alpha
maxDiff = max(abs(C - C_old))
```

```
C_old = C
}
# We return it in the same order as the main data
res = C[base$id, ]
return(res)
}
```

Note that the matrix representing the network `G` is such that it contains the information on networks at all periods and each row must correspond to an inventor-year. The vector `base$id` is an unique (character) identifier for each inventor-year of the data, and the row names of matrix `G` must be in a similar format.