groupedpaneldatamodels: A Python Library for Grouped Fixed and Interactive Effects Models

Micha den Heijer

(2690717)

June 25, 2025

Master Thesis Econometrics and Operations Research

Thesis committee:

Dr. Ignace de Vos (supervisor)

Dr. TBD (co-reader)

Abstract

Grouped Panel Data Models have many advantages over fully homogeneous or fully heterogeneous panel data models. [Examples we found in the results]. This thesis creates the `groupedpaneldatamodels`-package, creating the first Python-package to implement any Grouped Panel Data Model. The Python-package has been thoroughly verified and has been shown to be implemented properly. [Sentence about the results]

Keywords: Grouped Panel Data Models, Fixed Effects, Interactive Effects, Python package.

Contents

1 Introduction

Panel Data Models have been gaining traction in Econometrics. One of the main books on Panel Data Models, Baltagi (2021, p. vii), believes that part of the rise of these models is caused by the availability of software packages in programming languages such as Python, R, and Stata. Research has shown that the availability of models in programming languages can often increase the usage of these models, as Choodari-Oskooei and Morris (2016) reports that there are many well-known examples of model usage after a good and stable software implementation was made, additionally the lack of software implementations can in many cases be a key barrier to wider adoption (Anselin, 2010; Pullenayegum et al., 2015; Grayling and Wheeler, 2019). Notably, Wahlquist et al. (2018) reported that just providing replication code for new statistical models doubled the citation rate for freely accessible articles[1.1].

Panel Data Models have already existed for quite a while, specifically the Fixed Effects models, as the first papers on these models were published a while ago (Mundlak, 1961; Mundlak, 1978). A core assumption of the most Fixed Effects estimations is that the coefficients are partially homogeneous, implying that the individual effect stays constant over time, and that the slope coefficients are the same for each individual. Research has shown that individual effects are often not homogeneous over time and that shocks that effect many individuals slightly differently often occur (Giannone and Lenza, 2010; Carneiro et al., 2003; Cawley et al., 1997) and thus recently many researchers have been proposing alternative models that resolve this issue. One of the most well-known models proposing an alternative to the constant individual effects is the Interactive Fixed Effects model proposed by Bai (2009).

However, even for this model the assumption of complete homogeneity is made for the slope coefficients, thus assuming that the effect of the regressor is the same for each individual. A lot of research has already shown that in many cases this assumption does not hold (Pesaran and Smith, 1995; Holly et al., 2010; Dogan and Seker, 2016). The opposite, fully heterogeneous models, are difficult and almost impossible to estimate because of the incidental parameters problem, i.e., there are (almost) as many variables as datapoints.

Recently, many models have been introduced that use a middle ground. Well-known papers as Bonhomme and Manresa (2015), Ando and Bai (2016), and Bonhomme, Lamadon, et al. (2022), have introduced the concept of Grouped Panel Data Models, where coefficients may differ between individuals, however there are a limited number of groups of differing coefficients, making it much easier to estimate those values. Many of the models that have been introduced for these papers have already seen some usage, as for example at the time of writing Bonhomme and Manresa (2015) has been cited 617 times[1.2], and their usage appears to be increasing over time. Just this model has been used to link the effect of debt on different types of countries (Gómez-Puig et al., 2022), to link carbon emissions to income inequality (Grunewald et al., 2017), and it has been used to show that income risk is quite high in Spain and that this risk is higher for younger and poorer people (Arellano et al., 2022).

Many other Grouped Panel Data Models have also been introduced, such as Ando and Bai (2016), Su, Shi, et al. (2016), Su and Ju (2018), Ke et al. (2016), Mehrabani (2023), and Mugnier (2024) and many more. However, most of the models these papers have introduced do not have software packages available to estimate them and there has been to our knowledge not been any

[1.1] This paper specifically looked at the field of biostatistics, but the point still stands

[1.2] According to Google Scholar

publicly available Python package that implements one of these models. This is odd as Python is one of the most commonly used programming languages within Econometrics (Danielsson and Aguirre, 2020). There are some software implementations available that implement some of these models in other programming languages, e.g., the *PMDIF* R-package by Ando and Fayad (2022), implements a model somewhat similar to the one proposed by Ando and Bai (2016), and the *classifylasso* Stata-package by Huang et al. (2024) implements the model introduced by Su, Shi, et al. (2016). However, none are available in Python, and many of these packages are not complete, either estimating a different model or only estimating the parameters without standard errors. This may limit the usage of these models, as creating a correct software implementation of these models can be quite challenging. For most models is it often also quite unclear what the quality of the inference that is derived by the paper is, as many papers such as Ando and Bai (2016), Su, Shi, et al. (2016), and Su and Ju (2018) did not include any results from a simulation study on the accuracy of their standard errors.

This thesis aims to fill thise gaps by answering the research question: "How can Grouped Panel Data Models, specifically Grouped Fixed, and Interactive Effects models effectively be incorporated in a Python package and what is the performance of this package?". To answer this question this thesis is split up into a few different sections. Section 2 examines Grouped Panel Data Models, it introduces their origin, examines the two main groups (Fixed and Interactive Effects models), and compares the different algorithms that have been used to implement these models. Section 3 looks into the already existing software implementations of Grouped Panel Data Models. Section 4 summarizes the main challenges that can be experienced when writing econometric models into software. Section 5 then summarizes how the proposed package can be used; Section 6 states some method to ensure correctness and numerical verifiability of these implementations and Section 7 does a simulation study of this package with a performance evaluation. Finally, Section 8 has the concludes the current work.

2 Grouped Panel Data Models

Before introducing the Python-package developed in this paper, first a quick introduction on Panel Data (specifically Fixed Effects and Interactive Effects) is given. After that the different models that have been implemented in the package are introduced, accompanied by their algorithms and the justifications of each model. Finally, doing inference based on these models is discussed.

2.1 Panel Data Models

Before diving into Grouped Panel Models, we quickly examine Panel Data Models first and specifically focus on Fixed and Interactive Fixed Effects models. This section is mostly focused on the most relevant Panel Data Models for this paper. Much more complete books about Panel Data Models are Hsiao (2014) and Baltagi (2021), which this short introduction is based on.

Panel Data Models include data observed over multiple units over multiple periods and models them simultaneously (Hsiao, 2014). A well-known example of Panel Data is the Panel Study of Income Dynamics (Daumler et al., 2025), this dataset has tracked many thousands of individuals and families since 1968 in the United States. It tracks income, employment activity, health and many other statistics and was used to understand income dynamics, the effects of poverty and much more. This data set is mainly modeled using Fixed Effects and other Panel Data Models. Panel Data models have many advantages (Hsiao, 2014, pp. 4–10), since they often provide more accurate inference, are good at emulating dynamic relationships, controlling for omitted variables and can (in some cases) simplify computation and statistical inference.

One of the most basic Panel Data Models is the **Fixed Effects Model**, as introduced by Mundlak (1978), this model can be written as in Equation 2.1,

$$y_{it} = x_{it}'\beta + \alpha_i + \varepsilon_{it}, \quad i \in \{1, ..., N\} \tag{2.1}$$

$$t \in \{1, ..., T\} \tag{2.2}$$

where α_i is a scaler constant which represents the individual effect of the individual i, β is a $K \times 1$ vector representing the effect of representing the marginal effect of each regressor x_{it} and ε_{it} is the i.i.d. error with mean zero. To estimate this model, first β is estimated by running a standard OLS estimation on the demeaned variables of the model[2.1]. Then α_i could be recovered as follows, $\hat{a}_i = \bar{y}_i - \bar{x}_i'\hat{\beta}$ (Hsiao, 2014, pp. 34–39).

However, heterogeneity can also occur over time, there are many different papers written on how to deal with this heterogeneity. One of the most well-known models that deals with heterogeneity is introduced by Bai (2009), which introduced the **Interactive Fixed Effects Model**. This paper proposed the following model,

$$y_{it} = x_{it}'\beta + \lambda_i' F_t + \varepsilon_{it}, \quad i \in \{1, ..., N\}, \tag{2.3}$$

$$t \in \{1, ..., T\}, \tag{2.4}$$

where F_t is a vector of r unobserved common factors that differ over time and λ_i is a vector of factor loadings that represent how each individual unit responds to these factors. Thus ensuring

[2.1] The demeaned version of the model is defined as, $y_{it} - \bar{y}_i = (x_{it} - \bar{x}_i)'\beta + u_{it}$.

that this model allows for unobserved heterogeneity over time and over units (and additionally models cross-sectional dependence). To ensure that the model remains identifiable, the following restriction is imposed $F'F/T = I_r$, where $F = [F_1, F_2, ..., F_T]$.

This model is estimated using a Principle Components (PC) approach, Bai (2009) specifically proposes an iterative estimation method that estimates β first using Least Squares and then estimates the factors and factor loadings from the residuals by estimating the eigenvectors using Singular Value Decomposition. This procedure is repeated until convergence is reached[2.2].

Often these models need to be modified such that the slope coefficients and individual effects of these models are homogeneous (over individuals and time). As was stated in the introduction, this assumption is often not justified and this can lead to incorrect estimations. However, often this assumption is required to avoid the incidental parameters problem as discovered by Nickell (1981). This problem states that for short dynamic panels, i.e. panels where T is low, the estimation procedure produces biased estimates for α_i even as $N \to \infty$, as α_i always remains correlated with y_i. This assumption and problem are some of the biggest challenges for Panel Data Models as stated by Hsiao (2014, pp. 10–13). Grouped Panel Data Models have been created to solve both these problems.

2.2 Grouped Models

The first paper to introduce such a model was Sun (2005). The paper argued that while at the time Panel Data Models often have to assume to be homogeneous, assuming the opposite of homogeneity, complete heterogeneity, often introduces a lot of other challenges, as were stated previously. Therefore, instead of assuming either complete homogeneity or complete hetrogeneity, Sun (2005), proposes a middle ground, where there are a certain number of groups with unknown group membership, where within each group the parameters are homogeneous, between the groups they are heterogeneous. An example of such a model could be the following,

$$y_{it} = x'_{it}\beta_{g_i} + \alpha_{g_i} + \varepsilon_{it}, \quad \begin{matrix} i \in \{1, ..., N\} \\ t \in \{1, ..., T\} \end{matrix} \tag{2.5}$$

where $g_i \in \{1, ..., G\}$ represents is the grouping individual i is in and G represents the number of groups, and thus β_{g_i} represents the coefficient value of group g_i is in.

Sun (2005), argues that there could some theoretical justifications for grouped parameters, it argues that different model parameters could be caused by different type of steady states that may have been caused by different initial conditions. For example it has been shown that economic growth of different countries often converge around common groups of parameters (Sun, 2002). Hahn and Moon (2010) argues that there are often just a finite number of steady states, because many Game Theory games tend to have a finite number of steady states. Sun (2005) also justifies that splitting up a sample based on other pre-specified variables can often quite arbitrary, and having some sort of predefined algorithm can remove this arbitrariness.

The advantage of these Grouped Panel Data Models is that it solves the incidental parameters problem. As when we let G be fixed, then even when T grows very slowly (for example for short panels), letting $N \to \infty$, still ensures that the bias of the parameters goes to zero (Hahn and Moon, 2010).

[2.2] The paper actually proposes multiple methods for estimating the model, though this method is generally preferred

Recently, Grouped Panel Data Models have been gaining some traction, and thus many papers presenting new models have been proposed. All these models roughly fall into two groups: Grouped Fixed Effects Models and Grouped Interactive Fixed Effects Models.

2.3 Grouped Fixed Effects Models

Many different panel data models have been proposed that introduce some sort of Grouped Fixed Effects Model. Most models are mostly based on two techniques: a clustering-based methods and penalization-based methods. Both techniques are quite different, the clustering-based technique computes the objective value of each clustering that is considered and then stores the best one, whereas the penalization-based techniques applies a penalization to the likelihood function which is then optimized.

2.3.1 Clustering-Based Method

The most well-known and most cited paper in this field is Bonhomme and Manresa (2015), which proposed a clustering-based estimation method. The paper proposes the following model,

$$y_{it} = x_{it}'\beta + \alpha_{g_i,t} + \varepsilon_{it}, \quad \begin{matrix} i \in \{1, ..., N\} \\ t \in \{1, ..., T\} \end{matrix} \tag{2.6}$$

which allows for heterogeneity over (grouped) individuals and heterogeneity over time. Additionally the paper proposed another model which also allows for heterogeneity over (grouped) individuals over the coefficients[2.3], which is the main model that will be considered in this paper,

$$y_{it} = x_{it}'\beta_{g_i} + \alpha_{g_i,t} + \varepsilon_{it}, \quad \begin{matrix} i \in \{1, ..., N\} \\ t \in \{1, ..., T\} \end{matrix} \tag{2.7}$$

To estimate this model the following minimization problem needs to be solved,

$$Q_{\text{BM}}(\beta, \alpha, g) = \sum_{i=1}^{N} \sum_{t=1}^{T} \left(y_{it} - x_{it}'\beta_{g_i} - \alpha_{g_i,t}\right)^2 \tag{2.8}$$

$$(\hat{\beta}, \hat{\alpha}, \hat{g}) = \underset{(\beta,\alpha,g) \in B \times A^{GT} \times \Gamma_G}{\text{argmin}} Q_{\text{BM}}(\beta, \alpha, g), \tag{2.9}$$

where B, A^{GT} and Γ_G are the parameter spaces for each parameter respectively, and where $g = \{g_1, ..., g_N\}$ which denotes the specific grouping for each individual i. As this minimization problem is very difficult to solve directly, the paper suggested Algorithm 1 to solve this problem.

This method splits up the optimization problem into two different problems, first a clustering problem and then a standard OLS problem. Since each problem is relatively easy solve, this algorithm is quite easy to solve. However, since this is a non-convex optimization problem, many iterations need to be run to find the true optimum, until convergence is reached, where τ denotes the acceptable tolerance.

[2.3] These models will be the main focus of this thesis. Additionally this paper also proposed a unit-specific model as defined in Appendix A.

Algorithm 1 Bonhomme and Manresa (2015)

procedure BONHOMMEMANRESA(y, x, τ iterations)
  for k in iterations **do**
    Generate some starting values for $\hat{\beta}^{(0)}, \hat{\alpha}^{(0)}$, e.g. estimate a starting value based on a part of the sample
    Set iteration number $j = 1$
    while $\|Q_{\text{BM}}(\hat{\alpha}^{(j)}, \hat{\beta}^{(j)}, \hat{g}^{(j)}) - Q_{\text{BM}}(\hat{\alpha}^{(j-1)}, \hat{\beta}^{(j-1)}, \hat{g}^{(j-1)}\| > \tau$ **do**
      Compute groupings for all $i \in \{1, ..., N\}$

$$\hat{g}_i^{(j)} = \underset{g \in \{1,...,G\}}{\operatorname{argmin}} \sum_{t=1}^{T} \left(y_{it} - x_{it}'\beta_g - \alpha_{gt}\right) \tag{2.10}$$

      Compute $\alpha_g^{(j)}$ for all $g \in \{1, ..., G\}$

$$\hat{\alpha}_{gt}^{(j)} = \frac{1}{\sum_{i=1}^{N} \mathbf{1}\{\hat{g}_i^{(j)} = g\}} \sum_{i=1}^{N} \mathbf{1}\{\hat{g}_i^{(j)} = g\}(y_i - x_{it}'\hat{\beta}_{\hat{g}_i}^{(j)}) \tag{2.11}$$

      Compute $\beta_g^{(j)}$ for all $g \in \{1, ..., G\}$

$$\hat{\beta}_g^{(j)} = \underset{\beta_g}{\operatorname{argmin}} \sum_{i=1}^{N} \sum_{t=1}^{T} (y_{it} - \hat{\alpha}_{\hat{g}_i^{(j)},t} - x_{it}'\beta_g)^2 \tag{2.12}$$

      Update iteration number $j+ = 1$
    end while
    If the objective value $Q_{\text{BM}}(\hat{\beta}^{(j)}, \hat{\alpha}^{(j)}, \hat{g}^{(j)})$ store the parameters as $\hat{\alpha} = \hat{\alpha}^{(j)}, \hat{\beta} = \hat{\beta}^{(j)}, \hat{g} = \hat{g}^{(j)}$, if not ignore the results
  end for
  return $\hat{\alpha}, \hat{\beta}, \hat{g}$
end procedure

Since this is not ideal, the paper proposed an alternative Variable Neighborhood Search (VNS) algorithm as can be found in Appendix A.2, this algorithm does not need many repeated iterations to find the true optimum, however it is often much slower than the previous algorithm.

To estimate the number of groups information criteria can be used, the paper introduces the following BIC criterion to estimate the number of groups. Usually the BIC criterion is defined as follows,

$$\text{BIC} = -2 \log L + k \log n, \tag{2.13}$$

where L is the maximized log-likelihood, k is the number of parameters in the model and n is the number of observations in the model (Schwarz, 1978). In the supplement of Bonhomme and Manresa (2015) the following BIC criterion is derived,

$$\text{BIC}(G) = \frac{1}{NT} \sum_{i=1}^{N} \sum_{t=1}^{T} (y_{it} - \hat{\alpha}_{\hat{g}_i,t} - x_{it}'\hat{\beta}_{\hat{g}_i})^2 + \hat{\sigma}_\varepsilon^2 \frac{GT + N + K}{NT} \log NT, \tag{2.14}$$

where $\hat{\sigma}_\varepsilon^2$ is defined as the variance of the residuals. In the package a variant for this BIC is implemented which is just a rescaled version of this BIC criterion is implemented, as can be seen in Section 2.6.

The paper proposes multiple techniques to compute the standard errors. There is the standard large-N and T inference, though also the paper also proposes a much more complex large N, fixed T inference. Finally, it proposes a Bootstrap method, similarly to the method proposed in Section 2.5. The standard large-N, T inference method is defined as follows,

$$\widehat{\text{Var}}(\hat{\alpha}_{gt}) = \frac{\sum_{i:\hat{g}_i=g} \hat{v}_{it}^2}{\hat{N}_g}, \tag{2.15}$$

$$\widehat{\text{Var}}(\hat{\beta}_g) = \frac{1}{\hat{N}_g T} \widehat{\Sigma}_{\beta g}^{-1} \widehat{\Omega}_{\beta g} \widehat{\Sigma}_{\beta g}^{-1}, \text{ where} \tag{2.16}$$

$$\widehat{\Sigma}_{\beta g} = \frac{1}{\hat{N}_g T} \sum_{i:\hat{g}_i=g} \sum (x_{it} - \bar{x}_{\hat{g}_i t})(x_{it} - \bar{x}_{\hat{g}_i t})', \tag{2.17}$$

$$\widehat{\Omega}_{\beta g} = \frac{1}{\hat{N}_g T} \sum_{i:\hat{g}_i=g} \sum_{t=1}^{T} \sum_{s=1}^{T} \hat{v}_{it} \hat{v}_{is} (x_{it} - \bar{x}_{\hat{g}_i t})(x_{it} - \bar{x}_{\hat{g}_i t})', \tag{2.18}$$

where $\hat{v}_{it}$ represents the residuals, and is defined as $\hat{v}_{it} = y_{it} - x_{it}' \beta_{\hat{g}_i} - \hat{\alpha}_{\hat{g}_i, t}$. Additionally, $\hat{N}_g$ is the estimated number entries in group g, i.e. $\hat{N}_g = \sum_{i=1}^{N} \mathbf{1}\{\hat{g}_i = g\}$. This method remains valid even as for there is serial correlation between the residuals, other methods of estimating $\widehat{\Omega}_{\beta g}$ need to be used if the errors are related to each other in some sort of other way. Large N, fixed T inference is outside of the scope of this paper, though may be implemented in a future version of the package.

2.3.2 Penalization-Based Method

An alternative method for a Fixed Effects Model was proposed by Su, Shi, et al. (2016), which took an alternative path to estimate the groupings. It introduced a Classifier-Lasso method, where the estimator is penalized to enforce groupings between the parameters. This is quite a generalizable method and can easily be applied to many different problems. Additionally, by using this technique it remains possible to use simple optimization methods. An example of a model that can be estimated, as was proposed by the paper itself, is the following,

$$\begin{aligned} y_{it} = x_{it}' \beta_{g_i} + \alpha_i + \varepsilon_{it}, \quad & i \in \{1, ..., N\} \\ & t \in \{1, ..., T\} \end{aligned} \tag{2.19}$$

this model contains an individual effect α_i, and grouped coefficients β_{g_i}. To estimate this model the following minimization problem is introduced,

$$(\hat{b}, \hat{\alpha}, \hat{\beta}) = \underset{(b,\alpha,\beta) \in B^N \times A^{GT} \times B^G}{\text{argmin}} \sum_{i=1}^{N} \sum_{t=1}^{T} \left(y_{it} - x_{it}' b_i - \alpha_i\right)^2 + \frac{\kappa}{N} \sum_{i=1}^{N} \prod_{\gamma=1}^{G} \|b_i - \beta_\gamma\|_2, \tag{2.20}$$

where a new parameter b_i is introduced. This represents the coefficient of the individual i, without adding a penalization, this model would just represent a fully heterogeneous model. Additionally γ is introduced, which represents the group number. By introducing the penalization

$\frac{\kappa}{N}\sum_{i=1}^{N}\prod_{\gamma=1}^{G}\|b_i-\beta_\gamma\|$, the minimization procedure is incentivized (for a κ large enough) to group these coefficients together to the values β_g, which represents the β coefficient of group γ. Since the penalization contains a product, the penalization term for an individual i is minimized when $b_i \in \{\beta_\gamma, \gamma \in G\}$[2.4]. To retrieve g_i, which represents grouping each individual i belongs, we can simply state that $g_i = \sum_{\gamma=1}^{G}\gamma \mathbf{1}\{b_i = \beta_\gamma\}$.

In the paper a simple generalization of this method was proposed,

$$Q^{\kappa}_{\text{SSP}}(b,\alpha,\beta) = \frac{1}{NT}\sum_{i=1}^{N}\sum_{t=1}^{T} f(y_{it},x_{it};b_i,\alpha_i) + \frac{\kappa}{N}\sum_{i=1}^{N}\prod_{\gamma=1}^{G}\|b_i-\beta_\gamma\|_2, \tag{2.21}$$

where $f(\cdot)$ is a convex function that needs to be minimized to estimate the model. In most cases it is generally advantageous to use profiling, which estimates α_i first, by demeaning first[2.1]. Changing the objective function to,

$$Q^{\kappa}_{\text{SSP}}(b,\beta) = \frac{1}{NT}\sum_{i=1}^{N}\sum_{t=1}^{T} f(y_{it},x_{it};\hat{\alpha}_i(b),b_i) + \frac{\kappa}{N}\sum_{i=1}^{N}\prod_{\gamma=1}^{G}\|b_i-\beta_\gamma\|_2, \tag{2.22}$$

where $\hat{\alpha}(b) = \text{argmin}_\alpha \sum_{i=1}^{N}\sum_{t=1}^{T} f(y_{it},x_{it};b_i,\alpha_i)$. This paper does not consider implementing the generalization, though it does use the profiling, demeaning technique.

The optimization problem is not a simple convex problem that could easily be solved with an convex optimization algorithm such as the BFGS method (Broyden, 1970; Fletcher, 1970; Goldfarb, 1970; Shanno, 1970). However, it is conditionally convex for the case that β_γ is fixed for all γ except one, meaning that we can optimize over each of these separately using a convex optimizers.

$$Q^{\kappa}_{\text{SSP},\gamma,k}(b,\beta_\gamma) = Q^{\kappa}_{\text{SSP}}(b,[\hat{\beta}_1^{(k)},...,\hat{\beta}_{\gamma-1}^{(k)},\beta_\gamma,\hat{\beta}_{\gamma+1}^{(k-1)},...,\hat{\beta}_G^{(k-1)}]) \tag{2.23}$$

Therefore the paper proposes Algorithm 2, where $f(\cdot)$ is defined as the squared error as was proposed in the initial model in Equation 2.20.
The algorithm cleverly makes use of the conditionaly convexity of the problem, by only optimizing over a part of the problem, preferably with a convex optimizer. For this algorithm it is important that a reasonable value is chosen for κ. Preferably κ should be chosen using an Information Criterion. The generalized information criteria defined in Section 2.6, could be used as a valid information criterion, as it is just a rescaled version of the same information criterion.

The standard errors on $\hat{\beta}_g$ are computed similarly as was written in Equation 2.35[2.5] and the standard errors individual effects α_i are computed as follows,

$$\widehat{\text{Var}}(\hat{\alpha}_i) = \frac{1}{T}\sum_{t=1}^{T}\hat{v}_{it}^2, \tag{2.26}$$

where $\hat{v}_{it} = y_{it} - x_{it}'\hat{\beta}_{\hat{g}_i} - \hat{\alpha}_i$, where $\hat{g}_i = \text{argmin}_\gamma\|b_i-\beta_\gamma\|$.

[2.4] I.e., when b_i is one of the grouped coefficients β_γ

[2.5] Where the residuals are computed as stated below.

Algorithm 2 Su, Shi and Phillips Algorithm

procedure SUSHIPHILLIPS(y, x, κ, τ, iterations)
    Start with initial values for $\hat{b}_i^{(0)}$ and $\hat{\beta}^{(0)}$, s.t. $b_i \neq \beta_\gamma \ \ \forall i, \gamma$
    k = 1
    while $|Q_{\text{SSP}}(\hat{b}^{(k)}, \hat{\beta}^{(k)}) - Q_{\text{SSP}}(\hat{b}^{(k-1)}, \hat{\beta}^{(k-1)})| > \tau$ **do**
        for γ in G **do**
            Optimize for b and β_γ, with starting value $\hat{b}^{(k)}$ and $\hat{\beta}^{(k)}$

$$\text{argmin}_{b,\beta_\gamma} Q^{\kappa}_{\text{SSP},\gamma,k}(b, \beta_\gamma) = \tag{2.24}$$

$$\text{argmin}_{b,\beta_\gamma} \sum_{i=1}^{N}\sum_{t=1}^{T} \left(y_{it} - x_{it}' b_i - \alpha_i\right)^2 \tag{2.25}$$
$$+ \frac{\kappa}{N} \sum_{i=1}^{N} \|b_i - \beta_\gamma\| \prod_{j=1}^{\gamma-1} \|b_i - \hat{\beta}_j^{(k)}\| \prod_{j=\gamma+1}^{G} \|\hat{b}_i^{(\gamma-1)} - \hat{\beta}_j^{(k-1)}\|$$

        end for
        k += 1
    end while
end procedure

2.3.3 Alternative Models

There are many different models that have been based on these two models and with some simple modifications many of these other models could have been implemented. For example Bonhomme, Lamadon, et al. (2022) proposed a generalized version of the Bonhomme and Manresa (2015), which supports many non-linear models.

Additionally, there are many extensions that have been proposed to Su, Shi, et al. (2016). For example, Wang et al. (2018) suggests using an alternative model based on the CARDS (Clustering Algorithm in Regression via Data-driven Segmentation) penalty, this does not require a predefined number of groups to be defined and allows for the number of groups $G \to \infty$. Similarly, Mehrabani (2023) suggest an alternative penalty PAFGL (Pairwise Adaptive Group-Fused Lasso) penalty, which is fully convex and thus much easier and faster to compute.

These models have not (yet) been implemented in the Python-package due to time constraints. These constraints forced the choice to implement the models from the most popular and most cited models first, however these three models are good candidates for future expansions of the package.

2.4 Grouped Interactive Fixed Effects Models

Much of the most recent research has gone into Grouped Interactive Fixed Effects, as Interactive Fixed Effects models nest Fixed Effects models. This means that these models allow richer forms of unobserved heterogeneity, as they also allow to easily model much more cross-sectional heterogeneity and serial dependence.

As all Grouped Interactive Fixed Effects models are extensions of Grouped Fixed Effects models, there are once again two main types of models: clustering-based models and penalization-based models.

2.4.1 Clustering-Based Model

The clustering-based method proposed in Ando and Bai (2016), is very similar to and based on the model proposed by Bonhomme and Manresa (2015). The model the paper introduces is defined as follows[2.6].

$$y_{it} = x'_{it}\beta_{g_i} + f'_{g_i,t}\lambda_{g_i,i} + \varepsilon_{it}, \quad \begin{matrix} i \in \{1, ..., N\} \\ t \in \{1, ..., T\} \end{matrix} \tag{2.27}$$

where each group has their own factor $F_{g_i,t}$ and each individual has their own factor loading $\lambda_{g_i,i}$. To ensure that this model is identifiable [TBD]. Notably, in the specific case that $\lambda_{g_i,i} = 1 \ \forall g_i, i$, this model can be easily rewritten to the Grouped Fixed Effects model of the previous subsection. To estimate this model the following minimization procedure needs to be solved,

$$Q_{\text{AB}}(\beta, f, \lambda, g) = \sum_{i=1}^{N}\sum_{t=1}^{T}\left(y_{it} - x'_{it}\beta_{g_i} - f'_{g_i,t}\lambda_{g_i,i}\right)^2 \tag{2.28}$$

$$(\hat{\beta}, \hat{f}, \hat{\lambda}, \hat{g}) = \underset{(\beta,f,\lambda,g)\in B\times F^{GT}\times\Lambda^{GN}\times\Gamma_G}{\text{argmin}} Q_{\text{AB}}(\beta, f, \lambda, g) \tag{2.29}$$

where F^{GT} and Λ^{GN} are the parameter spaces of these coefficients. The algorithm proposed by Ando and Bai (2016) is quite similar to the algorithm to the algorithm proposed by Bonhomme and Manresa (2015), with an additional step to compute the factors and the factor loadings.
Notably the algorithm proposed above is slightly different than the algorithm proposed in the paper itself, where it was stated that for large N and T no multiple iterations are needed. However, to retain the ability to estimate for non-large N, T, these multiple iterations originally proposed in Bonhomme and Manresa (2015) are retained. Additionally, the paper proposed adding an additional penalty term to deal with High Dimensional Panel Data Models, this is implemented in the package, however due to the additional complexity, this is described in the Appendix B.

To decide the number of groups that are required, the paper suggests using Information Criteria as implemented in Section 2.6. Inference is also quite similar to the previous methods, however the paper does not derive any standard errors for the factor loadings $\lambda_{g_i,i}$ or the factors F_t. It does derive the standard errors for β, however notation the original paper uses is quite complex, thus a much simpler notation is introduced in this paper[2.7]

$$\widehat{\text{Var}}(\hat{\beta}_g) = \frac{1}{\hat{N}_g T}\widehat{\Sigma}_{\beta g}^{-1}\widehat{\Omega}_{\beta g}\widehat{\Sigma}_{\beta g}^{-1}, \text{ where} \tag{2.35}$$

$$\widehat{\Sigma}_{\beta g} = \frac{1}{\hat{N}_g T}\sum_{i:\hat{g}_i=g}\sum(\tilde{x}_{it} - \overline{\tilde{x}}_{\hat{g}_i t})(\tilde{x}_{it} - \overline{\tilde{x}}_{\hat{g}_i t})', \tag{2.36}$$

$$\widehat{\Omega}_{\beta g} = \frac{1}{\hat{N}_g T}\sum_{i:\hat{g}_i=g}\sum_{t=1}^{T}\sum_{s=1}^{T}\hat{v}_{it}\hat{v}_{is}(\tilde{x}_{it} - \overline{\tilde{x}}_{\hat{g}_i t})(\tilde{x}_{it} - \overline{\tilde{x}}_{\hat{g}_i t})', \text{ where} \tag{2.37}$$

$$\hat{v}_{it} = y_{it} - x'_{it}\beta_{\hat{g}_i} - \hat{\lambda}_{\hat{g}_i,i}\hat{F}_{\hat{g}_i,t}, \tag{2.38}$$

[2.6] The paper also introduces a model without grouped coefficients, this model is also implemented in the package, and is defined in Appendix B.

[2.7] Technically, this is asymptotically equivalent, but not exactly the same. For the samples sizes tested in this paper, $N >> T$, thus this should not be considered a problem.

Algorithm 3 Ando and Bai (2016)

procedure ANDOBAI(y, x, r, τ, iterations)
  for j in iterations **do**
    Generate some starting values for $\hat{\beta}^{(0)}, \hat{\alpha}^{(0)}$, e.g. estimate a starting value based on a part of the sample
    Compute initial factors and clusters as in Equation **??**
    Set iteration number $k = 1$
    while $|Q_{\text{AB}}(\hat{\beta}^{(k)}, \hat{f}^{(k)}, \hat{\lambda}^{(k)}, \hat{g}^{(k)}) - Q_{\text{AB}}(\hat{\beta}^{(k-1)}, \hat{f}^{(k-1)}, \hat{\lambda}^{(k-1)}, \hat{g}^{(k-1)})| > \tau$ **do**
      Compute groupings for all $i \in \{1, ..., N\}$

$$\hat{g}_i^{(k)} = \underset{g \in \{1,...,G\}}{\operatorname{argmin}} \sum_{t=1}^{T} \left(y_{it} - x_{it}'\beta_g - f_{g,t}'\lambda_{g,i}\right) \tag{2.30}$$

      Compute factors $F_{g,t}$ and factor loadings $\Lambda_{g,i}$ for all $g \in \{1, ..., G\}$ by computing the Principle Components of the residuals

$$\hat{w}_{it} = y_{it} - x_{it}'\hat{\beta}_{\hat{g}_i}^{(k)} \tag{2.31}$$

$$\hat{F}_g = eig(\hat{W}_g'\hat{W}_g)_r, \tag{2.32}$$

        where $\hat{W}_g$ are the stacked partial residuals $\hat{w}_{it}$ for which $\hat{g}_i = g$

$$\hat{\lambda}_{g_i,i}^{(k)} = F_{g_i} y_i / T \tag{2.33}$$

      Compute $\beta_g^{(k)}$ for all $g \in \{1, ..., G\}$

$$\hat{\beta}_g^{(k)} = \underset{\beta_g}{\operatorname{argmin}} \sum_{i=1}^{N} \sum_{t=1}^{T} (y_{it} - \hat{f}_{\hat{g}_i^{(k)},t} \hat{\lambda}_{\hat{g}_i^{(k)},i} - x_{it}'\beta_g)^2 \tag{2.34}$$

    end while
  end for
  If $Q_{\text{AB}}(\hat{\beta}^{(k)}, \hat{f}^{(k)}, \hat{\lambda}^{(k)}, \hat{g}^{(k)})$ is lower than the previous objective value, store $(\hat{\beta}, \hat{f}, \hat{\lambda}, \hat{g}) = (\hat{\beta}^{(k)}, \hat{f}^{(k)}, \hat{\lambda}^{(k)}, \hat{g}^{(k)})$
end procedure

2.4.2 Penalization-Based Model

The final model implemented in the paper is introduced by Su and Ju (2018) and uses the same C-Lasso method as introduced by Su, Shi, et al. (2016). The model proposed is almost similar to Ando and Bai (2016), as it is defined as,

$$y_{it} = x'_{it}\beta_{g_i} + F'_t\lambda_i + \varepsilon_{it}, \quad i \in \{1, ..., N\} \tag{2.39}$$

$$t \in \{1, ..., T\} \tag{2.40}$$

where the factors are not grouped anymore. Notably, when F_t has G components, it can be rewritten to the model described in the previous section. This means that the following model needs to be optimized to be estimated,

$$(\hat{\beta}, \hat{f}, \hat{\lambda}, \hat{g}) = \underset{(\beta,\alpha,g)\in B\times F^{GT}\times\Lambda^{GN}\times\Gamma_G}{\operatorname{argmin}} \sum_{i=1}^{N}\sum_{t=1}^{T}\left(y_{it} - x'_{it}\beta_{g_i} - f'_{g_i,t}\lambda_{g_i,i}\right)^2, \tag{2.41}$$

$$. \tag{2.42}$$

Similarly to Su, Shi, et al. (2016), to achieve the groupings a penalization technique Su and Ju (2018) uses a C-Lasso penalization technique. However, alternatively to Su, Shi, et al. (2016), it is not useful to insert the squared errors in the objective function, as optimizing that model will not take into account the restrictions on F_t. Therefore the paper proposes the following objective function that can be optimized,

$$\begin{aligned} Q_{\text{SJ}}(b, \beta) =& \frac{1}{T}\sum_{r=R+1}^{T}\mu_r\left(\frac{1}{N}\sum_{i=1}^{N}(Y_i - X_i b_i)(Y_i - X_i b_i)'\right) \\ &+ \frac{\kappa}{N}\sum_{i=1}^{N}\prod_{\gamma=1}^{G}\|b_i - \beta_\gamma\|_2 \end{aligned} \tag{2.43}$$

where $Y_i = [y_{i1}, ..., y_{iT}]$ and $X_i = [x_{i1}, ..., x_{iT}]$ and where $\mu_r(A)$ returns the r'th largest eigenvalue of the matrix A, and where R is the number of factors that this model includes. Each eigenvalue represents the strength of a common factor. Minimizing the sums of the smallest common factors than represents all the possible common factors that the model did not include in a common factor, essentially rewarding the optimizer to ensure that as much of the common factors are attributed to them. Since Su and Ju (2018) is based on the algorithm proposed by Su, Shi, et al. (2016), this problem is only conditionally convex. Therefore Algorithm 4 is proposed, which similarly to the algorithm of Su, Shi, et al. (2016) optimizes each of the convex blocks separately, which uses the following optimization function,

$$Q^{\kappa}_{\text{SJ},\gamma,k}(b, \beta_\gamma) = Q^{\kappa}_{\text{SJ}}(b, [\hat{\beta}^{(k)}_1, ..., \hat{\beta}^{(k)}_{\gamma-1}, \beta_\gamma, \hat{\beta}^{(k-1)}_{\gamma+1}, ..., \hat{\beta}^{(k-1)}_G]).$$

Algorithm 4 Su and Ju (2018)

procedure SUJU(y, x, κ, R, iterations)
  Start with initial values for $\hat{b}_i^{(0)}$ and $\hat{\beta}^{(0)}$, s.t. $b_i \neq \beta_\gamma \ \ \forall i, \gamma$
  k = 1
  while $|Q_{\text{SJ}}(\hat{b}^{(k)}, \hat{\beta}^{(k)}) - Q_{\text{SJ}}(\hat{b}^{(k-1)}, \hat{\beta}^{(k-1)})| > \tau$ **do**
    for γ in G **do**
      Optimize for b and β_γ, with starting value $\hat{b}^{(k)}$ and $\hat{\beta}^{(k)}$

$$\begin{aligned}
&\text{argmin}_{b,\beta_\gamma} Q^{\kappa}_{\text{SJ},\gamma,k}(b, \beta_\gamma) = \\
&\text{argmin}_{b,\beta_\gamma} \frac{1}{T} \sum_{r=R+1}^{T} \mu_r \left(\frac{1}{N} \sum_{i=1}^{N} (Y_i - X_i b_i)(Y_i - X_i b_i)' \right) \\
&\qquad + \frac{\kappa}{N} \sum_{i=1}^{N} \|b_i - \beta_\gamma\| \prod_{j=1}^{\gamma-1} \|b_i - \hat{\beta}_j^{(k)}\| \prod_{j=\gamma+1}^{G} \|\hat{b}_i^{(\gamma-1)} - \hat{\beta}_j^{(k-1)}\|
\end{aligned}$$

    end for
    k += 1
  end while
  Solve for the factors and compute the factor loadings

$$\left[\frac{1}{NT} \sum_{i=1}^{N} (Y_i - X_i \beta_{g_i})(Y_i - X_i \beta_{g_i})' \right] F = F V_{NT},$$

where $V_N T$ is a diagonal matrix of the R_0 largest eigenvalues

$$\lambda_i = \hat{F}'(Y_i - X_i \hat{\beta}_{g_i})/T$$

end procedure

The paper states explicitly that under the assumption of i.i.d. normal errors the information criteria stated in Section 2.6 can be used. Additionally, to do inference exactly the same standard errors as for Ando and Bai (2016) can be used for β.

2.5 Inference and Bootstrap

Each of the papers included a derivation for inference of the estimated parameters, however these are valid for large N, T, meaning that they may not be valid for all samples. Some papers (notably Bonhomme and Manresa (2015)) have derived alternative standard errors, for example for large N, but fixed T, however implementing these methods is quite complex and outside of the scope of this thesis.

Luckily, there are alternative methods to estimate the standard errors and confidence intervals of these models. The most well-known of these methods is the Bootstrap-method, which this package also includes. The advantage of this method is that it can just simple be implemented once, and then used for all models.

There are many different possible Bootstrap implementations, each with their own strengths and weaknesses. However, as each estimation procedure is quite expensive, a Bootstrap method that requires very few replications should be considered. This implies that percentile-based Bootstrap methods should not be considered, as they require way too many replications. Thus the

decision was made to implement a standard error, normal approximation non-parameteric Bootstrap technique, as is described in Algorithm 5, as was suggested by Efron and Tibshirani (1993, pp. 45, 168–169).

Algorithm 5 Bootstrap Standard Error and Confidence Interval Estimates

procedure BOOSTRAP($n_{\text{bootstrap}}$)
  Regularly estimate the model first, store estimates in parameter $\hat{\theta}$
  for i in n_{boostrap} **do**
    Generate a new sample of the data with replacement
    Estimate the model again on this sample, store estimates in coefficient $\hat{\theta}^{(i)}$
  end for
  Compute the standard errors $\widehat{\text{se}}_\theta = \text{Var}(\theta^{(i)})$
  Compute the confidence intervals $(\hat{\theta} - z_{1-\alpha/2} \cdot \widehat{\text{se}}_\theta, \hat{\theta} + z_{1-\alpha/2} \cdot \widehat{\text{se}}_\theta)$
end procedure

The advantage of this method is that we can get a good estimate of the standard errors, without having to rely on the assumptions made in inference (often large N and T). Some research has shown that doing more than 100 Bootstrap replications does not lead to improvements in the estimation of standard errors, thus this has been set as the default in the package (Goodhue et al., 2012).[2.8]

2.6 Group Selection and Information Criteria

Each model tends to include their an Information Criterion to allow a user to test the number of groups that need to be included. To simplify the implementation of these information criteria, a generalized formula of these criteria is implemented. Under the assumption that the residuals are i.i.d. and normally distributed the following information criteria can be defined,

$$\text{BIC} = n \log(\hat{\sigma}_\varepsilon^2) + k \log n, \tag{2.44}$$

$$\text{AIC} = n \log(\hat{\sigma}_\varepsilon^2) + 2k, \tag{2.45}$$

$$\text{HQIC} = n \log(\hat{\sigma}_\varepsilon^2) + 2k \log \log n, \tag{2.46}$$

where $\hat{\sigma}_\varepsilon^2$ is defined as the variance of the residuals, n (which in each of these models is NT for balanced panel data) the number of datapoints and k is the number of parameters in the model (Priestley, 1982; Manini et al., 2020; Lütkepohl, 2004).

Research has shown that each Information Criterion has its own advantages and performs best in different use cases. Research has shown that the AIC tends to be preferred when predictive performance needs to be optimized, the BIC tends to be preferred for model identification and the HQIC is some sort of balance in between these two Information Criteria (Sin and White, 1996).

[2.8] Further improvements could be made by implementing a Balanced Bootstrap method, as was suggested by Davidson et al. (1986).

3 Existing Software Packages for Grouped Panel Data Models

Grouped Panel Data Models have gained popularity since the publication of Bonhomme and Manresa (2015). While no Python-package implementing these methods have been created to our knowledge[3.1], other software has been created and implemented. This section gives a quick overview of the software packages that have been created that already implement Grouped Panel Data Models and suggest (potential) challenges for these packages.

Together with their paper, Bonhomme and Manresa (2015), published Stata program to replicate such that users could their findings, but also to use their model itself. Their model mainly programmed in FORTRAN, and is therefore quite fast[3.2], they have also implemented a Bootstrap estimation procedure for the standard errors. The code for the Stata library is however only available in compiled form, and thus it is quite difficult to create your own modifications to these models. Additionally, it appears that the compiled version of the code is only available for Windows computers. Bonhomme, Lamadon, et al. (2022) also published replication code, however this code has not been published in a package.

The *PMDIF* packages claims (Ando and Fayad, 2022) to have implemented the model proposed by Ando and Bai (2016). However, this code implements a slightly different model, namely the following model,

$$y_{it} = x'_{it}\beta_{g_i} + f'_{g_i,t}\lambda_{g_i,i} + \varepsilon_{it}, \quad \begin{array}{l} i \in \{1, ..., N\}, \\ t \in \{1, ..., T\}. \end{array} \tag{3.1}$$

This may still be a very useful model to implement, however it is not the same model as proposed in the paper. A big advantage of this package is that the code is written for Generalizable Linear Models (GLM), meaning that as long as you have the linking function, you can use this model for some sort of non-linear model.

Some of the penalized estimators have also been implemented in software libraries or packages. The *classifylasso* command, as implemented by Huang et al. (2024) is available for Stata and the *classo* package is available for R as implemented by Gao (2020). The *classifylasso* package seems to be slightly more developed as it is able to also estimate standard errors, confidence intervals and uses information criteria to automatically determine the number of groups. This makes this package quite easy to use and very complete, however the estimation procedure tends to be quite slow. The *classo* package has implemented the very fast ECOS optimizer, which may give it an advantage in estimation speed. However it does not automatically estimate the standard errors, thus requireing that users of this method manually implement some sort of Bootstrap method to estimate the standard errors.

The last package that is available is the *PAFGL* library that is available for R, as has been implemented by Haimerl (2024). This package implements the model specified in Mehrabani (2023), which has not been implemented in this version of the package. This package appears to be quite complete, providing the estimates for the coefficients, standard errors, information criteria. Notably it also supports endogeneous regressors, by supporting the Penalized Generalized

[3.1] A Python-package which creates some functions required by for the Grouped Fixed Effects estimator by Bonhomme, Lamadon, et al. (2022) has been published in Lamadon (2021). The package however does not fully provide an estimator, just some helper functions.

[3.2] Which is likely much faster than the same method implemented in this paper, when parallelism is disabled.

Method of Moments estimator suggested by Mehrabani (2023), which no other package (including this one) provides.

Each of these packages have their own advantages and disadvantages, however using compiled code or fast optimizers such as the Stata package of Bonhomme and Manresa (2015) and the *classo* library by Gao (2020) respectively can ensure that estimations are quick. Implementing a very complete package that has support for automatic group selection such as the *classifylasso* package can make usage very easy, which should be quite important when implementing a package.

4 Programming Considerations and Challenges

Before diving deep into the challenges the previous algorithms might face, it is important that the program remains and useful such that other users can also use and modify it. The best practices as described in Wilson et al. (2014), have (mostly) been used to ensure maintainability and ease of use. While this paper describes too many best practices to include in this thesis, some of the most important ones have been included. For example, a version control system is used, and all the code is written is the most high-level language as possible. However, unit-testing, where each part of the code is tested seperately, is not (yet) implemented due to the complexity of this task, though it would guarantee much more stable usage.

Hajivassiliou (2018) states some of the main computational challenges that often occur when implementing econometric models. There are four main challenges when implementing the models in this package: numerical stability, memory challenges, optimizer selection and finally performance. Each of these four challenges need to be solved appropriately, as incorrect implementations may give incorrect results, or even worse gives incorrect results for just some inputs.

4.1 Numerical Stability

Almost all computing is done using floating point numbers, these floating point numbers have a limited accuracy (Goldberg, 1991). This means that there are often (very) small rounding errors between the computations. When an algorithm is numerically unstable, these small errors can compound and these errors can dramatically increase. Some operations without proper precautions may be highly sensitive and thus be numerically unstable.

A well-known example, also present in this package, is the standard closed-form OLS solution $(X'X)^{-1}X'y$, however inverting a matrix can be very unstable if $X'X$ is near singular (Golub and Reinsch, 1970). Thus there are some techniques that ensure that this inversion is not required, such as QR Decomposition or Singular Value Decomposition (SVD). These techniques are also used in this package[4.1].

Additionally, the penalty term of both C-Lasso algorithms may be prone to overfitting, however this could easily be prevented by setting proper values for κ. Similarly, factor extraction of Ando and Bai (2016) and Su and Ju (2018) may be highly unstable if the number of factors is set too high.

4.2 Memory

The second challenge is memory usage, it is of vital importance that the package uses memory properly, as memory usage may be limited. This package uses *numpy* arrays, which are much more efficient compared to default data types implemented in Python (such as Lists, Tuples or DataFrames). This efficiency can be achieved as all elements are stored in a contiguous memory block using a single datatype, making it more expensive for arrays to grow dynamically, but much more memory efficient (Harris et al., 2020). Generally, this package limits the creation of copies of arrays, as these operations, even for *numpy*-arrays are quite slow.

To limit memory usage always the proper datatype is chosen, for example groupings are stored as unsigned 8-bit integers, meaning that the package supports up to 256 groups. Additionally, all

[4.1] QR Decomposition when X is quite stable (as it is slightly faster than SVD) and SVD when this may not be the case.

computations are done using 32-bit floating point numbers as during testing these provided plenty of precision and are half the memory size of regular 64 bit floating point numbers[4.2].

4.3 Optimizer Selection

If a function needs to be optimized, a proper optimization method needs to be chosen as not all are able to deal with every objective function. There are many components that could be considered when selecting a proper method, however in our case there are two main ones: 1) convexity and 2) dimensionality.

First, the two algorithms that use optimizers (Su, Shi, et al. (2016) and Su and Ju (2018)) both minimize (conditionally) convex objective functions. This means that each objective function has a global minimum and thus gradient-based could be used. Secondly, the number of parameters that each objective function has, can vary quite a bit, as this is based on the number of coefficients in β, the number of groups G. In some cases where G and X are quite large, dimensionality can be enormous, therefore a scaleable optimizer with low memory requirements needs to be chosen (Nash, 2014). In our package the L-BFGS method as proposed by Liu and Nocedal (1989) was chosen, as this method has low memory usage and is quite fast (and in addition is implemented in Python).

4.4 Performance

This may be one of the most difficult challenges to solve for these models, as all models introduced in the previous section are quite computationally heavy. A small data set with $N = 100$ and $T = 25$ can easily running the model from Bonhomme and Manresa (2015), can easiliy require 100,000 seperate estimation iterations, without including any bootstrap iterations! This means that all the code in these iterations needs to be as fast as possible, as the impact of just one slow line of code could easily explode the runtime. There are a lot of techniques to optimize the runtime of a program, in this program I have implemented the following three techniques: compilation, parallelization and memoization[4.3].

Python is an interpreted programming language, this means that all the code written is run directly by an interpreter, which reads the code and then runs it, instead of directly running the machine code. This has some major advantages for example not requiring compilation and dynamic typing. However, this generally comes at the cost of performance (Rossum, 1997). However, it is (in some cases) possible to compile Python code. This has been shown to deliver large gains if done properly. There are two main methods to compile Python code, the first is ahead-of-time (AOT) compilation and the second is just-in-time (JIT) compilation. AOT compilation can be achieved by using the *Cython* compiler (Behnel et al., 2011), however this can often be quite complex as code rewrites are necessary and different processing units need different compilation procedures. Often quite similar results can be achieved by using JIT compilation, where the first time a part of the code is run, it gets compiled and stored, and every time after that the same code is reached the compiled version is used. Using JIT compilation can often require some small rewrites in the Python code, however these are often much easier than rewrites for AOT compilation. JIT compilation can be used in Python by using the *Numba*-package (Lam et al., 2015).

[4.2] Note that computation using 32 bit floating point numbers is often not significantly faster for CPUs, but mainly has less memory usage

[4.3] This is in fact the correct spelling of this word.

Almost all new computers contain multiple processing units with multiple cores, allowing the computer to do multiple of the same computations at the same time. In a previous thesis the advantages of using these multiple processing units has been explored (Heijer, 2023), which has shown that using these techniques can in many cases lead to a large speedup of the programs. Parallelization can easily be implemented in many parts of the code, such as the Bootstrap iterations. Parallelization in Python is built-in and does not require any additional packages.

Finally, for function calls that are expensive (i.e., quite slow) and always return the same output for the same input, it can be advantageous to store the results of these function calls and return those values instead of doing the expensive computations again. This only makes sense in very specific circumstances but can delivery very large speedups when implemented correctly (Michie, 1968).

5 Package Usage

Thusfar, only the theoretical background has been explored, but this thesis also introduces a Python package for all these methods. This section will give a reasonably detailed description of how the package should be installed and how it should be used. Additionally, a more recent version of the documentation is available online at: https://groupedpaneldatamodels.michadenheijer.com.

5.1 Installation

Installation of the package is fairly straightforward for (most) machines with Python installed. As the package is published on the Python Package Index (PyPI)[5.1], it can easily be installed using the following command.

```
pip install groupedpaneldatamodels
```

If there are updates available can these be installed using:

```
pip install --upgrade groupedpaneldatamodels
```

In most cases it is advised to run the second command, as it automatically upgrades the dependencies of your machine if the package requires a newer version of a certain dependencies. To download the latest development version of the package, it can simply be downloaded from Github and be installed as follows:

```
git pull https://github.com/michadenheijer/groupedpaneldatamodels.git
cd groupedpaneldatamodel
pip install .
```

On most Windows machines the `pip` command may not be available, thus the alternative `python -m pip` may have to be used. In the case that multiple versions of Python are installed `pip3` may have to be used.

5.2 Usage

Importing the package into any Python file is also relatively simple and can be done using:

```
import groupedpaneldatamodels as gpdm
```

After importing the package into your code you can start using the package! An important detail, the package expects all Panel Data to be in a 3D *numpy*-array, where X is of size $N \times T \times K$and Y of size $N \times T \times 1$. Future work could allow the package to also support other input types (such as Pandas DataFrames), however to limit complexity only this input format is supported.

5.2.1 Grouped Fixed Effects

As was previously described, there are two Grouped Fixed Effects models implemented. As both models are quite similar, they are implemented in the same class: `GroupedFixedEffects`. Usage of this class is quite similar to usage of a model implemented in *statsmodels*. For example a GFE model could be used as follows.

[5.1] Additionally it can be visited on https://pypi.org/project/groupedpaneldatamodels/

```
model_gfe = gpdm.GroupedFixedEffects(y, x, 3, use_bootstrap=False)
model_gfe.fit(max_iter=100, tol=1e-4)
model_gfe.summary()
```

Code 1: Example usage of `GroupedFixedEffects` class

This defines a Grouped Fixed Effects model, to be estimated using the the default Bonhomme and Manresa (2015) model. Each model has different variables and settings that can be modified, they are described below, however there are some common parameters that need to be set for each model.

Parameter	Type	Required	Default	Description
`dependent`	`np.ndarray`	Yes	-	A 3D array of Y, structured by individual, time, and variables.
`exog`	`np.ndarray`	Yes	-	A 3D array of X, used as the regressors in the model.
`G`	`int`	Yes	-	The (maximum) number of groups that are estimated.
`use_bootstrap`	`bool`	No	`False`	Whether or not to estimate the standard errors using the Bootstrap method.
`model`	`str`	No	`"bonhomme_manresa"`	Which model to use: `"bonhomme_manresa"` or `"su_shi_phillips"`.
`heterogeneous_beta`	`bool`	No	`True`	Whether β is grouped-heterogeneous or completely homogeneous if set to `False`.
`entity_effects`	`bool`	No	`False`	Whether to include individual (entity-specific) fixed effects in the estimation.

Table 5.1: Overview of parameters that may be defined for each Grouped Fixed Effects Model

Fitting Each model additionally has their own parameters that may be set, these parameters have to be set in the `.fit()` command. Below are the specific parameters that are able to be set for the Bonhomme and Manresa (2015) model. The default parameters try to ensure that proper convergence will always be reached, however in the case that this procedure needs to be called many times, it may be advantageous to modify these parameters.

Parameter	Type	Required	Default	Description
`n_boot`	`int`	No	50	Number of Bootstrap replications
`max_iter`	`int`	No	10,000	Maximum number of optimization iterations
`tol`	`float`	No	10^{-6}	Acceptable tolerance to stop
`gfe_iterations`	`int`	No	100	Number of different starting points considered
`enable_vns`	`bool`	No	`False`	Usage of the VNS algorithm as described in the paper (not recommended for `heterogeneous_beta = True`, as this combination is very slow)

Table 5.2: Bonhomme and Manresa (2015)-specific parameters

Similarly, Su, Shi, et al. (2016), also has its own specific settings.

Parameter	Type	Required	Default	Description
`kappa`	`float`	No	0.1	κ as described in the algorithm, changing this hyperparameter is highly recommended
`n_boot`	`int`	No	50	Number of Bootstrap replications
`max_iter`	`int`	No	1,000	Maximum number of optimization iterations
`tol`	`float`	No	10^{-6}	Acceptable tolerance to stop
`only_bfgs`	`int`	No	`True`	Only use the L-BFGS optimizer, if `False`, then every iteration is switched between the Nelder-Mead and L-BFGS optimizer

Table 5.3: Su, Shi, et al. (2016)-specific parameters

Results After fitting the results are stored in the instance of the class. To just view the estimated parameters, `gfe_model.params` can be called, to view the standard errors the `gfe_model.params_standard_error` could be called[5.2]. However the easiest method to see the results of the estimation is by calling `gfe_model.summary()`, which displays a *statsmodels*-like summary about the estimation. By default this summary estimates the 95% confidence intervals of the estimated coefficients, however other confidence intervals could be computed using the `gfe_model.get_confidence_intervals(alpha)` function.

5.2.2 Grouped Interactive Fixed Effects

Similarly, the Grouped Interactive Fixed Effects have some parameters that need to be setup before estimating and some parameters that are relevant during fitting[5.3]. There are some parameters that are only valid for some some models. Example usage for this model could be as follows:

```
model_gife = gpdm.GroupedInteractiveFixedEffects(y, x, 3, use_bootstrap
    =False)
model_gife.fit(max_iter=100, tol=1e-4)
model_gife.summary()
```

Code 2: Example usage of `GroupedInteractiveFixedEffects` class

[5.2] There are also `gfe.params_bootstrap_standard_errors` and `gfe.params_analytical_standard_errors` if you specifically want to get the Bootstrapped or analytical standard errors.

[5.3] In general parameters that are set during fitting should be non-relevant hyperparameters, while the other parameters generally need to be defined before usage.

Parameter	Type	Required	Default	Description
`dependent`	`np.ndarray`	Yes	-	A 3D array of Y, structured by individual, time, and variables.
`exog`	`np.ndarray`	Yes	-	A 3D array of X, used as the regressors in the model.
`G`	`int`	Yes	-	The (maximum) number of groups that are estimated.
`use_bootstrap`	`bool`	No	`False`	Whether or not to estimate the standard errors using the Bootstrap method.
`model`	`str`	No	`"ando_bai"`	Which model to use: `"ando_bai"` or `"su_ju"`.
`heterogeneous_beta`	`bool`	No	`True`	Whether β is grouped-heterogeneous or completely homogeneous if set to `False`.
`R`	`int`	No	1	Number of factors used for the `"su_ju"` model
`GF`	`np.ndarray[int]`	No	`[1, ..., 1]`	Number of factors used for each of the groups of the `"ando_bai"` model[5.4]

Table 5.4: Overview of parameters that may be defined for each Grouped Interactive Fixed Effects Model

Fitting To fit the Interactive Effects models the same `gife_model.fit()` needs to be called, this works similarly to the Fixed Effects model.

Parameter	Type	Required	Default	Description
`n_boot`	`int`	No	50	Number of Bootstrap replications
`max_iter`	`int`	No	10,000	Maximum number of optimization iterations
`tol`	`float`	No	10^{-6}	Acceptable tolerance to stop
`gife_iterations`	`int`	No	100	Number of different starting points considered
`kappa`	`float`	No	0	Uses the SCAD Penalized Linear Regression if not set to zero, then is defined as the κ parameter used in the SCAD Penalty
`gamma`	`float`	No	3.7	Defines the γ from the SCAD penalty

Table 5.5: Ando and Bai (2016)-specific parameters

Similarly, for the Su and Ju (2018), we have the following parameters.

Parameter	Type	Required	Default	Description
kappa	float	No	0.1	κ as described in the algorithm, changing this hyperparameter is highly recommended
n_boot	int	No	50	Number of Bootstrap replications
max_iter	int	No	1,000	Maximum number of optimization iterations
tol	float	No	10^{-6}	Acceptable tolerance to stop
only_bfgs	int	No	True	Only use the L-BFGS optimizer, if False, then every iteration is switched between the Nelder-Mead and L-BFGS optimizer

Table 5.6: Su and Ju (2018)-specific parameters

Results The results can be shown similarly to the Grouped Fixed Effects model.

```
params = gife_model.params
params_se = gife_model.params_standard_errors
params_ci = gife_model.get_confidence_intervals(alpha=0.99)
gife_model.summary()
```

Code 3: Find and recall the results of the Grouped Interactive Fixed Effects Model

5.3 Information Criterion Selection

There are simple function available that allows for the selection of the best parameters, the grid_search_by_ic function. This function takes the following parameters and can be used as follows:

```
grid_search_by_ic(GroupedFixedEffects,
                  param_ranges = {"G": [2, 3, 4, 5, 6]},
                  init_params = {"dependent": y, "exog": x},
                  fit_params = {"max_iter": 100},
                  ic_criterion = "BIC")
```

Code 4: Example usage of grid_search_by_ic function

Parameter	Type	Required	Default	Description
model_cls	GroupedFixedEffectsModel or GroupedInteractiveFixedEffectsModel	Yes	-	Class of the type of model you want to select
param_ranges	dict[str, list]	Yes	-	Contains lists for each parameter of parameters to test
init_params	dict	Yes	-	Parameters with which the class is initialized
fit_params	dict	No	None	Parameters that are passed to the fit function
ic_criterion	"BIC", "AIC", or "HQIC"	No	"BIC"	Which IC criterion should be used

Table 5.7: Parameters that can be used for the grid_search_by_ic function

5.4 Expanding the Package

Implementing additional models should be relatively simple, as additional models for (interactive) fixed effects could easily be implemented to the two main classes. Additionally, a base class was implemented that could easily be expanded upon for new types of models. Each new model should return (at least) the following: 1) the estimated parameters (using some sort of proper order, to allow for Bootstrapped estimations) and 2) the estimated residuals. This would allow most functionality of the package to used (such as automatic parameter selection using ICs or automatic standard error calculation using Bootstrap).

6 Verification and Numerical Reliability

Analyzing the correctness of implementations of algorithms can be quite challenging, as just looking at the code often does not immediately suggest that there are any issues. Before verification techniques were commonly used, it was very common for software packages of econometric models to produce wildly incorrect results (Longley, 1967; McCullough and Vinod, 1999; Keeling and Pavur, 2007). To resolve this issue some benchmarks were created for specific models (Wampler, 1980; Elliott et al., 1989), however these benchmarks obviously don't work this package as they are specifically made for other problems.

A number of papers have suggested a few methods to asses the reliability of statistical software. McCullough and Vinod (1999) argued that every statistical software package should do two steps to ensure correctness and numerical reliability of estimation methods. First, all econometric and statistical software should clearly include which mathematical methods are used. This is done to ensure that users can decide if this method is accurate enough or at least know of potential flaws. At the time of publication, most econometric software did not state which method was used for estimating standard errors, potentially providing wildly incorrect results without the user knowing it. Second, every package should include its performance over standard benchmarks if they are available, or include benchmarks and provide the results if not. This gives users a clear estimate about the accuracy that should be expected of the results.

The paper additionally states that appropriate Random Number Generators (RNGs) should be used, with a long enough period. This paper introduces a Python package mainly using *numpy*, it uses the PCG64 RNG (O'Neill, 2014), which has a very long period of 2^{128}. However, RNGs generally don't function well when dealing with parallel code, therefore a hashing algorithm suggested by O'Neill (2015) and implemented in *Numpy* is used.

Finally, the paper states that finally states that the critical values of each package should be carefully considered. This package does not compute the critical values of a specific distribution itself, but uses the critical values computed by *Scipy*, which automatically verifies correctness of each of the critical values on every update (Virtanen et al., 2020).

6.1 Benchmark & Simulation Study Setup

To create a benchmark an appropriate simulation study needs to be set up. Morris et al. (2019) created the ADEMP (*Aims, Data-generating mechanisms, Estimands, Methods and Performance Measures*) framework. This framework has set up for each of the steps specific requirements to ensure that the simulation study (i.e., benchmark) is set up appropriately and correctly.

The *Aims* section, should state what properties are expected from the estimation technique and which aims would be desirable (but not required). Often it is also recommended to compare your specific method to other methods that are not specifically designed for this problem, this could be used to compare the performance of our estimator to other well-known methods.

The *Data-generating mechanisms* section should describe which DGP is used to generate the dataset, DGPs should either be relatively simple, fairly realistic or completely unrealistic to break a certain method. The *Estimants* section should state which coefficent is estimated, this could be relevant parameters, but could also be the performance of a test, or the quality of a model selection procedure.

The *Methods* part of the simulation study, should clearly state which methods are studied and with which hyperparameters. Additionally, the paper recommends that in other commonly used

methods should be compared to the new methods. Finally, the last section (before the results) proper *Performance measures* should be used. The paper states that common performance measures are Bias, Coverage, Confidence Interval Length, Squared Errors. These measures and the sample sizes they are based on should be clearly stated and defined.

7 Simulation Study and Performance Evaluation

Finally, to show the quality and performance of the models implemented in the package a simulation study is conducted. This simulation study attempts to show that the implementation of these models was done properly. It attempts to show that the Python-package can accurately estimate known DGP's. Showing that the implementation of these models was done correctly.

Additionally, this simulation study is the first to compare the quality of inference, i.e., the quality of the estimations of the standard errors, of Ando and Bai (2016) and the Su and Ju (2018) models. The quality of these standard errors is compared to the basic bootstrap algorithm described in Section 2.5.

7.1 Aims

While the goal of this thesis is not to show the validity of Grouped Panel Data Models or its weaknesses, it does want to show that similarly to the reference papers, that the derived properties of relevant estimators hold. Therefore our paper aims to show the following expected properties:

1. The slope estimates $\hat{\beta}_g$ should be and consistent and asymptotically normally distributed for large N, T.
2. The group memberships $\hat{g}_i$ should be consistently estimated.
3. The estimates for standard errors, of both the Bootstrap and Analytical approaches should be able to produce accurate $100(1-\alpha)\%$ confidence intervals.
4. The Information Criterion's (IC's) should be able to accurately select the correct model.

There are also some other aims that we would like that the implemented models have, we would prefer these properties:

1. The slope estimates should be consistent for large N, but fixed T[7.1].
2. The $\mathrm{Var}(\hat{\beta})$, should be as small as possible and should be lower than a fully heterogeneous model.
3. Every estimation should be able to run in a reasonable time frame and should not scale exponentially.

7.2 Data Generating Processes

We consider three Data Generating Processes (DGPs). Each of these DGP's is specifically created to test the performance of a specific model.

Of each DGP $n_{\text{sim}} = 500$ simulations are generated which are stored within the replication code[7.2]. This allows for different models to be run on the same data set and it allows other users

[7.1] We are explicitly not aiming for asymptotic normality or any other distribution for large N, fixed T as most papers have not derived these quantities (except notably for Bonhomme and Manresa, 2015, though these are not implemented).

[7.2] For some models due to time constraints the estimation $n_{\text{sim}} = 200$.

to exactly replicate the same results. Each simulation is generated with 64-bit floating point precision, due to limited storage and memory they are stored with 16-bit floating point precision[7.3].

Each simulated data set is generated with some different parameter settings, allowing us to see the effects on the quality of estimation. For each DGP the size of the data set is either $N = 100$ or $N = 200$, allowing us the see the effect of having a larger data set. Similarly, each data set is generated for either $T = 20$ or $T = 50$, allowing us to see the what the effect of longer data sets is. Additionally, to see the effect that multiple groupings have on the quality of estimation $G = 3$ or $G = 6$. For simplicity and similarity with the reference papers, x_{it} has three parameters, thus $K = 3$.

7.2.1 Grouped Fixed Effects

The first DGP is aimed at the model introduced by Bonhomme and Manresa (2015), this DGP has $\beta_\gamma = [\gamma, \gamma, \gamma]$, meaning that $\beta_1 = [1, 1, 1]$, it also has $\varepsilon_{it} \sim \mathcal{N}(0, 1)$. Additionally, $\alpha_{g_i t} = 0.8\alpha_{g_i t-1} + \eta_{g_i t}$, where $\eta_{g_i t} \sim \mathcal{N}(0, 1)$. Finally, $g_i \sim \mathcal{U}(\{1, ..., G\})$.

$$y_{it} = x'_{it}\beta_{g_i} + \alpha_{g_i,t} + \epsilon_{t,i} \tag{7.1}$$

7.2.2 Grouped Interactive Fixed Effects

The second DGP is very similar to the previous one, with the same parameters, however now with an Interactive component, specifically created for Ando and Bai (2016) and Su and Ju (2018). Each group has just one factor, which is similarly defined as $\alpha_{g_i t}$, meaning that $f_{g_i t} = 0.8f_{g_i t-1} + \eta_{g_i t}$, where $\eta_{g_i t} \sim \mathcal{N}(0, 1)$. Finally, $\lambda_{g_i i} = \mathcal{U}(1, 5)$.

$$y_{it} = x'_{it}\beta_{g_i} + \lambda_{g_i,i} f'_{g_i,t} + \epsilon_{t,i} \tag{7.2}$$

7.2.3 Grouped Fixed Effects with Individual Effects

The final DGP is specifically created for Su, Shi, et al. (2016), where β_g and ε_{it} are defined as previously. However, this DGP is the first to include individual effects, however for simplicity these are defined as $\alpha_i = 0 \; \forall i$.

$$y_{it} = x'_{it}\beta_{g_i} + \alpha_i + \epsilon_{it} \tag{7.3}$$

7.3 Estimands and other targets

There are a three main targets of this simulation study. The first is the point estimate of β_g, its estimated standard error and 95% confidence interval (both are of the bootstrapped and analytical quantities). The second is the quality of the estimated grouping g_i. Finally, we are interested in the quality of different Information Criteria to select the correct model.

[7.3] This small floating point downgrade could result in slightly higher than expected error rates, though these differences are likely very small and insignificant.

7.4 Methods

Each simulated data set is estimated by the relevant model using the estimation methods described in Section 2. For each data set the standard errors are estimated using the analytic method, however for data sets where $G = 6$ the estimation of the standard errors using Bootstrap is too slow for many repetitions, thus only analytical standard errors are considered. For clustering-based models 200 Bootstrap replications are used, while for the penalty-based models just 100 Bootstrap replications are used.

To show the relevance of Grouped Panel Data Models, these models and their estimation techniques are compared to a standard heterogeneous panel data model. A standard heterogeneous model as described by Baltagi (2021, pp. 269–272) could be used,

$$y_{it} = x_{it}'\beta_i + \alpha_i + \varepsilon_{it}, \tag{7.4}$$

where each iteration has their own slope coefficients β_i and each individual has its own individual effect α_i. This model could easily be estimated using a standard OLS estimation for each individual separately. Similarly, the standard errors and confidence intervals could easily be estimated using standard techniques, where the standard errors are computed as follows,

$$\begin{aligned}\widehat{\text{SE}}(\hat{\beta}_j) &= \sqrt{[\hat{\sigma}^2(X'X)^{-1}]_{jj}} \\ \text{CI}_{95\%}(\hat{\beta}_j) &= \left[\hat{\beta}_j - 1.96 \cdot \widehat{\text{SE}}(\hat{\beta}_j),\ \hat{\beta}_j + 1.96 \cdot \widehat{\text{SE}}(\hat{\beta}_j)\right]\end{aligned}$$

7.5 Performance Metrics

To compute the performance and quality of the different methods, different performance metrics are used, which are all defined in this section. First, the performance metrics to analyze the consistency and quality of the estimation of $\hat{\beta}$ are discussed. To measure the consistency of each estimation, we compute the estimated bias first. The bias is defined as follows,

$$\text{Bias} = \frac{1}{n_{\text{sim}}KG}\sum_{i=1}^{n_{\text{sim}}}\sum_{j=1}^{K}\sum_{g=1}^{G}\hat{\beta}_{gj}^{(i)} - \beta_{gj}. \tag{7.5}$$

To get an estimate of the quality of the estimation the Root Mean Squared Error (RMSE) is used. This is defined as follows,

$$\text{RMSE} = \sqrt{\frac{1}{n_{\text{sim}}}\sum_{i=1}^{n_{\text{sim}}}\left\|\hat{\beta}^{(i)} - \beta\right\|^2}. \tag{7.6}$$

To compute the quality of the confidence intervals, the coverage is computed. The computed coverage represents the fraction of confidence intervals that contain the true parameter. In case of the 95% confidence interval, we expect a coverage of 95%.

$$\text{Coverage} = \frac{1}{n_{\text{sim}}}\sum_{i=1}^{n_{\text{sim}}}\sum_{j=1}^{K}\mathbb{1}\left(\beta_j \in \text{CI}_{95\%}(\hat{\beta}_j^{(i)})\right) \tag{7.7}$$

As we prefer smaller confidence intervals, which means that the model has more power, we also compute the average Confidence Interval sizes.

n	t	G	k	corr	bias	rmse	conf. size	conf. size. boot.	coverage	coverage boot.	fit duration
100	20	3	3	1.000	0.001	0.040	0.149	0.154	0.934	0.939	0.040
100	50	3	3	1.000	-0.001	0.025	0.094	0.097	0.932	0.939	0.120
200	20	3	3	1.000	-0.001	0.027	0.106	0.108	0.947	0.948	0.077
200	50	3	3	1.000	0.000	0.018	0.067	0.068	0.944	0.943	0.208
100	20	6	3	0.999	-0.002	0.063	0.207	-	0.903	-	1.571
100	50	6	3	0.986	-0.001	0.117	0.133	-	0.903	-	4.449
200	20	6	3	1.000	0.001	0.041	0.150	-	0.925	-	4.118
200	50	6	3	0.997	-0.003	0.058	0.095	-	0.932	-	9.806

Table 7.1: Results for Bonhomme and Manresa (2015) with DGP 1

$$\text{Avg. CI Size} = \frac{1}{n_{\text{sim}}} \sum_{i=1}^{n_{\text{sim}}} \sum_{j=1}^{K} \sum_{g=1}^{G} \left(\hat{\beta}^{(i)}_{\text{upper},jg} - \hat{\beta}^{(i)}_{\text{lower},jg} \right) \tag{7.8}$$

To get the ratio of individuals that are classified correctly, we use the Correct Classification Rate (CCR). Since for each iteration g_i is unique, it needs to be compared to the correct grouping $g_i^{(j)}$.

$$\text{CCR} = \frac{1}{n_{\text{sim}}} \sum_{j=1}^{n_{\text{sim}}} \frac{1}{n} \sum_{i=1}^{n} \mathbb{1} \left(\hat{g}_i^{(j)} = g_i^{(j)} \right) \tag{7.9}$$

Finally, to compute the average runtime a simple average is computed, in this definition $\tau^{(i)}$ is the runtime of iteration i.

$$\text{Average Runtime} = \frac{1}{n_{\text{sim}}} \sum_{i=1}^{n_{\text{sim}}} \tau^{(i)} \tag{7.10}$$

7.6 Results

First we analyze the results of the grouped models, then we compare those results with the results of a fully heterogeneous model.

7.6.1 Grouped Models

The first implementation of the model proposed by Bonhomme and Manresa (2015), appears to perform quite well as can be seen in Table 7.1. We can clearly see that the model provides a consistent estimate as the bias is (near) zero for all combinations. We can also see that the quality of estimations increases the fewer groupings are required and the more data is available, as the RMSE tends to be lower. Interestingly, when there are more groupings required, increasing the time dimension does not improve estimation. We can also clearly see that estimated analytic standard errors tend to slightly underestimate the true standard errors, as they do not quite reach the 95% coverage they should provide. The Bootstrapped standard errors are slightly better, providing slightly better coverage and almost reaching the 95% coverage that they should provide. Finally, we can see that increasing the number of data points does very much increase the estimation time, suggesting that for very large data sets using this method may not be feasible.

n	t	G	k	corr	bias	rmse	conf. size	conf. size boot.	coverage	coverage boot.	fit duration
100	20	3	3	1.000	-0.002	0.075	0.237	0.266	0.922	0.946	0.242
100	50	3	3	1.000	-0.001	0.035	0.124	0.131	0.931	0.941	0.357
200	20	3	3	1.000	0.003	0.048	0.164	0.175	0.931	0.948	0.288
200	50	3	3	1.000	-0.000	0.024	0.087	0.090	0.934	0.943	0.520
100	20	6	3	0.990	0.006	0.143	0.340	-	0.896	-	6.526
100	50	6	3	0.997	0.003	0.071	0.175	-	0.916	-	8.659
200	20	6	3	1.000	-0.000	0.073	0.244	-	0.923	-	7.091
200	50	6	3	1.000	-0.000	0.035	0.125	-	0.930	-	12.145

Table 7.2: Results for Ando and Bai (2016) with DGP 2

n	t	G	k	corr	bias	rmse	conf. size	conf. size boot.	coverage	coverage boot.	fit duration
100	20	3	3	0.996	-0.002	0.065	0.176	0.285	0.816	0.939	0.981
100	50	3	3	1.000	-0.001	0.031	0.102	0.133	0.899	0.956	2.795
200	20	3	3	0.997	0.001	0.057	0.122	0.163	0.761	0.867	2.421
200	50	3	3	1.000	-0.000	0.024	0.071	0.088	0.861	0.920	6.649
100	20	6	3	0.942	0.003	0.253	0.345	-	0.630	-	1.896
100	50	6	3	0.981	0.001	0.178	0.192	-	0.712	-	5.286
200	20	6	3	0.963	-0.001	0.201	0.230	-	0.539	-	5.272
200	50	6	3	0.996	0.004	0.092	0.113	-	0.671	-	13.169

Table 7.3: Results for Su and Ju (2018) with DGP 2

The results of the Ando and Bai (2016) implementations are very similar to the results of the Bonhomme and Manresa (2015) implementation as can be seen in Table 7.2. The model also appears bo be consistent, although this implementation does not seem to suffer from worse estimation performance (as measured by the RMSE) for larger data sets as the previous model does. Comparing this model with the previous model, we can see that the RMSE is slightly higher, meaning that it performs slightly worse than the previous model.

We can also clearly see that the estimated analytical standard errors underestimate the true standard error, which again the Bootstrapped errors don't appear to suffer from, providing a slightly better standard error estimation at the cost of extra compute time.

In Table 7.3 we can see the results of the implementation of the model proposed by Su and Ju (2018). First, it appears to also be consistent, and able to accurately detect the groupings as each previous model was able to do. It does appear to perform slightly worse than the Ando and Bai (2016) model for large data sets, though it also appears to be effected much less by an increase in data set size. Meaning that larger data sets may be much faster to estimate with this model than with one of the other two models. Finally, the standard errors of appear to be consistently under estimated, as analytical coverage is way to low, and the Bootstrapped coverage is low in certain specifically for the case that $N = 200$ and $T = 20$.

Finally, we look at the implementation of the model proposed by Su, Shi, et al. (2016) in Table 7.4. We can clearly see that it appears to struggle with the same data sets as the previous models and it performs well when there is more data available or there are fewer groups to that need to be estimated. Interestingly, it appears that this model is the slowest of each of these models, which

n	t	G	k	corr	bias	rmse	conf. size	conf. size boot.	coverage	coverage boot.	fit duration
100	20	3	3	0.999	-0.001	0.055	0.156	0.196	0.845	0.916	0.952
100	50	3	3	1.000	-0.000	0.030	0.097	0.113	0.892	0.935	2.184
200	20	3	3	0.999	-0.000	0.043	0.109	0.133	0.796	0.874	2.971
200	50	3	3	1.000	0.001	0.022	0.068	0.078	0.877	0.908	7.189
100	20	6	3	0.982	-0.002	0.163	0.259	-	0.701	-	2.167
100	50	6	3	0.999	0.002	0.082	0.149	-	0.794	-	4.526
200	20	6	3	0.991	-0.008	0.131	0.175	-	0.573	-	9.444
200	50	6	3	1.000	-0.001	0.055	0.101	-	0.705	-	16.441

Table 7.4: Results for Su, Shi, et al. (2016) with DGP 3

n	t	G	k	bias	rmse	conf. size	coverage	bias	rmse	conf. size	coverage	bias	rmse	conf. size	coverage
100	20	3	3	0.001	0.425	1.727	0.950	0.001	0.936	3.619	0.950	-0.000	0.259	1.062	0.949
100	50	3	3	-0.001	0.269	1.062	0.952	-0.001	0.559	2.102	0.951	-0.001	0.149	0.594	0.950
200	20	3	3	-0.001	0.431	1.751	0.950	-0.001	0.941	3.633	0.950	0.000	0.259	1.060	0.950
200	50	3	3	0.001	0.270	1.068	0.950	-0.000	0.562	2.110	0.950	0.000	0.149	0.593	0.950
100	20	6	3	-0.001	0.425	1.730	0.952	-0.002	0.937	3.621	0.951	-0.000	0.258	1.061	0.951
100	50	6	3	0.000	0.270	1.064	0.949	-0.001	0.562	2.107	0.951	0.000	0.149	0.593	0.950
200	20	6	3	0.002	0.430	1.747	0.950	-0.003	0.939	3.624	0.949	-0.000	0.258	1.060	0.951
200	50	6	3	-0.000	0.271	1.071	0.950	0.000	0.561	2.107	0.950	-0.000	0.149	0.594	0.949
				DGP 1				DGP 2				DGP 3			

Table 7.5: Results for the heterogeneous model.

is somewhat unexpected as this model should be easier to estimate than the Su and Ju (2018). We finally notice that the analytical standard errors (which are exactly the same as used by Bonhomme and Manresa, 2015 and thus are correctly implemented), perform very poorly. The Bootstrapped standard errors perform slightly better, though they also understate the true standard errors.

7.6.2 Fully Heterogeneous Model

Comparing the grouped models with fully heterogeneous models, clearly shows the value of the Grouped Panel Data Models. In Table 7.5 we can see these results for all three DGPs. The model is clearly consistent as the estimates appear to not have any bias. However, we can clearly see that the RMSE, i.e., the quality of the estimation is much worse, though the larger T, the smaller effect this has. We can see that the coverage appears to be spot on the 95%, as should be expected, though to reach this coverage, confidence intervals need to be much bigger, as we can clearly see that the coverage size is approximately 4-20 times the size of compared to the grouped models.

8 Discussion and Conclusion

This paper successfully introduces the `groupedpaneldatamodels`-package, a simple to use Python package that is the first that makes Grouped Panel Data Models available to the platform. This package is relatively easy to use, as it hides most of the complexity of all of the implemented models behind a clear structure. This package implements advanced algorithms and uses specialized techniques that improve estimation speed and stability, such as JIT-compilation and efficient linear algebra by using LU decomposition over SVD for Bootstrapped estimations. Additionally, this package provides a method for automatic model selection using Information Criteria, allowing researchers to easily select the correct hyperparameters for each model. Combining all these techniques into a single package allows many more researchers to use Grouped Panel Data Models, whereas this was previously much more difficult, as full implementations of Grouped Panel Data models were rare and often not fully complete.

This thesis shows that the specific implementation of these models is correct and fairly accurate. All our results appear to also be closely aligned with already published papers. For the model introduced by Bonhomme and Manresa (2015) we get near identical results for the bias, additionally the RMSE of our implementations are very close to those suggested by Su, Shi, et al. (2016) and Su and Ju (2018). Suggesting that out implementations are correct.

In a simulation study for using this package, we have found that these grouped models are able to capture slope heterogeneity much better than fully heterogeneous models for our specific Data Generating Processes. As our the models that were implemented in this package were able to tend wildly outperform the fully heterogeneous model when comparing estimation quality by RMSE, showing that grouped models can lower the RMSE by a factor of 2 - 20. This improved performance is mainly visible for data sets where T is small G is limited and N is large. This means that these estimators are much more efficient for data sets where there are a lot of individuals but for each individual there just a few data points available, the models considered in this thesis perform exceedingly well. It should be noted that this simulation study was performed with well-specified models, meaning that that performance of these models for real world panels, which are rarely well-specified.

This paper was also the first paper, to our knowledge, to do a simulation study on the quality of the inference of the models introduced by Ando and Bai (2016) and Su and Ju (2018). We have found that the standard large N, T asymptotic results provide decent estimations for the clustering-based models, however they provide bad or even very bad results for the penalty-based models. This paper suggests using a Bootstrap method get more accurate estimations of the standard errors, which have outperformed analytical standard errors in providing better coverage.

The package of this thesis also has some general limitations that are important to consider. First, due to the computational complexity of Grouped Panel Data Models, estimation times, especially when using Bootstrapped standard errors, can be slow. Finetuning the hyperparameters of these models can help improve estimation speeds, for large data sets i.e., $N > 1000, T > 100$, estimation duration may be too large. We have also found that as G is increasing numerical stability can break down, notably for when T is small.

There are also some other limitations in this package that are not just the caused inherently by Grouped Panel Data Models themselves. For example, there is no support for Unbalanced Panel Data models, while these models make up the large marjority of available panel data set. Future support for unbalanced panel data should be relatively easy to implement, but was not implemented

due to time constraints and overall limitations. Another large improvement to this package could be made by introducing support for non-linear models, this could be implemented relatively easily by implementing linking functions. Additionally, this package only implements the basic large N, T standard errors, while large N, fixed T standard errors may actually be preferable in most situations. This package also provides Bootstrapped standard errors, but assumes normality of the parameters, which does not have to be the case for medium N and small T panels. Additionally, there are also only analytical errors available for individual effects and grouped individual effects, while these could also benefit from Bootstrapped standard errors.

Finally, there are some small usability problems that can make this package somewhat difficult to use. For example the package only supports a very specific inputs by requiring three dimensional *numpy*-arrays, which are not storing any labels or other data about these inputs that could be relevant. This makes it a slight hassle to verify which parameter belongs to each input. Small improvements such as these could make the package even easier to use and allow the implementation to reach a much wider range.

There are many improvements that could be make to the package that this paper introduces. However, it still has succesfully implemented four different Grouped Panel Data Models in an simple to use package. Future research on implementations of Grouped Panel Data Models could focus on two main targets. The first is implementing additional Grouped Panel Data Models to this (or other) package. Due to time constraints many models were implemented, interesting models that could be implemented could be based on Ke et al. (2016), Bonhomme, Lamadon, et al. (2022), Mehrabani (2023), Lumsdaine et al. (2023), or Mugnier (2024). Additionally, both Su, Shi, et al. (2016) and Su and Ju (2018) introduce a Penalized GMM estimator, which is able to deal with endogenous regressors, which was also not included in this package. Future work on the implementation of Grouped Panel Data models, could also focus on removing the limitations previously stated. For example, implementing support for unbalanced panels, could make this package much more usable for real-wold data sets.

References

Ando, Tomohiro and Jushan Bai (Jan. 2016). "Panel Data Models with Grouped Factor Structure Under Unknown Group Membership". en. In: *Journal of Applied Econometrics* 31.1. Number: 1, pp. 163–191. ISSN: 0883-7252, 1099-1255. DOI: 10.1002/jae.2467. URL: https://onlinelibrary.wiley.com/doi/10.1002/jae.2467 (visited on 02/10/2025).

Ando, Tomohiro and Hani Fayad (Feb. 2022). *PDMIF: Fits Heterogeneous Panel Data Models*. en. Institution: Comprehensive R Archive Network Pages: 0.1.0. DOI: 10.32614/CRAN.package.PDMIF. URL: https://CRAN.R-project.org/package=PDMIF (visited on 02/10/2025).

Anselin, Luc (Mar. 2010). "Thirty years of spatial econometrics". In: *Papers in Regional Science* 89.1, pp. 3–26. ISSN: 1056-8190. DOI: 10.1111/j.1435-5957.2010.00279.x. URL: http://dx.doi.org/10.1111/j.1435-5957.2010.00279.x.

Arellano, Manuel, Stéphane Bonhomme, Micole De Vera, Laura Hospido, and Siqi Wei (2022). "Income risk inequality: Evidence from Spanish administrative records". In: *Quantitative Economics* 13.4, pp. 1747–1801. ISSN: 1759-7323. DOI: 10.3982/qe1887. URL: http://dx.doi.org/10.3982/QE1887.

Bai, Jushan (2009). "Panel Data Models With Interactive Fixed Effects". en. In: *Econometrica* 77.4, pp. 1229–1279. ISSN: 0012-9682. DOI: 10.3982/ECTA6135. URL: http://doi.wiley.com/10.3982/ECTA6135 (visited on 06/07/2025).

Baltagi, Badi H. (2021). *Econometric Analysis of Panel Data*. en. Springer Texts in Business and Economics. Cham: Springer International Publishing. ISBN: 978-3-030-53952-8 978-3-030-53953-5. DOI: 10.1007/978-3-030-53953-5. URL: https://link.springer.com/10.1007/978-3-030-53953-5 (visited on 04/15/2025).

Behnel, Stefan, Robert Bradshaw, Craig Citro, Lisandro Dalcin, Dag Sverre Seljebotn, and Kurt Smith (Mar. 2011). "Cython: The Best of Both Worlds". In: *Computing in Science & Engineering* 13.2, pp. 31–39. ISSN: 1521-9615. DOI: 10.1109/mcse.2010.118. URL: http://dx.doi.org/10.1109/MCSE.2010.118.

Bertrand, Q., Q. Klopfenstein, P.-A. Bannier, G. Gidel, and M. Massias (2022). "Beyond L1: Faster and better sparse models with skglm". In: *NeurIPS*.

Bonhomme, Stéphane, Thibaut Lamadon, and Elena Manresa (2022). "Discretizing Unobserved Heterogeneity". en. In: *Econometrica* 90.2. Number: 2, pp. 625–643. ISSN: 0012-9682. DOI: 10.3982/ECTA15238. URL: https://www.econometricsociety.org/doi/10.3982/ECTA15238 (visited on 02/15/2025).

Bonhomme, Stéphane and Elena Manresa (May 2015). "Grouped Patterns of Heterogeneity in Panel Data: Grouped Patterns of Heterogeneity". en. In: *Econometrica* 83.3. Number: 3, pp. 1147–1184. ISSN: 00129682. DOI: 10.3982/ECTA11319. URL: http://doi.wiley.com/10.3982/ECTA11319 (visited on 02/10/2025).

Broyden, C. G. (1970). "The Convergence of a Class of Double-rank Minimization Algorithms 1. General Considerations". In: *IMA Journal of Applied Mathematics* 6.1, pp. 76–90. ISSN: 1464-3634. DOI: 10.1093/imamat/6.1.76. URL: http://dx.doi.org/10.1093/imamat/6.1.76.

Carneiro, Pedro, Karsten T Hansen, and James J Heckman (2003). *Estimating Distributions of Treatment Effects with an Application to the Returns to Schooling and Measurement of the Effects of Uncertainty on College*.

Cawley, John, Karen Conneely, James Heckman, and Edward Vytlacil (1997). "Cognitive Ability, Wages, and Meritocracy". In: *Intelligence, Genes, and Success*. Springer New York, pp. 179–192. ISBN: 9781461206699. DOI: 10.1007/978-1-4612-0669-9_8. URL: http://dx.doi.org/10.1007/978-1-4612-0669-9_8.

Choodari-Oskooei, Babak and Tim Morris (2016). "Quantifying the uptake of user-written commands over time". In: *The Stata Journal* 16.1, pp. 88–95.

Danielsson, Jon and Alvaro Aguirre (Aug. 2020). *Which programming language is best for Economic Research: Julia, Matlab, python or R?* URL: https://cepr.org/voxeu/columns/which-programming-language-best-economic-research-julia-matlab-python-or-r.

Daumler, Davis, Esther Friedman, and Fabian T. Pfeffer (2025). *PSID-SHELF User Guide and Codebook, 1968–2021, Beta Release*. en. DOI: 10.7302/25205. URL: http://deepblue.lib.umich.edu/handle/2027.42/196543.

Davidson, A. C., D. V. Hinkely, and E. Schechtman (1986). "Efficient bootstrap simulation". In: *Biometrika* 73.3, pp. 555–566. ISSN: 1464-3510. DOI: 10.1093/biomet/73.3.555. URL: http://dx.doi.org/10.1093/biomet/73.3.555.

Dogan, Eyup and Fahri Seker (Apr. 2016). "An investigation on the determinants of carbon emissions for OECD countries: empirical evidence from panel models robust to heterogeneity and cross-sectional dependence". In: *Environmental Science and Pollution Research* 23.14, pp. 14646–14655. ISSN: 1614-7499. DOI: 10.1007/s11356-016-6632-2. URL: http://dx.doi.org/10.1007/s11356-016-6632-2.

Efron, Bradley and Robert J. Tibshirani (1993). *An Introduction to the Bootstrap*. New York: Chapman & Hall/CRC. ISBN: 978-0412042317.

Elliott, Alan C et al. (1989). "Benchmark Data Sets for Evaluating Microcomputer Statistical Programs." In: *Collegiate Microcomputer* 7.4, pp. 289–99.

Fan, Jianqing and Runze Li (Dec. 2001). "Variable Selection via Nonconcave Penalized Likelihood and its Oracle Properties". In: *Journal of the American Statistical Association* 96.456, pp. 1348–1360. ISSN: 1537-274X. DOI: 10.1198/016214501753382273. URL: http://dx.doi.org/10.1198/016214501753382273.

Fletcher, R. (Mar. 1970). "A new approach to variable metric algorithms". In: *The Computer Journal* 13.3, pp. 317–322. ISSN: 1460-2067. DOI: 10.1093/comjnl/13.3.317. URL: http://dx.doi.org/10.1093/comjnl/13.3.317.

Gao, Zhan (2020). *classo: Classifier-Lasso for Panel Data Grouping*. Version 0.0.0.9100. URL: https://github.com/zhan-gao/classo.

Giannone, Domenico and Michele Lenza (Jan. 2010). "The Feldstein-Horioka Fact". In: *NBER International Seminar on Macroeconomics* 6.1, pp. 103–117. ISSN: 2150-8372. DOI: 10.1086/648699. URL: http://dx.doi.org/10.1086/648699.

Goldberg, David (Mar. 1991). "What every computer scientist should know about floating-point arithmetic". In: *ACM Computing Surveys* 23.1, pp. 5–48. ISSN: 1557-7341. DOI: 10.1145/103162.103163. URL: http://dx.doi.org/10.1145/103162.103163.

Goldfarb, Donald (1970). "A family of variable-metric methods derived by variational means". In: *Mathematics of Computation* 24.109, pp. 23–26. ISSN: 1088-6842. DOI: 10.1090/s0025-5718-1970-0258249-6. URL: http://dx.doi.org/10.1090/S0025-5718-1970-0258249-6.

Golub, G. H. and C. Reinsch (Apr. 1970). "Singular value decomposition and least squares solutions". In: *Numerische Mathematik* 14.5, pp. 403–420. ISSN: 0945-3245. DOI: 10.1007/bf02163027. URL: http://dx.doi.org/10.1007/BF02163027.

Gómez-Puig, Marta, Simón Sosvilla-Rivero, and Inmaculada Martínez-Zarzoso (Mar. 2022). "On the heterogeneous link between public debt and economic growth". In: *Journal of International Financial Markets, Institutions and Money* 77, p. 101528. ISSN: 1042-4431. DOI: 10.1016/j.intfin.2022.101528. URL: http://dx.doi.org/10.1016/j.intfin.2022.101528.

Goodhue, Lewis, and Thompson (2012). "Does PLS Have Advantages for Small Sample Size or Non-Normal Data?" In: *MIS Quarterly* 36.3, p. 981. ISSN: 0276-7783. DOI: 10.2307/41703490. URL: http://dx.doi.org/10.2307/41703490.

Grayling, Michael J and Graham M Wheeler (2019). *A review of available software for adaptive clinical trial design*. DOI: 10.48550/ARXIV.1906.05603. URL: https://arxiv.org/abs/1906.05603.

Grunewald, Nicole, Stephan Klasen, Inmaculada Martínez-Zarzoso, and Chris Muris (Dec. 2017). "The Trade-off Between Income Inequality and Carbon Dioxide Emissions". In: *Ecological Economics* 142, pp. 249–256. ISSN: 0921-8009. DOI: 10.1016/j.ecolecon.2017.06.034. URL: http://dx.doi.org/10.1016/j.ecolecon.2017.06.034.

Hahn, Jinyong and Hyungsik Roger Moon (June 2010). "Panel Data Models with Finite Number of Multiple Equilibria". en. In: *Econometric Theory* 26.3. Number: 3, pp. 863–881. ISSN: 0266-4666, 1469-4360. DOI: 10.1017/S0266466609990132. URL: https://www.cambridge.org/core/product/identifier/S0266466609990132/type/journal_article (visited on 03/09/2025).

Haimerl, Paul (Feb. 2024). *PAGFL: Joint Estimation of Latent Groups and Group-Specific Coefficients in Panel Data Models*. DOI: 10.32614/cran.package.pagfl. URL: http://dx.doi.org/10.32614/CRAN.package.PAGFL.

Hajivassiliou, Vassilis A. (2018). "Computational Methods in Econometrics". In: *The New Palgrave Dictionary of Economics*. Palgrave Macmillan UK, pp. 1991–1997. ISBN: 9781349951895. DOI: 10.1057/978-1-349-95189-5_2725. URL: http://dx.doi.org/10.1057/978-1-349-95189-5_2725.

Harris, Charles R., K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant (Sept. 2020). "Array programming with NumPy". In: *Nature* 585.7825, pp. 357–362. ISSN: 1476-4687. DOI: 10.1038/s41586-020-2649-2. URL: http://dx.doi.org/10.1038/s41586-020-2649-2.

Heijer, Micha den (2023). "Speeding up Slow Monte Carlo Simulations using Parallel Computing Techniques". Bachelor thesis. Vrije Universiteit Amsterdam. URL: https://www.academia.edu/127126512/Speeding_up_Slow_Monte_Carlo_Simulations_using_Parallel_Computing_Techniques.

Holly, Sean, M. Hashem Pesaran, and Takashi Yamagata (Sept. 2010). "A spatio-temporal model of house prices in the USA". In: *Journal of Econometrics* 158.1, pp. 160–173. ISSN: 0304-

4076. DOI: 10.1016/j.jeconom.2010.03.040. URL: http://dx.doi.org/10.1016/j.jeconom.2010.03.040.

Hsiao, Cheng (2014). *Analysis of Panel Data*. en. ISBN: 978-1-139-83932-7.

Huang, Wenxin, Yiru Wang, and Lingyun Zhou (Mar. 2024). "Identify latent group structures in panel data: The classifylasso command". en. In: *The Stata Journal: Promoting communications on statistics and Stata* 24.1. Number: 1, pp. 46–71. ISSN: 1536-867X, 1536-8734. DOI: 10.1177/1536867X241233642. URL: https://journals.sagepub.com/doi/10.1177/1536867X241233642 (visited on 02/10/2025).

Ke, Yuan, Jialiang Li, and Wenyang Zhang (June 2016). "Structure identification in panel data analysis". en. In: *The Annals of Statistics* 44.3. Number: 3. ISSN: 0090-5364. DOI: 10.1214/15-AOS1403. URL: https://projecteuclid.org/journals/annals-of-statistics/volume-44/issue-3/Structure-identification-in-panel-data-analysis/10.1214/15-AOS1403.full (visited on 02/10/2025).

Keeling, Kellie B. and Robert J. Pavur (May 2007). "A comparative study of the reliability of nine statistical software packages". In: *Computational Statistics & Data Analysis* 51.8, pp. 3811–3831. ISSN: 0167-9473. DOI: 10.1016/j.csda.2006.02.013. URL: http://dx.doi.org/10.1016/j.csda.2006.02.013.

Lam, Siu Kwan, Antoine Pitrou, and Stanley Seibert (Nov. 2015). "Numba: a LLVM-based Python JIT compiler". In: *Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC*. SC15. ACM, pp. 1–6. DOI: 10.1145/2833157.2833162. URL: http://dx.doi.org/10.1145/2833157.2833162.

Lamadon, Thibaut (Oct. 2021). *pygrpfe: Grouped Fixed Effect helper functions for Python*. https://github.com/tlamadon/pygrpfe. Git commit d12ee27, accessed 18 Jun 2025.

Liu, Dong C. and Jorge Nocedal (Aug. 1989). "On the limited memory BFGS method for large scale optimization". In: *Mathematical Programming* 45.1–3, pp. 503–528. ISSN: 1436-4646. DOI: 10.1007/bf01589116. URL: http://dx.doi.org/10.1007/BF01589116.

Longley, James W. (1967). "An Appraisal of Least Squares Programs for the Electronic Computer from the Point of View of the User". In: *Journal of the American Statistical Association* 62.319, pp. 819–841. ISSN: 01621459, 1537274X. URL: http://www.jstor.org/stable/2283673 (visited on 06/11/2025).

Lumsdaine, Robin L., Ryo Okui, and Wendun Wang (Mar. 2023). "Estimation of panel group structure models with structural breaks in group memberships and coefficients". en. In: *Journal of Econometrics* 233.1. Number: 1, pp. 45–65. ISSN: 03044076. DOI: 10.1016/j.jeconom.2022.01.001. URL: https://linkinghub.elsevier.com/retrieve/pii/S0304407622000033 (visited on 02/10/2025).

Lütkepohl, Helmut (Aug. 2004). "Univariate Time Series Analysis". In: *Applied Time Series Econometrics*. Cambridge University Press, pp. 8–85. ISBN: 9780511606885. DOI: 10.1017/cbo9780511606885.003. URL: http://dx.doi.org/10.1017/CBO9780511606885.003.

Manini, Paolo F., Samuele Segoni, and Andrea Rosi (2020). "Upgrading Model Selection Criteria with Goodness of Fit Tests for Practical Applications". In: *Entropy* 22.4, p. 447. DOI: 10.3390/e22040447.

McCullough, B. D. and H. D. Vinod (1999). "The Numerical Reliability of Econometric Software". en. In: *Journal of Economic Literature* 37.2, pp. 633–665. URL: http://www.jstor.org/stable/2565215.

Mehrabani, Ali (Aug. 2023). "Estimation and identification of latent group structures in panel data". en. In: *Journal of Econometrics* 235.2. Number: 2, pp. 1464–1482. ISSN: 03044076. DOI: 10.1016/j.jeconom.2022.12.002. URL: https://linkinghub.elsevier.com/retrieve/pii/S030440762200207X (visited on 02/10/2025).

Michie, Donald (1968). ""Memo" functions and machine learning". In: *Nature* 218.5136, pp. 19–22.

Morris, Tim, Ian R. White, and Michael J. Crowther (Jan. 2019). "Using simulation studies to evaluate statistical methods". In: *Statistics in Medicine* 38.11, pp. 2074–2102. ISSN: 1097-0258. DOI: 10.1002/sim.8086. URL: http://dx.doi.org/10.1002/sim.8086.

Mugnier, Martin (Sept. 2024). *A Simple and Computationally Trivial Estimator for Grouped Fixed Effects Models*. en. Issue: arXiv:2203.08879 arXiv:2203.08879 [econ]. DOI: 10.48550/arXiv.2203.08879. URL: http://arxiv.org/abs/2203.08879 (visited on 02/12/2025).

Mundlak, Yair (1961). "Empirical Production Function Free of Management Bias". In: *Journal of Farm Economics* 43.1, pp. 44–56. DOI: 10.2307/1235460.

— (Jan. 1978). "On the Pooling of Time Series and Cross Section Data". In: *Econometrica* 46.1, p. 69. ISSN: 0012-9682. DOI: 10.2307/1913646. URL: http://dx.doi.org/10.2307/1913646.

Nash, John C. (2014). "On Best Practice Optimization Methods in R". In: *Journal of Statistical Software* 60.2. ISSN: 1548-7660. DOI: 10.18637/jss.v060.i02. URL: http://dx.doi.org/10.18637/jss.v060.i02.

Nickell, Stephen (Nov. 1981). "Biases in Dynamic Models with Fixed Effects". In: *Econometrica* 49.6. Publisher: JSTOR, p. 1417. ISSN: 0012-9682. DOI: 10.2307/1911408. URL: http://dx.doi.org/10.2307/1911408.

O'Neill, Melissa E. (Sept. 2014). *PCG: A Family of Simple Fast Space-Efficient Statistically Good Algorithms for Random Number Generation*. Tech. rep. HMC-CS-2014-0905. Claremont, CA: Harvey Mudd College. URL: https://www.cs.hmc.edu/tr/hmc-cs-2014-0905.pdf.

— (Apr. 30, 2015). *Developing a seed_seq alternative*. PCG, A Better Random Number Generator blog. URL: https://www.pcg-random.org/posts/developing-a-seed_seq-alternative.html (visited on 06/20/2025).

Pesaran, M.Hashem and Ron Smith (July 1995). "Estimating long-run relationships from dynamic heterogeneous panels". In: *Journal of Econometrics* 68.1, pp. 79–113. ISSN: 0304-4076. DOI: 10.1016/0304-4076(94)01644-f. URL: http://dx.doi.org/10.1016/0304-4076(94)01644-F.

Priestley, M. B. (1982). *Spectral Analysis and Time Series, Volumes 1–2*. 1st ed. Comprehensive two-volume set covering univariate and multivariate time series. London: Academic Press. ISBN: 978-0-12-564922-3.

Pullenayegum, Eleanor M., Robert W. Platt, Melanie Barwick, Brian M. Feldman, Martin Offringa, and Lehana Thabane (Aug. 2015). "Knowledge translation in biostatistics: a survey of current practices, preferences, and barriers to the dissemination and uptake of new statistical methods". In: *Statistics in Medicine* 35.6, pp. 805–818. ISSN: 1097-0258. DOI: 10.1002/sim.6633. URL: http://dx.doi.org/10.1002/sim.6633.

Rossum, Guido van (1997). *Comparing Python to Other Languages*. URL: https://www.python.org/doc/essays/comparisons/ (visited on 06/19/2025).

Schwarz, Gideon (Mar. 1978). "Estimating the Dimension of a Model". In: *The Annals of Statistics* 6.2. ISSN: 0090-5364. DOI: 10.1214/aos/1176344136. URL: http://dx.doi.org/10.1214/aos/1176344136.

Shanno, D. F. (1970). "Conditioning of quasi-Newton methods for function minimization". In: *Mathematics of Computation* 24.111, pp. 647–656. ISSN: 1088-6842. DOI: 10.1090/s0025-5718-1970-0274029-x. URL: http://dx.doi.org/10.1090/S0025-5718-1970-0274029-X.

Sin, Chor-Yiu and Halbert White (Mar. 1996). "Information criteria for selecting possibly misspecified parametric models". In: *Journal of Econometrics* 71.1–2, pp. 207–225. ISSN: 0304-4076. DOI: 10.1016/0304-4076(94)01701-8. URL: http://dx.doi.org/10.1016/0304-4076(94)01701-8.

Su, Liangjun and Gaosheng Ju (Oct. 2018). "Identifying latent grouped patterns in panel data models with interactive fixed effects". en. In: *Journal of Econometrics* 206.2. Number: 2, pp. 554–573. ISSN: 03044076. DOI: 10.1016/j.jeconom.2018.06.014. URL: https://linkinghub.elsevier.com/retrieve/pii/S0304407618301106 (visited on 02/10/2025).

Su, Liangjun, Zhentao Shi, and Peter C. B. Phillips (2016). "Identifying Latent Structures in Panel Data". en. In: *Econometrica* 84.6. Number: 6, pp. 2215–2264. ISSN: 0012-9682. DOI: 10.3982/ECTA12560. URL: https://www.econometricsociety.org/doi/10.3982/ECTA12560 (visited on 02/10/2025).

Sun, Yixiao (2002). *Catching up, Forging ahead, and Falling behind: A Panel Structure Analysis of Convergence Clubs*. URL: https://economics.ucsd.edu/~yisun/growth_empirics.pdf.

— (2005). *Estimation and Inference in Panel Structure Models*. en. URL: https://escholarship.org/uc/item/5tf1231k.

Virtanen, Pauli et al. (Feb. 2020). "SciPy 1.0: fundamental algorithms for scientific computing in Python". In: *Nature Methods* 17.3, pp. 261–272. ISSN: 1548-7105. DOI: 10.1038/s41592-019-0686-2. URL: http://dx.doi.org/10.1038/s41592-019-0686-2.

Wahlquist, Amy E., Lutfiyya N. Muhammad, Teri Lynn Herbert, Viswanathan Ramakrishnan, and Paul J. Nietert (Aug. 2018). "Dissemination of novel biostatistics methods: Impact of programming code availability and other characteristics on article citations". In: *PLOS ONE* 13.8. Ed. by Dena L. Schanzer, e0201590. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0201590. URL: http://dx.doi.org/10.1371/journal.pone.0201590.

Wampler, Roy H (1980). "Test procedures and test problems for least squares algorithms". In: *Journal of Econometrics* 12.1, pp. 3–22.

Wang, Wuyi, Peter C. B. Phillips, and Liangjun Su (Sept. 2018). "Homogeneity pursuit in panel data models: Theory and application". en. In: *Journal of Applied Econometrics* 33.6. Number: 6, pp. 797–815. ISSN: 0883-7252, 1099-1255. DOI: 10.1002/jae.2632. URL: https://onlinelibrary.wiley.com/doi/10.1002/jae.2632 (visited on 02/15/2025).

Wilson, Greg, D. A. Aruliah, C. Titus Brown, Neil P. Chue Hong, Matt Davis, Richard T. Guy, Steven H. D. Haddock, Kathryn D. Huff, Ian M. Mitchell, Mark D. Plumbley, Ben Waugh, Ethan P. White, and Paul Wilson (Jan. 2014). "Best Practices for Scientific Computing". In: *PLoS Biology* 12.1. Ed. by Jonathan A. Eisen, e1001745. ISSN: 1545-7885. DOI: 10.1371/journal.pbio.1001745. URL: http://dx.doi.org/10.1371/journal.pbio.1001745.

A Bonhomme and Manresa (2015)

A.1 Individual-Effects Extension

The individual effects extension suggested by Bonhomme and Manresa (2015), is defined as follows.

$$y_{it} = x_{it}'\beta_{g_i} + \alpha_{g_i,t} + \mu_i + \varepsilon_{it}, \quad \begin{matrix} i \in \{1, ..., N\} \\ t \in \{1, ..., T\} \end{matrix} \tag{A.1}$$

To estimate this model the following minimization problem needs to be solved,

$$Q_{\text{BM,ind}}(\beta, \alpha, g) = \sum_{i=1}^{N} \sum_{t=1}^{T} \left(y_{it} - x_{it}'\beta_{g_i} - \alpha_{g_i,t} - \mu_i\right)^2 \tag{A.2}$$

$$(\hat{\beta}, \hat{\alpha}, \hat{g}, \hat{\mu}) = \underset{(\beta,\alpha,g,\hat{\mu}) \in B \times A^{GT} \times \Gamma_G \times M}{\text{argmin}} Q_{\text{BM}}(\beta, \alpha, g, \mu). \tag{A.3}$$

To estimate this model it is generally preferred to just demean both y and x first, by getting $\tilde{y}_{it} = y_{it} - \bar{y}_i$ and $\tilde{x}_{it} = x_{it} - \bar{x}_i$. Then run the one of the provided algorithms first on $\tilde{y}$ and $\tilde{x}$. And finally retrieving the individual effects by setting $\hat{\mu}_i = \bar{y}_i - \bar{x}_i'\hat{\beta}_{\hat{g}_i}$.

A.2 Variable Neighborhood Search

The Variable Neighborhood Search algorithm described by Bonhomme and Manresa (2015) is defined as follows.

1. Start with some feasible starting values for $\beta^{(0)}$, $\alpha^{(0)}$. Set $j = 1$.
2. Perform one assignment step of the original algorithm to obtain initial groupings $g^{(0)}$.
3. Set $n = 1$.
4. Do a Neighborhood jump and a Local Search
   (a) *Neighborhood Jump*: Randomly select n units and move them to another group and perform an update step from the original algorithm.
   (b) *Local Search*: Systematically check if a relocation of a single individual to another groups improves the objective function. If it does, return to Step 3. If it does not, set $n+ = 1$ and return to Step 4. If $n > 10$, then no more improvements can be made and continue to the next step.
5. Set $j+ = 1$ If $j >$ max_iter, then stop, else return to Step 2.

Usage of this algorithm is encouraged under the assumption that β is homogeneous, but highly discouraged otherwise as it is very slow.

B Ando and Bai (2016)

B.1 Homogeneous Model

The homogeneous model proposed by Ando and Bai (2016), is defined as follows,

$$y_{it} = x'_{it}\beta + f'_{g_i,t}\lambda_{g_i,i} + \varepsilon_{it}, \quad \begin{array}{l} i \in \{1, ..., N\}, \\ t \in \{1, ..., T\}. \end{array} \tag{B.1}$$

The objective function and algorithm to compute this model is the same as the algorithm proposed in the main text, however each $\beta_{\hat{g}_i}$, should just be replaced with the homogeneous coefficient β.

B.2 High Dimensional Model

The high-dimensional model proposed by Ando and Bai (2016), has an objective function that is defined as follows,

$$Q_{\text{AB}}(\beta, f, \lambda, g) = \sum_{i=1}^{N}\sum_{t=1}^{T}\left(y_{it} - x'_{it}\beta_{g_i} - f'_{g_i,t}\lambda_{g_i,i}\right)^2 + NT \cdot p_{\kappa,\gamma}(|\beta|), \tag{B.2}$$

$$p_{\kappa,\gamma}(|\beta_j|) = \begin{cases} \kappa|\beta_j| & \text{if } |\beta_j| \leq \kappa, \\ \frac{\gamma\kappa|\beta_j| - 0.5(\beta_j^2 + \kappa^2)}{\gamma - 1} & \text{if } \kappa < |\beta_j| < \gamma\kappa, \\ \frac{\kappa^2(\gamma^2 - 1)}{2(\gamma - 1)} & \text{if } \gamma\kappa \leq |\beta_j|, \end{cases} \tag{B.3}$$

where the penalty is the well-known SCAD penalty as discussed by Fan and Li (2001). This high dimensional model is estimated by the Bertrand et al. (2022)-package, and replaces the standard OLS estimations in case it is used (Bertrand et al., 2022).