The Determinants of Capital Structure: An Empirical Study on Singapore Exchange Listed Firms

Yung Qi Yang[a] and Zhang Yuhang[a]

[a] *School of Economics, Singapore Management University, Singapore*

Course Project for DSA305 Panel Data Analysis

Instructor: Professor Zhenlin Yang

April 5, 2024

Abstract

It is theoretically understood that the capital structure of a firm drives internal strategic decisions, and gives a strong indication of its financial health, and ultimately, its overall performance in the market. As such, the factors affecting capital structure are often subjected to studies in the academic and corporate spheres, as they are in this paper. With its robust regulatory structures, and bright economic landscape, Singapore, and specifically the Singapore Exchange (SGX) offers us a great opportunity to investigate the empirical performance of theoretical frameworks of capital structure determinants. And perhaps, even more crucially, the premise of our chosen subject presents us with a model example of short panel data, and by extension, the use of its estimation techniques. To develop our model framework, we rely upon the brilliance of existing studies into the subject to select an initial array of prospective, significant variables. Using them, we investigate into their significance when contextualized to the publicly, available SGX data to narrow the field of parameters that are noteworthy to Singaporean-listed companies. We shall also dive deeper into the variety of estimation techniques commonly used in panel data analysis to refine the functionality of our model, and segments of the data to stress test our model's viability.

Key Words: Leverage level; Panel data econometrics; Pecking order theory; Correlated random effects

1. Introduction

It has been empirically observed in many countries that a firm's capital structure drives internal strategic decisions, and gives strong indication about its financial health, and ultimately, its overall performance in the market. Understanding the determinants of a firm's capital is thus often of interest for academics and practitioners alike. As one Asia's leading financial hubs, Singapore possesses an economic environment fortified with robust regulatory frameworks, extensive access to global capital markets, along with a good mixture of larger multinational corporations (MNCs) and smaller, domestic firms. Thus, bestowing us with an incredibly rich, diverse and complete dataset to test traditional iterations of key capital structure determinants models empirically. By employing rigorous quantitative analysis, we seek to identify and evaluate the influence of factors such as profitability, asset tangibility, firm size, growth opportunities, leverage, and market conditions on firms' capital structure choices.

Whilst extensive research has been conducted on capital structure determinants globally, there remains a dearth of not just empirical studies focusing specifically on Singapore-listed firms, as well as a more statistical-focused discussion on the use of different estimation methods available. Thus, with this research, we attempt to contribute to the existing body of literature on capital structure by offering not just insights specific to the Singaporean market, but also insights rooted within the discussion of the different estimation techniques of panel data structures. By narrowing our scope to this market, we aim to provide nuanced findings that contribute to a deeper understanding of capital structure dynamics within the Singaporean context.

Due to the nature of this research which signifies the cross-sectional pattern of a firm's choice on making financial decision combined with time-varying decisions reassured by different effects across years, panel data analysis techniques are highly suitable to achieve our aforementioned conclusion. For the rest of this paper, we will consider an entity to be each individual firm, which whose effects we expect to vary significantly because they would have different sets of corporate culture and values, working and decision-making processes and patterns affected by managers, decision-preferences resulting from personal preferences of shareholders etc. The potential effect on capital structure in and by each time period as a result of the specific year's economic environment and the stage of business cycle and credit cycle will be captured by considering the year of each financial report as a time panel.

Through a comprehensive analysis of empirical data gathered from Singapore exchange listed firms over a specified period, this study seeks to offer actionable insights for corporate finance practitioners, policymakers, and investors. By understanding the factors that drive

capital structure decisions, firms can make informed financing choices that optimize their capital structure and enhance their overall financial performance.

2. Data Source and Theoretical background

2.1. Data Source

The database we used is called Orbis, provided by Moody's Analytics, which is one of the largest credit rating and data services providers in the world. Our approach centered on the principle of minimizing missing values whilst maximizing the coverage across the 20-year period from 2004 to 2023. Given the variability in the listing durations of companies on the Singapore Exchange (SGX), we strategically selected dimensions that would yield a robust dataset conducive to meaningful analysis. This involved striking a balance between temporal depth and entity breadth to ensure a comprehensive representation of the Singapore-listed firms while maintaining data consistency and reliability.

In addition to the financial statement items, the ones from Balance Sheet, Income Statement and Cashflow Statement following International Financial Reporting Standards (IFRS), extracted from Orbis, our dataset incorporates other pertinent information including the Singapore Overnight Rate Average (SORA) and the Consumer Price Index (CPI) as proxies for market conditions and inflation dynamics, respectively. These supplementary variables enhance the richness of our dataset and enable a more comprehensive analysis of the determinants influencing capital structure decisions among Singapore-listed firms.

Following the cleaning process, we arrived at a condensed panel dataset characterized by an 8-year time index spanning from 2016 to 2023 and encompassing 216 unique firms in the entity index. This approach allowed us to focus on a consistent time frame while capturing a diverse cross-section of companies listed on the SGX.

2.2. Variables Chosen

As stated by Gaud et al. (2005), Bayrakdaroglu et al. (2013) and Neves et al. (2020), either under the circumstances of economic turbulence or stable and booming economy, a few factors derived from financial statements are proven to be significant in a few territories including Turkey, Switzerland and Portugal. We adopted their selection of variables, and added Free Cashflow for Firm (FCFF), Growth Potential, SORA and CPI of which we expected them to play an important role in affecting the firm's decision of capital structure. Furthermore, we have chosen leverage level, represented by net debt over total assets, as the capital structure for

a firm. Table 1 showcases the derivation of variables we used to fit our model:

Table 1. Derivations of variables used in our model

Acronym	Variable Name	Derivation
$SIZE$	The size of a firm based on its financial statement	log(Total Assets)
$PROFITABILITY$	The gross earnings of a firm as a portion of total assets	$\frac{\text{EBITA}}{\text{Total Assets}}$
$TANG$	Tangibility of a firm's assets	$\frac{\text{Fixed Assets}}{\text{Total Assets}}$
$RISK$	The sensitivity of the earnings before interest and tax expense to the changes of net sales	$\frac{\%\Delta\text{EBIT}}{\%\Delta\text{Net Sales}}$
$LIQUID$	The ability of a firm to liquidate its current assets to fulfil its current obligations	$\frac{\text{Current Assets}}{\text{Current Liabilities}}$
$NDTSHIELD$	The non-debt tax shield, an indicator of the firm's benefit from choosing non-debt financing channel	$\frac{\text{Fixed Assets}}{\text{Total Assets}}$
$MCAP$	Market capitalization, another indicator of a firm size, based on the market value of equity	-
$FCFF$	Free cashflow for firm, indicating the cashflow of operating activities after the deduction of non-cash charges and capital investment	-
$GROWPOT$	The growth potential of a firm based on the market	$\frac{\text{Market Capitalization}}{\text{Total Equity}}$
$SOLV$	The scale of earnings can be used to pay interest	$\frac{\text{EBIT}}{\text{Interest Expense}}$
$SORA$	Singapore Overnight Rate Average, indicator of risk-free rate in the market	-
CPI	Consumer Price Index, indicator of inflation rate	-
$LEVERAGE$	Dependent variable, indicator of the capital structure of a firm	$\frac{\text{Nedt Debt}}{\text{Total Assets}}$

With all the variables we have, we are going to find the significance of explanatory variables on the dependent variable by introducing the considerations of fixed effects brought by individual firms or the year due to the cycle in which the year was, random effects due to the high volatility of the capital markets from different channels, or correlated random effects that mixes the complexity of capital markets and the individual firm's decision based on different factors combined. The correlation among explanatory variables is given below in figure 1:

There are a few data points to be further investigated for the high level of correlation between some variables:

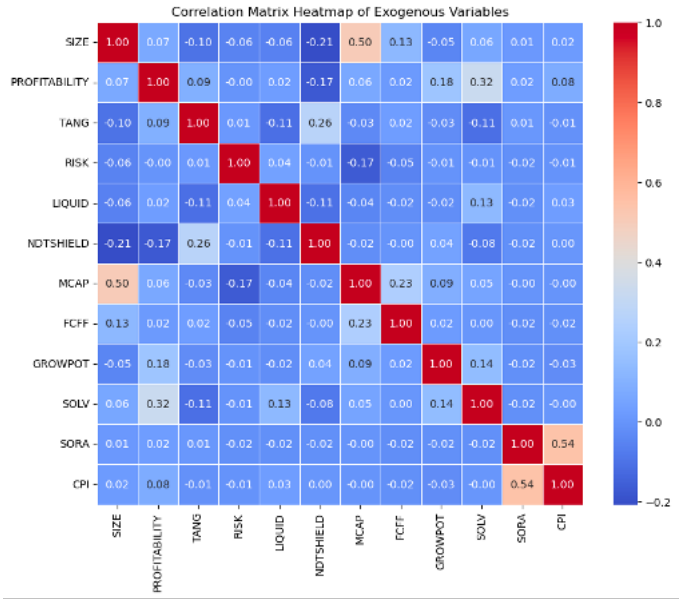

Figure 1: Correlation among explanatory variables

1. $SIZE$ and $MCAP$: Both variables are indicators of the size of a firm, based on total assets or the market value of equity. It is expected to be correlated, but we do not assume they are linearly correlated since they tend to move together but do not derive one from the other.

2. $SORA$ and CPI: Based on macroeconomic theory, the interest rate in the economy might be correlated to inflation level as researched and proven empirically by Baharumshah et al. (2019), and practically the central bank will conduct contractionary monetary policy (i.e. increase interest rate, SORA in our case) to seize the overheated inflation rate, represented by CPI. However, we do not remove either of them since they are not likely linearly correlated to each other, and we need to investigate the further individual or synergetic effect brought by them in our following models.

Overall, our target is to test the significance of each variable on the leverage level chosen by a firm, and consider the extensive effects such as fixed one and random one in either one-way or two-way, and with the ideology that within each cluster of company (an entity), the variance of residuals might not stand at the same point, leading the further consideration of robust estimation of parameters and the standard errors of parameters.

3. Model Fitting

3.1. Pooled Ordinary Least Square

To test whether there are fixed effects, random effects or correlated random effects, we need to run a Pooled Ordinary Least Square (OLS) as the base for null hypothesis. Besides the simple OLS, we also need to consider the nature of dataset: the possible heteroskedasticity drawn by the short panel dataset we produced. The model 3.1 is introduced:

$$\begin{aligned} LEVERAGE_{it} = \beta_0 &+ \beta_1 SIZE_{it} + \beta_2 PROFITABILITY_{it} + \beta_3 TANG_{it} + \beta_4 RISK_{it} \\ &+ \beta_5 LIQUID_{it} + \beta_6 NDTSHIELD_{it} + \beta_7 MCAP_{it} + \beta_8 FCFF_{it} \\ &+ \beta_9 GROWPOT_{it} + \beta_{10} SOLV_{it} + \beta_{11} SORA_t + \beta_{12} CPI_t + u_t \end{aligned} \tag{3.1}$$

From which we have a null hypothesis 3.2 for this pooled OLS model:

$$H_0 : \quad \beta_1 = \beta_2 = \cdots = \beta_{12} \tag{3.2}$$

Given in table 2, we have the first Pooled OLS result with overall F-statistic of 81.520 (p-value smaller than 0.0001) to reject the null hypothesis 3.2:

Table 2. Pooled OLS result for model 3.1

Variable	Parameter	Standard error	p-value
constant	-0.5894	0.0658	< 0.0001
$SIZE$	0.0417	0.0033	< 0.0001
$PROFITABILITY$	-0.4409	0.0604	< 0.0001
$TANG$	0.1953	0.0259	< 0.0001
$RISK$	-2.271×10^{-5}	2.019×10^{-5}	0.2697
$LIQUID$	-0.0545	0.0028	< 0.0001
$NDTSHIELD$	-0.0379	0.0896	0.6723
$MCAP$	-8.883×10^{-12}	1.934×10^{-12}	< 0.0001
$FCFF$	-1.517×10^{-13}	2.702×10^{-13}	0.9536
$GROWPOT$	-0.0021	0.0026	0.4143
$SOLV$	-0.0007	7.083×10^{-5}	< 0.0001
$SORA$	0.0099	0.0110	0.3706
CPI	-0.0012	0.0029	0.6700

Our strategy is to drop the irrelevant variables one-by-one (p-value greater than 0.01, which is significant at the confidence level of 99%) and check if the overall fitness of our model has improved. In the process of dropping, the changes in the significance level for the estimated

parameters with p-value smaller than 0.0001 in table 2 are trivial, whilst the rest of insignificant variables remained highly insignificant. Hence, we arrived at the final pooled OLS model 3.3:

$$LEVERAGE_{it} = \beta_0 + \beta_1 SIZE_{it} + \beta_2 PROFITABILITY_{it} + \beta_3 TANG_{it} + \beta_4 LIQUID_{it} + \beta_5 MCAP_{it} + \beta_6 SOLV_{it} + u_t \tag{3.3}$$

With the following result in table 3, it can be seen that all the variables at this stage are highly significant, indicating strong influence as the determinants for the firms to choose their leverage level. And the F-statistic for model 3.3 climbed to a higher value at 162.81, which reassures the overall fitness of our model.

Table 3. Pooled OLS result for model 3.3

Variable	Parameter	Standard error	p-value
constant	-0.5964	0.0628	< 0.0001
$SIZE$	0.0422	0.0032	< 0.0001
$PROFITABILITY$	-0.4442	0.0582	< 0.0001
$TANG$	0.1932	0.0249	< 0.0001
$LIQUID$	-0.0545	0.0027	< 0.0001
$MCAP$	-8.842×10^{-12}	1.85×10^{-12}	< 0.0001
$SOLV$	-0.0007	7.042×10^{-5}	< 0.0001

Following the model we have developed, we need to consider the problem of heteroskedasticity existing in our short panel data as shown in figure 2. We have tried out different approaches, such as heteroskedasticity robust estimator, heteroskedasticity-autocorrelation (HAC) estimator and clustered robust estimator. The results are summarized in table 4, from which we can conclude that the clustered robust model provides a relatively conservative estimation for our model. Therefore, we will adopt the result as stated in table 5, and extend this fitting method to the following models for fixed effects, random effects and other relevant fitting processes.

Figure 2: Residuals plot to check heteroskedasticity for model 3.2

Table 4. Comparison of robust estimation results

Estimation	F-statistic	p-value	Distribution
Non-robust	162.81	< 0.0001	$F(6, 1721)$
Heteroskedastic robust	113.38	< 0.0001	$F(6, 1721)$
HAC robust	1.201×10^6	< 0.0001	$F(6, 1721)$
Clustered robust	44.533	< 0.0001	$F(6, 1721)$

Table 5. Clustered-robust Pooled OLS result for model 3.3

Variable	Parameter	Standard error	p-value
constant	-0.5964	0.1401	< 0.0001
$SIZE$	0.0422	0.0068	< 0.0001
$PROFITABILITY$	-0.4442	0.1056	< 0.0001
$TANG$	0.1932	0.0519	0.0002
$LIQUID$	-0.0545	0.0076	< 0.0001
$MCAP$	-8.842×10^{-12}	-4.09×10^{-12}	0.0308
$SOLV$	-0.0007	0.0001	< 0.0001

3.2. Fixed Effects

Due to the nature of our dataset with panel futures, we will investigate if there are any one-way or two-way fixed effects in our model 3.4 where μ_i and λ_t specify the entity and time effects:

$$LEVERAGE_{it} = \alpha + \beta_1 SIZE_{it} + \beta_2 PROFITABILITY_{it} + \beta_3 TANG_{it} + \beta_4 LIQUID_{it} + \beta_5 MCAP_{it} + \beta_6 SOLV_{it} + \mu_i + \lambda_t + v_{it} \tag{3.4}$$

With the clustered robust estimation method introduced in the previous part, we have the null hypothesis 3.5 to 3.9 for joint F-test, conditional F-test for entity and time effect respectively, and marginal F-test for entity and time effect respectively:

$$H_0: \ \mu_1 = \mu_2 = \cdots = 0 \ \text{ and } \ \lambda_1 = \lambda_2 = \cdots = 0 \tag{3.5}$$

$$H_0: \ \mu_1 = \mu_2 = \cdots = 0 \ \text{ given } \ \lambda_1 = \lambda_2 = \cdots = \lambda_T \tag{3.6}$$

$$H_0: \ \lambda_1 = \lambda_2 = \cdots = 0 \ \text{ given } \ \mu_1 = \mu_2 = \cdots = \mu_n \tag{3.7}$$

$$H_0: \ \mu_1 = \mu_2 = \cdots = 0 \ \text{ allowing } \ \lambda_t\text{s to differ} \tag{3.8}$$

$$H_0: \ \lambda_1 = \lambda_2 = \cdots = 0 \ \text{ allowing } \ \mu_i\text{s to differ} \tag{3.9}$$

And the results of all the testing models are summarized in the table 6:

Table 6. Results for Testing Fixed Effects

Test	F-statistics	p-value	Distribution
(3.5)	25.3104	< 0.0001	$F(222, 1499)$
(3.6)	25.7635	< 0.0001	$F(215, 1506)$
(3.7)	1.0642	0.3842	$F(7, 1714)$
(3.8)	25.9912	< 0.0001	$F(215, 1499)$
(3.9)	3.2219	0.0021	$F(7, 1499)$

From the result, it can be concluded that the time effect under conditional and marginal tests are not significant, at which we do not reject the null hypothesis at the confidence level of 99.99%, leading to the conclusion where there is no strong indication of time effect in our model, whereas the entity effect showcases a strong significance for all the tests. Following the methodology provided in Pooled OLS part, we want to include the cluster robust estimation, based on either entity cluster or time cluster. The F-statistics are shown in table 7:

Table 7. Results for robust fixed effects estimation

Estimation	F-statistic	p-value	Distribution
Non-robust	79.286	< 0.0001	$F(6, 1506)$
Entity cluster robust	18.662	< 0.0001	$F(6, 1506)$
Time cluster robust	1708.8	< 0.0001	$F(6, 1506)$

From this result, we can conclude that the entity cluster robust estimation will provide a relatively conservative result. Hence, we will conclude, for the fixed effects part, the best model we have is stated as model 3.17:

$$\begin{aligned} LEVERAGE_{it} = \alpha + \beta_1 SIZE_{it} + \beta_2 PROFITABILITY_{it} + \beta_3 TANG_{it} + \beta_4 LIQUID_{it} \\ + \beta_5 MCAP_{it} + \beta_6 SOLV_{it} + \mu_i + v_{it} \end{aligned} \tag{3.10}$$

The estimated parameters with entity cluster effect true are listed in table 8:

Table 8. Clustered-robust Fixed Effect result for model 3.10

Variable	Parameter	Standard error	p-value
constant	-1.559	0.4675	0.0009
$SIZE$	0.0834	0.0238	0.0005
$PROFITABILITY$	-0.2270	0.0543	< 0.0001
$TANG$	0.4346	0.0807	< 0.0001
$LIQUID$	-0.0228	0.0062	0.0003
$MCAP$	-4.195×10^{-12}	3.294×10^{-12}	0.2030
$SOLV$	-0.0002	0.0001	0.0808

The result implies that, except for MCAP and SOLV, the variables are significant at the confidence level of 99.9%. The impact on capital structure based on leverage level of these variables is generally strong, with each of firm's own effect and heteroskedasticity feature embedded within each of the firm.

3.3. Random Effects

Whilst fixed effects models are typically used to estimate models of capital structures, we will still conduct our own investigation into the suitability of random effects models on our data even if it is just to eliminate them as a possibility. To do so, we must first estimate for the 1-way entity and/or time random effects models using the same specification as the fixed effects models prior. The model specification is listed below:

$$\begin{aligned} LEVERAGE_{it} = \alpha + \beta_1 SIZE_{it} + \beta_2 PROFITABILITY_{it} + \beta_3 TANG_{it} + \beta_4 LIQUID_{it} \\ + \beta_5 MCAP_{it} + \beta_6 SOLV_{it} + \mu_i + v_{it} \end{aligned} \tag{3.11}$$

After estimating the basic iterations of random effects model, we check for the preferential order of these models in order to get a sense on how to proceed with any comparative process with the fixed effects models estimated in the 3.2. To get an indication about the power of including time or/and entity random effects, we apply the Conditional, Marginal and Joint Lagrange Multiplier Tests onto our model to test for the null hypothesis of 3.12 to 3.13.

$$H_0: \quad \sigma_\mu^2 = \sigma_\lambda^2 = 0 \tag{3.12}$$

$$H_0: \quad \sigma_\mu^2 = 0 \ \text{ given } \ \sigma_\lambda^2 = 0 \tag{3.13}$$

$$H_0: \quad \sigma_\lambda^2 = 0 \ \text{ given } \ \sigma_\mu^2 = 0 \tag{3.14}$$

$$H_0: \quad \sigma_\mu^2 = 0 \ \text{ given } \ \sigma_\lambda^2 > 0 \tag{3.15}$$

$$H_0: \quad \sigma_\lambda^2 = 0 \ \text{ given } \ \sigma_\mu^2 > 0 \tag{3.16}$$

And obtain the following test results:

Table 9. Results for Testing Random Effects

Test	χ^2-statistic	p-value	Distribution
(3.12)	2931.4558	< 0.0001	$\chi^2(2)$
(3.13)	2931.4559	< 0.0001	$\chi^2(1)$
(3.14)	0.0212	0.8842	$\chi^2(1)$
(3.15)	0.8280	0.3629	$\chi^2(1)$
(3.16)	-0.8312	< 0.0001	$\chi^2(1)$

From the LM test statistics obtained, we accept the significance of including either only the entity effects or the significance of including both entity and time effects jointly. However, as we can observe from the insignificant conditional LM test for time effects, the time effects only model is not impactful over the vanilla pooled OLS estimation that we did prior. Unfortunately, the marginal LM test result is unable to give us an exact conclusion as its χ^2-statistic is negative which is not possible, thus, likely indicating an issue with the RandomEffects module in python used to estimate the time effects model used for in the test. Despite this misfortune, it would be reasonable, considering the insignificance of the conditional LM test along with the extremely similar joint and entity-conditional LM tests to conclude that the high significance of the joint LM test is likely due to the sole, large contributions of the entity random effects. We will therefore conclude that it is extremely likely that the 1-way entity random effects model is not only sufficient, but also preferable over the joint effects model.

However, knowing that the entity random effects model holds strong explanatory powers does not help us judge its favorability against the 1-way fixed effects model produced prior. To do so, we leverage upon the Hausman test to compare the 2 estimation techniques. When comparing the unadjusted time and/or entity models, for the null hypothesis of a random effects preference, we obtain the following test results:

Table 10. Results for Hausman Test of Random vs. Fixed Effects

Estimation	χ^2-statistic	p-value	Distribution
Entity & Time	67.0179	5.9048×10^{-12}	$\chi^2(7)$
Entity	128.1244	< 0.0001	$\chi^2(7)$
Time	7.6988	0.3599	$\chi^2(7)$

Both the joint and time effects models were also tested for preferential order of random effects vs. fixed effects specifications as there is no disadvantage to doing so. In fact, we can conclude that both 2-way and entity fixed effects models were preferred over their counterpart random effects models, giving us even greater validation about the sufficiency of entity effects in either models. Additionally, the significance of the time random effects model over the time fixed effects model indicates something interesting. That there might possibly be traces of time random effects captured inside our data. Of course, the comparison of different vanilla random vs. fixed effects models will warrant some discussion but it should be reminded that our focus thereafter still lies in the capture of entity fixed effects within our model specifications rather than in the exploration of either time or 2-way effect models. As such, the main conclusion of this section is that we must reject the null hypothesis of the Hausman test upon entity effects and conclude that the fixed effects estimation is preferred to the random effects estimation for entity effects models.

3.4. Correlated Random Effects

Despite our earlier conclusion, we cannot know from the value of the Hausman test statistic, the magnitude of preference of the entity fixed-effects model. Additionally, as discussed prior, the significance of the time random effects model likely indicates that the entity fixed effects preference is not strict, and that we must find some way to model for both the entity fixed effects as well as the potential traces of time random effects in our data. Coincidentally, we have the option of trying the correlated random effects model which allows us to partially control for the within cluster effects of our model through the inclusion of cluster-means, whilst providing the flexibility to account for any random effects as well. To do so, we will estimate the following entity effects correlated random model:

$$LEVERAGE_{it} = \alpha + \beta_1 SIZE_{it} + \beta_2 PROFITABILITY_{it} + \beta_3 TANG_{it} + \beta_4 LIQUID_{it} + \beta_5 MCAP_{it} + \beta_6 SOLV_{it} + \beta_7 \overline{SIZE_i} + \beta_8 \overline{PROFITABILITY_i} + \beta_9 \overline{TANG_i} + \beta_10 \overline{LIQUID_i} + \beta_11 \overline{MCAP_i} + \beta_12 \overline{SOLV_i} + \epsilon_i \tag{3.17}$$

Unlike in prior specifications, the entity cluster-means of each exogenous regressor is included in the regression. After we produce the above model, we apply the Wald-test to test for its significance against the 1-way entity random effects model and get the following result:

Table 11. Results for Wald Test of Correlated Random vs. Random Effects

Estimation	χ^2-statistic	p-value	Distribution
Entity	78.1680	< 0.0001	$\chi^2(6)$

From the Wald-test, the inclusion of the time-invariant regressors allows for the partial control of the fixed effects inherent to the model and is thus preferred to the random effects model. This is of course expected, as the preferential order of the 3 estimation methods should in theory be fixed ¿ correlated random ¿ random if our hypothesis of significant but small traces of time random effects is correct. unfortunately, this test result also creates a minor headache for us as it is mathematically impossible to conduct a comparative test for a fixed effects specification against a correlated random effects model. Thus, we do not actually know if the correlated random effects model is preferred to the fixed effects estimate as it is possible our little hypothesis is wrong and that the preference of the correlated random effects model to the random effects model is due to the partial capture of the within-entity effects, and that the inclusion of time random effects, where there is none, spoils its preference to the fixed effects model. Thus, we will have to apply a little imagination and argumentative logic to deciding on which model to present.

Firstly, the major advantage that the correlated random effects model has over the fixed effects model is that it allows for partial control of both fixed and random effects simultaneously. Unless we are sure that the data only has fixed effects, or that the random effects are negligible, it would be wise to prefer a correlated random effects model for its flexibility instead. In applying this logic, we are trying to account for the possibility rather that the probability of have time random effects. Fortunately for us, we can see that the cost of this logic is extremely low, especially as our 1-way individual random effects model has the following specification:

Table 12. Entity Correlated Random Effects result for model 3.17

Variable	Parameter	Standard error	p-value
constant	-1.559	0.1571	0.0006
$SIZE$	0.0834	0.0103	< 0.0001
$PROFITABILITY$	-0.2270	0.0363	< 0.0001
$TANG$	0.4346	0.0364	< 0.0001
$LIQUID$	-0.0228	0.0022	< 0.0001
$MCAP$	-4.195×10^{-12}	4.13×10^{-12}	0.3100
$SOLV$	-0.0002	5.636×10^{-5}	0.0005
$\overline{SIZE}$	-0.0413	0.0130	0.0015
$\overline{PROFITABILITY}$	-0.5395	0.2383	0.0237
$\overline{TANG}$	-0.2530	0.0756	0.0008
$\overline{LIQUID}$	-0.0482	0.0089	< 0.0001
$\overline{MCAP}$	-4.719×10^{-12}	6.212×10^{-12}	0.4476
$\overline{SOLV}$	-0.0006	0.0002	0.0114

Before we conclude, however, we must check if the correlated random effects model has managed to control the cluster correlations sufficiently.

Table 13. Comparison of robust estimation results

Estimation	F-statistic	p-value	Distribution
Non-robust	53.744	< 0.0001	$F(12, 1721)$
Entity cluster robust	36.989	< 0.0001	$F(12, 1715)$
Time cluster robust	7.977×10^{15}	< 0.0001	$F(12, 1715)$

As the specification retains almost all its overall significance, there really does not seem to be a great cost to using the correlated random effects specification over the fixed effects model. However, like before, this is not the end, and we must conduct a couple of robustness tests to proceed.

Like in the cluster robust fixed effects estimator, it appears that there are still entity cluster correlations present and unaccounted for in the model whereas an application of a time cluster robust method has only served to inflate the confidence of the model. Considering that the entity cluster robust method retains its overall significance at the extremely conservative 0.1% level, it only makes sense to report the following estimation of the cluster robust method:

Table 14. Entity Cluster-robust Entity Correlated Random Effects result for model 3.17

Variable	Parameter	Standard error	p-value
constant	-0.5374	0.1433	0.0002
$SIZE$	0.0834	0.0238	0.0005
$PROFITABILITY$	-0.2270	0.0544	< 0.0001
$TANG$	0.4346	0.0808	< 0.0001
$LIQUID$	-0.0228	0.0062	0.0003
$MCAP$	-4.195×10^{-12}	3.3×10^{-12}	0.2038
$SOLV$	-0.0002	0.0001	0.0813
$\overline{SIZE}$	-0.0413	0.0242	0.0877
$\overline{PROFITABILITY}$	-0.5395	0.2074	0.0094
$\overline{TANG}$	-0.2530	0.0986	0.0104
$\overline{LIQUID}$	-0.0482	0.0100	< 0.0001
$\overline{MCAP}$	-4.719×10^{-12}	3.488×10^{-12}	0.1763
$\overline{SOLV}$	-0.0006	0.0002	0.0034

Thus, we can conclude that the above specification provides a good inferential model on capital structure determinants of SGX listed companies over our overall dataset, and that most traditionally capital structure models will retain their significance when applied as so.

4. Conclusion and Limitations

4.1. Overall Interpretation

From the analysis, we conclude that there are a few notable financial factors that determines a firm's capital structure choice represented by net debt over total assets. The most significant ones and their explanations are as follows:

1. Natural log of total assets as the firm's size: A firm tends to increase the leverage level since they have the channel and larger pool for investors to invest in its debt-related financing activities.

2. EBITDA over total assets as profitability: A firm with a higher profitability power tends to borrow less in terms of financing since internal financing will be the most preferred with its sufficient capital pool inside the firm according to pecking order theory as proposed by Modigliani and Miller (1958).

3. Fixed assets over total assets as the tangibility of the firm's assets: A higher tangibility indicates a larger pool as collateral for loans or bond instruments. Therefore, the firms with more tangible assets will have more access to capital markets for debt financing.

4. Current ratio as the firm's liquidity to pay its short-term obligations: With a higher liquidity ratio, a firm has relatively smaller short-term obligations compared to the total assets, resulting in a lower level of indicated leverage ratio.

5. Interest coverage ratio as the firm's solvency ratio: A higher interest coverage ratio also indicates that the firm has sufficient capital for internal financing, leading to a less leveraged choice for capital structure.

6. Time invariance of profitability, tangibility, liquidity and solvency: It appears that SGX listed companies do not adjust these ratios to match the economic situations across years.

4.2. COVID Interpretation

However, the model was run across the entire data set of financial reports inclusive from 2016 to 2022. Due to the substantial effect of operating, pricing and all various strategies regarding a firm's operations by COVID from 2020 to 2022, we decided to split the dataset into 2 different segments by years, where the first segment is COVID-influenced segment and the second one is Non-COVID-influenced segment. For the different segments, we fitted the same

entity-clustered-robust correlated random effects model as stated in 3.17, and investigated to see if the parameters estimated vary by a large extent.

Table 15. COVID segmented model 3.17

Variable	Parameter	Standard error	p-value
constant	-0.4151	0.1587	0.0091
$SIZE$	0.0315	0.0417	0.4499
$PROFITABILITY$	-0.3764	0.1005	0.0002
$TANG$	0.5304	0.1285	< 0.0001
$LIQUID$	-0.0160	0.0057	0.0051
$MCAP$	-2.676×10^{-11}	1.068×10^{-11}	0.0125
$SOLV$	-7.34×10^{-5}	0.0001	0.5001
$\overline{SIZE}$	0.0027	0.0418	0.0647
$\overline{PROFITABILITY}$	0.3477	0.2045	0.0896
$\overline{TANG}$	-0.2970	0.1455	0.0.0416
$\overline{LIQUID}$	-0.0436	0.0066	< 0.0001
$\overline{MCAP}$	1.895×10^{-11}	1.123×10^{-11}	0.0920
$\overline{SOLV}$	-0.0006	0.0002	0.0143

Table 16. Non-COVID segmented model 3.17

Variable	Parameter	Standard error	p-value
constant	-0.6163	0.1552	0.0001
$SIZE$	0.1497	0.0256	< 0.0001
$PROFITABILITY$	-0.0551	0.0601	0.3601
$TANG$	0.3753	0.0927	0.0001
$LIQUID$	-0.0255	0.0068	0.0002
$MCAP$	-6.98×10^{-12}	3.539×10^{-12}	0.0488
$SOLV$	-0.0002	$9,271 \times 10^{-5}$	0.0417
$\overline{SIZE}$	-0.1036	0.0258	0.0001
$\overline{PROFITABILITY}$	-0.5980	0.2166	0.0059
$\overline{TANG}$	-0.2217	0.1087	0.0416
$\overline{LIQUID}$	-0.0474	0.0135	0.0004
$\overline{MCAP}$	-2.115×10^{-12}	2.962×10^{-12}	0.4755
$\overline{SOLV}$	-0.0006	0.0001	< 0.0001

As it turns out that our expectation is correct and there are a few, significant changes in the estimation results for two different segments aforementioned as shown in table 15 and 16. The noteworthy changes and possible reasons for their changes are as follows:

1. $SIZE$ becomes insignificant. During the COVID period, everyone was equally affected and larger firms were increasingly reliant on debt financing as well.

2. $PROFITABILITY$ stays highly significant. During the COVID period, firms that were profitable could choose to rely less upon debt, whereas, profitability did not give a clear indication of the firm's leverage structure.

3. $SOLV$ shifts to being insignificant. Likely due to widespread impact on the market outlook and increasing reliance upon debt, interest coverage ratio could no longer be associated with lower leverage structures.

4. Time-invariant $SIZE$ becomes insignificant. Like with the size parameter, firm's debt reliance behaviour no longer depended upon its size as most firms needed to increase debt financing reliance regardless. It thus, became impossible to observe the firm-specific effects on the model during this period.

Thus, it should be noted that the use of COVID separated results to understand the determinants of capital structure models of SGX-listed firms is crucial given the strong impact that

COVID had on companies, and the fact that the estimation results for segmented and overall data had a few, but not insignificant changes to the coefficient significance. Unfortunately, this research paper is constrained by time and cannot was unable to address all concerns and results found by our investigations. For instance, the weird and unexpected results upon our overall model's $LIQUID$ and $SOLV$ coefficients indicate that there might be possible endogeneity in our model, possibly indicating that the use of instrumental estimation techniques are warranted. Furthermore, when we ran our model across industry sector segmented data sets we ran into issues with the covariance matrices used in the correlated random effects estimations, hinting that our model may suffer from partial multicollinearity that became perfect when the data size was reduced. Lastly, the initial size of our data set and the final dimensions of the reduced data set once missing values were handled will and should provoke discussions on how missing data could be better handled. The implementation of unbalanced panel data estimation modules for instance, or the use of other data augmentation techniques like mean-filling, or even data predictive interpolation are all ideas that warrant further investigation in future papers.

References

[1] Baharumshah, A. Z., Soon, S.-V., & Wohar, M. E. (2019). Phillips curve for the Asian Economies: A nonlinear perspective. *Emerging Markets Finance and Trade, 57*(12), 3508–3537. https://doi.org/10.1080/1540496x.2019.1699789

[2] Bayrakdaroglu, A., Ege, I., & Yazici, N. (2013). A panel data analysis of Capital Structure Determinants: Empirical results from Turkish Capital Market. *International Journal of Economics and Finance, 5*(4). https://doi.org/10.5539/ijef.v5n4p131

[3] Gaud, P., Jani, E., Hoesli, M., & Bender, A. (2005). The capital structure of Swiss companies: An empirical analysis using dynamic panel data. *European Financial Management, 11*(1), 51–69. https://doi.org/10.1111/j.1354-7798.2005.00275.x

[4] Modigliani, F., & Miller, M. H. (1958). The Cost of Capital, Corporation Finance and the Theory of Investment. *The American Economic Review, 48*(3), 261–297.

[5] M'ng, J. C., Rahman, M., & Sannacy, S. (2017). The determinants of capital structure: Evidence from public listed companies in Malaysia, Singapore and Thailand. *Cogent Economics & Finance, 5*(1), 1418609. https://doi.org/10.1080/23322039.2017.1418609

[6] Neves, M. E., Serrasqueiro, Z., Dias, A., & Hermano, C. (2020). Capital structure decisions in a period of economic intervention. *International Journal of Accounting & Information Management, 28*(3), 465–495. https://doi.org/10.1108/ijaim-08-2019-0094

5. Appendix

5.1. Code for data cleaning

```
import pandas as pd
import numpy as np
import re

# Read raw data file
sgx = pd.read_excel("data/raw_sgx_data.xlsx", index_col= None)
sgx
sgx = sgx.replace({"n.a.": np.nan, "n.s.": np.nan, 0: np.nan})
sgx

# Removing "\nth" pattern and ".1" pattern (for duplicated column names) from
                                        the column names
sgx.columns = pd.Index([re.sub(pattern= "\.\d?\$", repl= "", string=
                               re.sub(pattern= " ?\n(th)? ?", repl= " ", string
                                                                    = col))
                                                                    for col
                                                                    in sgx.
                                                                    columns]
                                                                    )

sgx = sgx.loc[:, ~sgx.columns.duplicated()]

# Lengthening dataframe using columns with time periods
id_vars = ['Company name Latin alphabet', 'NACE Rev. 2, core code (4 digits)',
                                        'Consolidation code']
melted_sgx = pd.melt(sgx, id_vars= id_vars)

# Extracting year value from financial statement variable strings
melted_sgx["year"] = melted_sgx["variable"].str.extract(r"(\d{4})\$")[0]
melted_sgx["variable"] = melted_sgx["variable"].str.replace("\s+\d{4}\$|USD", "
                                        ", regex= True)
melted_sgx["variable"] = melted_sgx["variable"].str.strip(" ")
melted_sgx

# Widening dataframe using unique financial statement variables
pivoted_sgx = melted_sgx.pivot(index= id_vars + ["year"], columns= "variable",
                                        values= "value").reset_index().
                                        rename_axis(None, axis=1)
```

```
pivoted_sgx

# Dropping columns without any data
pivoted_sgx = pivoted_sgx.dropna(axis= 1, how= 'all')
pivoted_sgx.to_csv("data/sgx_data.csv", index= False)
sgx = pd.read_csv("data/sgx_data.csv")
sgx
cpi = pd.read_csv("data/sg_cpi.csv")
cpi

# Join the cpi and sgx data
sgx = pd.merge(left= sgx, right= cpi, left_on= 'year', right_on= 'Year', how= '
                                        left').drop(columns= 'Year')
sgx
sibor = pd.read_csv("data/sg_ir.csv", index_col= None)
sibor

# Use rates depending on the year as after 2014, SIBOR is defunct while SORA
                                        did not start till 2005
sibor['Singapore Interbank Rates'] = np.where(sibor['Year'] < 2014, sibor['
                                        SIBOR'], sibor['SORA']).astype('float')
sibor
# Get average annual interbank rates
sibor['Singapore Interbank Rates'] = sibor.groupby('Year')['Singapore Interbank
                                         Rates'].transform('mean')
sibor = sibor.drop_duplicates(subset= 'Year')[['Year', 'Singapore Interbank
                                        Rates']].reset_index()
sibor
# Join the interbank rates to the sgx data
sgx = pd.merge(left= sgx, right= sibor, left_on= 'year', right_on= 'Year', how=
                                         'left').drop(columns= 'Year')
sgx

list(sgx.columns)
# Select the fields of possible interest in the model
selected_fields = ['Company name Latin alphabet',
                   'NACE Rev. 2, core code (4 digits)',
                   'year',
                   'Total assets',
                   'Total current assets',
                   'Net property, plant & equipment',
                   'Total liabilities',
```

```
                    'Total current liabilities',
                    'Net debt',
                    'Income tax expense',
                    'Interest Expense',
                    'Total shareholders\' equity',
                    'Net profit',
                    'Net Sales',
                    'EBITDA',
                    'Earnings before interest & tax (EBIT)',
                    'Total depreciation, amortization & depletion',
                    'Net Cash from Operating Activities',
                    'Additions to Fixed Assets',
                    'Market capitalisation m',
                    'Price / earnings ratio - close',
                    'Interest coverage',
                    'Number of employees',
                    'Singapore Interbank Rates',
                    'Inflation Rate']
filtered_sgx = sgx[selected_fields]
# Count the number of usable records in the selected fields
filtered_sgx.apply(axis= 0, func= lambda x: x.count())
# Count the number of usable records in the selected fields
filtered_sgx.apply(axis= 0, func= lambda x: x.count())

# Create the column rename mapping dictionary
colname_mapping = {
    'Company name Latin alphabet' : 'Company',
    'NACE Rev. 2, core code (4 digits)' : 'NACE',
    'year' : 'Year',
    'Total assets': 'TA',
    'Total current assets': 'CA',
    'Net property, plant & equipment': 'FA',
    'Total liabilities': 'TL',
    'Total current liabilities': 'CL',
    'Net debt': 'DEBT',
    'Income tax expense': 'TAXEX',
    'Interest Expense': 'INTEX',
    'Total shareholders\' equity': 'TE',
    'Net profit': 'PROFIT',
    'Net Sales': 'SALES',
    'EBITDA': 'EBITDA',
    'Earnings before interest & tax (EBIT)': 'EBIT',
```

```
    'Total depreciation, amortization & depletion': 'DA',
    'Net Cash from Operating Activities': 'CFO',
    'Additions to Fixed Assets': 'CAPEX',
    'Market capitalisation m': 'MCAP',
    'Price / earnings ratio - close': 'PER',
    'Interest coverage': 'SOLV',
    'Number of employees': 'NEMP',
    'Singapore Interbank Rates': 'SORA',
    'Inflation Rate': 'CPI'
}

# Rename the fields
filtered_sgx = filtered_sgx.rename(columns = colname_mapping)
filtered_sgx

# Dropping groups which do not contain any information
fields_to_check = ["TA", "CA", "FA",
                   "TL", "CL", "DEBT", "TAXEX", "INTEX",
                   "TE", "PROFIT", "SALES", "EBITDA", "EBIT",
                   "DA", "CFO", "CAPEX", "MCAP", "PER", "SOLV",
                   "NEMP"]

filtered_sgx = filtered_sgx.groupby(by= "Company")
filtered_sgx = filtered_sgx.apply(lambda x: x if not x[fields_to_check].isna().
                                        all().all() else None)
filtered_sgx = filtered_sgx.reset_index(drop= True)
filtered_sgx

# Check the number of groups left
filtered_sgx.groupby(by= "Company").ngroups
# Count the number of un-usable records in the selected fields
filtered_sgx.apply(axis= 0, func= lambda x: x.isna().sum())
filtered_sgx.to_csv("data/filtered_sgx.csv", index= False)
sgx = pd.read_csv("data/filtered_sgx.csv", index_col= None)
sgx
# Encode Companies with a unique numeric identifier
sgx['Company Code'] = sgx['Company'].factorize()[0]
sgx['Company Code']
# Extract first 2 digits of the NACE field
sgx["NACE"] = sgx["NACE"].astype(str).str[:2]
sgx["NACE"]
# Reverse sign of depreciation, amortisation and depletion
```

```
sgx["DA"] = np.absolute(sgx["DA"])
sgx["DA"]
# Re-scale Market Capitalisation to units
sgx["MCAP"] = sgx["MCAP"] * 1000000
sgx["MCAP"]
# Measure of Firm Size
sgx["SIZE"] = np.log(sgx["TA"])
sgx["SIZE"]
# Measure of Profitability
sgx["PROFITABILITY"] = sgx["EBITDA"] / sgx["TA"]
sgx["PROFITABILITY"]
# Tangibility
sgx["TANG"] = sgx["FA"] / sgx["TA"]
sgx["TANG"]
# Measure of Business Risk
sgx["RISK"] = sgx.groupby(by= "Company")["EBIT"].pct_change() / sgx.groupby(by=
                                        "Company")["SALES"].pct_change()
sgx["RISK"]
# Ability to pay debts (Current Ratio)
sgx["LIQUID"] = sgx["CA"] / sgx["CL"]
sgx["LIQUID"]
# Debt Tax Shield
sgx["NDTSHIELD"] = sgx["DA"] / sgx["TA"]
sgx["NDTSHIELD"]
# Interest Expense
sgx["INTEX"] = sgx["EBIT"] - (sgx["PROFIT"] + sgx["TAXEX"])
sgx["INTEX"]
# Effective Tax Rate
sgx["TAXRATE"] = sgx["TAXEX"] / (sgx["EBIT"] - sgx["INTEX"])
sgx["TAXRATE"]
# Free  Cash Flow to Firm
sgx["FCFF"] = sgx["CFO"] + (sgx["INTEX"] * (1 - sgx["TAXRATE"])) - sgx["CAPEX"]
sgx["FCFF"]
# Growth Potential
sgx["GROWPOT"] = sgx["MCAP"] / sgx["TE"]
sgx["GROWPOT"]
# Dependent Variable
sgx["LEVERAGE"] = sgx["DEBT"] / sgx["TA"]
sgx["LEVERAGE"]
# Count the number of usable records in the selected fields
sgx.apply(axis= 0, func= lambda x: x.count())
```

```
## Years to select
years_to_select = [2015, 2016, 2017, 2018, 2019, 2020, 2021, 2022]
year_filtered_sgx = sgx[sgx['Year'].isin(years_to_select)]
year_filtered_sgx
# Count the percentage of un-usable records in the selected fields
year_filtered_sgx.apply(axis= 0, func= lambda x: (x.isna().sum() / x.count()) *
                                        100)
# PER and NEMP are almost unusable
year_filtered_sgx = year_filtered_sgx.drop(columns= ["PER", "NEMP"])
year_filtered_sgx
# Drop Companies with any NaN in data
year_filtered_sgx = year_filtered_sgx.groupby("Company")
year_filtered_sgx = year_filtered_sgx.apply(lambda x: x if not x.isna().any().
                                        any() else None)
year_filtered_sgx = year_filtered_sgx.reset_index(drop= True)
year_filtered_sgx
# Count the number of usable companies remaining in the data
year_filtered_sgx.groupby(by= "Company").size()
# Filter for COVID impacted years
covid_years = [2020, 2021, 2022]
covid_sgx = year_filtered_sgx[year_filtered_sgx['Year'].isin(covid_years)]
covid_sgx
# Filter for non-COVID impacted years
normal_sgx = year_filtered_sgx[~year_filtered_sgx['Year'].isin(covid_years)]
normal_sgx
```

5.2. Code for Pooled OLS

```
import numpy as np
import pandas as pd
import statsmodels.api as sm
from linearmodels.panel import PooledOLS
from statsmodels.iolib import save_pickle
import seaborn as sns
import matplotlib.pyplot as plt

sgx = pd.read_csv("data/clean_sgx.csv", index_col= None)
sgx.columns
# Set index to indicate that the data has both cross-sectional and time panels
sgx = sgx.set_index(['Company', 'Year'], drop= False)
sgx
```

```
sgx['Company'], sgx['Company Code'], sgx['Year'] = pd.Categorical(sgx['Company'
                                        ]), sgx['Company Code'], sgx['Year']
sgx
exog_vars = ['SIZE',
             'PROFITABILITY',
             'TANG',
             'RISK',
             'LIQUID',
             'NDTSHIELD',
             'MCAP',
             'FCFF',
             'GROWPOT',
             'SOLV',
             'SORA',
             'CPI']

exog = sgx[exog_vars]

corr_matrix = exog.corr()

# Create heatmap
plt.figure(figsize= (10, 8))
sns.heatmap(corr_matrix, annot= True, cmap= 'coolwarm', fmt= ".2f", linewidths=
                                          0.5)
plt.title('Correlation Matrix Heatmap of Exogenous Variables')
plt.show()
endo_var = 'LEVERAGE'
exog_vars = ['SIZE',
             'PROFITABILITY',
             'TANG',
             'RISK',
             'LIQUID',
             'NDTSHIELD',
             'MCAP',
             'FCFF',
             'GROWPOT',
             'SOLV',
             'SORA',
             'CPI']

endo = sgx[endo_var]
exog = sm.add_constant(sgx[exog_vars])
```

```
pooled_mod = PooledOLS(dependent = endo,
                       exog = exog)
pooled_res = pooled_mod.fit()
print(pooled_res.summary)
# 1st drop of FCFF
endo_var = 'LEVERAGE'
exog_vars = ['SIZE',
             'PROFITABILITY',
             'TANG',
             'RISK',
             'LIQUID',
             'NDTSHIELD',
             'MCAP',
             'GROWPOT',
             'SOLV',
             'SORA',
             'CPI']

endo = sgx[endo_var]
exog = sm.add_constant(sgx[exog_vars])
pooled_mod = PooledOLS(dependent = endo,
                       exog = exog)
pooled_res = pooled_mod.fit()
print(pooled_res.summary)
# 2nd drop of NDTS
endo_var = 'LEVERAGE'
exog_vars = ['SIZE',
             'PROFITABILITY',
             'TANG',
             'RISK',
             'LIQUID',
             'MCAP',
             'GROWPOT',
             'SOLV',
             'SORA',
             'CPI']
endo = sgx[endo_var]
exog = sm.add_constant(sgx[exog_vars])
pooled_mod = PooledOLS(dependent = endo,
                       exog = exog)
pooled_res = pooled_mod.fit()
print(pooled_res.summary)
```

```
# 3rd drop of CPI
endo_var = 'LEVERAGE'
exog_vars = ['SIZE',
             'PROFITABILITY',
             'TANG',
             'RISK',
             'LIQUID',
             'MCAP',
             'GROWPOT',
             'SOLV',
             'SORA']

endo = sgx[endo_var]
exog = sm.add_constant(sgx[exog_vars])
pooled_mod = PooledOLS(dependent = endo,
                       exog = exog)
pooled_res = pooled_mod.fit()
print(pooled_res.summary)
# 4th drop of SORA
endo_var = 'LEVERAGE'
exog_vars = ['SIZE',
             'PROFITABILITY',
             'TANG',
             'RISK',
             'LIQUID',
             'MCAP',
             'GROWPOT',
             'SOLV']
endo = sgx[endo_var]
exog = sm.add_constant(sgx[exog_vars])
pooled_mod = PooledOLS(dependent = endo,
                       exog = exog)
pooled_res = pooled_mod.fit()
print(pooled_res.summary)
# 5th drop of GROWPOT
endo_var = 'LEVERAGE'
exog_vars = ['SIZE',
             'PROFITABILITY',
             'TANG',
             'RISK',
             'LIQUID',
             'MCAP',
```

```
              'SOLV']
endo = sgx[endo_var]
exog = sm.add_constant(sgx[exog_vars])
pooled_mod = PooledOLS(dependent = endo,
                       exog = exog)
pooled_res = pooled_mod.fit()
print(pooled_res.summary)
# Final and 6th drop of RISK
endo_var = 'LEVERAGE'
exog_vars = ['SIZE',
              'PROFITABILITY',
              'TANG',
              'LIQUID',
              'MCAP',
              'SOLV']
endo = sgx[endo_var]
exog = sm.add_constant(sgx[exog_vars])
pooled_mod = PooledOLS(dependent = endo,
                       exog = exog)
pooled_res = pooled_mod.fit()
print(pooled_res.summary)
# Plot the residuals from the final model to check for heteroskedasticity
plt.figure(figsize= (10, 8))
plt.scatter(pooled_res.resids, pooled_res.predict())
plt.title('Residuals vs Fitted Values Plot')
plt.xlabel('Fitted Values')
plt.ylabel('Residuals')
plt.show()
# Heteroskedastic Robust Estimator
endo_var = 'LEVERAGE'
exog_vars = ['SIZE',
              'PROFITABILITY',
              'TANG',
              'LIQUID',
              'MCAP',
              'SOLV']
endo = sgx[endo_var]
exog = sm.add_constant(sgx[exog_vars])
pooled_mod = PooledOLS(dependent = endo,
                       exog = exog)
pooled_res = pooled_mod.fit(cov_type = 'robust')
print(pooled_res.summary)
```

```
# HAC Estimator
endo_var = 'LEVERAGE'
exog_vars = ['SIZE',
             'PROFITABILITY',
             'TANG',
             'LIQUID',
             'MCAP',
             'SOLV']
endo = sgx[endo_var]
exog = sm.add_constant(sgx[exog_vars])
pooled_mod = PooledOLS(dependent = endo,
                       exog = exog)
pooled_res = pooled_mod.fit(cov_type = 'kernel')
print(pooled_res.summary)
# Cluster robust estimator
endo_var = 'LEVERAGE'
exog_vars = ['SIZE',
             'PROFITABILITY',
             'TANG',
             'LIQUID',
             'MCAP',
             'SOLV']
endo = sgx[endo_var]
exog = sm.add_constant(sgx[exog_vars])
pooled_mod = PooledOLS(dependent = endo,
                       exog = exog)
pooled_res = pooled_mod.fit(cov_type = 'clustered', cluster_entity= True)
print(pooled_res.summary)
# Whiten for correlation between time clusters
endo_var = 'LEVERAGE'
exog_vars = ['SIZE',
             'PROFITABILITY',
             'TANG',
             'LIQUID',
             'MCAP',
             'SOLV']
endo = sgx[endo_var]
exog = sm.add_constant(sgx[exog_vars])
pooled_mod = PooledOLS(dependent = endo,
                       exog = exog)
pooled_res = pooled_mod.fit(cov_type = 'clustered', cluster_time= True)
print(pooled_res.summary)
```

```
endo_var = 'LEVERAGE'
exog_vars = ['SIZE',
             'PROFITABILITY',
             'TANG',
             'LIQUID',
             'MCAP',
             'SOLV']
endo = sgx[endo_var]
exog = sm.add_constant(sgx[exog_vars])
pooled_mod = PooledOLS(dependent = endo,
                       exog = exog)
pooled_res = pooled_mod.fit(cov_type = 'clustered', cluster_entity= True,
                                        cluster_time= True)
print(pooled_res.summary)

exog_vars = ['SIZE',
             'PROFITABILITY',
             'TANG',
             'LIQUID',
             'MCAP',
             'SOLV']

exog = sgx[exog_vars]

corr_matrix = exog.corr()

# Create heatmap
plt.figure(figsize= (10, 8))
sns.heatmap(corr_matrix, annot= True, cmap= 'coolwarm', fmt= ".2f", linewidths=
                                        0.5)
plt.title('Coefficient Correlation Matrix Heatmap of Final Model')
plt.show()

# producing final model
endo_var = 'LEVERAGE'
exog_vars = ['SIZE',
             'PROFITABILITY',
             'TANG',
             'LIQUID',
             'MCAP',
             'SOLV']
endo = sgx[endo_var]
```

```
exog = sm.add_constant(sgx[exog_vars])
final_mod = PooledOLS(dependent = endo,
                      exog = exog)
final_res = pooled_mod.fit(cov_type = 'clustered', cluster_entity= True)
```

5.3. Code for Fixed Effects

```
import numpy as np
import pandas as pd

import scipy.stats as stats
import statsmodels.api as sm
from linearmodels.panel import PanelOLS

from statsmodels.iolib import load_pickle, save_pickle

sgx = sgx[['Company Code', 'Year', 'LEVERAGE',
           'SIZE', 'PROFITABILITY', 'TANG', 'LIQUID', 'MCAP', 'SOLV']]
sgx['year'] = pd.Categorical(sgx['Year'])
sgx['compcode'] = pd.Categorical(sgx['Company Code'])
sgx = sgx.set_index(['Company Code', 'Year'])

endo = sgx.LEVERAGE
exog_vars = ['SIZE', 'PROFITABILITY', 'TANG', 'LIQUID', 'MCAP', 'SOLV']
exog = sm.add_constant(sgx[exog_vars])

def fixedeffect(endo = endo, exog = exog,
                entity_eff = False, time_eff = False,
                cov_type = 'unadjusted', **cov_kwargs):
    mod = PanelOLS(endo, exog,
                   entity_effects=entity_eff,
                   time_effects=time_eff)
    res = mod.fit(cov_type= cov_type, **cov_kwargs)

    return res
# entity FE
entity_fe_res = fixedeffect(endo, exog, entity_eff= True)
print(entity_fe_res.summary)
# time FE
time_fe_res = fixedeffect(endo, exog, time_eff= True)
print(entity_fe_res.summary)
# two-way FE
```

```
tw_fe_res = fixedeffect(endo, exog, entity_eff= True, time_eff= True)
print(entity_fe_res.summary)
# tests for FE
print(tw_fe_res.f_pooled)
print(entity_fe_res.f_pooled)
print(time_fe_res.f_pooled)
exog_vars_marginal_entity = ['SIZE', 'PROFITABILITY', 'TANG', 'LIQUID', 'MCAP',
                                        'SOLV', 'year']
exog_marginal_entity = sm.add_constant(sgx[exog_vars_marginal_entity])
print(fixedeffect(endo, exog_marginal_entity, entity_eff= True).f_pooled)
exog_vars_marginal_time = ['SIZE', 'PROFITABILITY', 'TANG', 'LIQUID', 'MCAP', '
                                       SOLV', 'compcode']
exog_marginal_time = sm.add_constant(sgx[exog_vars_marginal_time])
print(fixedeffect(endo, exog_marginal_time, time_eff = True).f_pooled)
entity_clus_entity_fe = fixedeffect(endo, exog, entity_eff= True, cov_type= '
                                       clustered', cluster_entity= True)
print(entity_clus_entity_fe.summary)
entity_clus_time_fe = fixedeffect(endo, exog, entity_eff= True, cov_type= '
                                       clustered', cluster_time= True)
print(entity_clus_time_fe.summary)
tw_clus_entity_fe = fixedeffect(endo, exog, entity_eff= True, time_eff= True,
                                       cov_type= 'clustered', cluster_entity=
                                       True)
print(tw_clus_entity_fe.summary)
entity_clus_tw_fe = fixedeffect(endo, exog, entity_eff= True, time_eff= True,
                                       cov_type= 'clustered', cluster_time=
                                       True)
print(entity_clus_tw_fe.summary)
```

5.4. Code for Random Effects

```
import pandas as pd
import numpy as np
from scipy import stats

import statsmodels.api as sm
from statsmodels.regression.linear_model import RegressionResults
from linearmodels.panel import RandomEffects
from linearmodels.panel.results import PanelResults, RandomEffectsResults,
                                       PanelEffectsResults

from statsmodels.iolib import load_pickle, save_pickle
```

```
from typing import Union

# 1-way entity RE
endo_var = 'LEVERAGE'
exog_vars = ['SIZE',
             'PROFITABILITY',
             'TANG',
             'LIQUID',
             'MCAP',
             'SOLV']

endo = sgx[endo_var]
exog = sm.add_constant(sgx[exog_vars])

entity_re_mod = RandomEffects(endo, exog)

entity_re_fit = entity_re_mod.fit()

print(entity_re_fit.summary)

# 1-way time RE
sgx = sgx.swaplevel('Year', 'Company Code')

endo_var = 'LEVERAGE'
exog_vars = ['SIZE',
             'PROFITABILITY',
             'TANG',
             'LIQUID',
             'MCAP',
             'SOLV']

endo = sgx[endo_var]
exog = sm.add_constant(sgx[exog_vars])

time_re_mod = RandomEffects(endo, exog)

time_re_fit = time_re_mod.fit()

print(time_re_fit.summary)

# FGLS 2-way RE
```

```
def get_weighting_matrix(time_panels: pd.Series, entity_panels: pd.Series):
    t = time_panels.nunique()
    n = entity_panels.nunique()

    J_n_bar = (1 / n) * np.ones(shape = (n, n))
    J_t_bar = (1 / t) * np.ones(shape = (t, t))
    I_n = np.identity(n = n)
    I_t = np.identity(n = t)

    E_n = I_n - J_n_bar
    E_t = I_t - J_t_bar

    Q_1 = np.kron(E_n, E_t)
    Q_2 = np.kron(E_n, J_t_bar)
    Q_3 = np.kron(J_n_bar, E_t)
    Q_4 = np.kron(J_n_bar, J_t_bar)

    return np.array([Q_1, Q_2, Q_3, Q_4])

def get_omega_i(weighting_matrix: np.array, resid: np.array):

    w_1 = (resid.T @ weighting_matrix[0] @ resid) / np.trace(weighting_matrix[0
                                                ])
    w_2 = (resid.T @ weighting_matrix[1] @ resid) / np.trace(weighting_matrix[1
                                                ])
    w_3 = (resid.T @ weighting_matrix[2] @ resid) / np.trace(weighting_matrix[2
                                                ])
    w_4 = w_2 + w_3 - w_1

    return np.array([w_1, w_2, w_3, w_4])

def get_rcorr_matrix(omega_matrix: np.array, weighting_matrix: np.array):
    omega = omega_matrix
    weight = weighting_matrix
    return omega[0] * weight[0] + omega[1] * weight[1] + omega[2] * weight[2] +
                                                 omega[3] * weight[3]

def TwoWayRandomEffects(Y: pd.Series, X: Union[pd.Series, pd.DataFrame],
                                        entity_panel: pd.Series, time_panel: pd
                                        .Series, epsilon: float= 0.0001,
                                        maxiter: int= 99):
    # Step 1: Run OLS of Y on X
```

```
ols = sm.OLS(Y, X)
residuals = ols.fit().resid
# Step 2: Get OLS weighting matrix
weight_matrix = get_weighting_matrix(time_panels= time_panel, entity_panels
                                     = entity_panel)
omega_matrix = get_omega_i(weight_matrix, residuals)
OMEGA = get_rcorr_matrix(omega_matrix, weight_matrix)

# Step 3: Get GLS residuals using weighting matrix
gls = sm.GLS(endog= Y, exog= X, sigma= OMEGA)
gls_residuals = gls.fit().resid
# Step 4: Update GLS weighting matrix
weight_matrix = get_weighting_matrix(time_panels= time_panel, entity_panels
                                     = entity_panel)
omega_matrix = get_omega_i(weight_matrix, gls_residuals)
OMEGA = get_rcorr_matrix(omega_matrix, weight_matrix)
# Step 5: Update GLS coefficient estimates
init_gls = ols ## Initial GLS model
iter_gls = sm.GLS(endog= Y, exog= X, sigma= OMEGA) ## Updated GLS model

i = 1
while np.max(abs(init_gls.fit().params - iter_gls.fit().params)) >= epsilon
                                     : ## If there is a significant
                                     difference in the model estimates,
                                     re-run the refining steps
    init_gls = iter_gls ## Set the initial GLS model to the most updated
                                         model
    # Step 3: Get GLS residuals using weighting matrix
    gls = sm.GLS(endog= Y, exog= X, sigma= OMEGA)
    gls_residuals = gls.fit().resid
    # Step 4: Update GLS weighting matrix
    weight_matrix = get_weighting_matrix(time_panels= time_panel,
                                         entity_panels= entity_panel)
    omega_matrix = get_omega_i(weight_matrix, gls_residuals)
    OMEGA = get_rcorr_matrix(omega_matrix, weight_matrix)
    # Step 5: Update GLS coefficient estimates
    iter_gls = sm.GLS(endog= Y, exog= X, sigma= OMEGA) ## Produce an
                                         updated GLS model
    i += 1
    if i == maxiter:
        print(f"Maximum of {maxiter} iterations reached before model
                                             convergence was achieved.")
```

```
            break

    print(f"{i} iterations of GLS re-specification performed")
    return gls

sgx = sgx.swaplevel('Company Code', 'Year')

endo_var = 'LEVERAGE'
exog_vars = ['SIZE',
             'PROFITABILITY',
             'TANG',
             'LIQUID',
             'MCAP',
             'SOLV']

endo = sgx[endo_var]
exog = sm.add_constant(sgx[exog_vars])

entity_panel = sgx['Company Code']
time_panel = sgx['Year']

tw_re_mod = TwoWayRandomEffects(endo, exog, entity_panel, time_panel)

tw_re_fit = tw_re_mod.fit()

print(tw_re_fit.summary())

# tests for 2-way RE
def cond_LM_stat(restricted_model: PanelResults, how: str):
    T = restricted_model.time_info.total.astype('int')
    n = restricted_model.entity_info.total.astype('int')

    J_n = np.ones(shape = (n, n))
    J_T = np.ones(shape = (T, T))

    I_n = np.identity(n = n)
    I_T = np.identity(n = T)

    u_tilda = restricted_model.resids

    if how.lower() == 'entity':
```

```
        LM_C_entity = ((n * T) / (2 * (T - 1))) * (1 - ((u_tilda.T @ np.kron(
                                                    I_n, J_T) @ u_tilda) / (u_tilda
                                                    .T @ u_tilda))) ** 2
        return LM_C_entity
    elif how.lower() == 'time':
        LM_C_time = ((n * T) / (2 * (n - 1))) * (1 - ((u_tilda.T @ np.kron(J_n,
                                                     I_T) @ u_tilda) / (u_tilda.T @
                                                     u_tilda))) ** 2
        return LM_C_time
    else:
        raise ValueError(f"'how' parameter should be either 'Entity' or 'Time'
                                                    and not {how}.")

def marg_LM_stat(restricted_model: RandomEffectsResults, how: str):
    if how.lower() == 'time':
        T = restricted_model.time_info.total.astype('int')
        n = restricted_model.entity_info.total.astype('int')
    elif how.lower() == 'entity':
        n = restricted_model.time_info.total.astype('int')
        T = restricted_model.entity_info.total.astype('int')
    else:
        raise ValueError(f"'how' parameter should be either 'Entity' or 'Time'
                                                    and not {how}.")

    J_n = np.ones(shape = (n, n))
    J_T = np.ones(shape = (T, T))

    J_n_bar = (1 / n) * J_n
    J_T_bar = (1 / T) * J_T

    I_n = np.identity(n = n)
    I_T = np.identity(n = T)

    E_n = I_n - J_n_bar
    E_T = I_T - J_T_bar

    u_tilda = restricted_model.resids

    if how.lower() == 'entity':
        sigma_v_sq = (1 / T*(n - 1)) * u_tilda.T @ np.kron(E_n, I_T) @ u_tilda
        sigma_2_sq = (1 / T) * u_tilda.T @ np.kron(J_n_bar, I_T) @ u_tilda
```

```
        Q_1 = (1 / sigma_2_sq) * u_tilda.T @ np.kron(J_n_bar, J_T_bar) @
                                                u_tilda
        Q_2 = (1 / (n - 1)*sigma_v_sq) * u_tilda.T @ np.kron(E_n, J_T_bar) @
                                                u_tilda

        LM_M_entity = ((np.sqrt(2) * sigma_2_sq * sigma_v_sq) / np.sqrt(T * (T
                                                - 1) * (sigma_v_sq ** 2 + (n -
                                                1) * sigma_2_sq ** 2))) *\
             ((1/sigma_2_sq) * (Q_1 - 1) + ((n-1) / sigma_v_sq) * (Q_2 - 1))

        return LM_M_entity

    else:
        sigma_v_sq = (1 / n*(T - 1)) * u_tilda.T @ np.kron(I_n, E_T) @ u_tilda
        sigma_1_sq = (1 / n) * u_tilda.T @ np.kron(I_n, J_T_bar) @ u_tilda

        R_1 = (1 / sigma_1_sq) * u_tilda.T @ np.kron(J_T_bar, J_n_bar) @
                                                u_tilda
        R_2 = (1 / (T - 1)*sigma_v_sq) * u_tilda.T @ np.kron(J_n_bar, E_T) @
                                                u_tilda

        LM_M_time = ((np.sqrt(2) * sigma_1_sq * sigma_v_sq) / np.sqrt(n * (n -
                                                1) * (sigma_v_sq ** 2 + (T - 1)
                                                 * sigma_1_sq ** 2))) *\
             ((1/sigma_1_sq) * (R_1 - 1) + ((T-1) / sigma_v_sq) * (R_2 - 1))

        return LM_M_time

def joint_LM_stat(restricted_model: PanelResults):
    return cond_LM_stat(restricted_model, 'Entity') + cond_LM_stat(
                                            restricted_model, 'Time')

pooled_ols_res = load_pickle('model/pooled_ols.pickle')
joint_stat = joint_LM_stat(pooled_ols_res)

joint_p = 1 - stats.chi2.cdf(joint_stat, 1)

print(f"Joint LM Statistic for 2-way Random Effects: {joint_stat}")
print(f"p-value of Joint LM test for 2-way Random Effects: {joint_p}")

cond_entity_stat = cond_LM_stat(pooled_ols_res, how= 'entity')
```

```
cond_entity_p = 1 - stats.chi2.cdf(cond_entity_stat, 1)

print(f"Conditional LM Statistic for Entity Random Effects: {cond_entity_stat}"
                                        )
print(f"p-value of Conditional LM test for Entity Random ffects: {cond_entity_p
                                        }")

cond_time_stat = cond_LM_stat(pooled_ols_res, how= 'time')

cond_time_p = 1 - stats.chi2.cdf(cond_time_stat, 1)

print(f"Conditional LM Statistic for Time Random Effects: {cond_time_stat}")
print(f"p-value of Conditional LM test for Time Random Effects: {cond_time_p}")

marg_entity_stat = marg_LM_stat(time_re_fit, how= 'entity')

marg_entity_p = 1 - stats.chi2.cdf(marg_entity_stat, 1)

print(f"Marginal LM Statistic for Entity Random Effects: {marg_entity_stat}")
print(f"p-value of Marginal LM test for Entity Random ffects: {marg_entity_p}")
marg_time_stat = marg_LM_stat(entity_re_fit, how= 'time')

marg_time_p = stats.chi2.cdf(marg_time_stat, 1)

print(f"Marginal LM Statistic for Time Random Effects: {marg_time_stat}")
print(f"p-value of Marginal LM test for Time Random ffects: {marg_time_p}")

def lr_test(restricted_model: PanelResults|RandomEffectsResults|
                                        PanelEffectsResults, unrestricted_model
                                        : PanelResults|RandomEffectsResults|
                                        PanelEffectsResults, df: int= 1):
    try:
        res_loglik = restricted_model.loglik
    except:
        res_loglik = restricted_model.llf

    try:
        unres_loglik = unrestricted_model.loglik
    except:
        unres_loglik = unrestricted_model.llf
```

```
    lr_stat = -2 * (res_loglik - unres_loglik)

    lr_p = 1 - stats.chi2.cdf(lr_stat, df)

    print(f"Log-Likelihood Test Statistic: {lr_stat}")
    print(f"Log-Likelihood Test Statistic: {lr_p}")
lr_test(pooled_ols_res, entity_re_fit, 1)
lr_test(pooled_ols_res, time_re_fit, 1)
lr_test(pooled_ols_res, tw_re_fit, 2)
lr_test(entity_re_fit, tw_re_fit, 1)
lr_test(time_re_fit, tw_re_fit, 1)

# CRE
endo_var = 'LEVERAGE'
exog_vars = ['SIZE',
             'PROFITABILITY',
             'TANG',
             'LIQUID',
             'MCAP',
             'SOLV']

sgx = sgx.reset_index(drop= True).set_index(['Company Code', 'Year'], drop=
                                            False)

endo = sgx[endo_var]

mean_exog_vars = ['avg' + var for var in exog_vars]
sgx[mean_exog_vars] = sgx[exog_vars].groupby(level= 'Company Code').transform('
                                            mean')
exog = sm.add_constant(sgx[exog_vars + mean_exog_vars])
sgx.drop(columns= mean_exog_vars)

entity_cre_mod = RandomEffects(endo, exog)

entity_cre_fit = entity_cre_mod.fit()

print(entity_cre_fit.summary)

endo_var = 'LEVERAGE'
exog_vars = ['SIZE',
             'PROFITABILITY',
             'TANG',
```

```
            'LIQUID',
            'MCAP',
            'SOLV']

sgx = sgx.reset_index(drop= True).set_index(['Year', 'Company Code'], drop=
                                            False)

endo = sgx[endo_var]

mean_exog_vars = ['avg' + var for var in exog_vars]
sgx[mean_exog_vars] = sgx[exog_vars].groupby(level= 'Year').transform('mean')
exog = sm.add_constant(sgx[exog_vars + mean_exog_vars])
sgx.drop(columns= mean_exog_vars)

time_cre_mod = RandomEffects(endo, exog, check_rank= False)

time_cre_fit = time_cre_mod.fit()

print(time_cre_fit.summary)

endo_var = 'LEVERAGE'
exog_vars = ['SIZE',
             'PROFITABILITY',
             'TANG',
             'LIQUID',
             'MCAP',
             'SOLV']

sgx = sgx.reset_index(drop= True).set_index(['Company Code', 'Year'], drop=
                                            False)

endo = sgx[endo_var]

mean_exog_vars = ['avg' + var for var in exog_vars]
sgx[mean_exog_vars] = sgx[exog_vars].groupby(level= 'Company Code').transform('
                                            mean')
exog = sm.add_constant(sgx[exog_vars + mean_exog_vars])
sgx.drop(columns= mean_exog_vars)

entity_panel, time_panel = sgx['Company Code'], sgx['Year']

tw_cre_mod = TwoWayRandomEffects(endo, exog, entity_panel, time_panel)
```

```
tw_cre_fit = tw_cre_mod.fit()

print(tw_cre_fit.summary())

def hausman_test(fixed_effects: PanelEffectsResults, random_effects:
                                        RandomEffectsResults|RegressionResults)
                                        :
    # (I) find overlapping coefficients:
    common_coef = list(set(fixed_effects.params.index).intersection(
                                            random_effects.params.index))

    # (II) calculate differences between FE and RE:
    b_diff = np.array(fixed_effects.params[common_coef] - random_effects.params
                                            [common_coef])
    df = len(b_diff)
    b_diff.reshape((df, 1))

    b_fe_cov = fixed_effects.cov
    try:
        b_re_cov = random_effects.cov
    except:
        b_re_cov = random_effects.cov_params()

    b_cov_diff = np.array(b_fe_cov.loc[common_coef, common_coef] -
                          b_re_cov.loc[common_coef, common_coef])
    b_cov_diff.reshape((df, df))

    # (III) calculate test statistic:
    hausman_stat = abs(np.transpose(b_diff) @ np.linalg.inv(b_cov_diff) @
                                            b_diff)
    hausman_p = 1 - stats.chi2.cdf(hausman_stat, df)

    print(f"Hausman Test Statistic: {hausman_stat}")
    print(f"Hausman Test Statistic: {hausman_p}")

hausman_test(tw_fe_fit, tw_re_fit)
hausman_test(entity_fe_fit, entity_re_fit)
hausman_test(time_fe_fit, time_re_fit)
endo_var = 'LEVERAGE'
exog_vars = ['SIZE',
             'PROFITABILITY',
```

```
              'TANG',
              'LIQUID',
              'MCAP',
              'SOLV']

mean_exog_vars = ['avg' + var for var in exog_vars]

hypothesis = " = ".join(mean_exog_vars) + " = 0"
wald_test = entity_cre_fit.wald_test(formula= hypothesis)

print(wald_test)
endo_var = 'LEVERAGE'
exog_vars = ['SIZE',
              'PROFITABILITY',
              'TANG',
              'LIQUID',
              'MCAP',
              'SOLV']

mean_exog_vars = ['avg' + var for var in exog_vars]

hypothesis = " = ".join(mean_exog_vars) + " = 0"
wald_test = time_cre_fit.wald_test(formula= hypothesis)

print(wald_test)
endo_var = 'LEVERAGE'
exog_vars = ['SIZE',
              'PROFITABILITY',
              'TANG',
              'LIQUID',
              'MCAP',
              'SOLV']

mean_exog_vars = ['avg' + var for var in exog_vars]

hypothesis_matrix = " = ".join(mean_exog_vars) + ' = 0'
wald_test = tw_cre_fit.wald_test(hypothesis_matrix, use_f= False, )

print(f"Wald-Test Statistic for 2-Way Correlated Random Effects: {wald_test.
                                        df_denom}")
print(f"Wald-Test Statistic for 2-Way Correlated Random Effects: {wald_test.
                                        statistic[0][0]}")
```

```
print(f"Wald-Test Statistic for 2-Way Correlated Random Effects: {wald_test.
                                        pvalue.item()}")
marg_cre_entity_stat = marg_LM_stat(restricted_model= entity_cre_fit, how= '
                                        Entity')
marg_cre_entity_p = 1 - stats.chi2.cdf(marg_cre_entity_stat, 1)

print(f"Marginal LM Statistic for Entity Correlated Random Effects: {
                                        marg_cre_entity_stat}")
print(f"p-value of Marginal LM test for Entity Correlated Random ffects: {
                                        marg_cre_entity_p}")
ind_clust_entity_cre = entity_cre_mod.fit(cov_type= 'clustered', cluster_entity
                                        = True)
print(ind_clust_entity_cre.summary)
```