

ASSET PRICING - EMPIRICAL APPLICATION 1

FACTORIAL MODEL AND RISK PREMIUM DECOMPOSITION - APT

Luis Miguel Fonseca
Stéphane Eloundou Mvondo
Natalia Cárdenas Frías

November 29, 2023

Introduction

Focusing on recent data from the French equity market, we want to better comprehend how the market prices systemic, non-diversifiable risk embedded in the risk premium of stocks, i.e. any expected compensation beyond the risk-free return. We base our analysis on a linear decomposition of said premium on different *factors* of risk in the spirit of the Arbitrage Pricing Theory (APT) pioneered by Ross [1976]. Unlike the CAPM model which considers a unique risk premium in the market, the Ross model gives a more detailed description of the pricing of aggregate risk by decomposing the contributions of different sources of risk. Here, a risky portfolio of j stocks¹ is compensated with k risk premia associated with the k *common factors* that the portfolio is exposed to.

1 Data

1.1 French stock market data

We decided to build a case study of the French market because it is a liquid and matured market, central in Europe. In the case of this analysis, we had trouble getting the data needed to perform it for other countries² and the fact that France has more publicly available data helped us choose it as our market of study.

We built a portfolio with 30 French stocks that we got from Yahoo Finance. For simplicity, the synthetic portfolio is composed of one stock of each company and its composition does not change during the period studied. Table 1 shows the companies that we used to create this portfolio, they are all publicly traded companies in France since the early 2000's in Euronext Paris. Importantly, we tried to have a certain diversity in the sectors

¹Let $j \in \{1, \dots, J\}$ with J sufficiently large so that all idiosyncratic risk can be fully diversified. We better explain the difference between idiosyncratic and aggregate in the context of the Ross model in Section 1.1

²Initially we thought about using data from the German market but we couldn't for instance find data for their inflation-linked bond yield that we use as an endogenous factor in Section 2.1.2

represented to be able to capture some diversification to risk even if the portfolio is too small and we do not reweight it. However, because we try to implement a version of the [Fama and French \[1992\]](#) factor analysis, we restrain ourselves from choosing financial companies as the authors do due to their high leverage. Other than these two conditions, the choice of the companies was mainly restricted to data availability on public 'long' series on firm-level data, notably on market capitalization and book-to-market ratio to be able to incorporate the [Fama and French \[1992\]](#) factors to our analysis.

Because we are interested in the underlying determinants of the risk premia, and due to data availability issues, we decided to have a broad analysis with monthly data for our selection of stocks for the period 2005-2022. Monthly data is for instance used by [Chen et al. \[1986\]](#). While this is not a very long period, it encompasses important moments in the financial markets in particular the Great Financial Crisis, the subsequent European Debt Crisis, and the Covid years. Also importantly, during most of this period (following the GFC), monetary policy fixed interest rates were extremely low driving down the return of sovereign debt for countries like France and Germany³ that could have been assimilated to the risk-free rate. This means that for an investor to get any returns, it had to hold risky assets. Moreover, the period following the GFC and up to 2021 was also characterized by extremely low inflation in the Euro Zone. This is interesting because the APT usually incorporates inflation risk as market risk, yet inflation was nowhere to be found for more than a decade. Seeing how the market incorporated this monetary reality is by itself an intriguing question.

1.2 Data description and sources

The other series that we use are the following and its sources, how they are used in the context of the analysis is described in subsequent sections.

- As a proxy for the free rate of the market we consider two measures:
 - The yield of short-term OAT, i.e. French treasuries taken from [Banque de France's website](#). As for most developed, stable countries, short-term sovereign bonds are taken as the risk-free asset as Governments are supposed to be more solvent than other agents in the economy, after all, they decide their income and could seize resources via taxes to meet their obligations.
 - The spot yield curve spot rate, for 3-month maturity of all government bonds rated triple A in the Euro Area, retrieved from the [ECB webpage](#). On top of the fact that this is a measure for short-term sovereign bonds, we consider this to be a relevant proxy for the French market due to the strong integration within the European capital market. If an investor decides that the French market becomes risky, she can easily move her investments to another European capital market that looks safer.
- To get the market rate, we use the return of the main index of the country, the CAC40 also taken from Yahoo Finance as for the components of our synthetic portfolio. In hindsight, we are not sure of the pertinence of comparing our portfolio to this index. While the composition of our portfolio is not the

³They were negative for certain maturities in real terms for a part of the time frame analyzed, pretty much since the European Debt Crisis until the inflation surge after Covid.

Company Name	Ticker	Industry
Accor	AC.PA	Hospitality
Air Liquide	AI.PA	Industrial Gases
Air France-KLM	AF.PA	Airlines
Airbus	AIR.PA	Aerospace
Biomerieux	BIM.PA	Biotechnology
BIC	BB.PA	Consumer Goods
Bouygues	EN.PA	Construction
Capgemini	CAP.PA	Information Technology
Carrefour	CA.PA	Retail
Casino	CO.PA	Retail
Dassault Aviation	AM.PA	Aerospace
Danone	BN.PA	Food and Beverage
Hermes International	RMS.PA	Fashion and Luxury
JCDecaux	DEC.PA	Advertising
Kering	KER.PA	Fashion and Luxury
L'Oreal	OR.PA	Cosmetics
LVMH	MC.PA	Fashion and Luxury
Michelin	ML.PA	Automotive
Nexans	NEX.PA	Electrical Equipment
Orange	ORA.PA	Telecommunications
Renault	RNO.PA	Automotive
Saint-Gobain	SGO.PA	Manufacturing
Sanofi	SAN.PA	Pharmaceuticals
Sodexo	SW.PA	Food Services
TF1	TFL.PA	Broadcasting
Thales	HO.PA	Aerospace and Defense
TotalEnergies	TTE.PA	Energy
Ubisoft	UBI.PA	Video Games
Vinci	DG.PA	Construction
Vivendi	VIV.PA	Entertainment

Table 1: Synthetic portfolio: Companies, Tickers, and Industries

same as the CAC40⁴, due to the data availability issues we've been mentioning, we see that our choices are heavily biased towards 'big name' companies that are those belonging to the index.

- We got the series of the exchange rate between the Euro and the US dollar from Yahoo Finance. It is read as the amount of USD needed to get one euro.
- The GDP series is taken from the [ECB webpage](#). It is available at a quarterly frequency and is available at market prices.
- The harmonized headline inflation rate is taken from the [INSEE webpage](#).
- For the market inflation expectation in a 10-year horizon, we use the break-even inflation rate published by [Agence France Trésor](#) online. Sadly, data is only available from 2013.
- For the implementation of the two additional [Fama and French \[1992\]](#) factors, we took different routes
 - We found the estimation of the factors published by K. French in his [online Data Library](#) that are

⁴Not all the stocks we chose are necessarily part of the CAC40 at every period studied, and the CAC40 is a weighted index that evolves over time.

constantly updated. Their estimations start in the 1990s and are made for different markets using the comprehensive CRSP dataset that is not freely available. He has an estimation for the European market that we downloaded to use but it is not clear which stocks are used to replicate their portfolio.

- To try to build these estimations ourselves for our portfolio meaning that a minima we need data on the market capitalization of each company during the time frame studied and its *book*. This information is hardly available without having access to platforms like Bloomberg or CRSP. The best information that we could find comes from [this](#) website that publishes the market capitalization and the price-to-book (the inverse of the book-to-market ratio) annual series for several stocks. The data is however cannot be directly downloaded from the site so we scrap it to get the series (see Code Appendix B). Our biggest fear with this source is that it is not clear at all where the information comes from even if they mention several quality [data providers](#) as their partners. Since it is the only source that resembles what is needed for this part we used it but we are not confident about it.

2 Empirical strategy and implementation

We implement a minimal approach to [Ross \[1976\]](#), namely using fewer factors than in [Chen et al. \[1986\]](#). We include both exogenous and endogenous macroeconomic risk factors as well as an approximation to implement [Fama and French \[1993\]](#) three-factor model which used stock-specific data.

Let the return R_j of the j -th component of her portfolio can be described by the following expression $\forall j \in \{1, \dots, N\}$:

$$R_j = \mathbb{E}[R_j] + \underbrace{\sum_{k=1}^K \beta_{j,k} f_k}_{\text{Systemic risk}} + \overbrace{u_j}^{\text{Idiosyncratic risk}} \quad (1)$$

Where $\mathbb{E}[R_j]$ is the expected return of asset j and R_j its return without dividend i.e. the first difference of the stock price. This is the so-called *factorial model* that includes two sources of risk:

- The investor faces centered idiosyncratic risks u_j , $\mathbb{E}[u_j] = 0$ that are assumed to be completely diversifiable with a portfolio "large enough" (N big) because they are independent of each other $u_j \perp u_{j'} \forall j \neq j'$.
- She also faces k different sources of aggregate risk, modeled by the linear combination of f_k centered *shocks* that influence all R_j with a sensitivity $\beta_{j,k}$. By definition, these risks cannot be diversified because they affect the returns of all asset and thus has to be compensated which is the focal point of our study.

In order to implement a regression analysis and estimation as the one that follows in this section, we shall also assume that the idiosyncratic risks are uncorrelated with aggregate risk $\text{corr}(u_j, f_k) = 0, \forall j, k$.

This section is organized as follows. Section 2.1 identifies the aggregate market risks $(f_k)_k$ that we are going to consider as the ones priced by the market. Section 2.2 runs a first regression analysis to identify the sensitivity of each return to each factor of risk, i.e. to identify the $(\beta_{j,k})_{j,k}$ from the factorial model (Eq. 1). It also implements an approximation to a parallel model that decomposes the return of stocks: the [Fama and French \[1992\]](#) that

also will lead us to estimate the sensitivity of the return of each return to a series of factors. Finally, section 2.3 uses the series of $(\hat{\beta}_{j,k})_{j,k}$ estimated before to implement an estimation of the *multibeta relationship* which is the regression that will allow us to get how much the market is remunerating the exposition to a market factor risk.

2.1 Identify the risk factors

We decided to explore the role of the following sources of risk for a first model inspired by Ross [1976] and Chen et al. [1986].

- The activity risk, measured by changes in GDP
- Inflation risk, measured both by the HICP of France for the short term and by the market inflation expectation in a 10-y horizon.
- Devaluation risk measured by the exchange rate between the Euro and the US dollar.

We then consider the factors of a complementary factor model, Fama and French [1992] which are not directly linked to risks.

2.1.1 Exogeneous factors

These are risk factors that do not depend directly on the financial markets or more precisely that are not deduced from a linear combination of the returns of financial assets. In our case, it is mainly the activity risk, the short-term interest rate (measured by the HICP), and the devaluation risk. Importantly, the measure of these risks is not directly measured by changes or by the level of the underlying variables as markets are informational efficient [Fama, 1970], and they have already priced in all relevant information conveyed by prices. In particular, all *predictable* movements, say in inflation, have already been incorporated by the market. This means that the factors of risk are actually the *surprises* in the movements of these variables.

To extract this unpredictable part of these variables we need to use the residuals of some time series model of the (stationary) variables⁵, in our case ARIMA(p, d, q) models. We used the function *auto.arima* from the R *forecasts* package that set de parameters optimally by minimizing the BIC between different specifications. We ended up with:

- ARIMA(0,1,0) (i.e. a random walk⁶) for the series of log GDP.
- ARIMA(0,1,1) for the exchange rate series.
- ARIMA(2,1,2) for the HICP series.

We then stored the residuals of each one of these models and merge it in the data frame with the stock return. We shall note that by construction the factor (i.e. the residuals) are by construction centered around zero. They are actually the innovations of the initial series for the macroeconomic variables.

⁵Evidently, we need that the series used are stationary before conducting any TS analysis. We tested for stationarity of our series using the ADP test and then differentiate the series that were non-stationary (see Appendix A). We didn't apply the ADF test to the series in logs afterward as the R function *auto.arima* automatically differentiates the series d times until they are stationary.

⁶We do find this result 'weird' but didn't find any mistake in the code and it does square with the factors being centered.

2.1.2 Endogeneous factors

These factors are linear combinations of the returns of financial assets. We used the *Breakeven Inflation* at a 10-year horizon which is simply the yield difference between 10-y OATi 0.10% (*Obligations Assimilables du Trésor*) indexed by the HICP and non-indexed 10-y OAT. This gap is assimilated to how markets price the possibility of having future inflation⁷. Once again we need to extract the unforecastable movements in this indicator to have an adequate measure of risk. We follow the same methodology as with the exogenous factors and retrieve the residuals of an ARIMA(2,1,2) model.

2.1.3 French-Fama factors

Fama and French [1993] can also be seen as an extension of the CAPM model but their factors, while significant specially in the US market, are hardly interpretable as risk factors. The authors show that the variation of the returns of an asset can be explained not only by the exposure to market risk as in the CAPM represented by the difference of the market return and the risk-free rate $[R_M - R_f]$, but also by a size and value premium in the following model.

$$R_j = \alpha_j + R_f + \beta_{m,j}[R_M - R_f] + \beta_S SMB + \beta_V HML + \varepsilon_j \quad (2)$$

The size premium refers to the observation that stocks with small market capitalizations tend to outperform stocks with larger ones and it is captured by the factor SMB, *small minus big*. It is computed as the difference in average returns of the 30% stocks with the smallest market capitalization and the average returns of the 30% stocks associated with the firms with the largest market capitalization. The value premium refers to the outperformance of "value stocks" i.e. those that have high book-to-market (B/M) and it is represented by the difference in an average return of the 50% of stocks with the highest B/M ratio (value stocks) and the 50% with lowest B/M ratio (growth stocks). We use this methodology to create our own SMB and HML factors at a yearly frequency.

As we previously mentioned, K. French made public his estimation of the factors for the European model using a more complex approach with "6 value-weight portfolios formed on size and book-to-market". SMB (Small Minus Big) is the average return on the three small portfolios minus the average return on the three big portfolios, and HML (High Minus Low) is the average return on the two value portfolios minus the average return on the two growth portfolios.

2.2 Estimate the sensitivities of the factorial model

2.2.1 Ross model

Given that we assumed that the idiosyncratic risks u_j are uncorrelated with the aggregate factor risks f_k we can estimate the factorial model in Eq. 1 with linear regression models⁸. We will also pose that $f_k \perp f_{k'} \forall k, k'$ in

⁷Note that the Breakeven inflation is the inflation rate that would equalize the return of these two sovereign bonds (due to AOA).

⁸The residuals will be uncorrelated with the explanatory variables

order to have a good identification of the parameters with linear regression estimators $\forall k_0, \beta_{j,k_0} = \frac{cov(R_j, f_{k_0})}{Var(f_{k_0})} \equiv \hat{\beta}_j$.

For this first estimation, we need to include a time dimension on the baseline factorial model $\forall j, \forall t$:

$$R_{j,t+1} = \mathbb{E}_t[R_{j,t+1}] + \sum_{k=1}^K \beta_{j,k} f_{k,t+1} + u_{j,t+1} \quad (3)$$

We will therefore estimate the following:

$$R_{j,t} \sim \alpha + \sum_{k=1}^K \beta_{j,k} \hat{f}_{k,t}$$

Note that for the macroeconomic risks $\hat{f}_{k,t}$ denotes the estimated errors we found in the previous section for each series with standard TS analysis. To do so, we split the dataset by stock and performed linear regressions for each subgroup.

Exogeneous factors only We performed a first version of this estimation using only the exogenous factors which allows us to have an analysis of the 2005-2022 period. Results are shown in the following table and graph. I am really sorry for the size of the table, I didn't manage to split the latex table.

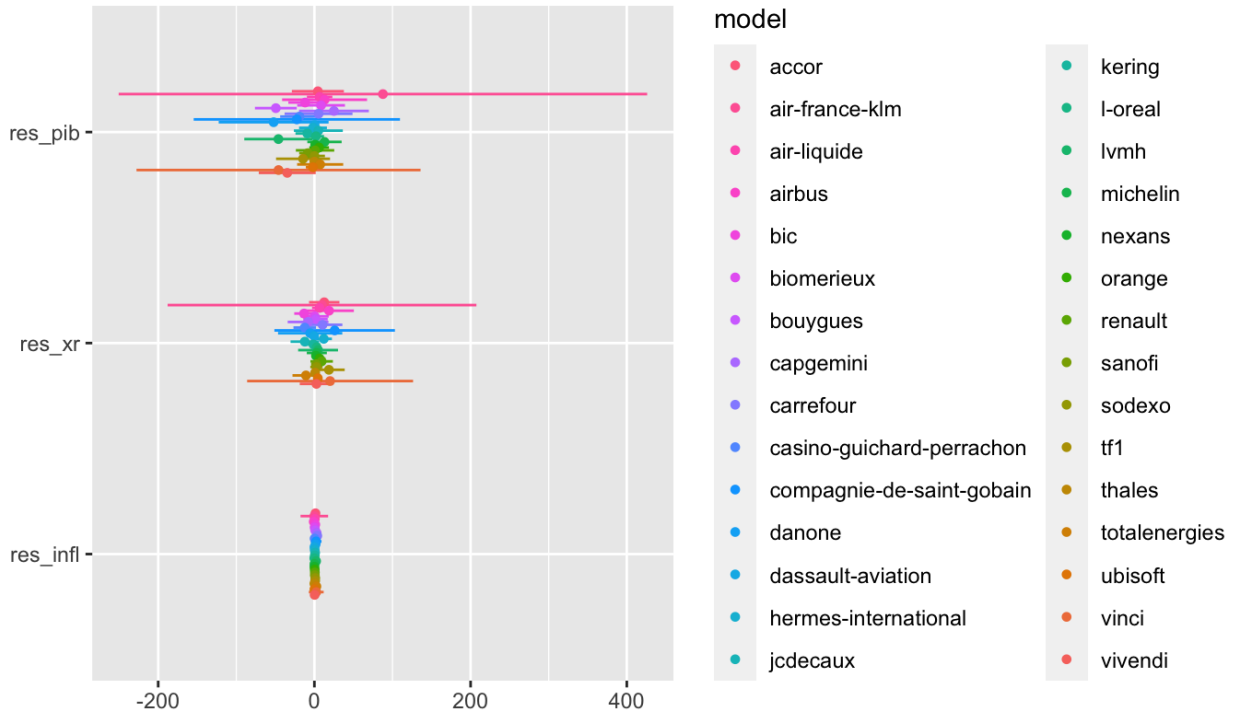


Figure 1: Betas: exogenous factors

Estimate the beta coefficients for each exogenous factor

	Dependent variable:																															
	Return																															
	accor	air-france-kin	air-liquide	airbus	bic	biomecurex	boisguyes	cagennini	carrefour	cuisine-guichard	permacon	compagnie-de-saint-gobain	danone	daussat-aviation	hermes-international	jadeaux	kering	l'oreal	lvh	michelein	nextans	orange	renault	sanofi	sodeao	tf1	thales	totalenergies	ubisoft	vinci	vivendi	
	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)	(13)	(14)	(15)	(16)	(17)	(18)	(19)	(20)	(21)	(22)	(23)	(24)	(25)	(26)	(27)	(28)	(29)	(30)	(31)	
res_pib	4.417	87.731	6.695	13.047	-11.895	8.568	-49.367***	25.025	5.304	-19.160			-22.302	0.258	-1.806	5.052	-9.056	2.063	-46.082**	12.901	0.998	6.052	0.842	-8.384	0.075	-14.533	-0.049	7.382	-1.963	-45.927	-34.750*	
	(16.882)	(771.635)	(8.268)	(27.583)	(10.889)	(15.496)	(13.562)	(22.659)	(22.047)	(12.504)			(66.982)	(1.664)	(9.034)	(15.882)	(7.705)	(5.203)	(22.063)	(11.139)	(2.013)	(6.387)	(12.460)	(5.642)	(6.801)	(17.521)	(0.030)	(14.959)	(4.566)	(92.241)	(18.497)	
res_xr	12.577	9.750	6.565	18.735	-13.262**	0.480	-8.086	10.413	-12.388*				-5.407	-0.441	12.003**	-12.236	-1.286	2.870	4.720	3.690	2.008*	6.672*	9.294	2.280	2.584	18.600*	0.999***	-10.808	4.367	20.101	2.579	
	(9.864)	(100.281)	(4.830)	(16.116)	(6.362)	(7.924)	(13.239)	(12.881)	(7.366)				(20.852)	(0.972)	(5.278)	(9.279)	(4.502)	(3.040)	(12.949)	(6.508)	(1.176)	(3.731)	(7.283)	(3.297)	(3.973)	(10.237)	(0.018)	(8.740)	(2.668)	(53.894)	(0.807)	
res_inf	1.383	-0.111	0.556	-0.840	1.128**	0.239	0.925	2.895**	3.652***	0.252			2.233	1.565	0.092	0.943	0.496	-0.089	2.148*	0.095	0.064	0.404	0.607	0.527	1.085***	0.959	0.001	2.524***	0.042	2.372	0.313	
	(0.880)	(8.947)	(0.431)	(1.438)	(0.568)	(0.808)	(0.707)	(1.181)	(1.149)	(0.652)			(1.860)	(0.087)	(0.471)	(0.828)	(0.402)	(0.271)	(1.155)	(0.581)	(0.105)	(0.333)	(0.650)	(0.294)	(0.355)	(0.913)	(0.002)	(0.780)	(0.238)	(4.808)	(0.964)	
Constant	-0.031	6.689**	0.059	-0.097	0.251	0.081	0.563***	0.366	0.556	0.196			3.082***	1.418**	0.006	0.328	0.138	-0.029	0.692**	-0.107	-0.011	-0.0002	0.366*	0.077	0.178*	0.310	-0.0002	0.598*	0.093	2.137	0.436	
	(0.263)	(2.674)	(0.129)	(0.430)	(0.170)	(0.241)	(0.211)	(0.353)	(0.343)	(0.195)			(1.044)	(0.026)	(0.141)	(0.247)	(0.120)	(0.081)	(0.345)	(0.174)	(0.031)	(0.099)	(0.194)	(0.088)	(0.106)	(0.273)	(0.0005)	(0.233)	(0.071)	(1.437)	(0.288)	
Observations	215	215	215	215	215	215	215	215	215	215			215	215	215	215	215	215	215	215	215	215	215	215	215	215	215	215	215	215	215	
R ²	0.021	0.001	0.020	0.009	0.042	0.002	0.068	0.033	0.090	0.025			0.005	0.014	0.006	0.028	0.014	0.015	0.005	0.038	0.008	0.017	0.027	0.013	0.029	0.046	0.025	0.938	0.052	0.014	0.003	0.017
Adjusted R ²	0.007	-0.013	0.004	-0.005	0.029	-0.012	0.055	0.019	0.037	0.011			-0.009	-0.008	0.014	-0.004	0.001	-0.009	0.024	-0.007	0.003	0.013	-0.002	0.015	0.032	0.011	0.937	0.039	-0.004	-0.011	0.003	
Robust Std. Error	30.980	1.882	6.280	2.479	3.538	3.088	5.159	5.020	2.447				8.126	0.379	2.057	3.616	1.754	1.185	5.046	2.536	0.458	1.454	2.438	1.285	1.548	3.989	0.007	3.406	1.040	21.003	4.212	
F-Statistic (df = 3, 211)	1.474	0.091	1.454	0.629	3.100**	0.129	5.162***	2.390*	3.715**	1.825			0.344	1.009	0.427	0.969	0.036	0.383	2.767**	0.538	1.230	1.946	0.891	2.096	3.370**	1.793	1.067772***	3.879***	0.974	0.224	1.290	
																									</							

Note:

*p<0.1; **p<0.05; ***p<0.01

Figure 2: Betas: exogenous factors

Exogenous and endogenous factors We add another specification to include the endogenous market expectation of long-term inflation. Sadly due to the short series on the Breakeven inflation, this analysis is conducted only on the period 2013-2022, still at a monthly frequency.

Exogenous and French and Fama factors The last specification includes the French and Fama factors given by K. French data library. As in the baseline it refers to monthly data on the 2005-2022 period.

2.2.2 French and Fama model

We also wanted to try to estimate the three French and Fama with the data we collected. We used the approach mentioned in Section 2.1.3 and compared the mean return of value stocks and non-value stocks as well as of growth vs non-growth stocks. Once again, due to data availability issues, we had to restrain our sample to study only the 2010-2022 period at a yearly frequency.

The results are not very significant but we think it is likely due to our data quality. We show the results for this but we will ignore this representation of the French and Fama model moving forward.

2.3 Estimate the remuneration of risk from the multibeta relationship

Under AOA, assuming that the factorial model (Eq. 1) is an accurate depiction of how equity is price, implies that the expected returns are constrained by a multibeta relationship of the following form $\exists \rho, \exists \lambda_1, \dots, \lambda_k$:

$$\mathbb{E}[R_j] = \rho + \sum_{k=1}^K \lambda_k \beta_{j,k} \quad (4)$$

where each λ_k is the parameter representing the market price of risk (the risk premium) that the market re-tributes for being exposed to a given risk factor f_k with a sensitivity $\beta_{j,k}$

To estimate the lambda coefficients we need to have a new dependent variable: the historical mean return of every stock in order to get a cross-sectional analysis. To do this last part of the analysis we store the sensitivities $\hat{\beta}_{j,k}$ and the intercepts $\hat{\rho}_j$ derived in the previous section and regress them over this historical mean $\mathbb{E}[\hat{R}_j] - \hat{\rho} = \overline{R_{j,t}} - \hat{\rho}_j$.

We decided to only keep the model with the exogenous factors and the French and Fama factors in monthly frequency and run:

$$\overline{R_{j,t}} - \hat{\rho}_j \sim \hat{\beta}_{pib,j} + \hat{\beta}_{xr,j} + \hat{\beta}_{infl,j} + \hat{\beta}_{HLM,j} + \hat{\beta}_{SMB,j} + \hat{\beta}_{Mkt.RF,j} \quad (5)$$

Appendix A Additional tables and figures

Macroeconomic factors

Table 2: Results ADF with trend and drift

	EuroUSD	CAC40	Inflation	PIB	RF AAA	OAT	1pct	5pct	10pct
tau3	-2.945	-2.028	-0.565	-4.716	-1.011	-0.720	-3.990	-3.430	-3.130
phi2	3.012	1.652	1.243	7.665	0.800	0.555	6.220	4.750	4.070
phi3	4.421	2.220	1.522	11.121	1.198	0.683	8.430	6.490	5.470

Table 3: Results ADF with drift

	EuroUSD	CAC40	Inflation	PIB	RF AAA	OAT	1pct	5pct	10pct
tau2	-1.841	-1.428	-0.444	-2.117	-1.545	-1.142	-3.460	-2.880	-2.570
phi1	1.791	1.277	0.438	2.589	1.195	0.803	6.520	4.630	3.810

Table 4: Results ADF with no trend nor drift

	EuroUSD	CAC40	Inflation	PIB	RF AAA	OAT	1pct	5pct	10pct
tau1	-0.618	0.414	0.353	0.721	-1.437	-0.995	-2.580	-1.950	-1.620

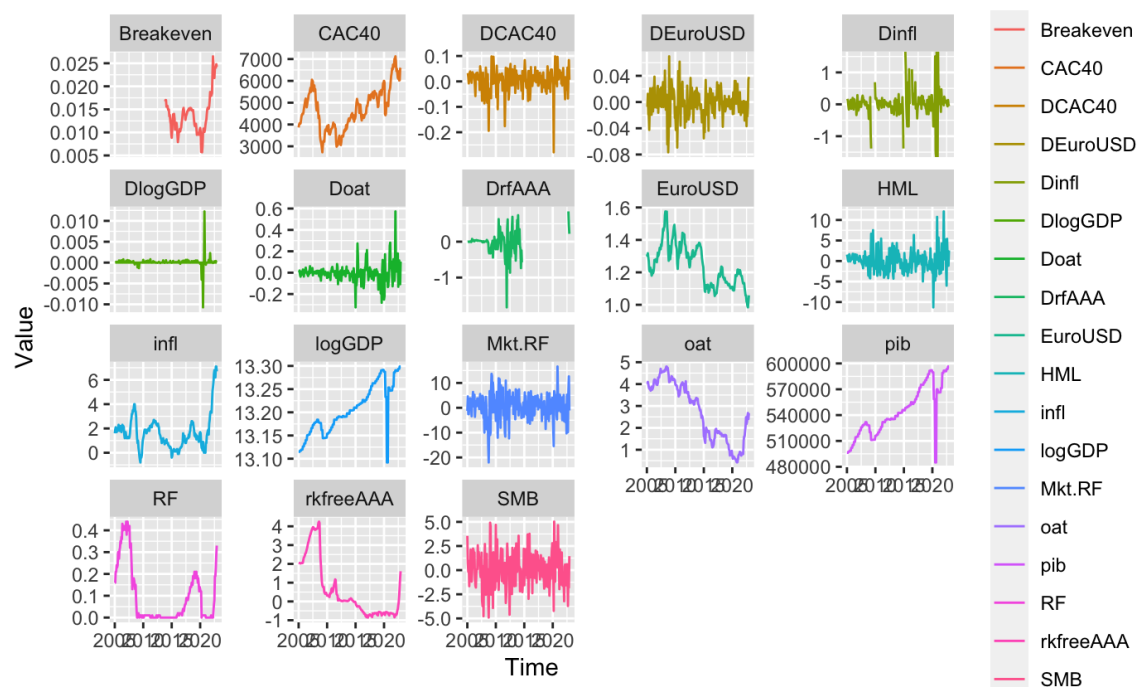


Figure 3: Factors: Series in level and in deltas

Appendix B Code - Data gathering and cleaning in Python

```

1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3 """
4 APi - Data gathering and Data Cleaning
5
6 Scrapping - Firm level data for French and Fama factors
7 Use Yahoo Finance API to get the financial data for all the stocks
8 Use Eurostat API to get the macro data
9 Two data sets (French-Fama factors and long term inflation expectation) are found online and
   have been downloaded in CSV file beforehang
10
11 Merge and clean the dataset
12
13 @author: nataliacardenasf
14 """
15
16 import pandas as pd
17 import numpy as np
18 import os
19
20 import requests
21 from bs4 import BeautifulSoup
22
23 #import pandas_datareader.data as web

```

```

24 import yfinance as yf
25 #from eurostatapiclient import EurostatAPIClient
26 import datetime
27
28
29 os.chdir('/Users/nataliacardenasf/Documents/GitHub/PROJECTS_AP_FE/AP 1')
30
31 company_names_lower = [
32     'air-liquide', 'airbus', 'bouygues', 'capgemini', 'carrefour', 'casino-guichard-perrachon',
33     , 'vivendi',
34     'kering', 'l-oreal', 'lvmh', 'michelin', 'orange', 'renault', 'sanofi', 'thales',
35     'totalenergies', 'vinci', 'compagnie-de-saint-gobain', 'ubisoft', 'tf1', 'danone',
36     'dassault-aviation', 'air-france-klm', 'accor', 'bic', 'hermes-international',
37     'jcdecaux', 'nexans', 'sodexo', 'biomerieux', "CAC40", "EuroUSD"]
38
39 tickers = [
40     'AI.PA', 'AIR.PA', 'EN.PA', 'CAP.PA', 'CA.PA', 'CO.PA', 'VIV.PA', 'KER.PA', 'OR.PA', 'MC.
41     PA',
42     'ML.PA', 'ORA.PA', 'RNO.PA', 'SAN.PA', 'HO.PA', 'TTE.PA', 'DG.PA', 'SGO.PA', 'UBI.PA', '
43     TFI.PA',
44     'BN.PA', 'AM.PA', 'AF.PA', 'AC.PA', 'BB.PA', 'RMS.PA', 'DEC.PA', 'NEX.PA', 'SW.PA', 'BIM.
45     PA', "^FCHI", 'EURUSD=X']
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999

```

```

66         data.append({'Year': year, 'MarketCap': market_cap, 'Company':
        company_name})
67         return pd.DataFrame(data)
68     return None
69
70 def scrape_price_book(url, company_name):
71     response = requests.get(url)
72     if response.status_code == 200:
73         soup = BeautifulSoup(response.content, 'html.parser')
74         table_body = soup.find('table', class_='table').find('tbody')
75         if table_body:
76             data = []
77             rows = table_body.find_all('tr')
78             for row in rows:
79                 cols = row.find_all('td')
80                 if len(cols) >= 2:
81                     year = cols[0].text.strip()
82                     pricebook = cols[1].text.strip()
83                     #variation = cols[2].text.strip()
84                     data.append({'Year': year, 'PriceBook': pricebook, 'Company': company_name
        })
85             return pd.DataFrame(data)
86     return None
87
88
89 # Scraping market cap data
90 dfmktcap = pd.DataFrame()
91 for url, company in zip(mktcap_urls, company_names_lower[:-2]):
92     data = scrape_market_cap(url, company)
93     if data is not None:
94         dfmktcap = pd.concat([dfmktcap, data])
95
96 #Scap price book
97 dfpricebook = pd.DataFrame()
98 for url, company in zip(pricebook_urls, company_names_lower[:-2]):
99     data = scrape_price_book(url, company)
100    if data is not None:
101        dfpricebook = pd.concat([dfpricebook, data])
102
103 #indexes
104 dfmktcap['Year'] = pd.to_datetime(dfmktcap['Year'])
105 dfmktcap['Year'] = pd.DatetimeIndex(dfmktcap['Year']).year
106
107 dfpricebook['Year'] = pd.to_datetime(dfpricebook['Year'])
108 dfpricebook['Year'] = pd.DatetimeIndex(dfpricebook['Year']).year
109
110
111 ##Merge datasets

```

```

112 final_firm = dfmktcap.copy()
113 final_firm = final_firm.merge(dfpricebook, how='outer', on=['Year', 'Company'])
114
115
116 del dfmktcap, dfpricebook, mktcap_urls, pricebook_urls, url, data, company
117
118
119 missing = final_firm[final_firm.isna().any(axis=1)]
120 missing = missing.sort_values(by=['Year'])
121 missing = missing.reset_index()
122 #have both data points for all firms for 2010-2022
123 # in 09 only missing data is from BIC, Carrefour, Ubisolft, AirFrance
124
125 #remove 2023
126 final_firm = final_firm[final_firm.Year != 2023]
127
128 #Get book to market ratio = inverse of price-book ratio
129 final_firm['PriceBook'] = pd.to_numeric(final_firm['PriceBook'], errors='coerce')
130 final_firm['PriceBook'].replace('nan', np.nan, inplace=True)
131 final_firm['BookMarket'] = final_firm['PriceBook'].apply(lambda x: x ** -1 if not pd.isnull(x)
    ) else np.nan)
132
133 #Clean MarketCap
134 final_firm['MarketCap'] = (final_firm['MarketCap'].replace({'\$: ': '', ' B': ''}, regex=True).
    astype(float) * 1_000) # Clear the letters, convert to float and scale to millions
135
136 #Date format
137 final_firm['Year'] = pd.to_datetime(final_firm['Year'], format='%Y')
138
139 #====Get monthly data
140 monthly_data = pd.DataFrame()
141 # Repeat the yearly data for each month and each firm
142 for index, row in final_firm.iterrows():
143     firm_data = pd.DataFrame()
144     monthly_year = pd.date_range(start=row['Year'], periods=12, freq='MS')
145     firm_data['Date'] = monthly_year
146     firm_data['Company'] = row['Company']
147     firm_data['MarketCap'] = row['MarketCap']
148     firm_data['BookMarket'] = row['BookMarket']
149     firm_data['PriceBook'] = row['PriceBook']
150     monthly_data = pd.concat([monthly_data, firm_data])
151
152 del index, monthly_year, row, firm_data
153
154 firms_year = final_firm.copy()
155 firms_month = monthly_data.copy()
156
157 del final_firm, monthly_data, missing

```

```

158
159 %%Get return data with Yahoo finance
160
161 start = datetime.datetime(2002, 1, 1)
162 end = datetime.datetime(2022, 12, 31)
163
164 #Get all data
165 data = yf.download(tickers, start=start,
166                     end=end)
167
168 #Focus on adjusted closed values only
169 adjclose=data['Adj Close']
170 adjclose = adjclose.set_axis(company_names_lower, axis=1)
171
172 #Use monthly data: mean of the months value
173 adjclose = adjclose.resample('1M').mean(numeric_only=True)
174 adjclose_y = adjclose.resample('1Y').mean(numeric_only=True)
175
176 ## extract CAC40 and exchange rate
177 cac_xrate_month = adjclose.loc[:, ["CAC40", "EuroUSD"]]
178 cac_xrate_year = adjclose_y.loc[:, ["CAC40", "EuroUSD"]]
179 adjclose = adjclose.drop(columns=["CAC40", "EuroUSD"])
180 adjclose_y = adjclose_y.drop(columns=["CAC40", "EuroUSD"])
181
182 #Reshape
183 prices_monthly = pd.melt(adjclose, value_vars=company_names_lower[0:-2], ignore_index=False)
184 prices_yearly = pd.melt(adjclose_y, value_vars=company_names_lower[0:-2], ignore_index=False)
185
186
187 del data, adjclose, adjclose_y
188
189 #I'm not getting right values for xrate when downloading in bulk
190 data = yf.download(['EURUSD=X', '^FCHI'], start=start, end=end)
191 adjclose=data['Adj Close']
192 adjclose = adjclose.set_axis(['EuroUSD', 'CAC40'], axis=1)
193 #adjclose = pd.DataFrame(adjclose, columns=['Date', 'EuroUSD'])
194
195 cac_xrate_month = adjclose.resample('1M').mean(numeric_only=True)
196 cac_xrate_year = adjclose.resample('1Y').mean(numeric_only=True)
197
198 del data, adjclose, start, end
199
200
201 %% Endogeneous factor: long term inflation expectation from external file
202
203 #upload the Agence France Tresor data
204 pi_endo= pd.read_excel('2023_11_01_rend_tit_ref_oatei.xls', skiprows=[0,1,2,3,4], usecols
                    =[0,3])

```

```

205 pi_endo.columns = ['Date', "Breakeven"]
206
207 pi_endo["Date"] = pd.to_datetime(pi_endo["Date"])
208 pi_endo= pi_endo.set_index(pi_endo["Date"])
209 pi_endo.drop(columns=['Date'])
210
211 #get monthly data
212 piendo_month = pi_endo.resample('1M').mean(numeric_only=True)
213
214 #get yearly data
215 piendo_year = pi_endo.resample('1Y').mean(numeric_only=True)
216
217 del pi_endo
218
219 ### French and Fama - their data
220
221 df = pd.read_csv('Europe_3_Factors.csv',skiprows=[0,1,2])
222
223 #montly data, need to fix dates
224 frenchfama_month = df.iloc[:399,:]
225
226 frenchfama_month['Unnamed: 0'] = frenchfama_month['Unnamed: 0'].astype(str) # Convert to
    string for manipulation
227 frenchfama_month['Year'] = frenchfama_month['Unnamed: 0'].str[:4] # Extract year from the
    encoded date
228 frenchfama_month['Month'] = frenchfama_month['Unnamed: 0'].str[4:] # Extract month from the
    encoded date
229 frenchfama_month['Date'] = pd.to_datetime(dict(year=frenchfama_month['Year'], month=
    frenchfama_month['Month'], day=1))
230 frenchfama_month.drop(['Year', 'Month', 'Unnamed: 0'], axis=1, inplace=True)
231 frenchfama_month = frenchfama_month.set_index(frenchfama_month['Date'])
232 frenchfama_month = frenchfama_month.drop(columns=["Date"])
233 frenchfama_month = frenchfama_month.loc['2002-01-01':]
234 frenchfama_month = frenchfama_month.astype(float)
235
236
237 #yearly data
238 frenchfama_year = df.iloc[402:,:]
239 frenchfama_year["Unnamed: 0"] = pd.to_datetime(frenchfama_year['Unnamed: 0'])
240 frenchfama_year.rename(columns={"Unnamed: 0":'Date'}, inplace=True)
241 frenchfama_year = frenchfama_year.set_index(frenchfama_year['Date'])
242 frenchfama_year = frenchfama_year.drop(columns=["Date"])
243 frenchfama_year = frenchfama_year.loc['2002-01-01':]
244 frenchfama_year = frenchfama_year.astype(float)
245
246 del df
247
248

```



```

249 ### Macro data
250 #APIs didn't work as planned
251
252 ## PIB Q
253 pib = pd.read_csv("ECB_PIB.csv")
254 pib.columns = ['Date', 'Q', 'pib']
255 pib['Date'] = pd.to_datetime(pib['Date'])
256 pib = pib.drop(columns=['Q'])
257 pib = pib.set_index(pib['Date'])
258 pib = pib.loc['2002-01-01':]
259 pib = pib.drop(columns=['Date'])
260
261 #monthly
262 pib_monthly = pib.resample('MS').ffill()
263 #yearly
264 pib_year = pib.resample('1Y').last()
265
266 del pib
267
268 ##risk free AAA
269 rkfreeAAA = pd.read_csv('ECB_yield.csv')
270 rkfreeAAA.columns=['Date', "time", "rkfreeAAA"]
271 rkfreeAAA['Date'] = pd.to_datetime(rkfreeAAA['Date'])
272 rkfreeAAA = rkfreeAAA.set_index(rkfreeAAA['Date'])
273 rkfreeAAA = rkfreeAAA.drop(columns=['time', "Date"])
274
275 rkfreeAAA_monthly = rkfreeAAA.resample("1M").mean(numeric_only=True)
276 rkfreeAAA_year = rkfreeAAA.resample("1Y").mean(numeric_only=True)
277
278 del rkfreeAAA
279
280 ##HICP
281 pi_month = pd.read_csv("HICP.csv", sep=';', encoding = 'latin1', skiprows=[0,1,2,3], usecols
    = [0,1], header=None)
282 pi_month.columns= ['Date', "infl"]
283 pi_month["Date"] = pd.to_datetime(pi_month['Date'], format = "%Y-%m")
284 pi_month=pi_month.set_index(pi_month["Date"])
285 pi_month = pi_month.drop(columns=['Date'])
286
287 #yearly
288 pi_year = pi_month.resample("1Y").mean(numeric_only=True)
289
290
291 ##OAT
292 oat = pd.read_csv('OAT.csv', sep=';', skiprows=[0,1,2,3,4,5], usecols=[0,5], header=None)
293 oat.columns = ['Date', 'oat']
294 oat['Date'] = pd.to_datetime(oat["Date"])
295 oat = oat.set_index(oat["Date"])

```

```

296 oat = oat.drop(columns=["Date"])
297 oat = oat.loc["2002-01-01":]
298
299 oat['oat'] = oat['oat'].replace("-", np.nan)
300 oat['oat'] = oat['oat'].str.replace(',', '.').astype(float)
301
302 oat_month = oat.resample('M').mean(numeric_only=True)
303 oat_year = oat.resample('1Y').mean(numeric_only=True)
304
305 del oat
306
307
308 ### Merge the dataframes
309
310 === MONTHLY
311 #macro stuff, only date index matter
312 monthly = pd.concat([cac_xrate_month, frenchfama_month, oat_month, pi_month, pib_monthly,
313                     piendo_month, rkfreeAAA_monthly], axis=1)
314 monthly = monthly.resample('M').last() #some tables encoded end of month, others on the 1st
315 monthly.to_csv('Monthly_series.csv')
316
317
318 #firm specific data
319 firms_month['Date'] = firms_month['Date'] + pd.offsets.MonthEnd(0) #all other df have eomonth
320                                     date
321 firms_month = firms_month.set_index(['Date'])
322 firms_month = firms_month.set_index('Company', append=True)
323
324 prices_monthly.columns = ['Company', 'value']
325 prices_monthly.set_index('Company', append=True)
326
327 monthly_stock = pd.merge(prices_monthly.reset_index(), firms_month.reset_index(), on=["Date",
328                                     "Company"], how='outer').set_index(["Date", "Company"])
329 monthly_stock.to_csv("Firm_monthly.csv")
330
331 #merge the two
332 merged_monthly = pd.merge(monthly, monthly_stock, left_index=True, right_index=True, how='
333                                     right')
334
335 #Export df in csv
336 merged_monthly.to_csv("DATA_month.csv")
337
338 === YEARLY
339 yearly = pd.concat([cac_xrate_year, frenchfama_year, oat_year, pi_year, pib_year, piendo_year,
340                     rkfreeAAA_year], axis=1)

```

```

339 yearly = yearly.resample('Y').last()
340
341 yearly.to_csv('Yearly_series.csv')
342
343 # =====
344 # #firm specific
345 # firms_year.rename(columns={"Year":'Date'}, inplace=True)
346 # firms_year['Date'] = firms_year['Date'] + pd.offsets.MonthEnd(0)
347 # firms_year = firms_year.set_index(['Date'])
348 # firms_year = firms_year.set_index('Company', append=True)
349 #
350 # prices_yearly.columns = ['Company', 'value']
351 # prices_yearly = prices_yearly.set_index('Company', append=True)
352 #
353 # yearly_stock = pd.merge(prices_yearly.reset_index(), firms_year.reset_index(), on=["Date",
354 #     "Company"], how='outer').set_index(["Date", "Company"])
355 # =====
356 #yearly_stock = monthly_stock.groupby('Company').resample('Y').mean()
357
358 monthly_data_reset = monthly_stock.reset_index()
359 yearly_stock = monthly_data_reset.groupby('Company').resample('Y', on='Date').mean()
360 #yearly_stock = yearly_stock.set_index(["Date", "Company"])
361
362 yearly_stock.to_csv("Firm_yearly.csv")
363
364
365 #merge the two
366 merged_year = pd.merge(yearly, yearly_stock, left_index=True, right_index=True, how='right')
367
368 merged_year.to_csv("DATA_yearly.csv")

```

Appendix C Code - Analysis in R

```
1 #%% AP 1
2 #%% ncardenasfrias
3
4 pacman::p_load(data.table, urca, dotwisker, tidyverse, ggplots, xts, stargazer, forecast, plm,
5                 ggplot2, tidyr)
6 library(dplyr)
7 setwd('/Users/nataliacardenasf/Documents/GitHub/PROJECTS_AP_FE/AP 1')
8
9 df <- fread("DATA_month.csv")
10 as.data.table(df)
11 #####
12 ## Identify the risk factors
13 #####
14
15 #Need to remove the predictable part to the endogeneous and exogeneous series
16
17 #upload the monthly data with the yearly factors and transform into a list of TS
18 data = read.csv('Monthly_series.csv')
19 data$Date <- as.Date(data$Date)
20
21 filtered_data <- data %>% #Filter rows between 2005 and 2022
22   filter(Date >= as.Date("2005-01-01") & Date <= as.Date("2022-12-31"))
23 time_series_cols <- filtered_data %>%
24   select(-Date)
25 time_series_list <- lapply(time_series_cols, function(col) {
26   ts_values <- ts(col, start = c(year(min(filtered_data$Date)), month(min(filtered_data$Date))
27     ), frequency = 12)
28   return(ts_values)
29 })
30 names(time_series_list) <- names(time_series_cols)
31
32 #plot series
33 time_series_df <- as.data.frame(time_series_list)
34 time_series_df$Date <- time(time_series_list[[1]])
35 time_series_long <- pivot_longer(time_series_df, cols = -Date, names_to = "Series", values_to
36   = "Value")
37 ggplot(time_series_long, aes(x = Date, y = Value, color = Series)) +
38   geom_line() +
39   facet_wrap(~ Series, scales = "free_y") +
40   labs(x = "Time", y = "Value")
41
42 ##check stationarity
43
44 lv = filtered_data[,c("EuroUSD", "CAC40", "infl", "pib", "Breakeven", "rkfreeAAA", 'oat')]
```

```

44
45 lv.adf.ln.trend = list(
46   XR = ur.df(lv$EuroUSD, type='trend', selectlags = c('BIC')),
47   cac = ur.df(lv$CAC40, type='trend', selectlags = c('BIC')),
48   infl = ur.df(lv$infl, type='trend', selectlags = c('BIC')),
49   pib = ur.df(lv$pib, type='trend', selectlags = c('BIC')),
50   #breakeven = ur.df(lv$Breakeven_no_na, type='trend', selectlags = c('BIC')),
51   rfAAA = ur.df(lv$rkfreeAAA, type='trend', selectlags = c('BIC')),
52   oat = ur.df(lv$oat, type='trend', selectlags = c('BIC'))
53 )
54
55 summary(lv.adf.ln.drift$XR)
56 print("levelVariable with drift and trend")
57 test = cbind(t(lv.adf.ln.trend$XR@teststat), t(lv.adf.ln.trend$cac@teststat),
58             t(lv.adf.ln.trend$infl@teststat), t(lv.adf.ln.trend$pib@teststat),
59             t(lv.adf.ln.trend$rfAAA@teststat), t(lv.adf.ln.trend$oat@teststat),
60             lv.adf.ln.trend$XR@cval)
61 #stargazer(test, out = 'Tables/trend_macro.tex')
62
63 lv.adf.ln.drift = list(
64   XR = ur.df(lv$EuroUSD, type='drift', selectlags = c('BIC')),
65   cac = ur.df(lv$CAC40, type='drift', selectlags = c('BIC')),
66   infl = ur.df(lv$infl, type='drift', selectlags = c('BIC')),
67   pib = ur.df(lv$pib, type='drift', selectlags = c('BIC')),
68   #breakeven = ur.df(lv$Breakeven_no_na, type='drift', selectlags = c('BIC')),
69   rfAAA = ur.df(lv$rkfreeAAA, type='drift', selectlags = c('BIC')),
70   oat = ur.df(lv$oat, type='drift', selectlags = c('BIC'))
71 )
72 print("levelVariable with drift ")
73 stat_macro_drift = cbind(t(lv.adf.ln.drift$XR@teststat), t(lv.adf.ln.drift$cac@teststat),
74                         t(lv.adf.ln.drift$infl@teststat), t(lv.adf.ln.drift$pib@teststat),
75                         t(lv.adf.ln.drift$rfAAA@teststat), t(lv.adf.ln.drift$oat@teststat),
76                         lv.adf.ln.drift$XR@cval)
77 #stargazer(stat_macro_drift, out='Tables/drift_macro.tex')
78
79 lv.adf.ln.none = list(
80   XR = ur.df(lv$EuroUSD, type='none', selectlags = c('BIC')),
81   cac = ur.df(lv$CAC40, type='none', selectlags = c('BIC')),
82   infl = ur.df(lv$infl, type='none', selectlags = c('BIC')),
83   pib = ur.df(lv$pib, type='none', selectlags = c('BIC')),
84   #breakeven = ur.df(lv$Breakeven_no_na, type='none', selectlags = c('BIC')),
85   rfAAA = ur.df(lv$rkfreeAAA, type='none', selectlags = c('BIC')),
86   oat = ur.df(lv$oat, type='none', selectlags = c('BIC'))
87 )
88 print("levelVariable with none ")
89 stat_macro_none = cbind(t(lv.adf.ln.none$XR@teststat), t(lv.adf.ln.none$cac@teststat),
90                       t(lv.adf.ln.none$infl@teststat), t(lv.adf.ln.none$pib@teststat),
91                       t(lv.adf.ln.none$rfAAA@teststat), t(lv.adf.ln.none$oat@teststat),

```

```

92         lv.adf.ln.none$XR@cval)
93 #stargazer(stat_macro_none, out='Tables/drift_none.tex')
94
95
96
97
98 #differentiate
99 filtered_data$logGDP = log(filtered_data$pib)
100
101 filtered_data$DlogGDP <- c(NA, diff(filtered_data$logGDP))
102 filtered_data$DCAC40 <- c(NA, diff(filtered_data$CAC40))
103 filtered_data$DEuroUSD <- c(NA, diff(filtered_data$EuroUSD))
104 filtered_data$Dinfl <- c(NA, diff(filtered_data$infl))
105 filtered_data$Doat <- c(NA, diff(filtered_data$oat))
106 filtered_data$DrfAAA <- c(NA, diff(filtered_data$rkfreeAAA))
107
108 time_series_cols <- filtered_data %>%
109   select(-Date)
110 time_series_list <- lapply(time_series_cols, function(col) {
111   ts_values <- ts(col, start = c(year(min(filtered_data$Date)), month(min(filtered_data$Date))
112     ), frequency = 12)
113   return(ts_values)
114 })
115 names(time_series_list) <- names(time_series_cols)
116
117 #plot series
118 time_series_df <- as.data.frame(time_series_list)
119 time_series_df$Date <- time(time_series_list[[1]])
120 time_series_long <- pivot_longer(time_series_df, cols = -Date, names_to = "Series", values_to
  = "Value")
121 ggplot(time_series_long, aes(x = Date, y = Value, color = Series)) +
122   geom_line() +
123   facet_wrap(~ Series, scales = "free_y") +
124   labs(x = "Time", y = "Value")
125
126 # Auto arima differentiates as much as needed to get stationary variables
127
128 logGDP_arima = auto.arima(filtered_data$logGDP) #ARIMA 0,1,0
129 logGDP_arima
130 EuroUSD_arima = auto.arima(filtered_data$EuroUSD) #ARIMA 0,1,1
131 EuroUSD_arima
132 infl_arima = auto.arima(filtered_data$infl) #ARIMA 1,1,1
133 infl_arima
134 breakeven_arima = auto.arima(filtered_data$Breakeven) #ARIMA 2,1,2
135 breakeven_arima
136
137 timestamps <- filtered_data$Date

```

```

138 res_pib = resid(logGDP_arima)
139 res_xr = resid(EuroUSD_arima)
140 res_infl = resid(infl_arima)
141 res_bkeven = resid(breakeven_arima)
142 min_length <- min(length(res_pib), length(res_xr), length(res_infl))
143 # Create a dataframe with aligned timestamps and residuals padded with NA for breakeven
144 residuals_df <- data.frame(
145   Date = timestamps[1:min_length],
146   res_pib = c(res_pib[1:min_length], rep(NA, times = max(0, length(timestamps) - min_length)))
147   ,
148   res_xr = c(res_xr[1:min_length], rep(NA, times = max(0, length(timestamps) - min_length))),
149   res_infl = c(res_infl[1:min_length], rep(NA, times = max(0, length(timestamps) - min_length)
150   )),
151   res_bkeven = c(res_bkeven[1:min_length], rep(NA, times = max(0, length(timestamps) - min_
152   length)))
153 )
154 #####
155 ## Estimate the beta coefficients
156 #####
157
158 data_firm = read.csv('Firm_monthly.csv')
159 data_firm$Date <- as.Date(data_firm$Date)
160
161 filtered_data_firm <- data_firm %>% #Filter rows between 2005 and 2022
162   filter(Date >= as.Date("2005-01-01") & Date <= as.Date("2022-12-31"))
163 time_series_cols_firm <- filtered_data_firm %>%
164   select(-Date)
165 time_series_list_firm <- lapply(time_series_cols_firm, function(col) {
166   ts_values_firm <- ts(col, start = c(year(min(filtered_data_firm$Date)), month(min(filtered_
167   data_firm$Date))), frequency = 12)
168   return(ts_values_firm)
169 })
170 names(time_series_list_firm) <- names(time_series_cols_firm)
171
172 ## to get return, differenciate value of the stock (within each company)
173 filtered_data_firm <- filtered_data_firm[order(filtered_data_firm$Company, filtered_data_firm$
174   Date), ]
175 filtered_data_firm$Return <- with(filtered_data_firm, ave(value, Company, FUN = function(x) c(
176   NA, diff(x))))
177
178 finalmonthly =merge(filtered_data_firm, filtered_data, by = "Date")
179 finalmonthly =merge(finalmonthly, residuals_df, by = "Date")
180
181 #split df by company to get a regression per stock

```

```

180 dta_bystock = split(finalmonthly, finalmonthly$Company)
181
182
183 #Exogeneous factors only
184 fit_lm <- function(data) {
185   lm(Return ~ res_pib + res_xr + res_infl, data = data)
186 }
187 regs_beta <- lapply(dta_bystock, function(subset) fit_lm(subset))
188
189 summary(regs_beta)
190 stargazer(regs_beta,
191           title = "Estimate the beta coefficients for each exogeneous factor",
192           column.labels = names(regs_beta),
193           out="Tables/betas_exo.tex")
194
195 library(dotwhisker)
196
197 dwplot(regs_beta)
198
199 #Include endofactor -> start in 2013
200 fit_lm_endo <- function(data) {
201   lm(Return ~ res_pib + res_xr + res_infl+ res_bkeven, data = data)
202 }
203 regs_beta_endo <- lapply(dta_bystock, function(subset) fit_lm_endo(subset))
204 stargazer(regs_beta_endo,
205           title = "Estimate the beta coefficients for exogeneous and endogeneous factors
206           (2013-2022)",
207           column.labels = names(regs_beta),
208           out="Tables/betas_exo_endo.tex")
209
210 dwplot(regs_beta_endo)
211
212 #Include FF factors
213 fit_lm_exoff <- function(data) {
214   lm(Return ~ res_pib + res_xr + res_infl+ HML+SMB+Mkt.RF, data = data)
215 }
216
217 regs_beta_exoff <- lapply(dta_bystock, function(subset) fit_lm_exoff(subset))
218
219 summary(regs_beta_exoff)
220 stargazer(regs_beta_exoff,
221           title = "Estimate the beta coefficients for each exogeneous factor and French and
222           Fama factors",
223           column.labels = names(regs_beta),
224           out="Tables/betas_exo_ff.tex")
225
226 dwplot(regs_beta_exoff)
227
228 ##### FF model with annual data and our estimation for this sample
229 Firm_yearly <- read_csv("Firm_yearly.csv")

```



```

226 Firm_yearly = merge(Firm_yearly,
227                     Firm_yearly%>%
228                       filter(!is.na(MarketCap))%>%
229                       group_by(Date)%>%
230                       dplyr::summarise(MedianMct=median(MarketCap)),
231                     by="Date")
232 DiffMC = merge(
233   Firm_yearly%>%
234     filter(!is.na(MarketCap) & MarketCap>MedianMct & !is.na(value))%>%
235     group_by(Date)%>%
236     dplyr::summarise(AvgTop = mean(value)),
237   Firm_yearly%>%
238     filter(!is.na(MarketCap) & MarketCap<MedianMct & !is.na(value))%>%
239     group_by(Date)%>%
240     dplyr::summarise(AvgBot = mean(value)),
241   by = "Date")
242 DiffMC$SMBest = DiffMC$AvgTop-DiffMC$AvgBot
243
244 Firm_yearly = merge(Firm_yearly,
245                     Firm_yearly%>%
246                       filter(!is.na(BookMarket))%>%
247                       group_by(Date)%>%
248                       dplyr::summarise(MedianBmt=median(BookMarket)),
249                     by="Date")
250 DiffBM = merge(
251   Firm_yearly%>%
252     filter(!is.na(BookMarket) & BookMarket>MedianBmt & !is.na(value))%>%
253     group_by(Date)%>%
254     dplyr::summarise(AvgTop = mean(value)),
255   Firm_yearly%>%
256     filter(!is.na(BookMarket) & BookMarket<MedianBmt & !is.na(value))%>%
257     group_by(Date)%>%
258     dplyr::summarise(AvgBot = mean(value)),
259   by = "Date")
260 DiffBM$HMLest = DiffBM$AvgTop-DiffBM$AvgBot
261
262 Firm_yearly =merge(Firm_yearly, DiffBM, by = "Date")
263 Firm_yearly =merge(Firm_yearly, DiffMC, by = "Date")
264
265 Data_year = read_csv("DATA_yearly.csv")
266 Firm_yearly =merge(Firm_yearly, Data_year, by = "Date")
267 Firm_yearly$Date = as.Date(Firm_yearly$Date)
268
269 Firm_yearly <- Firm_yearly[order(Firm_yearly$Company.x, Firm_yearly$Date), ]
270 Firm_yearly$return <- with(Firm_yearly, ave(value.x, Company.x, FUN = function(x) c(NA, diff(x
  ))))
271
272 Firm_yearly$DCAC40 = c(NA, diff(Firm_yearly$CAC40))

```

```

273 Firm_yearly$Doat = c(NA, diff(Firm_yearly$Doat))
274 Firm_yearly$mkt_free = Firm_yearly$DCAC40 - Firm_yearly$Doat
275
276 filtered_Firm_yearly <- Firm_yearly %>% #Filter rows between 2010 and 2022 (no missing values)
277   filter(Date >= as.Date("2010-01-01") & Date <= as.Date("2022-12-31"))
278
279 fit_lm_myFFonly <- function(data) {
280   lm(Return ~ mkt_free+HMLest+SMBest, data = data)
281 }
282 year_by_stock_FF = split(filtered_Firm_yearly, filtered_Firm_yearly$Company.x)
283 regs_beta_myFF <- lapply(year_by_stock_FF, function(subset) fit_lm_myFFonly(subset))
284
285 stargazer(regs_beta_myFF,
286           title = "Estimate the beta coefficients for computed French and Fama factors
287             (2010-2022)",
288           column.labels = names(regs_beta_myFF),
289           out="Tables/betas_myff.tex")
290
291
292
293
294
295
296 #####
297 ## Estimate the lambdas
298 #####
299
300
301 #Extract the beta coefficients of reg with exo and FF factors
302 # Initialize an empty dataframe to store coefficients and model names
303 coefficients_df <- data.frame(Model = character(), Intercept = numeric(),
304                               beta_pib = numeric(), beta_xr = numeric(),
305                               beta_infl = numeric(), beta_HML = numeric(),
306                               beta_SMB = numeric(), beta_Mkt.RF = numeric(),
307                               stringsAsFactors = FALSE)
308
309 # Iterate through subsets of data and fit models
310 for (i in seq_along(dta_bystock)) {
311   subset <- dta_bystock[[i]]
312   model <- fit_lm_exoff(subset)
313   coefficients <- c(model$coefficients[1], model$coefficients[-1])
314   row <- data.frame(Model = paste0("Model_", i),
315                     Intercept = coefficients[1],
316                     beta_pib = coefficients[2],
317                     beta_xt = coefficients[3],
318                     beta_infl = coefficients[4],
319                     beta_HML = coefficients[5],

```

```

320         beta_SMB = coefficients[6],
321         beta_Mkt.RF = coefficients[7])
322     coefficients_df <- rbind(coefficients_df, row)
323 }
324
325 coefficients_df
326 coefficients_df$Company = names(regs_beta_exoff) #added column with company name
327
328
329
330 #Get the historical mean return of every stock
331
332 historical_mean_returns <- finalmonthly %>%
333   group_by(Company) %>%
334   summarise(mean_return = mean(Return, na.rm = TRUE))
335
336 #merge two data sets
337 multibeta = merge(coefficients_df, historical_mean_returns, by="Company")
338
339 ## Get return - intercept
340 multibeta$mean_intercept = multibeta$mean_return - multibeta$Intercept
341
342 ## Run std linear model
343 model_multibeta = lm(mean_intercept ~ beta_pib+beta_xt+beta_infl+beta_HML+beta_SMB+beta_Mkt.RF
344   , data= multibeta)
345 summary(model_multibeta)
346 stargazer(model_multibeta, out='Tables/LM_multibeta.tex')
347
348 ## Use GLS to correct the regression
349 library(nlme)
350
351 #get the var-cov matrix of the initial regression to use as weights
352 residuals_list <- list()
353
354 # Loop through each model to extract residuals
355 for (i in seq_along(regs_beta_exoff)) {
356   # Get residuals for each model
357   residuals <- residuals(regs_beta[[i]])
358
359   # Store residuals in the list
360   residuals_list[[i]] <- residuals
361 }
362
363 # Combine residuals into a single matrix
364 combined_residuals <- do.call(rbind, residuals_list)
365
366 # Calculate variance-covariance matrix of combined residuals

```

```

367 var_cov_matrix_pooled <- cov(combined_residuals)
368
369 install.packages("MASS")
370 library(MASS)
371 inv_var_cov = solve(var_cov_matrix_pooled)
372
373 model_multibeta2 = gls(mean_intercept ~ beta_pib+beta_xt+beta_infl+beta_HML+beta_SMB+beta_Mkt.
    RF, data= multibeta, weights=varComb())
374
375
376
377 #####
378 #Test the validity of the multi-beta relationship
379 #####

```

References

- Nai-Fu Chen, Richard Roll, and Stephen A. Ross. Economic Forces and the Stock Market. *The Journal of Business*, 59(3):383–403, 1986. ISSN 0021-9398.
- Eugene F. Fama and Kenneth R. French. The Cross-Section of Expected Stock Returns. *The Journal of Finance*, 47(2):427–465, 1992. ISSN 0022-1082. doi: 10.2307/2329112.
- Eugene F. Fama and Kenneth R. French. Common risk factors in the returns on stocks and bonds. *Journal of Financial Economics*, 33(1):3–56, February 1993. ISSN 0304-405X. doi: 10.1016/0304-405X(93)90023-5.
- Stephen A Ross. The arbitrage theory of capital asset pricing. *Journal of Economic Theory*, 13(3):341–360, December 1976. ISSN 0022-0531. doi: 10.1016/0022-0531(76)90046-6.