A QUANTITATIVE RESEARCH ON CROSS-SECTIONAL DISPERSION AND EXPECTED RETURNS

Jinfan Hu , Yipu Jin, Zhengshun You, Yiming Zou

3/12/2022

1. Introduction

This paper is a replication of a research paper by Thanos Verousis and Nikolaos Voukelatos in 2016, Cross-Sectional Dispersion and Expected Returns. The team followed the original paper's concept with a wider timeline of market and stock data. The data analysis and calculation were executed in Python with Jupyter Notebook, along with packages including pandas, numpy, and statsmodels. The final result negligibly differs from the primitive paper, however the team determined that the factor is capable of explaining and predicting the market volatility and therefore, predicting the returns.

2. Paper Summary

In this article, we review and attempt to reproduce the methods and results of "Cross-Sectional Dispersion and Expected Returns" by Thanos Verousis and Nikolaos Voukelatos(Verousis and Voukelatos 2015), originally published in Quantitative Finance in January 2018. This paper discussed the relation between cross-sectional dispersion and stock returns, and found out that stocks with lower sensitivity to the cross-sectional dispersion generally have higher returns than stocks with high sensitivity to the cross-sectional dispersion. They asserted that the cross-sectional dispersion can be priced as a stated variable. They prove this by formulating that after accounting for its exposure to other systematic risk factors, the cross-sectional dispersion can still have a statistically significant risk premium. They also showed that the cross-sectional dispersion could be a good proxy for aggregated idiosyncratic risk. And after adjusting for a wild set of stock characteristics, market conditions and industry groupings, the results are still robust.

The Cross-Sectional Dispersion (CSD) depicts the extent to how individual stocks' returns are clustered or diverges from the market return in the cross-sectional level. Which provides a reasonable measure for the heterogeneity of how individual stocks perform according to their own characteristics(Ang and Zhang 2006). Since the stocks' returns are driven by both systematic and idiosyncratic risks, the CSD could also be a close proxy to the aggregated idiosyncratic risks. Previous works had examined that dispersion has a strong negative relationship with market returns. Some of them also provided evidence for CSD's connection with Momentum and Value factors. However, there has been very few discussions on how dispersion relating to individual stocks returns. The paper is dedicated to explore the relationship between Cross-Sectional Dispersion, Individual stocks returns, together with their idiosyncratic risks.

They used a wide range of definitions to calculate CSD, and eventually proved that the results are indifference to the method they chose. Due to the non-stationarity, they chose the first order difference of CSD, to be the analyzing target. They illustrated that the average idiosyncratic risk of individual stocks can be decomposed into dispersion and market volatility, and since market risk has little power of predicting the future returns, the dispersion can be a potentially strong predictor. They ran a bivariate regression model on market risk factor and the dispersion factor, and obtained the individual stocks' exposure on dispersion. By using the

methods of (Fama and French 2002), they sorted the stocks and formed two portfolios 1-5 (long first quartile short fifth quantile) and N-P (long stocks with negative exposure short stocks with positive exposure). They showed that both portfolios have statistically significant positive returns, therefore proved that dispersion has a negative relationship with individual stocks.

Although the portfolios' returns are significant, much previous research had indicated the strong relation between dispersion and other systematic risk. In order to prove that the portfolios' return do not came from holding other systematic risks, they regressed the portfolios' returns on some of the most commonly used systematic risk factors, including the excess market return MKT, the two other (Fama and French 2002) factors SMB and HML, with the (Carhart 1997) momentum factor MOM. After accounting for these factors, the portfolios' return remained significant. After that, they also prove the predictive power of CSD is indifference to other individual characteristics of stocks(Brennan and Subrahmanyam 1998). After adjusting individual stocks' return with systemic factors, they ran cross-sectional returns on some stocks' idiosyncratic risk factors. Other than CSD, they also included the size (log market capitalization in millions), an idiosyncratic momentum factor (given by the past 6-month stock return), the standard deviation, skewness and kurtosis of stock returns over the past six months, the dispersion of analysts' forecasts about the stock's future earnings normalized by the mean forecast (similar to (Diether and Scherbina 2002)), a liquidity measure, the percentage of stock returns explained by systematic risk, as well as the co-skewness and idiosyncratic volatility of stock returns. The result remained significant after adjusting for the idiosyncratic risk factors. Then they constructed the doubled-sorted portfolios based on selected factors, although the portfolios' returns seemed to be co-varied with some factors, it could not disavow the predictiveness of CSD. However, the later on tests showed that with longer testing-period, the portfolios' returns became less significant. One credible explanation for that is with more data coming, the estimate of dispersion became less accurate.

They finally proved that the CSD is a priced state variable. They employ the standard (Fama and Macbeth 1973) two-pass methodology to extract the risk premium of cross-sectional dispersion risks. After obtaining the time-series of double-sorted investment portfolios, they ran regression for the excess returns of each of the portfolios with a set of systematic risk factors, the result shows that the coefficient for CSD is significant at 5% level. After that, they incorporated more variables that may demonstrate dispersion of stocks returns. Those variables includes FVIX (based on monthly changes VIX), an aggregate measure of dispersion of beliefs about the future earnings of stocks in the market (FDISP), macroeconomic uncertainty index UNC(B. Bali T. and Tang 2015). After converting the time-series of cross-sectional means to a mimicking portfolio (FIDVOL), they include this aggregate idiosyncratic volatility factor in the augmented Regression. The results suggested that the factor is shown to associate with a negative risk premium, and the cross-sectional dispersion is a distinct risk-premium from premia associated with the established systematic risk factors.

In the paper, the authors argued that Cross-Sectional dispersion is a priced factor, they proved that after adjusting for various systematic and idiosyncratic risk factors, the CSD remains predictive. Overall, they proved CSD is associated with negative risk-premium, stocks that are highly sensitive to dispersion will on average generate less returns that stocks that are less sensitive, and this phenomenon could be explained by the CSD's approximation to the aggregated idiosyncratic risks for all stocks.

3. Hypothesis Overview

1. CSD has negative correlation to individual stocks' return. Stocks with low sensitivity to CSD generally have lower returns than the ones with higher sensitivity.

a. Mimicking portfolio for CSD monthly (1-5, N-P)
b. Monthly rolling regression on MKT, CSD

3. Sort based on CSD beta

4. The risk premium of CSD is distinct from the premia generated by other systematic risk factors

a. Risk adjusted returns, with MKT, SMB, HML, MOM
b. Regression on monthly returns

3. After accounting for various stock characteristics, market conditions and industry groupings, the risk-premium of CSD is still statistically significant.

a. Risk adjusted returns, after accounting for systematic factors, with size, momentum, std, skew, kurt,
b. Double sorted portfolios

4. After accounting for other measurements of aggregated risk, the CSD still has significant explanatory power over the cross-section of stocks' returns.

a. Mimicking portfolio FCSD
b. Regression on double sorted portfolios, maybe add Liquidity factor
c. Also do b. To VIX, Stock Variance SVAR, index of macro uncertainty, cross-sectional means of residual volatility

***Due to the lack of data for analysts' prediction, we can not test the affect of FDISP factor in section 5.3

4. Literature Review

Idiosyncratic risk matters!

Amit Goyal and Pedro Santa-Clara(Amit Goyal 2003) found a significant positive relation between average stock variance and the return on the market, as opposed to systematic risk only, in one of their papers: Idiosyncratic Risk Matters! Their results are robust to small sample inference problems, in different sample periods, for alternative definitions of risk, for different portfolios, and persist even after controlling for business cycle variables known to predict the stock market. In fact, they show that it is the idiosyncratic component of total risk that drives the predictability of the market. The idiosyncratic component accounts for over 80 percent of total stock variance and over 70 percent of its variation through time. Moreover, when they regress market returns on pure idiosyncratic risk, they still retain significant predictive ability. Further, they find that the variance of the market by itself has no forecasting power for the market return. Rather, they believe that it is total risk that matters to investors and is therefore priced in market returns. The significance of idiosyncratic risk as a forecaster of market returns derives only from its importance as a large component of total risk and the precision with which it is measured. Similarly, the general insignificance of systematic risk in forecasting the market is due to its low weight on total risk and the large error in its measurement(Angelidis and Tessaromatis 2015). They also show that idiosyncratic risk explains most of the variation of average stock risk through time and it is idiosyncratic risk that drives the forecastability of the stock market.

A model-free measure of aggregate idiosyncratic volatility and the prediction of market returns.

In addition, a recent literature has revealed that the stock return dispersion may possibly forecast market return.((Garcia and Martellini 2014)) The paper formally focused on the cross-sectional variance of stock returns and discussed it as a consistent and asymptotically efficient estimator, especially when targeting aggregate idiosyncratic volatility. Two advantages were listed in detail: the cross-sectional variance of stock returns is a model-free factor, and it is observable at any frequency. These two advantages are valuable because most of the previous approaches for estimating idiosyncratic volatility have used models that exploit monthly measures, with a construction of a time series analysis of daily returns. However, with this newly developed method for volatility estimation, it is possible to measure the cross sectional volatility and use it as a strong predictor, especially when predicting and forecasting future returns on the aggregate stock market and most importantly, at a daily frequency. Garcia et al., demonstrated that a portfolio's exposures to the aggregate idiosyncratic volatility risk predict the cross section of the expected returns, with the analysis of

cross section of size and book-to-market portfolios. They verified that cross-sectional dispersion is a viable predicting factor over future market returns, when analyzed and examined at both the monthly and daily frequency. Nonetheless, Garcia et al., barely tested whether it is possible to price the 25 and 100 size/book-to-market portfolio, while the paper in this replication thoroughly inspected dispersion as a priced factor in the cross-section of market and stock returns, comprehensively.

Does Idiosyncratic Risk Proxy for Risk Exposure?

Chen and Retkova proposed a new state variable that is measured as the cross-sectional dispersion of stock returns around the market return and reflects the aggregate level of idiosyncratic risk in the market. They hypothesized that it should be expected to be associated with a negative risk premium which means stocks are found to offer expected returns that vary according to their sensitivity to changes in dispersion. Furthermore, a zero-cost spread portfolio that is long (short) in stocks with low (high) dispersion betas produces a statistically and economically significant return. And they performed a set of robustness tests to avoid the bias of stocks' idiosyncratic characteristics. More importantly, they showed that the reported dispersion premium is distinct from premia that have been previously found to be offered by other systematic factors which are related to uncertainty or heterogeneity, such as volatility, dispersion of analysts' earnings forecasts, mean stock variance, macroeconomic uncertainty and the mean idiosyncratic volatility of stock returns.

Differences of Opinion and the Cross-Section of Stock Returns.

Diether and Scherbina prove that stocks with higher dispersion in analysts' earnings forecasts earn significantly lower future returns than other similar stocks. Further, they present evidence that standard multifactor risk-based explanations cannot account for this relation. They also shows consistency of the hypothesis as prices will reflect the optimistic view whenever investors with the lowest valuations do not trade. Further, they argue that any friction that prevents the revelation of negative opinions will produce the negative relation between dispersion and future returns. To this end, they demonstrate that the incentive structure of analysts could serve as another such friction.

The Cross-Section Of Volatility And Expected Returns

The paper examined the impact of innovations in aggregate volatility on the cross-sectional stock returns. As a result, they showed that the expected return is lower for individual stocks that are more sensitive to changes in market volatility. They also found that stocks with high idiosyncratic risk, compared to the Fama and French (1993) model, have extremely low expected returns. The paper proved this theory robust and cannot be explained by size, book-to-market, momentum, and liquidity factors. In the paper that we are replicating, they found that FCSD, the changes in CSD variable, has a relatively strong correlation with FVIX which is the monthly changes in VIX index. The finding is consistent with their negative dispersion risk premium theory.

Stock Market Dispersion, The Business Cycle And Expected Factor Returns

The recent paper by Angelidis et al. provides evidence that the stock market return dispersion can be used as an economic state variable to predict changes in economic activity, returns of value, and momentum premia. In details, a high return dispersion is related to worse business conditions, higher value premium, smaller momentum premium, and lower market returns. They proved that return dispersion is a good predictor of future economy development as it is found that a high return dispersion is often followed by increased unemployment and greater likelihood of a recession. As a predictor of market returns and momentum premia, return dispersion is robust across periods, geographic locations, and measuring methods of the dispersion. The paper confirmed return dispersion is a better indicator than idiosyncratic stock volatility as it introduces less noise. The finding is consistence with our paper since there is less investing opportunities and worse stock performance in a bad business cycle.

Macroeconomic Uncertainty and Expected Stock Returns

The paper constructed an ex-ante measure of economic uncertainty with cross-sectional dispersion in the macroeconomic fundamentals forecasts from professionals. They estimated the exposure of individual stocks to the economic uncertainty index, called by uncertainty beta, and discovered its correlation with future cross-sectional dispersion in stock returns. They found that stocks that are most sensitive to economic

uncertainty yield lowest return, thus giving evidence to the negative link between the uncertainty beta and future stock returns. The effect is distinct from the negative market volatility risk premium and is robust across time periods and large and liquid stocks. The negative relation is confirmed in the paper that we aimed to replicate

5. Replication

5.1. Key Software Components

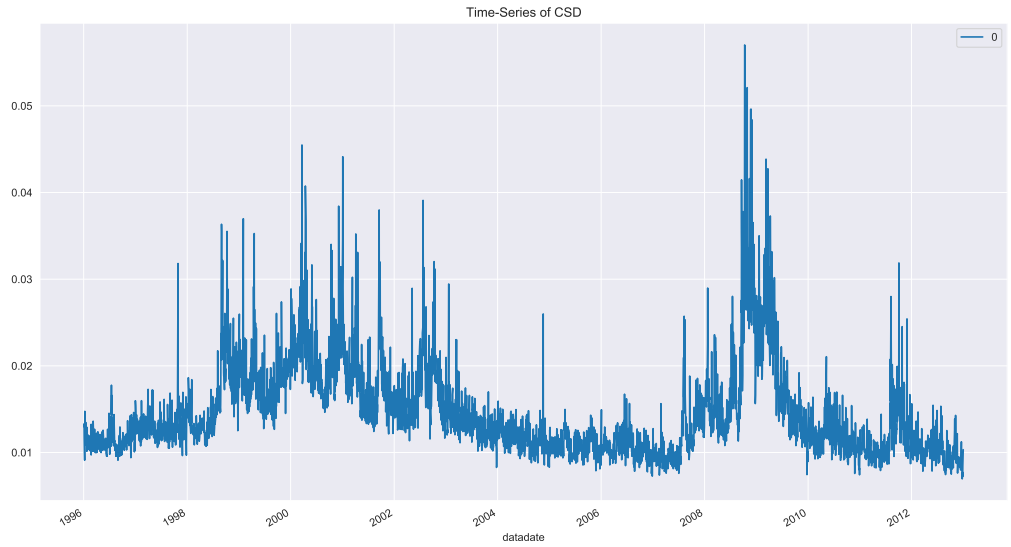
The paper that we aim to replicate examined the cross-section of equity returns from the US stock market, during January 1996 to December 2012. The original data was extracted from the Center for Research in Security Prices (CRSP) database, containing daily closing bid and ask quotes and trading volumes. The original method used the CRSP value-weighted index as a proxy for the aggregate market when computing the cross-sectional dispersion:

$$CSD_t = \frac{\sum_{j=1}^{N} |r_{i,t} - r_{mkt,t}|}{N-1}$$

In our replication, we obtained the stock market date from CRSP database and decided to use a time horizon from January 1990 to December 2019 for this replication, in purpose to further validate this factor. The data that the team acquired contains daily close, open, high, low price data. These data are carefully manipulated to construct a monthly return for every stock, with stock split and adjusting factor taken into account.

An important difference from the original research paper is that our team decided to use only the surviving stock from 1990 to 2019, which could be biased since these surviving stocks may represent only the profiting side of the overall US stock market. In this case, we decided to use the Standard & Poor 500 Composite Index to represent the market return $r_{mkt,t}$ since the S & P 500 index is constructed with top trending companies, which usually outperform the market. The team decided that this composition of the CSD factor is valid and proceeded on to further data analysis.

Plotting the daily frequency of the CSD time series from January 1990 to December 2019 in figure 1, we are able to calculate and conduct the mean and standard deviation, which are 1.33% and 0.54%. The original paper conducts a daily CSD mean and standard deviation of 1.16% and 0.44%, respectively. Results are relatively different because we used a different market return as benchmark, a different stock selection method, and a different timeline.

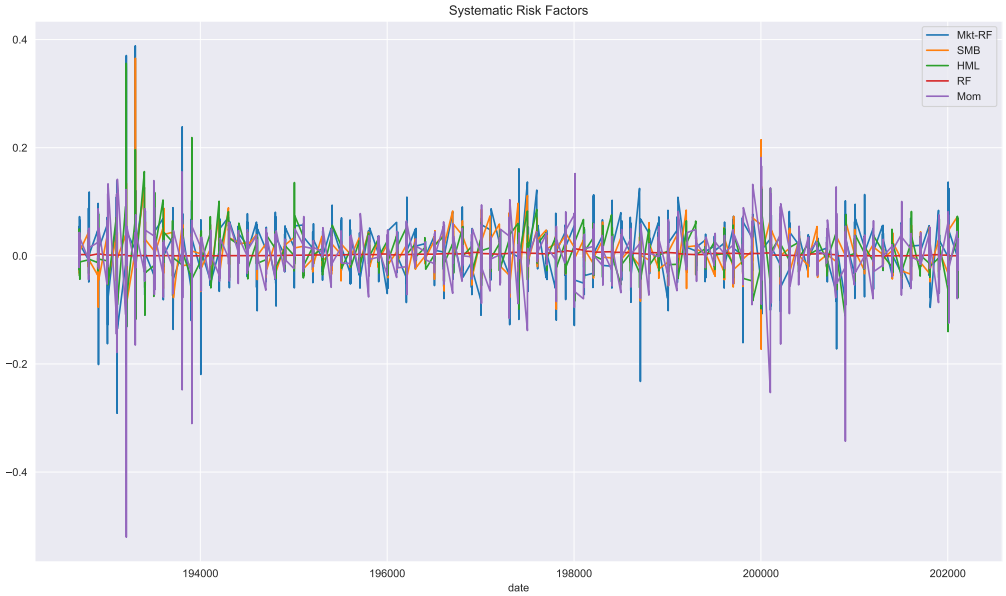


5.1.1 Data Source

We provide the source of the data we used and the methods we used to manage them to provide direction for future research. The stock data were obtained from CRSP database(CRSP, n.d.). We obtained the Fama-French Three factors as well as Carhart momentum factors from the data library in Kenneth French's website(French, n.d.), the VIX index from Bloomberg. The data of the index of the macroeconomic uncertainty was obtained from the website of Turan Bali(T. Bali, n.d.). Here are the brief descriptive analysis about our main data.

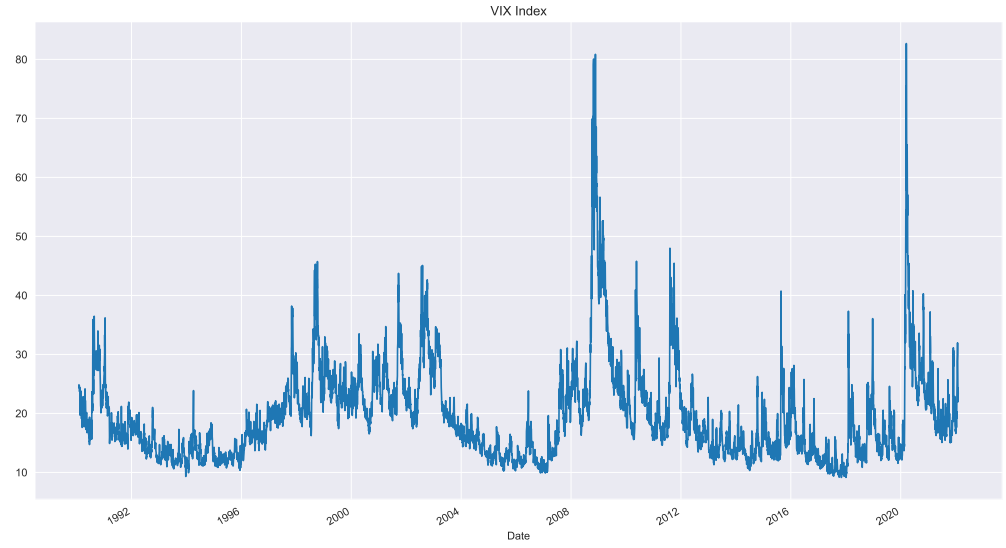
Factors

Fama-French three factors combined with Carhart momentum factors were seems as one of the standard source of systematic risk premia in many of the academic researches.



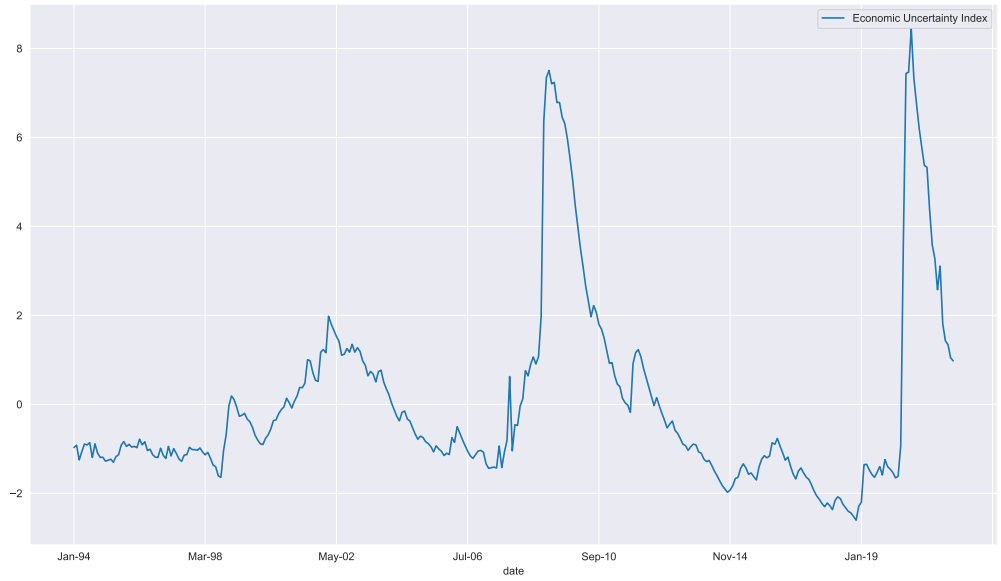
VIX

VIX index is the weighted average implied volatility for at-the-money S&P500 index options. It usually serves as a good proxy for the short-term market uncertainty.



Index of Macroeconomic uncertainty

This index is good measure for systematic risk in a macro level. It closely follows the fluctuation in business conditions and financial crisis period.



5.1.2. Data Management

The CRSP database we used contains all the stocks data since 1989 including fields like daily close price, daily trading volume, issue type of the asset etc. The original database has size of over 60 GB.In order to save our RAM and be able to extract required data efficiently, we read the original data by chunk using Python's pandas library, and stored the transformed data based on the fields in HDF format. Here are our main codes for storing the data.

```
import time

for date in ['2010-2019', '2000-2009', '1990-1999']:
  data = pd.read_csv(f"/content/drive/My Drive/FIN554/Data{date}.csv")
  data_new = data.loc[data.tic.isin(whole_list)]
  del data

  time.sleep(10)

  # Save the daily close price
  close_data = data_new.pivot_table(index = "datadate", columns = ["tic"], \
  values = ["prccd"], aggfunc = "mean")
  close_data.to_hdf(f"/content/drive/My Drive/FIN554/ClosePrice_2012.h5", \
  key = date, complevel = 9, complib = 'blosc')
  del close_data

  # Save the daily shares outstanding
  share_data = data_new.pivot_table(index = "datadate", columns = ["tic"], \
  values = ["cshoc"], aggfunc = "mean")
  share_data.to_hdf(f"/content/drive/My Drive/FIN554/ShareOutstanding_2012.h5", \
  key = date, complevel = 9, complib = 'blosc')
  del share_data
```

```
# Save the price adjusting factor
adj_data = data_new.pivot_table(index = "datadate", columns = ["tic"], \
values = ["ajexdi"], aggfunc = "mean")
adj_data.to_hdf(f"/content/drive/My Drive/FIN554/AdjustFactor_2012.h5", \
key = date, complevel = 9, complib = 'blosc')
del adj_data

trfd_data = data_new.pivot_table(index = "datadate", columns = ["tic"], \
values = ["trfd"], aggfunc = "mean")
trfd_data.to_hdf(f"/content/drive/My Drive/FIN554/TrfdFactor_2012.h5", \
key = date, complevel = 9, complib = 'blosc')
del trfd_data

# Delete original data frame and pause the program to save RAM
del data_new

time.sleep(10)
```

5.1.3 Software Usage

We conduct all our data processing task and statistical analysis in Python. We mainly used Pandas and Numpy modules for data management. The statistical analysis were conducted using statsmodels module. Our data and test results visualization were applied with Matplotlib and Seaborn modules in Python as well as knitr library in R.

5.2. Replication of Key Analytical Techniques

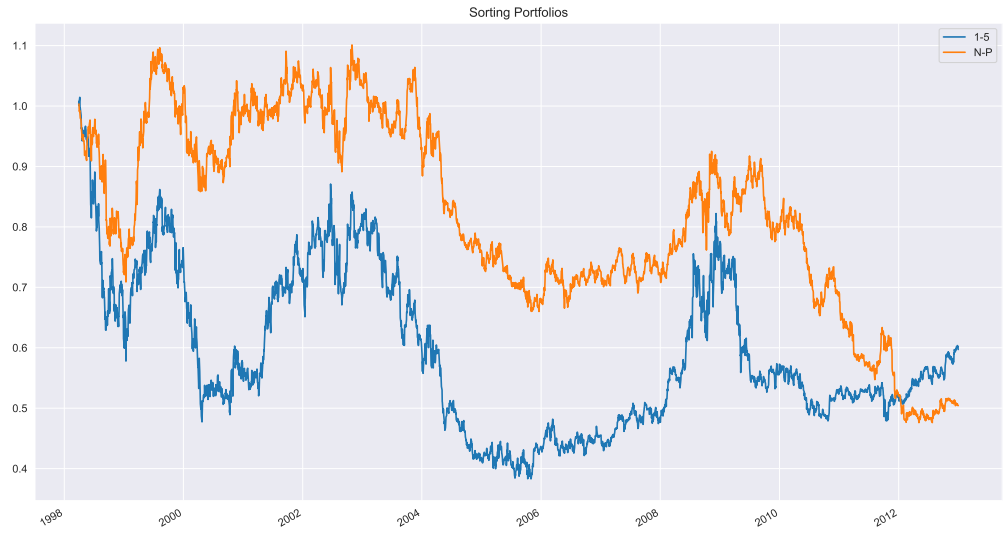
5.2.1 Mimicing portfolio for CSD monthly

We conducted regression on stocks' return and ΔCSD as well as MKT factor. Sort the $\beta\Delta CSD$ and form portfolios on different quintiles (1-5).

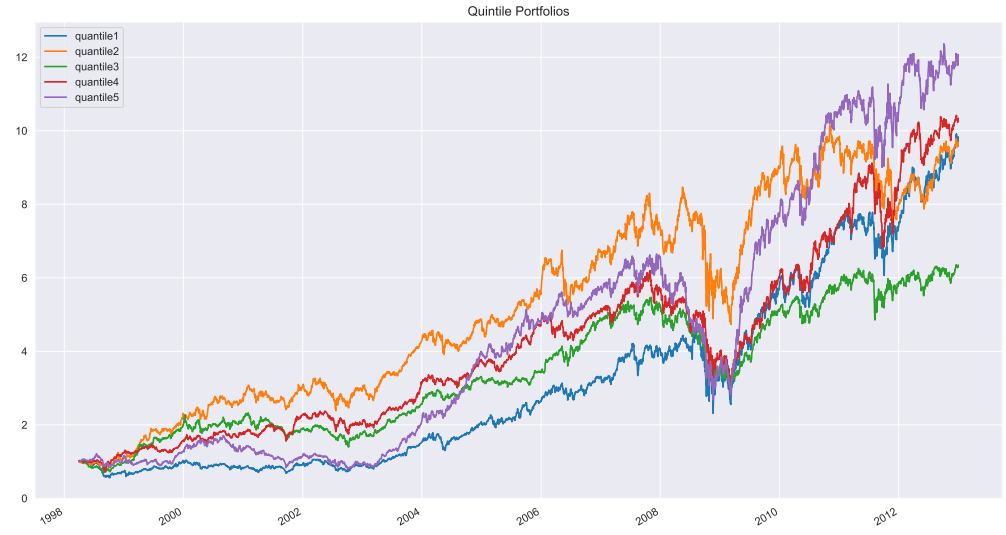
The performance of portfolios are as follows:

```
### Portfolio Returns
sorting = pd.read_csv("Sorting_return.csv", header = 0, index_col = 0)
quintile = pd.read_csv("Quintile_return.csv", header = 0, index_col = 0)
```

```
sorting.index = pd.DatetimeIndex(sorting.index)
sorting.add(1).cumprod().plot(figsize = (16,9), title = 'Sorting Portfolios')
```




```
quintile.index = pd.DatetimeIndex(quintile.index)
quintile.add(1).cumprod().plot(figsize = (16,9), title = 'Quintile Portfolios')
```



5.2.2 Determine the risk premium of CSD

In order to show the relationship between dispersion loadings and mean returns is robust to other aggregate factors that have been commonly found to explain the cross-section of stock returns, such as market return

MKT, the two additional (Fama and French 2002) factors SMB and HML, and the (Carhart 1997) momentum factor MOM. We first select entities that have survived from 1990 to 2019. The stock returns were adjusted by the adjusted factors in order to be used for time series analysis. We multiply each stocks' daily return with their adjusted factor with the following code.

```
price = pd.DataFrame(index = data_new.index)

for f in range(data_new.shape[1]):
  name = col_name[f]
  price[name] = data_new[name]*adj[name]
```

Then we only keep stocks that survive all the time. We drop stocks that contain N/A for more than 200 days.

Next, we generate the monthly return for each stock.

```
price_data['month'] = price_data['datadate'].astype(str).str[:6]
monthly_total = price_data.groupby('month')
first = price_data.groupby('month').first()
last = price_data.groupby('month').last()
month_return = last - first
month_return
month_return = month_return.drop(['datadate'], axis = 1)
month_return = month_return.reset_index()
month_return
```

Get monthly factors datas and combine with stock return.

```
raw_data = month_return
raw_data['month'] = raw_data['month'].astype(int)
df = month_return.merge(factors, how = 'left', left_on = 'month', right_on='date')
```

Doing regression for each stock.

```
import statsmodels.api as sm
X = df[my_list[900:904]]
X = sm.add_constant(X)
final = pd.DataFrame(index = ['const','Mkt-RF','SMB','HML','Mom'])
for g in range(1,899):
  Y = df[my_list[g]]
  result = sm.OLS(Y,X).fit()
  final[my_list[g]] = result.params.to_list()

name = list(final)
Diff = pd.DataFrame(index = df.index)
for g in range(final.shape[1]):
  con = final[name[g]][0]
  X1 = final[name[g]][1]
  X2 = final[name[g]][2]
  X3 = final[name[g]][3]
  X4 = final[name[g]][4]

  predict = factors['Mkt-RF']*X1 + factors['SMB']*X2 + factors['HML']*X3+ \
```

```
  factors['Mom   ']*X4  + con
  Diff[name[g]] = predict
```

After estimate the risk adjusted return, we examine the impact of stock characteristics(Chen and Petkova 2012). By using The stock characteristics such as CSD and the standard deviation, skewness(Harvey and Siddique 2000) and kurtosis of stock returns over the past six months.

```
name = list(KKK)
res = pd.DataFrame( index = ['cons', 'CSD','adjR2'])
for i in range(1):
  sk1 = list()
  std1 = list()
  kuro1 = list()
  for j in range(300):
    skvalue = original[name[i]]
    skvalue = skvalue[(66+j):(7.2+j)]
    sk = skvalue.skew()
    std = skvalue.std()
    kuro = skvalue.kurtosis()
    sk1.append(sk)
    std1.append(std1)
    kuro1.append(kuro)

  X['sk'] = sk1
  X['std'] = std1
  X['kuro'] = kuro1
  Y = KKK[name[i]].to_list()
  result = sm.OLS(Y,X).fit()
  res[name[i]] = result.params.to_list()
```

5.2.3 Test Risk Adjusted Returns

We run regression on the portfolio returns (1-5, N-P) with systematic factors, and use the significant of interseption terms to indentify the risk-adjusted returns of CSD.

```
### Fetch the monthly returns for mimicing portfolios
def monthly_return(x):
  return x.add(1).cumprod().iloc[-1]-1

one_five = one_five.to_frame(name = '1-5')
N_P = N_P.to_frame(name = 'N-P')

one_five['month'] = one_five.index.strftime('%Y-%m')
N_P['month'] = N_P.index.strftime('%Y-%m')

return_1_5 = one_five.groupby(['month']).apply(lambda x: monthly_return(x))
return_N_P = N_P.groupby(['month']).apply(lambda x: monthly_return(x))
```

```
return_1_5.index = return_1_5.index.str[:4] + return_1_5.index.str[-2:]
return_N_P.index = return_N_P.index.str[:4] + return_N_P.index.str[-2:]
```

```
### montly factors data
factors_monthly = pd.read_csv('/content/drive/My Drive/FIN554/Factors.csv', \
index_col = 0)
factors_monthly.index = factors_monthly.index.astype(str)
factors_monthly = factors_monthly.loc[return_1_5.index]

### access returns
return_1_5 = return_1_5.sub(factors_monthly.RF, axis = 0)
return_N_P = return_N_P.sub(factors_monthly.RF, axis = 0)
```

```
factors_monthly.rename(columns = {'Mom   ': 'Mom'}, inplace = True)
```

5.2.4 Test the robustness of models

1. formation window Formation window equals to 3 months:

```
### store the result in dictionary
def log_result_3month(result):
    beta_3month.append(result)

from multiprocessing import Pool
import os

pool = Pool(os.cpu_count())

# regression_data.fillna(method = 'ffill', inplace = True)
beta_3month = []

### 3month regression
months = regression_data.index.unique()
for i in tqdm(range(len(months)-2)):
  grouped_df = regression_data.loc[months[i]: months[i+2]]
  pool.apply_async(regression_stocks, args=(grouped_df,['Mkt-RF', 'CSD'], 'CSD'),\
  callback = log_result_3month)

pool.close()
pool.join()

beta_3month = pd.concat(beta_3month, axis=1).T

### save Beta of CSD factors
beta_3month.to_csv("/content/drive/My Drive/FIN554/CSD_Betas_3month.csv")

### read Beta of CSD factors
beta_3month = pd.read_csv("/content/drive/My Drive/FIN554/CSD_Betas_3month.csv",\
index_col = 0)
# beta_3month

### Reset the index of beta Dataframe into daily
beta_3month = beta_3month.shift()
beta_3month.index = pd.to_datetime(beta_3month.index)
beta_3month = beta_3month.resample('1d').mean()
```

```
beta_3month.pad(inplace = True)

### use data only after 1998-04-01
beta_3month = beta_3month.loc['19980401':]

### generate portfolio weights
No1, No2, No3, No4, No5, Neg, Pos, new_index = get_weights(beta_3month, return_daily.index)

N_P = long_short_return(Neg, Pos)
one_five = long_short_return(No1, No5)

X = sm.add_constant(factors.loc[N_P.index, ['Mkt-RF', 'SMB', 'HML', 'MOM']])

row_3month = [one_five.mean(), sm.OLS(one_five, X).fit().params['const'], N_P.mean(),\
sm.OLS(N_P, X).fit().params['const']]
row_3month = [round(x, 5) for x in row_3month]
row_3month
```

Formation window equals to 6 months:

```
### store the result in dictionary
def log_result_6month(result):
    beta_6month.append(result)

pool = Pool(os.cpu_count())

# regression_data.fillna(method = 'ffill', inplace = True)
beta_6month = []

### 6month regression
months = regression_data.index.unique()
for i in tqdm(range(len(months)-5)):
  grouped_df = regression_data.loc[months[i]: months[i+5]]
  pool.apply_async(regression_stocks, args=(grouped_df,['Mkt-RF', 'CSD'], 'CSD'),\
  callback = log_result_6month)

pool.close()
pool.join()

beta_6month = pd.concat(beta_6month, axis=1).T

### save Beta of CSD factors
beta_6month.to_csv("/content/drive/My Drive/FIN554/CSD_Betas_6month.csv")

### read Beta of CSD factors
beta_6month = pd.read_csv("/content/drive/My Drive/FIN554/CSD_Betas_6month.csv",\
index_col = 0)

### Reset the index of beta Dataframe into daily
beta_6month = beta_6month.shift()
beta_6month.index = pd.to_datetime(beta_6month.index)
beta_6month = beta_6month.resample('1d').mean()
beta_6month.pad(inplace = True)
```

```
### use data only after 1998-04-01
beta_6month = beta_6month.loc['19980401':]

### generate portfolio weights
No1, No2, No3, No4, No5, Neg, Pos, new_index = get_weights(beta_6month, return_daily.index)

N_P = long_short_return(Neg, Pos)
one_five = long_short_return(No1, No5)
X = sm.add_constant(factors.loc[N_P.index, ['Mkt-RF', 'SMB', 'HML', 'MOM']])

row_6month = [one_five.mean(), sm.OLS(one_five, X).fit().params['const'], \
N_P.mean(), sm.OLS(N_P, X).fit().params['const']]
row_6month = [round(x, 5) for x in row_6month]
row_6month
```

2. Sign of market Negative MKT:

```
mkt_neg = set(mkt.loc[mkt<0].index).intersection(set(one_five.index))

X = sm.add_constant(factors.loc[mkt_neg, ['Mkt-RF', 'SMB', 'HML', 'MOM']])

row_mkt_neg = [one_five.loc[mkt_neg].mean(), sm.OLS(one_five.loc[mkt_neg], X)\
.fit().params['const'], N_P.loc[mkt_neg].mean(), sm.OLS(N_P.loc[mkt_neg], X).fit().params['const']]
row_mkt_neg = [round(x, 5) for x in row_mkt_neg]
row_mkt_neg
```

Positive MKT:

```
mkt_pos = set(mkt.loc[mkt>0].index).intersection(set(one_five.index))

X = sm.add_constant(factors.loc[mkt_pos, ['Mkt-RF', 'SMB', 'HML', 'MOM']])

row_mkt_pos = [one_five.loc[mkt_pos].mean(), sm.OLS(one_five.loc[mkt_pos], X)\
.fit().params['const'], N_P.loc[mkt_pos].mean(), sm.OLS(N_P.loc[mkt_pos], X).fit().params['const']]
row_mkt_pos = [round(x, 5) for x in row_mkt_pos]
row_mkt_pos
```

Negative CSD:

```
csd_neg = set(csd.loc[csd<0].index).intersection(set(one_five.index))

X = sm.add_constant(factors.loc[csd_neg, ['Mkt-RF', 'SMB', 'HML', 'MOM']])

row_csd_neg = [one_five.loc[csd_neg].mean(), sm.OLS(one_five.loc[csd_neg], X)\
.fit().params['const'], N_P.loc[csd_neg].mean(), sm.OLS(N_P.loc[csd_neg], X).fit().params['const']]
row_csd_neg = [round(x, 5) for x in row_csd_neg]
row_csd_neg
```

Positive CSD:

```
csd_pos = set(csd.loc[csd>0].index).intersection(set(one_five.index))

X = sm.add_constant(factors.loc[csd_pos, ['Mkt-RF', 'SMB', 'HML', 'MOM']])

row_csd_pos = [one_five.loc[csd_pos].mean(), sm.OLS(one_five.loc[csd_pos], X)\
.fit().params['const'], N_P.loc[csd_pos].mean(), sm.OLS(N_P.loc[csd_pos], X).fit().params['const']]
row_csd_pos = [round(x, 5) for x in row_csd_pos]
row_csd_pos
```

```
from statsmodels.tsa.ar_model import AR

ar = AR(csd.dropna()).fit(1)
fit_data = ar.predict(start = csd.index[2], end = csd.index[-1])
csd_ar = csd.sub(fit_data)
csd_ar.name = 'CSD_AR'

mkt_copy = return_daily.factors['Mkt-RF'].copy(deep = True)
mkt_copy.index = pd.DatetimeIndex(mkt_copy.index.astype(str))

regression_ar = pd.concat([return_data, pd.concat([mkt_copy, csd_ar], axis = 1)],\
axis = 1, keys = ['stocks', 'factors'])

regression_ar.index = pd.DatetimeIndex(regression_ar.index).strftime('%Y-%m')
### store the result in dictionary
def log_result_ar(result):
    beta_ar.append(result)

pool = Pool(os.cpu_count())

# regression_data.fillna(method = 'ffill', inplace = True)
beta_ar = []

### regression
months = regression_ar.index.unique()
for i, grouped_df in regression_ar.groupby(level = 0):
  pool.apply_async(regression_stocks, args=(grouped_df,['Mkt-RF', 'CSD_AR'], 'CSD_AR'),\
  callback = log_result_ar)

pool.close()
pool.join()

beta_ar = pd.concat(beta_ar, axis=1).T

### save Beta of CSD factors
beta_ar.to_csv("/content/drive/My Drive/FIN554/CSD_Betas_AR1.csv")

beta_ar = pd.read_csv("/content/drive/My Drive/FIN554/CSD_Betas_AR1.csv", index_col = 0)
### Reset the index of beta Dataframe into daily
beta_ar = beta_ar.shift()
beta_ar.index = pd.to_datetime(beta_ar.index)
beta_ar = beta_ar.resample('1d').mean()
```

```
beta_ar.pad(inplace = True)

### use data only after 1998-04-01
beta_ar = beta_ar.loc['19980401':]

### generate portfolio weights
No1, No2, No3, No4, No5, Neg, Pos, new_index = get_weights(beta_ar, return_daily.index)

N_P = long_short_return(Neg, Pos)
one_five = long_short_return(No1, No5)

X = sm.add_constant(factors.loc[N_P.index, ['Mkt-RF', 'SMB', 'HML', 'MOM']])

row_ar = [one_five.mean(), sm.OLS(one_five, X).fit().params['const'], N_P.mean(),\
sm.OLS(N_P, X).fit().params['const']]
row_ar = [round(x, 5) for x in row_ar]
row_ar
```

3. CSD as AR(1) innovations CSD's performance under different settings and conditions:

Table 1: formation window

	mean	alpha	mean	alpha
3 months	-0.00017	-0.00022	-0.00004	-0.00006
6 months	-0.00012	-0.00016	-0.00022	-0.00024

Table 2: MKT Sign

	mean	alpha	mean	alpha
negative	-0.00032	0.00037	-0.00019	-0.00015
positive	0.00007	-0.00069	-0.00025	-0.00050

Table 3: CSD Sign

	mean	alpha	mean	alpha
negative	0.00038	0.00025	-0.00011	-0.00017
positive	-0.00066	-0.00062	-0.00035	-0.00033

Table 4: CSD Measure

	mean	alpha	mean	alpha
AR(1)	4e-05	2e-05	4e-05	4e-05

5.2.5 Price the Cross-Sectional Dispersion

We try to measure the price of CSD by showing that its price is immune of other cross-sectional risk measures(Brennan and Subrahmanyam 1998). Here we conduct the analysis of seven factors, MKT market

returns, $FVIX$ VIX volatilty index, $SVAR$ volatilty of returns of all stocks(arithmatic mean of squared returns), UNC the index of uncertainty, LIQ liquidity factor(Pastor and Stambaugh 2003), and $IDVOL$ aggregated idiosyncratic risk, which were obtained as the residual of regression for each stocks' return on MKT, SMB, HML and MOM factors. Here for the convinence of calculation, we use the factor value as the mimic portfolio returns of that factor.

```
# Read all additional data
### VIX
vix = pd.read_excel('/content/drive/My Drive/FIN554/VIX_daily.xlsx', sheet_name='Worksheet',\
index_col = 0).PX_LAST.diff()
vix.name = 'VIX'
### Macro Uncertainty Index
macro_risk = pd.read_csv('/content/drive/My Drive/FIN554/Economic Uncertainty Index.csv',\
index_col=0)

def format_index(data):
  if int(data.name[-2:])>=94:
    return '19' + data.name[-2:] + '-' +data.name[:3]
  else:
    return '20' + data.name[-2:] + '-' +data.name[:3]

macro_risk.index = pd.DatetimeIndex(macro_risk.apply(lambda x: format_index(x),\
axis =1).values)

### double sorting portfolios
def double_sorting(mkt_beta, target_beta, return_data, new_index, mv_df, factor_name):
  mkt_beta, target_beta = mkt_beta.align(target_beta, join = 'inner', axis = 0)
  portfolios = {}
  for i in range(5):
    No_mkt = ((mkt_beta.sub(mkt_beta.quantile(q=0.2*i, axis=1), axis = 0)>=0) & \
              (mkt_beta.sub(mkt_beta.quantile(q=0.2*(i+1), axis=1), axis = 0)<0)).astype(int)
    for j in tqdm(range(5), f'MKT quantile {i+1}'):
      No_target = ((target_beta.sub(target_beta.quantile(q=0.2*j, axis=1), axis = 0)>=0) & \
             (target_beta.sub(target_beta.quantile(q=0.2*(j+1), axis=1), axis = 0)<0) & No_mkt)\
             .astype(int)

      weights = No_target.loc[new_index].mul(mv_df.loc[new_index])
      weights = weights.div(weights.sum(axis = 1), axis = 0)
      returns = return_data.align(weights, join = 'right')[0].mul(weights).sum(axis=1)
      portfolios[f'factor_name{j+1}_mkt{i+1}'] = returns

  return pd.concat(portfolios, axis = 1)
```

MKT:

```
mkt = return_daily.factors['Mkt-RF'].copy(deep = True)
mkt.index = pd.DatetimeIndex(mkt.index.astype(str))

### construct regression data
regression_mkt = pd.concat([return_data.align(mkt, join = 'right', axis =0)[0],\
                            mkt], axis = 1, keys = ['stocks', 'factors'])
regression_mkt.index = pd.DatetimeIndex(regression_mkt.index.astype(str)).strftime("%Y-%m")

### store the result in dictionary
```

```
def log_result_mkt(result):
    beta_mkt.append(result)

pool = Pool(os.cpu_count())

# regression_data.fillna(method = 'ffill', inplace = True)
beta_mkt = []

### monthly regression
for idx, grouped_df in regression_mkt.groupby(level = 0):
  pool.apply_async(regression_stocks, args=(grouped_df,['Mkt-RF'], 'Mkt-RF'),\
  callback = log_result_mkt)

pool.close()
pool.join()

beta_mkt = pd.concat(beta_mkt, axis=1).T
### save Beta of CSD factors
beta_mkt.to_csv("/content/drive/My Drive/FIN554/MKT_Betas.csv")

beta_mkt = pd.read_csv("/content/drive/My Drive/FIN554/MKT_Betas.csv", index_col = 0)

beta_mkt = beta_mkt.shift()
beta_mkt.index = pd.to_datetime(beta_mkt.index)
beta_mkt = beta_mkt.resample('1d').mean()
beta_mkt.pad(inplace = True)
```

FCSD:

```
csd.index = pd.DatetimeIndex(csd.index.astype(str))

### Reset the index of beta Dataframe into daily
beta_df = beta_df.shift()
beta_df.index = pd.to_datetime(beta_df.index)
beta_df = beta_df.resample('1d').mean()
beta_df.pad(inplace = True)
### use data only after 1998-04-01
beta_df = beta_df.loc['19980401':]

fcsd = mimicing_portfolio(beta_df, return_data, mv_df, csd)
fcsd.name = 'FCSD'
fcsd.index = pd.DatetimeIndex(fcsd.index)
fcsd = fcsd.resample('1d').mean()
fcsd.pad(inplace = True)
```

VIX:

```
vix = vix.sort_index()
vix = vix.loc['1996-01-01': '2012-12-31']
```

SVAR:

```
svar = return_daily.stocks.pow(2).mean(axis = 1).diff()
svar.name = 'SVAR'
svar.index = pd.DatetimeIndex(svar.index.astype(str))
svar = svar.loc['1996-01-01': '2012-12-31']
svar = svar.pad()
```

UNC:

```
macro_risk = macro_risk.resample('1D').pad()
macro_risk.columns = ['UNC']
unc = macro_risk.UNC
unc = unc.loc['1996-01-01': '2012-12-31']
```

LIQ:

```
liq = pd.read_table('/content/drive/My Drive/FIN554/LIQ.txt', header = None, index_col = 0).iloc[:, 0]
liq.index = pd.to_datetime(liq.index.astype(str), format = "%Y%m")
liq.index.name = 'datatime'
liq.name = "LIQ"
liq = liq.resample('1D').mean()
liq.pad(inplace = True)
```

IDVOL:

```
### Daily factor data
factors = pd.read_csv("/content/drive/My Drive/FIN554/Factors_daily.csv", header = 0,\
index_col = 0)
factors.index = pd.DatetimeIndex(factors.index.astype(str))

### concatenate all factors
factors = pd.concat([factors, liq], axis = 1, join = 'inner')
factors.drop(columns = ['RF'], inplace = True)

### construct regression data
regression_idvol = pd.concat([return_data.align(factors, join = 'right', axis =0)[0],\
                          factors], axis = 1, keys = ['stocks', 'factors'])
regression_idvol.index = pd.DatetimeIndex(regression_idvol.index.astype(str))\
.strftime("%Y-%m")
regression_idvol = regression_idvol.loc['1998-04-01':]

### store the result in dictionary
def log_result_idvol(result):
    beta_idvol.append(result)

pool = Pool(os.cpu_count())

# regression_data.fillna(method = 'ffill', inplace = True)
beta_idvol = []

### monthly regression
for idx, grouped_df in regression_idvol.groupby(level = 0):
  pool.apply_async(regression_stocks, args=(grouped_df,['Mkt-RF', 'SMB', 'HML',\
```

```
  'MOM', 'LIQ'],
  'residual'), callback = log_result_idvol)

pool.close()
pool.join()

beta_idvol = pd.concat(beta_idvol, axis=0)
### save Beta of CSD factors
beta_idvol.to_csv("/content/drive/My Drive/FIN554/IDVOL_Betas.csv")

beta_idvol = pd.read_csv("/content/drive/My Drive/FIN554/IDVOL_Betas.csv",\
index_col = 0)
beta_idvol.index = pd.DatetimeIndex(beta_idvol.index.astype(str))
beta_idvol = beta_idvol.groupby(level = 0).std().mean(axis = 1)
idvol = beta_idvol.resample('1D').mean()
idvol.pad(inplace = True)
idvol.name = 'IDVOL'
```

In order to show that the risk premium of CSD is distinct from the other factors, we conduct double sorting to obtain 25 quintile portfolios at the beginning of each month.

```
double_sorted_csd = double_sorting(beta_mkt, beta_df, return_data, new_index, mv_df,\
'CSD')

### calculate the access returns
RF = return_daily.factors.RF.copy(deep = True)
RF.index = pd.DatetimeIndex(RF.index.astype(str))
RF = RF.align(double_sorted_csd, join = 'right', axis = 0)[0]
double_sorted_csd = double_sorted_csd.sub(RF, axis = 0)
```

Then we run the Fama-MacBeth two pass regression to test the significant of those factors' effect on CSD returns:

```
### Fama-MacBeth two pass regression
def two_pass_regression(reg_data, portfolios):
  ### first pass: time-series regression
  time_series_beta = []
  for portfolio in portfolios.columns:
    Y = portfolios.loc[:, portfolio]
    X = reg_data.align(Y, join = 'right', axis = 0)[0]
    X = sm.add_constant(X)
    model = sm.OLS(Y, X).fit()
    time_series_beta.append(pd.Series(model.params, name = portfolio))

  time_series_beta = pd.concat(time_series_beta, axis = 1).T
  # print(portfolios.mean(), time_series_beta)
  ### second pass: cross-sectional regression
  X_new = time_series_beta
  Y_new = portfolios.mean()
  results = sm.OLS(Y_new, X_new).fit()
  return results
```

5.3 Hypothesis Tests

Test1. CSD has negative correlation with expected returns

From the return of the sorting portfolo, we can conclude that CSD has a slight negative trend. However, from the t-test statistics of long-short portfolios 1-5 and N-P, the correlation is in fact not negative, and not significant. The difference might came from the different usage of data. In our analysis we use only the stocks that still are listing today, which intrduced survivor ship bias, and affect the significance. What's more, due to the missing data of market value in the late 90s, some of the stocks with significant negative exposure to CSD might not participate in the regression (we follow the rules which only use those stocks that has more than 15 observation in that month)

	Mean	Std	Pre-formation beta	Post-formation beta
1	0.0008	0.0170	-1.9174	0.1965
2	0.0006	0.0143	-0.6359	0.1461
3	0.0006	0.0132	0.0205	0.2282
4	0.0006	0.0134	0.6794	0.1896
5	0.0007	0.0161	1.9164	0.2154
1-5	-2.810391753545679e-05(-0.11)			
N-P	-0.0001353690564594488(-0.83)			

We used the monthly returns of sorting portfolios 1-5 and N-P to do the regression on traditional Fama-French three factors MKT, SMB, HML as well as Carhart's momentum factor MOM. From the regression results, we obtained a contradictory conclusion from Test1 which indicated that CSD has a significant negative risk-adjusted returns. The reason behind this might be that we use daily returns on Test1 but monthly returns on Test2, because monthly returns are more representative for factors like SML and HML, which might reactly slowly to the market change, so we conclude that in long term investing, CSD can generate a significant risk-premium which are distinct from other established systematic risk premia.

```
table2 = read.csv("Table2.csv")[,3:4]
rownames(table2) = c("constant", "MKT", "SMB", "HML", "MOM", "Adj.R^2")
knitr::kable(table2, "pipe", col.names = c('1-5', 'N-P'))
```

	1-5	N-P
constant	-0.1508(-14.56)	-0.1505(-15.33)
MKT	0.0033(1.35)	0.003(1.27)
SMB	0.0006(0.19)	0.0016(0.51)
HML	-0.0042(-1.26)	-0.0038(-1.23)
MOM	-0.0022(-1.04)	-0.0026(-1.29)
Adj.R^2	0.027	0.034

Test3. CSD can be price cross-sectionally

In order to prove that CSD can be priced cross-sectionally. We use Fama-MacBeth two-pass regression to test the returns of double-sorted portfolios (based on MKT and $FCSD$) on different measurement of cross-secional risk. From the table below we can see that all factors are not significant in the regression, indicating that CSD can be priced cross-sectionally, and is a distinct measurement of cross sectional risk.

	regression-1	regression-2	regression-3	regression-4	regression-5
constant	1.0(92.45)	1.0(47.47)	1.0(5.12)	1.0(93.15)	1.0(51.29)
MKT	0.0281(76.6)	0.0281(53.5)	0.0281(43.51)	0.0281(62.51)	0.0278(88.67)
SMB	0.006(70.17)	0.006(34.0)	0.006(64.44)	0.006(41.9)	0.0101(13.87)
HML	0.0046(40.51)	0.0046(6.68)	0.0046(19.3)	0.0046(42.44)	0.0064(18.14)
MOM	0.0204(82.14)	0.0204(34.63)	0.0204(30.71)	0.0204(48.73)	0.0166(0.98)
FCSD	-0.0236(0.95)	-0.0236(10.7)	-0.0236(75.62)	-0.0236(14.88)	-0.0224(1.54)
FLIQ	0.0001(38.9)	0.0001(18.75)	0.0001(38.02)	0.0001(91.97)	0.0001(89.06)
FVIX		0.002(64.21)	0.002(89.53)	0.002(47.34)	0.0026(28.44)
FSVAR			-0.0(82.33)	-0.0(38.35)	-0.0(72.84)
FUNC				-0.1082(66.82)	-0.0717(42.46)
FIDVOL					0.0347(47.2)
Adj.R^2	1.0	1.0	1.0	1.0	1.0

Compare with Original Paper

By extending the testing period, our results are different from the original paper. This indicate that the stability of CSD is not as strong as it's stated in the original paper

- Overlap
  - Using the return of sorting portfolios, our paper showed that the risk premium attained from CSD is in fact independent from other established systematic risk factors (all factors' t-test are significant under 5% confidence level, with a significant interception term)
  - The CSD is indeed a good measure of aggregated idiosyncratic risk of individual stocks, when conducting Fama-MacBeth two-pass regression with other measurements, it is significant that CSD is comprised with different idiosyncratic risk.(With Adjusted R-squared closed to 1.0)
- Difference
  - Our experiment in section 5.2.1 showed that the sorting portfolio returns does not have a very strong increasing order with smaller CSD beta. We also showed that the negative relationship between CSD and expect returns is rejected in the t-test, which differs from the results in original paper.

6 Future Work

Currently the team only analyzed the cross-sectional dispersion with Standard Poor 500 Composite Index data as our market return, while the original research paper used the CRSP value-weighted index as a proxy for aggregated market returns. It is possible that a different index or calculation for the market return could have delivered a different CSD factor, and thus a different regression result. Another important evaluation that needs to be implemented is the stock selection process in this replication. Currently the team only utilized stocks that were consistently traded. All delisted stocks were eliminated from our factor calculation. This definitely is a biased data handling approach and should be further scrutinized. It is almost certain that there is no way to predict if a stock will be delisted from the stock market or not, therefore it is inaccurate to eliminate all delisted stocks from the beginning, while conducting this data analysis.

However, it certainly opens up new opportunities for the team to look at the calculation of the Cross-Sectional Dispersion factor calculation. From the original formula:

$$CSD_t = \frac{\sum_{i=1}^{N} |r_{i,t} - r_{mkt,t}|}{N-1}$$

It is shown that the CSD is an equally weighted factor, but since we mentioned that eliminating delisted stocks brings a different CSD value, it is easy to imply that CSD could be expressed as a weighted factor in such a way that:

$$CSD_t^{'} = \frac{\sum_{i=1}^{N} \lambda_{i,t}^{*} |r_{i,t} - r_{mkt,t}|}{N-1}$$

Where i,t represents a weight, regarding different stocks. This weight can be informally calculated by the market cap proportion of the corresponding stock, or can be further optimized using a reinforcement learning approach such as Q-learning to determine the optimal value. This approach would make CSD a more refined factor that can be deliberately tuned.

Eliminate correlation Another improvement that can be examined is the correlation between a certain stock return and the market return. By eliminating the correlation of returns we might be able to calculate an unbiased CSD that represents the neutral fluctuation and volatility of the market. In this case we have

$$CSD_t^{''} = \frac{\sum_{i=1}^{N} \lambda_{i,t}^{*} |\rho_{i,t}^{*} r_{i,t} - r_{mkt,t}|}{N-1}$$

Where i,t represents the correlation between stock i and market return, which can be calculated by a linear regression of stock returns.

7 Conclusions

In this paper, we replicated the research of Cross-Sectional Dispersion and Expected Returns by Thanos Verousis and Nikolaos Voukelatos in 2016. We used data from January 1996 to December 2012 to test the relationship between Cross-sectional Dispersion and the expected stock returns. First, we computed the ΔCSD factor, regression on all the stocks to obtain a mimicking portfolio. We then show that the mimicking portfolio had mild negative returns, and further proved that CSD has negative relationship with stocks' expected returns. In order to show that CSD is a new factor, we test the risk premium of CSD with the risk premia of some established systematic risk factors, the results indicated that CSD is a distinct risk factor. And it has similar performance under different market conditions, demonstrating the robustness of this factor. We finally use different measurement of cross-sectional risk to prove that CSD is a priced factor, and can be a distinct measures of aggregated idiosyncratic risk of individual stocks cross-sectionally.

We denote that the data we use is slightly different from the original paper, due to the missing data. So the results were less significant than the original paper. We suggest further analysis on more recent period. Because of the pandemic, the fundamental of market has experienced some changes, the factors that were once significant might generate less returns compared to the past. We also discovered that the alternative calculation methodology for CSD may generate different results, so further study on the robustness with calculation method is strongly suggested.

Appendix

Main Python codes.

```
# Reference: https://www.statsmodels.org/dev/generated/statsmodels.regression.linear_model.OLS.html
def regression_stocks(stock_prices, factors, beta):
  '''
  stock_prices: dataframe of stocks and factors returns
```

```
  factors: a list of factors to do the regression
  '''
  betas = []
  for stock in tqdm(stock_prices.stocks.columns, f"regression on {stock_prices.index[-1]}"):
    X_Y = pd.concat([stock_prices.factors.loc[:, factors], stock_prices.stocks\
    .loc[:, stock]],
    axis = 1)
    X_Y = X_Y.dropna()
    if X_Y.shape[0]<15:
      if beta == 'residual':
        betas.append(pd.Series(np.nan, index = stock_prices.index))
      else:
        betas.append(None)
      continue
    X = X_Y.loc[:, factors]
    Y = X_Y.loc[:, stock]
    X = sm.add_constant(X)
    model = sm.OLS(Y,X)
    results = model.fit()
    if beta == 'residual':
      betas.append(Y.sub(results.predict(X)))
    else:
      betas.append(results.params[beta])
  if beta == 'residual':
    return_df = pd.concat(betas, axis =1)
    return_df.columns = stock_prices.stocks.columns
    return return_df
  else:
    return pd.Series(betas, index = stock_prices.stocks.columns, name =\
  stock_prices.index[-1])


def get_weights(beta_df, ref_index, mv_df):
  ### Align the Index
  new_index = sorted(list(set(beta_df.index).intersection(set(pd.DatetimeIndex
  (ref_index.astype(str))))))
  ### Quantile1-Quantile5
  No5=(beta_df.sub(beta_df.quantile(q=0.8, axis=1), axis = 0)>=0).loc[new_index]\
  .mul(mv_df.loc[new_index])
  No5 = No5.fillna(0)
  No5 = No5.div(No5.sum(axis = 1), axis = 0)

  No4=((beta_df.sub(beta_df.quantile(q=0.8, axis=1), axis = 0)<=0) & \
  (beta_df.sub(beta_df.quantile(q=0.6, axis=1), axis = 0)>0)).loc[new_index]\
  .mul(mv_df.loc[new_index])
  No4 = No4.fillna(0)
  No4 = No4.div(No4.sum(axis = 1), axis = 0)

  No3=((beta_df.sub(beta_df.quantile(q=0.6, axis=1), axis = 0)<=0) & \
  (beta_df.sub(beta_df.quantile(q=0.4, axis=1), axis = 0)>0)).loc[new_index]\
  .mul(mv_df.loc[new_index])
  No3 = No3.fillna(0)
  No3 = No3.div(No3.sum(axis = 1), axis = 0)
```

```
  No2=((beta_df.sub(beta_df.quantile(q=0.4, axis=1), axis = 0)<=0) & \
  (beta_df.sub(beta_df.quantile(q=0.2, axis=1), axis = 0)>0)).loc[new_index]\
  .mul(mv_df.loc[new_index])
  No2 = No2.fillna(0)
  No2 = No2.div(No2.sum(axis = 1), axis = 0)

  No1=(beta_df.sub(beta_df.quantile(q=0.2, axis=1), axis = 0)<=0).loc[new_index]\
  .mul(mv_df.loc[new_index])
  No1 = No1.fillna(0)
  No1 = No1.div(No1.sum(axis = 1), axis = 0)
  ### Negative-Positive
  Neg=(beta_df < 0).loc[new_index].mul(mv_df.loc[new_index])
  Neg = Neg.fillna(0)
  Neg = Neg.div(Neg.sum(axis = 1), axis = 0)
  Pos=(beta_df > 0).loc[new_index].mul(mv_df.loc[new_index])
  Pos = Pos.fillna(0)
  Pos = Pos.div(Pos.sum(axis = 1), axis = 0)

  return No1, No2, No3, No4, No5, Neg, Pos, new_index
```

```
### close price data
close_df = pd.concat([pd.read_hdf("/content/drive/My Drive/FIN554/ClosePrice.h5",\
key = '1990-1999').prccd,\
                      pd.read_hdf("/content/drive/My Drive/FIN554/ClosePrice.h5",\
                      key = '2000-2009').prccd,\
                      pd.read_hdf("/content/drive/My Drive/FIN554/ClosePrice.h5",\
                      key = '2010-2019').prccd],\
                      axis = 0, join = 'inner')
### select data after 1995-12-31
close_df = close_df.loc[close_df.index > 19951231 ]
### delete the date and stocks that are all empty
close_df.dropna(how = 'all', inplace = True)
close_df.dropna(how = 'all', axis = 1, inplace = True)

### adjust factors data
adjf_df = pd.concat([pd.read_hdf("/content/drive/My Drive/FIN554/AdjustFactor.h5",\
key = '1990-1999').ajexdi,\
                     pd.read_hdf("/content/drive/My Drive/FIN554/AdjustFactor.h5",\
                     key = '2000-2009').ajexdi,\
                     pd.read_hdf("/content/drive/My Drive/FIN554/AdjustFactor.h5",\
                     key = '2010-2019').ajexdi],\
                     axis = 0, join = 'inner')
### select data after 1995-12-31
adjf_df = adjf_df.align(close_df, join = 'right')[0]

### trfd factors data
trfd_df = pd.concat([pd.read_hdf("/content/drive/My Drive/FIN554/TrfdFactor.h5",\
key = '1990-1999').trfd,\
                     pd.read_hdf("/content/drive/My Drive/FIN554/TrfdFactor.h5",\
                     key = '2000-2009').trfd,\
                     pd.read_hdf("/content/drive/My Drive/FIN554/TrfdFactor.h5",\
                     key = '2010-2019').trfd],\
                     axis = 0, join = 'inner')
```

```
### select data after 1995-12-31
trfd_df = trfd_df.align(adjf_df, join = 'right')[0]

### calculate the adjusted close returns
adj_price = close_df.div(adjf_df)
adj_return = adj_price.mul(trfd_df).pct_change()
### replace infinity with previous return
adj_return.replace(np.inf, np.nan, inplace = True)
adj_return.pad(inplace = True)

### Daily MKT and RF factor data
mkt_rf_daily = pd.read_csv("/content/drive/My Drive/FIN554/Factors_daily.csv", \
header = 0, index_col = 0).loc[:, ["Mkt-RF", "RF"]].div(100)

### Calculate the CSD Factors
num_stocks = adj_return.shape[1]
### calculate the first order difference of CSD as our main variable
csd = adj_return.sub(mkt_rf_daily.loc[adj_return.index, "Mkt-RF"], axis= 0).abs()\
.sum(axis = 1).div(num_stocks-1).diff()
mkt_rf_daily['CSD'] = csd

### calculate the access returns
return_daily = pd.concat([adj_return, mkt_rf_daily], axis = 1, join = 'inner', \
keys = ['stocks', 'factors'])
return_daily.stocks = return_daily.stocks.sub(return_daily.factors.RF, axis = 0)
```

```
### Market Share as Portfolio Weight
# *** Before 1998-04-01 only 3 or 4 not N/A per month ***
### Market Share data
mv_df = pd.concat([pd.read_hdf("/content/drive/My Drive/FIN554/ShareOutstanding.h5",\
key = '1990-1999').cshoc,\
                   pd.read_hdf("/content/drive/My Drive/FIN554/ShareOutstanding.h5",\
                   key = '2000-2009').cshoc,\
                   pd.read_hdf("/content/drive/My Drive/FIN554/ShareOutstanding.h5",\
                   key = '2010-2019').cshoc],\
                   axis = 0, join = 'inner')
### select data after 1995-12-31
mv_df = mv_df.loc[mv_df.index > 19951231 ]

mv_df = mv_df.align(return_daily.stocks, join = 'right')[0].mul(close_df)
mv_df.index = pd.DatetimeIndex(mv_df.index.astype(str))
```

```
### group by month to generate regression data
regression_data = return_daily.copy(deep = True)
regression_data.index = pd.DatetimeIndex(return_daily.index.astype(str)).strftime("%Y-%m")
```

```
# Multithreading to speed up
# https://stackoverflow.com/questions/8533318/multiprocessing-pool-
# when-to-use-apply-apply-async-or-map

### store the result in dictionary
def log_result(result):
    beta_df.append(result)
```

```
pool = Pool(os.cpu_count())

# regression_data.fillna(method = 'ffill', inplace = True)
beta_df = []

### monthly regression
for idx, grouped_df in regression_data.groupby(level = 0):
  pool.apply_async(regression_stocks, args=(grouped_df,['Mkt-RF', 'CSD'], 'CSD'),\
  callback = log_result)

pool.close()
pool.join()

beta_df = pd.concat(beta_df, axis=1).T
```

Reference

Amit Goyal, Pedro Santa-Clara. 2003. "Idiosyncratic Risk Matters!" *Journal of Finance* 98: 975–1008.

Ang, Hodrick, A., and X. Zhang. 2006. "The Cross-Section Ofvolatility and Expected Returns." *Journal of Finance* 61: 259–99.

Angelidis, Sakkas, T., and N. Tessaromatis. 2015. "Stock Market Dispersion, the Business Cycle and Expected Factor Returns." *Journal of Banking and Finance* 59: 265–79.

Bali, Brown, T., and Y. Tang. 2015. "Macroeconomic Uncertainty and Expected Stock Returns."

Bali, Turan. n.d. https://sites.google.com/a/georgetown.edu/turan-bali/data-working-papers.

Brennan, Chordia, M., and A. Subrahmanyam. 1998. "Alternative Factor Specifications, Security Characteristics, and the Cross-Section of Expected Returns." *The Journal of Financial Economic* 49 (4): 345–73.

Carhart, M. 1997. "On Persistence in Mutual Fund Performance." *Journal of Finance* 52: 57–82.

Chen, Z., and R. Petkova. 2012. "Does Idiosyncratic Risk Proxy for Risk Exposure?" *Review of Financial Studies* 25: 2745–87.

CRSP. n.d. https://www.crsp.org/.

Diether, C., K. Malloy, and A. Scherbina. 2002. "Differences of Opinion and the Cross-Section of Stock Returns." *Journal of Finance* 57: 2113–41.

Fama, E., and K. French. 2002. "Differences of Opinion and the Cross-Section of Stock Returns." *Journal of Finance* 33: 3–56.

Fama, E., and J. Macbeth. 1973. "Risk, Return, and Equilibrium: Empiricaltests." *Journal of Political Economy* 71: 607–36.

French, Kenneth. n.d. https://mba.tuck.dartmouth.edu/pages/faculty/ken.french/data_library.html.

Garcia, Mantilla-Garcia, R., and L. Martellini. 2014. "A Model-Free Measure of Aggregate Idiosyncratic Volatility and the Prediction of Market Returns." *Journal of Financial and Quantitative Analysis*. https://papers.ssrn.com/sol3/papers.cfm?abstract_id=1973471.

Harvey, C., and A Siddique. 2000. "Conditional Skewness in Asset Pricing Tests." *Journal of Finance* 55 (4): 1263–95.

Pastor, L., and R. Stambaugh. 2003. "Liquidity Risk and Expected Stock Returns." *Liquidity Risk and Expected Stock Returns* 111: 642–85.

Verousis, Thanos, and Nikolaos Voukelatos. 2015. "Cross-Sectional Dispersion and Expected Returns."