Global Economic Indicators Between 1987-2021 For the Top 10 Highest GDP Countries in 2021

Phase 2 - Report

Syed Hassan - shassa25@illinois.edu
Gordon Ochi - gochi2@illinois.edu
Noah Rogers - noahrr2@illinois.edu

1. **Description of Actual Data Cleaning Workflow**

The data cleaning proceeded mostly as according to our Phase 1 plan, with a few exceptions found for when certain tools were more appropriate for a given task than what we had originally envisioned. A high-level overview of the data cleaning, as pertaining to the Tasks 1-4 outlined in our Phase 1 plan, is given in the next few paragraphs.

For Task 1, the primary goal was to permute the axes of the data such that rows were by time indice and secondarily by country, and the macroeconomic statistics were on the columns. This would make the dataset much easier to work with in OpenRefine in Tasks 2.1 and 2.2. Permuting of the axes was accomplished through a python script utilizing the Pandas package. In this step we also dropped two unnecessary columns, 'Country Code' and 'Series Code', and did some basic checking and filling for NaN values. The dataset was also split into a monthly dataset, and quarterly and yearly dataset, to help alleviate some of the empty space issues found when certain statistics were only logged monthly/quarterly. The outputs of this function were two new csv's, macroecon_qy_df.csv for the quarterly and yearly data, and macroecon_monthly_df.csv for the monthly data. See the file CS513_GroupProject_Task1.pdf in the supplementary material for a YesWorkflow diagram of this process.

Task 2 had two subtasks, 2.1 for the quarterly and yearly data, and 2.2 for the monthly data. These were targeted for OpenRefine, although we ultimately found that the correlation-type metrics were best done with python and Pandas. The process began with opening the csvs in OpenRefine, where the data for the various statistics were converted to number format. We then went through each macroeconomic statistic column and checked the distributions with facets, and in certain cases discussed dropping certain columns based on either redundancy or application to our use case U_1.

We removed the columns "GDP, current LCU, millions, seas. Adj", "Stock Markets, LCU", "GDP, constant 2010 LCU, millions, seas. Adj.", as the LCU acronym specifies Local Currency Units, and we decided our use case was primarily US dollar denominated. The column, "GDP, constant 2010 US$, millions, seas. Adj", was removed as we were not particularly interested in the GDP in 2010 US dollars, and this was a bit redundant with the GDP in current US$ column. Due to redundancies the columns, "CPI Price, % y-o-y, median weighted, seas. Adj", "Exports Merchandise, Constant US$, seas. Adj.", "Exports Merchandise, Constant US$, not seas. Adj.", "Imports Merchandise, Constant US$, millions, not seas. Adj.", "Imports Merchandise, Constant US$, millions, seas. Adj.", were removed. We also found that the import/export columns with 'Price' in the name had to do more with taxes levied on the products than the actual volume of transactions, and did not necessarily fit our use case. Columns, "Imports Merchandise, Customs, Price, US$, not seas. Adj.", "Imports Merchandise, Customs, Price, US$, Seas. Adj.", "Exports Merchandise, Customs, Price, US$, not seas. Adj.", "Exports Merchandise, Customs, Price, US$, seas. Adj.", were thus removed. The column, "Total Reserves", was also removed as we thought it did not do much for our use case.

After removing redundant and/or unnecessary columns, we set about addressing some of our desired queries and creating some new columns based on the combination of existing columns. First we created a seasonally adjusted Import-Export ratio, which is defined as the import value minus the export value, divided by the Gross Domestic

Product (GDP). The same procedure was done for the Export-Import ratio, which is defined as the export value minus the import value divided by the GDP. These computations return a value between -1 and 1, which inform the trade balance of a nation.

The next column added was a metric for the Industrial Production as a percent of GDP, which was done by adding a new column based on the Industrial production column and dividing by the GDP. This was only done for seasonally adjusted Industrial Production, as we only had seasonally adjusted GDP.

Next up we did some comparisons with the CPI data, where CPI stands for Consumer Price Index and is a measure of inflation. Here we first made a new column by dividing the seasonal CPI by the not seasonal CPI, to see the relation of the seasonal adjustments. This was repeated for core CPI, which is a similar inflation metric (although it computes the inflation on a restricted number of categories). To answer one of our queries regarding the Federal Monetary Policy preference of the core CPI metric, we also computed new seasonally adjusted and not seasonally adjusted columns for comparing the Core CPI to the standard CPI. These were defined as Core CPI divided by CPI; seasonally adjusted or not seasonally adjusted.

Lastly in the OpenRefine portion of Task 2.1, we renamed two columns. The column, “Stock Markets, US$,,,,”, was renamed to, “Stock Markets, US$, Billions”, as we thought this was better notation. The column, “Exchange rate, new LCU per USD extended backward, period average”, was renamed to, “Exchange rate, new LCU per USD, extended backward, period average (USD to EURO)”, which we thought helped clarify that this is the conversion to Euros. There is another column, “Exchange, rate, old LCU per USD, extended backward, period average”, which pertains to the various EU countries’ currency units before they switched over to the Euro.

At the end of the OpenRefine portion of Task 2.1, we realized that our correlation-type queries were better done with a different tool. Here we opted to use a quick Pandas script to compute normalized relations among a few columns. Our work in OpenRefine was exported to an intermediary csv file, which was subsequently imported into our python/pandas script. We first did a quick check and fill for NaN values. Then we computed the normalized GDP to normalized unemployment rate, where each of GDP and unemployment rate are normalized 0.0 to 1.0 before dividing the two: normalized GDP divided by normalized unemployment rate. We note that this is not necessarily the type of metric we first had in mind for the correlations, but due to the need to have a similar input and output format, it was decided that this would still show the overall relationship we were interested in. This process was repeated for the retail sales volume to CPI, and for the stock market value to CPI. The final product of this script was saved as Task2.1_quarterly_and_yearly_data_postPandas.csv. See our YesWorkflow diagram CS513_GroupProject_Task2.1.pdf in the supplementary material for a visualization.

Task 2.2 proceeded quite similarly to Task 2.1, and with some contingencies for the statistics that were only monthly. Many of the data cleaning aspects in OpenRefine proceeded the same way (as noted in the summary of data changes section), however pandas helper code needed to be adjusted for handling monthly updates. Data changes in general needed to be adjusted just to display the monthly data in a usable manner for Task 3.

Task 3 regarded computing some combination statistics that looked at relations among countries. Our approach differed from our original plan in a few ways. First, we came to the realization that Pandas would be a better tool for looking at relations among the variables than our originally targeted Datalog or SQLite. Secondly, we decided to add columns to the two existing datasets, rather than making a third dataset of fairly limited data. Another small change is that we made the metrics a bit more focused on the United States, as many metrics are U.S. dollar denominated, and we ultimately decided our use case was focused more on the United States and its relations with other top GDP countries.

In Task 3, there were three smaller tasks that were completed. We used the two datasets from Tasks 2.1 and 2.2 to compare the following information from all other countries to the USA: Import-Export Ratio, Unemployment Rate, and Stock Market Total Value. For the Import-Export Ratio, we divided the rate of all other countries' by USA's Import/Export value. We derived relations in regards to Unemployment rate by dividing all other countries' value by USA's Unemployment rate. Finally, we calculated the Stock Market Value relations by dividing the value from all other countries by the value from USA's Stock Market Total value for a given year. Some minor cleanup on the year column and Pandas-generated duplicate index was also performed at the end of the script. YesWorkflow was then used to generate a pdf that visually describes this process. See the supplementary material for a diagram of the Task 3 Pandas/Python workflow.

In Task 4 we completed the various YesWorkflow diagrams. For each component that using OpenRefine, we utilized the or2yw tool to convert the json log files into YesWorkflow diagrams. See the supplementary material for these, although we note that the or2yw generated diagram output was perhaps less than ideal. For all the pandas/python portions, we marked our scripts with the standard @func nomenclature of YesWorkflow, and utilized YesWorkflow and graphviz to generate flow charts of our processes. These are also included in the supplementary material. OpenRefine json files and python scripts are also included in the supplementary material.

2. Narrative and Motivation

Our main use case, U_1, was an in-depth exploratory analysis of the macroeconomics dataset, relating countries and various statistics, and perhaps lending itself to predictive analysis for future data. In proceeding with the workflow *W* documented herein, it was also decided that it would focus primarily on the United States, partially for conciseness and partially because most macroeconomic quantities are U.S. dollar denominated anyway. We chose a combination of Python/Pandas and OpenRefine to accomplish our task. Python and Pandas were useful for the axes permutations necessary at the start, and handy for some of the more intricate calculations we needed to do for relating variables and countries. OpenRefine was useful for checking data consistency, and some basic combinations of columns. The numeric facets provide a quick method of seeing if values in columns are realistic. For visualization of the data cleaning procedures, YesWorkflow and an OpenRefine to YesWorkflow tool, or2yw, were used.

3. Documentation of Data Quality Improvement

Our improvements largely have to do with the addition of new columns/statistics, and the splitting of data into quarterly and yearly statistics, and monthly statistics. We found no obvious outliers through inspection of the numeric facets in OpenRefine (although some statistics that appeared as outliers at first were actually valid values - looking at you Italy with 1000%+ year-over-year inflation!). Empty values at time indices, related to certain statistics being only monthly/quarterly/yearly, were primarily addressed through splitting the datasets as previously specified. That being said, data collection for some older dates was still inconsistent, and some empty values remain in our finished dataset. This is a minor impediment to our use case though, as the empty values can be easily handled by queries.

a. **Monthly, Quarterly and Yearly, datasets split.**

The usefulness of splitting the datasets can be seen from this example query looking at the GDP statistic for the United States in 1987. At first it is listed as a row, with many ".." in the place of empty data, since the data is not listed monthly. See the query just below.

Country	Country C	Series	Series Coc	1987 [1987]	1987M01 [	1987M02 [	1987M03 [	1987M04 [	1987M05 [	1987M06 [	1987M07 [	1987M08 [	1987M09 [	1987M10 [	1987M11 [	1987M12 [	1987
United Sta	USA	Core CPI,not seas.adj,,,	CORENS	53.70628	52.6158	52.75211	53.07016	53.34278	53.43366	53.47909	53.6154	53.88803	54.25152	54.56958	54.75132	54.70589	..
United Sta	USA	CPI Price, % y-o-y, media	CPTOTSAX	..	..	..	..	..	..	..	..	..	..	..	..	..	..
United Sta	USA	CPI Price, seas. adj.,,,	CPTOTSAX	52.24043	51.22122	51.40514	51.58905	51.81895	51.95689	52.18679	52.32473	52.55462	52.73854	52.87648	53.0604	53.15236	..
United Sta	USA	CPI Price, % y-o-y, not se	CPTOTSAX	3.578212	1.364877	1.914312	2.84143	3.679853	3.669725	3.747715	3.926941	4.288321	4.272727	4.355717	4.528986	4.33213	..
United Sta	USA	Core CPI,seas.adj,,,	CORESA	53.5618	52.53067	52.66665	52.84794	53.16521	53.34651	53.43716	53.61845	53.79975	54.02637	54.29831	54.43429	54.57026	..
United Sta	USA	CPI Price,not seas.adj,,,	CPTOTNSX	52.43739	51.31826	51.50286	51.73361	52.0105	52.1951	52.3797	52.51815	52.79505	53.07194	53.21039	53.25654	53.25654	..
United Sta	USA	GDP,current LCU,millions	NYGDPMK	4855400	..	..	..	..	..	..	..	..	..	..	..	..	11
United Sta	USA	GDP,current US$,millions	NYGDPMK	4855400	..	..	..	..	..	..	..	..	..	..	..	..	11

Splitting the data quarterly and yearly, along with the axes permutations, allows for an easier view of the various statistics. This is shown for the quarterly and yearly data below. Note also that we can more quickly compare amongst the countries of interest for each macroeconomic variable.

Time	Country	Core CPI,r	Core CPI /	CPI Price,	CPI seasor	CPI Price,	Core CPI,s	Core CPI /	Core CPI S	CPI Price,r	GDP,curre
1987 [1987]	Brazil										
1987 [1987]	Canada	61.15591	1.027973	59.12575	0.993848	4.357171				59.49175	433144.1
1987 [1987]	China			28.45207	1.005377	7.24936				28.29991	
1987 [1987]	France			65.25497	0.995484	3.304534				65.55098	999232.4
1987 [1987]	Germany			63.65516	0.997657	0.183928				63.80462	1286346
1987 [1987]	India			18.6945	0.997629	8.794978				18.73893	
1987 [1987]	Italy			47.4444	0.998525	4.701212				47.51446	642039.9
1987 [1987]	Japan	89.48473	1.005532	88.71754	0.996911	0.124835				88.99241	2538700
1987 [1987]	United Kingdom			53.94384	0.994875	4.144622				54.22174	813924.6
1987 [1987]	United Sta	53.70628	1.024198	52.24043	0.996244	3.578212	53.5618	1.025294	0.99731	52.43739	4855400

Similar can be said for the monthly data, as far as axes permutations and separating the monthly statistics. Note that this also makes the monthly dataset start in 1990, implying that data was not collected for any country on a monthly basis before then. See example below. Also note that there is no GDP column in the monthly data, as that data is not logged on a monthly basis.

Time	Country	Core CPI,r	CPI Price,	CPI Price,	Core CPI,s	CPI Price,r
1990M01 [	Canada	69.26523	66.15411	5.443177		66.63771
1990M01 [	France		70.80587	3.568402	72.14992	70.96912
1990M01 [	Japan	93.75629	92.88061	3.322073		92.85714
1990M01 [	United Kingdom		61.16888	5.751456		61.02837
1990M01 [	United Sta	59.97655	58.62392	5.19802	59.87318	58.79448

b. Seasonally adjusted import-export ratio.

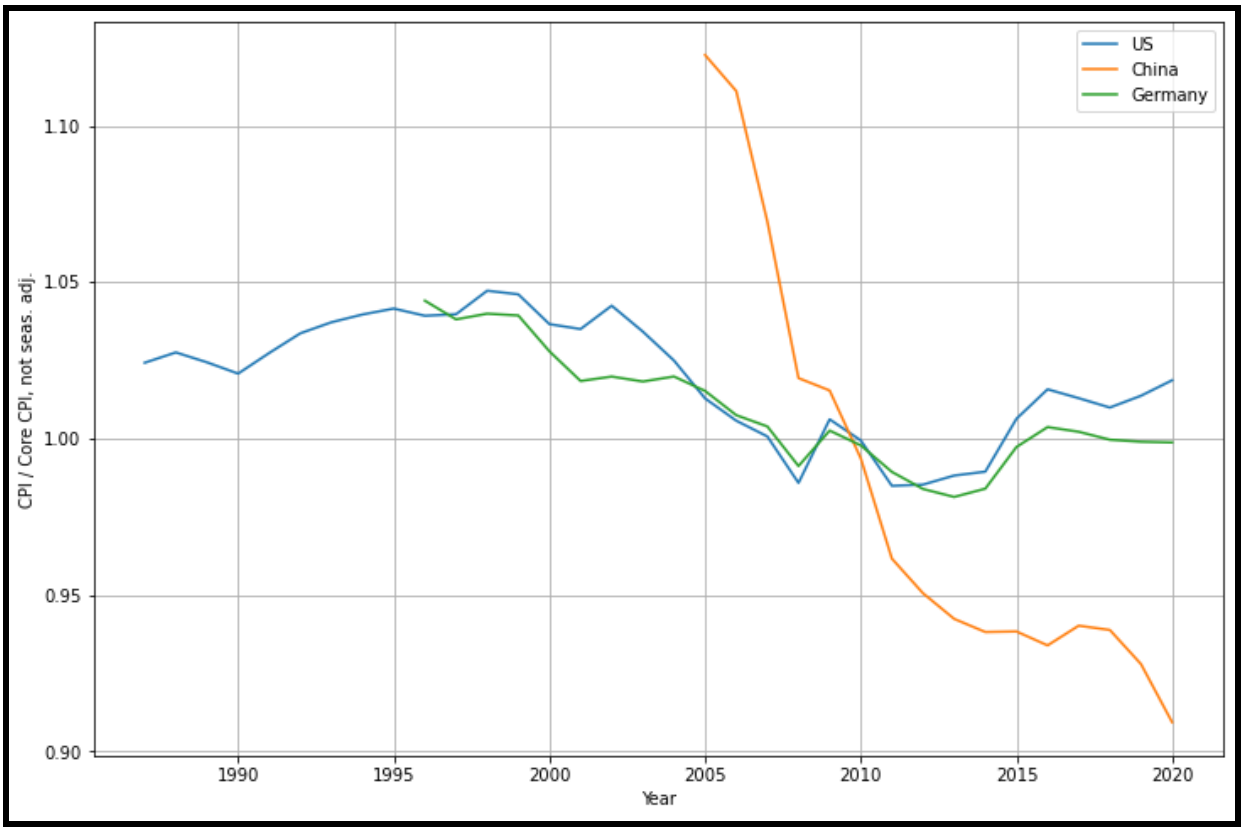
Here we look at the utility of one of our combination metrics, the import-export ratio, by plotting it for the United States (blue) and China (orange). In this case we can see that the United States typically imports more than it exports (positive values), while China typically exports more than it imports (negative values).

c. Industrial Production as Percent of GDP.

Another combined statistic we created was the Industrial Production as a Percent of GDP, found by dividing the GDP by Industrial Production value. Below we plot this data on a yearly basis for the United States, China, and Germany. The United States has a steadily decreasing contribution of industrial production to overall GDP (blue), while China (orange) and Germany (green) generally have elevated contributions of the industrial production to GDP. Interestingly, there is a noticeable dip for each country for the subprime mortgage crisis, approximately 2007-2008.

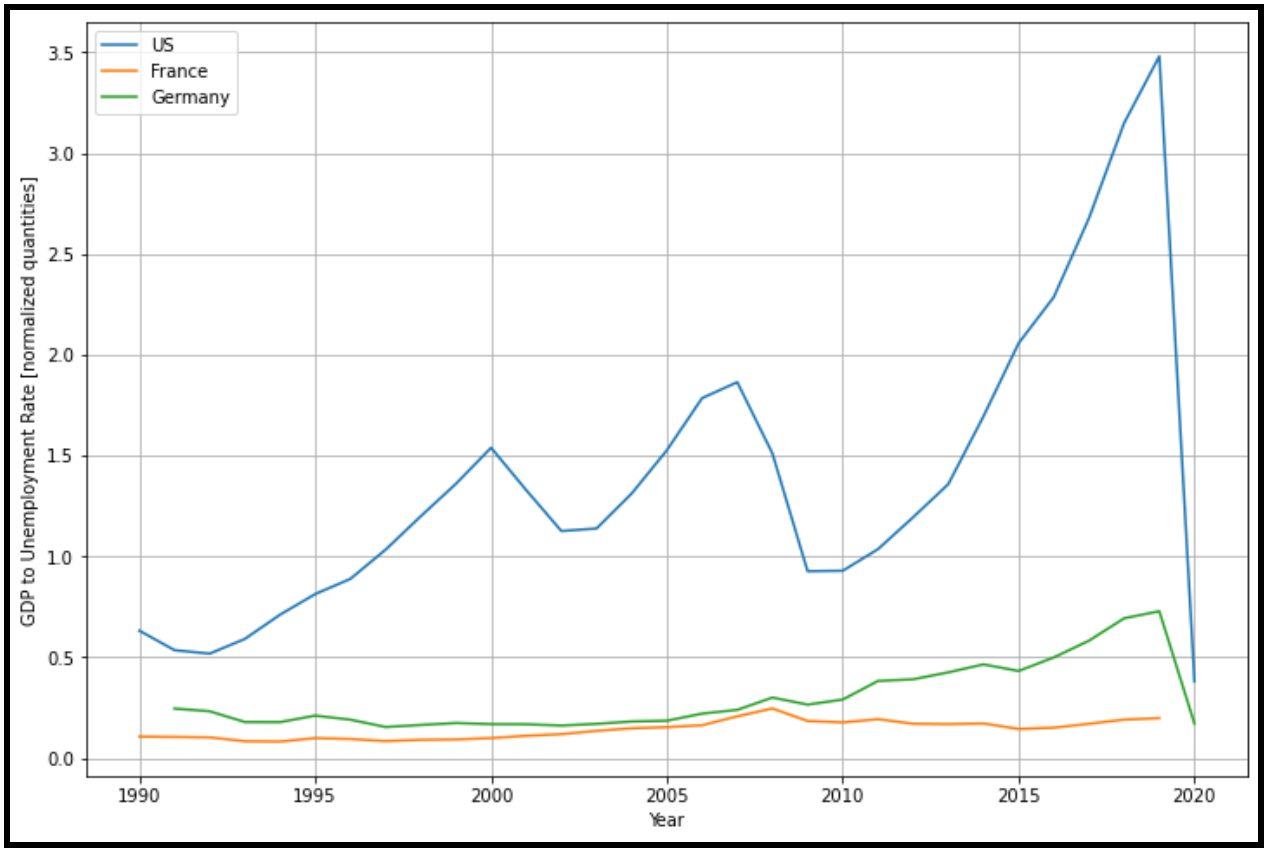
d. Core CPI / CPI, not seasonally Adjusted.

One of our queries had to do with the question, is core CPI a better measure of inflation than CPI? Below we plot the quantity CPI / Core CPI, not seasonally adjusted, for the United States (blue), China (orange), and Germany (green). For the United States and Germany, we find that generally CPI runs higher than Core CPI, with the exception of the subprime mortgage crisis in 2007-2008 and the approximate recovery period for the economies. This for the most part confirms one of the authors' precepts, that the Fed prefers Core CPI because it is typically a bit lower than CPI. Although both CPI metrics are arguably highly manipulated measures of inflation in the first place. Lastly, China's CPI / Core CPI is quite interesting, perhaps implying that inflation for non-food and energy factors has increased much faster than inflation for food and energy in the last 15 years or so.



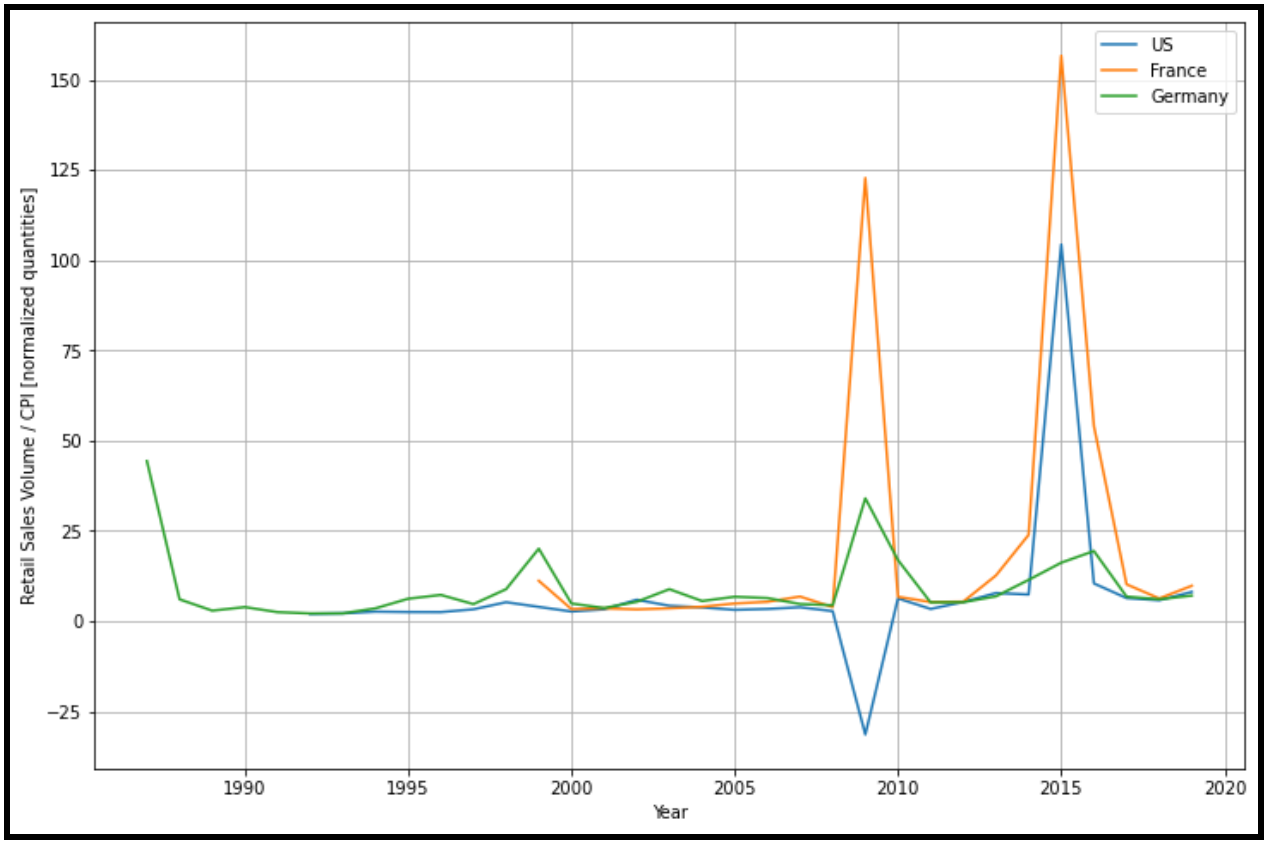
e. Normalized GDP / Normalized Unemployment Rate.

Here we look at the GDP divided by the unemployment rate, with both quantities being normalized 0.0 to 1.0 by dividing by their maximum value. Larger values indicate GDP increasing faster than the unemployment rate (or vice-versa). These are plotted for the United States (blue), France (orange), and Germany (green). It is interesting to point out dips in this quantity for United States around the time of the dot-com bubble (year 2000) and the subprime mortgage crisis (2007). Also of great interest is how this quantity dips incredibly rapidly for the United States and Germany at the onset of the Covid-19 pandemic in late 2019 / early 2020.



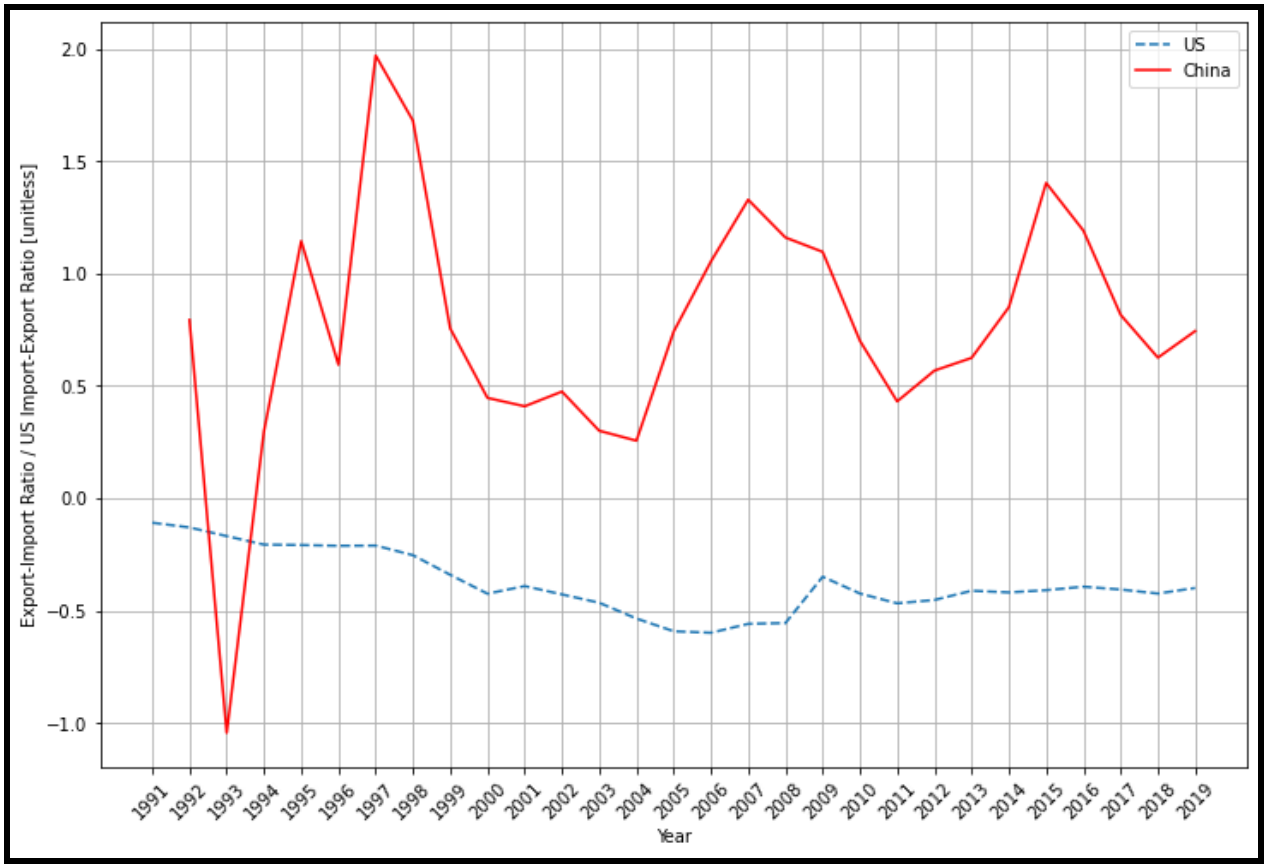
f. Normalized Retail Sales Volume / Normalized CPI.

Here we look at the retail sales volume and its relation to CPI, specifically the retail sales volume divided by the CPI %y-o-y, where each component is normalized according to its maximum value. This is plotted for the United States (blue), France (orange), and Germany (green). Larger values would generally indicate much larger retail sales volumes, while negative values (United States) indicate a negative %y-o-y inflation rate. Unfortunately this does not seem like an overly useful metric, as it's not clear whether large peaks are due to large retail sales volumes or very low inflation.



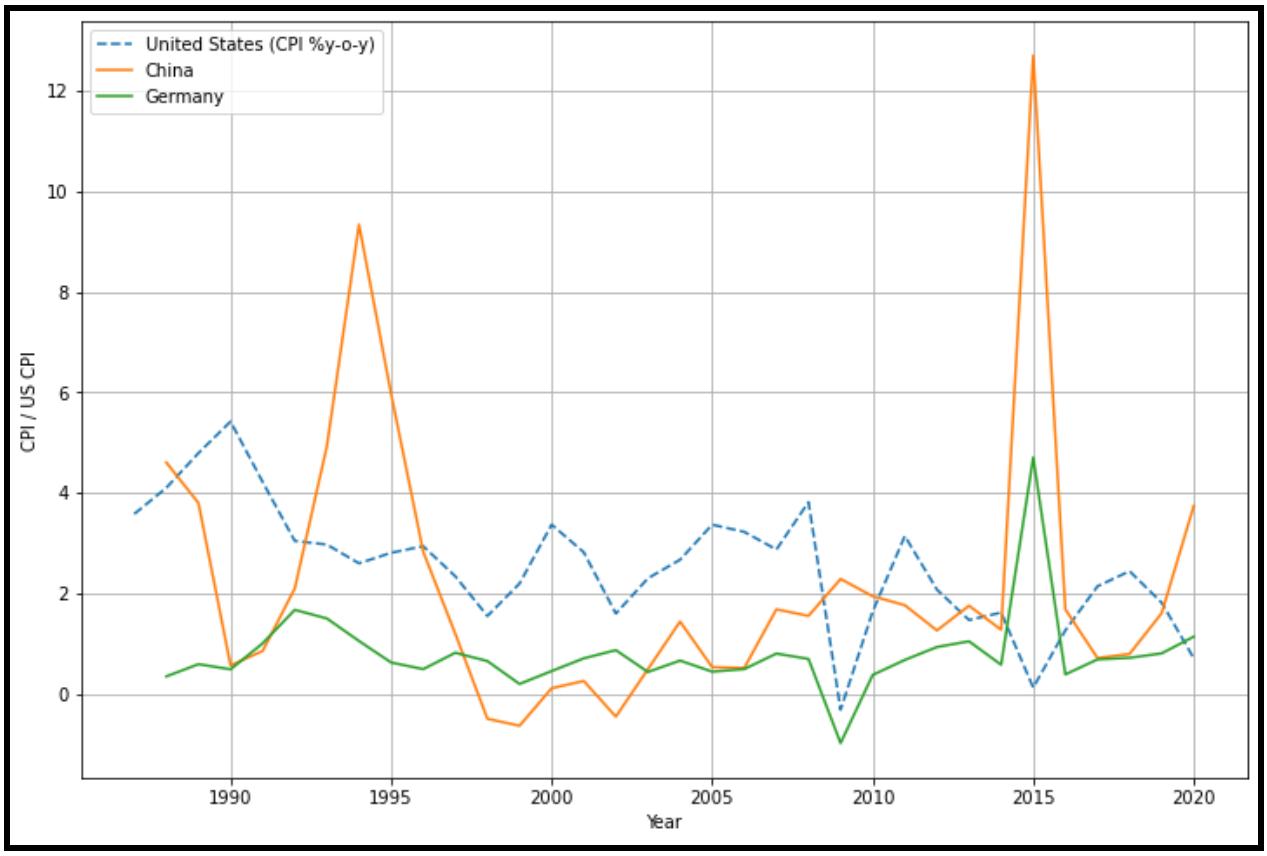
g. Export-Import Ratio / US Import-Export Ratio

Here we queried and plotted the Chinese Export-Import ratio divided by the U.S. Import-Export Ratio (red). The US Import-Export Ratio is also shown with the dotted blue line for comparison purposes, and it is multiplied by a constant value of 10.0 to better show trends in this quantity. It is observable that the United States has generally imported more than it exported, while China has largely increased its exports relative to its imports.



h. CPI / US CPI

In this metric we were interested in the relation of a country's CPI to the United States' CPI, and whether US CPI could serve as a leading indicator for another country's CPI. We computed the quantity, CPI / US CPI, for China (orange) and Germany (green). For interest in the leading indicator part, we also plotted the United States CPI %y-o-y in the dashed blue line. For starters, we can observe that China and Germany typically have significantly higher inflation than the United States, as evidenced by the quantity CPI / US CPI typically being greater than 0.0. It seems the peak in U.S. inflation in about 1990 led to peaks in inflation in China in approximately 1994, meaning U.S. inflation is potentially a leading indicator for China's inflation in this case, although this trend is not as apparent in later years.



4. **Summary of Data Changes**

   The use of '/' below signifies a division operation, typically with regards to two normalized statistics.

   a. **Task 1**
      i. Drop 'Country Code' and 'Series Code" columns.
      ii. Fill NaN values with empty value.
      iii. Split the dataset into a quarterly and yearly dataset, and a monthly dataset (two datasets).
      iv. For each now separate dataset, permute the axes such that the rows are by time and secondarily by country, and the columns are the statistics.
      v. Export these datasets as csv for Task 2 subtasks.

   b. **Task 2.1**
      i. Removed columns "GDP, current LCU, millions, seas. Adj", "Stock Markets, LCU", "GDP, constant 2010 LCU, millions, seas. Adj."
      ii. Removed "GDP, constant 2010 US$, millions, seas. Adj" column.
      iii. Removed columns: "CPI Price, % y-o-y, median weighted, seas. Adj", "Exports Merchandise, Constant US$, seas. Adj.", "Exports Merchandise, Constant US$, not seas. Adj.", "Imports Merchandise, Constant US$, millions, not seas. Adj.", "Imports Merchandise, Constant US$, millions, seas. Adj.".

iv. Removed columns: “Imports Merchandise, Customs, Price, US$, not seas. Adj.”, “Imports Merchandise, Customs, Price, US$, Seas. Adj.”, “Exports Merchandise, Customs, Price, US$, not seas. Adj.”, “Exports Merchandise, Customs, Price, US$, seas. Adj.”.
v. Removed column “Total Reserves”.
vi. Added column seasonally adjusted import-export ratio.
vii. Added column seasonally adjusted export-import ratio.
viii. Added column Industrial Production as Percent of GDP.
ix. Added column Seasonal CPI / not seasonal CPI.
x. Added column Core CPI / CPI, seasonally adjusted.
xi. Added column Core CPI / CPI, not seasonally adjusted.
xii. Column, “Stock Markets, US$,,,,”, renamed to, “Stock Markets, US$, Billions”.
xiii. Column, “Exchange rate, new LCU per USD extended backward, period average”, renamed to, “Exchange rate, new LCU per USD, extended backward, period average (USD to EURO)”.
xiv. Added column normalized GDP / normalized unemployment rate.
xv. Added column normalized retail sales volume / normalized CPI.
xvi. Added column normalized Stock Market Total Value / normalized CPI.

c. Task 2.2

i. Drop ‘Country Code’ and ‘Series code’ columns
ii. Fill NaN entries with empty string values
iii. Convert All String Numerics to Number Types in OpenRefine (they were read in as strings)
iv. Remove the following columns: “CPI Price, % y-o-y, median weighted, seas. Adj.,”, "GDP,current LCU,millions,seas. Adj.,", "Stock Markets, LCU,,,", "Imports Merchandise, Customs, Price, US$, not seas. Adj.", "Imports Merchandise, Customs, Price, US$, seas. Adj.", "GDP,constant 2010 LCU,millions,seas. Adj.,", "GDP,constant 2010 US$,millions,seas. Adj.,", "Exports Merchandise, Customs, Price, US$, seas. Adj.", "Exports Merchandise, Customs, Price, US$, not seas. Adj.", "Imports Merchandise, Customs, constant US$, millions, not seas. Adj.", "Imports Merchandise, Customs, constant US$, millions, seas. Adj.", "Exports Merchandise, Customs, constant US$, millions, seas. Adj.", "Exports Merchandise, Customs, constant US$, millions, not seas. Adj.",
v. Rename “Stock Markets, US$,,,” to “Stock Markets, US$, Billions”
vi. Post Processing:
    1. Remove column GDP,current US$,millions,seas. Adj.,
    2. Convert “Industrial Production, constant US$, seas. Adj.,,” and "Industrial Production, constant US$,,," columns to number values

d. Task 3

i. Fill NaN entries with empty string values
ii. Added column Import/Export
iii. Added column Export/Import
iv. Added column Retail Sales

v. Added column Industrial Product GDP
vi. Added column Core CPI
vii. Added column US Unemployment as leading/trailing.

5. **Conclusions**
    a. Our work in cleaning the dataset described herein, and synthesizing new statistics, made possible the analysis of historical macroeconomic statistics for the top 25 countries by GDP in the year 2021. As documented through the various plots and queries in Section 3, there are some interesting relations that can be drawn among the macroeconomic statistics, and among various countries.
6. **Lessons Learned**
    a. The lessons learned from this project are mainly in the realm of the handling of large amounts of international financial data. We learned tacit methods for cleaning this dataset, along with implicit knowledge of macroeconomic variables and their meaning. We became more familiar with OpenRefine and learned how to produce graphs/workflows, and developed our critical thinking with choosing what is the best tool to use at a given time. Along with this we were able to combine the usage of python's pandas library with OpenRefine to help calculate desired information, which can both serve as powerful and useful data mining and data cleaning tools with proper knowledge. Another lesson learned was understanding the strengths and weaknesses of each data cleaning tool, and when/where/why to apply one or the other. Lastly, working with YesWorkflow and the or2ywTool provided a hands-on method for implementing provenance in our workflows.
7. **Future Work**
    a. Implementations for the main use case $\mathbf{U_1}$ using this dataset could be optimizing foreign trading costs for your company or something along the lines of providing analysis on monetary statistics between the highest GDP nations. Due to the nature of the data, the use cases would have fallen under more business and financial orientated products, but still provides helpful results for research into international economics as well. Numerous plots can be generated from our dataset, for investigating financial and geopolitical strategies.