**FX Value as a construct of PPP scores has been around for decades, and yes it has good portfolio characteristics, can have positive drift or expected value, etc. Based on some of the steps above, think a bit more around the concept of “value”, especially leaning on some of the lecture notes around both quantitative and fundamental versions. Leveraging data accessible to you (if you have a specific ask, Dr. Jackson and I will do our best to provide time series from BBG), research, test, and implement a new metric for “value” in the FX G10 world. Once you have your equity series, be sure to comment on things like cost, capacity, robustness, etc. Note your grade will have nothing to do with the actual performance of the strategy but the research process, explanation of steps and results, and critical thinking about the strengths and weaknesses.**

The Traditional Definition (PPP-Based): Value in FX trading has historically been associated with Purchasing Power Parity (PPP). A currency is considered undervalued if the market exchange rate is below the PPP rate, and overvalued if above.

For a more robust metric, "value" can be defined as the combination of several economic indicators that collectively suggest whether a currency is priced appropriately relative to its fundamental economic conditions.

Our first approach was to look at the inflation rate of each of the 10 countries we are studying, as this could be an interesting way to measure how good the economies are doing. However we encountered 2 main problems with this approach. Firstly, inflation metrics are released monthly, which is too little robust for our trading algorithm. The second problem we found is that inflation could be ambivalent in the sense that it may be a good signal for the market (good economies grow, which leads to greater inflation), or a bad signal (high rates or even uncontrollable inflation destroy economies).

* Long and short term ideas
* Volatility of the metric?

Weighted sum:

* PPP
* GDP
* CAB: positive means x > i => which means more (csv file)
* RIR (fred)

‘

Krish Write-up Part 4

from fredapi import Fred

import pandas as pd

fred = Fred(api\_key='321d7bf2c5feafe84bc8c7b84a252cdc')

# Dictionary to store country codes and their corresponding FRED series IDs for interest rates

interest\_rate\_series = {

'Australia': 'IR3TIB01AUM156N', # Example: 3-Month or 10-Year bond yield

'Canada': 'IR3TIB01CAM156N', # Example: 3-Month or 10-Year bond yield

'European Union': 'IR3TIB01EZM156N', # Euro Area 3-Month or 10-Year bond yield

'Japan': 'IR3TIB01JPM156N', # Example: 3-Month or 10-Year bond yield

'New Zealand': 'IR3TIB01NZM156N',# Example: 3-Month or 10-Year bond yield

'Norway': 'IR3TIB01NOM156N', # Example: 3-Month or 10-Year bond yield

'United Kingdom': 'IR3TIB01GBM156N', # Example: 3-Month or 10-Year bond yield

'Sweden': 'IR3TIB01SEM156N', # Example: 3-Month or 10-Year bond yield

'Switzerland': 'IR3TIB01CHM156N',# Example: 3-Month or 10-Year bond yield

'United States': 'DGS10' # 10-Year Treasury yield as a nominal interest rate

}

# Fetch interest rate data for each country and store in a dictionary

interest\_rate\_data = {}

for country, series\_id in interest\_rate\_series.items():

try:

data = fred.get\_series(series\_id, observation\_start='1990-01-01')

interest\_rate\_data[country] = data

except Exception as e:

print(f"Error fetching data for {country}: {e}")

# Convert the data into a unified DataFrame

df\_interest\_rates = pd.DataFrame(interest\_rate\_data)

df\_interest\_rates.head(100)

—----

cab = pd.read\_csv('current\_account\_balance.csv', skiprows=2)

# get rid of time

cab['Category'] = pd.to\_datetime(cab['Category']).dt.date

new\_metric\_df = merged\_df.copy()

# add the cab data to the new\_metric\_df matched by their time

new\_metric\_df['Dates'] = pd.to\_datetime(new\_metric\_df['Dates'], errors='coerce')

cab['Category'] = pd.to\_datetime(cab['Category'], errors='coerce')

if new\_metric\_df['Dates'].isnull().any() or cab['Category'].isnull().any():

print("There are non-date values in 'Dates' or 'Category' columns that have been converted to NaT.")

# Proceed with the merge after sorting

new\_metric\_df = new\_metric\_df.sort\_values('Dates')

cab = cab.sort\_values('Category')

# Merge using pd.merge\_asof with backward direction

new\_metric\_df = pd.merge\_asof(

new\_metric\_df,

cab,

left\_on='Dates',

right\_on='Category',

direction='backward'

)

new\_metric\_df.head()