Stat Arb: Archimedean Copulas

Health Care and Gold

Ryan Finegan

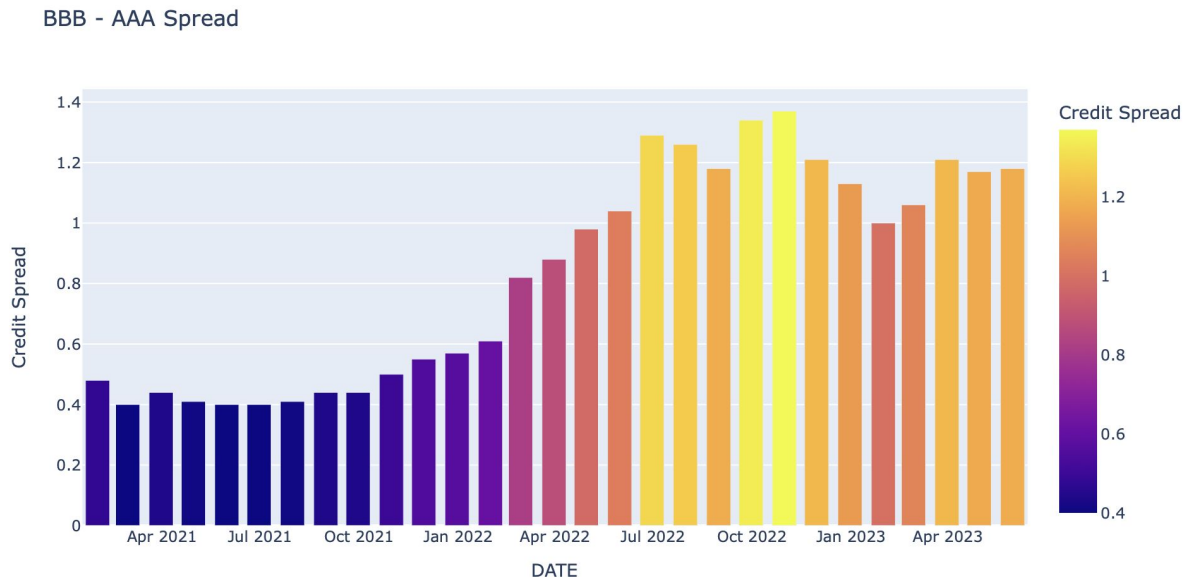
1. Trade is more based on macro view, but buying at a good price is discussed with brief overview of pairs trading with Copulas
2. Macro: Produced with Plotly Python and utilized Python and RStudio's FRED API
3. Security Pricing: Utilized Python to scrape Yahoo (might not be too accurate on daily pricing)

Best Idea:
Buy GLD US Equity @ $187.46

Current Macro

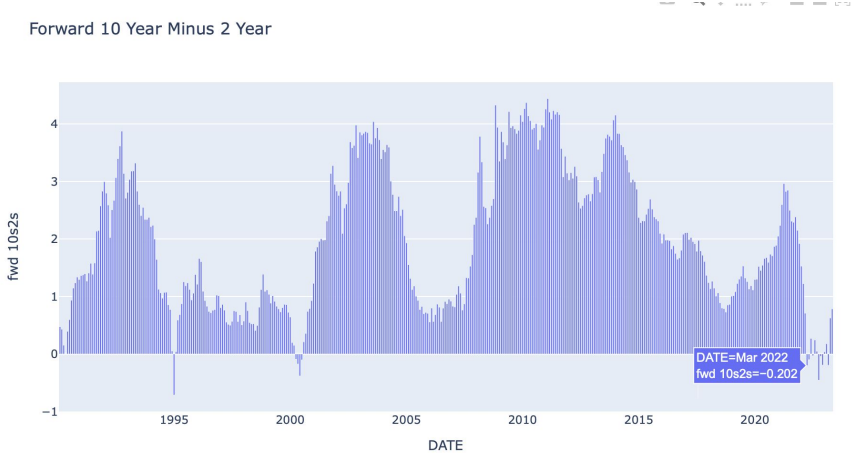
BBB spread rising relative to AAA

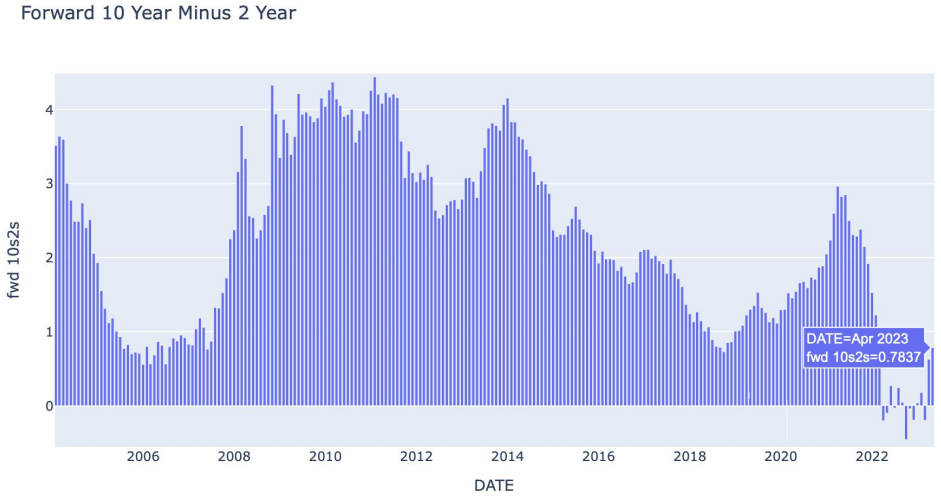
- Credit spread widening is a bearish signal and has prompted other historic market downturns



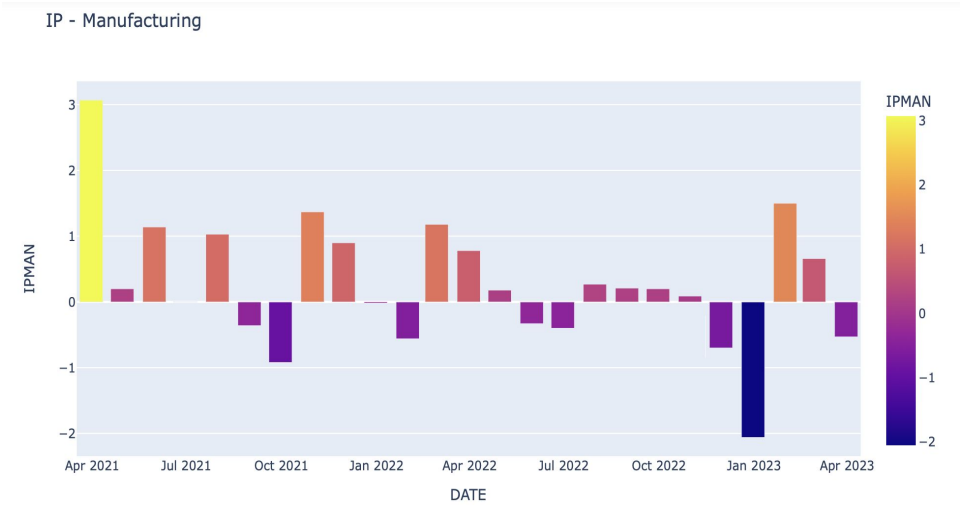
1 Year Forward 10s and 2s

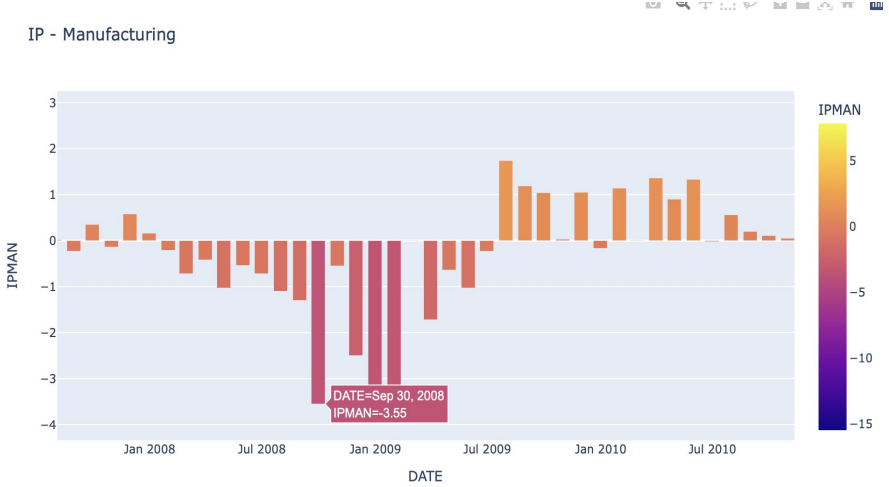
- One year forward yield curve is a good indicator of good and bad markets
- Steepening / accelerating means real growth





Industrial Production - ROC

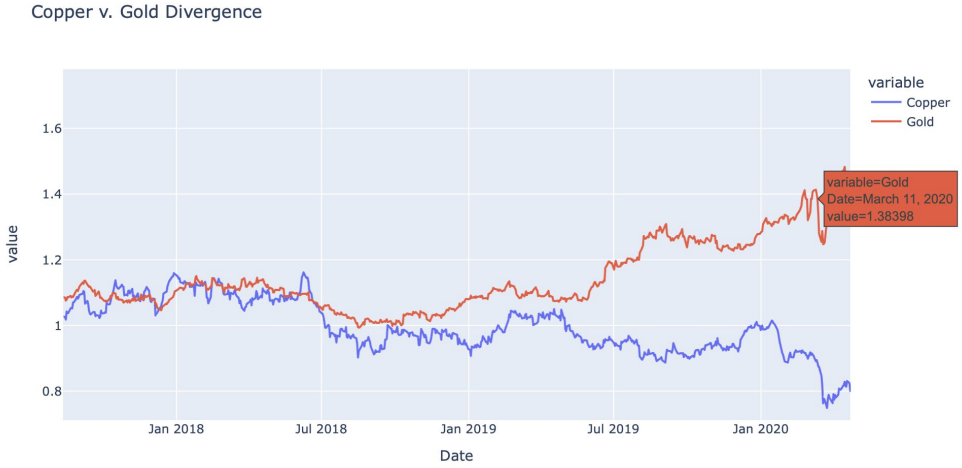


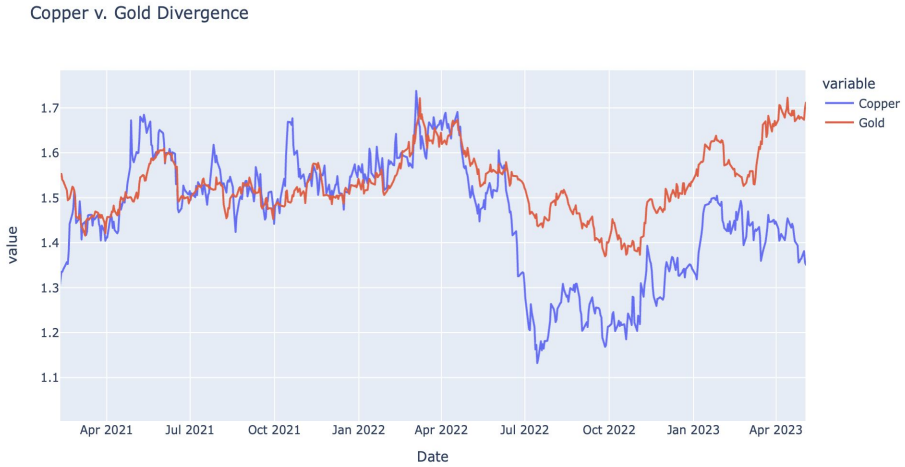


- Percentage down M/M:
  - April 2008: ~ (-1.03%)
  - March 2020: ~ (-4.5%)
  - April 2020: ~ (-15.5%)
  - March 2023: ~(-0.53%)

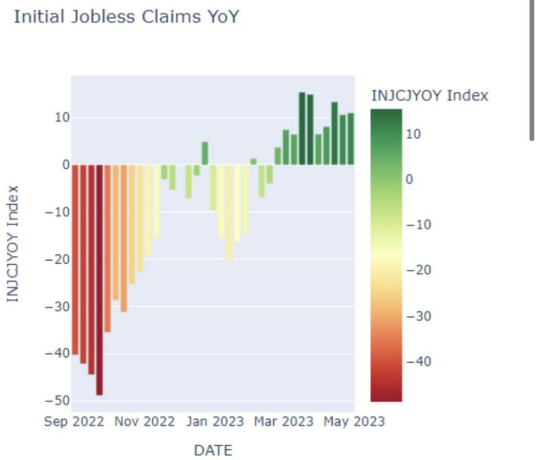
Copper - Gold Divergence

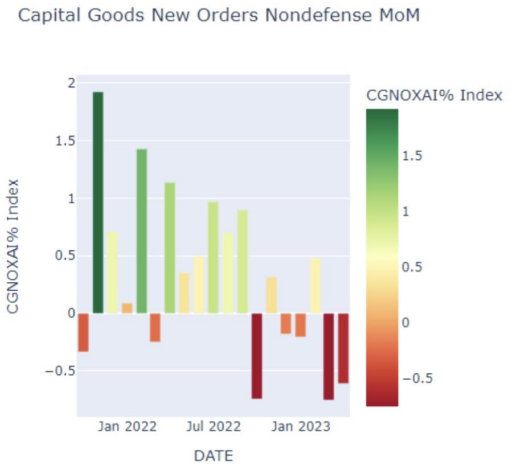
- Copper and gold diverge because gold is a safe haven asset
- Copper does well when there is economic growth because it is utilized in many industrial applications

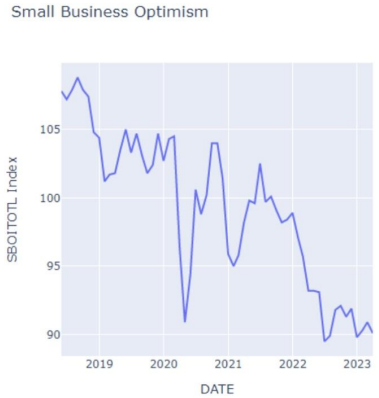




Capital Goods New Orders / Jobless Claims / Small Business Optimism

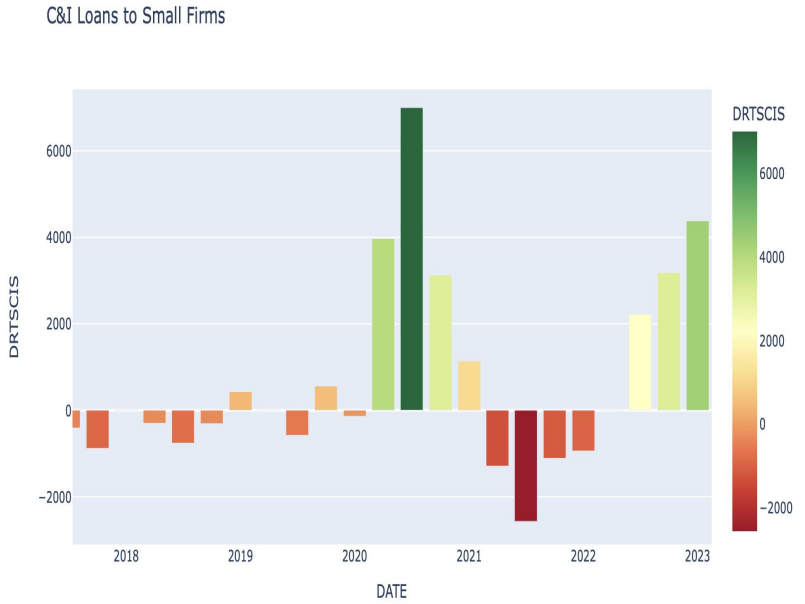


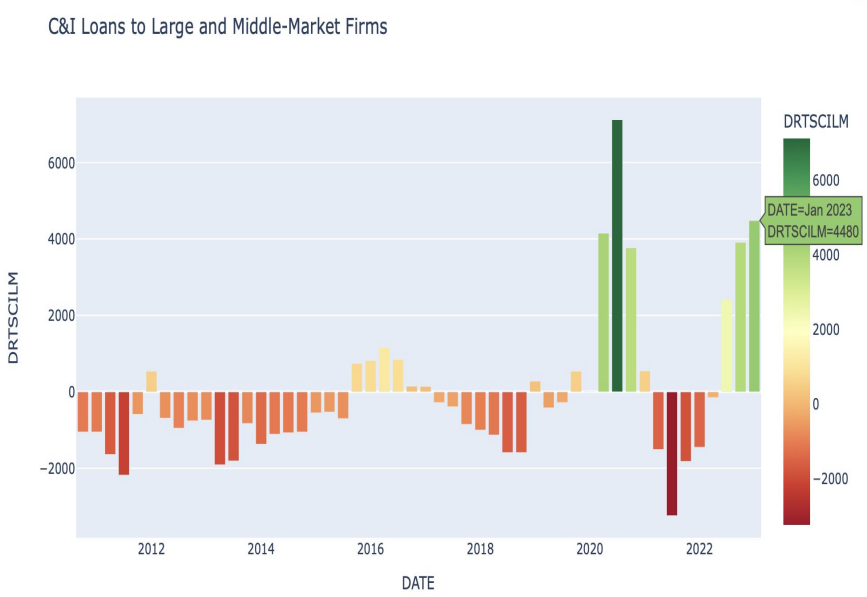




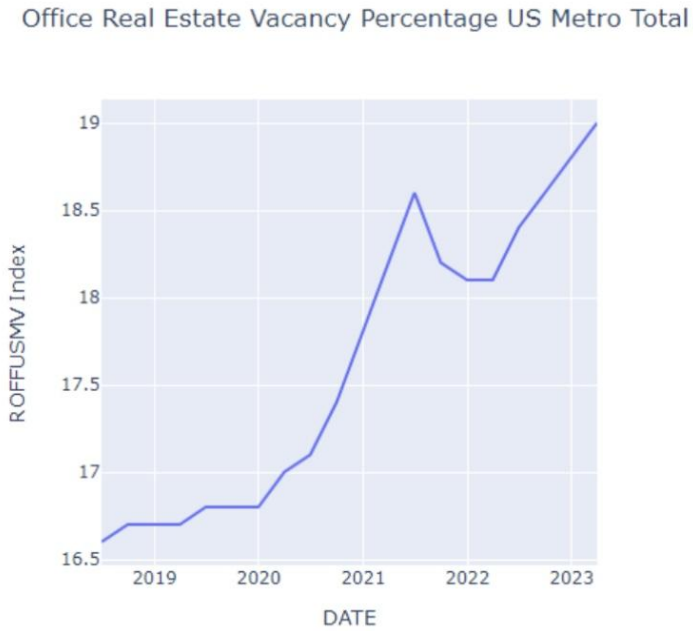
Senior Loan Officer Survey

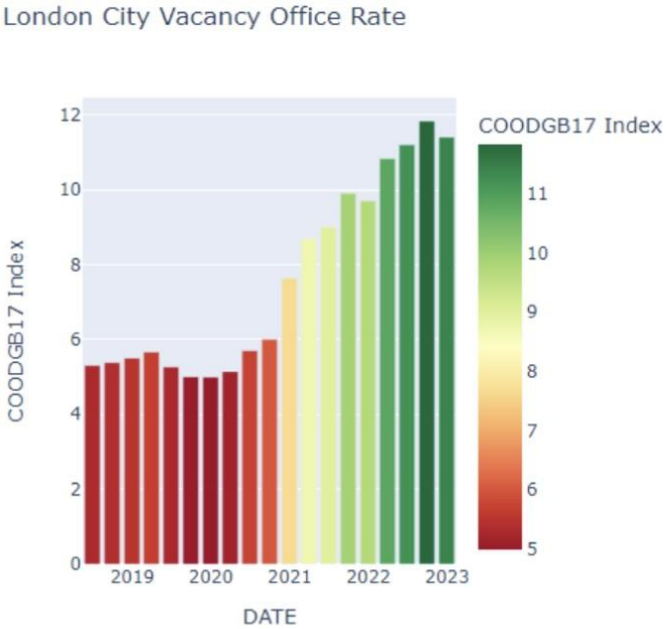
- Lending institutions standards rise during economic turbulence





CRE Data


Office Real Estate Vacancy Percentage US Metro Total
ROFFUSMV Index
19
18.5
18
17.5
17
16.5
2019
2020
2021
2022
2023
DATE


London City Vacancy Office Rate
COODGB17 Index
12
10
8
6
4
2
0
2019
2020
2021
2022
2023
DATE
COODGB17 Index
11
10
9
8
7
6
5

Trade Ideas

- Long GLD US Equity (Long Only)
  - Probably the best option
  - XLV and GLD do well in this regime (declining Growth and Inflation)
  - Choosing the undervalued security in the coupled (copula) pair should offer more value
- Long GLD - Short XLV
  - Stat Arb: Not capital intensive
  - Banking on undervalued pair to “catch up” to overvalued
  - Exit both when conditional probabilities cross 0.5 (calculated daily EOD)
- Conditional Probabilities Using the Frank Copula
  - GLD is undervalued compared to XLV (will be shown in a few slides)

Why GLD?

- Gold is trending up
  - from equities like miners to the commodity itself
- Safe haven when markets are in decline
- Alternative asset that isn’t consumed and could be used to conserve wealth

Why XLV?

- Health Care is known to be a very reliable defensive sector
- There’s consumer demand no matter the economic cycle

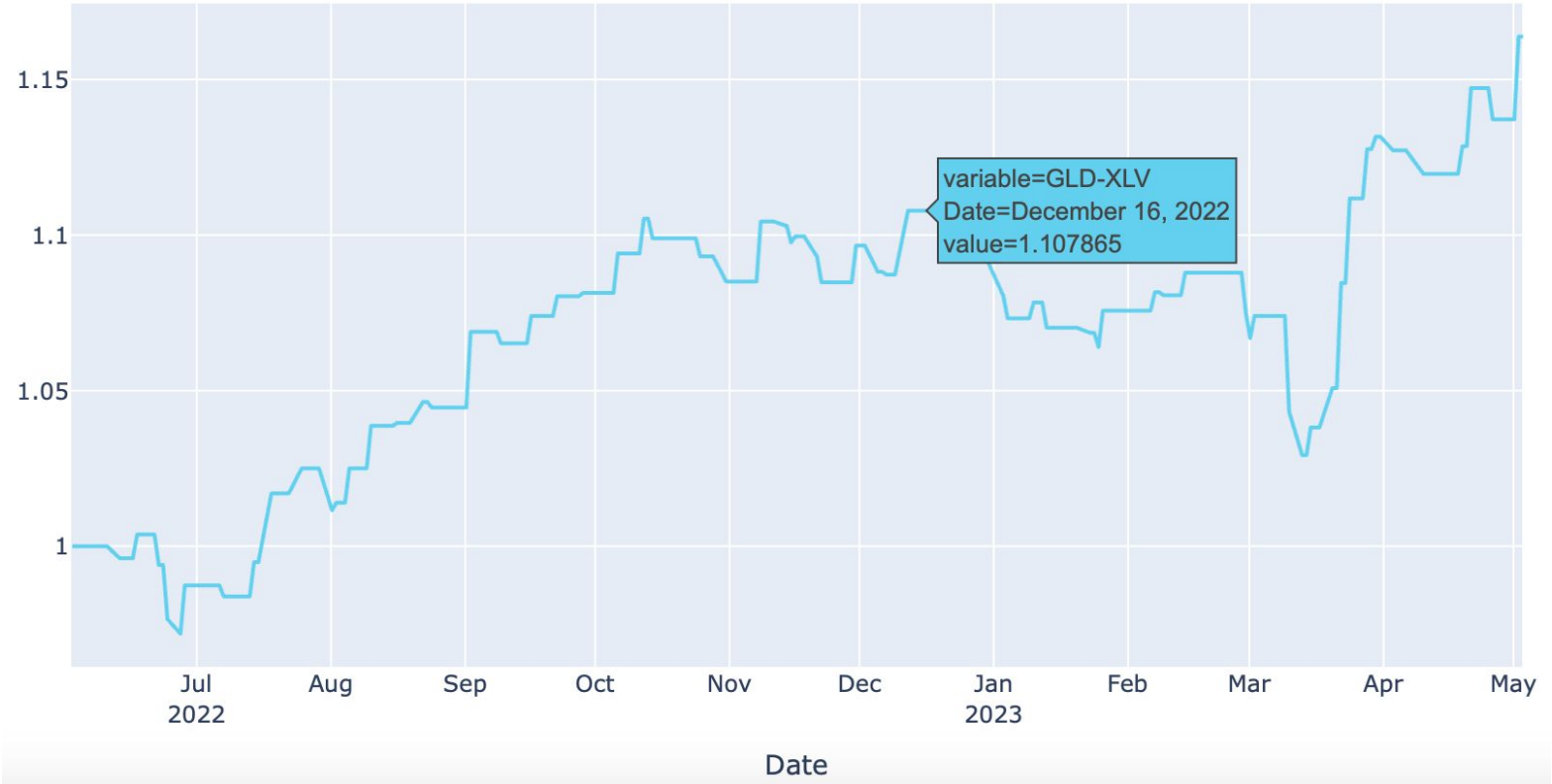
What's the Relationship?

- Both perform better in declining markets
  - Not cyclical bets like XLY, XLB, XLK

Systematic Backtest

- Used a mixture of Archimedean Copulas (Clayton, Gumbel, Frank)
  - Utilized a training (Jan 2021 to May 2022) and testing (June 2022 to April 2023) period
  - Copulas can be used to get the conditional probabilities of each security
  - Long the security with a conditional probability that is less than 0.4
  - Short the security with a conditional probability that is greater than 0.6
  - Close both positions when both conditional probabilities cross 0.5 threshold

Backtest - June 2022 through April 2023



Trading Using Copulas

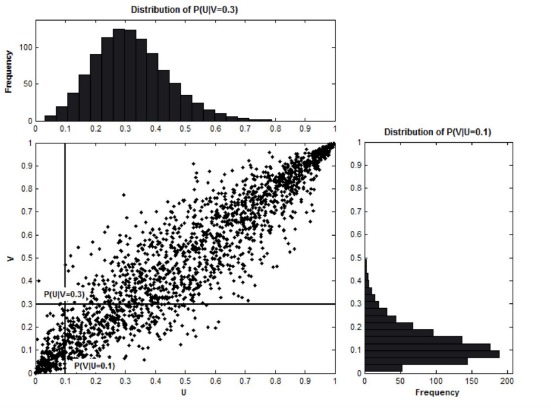
- Select securities that do well in the current market regime
  - Defensive / Safe Haven Assets
- Select pairs using Kendall Tau correlation coefficient
  - Use this instead of Pearson Rho because its calculated on order, not value (stability)
  - Doesn’t change when the distribution is changed (probability integral transform)
- Pick a training period
  - Used January 2021 to June 2022 as training
  - Sorted the asset pairs on highest Tau correlated daily log returns (could use weekly)

Copulas - ID Marginal Distributions

- Identify marginal distributions relative to the log returns of each stock pair
- Fit each log return of the screened securities to the student t marginal distribution
  - Used Student t because security returns are usually not normal and have fatter tails
  - I used Kolmogorov-Smirnov Test to see if the distributions fit the data well
  - Removed all securities that failed the KS Test (p-values lower than 0.1 meaning that the selected parametric distribution (Student T) for the security wasn’t a good fit

Probability Integral Transform

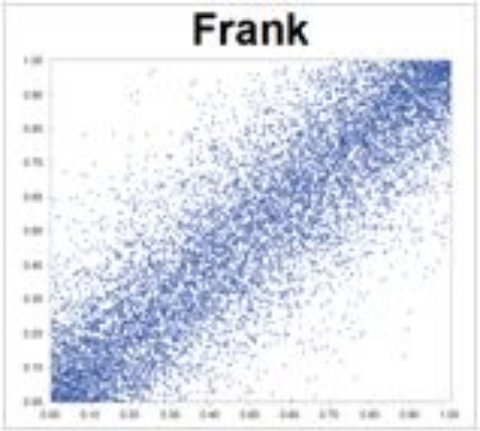
- Probability Integral Transform is used on the PDFs of the marginal returns to make them uniform (needed for coupling the securities)
  - Transform the PDF into a CDF to make it uniform (standardized
- Do this so copulas can be fit to the marginal distributions



“Trading strategies with copulas.” by Stander, Yolanda, Daniël Marais, and Ilse Botha.

Fit Copulas on Transformed Marginal Returns

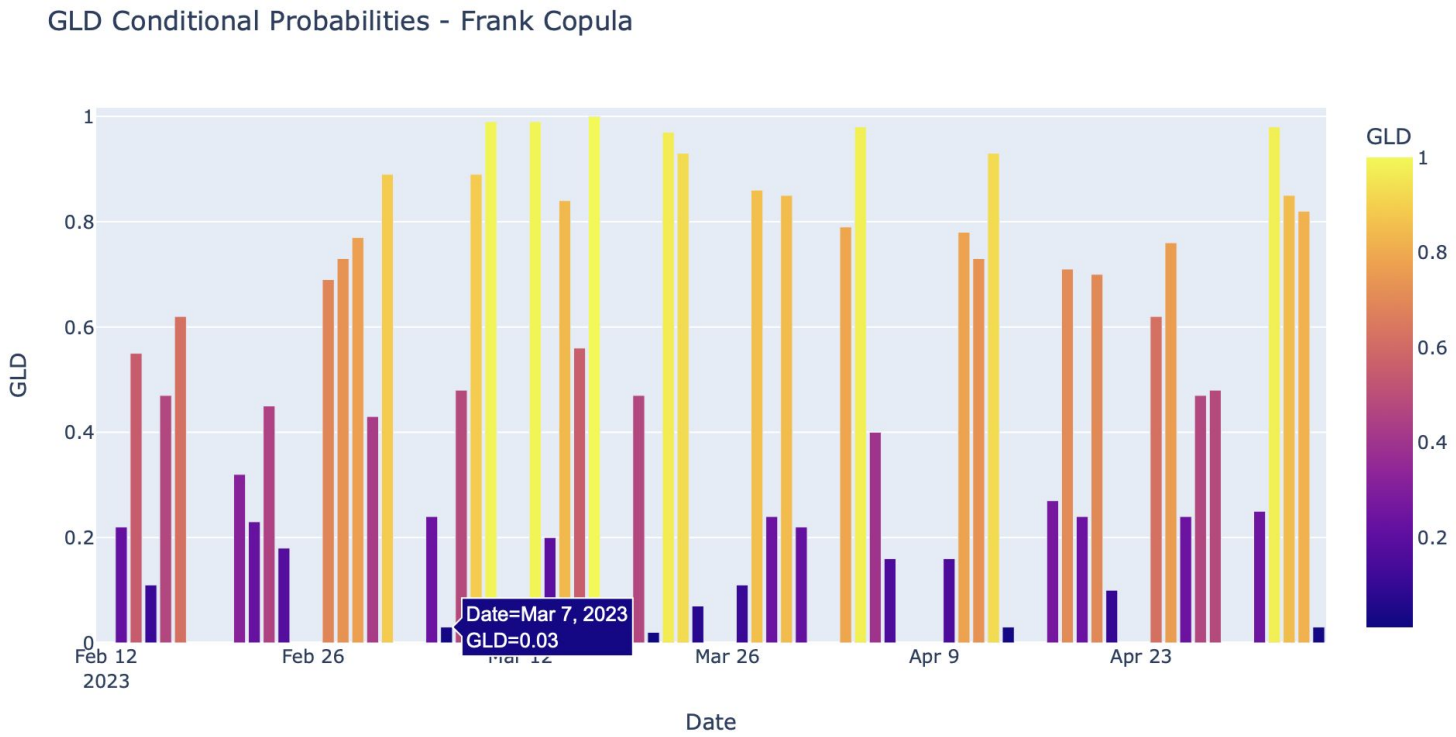
- Used Frank, Clayton, Joe, and Gumbel to fit to the marginal distributions
- Optimal copulas were selected to decouple the marginal distributions and it was selected using the lowest AIC according to log likelihood
- Frank copulas were used on the XLV and GLD pair
- Much like a Gaussian Copula with little tail dependence



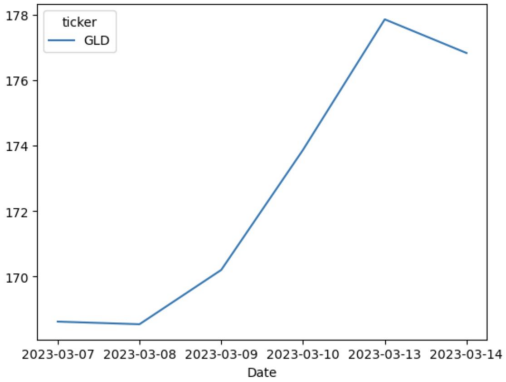
Last Week's Conditional Probabilities

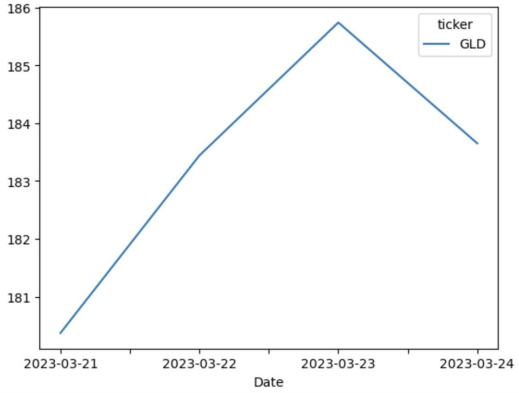
Date	GLD	XLV
2023-05-01	0.25	0.72
2023-05-02	0.98	0.24
2023-05-03	0.85	0.38
2023-05-04	0.82	0.14
2023-05-05	0.03	0.85

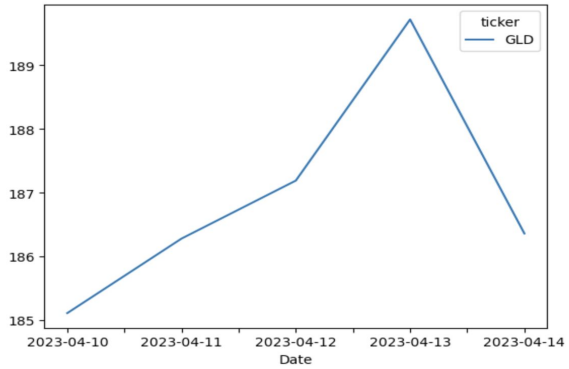
Conditional Probabilities (Frank Copula) - GLD US Equity



GLD Long Only -
Last three times @ less than 0.2 conditional probability







Date	GLD Prob	GLD Price
2023-03-07	0.03	168.62
2023-03-08	0.48	168.54
2023-03-09	0.89	170.20
2023-03-10	0.99	173.87
2023-03-13	0.99	177.86
2023-03-14	0.20	176.83

Date	GLD Prob	GLD Price
2023-03-21	0.02	180.37
2023-03-22	0.97	183.44
2023-03-23	0.93	185.74
2023-03-24	0.07	183.65

Date	GLD Prob	GLD Price
2023-04-10	0.16	185.11
2023-04-11	0.78	186.28
2023-04-12	0.73	187.19
2023-04-13	0.93	189.72
2023-04-14	0.03	186.36

Appendix

```python
class Trading:

    def __init__(self, condition, start, end):
        if condition == 1:
            self.period1 = start
            self.period2 = end
        else:
            self.period1 = int(time.mktime((dt.datetime.now() - dt.timedelta(365*35)).timetuple()))
            self.period2 = int(time.mktime(dt.datetime.now().timetuple()))

        self.fred_start = str(dt.datetime.now() - dt.timedelta(365*35)).split()[0]
        self.fred_end = dt.datetime.now()
        self.interval = '1d'
        self.ticker_list = ["^GSPC","XLV","VNQ","XLP","XLU","TLT","GDX","GLD","MUB","SPLV","DEF","XLRE","IEF",
                            "GDXJ","MBB", "UUP","IIGD","PINK","XLY","XLP","HG=F","GC=F"]

    def yahoo_universe(self):
        web3 = []
        failed = []
        for security in self.ticker_list:
            try:
                interval = '1d'
                query_string = f'https://query1.finance.yahoo.com/v7/finance/download/{security}?period1={self.perio
                df = pd.read_csv(query_string).set_index("Date")
                df['ticker'] = security
                df.columns = ['Open', 'High', 'Low','Close','Adj Close','Volume','ticker']
                web3.append(df)
            except:
                failed.append(security)
                pass
            final = pd.concat(web3)
            df_final = final.reset_index()
        return df_final, web3, failed
```

```
def trading_sql(self, data_file):
    try:
        conn = sqlite3.connect('tradable_universe')
        data_file.to_sql("consensusinvestable", conn, if_exists='replace', index = False)
        conn.commit()
        conn.close()
        print("Updating the sql table was a success")
    except:
        print("Was unsuccessful in uploading the dataframe into the datatable. ")


def to_trading_sql(self, data):

    """
    Inputs: dataframe you want to upload
    Outputs: saves SQL table to the investable database
    """

    try:
        connection = sqlite3.connect(config.db_file)
        data.to_sql("consensusinvestable", connection, if_exists='replace', index=False)
        connection.commit()
        connection.close()
        print("Uploading dataframe to database worked successfully.")
    except:
        print("Was unable to save the dataframe into the investable datatable.")
```

```python
def load_sql(self):
    try:
        conn = sqlite3.connect('tradable_universe')
        df = pd.read_sql_query("SELECT * from consensusinvestable", conn)
        conn.commit()
        conn.close()
    except:
        print("Could not retrieve the datatable. ")
    return df


def ticker_filter(self, ticker, final):
    return final[final['ticker']==f"{ticker}"]


def dataframe(self, tickers, final, start, end):

    """
    Inputs: The tickers you want to filter by and a raw unfiltered data query from trading database.
    Outputs: A dataframe with the columns being the names of the selected tickers and the values being
    the adjusted adjusted close prices.
    """

    new_df = []
    for ticker in tickers:
        new = final[final['ticker']==f"{ticker}"]
        new_df.append(new)
    main = pd.concat(new_df)
    df = main.pivot_table(index = main.index, columns = "ticker", values = 'Adj Close')
    return df[start:end]
```