Bond Pricing and Yield Exploration: Technical Report

By Alexander Wilkins

Introduction

As a Computer Science and Finance double major, I am constantly looking for projects that let me sit at the intersection of both disciplines. Bonds are a corner of finance that, in my experience, tend to get skipped over in the classroom and glossed over in casual conversation in favor of the more exciting world of equities. Yet their importance is hard to overstate: bonds fund governments and corporations, anchor institutional portfolios, and serve as a primary mechanism through which the Federal Reserve transmits monetary policy to the broader economy. Going into this project, I assumed bonds were fairly straightforward instruments defined by a handful of concrete inputs. A conversation with an analyst at a well-respected hedge fund quickly changed my perspective. He told me that, in practice, bonds are often viewed as the more abstract instrument compared to equities, largely because their pricing hinges on the discount rate, a figure driven by an enormous and constantly shifting set of macroeconomic forces. That idea stuck with me and became the central motivation for this project.

With that framing in mind, I set out to build a bond pricing and yield analysis tool in Python that would let me explore these dynamics hands-on. The goals were fourfold: accurately price treasury bonds using live yield data, compute meaningful risk metrics from that pricing, visualize how bond prices respond to changes in coupon rate and maturity, and dig into the macroeconomic factors that shape the yield curve itself. The project draws on real-time data from the Federal Reserve Economic Data API (FRED) and Yahoo Finance, numerical methods from NumPy and SciPy, and interactive visualizations built with Matplotlib and Plotly. It is not a comprehensive bond trading system, but rather a focused analytical tool designed to build genuine intuition for how these instruments behave and how sensitive they are to changes in interest rates.

System Design

The system is built entirely in Python across five modules: get_data.py, utils.py, pricing.py, charts.py, and main.py. They are organized around four core functions: data ingestion, bond pricing, risk metrics, and visualization.

Data Ingestion

The data layer connects to two distinct sources. The primary source is the Federal Reserve Economic Data API (FRED), accessed via the fredapi library. On each execution of get_data.py, the tool pulls seven U.S. Treasury yield series (DGS3MO, DGS1, DGS2, DGS3, DGS5, DGS7, DGS10), the federal funds rate (FEDFUNDS), core CPI (CPILFESL), and the unemployment rate (UNRATE). The second source is Yahoo Finance, accessed via yfinance, which supplies monthly SPY closing prices for use in the correlation analysis. All series are written to CSV files in a local data/ directory and read from there at runtime, keeping the analysis fast and separating the data fetch step from the analytical pipeline. A FRED API key

is required and is loaded from a .env file via python-dotenv, keeping credentials out of the codebase.

Bond Pricing

The pricing layer in pricing.py implements the standard discounted cash flow formula for a fixed-coupon bond. For each target maturity, the tool interpolates the appropriate discount rate from the live yield curve using NumPy's interp function. This was a necessary design decision because FRED provides data only at the benchmark maturities, and the pricing calculator needs a discount rate for every integer maturity from 1 to 10 years. The core formula computes the present value of the annuity of coupon payments plus the present value of the face value repaid at maturity:

$$\text{Price} = (C / r) \times (1 - (1 + r)^{-T}) + F / (1 + r)^{T}$$

where C is the annual coupon payment, r is the interpolated discount rate, T is the maturity in years, and F is the face value. The face value used in the interactive chart defaults to $2,000 and is adjustable within the function parameters.

Risk Metrics

Beyond price, the tool computes three standard fixed income risk metrics via the bond_metrics function: Macaulay duration, modified duration, and convexity. These are the quantities practitioners actually use to manage interest rate exposure, and implementing them correctly required working carefully with the full cash flow schedule rather than relying on simplified approximations.

Macaulay duration is the weighted average time to receive the bond's cash flows, where each weight is that cash flow's share of the bond's total present value. It measures, in years, how long it takes on average to be repaid:

$$D_{mac} = \text{sum}(t \times PV(CF_t)) / \text{Price}$$

Modified duration rescales Macaulay duration to give a direct estimate of price sensitivity to a small change in yield. Specifically, it approximates the percentage change in bond price for a one-unit change in yield:

$$D_{mod} = D_{mac} / (1 + r)$$

Convexity captures the curvature of the price-yield relationship, which modified duration alone misses. Because the price-yield curve is convex rather than linear, duration underestimates price gains and overestimates price losses when yields move by a large amount. Convexity provides the second-order correction:

$$\text{Convexity} = \text{sum}((t^2 + t) \times PV(CF_t)) / (\text{Price} \times (1 + r)^2)$$

The tool also solves for yield-to-maturity (YTM) using SciPy's brentq root-finding algorithm. Given a market price, YTM is the single discount rate that makes the bond's discounted cash

flows equal to that price. There is no closed-form solution for YTM, so numerical root-finding is the standard approach. Brent's method was chosen because it is guaranteed to converge when given a valid bracket, which here is any yield between 0.0001 and 10.0. All four metrics are printed to the console on each run of main.py.

Visualization and Interface

The visualization layer in charts.py generates four distinct outputs. First, a static Matplotlib yield curve plots the most recent yield at each of the five benchmark maturities, giving an immediate snapshot of the current interest rate environment. Second, an interactive Plotly bond pricing chart allows the user to adjust the coupon rate via a slider and watch the bond price curve update across all maturities in real time. Third, a Matplotlib time-series chart overlays historical treasury yields for the 3-month, 2-year, and 10-year maturities alongside the federal funds rate over the past 10 years. Fourth, a 2x2 panel of correlation charts plots each of the four yield series against unemployment, inflation, and SPY. All series in the correlation panels are normalized to z-scores using scikit-learn's StandardScaler before plotting, which puts yields and macroeconomic indicators on a common scale and makes the directional relationships between them visually interpretable despite large differences in their raw units. The correlation matrix is also exported to output/correlations.csv at runtime.

```
def price_bond(face_value, coupon_rate, maturities):  3 usages
    prices = []
    for maturity in maturities:
        discount_rate = np.interp(maturity, current_yields['Maturity'], current_yields['Yield'])/100
        coupon_payment = (face_value * coupon_rate)
        discounted_payments = (coupon_payment / discount_rate) * (1-(1/(1+discount_rate))**maturity)
        discounted_face_value = (face_value / (1+discount_rate)**maturity)
        present_value = discounted_payments + discounted_face_value
        print(f"Bond Price: ${present_value:.2f}")
        prices.append(present_value)
    return prices
```

Figure 1. The price_bond function, which interpolates live FRED yield data to discount coupon payments and face value at each target maturity.

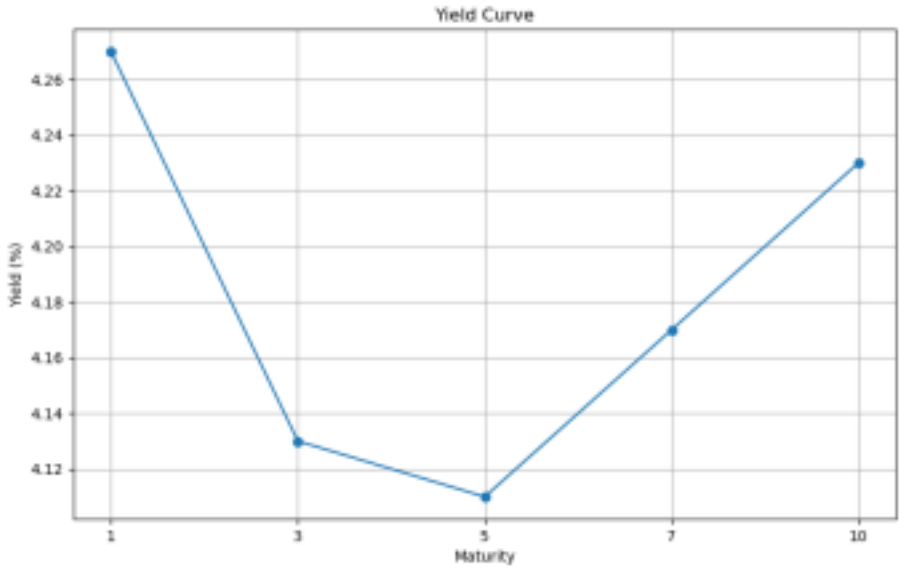

Figure 2.

Live yield curve generated from FRED data at the time of execution, plotting treasury yields at the 1-, 3-, 5-, 7-, and 10-year maturities. The non-linear shape reflects the partially recovered yield curve present in the data at that time.

Figure 3.

Interactive bond pricing chart showing Bond Price vs. Maturity at a 5.00% coupon rate. The Plotly slider allows the user to adjust the coupon rate from 1% to 10% and observe the resulting price curve in real time.

Evaluation

This project accomplished what it set out to do, though not without a few honest lessons along the way. The piece I am most proud of is the pricing and risk metrics layer. Getting the pricing logic right required careful attention to how the discount rate is sourced and interpolated. Early versions of the tool used a naive fixed discount rate, which produced clean-looking but fundamentally inaccurate results. Switching to live FRED yields and interpolating across the maturity spectrum produced a much more realistic picture, including the non-linearity visible in Figure 2 that a fixed rate would have masked entirely. Extending the tool to compute duration and convexity pushed that further. Macaulay duration and modified duration are straightforward once the cash flow schedule is set up correctly, but convexity required more careful thought about what the second-order approximation is actually capturing and why it matters. Implementing YTM via Brent's method was similarly instructive. The fact that there is no closed-form solution for YTM is something I had read about but never fully appreciated until I had to solve for it numerically and understand why the root-finding bracket needs to be set the way it does.

The interactive pricing chart in Figure 3 made it genuinely easy to build intuition for concepts that are hard to grasp from formulas alone. Dragging the coupon rate slider and watching the price curve shift makes the inverse relationship between yield and price, and the way that relationship steepens at longer maturities, immediately visible. It also makes the duration story concrete: a high-coupon bond at the left of the slider moves less for a given yield change than a low-coupon bond at the right, which is exactly what modified duration predicts.

Figure 4. U.S. Treasury yields over time (2015 to 2025) for the 3-month (DGS3MO), 2-year

(DGS2), and 10-year (DGS10) maturities, alongside the federal funds rate (FEDFUNDS). The sharp convergence beginning in 2022 reflects the Federal Reserve's aggressive rate hiking cycle.

The time-series chart in Figure 4 makes one pattern immediately clear: shorter-maturity treasury yields track the federal funds rate much more tightly than longer-maturity yields do. The 3-month yield and the federal funds rate are nearly indistinguishable for most of the period shown. The 10-year yield, by contrast, diverges meaningfully in both directions at different points, reflecting the market's forward-looking expectations about growth and inflation rather than just the current policy rate. This is also where the duration intuition from the pricing layer connects back to the yield analysis: a longer-duration bond is more exposed to exactly the kind of yield movements visible in the 10-year series.

The macroeconomic correlation analysis was the most intellectually humbling part of the project. Going in, I expected to find clean, interpretable relationships between variables like inflation or unemployment and treasury yields. What I found instead was a set of correlations that shift dramatically depending on the time window, the maturity of the bond, and the broader economic regime. The four-panel chart in Figure 5 overlays unemployment (UNRATE), inflation (CPILFESL), and SPY against each of the tracked yields, all normalized to z-scores so the directional relationships are visible across series with very different raw scales. The 2020 spike in unemployment is the dominant visual feature in every panel and makes it difficult to draw conclusions that generalize beyond that single event.

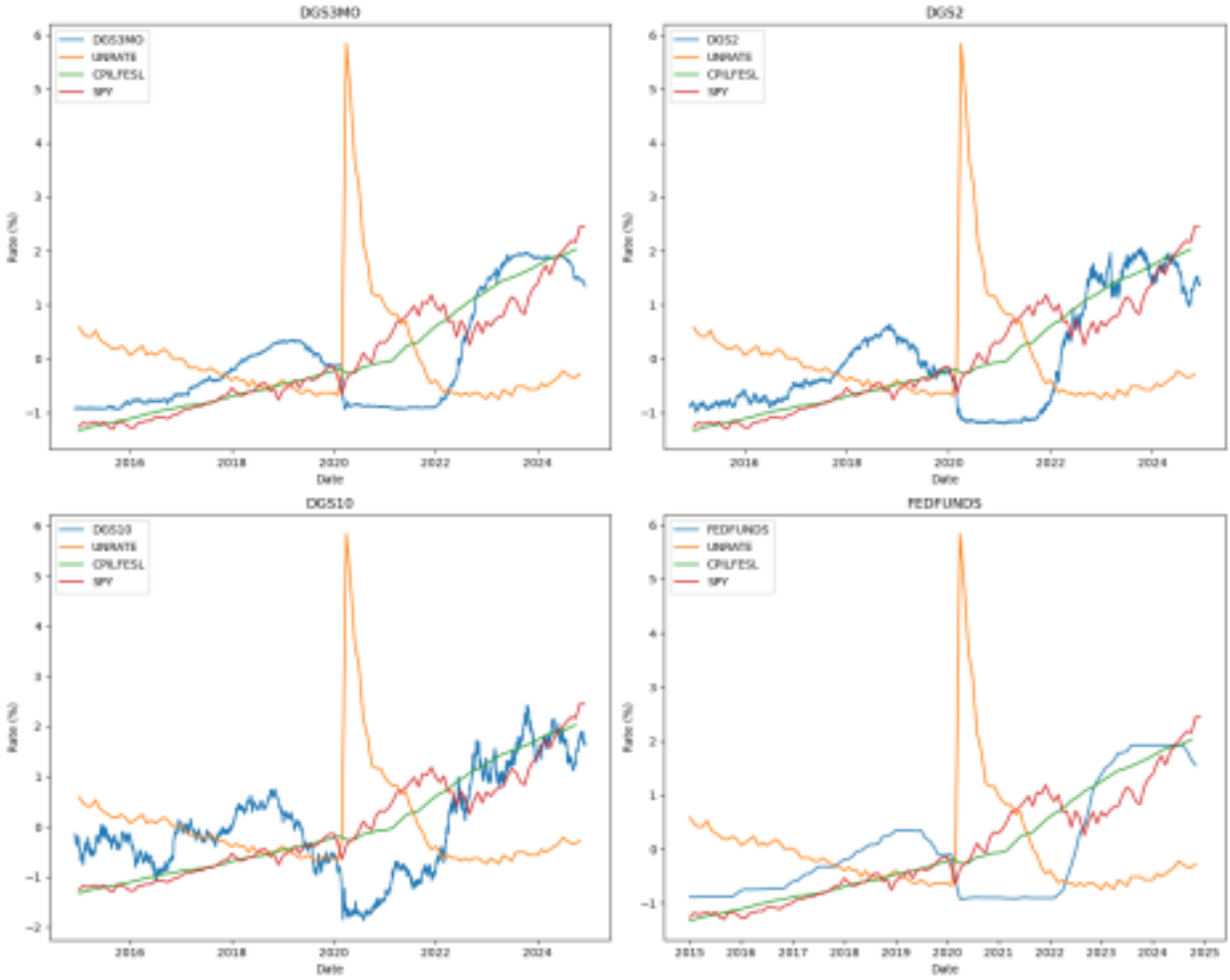

Figure 5. Macroeconomic factor correlations across treasury maturities and the federal funds rate. Each panel overlays unemployment (UNRATE), inflation (CPILFESL), and SPY against the respective yield series, all normalized to z-scores via StandardScaler, from approximately 2015 to 2025.

I came away from this section with a much deeper respect for why yield forecasting is genuinely hard, even for sophisticated market participants. The relationships are real, but they are regime-dependent and heavily distorted by outlier periods like 2020. Drawing firm conclusions would require a more rigorous modeling approach, a longer time horizon, and likely a method for controlling for structural breaks in the data. That is exactly the direction I would take this project next, alongside expanding the risk metrics layer to model how a bond portfolio's aggregate duration and convexity exposure shifts as the yield curve moves.

One design decision worth noting: the tool focuses exclusively on U.S. Treasury bonds. This was a deliberate narrowing from an earlier plan that included corporate and municipal bonds. Those instruments introduce credit risk, liquidity premiums, and issuer-specific factors that would have significantly complicated both the data sourcing and the pricing model. Restricting scope to Treasuries kept the analysis clean and focused on the macroeconomic dynamics that were the real subject of interest.

Overall, this project served its core purpose: building a hands-on, data-driven understanding of fixed income instruments using real market data and Python's scientific computing stack. The tool is functional, the visualizations are meaningful, and the process of building it left me with more questions than answers about the bond market, which I consider a success. I plan to continue expanding this project, particularly around the yield curve analysis, the modeling of macroeconomic factor relationships, and portfolio-level duration and convexity management.