SIMULATING FOREX TRADING WITH EXCHANGE RATE FORECASTS

ZACHARY EVANS, TEAYOUN KIM, DANIEL TREUHAFT, RYAN WOOD

Abstract. The foreign currency exchange (Forex) market is one of the largest financial markets in the world. Forex trading is characterized by rapid fluctuations in value. While small, the constant variations in relative currency prices provide opportunity for traders or intelligent algorithm to accrue modest profits. As a proof of concept, we pair K-Nearest Neighbors and ARIMA regression models with an investing algorithm to simulate automated trading on historical Euro-to-USD Forex data. We find that KNN models offer better-than-random predictions of future exchange rates. Additionally our automated investor, using the KNN predictor, averaged modest daily profits in simulation. Significantly outperforming both random and heuristic baselines. We analyze the above results in context of our simplifying assumptions, avenues for future research, and differences between simulation and the live market.

1. Problem Statement and Motivation

In today's era of globalization, large international corporations commonly sell products in countries far from where those products are produced. Thus, maintaining complex international supply chains requires companies to exchange large balances of one currency for another. Exchange rates vary rapidly, sometimes on the order of fractions of a second. While these variations are typically small, they translate into meaningful gains and losses when the cash quantity exchanged is very large. Strategically timing exchanges based on statistical forecasts thus offers the potential for large savings.

In this paper, we analyze Forex exchange rates between the US Dollar and the Euro. We choose these currencies because they are among the most-traded currencies on the market [1]. Other researchers have successfully considered external factors such as inflation rates, relative economic strength, and other general economic indicators to predict future exchange rates [2]. We recognize the utility offered by these approaches, but we limit the scope of this paper to solely consider previous exchange rates as predictors of future rates. Narrowing our focus affords additional time to more thoroughly explore feature engineering and hyperparameter possibilities for the models discussed below.

In summary, we focus on the following objectives:

Date: January 16, 2026.

- Analyzing historical data for seasonality and/or other relevant trends
- Predicting upcoming exchange rates, given previous rates
- Simulating automated trading to measure the profitability of our approach

2. Data

When looking for data, we wanted to make sure our dataset had information about the exchange rates of Euro to USD. We eventually landed on histdata.com as a free site that provides exchange rates for several currencies. The site says that "We're a group of traders and strategy developers that also need to use historical Forex data to develop our own trading strategies and [expert advisors]. These data is then shared for free to help you out testing your own/bought strategies and expert advisors" [1]. This provides some validity since this the providers of the data are also using it. The biggest benefit of the site though was it claimed to have exchange rates for several countries for every minute since the year 2000. The dataset provided 8,380,434 data points, with each point containing the following values: Date, Open, Close, Max, and Min. These are defined as follows:

- Date/Time: The date and time that was used to define when these data points take place at
- Maximum: The maximum exchange rate that was hit within the minute
- Minimum: The minimum exchange rate that was hit within the minute
- Open: The initial amount you could trade USD to EURO at the beginning of the minute
- Close: The final exchange rate with which you can trade USD to EURO at the end of the minute

One of the first things we noticed about this dataset was that the time increments between points were not constant. Some of the points were for adjacent minutes, while other points had gaps of over an hour between them with no data in between. With a rough estimate of how many data points there should be in the total time interval, we found that we had data for about 68% of the minutes over the time span covered. Due to the uneven spacing between the points, we decided to linearly interpolate between points in order to avoid dropping almost all of the data. We also realized that it made sense to train using only the average price for each minute instead of trying to predict all four values. We estimated the average price during each minute to be the average of the open and close price, which we thought was reasonable because there generally wasn't much variance between those two values. This also resolved issues of the open price of one minute not matching the close price of the minute before. Our interpolated averages were then created by linearly interpolating from the close value of the previous known point to the open value of the next known point. We also interpolated the

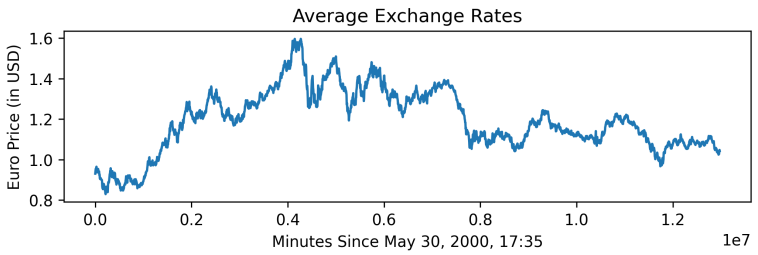

FIGURE 1. The average cost of a Euro in USD in each minute from 2000 to 2025. The maximum and minimum prices follow the average price closely.

minimum and maximum rates for each minute by just taking the lower of the minimums around each unknown point as its minimum, and similarly for maximums. Our time scaled average data points are plotted in Figure 1.

Our training, validation, and testing process will be as follows:

(1) First, we choose a slice of data to use. This is partly because our entire dataset is often too big to work with, but also allows us to test in both stable and unstable regions of the market.
(2) To avoid cross contamination, our split will be taking the first 70 percent of points as the training set, the next 15 percent as the validation set, and the last 15 percent as the final test set.
(3) Models with varying hyperparameters are trained on the training data only, and then their predictions are compared on the validation dataset using their MSE or AIC, depending on which makes sense for each model. Then, the hyperparameter combination that does the best is used for the final testing model.
(4) For final testing, we will train on both the training and validation datasets, and then predict over the test set. Instead of using the metrics from the validation section, we compare how well the models do when we use their predictions to decide when to exchange money.

3. METHODS

3.1. **Classical Decomposition.** The first method we applied to the data was a classical decomposition. This consists of deciding on a period and then splitting the data into trend (T(t)), seasonal (S(t)), and remainder (R(t)) components. Our main goal with this method is to look for seasonality in the data, which can help us identify long-term patterns in the data. Using these, we might be able to find a specific time each hour, day, month, or year in which trading rates are generally better or worse, which would help for all three of our main objectives.

With a chosen period τ, the trend is calculated using a moving average with diameter τ around each data point V_k, representing the average

exchange rate in minute k. After calculating the trend, we calculate the detrended data values as $W_k = V_k - T_k$, and use these to calculate the seasonal portion of the data. The seasonal portion is calculated by splitting the detrended data into sections with length τ, and then S_j is the average of the jth index of each section for each $j \in \{0, ..., \tau - 1\}$. The remainder R_k is then given by $R_k = W_k - S_j$, where $j = k \mod \tau$.

For this analysis, we chose periods of one hour, one day, one month, and one year to look for seasonal components on each of these time scales. If we could find strong evidence of seasonal behavior for any of these periods, it would help to answer the question of when the best time to exchange currency is in the case that we only want to exchange once.

3.2. **ARIMA Modeling.** Autoregressive Moving Average (ARMA) models are a staple of statistical time series forecasting and thus are a natural approach to our problem. Additionally these models can encapsulate all of the information contained in Kalman Filters and HMMs, so fitting with ARIMA models makes these models obsolete. The Wold Theorem guarantees that all covariance-stationary time series can be modeled well with an ARMA model. There is good reason to believe our dataset is not covariance-stationary - even on a daily time scale, economic data often includes trends. Consequently, we applied Autoregressive Integrated Moving Average (ARIMA) models in the hopes that differencing might de-trend the data and render the resulting data approximately stationary.

Model fitting was performed using the `innovations_mle` fit algorithm of the `statsmodel.tsa.arima.model.ARIMA` class. We referred to the Akaike Information Criterion (AIC) score, which penalizes both final loss and parameter complexity, as part of model selection. Before each prediction, we trained an ARIMA model on 120 consecutive minutes of data chosen from a random initial index. After comparing AIC scores, we then had the selected ARIMA model generate a forecast over the subsequent 10 minute horizon and compared its results to the actual exchange rates. We repeated this process for random 130-minute slices of the dataset to estimate the average accuracy of our process.

The decision to restrict each model to train on only 120 minutes prior to the prediction horizon was made primarily in order to reduce training time, yet we also hypothesized that this would not negatively impact the model's performance because data further in the past would quickly become less informative. We selected a 10-minute horizon after examining graphs that appeared to indicate that the relative direction (increase/decrease) of the actual price more often matched the prediction after 10 minutes than after 1 minute.

3.3. **KNNR.** As an alternative to ARIMA models, we trained a K-Nearest Neighbors Regression (KNNR) model. This model is an extension of the normal K-Nearest Neighbors method that we trained on small windows of 6 consecutive data points. In order to predict a future value, the model

compared the previous 5 points to other 5-tuples it had previously seen in the dataset, found the k most similar windows, and used an average of the final points for each of those windows as the predicted next value.

For this method, we did the standard train-validate-test split mentioned in the data section, where the hyperparameters we searched over were both the window size and the number of nearest neighbors to use in the averaging. However, KNNR typically only finds one new point at once, so for the validation and test set we did two different tests. In one, we used its predicted next value to predict future values, generating predictions over the entire validation or test set from just the first n data points. This test would tell us how well we could predict the best exchange rate over a larger future period. In the other test, we predicted one new point at a time, and then updated our window with the actual data values to predict the next point. This predicts only the next minute from each minute in the validation or test dataset, which is useful to know if we want to use many exchanges to generate profit.

4. Results and Analysis

4.1. **Classical Decomposition.** After testing various periods in our classical decomposition, we found limited evidence of any seasonality in our data. Using a year-long period, the decomposition of the data can be seen in Figure 5. We first look at what the expected seasonal component would be with this amount of error if the data had zero natural seasonal component. Since we have about 20 years of data, the values in the seasonal component are found by averaging 20 de-trended data values. Assuming the remainder values ε are independent and approximately normally distributed with mean 0 and standard deviation 0.03 (since nearly all remainder values are between 0.1 and -0.1), and that S is entirely made of remainder terms, we have $\mathbb{E}[S] = 0$, and $\mathrm{Var}(S) = \frac{1}{20}\mathrm{Var}(\varepsilon)$, so the expected standard deviation of the seasonal component is $\sigma_S = \frac{1}{\sqrt{20}}\sigma_\varepsilon = 0.007$. Nearly all of our measured seasonal components are within two standard deviations using this value, so there is no statistical evidence that any of our measured seasonal component is more than just error. We found that similar analysis using different seasonal periods got similar results, so we proceed assuming that there is negligible seasonality, or that seasonality is impossible to distinguish from the remainder.

This result makes predicting the optimal exchange rate over a large time interval much more difficult than we had initially hoped, since there are no times of the day, month, or year that we can tell are generally better than others.

4.2. **Hypothesis Testing.** Where applicable, we performed one-sided t-tests to evaluate our results. We identified 4 null hypotheses to test a priori, as follows:

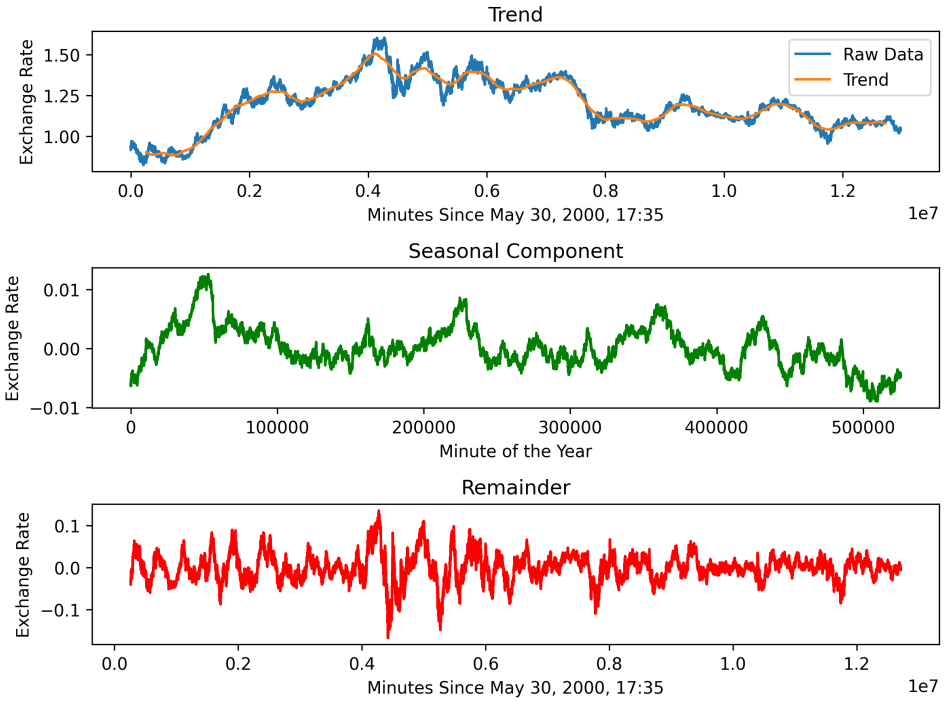

FIGURE 2. The classical decomposition of the average exchange rate into trend, seasonal, and remainder components. Note that the magnitudes of the seasonal part are around a tenth or the magnitudes of the error, indicating low seasonality.

(1) The ARIMA model predicts whether the exchange price in 10 minutes will increase or decrease with 50% accuracy
(2) The KNN model predicts whether the exchange price in 1 minute will increase or decrease with 50% accuracy
(3) The average daily profit from trading with an ARMA model is higher than that of our heuristic baseline
(4) The average daily profit from trading with a KNN model is higher than that of our heuristic baseline

We selected a significance level of $\alpha = 0.05$ a priori, but to account for multiple comparisons, we applied the Bonferroni correction. As a result, each individual t-test was compared against the threshold $\alpha_{bonferroni} = 0.0125$.

4.3. **ARIMA.** ARIMA testing yielded mixed results. Over a sample of 100 random 130-minute segments of the dataset, the ARIMA model trained on the first 120 minutes correctly predicted whether the exchange rate at the 130th minute would increase or decrase 60 out of the 100 times. See Figure 3 for ARIMA's prediction on two of the random 130-minute slices.

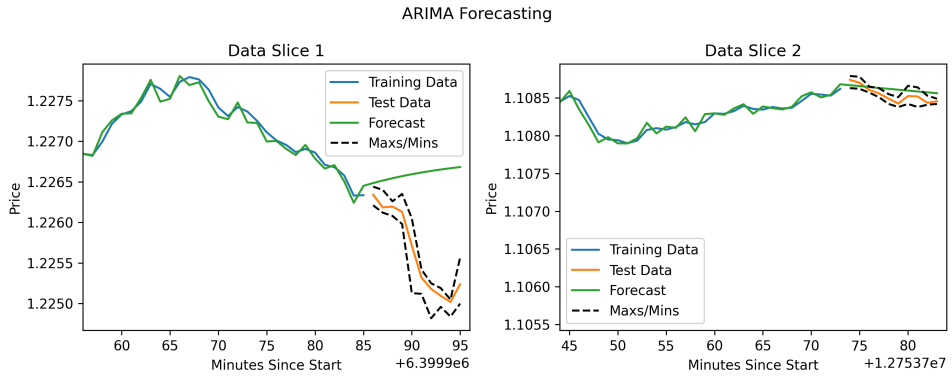

FIGURE 3. Estimates made by our ARIMA model on the last 40 points on two of our data slices. The last 10 points are the predicted values. The first slice is in a slice of high market instability, and the model does poorly as a result. The model does much better on the second slice, and we found that it chose the correct direction about 60% of the time.

We performed a one-directional t-test to investigate the null hypothesis that the ARIMA model correctly predicts the direction of upcoming price changes only half the time. At 0.0225, the above p-value is less than our original significance threshold of 0.05, but it is higher than our Bonferroni-adjusted threshold of 0.0125. As such, the data is inconclusive and insufficient to reject the null hypothesis. We attempted to simulate live trading with our ARIMA models to further explore the model's efficacy; see Section 4.5.

4.4. **KNNR.** Results for the KNN regressor were more conclusive. When limited to predicting upcoming exchange rates one minute at a time, the KNN regressor performed well, with mean absolute error (MAE) of $3.83 \cdot 10^{-5}$. Unlike the ARMA model, the accuracy of KNN predictions deteriorated rapidly as the prediction horizon increased past one minute.

After training the KNN on the training set obtained from our train-validation-test split, we evaluated whether it could correctly predict whether the price would increase or decrease at 1000 random data points from the validation set. Computing a one-direction t-test on this sample against the null hypothesis that the KNN's predictions were not better than random yielded a p-value of $2.87 \cdot 10^{-35}$. We thus reject the null hypothesis in this case.

4.5. **Investor Class.** Ultimately, the success of economic time series forecasting can be concretely measured by whether the method is profitable in a trading scenario. Since genuine trading would be ill-advised at this stage, we instead simulated automated day-trading scenarios where our predictions

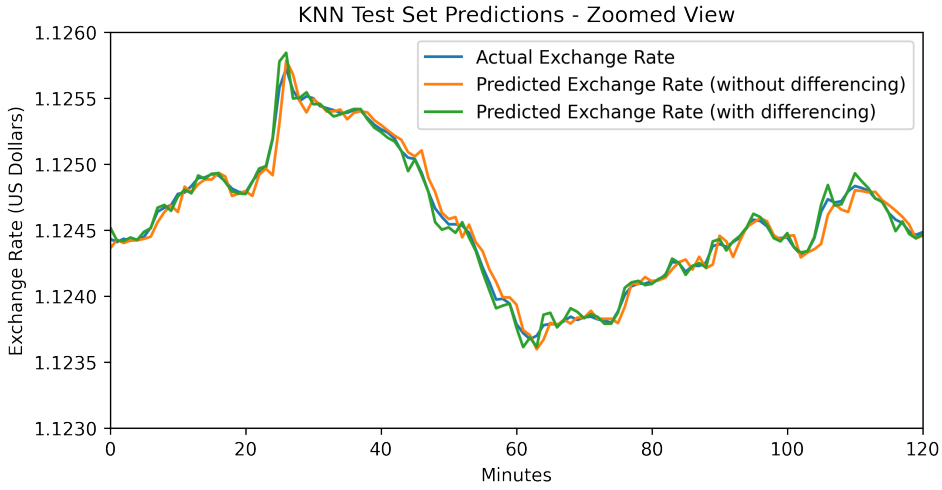

FIGURE 4. Predictions from KNN plotted against the true values. We plot how effective KNN predictions are on the raw data and on the first difference of the data. This plot shows that both sets of predictions are good, but the predictions on the first difference are slightly better.

determined purchasing and selling decisions. Applying principles of object-oriented programming, we implemented an *Investor* class that accepts both *Strategy* and *Forecaster* class objects in its constructor. This code framework allowed us to flexibly experiment with different combinations of models and strategy algorithms.

After some experimentation, we settled on a straightforward trading strategy for our tests: If the forecaster predicted the upcoming price of the Euro would increase more than \$0.0001, the investor would purchase as many Euros as possible given its current balance. Conversely, if the forecaster predicted a drop in the Euro-to-USD exchange rate exceeding \$0.0001, the investor would attempt to purchase as many dollars as possible. Otherwise, the investor declined to trade, let the minute elapse, and generated another forecast.

We paired this strategy with ARIMA and KNNR models as Forecasting objects. Each investor was assumed to begin with an initial account balance of \$1 million USD. We then allowed the KNN investor to simulate trading on 1000 random 480-minute segments of the test (not validation) dataset. We measured the model's profit or loss at the end of each 480-minute "trading day". Due to computational constraints, we only simulated 40 trading days with the ARIMA-augmented investor.

An important thing to notice is that overall, the price of the Euro against the US dollar rose more often than it fell in our dataset. Consequently, assuming no transaction fees, an investor who begins with a balance consisting

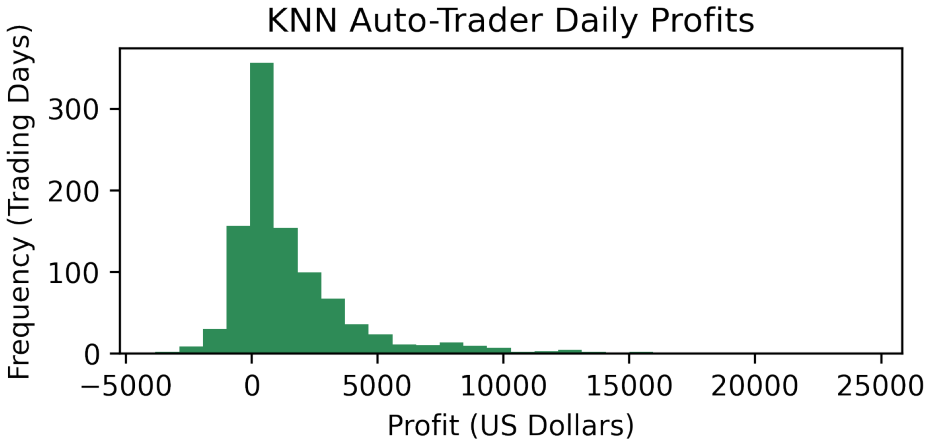

FIGURE 5. Approximate distribution of daily KNN trader returns. Key features are that the histogram is right-skewed and that while small losses are reasonably common, large losses are rare.

of US Dollars would, on average, gains a positive return even if they traded randomly over this time period.

We mitigated this confounding influence by comparing our automated traders' performance against two baselines: an investor that traded randomly throughout the trading day, as well as an investor that bought Euros at the beginning of the day then traded back to US Dollars at the end of the day.

The KNN-augmented investor averaged a daily return of \$1544.81 US Dollars. Notably, over the 1000 480-minute segments included in this simulation, the price of the Euro increased by about \$0.0004 each day. This led to the baseline investor that bought all Euros at the beginning of each day to also average a positive daily profit, at 103.90 a day. Even after adjusting for multiple comparisons, the difference between these means is significant; the corresponding t-test returned a p-value of $5.75 \cdot 10^{-52}$.

The distribution of daily returns obtained by the KNN investor is right-skewed; see Figure 5. Sensitive to the possibility that the few unusually large returns might be artifacts of the data or otherwise less likely to occur in the live market, we verified that the median daily return was \$703 US dollars. Thus even after excluding outliers, the KNN investor still offers robust performance.

Unfortunately, the investor that used our ARIMA model as a forecaster failed to return a positive daily profit and performed marginally worse than random trading. On a principal of \$1 million USD, it lost an average of \$23 dollars per day.

5. Ethical Implications and Conclusions

There are several reasons why our model may not be valid. After talking with an analyst at Goldman Sachs about currency exchange, we realized that had assumed that our data is continuous, but it actually makes discrete jumps every time a new price is offered. This assumption that we made may affect how our models were trained, which may in turn affect how our models turned out.

Additionally, we have made the assumption that there are no transaction fees when exchanging currency. In reality, most exchange places require you to make an exchange fee [3]. Due to small market fluctuations, even minor exchange fees can significantly impact profits. Combined with model uncertainty, the feasibility of actually making money off of the foreign exchange market is limited.

Another problem with our model is that even if we had a working model, as we used the model to make money, our transactions could cause a change in the market. This would introduce a new factor that goes into future exchange rates that our model has not been trained to account for, and thus might cause its predictions to fail more frequently than we are currently predicting.

Furthermore, there may be ethical concerns associated with publicly sharing the model's predictions. Since the model's forecasts are not perfect, others should not use its predictions to determine their financial decisions. If used in this way, it could lead to unintended and potentially harmful consequences for a country and those individuals.

6. Conclusion

Our work suggests several avenues of further research. First, we could develop more sophisticated trading strategies, perhaps drawing from financial research literature, to make better use of our forecasts. Similarly, we could incorporate data about inflation rates from each country as well as a few economic indicators to make better predictions. Second, it may be desirable to estimate risk by forecasting upper and lower bounds on the exchange rate, helping investors know how much they stand to lose. Third, trading strategies could be altered to account for transaction processing fees, which would necessitate fewer, higher-value trades to remain profitable.

Despite these limitations, our work demonstrates that when paired with careful automated training, statistical forecasting tools such as ARIMA and KNN models can generate modest returns on large account balances, such as those held by publicly traded firms.

References

[1] "HistData.com – Free Forex Historical Data." Histdata.com, 2020, www.histdata.com/. Accessed 12 Apr. 2025.

[2] Nguyen, Joseph. "Learn 3 Common Ways to Forecast Currency Exchange Rates." Investopedia, Investopedia, www.investopedia.com/articles/forex/11/4-ways-to-forecast-exchange-rates.asp. Accessed 12 Apr. 2025.

[3] Konovodoff, Alexis. "Best Places to Exchange Currency without Paying Huge Fees." Wise, Wise, 24 Dec. 2024, wise.com/us/blog/the-best-places-to-exchange-currency-without-huge-fees.

[4] Treuhaft, D, Evans, Z, Kim, T, Wood, Ryan. "Github of Currency Exchange Programs" 2025, https://github.com/danman778/CurrencyExchange. Accessed 13 Apr. 2025