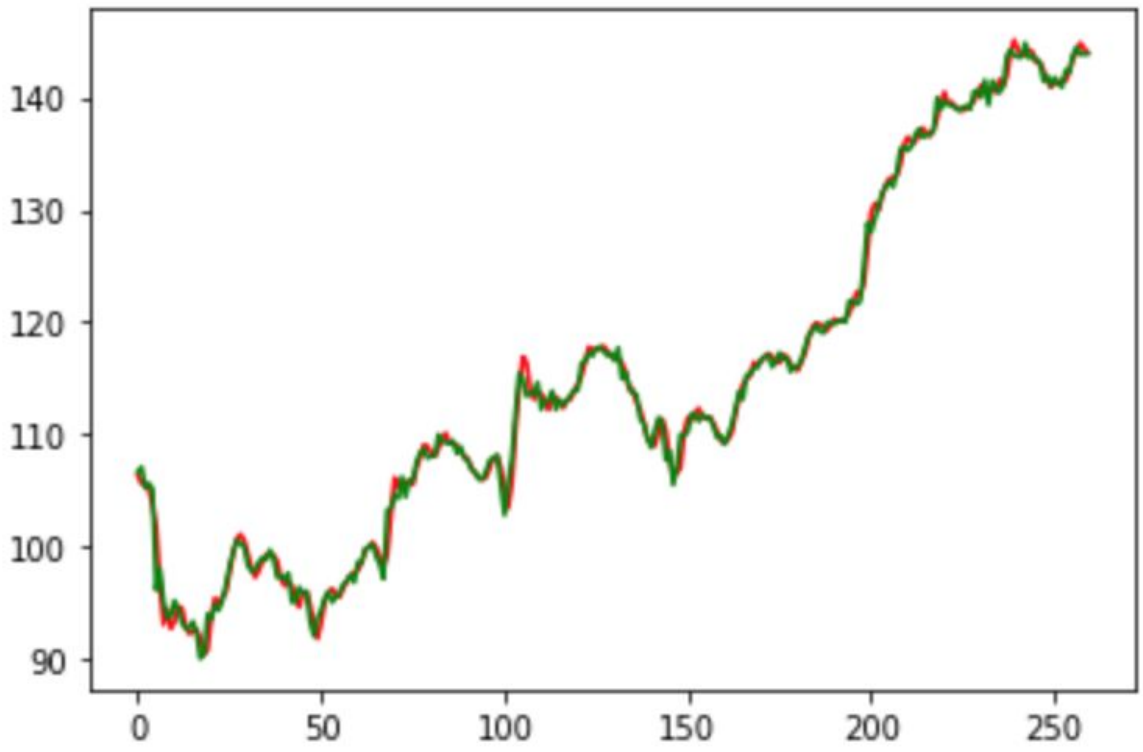
Andy Liu - July 30, 2020

(Paper - I found a free PDF with the full text, which I'll be uploading to this folder shortly)

This paper compared a variety of Wavelet Transform + ARIMA predictive models on the task of predicting future commodity prices for a variety of metals. Its forecasting framework first used a wavelet-based MRA (multiresolution analysis) component, which was based off of a variety of discrete wavelet transforms (DWT) and maximal-overlap discrete wavelet transforms (MODWT), each of which used a different wavelet or number of layers. This produced a "smooth series" (the smoothed data) and a set of "detail series".

Future data was predicted by training separate ARIMA models on the smooth series and each of the detail series, then adding up the predicted values for each of the series, thus obtaining a forecast of the original time series. I am currently working on implementing a similar algorithm for stock market prices. The paper found that, generally, MODWT is superior for this application as compared to DWT, and was able to obtain high accuracies with sufficient layers and a MODWT-Arima model.

I was able to replicate the MODWT-ARIMA model on Apple daily data from 2000-2020. I found success with this model, achieving a MSE of 0.936, and an MAPE of 0.608%, by implementing their predictive model in Python. Below is a graph containing the actual (green) and predicted (red) values over the entire test period. I used a (10,1,1) ARIMA on the smooth series and a (1,1,0) on the detailed series (parameters were computed using autocorrelation and partial autocorrelation tests) before successively training the ARIMA and using it to make next-day predictions.



August 4, 2020

I have recently been able to use the Python AutoARIMA library to facilitate the training of this model. My updated work has been committed to my Github repo.

The AutoARIMA library selects **p** and **q** parameters for an ARIMA model by calculating the AIC (Akaike information criterion) for each set of parameters and selecting the combination that yields the lowest AIC. AIC is calculated to be -2(log-likelihood) + 2K, meaning that better-fitting, less complex models are favored by AutoARIMA. The **d** parameter must be selected manually; I did so by running the KPSS and ADF stationarity tests to figure out how many times the data needed to be differenced before becoming stationary, selecting the larger of the two outputs from our tests. I periodically recalculated ARIMA parameters with AutoARIMA

in order to maintain an updated model, before carrying on as before. Other than the usage of AutoARIMA, I used the same algorithm to generate one-day-ahead forecasts on the test dataset. This model yielded positive results as well, with a MAPE of 1.59% and a RMSE of 4.35 (this did require a few anomalies to be removed from the end of the test set, however).

One issue I found with these models is how computationally expensive AutoARIMA can be. Combined with having to update the model daily with new data, this model may not be able to deal with minute resolution data.