[36-471] Final Project Report

Vihan Karnala, Arnav Paliwal, Ryan Zhang, Alan Zhu

April 2024

1 Executive Summary

The soybean industry in the United States is a $100+ billion industry with many stakeholders, from farmers to processors to consumers. In particular, soybean futures are a crucial instrument, indicating the health of the industry and allowing stakeholders to hedge their products. We are interested in better understanding how soybean futures change over time, particularly after taking climate data into account.

We obtain soybean futures pricing data from 2004 to 2024, as well as climate data from a representative location for soybean farmers. We then fit various models to our data, from univariate ARMA and GARCH models on soybean futures returns to multivariate time series regression and VAR models that include climate data.

We found that the volatility in futures returns is well-modeled by a GARCH model. However, precisely and accurately predicting future returns was difficult due to high noise and low signal in our data. We believe more complex models and/or more variables would be needed to produce models that would be useful in terms of predicting future returns.

2 Introduction

The soybean industry is a large contributor to the US economy, with an impact of $124 billion [United Soybean, 2023]. The US also accounts for nearly 30% of the world's soybean production [USDA, 2024b] and exports nearly $30 billion worth of soybeans annually [USDA, 2024a].

Within the soybean industry, futures are a crucial financial instrument for producers, consumers, and traders. Soybean futures allow farmers to obtain guaranteed prices on their products, hedging against sudden dips in the market, and lets purchasers hedge against spikes. Thus, we aim to gain a better understanding of how the price of soybean futures changes in order to help both producers and consumers make the most out of the market.

To address this question, we will begin with univariate analysis on soybean futures prices, aiming to model the changes in price with just historical price data. Next, we will investigate whether other data, specifically climate data, can help improve our model. Our inclusion of climate data is motivated by the fact that soybeans, like all plants, grow best when the weather is most suitable for its growth.

2.1 Data

We use data collected over 20 years, from March 31, 2004 to April 1, 2024. For future prices, we obtain closing prices for each trading day from Investing.com [Investing.com, 2024].

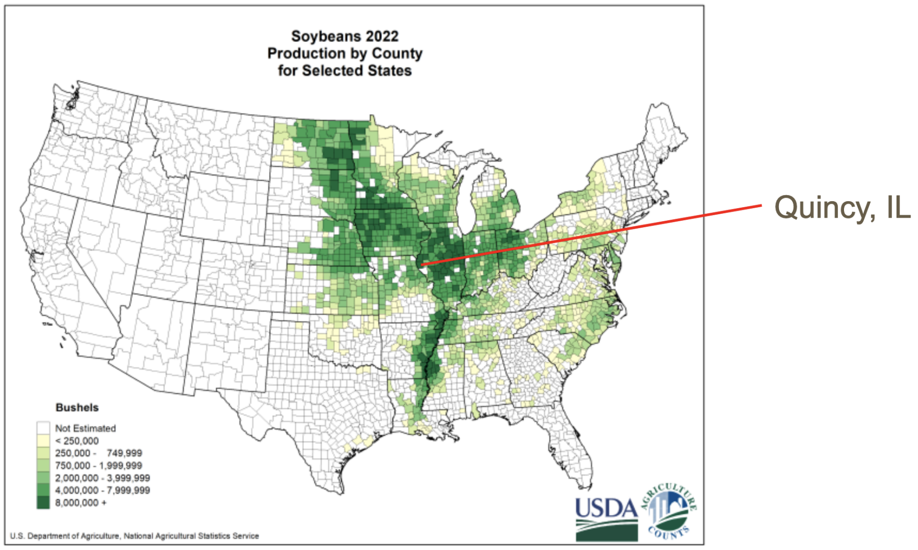


Figure 1: Production of soybean in the United States, and location of Quincy, Illinois. Map from United States Department of Agriculture [USDA, 2023].

We chose to use climate data from Quincy, Illinois. As seen in Figure 1, western Illinois is close to the center of the regions where soybeans are grown in the US. Quincy is home to a regional airport in western Illinois, allowing us to access high quality climate data.

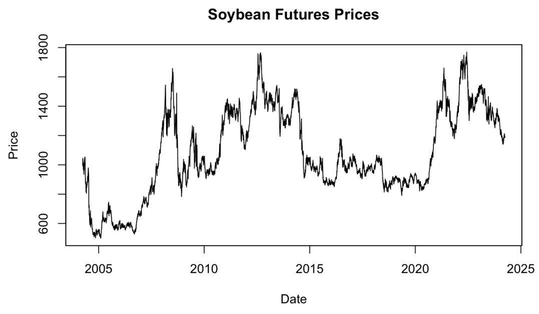
In particular, we collected humidity, temperature, and precipitation data. Hourly humidity data was obtained through the Open Meteo API [Zippenfenig, 2023] and aggregated into daily averages, daily temperature and precipitation data was obtained through the National Oceanic and Atmospheric Administration's Applied Climate Information System [NOAA, 2024]. Trace amounts of precipitation were interpreted as 0 precipitation, and missing values were filled in through interpolation between last and next known values.

Finally, we excluded all non-trading days from the data to ensure we would have have any missing price data. Exclusion is preferable to forward filling the previous trading day's closing price in order to maintain equivalence in the days of the week; forward filling would increase the influence of the Friday closing price.

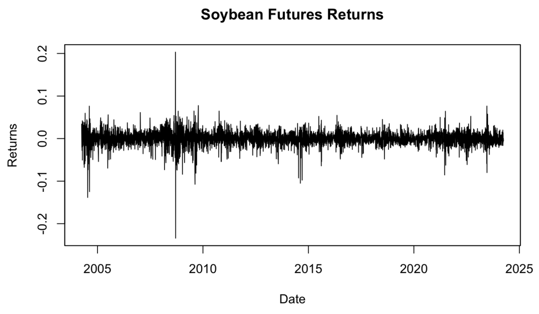
3 Methods

The methods section proceeds as follows: 1) we discuss the soybeans futures prices time series and justify transforming it into daily returns, 2) we discuss the univariate models we fit, and 3) we discuss the multivariate models we fit.

3.1 Returns



(a) Soybean futures prices.



(b) Soybean futures returns.

Figure 2: Plots of the soybean futures prices and returns from March 2004 to April 2024.

A plot of the soybean futures price time series is presented in Figure 2a. This time series is non-stationary, exhibiting a slight overall upward trend and regular smaller trends. Following standard practice [Tsay, 2005], we instead use daily returns, which is calculated as the change in price divided by the current price (and may be approximated by differencing the logarithm of the price).

A plot of the returns time series is presented in Figure 2b. There is no discernible trend, either global or local, though there appears to be some heteroscedasticity, best evidenced by the large positive and negative spike at around 2008. This will inform our univariate modeling going forward.

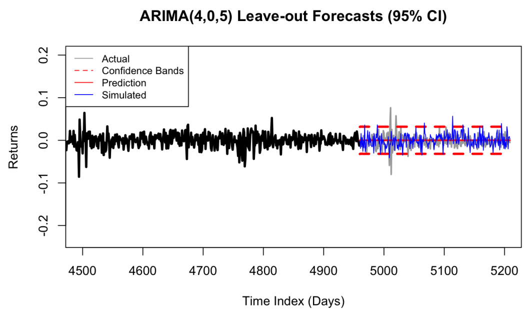
3.2 Univariate Modeling

3.2.1 ARMA

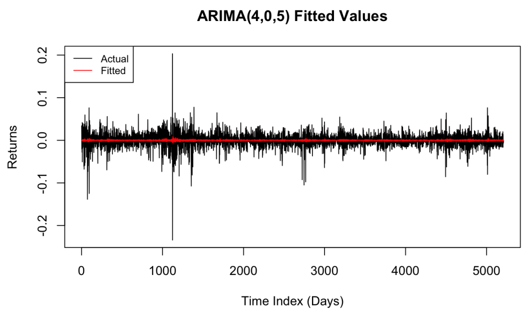
We begin by fitting an ARMA model to the returns time series. `auto.arima()` selects an ARMA$(4, 5)$ model using AIC. However, the quality of the model is poor: simulations from the model (Figure 3a) do not achieve the same variances in magnitude as the actual returns time series, and the fitted values (Figure 3b) are close to zero and do not follow the returns. Analysis of the residuals (Figures 3c and 3d) indicates that though they are plausibly white noise, they do not appear to be normally distributed, having a heavier tail. The residuals also appear to be heteroscedastic, with large-magnitude residuals clustering near each other. For these reasons we believe the ARMA fit is poor and will focus on other models in following sections; nonetheless, for completeness, we present limited discussion of the results here and include the fitted coefficients in Appendix A in Table 6.

3.2.2 GARCH

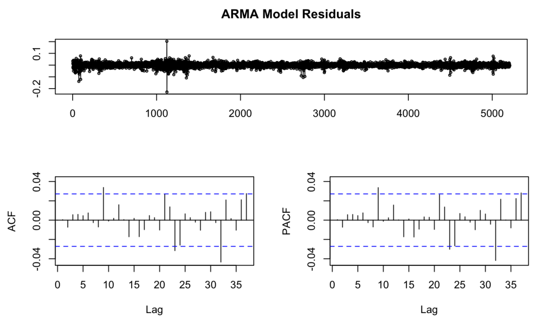
Our main univariate model is the GARCH model, with which we aim to account for the heteroscedasticity in both the returns and the residuals through fitting a GARCH model. For a GARCH model to be appropriate for the returns, we need that the returns are white noise and that the squared returns follow some ARMA process. Examination of the ACF and PACF plots of returns and squared returns (Figure 4a) indicate that the returns are plausibly white noise (no



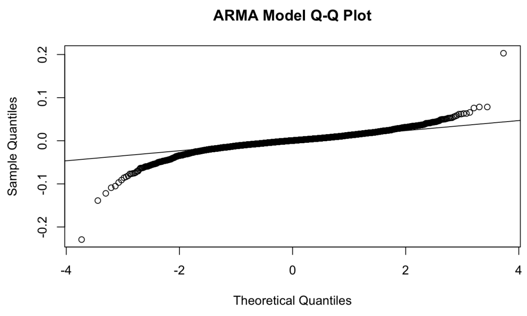
(a) Simulation plot.



(b) Fitted vs. actual plot.

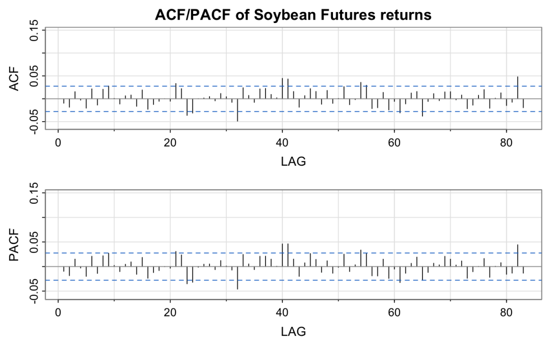


(c) Residual plots.

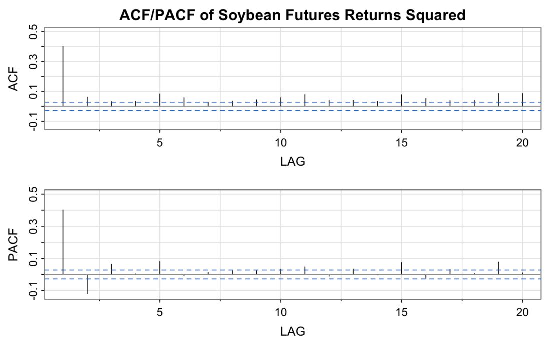


(d) Residual Q-Q plot.

Figure 3: Forecast and residual plots of an ARMA$(4,5)$ model fitted to returns.

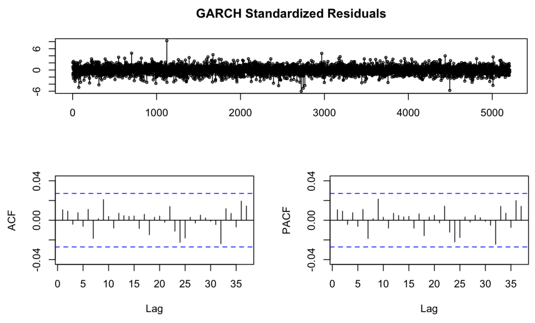


(a) ACF and PACF of returns.

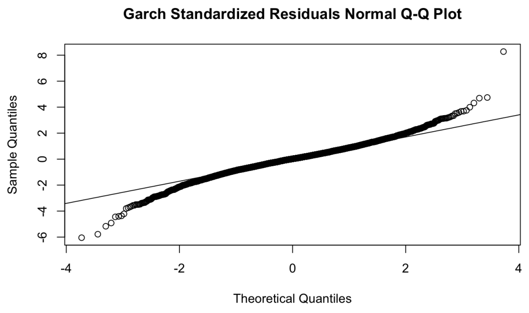


(b) ACF and PACF of squared returns.

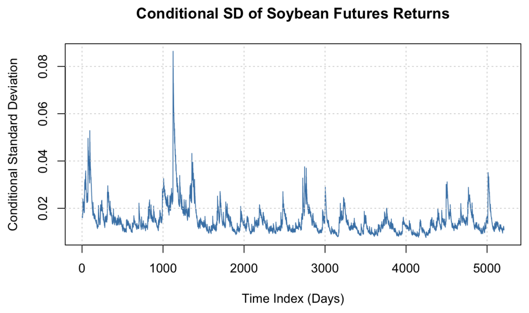
Figure 4: ACF and PACF plots for fitting a GARCH model to returns.



(a) Residual plots.



(b) Standardized residual Q-Q plot.



(c) Conditional standard deviation plot.

Figure 5: Residuals and conditional standard deviation of GARCH fit.

Order	AIC
$(1,1)$	-5.6849
$(1,2)$	-5.6855
$(2,1)$	-5.6843
$\mathbf{(2,2)}$	$\mathbf{-5.6857}$
$(1,3)$	-5.6855
$(3,1)$	-5.6839
$\mathbf{(6,4)}$	$\mathbf{-5.6857}$

Table 1: AIC for GARCH models fitted on returns.

large ACF or PACF values). Rapid decay in the ACF and PACF of the squared returns (Figure 4b) indicates that an ARMA mode could be suitable for the squared returns.

We used `auto.arima()` to fit an ARMA model to the squared residuals, and identified an ARMA$(6,4)$ model. This tells us that the returns follow a GARCH$(6,4)$ model. However, we are most likely over-fitting to noise since this is very high order, so we instead examined the AIC of lower order models.

From the AICs (Table 1), we identify a GARCH$(2,2)$ model, which had an AIC that was rather close to the AIC of the original GARCH$(6,4)$. We thus fit a GARCH$(2,2)$ model.

Diagnostics on this model indicate the fit is good: the standardized residuals (Figure 5a) look to be plausibly white noise with no heteroscedasticity. Moreover, the residuals are now more normal than in the ARMA fit (Figure 5b). Finally, a quick look at the plot of the conditional standard deviations (Figure 5c) reveals that the conditional standard deviations are larger precisely where there is higher variance in the returns (Figure 2b). We will thus discuss this model further in the Results section.

3.3 Multivariate Modeling

In multivariate modeling we aim to include climate data to better model returns by using all of the available information that we have access to. This additional information is the weather data that we can lag and then incorporate into our model for futures returns. To determine the appropriate lagged climate data to use, we examine the AIC.

3.3.1 Time Series Regression

We aim to fit our Time Series Regression model using lagged climate data and also lagged returns data. This provides the regression model to add extra information to the information that the ARMA model we fit earlier. In order to fit such a model, we used the `arima()` function with the lagged regression being put into the `xreg` term. By comparing AIC values (Table 2), we chose a lag of 2. Though a lag of 1 had a lower AIC, we wanted to have more lagged data to work with, and so we chose the lag with the next lowest AIC. This gave us an ARMA(2,1) fit.

Then, we aim to check the residuals to evaluate our models. From Figure 6a, we can see that the residuals are heteroscedastic. This means that the IID assumption for the residuals is not held up in the time series regression which suggests that the assumptions for the standard errors the model produces are not met. In order to account for this, we want to try fitting a GARCH model. In order to check if a GARCH model really makes sense, we analyze the square residuals of the

Lag	AIC
1	$\mathbf{-21322.23}$
2	-21314.17
3	-21311.8
4	-21308.46
5	-21310.04

Table 2: AIC for time series regression models fitted on returns with lagged climate and returns data.

time series in Figure 6b. These residuals appear to be from an ARMA process by looking at how the PACF and ACF both tail off. Thus, we fit an ARMA(4,2) process to them. This informs us to fit a GARCH(4,2) process to the residuals.

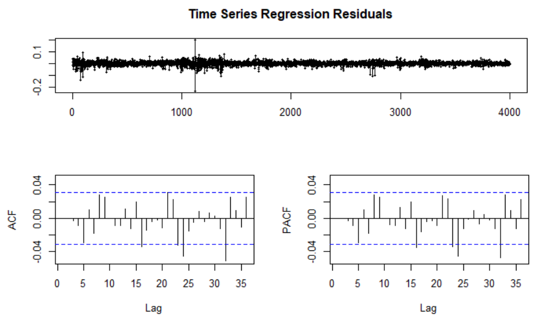
In order to check the GARCH fit, we check the GARCH standardized residuals. From Figure 6c, we can see that the residuals seem much more stationary with little to no outliers. In addition, the PACF/ACF do not violate the Gaussian white noise hypothesis. If we look at the normal QQ plot of these residuals in Figure 6d, we see that while there are slightly heavy tails they aren't that large in deviations from the QQ line. This suggests that the residuals are normal enough.

After GARCH fitting and validation, we plot the conditional SD superimposed on top of the original residuals in Figure 6e, and we can see that the volatility in the residuals is well captured by the GARCH model. We will use this captured volatility to inform our error predictions around our estimates from the time series regression model in the results section down below.

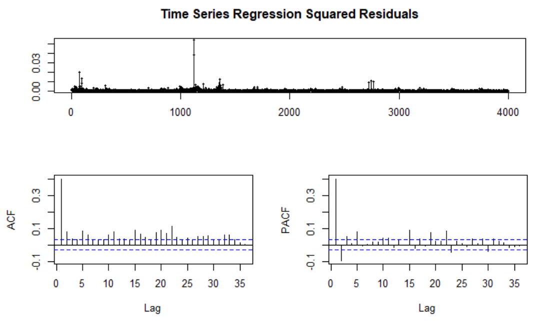
3.3.2 VAR

One of the drawbacks of the Time Series regression is that the interdependencies between weather variables are not accounted for. In order to account for these interdependencies, we fitted a VAR model using the `VARSelect()` function in R. We found that a lag of 5 was best at maximizing HQC, we chose this because different information criteria outputted lags of 3, 5, and 11 and we wanted to select a middle ground for optimizing the trade off between information and parsimony. Afterward, we looked at the residuals in Figure 7a and saw that there does seem to be heteroscedasticity just like the time series regression, going to the squared residuals in Figure 7b shows a decrease in heteroscedasticity and ACF/PACF models that look like they could represent an ARMA process. We then decide to fit the residuals as GARCH as we did with the time series regression. This time, the GARCH(2,2) model was fit by inspecting the ACF/PACF of the squared residuals and checking the following AIC.

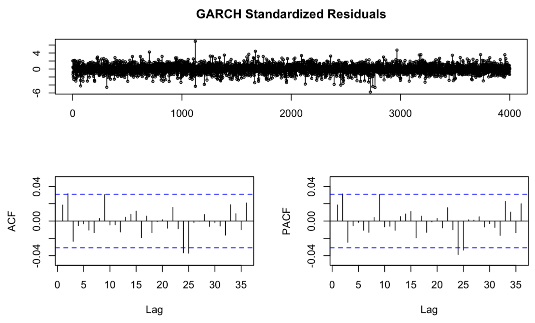
In order to check the GARCH fit, we check the GARCH standardized residuals. From Figure 7c, we can see that the residuals seem much more stationary with little to no outliers. In addition, the PACF/ACF do not violate the al white noise hypothesis. If we look at the normal QQ plot of these residuals in Figure 7d, we see that while there are slightly heavy tails they aren't that large in deviations from the QQ line. In addition, these tails seem less heavy than that from the time series regression suggesting better validity of the overall fitting. From this, we get that the GARCH is a good fit for the model. This support is even more well supported by the 2 conditional SD plot in Figure 7e where the volatility in the residuals seems well modeled by the GARCH(2,2) fit.



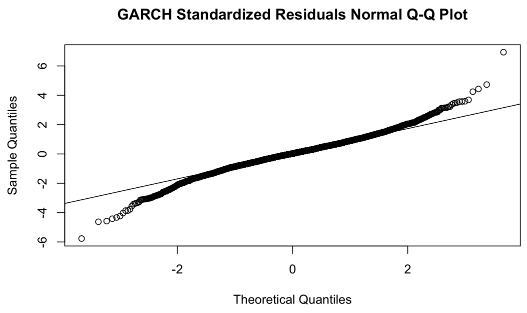
(a) Time series regression residuals.



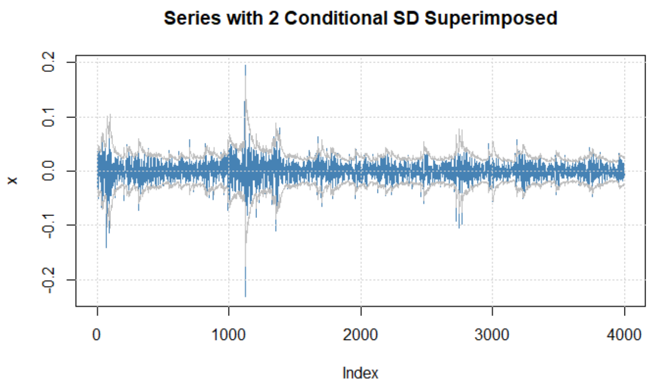
(b) Time series regression squared residuals.



(c) Time series regression standardized residuals.

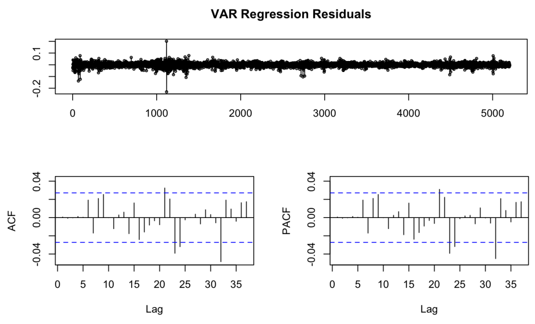


(d) Time series regression standardized residuals Q-Q plot.

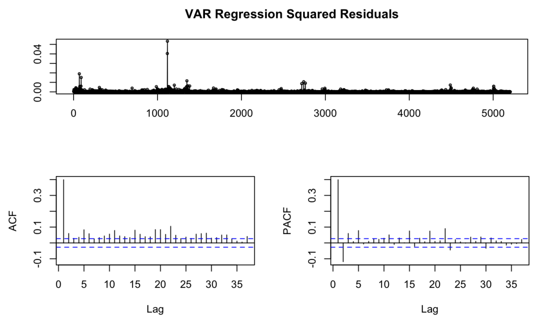


(e) Time series regression residuals with Conditional SD from GARCH$(4, 2)$ fit.

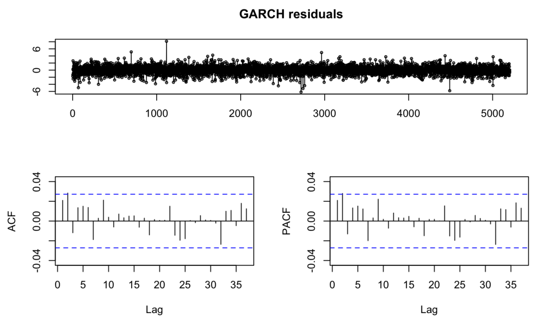
Figure 6: Residuals from time series regression and GARCH fitting.



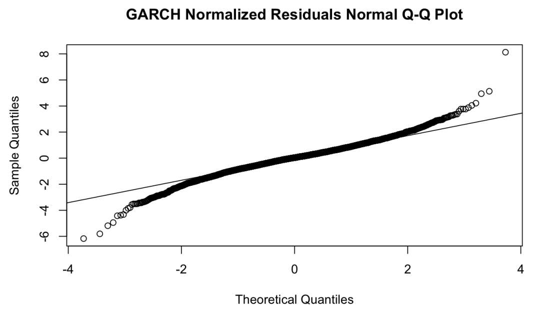
(a) VAR residuals.



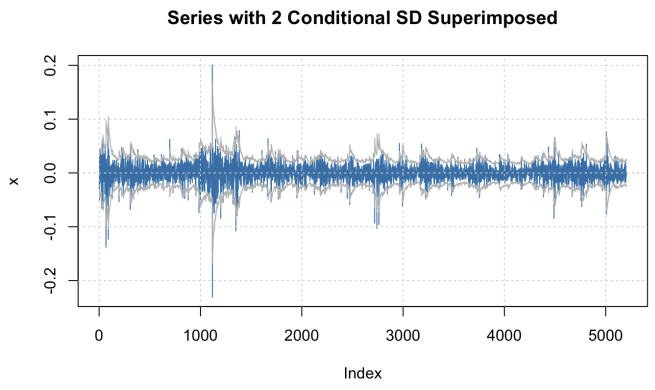
(b) VAR squared residuals.



(c) VAR and GARCH standardized residuals.



(d) VAR and GARCH Standardized residuals Q-Q plot.



(e) VAR residuals with Conditional SD from GARCH$(2, 2)$ fit.

Figure 7: Residuals from VAR and GARCH fitting.

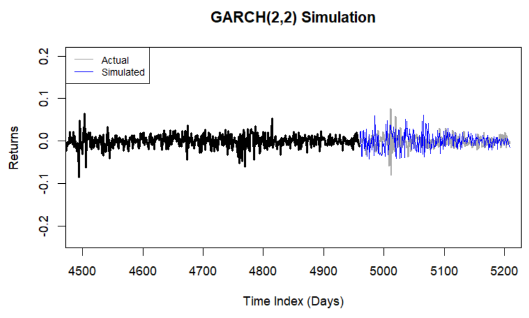
	mu	omega	alpha1	alpha2	beta1	beta2
	2.8355e-04	5.1411e-06	9.7466e-02	2.5357e-02	1.4022e-01	7.1886e-01
s.e.	1.744e-04	1.055e-06	1.306e-02	2.146e-02	1.439e-01	1.314e-01

Table 3: Coefficients for GARCH$(2, 2)$ model.

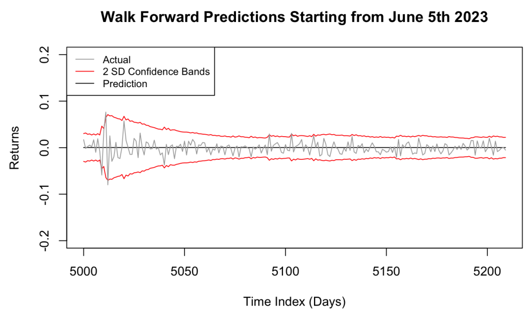
4 Results

We now evaluate the results of fitting the models. We check the quality of the fitted values, forecasts, and residuals. An ideal model would have accurate fitted values, useful forecasts and/or simulations, and roughly normal residuals. Given the difficulty of modeling the trend in financial data, we would be satisfied with a model that could accurately model the variances of the returns data (i.e., accurate simulations), but being able to model means as well (i.e., accurate forecasts) would be welcome.

4.1 Univariate Modeling



(a) Simulation plot.



(b) Fitted vs. actual plot.

Figure 8: Forecast and standardized residual plots of a GARCH$(2, 2)$ model fitted to returns.

Our primary univariate modeling result is via the GARCH$(2, 2)$ model. Coefficients are presented in Table 3. At a glance, it appears that the ω, α_1, and β_2 terms are significant, indicating that the variance is being modeled based on previous variances. μ does not appear to be significant, indicating there is no evidence the mean return is non-zero.

Moreover, the simulations are able to replicate the changes in variance similar to the actual data (Figure 8a), which indicates that our GARCH fit is able to model the variance well. This is further reinforced by the actual vs. fitted plot with confidence bounds (Figure 8b), where we see the fitted variance increase to accommodate for sudden spikes, allowing the confidence range to include the period of higher variance following large changes.

Note, however, that the predicted and fitted means for the GARCH model are essentially constant at 0. This is to be expected as GARCH models are meant to model the changing variance of a process, not its mean. In this regard, we evaluate this as a good model for the variance in the returns time series. We now present the results of our multivariate fits, which we aim to use to

	ar1	ar2	ma1	intercept
	-0.2290	-0.1198	0.2461	0.0020
s.e.	0.1402	0.0861	0.5233	0.0018
	humid.lag.1	temp.lag.1	precip.lag.1	futures.returns.lag.1
	0	-1e-04	-0.0018	-0.0250
s.e.	0	0e+00	0.0009	0.4234
	humid.lag.2	temp.lag.2	precip.lag.2	futures.returns.lag.2
	0	0	0.0011	0.1103
s.e.	0	0	0.0010	0.1698

Table 4: Coefficients for time series regression model with lag 2.

better model means.

4.2 Multivariate Modeling

4.2.1 Time Series Regression

Our final fit for our time series regression was an ARMA(2,1) mean with parameters as presented in Table 4. We then were able to fit the residuals with a GARCH(4,2) model with parameters as presented in Table 7.

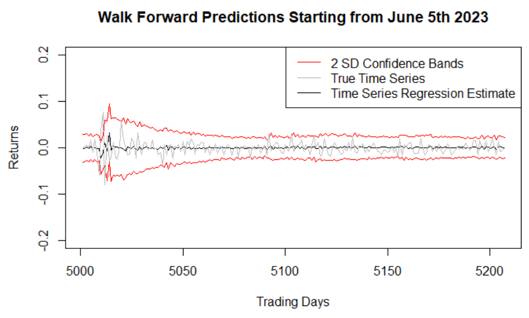
From the coefficients in Table 7, we see that this model has coefficients of zero or near zero for temperature and humidity but has non-zero coefficients for precipitation and the lagged returns. This model suggests that it's mostly temperature that contributes to the estimate of the returns; however, since it doesn't model the interdependencies between humidity, temperature, and precipitation, it could simply be attributing the other variables to the precipitation variables.

In order to evaluate the overall fit of this model, we will do walk forward analysis. This consists of predicting one step ahead and then comparing the set of one step ahead predictions to the true values. We will also get walk forward conditional standard deviations and add them as our 2 SD intervals. We can see this walk forward analysis in Figure 9a, the GARCH bands model the volatility of the time series while the time series regression helps follow the small changes in the trend of the returns. This model looks like it does a pretty good job at capturing both the estimate of the returns and also the volatility in them.

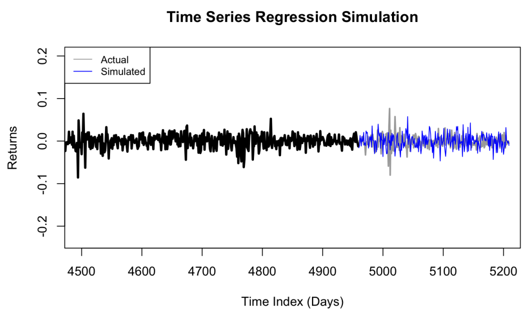
Additionally, we can analyze a simulation from the time series regression. In Figure 9b, we can see that the time series regression itself doesn't have a perfect representation of the volatility. This makes sense, as we know the error assumptions in the time series regression were wrong, which is why we added the GARCH modeling to better model uncertainty. Thus, while this behavior is not necessarily good, it is expected.

4.2.2 VAR

Our final fit for our VAR model was a lag 5 model with parameters as presented in Table 5. Since VAR models for interdependencies between our different climate variables, it ends up producing smaller but non-zero coefficients for all the variables. Due to this fact, it likely better reflects the



(a) Walk-forward estimates.



(b) Single simulation vs actual.

Figure 9: Time series regression estimates and simulations.

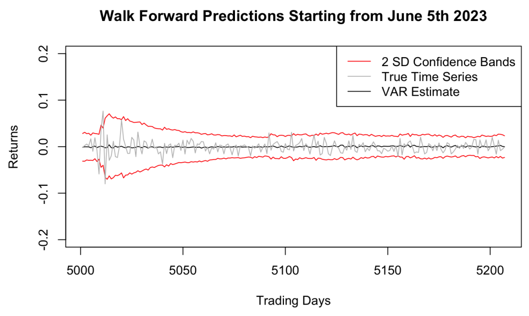
	humid.l1	temp.l1	precip.l1	futures.returns.l1	
	2.832688e-05	-4.890620e-05	1.524098e-03	-1.107678e-02	
s.e.	2.422261e-05	3.201588e-05	7.811666e-04	1.388797e-02	
	humid.l2	temp.l2	precip.l2	futures.returns.l2	
	-5.619372e-05	6.835772e-05	7.261996e-04	-1.672650e-02	
s.e.	2.736047e-05	4.149816e-05	7.851395e-04	1.388781e-02	
	humid.l3	temp.l3	precip.l3	futures.returns.l3	
	-1.624608e-05	-1.519102e-05	-1.797180e-04	1.273022e-02	
s.e.	2.734529e-05	4.165447e-05	7.864920e-04	1.388485e-02	
	humid.l4	temp.l4	precip.l4	futures.returns.l4	
	5.177312e-05	-9.013461e-05	-5.124654e-04	-5.202161e-03	
s.e.	2.722482e-05	4.164795e-05	7.840356e-04	1.388435e-02	
	humid.l5	temp.l5	precip.l5	futures.returns.l5	const
	-3.680099e-05	4.071543e-05	2.889234e-04	-2.251488e-02	4.262263e-03
s.e.	2.387811e-05	3.251463e-05	7.705283e-04	1.386710e-02	2.040535e-03

Table 5: Coefficients for VAR with lag 2.

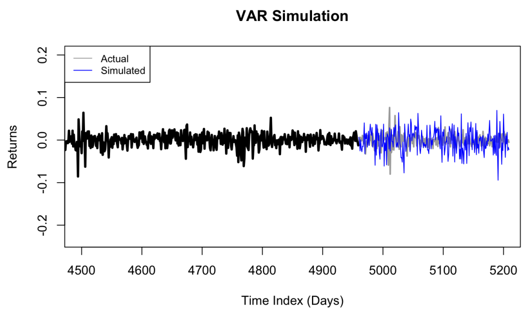
contributions of each individual climate variable to our process. Our final fit for the residuals of our process was GARCH(2,2).

In order to evaluate the overall fit of our model we will do walk-forward analysis as before. In Figure 10a, we see that our confidence bands do contain the volatility of the process as well as the time series regression of the GARCH model does. If we look at the simulation of the VAR in Figure 10b, we can also see that we better match the variability in the true data in comparison to the time series regression model. This makes sense as the residual assumptions for the time series regression model are much stronger.

Overall, the VAR + GARCH Residuals model seems to fit the process slightly better than the time series regression + GARCH residuals from the simulation and residual analysis from the methods section.



(a) Walk-forward estimates.



(b) Single simulation vs actual.

Figure 10: VAR estimates and simulations.

5 Discussion

Among our univariate models, GARCH allowed us to fit the heteroscedasticity that we observed in the ARIMA model residuals, and the residuals of the GARCH better satisfy normality assumptions of the model. Furthermore, the GARCH simulation also reflects the heteroscedasticity of the underlying data, and the walk forward prediction appears to model the variance very well. In fact, the two standard deviation bounds tightly enclose the original time series.

In the multivariate setting, we were able to integrate the weather data as covariates in order to reach slightly better point estimates compared to the univariate GARCH model. While initially, this comes with the trade-off of losing out on the heteroscedastic modeling of the volatility in the futures returns we accounted for this trade-off by modeling the residuals of these models with a GARCH process. This resulted in both the VAR and Time Series regression providing us marginally better bounds within 2 confidence intervals in comparison to the GARCH model. Between these two multivariate models, it seems that the time series regression has more parsimony as it uses fewer parameters and some of the parameters it uses are essentially zero and can be dropped. The trade off is that it doesn't model the interdependencies between the climate data, which means the VAR likely provides us with a better understanding of how these variables interact in the real world while upping the complexity of the model.

5.1 Limitations and Future Work

Unfortunately, none of our models allowed us to effectively estimate the returns. The univariate GARCH model does not model trends, and our attempts to model trends using the multivariate models with climate data were only marginally more effective. This makes sense, as futures returns data in general is a very noisy process due to the large frequency of trades made. In addition, there are many other input variables that make the process more noisy, from which we are trying to extract a much weaker local climate-based signal. Additionally, the data we are taking is specifically from a region in America that produces a decent amount of soybean and not even representative of all soybean exporting areas. Thus, it makes sense that we have parameters of small magnitude that only slightly inform estimates, given that they only encompass a small portion of the soybean growth process, which is only part of the soybean futures pricing process.

Future work could aim to better capture these other influences. For example, we can include climate information from other regions. We can use data from many different places in the US, which can be more representative of the true climate and be less noisy in aggregate. Additionally, we can include metrics measuring market sentiment, geopolitical stability, and other known influences on financial markets.

An alternative approach would see us using smoothed data, which can help us reduce the noise in the data and boost the signal. For example, we can use weekly or monthly aggregates instead of the current daily data.

Overall, this report performs promising early work towards modeling the volatility in the soybean futures market and making predictions using climate data, but more work is needed to effectively extract signals from the noise.

References

Investing.com. Us soybeans futures price. https://www.investing.com/commodities/us-soybeans, 2024. Accessed: 2024-04-19.

NOAA. Applied climate information system. https://scacis.rcc-acis.org/, 2024. Accessed: 2024-04-19.

Ruey S Tsay. *Analysis of financial time series*. John Wiley & Sons, 2005.

United Soybean. New study finds u.s. soybean industry has $124 billion impact on the united states economy. https://www.unitedsoybean.org/hopper/new-study-finds-u-s-soybean-industry-has-124-billion-impact-on-the-united-states-economy/, 2023. Accessed: 2024-04-19.

USDA. Soybeans production by county. https://www.nass.usda.gov/Charts_and_Maps/Crops_County/sb-pr.php, 2023. Accessed: 2024-04-19.

USDA. Soybeans. https://fas.usda.gov/data/commodities/soybeans, 2024a. Accessed: 2024-04-19.

USDA. Soybean explorer. https://ipad.fas.usda.gov/cropexplorer/cropview/commodityView.aspx?cropid=2222000, 2024b. Accessed: 2024-04-19.

Patrick Zippenfenig. Open-meteo.com weather api, 2023. URL https://open-meteo.com/.

Here we present the coefficients fit for the ARMA$(4,5)$ model we rejected in the methods section above, for completion.

A Tables of Parameters

Below we include fitted coefficients of models not discussed in depth in the main report.

	intercept	ar1	ar2	ar3	ar4
	0e+00	-0.5158	0.0967	-0.6492	-0.7480
s.e.	2e-04	0.1834	0.1088	0.0501	0.1582
	ma1	ma2	ma3	ma4	ma5
	0.5059	-0.1127	0.6559	0.7413	-0.0448
s.e.	0.1835	0.1087	0.0496	0.1555	0.0166

Table 6: Coefficients for ARMA$(4,5)$ model.

	mu	omega	alpha1	alpha2
	4.2550e-05	4.8203e-06	1.0588e-01	1.0000e-08
s.e.	2.060324e-04	1.758194e-06	1.838013e-02	4.961200e-02
	alpha3	alpha4	beta1	beta2
	1.0000e-08	1.0000e-08	3.2119e-01	5.5589e-01
s.e.	3.290023e-02	3.021720e-02	4.245067e-01	3.853316e-01

Table 7: Coefficients for GARCH$(4,2)$ model on residuals of time series regression.

	mu	omega	alpha1	alpha2	beta1	beta2
	5.695071e-18	5.219122e-06	9.504497e-02	2.477280e-02	1.457927e-01	7.154571e-01
s.e.	1.763360e-04	1.085526e-06	1.312542e-02	2.198874e-02	1.541942e-01	1.411038e-01

Table 8: Coefficients for GARCH$(2,2)$ model on residuals of VAR.

B Code

Here we present the `R` code used to perform our analyses and generate figures.

```r
# read data
temp_precip = read_csv("data/quincy_il_temp_precip.csv", show_col_types = FALSE)
humid_hourly = read_csv("data/quincy_il_humidity.csv", show_col_types = FALSE)
futures_1 = read_csv("data/soybeans_to_2023.csv", show_col_types = FALSE)
futures_2 = read_csv("data/soybeans_after_2023.csv", show_col_types = FALSE)
futures = bind_rows(futures_1, futures_2)
```

```r
# parse futures data
futures$Date = as.Date(futures$Date, "%m/%d/%Y")
futures = complete(futures, Date = seq(min(Date), max(Date), by = "day"))

# parse temp and precip data
temp_precip$AvgTemperature =
  suppressWarnings(as.numeric(temp_precip$AvgTemperature))
temp_precip$Precipitation =
  suppressWarnings(as.numeric(temp_precip$Precipitation))

# aggregate humid data
humid_hourly$date <- as.Date(humid_hourly$time)
humid = aggregate(`relative_humidity_2m (%)` ~ date, humid_hourly, mean)
colnames(humid) <- c("date", "humidity")

# convert to xts for handling nas
futures_ts = xts(futures$Price, futures$Date)
humid_ts = xts(humid$humidity, humid$date)
temp_ts = xts(temp_precip$AvgTemperature, temp_precip$Date)
precip_ts = xts(temp_precip$Precipitation, temp_precip$Date)

# Handle nas: carry-forward futures, interpolate humid and temp, 0 for precip
futures_ts = na.locf(futures_ts)
humid_ts = na.approx(humid_ts)
temp_ts = na.approx(temp_ts, maxgap=Inf)
precip_ts = na.fill(precip_ts, 0.0)

# collect to df
all_df = data.frame(date = futures$Date,
                    futures_ts, humid_ts, temp_ts, precip_ts)

# retain only trading data
trading_futures = bind_rows(futures_2, futures_1)
trading_dates = as.Date(trading_futures$Date, "%m/%d/%Y")
trading_df = all_df[all_df$date %in% trading_dates,]
trading_ts = xts(cbind(trading_df$futures_ts, trading_df$humid_ts,
                       trading_df$temp_ts, trading_df$precip_ts),
                 trading_df$date)
colnames(trading_ts) = c("futures", "humid", "temp", "precip")

# create returns
noweekend.ts <- trading_ts$futures
no.weekend.returns <- diff(log(noweekend.ts))[-1]

# plot time series and returns
plot.zoo(noweekend.ts, main = "Soybean Futures Prices", ylab = "Price",
     xlab = "Date")
plot.zoo(no.weekend.returns, main = "Soybean Futures Returns",
         ylab = "Returns", xlab = "Date")
acf2(no.weekend.returns, main = "ACF/PACF of Soybean Futures returns")

# Auto fit ARMA
# returns.arima = auto.arima(no.weekend.returns, max.p = 5, max.q = 5,
#                            max.order = 14, stationary = T, seasonal = F,
#                            trace = F, stepwise = F, approximation = F)

# Selected ARMA(4, 5)
returns.arima <- arima(no.weekend.returns, order=c(4,0,5))
summary(returns.arima)
AIC(returns.arima)

# plot fit
plot(ts(as.numeric(no.weekend.returns)), lwd=1, xlab = "Time Index (Days)",
     ylab = "Returns", main = "ARIMA(4,0,5) Fitted Values", type='l')
lines(fitted(returns.arima), col="red", lwd=1)
legend("topleft", legend=c("Actual", "Fitted"),
       col=c("black", "red"), lty=c(1,1), cex=0.8)

# qq plot
qqnorm(returns.arima$residuals, main="ARMA Model Q-Q Plot")
qqline(returns.arima$residuals)

# residual plots
tsdisplay(returns.arima$residuals, main="ARMA Model Residuals")

# simulate last 250 trading days
reduced.ts <- no.weekend.returns[1:(length(no.weekend.returns) - 250)]
n_ahead <- length(no.weekend.returns) - length(reduced.ts)

# refit model to reduced data
reduced.returns.arima <- arima(reduced.ts, order=c(4,0,5))
returns.arima.forecast <- predict(reduced.returns.arima,
                                  n.ahead = n_ahead, conf = 0.95)

# plot simulation
plot(x=1:length(no.weekend.returns), y=coredata(no.weekend.returns),
     col="darkgray", lwd=2, xlim=c(4500,5200), xlab="Time Index (Days)",
     ylab="Returns", main="ARIMA(4,0,5) Leave-out simulations (95% CI)",
     type='l')
lines(x=1:length(reduced.ts), y=coredata(reduced.ts), lwd=3)
lines(x=(length(reduced.ts)+1):length(no.weekend.returns),
      y=returns.arima.forecast$pred, col="red", lwd=2)
lines(x=(length(reduced.ts)+1):length(no.weekend.returns),
      y=returns.arima.forecast$pred-1.96*returns.arima.forecast$se,
      col="red", lty=2, lwd=3)
lines(x=(length(reduced.ts)+1):length(no.weekend.returns),
      y=returns.arima.forecast$pred+1.96*returns.arima.forecast$se,
      col="red", lty=2, lwd=3)
lines(simulate(reduced.returns.arima, nsim=250), col='blue')
legend("topleft",
       legend=c("Actual", "Confidence Bands", "Prediction", "Simulated"),
       col=c("darkgray", "red", "red", "blue"), lty=c(1, 2, 1, 1), cex=0.8)

# create squared returns
no.weekend.returns.sq <- no.weekend.returns^2
acf2(no.weekend.returns.sq, main="ACF/PACF of Soybean Futures Returns Squared",
     max.lag=20)

# auto fit arma on squared residuals
# gives AMRA(6, 4)
# no.weekend.returns.sq.ARMA <-
#   auto.arima(no.weekend.returns.sq, max.p=7, max.q=7, max.order=14,
#              stationary=T, seasonal=F, trace=T, stepwise=F, approximation=F)
# summary(no.weekend.returns.sq.ARMA)

# find parsimonious model
# select GARCH(2, 2)
# gfit64 <-
#   garchFit(~garch(6,4), data=no.weekend.returns, cond.dist='norm', trace=F)
# summary(gfit64)
# gfit11 <-
#   garchFit(~garch(1,1), data=no.weekend.returns, cond.dist='norm', trace=F)
# summary(gfit11)
# gfit12 <-
#   garchFit(~garch(1,2), data=no.weekend.returns, cond.dist='norm', trace=F)
# summary(gfit12)
# gfit21 <-
#   garchFit(~garch(2,1), data=no.weekend.returns, cond.dist='norm', trace=F)
# summary(gfit21)
gfit22 <-
  garchFit(~garch(2,2), data=no.weekend.returns, cond.dist='norm', trace=F)
# summary(gfit22)
# gfit13 <-
#   garchFit(~garch(1,3), data=no.weekend.returns, cond.dist='norm', trace=F)
# summary(gfit13)
# gfit31 <-
#   garchFit(~garch(3,1), data=no.weekend.returns, cond.dist='norm', trace=F)
# summary(gfit31)
gfit <- gfit22

# perform walk forward
next_pred = function(end, ts) {
  temp <- garchFit(~garch(2,2), data=ts[1:end], trace = F)
  return(predict(temp,n.ahead = 1))
}
walk_pred = data.frame(meanForecast = 0, meanError = 0, standardDeviation = 0)
for (i in 4500:5209) {
  walk_pred = rbind(walk_pred, next_pred(i, no.weekend.returns))
}
walk_pred = walk_pred[-1,]

# plot walk forward
plot(x=5000:5209, y=ts(walk_pred$meanForecast), type="l", ylim = c(-0.2,.2),
     col="black", xlab="Time Index (Days)", ylab="Returns",
     main = "Walk Forward Predictions Starting from June 5th 2023")
lines(x=5000:5209, y=ts(walk_pred$meanForecast+2*walk_pred$standardDeviation),
      col="red")
lines(x=5000:5209, y=ts(walk_pred$meanForecast-2*walk_pred$standardDeviation),
      col="red")
lines(x=5000:5209, y=ts(no.weekend.returns[5000:5209]), col="darkgrey")
legend("topleft", legend=c("Actual", "2 SD Confidence Bands", "Prediction"),
       col=c("darkgray", "red", "black"), lty=c(1, 1, 1), cex=0.8)

# plotting functions from fGarch source code, modified to change xlab and main
custom.plot.1 <- function(x, ...)
{
    # A function implemented by Diethelm Wuertz

    # Description:
    #   Internal plot function

    # 1. Time Series:
    xseries = x@data
    plot(xseries, type = "l", col = "steelblue", ylab = "Returns",
        main = "Soybean Futures Returns", xlab = "Time Index (Days)")
    abline(h = 0, col = "grey", lty = 3)
    grid()
}
custom.plot.2 <-
    function(x, ...)
{
    # A function implemented by Diethelm Wuertz

    # Description:
    #   Internal plot function

    # 2. Conditional SD:
    xcsd = volatility(x, "sigma")
    plot(xcsd, type="l", col="steelblue", ylab="Conditional Standard Deviation",
        main="Conditional SD of Soybean Futures Returns", xlab="Time Index (Days)")
    abline(h = 0, col = "grey", lty = 3)
    grid()
}

# plot
custom.plot.1(gfit)
custom.plot.2(gfit)

# standardized residuals
resid = residuals(gfit) / gfit@sigma.t
tsdisplay(resid, main = "GARCH residuals")
qqnorm(resid, main = "GARCH Standardized Residuals Normal Q-Q Plot")
qqline(resid)

# refit model to reduced data
gfit <- garchFit(~garch(2,2), data=reduced.ts, cond.dist = "std", trace=F)
gfit.forecast <- predict(gfit, n.ahead = n_ahead, conf = 0.95)

# plot
spec = garchSpec(model = list(alpha = coef(gfit)[3:4], beta = coef(gfit)[5:6],
                              omega = coef(gfit)[2], mu = coef(gfit)[1]))
plot(x=1:length(no.weekend.returns), y=coredata(no.weekend.returns),
     col="darkgray", lwd=2, xlim=c(4500,5200), xlab="Time Index (Days)",
     ylab="Returns", main="GARCH(2,2) Leave-out Forecasts (95% CI)", type='l')
lines(x=1:length(reduced.ts), y=coredata(reduced.ts), lwd=3)
lines(x =(length(reduced.ts)+1):length(no.weekend.returns),
      y=as.numeric(
        garchSim(spec, n=250, n.start=length(no.weekend.returns)-250)),
      col='blue')
legend("topleft", legend=c("Actual", "Simulated"),
       col=c("darkgray", "blue"), lty=c(1, 1), cex=0.8)

# create multi-ts with all data
full_ts = cbind(trading_ts[-1, -1], no.weekend.returns)
colnames(full_ts) = c("humid", "temp", "precip", "futures returns")

# get aics for lags
```

```r
lag = 2
# covariates = full_ts
# dates = index(returns)[-c((length(returns)-(lag-1)):length(returns))]
# returns = full_ts£futures
# lagged_cov = covariates[-c(1:lag)]
# index(lagged_cov) = dates
#
for (iter in (lag-1):1) {
  next_lagged_cov =
    covariates[-c(1:iter,(length(returns)-(lag-iter-1)):length(returns))]
  index(next_lagged_cov) = dates
  lagged_cov = merge(lagged_cov, next_lagged_cov, all=TRUE)
}
lag_reg = auto.arima(returns[1:4000], xreg = lagged_cov[1:4000])
summary(lag_reg)


# time series regression
returns = full_ts$futures
covariates = full_ts
dates =
  index(returns)[-c(length(returns)-2, length(returns)-1, length(returns))]
covariates.lag1 =
  covariates[-c(1,length(index(covariates))-1,length(index(covariates)))]
index(covariates.lag1) = dates
covariates.lag2 = covariates[-c(1,2,length(index(covariates)))]
index(covariates.lag2) = dates
covariates.lag3 = covariates[-c(1,2,3)]
index(covariates.lag3) = dates
covariates = merge(covariates.lag1, covariates.lag2, all = TRUE)
covariates = merge(covariates, covariates.lag3, all = TRUE)
returns = returns[-c(length(returns)-1, length(returns))]

# fit arima
# auto.arima(returns[1:4000], xreg = covariates[1:4000])
order_fit = Arima(returns[1:4000], order = c(2,0,1), xreg = covariates[1:4000])
summary(order_fit)
resid = (order_fit$residuals)
resid_sq = resid^2

# residual plots
tsdisplay(resid, main = "Time Series Regression Residuals")
tsdisplay(resid_sq, main = "Time Series Regression Squared Residuals")

# conditional sd plots
arima_resid_sq = arima(resid_sq, order = c(4,0,2))
```

```r
garch_resid = garchFit( ~ garch(4,2), data = resid, trace = F)
plot(garch_resid, which = 3,
     main = "Series with 2 Conditional SD Superimposed",
     ylab = "Returns")


# predict using model
next_pred = function(returns, covariates, end) {
  train_ret = returns[1:end]
  train_cov = covariates[1:end]
  test_cov = covariates[end+1]
  fit = arima(train_ret, order = c(2,0,1), xreg = train_cov)
  garch_fit = garchFit( ~ garch(4,2), data = fit$residuals, trace = F)
  sd = predict(garch_fit, n.ahead = 1)$standardDeviation
  temp = predict(fit, newxreg = test_cov)
  newrow = data.frame(lower = temp$pred - 2*sd,
                      estimate = temp$pred,
                      upper = temp$pred + 2*sd)

  return(newrow)
}
preds_frame = data.frame(lower = 0, estimate = 0, upper = 0)
for (i in 5000:5206) {

  preds_frame = rbind(preds_frame,
                      next_pred(returns, covariates, i))
}
preds_frame = preds_frame[-1,]

# plot walk forward
plot(ts(preds_frame$estimate, start  = 5001),ylim = c(-0.2,.2),
     ylab = "Returns",
     xlab = "Trading Days",
     main = "Walk Forward Predictions Starting from June 5th 2023")
lines(ts(preds_frame$lower, start  = 5001),col="red")
lines(ts(preds_frame$upper, start  = 5001),col="red")
lines(ts(returns[5001:5207], start  = 5001),col="grey")
legend("topright",
       legend = c("2 SD COnfidence Bands",
                  "True Time Series",
                  "Time Series Regression Estimate"),
       col = c("red", "grey", "black"),
       lty = 1)

# standardized residuals
```

```r
resid = residuals(garch_resid) / garch_resid@sigma.t
tsdisplay(resid, main = "GARCH Standardized Residuals")
qqnorm(resid, main = "GARCH Standardized Residuals Normal Q-Q Plot")
qqline(resid)

# simulations plot
spec =
  garchSpec(model=list(alpha=coef(garch_resid)[3:6], beta=coef(garch_resid)[6:7],
                              omega=coef(garch_resid)[2], mu=coef(garch_resid)[1]))
plot(x=1:length(no.weekend.returns), y=coredata(no.weekend.returns),
     col="darkgray", lwd=2, xlim=c(4500,5200), xlab="Time Index (Days)",
     ylab="Returns", main="Time Series Regression Simulation", type='l')
lines(x=1:length(reduced.ts), y=coredata(reduced.ts), lwd=3)
lines(x =(length(reduced.ts)+1):length(no.weekend.returns),
      y=as.numeric(
        garchSim(spec, n=250, n.start=length(no.weekend.returns)-250))+
        simulate(order_fit, nsim=250, xreg=covariates[4001:4250]),
      col='blue')
legend("topleft", legend=c("Actual", "Simulated"),
       col=c("darkgray", "blue"), lty=c(1, 1), cex=0.8)

# select var fit
# VARselect(full_ts, lag.max=100)

# lag 5 selected
var_fit = VAR(full_ts, p=5)$varresult$futures.returns
summary(var_fit)
var_resid = (var_fit$residuals)
var_resid_sq = var_resid^2

# residual plots
tsdisplay(var_resid, main = "VAR Regression Residuals")
tsdisplay(var_resid_sq, main = "VAR Regression Squared Residuals")

# conditional sd plots
# auto.arima(var_resid_sq)
arima_var_resid_sq = arima(var_resid_sq, order = c(0,1,2))
garch_var_resid = garchFit( ~ garch(2,2), data = var_resid, trace = F)
plot(garch_var_resid, which = 3,
     main = "VAR Residuals with 2 Conditional SD Superimposed",
     ylab = "Returns")

# predict using model
# var_next_pred = function(end) {
#   fit = VAR(full_ts[1:end], p=5)
#   garch_fit = garchFit( ~ garch(2,2),
#                          data=fit£varresult£futures.returns£residuals, trace = F)
#   sd = predict(garch_fit, n.ahead = 1)£standardDeviation
#   temp = predict(fit, n.ahead=1)£fcst£futures.returns[1]
#   newrow = data.frame(lower = temp - 2*sd,
#                        estimate = temp,
#                        upper = temp + 2*sd)
#
#   return(newrow)
# }
# var_preds_frame = data.frame(lower = 0, estimate = 0, upper = 0)
# for (i in 5000:5206) {
#   var_preds_frame = rbind(var_preds_frame,
#                            var_next_pred(i))
# }
# var_preds_frame = var_preds_frame[-1,]
#
# # plot walk forward
# plot(ts(var_preds_frame£estimate, start  = 5001),ylim = c(-0.2,.2),
#      ylab = "Returns",
#      xlab = "Trading Days",
#      main = "Walk Forward Predictions Starting from June 5th 2023")
# lines(ts(var_preds_frame£lower, start  = 5001),col="red")
# lines(ts(var_preds_frame£upper, start  = 5001),col="red")
# lines(ts(returns[5001:5207], start  = 5001),col="grey")
# legend("topright",
#        legend = c("95% Confidence Bands",
#                   "True Time Series",
#                   "VAR Estimate"),
#        col = c("red", "grey", "black"),
#        lty = 1)

# standardized residuals
resid = residuals(garch_var_resid) / garch_var_resid@sigma.t
tsdisplay(resid, main = "GARCH residuals")
qqnorm(resid, main = "GARCH Normalized Residuals Normal Q-Q Plot")
qqline(resid)

# simulations plot
spec = garchSpec(
  model=list(alpha=coef(garch_var_resid)[3:4], beta=coef(garch_var_resid)[5:6],
             omega=coef(garch_var_resid)[2], mu=coef(garch_var_resid)[1]))
plot(x=1:length(no.weekend.returns), y=coredata(no.weekend.returns),
     col="darkgray", lwd=2, xlim=c(4500,5200), xlab="Time Index (Days)",
     ylab="Returns", main="VAR Simulation", type='l')
```

```r
lines(x=1:length(reduced.ts), y=coredata(reduced.ts), lwd=3)
lines(x =(length(reduced.ts)+1):length(no.weekend.returns),
      y=as.numeric(
        garchSim(spec, n=250, n.start=length(no.weekend.returns)-250))+
        fitted(var_fit)[4001:4250],
      col='blue')
legend("topleft", legend=c("Actual", "Simulated"),
       col=c("darkgray", "blue"), lty=c(1, 1), cex=0.8)
```