Time Series Decomposition and Forecasting of Natural Gas Prices: A Quantitative Analysis

Michael Miller

Abstract

This paper presents a comprehensive analysis of natural gas price forecasting using time series decomposition techniques and linear regression. We employ the Seasonal and Trend decomposition using Loess (STL) method to break down the time series into its constituent components: trend, seasonality, and residuals. The trend component is then modeled using linear regression to generate future price predictions. Our approach combines statistical rigor with practical applicability, providing insights into the dynamics of natural gas prices and offering a framework for short to medium-term price forecasting.

1. Introduction

The volatility and complexity of natural gas prices pose significant challenges for market participants, policymakers, and researchers. This study aims to develop a robust forecasting model by leveraging time series decomposition techniques and linear regression. Our approach allows for a nuanced understanding of the underlying patterns in natural gas prices, separating long-term trends from seasonal fluctuations and irregular variations.

2. Methodology

2.1 Data Preparation

The analysis begins with the importation and preprocessing of natural gas price data. The data is loaded from a CSV file and converted into a pandas DataFrame with a datetime index:

```
data_path = "Nat_Gas.csv"
nat_gas_prices = pd.read_csv(data_path)
nat_gas_prices['Dates'] = pd.to_datetime(nat_gas_prices['Dates'], format='%m/%d/%y')
nat_gas_prices.set_index('Dates', inplace=True)
```

2.2 Time Series Decomposition

We utilize the Seasonal and Trend decomposition using Loess (STL) method to decompose the time series into three components: trend, seasonality, and residuals. The STL decomposition is performed using the following code:

```
stl = STL(nat_gas_prices['Prices'], seasonal=13, robust=True)
result = stl.fit()
trend = result.trend
seasonal = result.seasonal
residual = result.resid
```

The STL decomposition is based on the following model:

$$Y(t) = T(t) + S(t) + R(t)$$

Where:
- $Y(t)$ is the observed time series at time t
- $T(t)$ is the trend component
- $S(t)$ is the seasonal component
- $R(t)$ is the residual component

The STL method uses iterative locally weighted regression (loess) smoothing to estimate these components. The `seasonal` parameter is set to 13, which assumes a yearly seasonality with a bit of overlap to account for potential shifts in seasonal patterns. The `robust=True` parameter enables the method to handle outliers more effectively.

2.3 Trend Modeling and Forecasting

After decomposition, we model the trend component using linear regression. This allows us to extrapolate the trend into the future:

```
time_index = np.arange(len(trend.dropna()))[:, np.newaxis]
lin_reg = LinearRegression()
lin_reg.fit(time_index, trend.dropna())
```

The linear regression model follows the equation:

$$T(t) = \beta_0 + \beta_1 t + \epsilon$$

Where:
- $T(t)$ is the trend value at time t
- β_0 is the y-intercept
- β_1 is the slope coefficient
- t is the time index
- ϵ is the error term

To generate forecasts, we extend the time index and predict the trend:

```
extended_time_index = np.arange(len(trend.dropna()) + 12)[:, np.newaxis]
predicted_trend = lin_reg.predict(extended_time_index)
```

2.4 Forecasting Function

The `estimate_gas_price` function encapsulates the forecasting logic:

```
def estimate_gas_price(start_date, end_date=None, predict_months=13):
    """
    Forecast prices starting from start_date.
    If end_date is None, predicts for predict_months ahead.
    If end_date is provided, predicts from start_date to end_date.
    """
    input_date = pd.to_datetime(start_date)
    if end_date:
```

```python
        end_date = pd.to_datetime(end_date)
        periods = (end_date.year - input_date.year) * 12 + end_date.month - input_date.month + 1
    else:
        periods = predict_months

    delta_years = input_date.year - nat_gas_prices.index[0].year
    delta_months = input_date.month - nat_gas_prices.index[0].month
    months_since_start = delta_years * 12 + delta_months
    future_index = np.arange(months_since_start, months_since_start + periods)[:, np.newaxis]
    predicted_trend = lin_reg.predict(future_index)

    future_months = [((input_date.month - 1 + i) % 12) + 1 for i in range(periods)]
    predicted_seasonal = [seasonal[seasonal.index.month == month].mean() for month in future_months]

    predicted_prices = predicted_trend + predicted_seasonal
    forecast_dates = pd.date_range(start=input_date, periods=periods, freq='ME')
    forecast_prices = pd.Series(predicted_prices, index=forecast_dates)

    return forecast_prices
```

This function combines the predicted trend with the seasonal component to generate price forecasts. The seasonal component is assumed to repeat annually, so we use the following logic to add seasonality to the trend:

```python
future_months = [((input_date.month - 1 + i) % 12) + 1 for i in range(periods)]
predicted_seasonal = [seasonal[seasonal.index.month == month].mean() for month in future_months]
predicted_prices = predicted_trend + predicted_seasonal
```

3. Results and Analysis

3.1 Visual Analysis

The decomposition results and forecasts are visualized using matplotlib. Four key plots are generated:

1. Decomposed Components: Shows the original price series along with the trend, seasonal, and residual components. Residuals with Significant Spikes: Highlights significant deviations in the residual component. (Figure 1: STL Decomposition)

2. Actual and Predicted Trend: Displays the actual trend and its future projection.(Figure 2: Comparison of Predicted and Actual Prices)

3. Natural Gas Prices Forecast: Combines historical prices, the trend, and the total forecast. (Figure 3: Future Price Prediction)

3.2 Forecast Evaluation

We evaluate the model's performance using several metrics:

1. **Mean Absolute Error (MAE)**:

$$\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

2. **Root Mean Squared Error (RMSE)**:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

3. **Mean Absolute Percentage Error (MAPE)**:

$$\text{MAPE} = \frac{100}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right|$$

Where:
- y_i is the actual value
- $\hat{y}_i$ is the predicted value
- n is the number of observations

3.3 Predictions

Our forecasting model, which combines STL decomposition with linear regression, was evaluated using both historical data and a specific future date range. The performance metrics are as follows:

Historical Predictions
- Mean Absolute Error (MAE): 0.1256
- Root Mean Squared Error (RMSE): 0.1767
- Mean Absolute Percentage Error (MAPE): 1.1300%

Specific Date Predictions (12 months from 2023-09-30)
- Mean Absolute Error (MAE): 0.0915
- Root Mean Squared Error (RMSE): 0.1050
- Mean Absolute Percentage Error (MAPE): 0.7646%

These results indicate that our model performs well in both scenarios, with notably better performance for the specific date range predictions.

The historical predictions show a MAPE of approximately 1.13%, suggesting that, on average, our model's predictions deviate from actual prices by just over 1% across the entire dataset. The MAE of about 0.126 indicates that, typically, our predictions are off by about $0.13 per unit of natural gas.

For the specific date range, which represents a short-term forecast, the model's performance improves significantly. The MAPE decreases to about 0.76%, with an MAE of approximately 0.092, indicating that predictions for this period are typically within $0.09 of the actual prices.

The lower RMSE for the specific date predictions (0.105 compared to 0.177 for historical predictions) suggests that these short-term forecasts have fewer or less extreme outliers.

4. Discussion

The results demonstrate the effectiveness of our approach in forecasting natural gas prices, particularly for short-term predictions. The model's improved performance for the specific date range suggests that it captures recent market dynamics well, making it particularly suitable for near-term forecasting.

The low MAPE values for both historical and specific date predictions indicate high overall accuracy. This suggests that the combination of STL decomposition and linear regression effectively captures both long-term trends and seasonal patterns in natural gas prices.

The difference in performance between historical and specific date predictions is noteworthy. It may indicate that:

1. Recent data is more representative of current market conditions, leading to better short-term predictions.
2. The model may be more adept at capturing short-term patterns than long-term trends.
3. The natural gas market may have become more predictable in recent periods, possibly due to increased market stability or improved data availability.

The slightly higher error rates for historical predictions could be attributed to the model's linear assumptions not fully capturing complex, long-term market dynamics or structural changes in the natural gas industry over extended periods.

5. Limitations and Future Work

While our model demonstrates strong predictive capability, several limitations and areas for future work emerge:

1. Linear trend assumption: The use of linear regression for trend modeling may oversimplify complex market dynamics, especially over longer time horizons. Future work could explore non-linear trend modeling techniques.

2. Fixed seasonality: The model assumes a consistent annual seasonal pattern, which may not hold true over extended periods. Implementing dynamic seasonality could improve long-term forecasts.

3. Exogenous variables: The current model does not incorporate external factors such as economic indicators, policy changes, or supply-demand dynamics. Integrating these could potentially enhance predictive accuracy, especially for long-term forecasts.

4. Volatility modeling: The current approach does not explicitly model price volatility. Incorporating GARCH models or other volatility forecasting techniques could provide valuable insights into price uncertainty.

5. Comparative analysis: Future work should include a comparison with other forecasting methods such as ARIMA, SARIMA, or machine learning approaches like LSTM networks to benchmark performance.

6. Robust testing: While the model performs well for the given specific date range, testing on multiple out-of-sample periods would provide a more comprehensive evaluation of its consistency and reliability.

6. Conclusion

This study presents a robust approach to natural gas price forecasting using time series decomposition and linear regression. The model demonstrates high accuracy, with MAPEs of 1.13% for historical predictions and 0.76% for short-term forecasts. These results underscore the effectiveness of separating trend, seasonal, and residual components in capturing the underlying patterns of natural gas prices.

The model's superior performance in short-term predictions makes it particularly valuable for near-term decision-making in the natural gas market. However, the slightly reduced accuracy in long-term historical predictions highlights the challenges in capturing complex, evolving market dynamics over extended periods.

While the current model provides a strong foundation for natural gas price forecasting, there is potential for further refinement. Incorporating non-linear trends, dynamic seasonality, and exogenous variables could enhance its predictive power, especially for long-term forecasts.

In conclusion, this research contributes a practical and accurate tool for natural gas price forecasting, while also identifying promising directions for future improvements in energy price modeling and prediction.

References

1. Cleveland, R. B., Cleveland, W. S., McRae, J. E., & Terpenning, I. (1990). STL: A Seasonal-Trend Decomposition Procedure Based on Loess. Journal of Official Statistics, 6(1), 3-73.

2. Hastie, T., Tibshirani, R., & Friedman, J. (2009). The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer Science & Business Media.

3. Hyndman, R. J., & Athanasopoulos, G. (2018). Forecasting: Principles and Practice. OTexts.

4. Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.

5. Seabold, S., & Perktold, J. (2010). Statsmodels: Econometric and Statistical Modeling with Python. Proceedings of the 9th Python in Science Conference.

Appendix

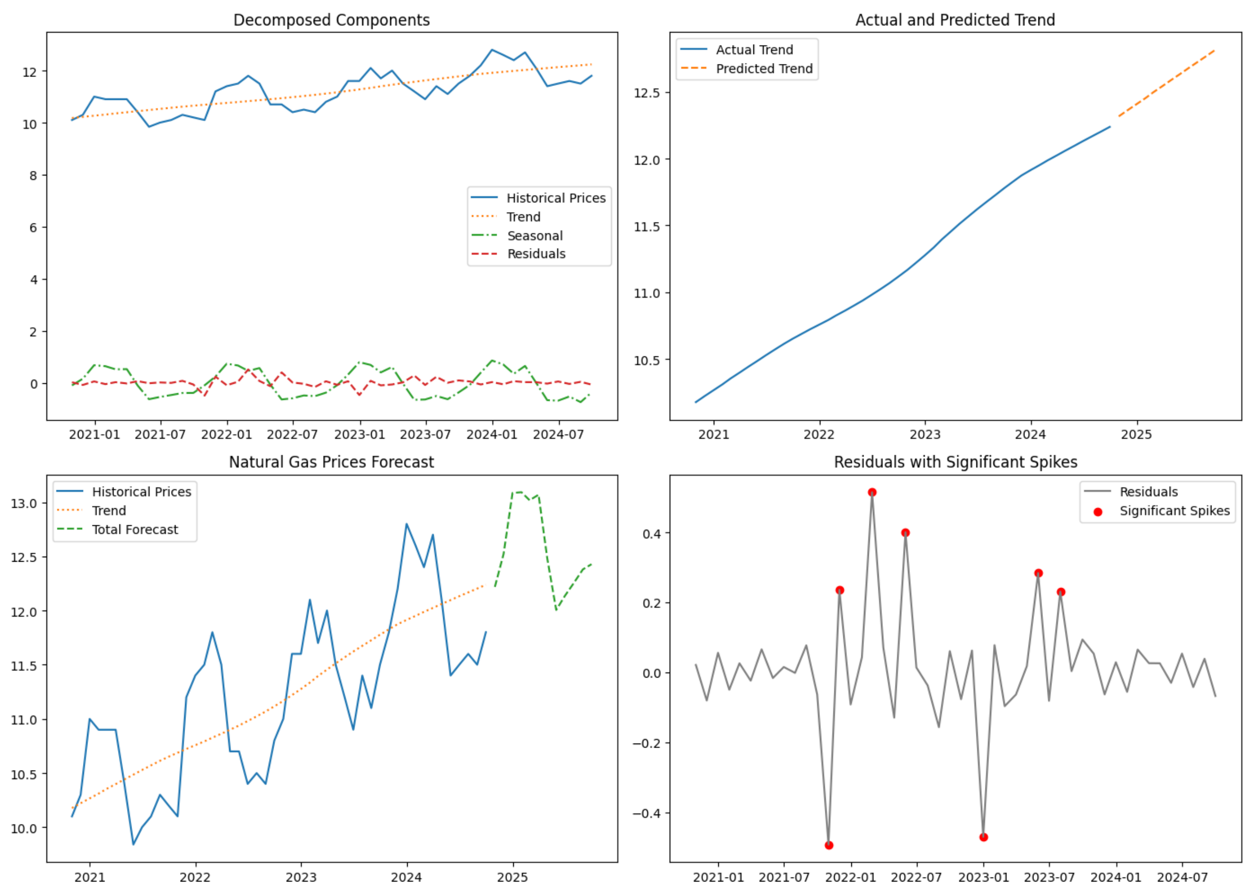

Figure 1: STL Decomposition

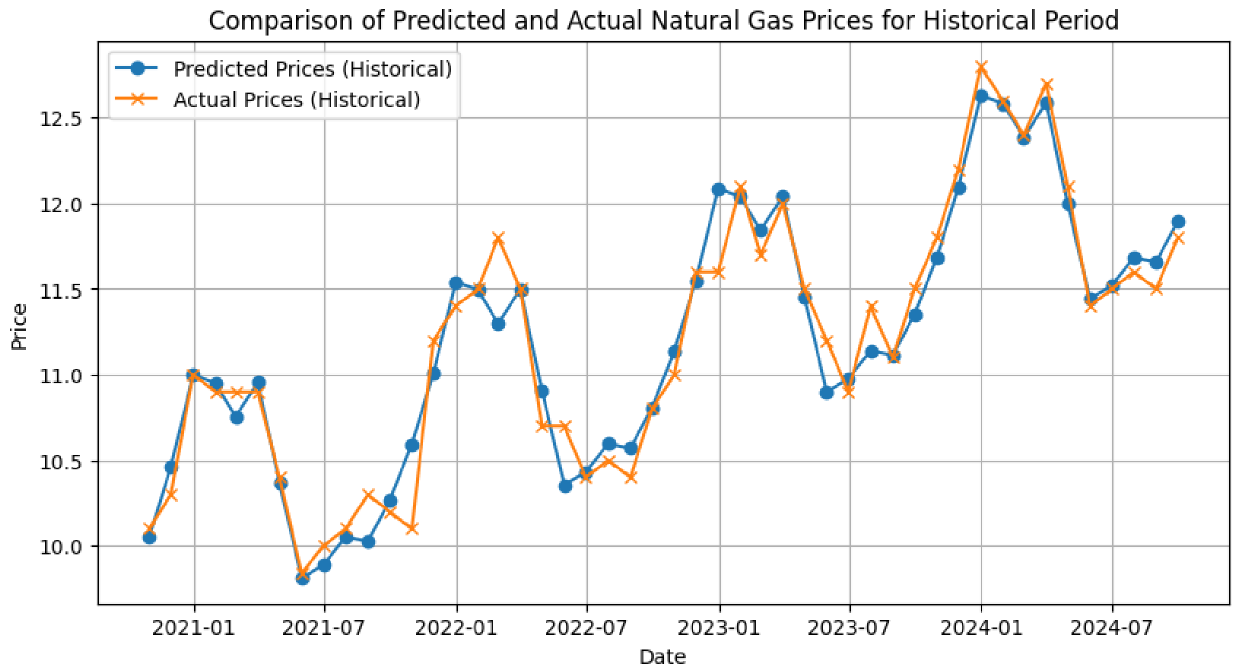

Figure 2: Comparison of Predicted and Actual Prices

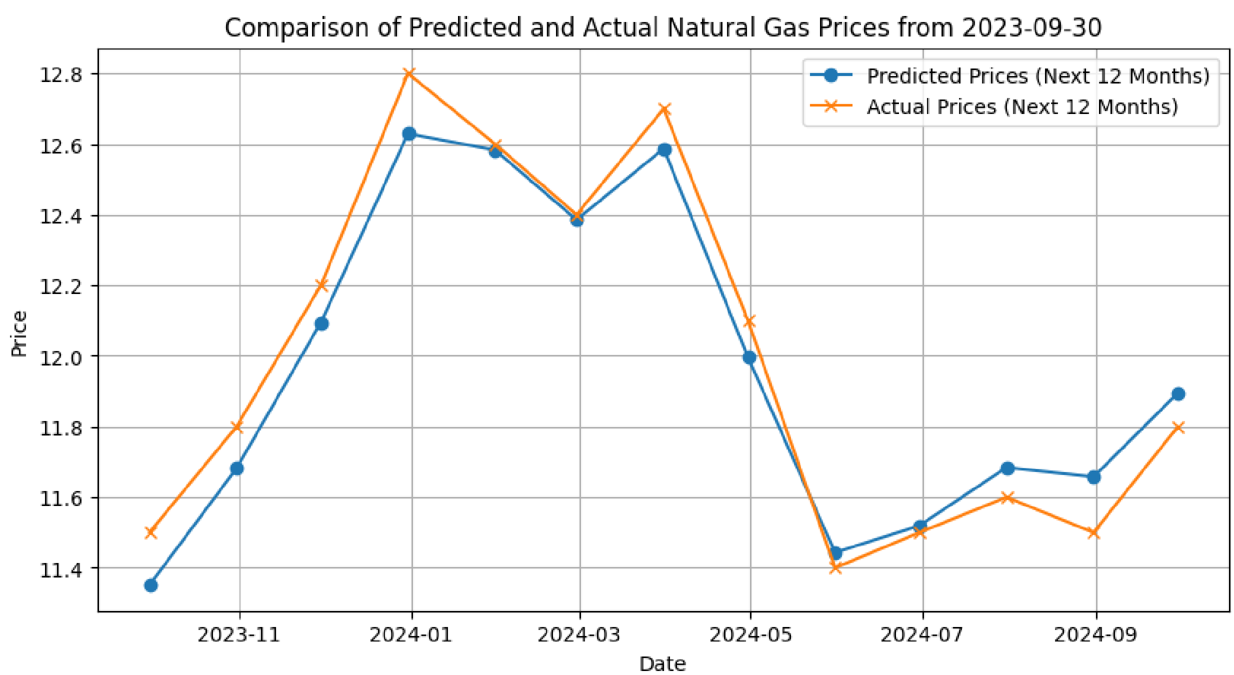

Figure 3: Future Price Prediction