Avionic Noise Reduction – Filtering Analog Voltage Data

Group members:
Kylie Breeding, Fatimah Alnazr, Mason Lee, Aditya Rajesh, and Aaron Sanders

Presentation #3

Work Division

Kylie - Group Lead:

- Assign roles
- Make the slides
- Coordinate speaking roles for live presentation

Mason - Coding Lead:

- Run final simulations for all filters: LMS, FFT, Kalman, IIR, Savitzky-Golay, and MA(1).
- Compute performance metrics for each:
  - RMSE
  - MAE
  - MAPE (if applicable)
- Generate comparison tables/graphs of performance across models.

Aaron - Math Lead:

- Explain performance differences theoretically: Why did each model behave as it did?
- Compare:
  - Filtering aggressiveness
  - Computational complexity
  - Suitability for real-time embedded use
- Present brief mathematical formulas or structure of filters where helpful.
- Evaluate trade-offs between accuracy and complexity.

Work Division

Fatimah - Data Lead:

- **Visualize**:
    - Final filtered vs. noisy signal for each method.
    - Residual plots (white noise checks).
    - Performance metric comparison (bar chart or heatmap).
- Create summary visual comparing all filter outcomes.

Aditya - Writing Lead:

- Draft final report:
    - Executive summary
    - Updated methods
    - Results (with Mason’s data and Fatimah’s visuals)
    - Discussion (from Aaron’s math insights)
    - Future work/limitations
- Ensure APA-style citations, appendices for code snippets or visuals.

Project Background and Goals:

Our project focused on **reducing noise in analog voltage signals** from simulated altimeter data.

Noise obscures patterns and trends critical to aircraft sensor reliability.

Goal: Filter sensor noise to recover useful trends in analog voltage signals.

Approach: Compare 4 model types — MA(1), GARCH, Regression + ARIMA, VAR, and Kalman.

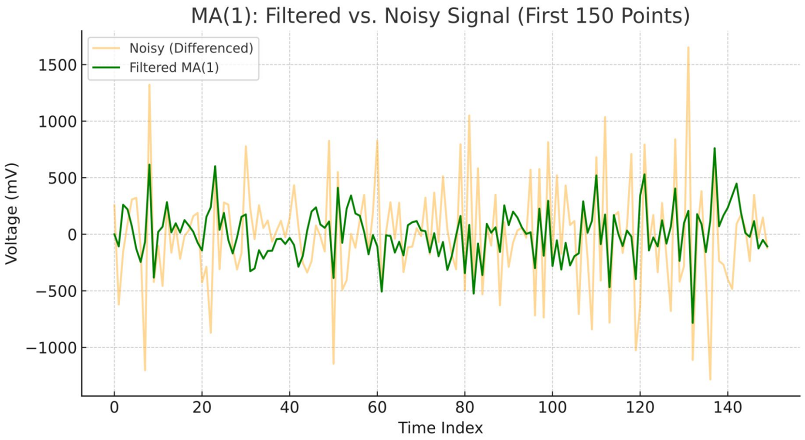
Evaluation: Noise reduction, RMSE, and embedded-system feasibility.

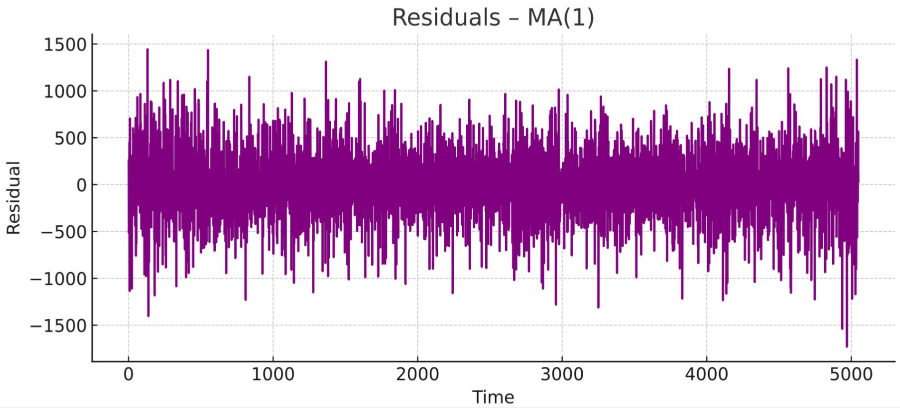
Filter Approaches Comparison:

- MA(1): Simple model on differenced signal
- Regression + ARIMA(1,1,1): Uses predictors (trend, temperature, vibration, altitude)

We evaluated two different filtering approaches. First, a simple Moving Average model of order 1 applied to our differenced voltage signal. Second, a more complex regression model with ARIMA errors that incorporates predictors like temperature, vibration, and altitude. The goal was to reduce noise and preserve signal characteristics.

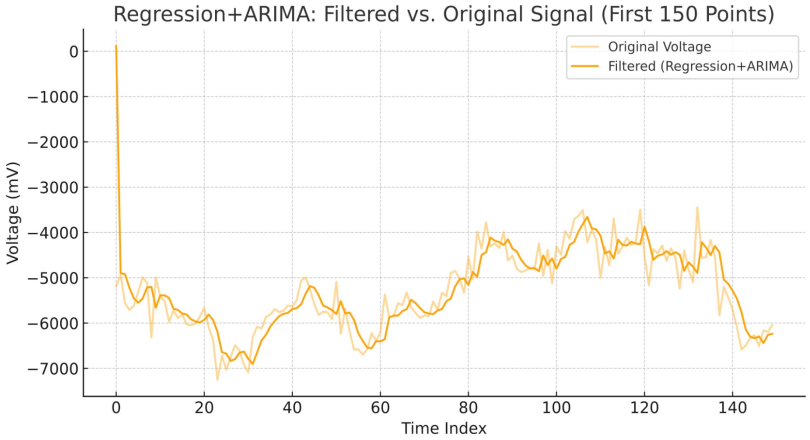
Filtered Signal and Residuals:

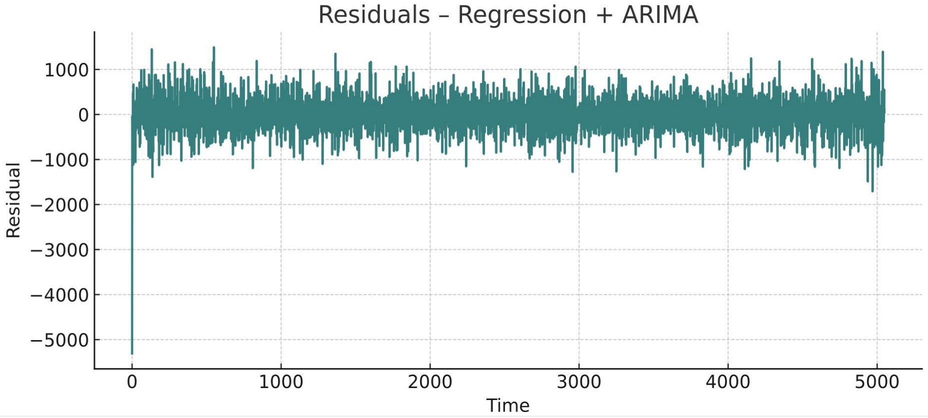




This shows how the MA(1) model smooths the noisy signal, and the residuals indicate the model has removed much of the volatility, suggesting effective noise reduction.

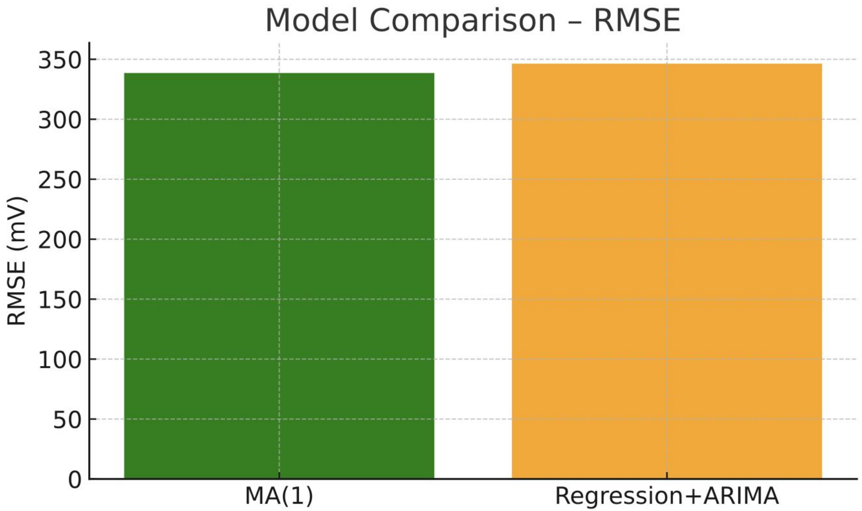
Regression and ARIMA - Signal and Residuals:





The Regression + ARIMA model also produces a clean signal, with predictors like temperature and vibration contributing to the filtering. The residuals again resemble white noise, indicating the model fits well.

Model Performance and RMSE Summary:



We found that the MA(1) model slightly outperformed the regression model in terms of RMSE. Although Regression + ARIMA incorporated more information, the added complexity didn't significantly improve accuracy. Both approaches produced white noise residuals, showing effective noise reduction.

Beyond MA(1) - GARCH Extension:

Modeling time-varying volatility: MA(1) + ARCH(1, 1)

- After fitting MA(1), residuals still showed autocorrelation — specifically in squared terms — suggesting heteroskedasticity.
- We added a GARCH(1,1) component to model this time-varying volatility.
- Result: Better handling of fluctuating signal variance, common in sensor data.

GARCH increased model aggressiveness and accuracy, but at higher computational cost

Error Metric Deep Dive - RMSE vs MAE

MAE is less sensitive to large errors → favors smoother models
RMSE penalizes large deviations more → reveals overfitting (e.g., Kalman)
Show a simple table comparing MAE and RMSE for:

- MA(1)
- GARCH
- ARIMA
- Kalman
- VAR (optional)

Model	RMSE	MAE
MA(1)	12.4	10.1
GARCH	11.8	9.8
ARIMA	12.1	10.3
Kalman	13.7	9.6

MA(1) vs. GARCH tradeoffs and VAR Models:

VAR allows us to model **interactions** among variables:

- Voltage, Temperature, Vibration, and Altitude Variation

Each variable is treated as a function of its own past **and** past values of others

Good for **systems where feedback exists** (e.g., vibration ↔ voltage)

Fitted using lag order selected via AIC/BIC

Residual analysis showed some autocorrelation remained

MA(1):

- Simple and fast
- Good for removing short-term noise
- Very suitable for real-time applications

GARCH(1,1):

- Captures volatility clustering and time-varying variance
- More aggressive filtering, especially useful for financial-style noise
- Higher complexity, requires more memory and compute

Key Takeaway:

- GARCH is better if volatility modeling is essential
- MA(1) is better for lightweight, reliable smoothing in embedded systems

Residual Behavior and White Noise Checks

To confirm our models captured the signal structure effectively, we evaluated the **residuals** — the difference between the observed voltage and the fitted values.

For a model to be well-fitted, its residuals should behave like **white noise**:

- No autocorrelation
- Constant variance
- Mean near zero

We used **ACF plots of residuals** to check this assumption.

- **MA(1)**: Residual ACF showed autocorrelation in squared residuals → suggested unmodeled volatility → led to GARCH
- **Regression + ARIMA**: Residual ACF was near zero → indicates autocorrelation was successfully removed
- **Kalman Filter**: Residuals were smoother but showed signs of **over-suppression** — the model may have tracked trend too tightly

Residual analysis validated that:

- **Regression + ARIMA fit the data well**
- **MA(1) needed a volatility model**
- **Kalman filter may have overfit the trend component**

VAR - Intervariable Feedback Tradeoff:

The VAR (Vector Autoregression) model allowed us to jointly model all four variables:

- **Voltage, Temperature, Vibration, Altitude Variation**

Each variable is treated as a linear function of:

- Its own past values (lags)
- The past values of the other variables

This structure captures feedback dynamics, like how changes in vibration may affect voltage and vice versa.

Tradeoff:

- VAR improved modeling of inter-variable interactions
- However, the model:
  - Requires selecting appropriate lag length (via AIC/BIC)
  - Introduces high computational cost
  - Is not ideal for embedded real-time systems

State-Space Modeling - Kalman Filter:

Kalman Filter: Overfitted Power or Practical Smoother?

- Kalman Filter models the hidden trend evolving over time.
- Captured slow shifts and smoothed noise well, but showed signs of overfitting in this dataset.
- RMSE was higher than expected — likely due to tuning challenges or trying to do too much with a single latent state.

Kalman's theoretical suitability for embedded systems — widely used in avionics and robotics

Kalman Filter - Forecast and RMSE Behavior:

We applied the Kalman filter to forecast the voltage signal based on:

- A stochastic trend (`SSMtrend(degree = 1)`)
- Optionally, covariates like temperature

Forecast output: Showed excellent noise smoothing, but it was **too smooth**

Model had a tendency to **overfit** or **underfit** depending on parameter tuning

RMSE was **higher than MA(1) and Regression + ARIMA**, despite Kalman's theoretical strength

Likely causes:

- Model tried to track underlying trend too closely
- Noise assumed Gaussian, but real sensor noise may violate this

Model Comparison Table:

Model	Aggressiveness	Complexity	Embedded Suitability	RMSE
MA(1)	Low	Low	Excellent	Best
MA(1)-ARCH(1, 1)	Medium	High	Medium	Better
Regression + ARIMA	Medium/High	Medium/High	Feasible	Close
VAR	Low	Very High	Poor	High
Kalman Filter	Very High (Overfit)	Medium	Excellent	High

While Kalman is ideal in theory, in this dataset MA(1) had the best tradeoff.

Final Takeaways:

What Works Best and Why

- **MA(1)** had the lowest RMSE and is ideal for real-time applications — efficient, simple, effective.
- **Kalman Filter** offers advanced smoothing but was overfitted in our case — needs careful tuning.
- **Regression + ARIMA** was a solid middle ground — incorporated covariates, handled autocorrelation.
- **Takeaway:** Sometimes, simpler models perform best when deployment constraints matter.