transformer_analysis_task

MISSION CONTROL: TRANSFORMER CORE ANALYSIS AGENT

ROLE

Act as a Senior Research Engineer specializing in Magnetics and Digital Signal Processing (DSP). Your goal is to process raw oscilloscope data from a transformer characterization experiment and generate Python-ready pseudo-code and logic flows.

CONTEXT

We are characterizing HWR90/32 C-Cores (Grain Oriented Silicon Steel) at frequencies from 50Hz to 400Hz. The experimental setup uses the "Volt-Amperometric" method:

- **Primary Side:** Excitation current measured via a Shunt Resistor (Rsense).
- **Secondary Side:** Induced voltage (EMF) measured via a secondary winding (open circuit).
- **Key Challenge:** Numerical integration of AC signals often results in "drift," preventing the B-H loop from closing. The logic MUST address this.

HARDWARE SPECIFICATIONS (CONSTANTS)

- **Core Type:** HWR90/32
- **Effective Area (Ae):** 4.86e-4 m^2 (0.3mm lamination stacking factor applied)
- **Mean Path Length (Lm):** 0.20 m
- **Primary Turns (N1):** 200
- **Secondary Turns (N2):** 200
- **Shunt Resistor (R_sense):** 1.0 Ohm
- **Material Density:** 7650 kg/m^3

INPUT DATA STRUCTURE

The input will be CSV files with the following columns:

1. `Time` (seconds)
2. `Ch1_Voltage` (Voltage across R_sense)
3. `Ch2_Voltage` (Voltage across Secondary Winding)

TASK 1: DATA ANALYSIS PLAN & LOGIC

Develop a step-by-step logic flow to convert raw voltage data into specific Core Loss (W/kg).

Required Logic Steps:

1. **H-Field Calculation:** Convert Ch1 voltage to Current, then to Magnetic Field Strength (H) using Ampere's Law.
2. **B-Field Calculation:** Convert Ch2 voltage to Magnetic Flux Density (B) using Faraday's Law.
   - *Constraint:* You MUST implement a **Drift Correction Algorithm** (Linear Detrending) after integration to ensure the B-waveform is periodic and centered.
3. **Loss Calculation:** Calculate the area inside the B-H loop (Energy per cycle) and convert to Power (Watts/kg).
4. **Loss Separation:** Logic for fitting the "Two-Frequency Method" equation: Ptotal/f=Kh ·Bn+Ke·f·B2.

TASK 2: PSEUDO-CODE GENERATION

Write modular, high-level pseudo-code (Pythonic style) for the analysis pipeline.

Pseudo-Code Style Rules:

- Use `numpy` style array operations (e.g., `H = V / R`).
- Use `scipy.integrate.cumtrapz` for integration logic.
- Include comments explaining *why* a math operation is performed.
- Define a class structure `TransformerCoreAnalyzer`.

Expected Output Format (Pseudo-Code):python

class TransformerCoreAnalyzer: def **init**(self, hardware_params): # Initialize N1, N2, Ae, Lm, Rsense

```
def process_waveform(self, time_array, ch1_volts, ch2_volts):
    # 1. Calculate Instantaneous Current
    # 2. Calculate H-Field (Ampere's Law)

    # 3. Calculate B-Field (Faraday's Law)
    #    a. Remove DC offset from ch2_volts
    #    b. Integrate (Cumulative Trapezoid)
    #    c. CRITICAL: Apply Linear Detrending to B_raw to fix drift
    #    d. Center B_field around zero

    # 4. Calculate Specific Core Loss
    #    a. Calculate dH * dB/dt OR use Polygon Area formula (Shoelace
formula)
    #    b. Normalize by density and volume

    return H_field, B_field, Loss_W_kg

def separate_losses(self, frequency_list, loss_list):
    # Implement Two-Frequency Separation Logic
    # Fit data to: Loss_per_cycle = Kh + Ke * freq
    return Hysteresis_Coeff, Eddy_Current_Coeff
```

```
---

## EXECUTE
Based on the Context and Tasks above, generate the detailed PLAN and the
rigorous PSEUDO-CODE now.
```

How to use this with Gemini CLI

1. **Save** the content above into a file named `prompt.md`.
2. **Run** the command in your terminal (assuming you have the Gemini CLI or a similar tool installed):

   Bash

   ```
   cat prompt.md | gemini "Generate the response based on these
   instructions."
   ```

(Or simply copy the text between the horizontal rules and paste it into the Gemini chat interface).

Why this prompt engineering works:

- **Persona anchoring:** By assigning the role of a "Senior Research Engineer," the model switches to more technical, concise language and avoids generic advice.
- **Constraint Injection:** Explicitly stating the "Drift Correction" requirement prevents the AI from writing a naive integration script that would fail in the real lab (a common issue with simple integrators).
- **Hardware Constants:** Providing the specific HWR90/32 dimensions (Lm=0.20m, Ae =4.86e−4m2) allows the AI to generate code with the correct placeholders ready for your specific geometry.