Research Proposal: Multi-Scale Plasticity in Test-Time Training (TTT)

1. Core Concept

This project investigates **Multi-Scale (Hierarchical) Plasticity** in Test-Time Training (TTT) models.

Standard TTT treats every layer of a neural network as equally "plastic" (updatable) during inference. We propose that different layers should adapt at different rates. This mimics biological intelligence, where sensory cortices adapt rapidly to new stimuli (e.g., light adaptation) while higher association cortices maintain stable, long-term semantic models.

The Hypothesis: We hypothesize that applying a non-uniform learning rate schedule across layers during test-time training can maintain performance (perplexity) while significantly reducing computational cost (FLOPs) compared to the standard "uniform update" baseline.

2. Motivation & Neuroscience Inspiration

- **Biological Plausibility:** The brain does not update all synapses uniformly. "Fast weights" (sensory buffers) operate on a rapid timescale, while "Slow weights" (semantic knowledge) resist quick changes.
- **The "Drift" Problem:** Updating *all* weights during a long test sequence risks "catastrophic forgetting" of the model's pre-trained logic or initial system prompt.
- **Efficiency:** If the majority of necessary adaptation occurs in the superficial layers (adjusting to local syntax or style), freezing the deep layers eliminates the need to backpropagate through the entire depth of the 1.3B parameter model, saving substantial memory and compute.

3. Differentiation from Existing Work

- **vs. Google's Nested Learning / HOPE:** Those approaches are *structural*, requiring specialized architectures trained from scratch (Pre-training focus).
- **Our Approach (Functional):** We apply "fast" and "slow" dynamics to generic architectures (TTT-Linear) purely at inference time. This is a **Test-Time Optimization** intervention that requires no pre-training or architectural changes.

4. Methodology

We will modify the update rule of the **TTT-Linear-1.3B** model using JAX. Instead of a single scalar learning rate η, we introduce a layer-dependent masking vector $\lambda^{(l)}$ implemented via `optax.multi_transform`:

$$\theta_{t+1}^{(l)} = \theta_t^{(l)} - (\eta \cdot \lambda^{(l)}) \nabla \mathcal{L}$$

Experimental Profiles (The "Plasticity Schedules")

We will define static "Plasticity Profiles" (λ) to test which parts of the network are essential for adaptation:

1. **The "Retina" Profile (Early-Only):**
   - **Settings:** High η for Layers 0-4, Frozen ($\eta = 0$) elsewhere.
   - **Hypothesis:** Sufficient for syntax/style adaptation; highest FLOP savings.
2. **The "Cortex" Profile (Late-Only):**
   - **Settings:** Frozen for Layers 0-20, High η for deep layers.
   - **Hypothesis:** Necessary for semantic adaptation but risks "drift."
3. **The "Uniform" Profile (Baseline):**
   - **Settings:** Standard TTT (all layers update equally).
   - **Hypothesis:** Maximum plasticity but highest cost and drift risk.

5. Key Research Questions

1. **The Efficiency Trade-off:** Can we retain >95% of TTT's perplexity reduction while updating only the first 25% of layers?
2. **The Stability Question:** Does freezing deep layers prevent "Instruction Drift" (overfitting to local noise) compared to the uniform baseline?
3. **Gradient Sensitivity:** Do gradient norms naturally decay in earlier layers, or is explicit freezing required to enforce stability?

6. Experimental Roadmap (2-Day Sprint)

Setup

- **Model:** `TTT-Linear-1.3B` (JAX/Flax implementation).
- **Data:** A subset of the Pile (validation set) to ensure rapid iteration.

Experiments

1. **Baseline Run:** Run standard TTT (Uniform Profile) to establish the "compute ceiling" and best-case perplexity.
2. **Ablation Study:** Run the "Retina" and "Cortex" profiles.
3. **Analysis:** Plot **Perplexity (y-axis) vs. FLOPs/Updates (x-axis)** for all three profiles.

Success Criteria

A successful project will demonstrate a Pareto frontier where "Retina" or "Cortex" profiles achieve similar perplexity to the baseline but with significantly fewer gradient updates or faster wall-clock time.