Non■Traditional Online Training Methods for Always■On Models

Design goal: learning that emerges from information flow statistics and internal regulation rather than fixed dataset epochs or a single scalar objective.

1. Information■Theoretic Reinforcement

bullet Drive adaptation where present latent improves future latent predictability.

bullet No external rewards. Objective is internal mutual information growth across small horizons.

bullet Use rolling-window estimators; keep compute O(1) per tick via EMA statistics.

Key equations:

```
Let z_t be latent state; estimate I(z_t; z_{t+\Delta}) ≈ H(z_{t+\Delta}) − H(z_{t+\Delta}|z_t).
```

```
Model H with Gaussian approximation: H(z) ≈ 0.5 · log det(2πe Σ_z). Maintain Σ via exponential moving covariance.
```

```
Conditional entropy via linear predictor: z■_{t+\Delta}=A z_t; residual ε=z_{t+\Delta}−z■; H(z_{t+\Delta}|z_t)≈0.5·log det(2πe Σ_ε).
```

```
Signal g_I = I■(z_t; z_{t+\Delta}) − τ_I (target level τ_I). Plasticity gate: η_eff = clamp(η_0 + k·g_I, η_min, η_max).
```

Online update sketch:

```
for each tick t: observe z_t, z_{t+Δ} update EMA covariance Σ_z and residual covariance Σ_ε from linear probe A (updated by least squares) I_hat = 0.5 * (logdet(Σ_z) - logdet(Σ_ε)) g_I = I_hat - tau_I eta_eff = clamp(eta0 + k * g_I, eta_min, eta_max) apply eta_eff to parameter subsets that reduced Σ_ε this tick
```

2. Entropy■Regulated Flow Control

- bullet Maintain a target entropy band for latent transitions to avoid collapse or chaos.
- bullet Adjust plasticity, noise injection, and dropout to keep transition entropy in range.

Key equations:

$\Delta z_t = z_t - z_{t-1}$. Transition entropy $H_\Delta \approx 0.5 \cdot \log\det(2\pi e\, \Sigma_\Delta)$, with Σ_Δ EMA of Δz_t.

Homeostat $e = (H_\Delta - \mu_T) / \sigma_T$. Plasticity $\lambda = \sigma(\alpha \cdot e)$ mapped to $[\lambda_{min}, \lambda_{max}]$.

Noise scale $\sigma_{noise} = \sigma_0 \cdot \exp(-\beta \cdot e)$ (more noise when entropy is low).

Dropout $p_{drop} = \mathrm{clamp}(p_0 + \gamma \cdot (e>0\ ?\ -e : |e|), 0, p_{max})$.

Online update sketch:

```
per tick: Δz = z_t - z_{t-1}; update Σ_Δ HΔ = 0.5 * logdet(2πe Σ_Δ) e = (HΔ - mu_T) /
sigma_T set learning-rate, noise, and dropout via λ(e), σ_noise(e), p_drop(e)
```

3. Predictive Symbiosis (Cross■Modal Co■Prediction)

bullet Organs co■predict each other, enforcing a shared manifold without labels.

bullet Only update modules whose cross■prediction error improved over their own EMA.

Key equations:

Let f_v(x_t) = z^v_t, g_a(y_t) = z^a_t. Symmetric prediction: z■^v_{t+1} = P_{va}(z^a_t), z■^a_{t+1} = P_{av}(z^v_t).

Local tension L_sym = ||z^v_{t+1} − z■^v_{t+1}||^2 + ||z^a_{t+1} − z■^a_{t+1}||^2.

Gate by improvement δ = L■_prev − L_sym. Update only if δ > 0.

Online update sketch:

```
per tick with both modalities: compute z_v, z_a predict next states via P_va, P_av L =
mse(z_v_next, zhat_v) + mse(z_a_next, zhat_a) if L < ema(L): allow weight updates for
P_va/P_av and encoders; else freeze this tick
```

4. Self■Referential Distillation

- bullet Spawn a shadow network S (small). Train S to imitate main network M online.
- bullet Use divergence D(M,S) as a structural growth signal; reconcile periodically.

Key equations:

```
Outputs o_M, o_S. Distill S by L_distill = ||o_M − o_S||^2.
```

```
Divergence signal g_D = ||o_M − o_S||_2 (or KL if probabilistic).
```

```
Plasticity gate for M: η_eff = η_0 · (1 + κ · g_D) within bounds.
```

Online update sketch:

```
every K ticks: update S ← argmin ||o_M - o_S||^2 with small LR compute g_D = norm(o_M -
o_S) if g_D > τ: increase plasticity or spawn a small adapter layer in M
```

5. Temporal Fractal Replay

bullet Continuously replay stored compressed states at multiple time scales (1, 2, 4, 8… steps back).

bullet Adaptation occurs by aligning present dynamics with multi■scale echoes, preventing drift.

Key equations:

```
Let C_k be a circular buffer with stride 2^k of compressed states c_t.
```

```
For each k, prediction residual r_k = c_t − F(C_k[t]). Minimize ∑_k w_k ||r_k||^2 but only
if it reduces long■horizon variance.
```

```
Variance criterion: update if Var_k^{new} < Var_k^{EMA} − ε.
```

Online update sketch:

```
per tick: for k in {0..K}: fetch echo c_{t-2^k}; predict forward via F r_k = c_t -
F(c_{t-2^k}) if variance(r_k) improves EMA: do a tiny update on F and encoders
```

6. Internal Curiosity Economy

bullet Assign each latent channel i a plasticity budget b_i.

bullet Budget increases with novelty discovery and decreases with easy predictions.

Key equations:

Novelty $n_i = 1 - \cos(z_i, \mu_i)$ where μ_i is EMA centroid of channel i.

Budget update: $b_i \leftarrow \mathrm{clamp}(b_i + \rho_n \cdot (n_i - \tau_n) - \rho_e \cdot \max(0, \tau_e - e_i), b_{min}, b_{max})$,

where e_i is per■channel prediction error, τ_n desired novelty, τ_e minimal error.

Effective LR per channel: $\eta_i = \eta_0 \cdot (b_i / b_{max})$.

Online update sketch:

```
for channel i each tick: compute novelty n_i and error e_i update budget b_i scale
optimizer LR or gradient mask by η_i
```

7. Information■Equilibrium Decay (Synaptic Tagging)

bullet Consolidate only when prediction error variance stabilizes, preventing overfitting to bursts.

bullet Implements long■term retention without global checkpoints.

Key equations:

Track windowed mean μ_E and variance σ_E^2 of error E_t.

Stability score $s = 1 / (1 + \sigma\_E^2)$. Consolidate when $s > s\_{thr}$ and $|\mu\_E - \mu\_E^{EMA}| < \delta$.

Consolidation operator: $\theta \leftarrow (1 - \lambda)\theta + \lambda \cdot \theta$■ where θ■ are weights after micro■updates; λ depends on s.

Online update sketch:

```
per window W: compute μ_E, σ_E^2 if stability high: blend current weights into a
slow-moving consolidated copy otherwise: keep learning fast but do not consolidate
```

8. Signal Resonance Training

bullet Detect resonant structure between input dynamics and internal activations in frequency space.

bullet Strengthen connections that co■oscillate; weaken those that remain phase■incoherent.

Key equations:

```
Compute short FFTs: X_f = FFT(x_{t-L:t}), Z_f = FFT(z_{t-L:t}).
```

```
Cross■power spectral density: S_xz(f) = X_f · conj(Z_f). Coherence C(f) = |S_xz(f)|^2 /
(S_xx(f) S_zz(f)).
```

```
Resonance score R = ∑_f w_f · C(f). Update gate g_R = R − τ_R.
```

```
Weight change ΔW ∝ g_R · (∂z/∂W) projected onto frequencies with highest coherence.
```

Online update sketch:

```
every L ticks: compute FFT windows for input and latent measure coherence C(f); get
resonance R modulate gradients by g_R concentrating updates on resonant pathways
```

Integration Sketch

Combine methods via gates. At each tick, compute signals $\{g_I, e, \delta, g_D, Var_k, b_i, s, g_R\}$. Map them to per■module plasticity multipliers and gradient masks. Ensure O(1) memory per method using EMA statistics and bounded buffers.

Notation: logdet denotes the logarithm of the matrix determinant; Σ terms are covariance estimates tracked online via exponential moving averages; clamp() limits to a range.