Learning Dynamics of LISP: A Gradient-Free Constraint-Satisfaction Family Containing Backpropagation

Vardan Grigoryants VARDANG@SYNOPSYS.COM, GRIG.VAR@GMAIL.COM
Alexander Hakobyan ALEXANH@SYNOPSYS.COM
Synopsys Inc.

Abstract

We characterize the learning dynamics of *Layerwise Inequality-Sign Propagation* (LISP), a credit-assignment rule that is gradient-free and avoids activation derivatives: forward and backward passes execute without autograd, and layers exchange only a constraint-violation vector $\mathbf{d}^{(l)}$ modulated by a binary monotonicity indicator Π_σ. We prove that (i) the raw LISP output update is gradient descent on a smooth squared hinge loss with $O(1/T)$ convergence, and (ii) for ReLU networks without batch normalization, LISP with cross-entropy is *value-equivalent* to backpropagation almost everywhere—placing standard BP as one point inside the gradient-free LISP family. Across >3,000 runs on CIFAR-10/100, LISP trails tuned BP by 0.4–2.8pp on clean data but **outperforms BP+CE by 2.8pp under 40% label noise**; a controlled ablation isolates per-sample RMS normalization as the operative mechanism, and transferring it to standard backprop reproduces the effect.

AI assistance disclosure. Writing assistance from a large language model was used for drafting and editing. All scientific content, experiments, and conclusions reflect the authors' own work.

1. Introduction

The learning dynamics of deep networks are almost universally studied through the lens of gradient flow: weights evolve along $-\nabla\mathcal{L}$, and theoretical analyses characterize convergence in overparameterized regimes [1, 7]. This paper asks: *what are the learning dynamics when gradient computation is removed from hidden layers?*

We study *Layerwise Inequality-Sign Propagation* (LISP), which reformulates training as a *constraint satisfaction* problem. For a C-class classifier with logits $\mathbf{z} \in \mathbb{R}^C$ and true class y, correct classification with margin m requires:

$$z_y - z_j \geq m \quad \forall\, j \neq y. \tag{1}$$

LISP departs from backpropagation in two ways. *First*, the signal propagated through the network is not a loss gradient but a vector of *constraint-violation magnitudes*—non-zero only where the margin inequalities (1) are violated, and sparse by construction. *Second*, at hidden layers this signal is gated not by the activation derivative $\sigma'(z)$ but by a binary *responsiveness indicator* $\Pi_\sigma(z)$, equal to 1 wherever σ is locally strictly increasing and 0 otherwise, applied elementwise: $\Pi_{\text{ReLU}}(z) = \mathbf{1}[z > 0]$, $\Pi \equiv 1$ for sigmoid/tanh (strictly increasing everywhere), and $\Pi \equiv 0$ for step/sign (optionally overridden to $\Pi \equiv 1$, the STE special case). No activation derivatives are computed—only constraint violations and monotonicity are used.

LISP is gradient-free; BP is a value-equivalent special case. Algorithmically, LISP never differentiates an activation: Π_σ is read off from the monotonicity of σ, and no `loss.backward()`

is called. For ReLU networks without batch normalization, $\Pi_{\mathrm{ReLU}} = \mathrm{ReLU}'$ a.e., so LISP with cross-entropy *recovers backpropagation exactly* (Theorem 3)—a *value* equivalence placing BP as one point inside the gradient-free LISP family. For sigmoid/tanh, $\Pi = 1$ replaces the vanishing $\sigma'(z) \to 0$; for step/sign, the override $\Pi \equiv 1$ is the STE special case. Outside this special case the two genuinely diverge; the experiments isolate two explicit sources: **(D1)** per-sample RMS normalization of the error vector, and **(D2)** the BN backward Jacobian—where distinct dynamics such as noise robustness emerge.

Contributions. (C1) Unification. LISP is a gradient-free credit-assignment family in which standard BP appears as a value-equivalent special case (ReLU + CE + no BN; Theorem 3). **(C2) Descent dynamics.** The raw LISP output update is gradient descent on a smooth squared hinge loss (Theorem 1); under update clipping and empirically verified hidden-layer alignment, multi-layer LISP satisfies a descent inequality on S with bounded hidden-layer perturbation (Theorem 4). **(C3) Mechanism.** Per-sample RMS normalization is the operative ingredient for label-noise robustness; transferring it alone to BP reproduces the effect (§4). **(C4) Empirics.** Across >3,000 runs, LISP trails tuned BP by 0.4–2.8pp on clean CIFAR-10/100 but exceeds BP+CE by 2.8pp under 40% label noise.

Related work. Alternatives to backpropagation [19] include feedback alignment [14, 16], local error signals [17], target propagation [13], equilibrium propagation [20], Forward-Forward [8], and error-driven input modulation [5]. The STE [2, 9] replaces σ' with an identity surrogate; Yin et al. [24] analyze its convergence, and LISP's Π generalizes STE by preserving monotonicity. SignSGD [3] exploits gradient signs for compression; LISP propagates constraint-violation signs without computing gradients. Large-margin objectives [22] replace only the loss; LISP defines the entire learning rule.

2. LISP as Per-Sample-Normalized Margin BP

Consider an L-layer network with preactivations $\mathbf{z}^{(l)} = \mathbf{W}^{(l)}\mathbf{h}^{(l-1)} + \mathbf{b}^{(l)}$ and activations $\mathbf{h}^{(l)} = \sigma_l(\mathbf{z}^{(l)})$.

Error signal (output). For each violating class $j \neq y$ (where $z_y - z_j < m$), let $v_j = m - (z_y - z_j) > 0$. The raw signal

$$\mathbf{d}^{(L)}[y] = +\textstyle\sum_{j\neq y} v_j, \quad \mathbf{d}^{(L)}[j] = -v_j \tag{2}$$

satisfies $\mathbf{d}^{(L)} = -\nabla_{\mathbf{z}^{(L)}} S$, where

$$S \;=\; \tfrac{1}{2N} \sum_i \sum_{c\neq y_i} \big[\max\big(0,\, m - (z_{y_i} - z_c)\big)\big]^2.$$

LISP-margin uses the single worst violator $\arg\max_j v_j$; LISP-CE replaces (2) by $\mathbf{d}^{(L)} = \mathbf{y} - \mathrm{softmax}(\mathbf{z})$, the negative softmax-CE gradient.

Raw backward propagation. For $l = L, \ldots, 1$:

$$\Delta\mathbf{W}^{(l)} = \mathbf{d}^{(l)} \, (\mathbf{h}^{(l-1)})^\top / N, \tag{3}$$

$$\mathbf{d}^{(l-1)} = \big((\mathbf{W}^{(l)})^\top \mathbf{d}^{(l)}\big) \odot \Pi_{\sigma_{l-1}}(\mathbf{z}^{(l-1)}), \tag{4}$$

followed by $\mathbf{W}^{(l)} \leftarrow \mathbf{W}^{(l)} + \eta\, \Delta\mathbf{W}^{(l)}$, with optional update clipping $\|\Delta\mathbf{W}^{(l)}\|_F \le \varepsilon \|\mathbf{W}^{(l)}\|_F$.

Practical implementation: two explicit deviations from BP. The experimental version of LISP adds exactly two ingredients to (3)–(4):

(D1) *Per-sample RMS normalization.* Before forming the outer product and before propagation, each sample's error vector is rescaled $\mathbf{d}_i^{(l)} \mapsto \mathbf{d}_i^{(l)} / \|\mathbf{d}_i^{(l)}\|_{\text{rms}}$. This is a sample-wise preconditioner: it equalizes contributions across examples regardless of raw violation magnitude.

(D2) *BN-aware backward.* Through a BN block with scale γ, batch variance Var, and normalized inputs $\hat{\mathbf{x}}$, LISP applies the full BN Jacobian

$$J_{\text{BN}}(\mathbf{d}) \;=\; \tfrac{\gamma}{\sqrt{\text{Var}+\epsilon}}\Big(\mathbf{d} - \tfrac{1}{B}\mathbf{1}\mathbf{1}^\top\mathbf{d} - \tfrac{1}{B}\hat{\mathbf{x}}\hat{\mathbf{x}}^\top\mathbf{d}\Big),$$

identical to BP's; (D1) is then applied to the result.

Theorems 1 and 3 concern the *raw* rule (3)–(4); the role of (D1) is isolated empirically in §4 (BP+NormHinge). For ReLU + softmax-CE without BN, both (D2) and the directional effect of (D1) vanish, and LISP coincides with BP a.e. (Theorem 3). For step/sign activations the override $\Pi \equiv 1$ is the STE [2].

3. Convergence Theory

3.1. Output Layer: Squared-Hinge Gradient Descent

Theorem 1 (Output-Layer Gradient Descent) *With $\mathbf{h}^{(L-1)}$ held fixed, the raw LISP output update $\Delta\mathbf{W}^{(L)} = \frac{1}{N}\sum_i \mathbf{d}_i^{(L)}(\mathbf{h}_i^{(L-1)})^\top$ satisfies $\Delta\mathbf{W}^{(L)} = -\nabla_{\mathbf{W}^{(L)}} S(\boldsymbol{\theta})$, where S is convex and L_S-smooth in $\mathbf{W}^{(L)}$.*

Proof Logits are affine in $\mathbf{W}^{(L)}$ for fixed $\mathbf{h}^{(L-1)}$. Setting $f_{ic} = m - (z_{y_i} - z_c)$, each term $v_{ic}^2/2 = [\max(0, f_{ic})]^2/2$ has gradient $v_{ic} \cdot [-(\mathbf{e}_{y_i} - \mathbf{e}_c)(\mathbf{h}_i^{(L-1)})^\top]$; summing matches (2). ■

Corollary 2 ($O(1/T)$ rate) *With $\eta = 1/L_S$,*

$$\min_{0\le t\le T} S(\mathbf{W}_t^{(L)}) \;\le\; \frac{L_S\,\|\mathbf{W}_0^{(L)} - \mathbf{W}^{(L)*}\|_F^2}{2T}$$

for any feasible $\mathbf{W}^{(L)*}$.

Remark on (D1). Theorem 1 applies to the *raw* signal. The per-sample RMS rescale (D1) replaces $\mathbf{d}_i$ with $\mathbf{d}_i/\|\mathbf{d}_i\|_{\text{rms}}$, downweighting samples with large raw violations and upweighting small ones; the resulting update is a stochastic preconditioner on $-\nabla S$. We isolate its effect via the BP+NormHinge ablation in §4.

Theorem 3 (ReLU Equivalence (no BN)) *For ReLU networks without batch normalization and with cross-entropy error signal $\mathbf{d}^{(L)} = \mathbf{y} - \text{softmax}(\mathbf{z}^{(L)})$, LISP with $\Pi_{ReLU}(z) = \mathbf{1}[z > 0]$ produces the same layerwise error signals as backpropagation almost everywhere, hence identical weight updates.*

Proof For softmax cross-entropy, $\partial\mathcal{L}_{\text{CE}}/\partial\mathbf{z}^{(L)} = \text{softmax}(\mathbf{z}^{(L)}) - \mathbf{y}$, so $\mathbf{d}^{(L)} = -\partial\mathcal{L}_{\text{CE}}/\partial\mathbf{z}^{(L)}$. For ReLU, $\Pi_{\text{ReLU}}(z) = \mathbf{1}[z > 0]$ agrees with $\text{ReLU}'(z)$ a.e. The backward recursions coincide a.e. by induction over layers. ■

Reading. Theorem 3 is a *value* equivalence between two procedurally distinct rules: gradient-free LISP and gradient-based BP produce the same update vector a.e. in the ReLU+CE+no-BN special case, even though LISP never computes σ' or invokes autograd. Empirical deviations from BP outside this case are attributable to the genuinely different Π_σ for non-ReLU activations and to the explicit ingredients (D1)/(D2) of §2.

3.2. Multi-Layer Descent under Update Clipping

We extend Theorem 1 to all layers updating simultaneously. Let $\Delta\theta_t = \eta(\Delta\mathbf{W}^{(1)}, \ldots, \Delta\mathbf{W}^{(L)})$ denote a raw LISP step, and assume: **(A1)** update clipping $\|\Delta\mathbf{W}^{(l)}\|_F \le \varepsilon\|\mathbf{W}^{(l)}\|_F$ for all $l < L$; **(A2)** L_S-Lipschitz gradient of S on the parameter trajectory; **(A3)** hidden-layer alignment: $\langle\nabla_{\mathbf{W}^{(l)}} S,\ \Delta\mathbf{W}^{(l)}\rangle \le 0$ for all $l < L$.

Theorem 4 (Multi-Layer Descent) *Under (A1)–(A3), with $\eta \le 1/L_S$, every LISP step satisfies*

$$S(\theta_{t+1}) \ \le\ S(\theta_t) \ -\ \tfrac{\eta}{2}\,\|\nabla_{\mathbf{W}^{(L)}} S(\theta_t)\|_F^2 \ +\ \tfrac{L_S\,\eta^2\,\varepsilon^2}{2}\sum_{l<L}\|\mathbf{W}^{(l)}\|_F^2. \tag{5}$$

Proof L_S-smoothness gives $S(\theta_{t+1}) \le S(\theta_t) + \langle\nabla S,\ \eta\Delta\theta_t\rangle + \frac{L_S}{2}\|\eta\Delta\theta_t\|^2$. The inner product decomposes as $\sum_l\langle\nabla_{\mathbf{W}^{(l)}} S,\ \eta\Delta\mathbf{W}^{(l)}\rangle$. By Theorem 1, the output-layer term equals $-\eta\|\nabla_{\mathbf{W}^{(L)}} S\|_F^2$; by (A3) each hidden-layer term is ≤ 0. Hence $\langle\nabla S,\ \eta\Delta\theta_t\rangle \le -\eta\|\nabla_{\mathbf{W}^{(L)}} S\|_F^2$. For the quadratic, $\|\eta\Delta\theta_t\|^2 = \eta^2(\|\nabla_{\mathbf{W}^{(L)}} S\|_F^2 + \sum_{l<L}\|\Delta\mathbf{W}^{(l)}\|_F^2)$. Applying (A1) to hidden layers and using $\eta \le 1/L_S$ to absorb the output-layer quadratic into the descent term yields (5). ■

Interpretation. Whenever the residual is dominated by the descent term—which holds for small ε—LISP is a descent method on S. Assumption (A3) requires that hidden-layer updates do not increase S; this is directly supported by per-layer cosine similarity between LISP updates and BP gradients (>99.9% on CIFAR-100; mean ∼74% on CIFAR-10; Appendix D).

4. Experiments

We evaluate LISP across >3,000 runs on CIFAR-10/100 [12] with PlainConvBN (6-layer ConvNet, BN) and CifarResNet (residual variant). LISP-CE uses cosine LR [15], Adam [11], and 600 epochs. All configurations use CutOut [6] except LISP-margin on PlainConvBN. LISP-margin uses SGD with momentum 0.9 and 200 epochs; BP baselines use SGD with cosine LR. CifarResNet runs employ label smoothing [21] and EMA [18]. Extended BP+SGD to 600 epochs shows no improvement beyond epoch 200 (Figure 1, dotted).

4.1. CIFAR-10/100: Matching Backprop

On the standard (uncorrupted-label) benchmarks, LISP-CE+Adam reaches 94.4% on CIFAR-10 PlainConvBN (0.4pp below BP+SGD), 95.8% on CifarResNet (vs. 96.2%, −0.4pp), and 72.7% on CIFAR-100 (vs. 75.5%, −2.8pp). Higher *training* accuracy for LISP-CE on each dataset (99.4% vs. 98.6% on CIFAR-10; 97.9% vs. 95.4% on CIFAR-100) localizes the gap to generalization, not optimization—consistent with Theorem 4 (LISP descends S under our clipping regime). Direct Feedback Alignment [16], another BP-free method using fixed random feedback weights, reaches only 75.1% on CIFAR-10 PlainConvBN under identical conditions (600 epochs, same regularization), a 19.7pp gap that highlights LISP's unique effectiveness among gradient-free alternatives.

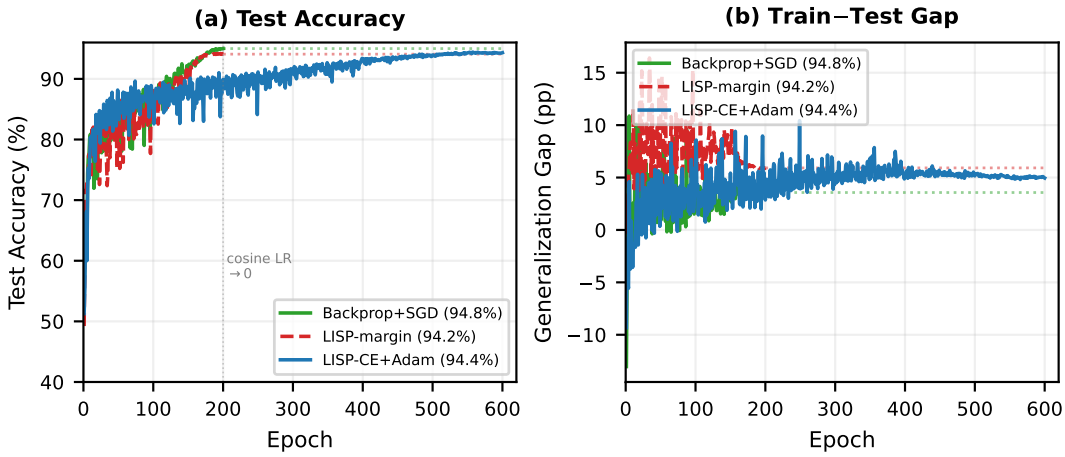

Figure 1: Learning dynamics on CIFAR-10 PlainConvBN. **(a)** Test accuracy: LISP-CE+Adam reaches 94.4% (within 0.4pp of BP+SGD at 94.8%). **(b)** Generalization gap. Dotted: BP plateau after cosine LR ends.

Table 1: Test accuracy (%). Mean±std over 2–3 seeds. DFA included as a BP-free reference; tabular benchmarks in Appendix F.

Dataset	BP+SGD	BP+AdamW	LISP-margin	LISP-CE
CIFAR-10 (PlainConvBN)	**94.8**$_{\pm 0.1}$	93.3$_{\pm 0.2}$	94.2$_{\pm 0.2}$	94.4$_{\pm 0.1}$
CIFAR-10 (CifarResNet)	**96.2**$_{\pm 0.2}$	95.2$_{\pm 0.1}$	95.6$_{\pm 0.1}$	95.8$_{\pm 0.1}$
CIFAR-100 (PlainConvBN)	**75.5**$_{\pm 0.1}$	69.9$_{\pm 0.5}$	72.5$_{\pm 0.1}$	72.7$_{\pm 0.2}$
BP-free reference (CIFAR-10, PlainConvBN, 600 epochs):				
DFA [16]		75.1$_{\pm 1.0}$		

4.2. Label Noise: +2.8pp over BP+CE; mechanism is (D1)

Under 40% uniform label corruption on CIFAR-10 PlainConvBN, **LISP-margin reaches 84.8% vs. 81.9% for BP+CE (+2.8pp; Appendix E)**. Two ablations isolate the cause: (i) BP+Hinge (same loss family as LISP-margin, no normalization) degrades *faster* than BP+CE—ruling out the loss; (ii) BP+NormHinge (BP with ingredient (D1) ported in) *matches* LISP-margin at 40% noise. The operative dynamical phenomenon is the per-sample preconditioner of §3: it caps the influence of mislabelled, large-violation samples regardless of the credit-assignment rule.

References

[1] Zeyuan Allen-Zhu, Yuanzhi Li, and Zhao Song. A convergence theory for deep learning via over-parameterization. In *ICML*, 2019.

[2] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

[3] Jeremy Bernstein, Yu-Xiang Wang, Kamyar Azizzadenesheli, and Animashree Anandkumar. signSGD: Compressed optimisation for non-convex problems. In *ICML*, 2018.

[4] Tianqi Chen and Carlos Guestrin. XGBoost: A scalable tree boosting system. In *KDD*, 2016.

[5] Giorgia Dellaferrera and Gabriel Kreiman. Error-driven input modulation: Solving the credit assignment problem without a backward pass. In *ICML*, pages 4937–4955, 2022.

[6] Terrance DeVries and Graham W. Taylor. Improved regularization of convolutional neural networks with Cutout. *arXiv preprint arXiv:1708.04552*, 2017.

[7] Simon Du, Jason Lee, Haochuan Li, Liwei Wang, and Xiyu Zhai. Gradient descent finds global minima of deep neural networks. In *ICML*, 2019.

[8] Geoffrey Hinton. The forward-forward algorithm: Some preliminary investigations. *arXiv preprint arXiv:2212.13345*, 2022.

[9] Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Binarized neural networks. In *NeurIPS*, 2016.

[10] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. LightGBM: A highly efficient gradient boosting decision tree. In *NeurIPS*, 2017.

[11] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

[12] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

[13] Dong-Hyun Lee, Saizheng Zhang, Asja Fischer, and Yoshua Bengio. Difference target propagation. In *ECML PKDD*, 2015.

[14] Timothy P. Lillicrap, Daniel Cownden, Douglas B. Tweed, and Colin J. Akerman. Random synaptic feedback weights support error backpropagation for deep learning. *Nature Communications*, 7:13276, 2016.

[15] Ilya Loshchilov and Frank Hutter. SGDR: Stochastic gradient descent with warm restarts. In *ICLR*, 2017.

[16] Arild Nøkland. Direct feedback alignment provides learning in deep neural networks. In *NeurIPS*, 2016.

[17] Arild Nøkland and Lars Hiller Eidnes. Training neural networks with local error signals. In *ICML*, 2019.

[18] Boris T. Polyak and Anatoli B. Juditsky. Acceleration of stochastic approximation by averaging. *SIAM Journal on Control and Optimization*, 30(4):838–855, 1992.

[19] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986.

[20] Benjamin Scellier and Yoshua Bengio. Equilibrium propagation: Bridging the gap between energy-based models and backpropagation. *Frontiers in Computational Neuroscience*, 11:24, 2017.

[21] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, 2016.

[22] Yichuan Tang. Deep learning using linear support vector machines. In *ICML Workshop on Challenges in Representation Learning*, 2013.

[23] Joaquin Vanschoren, Jan N. van Rijn, Bernd Bischl, and Luis Torgo. OpenML: Networked science in machine learning. *SIGKDD Explorations*, 15(2):49–60, 2013.

[24] Penghang Yin, Jiancheng Lyu, Shuai Zhang, Stanley Osher, Yingyong Qi, and Jack Xin. Understanding straight-through estimator in training activation quantized neural nets. In *ICLR*, 2019.

Appendix A. Local Direction Preservation

The responsiveness indicator Π_σ is justified by the following observation.

Proposition 5 (Local Direction Preservation) *Let $\sigma : \mathbb{R} \to \mathbb{R}$ be non-decreasing and define $\Delta h = \sigma(z + \Delta z) - \sigma(z)$. If σ is strictly increasing on some interval $(z - \epsilon, z + \epsilon)$, then for all Δz with $0 < |\Delta z| < \epsilon$ we have $\operatorname{sign}(\Delta h) = \operatorname{sign}(\Delta z)$.*

Proof For $0 < |\Delta z| < \epsilon$, both z and $z + \Delta z$ lie in $(z - \epsilon, z + \epsilon)$ where σ is strictly increasing, hence $\sigma(z + \Delta z) > \sigma(z)$ iff $\Delta z > 0$. ■

This formalizes the responsiveness intuition: when a unit is locally responsive, the sign of a small preactivation change matches that of the induced activation change. Π_σ encodes responsiveness, while magnitudes are handled by the per-sample rescale (D1).

Appendix B. Discussion

Unification. The ReLU equivalence theorem (Theorem 3) places standard cross-entropy backpropagation as one point inside a broader gradient-free family: LISP never differentiates an activation, yet for ReLU networks without batch normalization it produces the same weight updates as BP almost everywhere. The same family also subsumes the STE for non-differentiable activations, providing a unified view of constraint-based credit assignment.

Mechanism identification. Where LISP diverges from BP—when per-sample RMS normalization (D1) and BN-aware propagation (D2) are active—we observe improved robustness to label noise (+2.8pp over BP+CE at 40% corruption). The ablation identifying (D1) as the operative ingredient, and the successful transfer of (D1) to standard backprop (BP+NormHinge), yield a concrete methodological insight: per-sample signal normalization caps the influence of mislabelled high-violation samples regardless of the underlying credit-assignment rule.

Open directions. The 0.4–2.8pp generalization gap on clean data persists despite LISP reaching higher training accuracy, pointing to a regularization mismatch rather than an optimization failure. LISP's per-sample normalization equalizes gradient contributions across samples, acting as implicit regularization that differs from BP's. Closing this gap through targeted regularization, extending Theorem 4 to formally account for (D1) and (D2), and relaxing (A3) via a purely structural argument are the natural next steps.

Constraint dropout. We randomly zeroed per-sample error signals during backward propagation with inverted-dropout scaling. This is analogous to standard dropout but applies to the LISP signal path rather than forward activations. Despite testing dropout rates 0.1–0.3, we observed $<$1pp improvement on CIFAR-100, suggesting that regularization mechanisms effective for forward activations do not trivially transfer to the constraint signal path. Standard forward dropout (0.2–0.3) combined with strong weight decay (0.05) proved more effective for LISP generalization.

Appendix C. Computational Efficiency

Table 2 compares per-epoch wall time on CIFAR-10 PlainConvBN. LISP-margin runs at comparable speed to backprop; LISP-CE+Adam has higher per-epoch cost but uses $3\times$ the epochs.

Table 2: Per-epoch wall time (seconds) on CIFAR-10 PlainConvBN (single GPU).

Method	**Time/epoch (s)**	**Epochs**	**Total**
BP+SGD	6.0$\pm$2.4	200	20.0min
LISP-margin	3.7$\pm$0.0	200	12.2min
LISP-CE+Adam	9.6$\pm$2.0	600	1.6h

Appendix D. Gradient Alignment

Figure 2 shows per-layer cosine similarity between LISP weight updates and backpropagation gradients. On CIFAR-100, all layers achieve $>$99.9% alignment throughout training, indicating LISP updates are nearly identical to BP gradients at every layer. On CIFAR-10, alignment averages $\sim$74% with layer-wise variation showing earlier layers (L0–L3) generally have higher alignment than later layers (L4–L7). Both datasets are consistent with the descent picture of Theorem 4: positive alignment validates assumption (A3), and the LISP step is dominated by the output-layer gradient term with a small hidden-layer perturbation residual.

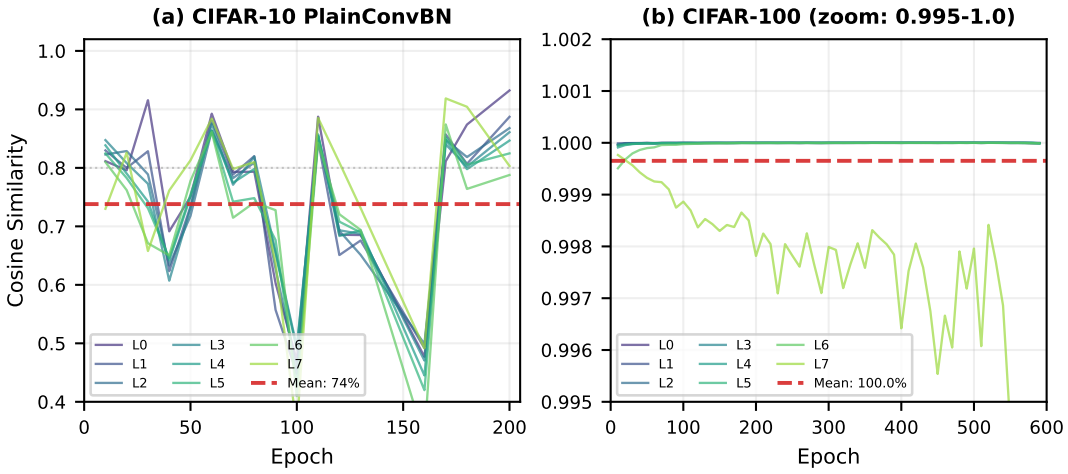

Figure 2: Per-layer cosine similarity between LISP updates and BP gradients. **(a)** CIFAR-10 shows layer-wise alignment (L0–L7) with mean $\sim$74%. **(b)** CIFAR-100 (zoomed y-axis: 0.995–1.0) shows all layers at $>$99.9% alignment.

Appendix E. Label Noise Robustness (Extended)

We corrupt training labels uniformly at random at rates 0%, 10%, 20%, and 40%. Figure 3 shows that LISP-margin degrades more gracefully than BP+CE. Crucially, BP+Hinge (same loss without normalization) degrades *faster* than BP+CE, ruling out the loss function as the source of robustness. BP+NormHinge—backprop with per-sample RMS normalization applied to the gradient—matches LISP's robustness at 40% noise, confirming that the normalization mechanism transfers directly to standard training.

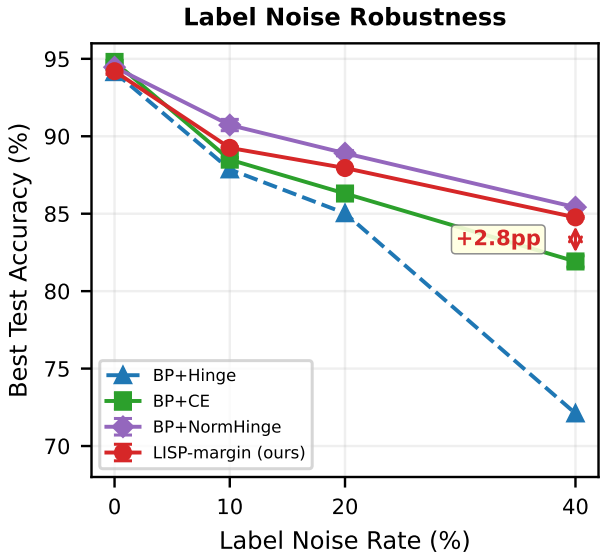

Figure 3: Label noise robustness on CIFAR-10 PlainConvBN. Mean±std over 2–3 seeds. LISP-margin outperforms BP+CE at 40% noise; BP+NormHinge matches LISP, confirming per-sample normalization as the key mechanism.

Appendix F. Tabular Benchmark Results

Table 3 extends the comparison to 8 OpenML tabular benchmarks [23] using a 2-layer MLP (256 hidden units) with 5-fold cross-validation. The best LISP variant matches or exceeds the best backprop MLP on 6 of 8 datasets (including ties). Tree-based methods (XGBoost, LightGBM) dominate on the largest dataset (electricity, 45k samples) [4, 10].

Table 3: Test accuracy (%) on 8 OpenML tabular benchmarks. Best **neural** method per dataset is **bolded**. 2-layer MLP (256 hidden), mean±std over 5-fold CV.

Dataset	XGBoost	LightGBM	BP+SGD	BP+AdamW	LISP-margin	LISP-CE
iris (150/4/3)	94.0±3.3	94.7±4.0	96.0±4.9	95.3±5.8	**96.0**±4.9	**96.0**±4.9
wine (178/13/3)	98.3±1.4	97.8±2.1	98.9±1.4	98.3±1.4	98.3±1.4	**100.0**±0.0
blood (748/4/2)	75.0±2.2	75.0±1.3	81.4±0.5	81.3±0.9	81.3±0.6	**81.6**±1.1
vehicle (846/18/4)	76.4±1.8	76.1±1.3	**85.2**±2.2	84.2±2.7	83.1±1.1	84.3±2.3
credit-g (1k/7/2)	67.1±0.6	66.2±2.2	**71.0**±3.3	69.5±1.9	69.9±3.1	69.5±2.1
segment (2.3k/19/7)	98.6±0.5	98.4±0.5	98.2±0.4	98.1±0.4	98.0±0.5	**98.2**±0.4
phoneme (5.4k/5/2)	90.1±0.5	89.8±0.5	90.0±0.8	89.2±0.5	**90.0**±0.7	89.9±0.6
electricity (45k/7/2)	88.2±0.3	88.0±0.3	82.7±0.4	83.4±0.3	81.9±0.4	**83.8**±0.4

Appendix G. Alternative Propagation Modes

The default LISP backward rule uses the *magnitude* mode where $d^{(\ell)} = d^{(\ell+1)} \cdot \mathbf{1}[z^{(\ell)} > 0]$ for ReLU activations. We also implement an *inverse* mode that maps desired post-activation changes to pre-activation space via $\Delta z = \sigma^{-1}(h + \Delta h) - \sigma^{-1}(h)$, which amplifies signals near saturation (opposite of vanishing gradients).

For saturating activations (sigmoid, tanh), inverse mode provides an alternative to the $\Pi \equiv 1$ responsiveness used by default. For ReLU, inverse mode reduces to magnitude mode since σ^{-1} is identity on positive reals—the two are mathematically equivalent. Table 4 compares test accuracy on CIFAR-10 PlainConvBN for sigmoid and tanh under identical hyperparameters.

Table 4: Test accuracy (%) comparing propagation modes on CIFAR-10 PlainConvBN for saturating activations. All runs: LISP-Adam, EMA stabilization, cosine LR, 600 epochs, label smoothing 0.1, weight decay 0.005, no cutout, no dropout.

Activation	**Magnitude**	**Inverse**	Δ
Tanh	89.9	90.1	+0.2
Sigmoid	86.8	87.3	+0.5

Despite the theoretical appeal of amplifying signals at saturation, inverse mode shows no significant improvement over magnitude mode ($\pm$0.5pp). The simpler magnitude mode ($\Pi \equiv 1$) is therefore recommended for saturating activations; it matches inverse mode's accuracy while avoiding the numerical overhead of computing σ^{-1}.