BOUNDED PRECISION-GEOMETRY SCALING FOR ROBUST MULTI-TASK LEARNING UNDER LOSS SCALE MISMATCH

Anonymous authors
Paper under double-blind review

ABSTRACT

Multi-task learning often combines losses that span several orders of magnitude, causing homoscedastic uncertainty weighting to degrade severely. We propose Bounded Precision-Geometry Scaling (BPGS), a method that maps each task's log-variance through a bounded sigmoid parameterisation anchored to detached batch loss statistics, and decouples network optimisation from uncertainty optimisation. Its normalised task weights are provably invariant to uniform rescaling under non-degenerate loss scales. We evaluate BPGS on synthetic stress tests and three real-world benchmarks: NYUv2 dense prediction, Yeast multi-label classification, and RF1 multi-target regression. Under pure loss rescaling from $\times 1$ to $\times 1000$, its macro score changes from 0.777 to 0.778, whereas Kendall weighting drops from 0.780 to 0.637; ℓ_1-normalising Kendall's weights does not close the gap. On NYUv2, BPGS records the lowest depth absolute relative error (0.223), depth RMSE (0.790), and total loss (1.891) among all compared methods, including Nash-MTL. Sensitivity studies on batch size and calibration show small variation across the tested ranges, and runtime overhead relative to Kendall is under 1%. BPGS posts the highest Yeast micro-F1 (0.616) and is competitive on RF1, though PCGrad leads RMSE and MAE there. These findings establish BPGS as a scale-robust alternative to homoscedastic uncertainty weighting, notably effective when loss-scale disparities dominate multi-task optimization.

1 INTRODUCTION

In multi-task learning, raw losses often differ in magnitude; for example, a pixel-level cross-entropy may dominate a regression residual by several orders of magnitude. If the weighting mechanism scales with loss magnitude, tasks with larger losses dominate the shared gradients, diminishing the updates for smaller-loss tasks, preventing effective training, and destabilising learned task-weighting parameters Caruana (1997).

Homoscedastic uncertainty weighting addresses this disparity by learning one log-variance per task Kendall et al. (2018). This works well when the losses have similar scales. However, under severe scale disparity, its unconstrained parameters can translate the scale gap into large precision shifts, which can destabilise optimisation. PCGrad Yu et al. (2020), a gradient surgery method, tries to resolve directional conflicts among task gradients but leaves the scalar loss weighting problem unaddressed.

To address the gap left by the aforementioned methods, we propose Bounded Precision-Geometry Scaling (BPGS). BPGS confines every task's learned uncertainty to a finite interval by mapping each latent coordinate through a sigmoid and re-centring the output on detached batch log-loss statistics. A split objective keeps weight adaptation and network training on separate computational paths. We study whether this bounded, batch-anchored parameterisation can remain robust to severe loss-scale disparities without hurting performance on standard multi-task benchmarks.

We make four contributions:

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

1. We introduce BPGS as a bounded, batch-aware, split uncertainty-weighting method and prove that its normalised task weights are strictly invariant to uniform loss rescaling under non-degenerate loss scales.

2. Across three orders of magnitude of loss scaling ($\times 1$–$\times 1000$), BPGS maintains stable performance, whereas standard homoscedastic weighting Kendall et al. (2018) degrades markedly; adding an ℓ_1 normalisation constraint does not recover this stability.

3. On the NYUv2 dataset, BPGS outperforms competitive baselines, including Nash-MTL Navon et al. (2022), in depth estimation accuracy and aggregate multi-task loss. Detailed ablations isolate the empirical contributions of the batch-anchored parameterisation, first-batch calibration, and decoupled stop-gradient objective.

4. We systematically evaluate sensitivity to batch size, calibration parameters, and runtime overhead on NYUv2, and demonstrate that the performance gains of BPGS extend to multi-task tabular benchmarks (Yeast and RF1).

2 Related Work

Multi-task networks typically optimise shared representations across multiple objectives Caruana (1997) and often rely on adaptive weighting schemes to balance task contributions. BPGS builds directly on homoscedastic uncertainty weighting Kendall et al. (2018), which assigns every task a learned log-variance that simultaneously scales the task loss and regularises the uncertainty estimate. Other adaptive methods reweight tasks based on training dynamics: GradNorm Chen et al. (2018) balances gradient magnitudes to synchronise learning progress, while Dynamic Task Prioritization Guo et al. (2018) emphasises lagging objectives. In contrast, BPGS reparameterises the learned variance through a bounded mapping anchored to detached batch loss statistics. The design specifically targets loss-scale robustness rather than directional gradient conflict.

Gradient manipulation methods resolve task conflict directly in gradient space rather than in the loss domain. Sener and Koltun Sener & Koltun (2018) frame multi-task learning as multi-objective optimisation and adapt the Multiple Gradient Descent Algorithm (MGDA) Désidéri (2012) to converge toward Pareto-stationary solutions. Extending this paradigm, gradient surgery techniques explicitly alter the update direction to minimise negative transfer: PCGrad Yu et al. (2020) orthogonally projects conflicting task gradients, while CAGrad Liu et al. (2021a) regularises the update toward conflict-averse directions. Game-theoretic and accelerated multi-objective formulations such as Nash-MTL Navon et al. (2022), IMTL-G Liu et al. (2021b), and FAMO Liu et al. (2023) similarly operate in gradient space by solving per-step optimisation subproblems. In contrast, BPGS operates in the loss domain through scalar uncertainty reparameterisation, neither modifying gradient geometry nor adding iterative optimisation overhead. Our benchmarks compare BPGS against standard loss-weighting methods alongside representative gradient-level methods, including Nash-MTL on NYUv2 and PCGrad on tabular tasks.

The resulting comparison set spans both paradigms, scoped to the scale-robustness question rather than a comprehensive multi-task optimisation survey.

3 Method

3.1 Problem setup

Let T denote the number of tasks and w the shared network parameters. The loss vector

$$L(w) = \big(L_1(w), \ldots, L_T(w)\big) \tag{1}$$

collects all raw task losses for the current batch. We seek adaptive task weights that stay informative even when these losses differ by orders of magnitude. BPGS approaches the problem by learning one latent uncertainty coordinate per task and mapping it into a bounded log-variance chart, as described next.

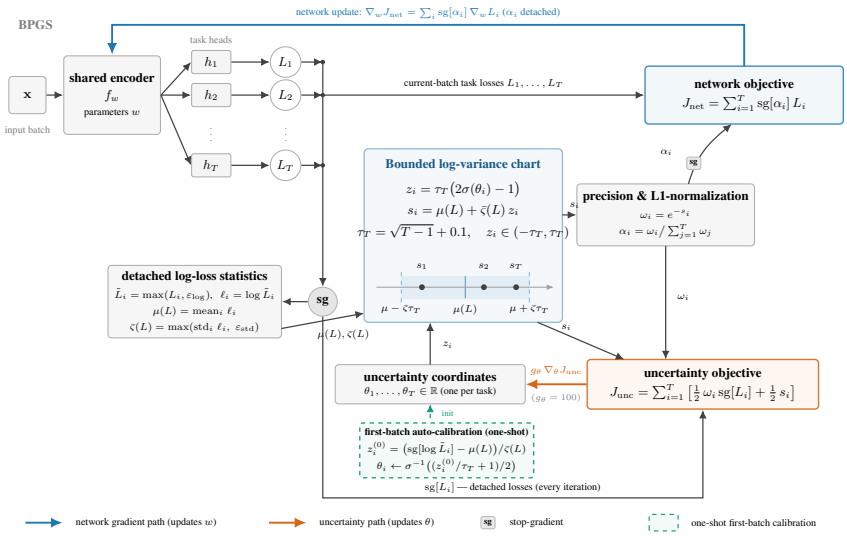

Figure 1: Schematic overview of BPGS. Detached statistics $\mu(L), \bar{\varsigma}(L)$ anchor a bounded chart for s_i; precisions give weights α_i. Network and uncertainty passes are split with stop-gradients; see §3.2–3.4.

3.2 Canonical BPGS chart

The batch-aware form of BPGS begins by computing detached log-loss statistics from the current minibatch:

$$\widetilde{L}_i = \max(L_i, \varepsilon_{\log}), \qquad \mu(L) = \frac{1}{T}\sum_{i=1}^{T} \text{sg}[\log \widetilde{L}_i], \tag{2}$$

and

$$\bar{\varsigma}(L) = \max\left(\sqrt{\frac{1}{T}\sum_{i=1}^{T}\big(\text{sg}[\log \widetilde{L}_i] - \mu(L)\big)^2}, \varepsilon_{\text{std}}\right), \tag{3}$$

where $\varepsilon_{\log}, \varepsilon_{\text{std}} > 0$ are small numerical stability floors preventing zero-argument logarithms and degenerate zero variance. Each task owns a single learnable uncertainty coordinate $\theta_i \in \mathbb{R}$. The latent radius depends on the task count:

$$\tau_T = \sqrt{T-1} + 0.1, \tag{4}$$

and the bounded latent coordinate is

$$z_i(\theta_i) = \tau_T\big(2\sigma(\theta_i) - 1\big), \tag{5}$$

where $\sigma(\cdot)$ is the logistic sigmoid. Combining z_i with the batch statistics gives the canonical BPGS log-variance:

$$s_i(\theta_i; L) = \mu(L) + \bar{\varsigma}(L)\, z_i(\theta_i). \tag{6}$$

The radius in Eq. (4) follows from the geometry of a population-standardised T-vector. In the unclipped regime ($\bar{\varsigma}(L) > \varepsilon_{\text{std}}$), its coordinates have zero mean and unit average square. If one coordinate has magnitude M, the other $T-1$ coordinates must sum to $-M$; their squared sum is minimised when they are equal. Hence $T \geq M^2 + (T-1)(M/(T-1))^2$, giving $M \leq \sqrt{T-1}$. The added 0.1 is a numerical margin that keeps first-batch inverse-sigmoid calibration away from the chart boundary. From s_i we obtain the task precision

$$\omega_i(\theta_i; L) = e^{-s_i(\theta_i; L)}, \tag{7}$$

and the network weight by ℓ_1 normalisation:

$$\alpha_i(\theta_i; L) = \frac{\omega_i(\theta_i; L)}{\sum_{j=1}^{T} \omega_j(\theta_j; L)}. \tag{8}$$

The net effect: every task's log-variance lives in a finite, batch-anchored interval rather than on the entire real line.

Before the first network update, BPGS runs a one-shot auto-calibration step. It maps the detached log-loss statistics of the initial batch into the bounded chart to set the starting θ_i values. The ablation study (Section 5.3) tests this initialisation rule separately from the batch-aware chart.

3.3 Split optimisation objectives

Network parameters and uncertainty coordinates are trained under separate objectives. The network loss is

$$J_{\text{net}}(w \mid \theta, L) = \sum_{i=1}^{T} \text{sg}[\alpha_i(\theta_i; L)]\, L_i, \tag{9}$$

where $\text{sg}[\cdot]$ denotes stop-gradient. The uncertainty objective is

$$J_{\text{unc}}(\theta \mid L) = \sum_{i=1}^{T} \left[\frac{1}{2}\omega_i(\theta_i; L)\, \text{sg}[L_i] + \frac{1}{2} s_i(\theta_i; L) \right]. \tag{10}$$

While Eq. (10) shares its loss form with Kendall et al. Kendall et al. (2018), optimization is performed with respect to the latent coordinate θ_i rather than an unconstrained log-variance. Confining $s_i(\theta_i; L)$ to a batch-conditional interval prevents unbounded parameter drift and fundamentally stabilizes the optimization dynamics.

A fixed gradient scale $g_\theta = 100$ is applied to the uncertainty update across all reported configurations. It compensates for the smaller magnitudes of the uncertainty gradients and affects only update speed, not the weighting rule itself (full optimiser settings: Appendix A).

Both passes draw on the same current-batch losses at every iteration. In J_{net}, the normalised weights are detached so that weight changes do not reshape the network gradient. Symmetrically, J_{unc} detaches the losses and batch statistics. This separation prevents the two updates from interfering within a single step.

3.4 Batch-conditional boundedness

The formal property underpinning the robustness argument is that s_i is bounded conditionally on the current batch. Fix a batch L and take any finite θ_i:

$$\mu(L) - \bar{\varsigma}(L)\tau_T < s_i(\theta_i; L) < \mu(L) + \bar{\varsigma}(L)\tau_T. \tag{11}$$

The bound follows from $\sigma(\theta_i) \in (0, 1)$, which forces

$$z_i(\theta_i) \in (-\tau_T, \tau_T). \tag{12}$$

Substituting into the affine map for s_i yields the stated interval. As a consequence, the induced precision $\omega_i(\theta_i; L)$ is strictly positive and bounded for every fixed batch. Note the qualifier: the guarantee is batch-conditional, not globally independent of the observed loss statistics.

Boundedness alone does not ensure learnability near the boundary. As z_i approaches $\pm\tau_T$, the sigmoid derivative vanishes and recovery may stall. We inspected the logged trajectories from all main-paper runs (Appendix G). The most extreme ratio observed was $|z_i|/\tau_T = 0.93$, occurring at initialisation on NYUv2; by the final epoch, the caps settled to at most 0.84 (NYUv2), 0.69 (Yeast), and 0.89 (RF1). Even the closest point to the boundary retained a derivative factor of roughly 0.033—about 13% of its maximum—and final-epoch values reached approximately 0.074. No logged run exhibited boundary locking. This margin is empirical, not a formal guarantee for arbitrary tasks or training regimes.

3.5 Uniform rescaling invariance

Proposition 1 (Uniform rescaling invariance). *Let $L'_i = cL_i$ for all tasks $i \in \{1, \ldots, T\}$ with scalar multiplier $c > 0$. For any loss vector satisfying $L_i > \varepsilon_{\log}$ and $\bar{\varsigma}(L) > \varepsilon_{\text{std}}$ on both batches, the normalised task weights satisfy $\alpha_i(\theta_i; L') = \alpha_i(\theta_i; L)$ for all i. First-batch auto-calibration also produces identical initial θ_i values under the same rescaling.*

Proof. Uniform rescaling adds $\log c$ to every log-loss. Therefore $\mu(L') = \mu(L) + \log c$ while $\bar{\varsigma}(L') = \bar{\varsigma}(L)$, and $z_i(\theta_i)$ is unchanged. Hence $s_i(\theta_i; L') = s_i(\theta_i; L) + \log c$, so each raw precision is multiplied by the same factor $1/c$, which cancels in the ℓ_1 normalisation. The standardised coordinates used for first-batch calibration are unchanged by the same additive shift. □

This result is about normalised weights on a single batch; it says nothing about full optimiser trajectories or final predictive metrics. The pure-rescaling experiment (Section 5.1) probes that broader empirical behaviour.

4 Experimental Setup

4.1 Benchmarks

We evaluate BPGS across different benchmark and stress settings. The main dense-prediction comparison uses the NYUv2 multi-task benchmark Silberman et al. (2012), which contains 795 training images and 654 validation images with three tasks: semantic segmentation, depth estimation, and surface-normal prediction. This comparison runs for 120 epochs and reports final-epoch metrics aggregated over seeds.

A separate NYUv2 ablation study uses a fixed 50% training subset with the validation split unchanged. It compares four BPGS variants against Kendall uncertainty weighting to isolate the bounded chart and the initialisation rule: the canonical batch-aware auto-calibrated form, a batch-aware fixed-initialisation form, a stateless fixed-initialisation form, and a stateless auto-calibrated form. The ablation runs for 60 epochs over the same three training seeds.

Two real-data benchmarks evaluate the generalisability of the gains beyond dense prediction. Yeast Elisseeff & Weston (2001) is a 14-label multi-label classification task with 1,500 training and 917 validation examples. RF1 Spyromitros-Xioufis et al. (2016) is an 8-target regression task with 4,108 training and 5,017 validation examples. Both report final task metrics aggregated over three seeds from runs of up to 120 epochs.

Four controlled studies reuse the same NYUv2 50% subset. The stop-gradient study pairs canonical BPGS against a coupled variant. The batch-size study sweeps sizes 4, 8, and 16. Both run for 60 epochs over three seeds. A first-batch study varies only the loader seed (1001, 2001, 3001) while fixing the model seed, isolating calibration sensitivity. The overhead study records per-epoch wall-clock time and peak CUDA memory for BPGS versus Kendall over 60 epochs and three seeds.

Three synthetic studies test scale robustness. The pure loss-rescaling study applies post-hoc multipliers $\times 1$, $\times 10$, $\times 100$, and $\times 1000$ to check invariance to raw loss magnitude. The scale-stress study uses the same grid in a synthetic multi-task setting where increasing scale imbalance can degrade all methods. The heterogeneous mixed-stress study adds three regimes—*clean*, *conflict*, and *noisy*—to test what happens when tasks differ not only in scale but also in interaction structure.

A separate diagnostic asks whether normalisation alone explains the rescaling result: it compares BPGS, Kendall, and Kendall with ℓ_1-normalised precision weights on the same rescaling grid. This run uses seeds 42, 123, and 999 and is reported apart from the main rescaling table.

4.2 Compared methods and metrics

The comparison set varies by benchmark. On NYUv2, the main comparison includes BPGS, Static, Kendall uncertainty weighting Kendall et al. (2018), UWSO, and Nash-MTL Navon et al. (2022). The ablation restricts to the four BPGS variants against Kendall. Yeast and RF1 compare BPGS against Static, Kendall, PCGrad Yu et al. (2020), a GradNorm-inspired gradient-norm proxy Chen et al. (2018), and UWSO. The pure loss-rescaling and scale-stress studies compare BPGS, Kendall, and UWSO. The heterogeneous mixed-stress study uses the same three plus PCGrad.

For NYUv2, we report segmentation mIoU, depth absolute relative error, depth RMSE, mean angular error for normals, normal accuracy within 11.25°, total validation loss, and in the main benchmark also Δ_M against the Static baseline Liu et al. (2019). Yeast metrics are macro-F1, micro-F1, and Hamming accuracy. RF1 metrics are R^2, RMSE, and MAE. The synthetic stress studies use macro

Table 1: Pure loss rescaling: Macro Score across scale factors. Mean ± std over 3 seeds.

Method	$\times 1$	$\times 10$	$\times 100$	$\times 1000$
Kendall	$\mathbf{0.780 \pm 0.007}$	0.769 ± 0.013	0.714 ± 0.017	0.637 ± 0.015
UWSO	0.622 ± 0.073	0.674 ± 0.036	0.688 ± 0.035	0.681 ± 0.014
BPGS	0.777 ± 0.012	$\mathbf{0.784 \pm 0.008}$	$\mathbf{0.781 \pm 0.001}$	$\mathbf{0.778 \pm 0.006}$

score as the primary statistic, with worst-task score added for the heterogeneous analysis and stress diagnostics.

4.3 Reporting protocol

Unless stated otherwise, values are means ± standard deviations across seeds. The main NYUv2, Yeast, RF1, rescaling, and heterogeneous-stress results use seeds 42, 43, and 44. The synthetic scale-stress study in Table 5 uses 10 seeds. Synthetic stress results aggregate per-seed summaries, whereas NYUv2, Yeast, and RF1 tables use the final recorded epoch. The supplementary materials contain the code, configurations, and reproduction instructions.

Because checkpoint summaries are not uniformly available across methods, mixing best- and final-checkpoint numbers would introduce method-dependent extraction rules. We therefore use final epoch metrics for all paper tables. Final NYUv2 runs select checkpoints by `val/miou` in max mode, while the ablation and controlled studies use the dataset default `val/total_loss` in min mode. The uncertainty update uses $g_\theta = 100$ and full settings are in Appendix A.

5 Results

5.1 Robustness to loss-scale mismatch

Figure 2 separates two perturbation types as the right column isolates pure post-hoc loss rescaling, where only the numerical scale changes. In that setting, BPGS is near-invariant: its macro score stays at 0.777, 0.784, 0.781, and 0.778 from $\times 1$ through $\times 1000$ (Table 1 and Appendix Figure 4). Proposition 1 shows that the normalised weights are invariant under this rescaling in the non-degenerate regime. The table verifies that this invariance carries through to training outcomes. Kendall-style uncertainty weighting starts slightly higher at $\times 1$, but drops monotonically to 0.637 at $\times 1000$, while UWSO improves from a lower $\times 1$ baseline.

Normalisation is not sufficient. On a separate three-seed diagnostic, ℓ_1-normalised Kendall improves over unnormalised Kendall at $\times 1000$ (0.689 versus 0.658) but still falls by 0.105 from $\times 1$ to $\times 1000$. BPGS falls by 0.004 (Appendix B). This diagnostic uses a different seed set from the main rescaling table, so we treat it as evidence that normalisation alone is insufficient, not as a seed-matched estimate of the main-table gap.

The left column of Figure 2 is the difficult test as the perturbation changes the optimisation problem, not only its units. All three methods degrade as the factor increases, but BPGS leads at every tested multiplier, reaching 0.543 at $\times 1000$ versus 0.499 for Kendall and 0.494 for UWSO. The relative drop from $\times 1$ to $\times 1000$ is 27.0% for BPGS, compared with 32.3% for Kendall and 28.5% for UWSO (Appendix Table 6; Appendix Figure 3). The BPGS–Kendall gap at $\times 1000$ is 0.044, or about 1.1 combined standard deviations, a comparison of means rather than a formal significance test.

The heterogeneous mixed-stress study is less favourable to BPGS than the scale-mismatch studies (Appendix Table 7 and Appendix Figure 6). Kendall and PCGrad attain higher macro scores in the noisy and conflict regimes, whereas BPGS gives the strongest worst-task score in the clean regime at 0.301 while UWSO collapses to 0 throughout. These results limit the rescaling advantage to scale-dominated settings and do not extend it to conflict-dominated ones.

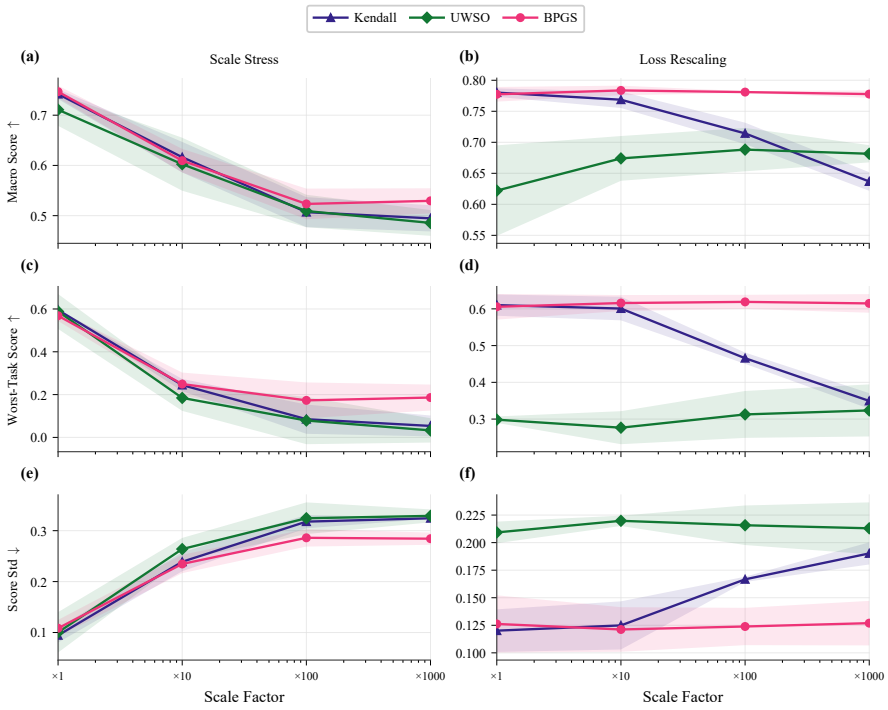

Figure 2: Synthetic robustness diagnostics under scale mismatch. Left column: scale stress, where performance is evaluated as task scales become increasingly imbalanced. Right column: pure loss rescaling, where only the reported magnitude changes. BPGS remains nearly invariant under pure rescaling and retains the strongest macro score at the largest scale-stress factors.

Table 2: NYUv2 dense prediction benchmark. All methods trained for 120 epochs. Final-epoch metrics reported as mean ± std over 3 seeds. Δ_M is computed against the Static baseline; the Nash-MTL value is derived from the reported mean metrics and is shown approximately.

Method	mIoU ↑	Abs Rel ↓	RMSE ↓	Angle ↓	Within 11.25° ↑	Total Loss ↓	Δ_M ↑
Static	$\mathbf{0.341 \pm 0.006}$	0.239 ± 0.000	0.829 ± 0.011	35.934 ± 2.055	0.169 ± 0.009	1.992 ± 0.033	0.000 ± 0.000
Kendall	0.281 ± 0.013	0.229 ± 0.001	0.808 ± 0.003	28.629 ± 0.431	0.231 ± 0.007	1.977 ± 0.029	0.022 ± 0.000
UWSO	0.294 ± 0.007	0.228 ± 0.003	0.815 ± 0.012	$\mathbf{26.110 \pm 0.383}$	$\mathbf{0.271 \pm 0.007}$	1.936 ± 0.023	0.059 ± 0.000
BPGS	0.312 ± 0.001	$\mathbf{0.223 \pm 0.002}$	$\mathbf{0.790 \pm 0.010}$	26.851 ± 0.334	0.257 ± 0.005	$\mathbf{1.891 \pm 0.014}$	$\mathbf{0.078 \pm 0.000}$
Nash-MTL	0.252 ± 0.027	0.242 ± 0.009	0.856 ± 0.029	30.180 ± 1.010	0.211 ± 0.011	2.071 ± 0.070	≈ -0.038

5.2 Main NYUv2 benchmark

Table 2 shows that no method dominates every metric. Static achieves the best segmentation mIoU, and UWSO is strongest on both surface-normal metrics. BPGS leads on depth and on the aggregate: best absolute relative error (0.223), depth RMSE (0.790), total loss (1.891), and Δ_M (0.078). Appendix Figures 9–12 visualise these per-metric comparisons.

Relative to the strongest competing values, BPGS improves absolute relative error from 0.228 under UWSO to 0.223 and RMSE from 0.808 under Kendall to 0.790, and lowers total loss from 1.936 to 1.891. The NYUv2 comparison therefore shows that the bounded parameterisation holds its own on a standard dense-prediction benchmark while retaining its scale-robustness properties.

Nash-MTL has the weakest mean mIoU (0.252) and largest mIoU standard deviation (0.027), though its within-11.25° accuracy (0.211) exceeds Static (0.169). It serves as a recent multi-objective reference point under the same protocol. The single-benchmark comparison does not permit broader conclusions.

Table 3: Ablation study: BPGS variants vs. Kendall uncertainty weighting on NYUv2. All methods trained for 60 epochs. Mean ± std over 3 seeds.

Method	mIoU ↑	Abs Rel ↓	RMSE ↓	Angle ↓	Within 11.25° ↑	Total Loss ↓
Kendall	0.137 ± 0.006	0.294 ± 0.010	0.990 ± 0.016	38.681 ± 0.936	0.130 ± 0.005	2.546 ± 0.035
BPGS (canonical)	0.128 ± 0.008	$\mathbf{0.283 \pm 0.005}$	$\mathbf{0.988 \pm 0.043}$	$\mathbf{33.834 \pm 1.068}$	$\mathbf{0.161 \pm 0.015}$	$\mathbf{2.488 \pm 0.075}$
BPGS (BA-fixed)	$\mathbf{0.143 \pm 0.006}$	0.291 ± 0.005	0.994 ± 0.022	39.097 ± 0.478	0.130 ± 0.004	2.524 ± 0.047
BPGS (SL-auto)	0.058 ± 0.006	0.331 ± 0.003	1.305 ± 0.015	34.661 ± 1.310	0.147 ± 0.024	6.142 ± 1.332
BPGS (SL-fixed)	0.120 ± 0.007	0.287 ± 0.006	1.013 ± 0.025	35.183 ± 0.963	0.142 ± 0.010	2.566 ± 0.051

Table 4: Real-data transfer benchmarks. Final-epoch task metrics reported as mean ± std over 3 seeds. Higher is better for Yeast metrics and RF1 R^2; lower is better for RF1 RMSE and MAE.

Method	Macro-F1 ↑	Micro-F1 ↑	Hamming Acc ↑	R^2 ↑	RMSE ↓	MAE ↓
Static	0.430 ± 0.008	0.610 ± 0.009	0.772 ± 0.003	-0.453 ± 0.127	29.875 ± 0.485	23.376 ± 0.451
Kendall	0.431 ± 0.003	0.609 ± 0.001	0.768 ± 0.002	-0.439 ± 0.164	29.801 ± 0.821	23.219 ± 0.790
UWSO	0.142 ± 0.014	0.486 ± 0.008	0.767 ± 0.002	$\mathbf{-0.408 \pm 0.193}$	30.144 ± 1.137	23.449 ± 1.230
PCGrad	0.430 ± 0.007	0.604 ± 0.010	0.769 ± 0.002	-0.432 ± 0.107	$\mathbf{29.596 \pm 0.385}$	$\mathbf{23.098 \pm 0.368}$
GradNormProxy	0.408 ± 0.028	0.597 ± 0.014	0.772 ± 0.001	-0.508 ± 0.248	30.033 ± 0.482	23.380 ± 0.582
BPGS	$\mathbf{0.436 \pm 0.006}$	$\mathbf{0.616 \pm 0.008}$	$\mathbf{0.773 \pm 0.003}$	-0.429 ± 0.213	30.158 ± 0.952	23.548 ± 0.908

5.3 Ablation on chart and initialisation

Table 3 isolates the batch-aware chart from initialisation. The stateless auto-calibrated variant collapses (mIoU 0.058; total loss 6.142): without the batch-aware chart, first-batch calibration alone cannot sustain training. Appendix Figure 7 traces this collapse in the validation curves, and Appendix Figure 8 contrasts the task-weight trajectories.

Among the stable variants, the best values split between the two batch-aware forms. The canonical batch-aware BPGS variant leads on five of six metrics: depth absolute relative error, depth RMSE, mean angular error, normal accuracy within $11.25°$, and total loss. The batch-aware fixed-initialisation variant leads on mIoU. The stateless fixed variant never matches the canonical batch-aware variant on any metric and trails both batch-aware forms on mIoU and total loss. The batch-aware chart is therefore the critical component. The initialisation rule shifts which task benefits most, without destabilising the overall run.

A separate ablation isolates the split stop-gradient while keeping the bounded, batch-aware chart and initialisation fixed (Appendix E). Removing it alters the uncertainty trajectory: $\theta_{\max}$ grows from 3.11 to 3.99 instead of decaying from 3.08 to 2.44, and its final seed-to-seed standard deviation rises from 0.014 to 0.192. Segmentation mIoU drops from 0.198 to 0.190. The split is therefore a substantive design choice, not an implementation detail.

5.4 Sensitivity to batch statistics and calibration

The batch-aware chart was stable over the tested batch sizes. Across batch sizes 4, 8, and 16, seed-averaged validation metrics have coefficients of variation below 3.5%, with most below 1.2%; final $\theta_{\max}$ ranges from 2.40 to 2.59 (Appendix E). When only the first loader batch is changed, all reported task metrics have coefficients of variation below 2%, and the final $\theta_{\max}$ values differ by at most 0.3%. These studies bound the claim to the tested ranges; they do not establish insensitivity to arbitrary batch statistics.

5.5 Transfer to Yeast and RF1

The real-data benchmarks separate methods less clearly than the synthetic stress tests, so we read them as consistency checks rather than ranking evidence. On Yeast, BPGS is best on all three reported metrics, with macro-F1 0.436, micro-F1 0.616, and Hamming accuracy 0.773 (Table 4; Appendix Figure 13). On RF1, BPGS is not the top method: PCGrad gives the lowest RMSE and MAE, and UWSO gives the best mean R^2. BPGS nonetheless posts the second-best R^2 (-0.429) with RMSE and MAE within about 2% of the best values (Appendix Figure 14).

5.6 Computational overhead

On the NYUv2 50% subset, BPGS takes 40.979 seconds per epoch versus 40.916 for Kendall and uses 3368 MB peak GPU memory versus 3339 MB: increases of 0.15% and 0.87%, respectively (Appendix F). The method adds three task-level parameters to an 18.9M-parameter network. At this scale, the batch statistics and separate uncertainty update add no measurable efficiency penalty.

6 Discussion and Limitations

Collectively, the empirical results validate BPGS across its primary design objectives: controlled rescaling tests verify its invariance under extreme loss-scale disparities, while standard multi-task benchmarks show that this stability does not worsen general task performance. Consequently, BPGS provides a targeted, scale-robust loss-weighting mechanism that complements existing gradient optimisers instead of attempting to replace them.

The study has clear limits. Our empirical evaluation is restricted to a focused setup that includes NYUv2, two tabular benchmarks, and a synthetic scaling experiment. Extending the validation to additional dense-prediction benchmarks is an immediate next step. The primary benchmark results reflect three random seeds (with 10-seed evaluations reserved for scaling analyses), preventing definitive fine-grained ranking claims among closely performing baselines. As BPGS regulates scalar loss weights rather than gradient trajectories, gradient surgery methods like PCGrad and loss-weighting baselines like Kendall retain an advantage in regimes where directional gradient conflict dominates over scale imbalance. Combining BPGS with such methods is untested future work.

From an optimisation standpoint, bounding the uncertainty parameters introduces a theoretical risk of gradient saturation near the chart boundaries. As reported in Section 3.4, our experiments show an empirical maximum of $|z_i|/\tau_T = 0.93$ without observing saturation, but this operating margin could change under different architectures or learning rates. Furthermore, our theoretical analysis establishes batch-conditional boundedness and normalised-weight invariance but does not extend to formal convergence rates or asymptotic optimality. The diagnostic sensitivity analyses cover only batch sizes 4–16 on NYUv2, and our dense-prediction evaluation includes Nash-MTL as the primary representative baseline, leaving benchmarks against other multi-objective and meta-weighting methods (e.g., CAGrad, IMTL-G, FAMO, and Auto-Lambda Liu et al. (2022)) to future empirical studies.

The runtime study quantifies cost only on the controlled NYUv2 subset and one GPU setting. Potential societal impact is indirect. BPGS may improve multi-output models used in different settings by providing steadier task weights, but downstream deployments still inherit the data, labeling, and evaluation failures of their tasks. In safety-sensitive settings, improving one task while weakening another can be obscured by aggregate scores. We therefore report per-task metrics and state where BPGS is not the best method.

7 Conclusion

We propose BPGS, a bounded, batch-anchored reparameterisation of uncertainty weighting designed for multi-task learning under disparate loss scales. Theoretically, BPGS guarantees that normalised task weights remain invariant to uniform loss rescaling under non-degenerate loss scales. Empirically, macro scores shift by less than 0.01 across the $\times 1$–$\times 1000$ rescaling range, demonstrating substantially greater stability than existing baselines under severe scale disparities. Across multiple benchmarks, BPGS achieves the lowest aggregate multi-task loss and yields the best depth metrics and lowest total loss on the NYUv2 dataset, while other methods lead on segmentation and normals; on tabular tasks such as Yeast and RF1, it yields leading or highly competitive performance.

We do not claim a universal ranking, and BPGS targets regimes where loss-scale disparity, not directional gradient conflict, dominates instability, complementing rather than replacing gradient-space methods. Future work includes broader dense-prediction evaluation and hybrids combining bounded weighting with conflict-aware optimisers.

AI USE STATEMENT

Generative AI tools were used to provide external feedback on mathematical derivations, correct grammar and improve language, generate initial figure drafts, and assist with the literature review. The authors independently reviewed all mathematical derivations and verified the relevant literature against the sources. The authors manually revised and corrected AI-generated figure drafts using the underlying source data before inclusion. All other aspects of the work were carried out by the authors. The authors take full responsibility for the final content of this work, including all text, claims, figures, and other artefacts produced with the assistance of generative AI.

ETHICS STATEMENT

This work studies loss weighting on standard public benchmarks (NYUv2, Yeast, RF1) and synthetic stress tests. It does not involve human subjects, sensitive data, or high-risk generative models. Per-task metrics are reported, including cases where the proposed method is not best, to avoid masking per-task failures behind aggregate scores.

REPRODUCIBILITY STATEMENT

Datasets, splits, seeds, budgets, and hyperparameters are described in the Experimental Setup and Appendix A. Proofs appear in the Method section. Code and configurations for reproduction will be provided as anonymized supplementary material at submission.

REFERENCES

Rich Caruana. Multitask learning. *Machine Learning*, 28(1):41–75, 1997. doi: 10.1023/A:1007379606734.

Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 794–803, 2018.

Jean-Antoine Désidéri. Multiple-gradient descent algorithm (MGDA) for multiobjective optimization. *Comptes Rendus Mathematique*, 350(5–6):313–318, 2012. doi: 10.1016/j.crma.2012.03.014.

André Elisseeff and Jason Weston. A kernel method for multi-labelled classification. In *Advances in Neural Information Processing Systems*, pp. 681–687, 2001.

Michelle Guo, Albert Haque, De-An Huang, Serena Yeung, and Li Fei-Fei. Dynamic task prioritization for multitask learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 282–299, 2018.

Alex Kendall, Yarin Gal, and Roberto Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7482–7491, 2018.

Bo Liu, Xingchao Liu, Xiaojie Jin, Peter Stone, and Qiang Liu. Conflict-averse gradient descent for multi-task learning. In *Advances in Neural Information Processing Systems*, 2021a.

Bo Liu, Yihao Feng, Peter Stone, and Qiang Liu. FAMO: Fast adaptive multitask optimization. In *Advances in Neural Information Processing Systems*, 2023.

Liyang Liu, Yi Li, Zhanghui Kuang, Jing-Hao Xue, Yimin Chen, Wenming Yang, Qingmin Liao, and Wayne Zhang. Towards Impartial Multi-Task Learning. In *International Conference on Learning Representations*, 2021b. URL https://openreview.net/forum?id=IMPnRXEWpvr.

Shikun Liu, Edward Johns, and Andrew J. Davison. End-to-end multi-task learning with attention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.

Shikun Liu, Stephen James, Andrew J. Davison, and Edward Johns. Auto-lambda: Disentangling dynamic task relationships. *Transactions on Machine Learning Research*, 2022.

Aviv Navon, Aviv Shamsian, Idan Achituve, Haggai Maron, Kenji Kawaguchi, Gal Chechik, and Ethan Fetaya. Multi-task learning as a bargaining game. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 16428–16446, 2022.

Ozan Sener and Vladlen Koltun. Multi-task learning as multi-objective optimization. In *Advances in Neural Information Processing Systems*, 2018.

Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from RGBD images. In *European Conference on Computer Vision*, 2012.

Eleftherios Spyromitros-Xioufis, Grigorios Tsoumakas, William Groves, and Ioannis Vlahavas. Multi-target regression via input space expansion: treating targets as inputs. *Machine Learning*, 104(1):55–98, 2016. doi: 10.1007/s10994-016-5546-z.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. In *Advances in Neural Information Processing Systems*, 2020.

A Reproducibility Details

Unless stated otherwise, all experiments aggregate results over three random seeds (42, 43, and 44), except the synthetic scale-stress benchmark, which evaluates over 10 seeds (42–51). For real-world benchmarks (NYUv2, Yeast, and RF1), metrics are reported at the final training epoch across all methods to ensure uniform evaluation without checkpoint selection bias.

Optimization and numerical safeguards. All neural network models are trained using AdamW with linear learning-rate warmup and cosine decay. In BPGS, the batch log-loss statistics $\mu(L)$ and $\bar{\varsigma}(L)$ as well as the normalized weights α_i are detached during network updates, while task losses are detached during uncertainty updates with fixed gradient scale $g_\theta = 100$. Numerical stability floors are set to $\varepsilon_{\log} = 10^{-8}$ for logarithmic arguments and $\varepsilon_{\mathrm{std}} = 10^{-4}$ for standard deviation. Gradient clipping is set to 10.0 for BPGS and 1.0 for Kendall.

Main NYUv2 benchmark. The standard NYUv2 benchmark contains 795 training and 654 validation images (288×384 resolution) evaluated on a 3-task SegNet architecture (≈ 18.9M parameters): 13-class semantic segmentation (cross-entropy loss, ignore index 255), depth estimation (masked L1 loss), and surface-normal prediction (cosine distance loss). Training runs for 120 epochs with batch size 8, initial learning rate 5×10^{-4} decaying with cosine schedule to 10^{-6} (100 warmup steps), weight decay 10^{-4}, gradient clipping 10.0, deterministic CPU data augmentation, and mixed precision.

NYUv2 ablation. The ablation uses a fixed 50% training subset (398 images, stratified by majority semantic class and depth quartile using seed 11) with the same SegNet architecture and validation split. Models are trained for 60 epochs with batch size 8, initial learning rate 5×10^{-4} (decaying to 10^{-6} after 100 warmup steps), weight decay 10^{-4}, and mixed precision.

Yeast and RF1. Yeast and RF1 use 4-layer shared-trunk MLPs (hidden dimension 256, dropout 0.1). Yeast consists of 1,500 training and 917 validation examples for 14-label classification under binary cross-entropy loss; models are trained for up to 120 epochs with batch size 128, learning rate 5×10^{-4} decaying to 10^{-5} (200 warmup steps), weight decay 10^{-4}, gradient clipping 10.0, and early stopping patience of 15 epochs on validation loss. RF1 consists of 4,108 training and 5,017 validation examples for 8-target regression under MSE loss, trained with batch size 256 and identical optimizer settings.

Synthetic stress benchmarks. Synthetic scale-stress uses $T = 4$ tasks over 10 seeds (42–51); pure loss-rescaling uses $T = 5$ tasks over seeds 42, 43, and 44; and heterogeneous mixed-stress uses $T = 8$ tasks across three regimes: *clean* ($\rho = 0.45, \eta = 0.05$), *noisy* ($\rho = 0.15, \eta = 0.14$), and *conflict* ($\rho = -0.35, \eta = 0.10$).

Baseline methods. The `GradNormProxy` baseline implements a gradient-norm balancing proxy inspired by GradNorm. The UWSO baseline implements an analytical inverse-loss weighting rule with fixed softmax temperature $T_{\text{temp}} = 2.0$.

Additional NYUv2 and diagnostic studies. The stop-gradient, batch-size, first-batch, and computational overhead studies all use the 50% NYUv2 subset for 60 epochs over three seeds (42, 43, 44). For Nash-MTL, training follows the 120-epoch NYUv2 protocol with batch size 8, AdamW learning rate 5×10^{-4} (cosine decay to 10^{-6}, 100 warmup steps), weight decay 10^{-4}, mixed precision, and gradient clipping 1.0. The normalization ablation evaluates on the pure loss-rescaling grid over seeds 42, 123, and 999.

Reproducibility disclosures. All experiments use standard public benchmarks cited in the main text. Complete source code, configuration files, and reproduction scripts are provided as anonymized supplementary material at submission. Computational overhead measurements are systematically reported for the controlled benchmark setting in Appendix F.

B Additional Stress Results

Table 5: Synthetic scale stress: Macro Score across scale factors. Mean ± std over 10 seeds.

Method	$\times 1$	$\times 10$	$\times 100$	$\times 1000$
Kendall	0.738 ± 0.017	0.607 ± 0.038	0.522 ± 0.030	0.499 ± 0.025
UWSO	0.690 ± 0.034	0.604 ± 0.040	0.525 ± 0.035	0.494 ± 0.024
BPGS	$\mathbf{0.743 \pm 0.018}$	$\mathbf{0.616 \pm 0.039}$	$\mathbf{0.547 \pm 0.032}$	$\mathbf{0.543 \pm 0.032}$

Table 6: Relative Macro Score change (%) from $\times 1 \rightarrow \times 1000$. Positive values indicate degradation; negative values indicate improvement.

Method	Scale Stress	Loss Rescaling
Kendall	32.3%	18.4%
UWSO	28.5%	$\mathbf{-9.5}\%$
BPGS	$\mathbf{27.0}\%$	0.0%

Table 7: Heterogeneous mixed stress: Macro Score and Worst-Task Score across regimes. Mean ± std over 3 seeds.

Method	Clean		Noisy		Conflict	
	Macro	Worst	Macro	Worst	Macro	Worst
Kendall	0.687 ± 0.010	0.267 ± 0.056	$\mathbf{0.659 \pm 0.007}$	$\mathbf{0.242 \pm 0.045}$	$\mathbf{0.664 \pm 0.012}$	$\mathbf{0.211 \pm 0.041}$
UWSO	0.437 ± 0.028	0.000 ± 0.000	0.438 ± 0.020	0.000 ± 0.000	0.433 ± 0.013	0.000 ± 0.000
PCGrad	$\mathbf{0.688 \pm 0.016}$	0.274 ± 0.055	0.656 ± 0.007	0.241 ± 0.046	0.660 ± 0.011	0.205 ± 0.048
BPGS	0.679 ± 0.017	$\mathbf{0.301 \pm 0.052}$	0.641 ± 0.013	0.176 ± 0.076	0.652 ± 0.012	0.198 ± 0.063

C Ablation and NYUv2 Figures

Nash-MTL's per-step coefficients varied substantially across batches in all three runs. This observation is descriptive only: the present logs do not distinguish solver variation from an intrinsic property of the method on this benchmark.

Table 8: Normalization ablation on pure loss rescaling. Macro score (mean $\pm$ std over seeds 42, 123, and 999). L1-normalizing Kendall improves the largest-scale result but does not reproduce the BPGS scale sensitivity.

Scale	BPGS	Kendall	Kendall+L1
$\times 1$	0.789 ± 0.018	0.788 ± 0.014	0.794 ± 0.017
$\times 10$	0.791 ± 0.015	0.781 ± 0.023	0.780 ± 0.022
$\times 100$	0.788 ± 0.016	0.734 ± 0.037	0.742 ± 0.029
$\times 1000$	0.785 ± 0.017	0.658 ± 0.042	0.689 ± 0.034
$\Delta(\times 1 \to \times 1000)$	-0.004	-0.130	-0.105

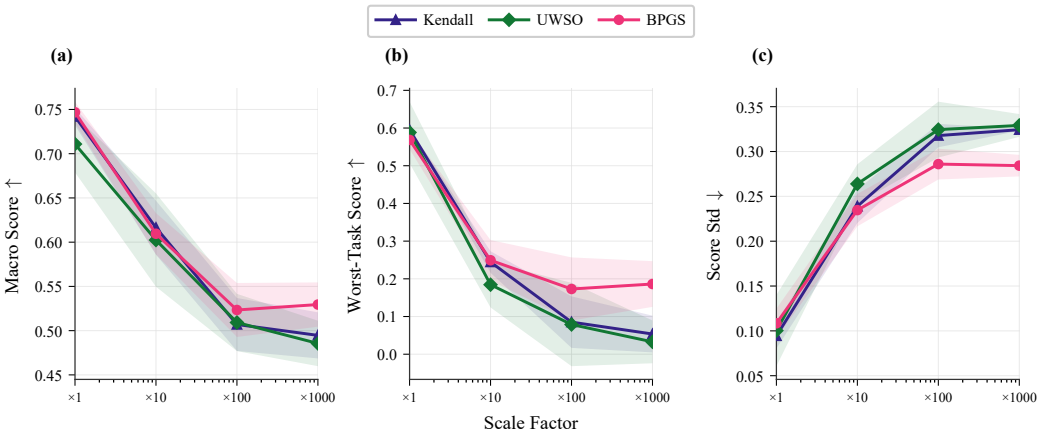

Figure 3: Synthetic scale-stress curves. As the scale factor increases, all methods degrade, but BPGS retains the strongest macro score at every tested factor and tracks the strongest worst-task score at the larger stress levels.

D Real-Data Training Curves

E Sensitivity Analyses

The stop-gradient study changes only the coupling of the uncertainty update. The batch-size and first-batch studies are controlled diagnostics on the NYUv2 subset, not claims about all dataset sizes, batch regimes, or initialization procedures.

F Computational Overhead

The measurements report the controlled BPGS–Kendall comparison on a single GPU setting. They quantify the added batch-statistics computation and uncertainty pass, but do not establish a hardware-independent efficiency guarantee.

G Bounded-Chart Behavior

We reconstruct $z_i = (s_i - \mu)/\bar{\varsigma}$ from the logged batch statistics and log-variances. At the closest observed point, $\sigma(\theta_i)(1 - \sigma(\theta_i)) \approx 0.033$, about 13% of its maximum 0.25; by the final epochs the corresponding value is approximately 0.074. These measurements show an empirical margin from saturation in the logged runs, not a general guarantee.

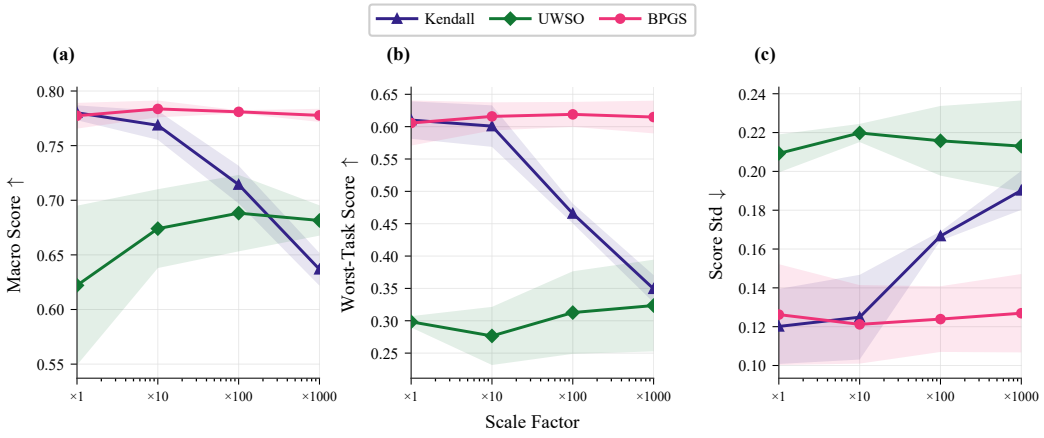

Figure 4: Pure loss-rescaling curves. BPGS remains nearly invariant across the entire rescaling range, while Kendall degrades sharply at larger factors and UWSO improves only from a substantially weaker starting point.

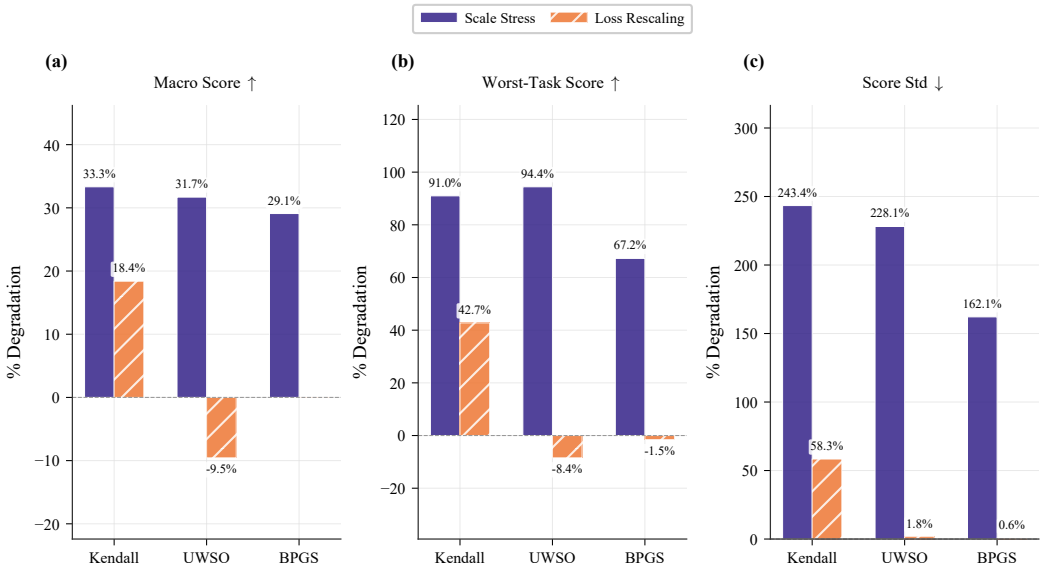

Figure 5: Relative degradation comparison between synthetic scale stress and pure loss rescaling. The figure shows that BPGS has the smallest macro-score degradation under scale stress and essentially no macro-score degradation under pure rescaling.

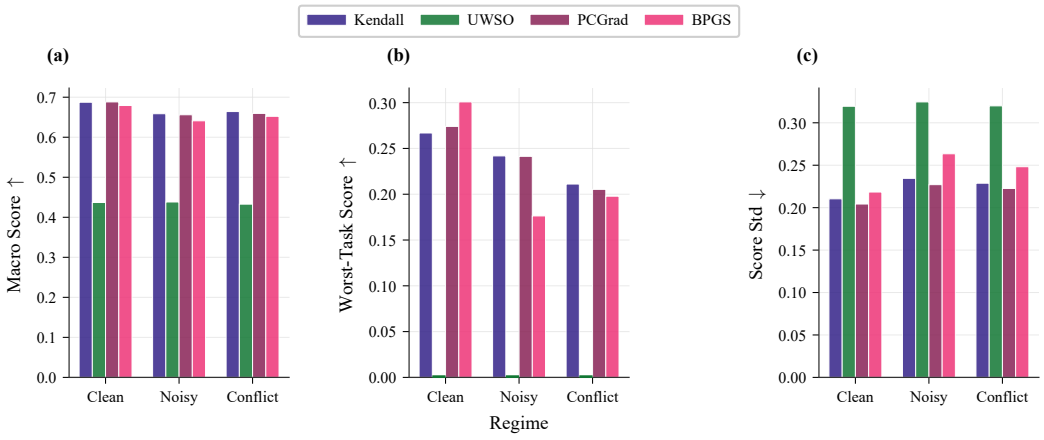

Figure 6: Heterogeneous mixed-stress summary. BPGS is not the best method in every regime, but it achieves the strongest worst-task score in the clean regime and remains well above the degenerate UWSO worst-task score across all regimes.

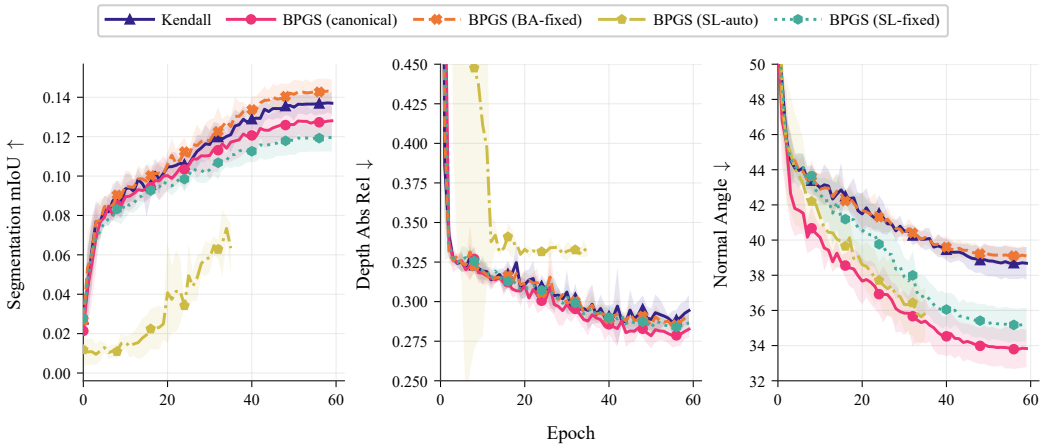

Figure 7: NYUv2 ablation validation curves by task. The stateless auto-calibrated variant fails on segmentation and remains weak on depth, while the batch-aware variants maintain stronger trajectories throughout training.

Table 9: Stop-gradient ablation on the NYUv2 50% subset (three seeds, 60 epochs). Removing the stop-gradient changes the uncertainty dynamics and lowers segmentation mIoU.

Metric	Split on	Split off
$\theta_{\max}$, epoch 0	3.082 ± 0.017	3.109 ± 0.011
$\theta_{\max}$, final	2.438 ± 0.014	3.993 ± 0.192
Segmentation mIoU	0.1983	0.1897
Depth RMSE	0.8433	0.8544
Total loss	2.1278	2.1419

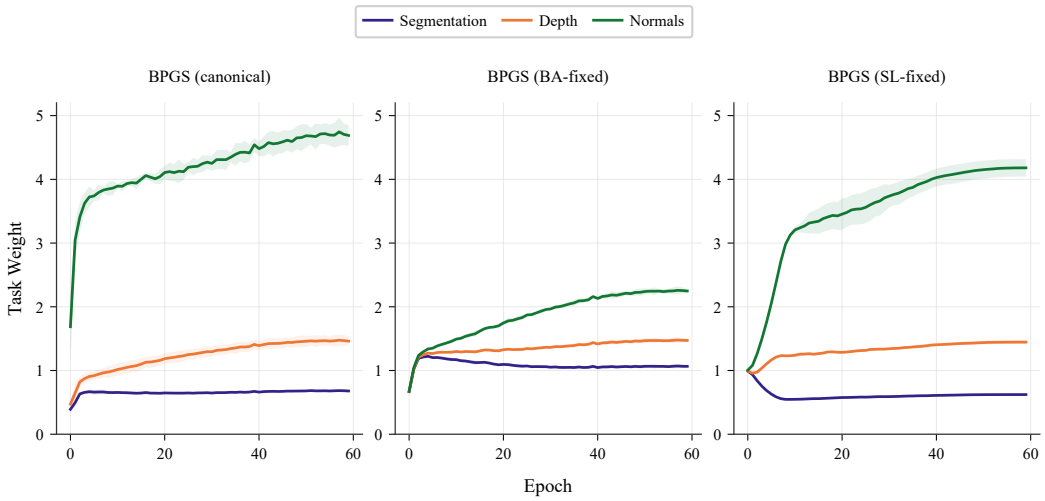

Figure 8: Task-weight dynamics for representative stable BPGS variants on the NYUv2 ablation. The batch-aware variants keep task weights in distinct but controlled regimes, whereas the stateless fixed variant shows a different long-run allocation pattern, especially for the surface-normal task.

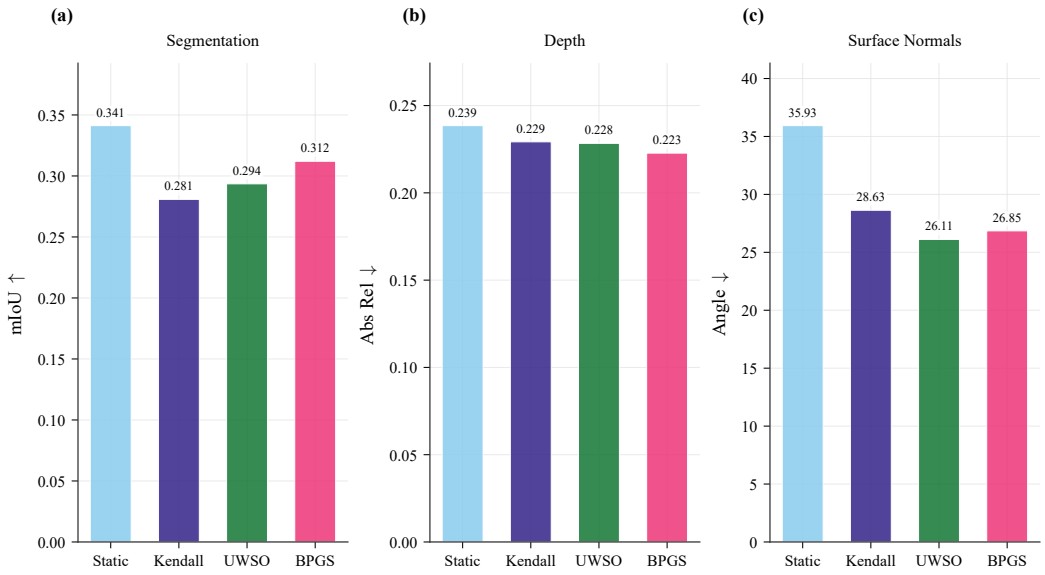

Figure 9: NYUv2 per-task performance comparison. The figure makes the trade-off pattern visually explicit: Static is strongest on segmentation, UWSO is strongest on surface normals, and BPGS is strongest on both depth metrics.

Table 10: Batch-size sensitivity of canonical BPGS on the NYUv2 50% subset (three seeds per batch size, 60 epochs). Values are seed-averaged final metrics.

Metric	bs=4	bs=8	bs=16	CV
Total loss	2.1468	2.1226	2.1507	0.58%
Depth Abs Rel	0.2411	0.2421	0.2451	0.70%
Depth RMSE	0.8643	0.8403	0.8490	1.17%
Normal angle	29.400	29.223	29.754	0.75%
Segmentation mIoU	0.1955	0.1932	0.1810	3.35%
$\theta_{\max}$, final	2.396	2.441	2.589	–

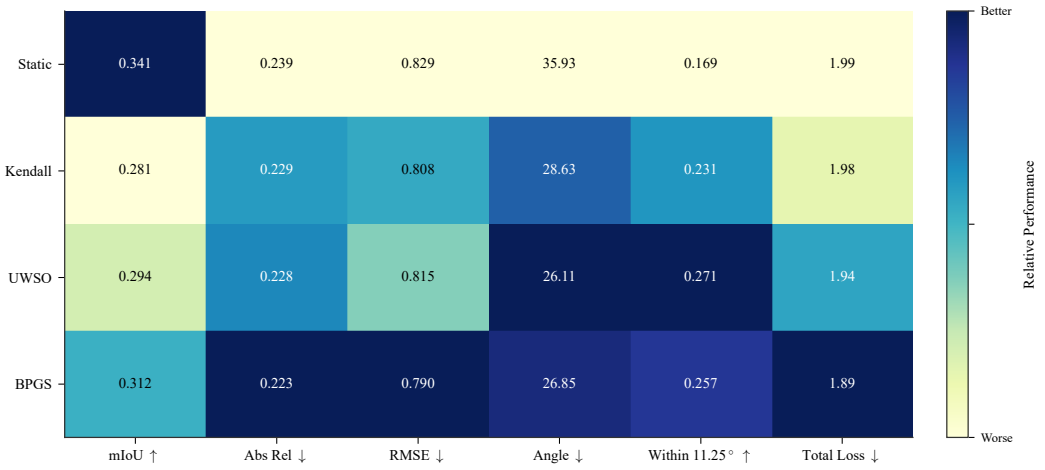

Figure 10: NYUv2 performance heatmap across reported metrics. Darker cells indicate stronger relative performance after accounting for metric direction. BPGS concentrates its gains on depth and total loss, while Static and UWSO remain stronger on some other tasks.

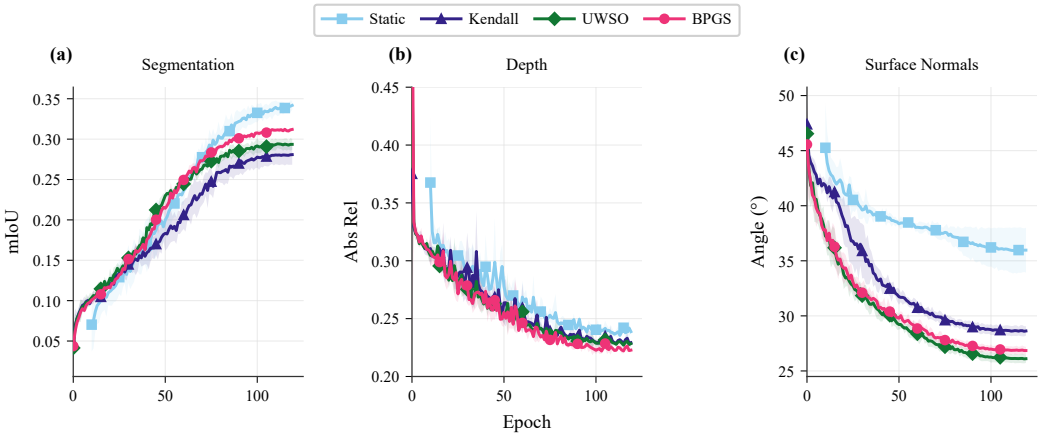

Figure 11: NYUv2 validation training curves by task. BPGS tracks the strongest depth trajectory and remains competitive on segmentation and surface normals over the full 120-epoch budget.

Table 11: First-batch calibration sensitivity on the NYUv2 50% subset. Only the loader seed changes; the model seed is fixed.

Metric	1001	2001	3001	CV
Total loss	2.1145	2.1040	2.1090	0.20%
Depth Abs Rel	0.2385	0.2392	0.2411	0.46%
Depth RMSE	0.8348	0.8302	0.8325	0.23%
Normal angle	28.9205	28.8975	29.1069	0.32%
Segmentation mIoU	0.1912	0.1866	0.1952	1.84%
$\theta_{\max}$, final	2.4451	2.4382	2.4380	0.3% spread

Table 12: Computational overhead on the NYUv2 50% subset (three seeds, 60 epochs).

Quantity	BPGS	Kendall	Difference
Per-epoch time (s)	40.979	40.916	+0.15%
Peak GPU memory (MB)	3368.36	3339.34	+0.87%
Total time, 60 epochs (s)	2461.6	2457.9	+3.7
Trainable parameters	18,869,143	18,869,140	+3

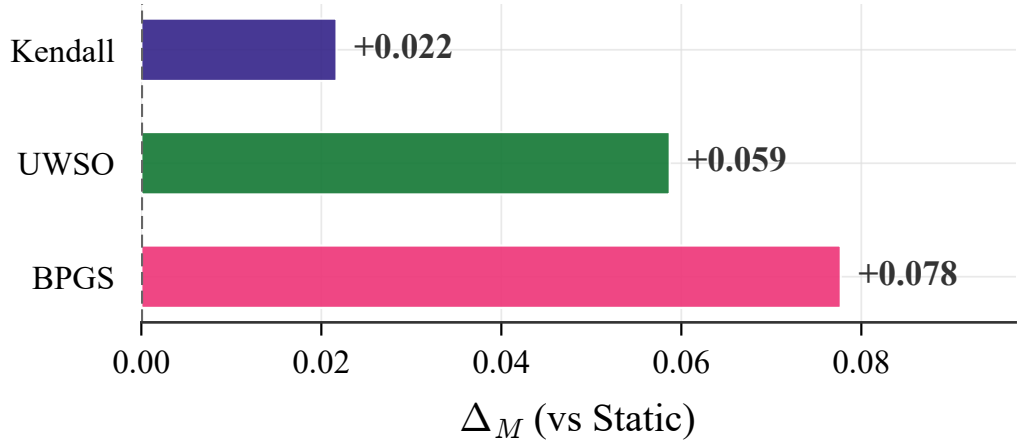

Figure 12: NYUv2 Δ_M comparison against the Static baseline. BPGS attains the strongest aggregate improvement among the adaptive methods evaluated in the main benchmark.

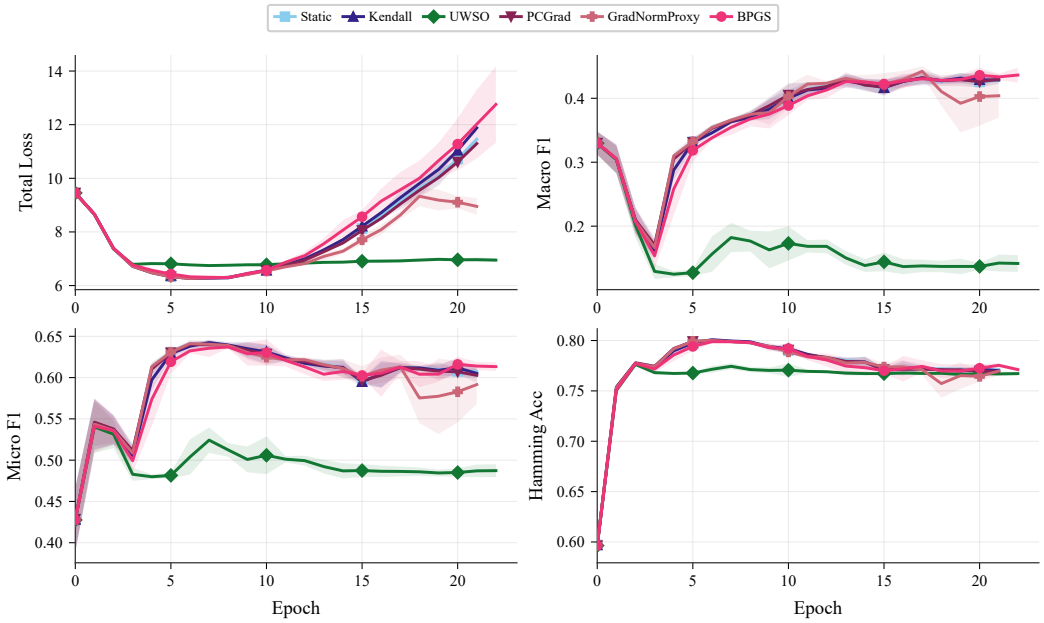

Figure 13: Yeast training curves. BPGS tracks the strongest final micro-F1 and macro-F1 among the compared methods, while UWSO remains substantially weaker throughout training.

Table 13: Closest observed approach to the bounded-chart radius across logged main-paper runs. The maximum occurs at initialization; no run reaches the boundary.

Dataset	T	max $\lvert z_i\rvert/\tau_T$	final cap
NYUv2	3	0.9327	≤ 0.84
Yeast	14	0.7017	≤ 0.69
RF1	8	0.8937	≤ 0.89

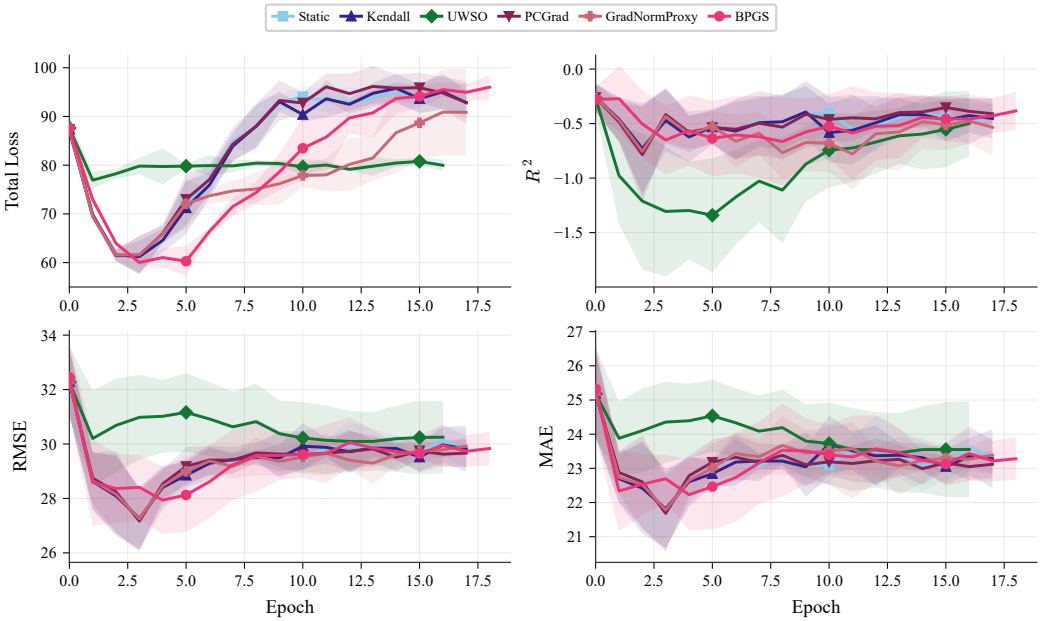

Figure 14: RF1 training curves. The methods remain closer than in the synthetic stress studies, which is consistent with the paper's claim that RF1 supports competitiveness rather than clear superiority.