Where Does Adaptation Matter?

Layer-wise Importance of Parameter-Efficient Adaptation of Vision Foundation Models to Seismic Denoising

Okan Demir Baykal[1]

Supervisor(s): Dr. Jing Sun[1], **Dr. Eric Verschuur**[2], **Dr. Tiexing Wang**[3], **Jiahua Zhao**[2,4]

[1]**EEMCS, Delft University of Technology, The Netherlands**
[2]**CEG, Delft University of Technology, The Netherlands**
[3]**R&D Department, Shearwater GeoServices, UK**
[4]**The Cyprus Institute, Cyprus**

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
June 21, 2026

Name of the student: Okan Demir Baykal
Final project course: CSE3000 Research Project
Thesis committee: Dr. Jing Sun[1], Dr. Eric Verschuur[2], Dr. Tiexing Wang[3], Jiahua Zhao[2,4], Dr. Petr Kellnhofer[1]

An electronic version of this thesis is available at http://repository.tudelft.nl/.

Abstract

Seismic trace denoising is a long-standing problem in geophysical data processing, and recent work has shown that vision foundation models pre-trained on natural images can be adapted to it parameter-efficiently rather than trained from scratch. Such adaptation is typically applied uniformly across all transformer layers, but it is not known where along the network the adaptation effort is actually needed — that is, where the representation gap between natural images and seismic data is concentrated. We investigate this question for two structurally distinct families of parameter-efficient fine-tuning (PEFT): Low-Rank Adaptation (LoRA), which injects a low-rank update into the attention projections, and Pfeiffer bottleneck adapters, which insert a residual MLP module after the feed-forward sub-layer. Using a DINOv3 ViT-S/16 backbone on active-source seismic image denoising and holding the per-layer parameter budget fixed across both mechanisms, we sweep adaptation placement across restricted subsets of the twelve transformer layers within the DINOv3 architecture and measure denoising quality at each placement. We find that adaptation is strongly concentrated in the early layers: placing modules on only the first four layers recovers most of the denoising quality of full adaptation at one third of the adapter parameters, and even a single early layer is already competitive with full adaptation, while a roughly monotonic early-to-late importance ordering holds across placements. Crucially, this profile is near-identical for the two mechanisms at matched budget, indicating that the effect is a property of *where* adaptation is applied rather than of the particular PEFT design. These results suggest that, for this task and backbone, the natural-image-to-seismic gap is primarily a low-level, input-stage shift, and that early-layer-heavy placement is an effective and economical default for PEFT-based adaptation of vision foundation models to seismic data.

1 Introduction

A seismic recording images the subsurface as a field of coherent reflection events — the layered arrivals that trace geological structure — overlaid by incoherent noise introduced during acquisition. Seismic denoising is the task of recovering the coherent signal from this corrupted record (Figure 1), and it is a long-standing problem in geophysical data processing because the quality of every downstream interpretation step depends directly on how cleanly the signal is recovered.

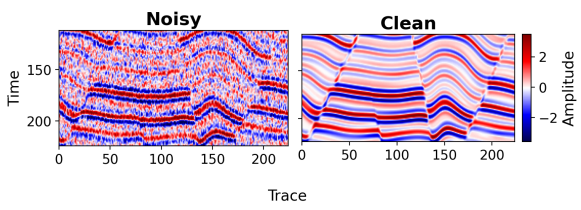

Figure 1: Example of a noisy/clean seismic image pair from the active-source dataset of Sheng et al., cropped.

Classical denoising methods based on linear and nonlinear filtering have been studied for decades but require careful per-dataset parameter tuning and degrade when signal and noise overlap in their spectral or kinematic properties [8]. Deep learning approaches have made substantial progress on this problem, but typically require task-specific models trained from scratch on curated datasets, limiting their generalisation across surveys. This has motivated foundation models pretrained on broad data and adapted to downstream tasks: Sheng et al. [8] introduced the Seismic Foundation Model (SFM), a Vision Transformer pretrained on millions of seismic patches, with strong performance across downstream tasks including denoising. However, training a domain-specific foundation model from scratch requires substantial data and compute. An alternative is to adapt existing vision foundation models pretrained on natural images: Zhao et al. [13] showed that DINOv3 [9], a general-purpose vision foundation model, can be adapted to seismic denoising at modest computational cost using Low-Rank Adaptation (LoRA) [3].

LoRA belongs to the broader family of *parameter-efficient fine-tuning* (PEFT) methods, which adapt a frozen backbone by training only a small number of added parameters. We study it alongside a structurally different design, the Pfeiffer bottleneck adapter [6], so we can separate effects intrinsic to a particular adapter from properties of the adaptation problem itself — such as where in the network adaptation is actually needed.

PEFT is conventionally applied uniformly across all transformer layers. Yet a ViT's layers are not interchangeable: early layers handle low-level structure, while later layers abstract semantics. Hence, the natural-image-to-seismic mismatch may concentrate at one end of the network rather than spread evenly across it. Whether it does, and where, has not been studied for seismic data. This research therefore investigates the following questions:

RQ1 *Which layers of DINOv3 are most important to adapt when transferring to seismic denoising?*

RQ2 *Is the resulting layer-wise importance profile consistent across structurally distinct PEFT methods and different datasets?*

RQ3 *What does that profile reveal about where the natural-image-to-seismic representation gap is concentrated?*

To answer them, we insert PEFT modules — LoRA and Pfeiffer adapters — into restricted subsets of DINOv3 layers and evaluate denoising performance on the active-source dataset of Sheng et al. [8], complemented by a post-hoc analysis of the trained per-layer Pfeiffer adapter weights to localise where adaptation accumulates, and we repeat the placement sweep on a second, synthetic dataset (ThinkOnward's *Image Impeccable* denoising challenge [10]) to test whether the profile transfers.

2 Background

2.1 Vision Transformers and feature hierarchies.

A Vision Transformer (ViT) divides an image into fixed-size patches, embeds each patch as a token, and processes the resulting sequence through a stack of transformer layers, each combining multi-head self-attention with a token-wise feed-forward network (FFN) under residual connections and layer normalisation. DINOv3 [9] is a general-purpose ViT pretrained on natural images with a self-supervised objective, yielding features that transfer well to a range of downstream tasks. As in convolutional networks, representations in a ViT are organised hierarchically: early layers respond to low-level structure such as edges and local texture, while later layers encode increasingly abstract, semantic content. This hierarchy is central to the present work, since it implies that the mismatch between natural-image and seismic data does not have to be uniform across depth — it may be concentrated at one end of the network.

2.2 Parameter-efficient fine-tuning.

Parameter-efficient fine-tuning (PEFT) adapts a pretrained model to a new task by training only a small number of additional parameters while the backbone remains frozen, avoiding the cost and overfitting risk of full fine-tuning. It originated in NLP — bottleneck adapters [2] and low-rank weight updates (LoRA) [3] were both introduced for adapting large language models. The same techniques now transfer to vision transformers [1]. The family spans several structurally distinct designs; this study uses two representatives, LoRA and the Pfeiffer bottleneck adapter [6], whose exact formulations are deferred to Section 3.3.

2.3 Layer-wise adaptation.

Adaptation is conventionally applied uniformly across all transformer layers, but recent work questions whether this is necessary. Surgical fine-tuning [4] shows that tuning only a contiguous subset of layers, while freezing the rest, can match or outperform tuning the whole network, and crucially that *which* subset is best depends on the *type* of distribution shift: input-level shifts such as image corruptions are best addressed by adapting the earliest layers, while feature- and output-level shifts favour progressively later ones. Notably, surgical fine-tuning operates in the full fine-tuning regime — it updates the entire weights of the selected layers rather than inserting small parameter-efficient modules. Within PEFT, related approaches instead allocate a fixed budget non-uniformly across layers, for example by dynamically reallocating low-rank capacity during training [12] or selecting which layers to tune for efficiency [1]. Whether the depth-dependence established for full-weight tuning also governs adapter-based transfer, and which end of the network the natural-image-to-seismic shift favours, has not been studied.

3 Methodology

3.1 Backbone and pipeline

The backbone is DINOv3 ViT-S/16 [9], consisting of $L = 12$ transformer layers with hidden dimension $d = 384$ and patch size 16. The backbone is frozen throughout and adapted by inserting a parameter-efficient fine-tuning (PEFT) module — either LoRA or a Pfeiffer adapter (Section 3.3) — into a chosen subset of its transformer layers, while a trainable convolutional decoder maps the resulting features back to a denoised patch. The frozen backbone contains roughly 21.6M parameters and receives no gradient updates; the only trainable components are the convolutional decoder ($\approx$ 1.27M parameters) and the inserted PEFT modules ($\approx$ 49.6k parameters per layer for Pfeiffer, $\approx$ 49.2k for LoRA), so that even full 12-layer adaptation trains under 0.6M backbone-side parameters. The full denoising pipeline — input preprocessing, the convolutional decoder, the training loop, and the loss function — is adopted from Zhao et al. [13], with preprocessing, the training protocol, and evaluation detailed in Sections 3.2, 3.6, and 3.7. The benefit of reusing this pipeline is that it is already validated for LoRA-based seismic denoising, so we can study layer placement on a proven setup rather than re-engineering one. We train both mechanisms under identical conditions, so any difference between them reflects the adaptation mechanism rather than confounding implementation choices. We disable the kurtosis-guided test-time adaptation step described by Zhao et al., since the question of interest is the supervised, train-time contribution of PEFT modules at different layers.

3.2 Preprocessing

The same preprocessing is applied during training and at test time. Each seismic section is first standardised per sample by subtracting its own mean and dividing by its own standard deviation. For paired data, the clean target is normalised with the statistics of its noisy input, so that the two remain on a common scale. Because DINOv3 expects three-channel RGB images, the single-channel seismic input is then replicated across three channels. Sections are processed in 224×224 patches with a stride of 112 pixels (50% overlap). At inference, the per-patch predictions are de-normalised with the stored per-sample statistics and stitched back into a full section, averaging over the overlapping regions.

3.3 PEFT mechanisms

We study two structurally distinct families of PEFT, chosen so that they differ in both *where* they intervene in a transformer layer and *how* they modify its forward pass. Because both are applied to the same transformer layers and matched in per-layer parameter budget (Section 3.4), any difference in behaviour between them is attributable to the adaptation mechanism rather than to differences in budget or placement, and the layer-wise comparisons that follow isolate *where* adaptation is applied. We describe each mechanism in turn:

Pfeiffer adapters. A Pfeiffer bottleneck adapter [6] is a small residual module inserted after the feed-forward sub-layer of a transformer layer. Given a hidden representation $h \in \mathbb{R}^d$, the adapter applies

$$h \leftarrow h + W_{\text{up}}\, \sigma(W_{\text{down}}\, h)\,,$$

where $W_{\text{down}} \in \mathbb{R}^{r\times d}$ and $W_{\text{up}} \in \mathbb{R}^{d\times r}$ are the trainable down- and up-projection matrices, $r \ll d$ is the bottleneck dimension, and σ is a GELU nonlinearity. In our implementation the adapter is attached via a forward hook on the MLP sub-module of each targeted transformer layer, so that the adapter output is added to the layer's MLP output prior to the layer-level residual connection. The bottleneck dimension is fixed at $r = 64$, yielding approximately 49,600 trainable parameters per layer. Following standard practice, the up-projection W_{up} is initialised to zero, so that each adapter begins as an identity map and the pretrained forward pass is recovered exactly at the start of training.

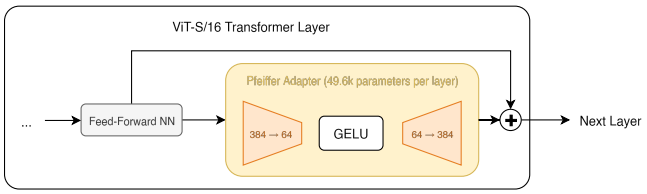

Figure 2: Pfeiffer adapter. A bottleneck adapter inserted after the feed-forward sub-layer of a transformer layer: a down-projection ($384 \to 64$), a GELU nonlinearity, and an up-projection ($64 \to 384$), added back through a residual connection (≈50k trainable parameters per layer).

LoRA. Low-Rank Adaptation [3] keeps the pretrained weights frozen and learns a low-rank update in parallel with them. For a frozen projection matrix $W_0 \in \mathbb{R}^{d\times d}$ acting on a hidden representation h, the adapted output is

$$W_0\, h \;\longrightarrow\; W_0\, h + \tfrac{\alpha}{r}\, B\, A\, h,$$

where $A \in \mathbb{R}^{r\times d}$ and $B \in \mathbb{R}^{d\times r}$ are the trainable low-rank factors, $r \ll d$ is the rank, and α is a fixed scaling factor. The factor B is initialised to zero so that the update is inactive at the start of training. We apply LoRA to the four attention projections of each layer — the query, key, value, and output projections — with rank $r = 16$, scaling $\alpha = 64$, and dropout 0.1, using the `peft` library. This yields 49,152 trainable parameters per layer, matching the Pfeiffer budget to within 0.9%.[1] Because both mechanisms are inserted into the same layers at this matched per-layer budget, the layer-subset comparisons in Section 3.4 isolate *where* adaptation is applied rather than how much capacity it has.

[1]The 448-parameter difference is exactly the Pfeiffer adapter's down- and up-projection bias vectors ($64 + 384$), which the LoRA formulation does not include.

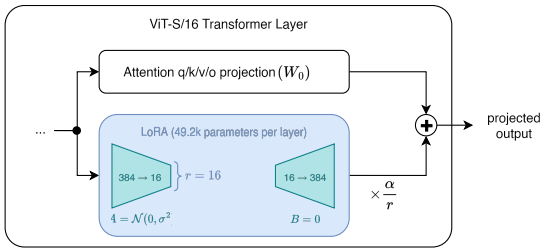

Figure 3: LoRA adaptation. A trainable low-rank update $\frac{\alpha}{r}BA$ is added in parallel to each frozen attention projection W_0 of a transformer layer; only the factors A and B are trained, while W_0 stays frozen.

3.4 Experimental conditions

The following configurations are compared to systematically evaluate layer-wise adaptation importance:

- **Decoder-only baseline.** The backbone is frozen and no adaptation modules are inserted; only the decoder is trained. This serves as the lower-bound reference for what the pretrained DINOv3 features deliver with no adaptation.
- **Full (uniform) adaptation.** Adaptation modules—LoRA [3] at rank 16 or Pfeiffer adapters—are inserted into all twelve transformer layers. This isolates the effect of the adapter mechanism at matched adapter parameter budget and uniform layer coverage, with the LoRA variant reproducing the configuration of Zhao et al. [13].
- **Four-layer segments (early, middle, late).** Adapters (evaluated separately for both LoRA and Pfeiffer) are inserted only into the first four layers (1–4), the middle four layers (5–8), or the last four layers (9–12). These conditions utilise one third of the full-coverage parameter budget and provide a coarse, layer-level view of the layer-wise importance profile.
- **Single-layer sweep.** A single adapter module (evaluated separately for both LoRA and Pfeiffer) is inserted into exactly one of the twelve transformer layers, swept across all layers $l \in \{1, \ldots, 12\}$ in independent runs. This provides a high-resolution, layer-by-layer diagnostic of where the natural-image-to-seismic representation gap is concentrated at minimal adapter parameter cost.

3.5 Post-hoc weight analysis

Beyond performance sweeps, we analyse the trained adapter weights of the full-coverage Pfeiffer model directly. We restrict this analysis to Pfeiffer by design. Its down- and up-projections are explicit, separately stored matrices, so each can be profiled on its own. LoRA's update is instead a single low-rank product $\frac{\alpha}{r}BA$ whose factors are not two separable projections, leaving no matched per-matrix quantity to compare across mechanisms. This analysis is therefore mechanism-specific corroboration of the cross-mechanism sweeps, not a claim about LoRA.

For each layer ℓ, we compute two summary statistics for both the down-projection ($W_{\text{down}}^{(\ell)}$) and up-projection ($W_{\text{up}}^{(\ell)}$) matrices independently. The first is the Frobenius norm ($\|W\|_F = \sqrt{\sum_{ij} W_{ij}^2}$), serving as a measure of the overall magnitude of the learned update. The second is the effective rank [7], an entropy-based measure of the dimensional complexity of the learned transformation. Given the singular values $\sigma_1, \ldots, \sigma_k$ of W, it is defined as

$$\text{erank}(W) = \exp\left(-\sum_i p_i \log p_i\right), \qquad p_i = \frac{\sigma_i}{\sum_j \sigma_j}, \tag{1}$$

yielding a value in $[1, r]$. A low effective rank indicates that the matrix exploits only a small subspace of its available bottleneck capacity, while a value close to the bottleneck dimension $r = 64$ indicates full capacity utilisation. These per-layer curves serve as a diagnostic readout of where the trained uniform-adapter model concentrates its learned updates, corroborating the empirical layer-subset performance results from within the Pfeiffer parameterisation.

3.6 Training protocol

All conditions share an identical optimisation setup; the only variation between runs is the set of trainable parameters defined by the condition. The backbone is frozen throughout and only the inserted PEFT module (or decoder) parameters receive gradient updates.

Optimiser and schedule. We use AdamW [5] with an initial learning rate of 6×10^{-4} and weight decay 10^{-2}. The learning rate follows a linear warm-up over the first 10 epochs, after which it decays towards zero along a cosine schedule whose horizon is fixed at the full 50-epoch budget. Training runs for at most 50 epochs but may terminate earlier under the early-stopping criterion described below; the cosine horizon remains pegged to 50 epochs irrespective of where a run stops, so that runs reaching the full budget follow an identical schedule.

Loss function. The training objective is a balanced combination of mean-squared error and MS-SSIM:

$$\mathcal{L} = \tfrac{1}{2}\,\mathcal{L}_{\text{MSE}}(\hat{x}, x_{\text{clean}}) + \tfrac{1}{2}(1 - \text{MS-SSIM}(\hat{x}, x_{\text{clean}})), \tag{2}$$

with an 11×11 Gaussian window and the standard five-scale weighting [11].

Data, validation split, and checkpoint selection. The 2,000 paired (noisy, clean) examples from the Sheng et al. [8] training split are partitioned into 1,600 training and 400 validation examples by a fixed seeded split. The split seed is held constant across every run and every condition, so that all configurations are selected and compared against an identical held-out validation set. Training uses a batch size of 24 patches per iteration. At the end of every epoch we evaluate mean MS-SSIM-C on the 400 validation examples, and the checkpoint achieving the highest validation MS-SSIM-C is retained as the model for that run; selection is therefore based on generalisation to held-out data rather than on the final-epoch state. Early stopping with a patience of 10 epochs, keyed on validation MS-SSIM-C, halts training once the validation metric has not improved for ten consecutive epochs.

Repetitions and hardware. Each experimental condition is run for five independent repetitions. Repetitions differ by an explicit per-run random seed, derived deterministically from the run index and recorded for each run, which governs adapter and decoder initialisation as well as training-time data ordering; the validation split seed is held fixed across all repetitions so that only the trainable initialisation and ordering vary. All results are reported as mean ± standard deviation across the five runs. When comparing two conditions, we apply a two-sided Welch's (unequal-variance) two-sample t-test to their five per-run mean scores, computing each run's mean over the test set first so that runs, not individual patches, are the units of comparison; we report the per-metric p-value and treat $p < 0.05$ as significant.[2] All training is performed on a single NVIDIA RTX 3080 GPU.

3.7 Evaluation

Primary test set. The test set is the test partition of the same SFM dataset of Sheng et al. [8] used for training, comprising 4,000 unlabelled noisy field recordings for which no clean reference traces are available at evaluation time. For this test set, we adopt two reference-free proxy metrics, both derived from the multi-scale structural similarity index (MS-SSIM) [11].

MS-SSIM. The primary metric is MS-SSIM$(x_{\text{in}}, \hat{x})$, the structural similarity between the noisy input and the denoised output; higher values indicate that coherent signal structure is preserved while incoherent noise is suppressed. Because it compares the output to the noisy input rather than to a clean reference, this proxy cannot by itself separate effective denoising from inaction: a do-nothing model that returns its input scores near-perfectly. In practice MS-SSIM and the full-reference MS-SSIM-C rank our configurations consistently, so it remains a useful proxy where clean labels are unavailable. We use the standard five-scale MS-SSIM [11] with an 11×11 Gaussian window, and note two implementation choices for reproducibility: the per-scale maps are rescaled from $[-1, 1]$ to $[0, 1]$ via $(s + 1)/2$ before the across-scale product, and the dynamic range L in the constants $C_1 = (0.01L)^2$, $C_2 = (0.03L)^2$ is inferred per image ($L = 2$ for the per-sample-standardised inputs, $L = 255$ for raw $[0, 255]$ data). This implementation is adopted from Zhao et al. [13] and is used identically for

[2] The reported p-values are uncorrected for multiple comparisons. The borderline results ($p \approx 0.05$) would not survive a correction such as Holm–Bonferroni and should be read with corresponding caution; the smaller p-values ($p \leq 0.01$) that underpin our main claims comfortably survive it.

the training loss (Section 3.6), the reference-free proxy, and the full-reference MS-SSIM-C below.

MS-SSIM-R. The secondary metric follows the MS-SSIM-R signal-leakage measure of Zhao et al. [13]:

$$\text{MS-SSIM-R} = \text{MS-SSIM}(x_{\text{in}},\ x_{\text{in}} - \hat{x})\,, \qquad (3)$$

which quantifies how much of the input's signal structure has leaked into the extracted noise residual; lower values are better. A high MS-SSIM together with a low MS-SSIM-R characterises a model that separates signal from noise without discarding coherent energy. Like MS-SSIM, this proxy does not penalise inaction — a do-nothing model also scores well — so neither reference-free metric can rule out inaction on its own.

Full-reference metrics. Where clean references are available — the held-out validation split and the generalisation set introduced below — we additionally report mean absolute error (MAE), peak signal-to-noise ratio (PSNR), and the full-reference MS-SSIM-C $\equiv$ MS-SSIM$(x_{\text{clean}}, \hat{x})$. Lower MAE and higher PSNR and MS-SSIM-C indicate closer agreement with the ground truth. Because these compare the output to the clean label rather than the noisy input, they cannot be satisfied by leaving the input untouched; MS-SSIM-C is the full-reference counterpart of the primary proxy and resolves the inaction ambiguity. MS-SSIM-C is also the quantity used for checkpoint selection (Section 3.6) on the held-out validation split, while the reference-free proxies are computed only on the unlabelled test partition and never enter selection.

Secondary test set. To probe whether our models behave the same way on a different dataset, we evaluate them on a generalisation set drawn from the ThinkOnward *Image Impeccable* challenge[3] — a collection of *synthetic* 3D seismic volumes ($1259 \times 300 \times 300$) supplied as paired noisy and clean versions. We use the first part of this dataset (15 volumes), extracting evenly spaced inline sections and tiling them into 224×224 patches to obtain 450 clean/noisy pairs. Each volume is rescaled to $[0, 255]$ (clipped at the 2nd and 98th percentiles); because inputs are standardised per sample (Section 3.2), this amplitude convention — different from the roughly $[-8, 8]$ range of the SFM data — is invisible to the network and affects only visualisation. This set is held out and never used for training.

4 Results

4.1 Uniform Adaptation: Full Coverage vs. Baseline

To establish the performance boundaries of adaptation, we first compare the full-coverage conditions (full LoRA and full Pfeiffer) against the decoder-only baseline. Figure 4 shows a representative section denoised by all three. Operating on the unadapted backbone, the decoder-only model already suppresses much of the incoherent noise and recovers the broad reflector structure — it is a working denoiser, not a failure case — but it does so coarsely, smoothing over the finer reflector detail and fault geometry. Both LoRA and Pfeiffer adaptation visibly sharpen these features, restoring continuous, well-resolved reflectors, and the two mechanisms yield similar outputs. Adapting the frozen backbone thus recovers fine detail that the decoder, despite its far larger parameter count, cannot recover on its own.

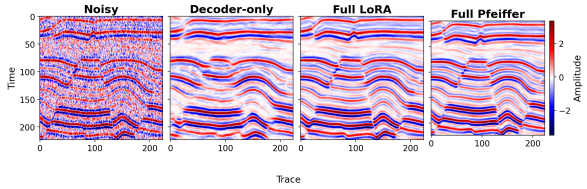

Figure 4: Denoising comparison on a single seismic section. From left to right: the noisy input, and the outputs of the decoder-only baseline, the full LoRA adaptation, and the full Pfeiffer adaptation. The decoder-only model suppresses noise but reconstructs reflectors coarsely, blurring fine detail and geometry. Adapted models recover sharper, more continuous details and perform similarly.

This qualitative picture is reinforced by the metrics (Figure 5). Both LoRA and Pfeiffer-adapted models substantially outperform the decoder-only model, raising MS-SSIM from 0.660 to 0.940 (LoRA) and 0.943 (Pfeiffer) and lowering MS-SSIM-R from 0.874 to 0.636 and 0.624 respectively. At a matched parameter budget the two adaptations perform almost identically, differing by only 0.002 in MS-SSIM and 0.012 in MS-SSIM-R, and a Welch's two-sample t-test (Section 3.6) finds no significant difference on either metric ($p = 0.06$, $p = 0.18$). With only $n = 5$ runs these tests are underpowered, however: Pfeiffer is consistently ahead on both metrics with a large effect size on MS-SSIM (Cohen's $d \approx 1.4$), so a small genuine edge cannot be ruled out, being too fine to resolve at five runs. We therefore claim *practical*, rather than statistical, equivalence: the two mechanisms are interchangeable for our purposes, so the placement sweeps that follow vary *where* adaptation is applied while holding the adapter design effectively fixed.

[3] https://thinkonward.com/app/c/challenges

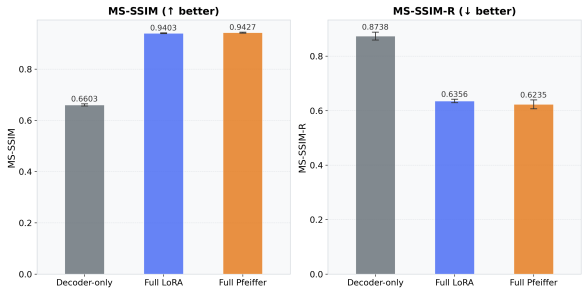

Figure 5: Uniform adaptation performance. Denoising quality (MS-SSIM, higher is better) and signal leakage (MS-SSIM-R, lower is better) of the decoder-only baseline compared to full 12-layer adaptation using LoRA and Pfeiffer adapters.

4.2 Coarse Localisation: 4-Layer Sweeps

Having established the efficacy of full adaptation, we restrict the parameter budget to a single four-layer segment to localise where adaptation is most critical. Figure 6 shows a clear, monotonic early-to-late importance ordering for both mechanisms. Placing adapters exclusively in the first four layers (the early segment) recovers approximately 99% of full-adaptation MS-SSIM (LoRA 0.934 and Pfeiffer 0.938, against 0.940 and 0.943 for full coverage) at one third of the adapter parameter budget (Table 3), and comes close to full adaptation on signal leakage as well (MS-SSIM-R $\approx$ 0.62–0.64 for both): the gap from full coverage is negligible for Pfeiffer (Cohen's $d \approx 0.1$) and, for LoRA, a small but consistent difference that five runs cannot resolve. The middle segment yields lower MS-SSIM and higher leakage, and the late segment lower still ($\approx$ 0.73 MS-SSIM), though all three segments remain well above the decoder-only baseline (0.66). The ordering is consistent across both LoRA and Pfeiffer and across both metrics, indicating that the domain shift is localised to the early layers rather than distributed uniformly across depth.

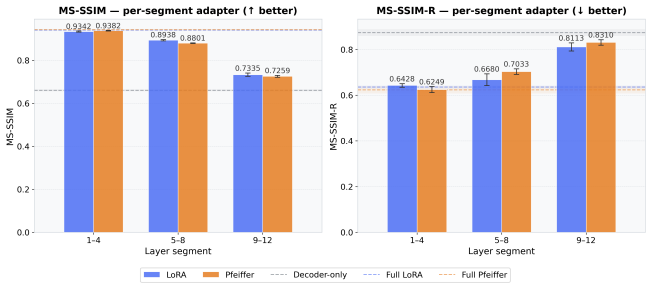

Figure 6: Coarse localisation via 4-layer sweeps. Denoising quality and signal leakage when adapters are restricted to the early (1–4), middle (5–8), and late (9–12) layers. Early-layer adaptation clearly outperforms middle and late placements.

4.3 High-Resolution Localisation: Single-Layer Sweeps

The coarse localisation is refined by the single-layer sweeps, isolating the contribution of individual layers. As shown in Figure 7, adaptation efficacy is concentrated in the earliest layers for both mechanisms. A single LoRA adapter placed in layer 1 already reaches full 12-layer LoRA adaptation on MS-SSIM — in fact slightly but reliably exceeding it (0.943 versus 0.940; $p = 0.035$). A single Pfeiffer adapter in layer 1 reaches MS-SSIM $\approx$ 0.92 against $\approx$ 0.94 for full Pfeiffer and matches full Pfeiffer on signal leakage (MS-SSIM-R $\approx$ 0.63). Aside from a departure from monotonicity among the earliest layers (a second-layer dip examined in Section 5), efficacy decreases with depth through the middle layers and, for the latest layers, approaches or falls to the decoder-only baseline.

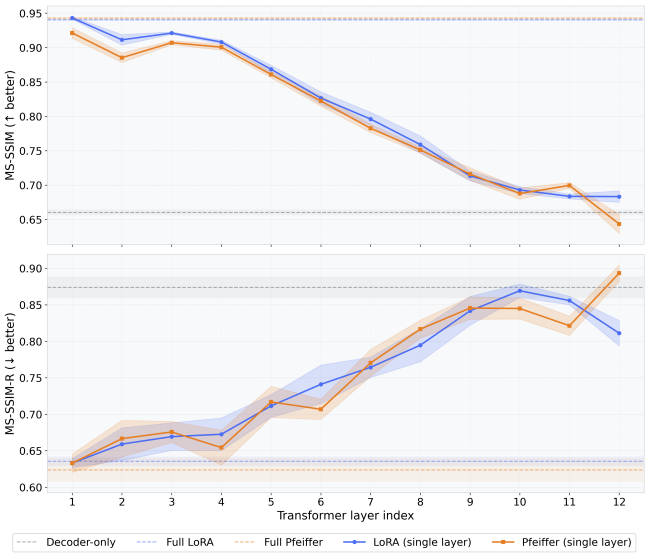

Figure 7: High-resolution single-layer sweeps. MS-SSIM and MS-SSIM-R scores for models utilising exactly one layer with adaptation, swept across all 12 layers. Performance peaks in the earliest layers and decays with depth.

4.4 Generalisation to a Second Dataset

The sweeps so far use a single dataset, leaving open whether the early-layer importance profile reflects the denoising task or merely an idiosyncrasy of the SFM data. We therefore repeat the single-layer sweeps on the independent, synthetic Image Impeccable set (Section 3.7). The full-reference metrics for the uniform-coverage conditions, available wherever clean labels exist, confirm the headline effects on this second dataset (Table 1): full adaptation again improves sharply over the decoder-only baseline on every metric and on both datasets—on Image Impeccable, MS-SSIM-C rises from 0.77 to 0.91, PSNR from 15.9 to 20.4 dB, and MAE falls from 28.6 to 17.0—and the performance parity between the two mechanisms holds on this second dataset as well,

agreeing to within 0.001 MS-SSIM-C under full coverage on both labelled sets.

Table 1: Full-reference denoising metrics against the clean label (mean over five runs) for the decoder-only baseline and full LoRA/Pfeiffer adaptation, on the two datasets with clean references. MAE and PSNR are amplitude-scale dependent and comparable only within a dataset block; MS-SSIM-C is scale-invariant.

	MAE↓	PSNR (dB)↑	MS-SSIM-C↑
SFM held-out validation			
Decoder-only	0.546	21.30	0.750
Full LoRA	0.202	30.59	0.938
Full Pfeiffer	0.199	30.72	0.939
Image Impeccable (synthetic)			
Decoder-only	28.59	15.91	0.770
Full LoRA	17.01	20.44	0.910
Full Pfeiffer	17.04	20.43	0.910

Beyond these uniform-coverage anchors, the layer-wise importance profile itself reproduces (Figure 8). Overlaying the single-layer sweeps of both datasets on the same two reference-free MS-SSIM metrics—so that the SFM test-set curves of Section 4.3 and the Image Impeccable curves are directly comparable—shows efficacy concentrated in the earliest layers and decaying with depth for both datasets and both mechanisms; the single layer-1 LoRA adapter again slightly exceeds full LoRA, and even the minor second-layer dip recurs. That the four sweep curves track one another so closely indicates that the layer-wise importance profile is a property of the natural-image-to-seismic transfer itself, robust to both the choice of adapter mechanism and the evaluation dataset.

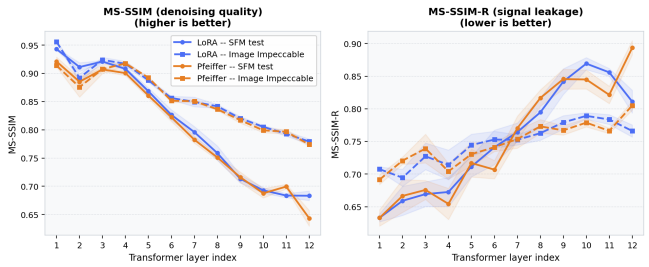

Figure 8: Cross-dataset generalisation of the single-layer sweeps. Denoising quality (MS-SSIM, left) and signal leakage (MS-SSIM-R, right) for a single adapter in each of the 12 layers; mechanism is encoded by colour (LoRA / Pfeiffer) and dataset by linestyle (SFM test set, solid; Image Impeccable, dashed), with bands of ±1 standard deviation over five runs. The profile peaks in the earliest layers and decays with depth — including the shared second-layer dip in denoising quality (left) — regardless of mechanism or dataset.

4.5 Mechanism-Specific Corroboration: Pfeiffer Weight Analysis

Finally, we analyse the trained weights of the full-coverage Pfeiffer model to ask whether the network intrinsically favours early-layer updates when given capacity at every layer. As noted in Section 3.5, this reading is available only for Pfeiffer; it corroborates the cross-mechanism sweeps from within one mechanism rather than forming an independent test.

Figure 9 plots the Frobenius norm and effective rank of the down- and up-projections (W_{down}, W_{up}) across the 12 layers, averaged over five runs. Two patterns stand out. First, both projections are front-loaded — the update magnitude (Frobenius norm) is concentrated in the early layers and falls away through the deeper ones — though the peak location differs: W_{up} is largest at the very first layer, whereas W_{down} peaks across the first few (layers 2–4) before declining. This broadly tracks the performance sweeps, but less cleanly than the placement results themselves. Second, the two projections differ sharply in effective rank: W_{down} stays near-full ($\approx$ 57–60) at every layer, whereas W_{up} is consistently lower and noisier ($\approx$ 53 at the first layer, falling and fluctuating through the deeper layers). Unlike the smooth performance decay, this rank profile is only loosely monotonic.

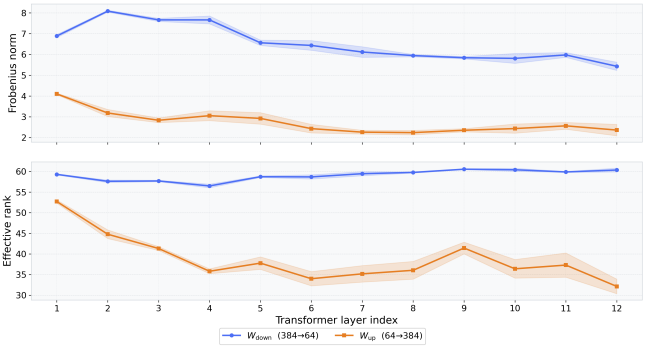

Figure 9: Layer-wise Frobenius norms and effective ranks of the full-coverage Pfeiffer model, averaged over five runs. The magnitude of the learned updates (top) is concentrated in the early layers for both projections. The effective rank (bottom) shows that the up-projection uses markedly less of its dimensional capacity than the down-projection, most so in the deeper layers, corroborating the early-layer importance observed in the empirical sweeps.

5 Discussion

Across both adapter mechanisms, denoising performance concentrates overwhelmingly in the earliest transformer layers: a single layer-1 adapter recovers most of the quality of full twelve-layer adaptation — matching or slightly exceeding it for LoRA, and within a small margin for Pfeiffer — at one twelfth of the adapter parameter budget. That two architecturally distinct PEFT mechanisms yield the same layer-wise importance profile indicates the effect is a property of *where* adaptation is applied rather than of how the adapter is constructed. This strong concentration of adaptation importance in the earliest layers suggests that the representation gap between natural images (the DINOv3 pretraining domain) and seismic recordings is fundamentally a low-

level, input-stage shift. While natural-image tasks rely heavily on deep, abstract semantic representations (e.g., object identity), active-source seismic denoising depends on structural coherence, local wavelets, and phase alignments. Consequently, recalibrating the network's foundational feature extractors at the input stage is largely sufficient to bridge the domain gap, rendering deep-layer adaptation mostly redundant for this task.

A second line of evidence comes from the decoder-only baseline. Adding even a single early-layer adapter — under 4% more trainable parameters on top of the shared 1.27M-parameter decoder — raises full-reference MS-SSIM-C from 0.750 to above 0.93 (Table 1), a gain the decoder cannot reach on its own at any point in our experiments. Because the decoder is identical across conditions, the gain is attributable to the early-layer adaptation rather than to decoder capacity; we leave open only the finer question of why, since we did not examine the intermediate representations.

A third, mechanism-internal line of evidence comes from the trained Pfeiffer weights (Figure 9). The large magnitude gap between W_{down} and W_{up} should not be over-read: it is largely an artefact of standard adapter initialisation, which sets W_{up} near zero so the module begins training as an identity map. The effective-rank asymmetry, however, is a genuine structural signal — W_{down} holds a near-full rank at every layer while W_{up} stays well below it and declines with depth. This suggests the adapter continually reads a high-dimensional summary of the residual stream but writes back an intrinsically lower-dimensional correction, increasingly so in the deeper layers. Together with the early-concentrated update magnitudes, the weight geometry echoes the placement sweeps from within a single mechanism: the network spends both its magnitude and its dimensional capacity where it matters most, in the earliest layers.

While the macro-trend heavily favours early-layer adaptation, the high-resolution sweeps reveal more nuanced behaviour. Within the earliest layers the decay is not smooth but exhibits a distinct, non-monotonic perturbation whose shape differs slightly between the two mechanisms. For Pfeiffer, the second layer underperforms the highly performant first, third, and fourth layers; for LoRA, the first and third layers are the strongest, with both the second and fourth layers dipping below them. This dip is statistically robust (Appendix B), present across adaptation mechanisms, and recurs on the Image Impeccable set (Figure 8), so it is a genuine departure from monotonicity rather than run-to-run noise. We do not attempt to explain its origin here.

Finally, while LoRA and Pfeiffer adapters exhibit closely matched performance profiles across all placements, small discrepancies remain, plausibly reflecting their distinct architectural interventions — LoRA modifies the attention projections in parallel, whereas Pfeiffer acts on the feed-forward output sequentially. These are second-order perturbations on the shared early-layer profile rather than departures from it.

6 Conclusions and Future Work

This study investigated the layer-wise importance of parameter-efficient fine-tuning when transferring the DINOv3 vision foundation model to active-source seismic denoising, comparing two structurally distinct mechanisms: LoRA and Pfeiffer adapters. We found that the representation gap between natural images and seismic data is strongly concentrated in the earliest stages of the network. A single adapter of either type placed in the initial layers recovers most of the denoising performance of a fully adapted model at a fraction of the adapter parameter cost. Critically, both mechanisms trace the same performance profile across every configuration we tested, so what matters is *where* adaptation is applied, not which adapter is applied. For adapting the DINOv3 backbone to seismic data, the efficient approach is therefore to adapt only the earliest layers and leave the deeper ones unmodified, rather than spreading adapters uniformly across all layers as is typically done.

Several directions would strengthen and extend these findings. The most immediate is to test whether the early-layer profile holds for other backbones, which would establish whether the profile reflects DINOv3 specifically or natural-image pretraining more broadly, and whether finer features such as the second-layer dip survive a change of backbone. The sweeps could also be extended to the other downstream tasks these models serve — regression tasks such as interpolation or reflectivity-model inversion, and segmentation tasks such as facies classification — to test whether the same placement profile governs adaptation beyond denoising. The mechanism axis can likewise be broadened to further PEFT families, for example prefix-tuning or $(IA)^3$, to test whether the placement profile is invariant beyond the two adapter types studied here. A complementary direction would shift the emphasis from efficiency to performance: this study minimised and equalised the per-layer budget, but future work could instead spend a fixed budget where it matters most, allocating capacity asymmetrically by concentrating rank or width in the first few layers and omitting it from the rest, or trading decoder capacity for adapter capacity by pairing a lighter decoder with richer early-layer adaptation. Finally, the profile's transfer across different seismic data — particularly passive recordings, whose noise differs substantially from active-source data — remains open.

7 Responsible Research

The primary responsibility concern for a study of this kind is reproducibility. All experimental conditions share an identical backbone (DINOv3 ViT-S/16), optimiser, learning-rate schedule, batch size, and epoch budget, so that the only variable across runs is the set of layers receiving adapters. The train/validation split is determined by a fixed seed held constant across every run and condition, so that all configurations are selected against an identical held-out set; each repetition uses an explicit per-run seed derived from the

run index, and these seeds are recorded alongside the reported metrics so that individual runs can be reproduced from the saved artifacts. Checkpoint selection retains the epoch of highest validation MS-SSIM-C rather than the final-epoch state, and all reported figures are averaged over five independent repetitions. The pipeline builds on the codebase of Zhao et al. [13], whose preprint we cite directly. The full codebase — training scripts, adapter implementations, and evaluation pipeline — is available at https://github.com/baykalokandemir/layerwise-peft-seismic; access is provided to the thesis committee, and the repository will be released publicly following the defence.

Because clean reference traces are unavailable for the test partition, denoising quality on the test set is measured with reference-free proxy metrics rather than a ground-truth MS-SSIM($x_{\text{clean}}, \hat{x}$). This is a genuine limitation: as detailed in Section 3.7, neither reference-free proxy can by itself separate effective denoising from inaction, so we base the conclusions on the full-reference metrics (MS-SSIM-C, PSNR, MAE) when available, which rule this out. Relatedly, our central conclusion, while reproduced across two adaptation mechanisms and two datasets, is still demonstrated within a deliberately narrow scope — one task and one backbone family. The agreement between two structurally distinct PEFT mechanisms strengthens the case that the early-layer profile is a property of the adaptation problem rather than of any one adapter, but generalising to PEFT as a whole remains an inference from two representatives rather than a proof, and we frame the early-layer finding accordingly.

Finally, the approach is favourable on computational grounds, training at most a few hundred thousand parameters on a single consumer GPU rather than fully fine-tuning the backbone or pretraining a seismic foundation model from scratch.

A Appendix: Use of Generative AI Tools

In accordance with the TU Delft guidelines on the use of generative AI in end projects, we disclose here how such tools were used during this research. Generative AI was used for proofreading, literature-search support, code scaffolding and automation, and formatting. All research design, experimental work, results, analysis, and conclusions are our own. We retain full responsibility for the accuracy, originality, and integrity of all content. Where AI was used to summarize literature, it was prompted to provide sources, which were then verified against the original papers.

Models used. Claude Opus 4.7, Claude Opus 4.8, and Claude Code.

The examples below are representative of the type of assistance requested and are not an exhaustive list of prompts used.

Writing support.

- "Provide alternative formulations of this sentence: …"
- "How appropriate is the strength of the language used in this paragraph?"
- "Check for spelling inconsistencies in this file: …"
- "Locate instances where we made the following claim: …"

Literature support.

- "What are the main alternative PEFT designs other than Low-Rank Adaptation?"
- "What were the five downstream tasks tested in the SFM paper?"
- AI was prompted to cite sources for factual claims; all such sources were verified against the original papers.

Programming and automation support.

- "Decompose the functionality in this Jupyter notebook into smaller Python files according to this structure: …"
- "Create a script to queue the training loops for models in these configurations: …"
- "Edit this testing script so that the `.csv` schema looks like this: …"
- Generating regular expressions to locate content within the manuscript, small inline scripts for file manipulation, and populating fields involving complex directory paths.
- "Compare the numbers in the current version of this table to our results in `x.csv`."

Visualization support.

- "Using our data format for `x.csv` and `y.csv`, create a script to plot metrics A and B as a line chart. Use variables for the titles, labels, colors, and data directories for us to populate."

Formatting support.

- "Make this table span across two columns instead of one."
- "Turn the results from our precomputed CSV file into a LaTeX table."

B Statistical Significance Tests

Table 2 reports the full set of Welch's two-sample t-tests underlying the significance claims in Sections 4 and 5. Each test compares two conditions on their five per-run mean scores, as described in Section 3.6.

Table 2: Welch's two-sample t-tests on the five per-run mean scores (Δ = first − second condition; L = LoRA, Pf = Pfeiffer). d is Cohen's d (pooled SD); * marks $p < 0.05$. p-values are uncorrected for multiple comparisons (see Section 3.6).

	MS-SSIM			MS-SSIM-R		
Comparison	Δ	p	d	Δ	p	d
Mechanism equivalence (§4.1)						
Pf vs. L, full	+0.0024	0.061	+1.38	−0.0120	0.179	−0.98
Subset vs. full (§4.2–4.3)						
Early vs. full, L	−0.0061	0.002*	−2.95	+0.0072	0.153	+1.00
Early vs. full, Pf	−0.0044	0.003*	−2.78	+0.0014	0.887	+0.09
Layer 1 vs. full, L	+0.0027	0.035*	+1.60	−0.0025	0.536	−0.41
Layer 1 vs. full, Pf	−0.0215	0.003*	−3.85	+0.0094	0.335	+0.65
Second-layer dip (§5)						
Layer 2 vs. 1, L	−0.0318	$< 0.001^*$	−5.73	+0.0260	0.058	+1.59
Layer 2 vs. 3, L	−0.0098	0.044*	−1.76	−0.0102	0.457	−0.49
Layer 2 vs. 1, Pf	−0.0360	$< 0.001^*$	−4.89	+0.0336	0.038*	+1.69
Layer 2 vs. 3, Pf	−0.0218	0.001*	−4.05	−0.0092	0.504	−0.45

C Parameter Counts Across Configurations

Table 3 reports the trainable parameter counts for each coverage level and mechanism.

Table 3: Trainable parameter counts across configurations. *Adapter* counts only the inserted PEFT parameters; *Total tr.* adds the trained decoder (1.27 M); the frozen backbone (21.6 M) is shared by all rows and excluded. Single-layer places one adapter (one layer), coarse four (a four-layer segment), and full all twelve.

		Trainable parameters	
Method	Coverage	Adapter	Total trainable
Decoder-only		0	1.27 M
LoRA	Single-layer	49.2 K	1.32 M
	Coarse	196.6 K	1.47 M
	Full	589.8 K	1.86 M
Pfeiffer	Single-layer	49.6 K	1.32 M
	Coarse	198.4 K	1.47 M
	Full	595.2 K	1.87 M

D Per-Configuration Results

Table 4 reports the full set of denoising metrics for every trained configuration, averaged over the five repetitions.

References

[1] Alessio Devoto, Federico Alvetreti, Jary Pomponi, Paolo Di Lorenzo, Pasquale Minervini, and Simone Scardapane. Adaptive layer selection for efficient vision transformer fine-tuning, 2024.

[2] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for NLP. In *International Conference on Machine Learning (ICML)*, pages 2790–2799, 2019.

[3] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations (ICLR)*, 2022.

[4] Yoonho Lee, Annie S. Chen, Fahim Tajwar, Ananya Kumar, Huaxiu Yao, Percy Liang, and Chelsea Finn. Surgical fine-tuning improves adaptation to distribution shifts. In *International Conference on Learning Representations (ICLR)*, 2023.

[5] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*, 2019.

[6] Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. AdapterFusion: Non-destructive task composition for transfer learning. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 487–503, 2021.

[7] Olivier Roy and Martin Vetterli. The effective rank: A measure of effective dimensionality. In *15th European Signal Processing Conference (EUSIPCO)*, pages 606–610, 2007.

[8] Hanlin Sheng, Xinming Wu, Xu Si, Jintao Li, Sibio Zhang, and Xudong Duan. Seismic foundation model: A next generation deep-learning model in geophysics. *Geophysics*, 90(2):IM59–IM79, 2025. doi: 10.1190/geo2024-0262.1.

[9] Oriane Siméoni, Huy V. Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, et al. DINOv3, 2025.

[10] ThinkOnward. Image impeccable denoising challenge dataset. https://thinkonward.com/app/c/challenges/image-impeccable, 2023. Accessed: 2026-06-19.

[11] Zhou Wang, Eero P. Simoncelli, and Alan C. Bovik. Multiscale structural similarity for image quality assessment. In *The Thirty-Seventh Asilomar Conference on Signals, Systems & Computers*, pages 1398–1402, 2003.

[12] Qingru Zhang, Minshuo Chen, Alexander Bukharin, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. AdaLoRA: Adaptive budget allocation for parameter-efficient fine-tuning. In *International Conference on Learning Representations (ICLR)*, 2023.

[13] Jiahua Zhao, Umair bin Waheed, Jing Sun, Yang Cui, Nikos Savva, and Eric Verschuur. Parameter-efficient adaptation of pre-trained vision foundation models for active and passive seismic data denoising, 2026. URL https://arxiv.org/abs/2605.10953.

Table 4: Per-configuration denoising results, mean over five repetitions. *SFM test* uses the reference-free proxies (no clean labels); *SFM validation* and *Image Impeccable* additionally report full-reference metrics against the clean label. ↑/↓ indicate whether higher/lower is better. All SSIM columns are multi-scale: SSIM = reference-free MS-SSIM(input, output), SSIM-R = signal-leakage MS-SSIM-R, SSIM-C = full-reference MS-SSIM(clean, output). PSNR/MAE scales differ between datasets (per-sample standardised SFM vs. $[0, 255]$ Image Impeccable) and are not cross-dataset comparable.

	SFM test		SFM validation					Image Impeccable				
Placement	SSIM↑	SSIM-R↓	PSNR↑	MAE↓	SSIM↑	SSIM-R↓	SSIM-C↑	PSNR↑	MAE↓	SSIM↑	SSIM-R↓	SSIM-C↑
Decoder-only baseline												
(no adapters)	0.660	0.874	21.30	0.55	0.678	0.857	0.750	15.91	28.59	0.779	0.788	0.770
Pfeiffer												
Single-layer sweeps												
Layer 1	0.921	0.633	29.84	0.22	0.801	0.797	0.932	19.40	19.65	0.914	0.692	0.877
Layer 2	0.885	0.667	29.11	0.24	0.799	0.799	0.928	17.51	24.38	0.876	0.721	0.842
Layer 3	0.907	0.676	28.78	0.24	0.795	0.803	0.926	18.68	20.87	0.908	0.739	0.871
Layer 4	0.901	0.654	28.11	0.26	0.790	0.804	0.920	18.92	20.04	0.918	0.704	0.884
Layer 5	0.861	0.717	25.98	0.32	0.768	0.812	0.888	18.51	20.46	0.893	0.731	0.873
Layer 6	0.822	0.707	24.32	0.39	0.747	0.825	0.854	17.37	23.83	0.852	0.741	0.839
Layer 7	0.783	0.770	23.53	0.43	0.734	0.830	0.834	17.48	23.15	0.851	0.754	0.839
Layer 8	0.751	0.817	22.89	0.46	0.722	0.837	0.815	17.17	24.09	0.836	0.773	0.826
Layer 9	0.716	0.845	22.28	0.49	0.708	0.841	0.794	16.63	25.82	0.817	0.767	0.809
Layer 10	0.688	0.845	21.83	0.51	0.697	0.847	0.777	16.18	27.40	0.799	0.779	0.791
Layer 11	0.699	0.821	21.57	0.53	0.689	0.848	0.765	16.14	27.66	0.797	0.766	0.788
Layer 12	0.643	0.893	21.25	0.55	0.677	0.858	0.748	16.00	28.48	0.774	0.805	0.765
Coarse sweeps												
Layers 1–4	0.938	0.625	30.16	0.21	0.803	0.797	0.935	20.32	17.21	0.945	0.709	0.905
Layers 5–8	0.880	0.703	26.58	0.30	0.774	0.809	0.897	19.13	18.99	0.910	0.698	0.889
Layers 9–12	0.726	0.831	22.39	0.48	0.712	0.838	0.799	16.76	25.44	0.820	0.769	0.811
Full adaptation												
Layers 1–12	0.943	0.624	30.72	0.20	0.805	0.792	0.939	20.43	17.04	0.952	0.690	0.910
LoRA												
Single-layer sweeps												
Layer 1	0.943	0.633	30.49	0.21	0.807	0.796	0.935	20.96	15.99	0.956	0.708	0.915
Layer 2	0.911	0.659	29.68	0.22	0.802	0.798	0.932	18.33	23.10	0.892	0.694	0.859
Layer 3	0.921	0.669	29.16	0.23	0.799	0.799	0.929	19.23	19.47	0.924	0.727	0.884
Layer 4	0.908	0.673	28.59	0.25	0.795	0.801	0.925	18.97	20.03	0.917	0.714	0.883
Layer 5	0.868	0.711	26.64	0.30	0.776	0.809	0.900	18.29	21.30	0.888	0.744	0.865
Layer 6	0.827	0.741	25.14	0.36	0.757	0.819	0.872	17.67	23.16	0.856	0.753	0.841
Layer 7	0.796	0.764	24.17	0.40	0.744	0.827	0.851	17.48	23.52	0.850	0.752	0.838
Layer 8	0.759	0.795	23.26	0.44	0.728	0.833	0.826	17.18	24.11	0.842	0.763	0.830
Layer 9	0.713	0.842	22.64	0.47	0.715	0.840	0.805	16.86	25.15	0.820	0.779	0.811
Layer 10	0.693	0.869	22.06	0.50	0.703	0.843	0.785	16.45	26.35	0.805	0.789	0.796
Layer 11	0.684	0.856	21.69	0.52	0.692	0.850	0.770	16.14	27.58	0.793	0.784	0.785
Layer 12	0.683	0.811	21.49	0.53	0.686	0.849	0.761	15.87	28.88	0.779	0.766	0.772
Coarse sweeps												
Layers 1–4	0.934	0.643	30.12	0.21	0.803	0.795	0.936	20.41	17.02	0.948	0.706	0.907
Layers 5–8	0.894	0.668	26.93	0.29	0.778	0.807	0.903	18.94	19.63	0.908	0.716	0.883
Layers 9–12	0.734	0.811	22.69	0.47	0.717	0.836	0.808	16.89	24.99	0.827	0.761	0.818
Full adaptation												
Layers 1–12	0.940	0.636	30.59	0.20	0.804	0.795	0.938	20.44	17.01	0.951	0.698	0.910