Distributed training of 3D seismic foundation models at scale

*Manoj Alwani[†**], Haotian An[†**], Andy Lapastora[*], Prashanth Ramaswamy[*], Bingchen Liu[*], Ben Lasscock[**], Alejandro Valenciano[**], Altay Sansal[**], Sathiya Namasivayam[**], Rohit Thekkanal[*], Jared Kramer[*], Arun Ramanathan**

† **Equal Contribution**

* ***Amazon Web Services***

** ***TGS***

Summary

Developing Seismic Foundation Models (SFMs) for 3D volumetric data requires addressing challenges in data management, computational efficiency, and the need for expanded spatial context. This paper presents a systematic approach to scaling 3D SFM training based on the Vision Transformer-Masked AutoEncoder (ViT-MAE) (He et al., 2021) to 16 nodes (128 GPUs). We show that our approach achieves near-linear scaling from 1 to 16 nodes. We demonstrate that, across multiple nodes, streaming data directly from object storage to each node in the distributed training cluster achieved 5.7× cluster-wide throughput and identified the DeepSpeed ZeRO Stage2 (Rajbhandari et al., 2020) training framework as optimal for our memory profile (127% faster than ZeRO Stage3). Through context parallelism using ring attention, we enable processing of 1536×1536×2048 voxel seismic volumes, a 4.5x volumetric expansion that was previously infeasible due to memory constraints, allowing the model to see a larger geological context critical for interpretation tasks.

1. Introduction

Seismic Foundation Models (Sansal et al., 2025) enable pre-trained representations that are adaptable to diverse geological interpretation tasks, but developing effective SFMs for 3D volumetric data presents three fundamental challenges:

Data Management at Scale. Seismic surveys generate massive 3D volumes—often billions of voxels—in specialized formats. Unlike standard image datasets, these volumes require careful data pipeline design to prevent I/O bottlenecks from undermining GPU utilization during distributed training.

Computational Efficiency. Training foundation models on 3D volumetric data is computationally intensive, with cloud infrastructure costs scaling rapidly. Achieving cost-effective scaling requires systematic optimization of distributed training strategies, batch processing, and communication patterns.

Spatial Context Expansion. Geological interpretation requires multi-scale analysis: fine-grained features (fractures, thin-bed reflectors) provide local detail, while regional structures (fault systems, depositional patterns, salt bodies) reveal broader context. However, the quadratic memory scaling of self-attention in Vision Transformers (Dosovitskiy et al., 2020) constrains processable volume sizes, limiting the model's ability to integrate information across scales.

While context parallelism (Ratner et al., 2023) has been extensively explored in language models, it's use in Vision Transformers (ViT) remains limited. Most ViT research addresses bottlenecks through data (Li et.al. 2020), tensor (Narayanan et al., 2021) and pipeline parallelism (Guan et al., 2024), as sequence lengths are modest in 2D image processing (for ImageNet (Deng et al., 2009), the largest computer vision benchmark, 3×256×256 images with 16×16 patches yield only 768 tokens). For seismic foundation models, the context window size becomes the primary limitation rather than model parameters or batch size, making context parallelism a particularly suitable approach for expanding analytical capacity beyond individual GPU memory constraints.

This paper builds upon the pre-trained 3D ViT-MAE Seismic Foundation Model (SFM) developed by Sansal et al. (2025), a 1.8-billion-parameter model originally trained on a single node. We demonstrate systematic scaling from single node to multi-node configurations (up to 128 GPUs across 16 nodes) and extend the context length through context parallelism. To the best of our knowledge, this is the first applied study to implement context parallelism within the ViT-MAE training framework for volumetric seismic data. Our contributions include: (1) empirical comparison of data streaming versus shared file systems at scale, (2) systematic evaluation of distributed training frameworks under realistic memory constraints, (3) identification of CPU-GPU communication bottlenecks, (4) demonstration of near-linear scaling with increasing node count, and (5) successful adaptation of ring attention to MAE pre-training with novel solutions for stochastic masking and decoder sequence management in 3D volumetric contexts.

2. Methodology

2.1 Data pipeline Design

The training corpus of SFM consists of 3D seismic reflection volumes stored in MDIO format (Sansal et al., 2023), a cloud-native container built on Zarr arrays (Miles et al. 2023). Individual volumes contain up to billions of voxels representing post-stack seismic data.

We evaluated two data delivery strategies: (1) Shared file system using FSx for Lustre (1000 MBps/TiB total provisioned throughput) and (2) Streaming data directly from object storage (S3) using MDIO's multi-threaded I/O. The critical distinction lies in throughput scaling: Streaming from object storage directly establishes independent connections per node (4-5 GB/s each), allowing total bandwidth to grow linearly with cluster size, while shared file systems impose a throughput ceiling that becomes a bottleneck as clusters expand.

2.2. Incremental Scaling Strategy

Training large foundation models on multi-node GPU clusters incurs substantial computational costs. Committing to large-scale experiments without first validating performance characteristics at smaller scales risks significant resource expenditure on suboptimal configurations (Allen Institute for AI. 2025, Labuna 2025) that may suffer from undetected bottlenecks or inefficient hyperparameter choices.
To mitigate this risk, we adopted a bottom-up scaling methodology in two phases: first, scaling the GPU count while maintaining fixed input dimensions, then expanding the spatial context through continual pre-training on higher-resolution volumes.

2.2.1 Scaling to multiple nodes

We established single-node baselines before expanding to 2-node and 16-node configurations. At each step, we profiled the pipeline to identify whether GPU compute, inter-node communication, or data loading became the throughput-limiting factor, revealing optimal configurations to train the model at scale.

2.2.2 Scaling Spatial Context: From 256x256x256 to 1536x1536x2048 Voxels

The initial model was trained on 256^3 voxel volumes with a patch size of 16^3, yielding 4K tokens (Sansal et al., 2025). Expanding to 1536×1536×2048 voxels (1.17M tokens) presents fundamental memory constraints due to quadratic attention (Dao et.al , 2022) scaling.

We adopted a two-stage approach: after multi-node scaling on 256^3 volumes, we extended the model by continuing pre-training at higher resolutions (512^3, 1024^3, 1536×1536×2048), incorporating context parallelism (Section 2.2.3) to spread memory consumption across the training cluster. Training on large 3D volumes enables the model to capture both fine-grained local features and broader regional patterns.

2.2.3 Context Length Expansion via Context Parallelism

We implemented context parallelism based on ring attention (Liu et al., 2023), where the input sequence is partitioned across GPUs. Each device computes local attention over its segment while key-value tensors are passed in a ring topology. For a sequence of length N across P devices, each store only N/P tokens, reducing peak memory by a factor of P while maintaining identical outputs. We used PyTorch's native context-parallel API for seamless integration with the ViT-MAE codebase.

Context parallelism for the ViT-MAE architecture introduced two constraints:

Constraint 1: Retained patches must be divisible by the context-parallel group size. We implemented adaptive masking that dynamically adjusts the mask ratio within ±2% of the target masking proportion to ensure divisibility while preserving the target masking proportion.

Constraint 2: The decoder sequence (retained tokens + mask tokens) must be evenly divisible. We implemented a sequence management module that pads or adjusts the mask token count to ensure uniform partitioning.

2.2.4 Distributed Training Framework Selection

Training ViT-MAE across multiple GPUs requires distributing model state (parameters, gradients, optimizer states) across devices. We evaluated three strategies: DeepSpeed ZeRO Stage 2 (Rajbhandari et al., 2020), ZeRO Stage 3, and PyTorch FSDP2 (Zhao et al., 2023).

ZeRO-2 partitions gradients and optimizer states while replicating parameters across GPUs, avoiding parameter all-gather communication during forward/backward passes. This minimizes overhead when parameters fit within per-GPU memory.

ZeRO-3 additionally shards parameters across devices, reducing peak memory but requiring parameter gathering before each layer's forward pass. While enabling larger

models introduces substantial communication overhead when sharding is unnecessary.

FSDP2 implements comparable full sharding within native PyTorch, providing similar memory-communication trade-offs to ZeRO-3 with tighter Autograd integration.

2.2.5 Multi-Node Scaling Infrastructure

Training was scaled to 128 NVIDIA H200 GPUs across 16 nodes with 3,200 Gbps EFA networking, managed using Amazon SageMaker HyperPod (Amazon Web Services, 2025) for automated health monitoring, fault recovery, and distributed checkpointing.

3. Results and Analysis

All experiments were conducted on p5en.48xlarge nodes until mentioned, each equipped with 8 NVIDIA H200 GPUs with 141 GB HBM3e memory per GPU. The Global Batch Size **(GBS)** was calculated as: Number of Nodes × Number of GPUs per Node × Per-GPU Batch Size × Gradient Accumulation Steps.

3.1 Data Pipeline Performance: Direct Streaming vs. Shared File System

To evaluate data pipeline performance, we compared shared file system against direct streaming from object storage on a 2-node cluster with 16 GPUs and a Global Batch Size of 1,536. Shared file system (FSx for Lustre) provisions throughput proportionally to storage capacity (1,000 MBps/TiB), yielding only ~2.4 GB/s cluster-wide throughput for our 2.4 TiB storage, creating a severe data pipeline bottleneck that left GPUs idle during training. In contrast, streaming directly from object storage (S3) sustained 4-5 GB/s per node (8-10 GB/s across the cluster), maintaining high GPU utilization throughout training. As shown in Table 1, direct streaming achieved 5.7× higher throughput compared to shared file system.

Data Source	GBS	Throughput (samples/second)
Shared File System (FSx)	1536	34.82
Direct streaming (S3)	1536	**196.92**

Table1: Data source throughput comparison

3.2. Optimizations for scale

Building on the data pipeline improvements from Section 3.1, we conducted an ablation study examining batch size, gradient accumulation steps, data loader configuration, and distributed training frameworks at 16 nodes (128 GPUs) with a GBS of 8,192. The following section shows insights from our studies.

3.2.1 Distributed training framework comparison

As shown in Figure 2, DeepSpeed ZeRO-3's parameter all-gather operations at each layer introduced substantial communication overhead, resulting in more than 2× throughput degradation compared to ZeRO-2. Since the model's gradients and optimizer states can be fully accommodated by the 141 GB HBM3e memory per H200 GPU, ZeRO-2's gradient and optimizer partitioning strategy proved optimal—parameter sharding provided no practical benefit while incurring significant performance penalties.

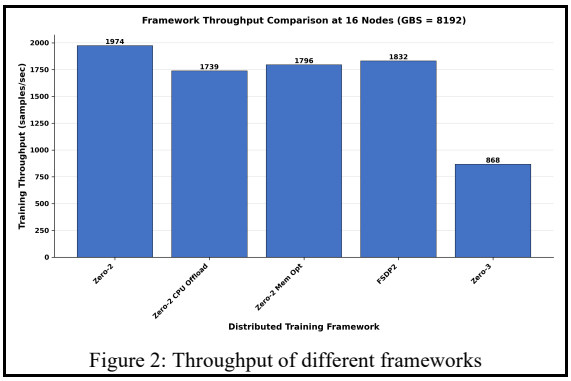

Figure 2: Throughput of different frameworks

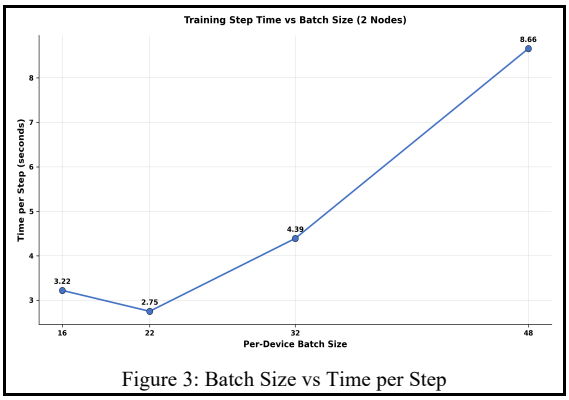

Figure 3: Batch Size vs Time per Step

3.2.2 Batch size scaling behavior

As shown, in Figure 3, Training throughput did not increase monotonically with per-GPU batch size, even when GPU memory capacity remained available. Beyond a critical threshold, batch processing time increased significantly due to CPU-GPU communication bottlenecks during micro-batch preparation and transfer. This finding reveals that data

movement overhead, rather than compute capacity or memory constraints, can become the limiting factor in distributed training at scale

3.3 Linear scaling with the number of nodes

Using the optimized parameters and ZeRO-2 framework, we extended our experiments to larger multi-node configurations. Table 2 demonstrates that the optimized configurations achieve near-linear scaling with the number of nodes, delivering a throughput exceeding >16× when scaling from a single node to 16 nodes.

Nodes	Per GPU Batch Size	Global Batch Size	Throughput samples/sec
1	32	512	106.44
2	32	1024	233.50
8	32	4096	898.25
16	32	8192	1973.97

Table 2: Throughput with increasing number of nodes

3.4. Context length expansion via Context Parallelism

As described in Section 2.3, we implemented the Ring Attention-based context parallelism (CP) to enable processing of larger seismic volumes. Table 3 shows training throughput across different input resolutions with a patch size of 16^3.

Input size	Context Length	Global Batch Size	Samples /Second CP=0	Samples /Second CP=16
(256,256,256)	4K	8192	**1973.97**	676.58
(512,512,512)	33K	2560	**132.78**	63.71
(1024,1024,1024)	262K	512	3.66	**9.72**
(1536,1536,2048)	1.17M	128	OOM	**1.06**

Table 3: Effect of context parallelism for larger cubes. Here, OOM refers to Out-Of-Memory

Table 3 shows that for small volumes up to 512^3, context parallelism across 16 nodes (CP=16) incurs overhead from partitioning and distributing context across devices, resulting in lower throughput than standard training. However, for larger volumes (1024^3), context parallelism achieves ~3× higher throughput. Critically, context parallelism enabled processing of 4.5× larger volumes (1536×1536×2048) with 1.17M token context length—configurations that cause out-of-memory errors without context parallelism.

4. Discussion

This work has a real impact by shortening model development cycles, reducing compute costs, and speeding up applications such as AI-assisted geological interpretation. These methods can be easily adapted to related fields such as mineral exploration, carbon capture monitoring, and geothermal energy, creating a reusable framework for subsurface foundation models within the earth science community.

5. Conclusions

This work presents a systematic approach to scaling 3D ViT-MAE seismic foundation models from single-node to multi-node configurations while enabling context-length expansion via context parallelism. Our incremental scaling methodology—progressing from small GPU clusters to large-scale deployments and extending context windows through continual pre-training—provides a cost-effective validation framework for developing large-scale foundation models in geoscience applications. This approach reduces computational risk and enables efficient resource allocation during the model development lifecycle.