A study on lossy compression for background wavefield storage in LSM

Átila Saraiva Quintela Soares, and Mauricio Sacchi, University of Alberta*

SUMMARY

Least-squares reverse time migration (LSRTM) stands out as an effective method for delineating complex geological formations, offering a way to offset the limitations posed by limited offset data. However, a significant challenge arises from the computation of gradients in each iteration, which demands storing the entire background wavefield. For voluminous 3D models, this requirement can escalate to storing terabytes of data for every shot, underscoring the limitations even when employing advanced strategies like optimal checkpointing, enhanced boundary conditions, and reduced wavefield reconstruction due to the massive size of these models. Albeit a simple approach, archiving the entire background wavefield on disk becomes increasingly relevant. Nonetheless, this approach introduces several complications, including concerns over the lifespan of storage mediums, performance bottlenecks, and space usage constraints. An alternative strategy involves leveraging computation in exchange for an expanded storage footprint, in terms of both space and durability, by applying compression techniques. Nevertheless, when it comes to scientific data, employing lossless compression on floating-point numbers often falls short in significantly reducing space requirements, which brings to light the potential benefits and necessity for lossy compression methods. This study focuses on exploring the impact of varying degrees of lossy compression, specifically utilizing the ZFP compression algorithm, on the convergence, performance, and quality of LSM images, with the goal of providing a deeper understanding through practical experiments.

INTRODUCTION

The least-squares reverse time migration (LSRTM) main performance bottleneck consists of the requirement to preserve the forward background wavefield for computing the gradient via the adjoint state method during reverse propagation. This challenge is not unique to LSRTM but extends to other adjoint method-based inversion/optimization techniques and time-dependent partial differential equation (PDE) solvers, as documented in the literature (Cardesa et al., 2020; Hascoet and Pascual, 2013; Dussaud et al., 2008), including full waveform inversion (FWI). Although strategies such as checkpointing (Griewank, 1992; Symes, 2007), wavefield reconstruction with decimation(Yang et al., 2016), and innovative boundary condition methods (Dussaud et al., 2008; Yang et al., 2014) have been proposed to mitigate the storage burden, these techniques still necessitate considerable storage capacity and computational power, especially for large 3D models.

This research delves into the consequences of applying lossy compression, with a focus on the ZFP algorithm for the storage of the background wavefield during LSRTM procedures. It seeks to fill a research void concerning the effects of compression on the convergence rates and the quality of images produced through least-square migration techniques. This inquiry extends previous studies on compression's role in forward wave inversion (FWI) and reverse time migration (RTM) (Kukreja et al., 2022; Huang et al., 2023), with the objective of shedding light on the compromises associated with the use of lossy compression in computing-intensive applications like LSRTM.

Throughout this document, vectors are represented as bold lowercase letters, linear operators by bold uppercase, functional mappings by uppercase, and other functions in lowercase and constants are signified using Greek symbols.

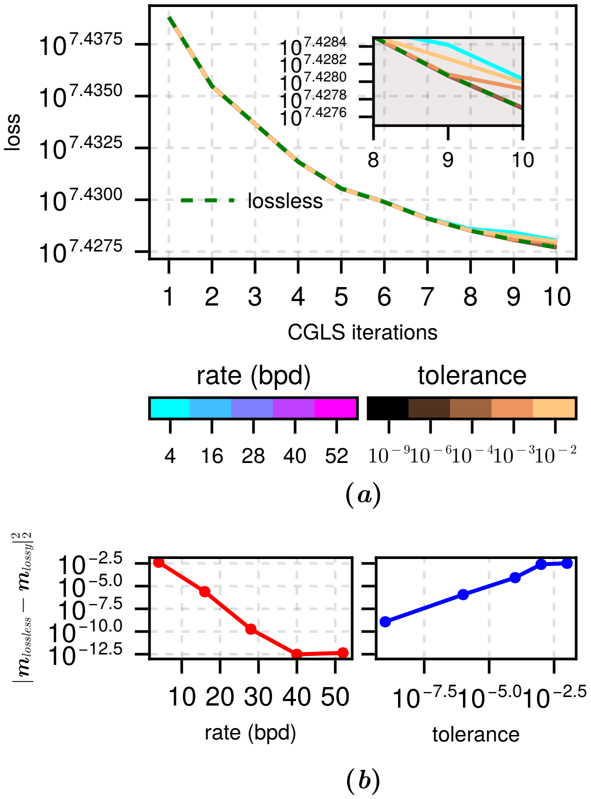
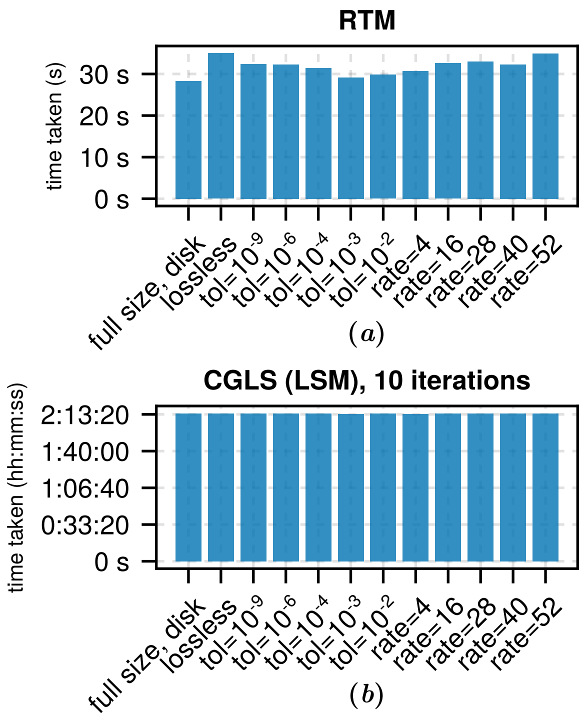


Figure 1: Convergence curves for different rates and tolerances (a). L2 norm of the lossless least-squares model and the lossy one for different rates and tolerances (b).



(a)

(b)

Figure 2: Original size=168.61 GB

LEAST-SQUARES REVERSE TIME MIGRATION

The code implemented for this work, has forward and adjoint states, and gradient based on the L2 norm cost function:

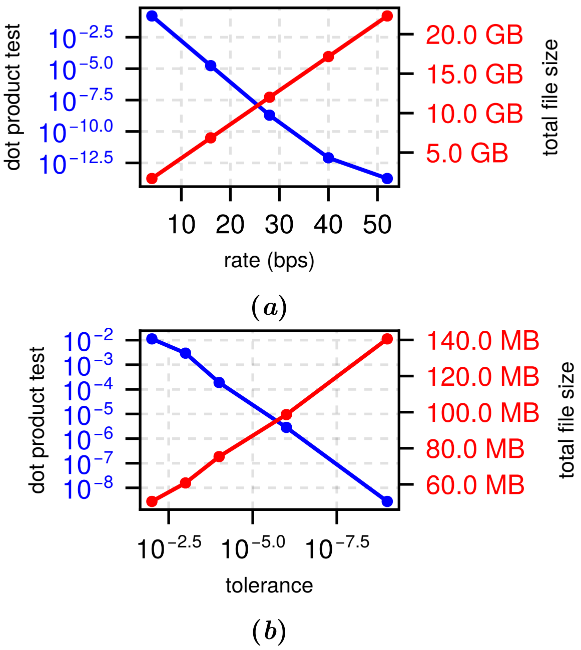
$$J(\mathbf{m}) = \frac{1}{2}\sum_{s,r}\int_0^T \left(S_{s,r}p_s(\mathbf{x}) - d_{s,r}(\mathbf{x})\right)^2 dt, \qquad (1)$$

where $S_{s,r}$ is the source-receiver sampling operator, p_s is the forward state variable, m is the reflectivity model, and $d_{s,r}$ is the data. The least-squares migration technique has a linear forward state equation mapping with relation to the model, which allows for least-squares optimization tecniques such as conjugate gradient. We used the following parametrization for the forward-state equations:

$$F(m) = \begin{cases} p_o(\mathbf{x},0) = 0, \\ \frac{\partial p_o(\mathbf{x},0)}{\partial t} = 0, \\ \frac{1}{c_o(\mathbf{x})^2}\frac{\partial^2 p_o(\mathbf{x},t)}{\partial t^2} - \nabla^2 p_o(\mathbf{x},t) = f_s(\mathbf{x},t), \\ \frac{1}{c_o(\mathbf{x})^2}\frac{\partial^2 \delta p(\mathbf{x},t)}{\partial t^2} - \nabla^2 \delta p(\mathbf{x},t) = \frac{\partial^2 p_o(\mathbf{x},t)}{\partial t^2} m, \end{cases} \qquad (2)$$

where p_o is the background wavefield, c_o is the background velocity model, δp is the perturbation wavefield. The adjoint state equations are:

$$F^*(m) = \begin{cases} \lambda(\mathbf{x},T) = 0, \\ \frac{\partial \lambda(\mathbf{x},T)}{\partial t} = 0, \\ \frac{1}{c_o(\mathbf{x})^2}\frac{\partial^2 \lambda(\mathbf{x},t)}{\partial t^2} - \nabla^2 \lambda(\mathbf{x},t) = \sum_r \left(S_{s,r}^T(S_{s,r}\delta p(\mathbf{x},t) - d_{s,r}\right), \end{cases} \qquad (3)$$



(a)

(b)

Figure 3: Dot product test result vs rate and tolerance, matched with size of background wavefield.

where λ is the adjoint state variable, and S^T is the adjoint of the sampling operator. Finally, the gradient is given by:

$$\frac{\partial}{\partial m}J(m) = \sum_s \int_0^T \lambda(\mathbf{x},t)\frac{\partial^2 p_o(\mathbf{x},t)}{\partial t^2}dt. \qquad (4)$$

Please observe that the gradient calculation requires the background wavefield p_o to be saved at all time steps for it to be correlated with λ. Given λ has final boundary conditions (i.e. $\lambda(\mathbf{x},T) = 0$), the efficient way to compute it is to propagate the adjoint wavefield associated with it backwards together with the calculation of the gradient in Equation 4, which constitutes the main problem of the LSRTM and RTM in general. In this work, the ZFP compression algorithm is used to compress p_o to decrease the storage size it occupies.

In this work, I actually use a relative dot product test result defined by:

$$\alpha = \left| \frac{\langle \mathbf{L}\hat{\mathbf{m}}, \hat{\mathbf{d}}\rangle - \langle \hat{\mathbf{m}}, \mathbf{L}^*\hat{\mathbf{d}}\rangle}{\langle \mathbf{L}\hat{\mathbf{m}}, \hat{\mathbf{d}}\rangle} \right|, \qquad (5)$$

where $\hat{\mathbf{m}}$ is a random model, and $\hat{\mathbf{d}}$ is a random data. $\mathbf{L}$ is the linear operator that represents the linear operator that in the foward pass applies the forward state equations and in the adjoint pass, applies the gradient.

OVERVIEW OF ZFP COMPRESSION TECHNIQUE

ZFP is an innovative compression algorithm tailored for multi-dimensional arrays, employing a unique approach by segmenting the data into blocks of 4^d elements, where d is the number of dimensions of the input array. This strategy, reminiscent of techniques used in graphical texture compression, allows for independent processing of each block, enhancing the efficiency and flexibility of the compression process.

In the experiments performed in this work, the multifile compressor of the `SequentialCompression.jl` package[*] was used. It consists of splitting the input array in its last dimension, compressing each part in parallel using the ZFP algorithm, and saving the result in different files, one per thread. Additionally, the experiments in this work base themselves on analyzing rate and tolerances, and their effect on least-squares migration, given precision has very similar behaviour to rate.

There are four compression modes for ZFP, fixed-precision, fixed-tolerance, fixed-rate and lossless. Since the ZFP algorighm first segments the data into 4^d blocks, where d is the number of dimensions, these modes are specified in a per-block basis. Both the fixed-precision and fixed-rate allow the user to specify the number of bit-planes for the block or individual values respectively. The fixed-accuracy allows the user to specify the desired maximum absolute value difference between the lossy compressed blocks and the original uncompressed block of the original data.

NUMERICAL EXPERIMENT

This study involved conducting 10 Conjugate Gradient Least Squares (CGLS) iterations and a basic Reverse Time Migration (RTM) using the linear operator from Section , employing the `SequentialCompression.jl` for wavefield compression. The experiment utilized a downsampled, smoothed Marmousi model(Brougois et al., 1990) sized at $nz, nx = 201, 512$ and grid spacing $dz, dx = 10, 10$ m. With 100 shots spaced evenly from $x = 100$ m to $x = L_x$ m, where L_x represents the model's width, and an absorbing boundary of $nb = 50$, the simulation ran for $nt = 2048$ timesteps at a max frequency of 15Hz, using observed data from the full second-order acoustic wave equation.

Figure 1a illustrates the convergence across different compression rates and tolerances, showing deviation from lossless compression starting at the 8th iteration for lower rates and higher tolerances. Figure 1b examines the L2 norm differences between lossless and lossy models, revealing a direct link between tolerance, rate, and error, with the latter decreasing as the rate increases. Despite the per-block error control, this linear trend underscores the tolerance's impact on overall error.

Figure 4 compares the least squares migrated images under various rates and tolerances, indicating minimal perceptual differences. These results highlight significant space savings, defined as:

$$\text{space saving} = 1 - \frac{\text{compressed size}}{\text{uncompressed size}}, \tag{6}$$

with uncompressed and compressed sizes calculated over all shots. Notably, even at the lowest rate and highest tolerance, space savings can exceed 90%, based on an original uncompressed wavefield size of 168.61 Gb.

A secondary experiment evaluated the dot product test outcome, using the same velocity model but with random reflectivities and data, as detailed in Equation 5. Figure 3 show that there is a inverse proportinaly relationship between rate and the dot product test result, with a sweet spot between total file size of the background wavefield and the dot product test result being 30 bits per double. For tolerance it shows that the lower the tolerance, the lower the dot product test result, with a sweet spot of total file size of the background wave field and dot product test result around 10^{-6}. The values of total fize size however are in different orders of magnitude for rate and tolerances in this case, most likely because of the nature of random reflectivity, with tolerances giving much lower sizes.

CONCLUSIONS

The findings highlight the efficiency of implementing the ZFP compression algorithm in the LSRTM (Least Squares Reverse Time Migration) process, demonstrating minimal impact on convergence and image quality. The integrity of the approach is supported by dot product test outcomes, indicating negligible errors introduced by the compression, thereby preserving data integrity within acceptable loss parameters. The achieved space reductions are notable, even with stricter compression settings, emphasizing the ZFP algorithm's effectiveness in reducing data volume.

To sum up, this investigation confirms the feasibility of integrating the ZFP compression algorithm into LSRTM operations. It paves the way for improved data handling and computational efficiency in seismic inversion activities. This study suggests a viable strategy for managing large-scale data in geophysical exploration, offering insights that could be applicable across various fields facing challenges with large data volumes.

ACKNOWLEDGMENTS

We want to thank the Signal Analysis and Imaging Group sponsors at the University of Alberta for supporting the stimulating research environment that allowed the preparation of this work.

[*] The package can be found at the GitHub repository: `https://github.com/AtilaSaraiva/SequentialCompression.jl`.

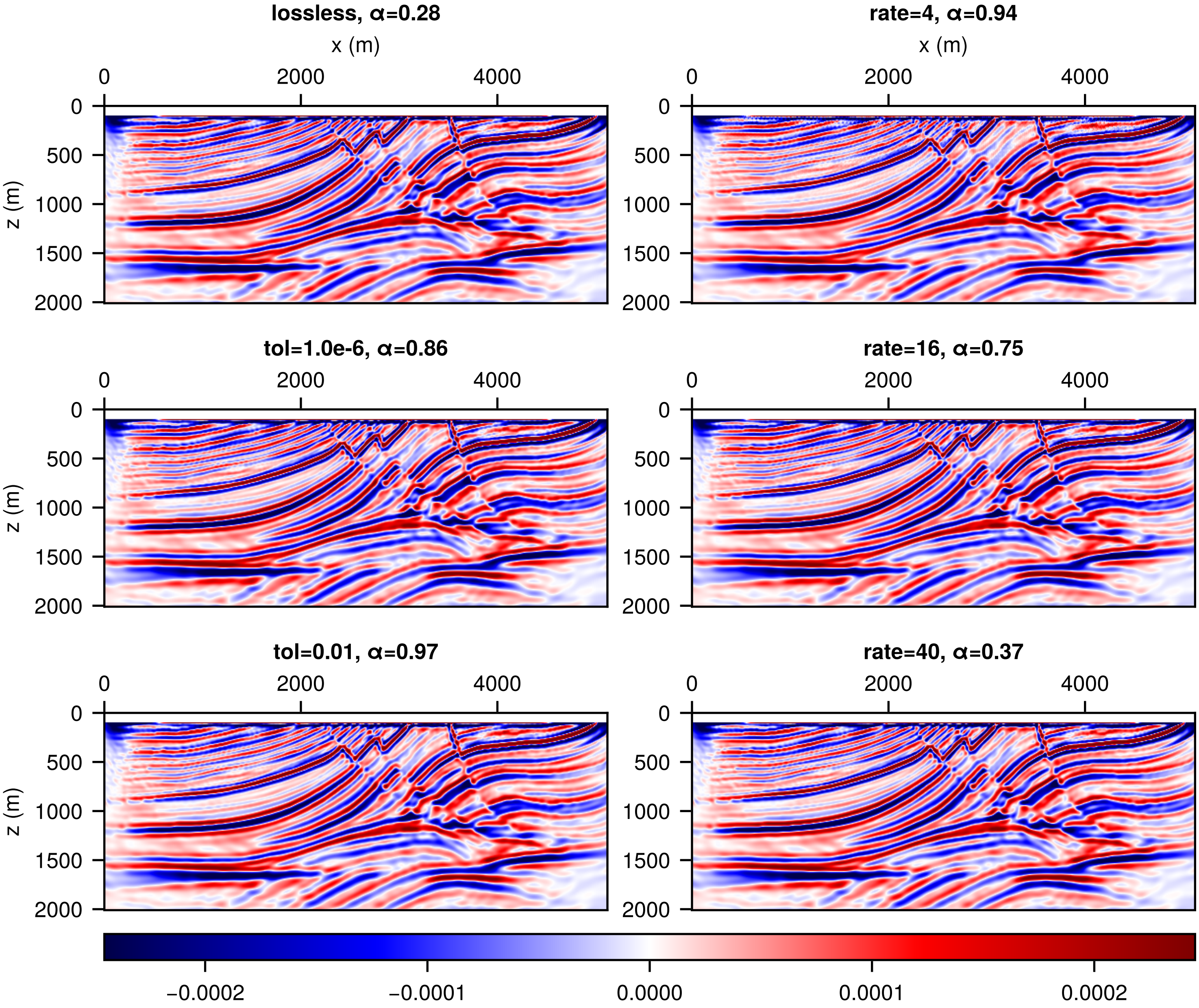


Figure 4: CGLS migrated image after ten iterations for different compression tolerances and rates. Original summed over shots background wavefield size=168.61 GB