Joint Inversion of Time-Lapse Surface Gravity and Seismic Data for Monitoring of 3D CO_2 Plumes via Deep Learning

Adrian Celaya *
Rice University
Houston, TX 77005

Mauricio Araya-Polo
TotalEnergies EP Research and Technology USA
Houston, TX 77002

Abstract

Geological storage of CO_2 is a key climate change mitigation strategy. As important as location selection and injection planning is monitoring that the gas is contained for long periods of time. We introduce a fully 3D, deep learning-based approach for the joint inversion of time-lapse surface gravity and seismic data for reconstructing subsurface density and velocity models. The target application of this proposed inversion approach is the prediction of subsurface CO_2 plumes as a complementary tool for monitoring CO_2 sequestration deployments. Our joint inversion technique outperforms deep learning-based gravity-only and seismic-only inversion models, achieving improved density and velocity reconstruction, accurate segmentation, and higher R-squared coefficients. Future work will focus on validating our approach with larger datasets, simulations with other geological storage sites, and, ultimately, field data.

1 Introduction

Reducing CO_2 concentration in the atmosphere is critical to control climate change. These efforts involve implementing various technologies, such as efficient fossil-based fuel consumption, expanding absorption sources through afforestation/reforestation, and adopting CO_2 capture, utilization, and storage (CCUS) techniques. Among the CCUS technologies currently deployed worldwide, CO_2 geological storage has emerged as a promising approach. This technology involves capturing CO_2 from fixed sources or directly from air, then injecting it into underground formations.

Injecting CO_2 is just part of the process; during and after injection, ensuring the integrity of these geological sites necessitates ongoing monitoring. Regulatory authorities mandate the demonstration of storage volume containment and the detection of any potential CO_2 leakage or unwanted migration. Given the time scales involved in monitoring CO_2 storage sites, traditional monitoring techniques based on borehole sensors or surface seismic monitoring may not be practical or economically viable. Remote sensing by other modes, such as gravity, might be economically viable and technically feasible if combined with traditional seismic approaches.

Once data from the field is recorded, geophysical inversion techniques are deployed. These widely used techniques for interpreting data sets and recovering subsurface physical models can be crucial in monitoring CO_2 storage sites [1, 2]. The recovered models, which encompass parameters such as velocity, density, resistivity, or saturation, provide essential information about the structural and compositional characteristics of the subsurface. Integrating multiple datasets collected over the same area is known as geophysical inversion [3, 4]. This approach enhances the reconstruction

*Adrian Celaya is affiliated to TotalEnergies EP Research and Technology USA and Ph.D. student in the Department of Computational Applied Mathematics and Operations Research at Rice University.

LatinX in AI (LXAI) Research Workshop @ NeurIPS 2023.

of subsurface structures and facilitates the identification of changes or anomalies associated with activities like CO_2 storage [5]. However, effectively integrating the information embedded in multiple geophysical datasets presents a practical challenge from a computational point of view (i.e., storage, memory, and compute time), which compounds the inherent difficulties of solving nonlinear inversion problems [6]. These challenges especially become apparent when working with realistic 3D synthetic or field data.

In this work, we develop an effective *supervised* 3D deep learning (DL)-based inversion method to recover high-resolution subsurface CO_2 plumes from both surface gravity and seismic data. To the best of our knowledge, this is the first fully 3D approach for the joint inversion of surface gravity and seismic data, which is tested on realistic, physics-simulated CO_2 plumes.

2 Previous Work

Conventional inversion methods find a model with the minimum possible structure and whose forward response fits the observed data [7–9]. The minimum structure is achieved by minimizing model roughness through a least squares regression, resulting in a smooth model. While the least squares regression produces smooth subsurface models, the predicted models are often larger and exhibit smaller density values than the actual model [10, 11].

DL is an emerging alternative to traditional geophysical inversion [12–15]. Over the last several years, deep convolutional neural networks (CNNs) have achieved state-of-the-art results in various computer vision applications such as image classification, segmentation, and generation [16–18]. CNNs have recently been used for inversion of seismic [19–21], electromagnetic imaging [22, 23], electrical resistivity [24, 25], and time-lapse surface gravity data [26–29]. However, these works focus solely on inverting a single modality and do not explore joint inversion with multiphysical data.

Using simulated CO_2 plumes from the onshore Kimberlina site, Um et al. developed a 2D DL architecture to perform a joint inversion with seismic, electromagnetic, and gravity data [5]. Additionally, they use a modified version of their architecture to invert their imaging data types individually. In each case, their DL-based approach can recover CO_2 plumes. However, their approach still does not perform DL-based inversion in a fully 3D setting. Hu et al. present a physics-informed DL-based approach for inverting electromagnetic and seismic data for recovering subsurface anomalies [3]. While their approach successfully reconstructs the anomalies, this approach also uses 2D data and does not consider 3D data or the computational cost of implementing physics-informed inversion for such data; as opposed to our work, which implements fully 3D joint inversion with seismic and surface gravity data.

3 Problem Statement

Classical inversion techniques (for seismic [6]) aim to minimize a cost function that measures the difference between the forward response of a given subsurface model and the observed data when the subsurface is directly stimulated. Let F be our forward operator, $\mathbf{m}$ be a subsurface model, and $\mathbf{d}_{\text{obs}}$ be the observed data during stimulation. Then the classical inversion problem can be written as

$$\min_{\mathbf{m}} ||F(\mathbf{m}) - \mathbf{d}_{\text{obs}}||_2^2. \tag{1}$$

Note that this problem takes a single input (i.e., $\mathbf{d}_{\text{obs}}$) and produces a single predicted subsurface model.

In contrast, joint inversion is an extension of the classical formulation given by (1) that maps multiple inputs (i.e., gravity, seismic, and electromagnetic data) to multiple outputs (i.e., density, velocity, and resistivity models). For a given number of inputs $\mathbf{d}_{\text{obs}}^1, \ldots, \mathbf{d}_{\text{obs}}^n$ and appropriate forward operators $F_1, \ldots, F_n$, our joint problem is given by

$$\min_{\mathbf{m}^1, \ldots \mathbf{m}^n} \sum_{i=1}^{n} ||F_i(\mathbf{m}^i) - \mathbf{d}_{\text{obs}}^i||_2^2 + \alpha \sum_{i \neq j} \Phi(\mathbf{m}^i, \mathbf{m}^j), \tag{2}$$

where Φ is a coupling function used to link different physical models via known petrophysical relations or other metrics like SSIM (in a Machine Learning context), and α is a parameter controlling

the contribution of the coupling term [30, 31]. Joint inversion is much more time-consuming than independent inversions because of the additional terms in the cost function and the need to exchange information between different models via the coupling terms [3]. There also is a need to determine or adjust the weighting parameter α, which adds another layer of complexity to the standard joint inversion method [3].

Our goal is to train a DL-model that can accurately predict the changes in subsurface density and velocity given corresponding variations in surface gravity and seismic data.

4 Data Preparation

Located 60km off the western coast of Norway, the Johansen formation [32] is a promising CO_2 storage site with a theoretical capacity exceeding 1Gt of CO_2. The formation encompasses an aquifer with an approximate thickness of 100m, spans 100km in the north-south direction and 60km from east to west, and is at a depth that ranges from 2,200m to 3,100m below sea level. This setting provides optimal pressure and temperature conditions for injecting CO_2 in a supercritical state.

We generate data based on the Johansen formation using the process described in [29]. That is, we generate a number of distinct geological realizations that vary in porosity and permeability. We conduct a fluid flow simulation that assumes a 100-year injection period followed by a 400-year migration period. This process produces subsurface changes in density. To convert these density models to V_p models for forward seismic modeling, we use the following conversion:

$$V_p(\mathbf{r}) = \sqrt{\frac{\kappa + \frac{4}{3}\mu}{\rho(\mathbf{r})}}, \tag{3}$$

where $\mathbf{r}$ is the spatial coordinate in our model, $\kappa = 8.14$GPa is the bulk modulus, $\mu = 1.36$GPa is the shear modulus, and ρ is the density value [33, 34]. Here, we assume that the formation comprises 80% shale and 20% sandstone to get the values for κ and μ [35, 36]. Using this process, we generate 180 density/velocity pairs. For preprocessing, we resample each density and velocity model from their original resolution of $440\times530\times145$ to $256\times256\times128$.

4.1 Modeling Gravity

Given a density perturbation $\Delta\rho$ observed between the current and original (i.e., base) acquisition, the gravity field recorded at a station located at $\mathbf{r}'$ can be expressed as:

$$\mathbf{g}(\mathbf{r}') = \gamma \iiint_V \frac{\mathbf{r} - \mathbf{r}'}{||\mathbf{r} - \mathbf{r}'||_2^3} \Delta\rho(\mathbf{r}) dV \tag{4}$$

where V is the volume of the reservoir, and γ is Newton's gravitational constant. For more details on gravity modeling, see [29]. Assuming that gravity sensors are placed in a uniform grid every 500m, our surface gravity maps are size 88×106. An illustration of a change in surface gravity for a given density permutation in the subsurface is shown in Figure 1 (right).

4.2 Forward Seismic Modeling

Forward seismic modeling approximates the behavior of seismic waves propagating through a mechanical medium $\mathbf{m}$ and is given by the elastic wave equation:

$$\frac{\partial^2 u}{\partial t^2} - \mathbf{V}_p^2 \nabla(\nabla \cdot u) - \mathbf{V}_s^2 \Delta u = \mathbf{f}, \tag{5}$$

where $\mathbf{u} = \mathbf{u}(x, y, z, t)$, is the seismic wave displacement, $\mathbf{V}_p$ is P-wave velocity (compression/rarefaction), $\mathbf{V}_s$ is S-wave velocity (shear stress), and $\mathbf{f}$ is the perturbation source (i.e., shot) function [37]. While (5) more accurately describes seismic wave propagation, it is often preferred (as in this work) to approximate the solution $\mathbf{u}$ by the acoustic wave equation, which assumes only P-waves and requires less computational resources and parameters, as compared to solving (5) [37].

3D seismic data consumes a large amount of memory and storage, up to dozens of TB for the raw data in our case. We compute spatial decimation evenly, but temporal, which in seismic signals

represents depth, decimation favors samples that convey information in the area of interest (i.e., the reservoir). Further, we collapse our seismic data by adding the recorded data from each shot and then boost the later time signals by multiplying by a monotonically increasing function. This method is fully described in [37]. Finally, we resize each collapsed and boosted seismic cube from its original resolution of $167 \times 154 \times 941$ to $256 \times 256 \times 512$ by first taking every other slice (i.e., time step) in the z-direction and then linearly interpolating to the final resolution.

Like forward gravity modeling, we look at the differences between the original or base seismic data and the current data. Figure 1 (left) illustrates an example of the change in seismic data from the original and current acquisitions.

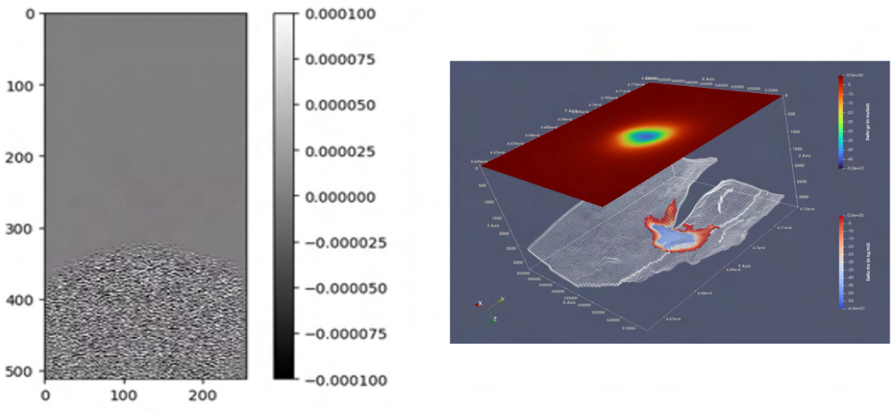

Figure 1: (Left) Example of seismic difference cube. (Right) Illustration of a change in surface gravity for a given density perturbation in the subsurface [29].

5 Methods

5.1 Network Architecture

We use a modified 3D U-Net architecture to map 2D surface gravity maps and 3D seismic cubes to subsurface changes in density and velocity. Each input is fed into a dedicated encoder branch. The first step for the surface gravity encoder branch is to resize the input to match the reservoir geometry via linear interpolation. Then the 2D features are converted into a 3D volume using a pointwise convolution, where the number of channels equals the subsurface model's depth (i.e., the output). This resulting 3D volume is the input to the first 3D encoder branch of the network. The seismic encoder branch starts with a 3D seismic cube as input. However, the seismic cubes are larger in the depth dimension than the network output. To address this, we use two convolutional layers with strides $1 \times 1 \times 2$ to reduce the depth dimension of the seismic cube from 512 to 128. The resulting resized seismic features are the input to the second encoder branch of our architecture. In each resolution level of our encoder branch, we apply two convolutional layers and downsampling via max pooling four times. A convolutional layer consists of two convolutions, each followed by batch normalization and a ReLU non-linearity.

At the bottleneck of our architecture, we concatenate the encoded seismic and gravity features and apply two convolutional layers to merge the individual features. We also apply autoencoder regularization. However, because we have two inputs, we separately decode the seismic and gravity features (before concatenation and convolution) to reconstruct their respective inputs.

Our decoder branch upsamples the combined features and concatenates them with the corresponding features from the encoder branches in the skip connections. Like with the network proposed by [29], we split the output of the decoder branch into three separate outputs: two regression branches to predict density and velocity and one for segmenting the plume. Figure 2 shows a detailed sketch of our proposed architecture.

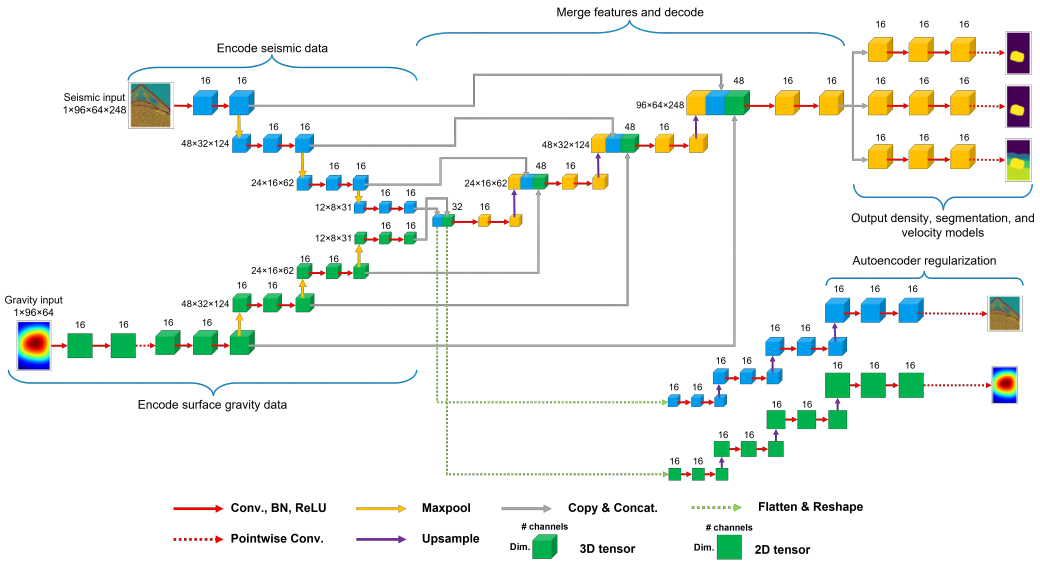

Figure 2: Detailed sketch of our proposed architecture for jointly inverting surface gravity and seismic data. Our architecture uses two separate encoder branches for each input and decodes them jointly. The output of the decoder branch splits into three separate outputs: two regression branches to predict density and velocity and one for segmenting the plume. Additionally, this architecture uses autoencoder regularization.

5.2 Loss Function

Our loss function consists of three components - segmentation, regression, and autoencoder losses.

The segmentation loss is the Generalized Dice Loss (GDL) proposed by Sudre et al. [38]. Unlike the original Dice loss proposed by [39], the GDL uses weighting terms to account for class imbalance. This loss function is given by

$$\mathcal{L}_{gdl} = \frac{\sum_{k=1}^{C} w_k ||T^k - P^k||_2^2}{\sum_{k=1}^{C} w_k \left(||T^k||_2^2 + ||P^k||_2^2\right)}, \tag{6}$$

where C denotes the number of segmentation classes, P^k is the k-th class in the predicted mask, and T^k is the same for the ground truth. The term w_k is the weighting term for the k^{th} class and is given by $w_k = \left(\frac{1}{\sum_{j=1}^{C} \frac{1}{N_j}}\right) \frac{1}{N_k}$. Here, N_k is the total number of pixels belonging to the class k over the entire dataset. Note that the weights w_k are pre-computed and remain constant throughout training. In our case, the number of segmentation classes equals two; background and foreground. The computed class weights are approximately 0.0015 and 0.9985 for the background and foreground classes, respectively.

The regression and autoencoder losses use the mean squared error loss for their respective inputs. For a general input pair (T, P), this loss is given by $\frac{1}{N}||T - P||_2^2$. For the regression tasks (i.e., density and velocity reconstruction) the inputs to this loss function are the true and predicted density and velocity models. For the autoencoder branches, the inputs to this loss function are the true and reconstructed gravity and seismic inputs.

Our overall loss function is a weighted convex combination of the previously described components and is given by

$$\mathcal{L} = 0.375\mathcal{L}_{\rho} + 0.2\mathcal{L}_{gdl} + 0.375\mathcal{L}_{v} + \frac{0.05}{2}\left(\mathcal{L}_{ae}^{grav} + \mathcal{L}_{ae}^{seis}\right), \tag{7}$$

where $\mathcal{L}_{\rho}$, $\mathcal{L}_{gdl}$, $\mathcal{L}_{v}$, $\mathcal{L}_{ae}^{grav}$, and $\mathcal{L}_{ae}^{seis}$ are the density model, segmentation, velocity model, gravity autoencoder, and seismic autoencoder losses, respectively. Note that we select these weights via a partial grid search.

5.3 Training and Testing Protocols

To train our neural networks, we use the Adam optimizer [40] with an initial learning rate of 0.001 and a cosine decay schedule with restarts [41]. We train our model to convergence ($\approx$ 400 epochs) with a batch size of 8. We use 80% of our dataset as a training set, use 10% of the training data as a validation set, and use the remaining 20% of the data as a test set. To compare the effect of joint inversion vs. inversion with only gravity or seismic data, we use the architecture described in [29] for the portions of our dataset corresponding to either gravity or seismic inversion. Note that the models used for individual inversion only output the properties corresponding to that particular inversion problem. For example, the network used for purely seismic inversion only produces a velocity model as a prediction.

To evaluate the validity of our predicted inversions, we utilize the following metrics: mean squared error in kg/m^3 between the true and predicted density models, mean squared error in m/s between the true and predicted velocity models, mean squared error in μGals between the observed data and the gravity response of the predicted density model, the R-squared coefficient between the true and predicted models (for density and velocity), and the Dice coefficient between the non-zero masks of the true and predicted plume geometry.

Our models are implemented in Python using PyTorch (v2.0.1) and trained on four NVIDIA A100 GPUs [42]. At test time, our DL-based methods produce predictions of size 256$\times$256$\times$128. We resample this output to the original grid resolution of 440$\times$530$\times$145 via linear interpolation to produce our final prediction. Given the sample size, we develop a data parallelism approach by using PyTorch's Distributed Data Parallel implementation. The in-node scalability is nearly ideal, with epochs taking 190 seconds running on 1 GPU and ending up in 45 seconds when running on 4 GPUs.

Method	MSE (kg/m^3)	MSE (m/s)	MSE (μGal)	Dice	R-Squared	
					Density	Velocity
Gravity	0.311 (0.215)	-	**0.601 (1.185)**	0.589 (0.034)	0.769 (0.12)	-
Seismic	-	0.121 (0.100)	-	0.579 (0.096)	-	0.715 (0.175)
Joint	**0.268 (0.174)**	**0.084 (0.057)**	0.634 (1.241)	**0.621 (0.038)**	**0.801 (0.095)**	**0.800 (0.095)**

Table 1: Joint inversion vs. inversion with only gravity or seismic data. Here, we see that our proposed joint inversion generally outperforms gravity and seismic-only inversion for all metrics except for the mean squared error between the observed data and the gravity response of the predicted density model, where the results vs. the gravity-only model are comparable.

Figure 3: Training and validation loss curves on a logarithmic scale for our DL-based joint inversion. Here, both losses converge, indicating that our proposed joint inversion architecture successfully learns a mapping from our surface gravity and seismic data to 3D subsurface density and velocity models.

6 Results

We train our joint inversion model using the methods described in Section 5. Figure 3 shows the training and validation loss curves on a logarithmic scale. This figure shows that both losses converge, indicating that our proposed joint inversion architecture successfully learns a mapping from our surface gravity and seismic data to 3D subsurface density and velocity models.

In Table 1, we see that our proposed joint inversion generally outperforms gravity and seismic-only inversion for all metrics except for the mean squared error between the observed data and the gravity response of the predicted density model, where the results vs. the gravity only model are comparable. Figure 4 shows a 2D cross-section slice from predictions from each model. Visually, the density models produced by the gravity-only and our joint architectures are similar. However, visual differences exist between the seismic-only reconstructed velocity model and the velocity model produced from our joint architecture (i.e., top right corner); those can also be observed in Figure 5 for a different sample (i.e., at different times of plume's migration).

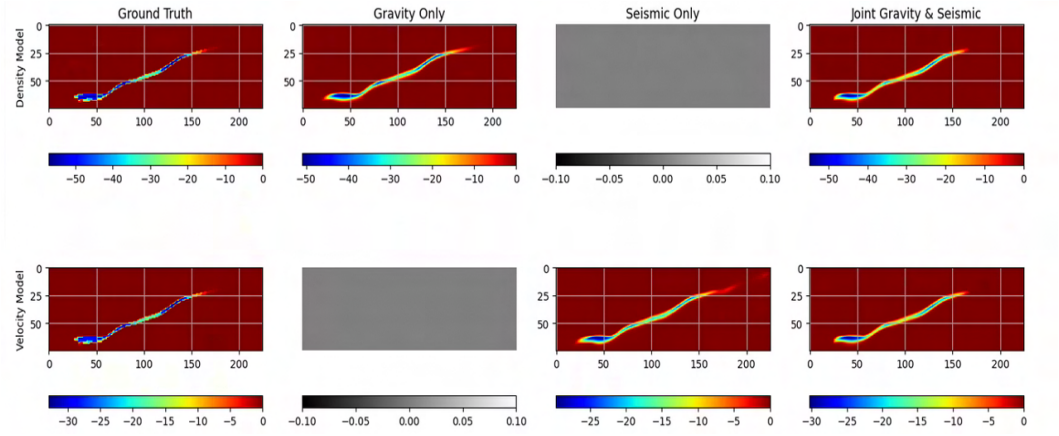

Figure 4: From left to right 2D cross-section slice from ground truth, gravity-only, seismic-only, and joint models. The top row shows density models, and the bottom row shows velocity models. Visually, the density models produced by the gravity-only and our joint architectures are similar. However, visual differences exist between the seismic-only reconstructed velocity model and the velocity model produced from our joint architecture (i.e., top right corner).

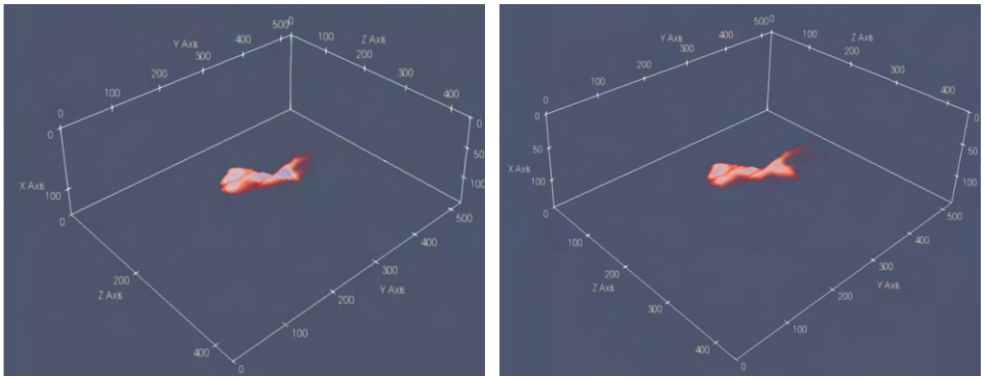

Figure 5: 3D view of label (left) -prediction (right) pair of velocity variation given the CO_2 plume location after 50 years of migration from injection point.

7 Discussion

Our results demonstrate the potential benefits of DL-based joint inversion. The joint inversion model consistently outperforms DL-based gravity-only and seismic-only inversion models across various evaluation metrics. This performance suggests that the fusion of surface gravity and seismic data can lead to more accurate subsurface models. The improved performance of the joint inversion model in terms of density and velocity reconstruction, segmentation accuracy, and R-squared coefficients indicates the effectiveness of the proposed approach in capturing the complexity of subsurface CO_2 plumes.

The benefits of DL-based joint inversion are limited from a computational perspective because combining two different datasets requires more memory and time during training. Our joint inversion model takes roughly 45s per epoch on 4 A100 GPUs with a batch size of 8. In contrast, DL-based gravity inversion takes roughly 20s per epoch, and DL-based seismic inversion takes roughly 30s per epoch for the same number of GPUs and batch size. Additionally, joint inversion takes just over 400 epochs to converge to a solution vs. 200 for both the gravity and seismic-only approaches. The greater number of epochs is possibly explained by the more complex relationship our joint inversion approach has to resolve vs. the single mode methods. Regarding inference, our joint model is comparable to the gravity and seismic-only models, producing predictions in less than one second on a single A100 GPU.

We utilize the PocketNet approach proposed by [43] in our joint architecture. This approach takes advantage of the similarity between the U-Net architecture and geometric multigrid methods to drastically reduce the number of parameters [43, 44]. Additionally, we replace the transposed convolution with trilinear upsampling. With these modifications, we reduce the number of parameters from roughly 33,000,000 to 349,000, yielding a roughly 40% decrease in the time per epoch. Additionally, these modifications allow us to use a larger batch size (8 instead of 4).

Our loss function is a weighted (and convex) combination of several components that address different aspects of the joint inversion problem, like regression, segmentation, and regularization. Other loss functions for these various components may further improve our results. For example, the focal loss for the segmentation component would be a potential candidate since it does not require the computation of weights like the GDL [45]. As previously stated, we develop the convex combinations of weights for the loss function via a partial grid search over the parameter space. The only weight that can significantly affect the results is the weight of the autoencoder regularizer. The selected weight for each autoencoder is 0.025, but we tested these for values ranging from 0 to 0.1. The accuracy remains comparable for non-zero values, but for the value zero (i.e., no autoencoder), we notice a sizable decrease in accuracy for our results. The other weights are relatively balanced, and their perturbations do not significantly impact our results.

While developing our joint inversion model, we observe that the optimization landscape is difficult, with some local minimums corresponding to good values for density model misfit and R-squared and vice versa for velocity models. Adding a coupling term like in the classical joint inversion formulation given by 2 may help avoid getting trapped in local minimums during training. Future work will focus on formulating and testing a term based on the Dice score. The intuition here is to enforce via our modified loss function that the plumes occupy the same physical space and geometry.

Our data comes from simulations of a single geologic formation (Johansen). Further work is needed to test the generalization capability of our joint inversion model to other datasets based on simulations of other CO_2 storage sites (i.e., Snohvit and Kimberlina [46]) and on field data. However, time-lapse CCUS monitoring with gravity (and other non-seismic methods) is a data-poor field. Sufficiently detailed reservoir models for CCUS, especially field data, are hard to come by, and access is limited [29, 46].

8 Conclusions

We developed an effective DL-based joint inversion method to recover high-resolution, subsurface CO_2 density and velocity models from surface gravity and seismic data. We train our joint DL architecture on realistic, physics-simulated CO_2 plumes, surface gravity, and seismic data. This training approach mirrors real-world site data collection. Our joint inversion outperforms gravity and seismic-only inversion techniques for our selected metrics. While there is room for improvement, the results presented here are promising and represent, to the best of our knowledge, the first fully 3D DL-based joint inversion of surface gravity and seismic data derived from a physics simulation of a proposed CO_2 storage site.

Acknowledgments and Disclosure of Funding

This work was supported by TotalEnergies EP Research & Technology USA, LLC.

References

[1] M. Fawad and N. H. Mondol, "Monitoring geological storage of co2: A new approach," *Scientific Reports*, vol. 11, no. 1, p. 5942, 2021.

[2] T. Alyousuf, Y. Li, and R. Krahenbuhl, "Machine learning inversion of time-lapse three-axis borehole gravity data for co2 monitoring," in *SEG International Exposition and Annual Meeting*, p. D011S164R002, SEG, 2022.

[3] Y. Hu, X. Wei, X. Wu, J. Sun, J. Chen, Y. Huang, and J. Chen, "A deep learning-enhanced framework for multiphysics joint inversion," *GEOPHYSICS*, vol. 88, no. 1, pp. K13–K26, 2023.

[4] Y. Sun, B. Denel, N. Daril, L. Evano, P. Williamson, and M. Araya-Polo, "Deep learning joint inversion of seismic and electromagnetic data for salt reconstruction," *SEG Technical Program Expanded Abstracts 2020*, pp. 550–554, 2020.

[5] E. S. Um, D. Alumbaugh, M. Commer, S. Feng, E. Gasperikova, Y. Li, Y. Lin, and S. Samarasinghe, "Deep-learning multiphysics network for imaging CO_2 saturation and estimating uncertainty in geological carbon storage," *Geophysical Prospecting*, 2022.

[6] A. Tarantola and B. Valette, "Inverse problems = quest for information," *Journal of Geophysics*, vol. 50, pp. 159–170, October 1981.

[7] C. De Groot-Hedlin and S. Constable, "Occam's inversion to generate smooth, two-dimensional models from magnetotelluric data," *GEOPHYSICS*, vol. 55, pp. 1613–1624, 12 1990.

[8] Y. Li and D. W. Oldenburg, "3-d inversion of gravity data," *GEOPHYSICS*, vol. 63, no. 1, pp. 109–119, 1998.

[9] M. Nabighian, V. Grauch, R. Hansen, T. LaFehr, Y. Li, J. Peirce, J. Phillips, and M. Ruder, "75th anniversary," *The historical development of the magnetic method in exploration: Geophysics*, vol. 70, no. 6, 2005.

[10] O. Boulanger and M. Chouteau, "Constraints in 3D gravity inversion," *Geophysical Prospecting*, vol. 49, no. 2, pp. 265–280, 2001.

[11] M. Rezaie, A. Moradzadeh, and A. N. Kalateh, "Fast 3d inversion of gravity data using solution space priorconditioned lanczos bidiagonalization," *Journal of Applied Geophysics*, vol. 136, pp. 42–50, 2017.

[12] K. H. Jin, M. T. McCann, E. Froustey, and M. Unser, "Deep convolutional neural network for inverse problems in imaging," *IEEE Transactions on Image Processing*, vol. 26, no. 9, pp. 4509–4522, 2017.

[13] M. Araya-Polo, J. Jennings, A. Adler, and T. Dahlke, "Deep-learning tomography," *The Leading Edge*, vol. 37, no. 1, pp. 58–66, 2018.

[14] Y. Kim and N. Nakata, "Geophysical inversion versus machine learning in inverse problems," *The Leading Edge*, vol. 37, no. 12, pp. 894–901, 2018.

[15] F. Yang and J. Ma, "Deep-learning inversion: A next-generation seismic velocity model building method," *GEOPHYSICS*, vol. 84, no. 4, pp. R583–R599, 2019.

[16] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009.

[17] S. Bakas *et al.*, "Identifying the best machine learning algorithms for brain tumor segmentation, progression assessment, and overall survival prediction in the brats challenge," *arXiv preprint arXiv:1811.02629*, 2018.

[18] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4401–4410, 2019.

[19] A. Adler, M. Araya-Polo, and T. Poggio, "Deep learning for seismic inverse problems: Toward the acceleration of geophysical analysis workflows," *IEEE Signal Processing Magazine*, vol. 38, no. 2, pp. 89–119, 2021.

[20] J. Chen, C. Schiek-Stewart, L. Lu, S. Witte, K. Eres Guardia, F. Menapace, P. Devarakota, and M. Sidahmed, "Machine learning method to determine salt structures from gravity data," in *SPE Annual Technical Conference and Exhibition*, OnePetro, 2020.

[21] S. Li, B. Liu, Y. Ren, Y. Chen, S. Yang, Y. Wang, and P. Jiang, "Deep-learning inversion of seismic data," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 58, no. 3, pp. 2135–2149, 2020.

[22] D. Colombo, W. Li, E. Sandoval-Curiel, and G. W. McNeice, "Deep-learning electromagnetic monitoring coupled to fluid flow simulators," *GEOPHYSICS*, vol. 85, no. 4, pp. WA1–WA12, 2020.

[23] S. Oh, K. Noh, S. J. Seol, and J. Byun, "Cooperative deep learning inversion of controlled-source electromagnetic data for salt delineation," *GEOPHYSICS*, vol. 85, no. 4, pp. E121–E137, 2020.

[24] B. Liu *et al.*, "Deep learning inversion of electrical resistivity data," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 58, no. 8, pp. 5715–5728, 2020.

[25] M. Shahriari, D. Pardo, A. Picón, A. Galdran, J. Del Ser, and C. Torres-Verdín, "A deep learning approach to the inversion of borehole resistivity measurements," *Computational Geosciences*, vol. 24, no. 3, pp. 971–994, 2020.

[26] Q. Yang, X. Hu, S. Liu, Q. Jie, H. Wang, and Q. Chen, "3-d gravity inversion based on deep convolution neural networks," *IEEE geoscience and remote sensing letters*, vol. 19, pp. 1–5, 2022.

[27] W. Yu-Feng, Z. Yu-Jie, F. Li-Hua, and L. Hong-Wei, "Three-dimensional gravity inversion based on 3D U-Net++," *Applied Geophysics*, vol. 18, no. 4, pp. 451–460, 2021.

[28] R. Huang, S. Liu, R. Qi, and Y. Zhang, "Deep learning 3D sparse inversion of gravity data," *Journal of Geophysical Research: Solid Earth*, vol. 126, no. 11, p. e2021JB022476, 2021.

[29] A. Celaya, B. Denel, Y. Sun, M. Araya-Polo, and A. Price, "Inversion of time-lapse surface gravity data for detection of 3-d co2 plumes via deep learning," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 61, pp. 1–11, 2023.

[30] M. Moorkamp, B. Heincke, M. Jegen, A. W. Roberts, and R. W. Hobbs, "A framework for 3-d joint inversion of mt, gravity and seismic refraction data," *Geophysical Journal International*, vol. 184, no. 1, pp. 477–493, 2011.

[31] P. G. Lelièvre, C. G. Farquharson, and C. A. Hurich, "Joint inversion of seismic traveltimes and gravity data on unstructured grids with application to mineral exploration," *Geophysics*, vol. 77, no. 1, pp. K1–K15, 2012.

[32] G. T. Eigestad, H. K. Dahle, B. Hellevang, F. Riis, W. T. Johansen, and E. Øian, "Geological modeling and simulation of co 2 injection in the johansen formation," *Computational Geosciences*, vol. 13, pp. 435–450, 2009.

[33] E. Hoek and J. D. Bray, *Rock slope engineering*. CRC press, 1981.

[34] W. G. Pariseau, *Design analysis in rock mechanics*. CRC Press, 2017.

[35] A. Sundal, J. P. Nystuen, K.-L. Rørvik, H. Dypvik, and P. Aagaard, "The lower jurassic johansen formation, northern north sea – depositional model and reservoir characterization for co_2 storage," *Marine and Petroleum Geology*, vol. 77, pp. 1376–1401, 2016.

[36] G. Eigestad, H. Dahle, B. Hellevang, F. Riis, W. Johansen, and E. Øian, "Geological modeling and simulation of co_2 injection in the johansen formation," *Computational Geosciences*, vol. 13, pp. 435–450, 12 2009.

[37] M. Gelboim, A. Adler, Y. Sun, and M. Araya-Polo, "Encoder–decoder architecture for 3d seismic inversion," *Sensors*, vol. 23, no. 1, p. 61, 2022.

[38] C. H. Sudre, W. Li, T. K. M. Vercauteren, S. Ourselin, and M. J. Cardoso, "Generalised dice overlap as a deep learning loss function for highly unbalanced segmentations," in *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*, pp. 240–248, Springer International Publishing, 2017.

[39] F. Milletari, N. Navab, and S.-A. Ahmadi, "V-net: Fully convolutional neural networks for volumetric medical image segmentation," in *2016 fourth international conference on 3D vision (3DV)*, pp. 565–571, IEEE, 2016.

[40] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[41] I. Loshchilov and F. Hutter, "SGDR: Stochastic gradient descent with warm restarts," in *International Conference on Learning Representations*, 2017.

[42] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Z. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," *CoRR*, vol. abs/1912.01703, 2019.

[43] A. Celaya, J. A. Actor, R. Muthusivarajan, E. Gates, C. Chung, D. Schellingerhout, B. Riviere, and D. Fuentes, "Pocketnet: A smaller neural network for medical image analysis," *IEEE Transactions on Medical Imaging*, vol. 42, no. 4, pp. 1172–1184, 2023.

[44] J. He and J. Xu, "Mgnet: A unified framework of multigrid and convolutional neural network," *Science china mathematics*, vol. 62, pp. 1331–1354, 2019.

[45] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE international conference on computer vision*, pp. 2980–2988, 2017.

[46] D. Alumbaugh, E. Gasperikova, D. Crandall, M. Commer, S. Feng, W. Harbert, Y. Li, Y. Lin, and S. Samarasinghe, "The kimberlina synthetic multiphysics dataset for co2 monitoring investigations," *Geoscience Data Journal*, 2023.