Data-Driven Approach to Inversion Analysis of Three-dimensional Inner Soil Structure via Wave Propagation Analysis

Takuma Yamaguchi[1], Tsuyoshi Ichimura[1,3], Kohei Fujita[1], Muneo Hori[2], Lalith Wijerathne[1], and Naonori Ueda[3]

[1] Earthquake Research Institute and Department of Civil Engineering, The University of Tokyo, Bunkyo, Tokyo, Japan
{yamaguchi, ichimura, fujita, lalith}@eri.u-tokyo.ac.jp
[2] Research Institute for Value-Added-information Generation, Japan Agency for Marine-Earth Science and Technology, Yokohama, Kanagawa, Japan
horimune@jamstec.go.jp
[3] Center for Advanced Intelligence Project, RIKEN, Tokyo, Japan
naonori.ueda@riken.jp

Abstract. Various approaches based on both computational science and data science/machine learning have been proposed with the development of observation systems and network technologies. Computation cost associated with computational science can be reduced by introducing the methods based on data science/machine learning. In the present paper, we focus on a method to estimate inner soil structure via wave propagation analysis. It is regarded as one of the parameter optimization approaches using observation data on the surface. This application is in great demand to ensure better reliability in numerical simulations. Typical optimization requires many forward analyses; thus, massive computation cost is required. We propose an approach to substitute evaluation using neural networks for most cases of forward analyses and to reduce the number of forward analyses. Forward analyses in the proposed method are used for producing the training data for a neural network; thereby they can be computed independently, and the actual elapsed time can be reduced by using a large-scale supercomputer. We demonstrated that the inner soil structure was estimated with the sufficient accuracy for practical damage evaluation. We also confirmed that the proposed method achieved estimating parameters within a shorter timeframe compared to a typical approach based on simulated annealing.

Keywords: Data-driven computing, finite element analysis, conjugate gradient method, GPU computing

1 Introduction

The demand for exploiting the power of Big Data has been increasing due to the rapid development of observation systems and network technologies (e.g., introduction of Internet of Things and 5th Generation networks). Thereafter, integration of computational science, data science, and machine learning has been proposed (Big Data & Extreme Computing (BDEC) [1]). The main purpose of this project is to provide the available data to computer systems and to produce new information with social value by data processing and computations. It is assumed that the data are supplied in real time; therefore, large amount of computer power will be required. Several large-scale computer systems are designed for data science and machine learning (e.g., AI Bridge Cloud Infrastructure (ABCI) [2]), and therefore, we can see that these approaches are being expanded. From the perspective of computational science, computation cost such as power consumption becomes a more important issue with the development of computation environments. Introduction of data science and machine learning can facilitate reducing the computation cost.

We focus on the estimation of the inner structure whose material properties vary by location. This problem has been discussed in various fields including biomedicine [4] and gas and oil exploration [12], and it is also important for damage evaluation in the case of earthquake disasters. Here, we estimate the inner soil structure (boundary surfaces of soil layers) required in ground shaking analysis: in this problem, material properties of each ground layer in the domain are relatively easy to estimate by geological survey. However, direct measurement of the position of boundaries between soil layers with different material properties is difficult. Reference [8] notes that the inner soil structure has significant effects on the distribution of displacement on the ground surface and strain in underground structures. The inner soil structure is not available with high resolution or appropriate accuracy, which deteriorates the accuracy of simulations even if numerical methods capable of modeling complex geometry is used.

One of the realistic ways to address this issue is to introduce an optimization method using the observation data on the ground surface in the case of a small earthquake. If we could generate many models and conduct wave propagation analysis for each of them, it would be possible to select a model capable of reproducing available observation data most accurately. The use of optimized models may help increase reliability of damage evaluation. This procedure requires a large number of forward analyses. Therefore, the challenge is an increase in the computation cost for many analyses with the large number of degrees of freedom.

In our previous study [14], we proposed a simulated annealing method with reduction in computation cost required in each forward analysis by overlapping model generation on CPUs and finite element solver computations on GPUs. We demonstrated that one boundary surface between two layers in the soil structure was estimated by 1,500 wave propagation analyses using finite element models with 3,000,000 degrees of freedom in 13 hours. However, in this application, we have to estimate the three-dimensional inner soil structure using the data on two-dimensional ground surface. In cases when we estimate multiple boundary surfaces, the convergence of optimization methods can deteriorate, as the number of control parameters becomes much larger than the number of observation points. Taking this into account, we note that practical estimation cannot be materialized by just accelerating each forward analysis. Reduction in the number of forward analyses is another important issue to consider.

The approach to introduce data science or machine learning into computational science has been proposed aiming to reduce the number of forward analyses. For instance, [9] applied a machine learning-based methodology for microstructure optimization and materials design. Generally, evaluation using data science or machine learning is faster than forward analysis; therefore, more trials can be evaluated within the same timeframe.

We examine the applicability of the data-driven approach to estimation of the inner soil structure. To address this problem, we combine the wave propagation analysis and neural network methodology. We conduct many forward analyses and use these results as the training data for a neural network. By implementing the neural network that takes control parameters as input and outputs error levels of models, we can extract the parameters that are expected to reproduce observation data more accurately from many samples. Computation cost is reduced by replacing forward analysis with inference using neural networks. In addition, forward analyses can be computed independently; therefore, we can reduce the actual elapsed time by using large-scale supercomputers. In the application example, we demonstrate that the inner soil structure has a considerable effect on the strain distribution, and that the proposed method can estimate the soil structure with the sufficient accuracy for damage estimation within a shorter timeframe compared with a typical approach based on simulated annealing.

2 Methodology

In the present paper, we propose a method that estimates the inner soil structure of the target domain by conducting a large number of wave propagation analyses and choosing the inner structure with the maximum likelihood. Here, we estimate boundary surfaces of the domains with different material properties. For simplicity, we assume that the target domain has a stratified structure, and that the target parameters considered for optimization are elevations of the boundary surfaces on control points which are located at regular intervals in x and y directions. The set of control parameters is denoted by $\boldsymbol{\alpha}$. Boundary surfaces are generated in the target domain by interpolating elevations at the control points using bi-cubic functions. We employ the time history of waves on the ground surface $\mathbf{v}^{\mathrm{ref}}_{i_{\mathrm{obs}}} (i_{\mathrm{obs}} = 1, ..., n_{\mathrm{obs}})$, where n_{obs} is the number of observation points. In addition, we assume that these waves do not contain noise. We define an error between a generated and a reference model as follows:

$$E = \sqrt{\frac{\sum_{i_{\mathrm{obs}}=1}^{n_{\mathrm{obs}}} \sum_{i_{\mathrm{t}}=1}^{n_{\mathrm{t}}} \|\mathbf{v}_{i_{\mathrm{obs}},i_{\mathrm{t}}} - \mathbf{v}^{\mathrm{ref}}_{i_{\mathrm{obs}},i_{\mathrm{t}}}\|^2}{\sum_{i_{\mathrm{obs}}=1}^{n_{\mathrm{obs}}} \sum_{i_{\mathrm{t}}=1}^{n_{\mathrm{t}}} \|\mathbf{v}^{\mathrm{ref}}_{i_{\mathrm{obs}},i_{\mathrm{t}}}\|^2}}, \tag{1}$$

where i_{t} is each time step, and $\mathbf{v}_{i_{\mathrm{obs}},i_{\mathrm{t}}}$ is the velocity on i_{obs}-th observation point at the i_{t}-th time step, which can be computed from the parameters $\boldsymbol{\alpha}$ and the known input waves obtained by pullback analysis. We assume that the models that reproduce the observation data closely have smaller error E. Therefore, we search $\boldsymbol{\alpha}$ that minimizes E. There are several gradient-based methods applied to optimization, for example, three-dimensional crustal structure optimization proposed in [11]. These methods have the advantage that the number of trials is small; however, it may be difficult to escape from a local solution if control parameters have a low sensitivity to the error function. Simulated annealing [7] is one of the most robust approaches; however, the convergence rate is not high for problems with many control parameters. If we obtain $\mathbf{v}_{i_{\mathrm{obs}},i_{\mathrm{t}}}$ by forward analysis for each parameter, a large number of forward analyses are required. Reduction in the number of cases to compute and introduction of parallel computations of forward analyses are essential to conduct the target estimation within a realistic timeframe. Accordingly, we introduce an approach based on machine learning employing the results of forward analyses. We define the neural networks that can be used to estimate the error E based on the input parameters $\boldsymbol{\alpha}$. The details about the neural networks and forward analysis are described in the following subsections.

2.1 Introduction of neural networks

The proposed algorithm based on neural networks is described as Algorithm 1. Firstly, we conduct n_0 forward analyses to generate the adequate training data (Algorithm 1, lines 3-4). We have to use the parameter sets that are scattered in the control parameter space. In the present study, a random number retrieved from the normal distribution is added up to initial elevation of one control point. We fluctuate elevations under the constraint that the order of layers is consistent in all of the models. We obtain errors for all parameters by performing forward analyses with generated models. Each case can be computed independently; therefore, elapsed time is reduced when we use a large-scale computer system.

Next, we implement neural networks to estimate error E roughly (Algorithm 1, line 5). Here, input parameters of neural networks consist of the fluctuation amount of elevation at each control point. To improve the performance, we normalize these values, so that the value of -3σ is set equal to 0, and the value of 3σ is set equal to 1. Here, we generate the classifiers instead of regression models, as suggested by [6], due to the fact that the number of samples constrained by the massive computation cost is limited for modeling complex modes of the error function. We classify errors into 10 levels that are equally divided between the maximum and minimum errors in n_0 sets of parameters. We define one classifier for each observation point so that contribution of each point to the error becomes clearer. We define the point-wise error based on the original error in Eq. (1) as follows:

$$E_{i_{\mathrm{obs}}} = \sqrt{\frac{\sum_{i_{\mathrm{t}}=1}^{n_{\mathrm{t}}} \|\mathbf{v}_{i_{\mathrm{obs}},i_{\mathrm{t}}} - \mathbf{v}^{\mathrm{ref}}_{i_{\mathrm{obs}},i_{\mathrm{t}}}\|^2}{\sum_{i_{\mathrm{t}}=1}^{n_{\mathrm{t}}} \|\mathbf{v}^{\mathrm{ref}}_{i_{\mathrm{obs}},i_{\mathrm{t}}}\|^2}}. \tag{2}$$

We divide the range of the possible values of $E_{i_{\mathrm{obs}}}$ into ten levels $(1-10)$ equally, and the implemented neural networks learn the classification of these levels. A final evaluation value for each parameter set is obtained by summation across levels in all neural networks. We assume that the parameters with smaller evaluation values provide smaller E. We have to define a neural network per observation point in this procedure; however, associated computation cost is insignificant, as each network can be trained independently, and the number of the training data elements is at most 10^3.

We perform inference using the generated neural networks. A dataset consisting of $n_1(\gg n_0)$ cases is inputted into the neural networks (Algorithm 1, lines 6-7). We extract n_0 cases that have the lowest estimated error levels and compute actual errors E by using forward analyses. Parameters that achieves the smallest E are chosen in Algorithm 1, lines 8-10. Further improvement can be achieved by iterating the same procedures using the estimated parameters as required (Algorithm 1, line 11).

Algorithm 1: Algorithm based on neural networks for estimation of the inner structure.

```
1  Data: initial parameters $\boldsymbol{\alpha}_{\mathrm{base}}$, observation data $\mathbf{v}^{\mathrm{ref}}_{i_{\mathrm{obs}}}$ ($i_{\mathrm{obs}} = 1, ..., n_{\mathrm{obs}}$),
         counter $i_{\mathrm{trial}} = 1$
2  for $i_{\mathrm{trial}} \leq MaxTrialNumber$ do
3      generate $n_0$ samples based on $\boldsymbol{\alpha}_{\mathrm{base}}$ to produce $TrainingDataset_{i_{\mathrm{trial}}}$
4      for each $\boldsymbol{\alpha}$ in $TrainingDataset_{i_{\mathrm{trial}}}$ do
           compute actual error $E_{i_{\mathrm{obs}}}$
5      for each observation point $i_{\mathrm{obs}}$ do
           produce classifier of $E_{i_{\mathrm{obs}}}$ for given $\boldsymbol{\alpha}$
6      generate $n_1(\gg n_0)$ samples based on $\boldsymbol{\alpha}_{\mathrm{base}}$ to produce
        $InferenceDataset_{i_{\mathrm{trial}}}$
7      for each $\boldsymbol{\alpha}$ in $InferenceDataset_{i_{\mathrm{trial}}}$ do
           estimate $E_{i_{\mathrm{obs}}}$ using classifiers
8      choose $n_0$ samples from $InferenceDataset_{i_{\mathrm{trial}}}$ which provides the
        lowest $E$ to construct $SuperiorDataset_{i_{\mathrm{trial}}}$
9      for each $\boldsymbol{\alpha}$ in $SuperiorDataset_{i_{\mathrm{trial}}}$ do
           compute actual error $E$
10     choose $\boldsymbol{\alpha}_{\mathrm{best}}$, which provides the lowest $E$
11     update $\boldsymbol{\alpha}_{\mathrm{base}} \leftarrow \boldsymbol{\alpha}_{\mathrm{best}}$, $i_{\mathrm{trial}} \leftarrow i_{\mathrm{trial}} + 1$
```

2.2 Finite element analyses

In the proposed scheme, more than 10^3 finite element analyses are required; therefore, it is important to conduct them in a shorter time possible. We assume that we use the observation data in the case of an earthquake small enough to ignore nonlinearity for estimation of the inner structure. Therefore, we focus on linear wave propagation analysis. The target equation is defined as follows: $\left(\frac{4}{dt^2}\mathbf{M} + \frac{2}{dt}\mathbf{C} + \mathbf{K}\right)\mathbf{u}_{i_{\mathrm{t}}} = \mathbf{f}_{i_{\mathrm{t}}} + \mathbf{C}\mathbf{v}_{i_{\mathrm{t}}-1} + \mathbf{M}\left(\mathbf{a}_{i_{\mathrm{t}}-1} + \frac{4}{dt}\mathbf{v}_{i_{\mathrm{t}}-1}\right)$, where $\mathbf{u}$, $\mathbf{v}$, $\mathbf{a}$, and $\mathbf{f}$ are displacement, velocity, acceleration, and force vector, respectively; and $\mathbf{M}$, $\mathbf{C}$, and $\mathbf{K}$ are mass, damping, and stiffness matrix, respectively. In addition, dt denotes the time increment, and i_{t} is the number of time steps. For the damping matrix $\mathbf{C}$, we apply Rayleigh damping and compute it by linear combination as follows: $\mathbf{C} = \alpha\mathbf{M} + \beta\mathbf{K}$. Coefficients α and β are set so that $\int_{f_{\mathrm{min}}}^{f_{\mathrm{max}}}(h - \frac{1}{2}(\frac{\alpha}{2\pi f} + 2\pi f\beta))^2 df$ is minimized, where f_{max}, f_{min}, and h

are maximum/minimum targeting frequency and damping ratio. We apply the Newmark-β method with $\beta = 1/4$ and $\delta = 1/2$ for time integration. Vectors $\mathbf{v}_{i_t}$ and $\mathbf{a}_{i_t}$ can be described as follows: $\mathbf{v}_{i_t} = -\mathbf{v}_{i_t-1} + \frac{2}{dt}(\mathbf{u}_{i_t} - \mathbf{u}_{i_t-1})$ and $\mathbf{a}_{i_t} = -\mathbf{a}_{i_t-1} - \frac{4}{dt}\mathbf{v}_{i_t-1} + \frac{4}{dt^2}(\mathbf{u}_{i_t} - \mathbf{u}_{i_t-1})$. We obtain displacement vector $\mathbf{u}_{i_t}$ by solving the linear equation and updating vectors $\mathbf{v}_{i_t}$ and $\mathbf{a}_{i_t}$ accordingly. Generation of finite element models and computation of the finite element solver are the most computationally expensive parts. We automatically generate finite element models using the method proposed by [5]. This method applies CPU computations using OpenMP. In the solver part, we apply the OpenACC-accelerated solver based on the conjugate gradient method described in our previous study [13]. The solver combines the conjugate gradient method with adaptive preconditioning, geometric multigrid method, and mixed precision arithmetic to reduce the amount of arithmetic counts and the data transfer size. In the solver, sparse matrix vector multiplication is computed by using the Element-by-Element method. It is applied to compute the element matrix on-the-fly and allows reducing the memory access cost. Specifically, the multiplication of matrix $\mathbf{A}$ and vector $\mathbf{x}$ is defined as $\mathbf{y} = \sum_{i=1}^{n_{\mathrm{elem}}}(\mathbf{Q}^{(i)T}(\mathbf{A}^{(i)}(\mathbf{Q}^{(i)}\mathbf{x})))$, where $\mathbf{y}$ is the resulting vector, and n_{elem} is the number of elements in the domain; $\mathbf{Q}^{(i)}$ is a mapping matrix to make a transition from the local node numbers in the i-th element to the global node numbers; and $\mathbf{A}^{(i)}$ is the i-th element matrix that satisfies the following: $\mathbf{A} = \sum_{i=1}^{n_{\mathrm{elem}}} \mathbf{Q}^{(i)T}\mathbf{A}^{(i)}\mathbf{Q}^{(i)}$, where $\mathbf{A}^{(i)} = \frac{4}{dt^2}\mathbf{M}^{(i)} + \frac{2}{dt}\mathbf{C}^{(i)} + \mathbf{K}^{(i)}$. The solver proposed by [13] originally includes the procedure to extract parts with bad convergence to be extensively solved in the preconditioning part; however, we skip it for simplicity. In addition, only single and double precision numbers are used, and the custom data type FP21 is not used in the computations.

3 Application Example

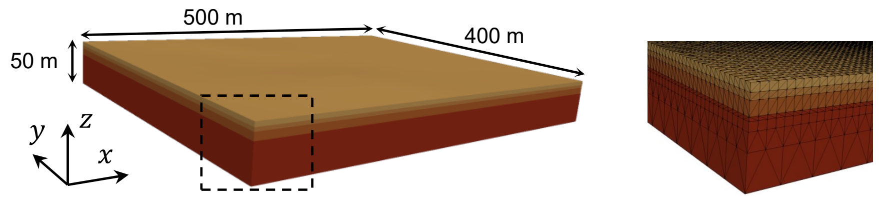
3.1 Definition of the considered problem

We apply the developed method to estimate the soil structure, aiming to verify its efficiency. The supercomputer ABCI [2], operated by the National Institute of Advanced Industrial Science and Technology, is used for each forward analysis. Each compute node of ABCI has four NVIDIA Tesla V100 GPUs and two Intel Xeon Gold 6148 CPUs (20 cores). The GPUs in each compute node are connected via NVLink, with a bandwidth of 50 GB/s in each direction. We conduct each forward analysis using one compute node of ABCI. We assign one GPU for the finite element solver using MPI and use all CPU cores for generating models with OpenMP. Moreover, a GPU cluster composed of the IBM Power System AC922, which has four NVIDIA Tesla V100 GPUs and two IBM Power9 CPUs (16 cores) per compute node, is used for learning and inference of neural networks.

The target domain has four layers, and we define their boundary surfaces. In the case of this problem, material properties of the soil structure are deterministic. These properties are described in Table 1. The target domain is of the following size: $0\,\mathrm{m} \leq x \leq 500\,\mathrm{m}$, $0\,\mathrm{m} \leq y \leq 400\,\mathrm{m}$, and $-50\,\mathrm{m} \leq z \leq 1.62\,\mathrm{m}$. A

Table 1: Material properties of the target domain. V_p, V_s, and ρ are primary and secondary wave velocity, and density, respectively. h is the damping ratio.

	V_p(m/s)	V_s(m/s)	ρ (kg/m^3)	h
1st layer	1500	130	1700	0.02
2nd layer	1600	220	1800	0.02
3rd layer	1600	160	1700	0.02
4th layer	2000	400	2000	0.001



(a) Entire model **(b)** Close-up view

Fig. 1: One of finite element models in the analysis.

finite element model with approximately 1,400,000 degrees of freedom is generated. Figure 1 represents one of the finite element models used in the analysis. The resolution was 5 m at maximum.

Twelve observation points are located at points $(x, y) = (100i, 100j)$ (i=1-4, j=1-3) on the ground surface. We assume that the observation data at these points are available without noise. In fact, we target the low frequency band up to 2.5 Hz; therefore, the influence of noise is small. The control points are located at regular intervals in x and y directions. We denote the elevation of the k-th layer on points $(x, y) = (100i, 100j)$ (i=0-5, j=0-4) as α_{ij}^k(m). The points $x = 0$, $x = 500$, $y = 0$, and $y = 400$ are the edges of the domain, and we assume that α_{ij}^k are constant among these points for each layer. Boundary surfaces with reference parameters are represented in Fig. 2. We input the observed small earthquake wave for 80 seconds to the bottom surface of our target model. Within the current problem settings, the target frequency is as much as 2.5 Hz, so we apply a band-pass filter to remove unnecessary frequency bands. Time increment used in the analysis is 0.01 seconds; therefore, each wave propagation analysis comprises 8,000 times steps.

We perform testing on the two cases of the control parameters. The details are provided in Table 2. In case 1, we only estimate the elevation of the 4th layer. The elevations of other layers are known. In case 2, we estimate the elevations of the 2nd, 3rd, and 4th layers. The elevation of the 1st layer, which is the ground surface, is known. It should be noted that in case 2, there are more control parameters.

3.2 Estimation by a typical approach

First, we apply the approach proposed by [14] for parameter estimation, which is based on very fast simulated annealing [7]. Each control parameter is changed

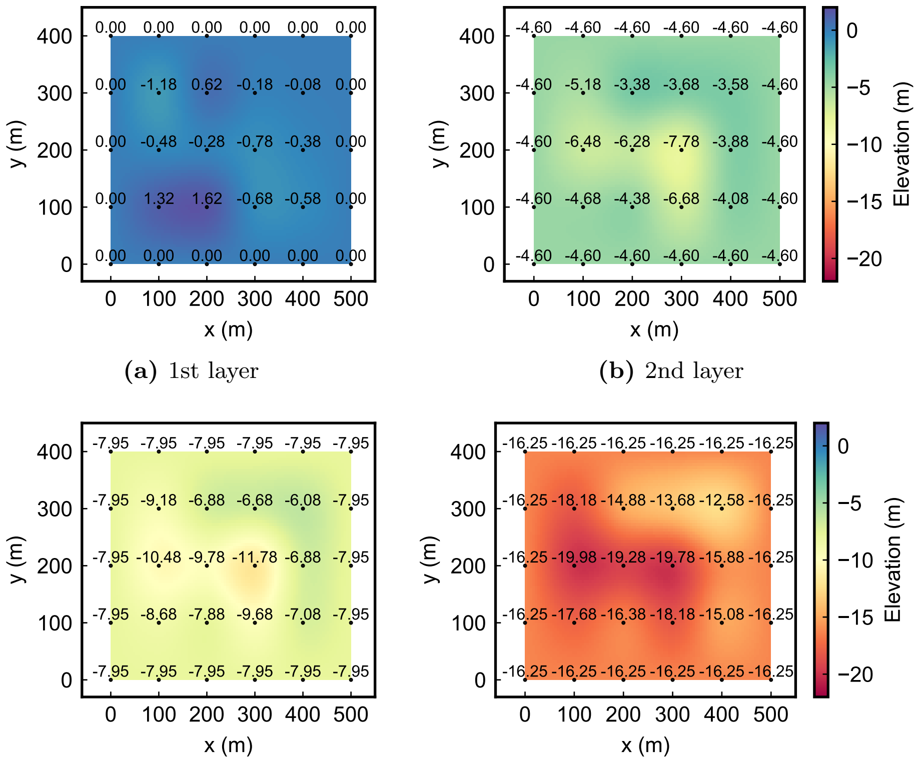
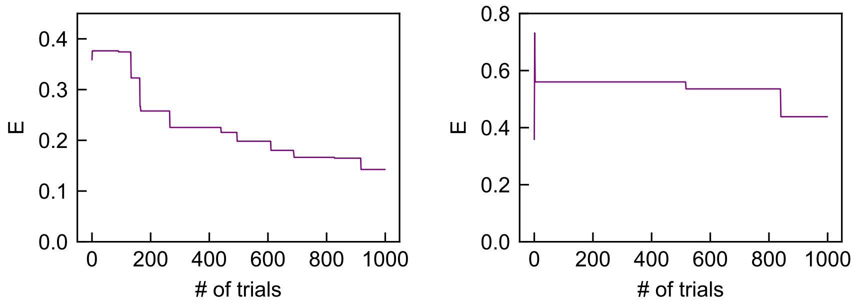


Fig. 2: Distribution of actual elevation (m) of each layer. The value of control parameters α_{ij}^{k} are also described.

Table 2: Problem settings for the control points.

	control parameters	# of parameters
case 1	$\alpha_{ij}^{k}(i\text{=1-4, }j\text{=1-3, }k\text{=4})$	12
case 2	$\alpha_{ij}^{k}(i\text{=1-4, }j\text{=1-3, }k\text{=2-4})$	36

by multiplication of the difference between the lower and the upper limit values of the target parameter and the ratio r. r is computed as follows: $sgn(u - 0.5)T_i\left[(1+1/T_i)^{|2u-1|}-1\right]$, where u is extracted from the uniform distribution between [0.0, 1.0], and T_i is a temperature parameter in simulated annealing. The temperature at i-th trial is defined using initial temperature T_0 and the number of control points D as follows: $T_i = T_0\exp(-ci^{\frac{1}{D}})$, where parameter c is defined by T_0, D, lowest temperature T_f, and the number of trials i_f as $T_f = T_0\exp(-m)$, $i_f = \exp n$, and $c = m\exp(-\frac{n}{D})$. Here, $D = 12$ in case 1 and $D = 36$ in case 2. We set the number of trials $i_f = 1000$; and $c = 8.18$ in case 1 and $c = 12.00$ in case 2, respectively. These parameters satisfy the requirement that the parameters that increase the value of the error function by ΔE are adopted with the probability of 80% at the initial temperature, where ΔE is



(a) case 1 **(b)** case 2

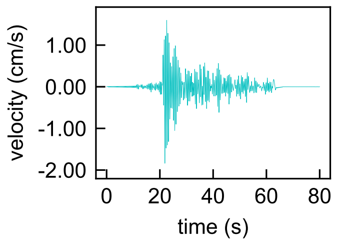
Fig. 3: History of the error E calculated by Eq. (1) in simulated annealing.

the value of the error function obtained in the initial model. In addition, the parameters which increase the value of the error function by $\Delta E \times 10^{-5}$ are adopted with the probability of 0.1% at the lowest temperature. These settings for c, T_0, and T_f are the same as those in [14].

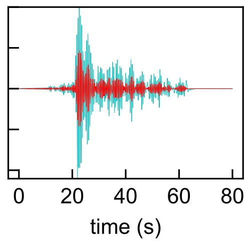
The histories of the error value for both cases are presented in Fig. 3. In case 1, the parameters are updated at 100-trial intervals, on average. In contrast, the parameters are rarely updated in case 2. The final value of E is 39.7% of the initial parameters in case 1 and 122.2% in case 2. Fig. 4 describes the time history of velocity on point $(x, y, z) = (400, 300, -0.08)$ when conducting wave propagation analysis with both estimated parameters. Analyzing these results, for case 1, we can see that the obtained wave has come close to that of the reference model to some extent. For case 2, the result obtained using the estimated model completely differs from that of the reference model. We assume that a much larger number of trials would be required for case 2.

3.3 Estimation by the proposed method

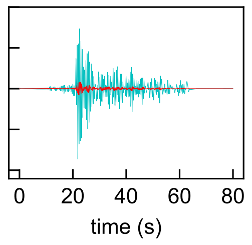
Next, we apply the proposed method for case 2, which is presumed to be more difficult. We introduce neural networks for classification of error $E_{i_{\mathrm{obs}}}$ in Eq. (2) according to the problem settings described above. We add random numbers following the normal distribution with $3\sigma = 5\,\mathrm{m}$ to the control parameters α_{ij}^k. We generate $n_0 = 1000$ parameter sets and conduct forward analysis for each case. Here, we assign 800 cases for the training data and 200 cases for the test data. We produce twelve neural networks for each of the twelve observation points in the target domain. We employ Pytorch [10] to develop neural networks. Learning and inference are accelerated by GPUs. We consider fully connected layers for the neural networks. The number of units, the number of layers, and dropout rate, which are representative hyperparameters in neural networks, are optimized using optuna [3]. This framework searches hyperparameters that minimizes the summation of $\frac{|L^{\mathrm{ref}} - L|}{L^{\mathrm{ref}}}$ in the test data, where L is the level of error judged by a neural network ($1 \leq L \leq 10$), and L^{ref} is the actual level of error derived by forward analysis ($1 \leq L^{\mathrm{ref}} \leq 10$). We use this index aiming to obtain more accurate classification for lower levels of error. The number of units, the number of



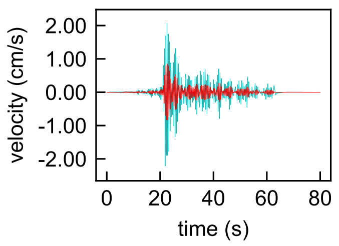
(a) reference model (cases 1, 2)



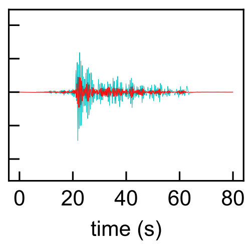
(b) initial model (case 1)



(c) estimated model (case 1)



(d) initial model (case 2)



(e) estimated model (case 2)

Fig. 4: x component of velocity at $(x, y) = (400, 300)$ on the ground surface in the analysis (blue lines). Differences with the results of the reference model are also represented in (b) to (e) as red lines.

layers, and dropout rate are fluctuated within the range of 6-72, 3-8, and 0.1-0.9, respectively. The number of trials is set to 50. We use a rectified linear unit for the activation function. In addition, we use Adam as the optimizer. Here, the batch size is set to 300, and the number of epochs is set to 1,000.

Optimized neural networks output correct levels with the probability of 71% and levels with an error of no more than plus/minus one level with the probability of 99% on average. We infer the error for n_1 =2,000,000 parameter sets using the generated neural networks and extract n_0 =1,000 cases that have lower estimated error levels. Next, we compute the actual error for the extracted parameter sets ($SuperiorDataset_1$ in Algorithm 1) by forward analyses. As a result, error E is generally reduced compared to the original dataset ($TrainingDataset_1$ in Algorithm 1), as represented in Fig. 5. This figure also includes the result obtained by repeating the same procedure once again ($SuperiorDataset_2$). The error distribution is shifted to the left side of the previous distribution. Analyzing this figure, we can confirm that the approach based on neural network is effective for extracting parameters that provide smaller errors. By using neural networks twice, the error E is reduced to 21% with respect to that of the initial parameters, as shown in Table 3.

For confirmation of the accuracy of the estimated model, we conduct wave propagation analysis using the estimated parameters. Figure 6 describes the time history of velocity on point $(x, y, z) = (400, 300, -0.08)$, and Fig. 7 represents the distribution of displacement on the ground surface at time $t = 30.00$ s. We

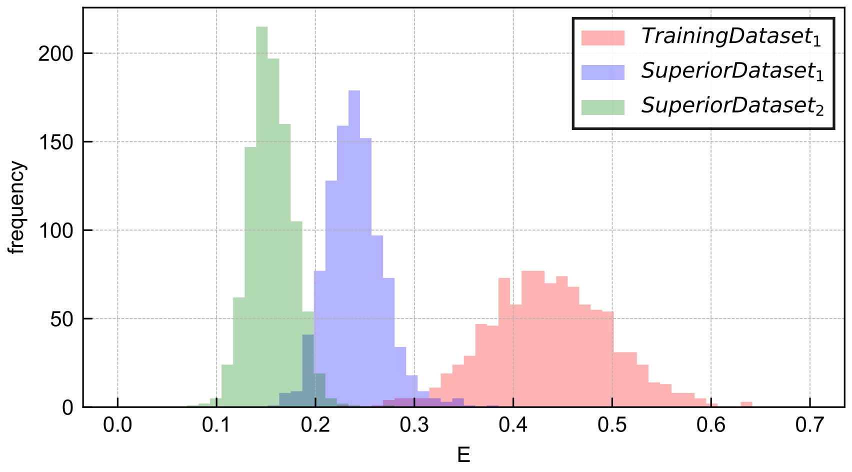


Fig. 5: Histogram of the error E for initial dataset ($TrainingDataset_1$ in Algorithm 1) and datasets extracted by using neural networks ($SuperiorDataset_1$, $SuperiorDataset_2$). Each dataset includes 1,000 sets of parameters.

Table 3: Error calculated by Eq. (1) in the proposed method. Results when generating AI once ($i_{\text{trial}} = 1$ in Algorithm 1) and twice ($i_{\text{trial}} = 2$) are listed.

	initial parameters	estimated parameters	
		$i_{\text{trial}} = 1$	$i_{\text{trial}} = 2$
E	0.358594	0.165238	0.077423

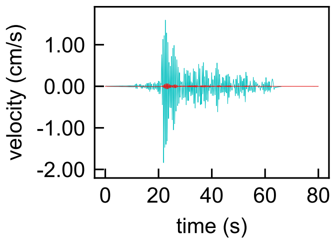
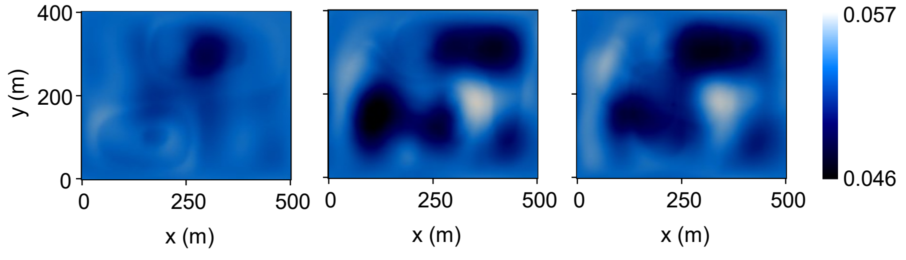


Fig. 6: x component of velocity at $(x, y) = (400, 300)$ on the ground surface using the model estimated by the proposed method (blue line). Differences with the results of the reference model are also represented by the red line.

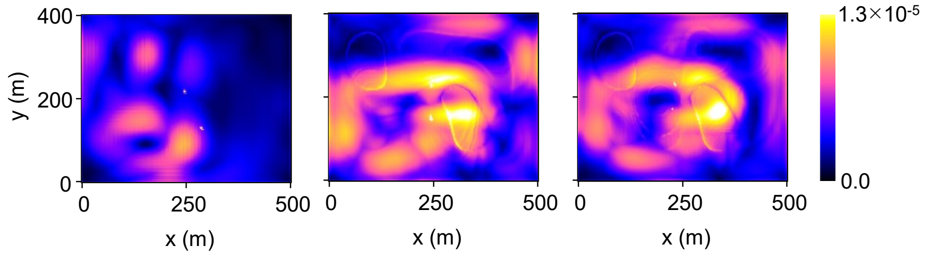
confirm that the results obtained by the estimated model and reference model are sufficiently consistent.

Finally, we computed the distribution of maximum principal strain, which can be utilized for screening of underground structures which might be damaged. Reference [8] notes that strain should be taken into consideration rather than velocity for damage estimation of buried pipelines, which is a typical type of underground structure. Figure 8 represents the strain distribution on the surface at $z = -1.2$ m. We confirm that the distribution of strain is estimated with sufficient accuracy by our proposed method. These strain distributions obtained



(a) initial model **(b)** reference model **(c)** estimated model

Fig. 7: Norm distribution of displacement (cm) on the ground surface at $t = 30.00s$ in the analysis.



(a) initial model **(b)** reference model **(c)** estimated model

Fig. 8: Norm distribution of max principal strain on the plane $z = -1.2$ in the analysis.

by using the estimated model and those obtained by the initial model are considerably different, including areas with high strain. The obtained result shows that the proposed estimation is important to assure the reliability of evaluation.

3.4 Evaluation of computation cost

In this study, we evaluate the computation cost for case 2. In the approach based on simulated annealing, we computed 1,000 forward analyses sequentially. Each trial took twelve min to complete; and therefore, the total elapsed time was 12 min $\times$ 1,000 cases = 12,000 min $\simeq$ 8 days. In fact, the greater number of trials would be necessary, as the value of error remained high. In the proposed method, we set $MaxTrialNumber$ in Algorithm 1 to be 2 and iterated the generation of a neural network with 1,000 forward analyses and computation of top 1,000 cases extracted by neural networks twice. In total, we constructed neural networks twice and conducted 4,000 forward analyses. It required 30 min to define each neural network; therefore, one hour was required for training the neural networks in total. We used the supercomputer ABCI to perform wave propagation analysis. About 200 nodes were simultaneously available on average, although the number of available compute nodes depended on the utilization of the system. As we computed 200 cases in parallel, the elapsed time for forward analyses was 12 min $\times$ (4,000 cases / 200 cases) = 240 min = 4 hours. Computation cost in other parts was negligible; therefore, the total elapsed time was approximately

1 hours + 4 hours = 5 hours. Although the number of the computed cases was larger than that of simulated annealing, the actual elapsed time was reduced owing to parallel computations. As a result of this comparison, we confirmed the effectiveness of the proposed method.

4 Conclusion

Introduction of data science and machine learning is one of the effective approaches to reduce the computation cost associated with computational science. In the present paper, we focus on estimation of the inner soil structure via wave propagation analysis. This estimation of the inner structure is important to improve reliability in numerical simulations. However, massive computation cost is required, as typical optimization requires many forward analyses.

We applied the data-driven approach to this problem aiming to reduce the computation cost. The proposed method combined neural networks and wave propagation analysis. We generated the training data by executing many forward analyses on a large-scale supercomputer. We implemented a neural network that took the parameters of inner structures as input and outputted error levels based on the observation data. Applying the neural network, we extracted the parameter sets expected to reproduce the observation data closely. Computation cost required in inference was negligible; therefore, many cases could be roughly evaluated in a shorter time.

In the application example, we estimated the soil structure using the observation data with a total of 4,000 wave propagation analyses. We confirmed that the estimated model had the sufficiently consistent results compared with the reference model via evaluation of a strain distribution. Each forward analysis required in the proposed method was computed in parallel; and thereby the actual elapsed time was reduced by using a large-scale supercomputer. We confirmed the effectiveness of the proposed method via performance comparison with the typical approach based on very fast simulated annealing. It is future task to examine validity of our proposed method for more complex models.

The demand for utilization of Big Data will continue to increase further, and the development of computation environments, observation systems, and network technologies will follow accordingly. In the present paper, we outlined the importance of developing an algorithm that enables capacity computing and maximizes utilization of computer resources for processing the large quantity of data and obtaining new information within a realistic timeframe.

Acknowledgments

Our results were obtained using computational resource of the AI Bridging Cloud Infrastructure (ABCI) at the National Institute of Advanced Industrial Science and Technology. We acknowledge support from Japan Society for the Promotion of Science (18H05239 and 18K18873).

References

1. Big Data and Extreme-scale Computing, `https://www.exascale.org/bdec/`

2. Computing Resources of AI bridging Clound Infrastructure, https://abci.ai/en/about_abci/computing_resource.html
3. Akiba, T., Sano, S., Yanase, T., Ohta, T., Koyama, M.: Optuna: A Next-Generation Hyperparameter Optimization Framework. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. p. 2623–2631. KDD 19, Association for Computing Machinery, New York, NY, USA (2019). https://doi.org/10.1145/3292500.3330701, https://doi.org/10.1145/3292500.3330701
4. Clement, R., Schneider, J., Brambs, H.J., Wunderlich, A., Geiger, M., Sander, F.G.: Quasi-automatic 3D finite element model generation for individual single-rooted teeth and periodontal ligament. Computer methods and programs in biomedicine **73**(2), 135–144 (2004)
5. Ichimura, T., Agata, R., Hori, T., Hirahara, K., Hashimoto, C., Hori, M., Fukahata, Y.: An elastic/viscoelastic finite element analysis method for crustal deformation using a 3-d island-scale high-fidelity model. Geophysical Journal International **206**(1), 114–129 (2016)
6. Ichimura, T., Fujita, K., Yamaguchi, T., Hori, M., Lalith, M., Ueda, N.: Fast Multi-Step Optimization with Deep Learning for Data-Centric Supercomputing. In: 4th International Conference on High Performance Compilation, Computing and Communications (accepted) (2020)
7. Ingber, L.: Very fast simulated re-annealing. Mathematical and computer modelling **12**(8), 967–973 (1989)
8. Liang, J., Sun, S.: Site effects on seismic behavior of pipelines: a review. Journal of pressure vessel technology **122**(4), 469–475 (2000)
9. Liu, R., Kumar, A., Chen, Z., Agrawal, A., Sundararaghavan, V., Choudhary, A.: A predictive machine learning approach for microstructure optimization and materials design. Scientific reports **5**(1), 1–12 (2015)
10. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: PyTorch: An Imperative Style, High-Performance Deep Learning Library. In: Wallach, H., Larochelle, H., Beygelzimer, A., dAlché Buc, F., Fox, E., Garnett, R. (eds.) Advances in Neural Information Processing Systems 32, pp. 8024–8035. Curran Associates, Inc. (2019), http://papers.nips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf
11. Quinay, P.E.B., Ichimura, T., Hori, M.: Waveform Inversion for Modeling Three-Dimensional Crust Structure with Topographic Effects. Bulletin of the Seismological Society of America **102**(3), 1018–1029 (2012)
12. Warner, M., Ratcliffe, A., Nangoo, T., Morgan, J., Umpleby, A., Shah, N., Vinje, V., Štekl, I., Guasch, L., Win, C., et al.: Anisotropic 3D full-waveform inversion. Geophysics **78**(2), R59–R80 (2013)
13. Yamaguchi, T., Fujita, K., Ichimura, T., Naruse, A., Lalith, M., Hori, M.: GPU implementation of a sophisticated implicit low-order finite element solver with FP21-32-64 computation using OpenACC. Sixth Workshop on Accelerator Programming Using Directives (2019), https://waccpd.org/program/
14. Yamaguchi, T., Ichimura, T., Fujita, K., Hori, M., Wijerathne, L.: Heuristic Optimization with CPU-GPU Heterogeneous Wave Computing for Estimating Three-Dimensional Inner Structure. In: International Conference on Computational Science. pp. 389–401. Springer (2019)