# Real-time Ambient Noise Subsurface Imaging in Distributed Sensor Networks

Maria Valero*, Goutham Kamath†, Jose Clemente*, Yao Xie‡, Fan-Chi Lin§ and WenZhan Song*
*College of Engineering, University of Georgia
†Computer Science Department, Georgia State University
‡H. Milton Stewart School of Industrial and Systems Engineering, Georgia Institute of Technology
§Department of Geology and Geophysics, University of Utah
Email: {maria.valero,jclementes,wsong}@uga.edu, gkamath1@student.gsu.edu, FanChi.Lin@utah.edu, yao.xie@isye.gatech.edu

***Abstract*—Ambient Noise Seismic Imaging (ANSI) is a recently developed geophysical methodology to image the shallow subsurface structures of earth using ambient/environment noise as the source. Integrating ANSI computing within distributed sensor networks will enable real-time continuous monitoring of subsurface dynamics for sustainability studies. However, the research challenges associated with this innovative approach are significant. Traditional data collection using sensor networks imposes practical difficulty for real-time applications, because of the sheer amount of data and large-dense sensor arrays versus limited network bandwidth. This paper is the first to investigate how to utilize the computing capabilities of sensor nodes to perform the computation of ANSI under resource constraints. We explored two distributed approaches (aggregation and consensus) for computing ambient noise eikonal tomography and obtaining phase velocity maps. We performed experiments using CORE emulator to obtain phase velocity maps on real data from USArray Transportable Array. Results demonstrate that our approaches can illuminate phase velocities under network constraints. We also show that the proposed aggregation and consensus algorithms not only balance the computation load but also achieve low communication cost and high data loss tolerance.**

***Index Terms*—subsurface imaging, ambient noise, eikonal tomography, sensor networks, distributed computation.**

## I. Introduction

Subsurface Imaging is a technique widely used in geophysical exploration for investigating structures under the surface of earth. Understanding and addressing the subsurface sustainability has a significant impact on natural, social, and economic issues across the globe. Real-time imaging of shallow underground structures is essential to assess the sustainability and potential hazards of geological structures [1]. The recent landslide tragedy in Washington state[1] is yet another example showing that understanding the subsurface sustainability is crucial to public safety.

To fully utilize the dense seismic array when there are few earthquakes or other obvious events, a new approach called Ambient Noise Seismic Imaging (ANSI) [2]–[6] has been developed. ANSI uses vibrations from random sources in the earth to first estimate the Green functions between pairs of stations and then invert them for obtaining 3D earth structures. By cross-correlating continuous seismic background noise recorded by two sensors, we can extract the seismic surface waves between them and study earth properties. One advantage of this approach is that it can be used to obtain superficial structures based on persistent background noise (ambient noise/eikonal tomography) instead of earthquake events (traveltime/inversion tomography). Moreover, recently, there is a growing interest in applying ambient noise tomography to a large N array (more than 1000 stations) to resolve the top few kilometers of the earth structure with high-resolution[3]. For such a study, eikonal tomography based on waveform tracking can be used as ANSI technique to efficiently determine 2D surface wave phase velocity maps at different periods that represent shallow structures of earth.

The problem is that the existing ambient noise tomography uses post-processing approaches to recover subsurface structures, and they do not yet have the capability of obtaining information in real-time. Current approaches involve manual collection of raw seismic data from the sensors to a central server for post processing and analysis. This data collection problem cannot be solved using simple data transmission in sensor networks because of the sheer amount of data and large-dense sensor arrays versus limited network bandwidth and reliability. Also, it introduces a bottleneck in computation and increases the risk of data loss in case of node failures, especially near the central station. To address these challenges, this paper investigates how to utilize the computing capabilities of sensor nodes to perform in-situ ANSI computation under resource constraints.

Computing in networks is also called fog computing, which is an emerging computing paradigm where the data processing, networking, storage and analytics are performed closer to the devices (IoT) and applications. We have pioneered the development of such computing methodology for seismic imaging applications [7]–[11]. Fog computing paradigm establishes the computation directly at the edge of the network. Here, decentralized devices communicate and potentially cooperate among themselves delivering a new range of applications and services like connected cars, smart grids, smart traffic lights, mobile computing systems and many sensor networks

[1]http://www.dnr.wa.gov/programs-and-services/geology/geologic-hazards/landslides#resources

applications.

The main contribution of this paper is to develop two decentralized approaches, aggregation and consensus, for computing tomography of ambient noise over a network of sensors using computing capabilities that allow sensors to analyze, compute and share their own results. We compared both approaches and identified pros and cons of each one of them from the eikonal tomography point of view. For aggregation approach, we used a multi-tree solution, and we followed the specifications described in [12] and [13]. For consensus approach, we obtained the complete phase velocity map by a general agreement between all sensors that have partial maps (computed individually) following studies presented in [14] and [15].

We performed experiments using CORE[2] emulator and used real data from EarthScope Transportable Array (USArray)[3], which records seismic data across the United States. Since transmission range between nodes is important for communication, we showed computing eikonal tomography and obtaining 2D phase velocity maps is possible based on the information from neighbor nodes only.

The rest of the paper is organized as follows. Section II presents the related work about eikonal tomography applications and tomography over sensor networks. Section III provides background about the eikonal method. In Section IV, we present eikonal tomography algorithms and their distributed implementation using aggregation and consensus techniques. In Section V we carry out experiments using real ambient noise data from USArray. The conclusion and future work are presented in Section VI.

## II. Related Work

The use of ambient noise tomography to extract surface wave phase velocity maps has been widely studied in geophysical fields. This method has been performed around the world (e.g US[16], Asia[17], New Zealand and Australia[18], and more). Even though these approaches were successful for obtaining phase velocity maps, the majority of them have been developed to treat the traveltime between every station pair independently. Other approaches, like the one presented by *Lin et al. (2008)*[1], have utilized the array process to treat all measurements together to improve the resolution of the tomography result. However, computationally, these approaches lack real-time measurements. All of them need to collect raw time series data from the sensors to a centralized server and then process the information.

Taking advantage of the capabilities of current sensor networks, computation of seismic data across a large array of sensors is possible, which avoids centralized approaches. Different techniques for distributed computing and aiming seismic tomography were presented in [7]–[11]. All of them have been successful in obtaining 2D and 3D seismic tomography by solving inversion problems.

[2]http://cs.itd.nrl.navy.mil/work/core/

[3]http://www.earthscope.org/science/observatories/usarray

However, even though we can set an inversion problem such those presented above for computing seismic ambient noise tomography, eikonal tomography represents a new technique that does not require initial assumptions. The current approaches that perform an inversion problem usually require a ray tracing to compute traveltimes, which implies knowledge about the medium and how waves propagate[19]. However, eikonal tomography needs neither a priori information nor ray tracing. This makes it possible to perform computations more efficiently in every node. To the best to our knowledge, this is the first attempt to compute ambient noise eikonal tomography for obtaining ANSI under distributed constraints using network sensor computing capabilities.

## III. Background

### A. Eikonal Tomography

To compute ambient noise wave surface tomography, several steps are required. These involve pre-processing of raw time series data for signal conditioning[20], cross-correlation[2] and frequency time analysis[20]. The above are beyond the scope of this paper and we refer to [1] for details. In this paper, we performed ambient noise eikonal tomography in every sensor by assuming that the delay traveltimes have been computed during the stages of cross-correlation and frequency time analysis. Traveltime measurements are the input of our algorithms.

The use of eikonal tomography has two major advantages. First, the method does not need an initial model of the medium for computing. Second, even though the method accounts for bent rays, the ray tracing is not needed. In eikonal tomography, the gradient of the travel times provides information about local direction and travel of the wave, hence, deriving phase velocity maps is possible.

*1) Eikonal Equation:* Once the phase traveltime $\tau(r_i, r)$ are known for positions $r$ (arbitrary point in the medium) relative to an effective source $r_i$, the eikonal tomography is performed. The eikonal equation[3] is based on the solution of Helmholtz equation:

$$\frac{1}{c_i(r)^2} = |\nabla\tau(r_i, r)|^2 - \frac{\nabla^2 A_i(r)}{A_i(r)\omega^2} \tag{1}$$

At high frequencies, when the right-hand term is small enough, it can be dropped as:

$$\frac{\hat{k}_i}{c_i(r)} \cong \nabla\tau(r_i, r) \tag{2}$$

where, $c_i$ is the phase velocity for event $i$ at position $r$. $\hat{k}_i$ is the unit wave number vector for the event $i$ at position $r$. $\omega$ is the frequency and $A$ is the amplitude of an elastic wave at position $r$. The gradient is computed relative to the field vector $r$. Equation 2 is derived from equation 1 by ignoring the second term from the right hand side. These conclude that the gradient of the phase traveltime is related to the local slowness at $r$ position, and the direction of propagation of the wave (azimuth) denotes the local direction of the wave. Dropping the

second term on the right-hand side of equation 2 is justified when either the frequency is high or the amplitude variation is small[3]. When eikonal tomography is used, there is no need for a tomographic inversion because taking the gradient of the phase traveltime surface gives the local phase speed as a function of the direction of propagation of the wave.

*2) Isotropic Wave Speeds and Azimuthal Anisotropy:* Applying eikonal equation 2 can introduce some errors and usually the phase velocity map is noisy due to imperfections in traveltime surface calculation. To overcome this issue, a mean slowness $S_0$ and standard deviation $\sigma_{S_0}$ are calculated in order to obtain the isotropic phase speed.

$$S_0 = \frac{1}{n}\sum_{i=1}^{n} S_i \tag{3}$$

$$\sigma_{S_0} = \frac{1}{n(n-1)}\sum_{i=1}^{n}(S_i - S_0)^2 \tag{4}$$

where $n$ is the number of effective sources, and $S_i$ the distribution of slowness measurements. The isotropic phase speed $c_0$, and its uncertainty $\sigma_{c_0}$ are calculated using equations 5 and 6 respectively.

$$c_0 = \frac{1}{S_0} \tag{5}$$

$$\sigma_{c_0} = \frac{1}{S_0^2}\sigma_{S_0} \tag{6}$$

Traditionally, to compute phase velocity maps using eikonal tomography (1) we need to generate a grid of arbitrary points ($r$) in the field using interpolation of traveltimes, (2) we need to construct a phase traveltime surface for obtaining slowness and azimuth vectors in every effective source relative to each arbitrary point in the grid, (3) we have to calculate the mean slowness and standard deviation of the phase traveltime surface to overcome errors, and (4) invert the final slowness vector to obtain the velocity map.

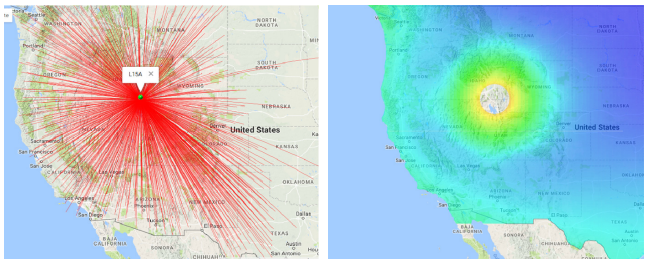
## IV. Distributed Eikonal Tomography

For computing distributed eikonal tomography, we proposed two distributed approaches: aggregation, which aggregates information in a bottom-up fashion, and consensus, which uses information of neighbors for computing a consensual average.

In the aggregation approach, every station calculates its slowness and azimuth vectors using an algorithm called *Phase Traveltime Surface* (algorithm 1). Then, an aggregation process begins between all nodes. A tree of nodes is arranged, and starting from leaf nodes, every node transmits its information to its parent. The parent has to complete an *Aggregated Isotropic Speed Map Algorithm (AISM)* which computes the average of the slowness of its children according to a calculated weight that indicates the percent of contribution of the child to the computation. In the consensus approach, once every node has calculated its slowness and azimuth vector using algorithm 1, it starts communication with its neighbors. This node sends its values and every one maintains a consensual group of values according to their weights. After nodes reach consensus or the maximum number of iterations, all nodes in the network have a complete phase velocity map.

### A. Phase Traveltime Surface Algorithm

Every node has to construct its own phase traveltime surface based on its traveltime measurements. To construct the phase traveltime surface it is necessary to interpolate traveltime data onto a finer and regular grid. For example, Figure 1(a) shows a central station and its efective sources (red lines). Figure 1(b) shows how the travel traveltimes are interpolated in a grid (in this case $0.2° \times 0.2°$ grid is used [3].)




(a) (b)

Fig. 1. Illustrating interpolation method for a central station and many virtual sources. (a) Central station L15A and its sources(USArray). (b) Result after applying minimum curvature interpolation.

**Algorithm 1** Phase TravelTime Algorithm (PTT)

1: Set $r$ as effective sources
2: Interpolate all $\tau$ of $r$ onto a $G^\circ \times G^\circ$ grid size $x \times y$
3: Perform second interpolation of $\tau$ with extra tension
4: **for** each point $k$ in the interpolated grid **do**
5: Calculate $\nabla\tau_k$
6: Obtain Slowness $S_k$
7: Calculate Azimuth $A_k$
8: **end**

In the Algorithm 1, $S_k$ and $A_k$ are the slowness and azimuth for each point $k$ in the grid respectively. The slowness vector ($S_k$) is composed of the horizontal and vertical slowness, and the gradient of a wave at position $k$ (here defined as the traveltime $\tau$) is equal to the local slowness. The output of this algorithm is the calculated slowness and azimuth for every center station. So, in our study, every station calculates its own slowness and azimuth vectors individually.

### B. Centralized Isotropic Speed Map Algorithm

Algorithm 1 gives the slowness and azimuth vectors centered at every central station. However, there are irregularities present in the phase traveltime method. To mitigate the effect of these irregularities, statistical averaging is used in the isotropic phase speed calculation (Algorithm 2) for obtaining the final phase velocity map. Suppose that total surface dimension is $x \times y$. The statistical average used is a weighted average where the weight is calculated based on azimuth values collected from all effective sources of the point. Then sum of slowness for point $k$ is calculated using the following equation:

$$SS = \sum_{k=0}^{x\times y} \frac{sw_k}{\sum_{j=0}^{x\times y} sw_j} \times S_k \tag{7}$$

where $x \times y$ denotes the total number of slowness values and $S$ and $sw$ represent the slowness and weight of every point in the grid $x, y$, respectively. Standard deviation ($\sigma$) of the slowness is determined by removing the uncertainty for each point in the surface. $\sigma$ can be expressed as

$$\sigma = \sqrt{\frac{\sum_{k=0}^{x \times y} sw_k * (S_k - SS)^2}{(1 - \sum_{k=0}^{x \times y} sw_k^2)}} \quad (8)$$

In the centralized approach, the *Isotropic Algorithm* (Algorithm 2) is used for computing the final map. Table I shows the variable reference for the Algorithm 2.

TABLE I
VARIABLE REFERENCE FOR ALGORITHM 2.

| Variable Name | Meaning |
|---|---|
| $x$ and $y$ | Size of the grid rows and columns |
| $k$ | Vector form of the grid $x$,$y$ |
| $S$ | Slowness |
| $A$ | Azimuth |
| $w$ | weight |
| $W$ | accumulated weight for every station |
| $sw$ | weight proportion |
| $SF$, $AF$, $WF$ | Final vectors for slowness, azimuth and weight |

**Algorithm 2** Isotropic Algorithm($S_{n_k}$, $A_{n_k}$,$w_{n_k}$)

```
Receive S_{n_k}, A_{n_k} and w_{n_k} of all n stations in consideration.
Initialize Resolution in x × y
for each point p in x × y grid do
    for l ← 0 until n do
        W_l = 0
        for m ← 0 until n do
            if |A_{l_p} − A_{m_p}| < 25 then
                W_l+ = w_{l_p}
            end if
        end
        sw_l ← 1/W_l
    end
    Calculate SS
    Calculate σ
    for l ← 0 until n do
        if (S_{l_p} − SS) < 2.0 ∗ σ then q+ = sw_l
        else WF_p+ = w_{l_p}
    end
    Set a = 0 and b = 0
    for l ← 0 until n do
        if (S_{l_p} − SS < 2.0 ∗ σ then
            a+ = (sw_l / q) ∗ S_{l_p}
            b+ = (sw_l / q) ∗ A_{l_p}
        end if
    end
    Set SF_p = a, AF_p = b
end
Return vectors SF, AF, WF
```

In this approach, the number of stations in consideration ($n$) is the total number of the stations in the network, and $w_{n_k}$ starts in 1 to all stations. This is the key part of the centralized algorithm. The algorithm computes weights taking into consideration local slowness for computing the total slowness of all stations. Standard deviation of slowness is computed to suppress unreliable measurements, and the final slowness vector is computed again with the new weights after these suppressions.

The output of Algorithm 2 is a vector of the final slowness ($SF$). The centralized approach then calculated the phase velocity map using the inverse of this slowness ($1/SF$).

We designed this algorithm to make it highly parallelized. If a distributed approach wants to use this algorithm, it only needs to assign weights according to the iteration in process.

### C. Aggregated Isotropic Speed Map Algorithm (AISM)

In order to exploit the advantages of eikonal method for parallelizing, we design an approach in which every node computes a partial slowness. Then, we need a technique for aggregating this partial information into a final phase velocity map.

For aggregating information, a simple approach could be to propose a tree structure, where the tree is a spanning tree of all nodes in the network. However, this approach lacks practical validity since if one node fails after it has received the information and before it has sent the result to its parent, the information of that particular subtree will be absent of the final result. To overcome this, we propose to use an aggregation algorithm based on $k$ spanning trees (2 or 3) called *MultipleTree* algorithm[13]

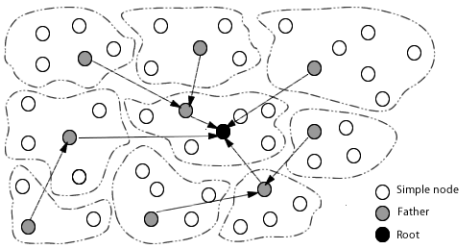



Fig. 2. Example of $T_1$ tree in the aggregation process. (Modified from [14])

In this approach, we can view the broadcast as a Breadth-First Search (BFS) in the network. Every node $u$ is associated with its level $l(u)$. This level is the length of $u$ shortest path to the root and it is computed during BFS in the $T_j$ independent spanning tree. At the beginning, the level of every node $u$ is $l(u) = \infty$, and the level of the root $r$ is $l(r) = 0$. For creating the $T_j$ spanning tree, the $r$ node makes a broadcast. This message contains the value of $T_j$ and root level $l(r) = 0$. When a node $u$ receives a message from a node $v$ contained $l(v)$, $u$ checks its $l(u)$ value to see if $l(u) = \infty$. If this happens, $u$ sets its level to $l(u) = l(v) + 1$ and forward the query to its neighbors; otherwise, $u$ stores the level of $v$ as $l(v)$. The tree has been formed when all $l$-values are less than $\infty$. An example of $T_j$ tree is shown if Figure 2. Once the $T_j$-tree is formed, we define $F(v) = \{u \in N(u)$ such that $l(u) < l(v)\}$ as the potential parents of node $v$. Every node picks uniformly at random $T_j$ parents from the set $F(v)$ (one parent for tree), and it is the way for each $T_j$ independent spanning tree to communicate.

The algorithm then computes the aggregation function in a bottom-up fashion for each $T_j$ spanning tree. Every node has to

wait until it receives the information from its children or reaches maximum time. Then the node aggregates the information and sends it to all of its parents in the $T$ trees. At the end, the final information is computed with the roots that are available.

**Algorithm 3** Aggregated Isotropic Speed Map Algorithm (AISM)

1: Initialize nodesID for all nodes $n$.
2: Initialize node level $l(u) = \infty$
3: **for** all nodes $n$ **do in parallel**
4: Execute PHASE TRAVELTIME ALGORITHM
5: **end**
6: Select randomly node root $r$ for treeID $T_j$
7: Set $l(r) = 0$ for all roots
8: **for** all nodes $n$ with $k$ points in the grid **do in parallel**
9: **if** $n = r$ **then**
10: Send broadcast to neighbors with $l(r)$ and treeID $T_j$
11: **While** until receive from all $n$ children or reach MaxTime
12: Receive ($S^t_{n_k}$,$A^t_{n_k}$,$w^t_{n_k}$) from its $n$ children
13: Added its own values to the vector $S^t_{n_k}$,$A^t_{n_k}$,$w^t_{n_k}$
14: Update $S^{t+1}_k$,$A^{t+1}_k$,$w^{t+1}_k$ = ISOTROPIC ($S^t_{n_k}$,$A^t_{n_k}$,$w^t_{n_k}$)
15: Combine with other roots in other trees $T$
16: Calculate final map $(1/S_k)$
17: Output phase velocity map
18: **else**
19: Receive $l(u)$ and tree number $T_j$ from parent node
20: Set own level $l(v) = l(u) + 1$ for tree $T_j$
21: **if** $n = leaf$ **then**
22: Send ($S_k$,$A_k$,$w_k$) to its father
23: **else**
24: Send broadcast to neighbors with $l(u)$ and treeID $T_j$
25: **While** until receive from all $n$ children or reach MaxTime
26: Receive ($S^t_{n_k}$,$A^t_{n_k}$,$w^t_{n_k}$) from its $n$ children
27: Added its own values to the vector $S^t_{n_k}$,$A^t_{n_k}$,$w^t_{n_k}$
28: Update $A^{t+1}_k$,$S^{t+1}_k$,$w^{t+1}_k$ = ISOTROPIC ($S^t_{n_k}$,$A^t_{n_k}$,$w^t_{n_k}$)
29: Send $A^{t+1}_k$,$S^{t+1}_k$,$w^{t+1}_k$ to its father
30: **end if**
31: **end if**
32: **end**

Algorithm 3 is called *Aggregated Isotropic Speed Map Algorithm (AISM)*. This algorithm starts setting all nodes in the network and establishes their levels in $\infty$. All nodes in parallel compute *Phase TravelTime Algorithm* (Step 4). Then $j$ trees are formed and roots are selected uniformly random (Step 6,7). During steps 8-32, nodes computes the aggregation in each tree $T_j$. Basically, if a node is a leaf, it sends its vector values $S_k$,$A_k$ and $w_k$ to its parent. These values correspond to its slowness vector, azimuth vector and its weight vector. For leaf, $w_k = 1$. When a parent receives all vector of all its $n$ children ($S^t_{n_k}$,$A^t_{n_k}$ and $w^t_{n_k}$), it computes the *Isotropic algorithm* for aggregating the information of its children nodes and itself. Notice that the *Isotropic algorithm* is able to aggregate partial information of all nodes in consideration. After receiving the aggregated vectors ($S^{t+1}_k$,$A^{t+1}_k$ and $w^{t+1}_k$), the node broadcast its information to its own parent. The process continues until node roots receive the information. Once each root of each tree $T_j$ has its complete map, the *Isotropic algorithm* is used to aggregate all maps of the roots in one. Then one random root computes the final phase velocity map using the inverse of the final slowness $(1/S_k)$.

### D. Consensus Isotropic Speed Map Algorithm (CISM)

In consensus algorithms, every node over the network has a initial value or group of values (measurements), and they aim to calculate the average of all these values through a distributed linear iteration method. A major advantage of these algorithms is they can calculate an average of the network measurements iteratively and distributively using local information exchange among neighbors and calculation of weighted sums at every node. Some papers have shown that consensus algorithms are effective for obtaining consensus values over a network[21].

The main idea is described as follows. Consider a graph $G = (V, E)$ where $V = \{1...n\}$ represents the vertices or nodes in the network, and $(i, j) \in E$ represents the edges or path of communication between nodes $i$ and $j$. Let $N_i$ be the set of neighbors of node $i$ and $N_{[i]}$ be the neighborhood of node $i$. Every node $i$ initially has a local value $S_i \in \mathbb{R}$. The goal is to compute $\bar{S} = \frac{1}{n}(\sum_{i=1}^{n} S_i)$ in a distributed fashion. Let $x_i(0) = S_i$. At time $t + 1$ every node transmits its current estimate to its neighbors and then updates its information $x_i(t+1) = \sum_{j \in N[i]} w_{ij} x_i(t)$ where $w$ is the weighted sum at every node.

**Algorithm 4** Consensus Isotropic Speed Map Algorithm (CISM)

1: Initialize nodesID for all nodes $n$.
2: Set $w_k = 1$ for all nodes $n$
3: **for** all nodes $n$ with points $k$ in the grid **do in parallel**
4: Execute PHASE TRAVELTIME ALGORITHM
5: **end**
6: **for** $t \leftarrow 0$ until convergence or maximum number of iteration **do**
7: Select randomly a neighbor and send $(S^t_k, A^t_k, w^t_k)$
8: Receive $(S^t_{n_k}, A^t_{n_k}, w^t_{n_k})$ of its neighbor $\in N_{[k]}$
9: Update $(S^{t+1}_k, A^{t+1}_k, w^{t+1}_k)$ = ISOTROPIC$(S^t_{n_k}, A^t_{n_k}, w^t_{n_k})$
10: **end**
11: Calculate final map $(1/S_k)$
12: Output phase velocity map

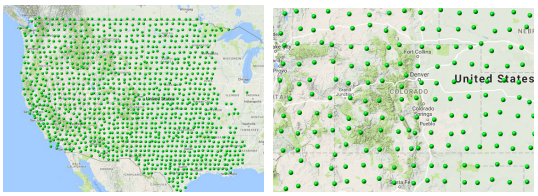
Algorithm 4 is called *Consensus Isotropic Speed Map Algorithm (CISM)*. The algorithm starts by setting all nodes of the network with an ID and a initial weight of one. Then, all nodes execute in parallel the *Phase TravelTime Algorithm*. Until they converge or reach the maximum number of iterations, all nodes execute in parallel steps 7 to 10. First, a node select randomly a neighbor and send its slowness, azimuth and weight values $(S^t_k, A^t_k, w^t_k)$ in time $t$. The node then receives the values of its neighbor $(S^t_{n_k}, A^t_{n_k}, w^t_{n_k})$ in time $t$. Using the *Isotropic Algorithm*, which has been designed to compute a weighted aggregation of information, every node obtains new averaged values for its slowness, azimuth, and weight $(S^{t+1}_{n_k}, A^{t+1}_{n_k}, w^{t+1}_{n_k})$ at time $t + 1$. For verifying convergence, every node measures the relative error between $S^t_k$ and $S^{t+1}_k$. When the consensus or the maximum number of iterations has been reached, in every node the final map is computed using the inverse of the final slowness vector $S_k$.

## V. EXPERIMENTAL RESULTS

### A. Database

For our study, we used data collected from 1211 stations of the USArray Transportable array database. Figure 3(a) and 3(b) shows these stations scattered along the west coast and central

United States. Even though spacing nodes of the USArray is based on radio of kilometers, we used this data to compare final results by assuming that the nodes are spreading in smaller distances. This assumption is made because eikonal tomography can be very well applied for monitoring shallow subsurface spanning few kilometers and a dense array of stations can be easily deployed with space in 10s meters[4]. Also, we assumed sensor nodes are tiny but powerful computational units (e.g. Beaglebone Black, Raspberry Pi).




(a) (b)

Fig. 3. Stations used in our study (a) Stations in the USArray. (b) Zoom on station in the state of Colorado.

### *B. AISM and CISM Performance and Accuracy.*

In this experiment, we setted up nodes on the CORE emulator to compute eikonal tomography for ANSI. Every node can be considered as a effective central node. Figure 4 shows a graphic interface of CORE emulator and some nodes computing phase travel time surfaces. Results of these partial surfaces after applying algorithm 1 are shown in Figure 5.

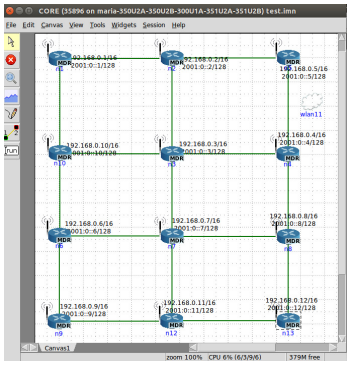
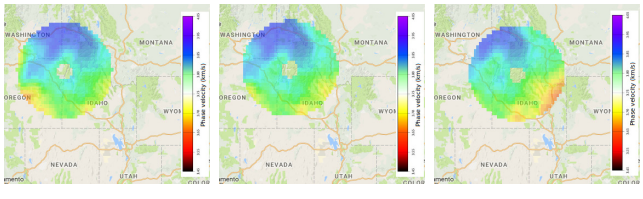


Fig. 4. CORE emulator GUI.



(a) (b) (c)

Fig. 5. Partial maps on every virtual receiver station. (a) Station G10A partial velocity map. (b) Station G11A partial velocity map. (c) Station G12A partial velocity map.

Observe that every effective central node computes its partial map using the traveltimes available from its neighbors.

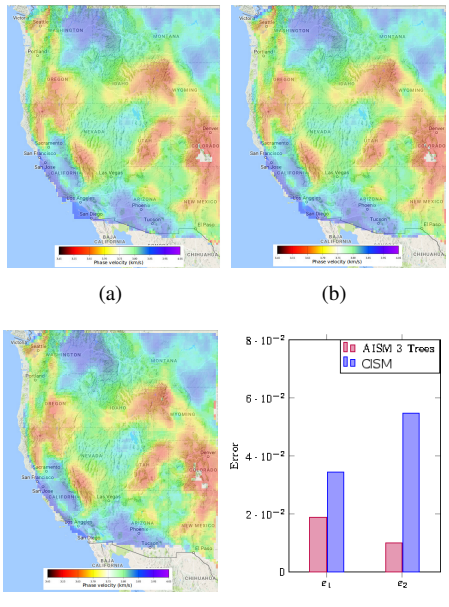
We computed AISM (3 trees) and CISM algorithms for getting final phase velocity maps over the network. Results of applying each method are shown in Figure 6. Notice that Figure 6(a) refers to the centralized approach, Figure 6(b) shows the AISM (3 trees) algorithm result, and Figure 6(c) the CISM algorithm result.

We compared these results with the centralized map for validating the accuracy of AISM and CISM algorithms. Using $\tilde{m}$, $m^*$ and $\bar{m}$ to represent the centralized model, the proposed distributed model and the mean value of $m^*$ respectively, we used the following quantitative measures of distance from the centralized model to evaluate the estimation quality.

$$e_1 = (\sum_{i=1}^{n}(\tilde{m}_i - m_i^*)^2 / \sum_{i=1}^{n}(m_i^* - \bar{m})^2)^{1/2}$$

$$e_2 = \sum_{i=1}^{n} |\tilde{m}_i - m_i^*| / \sum_{i=1}^{n} |m_i^*|$$

These represent the normalized root mean squared distance and the average value distance respectively. The result is shown in Figure 6(d). Notice that the distance from the centralized approach is higher in the CISM algorithm. This can be due to the probabilistic nature of the algorithm since it selects randomly one neighbor node to exchange data in each iteration. After some iterations certain nodes may only exchange information with the same neighbor. On the other hand, distance from centralized approach is considerably less in the AISM algorithm. Basically, this is because AISM uses all the information of the nodes to compute an aggregated map.




(a) (b) (c) (d)

Fig. 6. Comparison between final phase velocity maps. (a) Centralized Method. (b) AISM Algorithm. (c) CISM Algorithm. (d) Measures of Distance from centralized approach

### *C. Robustness and Loss Tolerance.*

In the next set of experiments, loss tolerance and robustness of AISM and CISM are evaluated. The algorithm runs with

the same configuration for three different packet loss ratios of 10%, 20% and 40% in the emulator and the results are shown in Figure 7. In this case, we ran the AISM Algorithm with two configurations: AISM using only one tree, and AISM using three trees.

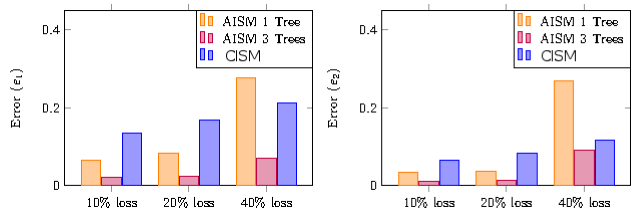



Fig. 7. Distance from centralized approach with package loss. (a) Error $e_1$. (b) Error $e_2$

Observe that for AISM (3 trees), the distance from the centralized approach is less than AISM (1 tree) and CISM even with severe package loss. This is due to the utilization of the multitrees approach. Even if one branch of one tree is lost, the other trees can recover the data with their own measurements. Since three roots are placed randomly in the network, warranties of fault-tolerance are established. If only one tree is used (AISM one tree), package loss is severe. Notice that, as expected, CISM performs well under package loss constraints. Even with 40% fewer transmitted packages, CISM is able to recover a closer phase velocity maps compared to centralized approach. This is mainly because in CISM algorithm, every node spreads its information to its neighbors in every iteration.

Even though AISM (3 trees) seems to be more fault-tolerant than CISM, when the loss ratio increases, CISM is more stable and it does not show significant difference in the distance from the centralized approach. In contrast, AISM with three trees shows a considerable increment of its distance from the centralized map. Figure 8 shows error results for both algorithms when loss ratio increases.

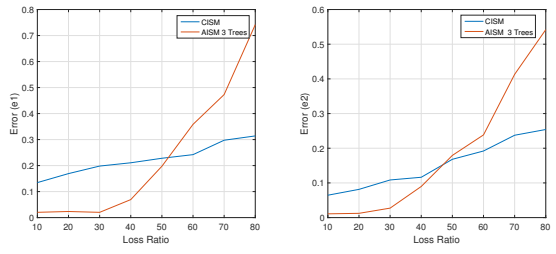



Fig. 8. Loss ratio comparison. (a) Error $e_1$. (b) Error $e_2$

Figure 9 gives the 2D tomography with packet loss and we can see that with 10% or even 40% packet loss, there is no significance difference in terms of the image reconstruction when compared to the results with no packet loss. We selected Figure 9 as a zoom of one part of the maps in Figure 6a and 6b. The black line is the ground true area of study obtained with the centralized approach. Notice that in both algorithms, this area shows small variation respecting centralized results even with severe package loss.

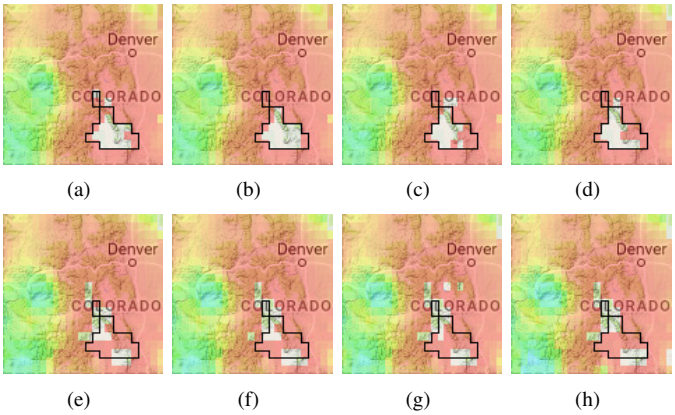



Fig. 9. Loss tolerance and robustness of AISM (3 trees) and CISM with a map with incomplete parts. (a) AISM no package loss. (b) AISM 10% Loss. (c) AISM 20% Loss. (d) AISM 40% Loss. (e) CISM no package loss. (f) CISM 10% Loss. (g) CISM 20% Loss. (h) CISM 40% Loss.

### *D. Communication Cost*

In this section, we compared the communication cost of the centralized algorithm with proposed distributed algorithms in terms of number of messages exchanged to reach the solution. This is computed based on the number of messages that every node receives during the computation. From Figure 10(a) we can see that communication cost in a centralized setup is high near the "central node" as all the slowness, azimuth information is transferred over the network before computation. Figure 10(b) shows the communication pattern for AISM algorithm using three trees. Notice that the communication cost is less compared centralized scheme. This is mainly because AISM distributed the computation load in trees of nodes. Observe also that the communication is greater at the roots of the trees. In Figure 10(c) we present the communication pattern of CISM algorithm. Observe that this communication pattern is almost flat which indicates CISM is able to balance the computation load across the network.

Communication volume (Figure 10(d)) is less in CISM because we stop the computation when we obtain an acceptable velocity map (when we can recover a image of velocity map for the expected resolution), and we do not wait for convergence. However, if the algorithm runs with a high number of iterations, its communication volume may be higher than AISM with three tress.

With the previous results, we can conclude that both methods produce similar results to the centralized approach in terms of image reconstruction. The AISM has acceptable communication cost, since children communicate with their parents only one time. Also, reliability under package loss constraints depends on the number of trees involved in the computation. On the other hand, the CISM is robust even if a node fails, its neighbors have enough information to maintain a reliable final map. However, communication cost is directly dependent on the number of iterations required for reaching consensus, since every node

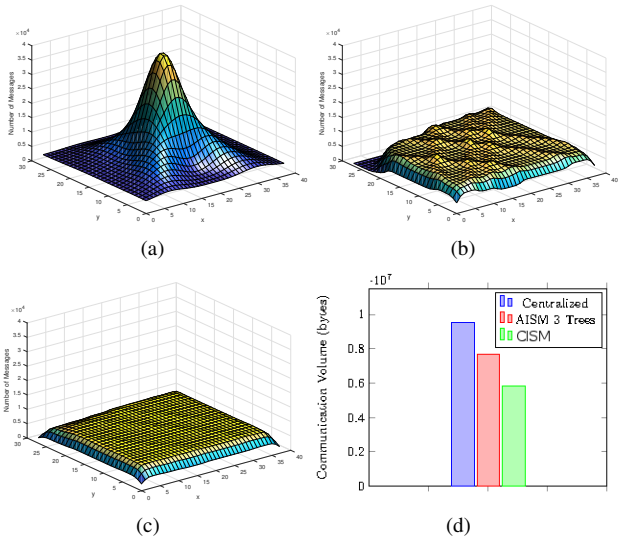



Fig. 10. Communication cost. (a) Centralized Approach. (b) AISM Algorithm (3 trees). (c) CISM Algorithm. (d) Communication volume

has to communicate many times with their neighbors in order to reach this consensus. So, if the number of iterations is high, the communication cost will be high too.

## VI. Conclusion and Future Work

In this paper, we presented two distributed methods for computing ambient noise subsurface tomography and imaging subsurface velocities in real-time using network networks. We showed that computing information at the edge level and cooperating with neighbors make it possible to illuminate near-surface velocities using the eikonal method. We showed that both distributed algorithms produce results close to the centralized approach, and they balance communication across the network. Furthermore, we also tested our algorithms under natural conditions of sensor networks, such as loss of packages, and showed they are robust in terms of loss tolerance. For future work, we want to improve the communication cost of the algorithms by reducing the size and number of messages between nodes and establishing a strong stopping criterion. Also, we want to include previous steps of ambient noise study such as cross-correlation and frequency time analysis for having a complete system to analyse ambient noise.

## VII. Acknowledgements

Our research is partially supported by NSF CyberSEES Program (NSF-1442630).

## References

[1] F.-C. Lin, M. P. Moschetti, and M. H. Ritzwoller, "Surface wave tomography of the western united states from ambient seismic noise: Rayleigh and love wave phase velocity maps," *Geophysical Journal International*, vol. 173, no. 1, pp. 281–298, 2008.

[2] M. H. Ritzwoller, F.-C. Lin, and W. Shen, "Ambient noise tomography with a large seismic array," *Comptes Rendus Geoscience*, vol. 343, no. 8, pp. 558–570, 2011.

[3] F.-C. Lin, M. H. Ritzwoller, and R. Snieder, "Eikonal tomography: surface wave tomography by phase front tracking across a regional broad-band seismic array," *Geophysical Journal International*, vol. 177, no. 3, pp. 1091–1110, 2009.

[4] F.-C. Lin, D. Li, R. W. Clayton, and D. Hollis, "High-resolution 3D shallow crustal structure in long beach, california: Application of ambient noise tomography on a dense seismic array," *Geophysics*, vol. 78, no. 4, pp. Q45–Q56, 2013.

[5] N. M. Shapiro, M. Campillo, L. Stehly, and M. H. Ritzwoller, "High-resolution surface-wave tomography from ambient seismic noise," *Science*, vol. 307, no. 5715, pp. 1615–1618, 2005.

[6] H. Yao, R. D. van Der Hilst, and V. Maarten, "Surface-wave array tomography in SE Tibet from ambient seismic noise and two-station analysis–I. Phase velocity maps," *Geophysical Journal International*, vol. 166, no. 2, pp. 732–744, 2006.

[7] G. Kamath, L. Shi, W.-Z. Song, and J. M. Lees, "Distributed travel-time seismic tomography in large-scale sensor networks," *Journal of Parallel and Distributed Computing*, vol. 89, 2016.

[8] L. Zhao, W.-Z. Song, L. Shi, and X. Ye, "Decentralized seismic tomography computing in cyber-physical sensor systems," *Cyber-Physical Systems, Taylor; Francis*, 2015.

[9] G. Kamath, L. Shi, E. Chow, and W. Song, "Distributed tomography with adaptive mesh refinement in sensor networks," *International Journal of Sensor Network*, 2015. [Online]. Available: http://sensorweb.engr.uga.edu/wp-content/uploads/2016/08/DropboxChooserAPI_KSCS-IJSNET2015.pdf

[10] G. Kamath, W.-Z. Song, P. Ramanan, L. Shi, and J. Yang, "Dristi: Distributed real-time in-situ seismic tomographic imaging," in *14th International Conference on Ubiquitous Computing and Communications (IUCC)*, Liverpool, UK, 2015.

[11] L. Shi, W.-Z. Song, M. Xu, Q. Xiao, J. M. Lees, and G. Xing, "Imaging volcano seismic tomography in sensor networks," in *The 10th Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks (IEEE SECON)*, 2013. [Online]. Available: http://sensorweb.engr.uga.edu/wp-content/uploads/2016/08/DropboxChooserAPI_SSXX-SECON2013.pdf

[12] G. Kamath, L. Shi, and W.-Z. Song, "Component-average based distributed seismic tomography in sensor networks," in *2013 IEEE International Conference on Distributed Computing in Sensor Systems*. IEEE, 2013, pp. 88–95.

[13] M. Bawa, H. Garcia-Molina, A. Gionis, and R. Motwani, "Estimating aggregates on a peer-to-peer network," *submitted for publication*, 2003.

[14] G. Kamath, L. Shi, and W.-Z. Song, "Component-average based distributed seismic tomography in sensor networks," in *The 9th IEEE International Conference on Distributed Computing in Sensor Systems (IEEE DCOSS)*, 2013.

[15] T. C. Aysal, M. E. Yildiz, A. D. Sarwate, and A. Scaglione, "Broadcast gossip algorithms for consensus," *IEEE Transactions on Signal processing*, vol. 57, no. 7, pp. 2748–2761, 2009.

[16] G. Ekström, G. A. Abers, and S. C. Webb, "Determination of surface-wave phase velocities across USArray from noise and Aki's spectral formulation," *Geophysical Research Letters*, vol. 36, no. 18, 2009.

[17] L. Fang, J. Wu, Z. Ding, and G. Panza, "High resolution rayleigh wave group velocity tomography in North China from ambient seismic noise," *Geophysical Journal International*, vol. 181, no. 2, pp. 1171–1182, 2010.

[18] P. Arroucau, N. Rawlinson, and M. Sambridge, "New insight into cainozoic sedimentary basins and palaeozoic suture zones in southeast Australia from ambient noise surface wave tomography," *Geophysical Research Letters*, vol. 37, no. 7, 2010.

[19] P. Gouédard, H. Yao, F. Ernst, and R. D. van der Hilst, "Surface wave eikonal tomography in heterogeneous media using exploration data," *Geophysical Journal International*, vol. 191, no. 2, pp. 781–788, 2012.

[20] G. Bensen, M. Ritzwoller, M. Barmin, A. Levshin, F. Lin, M. Moschetti, N. Shapiro, and Y. Yang, "Processing seismic ambient noise data to obtain reliable broad-band surface wave dispersion measurements," *Geophysical Journal International*, vol. 169, no. 3, pp. 1239–1260, 2007.

[21] L. Zhao, W.-Z. Song, and X. Ye, "Fast decentralized gradient descent method and applications to in-situ seismic tomography," in *IEEE International Conference on Big Data (IEEE BigData 2015)*, 2015.