

Imaging Seismic Tomography in Sensor Network

Lei Shi[†], Wen-Zhan Song[†], Mingsen Xu[†], Qingjun Xiao[†], Jonathan M. Lees[‡] and Guoliang Xing[§]

[†]Department of Computer Science, Georgia State University

[‡]Department of Geological Sciences, University of North Carolina at Chapel Hill

[§]Department of Computer Science and Engineering, Michigan State University

Email: [†]wsong@gsu.edu, [‡]jonathan.lees@unc.edu, [§]glxing@msu.edu

Abstract—Tomography imaging, applied to seismology, requires a new, decentralized approach if high resolution calculations are to be performed in a sensor network configuration. The real-time data retrieval from a network of large-amount wireless seismic nodes to a central server is virtually impossible due to the sheer data amount and resource limitations. In this paper, we present a distributed multi-resolution evolving tomography algorithm for processing data and inverting volcano tomography in the network, while avoiding costly data collections and centralized computations. The new algorithm distributes the computational burden to sensor nodes and performs real-time tomography inversion under the constraints of network resources. We implemented and evaluated the system design in the CORE emulator. The experiment results validate that our proposed algorithm not only balances the computation load, but also achieves low communication cost and high data loss tolerance.¹

Index Terms—Distributed Computing, In-network Processing, Seismic Tomography, Sensor Network

I. INTRODUCTION

Existing volcano instrumentation and monitoring systems do not yet have the capability to recover physical dynamics with sufficient resolution. At present, raw seismic data are typically collected at central observatories for post processing. Seismic sampling rates for volcano monitoring are usually in the range of 16-24 bit at 50-200Hz. With such high-fidelity sampling, it is virtually impossible to collect raw, real-time data from a large-scale dense sensor network, due to severe limitations of energy and bandwidth at current, battery-powered sensor nodes. As a result, at some most threatening, active volcanoes, fewer than 20 nodes [22] are thus maintained. This limits our ability to understand volcano dynamics and physical processes inside volcano conduit systems. Substantial scientific discoveries on the geology and physics of active volcanism would be imminent if seismic resolution could be increased by an order of magnitude or more.

Nowadays, the sensor network technology has matured to the point where it is possible to deploy and maintain a large-scale network for volcano monitoring and utilize the computing power of each node for distributed tomography inversion. Tomography algorithms commonly in use today cannot be easily implemented under field circumstances proposed here because they rely on centralized algorithms and require massive amounts of raw seismic data collected on a central processing unit. Thus, real-time volcano tomography

of high resolution requires a new approach with respect to computation algorithm and network design.

In this paper, a distributed multi-resolution evolving tomography algorithm is proposed. This algorithm partitions the volcano structure geometrically and divides the tomography inversion problem accordingly for distributing the computation load to the network. The tomography result is refined as more earthquakes are recorded over time. Compared with the centralized method with data collection, this algorithm performs real-time data processing and tomography inversion in the network while meeting the severe resource (bandwidth, energy, computing power, memory, etc.) constraints. To our best knowledge of the literature, this work is the first attempt to distribute seismic tomography computation in the sensor network. The algorithm proposed here has application to the fields far beyond the specifics of volcano, e.g., oil field explorations have similar problems and needs.

The rest of the paper is organized as follows. Section II presents the background knowledge of seismic tomography and the related study of parallel and distributed least-squares methods. We then propose the algorithm in section III. Section IV discusses the system design and implementation; evaluates the algorithm and system performance with emulations. Section V concludes the paper.

II. BACKGROUND AND RELATED STUDY

Static tomography inversion for 3D structure, applied to volcanoes and oil field explorations, has been explored since the late 1970's [5], [26], [8]. In volcano applications, tomography inversion used passive seismic data from networks consisting of tens of nodes, at most. The development and application to volcanoes include Mount St. Helens [7], [9], [27], Mt. Rainier [15], Kliuchevskoi, Kamchatka, Russia [11], and Unzen Volcano, Japan [16]. At the Coso geothermal field, California, researchers have made significant contributions to seismic imaging by coordinating tomography inversions of velocity [29], anisotropy [12], attenuation [28] and porosity [13].

However, the resolution for such inversions is typically in kilometers or even tens of kilometers. Details pertaining to the complex plumbing systems of volcanoes cannot be resolved due to the lack of nodes coverage on the edifice where signals from the conduit system emanate. In petroleum exploration applications of time-lapse subsurface imaging, thousands of nodes have been incorporated. But the 3D imaging is still based on centralized off-line processing and is typically accomplished by multiple active-source recordings where variations over multiple year spans are the main goal [8]. The time

¹Our research is partially supported by NSF-CNS-1066391, NSF-CNS-0914371, NSF-CPS-1135814 and NSF-CDI-1125165.

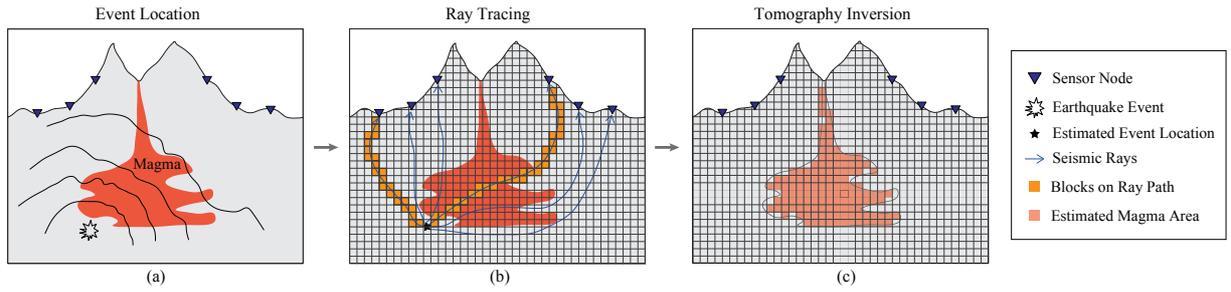


Fig. 1: Procedure of Seismic Tomography

scales involved in real-time hazard mitigation, on the other hand, are tens of minutes to hours. To achieve effective disaster warnings and timely responses, new schemes and methodologies are required to solve the real-time seismic tomography problem. This is one critical motivation for developing a distributed real-time volcano tomography inversion algorithm.

The approach employed for 3D seismic tomography in this paper is travel-time seismic tomography, which uses P-wave arrival times at sensor nodes to derive the internal velocity structure of the volcano; this model is continuously refined and evolving, as more earthquakes are recorded over time. The basic procedure of seismic tomography illustrated in Figure 1 involves three steps.

(1) *Event Location*. Once an earthquake happens, the nodes detected seismic disturbances can determine P-wave arrival times, which are then used to estimate the earthquake source location and origin time in the volcanic edifice (Figure 1(a)).

(2) *Ray Tracing*. Following each earthquake, seismic rays propagate to nodes and pass through anomalous media. These rays are perturbed and thus register anomalous residuals. Given the source locations of the seismic events and current velocity model of the volcano, ray tracing is to find the ray paths from the seismic source locations to the nodes (Figure 1(b)).

(3) *Tomographic Inversion*. The traced ray paths, in turn, are used to image a 3D tomography model of the velocity structure within the volcano. As shown in Figure 1(c) the volcano is partitioned into small blocks and the seismic tomography problem can be formulated as a large, sparse matrix inversion problem. Suppose that there are N nodes and J earthquakes, we consider a perturbation approach here. Let \mathbf{s}^* be the slowness (reciprocal of velocity) model of the volcano with resolution M (blocks). \mathbf{s}^* can be assumed to be a reference model, \mathbf{s}^0 , plus a small perturbation \mathbf{s} , i.e., $\mathbf{s}^* = \mathbf{s}^0 + \mathbf{s}$.

We can estimate the ray travel times in step (1) by the arrival times and estimated event origin times. Let $\mathbf{t}_i^* = [t_{i1}^*, t_{i2}^*, \dots, t_{iJ}^*]^T$, where t_{ij}^* is the travel time experienced by node i in the j -th event. Based on the ray paths traced in step (2), the travel time of a ray is the sum of the slowness in each block times the length of the ray within that block, i.e., $t_{ij}^* = \mathbf{A}_i[j, m] \cdot \mathbf{s}^*[m]$ where $\mathbf{A}_i[j, m]$ is the length of the ray from the j -th event to node i in the m -th block and $\mathbf{s}^*[m]$ is the slowness of the m -th block. Let $\mathbf{t}_i^0 = [t_{i1}^0, t_{i2}^0, \dots, t_{iJ}^0]^T$ be the unperturbed travel times where $t_{ij}^0 = \mathbf{A}_i[j, m] \cdot \mathbf{s}^0[m]$. In the rest of the paper, we use *observed travel time* and *predicted travel time* to indicate the same meanings of t_{ij}^* and t_{ij}^0 . In

matrix notation we have the following equation,

$$\mathbf{A}_i \mathbf{s}^* - \mathbf{A}_i \mathbf{s}^0 = \mathbf{A}_i \mathbf{s} \quad (1)$$

where $\mathbf{A}_i[j, m]$ represents the element at the j -th row and m -th column of matrix $\mathbf{A}_i \in \mathbb{R}^{J \times M}$. Let $\mathbf{t}_i = [t_{i1}, t_{i2}, \dots, t_{iJ}]^T$ be the travel time residual such that $\mathbf{t}_i = \mathbf{t}_i^* - \mathbf{t}_i^0$, equation (1) can be rewritten as,

$$\mathbf{A}_i \mathbf{s} = \mathbf{t}_i \quad (2)$$

We now have a linear relationship between the travel time residual observations, \mathbf{t}_i , and the slowness perturbations, \mathbf{s} . Since each ray path intersects with the model only at a small number of blocks compared with M , the design matrix, \mathbf{A}_i , is sparse. The seismic tomography inversion problem is to solve the system,

$$\mathbf{A} \mathbf{s} = \mathbf{t} \quad (3)$$

where $\mathbf{A} = [\mathbf{A}_1^T, \mathbf{A}_2^T, \dots, \mathbf{A}_N^T]^T$ and $\mathbf{t} = [\mathbf{t}_1^T, \mathbf{t}_2^T, \dots, \mathbf{t}_N^T]^T$. This system is usually overdetermined and the inversion aims to find the least-squares solution \mathbf{s} such that,

$$\mathbf{s} = \arg \min_{\mathbf{s}} \|\mathbf{t} - \mathbf{A} \mathbf{s}\|^2 \quad (4)$$

In seismic tomography, the event location can also be formulated as a least-squares problem by Geiger's Method [2], and the estimation vector is of length 4 (event origin time and 3D coordinates). Since the dimension of the estimation vector is fixed and small, a centralized solution can be applied in the network for this problem. In step (2), each node traces ray path based on a reference model. This can be naturally distributed since the ray tracing computation is entirely local. The third step is the most computationally intensive and time consuming aspect of high resolution seismic tomography. The sparse system can be solved by conjugate gradient method or row action method [10]. However, designed for high-performance computers, these centralized approaches need significant amount of computational/memory resources and require the knowledge of global information. As a result, they cannot be directly distributed in wireless sensor network. Thus, the key research challenge here is how to solve the least-squares problem in step (3) distributedly under the severe constraints of wireless sensor network. In this paper, we focus on distributed tomography inversion algorithm, while assuming that the event arrival timing at each node has been extracted from the raw seismic data by each node itself [21], [24], as well as that the event location and ray tracing have been done.

The methods to solve least-squares problem mainly fall into two categories, direct methods and iterative methods. Iterative methods for solving large sparse linear systems of equations are advantageous over the classical direct solvers, especially for huge systems [3]. Methods of parallelizing least-squares solutions on distributed memory architecture have been studied for both direct and iterative methods, but there are few studies on distributing the least-squares solutions from a wireless sensor network point of view.

Straková, Gansterer and Zemen investigate randomized algorithms based on gossiping for distributed QR factorization [23]. This algorithm demands asynchronous randomized information exchange among nodes. The problem is that gossiping based algorithms converge slow and a back substitution node by node is required to get the least-squares solution where the delay is introduced. Renaut proposed a parallel multisplitting solution of the least-squares problem [17] where the solutions to the local problems are recombined using weighting matrices. The problem to distribute this algorithm in the network is that it requires broadcast communications in each iteration. Yang and Brent describe a modified conjugate gradient least-squares method to reduce inner products global synchronization points and improve the parallel performance [30]. This can also be potentially distributed over the network but the broadcast communication is still required in each iteration.

In the literature of signal processing, there are a few studies on consensus-based Distributed Least Mean Square (D-LMS) algorithms [14], [19] in sensor network. These algorithms let each node maintain its own local estimation and, to reach the consensus, exchange the local estimation only within its neighbors. This can also be used for getting least-squares solutions statistically. The problem is that consensus-based methods can be slow [25] on convergence and the communication cost highly depend on the dimension of estimation. In large-scale network, consensus based algorithms introduce not only high communication overhead but also long delays involving frequent multi-hop communications. Sayed and Lopes developed a Distributed Recursive Least-Squares (D-RLS) strategy by appealing to collaboration techniques to achieve an exact recursive solution [18]. But it requires a cyclic path in the network to perform the computation node by node while exchanging a large dense matrix between nodes.

TABLE I: Communication Cost Analysis

	Algorithm	Communication Cost
1	D-MS	kN^2m
2	D-MCGLS	$(k+1)N(m+n+2N)$
3	D-LMS	$kN(D_{avg}+1)n$
4	D-CE	kNn
5	D-RLS	$N(n+n^2)$
6	D-CARP	$2kNn$

Kaczmarz's row projection method [6], also known as Algebraic Reconstruction Technique (ART), was one of the first iterative methods used for large nonsymmetric systems. Its main advantages are robustness and cyclic convergence on inconsistent systems. Dan and Rachel proposed scheme performs Kaczmarz row projections within the blocks and merges the results by certain component-averaging operations, called

component-averaged row projections (CARP) [3]. CARP is a robust and memory efficient method to solve sparse systems. However, to distribute CARP in the network, the broadcast communication is required and the performance depends on the partition number of the equation system.

Table I gives an analysis of the communication cost to distribute the methods mentioned above. For more details about how to distribute the algorithms and the communication analysis with emulation evaluation, please refer to our technical report on a survey of distributed least-squares iterative methods in networks [20]. Considering the least-squares problem in equation (4) where $\mathbf{A} \in \mathbb{R}^{m \times n}$ ($m \geq n$), $\mathbf{s} \in \mathbb{R}^n$ and $\mathbf{t} \in \mathbb{R}^m$. Suppose that the algorithm converges within k iterations in a multi-hop network of N nodes, D_{avg} denotes the average node degree of the network.

The methods mentioned above have been proved to be convergent, but there are several system design problems to distribute them in the wireless sensor network due to high communication cost or long delay. In high resolution seismic tomography, we intend to solve a large sparse system (n can be as large as tens of thousands even millions) in a large-scale network (hundreds of nodes), i.e., $N \ll m$ and $N \ll n$. Considering this, the multisplitting method and CARP have less communication cost if all algorithms converge in the same order of iteration number. But the iteration number highly depends on the matrix condition number (from the evaluation in [20], these methods need hundreds to thousands iterations to converge over a network with hundreds of nodes). Besides, some methods either requires broadcast communication per iteration or a path in the network to perform the computation node by node. This results to either high communication cost or long delay if the number of iterations is large. Since the convergence analysis of these methods is based on the information completeness, the solution is unpredictable if data loss happens in the communication among iterations. This introduces difficulties to maintain a stable wireless network. In next section, we will discuss how to address these challenges and present a distributed least-squares algorithm which avoids frequent broadcast communication and long delay, at the same time, can still approximate the least-squares solution.

III. ALGORITHM

We developed a new tomography partition and computation distribution algorithm with a multi-resolution evolving scheme to distribute the computation load, reduce the communication cost and approximate the least-squares solution of the seismic tomography inversion problem in the network. To distribute the computation load, we first partition the volcano structure geometrically and the system $\mathbf{A}\mathbf{s} = \mathbf{t}$ correspondingly. Then some nodes are selected as *landlords* to compute part of the tomography model. The computation on each landlord is entirely local so that the communication cost is bounded. Since the computation on each landlord only uses part of the system $\mathbf{A}\mathbf{s} = \mathbf{t}$, the result is not equivalent to the solution of the original system. To approximate the optimal solution, we introduce the multi-resolution evolving scheme: the network initially computes a coarse resolution tomography without partition when small amount of earthquake events arrive; as more and more earthquake events arrive, the network will compute

finer and finer resolution tomography with more partitions. The intuition behind this is that the network first computes an outline of the volcano structure in a low resolution then fills up with finer details inside. With the multi-resolution evolving scheme, we do not need to wait for all computation done and can retrieve the intermediate results under low resolutions in a real-time manner.

In this section, first we use an example to show the idea of the tomography partition and the computation distribution over the network; second we introduce the multi-resolution evolving scheme and give the description of the algorithm.

A. Tomography Partition

The example in Figure 2 illustrates that how to partition the volcano structure geometrically (vertically) and the corresponding system $\mathbf{A}\mathbf{s} = \mathbf{t}$ for distributing the computation in the network.

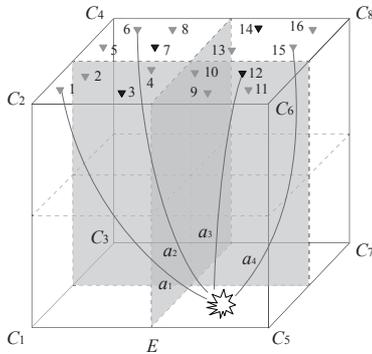


Fig. 2: Tomography Partition. The resolution of cube E is $2 \times 2 \times 2$ (8 blocks C_1 to C_8) and 16 sensor nodes are deployed on top of E .

In this example, cube E is vertically partitioned into 4 parts (E_1 to E_4) and there are 4 sensor nodes in each partition, e.g., node 1, 2, 3 and 4 are on top of E_1 consisting of blocks C_1 and C_2 . Suppose that one earthquake happens in block C_5 and node 1, 6, 12 and 15 detect this event. Once the event location is done, these 4 nodes do ray tracing individually and get 4 ray paths a_1 , a_2 , a_3 and a_4 . Assume that a_1 penetrates C_5 , C_1 and C_2 , a_2 penetrates C_5 , C_1 , C_3 and C_4 , a_3 penetrates C_5 and C_6 , and a_4 penetrates C_5 , C_7 and C_8 . These 4 ray paths can form a system $\mathbf{A}\mathbf{s} = \mathbf{t}$ as following,

$$\begin{bmatrix} a_{1,1} & a_{1,2} & 0 & 0 & a_{1,5} & 0 & 0 & 0 \\ a_{2,1} & 0 & a_{2,3} & a_{2,4} & a_{2,5} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & a_{3,5} & a_{3,6} & 0 & 0 \\ 0 & 0 & 0 & 0 & a_{4,5} & 0 & a_{4,7} & a_{4,8} \end{bmatrix} \cdot [s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8]^T = [t_1, t_2, t_3, t_4]^T$$

where $a_{l,m}$ is the intersecting length of the l -th ray path and the m -th block, s_m is the slowness perturbation of m -th block, and t_l is the travel time residual observation of the l -th ray path. Notice that each column in \mathbf{A} contains the lengths of all the ray paths which penetrate the corresponding block. So

the vertical partition of cube E can be mapped to a column partition on the system $\mathbf{A}\mathbf{s} = \mathbf{t}$,

$$\begin{bmatrix} a_{1,1} & a_{1,2} & 0 & 0 & a_{1,5} & 0 & 0 & 0 \\ a_{2,1} & 0 & a_{2,3} & a_{2,4} & a_{2,5} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & a_{3,5} & a_{3,6} & 0 & 0 \\ 0 & 0 & 0 & 0 & a_{4,5} & 0 & a_{4,7} & a_{4,8} \end{bmatrix} \cdot [s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8]^T = \begin{bmatrix} t_{1,1} + t_{1,2} + t_{1,3} + t_{1,4} \\ t_{2,1} + t_{2,2} + t_{2,3} + t_{2,4} \\ t_{3,1} + t_{3,2} + t_{3,3} + t_{3,4} \\ t_{4,1} + t_{4,2} + t_{4,3} + t_{4,4} \end{bmatrix}$$

where $t_{l,m}$ is the partial travel time residual of the l -th ray path in the m -th block. The system can be expressed as,

$$[\mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3, \mathbf{A}_4] \cdot [s_1, s_2, s_3, s_4]^T = [t_1 + t_2 + t_3 + t_4]$$

where \mathbf{A}_p is column partition of \mathbf{A} corresponding to tomography partition p , t_p is the partial time residuals for \mathbf{A}_p , and s_p is the partial slowness perturbation model of the blocks in partition p . Then in each partition, one node is selected as the landlord, e.g., 3, 7, 12 and 14 are landlords in Figure 2. The landlord in partition p solves the subsystem $\mathbf{A}_p \cdot s_p = t_p$, and the global tomography can be obtained by combining all s_p .

Next is how to partition the travel time residual of each ray. Since the travel time residual is based on the observation of P-wave arrival time which usually contains noise, it is difficult to partition the travel time residual exactly for each partial ray. Here an approximation method is employed to derive the partial travel time residuals based on the reference model. For example, assume that the reference slowness model for E in Figure 2 is $\mathbf{s}^0 = [\hat{s}_1, \hat{s}_2, \hat{s}_3, \hat{s}_4, \hat{s}_5, \hat{s}_6, \hat{s}_7, \hat{s}_8]^T$. Let T_2 be the observed travel time of ray a_2 and the predicted travel time of a_2 is $T_2(0) = a_2 \cdot \mathbf{s}^0$, thus the travel time residual $t_2 = T_2 - T_2(0)$. Then the partial predicted travel time can be approximated by the reference slowness model,

$$\begin{aligned} T_{2,1}(0) &= \vec{a}_{2,1} \cdot [\hat{s}_1, \hat{s}_2]^T & T_{2,2}(0) &= \vec{a}_{2,2} \cdot [\hat{s}_3, \hat{s}_4]^T \\ T_{2,3}(0) &= \vec{a}_{2,3} \cdot [\hat{s}_5, \hat{s}_6]^T & T_{2,4}(0) &= \vec{a}_{2,4} \cdot [\hat{s}_7, \hat{s}_8]^T \end{aligned}$$

where $T_{2,1}$ is the partial travel time of ray a_2 in partition E_1 , $\vec{a}_{2,1}$ is the part of ray path a_2 in E_1 . The travel time residual then is proportionally partitioned according to the predicted travel time,

$$\begin{aligned} t_{2,1} &= t_2 \cdot \frac{T_{2,1}(0)}{T_2(0)} & t_{2,2} &= t_2 \cdot \frac{T_{2,2}(0)}{T_2(0)} \\ t_{2,3} &= t_2 \cdot \frac{T_{2,3}(0)}{T_2(0)} & t_{2,4} &= t_2 \cdot \frac{T_{2,4}(0)}{T_2(0)} \end{aligned}$$

Here we formalize the estimation of the partial travel time residuals. Let $\vec{a}_{l,p}$ be the partial ray path of the l -th ray in partition E_p and let $t_{l,p}$ be the corresponding partial travel time residual, $t_{l,p}$ can be estimated as,

$$t_{l,p} = t_l \cdot \frac{T_{l,p}(0)}{T_l(0)}$$

where $T_{l,p}(0)$ is the predicted travel time of l -th ray in partition E_p . Assume that $\hat{s}_p(0)$ is the partial slowness reference model of partition E_p then,

$$T_{l,p}(0) = \vec{a}_{l,p} \cdot \hat{s}_p(0)$$

since $t_l = T_l - T_l(0)$ and $T_l(0) = a_l \cdot s^0$, the partial travel time residual $t_{l,p}$ is,

$$t_{l,p} = T_l \cdot \frac{\tilde{a}_{l,p} \cdot \hat{s}_p(0)}{a_l \cdot s^0} - \tilde{a}_{l,p} \cdot \hat{s}_p(0)$$

The previous discussion explains how to partition the tomography as well as $\mathbf{A}\mathbf{s} = \mathbf{t}$. Next we will show that how the partition and the computation distribution can be done in the network. First, after the ray tracing done, each node needs to send the partial ray path and the estimated partial travel time residual to the corresponding landlords for constructing the subsystems. Then each landlord can compute the partial slowness perturbation and broadcast it to the network so that each node can update its reference model part by part for future ray tracing. So there are two communication patterns in the network, unicast for sending partial rays and broadcast to synchronize the reference model. Both of them only happen once in one computation.

B. Multi-resolution Evolving Tomography

In this section, we discuss the details about the multi-resolution evolving scheme and give the description of the proposed algorithm. Figure 3 illustrates how the multi-resolution evolving scheme works following the example in Figure 2. Notice that the computation of the partial travel time residuals highly depends on the slowness reference model. In the multi-resolution evolving scheme, the tomography model is not partitioned initially so that a good initial guess of the slowness model can be derived. This initial guess is used to estimate the partial travel time residuals later for approximating the optimal solution.

Suppose that the resolution of the tomography model is $d \times d \times d$ in the beginning, a single landlord (node 10 in the example) will compute the first perturbation for the reference model. Here d is a configurable parameter, in the experiment setup of this paper, we use $d = 8$ (refer to section IV-A). Then the resolution increases to $2d \times 2d \times 2d$, and the tomography model is partitioned into 4 parts and distributed to 4 landlords (node 3, 7, 12 and 14) for computation as illustrated in Figure 3. Notice that, here the resolution of each partition is $d \times d \times 2d$. The aforementioned partition procedure will be recursively applied in each partition when sufficient more new earthquake events arrive, until the required resolution is achieved. Thus, at the $(r+1)$ -th ($r = 0, 1, 2, \dots$) resolution, the tomography model has resolution $2^r d \times 2^r d \times 2^r d$ and is partitioned into 4^r parts and evenly distributed to $\max(N, 4^r)$ landlords.

Now we give the formal description of the Distributed Multi-resolution Evolving Tomography (DMET) algorithm, see Algorithm 1. Suppose that there are q different resolution levels in our multi-resolution scheme, let d be the initial resolution dimension.

At the $(r+1)$ -th resolution, the tomography model E with resolution Q_r is vertically and evenly partitioned into P_r different parts, then the p -th part E_p contains Q_r/P_r blocks ($Q_r = 2^r d \times 2^r d \times 2^r d$ and $P_r = 4^r$). The system $\mathbf{A}\mathbf{s} = \mathbf{t}$ can be partitioned by columns accordingly,

$$[\mathbf{A}_1, \mathbf{A}_2, \dots, \mathbf{A}_{P_r}] \cdot [\mathbf{s}_1, \mathbf{s}_2, \dots, \mathbf{s}_{P_r}]^T = [\mathbf{t}_1 + \mathbf{t}_2 + \dots + \mathbf{t}_{P_r}]$$

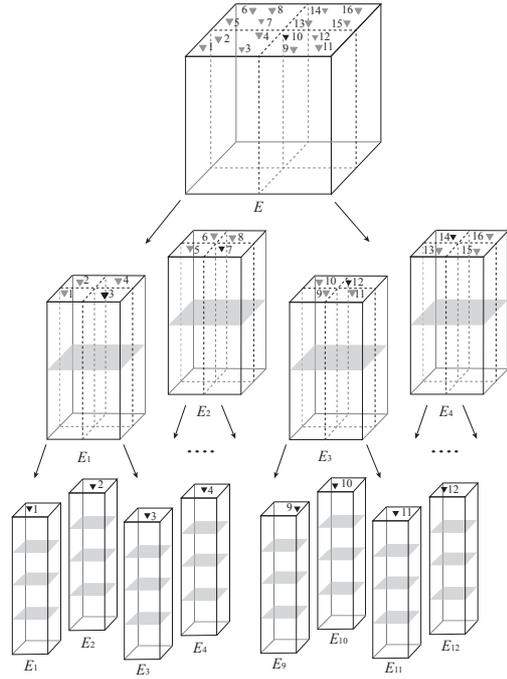


Fig. 3: Multi-resolution Evolving Tomography

and for E_p the following subsystem is constructed on a landlord,

$$\mathbf{A}_p \mathbf{s}_p = \mathbf{t}_p$$

where $1 \leq p \leq P_r$.

Initialize line 1-4: Each node initialize its ID, the resolution level q and initial dimension d . Besides, each node initializes a landlord list $\{\mathcal{H}_1, \mathcal{H}_2, \dots, \mathcal{H}_q\}$ where \mathcal{H}_{r+1} is a node list $\{h_p | 1 \leq p \leq P_r\}$ and h_p indicates the landlord for partition p at the $(r+1)$ -th resolution. This tells each node where to send the partial ray information. Then the resolution and partition parameters and a slowness reference model for ray tracing are initialized.

Initialize line 5-8: Set the landlord list as \mathcal{H}_1 (only one landlord in it). If this node is the landlord, it also initializes a ray counter and a time out threshold T_{th} which controls the waiting time for a landlord to receive partial ray information from other nodes. Because it is hard to know how many partial rays will be received by the landlord since the event activity is unpredictable.

Repeat line 1-3: After the initialization, each node will act based on the event detection and message reception. Once an event is detected by a node, either a landlord or a common node, it will trace the ray path (as assumed in previous discussion, the event location has already been estimated in the network and every node knows it). Then the node computes and sends the partial ray information to the corresponding landlords according the landlord list \mathcal{H}_{r+1} .

Repeat line 4-19: If the node is a landlord in partition p' , when it receives the first partial ray for current resolution, the node will start a timer T_c . Otherwise, if the timer is less than the threshold T_{th} , the landlord will add the received partial ray

Algorithm 1 Distributed Multi-resolution Evolving Tomography

```

Initialize
1: Node ID  $id$ , the resolution level  $q$  and dimension  $d$ 
2: Landlord lists  $\{\mathcal{H}_1, \mathcal{H}_2, \dots, \mathcal{H}_q\}$ 
3:  $r \leftarrow 0, Q_r \leftarrow d \times d \times d, P_r \leftarrow 1$ 
4: Slowness reference model  $\mathbf{s}^0$  of resolution  $Q_r$ 
5: Set landlord list to  $\mathcal{H}_1, p' \leftarrow$  the partition index this node locates
6: if  $id$  is equal to  $h_{p'}$ 
7:    $rc \leftarrow 0$ , set the time out threshold  $T_{th}$ 
8: endif
Repeat
1: Upon the detection of an event
2: Trace the ray path  $a_l$ , compute  $\bar{a}_{l,p}$  and  $t_{l,p}$  for  $1 \leq p \leq P_r$ 
3: Send  $\bar{a}_{l,p}$  and  $t_{l,p}$  to landlord  $h_p$ .
4: Upon the reception of  $\bar{a}_{l,p}$  and  $t_{l,p}$ 
5:   if  $id$  is equal to  $h_p$ 
6:     if  $rc$  is equal to 0
7:        $rc \leftarrow rc + 1$ , start timer  $T_c$ 
8:     else
9:       if  $T_c < T_{th}$ 
10:        Add  $\bar{a}_{l,p}$  and  $t_{l,p}$  to  $\mathbf{A}_{p'}\mathbf{s}_{p'} = \mathbf{t}_{p'}$ 
11:      else
12:        Solve least-squares problem  $\mathbf{A}_{p'}\mathbf{s}_{p'} = \mathbf{t}_{p'}$ 
13:        Broadcast  $\mathbf{s}_{p'}$  to all other nodes
14:         $rc \leftarrow 0$ , reset  $T_c$ , clear system  $\mathbf{A}_{p'}\mathbf{s}_{p'} = \mathbf{t}_{p'}$ 
15:      endif
16:    endif
17:   else
18:     Transfer  $\bar{a}_{l,p}$  and  $t_{l,p}$  to  $h_p$ 
19:   endif
20: Upon the reception of  $\mathbf{s}_p$  from landlord  $h_p$ 
21: Update the corresponding part of  $\mathbf{s}^0$  with  $\mathbf{s}_p$ 
22: if all  $\mathbf{s}_p (1 \leq p \leq P_r)$  have been received
23:   if  $r + 1$  is equal to  $q$ 
24:     TERMINATE
25:   else
26:      $r \leftarrow r + 1, Q \leftarrow 2^r d \times 2^r d \times 2^r d, P \leftarrow 4^r$ 
27:     Increase the resolution of  $\mathbf{s}^0$  to  $Q$ 
28:     Partition the tomography model into  $P_r$  parts
29:     Set landlord list to  $\mathcal{H}_{r+1}$ 
30:      $p' \leftarrow$  the partition index this node locates
31:     if  $id$  is equal to  $h_{p'}$ 
32:        $rc \leftarrow 0$ , set time out threshold  $T_{th}$ 
33:     endif
34:   endif
35: endif

```

into the subsystem $\mathbf{A}_{p'}\mathbf{s}_{p'} = \mathbf{t}_{p'}$. Once the timer T_c is out, the node will compute the partial slowness perturbation $\mathbf{s}_{p'}$ from constructed $\mathbf{A}_{p'}\mathbf{s}_{p'} = \mathbf{t}_{p'}$. Then the landlord broadcasts $\mathbf{s}_{p'}$ to all other nodes and reset the parameters. If the node is not a landlord, it will transfer the received partial ray information to the corresponding landlord.

Repeat line 20-35: Once a node receives the slowness perturbation \mathbf{s}_p from landlord h_p . It will update the corresponding part of the slowness reference model \mathbf{s}^0 with \mathbf{s}_p . After the partial slowness perturbations from all landlords have been received, i.e., the entire slowness reference model has been updated. The algorithm will TERMINATE if the required resolution is met; otherwise, the node will set r to $r + 1$, calculate current Q_r and P_r , and increase the resolution of reference model \mathbf{s}^0 to Q_r . There are different ways to increase the resolution of the model, e.g., all the blocks in higher resolution just use the slowness value of the block it belongs

to in the lower resolution. Then each node will partition the model into P_r parts. Also, the node will set the appropriate landlord list. If the node is a landlord at $(r + 1)$ -th resolution, it will initial a ray counter and set a timeout threshold (the threshold is larger in higher resolution since more rays are needed for higher resolution tomography computation).

Notice that the algorithm here is based on a cube tomography model, in reality the model is not always a cube and the partition may depend on the deployment of the sensor network. This algorithm can be applied to different models, it only needs to change the resolution evolving and partition scheme, i.e., how to set Q_r and P_r in resolution level r .

IV. EVALUATION AND VALIDATION

We implemented and evaluated the algorithm performance with system design in the CORE² and EMANE³ network emulators [1]. The experiment results validate that our proposed algorithm not only balances the computation load, but also achieves low communication cost and high data loss-tolerance.

The advantage of emulation is that the code developed over the emulator can be transplanted to a Linux-based device virtually without any modifications. This is because the emulation tool allocates for each network node a Linux virtual machine. We select CORE and EMANE as the development and evaluation platform, because the real sensor nodes in our VolcanoSRI project⁴ will be some low-power Linux-based devices such as android devices. CORE and EMANE will allows us to closely emulate the future deployment.

A. Experiment Setup and Implementation

Algorithm 1 is designed to compute the tomography in a wireless mesh network and requires both unicast and broadcast communication. On most volcanoes it is hard to rely on the pre-existing infrastructures (e.g. cellular infrastructure). Therefore, we adopt Disruption-Toleration Networks (DTN) techniques to maintain efficient and reliable end-to-end connectivity that spans many hops for data delivery. In our design, the data is buffered in a bundle and then transferred hop by hop in a store-and-forward manner until it arrives at the destination. Our implementation of DTN technique does not make any changes to underlying network services, it uses TCP for one-hop reliable bundle transfer, and uses OLSR routing protocol to indicate the next hop. We also implement a bundle cache component above the TCP/IP layer. This component can recover the lost bundle locally.

Besides unicast, we implement a delay-tolerant broadcasting service based on the NACK-Oriented Reliable Multicast (NORM) protocol⁵. Using NORM interface, one node can push a bundle reliably to its one-hop neighbors. Our cache component can receive and store this broadcast bundle, and rebroadcast it again with NORM, to the nodes that are two hops away, and so on so forth. A redundancy check module is developed in the cache component guarantees each node receives the same bundle at most once.

²<http://cs.itd.nrl.navy.mil/work/core/>

³<http://cs.itd.nrl.navy.mil/work/emane/>

⁴<http://sensorweb.cs.gsu.edu/?q=VolcanoSRI>

⁵<http://cs.itd.nrl.navy.mil/work/norm/>

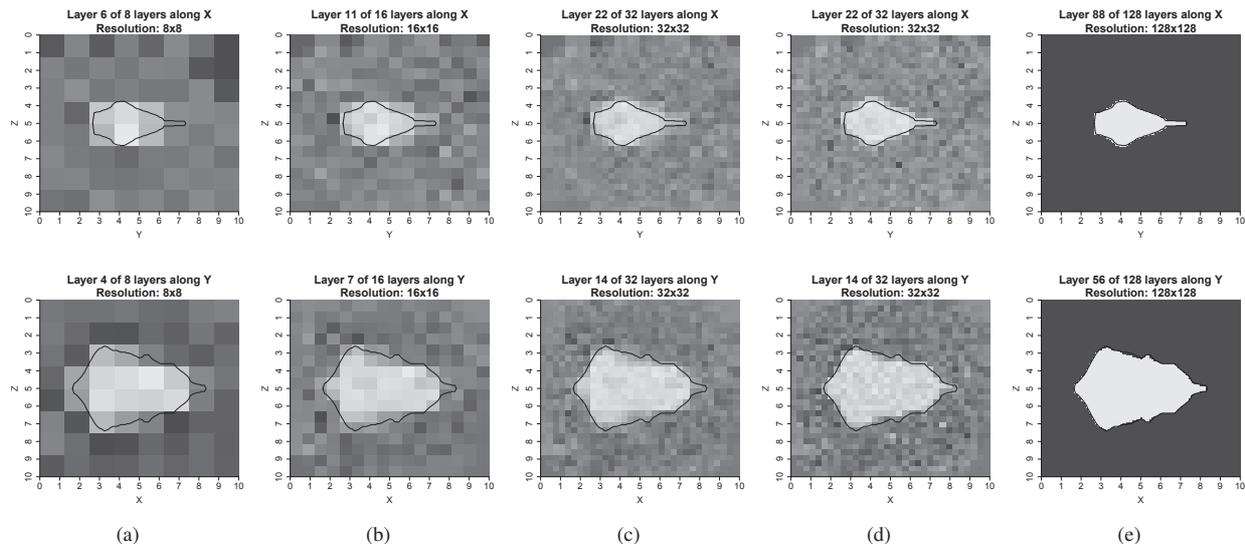


Fig. 4: Vertical slices of tomography model. Column (a), (b) and (c) are the results from DMET algorithm with resolution dimension 8, 16, 32 respectively while column (d) is the result of centralized algorithm and column (e) is the ground truth.

The evaluation of algorithm is illustrated by simulating seismic data on a synthetic model of resolution 128^3 consisting of a magma chamber (low velocity area) in a 10 km^3 cube. 100 stations are randomly distributed on top of the cube and form a mesh network as shown in Figure 5(a) (the black triangles indicate the landmarks in different resolution levels). 650 events are similarly distributed, Figure 5(b) which shows the vertical view of the event sources distribution.

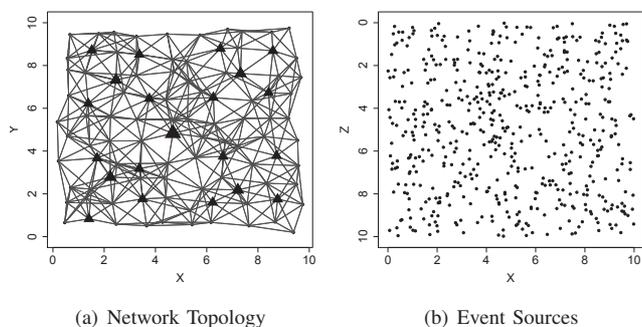


Fig. 5: Stations and Events Distribution

We evaluate the algorithm starting with resolution 8^3 and one partition, evolving to resolution 16^3 with 4 partitions and complete at resolution 32^3 with 16 partitions. 50, 100 and 400 events are generated for resolution from low to high to make sure the system is overdetermined. To simulate the event location estimation and ray tracing errors, a White Gaussian Noise is added to the travel time to generate the sensor node observations (arrival times).

To simulate the sensing behavior of the nodes in the network, two subnetworks are built in CORE emulator. One is the wireless mesh network in Figure 5(a) with EMANE to model the link and physical layer connectivity. The other

is a simple wired network where one generator node wired connects with each node in the mesh network. This generator node will generate event at random time, compute the travel times from this event to each node based on the ground truth, and send the event location and travel time to corresponding node with noise (suppose that the event location has been done as we discussed above). So the nodes in mesh network are blind to the event activity and thus simulate the sensing behavior.

The Bayesian version of ART method [4] is used in the experiments to compute the tomography. We use the relative update of the estimation between two sweeps (one sweep means that all the ray paths in the system are used once for estimation update) as the stopping criteria. Besides, a centralized collection scheme (one node collects all the ray information and perform centralized computation) is implemented in the emulator with the same data set to compare with our DMET algorithm. Notice that the Bayesian ART method solves the system $\mathbf{A}\mathbf{s} = \mathbf{t}$ to minimize $\|\mathbf{t} - \mathbf{A}\mathbf{s}\|^2 + \lambda^2 \|\mathbf{s}\|^2$ where λ is the trade-off parameter that regulates the relative importance we assign to models that predict the data versus models that have a characteristic, a priori variance.

B. Correctness and Accuracy

To validate the correctness and accuracy of the algorithm, we first visualize the tomography result in vertical slices in Figure 4. Each row of figures shows the same tomography slice on corresponding layer along with X or Y axes (the total layers of each figure is equal to the resolution dimension of the result). The black polygons give the cross section outline of the magma chamber surface. We can see that at the lowest resolution 8^3 , the result can hardly indicate the outline of the magma chamber since the block size is big, especially for the small cross section in the first row. But it gives a good start point for the higher resolution to further refine the result. At resolution dimension 16, the result can closely show the

outline of the magma chamber. The result in column (c) at resolution 32^3 is already very close to the centralized solution in column (d).

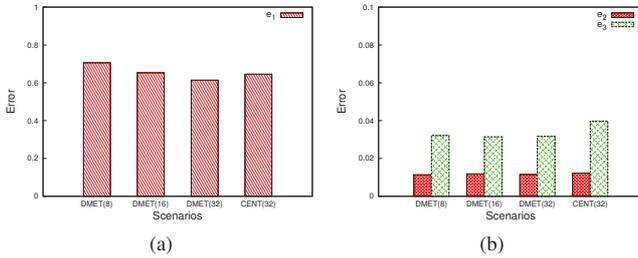


Fig. 6: Measures of Distance from Synthetic Model

Using \tilde{s} , s^* and \bar{s} to represent the synthetic model, the reconstructed model and the mean value of s^* respectively, we use the following quantitative measures of distance from the synthetic model provided in [10] to evaluate the estimation quality,

$$\begin{aligned}
 e_1 &= \left(\sum_{i=1}^n (\tilde{s}_i - s_i^*)^2 / \sum_{i=1}^n (s_i^* - \bar{s})^2 \right)^{\frac{1}{2}} \\
 e_2 &= \sum_{i=1}^n |\tilde{s}_i - s_i^*| / \sum_{i=1}^n |s_i^*| \\
 e_3 &= \max |\tilde{s}_i - s_i^*|
 \end{aligned}$$

These represent the normalized root mean squared distance, the average absolute value distance and the worst case distance respectively. The result is shown in Figure 6, DMET(8) means that the distance analysis of DMET algorithm with resolution 8^3 and CENT(32) indicates the result of centralized algorithm. First we observe that in DMET algorithm, the distances are decreasing along with the increase of the resolution. The distances in DMET(32) are even smaller than CENT(32), this is because that we use the relative update as the stop criteria in Bayesian ART method and the centralized algorithm may stop before the distance is small enough. This analysis can imply that the multi-resolution evolving scheme can give a good approximation on each resolution level for estimating partial travel times and the computation can approximate the centralized solution while not diverging in the local computation in each partition.

C. Communication and Computation

From the experiment result and analysis, it is validated that DMET gives a good result to approximate the tomography compared with the centralized algorithm. Next, we will compare the communication and computation cost between DMET and the centralized algorithms. Since the raytracing algorithm is very expensive to perform, we assume that each node does raytracing itself and sends the indexed ray path to a base station (centralized) or to the corresponding landlords. Also, after centralized computation done on the base station, it will broadcast the model to the network for future raytracing since the whole tomography process is iterative.

Figure 7 gives the communication and computation cost analysis where DMET(T) is the total cost for DMET algorithm, CENT(M) and CENT(C) are the cost of centralized

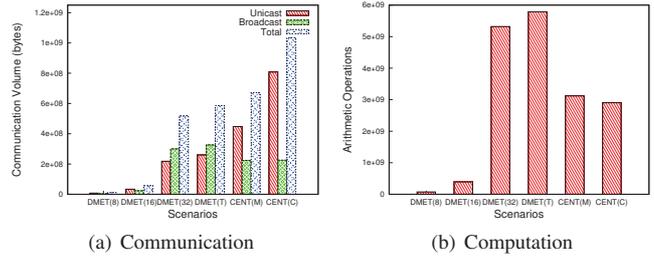


Fig. 7: Communication and computation cost analysis

algorithm when the base station is in the center and the corner of the network respectively. In Figure 7(a), we can see that the total unicast cost of DMET is much less than centralized algorithm (about one third of CENT(C)) since the centralized data collection will cause more interference and congestion when the data rate is high. At the same time, the broadcast cost of DMET is more than centralized algorithm because multiple nodes need to broadcast in DMET. To count the arithmetic operations in the computation of the base station and landlords, Figure 7(c) gives the computation cost for centralized algorithm and DMET. The total cost of DMET is higher than centralized since we partition and distribute the system to landlords, and the computation on landlords are entirely local and it is lack of the global information to constraint the problem for fast convergence.

The experiment results above have shown that the total communication cost of DMET is less than the centralized algorithm while the total computation cost is higher. Next, we count the communication and computation cost on each node in the network to show that DMET balances the communication and distribute the computation load in the network. Figure 8(a), (b) and (c) visualize the communication cost on each node in the network for 3 different scenarios with heat maps. From Figure 8(a) and (b), we can see that the communication cost in centralized scenario for the nodes near the base station (either in the center or corner) are much higher than other nodes since all the messages need to be delivered through them. The communication cost on each node in our algorithm shown in Figure 8(c) is more balanced. In Figure 8(d), the first column is the computation load on node 1 for CENT(C) algorithm, all other columns are the computation load on corresponding landlords of DMET algorithm. We can see that although the total computation cost of DMET is higher but the computation load is much more balanced.

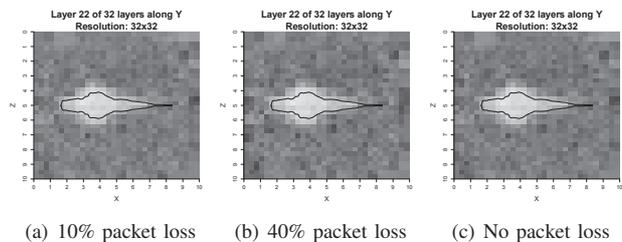


Fig. 9: Vertical slices of tomography model

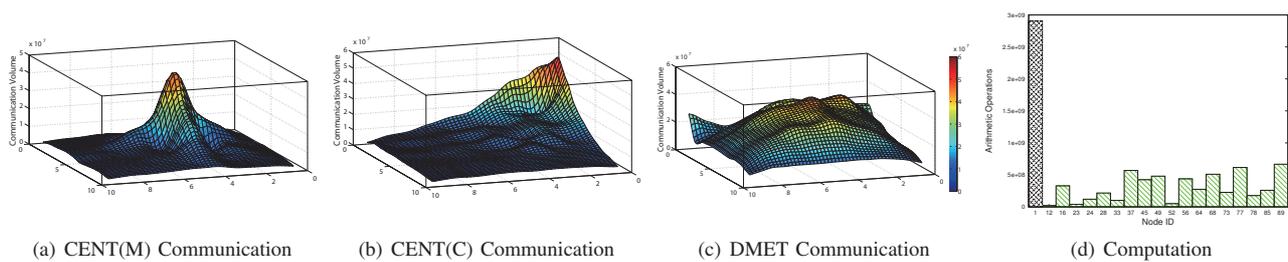


Fig. 8: Communication and Computation Load Balance

D. Data loss-tolerance and Robustness

Next we did an evaluation on the robustness of DMET. The algorithm runs with the same data set and two different packet loss ratios of 10% and 40% set in the emulator. Figure 9 shows the rendering of one vertical slice along Y axes. We can see that with 10% or even 40% packet loss, compared with the result without packet loss, It is hard to tell the differences. This validate the robustness of DMET algorithm which can be tolerant to a severe packet loss.

V. CONCLUSION

In this paper, we presented an innovative multi-resolution evolving tomography algorithm that distribute and balance the tomography inversion computation load to the network, while computing real-time high-resolution 3D tomography in the network. The experimental evaluation also showed that our proposed algorithm not only balances the computation load, but also achieves low communication cost and high data loss-tolerance.

REFERENCES

- [1] J. Ahrenholz, T. Goff, and B. Adamson. Integration of the CORE and EMANE Network Emulators. In *MILITARY COMMUNICATIONS CONFERENCE, 2011 - MILCOM 2011*, pages 1870–1875, 2011.
- [2] L. Geiger. Probability method for the determination of earthquake epicenters from the arrival time only. *Bull.St.Louis.Univ.*, 8:60–71, 1912.
- [3] D. Gordon and R. Gordon. Component-Averaged Row Projections: A Robust, Block-Parallel Scheme for Sparse Linear Systems. *SIAM Journal on Scientific Computing*, 27(3):1092–1117, 2005.
- [4] G. T. Herman, H. Hurwitz, A. Lent, and H.-P. Lung. On the Bayesian Approach to Image Reconstruction. *Information and Control*, 42:60–71, 1979.
- [5] H. M. Iyer and P. B. Dawson. *Imaging volcanoes using teleseismic tomography*. Chapman and Hall, 1993.
- [6] S. Kaczmarz. Angenäherte Auflösung von Systemen linearer Gleichungen. *Bulletin International de l'Académie Polonaise des Sciences et des Lettres*, 35:355–357, 1937.
- [7] J. M. Lees. The magma system of Mount St. Helens: non-linear high-resolution P-wave tomography. *Journal of Volcanology and Geothermal Research*, 53:103–116, 1992.
- [8] J. M. Lees. Seismic tomography of magmatic systems. *Journal of Volcanology and Geothermal Research*, 167(1-4):37–56, 2007.
- [9] J. M. Lees and R. S. Crosson. Tomographic Inversion for Three-Dimensional Velocity Structure at Mount St. Helens Using Earthquake Data. *Journal of Geophysical Research*, 94(B5):5716–5728, 1989.
- [10] J. M. Lees and R. S. Crosson. Bayesian Art versus Conjugate Gradient Methods in Tomographic Seismic Imaging: An Application at Mount St. Helens, Washington. *Institute of Mathematical Statistics*, 20:186–208, 1991.
- [11] J. M. Lees, N. Symons, O. Chubarova, V. Gorelchik, and A. Ozerov. Tomographic images of kluichevskoi volcano p-wave velocity. *AGU Monograph*, 172:293–302, 2007.
- [12] J. M. Lees and H. Wu. P wave anisotropy, stress, and crack distribution at Coso geothermal field, California. *Journal of Geophysical Research*, 104(B8):17955–17973, 1999.
- [13] J. M. Lees and H. Wu. Poisson's ratio and porosity at Coso geothermal area, California. *Journal of Volcanology and Geothermal Research*, 95(1-4):157–173, 2000.
- [14] G. Mateos, I. D. Schizas, and G. B. Giannakis. Performance Analysis of the Consensus-Based Distributed LMS Algorithm. *EURASIP Journal on Advances in Signal Processing*, 2009, 2010.
- [15] S. C. Moran, J. M. Lees, and S. D. Malone. P wave crustal velocity structure in the greater Mount Rainier area from local earthquake tomography. *Journal of Geophysical Research*, 104(B5):10775–10786, 1999.
- [16] S. Ohmi and J. M. Lees. Three-dimensional P- and S-wave velocity structure below Unzen volcano. *Journal of Volcanology and Geothermal Research*, 65(1-2):1–26, 1995.
- [17] R. A. Renaut. A Parallel Multisplitting Solution of the Least Squares Problem. *Numerical Linear Algebra with Applications*, 5(1):11–31, 1998.
- [18] A. H. Sayed and C. G. Lopes. Distributed Recursive Least-Squares Strategies Over Adaptive Networks. In *Signals, Systems and Computers, 2006. ACSSC '06. Fortieth Asilomar Conference on*, pages 233–237, Nov. 2006.
- [19] I. D. Schizas, G. Mateos, and G. B. Giannakis. Distributed LMS for consensus-based in-network adaptive processing. *IEEE Transactions on Signal Processing*, 57(6):2365–2382, 2009.
- [20] L. Shi, G. Kamath and W.-Z. Song. Distributed Least-Squares Iterative Methods in Networks: A Survey. <http://sensorweb.cs.gsu.edu/sites/default/files/publication/PDF/report/DistributedLS.pdf>.
- [21] R. Sleeman and T. van Eck. Robust automatic P-phase picking: an on-line implementation in the analysis of broadband seismogram recordings. *Physics of the Earth and Planetary Interiors*, 113(1-4):265–275, 1999.
- [22] W.-Z. Song, R. Huang, M. Xu, A. Ma, B. Shirazi, and R. Lahusen. Air-dropped Sensor Network for Real-time High-fidelity Volcano Monitoring. In *The 7th Annual International Conference on Mobile Systems, Applications and Services (MobiSys)*, June 2009.
- [23] H. Straková, W. N. Gansterer, and T. Zemen. Distributed QR factorization based on randomized algorithms. In *PPAM'11 Proceedings of the 9th international conference on Parallel Processing and Applied Mathematics - Volume Part I*, pages 235–244, 2012.
- [24] R. Tan, G. Xing, J. Chen, W. Song, and R. Huang. Quality-driven Volcanic Earthquake Detection using Wireless Sensor Networks. In *The 31st IEEE Real-Time Systems Symposium (RTSS)*, San Diego, CA, USA, 2010.
- [25] S.-Y. Tu and A. H. Sayed. Diffusion Strategies Outperform Consensus Strategies for Distributed Estimation over Adaptive Networks. *To appear in IEEE Transactions on Signal Processing*, 2012.
- [26] A. L. Vesnaver, F. Accaino, G. Bohm, G. Madrussani, J. Pajchel, G. Rossi, and G. D. Moro. Time-lapse tomography. *Geophysics*, 68(3):815–823, 2003.
- [27] G. P. Waite and S. C. Moranb. VP Structure of Mount St. Helens, Washington, USA, imaged with local earthquake tomography. *Journal of Volcanology and Geothermal Research*, 182(1-2):113–122, 2009.
- [28] H. Wu and J. M. Lees. Attenuation structure of Coso geothermal area, California, from wave pulse widths. *Bulletin of the Seismological Society of America*, 86(5):1574–1590, Oct. 1996.
- [29] H. Wu and J. M. Lees. Three-dimensional P and S wave velocity structures of the Coso Geothermal Area, California, from microseismic travel time data. *Journal of Geophysical Research*, 104(B6):13217–13233, 1999.
- [30] L. T. Yang and R. P. Brent. Parallel MCGLS and ICGLS Methods for Least Squares Problems on Distributed Memory Architectures. *The Journal of Supercomputing*, 29(2):145–156, 2004.