

5-2019

Distributed Compressive Sensing Algorithm for Photoacoustic Tomography

Maha AbdulWahab Hassan Shehada

Follow this and additional works at: https://scholarworks.uaeu.ac.ae/electric_theses



Part of the [Engineering Commons](#)

Recommended Citation

Hassan Shehada, Maha AbdulWahab, "Distributed Compressive Sensing Algorithm for Photoacoustic Tomography" (2019). *Electrical Engineering Theses*. 7.
https://scholarworks.uaeu.ac.ae/electric_theses/7

This Thesis is brought to you for free and open access by the Electrical Engineering at Scholarworks@UAEU. It has been accepted for inclusion in Electrical Engineering Theses by an authorized administrator of Scholarworks@UAEU. For more information, please contact fadl.musa@uaeu.ac.ae.

UAEU



جامعة الإمارات العربية المتحدة
United Arab Emirates University

United Arab Emirates University

College of Engineering

Department of Electrical Engineering

DISTRIBUTED COMPRESSIVE SENSING ALGORITHM FOR
PHOTOACOUSTIC TOMOGRAPHY

Maha AbdulWahab Hassan Shehada

This thesis is submitted in partial fulfilment of the requirements for the degree of
Master of Science in Electrical Engineering

Under the Supervision of Dr. Imad Barhumi

May 2019

Declaration of Original Work

I, Maha AbdulWahab Hassan Shehada, the undersigned, a graduate student at the United Arab Emirates University (UAEU), and the author of this thesis entitled "*Distributed Compressive Sensing Algorithm for Photoacoustic Tomography*", hereby, solemnly declare that this thesis is my own original research work that has been done and prepared by me under the supervision of Dr. Imad Barhumi in the College of Engineering at UAEU. This work has not previously been presented or published, or formed the basis for the award of any academic degree, diploma or a similar title at this or any other university. Any materials borrowed from other sources (whether published or unpublished) and relied upon or included in my thesis have been properly cited and acknowledged in accordance with appropriate academic conventions. I further declare that there is no potential conflict of interest with respect to the research, data collection, authorship, presentation and/or publication of this thesis.

Student's Signature: _____



Date: 29/9/2019

Copyright © 2019 Maha AbdulWahab Shehada
All Rights Reserved

Approval of the Master Thesis

This Master Thesis is approved by the following Examining Committee Members:

- 1) Advisor (Committee Chair): Dr. Imad Barhumi

Title: Associate Professor

Department of Electrical Engineering

College of Engineering

Signature Imad B.

Date 30/6/2019

- 2) Member (Internal Examiner): Dr. Falah Awwad

Title: Associate Professor

Department of Electrical Engineering

College of Engineering

Signature Falah Awwad

Date 30/6/2019

- 3) Member (External Examiner): Dr. Jiri Jaros

Title: Associate Professor

Department of Electrical and Computer Engineering


Institution: Brno University of Technology, Czech Republic

Signature Jiri Jaros

Date 30/6/2019

This Master Thesis is accepted by:

Dean of the College of Engineering: Professor Sabah Alkass

Signature  Date 24/9/2019

Dean of the College of the Graduate Studies: Professor Ali Almarzouqi

Signature  Date 29/9/2019

Copy 8 of 8

Abstract

Biomedical imaging techniques are playing an essential role in diagnosing different kinds of diseases, which always motivates the search for improving their sensitivity and accuracy. Photoacoustic Tomography (PAT) is one of the most powerful techniques. PAT has many advantages as it is less expensive and faster than Magnetic Resonance Imaging (MRI). It combines the advantages of optical imaging and ultrasound imaging as it provides high contrast, high penetration and high resolution images for biological tissues. Also, it uses non-ionizing radiation which is very safe for human health. The main challenge in PAT is that human tissues can be exposed only to a limited amount of radiation, so a full-view of PAT requires many transducers and a great number of measurements. This thesis aims to develop an efficient reconstruction algorithm of Photoacoustic (PA) images that uses few number of transducers, few number of measurements and offers low computational complexity while maintaining high quality of recovered images.

The proposed reconstruction algorithm depends on Compressive Sensing (CS) theory which is a signal processing technique that is capable of forming a full-view PAT images (under certain prerequisites) with few number of measurements. The proposed algorithm solves the CS problem using a distributed and parallel implementation of the Alternating Direction Method of Multipliers (ADMM). ADMM is a well-known method for solving convex optimization problems. A group of local processors that work in parallel with one global processor are used to form the images. The iterative algorithm of ADMM is distributed over local processors in such a way perfect reconstruction of images is possible.

Simulation results show that the proposed algorithm is powerful and successful in reconstructing different kinds of PA images with very high quality and significantly reduced computational complexity. Reducing the computational complexity is reflected on a much lower reconstruction time. Also, the algorithm requires lower cost and shorter acquisition time since the CS theory is used which allow the recovery of images from few number of samples and sensors. Although the idea of distributed ADMM has been introduced before in literature but to the best of our knowledge, this is the first work to apply distributed ADMM method in recovering photoacoustic images by distributing the iterative algorithm among multiple processors working in parallel.

Keywords: ADMM, PAT, compressive sensing, BP, distributed implementation, multiple processors.

Title and Abstract (in Arabic)

تطوير خوارزمية تعتمد على نظرية الاستشعار المضغوط و تطبيقها على التصوير المقطعي الصوتي

الملخص

يعتبر التصوير الطبقي الصوتي (Photoacoustic Tomography) نوع من أهم أنواع أجهزة التصوير الطبية و هو يعتمد على استعمال الليزر (ضوء) لتحفيز أنسجة الجسم على اطلاق موجات صوتية تعبر عن خصائص هذه الأنسجة. يعتبر التصوير الصوتي فعال و ذو دقة عالية و لديه من الخصائص ما يجعله ينافس الأنواع الأخرى من أجهزة التصوير الطبي. فهو يدمج بين مميزات الامتصاص الضوئي (Optical Imaging) و التصوير باستعمال الموجات فوق صوتية (Ultrasound Imaging) للحصول على صور ذات دقة عالية و على عمق كبير نسبياً. علاوة على ذلك، التصوير الصوتي يعتبر أسرع بكثير من التصوير باستعمال الرنين المغناطيسي (MRI) بالإضافة الى انه ذو تكلفة منخفضة. الأشعة المستعملة في التصوير الصوتي غير متأينة و بالتالي هي آمنة جداً على صحة الانسان. لقد أظهرت الدراسات الحديثة أن التصوير الصوتي فعال في مجالات طبية عدة منها : تحليل و مراقبة الأورام، تصوير وظائف الدماغ، تصوير أوعية الدم، و التصوير الداخلي للأوعية الدموية.

يجب أن لا تتعرض أنسجة جسم الانسان لكميات كبيرة من الأشعة حتى و إن كانت غير متأينة و هذا يمثل التحدي الأكبر لمجال التصوير الصوتي. لذلك يستعمل عادة عدد كبير جداً من أجهزة الاستشعار للحصول على صورة مقطعية كاملة و دقيقة للمنطقة المراد تشخيصها. و لكن هذا يزيد من تكلفة التصوير و يزيد من المدة اللازمة لتشكيل الصورة. إن الهدف من هذه الرسالة هو تطوير خوارزمية قادرة على تشكيل صور صوتية كاملة و ذات دقة عالية باستعمال عدد قليل من أجهزة الاستشعار، و في نفس الوقت غير معقدة حسابياً.

الخوارزمية المطروحة في هذه الرسالة تعتمد على نظرية الاستشعار المضغوط (Compressive Sensing). هذه النظرية تحت شروط معينة فعالة جداً في التقليل من عدد أجهزة الاستشعار اللازمة للحصول على صورة صوتية كاملة. معظم التطبيقات على هذه النظرية تستعمل جهاز (processor) مركزي واحد لتشكيل الصور. في هذه الرسالة يتم تشكيل الصور باستعمال عدد من الأجهزة المحلية و جهاز مركزي واحد. كل جهاز محلي مسؤول عن حل جزء من الخوارزمية المتعلق بعدد قليل جداً من أجهزة الاستشعار و من ثم ارسال الحلول المحلية للجهاز المركزي.

لقد أظهرت نتائج هذه الرسالة فعالية الخوارزمية المطروحة في التقليل من التعقيدات الحسابية المطلوبة من الجهاز المركزي لتشكيل مختلف الصور الصوتية و هذا قد انعكس على تشكيل الصور بوقت قصير جداً. بالإضافة إلى أن عدد أجهزة الاستشعار اللازمة قل بشكل كبير و الصور المسترجعة ذات دقة عالية.

مفاهيم البحث الرئيسية: استشعار مضغوط، عمليات حسابية موزعة، التصوير الطبقي الصوتي.

Acknowledgements

I would like to express my sincere gratitude to my supervisor, Dr. Imad Barhumi and Dr.Hanan Altous who have been tremendous mentors for me. I thank them for their encouragement, patience, motivation, help, guidance and support during the development of this work. They were the first motivators to study my master's degree.

My very profound gratitude goes to my mother Ahlam, my father AbdulWahab and my husband Osama for their continuous encouragement and unfailing support throughout the years of studying and working on this thesis. They were always available for me with their moral and emotional support. Their unconditional love and patience have been priceless. Their persistent help was at the end what made this work possible.

I would like also to express deep thanks to my sisters and brothers Nada, Khalid, Ala'a and Ahmed for their continuous moral support. My gratitude is extended to my friends Shaima, Ayshathul, Raghad and Maitha who made my study journey more enjoyable and interesting.

Dedication

To my beloved parents and husband

Table of Contents

Title	i
Declaration of Original Work	ii
Copyright	iii
Approval of the Master Thesis.....	iv
Abstract	vi
Title and Abstract (in Arabic)	viii
Acknowledgements	x
Dedication	xi
Table of Contents	xii
List of Tables	xiv
List of Figures	xv
List of Abbreviations	xvi
Chapter 1: Introduction	1
1.1 Overview	1
1.2 Statement of the Problem	2
1.3 Research Objectives	3
1.4 Thesis Organization.....	4
Chapter 2: Background	5
2.1 Photoacoustic Tomography.....	5
2.1.1 Advantages and applications.....	6
2.1.2 Photoacoustic wave generation	6
2.1.3 Photoacoustic wave equations.....	8
2.1.4 Pseudo-spectral matrix	9
2.1.5 Instrumentations	12
2.2 Compressive Sensing	13
2.2.1 Sparse Recovery Techniques	16
2.2.2 Basis Pursuit (BP)	16
2.2.3 Least Absolute Shrinkage Operator (LASSO).....	17
2.3 Alternating Direction Method of Multipliers (ADMM)	17
2.4 Related Works	20
2.4.1 Photoacoustic Tomography.....	20
2.4.2 Compressive sensing framework	21
2.4.3 Alternating direction method of multipliers.....	23
Chapter 3: Distributed Recovery Algorithm.....	25
3.1 Distributed ADMM BP	25

3.2 Relaxation Parameter	27
Chapter 4: Numerical Simulations and Results	30
4.1 Simulations Setup.....	30
4.2 ADMM BP Simulations	32
4.3 Distributed ADMM BP Simulations	36
4.4 Efficient Communication Links	40
Chapter 5: Conclusion and Future Work	43
5.1 Conclusion.....	43
5.2 Future Work	44
References	45
Appendix	49

List of Tables

Table 1: ADMM Basis Pursuit Algorithm	20
Table 2: The Proposed Distributed ADMM BP Algorithm.....	28
Table 3: The optical absorption coefficients of different breast tissue types.....	33
Table 4: Comparisons between three phantoms reconstruction while reducing the number of sensors (N_s)	35
Table 5: Distributed ADMM reconstructions using different no. of local units (M).....	38
Table 6: Distributed ADMM reconstructions using reduced number of samples	40
Table 7: Efficient communication links for the blood vessels phantom ($M = 4, 8$)	41

List of Figures

Figure 1: Principle of photoacoustic imaging	6
Figure 2: Inner and outer grids.....	11
Figure 3: Typical PAT system	13
Figure 4: Block Diagram of The Proposed Algorithm	29
Figure 5: Inner and outer grids with square sensors configuration.....	30
Figure 6: The measured pressure by one sensor using k -wave and the pseudo-spectral matrix (\mathbf{H} Matrix)	32
Figure 7: Original images and reconstructed images using ADMM	34
Figure 8: Reconstructed images using the proposed algorithm	38

List of Abbreviations

ADMM	Alternating Direction Method of Multipliers
ADMMBP	Alternating Direction Method of Multipliers Basis Pursuit
AMA	Alternating Minimization Algorithm
BP	Basis Pursuit
CS	Compressive Sensing
DOT	Diffuse Optical Tomography
FT	Fourier Transform
FFT	Fast Fourier Transform
HbR	Deoxy-hemoglobin
HbO ₂	Oxy-hemoglobin
LASSO	Least Absolute Shrinkage and Selection Operator
MATLAB	Matrix Laboratory
MRI	Magnetic Resonance Imaging
Nd:YAG	Neodymium-Doped yttrium aluminum garnet laser
NP-Hard	Non-deterministic Polynomial-time hardness
OCT	Optical Coherent Tomography
PA	Photoacoustic
PAT	Photoacoustic Tomography
RIP	Restricted Isometric Property
ROI	Region of Interest
SALSA	Split Augmented Lagrangian Shrinkage Algorithm
Ti:sapphire	Titanium Sapphire laser
USI	Ultrasound Imaging

Chapter 1: Introduction

1.1 Overview

Photoacoustic Tomography (PAT) is a powerful biomedical imaging modality that combines the advantages of ultrasound imaging and optical imaging. It breaks the spatial resolution limits associated with optical imaging such as Diffuse Optical Tomography (DOT) and Optical Coherent Tomography (OCT). Also, PAT is less expensive and much faster than Magnetic Resonance Imaging (MRI). Using PAT, biochemical parameters can be imaged with high resolution such as lipids, water, deoxy-hemoglobin (HbR) and oxy-hemoglobin (HbO₂) along with blood flow. Moreover, using a molecular contrast agent, highly specific molecular PAT can be realized. PAT is economical, can be made portable and uses non-ionizing radiation which is very safe for human health. Several clinical applications have found potential to use PAT such as breast imaging, joint imaging, intraoperative imaging, tumor vasculature imaging, brain imaging, and intravascular imaging [1].

The main challenge of biomedical imaging techniques including PAT is that human tissues can be exposed only to a limited amount of radiation, so a huge number of transducers and measurements are needed to form full-view Photoacoustic (PA) images. For PAT, an efficient solution to this challenge is to apply Compressive Sensing (CS) theory. Compressive sensing can be also referred to as compressive sampling or sparse signal recovery. It is a signal processing technique that can reconstruct a sparse signal or image accurately using few number of linear measurements. Sparse signals are those containing few number of nonzero elements. Many signals such as PA and audio signals are sparse either by nature or with respect

to another basis. CS has found its potential not only in imaging applications but also in radar and error correction. Different sparse recovery approaches are available such as greedy algorithms and convex optimization methods. One of the powerful algorithms for solving convex optimization problems including the CS problem is Alternating Direction Method of Multipliers (ADMM). ADMM can efficiently find a unique solution to CS optimization problem.

There are three significant traits that reconstruction approach or algorithm must possess. First, the algorithm should be fast. Second, it must provide uniform guarantee of performance which means it does not fail to recover any sparse signal. Third, the algorithm should have high stability meaning that if the signal to be recovered is perturbed slightly, the algorithm can still recover it with high accuracy. Stability of CS algorithm is essential in practice since the signal to be recovered may not be exactly sparse but close to being sparse such as the case of compressible signals. Compressible signals are those with coordinates decay according to power law. Also, the signals in practice are usually perturbed by noise.

1.2 Statement of the Problem

The advantages of PAT encourage improving the system performance by using an optimal and efficient reconstruction algorithm. Conventional reconstruction algorithms associated with CS and PAT rely on a centralized framework in which the whole measurements are processed using a central processor. Processing all measurements using a central processor may entail computational complexity especially in 3D PA images. Also, there is a gap between CS algorithms mentioned before. For example, convex optimization methods possess high stability and uniform

guarantee, but they require higher reconstruction time than greedy algorithms. On the other hand, greedy algorithms have lacked in stability and uniform guarantee of performance. Therefore, there is a need for reducing the computational complexity of processing the PA measurements in one processor and bridging the gap between sparse recovery algorithms. This can be achieved by developing a PAT recovery algorithm that uses an optimal number of measurements, with low computational complexity, uniform performance guarantee, fast run time and simultaneously maintains high quality of recovered images. The proposed algorithm is implemented using a distributed framework of ADMM algorithm. ADMM algorithm solves convex optimization problems, thus it provides high stability and uniform guarantee. The ADMM iterative algorithm is distributed over multiple local processors work in parallel with one global processor. This distributed framework reduces the computational complexity of the overall recovery algorithm which is reflected in a faster run time.

1.3 Research Objectives

This work aims to contribute to a growing research area of reconstructing PA images. Therefore, the main research objectives of this thesis are:

- 1) Implementing a reconstruction algorithm for PAT that has low computational complexity, uniform guarantee, uses few number of sensors, few number of measurements and provides high quality of recovered images.
- 2) Investigating the efficiency of the proposed algorithm using simulated images of different sparsity levels as well as using real numerical phantom.

1.4 Thesis Organization

This thesis is organized as follows. In Chapter 2, a background of PAT principals including its mathematical modeling and instrumentations are provided. Moreover, CS theory and ADMM algorithm are described along with some related works to PAT, CS and ADMM. The implementation of the proposed distributed ADMM Basis Pursuit (BP) algorithm is explained in Chapter 3. In Chapter 4, the numerical simulations and results are shown and discussed. Finally, the outcomes of this work and an insight to future works are concluded in Chapter 5.

Chapter 2: Background

In this chapter, PAT is introduced including its advantages, applications, mathematical modeling, forward operator matrix and instrumentations. Then Compressive Sensing (CS) theory along with some of its formulations and techniques are discussed. Later the well-known ADMM algorithm based on Basis Pursuit (BP) formulation is explained. After that, related works to PAT, CS and ADMM algorithm are presented. Finally, a discussion is made about the proposed distributed ADMM BP algorithm and how it enhances the available PA reconstruction algorithms.

2.1 Photoacoustic Tomography

Photoacoustic Tomography (PAT) is a biomedical imaging technique that depends on photoacoustic effect; which is the formation of sound waves following light absorption in a tissue sample. The PA effect principle is described in Figure1: a source of light (Laser pulses) is applied to tissues. The laser pulses heat the tissues and cause a localized pressure change and tissues thermoelastic expansion which at the end causes the acoustic waves to propagate from the tissues. Photoacoustic (PA) waves can be used to characterize the tissues and form clear images of them.

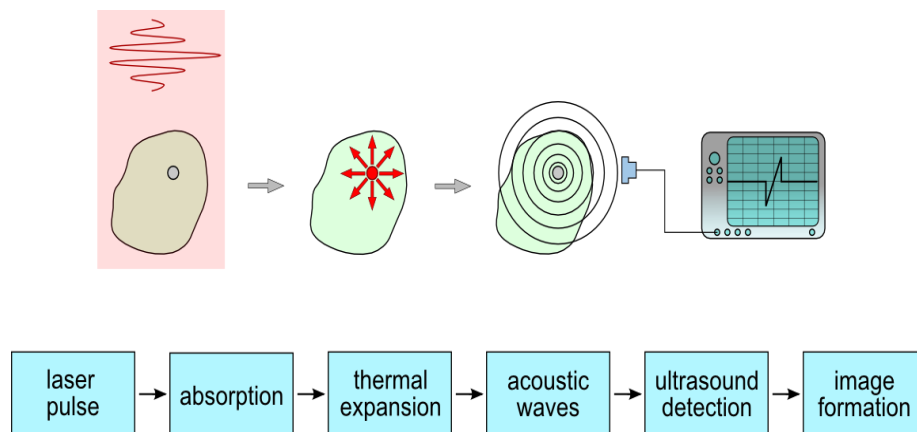


Figure 1: Principle of photoacoustic imaging

2.1.1 Advantages and applications

PAT has many advantages as it is less expensive, and faster than Magnetic Resonance Imaging (MRI). It combines the advantages of optical imaging and ultrasound imaging as it provides high contrast, high penetration and high resolution images for biological tissues [2]. Also, biochemical parameters such as oxy-hemoglobin (HbO_2), water (H_2O), and deoxy-hemoglobin (HbR) along with blood flow in tissues can be characterized in high resolution. PAT can be applied to many biomedical imaging fields such as: brain function imaging, tumor angiogenesis imaging, intravascular imaging and breast cancer imaging.

2.1.2 Photoacoustic wave generation

The generation of acoustic waves is usually done using short laser pulses of few nanoseconds in duration. The time scale in which the laser pulses must be delivered to enable tissues generate acoustic waves depends on two factors: the tissues physical characteristics and the time scale of energy dissipation. The time scale of

energy dissipation is represented as the stress (τ_s) and the thermal (τ_{th}) relaxation times. The stress time is given by [3]:

$$\tau_s = \frac{d_c}{c}, \quad (1)$$

where d_c is the dimension of the heated region or its spatial resolution, and c is the medium speed of sound (m/s). The thermal relaxation time is given by [3]:

$$\tau_{th} = \frac{d_c^2}{\alpha_{th}}, \quad (2)$$

where α_{th} is the thermal diffusivity (m^2/s). For soft tissues, α_{th} is in the range of $10^{-7} m^2/s$. Assuming that the dimension of an object is $1 mm$, the thermal relaxation time will be in the order of tens of seconds. However, the stress relaxation time is usually much smaller than thermal relaxation time. τ_s is on the order of few hundred nanoseconds for objects in the sub-mm or mm ranges. Therefore, to generate PA wave, the laser pulse width should be much shorter than thermal and stress relaxation times to ensure that heat conduction is negligible during laser pulses excitation and these are called the stress and thermal confinements. The temperature increase due to the absorbed laser pulses can be written as [3]:

$$T = \frac{\eta_{th} \mu_a F}{\rho C_v}, \quad (3)$$

where η_{th} is the percentage of light converted to heat, F is the optical fluence (J/cm^2), μ_a is the optical absorption coefficient (cm^{-1}), C_v is the specific heat capacity ($J/g \cdot K$), and ρ is the density (g/cm^3). If the thermal and stress confinements are met, one can find the initial pressure rise p_0 as [3] :

$$p_0 = \frac{\beta T}{\kappa} = \Gamma \eta_{th} \mu_a F, \quad (4)$$

where β is the thermal coefficient of volume expansion (K^{-1}), κ is the isothermal compressibility (Pa^{-1}), and Γ is called the Gruneisen parameter which is dimensionless and defined as [3]:

$$\Gamma = \frac{\beta}{\kappa\rho C_v}, \quad (5)$$

Γ and η_{th} are usually approximated as constants. To reconstruct an image we need to recover the initial pressure or in particular we need to recover μ_a since the other parameters in (4) are assumed to be constants [3].

2.1.3 Photoacoustic wave equations

After the initial pressure is generated, the acoustic waves start to propagate in the medium at the speed of sound. In a homogenous medium, the PA wave propagation can be expressed as [3]:

$$\frac{\partial^2 p(r,t)}{\partial t^2} - c^2 \nabla^2 p(r,t) = \Gamma \frac{\partial}{\partial t} H(r,t), \quad (6)$$

where $p(r,t)$ is the pressure signal, r is the spatial location, t is time, and $H(r,t)$ is the heat capacity defined as the thermal energy deposited per unit volume and per unit time which can be expressed as [3]:

$$H(r,t) = \rho C_v \frac{\partial T(r,t)}{\partial t}. \quad (7)$$

Using Green's functions, $p(r,t)$ can be obtained, providing the pressure signal at a transducer location r_s propagated from a source located at r over time t [3]:

$$p(r_s, t) = \frac{1}{4\pi c} \frac{\partial}{\partial t} \int \frac{p_0(r)}{|r_s - r|} \delta\left(\frac{t - |r_s - r|}{c}\right) dr, \quad (8)$$

where $p_0(r)$ is the initial pressure distribution based on spatial location r . After some mathematical simplifications, PAT equations can be modeled as a forward and inverse

problems. The forward problem is presented as the wave equation given in (8), while the inverse problem is presented as the initial pressure distribution $p_0(r)$ where the image is formed from. Forward and inverse problems can be written in a more compact form by assuming $\mathbf{x} = p_0(r)$, and $\mathbf{y} = p(r_s, t)$. The forward problem can be represented in a linear form as [4]:

$$\mathbf{y} = \mathbf{H}\mathbf{x}, \quad (9)$$

where \mathbf{H} is a matrix representing the forward operator of PA wave equations, \mathbf{x} is the initial pressure vector, and \mathbf{y} is the measurement vector of pressure. In this thesis, and from here on the compact form will be used to represent the initial pressure and the pressure waves.

2.1.4 Pseudo-spectral matrix

The construction of \mathbf{H} matrix depends on the adopted model, many forms of this matrix are found in literature [4], [5], [6]. In this thesis, we aim to simulate the PA measurements using k -wave MATLAB toolbox [7]. In order to reconstruct an image from simulated PA measurements, the forward operator matrix of PAT should give similar measurements as in the k -wave. So, a pseudo-spectral matrix has been chosen that gives accurate measurements as in k -wave. This matrix is implemented in the time domain as derived in [6]. To summarize the implementation of \mathbf{H} , one can start from the solution to the initial value problem in (6) [6]:

$$\left(\frac{\partial^2}{\partial t^2} - c^2 \nabla^2\right)p(r, t) = 0. \quad (10)$$

If the initial conditions were set to $p(\mathbf{r}, 0) = \Gamma h(\mathbf{r})$, and $\frac{\partial p}{\partial t}|_{t=0} = 0$, then the pseudo-spectral solution in Fourier space \mathbf{k} to the initial value problem at the time t and location \mathbf{r}' is given by:

$$p(\mathbf{r}', t) = \frac{1}{(2\pi)^3} \iint p_0(\mathbf{r}) e^{i\mathbf{k} \cdot (\mathbf{r}' - \mathbf{r})} \cos(ckt) d\mathbf{r} d\mathbf{k}, \quad (11)$$

where $k = |\mathbf{k}|$. Equation (11) is the basis of computing wave propagation for a particular time instant at all points in a plane, and it is used to construct the sensing matrix \mathbf{H} which is actually a pseudo-spectral matrix [6].

Consider to have two grids which are the imaging grid (inner grid) and the k-space grid (outer grid) as shown in Figure 2. The inner grid contains N_{in} grid points along each axis with coordinates x and y of the 2D grid, let's assume $\mathbf{r} = (x, y) = (m * d, n * d)$, where d is the grid spacing and $(m, n) \in [-N_{in}/2, N_{in}/2 - 1]$ are integers representing the inner index. The outer grid has N_{out} grid points along each axis. The Fourier transform frequency bins of the outer grid are given by:

$$\mathbf{k} = (k_x, k_y) = \frac{2\pi}{N_{out} * d} * (u, v), \quad (12)$$

where $(u, v) \in [-N_{out}/2, N_{out}/2 - 1]$ are integers representing the outer index. Then the Fourier transform matrix can be computed as [6]:

$$\mathbf{W}_{fwd}(i, j) = \frac{1}{N_{out}} e^{-\sqrt{-1}\mathbf{k}(i) \cdot \bar{\mathbf{r}}(j)}, \quad (13)$$

where $\mathbf{W}_{fwd} \in \mathbb{C}^{N_{out}^2 \times N_{in}^2}$, $\bar{\mathbf{k}}$ and $\bar{\mathbf{r}}$ are vectors resulting from vectorizing \mathbf{k} and \mathbf{r} respectively. The inverse Fourier transform matrix \mathbf{W}_{inv} is found only at sensor locations $\bar{\mathbf{r}}_s = (x_s, y_s)$. Assuming N_s sensor locations, \mathbf{W}_{inv} is given by [6]:

$$\mathbf{W}_{inv}(s, i) = \frac{1}{N_{out}} e^{-\sqrt{-1}\mathbf{k}(i) \cdot \bar{\mathbf{r}}_s(s)}, \quad (14)$$

where $\mathbf{W}_{inv} \in \mathbb{C}^{N_s \times N_{out}^2}$, and $s = 1, 2, \dots, N_s$ are the sensor indices. The Fourier inversion and wave propagation for a specific time instant is represented as \mathbf{K}_t , and it can be computed as:

$$\mathbf{K}_t = \mathbf{W}_{inv} \circ (\mathbf{1}\boldsymbol{\kappa}_t^T), \quad (15)$$

where $\boldsymbol{\kappa}_t = [\cos\{c\bar{\mathbf{k}}(1)t\}, \cos\{c\bar{\mathbf{k}}(2)t\}, \dots, \cos\{c\bar{\mathbf{k}}(N_{out}^2)t\}]^T$, $\mathbf{1}$ is a column vector of all ones with length N_s , and \circ represents element wise multiplication. One can construct matrix \mathbf{K} by stacking \mathbf{K}_t for different time instants. Assuming that \mathbf{K} matrix captures the response of sensors over the entire time steps, then the sensing matrix can be obtained by simple matrix multiplication [6]:

$$\mathbf{H} = \mathbf{K}\mathbf{W}_{fwd}, \quad (16)$$

where $\mathbf{H} \in \mathbb{C}^{N_s N_t \times N_{in}^2}$.

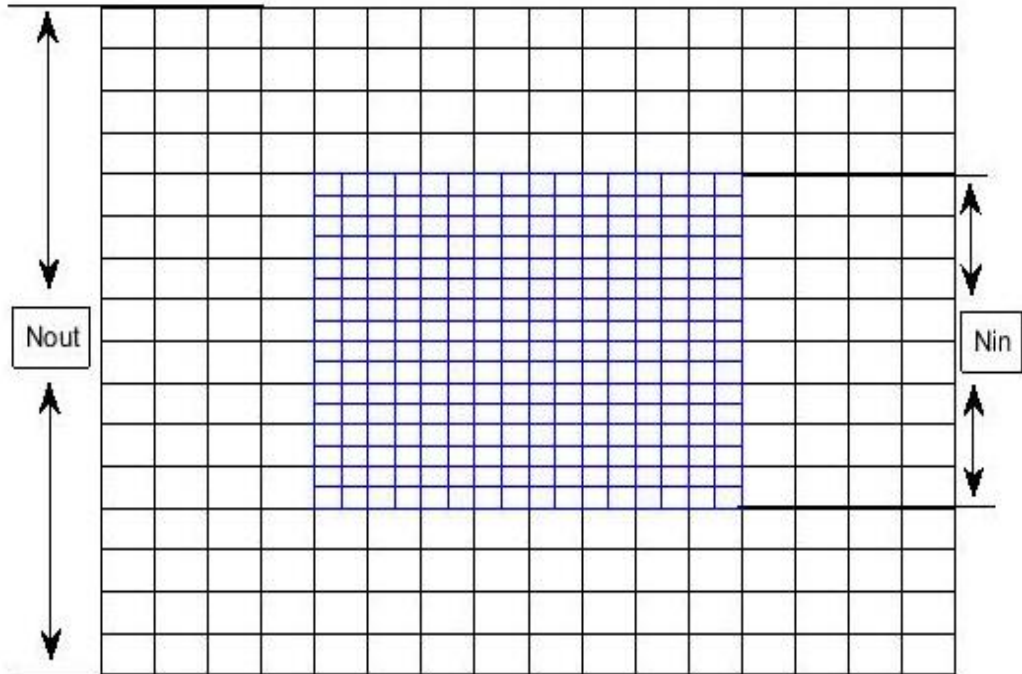


Figure 2: Inner and outer grids

2.1.5 Instrumentations

The instrumentations for PAT system consist of a source of short laser pulses, sensor scanning system and an acoustic signal detection system. Figure 3 shows a typical PAT system [1]. In this system, several types of pulsed laser could be used such as diode laser, Ti:Sapphire laser and Nd:YAG laser. An optical subsystem is used to couple the laser source with the object and generates acoustic waves. A single transducer is used to sense and receive the acoustic waves where the transducer and the object are immersed in a tank of water. A membrane is used to isolate the object which is here a rat from the water tank. A rotary system is used to rotate the transducer around the object. The transducer receives one set of data at multiple positions (e.g. 120 positions). This mechanical scanning system significantly increases the time needed for data acquisition. Many other transducer arrangements are possible such as linear array of transducers and circular arrangement of multiple transducers. There are many transducer types that can be used in PAT system. For example, the piezoelectric transducer which gives good results in detecting PA waves [8]. Also, ultrasound transducers can be used in PAT since the signal reception mechanism in Ultrasound Imaging (USI) and PAT are identical [9].

The acoustic signal received by the transducer is firstly amplified using a preamplifier and then amplified again using a receiver. The signal is then converted to digital signal using a data acquisition board and fed to the computer. The computer is responsible for reconstructing the image from these digital measurements. In Figure 3, a slice of the rat brain is shown which is recovered using the PAT system [1].

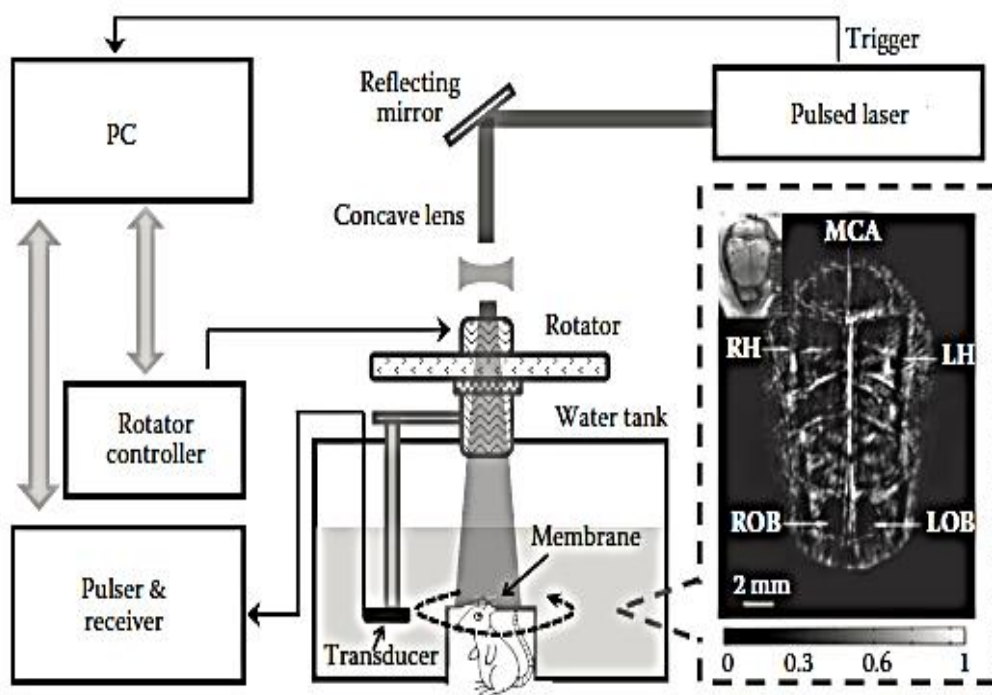


Figure 3: Typical PAT system

2.2 Compressive Sensing

Compressive Sensing (CS) theory is a signal processing technique for efficiently acquiring and reconstructing a sparse or a compressible signal. Sparse signal contains only few number of non-zero coefficients. However, compressible signal amplitude coefficients decrease rapidly if arranged in descending order as they decay with a power law. Many signals are compressible or sparse by nature, one of them is the photoacoustic signals. Thus, PA signals can be either recovered directly using compressive sensing, or they can be firstly transformed into another domain where they are sparser such as the Fourier domain, Wavelet domain, Curvelet domain ...etc.

CS is based on finding a solution of an optimization problem, where the number of unknowns is larger than the number of measurements. Based on CS theory, a small number of linear projections of a compressible signal contains enough information for reconstruction by directly sampling the signal to be recovered using a sparse representation. As, the number of measurements can be reduced for a given quality of reconstruction, CS theory can be used to form a full-view PAT image with less number of sensors and shorter acquisition times. Usually to reconstruct a signal we should satisfy the Nyquist criteria in which the sampling rate should be at least twice the modulating signal maximum frequency. However, using CS theory, the signal can be recovered in much lower rate than the sampling rate suggested by Nyquist. According to CS theory, the prerequisite for accurate reconstruction is the sparsity of the original signal \mathbf{x} , and the incoherency of the sensing modality [10].

The system matrix \mathbf{H} should satisfy certain properties. A strong property for exact reconstruction is the restricted isometric property (RIP). RIP typically holds for random matrices such as Gaussian, Bernoulli ...etc, but not for all deterministic matrices. RIP shows how well the distances between two columns in matrix \mathbf{H} are preserved by certain linear transformation. The matrix \mathbf{H} satisfies the RIP property for every K -sparse vector \mathbf{x} with constant $\delta_K \in (0,1)$ if [11]:

$$(1 - \delta_K)\|\mathbf{x}\|_2^2 \leq \|\mathbf{H}\mathbf{x}\|_2^2 \leq (1 + \delta_K)\|\mathbf{x}\|_2^2,$$

where K is the number of nonzero elements in \mathbf{x} . If the RIP property is satisfied, then the measurement vector \mathbf{y} corresponds only to one K -sparse vector \mathbf{x} (there are no two vectors \mathbf{x} that can give the same vector \mathbf{y}). In this way, the uniqueness of the solution is guaranteed. However, RIP property is NP-hard to compute [12]. A sufficient

condition for RIP is the mutual coherence of the sensing modality. The mutual coherence of a matrix can be computed easily, as it requires only $O(NM)$ operations, where N, M are the matrix number of columns and rows, respectively. Therefore, the incoherency of the sensing modality is considered as a prerequisite for CS theory instead of RIP. Mutual coherence measures the level of dependence between the columns of a matrix. Therefore, to provide incoherency, the mutual coherence should be as low as possible. It can be defined as follows:

$$\mu(\mathbf{H}) = \max_{i \neq j, 1 \leq i, j \leq N} \left| \frac{|h_j^* h_i|}{\|h_i\|_2 \|h_j\|_2} \right|,$$

where h_j denotes the j^{th} column of the matrix \mathbf{H} and h_j^* denotes its conjugate transpose. $\mu(\mathbf{H}) = 0$, if all the columns in the matrix \mathbf{H} are orthogonal. In case of CS, the number of rows N is lower than the number of columns M , thus the mutual coherence is strictly positive $\mu(\mathbf{H}) > 0$.

Mathematically, the sparsity of an image \mathbf{x} with N_{in}^2 pixels is defined as $\|\mathbf{x}\|_0 \ll N_{in}^2$, where $\|\mathbf{x}\|_0$ is the ℓ_0 norm defined as the number of non-zero elements in the vector \mathbf{x} . Compressive sensing theory tells us that if an image to be recovered is already sparse or can be transformed to a sparse image, then \mathbf{x} is the solution to the following optimization problem [4]:

$$\min_{\mathbf{x}} \|\mathbf{x}\|_0 \text{ s.t. } \mathbf{H} \mathbf{x} = \mathbf{y}. \quad (17)$$

ℓ_0 minimization is an NP-hard combinatorial problem which is computationally expensive to solve. However, if the image to be recovered is sufficiently sparse and under some conditions on the matrix \mathbf{H} , the solution of the ℓ_0 problem (17) can be

obtained by replacing the ℓ_0 norm with the ℓ_1 norm which is a convex optimization problem.

2.2.1 Sparse Recovery Techniques

In literature, several techniques are used to recover sparse signals, mainly convex optimization methods and greedy algorithms. Convex optimization methods have uniform guarantees of performance, which means they never fail to reconstruct any sparse signal. This is an important advantage over greedy algorithms. They are also stable which is significant in practice and thus they are applicable to real world problems. However, they have higher computational complexity compared to greedy algorithms. Greedy algorithms such as Orthogonal Matching Pursuit (OMP) [13] and Stagewise Orthogonal Matching Pursuit (SToMP) [14] compute the support of the signal iteratively but they do not provide the same guarantee of performance and stability as ℓ_1 - minimization methods. In this thesis, the focus will be on convex optimization methods since they provide higher guarantees and stability. Their computational complexity will be much improved using the proposed distributed implementation of the problem as will be discussed later. In the followings different formulations of CS problems are discussed.

2.2.2 Basis Pursuit (BP)

Basis Pursuit is formulated as a linear programming problem. It finds the sparse vector \mathbf{x} that has the smallest ℓ_1 norm and at the same time satisfies the equality constraint $\mathbf{H}\mathbf{x} = \mathbf{y}$. A vector \mathbf{x} can be recovered using BP formulation by solving the following equality-constrained optimization problem [10]:

$$\min_x \|\mathbf{x}\|_1 \text{ s.t. } \mathbf{H}\mathbf{x} = \mathbf{y}, \quad (18)$$

where $\|\mathbf{x}\|_1 = \sum_{n=1}^{N_{in}} |x_n|$ is the ℓ_1 norm.

2.2.3 Least Absolute Shrinkage Operator (LASSO)

LASSO is a method of solving ℓ_1 minimization problems similar to BP but with a quadratic constraint which can be formulated as:

$$\min_x \|\mathbf{x}\|_1 \text{ s.t. } \|\mathbf{H}\mathbf{x} - \mathbf{y}\|_2^2 < \epsilon^2, \quad (19)$$

where $\|\cdot\|_2$ is the ℓ_2 norm, and ϵ is the tolerance. BP and LASSO can be solved easily using convex optimization techniques such as interior point method which is implemented in CVX and ℓ_1 -MAGIC [15]. A fast implementation algorithm that allows to solve the problem is the ADMM which will be the focus of this thesis. ADMM allows for decoupling the problem into many sub-problems that can be solved in parallel.

2.3 Alternating Direction Method of Multipliers (ADMM)

ADMM is a very powerful algorithm and simple at the same time. It is well-suited to convex optimization problems and in particular to problems arising in machine learning. It combines the advantages of dual ascent method and the method of multipliers. In particular, it blends the dual ascent decomposability property with the method of multipliers convergence properties [16]. It can solve many problems efficiently, gives better results than other algorithms and is better suited for ℓ_1 norm problems. The main idea of ADMM is splitting the objective function into two objective functions; each depends on a distinct variable.

To apply ADMM to basis pursuit problem, an additional block of variables \mathbf{z} is added to the optimization variable. Following the ADMM formulation of the BP, problem (18) can be rewritten as [16]:

$$\min_{\mathbf{x}, \mathbf{z}} f(\mathbf{x}) + \|\mathbf{z}\|_1 \text{ s. t. } \mathbf{x} - \mathbf{z} = 0, \quad (20)$$

where $f(\mathbf{x})$ is an indicator function defined as:

$$f(\mathbf{x}) = \begin{cases} 0 & \mathbf{H}\mathbf{x} = \mathbf{y} \\ \infty & \text{otherwise} \end{cases}. \quad (21)$$

To solve the optimization problem (20), the augmented Lagrangian function is written as:

$$L_\rho(\mathbf{x}, \mathbf{z}, \boldsymbol{\mu}) = f(\mathbf{x}) + \|\mathbf{z}\|_1 + \boldsymbol{\mu}^T(\mathbf{x} - \mathbf{z}) + \frac{\rho}{2}\|\mathbf{x} - \mathbf{z}\|_2^2, \quad (22)$$

where $\boldsymbol{\mu}$ is a dual variable, and ρ is the augmented Lagrangian parameter. Based on the ADMM, the solution to (22) is obtained by alternating between \mathbf{x} , \mathbf{z} and $\boldsymbol{\mu}$ updates as follows [16]:

$$\mathbf{x}^{k+1} := \underset{\mathbf{x}}{\operatorname{argmin}} L_\rho(\mathbf{x}, \mathbf{z}^k, \boldsymbol{\mu}^k), \quad (23)$$

$$\mathbf{z}^{k+1} := \underset{\mathbf{z}}{\operatorname{argmin}} L_\rho(\mathbf{x}^{k+1}, \mathbf{z}, \boldsymbol{\mu}^k), \quad (24)$$

$$\boldsymbol{\mu}^{k+1} = \boldsymbol{\mu}^k + \rho(\mathbf{x}^{k+1} - \mathbf{z}^{k+1}), \quad (25)$$

where k is the iteration number. \mathbf{x}^{k+1} update depends on previous states of \mathbf{z}^k and $\boldsymbol{\mu}^k$, while \mathbf{z}^{k+1} update depends on the updated state \mathbf{x}^{k+1} and on previous state of the dual variable $\boldsymbol{\mu}^k$. The dual variable update $\boldsymbol{\mu}^{k+1}$ depends on updated states of \mathbf{x}^{k+1} and \mathbf{z}^{k+1} . A scaled dual variable form is obtained by assuming $\mathbf{u} = (1/\rho)\boldsymbol{\mu}$ in all the updates. \mathbf{x}^{k+1} update in (23) can be expressed as the projection onto $\{\mathbf{x} \in \mathbb{R}^n \mid \mathbf{H}\mathbf{x} = \mathbf{y}\}$, and can be written explicitly as [16]:

$$\mathbf{x}^{k+1} = (\mathbf{I} - \mathbf{H}^T(\mathbf{H}\mathbf{H}^T)^{-1}\mathbf{H})(\mathbf{z}^k - \mathbf{u}^k) + \mathbf{H}^T(\mathbf{H}\mathbf{H}^T)^{-1}\mathbf{y}. \quad (26)$$

\mathbf{z}^{k+1} update in (24) can be expressed using the soft thresholding operator S as [16]:

$$\mathbf{z}^{k+1} := S_{1/\rho}(\mathbf{x}^{k+1} + \mathbf{u}^k), \quad (27)$$

$$\mathbf{z}^{k+1} := \left(\mathbf{x}^{k+1} + \mathbf{u}^k - \frac{1}{\rho}\right)_+ - \left(-\mathbf{x}^{k+1} - \mathbf{u}^k - \frac{1}{\rho}\right)_+, \quad (28)$$

where $(a)_+ = \max(a, 0)$. This iterative algorithm converges to the optimal solution after satisfying the stopping criteria which depends on the primal and dual residuals.

The primal and dual residues are computed respectively at iteration k as [16]:

$$\mathbf{r}^k = \mathbf{x}^k - \mathbf{z}^k, \quad (29)$$

$$\mathbf{s}^k = \rho(\mathbf{z}^k - \mathbf{z}^{k-1}). \quad (30)$$

ADMM iterative algorithm terminates if a stopping criteria is satisfied. Different stopping criteria can be defined, for example:

$$\|\mathbf{r}^k\|_2 \leq \epsilon^{\text{pri}} \quad \text{and} \quad \|\mathbf{s}^k\|_2 \leq \epsilon^{\text{dual}},$$

where ϵ^{pri} is the primal tolerance, and ϵ^{dual} is the dual tolerance. In literature a relaxation parameter has been added to improve the ADMM convergence. As suggested in [16], the term \mathbf{x}^{k+1} in \mathbf{z}^{k+1} and $\boldsymbol{\mu}^{k+1}$ updates can be replaced by:

$$\mathbf{x}^{k+1} = \alpha \mathbf{x}^{k+1} + (1 - \alpha) \mathbf{z}^k, \quad (31)$$

where $\alpha \in [0, 2]$ is the relaxation parameter. If $\alpha > 1$ it is called an over-relaxation parameter, while if $\alpha < 1$ it is called an under relaxation parameter. The implementation of the ADMM Basis Pursuit is summarized in Table 1.

Table 1: ADMM Basis Pursuit Algorithm

<p>Inputs: $\mathbf{y}, \mathbf{H}, \boldsymbol{\alpha}, \boldsymbol{\rho}$</p> <p>Initialize $\mathbf{x}^0, \mathbf{z}^0, \mathbf{u}^0, \mathbf{k} = \mathbf{0}$</p> <p>While stopping criteria is not satisfied, do:</p> <p style="padding-left: 2em;">Step1. Update the variable \mathbf{x}^{k+1} using (23)</p> <p style="padding-left: 2em;">Step2. relax \mathbf{x}^{k+1} using $\boldsymbol{\alpha}$ as in (31)</p> <p style="padding-left: 2em;">Step3. Update the variable \mathbf{z}^{k+1} using (24)</p> <p style="padding-left: 2em;">Step4. Update the dual variable \mathbf{u}^{k+1} using (25)</p> <p style="padding-left: 2em;">$\mathbf{k} = \mathbf{k} + \mathbf{1}$</p> <p>end</p> <p>Outputs: \mathbf{z}^{k+1}</p>

2.4 Related Works

The followings summarize some related works to PAT, CS and ADMM respectively.

2.4.1 Photoacoustic Tomography

Several works have been done contributing to PAT field. In [3], the fundamentals and principles of PAT are presented along with its system characteristics, recent applications and major implementations. In [17], the authors have presented some current state-of-the-art photoacoustic imaging techniques and their outcomes related to clinical cancer applications. They explained many techniques such as photoacoustic computed tomography systems, stand-alone photoacoustic imaging systems and photoacoustic imaging systems resembling clinical ultrasound scanners. In [18], a realistic 3D numerical breast phantoms are developed for photoacoustic computed tomography and ultrasound computed tomography. The 3D phantoms describe the acoustic and optical breast properties and they are established by employing a clinical contrast-enhanced magnetic resonance data (MR).

Some researchers have used an exogenous agent to improve the contrast and penetration of PAT such as the case in [19]. The authors have reported a novel PA imaging scanner with a tyrosinase-based reporter system that makes tumor cells produce their own contrast. Experiments were done on mice and 3D images of xenografts formed of tyrosinase-expressing cells are obtained in vivo. A retroviral vector is used which permanently alter the genome so the image contrast is passed on to progeny cells which allow the study of the growth of these cells in the long term.

The authors of [20] have used a guided filtering approach to improve the total variation regularization method. The guiding image was obtained from linear back projection method. While in [7], the authors discussed in details the k-wave MATLAB toolbox for simulating and reconstructing PA signals. They presented several modeling examples for example they used data interpolation to improve time reversal reconstruction.

2.4.2 Compressive sensing framework

An intensive research has been conducted in understanding and explaining CS theory. Many of them have proposed CS framework for PAT. The works related to CS are summarized as follows. In [11], a survey was made to explain the compressive sensing idea along with its prerequisites and its reconstruction algorithms. The survey considers the CS formulations in signal processing applications using a commonly used transformation domains such as Fourier Transform (FT) domain, polynomial FT domain, combined time-frequency domain and Hermite transform domain. In [21], a survey has been conducted to discuss the construction of deterministic matrices used for CS and to present some of the disadvantages of using random matrices in CS. In

[22], an introduction to compressive sampling was provided with its two fundamental premises which are the sparsity and incoherency. Also, CS has been shown to be robust by corrupting the measured data with noise. Multiple random sensing matrices were presented and many CS applications were discussed.

One of the first works done that applies CS to PAT is in [4], where compressive sensing formalism was explained in details with its prerequisites. PAT was modeled in frequency domain and simulation results showed a dramatic reduction in the number of measurements needed for a given quality of recovered images. In [23], the theory of CS was used to improve image quality of full-view PAT with less ultrasound sensors where a circularly distributed asymmetric data acquisition frame was used. Firstly, a pre-imaging process was done using few number of sensors to form a low quality image. This pre-imaging process allows determining the Region of Interest (ROI) and then redistribute the sensors non-uniformly around a circle while most of the sensors are focused on the ROI. In [24], the acoustics topics of CS and holography are addressed. Using a sparsity constraint, CS reconstructs the direction of arrival of multiple sources. Many topics were also addressed such as sparse sensing in acoustic medium and sparse array configuration. In [25], the acquisition speed of PAT was increased dramatically by using spatial sparsity constraints with the development of PAT systems that are able to sub-sample the acoustic waves. The spatial sub-sampling was done using two models that were implemented using Fabry-Perot interferometer. The potential of the models were demonstrated through simulated data, experimental measured data, realistic numerical phantoms as well as in vivo experiments. In [26], 4D PAT was enhanced in terms of image quality by exploiting the additional temporal redundancy of measured data and coupling the image reconstruction methods with

sparsity constrained motion estimation models. In [27], the CS theory was used to increase the PAT imaging speed. The concept of sparsifying temporal transforms for 3D PAT was developed. The algorithm depends on two stages, the first one is recovering the complete pressure waves, while the second one is applying back-projection method which is a standard reconstruction algorithm. In [28], a distributed compressive sensing framework has been used to formulate photoacoustic signal recovery to exploit the intra- and inter- signal correlation. In [29], a number of sparse recovery algorithms were classified and their performance is tested and compared with each other. The performance of the recovery algorithms (CS algorithms) is tested based on recovery error, recovery time and covariance.

2.4.3 Alternating direction method of multipliers

In [16], a brief survey has been conducted to show the theory and history behind ADMM. Some of the ADMM applications were also discussed including Basis Pursuit (BP), Least Absolute Shrinkage Operator (LASSO), sparse logistic regression, support vector machines, covariance selection, and many others. In [30], optimization problems with multi-block linear constraints are solved using parallel randomized block coordinate method. The algorithm behaves like parallel randomized block coordinate descent. The proposed method outperform the state-of-the-arts methods in two applications which are the robust principal component and over-lapping group LASSO. In [31], a Total Variation (TV) problem was reformulated as a linear equality constrained problem using Alternating Direction Method (ADM). The ADM approach can be applied to multi- and single- channel images with impulsive or Gaussian noise. The computational complexity of the algorithm per-iteration is dominated by using

several Fast Fourier Transform (FFT). The approach has been simulated and compared to some of the state-of-the-art algorithms and results show that it outperforms them since it is more stable and efficient. In [32], a fast implementation of ADMM and another algorithm called the Alternating Minimization Algorithm (AMA). Global convergence bounds are provided for both classical and accelerated methods, in case the objective function is strictly convex. In [33], the basis pursuit deconvolution was performed to improve the PA reconstructed images accuracy of blurring models. An approximate blur matrix was built via the Lanczos bidagonalization and used in the simulations.

In this thesis, a distributed photoacoustic reconstruction algorithm is proposed that is capable of dramatically reducing the computational complexity while at the same time maintains high quality of recovered images. Conventional reconstruction of photoacoustic images relies on a centralized framework in which the whole measurements are processed using a central processor. Processing all measurements using a central processor may entail computational complexity especially in 3D images. Our proposed algorithm is based on splitting the optimization problem (recovery problem) into several sub-problems that can be solved iteratively in parallel. Each sub-problem is processed by a local processor with information exchange with a central (global) processor that works as a coordinator. Each local processor/unit is responsible to process the measured data of a small group of sensors. The proposed algorithm is based on a distributed implementation of the ADMM. The optimization problem is formulated using the Basis Pursuit (BP) formulation, but can be extended for other formulations. To the best of our knowledge, this is the first work that proposes a distributed PA recovery algorithm based on ADMM.

Chapter 3: Distributed Recovery Algorithm

In this chapter, the proposed implementation of distributed ADMM Basis Pursuit (BP) algorithm to recover PA images is discussed. Also, an explanation of the use of relaxation parameters for enhancing the optimization convergence is presented.

3.1 Distributed ADMM BP

ADMM BP is very powerful in finding the optimal solution; however for photoacoustic images with large dimensions (large N_{in}^2) the system matrix \mathbf{H} is usually huge and its size is even larger for 3D images. Also, in ADMM BP, the measurement vector is composed of measurements from all sensors. So, the computational complexity to process such problems is quite high. In this sense, a distributed and parallel implementation of the ADMM BP is proposed. In distributed ADMM BP, the whole system matrix and the measurements of all sensors are distributed over multiple local processors that work in parallel to find the optimal solution.

To split the ADMM optimization problem in equation (20) into smaller sub-problems that can be processed in parallel using multi processors, the sensing matrix \mathbf{H} is divided into M sub-matrices. Similarly, the measurement vector \mathbf{y} is divided into M measurement vectors:

$$\mathbf{H} = \begin{bmatrix} \mathbf{H}_1 \\ \mathbf{H}_2 \\ \cdot \\ \cdot \\ \mathbf{H}_M \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \\ \cdot \\ \cdot \\ \mathbf{y}_M \end{bmatrix}.$$

Since $\mathbf{H} \in \mathbb{R}^{N_s N_t \times N_{in}^2}$ then $\mathbf{H}_i \in \mathbb{R}^{m_i \times N_{in}^2}$ and $\mathbf{y}_i \in \mathbb{R}^{m_i}$, where $\sum_{i=1}^M m_i = N_s N_t$.

\mathbf{H}_i and \mathbf{y}_i denotes the i^{th} block or sub-problem that will be handled by the i^{th}

processor. According to these divisions, the optimization problem (20) can be formulated as:

$$\min_{\mathbf{x}_1, \dots, \mathbf{x}_M, \mathbf{z}} \sum_{i=1}^M f_i(\mathbf{x}_i) + \|\mathbf{z}\|_1 \text{ s.t. } \mathbf{x}_i - \mathbf{z} = 0, \text{ for } i = 1, \dots, M. \quad (32)$$

The indicator function in equation (20) is split into M sub-functions that each can be represented as a local function at the local processor. The \mathbf{z} variable is considered as the global variable that need to be received by all local processors. The optimization of the global variable \mathbf{z} need to be handled by the global processor. According to equation (32), the ADMM Basis Pursuit updates for the scaled form are as follows:

$$\mathbf{x}_i^{k+1} := \underset{\mathbf{x}_i}{\operatorname{argmin}} f_i(\mathbf{x}_i) + \frac{\rho}{2} \|\mathbf{x}_i - \mathbf{z}^k + \mathbf{u}_i^k\|_2^2, i = 1, \dots, M, \quad (33)$$

$$\mathbf{z}^{k+1} := \underset{\mathbf{z}}{\operatorname{argmin}} \|\mathbf{z}\|_1 + \frac{\rho}{2} \|\bar{\mathbf{x}}^{k+1} - \mathbf{z} + \bar{\mathbf{u}}^k\|_2^2, \quad (34)$$

$$\mathbf{u}_i^{k+1} := \mathbf{u}_i + (\mathbf{x}_i^{k+1} - \mathbf{z}^{k+1}), i = 1, \dots, M, \quad (35)$$

where $\bar{\mathbf{x}}^{k+1} = \frac{1}{M} \sum_{i=1}^M \mathbf{x}_i^{k+1}$ and $\bar{\mathbf{u}}^k = \frac{1}{M} \sum_{i=1}^M \mathbf{u}_i^k$ are the average values of \mathbf{x}_i^{k+1} and \mathbf{u}_i^k respectively. Similar to equation (26), \mathbf{x}_i^{k+1} update can be written explicitly as:

$$\mathbf{x}_i^{k+1} = (\mathbf{I} - \mathbf{H}_i^T (\mathbf{H}_i \mathbf{H}_i^T)^{-1} \mathbf{H}_i) (\mathbf{z}^k - \mathbf{u}_i^k) + \mathbf{H}_i^T (\mathbf{H}_i \mathbf{H}_i^T)^{-1} \mathbf{y}_i. \quad (36)$$

\mathbf{z} is updated based on the global variable consensus form using soft thresholding operator and the average values $\bar{\mathbf{x}}^{k+1}$ and $\bar{\mathbf{u}}^k$ as:

$$\mathbf{z}^{k+1} = (\bar{\mathbf{x}}^{k+1} + \bar{\mathbf{u}}^k - \frac{1}{\rho})_+ - (-\bar{\mathbf{x}}^{k+1} - \bar{\mathbf{u}}^k - \frac{1}{\rho})_+. \quad (37)$$

The distributed ADMM BP algorithm iterates until the primal and dual residues stopping criteria are met. In case of the proposed algorithm, the dual residue is the same as in equation (30), while the primal residue is computed as:

$$\mathbf{r}^{k+1} = \bar{\mathbf{x}}^{k+1} - \mathbf{z}^{k+1}. \quad (38)$$

3.2 Relaxation Parameter

Before updating \mathbf{z}^{k+1} in the global processor, a relaxation parameter α can be firstly applied to the average value $\bar{\mathbf{x}}^{k+1}$ as:

$$\bar{\mathbf{x}}^{k+1} = \alpha \bar{\mathbf{x}}^{k+1} + (1 - \alpha) \mathbf{z}^k, \quad (39)$$

and then applied to the average value of the dual updates $\bar{\mathbf{u}}^k$ as:

$$\bar{\mathbf{u}}^k = \alpha \bar{\mathbf{u}}^k + (1 - \alpha) \bar{\mathbf{x}}^{k+1}, \quad (40)$$

where $\bar{\mathbf{x}}^{k+1}$ in (40) is the relaxed average value found from (39). Adding an over-relaxation parameter in this manner will significantly improve the convergence of the distributed ADMM BP as shown in the simulations.

The proposed algorithm is summarized in Table 2. Each local processor finds and sends its local updates \mathbf{x}_i^{k+1} and \mathbf{u}_i^k to the global processor. The global processor updates the global variable \mathbf{z}^{k+1} using the average values of local updates based on (37). If the stopping criterion is met, the optimal solution is found and it is equal to the last updated value of \mathbf{z}^{k+1} , otherwise, the global processor will broadcast the updated \mathbf{z}^{k+1} to local processors again to do the next iteration.

Table 2: The Proposed distributed ADMM BP Algorithm

<p><i>Inputs to local unit i : $\mathbf{y}_i, \mathbf{H}_i$, for $i = 1, 2, 3, \dots, M$</i></p> <p><i>Inputs to global unit: α and ρ</i></p> <p>Initialize $\mathbf{u}_i^0, \mathbf{z}^0, \mathbf{k} = \mathbf{0}$</p> <p>While stopping criteria is not satisfied, do:</p> <p><i>Step1.</i> Update the local variable \mathbf{x}_i^{k+1} using (29)</p> <p><i>Step2.</i> Find $\bar{\mathbf{x}}^{k+1} = \frac{1}{M} \sum_{i=1}^M \mathbf{x}_i^{k+1}$, and $\bar{\mathbf{u}}^k = \frac{1}{M} \sum_{i=1}^M \mathbf{u}_i^k$</p> <p><i>Step2.</i> Relax $\bar{\mathbf{x}}^{k+1}$ using α as in (32)</p> <p><i>Step3.</i> Relax $\bar{\mathbf{u}}^k$ using α as in (33)</p> <p><i>Step4.</i> Update the global variable \mathbf{z}^{k+1} using (30)</p> <p><i>Step5.</i> Update the local variable \mathbf{u}_i^k using (28)</p> <p>$\mathbf{k} = \mathbf{k} + \mathbf{1}$</p> <p><i>end</i></p> <p><i>Output: \mathbf{z}^{k+1}</i></p>
--

The proposed algorithm is explained using a block diagram as shown in Figure

4. Each local processor collects and processes the measurements from its small group of sensors. The global processor has no access to sensors' data.

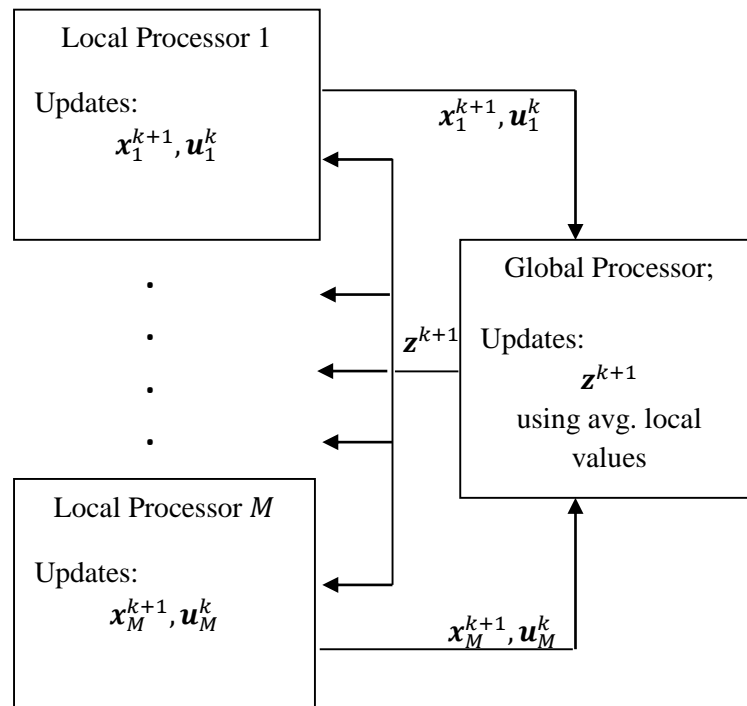


Figure 4: Block diagram of the proposed algorithm

Chapter 4: Numerical Simulations and Results

In this section, simulation results of ADMM BP and the proposed distributed ADMM BP are conducted using similar setups. The efficiency of the proposed algorithm compared to ADMM BP is tested in terms of the computational complexity and image quality.

4.1 Simulations Setup

The processor used in simulations is Intel® Core™ i5-2400 CPU @3.10 GHz with MATLAB R2016a. The system matrix was built as discussed in [6], with an inner grid of size $N_{in} = 64$, outer grid of size $N_{out} = 256$ and square sensors distribution of 67 sensors ($N_s = 67$) as shown in Figure 5.

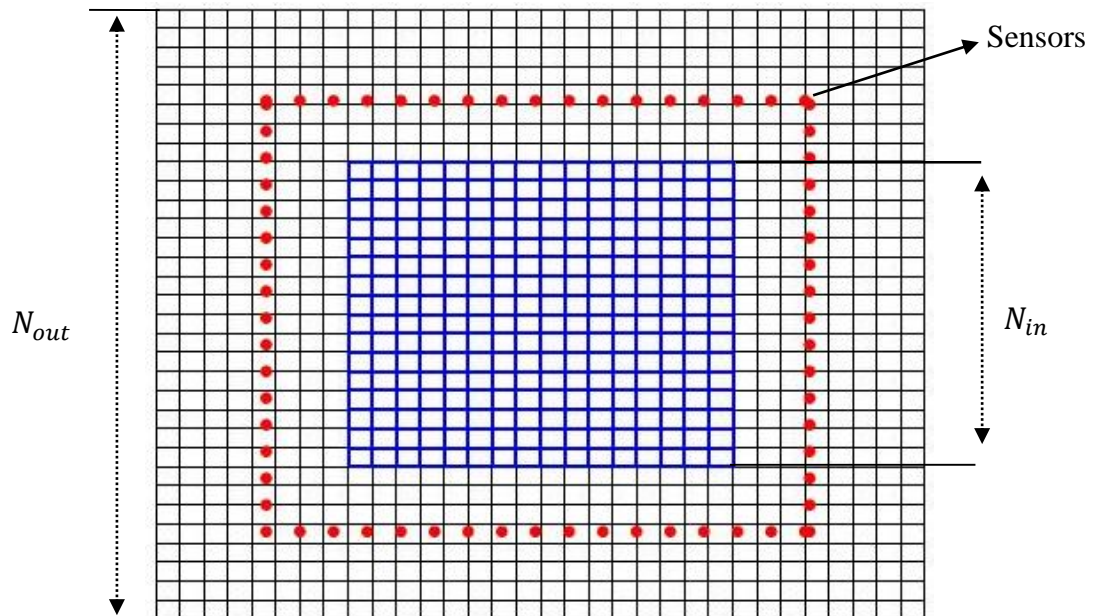


Figure 5: Inner and outer grids with square sensors configuration

The sensors were placed outside the imaging grid (inner) and in the outer grid. A grid spacing is taken as $d = 0.1 \text{ mm}$ and the medium acoustic speed is assumed to be $c = 1500 \text{ m/s}$. A maximum frequency is computed as $c/(2 \times d)$ and it is equal to 7.5 MHz . The acoustic signals are acquired during a time slot of $5\mu\text{s}$. Thus, based on Nyquist criteria, the number of samples needed to reconstruct an image at Nyquist rate is 75 ($N_t = 75$). Initially, the \mathbf{H} matrix is of dimension 5025×4096 , therefore it is an overdetermined system. Note that we started with this matrix dimension only for comparison reasons, later on the number of sensors and samples will be reduced by applying the CS theory. The measurements of our simulations were created by the pseudo-spectral matrix \mathbf{H} . These measurements have been compared first to measurements generated by k -wave under similar settings to ensure the validity of these measurements. k -wave is an open source MATLAB toolbox that is designed for time domain acoustic simulations in realistic media [7]. Figure 6 shows the measurement of one sensor only generated by k -wave and \mathbf{H} matrix. It is clear that both measurements match perfectly.

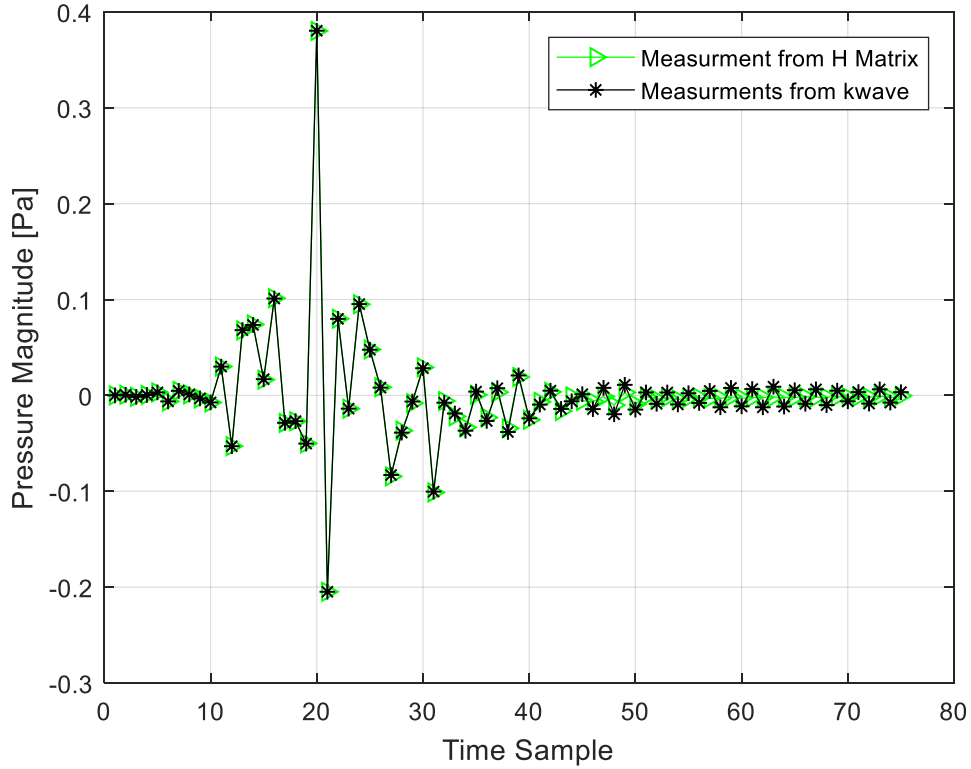


Figure 6: The measured pressure by one sensor using k -wave and the pseudo-spectral matrix (\mathbf{H} Matrix)

4.2 ADMM BP Simulations

The implementation of ADMM BP was done as described in Table 1, where \mathbf{y} is a column vector composed of measurements from all sensors. Based on the specified numerical values in the setup sub-section, the vector \mathbf{y} has initially a length of 5025. \mathbf{x}^0 , \mathbf{u}^0 and \mathbf{z}^0 are vectors of length 4096, and they are randomly initialized. An over-relaxation parameter $\alpha = 1.3$ is used. The augmented Lagrangian ρ was set to 1. The stopping criteria specified in (29) and (30) were applied with primal ϵ^{pri} and dual ϵ^{dual} stopping thresholds set to $\sqrt{N_{in}^2} \times 10^{-3}$.

The algorithm described in Table 1 is tested on two common images: the Shepp-Logan phantom and the blood vessels phantom. It is also tested on a realistic breast phantom that is generated based on real tissue characteristic values. The real phantom is available in [18]. The realistic breast phantom is a binary file written under UNIT8 data format, where each voxel contains a value that represents a specific tissue type. Blood vessel is given a value of 5, skin is given a value of 1, background is valued 0, fat and fibroglandular tissues are given values of 3 and 2 respectively. Replacing these values with their corresponding realistic initial pressure values based on acoustic properties of each tissue type, we can generate a realistic initial pressure image. For simulations, the optical absorption coefficients of breast tissues using a wavelength of 760 nm are shown in Table 3 [18].

Table 3: The optical absorption coefficients of different breast tissue types

Tissue Type	Background	Blood vessels	Skin	Fibroglandular	Fat
$\mu_a (cm^{-1})$	0	5	0.08	0.04	0.05

The initial pressure distribution can be generated based on values shown in Table 3 by using the initial pressure equation (4) with $\Gamma = 0.1$, $\eta_{th} = 1$, and the optical fluence $F = 100 J/m^2$ [18]. The initial pressure values has been inserted in its corresponding voxel type. The phantom available in [18] is a 3D phantom, for simplicity we took a 2D slice from it and resize it to 64×64 . The reconstructed images along with the original ones are shown in Figure 7.

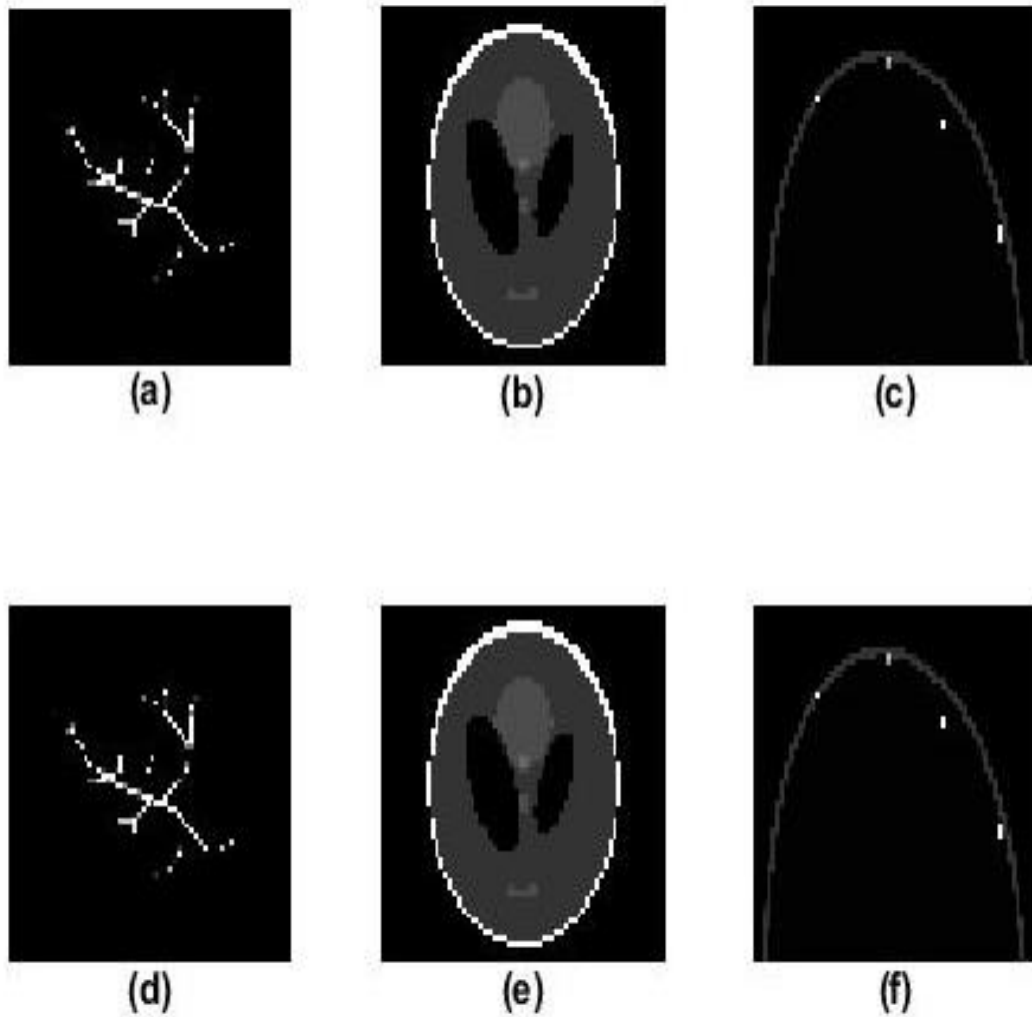


Figure 7: Original images and reconstructed images using ADMM. (a,b,c) Original images of blood vessels, Shepp-Logan and real phantom, respectively. (d,e,f) Reconstructed images at Nyquist rate and 67 sensors.

In case of reconstruction at Nyquist rate ($N_t = 75$) with $N_s = 67$, the inverse problem is overdetermined and reconstruction was done perfectly. However, one of the aims of this thesis is to optimize and reduce the number of sensors and samples using the CS theory. Firstly, the number of sensors are reduced and the algorithm performance is tested again. The results of these tests are shown in Table 4.

In all our tests, the Structural Similarity index (SSIM) is used to measure the quality of images [18]. SSIM ranges between -1 if there is no similarity between original image and reconstructed image, and 1 if both images are identical. The formula used to compute the SSIM value assuming to have two images x_1 and x_2 is written as [35]:

$$SSIM = \frac{(2\mu_{x_1}\mu_{x_2}+c_1)(2\sigma_{x_1x_2}+c_2)}{(\mu_{x_1}^2+\mu_{x_2}^2+c_1)(\sigma_{x_1}^2+\sigma_{x_2}^2+c_2)} \quad (41)$$

where μ_{x_1}, μ_{x_2} are the averages of x_1 and x_2 respectively. $\sigma_{x_1}, \sigma_{x_2}$ are the variances of x_1 and x_2 respectively. $\sigma_{x_1x_2}$ is the covariance of x_1 and x_2 . $c_1 = (0.01L)^2$ and $c_2 = (0.03L)^2$ are used to stabilize the division with weak denominator, where L is the pixel-values dynamic range.

Table 4: Comparisons between three phantoms reconstruction while reducing the number of sensors (N_s).

Phantom	N_s	Reconstruction Time (s)	Iterations	SSIM
Blood vessels	67	13.3431	6	1.00
	50	5.7900	6	1.00
	40	3.2082	6	1.00
	31	1.6058	6	1.00
Shepp- Logan	67	12.9243	10	1.00
	50	5.7243	10	1.00
	40	3.1940	10	1.00
	31	1.6571	10	1.00
Real phantom	67	13.3919	8	0.9999
	50	5.6814	8	0.9999
	40	3.1833	8	0.9999
	31	1.6282	8	0.9999

In Table 4, for the reduced number of sensors setup, the sensors locations were taken randomly for $N_s=40$ and $N_s=50$ from the distribution shown in Figure 5. For the case of $N_s=31$, the sensors were distributed evenly around the same square configuration. As shown in Table 4, reducing the number of sensors to 31 (out of 67) still allows for perfect reconstruction of the images. The computational complexity and run time are greatly reduced for this case. The run time complexity is less than 10% of the run time complexity when 67 sensors are used. Similar conclusions can be drawn for the cases $N_s=40$ and $N_s=50$. Acquisition with less than 31 sensors causes the ADMM algorithm to diverge as there are no sufficient measurements. Note in Table 4, the number of samples were fixed to 75.

4.3 Distributed ADMM BP Simulations

The main purpose is to reduce the computational complexity of PAT image reconstruction. From Table 4, it was found that the least number of sensors that allows for perfect reconstruction is 31 sensors, hence the distributed ADMM BP is tested only with this number of sensors. The 31 sensors were distributed evenly over a square outside the imaging grid similar to section 6.1. The same numerical values discussed in the setup sub-section are used. With 31 sensors, the pseudo-spectral matrix is of dimension 2325×4096 . Following the lines of [6], the arrangement of \mathbf{H} matrix is as follows: measurements of the 31 sensors for the first time sample comes consecutively after each other in the first 31 rows, and so on for the rest time samples.

In distributed ADMM BP, the whole number of sensors are divided into a number of small groups. Each local processor/unit is responsible to process data received from its own small group of sensors. Therefore, the \mathbf{H} matrix is rearranged

such that the measurements at all the time samples for each sensor comes consecutively after each other. Then the matrix is divided by rows while ensuring all the time samples for each sensor are taken. In the first experiment, the 31 sensors was divided into 4 groups ($M = 4$), the first three groups has 8 sensors while the last one has 7 sensors only. Each group is processed by a different local unit, so there is a total of 4 local units and one global unit. In this case, the system matrix \mathbf{H} is divided into four sub-matrices, the first 3 sub-matrices \mathbf{H}_i (for $i = 1,2,3$) are of size 600×4096 , the last sub-matrix \mathbf{H}_4 is of size 525×4096 . Each local unit i receives its own measurements vector \mathbf{y}_i ($i = 1,2,3,4$) from its own sensors group.

The implementation of distributed ADMM BP is done as described in Table 2. The local units are working simultaneously in parallel each with its corresponding \mathbf{y}_i , \mathbf{H}_i , \mathbf{x}_i and \mathbf{u}_i . An over-relaxation parameter $\alpha = 1.3$ is used and $\rho = 1$. The stopping criteria are the norms of primal and dual residues where the primal residue is computed based on (31) with $\epsilon^{\text{pri}} = \epsilon^{\text{dual}} = 0.064$. In the first experiment, the algorithm was applied to the three phantoms mentioned before, and it works successfully and perfectly in reconstructing all of the images as shown in Figure 8.

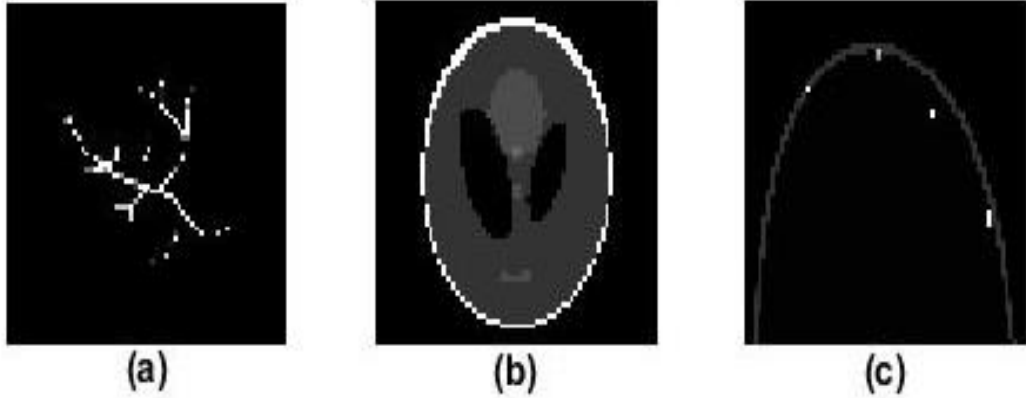


Figure 8: Reconstructed images using the proposed algorithm: a) Blood vessel, b) Shepp-Logan phantom and c) Real phantom at Nyquist rate ($N_t = 75$) using distributed BP ADMM with 4 local units ($M = 4$) and total of 31 sensors.

The algorithm has been tested also on different number of local units. Table 5 summarizes the results of reconstruction at Nyquist rate (75 time samples) obtained after dividing the sensors into 4, 8, 15 and 31 groups. In case of 31 groups, each local unit processes the measurements of one sensor only.

Table 5: Distributed ADMM reconstructions using different no. of local units (M)

Phantom	M	Reconstruction Time (s)	Iterations	SSIM
Blood Vessels	4	0.1992	16	1.00
	8	0.1934	21	0.9999
	15	0.2330	26	0.9994
	31	0.3244	37	0.9916
Shepp-Logan	4	0.2237	21	0.9956
	8	0.2240	24	0.9756
	15	0.2422	27	0.9429
	31	0.2836	32	0.9054
Real Phantom	4	0.1867	16	0.9998
	8	0.1816	19	0.9992
	15	0.2147	24	0.9989
	31	0.2905	31	0.9974

Comparing the proposed algorithm with ADMM BP results using 31 sensors, the proposed algorithm shows a dramatic reduction in computational complexity which is reflected on a faster running time. As shown in Table 5, for different M sub-problems, the run time has been reduced to around $1/8^{th}$ of the run time computed in Table 2 for 31 sensors. The SSIM value has been slightly affected but it is still very high for the three phantoms. There is a noticeable increase in the number of iterations compared to the ADMM BP, but this is not of much concern since the original images are reconstructed perfectly with a much faster running time. The number of iterations increases with higher number of groups (M), therefore dividing the problem into more sub-problems reduces the run time of local processors at each iteration, but not the total reconstruction time. Also, the SSIM value is slightly decreasing going from $M = 4$ to $M = 31$. For optimal results, there should be a balance between the number of sub-problems, quality of recovered image and the running time. Based on the results shown in Table 5, the optimal choice is when $M = 4$, since the images are perfectly reconstructed with the least time and least number of iterations.

As suggested by compressive sensing theory, a sparse image can be reconstructed at much lower rate than the Nyquist rate. The effect of reducing the time samples on the quality of reconstructed images is tested using the proposed algorithm with $M = 4$. Table 6 shows the results of reconstructing images using a much lower number of samples than 75. The number of samples has been taken randomly using a Gaussian random matrix which is known to satisfy the restricted isometric property (RIP) which is a significant property in CS theory for the system matrix.

Table 6: Distributed ADMM reconstructions using reduced number of samples

Phantom	Time samples	Reconstruction Time (s)	Iterations	SSIM
Blood Vessels	50	0.3424	32	0.9999
	30	0.6822	71	0.9995
	20	0.8462	94	0.9760
	15	1.3904	153	0.7137
Shepp-Logan	50	1.9836	214	0.7171
	30	2.0538	226	0.2458
Real Phantom	50	0.3905	41	0.9991
	30	0.8498	92	0.9973
	20	1.7779	200	0.9435
	15	2.1590	236	0.6946

As shown in Table 6, reaching 20 time samples still allows the reconstruction of blood vessel and real phantoms with very high SSIM value. While the quality of reconstructing the Shepp-Logan phantom is badly affected using less than 50 time samples, this is because it has less sparsity than the other two phantoms. Therefore, the results of reducing the number of time samples less than the Nyquist rate depends mainly on the image sparsity.

4.4 Efficient Communication Links

A key factor for a successful implementation of a distributed iterative algorithm is the convergence using delayed data or asynchronous updates at each subsystem. The delayed/ or asynchronous updates can be simulated by updating only a group of the local processors at each iteration and send their updates to the global unit. In the global unit, the updates of the remaining local processors are replaced by

values from previous iteration assuming that the global unit has a memory. The updating local processors are selected randomly at each iteration. Consequently, the non-sensitivity of the proposed distributed ADMM BP algorithm to delayed data/asynchronous updates can be utilized to reduce communication overheads between the local processors and the global one and mitigate link failures. Using a probabilistic model, the effect of outdated data on the convergence of the proposed distributed BP ADMM algorithm is investigated using the blood vessel phantom for $M = 4$ and $M = 8$. At first, 25% of the local units are assumed to face communication delays, so their updated data are not received by the global unit at that iteration. In this case, for $M = 4$, one local unit is chosen randomly, so one communication link is saved. Similarly, in case of $M = 8$, two local units are not transmitting their updates. The percentage of links saved is increased to 50% and 75% for both cases $M = 4,8$. Table 7 shows the results of this simulation.

Table 7: Efficient communication links for blood vessel phantom($M = 4, 8$)

Phantom	M	Percentage of links saved	Iterations	Time(s)	SSIM
Blood Vessels	4	25%	53	0.5146	0.9998
		50%	162	1.5259	0.9971
		75%	187	1.7419	0.9979
	8	25%	53	0.4882	0.9983
		50%	101	0.8981	0.9944
		75%	212	1.9148	0.9983

As shown in Table 7, the proposed algorithm is also successful in reconstructing images with asynchronous/delayed data. If the system faces some delays, still it can reconstruct the images with high SSIM value. Although it needs more number of iterations, the reconstruction time is still much lower than the reconstruction time of ADMM BP shown in Table 4. Considering the worst case scenario, where 75% of the links at each iteration are facing delays or communication problems, the algorithm still recovers the image successfully with high SSIM value.

Chapter 5: Conclusion and Future Work

5.1 Conclusion

The aim of this thesis work was to implement a low computational complexity photoacoustic image recovery algorithm that is able to reconstruct images perfectly using a few number of sensors and measurements, while at the same time possesses high stability and uniform performance guarantee.

The proposed algorithm is a distributed implementation of the Alternating Direction Method of Multipliers (ADMM) based on Compressive Sensing (CS) theory. The distributed iterative algorithm was formulated using the Basis Pursuit (BP) which provides high guarantee of performance and stability. The iterative algorithm was implemented using multiple local units/processors and one global processor. Local units work in parallel as each local unit processes data collected from a small group of sensors, solves a local optimization problem and exchanges information with the global unit. The global processor works as a coordinator on local units and has no access to sensors' measurements.

The proposed distributed algorithm can dramatically reduce the computational complexity and in turn the run time while maintaining high quality of reconstructed images. It showed a high guarantee of performance and stability in reconstructing different kinds of PA images with different sparsity levels. Furthermore, it has been shown that the algorithm is non-sensitive to communication delays or links failure. The optimal number of sensors that allows for perfect reconstruction of an image of resolution 64×64 was found to be 31 sensors. The algorithm was successful in

recovering images using different number of local units. The optimal number was found to be 4 local units, as it provided the highest SSIM value and the lowest running time. In case of 4 local units, the running time of the algorithm was only around 0.2 seconds.

5.2 Future Work

Simulations of the proposed algorithm can be investigated using 3D and 4D images. Also, testing the proposed algorithm on experimental data will be a significant future work.

References

- [1] H. Jiang, *Photoacoustic tomography*. Gainesville, FL, USA: Taylor & Francis Group, 2015.
- [2] K. J. Francis, P. Rajalakshmi, and S. Channappayya, “Wavelet domain frequency interpolation for photo-acoustic tomography,” presented at the International Conference on Medical Imaging, m-Health and Emerging Communication Systems (MEDCOM), India, 2014.
- [3] Z. Yong, J. Yao, and L. V. Wang, “Tutorial on photoacoustic tomography,” *Journal of Biomedical Optics*, vol. 21, no. 6, pp. 1–14, 2016.
- [4] J. Provost and F. Lesage, “The application of compressed sensing for photo-acoustic tomography,” *IEEE Transactions on Medical Imaging*, vol. 28, no. 4, pp. 585–594, 2009.
- [5] J. Meng, L. V. Wang, L. Ying, D. Liang, and L. Song, “Compressed-sensing photoacoustic computed tomography in vivo with partially known support,” *Optics Express*, vol.15, pp. 16510-16523, 2012.
- [6] K. J. F. Sumohana S. Channappayya, “A Simple and accurate matrix for model based photoacoustic imaging,” presented at the IEEE 18th International Conference on e-Health Networking, Munich, Germany, 2016.
- [7] B. E. Treeby and B. T. Cox, “k-Wave: MATLAB toolbox for the simulation and reconstruction of photoacoustic wave fields,” *Journal of Biomedical Optics*, vol. 15, pp. 021314–021314, 2010.
- [8] T. Szabo and P. A. Lewin, “Piezoelectric materials for imaging,” *Journal of Ultrasound in Medicine*, vol. 26, no. 3, pp. 283–288, 2007.
- [9] J. Kim *et al.*, “Programmable real-time clinical photoacoustic and ultrasound imaging system,” *Sci. Rep.*, vol. 6, pp. 35137-35200, 2016.
- [10] A. Sharma, “compressive sensing,” *International Journal of Engineering Technologies and Management Research*, vol. 5, no. 2, pp. 249–259, 2018.

- [11] I. OroviT, V. PapiT, C. Ioana, X. Li, and S. StankoviT, “Compressive sensing in signal processing: algorithms and transform domain formulations,” *Mathematical Problems in Engineering*, vol. 2016, pp. 16-80, 2016.
- [12] I. Rish and G. Grabarnik, *Sparse modeling: theory, algorithms, and applications*. 2014.
- [13] J. A. Tropp and A. C. Gilbert, “Signal recovery from random measurements via orthogonal matching pursuit,” *IEEE Transactions on Information Theory*, vol. 53, no. 12, pp. 4655–4666, 2007.
- [14] D. Donoho, Y. Tsaig, I. Drori, and J.-L. Starck, “Sparse solution of underdetermined systems of linear equations by stagewise orthogonal matching pursuit,” *IEEE Transactions on Information Theory*, vol. 58, no. 2, pp. 1094–1121, 2012.
- [15] E. Candes and R. Justin, “l1-MAGIC : recovery of sparse signals via convex programming,” 2005. [Online]. Available: <http://ci.nii.ac.jp/naid/10027600363/>.
- [16] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, “Distributed optimization and statistical learning via the alternating direction method of multipliers,” *Foundations and Trends® in Machine Learning*, vol. 3, pp. 1–122, 2010.
- [17] K. S. Valluru and J. K. Willmann, “Clinical photoacoustic imaging of cancer,” *Ultrasonography*, vol. 35, no. 4, pp. 267–280, 2016.
- [18] Y. Lou, W. Zhou, T. Matthews, C. Appleton, M. Appleton, and M. A. Anastasio, “Generation of anatomically realistic numerical phantoms for photoacoustic and ultrasonic breast imaging,” *Journal of Biomedical Optics*, vol. 22, no. 4, pp. 410–490, 2017.
- [19] A. P. Jathoul *et al.*, “Deep in vivo photoacoustic imaging of mammalian tissues using a tyrosinase-based genetic reporter,” *Nature Photonics*, vol. 9, pp. 239–246, 2015.
- [20] N. Awasthi, S. Kumar Kalva, M. Pramanik, and P. K. Yalavarthya, “Image-guided filtering for improving photoacoustic tomographic image reconstruction,” *Journal of Biomedical Optics*, vol. 23, no. 9, pp. 1–22, 2018.

- [21] Y. Shin and T. L. N. Nguyen, “Deterministic sensing matrices in compressive sensing: a survey,” *The ScientificWorld Journal*, vol. 2013, p. 6–112, 2013.
- [22] E. J. Candes and M. B. Wakin, “An introduction to compressive sampling,” *IEEE Signal Process. Mag.*, vol. 25, no. 2, pp. 21–30, 2008.
- [23] M. Caoa *et al.*, “Full-view photoacoustic tomography using asymmetric distributed sensors optimized with compressed sensing method,” *Biomedical Signal Processing and Control*, vol. 21, pp. 19–25, 2015.
- [24] P. Gerstoft, C. F. Mecklenbräuker, W. Seong, and M. Bianco, “Introduction to compressive sensing in acoustics,” *The Journal of the Acoustical Society of America*, vol. 143, no. 6, pp. 3731–3736, 2018.
- [25] P. Beard *et al.*, “Accelerated high-resolution photoacoustic tomography via compressed sensing,” *Physics in Medicine and Biology*, vol. 61, pp. 8908–8940, 2016.
- [26] F. Lucka *et al.*, “Enhancing compressed sensing 4D photoacoustic tomography by simultaneous motion estimation,” *SIAM Journal on Imaging Sciences*, vol. 11, no. 4, pp. 2224–2253, 2018.
- [27] M. Haltmeier, T. Berer, S. Moon, and P. Burgholzer, “Compressed sensing and sparsity in photoacoustic tomography,” *Journal of Optics*, vol. 18, no. 11, p. 4–114, 2016.
- [28] K. J. Francis, P. Rajalakshmi, and S. S. Channappayya, “Distributed compressed sensing for photo-acoustic imaging,” presented at the 2015 IEEE International Conference on Image Processing (ICIP), pp. 1513–1517, 2015.
- [29] Y. Arjoun, N. Kaabouch, H. El Ghazi, and A. Tamtaoui, “Compressive sensing: performance comparison of sparse recovery algorithms,” in *2017 IEEE 7th Annual Computing and Communication Workshop and Conference (CCWC)*, pp. 1–7, 2017.
- [30] H. Wang, A. Banerjee, and Z.-Q. Luo, “Parallel direction method of multipliers,” in *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 1*, Cambridge, MA, USA, pp. 181–189, 2014.

- [31] M. Tao and J. Yang, “Alternating direction algorithms for total variation deconvolution in image reconstruction,” TR0918 Department of Mathematics, Nanjing University., 2009.
- [32] T. Goldstein, B. O’Donoghue, S. Setzer, and R. Baraniuk, “Fast alternating direction optimization methods,” *SIAM J. Imaging Sci.*, vol. 7, no. 3, pp. 1588–1623, 2014.
- [33] J. Prakash, A. S. Raju, C. B. Shaw, M. Pramanik, and P. K. Yalavarthy, “Basis pursuit deconvolution for improving model-based reconstructed images in photoacoustic tomography,” *Biomedical Optics Express*, vol. 5, no. 5, pp. 1363–1377, 2014.
- [34] G. Renieblas, A. Nogués, A. González, N. Gómez-Leon, and E. Guibelalde del Castillo, “Structural similarity index family for image quality assessment in radiological images,” *Journal of Medical Imaging*, vol. 4, no. 3, pp. 1–11, 2017.
- [35] M. Khadtare, “GPU based image quality assessment using structural similarity (SSIM) index,” 2015.

Appendix

```

%-----
% MATLAB CODES
%-----

%-----
%Code 1:
Constructing the forward matrix H and generate measurements from it
%-----

clc;
clear all;
close all;

Nin=32;           %Imaging grid size Nin^2
Nout=128;        %K space grid size Nout^2
d=0.1e-3;        %Grid spacing
c=1500;          %sound speed [m/s]
max_freq=c/(2*d); %Maximum frequency
Fs=max_freq*2;  %Sampling frequency
t=5e-6;          %Time period of measuring acoustic waves
% t=3e-6;
Nt=t*Fs;         %number of samples

%Sensor distribution
%-----
skip=4;
sensor_grid=38;
sensor.mask = zeros(sensor_grid, sensor_grid);
sensor.mask(1, 1:skip:sensor_grid) = 1;
sensor.mask(end, 1:skip:sensor_grid) = 1;
sensor.mask(1:skip:sensor_grid, 1) = 1;
sensor.mask(1:skip:sensor_grid, end) = 1;
Ns=nnz(sensor.mask); %total number of sensors
kgrid_sensor= kWaveGrid(sensor_grid, d, sensor_grid, d);
[cart_data, order_index]=grid2cart(kgrid_sensor,sensor.mask);
xs=cart_data(1,:);
ys=cart_data(2,:);
rs_vec=[xs(:),ys(:)];
% rs_vec=rs_vec(1:2:Ns,:); %uncomment to make one sensor on and
one off from the whole number of sensors Ns
% [p, Ns]=size(1:2:Ns); %uncomment to change number of sensors
to the new number of sensors

%-----
%Initial Pressure Distribution Based on Realistic data
%-----

p0_magnitude = 2;
p0 = p0_magnitude * loadImage('EXAMPLE_source_two.bmp');
x0 = resize(p0, [Nin, Nin]);

% lamda=760;           %The wavelength (nm)

```

```

% gama=0.1; %The gruneisen parameter
(dimensionless)
% eita=1; %The percentage of absorbed light
converted to heat
% F=100; %The optical fluence [J/m^2] (depends
on lamda: 0.02*10^(2*((0.8)-0.7))
% mua=[0 0 0.04 0.05 1 5]*100; %Absorbtion Coefficient[1/m], idx
0:BackGround 1:Nothing 2:Fibro 3:fat 4:skin 5:blood
% disc1=load('phan2d'); %Loading the 3D breast image
% disc=disc1.phan; %Assigning values to disc
% disc2d=disc(110:173,120:183,400); %Take a 2d slice from the 3D
image
%
% %Note the initial pressure is in the range of 10kpa
% disc2d=cast(disc2d,'double'); %Changes the class of disc2d from
uint8 to double
%
% % A loop for assigning each voxel value the real initial pressure
value at lamda=760
% for i=0:5
% disc2d(disc2d==i)=F*mua(i+1)*eita*gama; %Initial
pressure equation (p0=eita*F*ua*gama)
% end
% x0=disc2d; %Assigning initial pressure values to x0
[Pa]

%-----
%Constructing The Forward Matrix H
%-----

%Define the imaging grid of size Nin^2
kgrid=kWaveGrid(Nin, d, Nin, d);
x=kgrid.x;
y=kgrid.y;
r_vec=[x(:), y(:)];

%Define the kspace grid of size Nout^2
kkgrid=kWaveGrid(Nout, 1, Nout, 1);
u=kkgrid.x; v=kkgrid.y;
kx=2*pi/Nout/d*u;
ky=(2*pi/(Nout*d))*v;
k_vec=[kx(:), ky(:)];
k_vec_k=sqrt(kx(:).^2+ky(:).^2);

% Uncomment if you want to define a centered Cartesian circular
sensor
% sensor_radius = 3e-3; % raduis is 5mm out side the imaging
grid [m]
% num_sensor_points = Ns;
% cart_sensor_mask = makeCartCircle(sensor_radius,
num_sensor_points);
% xs=cart_sensor_mask(1,:);
% ys=cart_sensor_mask(2,:);
% rs_vec=[xs(:),ys(:)];

%plot sensor distribution
figure(1)

```

```

plot(xs,ys,'o');
title('Sensors Distribution');xlabel('[m]');ylabel('[m]');
grid on
tic

%Find the forward discrete fourier transform
Wfwd=zeros(Nout^2,Nin^2);
for i=1:Nout^2
    for j=1:Nin^2
        Wfwd(i,j)=(1/Nout)*exp(-sqrt(-1)*dot(k_vec(i,:),r_vec(j,:)));
    end
end

%Initialization
Winv=zeros(Ns,Nout^2);
K=zeros(Ns*Nt,Nout^2);
Q=1:Ns;

%Find the inverse discrete fourier transform
for s=1:Ns
    for i=1:Nout^2
        Winv(s,i)=(1/Nout)*exp(sqrt(-1)*dot(k_vec(i,:),rs_vec(s,:)));
    end
end

%Construct K by stacking Kt_Matrix of all time samples.
Col_one=ones(Ns,1);
delta_t=t/Nt;
for sample=0:Nt-1
    kt=(cos(c*k_vec_k(:)*(sample*delta_t)));
    kt_matrix=Col_one*kt';
    Kt_Matrix=Winv.*kt_matrix;
    K(Q,:)=Kt_Matrix;
    Q=Q(end)+1:Q(end)+Ns;
end

%Find the H Matrix
H_Matrix=K*Wfwd;
save H_Matrix32_Nt75_Ns20 H_Matrix Nout Nin cart_data Ns %saves
the H_Matrix to be used in any simulations of same configuration
(same system).

```

```

%-----
Code 2:
ADMM code for reconstructing an image using basis pursuit
%-----

clc;
close all;
clear all;

% load H_Matrix32_Nt10_Ns39;
load H_Matrix64_Nt30_Ns35;
% load H_Matrix; %loads the full H matrix which consists of 67
sensors and 75 time samples for image size 64 by 64
[a,w]=size(cart_data);
[L,O]=size(H_Matrix);
Ns=w; %total number of sensors used to form H matrix
Nt=L/Ns; %total number of time samples based on nyquist rate,
since dimension of H is Ns*Nt X Nin*Nin then Nt=L/Ns
N=Nin; %Number of grid points

% Sensor-Option1:-----
% Reducing the number of active sensors to 34 instead of 67 by
taking
% the measurmenst of one sensor at all time samples and skip the
following
% sensor measurments (One sensor is ON and one is OFF).
% Comment the 7 following lines if you do not want to use this
option.
% -----
% H_Matrix1=[];
% for Q=1:Nt
% H_sub1=H_Matrix(1:2:Ns,:); %Makes one sensor on and one off from
the whole 67 sensors
% H_Matrix1=[H_Matrix1;H_sub1];
% H_Matrix(1:Ns,:)=[];
% end
% [d,N_Active_sensors]=size(1:2:Ns); %Number of On sensors is
N_Active_sensors

% Sensor-Option2:-----
% Activate less number of sensors randomly from the 67 sensors.
% Comment the 8 following lines if you do not want to use this
option.
% -----
% H_Matrix2=[];
% nind=randperm(Ns); %Distribute sensors randomly
% for Q=1:Nt
% H_sub1=H_Matrix(sort(nind(1:32)),:); %Activate less num of
sensors in random manner
% H_Matrix2=[H_Matrix2;H_sub1];
% H_Matrix(1:Ns,:)=[];
% end
% [d, N_Active_sensors]=size(nind(1:32)); %N_Active_sensors is the
number of activated sensors

% Time samples-Option3:-----

```

```

% Taking fewer number of time samples randomly from the full H
Matrix
% Comment the 4 following lines if you do not want to use this
option.
% -----
% Nt_new=50;          %Number of samples required to be taken randomly
% H_Random=randblock(H_Matrix2,[N_Active_sensors Nin^2]);
% A function for randomly redistribute the samples in H (here H_Random
contains all samples but arranged randomly)
% % H_Random=randblock(H_Matrix,[N_Active_sensors Nin^2]);%Uncomment
if you have used option 1 or 2, and comment the above line
% H_Matrix3=H_Random(1:N_Active_sensors*Nt_new,:);
% Takes the required number of samples from the random matrix
H_Random

% Time samples-Option4:-----
% Taking only the odd number of samples from the full H Matrix
% Comment the 9 following lines if you do not want to use this
option.
% -----
% R=1;
% N_1=Ns;
% H_Matrix4=[];
% [S,Nt_odd]=size(1:2:Nt); %Nt_new is the number of odd samples
% for Q=1:Nt_odd
% H_Matrix4=[H_Matrix4; H_Matrix(R:N_1,:)];
% R=R+2*Ns; %takes only the odd number of samples (sample number
1, 3, 5, 7 ...etc)which reduces the taken samples to almost half Nt
% N_1=R+Ns-1;
% end

% Option5:-----
% Taking only the first number of samples from the full H Matrix and
neglect the rest.
% Comment the following line if you do not want to use this option.
% -----
% H_Matrix5=H_Matrix(1:Ns*64, :); %Takes the first 64 samples out of
the 75.

% Option6:-----
% Rearrange H Matrix such that each block contains all the samples
for the
% first sensor. Then take fewer number of samples from this new H
matrix
% using a random matrix that satisfies the restricted isometric
property
% such as bernoulli matrix.
% Comment the following line if you do not want to use this option.
% -----
% H_New=[];
% rowdist=repelem([Ns Ns Ns],25); %creates an array of Ns values
repeated Nt times. here 3*25=75=Nt
% H_Cell = mat2cell(H_Matrix,rowdist); %divides H into cells or
blocks of size Ns*Nin^2, thus we will have Nt blocks
% for indx=1:Nt

```

```

%     H_New=[H_New, H_Cell{indx}]; %This will put all the cells
besides each other, thus each sensor will have all the measurments
for all the samples on the same row.
% end
%
% H_Matrix6=[];
% for indx1=1:Ns
%     H_New1=vec2mat(H_New(indx1,:),Nin^2); % arranges the each
row in H_New (where each row corresponds to a sensor),to a block of
size Nt*Nin^2
%     H_Matrix6=[H_Matrix6;H_New1]; %This matrix has Ns
blocks of size Nt*Nin^2
% end
% % load H_Matrix_EachSensorinBlock
% % Nt_new=22;
% % bernoulli=binornd(1,0.5,[Nt_new*Ns,Nt*Ns]); %1 is the number of
sample repetition which restrict the possible values to 0 and 1, 0.5
is the propability of occurance , last two parameters are the
dimensions of my sensing matrix
% % bernoulli=(bernoulli*2)-1;
% % Gaussian=randn(Nt_new*Ns,Nt*Ns);
% % Gaussian=orth(Gaussian);
% % H_lessSamples=Gaussian*H_Matrix6; %This will take less number
of samples randomly from H

% %-----
% %Finding the initial pressure
% %-----
% lamda=760; %The wavelength (nm)
% gama=0.1; %The gruneisen parameter
(dimensionless)
% eita=1; %The percentage of absorbed light
converted to heat
% F=100; %The optical fluence [J/m^2](depends
on lamda: 0.02*10^(2*((0.8)-0.7))
% mua=[0 0 0.04 0.05 1 5]*100; %Absorbtion Coefficient[1/m], idx
0:BackGround 1:Nothing 2:Fibro 3:fat 4:skin 5:blood
% disc1=load('phan2d'); %Loading the 3D breast image
% disc=disc1.phan; %Assigning values to disc
% disc2d=disc(110:173,120:183,400); %Take a 2d slice from the 3D
image
%
% %Note the initial pressure is in the range of 10kpa
% disc2d=cast(disc2d,'double'); %Changes the class of disc2d from
uint8 to double
%
% % A loop for assigning each voxel value the real initial pressure
value at lamda=760
% for i=0:5
%     disc2d(disc2d==i)=F*mua(i+1)*eita*gama; %Initial
pressure equation (p0=eita*F*ua*gama)
% end
% x0=disc2d; %Assigning initial pressure values to x0
[Pa]
%
% xi=x0(:);
p0_magnitude = 2;

```

```

p0 = p0_magnitude * loadImage('EXAMPLE_source_two.bmp');
x0=resize(p0, [N,N]);
xi=x0(:);
A=[real(H_Matrix);imag(H_Matrix)];
% A=real(H_Matrix6)+(1e-3*ir(:,1:4096));
% A=H_Matrix6;
y=A*xi;
figure(1)
histogram(xi);
title('Histogram of Real Phantom');
xlabel('Initial Pressure Intensity');
ylabel('Redundancy of each intensity value');
figure(2),
plot(xi, 'r*')
hold on

% ADMM
%-----
rho=1;
alpha=1.3;
QUIET = 0;
MAX_ITER = 5000;
[m ,n] = size(A);
% x = zeros(n,1); %Initialize the vector x
% z = zeros(n,1); %Initialize the vector z
% u = zeros(n,1); %Initialize the dual variable
x = 100*rand(n,1); %Initialize the vector x*
% x=(A'*A)'*A'*y;
z = 0*rand(n,1); %Initialize the vector z
u = 0*rand(n,1); %Initialize the dual variable
if ~QUIET
    fprintf('%3s\t%10s\t%10s\n', 'iter','r norm', 'objective');
end
% Iterations update
AAt = A*A';

PO=eye(n) - (A' * (AAt \ A));

t_start = tic; % start counting the time needed to run the whole
program
POP=(A' * (AAt \ y));
for k = 1:MAX_ITER
    % x-update
    x = PO*(z - u) + POP; %projection onto Ax=b
    % z-update with over relaxation parameter
    z_previous = z;
    % x_relaxed = alpha*x + (1 - alpha)*z_previous; %The over
relaxation will speed up the convergence of iteration process
    % z = max(0, (x_relaxed+u)-(1/rho)) - max(0, -(x_relaxed+u)-
(1/rho)); %Using formula of soft thresholding
    z = max(0, (x+u)-(1/rho)) - max(0, -(x+u)-(1/rho));
%Dual Update
    % u = u + (x_relaxed - z);
    u = u + (x - z);
    if ~QUIET
        fprintf('%3d\t\t%10.4f\t%10.4f\t%10.4f\n', k,norm(x -
z),rho*norm(z-z_previous),norm(x,1));
    end
end

```



```

        end
        if (norm(x - z)<=0.01) && (rho*norm(z-z_previous))<0.01 %if norm
2 of (x-z)<=0.001 then terminate the iterations
            break;
        end
    end
end
if ~QUIET
    toc(t_start); %End counting the time needed for finding the
iterations
end
x_hat=z;

%Comparing the image with the reconstructed one
%-----
plot(x_hat, 'go');
hold on
plot(x0(:), 'r*');
legend('Original image', 'Reconstructed image');
title('Original x vs Reconstructed x');
figure(3),
subplot(2,1,1);
imshow(x0, []);
title('Original Image');
subplot(2,1,2);
X_hat=reshape(x_hat, N,N);
imshow(X_hat, []);
title('Reconstructed Image (ADMM)');

% (1)Using Mean Square Error
%-----

MSE=mean((x0(:)-x_hat(:)).^2);
fprintf('The MSE value is %0.15f.\n',MSE);

%(2)Relative error
%-----
RE = mean((x0(:)-x_hat(:)).^2)/mean(x(:).^2);
fprintf('The relative error is %0.15f \n',RE);

%(2)Similarity index
%-----
[ssimval, ssimmap] = ssim(X_hat,x0);

fprintf('The SSIM value is %0.4f.\n',ssimval);

```

```

%-----
Code 3:
ADMM code for reconstructing an image using Distributed
implementation of ADMM basis pursuit
%-----

clc;
close all;
clear all;

load H_Matrix64_Nt75_Ns31; %loads the full H matrix which consists
of 67 sensors and 75 time samples for image size 64 by 64
[a,w]=size(cart_data);
[S,O]=size(H_Matrix);
Ns=w; %total number of sensors used to form H matrix
Nt=S/Ns; %total number of time samples based on nyquist rate,
since dimension of H is Ns*Nt X Nin*Nin then Nt=L/Ns
N=Nin; %Number of grid points

% Option:-----
% Rearrange H Matrix such that each block contains all the samples
for the first sensor and so on.
% Comment the following lines if you do not want to use this option.
% -----
H_New=[];
rowdist=repelem([Ns Ns Ns],25); %creates an array of Ns values
repeated Nt times. here 3*25=75=Nt
H_Cell = mat2cell(H_Matrix,rowdist); %divides H into cells or blocks
of size Ns*Nin^2, thus we will have Nt blocks
for indx=1:Nt
    H_New=[H_New, H_Cell{indx}]; %This will put all the cells
besides each other, thus each sensor will have all the measurments
for all the samples on the same row.
end

H_Matrix6=[];
for indx1=1:Ns
    H_New1=vec2mat(H_New(indx1,:),Nin^2); % arranges the each row
in H_New (where each row corresponds to a sensor),to a block of size
Nt*Nin^2
    H_Matrix6=[H_Matrix6;H_New1]; %This matrix has Ns
blocks of size Nt*Nin^2
end

%plot sensor distribution
figure(1)
xs=cart_data(1,:);
ys=cart_data(2,:);
plot(xs,ys,'*');
title('Sensors Distribution');xlabel('[m]');ylabel('[m]');

%-----
%Finding the initial pressure
% %-----
-----
p0_magnitude = 2;
p0 = p0_magnitude * loadImage('EXAMPLE_source_two.bmp');

```

```

x0=resize(p0, [N,N]);
xi=x0(:);

sensor_b1=[1,9,22,31];
sensor_b2=[2,11,20,30];
sensor_b3=[3,13,18,29];
sensor_b4=[4,15,16,28];
sensor_b5=[5,17,14,27];
sensor_b6=[6,19,12,26];
sensor_b7=[7,21,10,25];
sensor_b8=[8,23,24];

H1=[H_Matrix6((sensor_b1(1)-1)*Nt+1:(sensor_b1(1)-
1)*Nt+Nt,:);H_Matrix6((sensor_b1(2)-1)*Nt+1:(sensor_b1(2)-
1)*Nt+Nt,:);H_Matrix6((sensor_b1(3)-1)*Nt+1:(sensor_b1(3)-
1)*Nt+Nt,:);H_Matrix6((sensor_b1(4)-1)*Nt+1:(sensor_b1(4)-
1)*Nt+Nt,:)]];
H2=[H_Matrix6((sensor_b2(1)-1)*Nt+1:(sensor_b2(1)-
1)*Nt+Nt,:);H_Matrix6((sensor_b2(2)-1)*Nt+1:(sensor_b2(2)-
1)*Nt+Nt,:);H_Matrix6((sensor_b2(3)-1)*Nt+1:(sensor_b2(3)-
1)*Nt+Nt,:);H_Matrix6((sensor_b2(4)-1)*Nt+1:(sensor_b2(4)-
1)*Nt+Nt,:)]];
H3=[H_Matrix6((sensor_b3(1)-1)*Nt+1:(sensor_b3(1)-
1)*Nt+Nt,:);H_Matrix6((sensor_b3(2)-1)*Nt+1:(sensor_b3(2)-
1)*Nt+Nt,:);H_Matrix6((sensor_b3(3)-1)*Nt+1:(sensor_b3(3)-
1)*Nt+Nt,:);H_Matrix6((sensor_b3(4)-1)*Nt+1:(sensor_b3(4)-
1)*Nt+Nt,:)]];
H4=[H_Matrix6((sensor_b4(1)-1)*Nt+1:(sensor_b4(1)-
1)*Nt+Nt,:);H_Matrix6((sensor_b4(2)-1)*Nt+1:(sensor_b4(2)-
1)*Nt+Nt,:);H_Matrix6((sensor_b4(3)-1)*Nt+1:(sensor_b4(3)-
1)*Nt+Nt,:);H_Matrix6((sensor_b4(4)-1)*Nt+1:(sensor_b4(4)-
1)*Nt+Nt,:)]];
H5=[H_Matrix6((sensor_b5(1)-1)*Nt+1:(sensor_b5(1)-
1)*Nt+Nt,:);H_Matrix6((sensor_b5(2)-1)*Nt+1:(sensor_b5(2)-
1)*Nt+Nt,:);H_Matrix6((sensor_b5(3)-1)*Nt+1:(sensor_b5(3)-
1)*Nt+Nt,:);H_Matrix6((sensor_b5(4)-1)*Nt+1:(sensor_b5(4)-
1)*Nt+Nt,:)]];
H6=[H_Matrix6((sensor_b6(1)-1)*Nt+1:(sensor_b6(1)-
1)*Nt+Nt,:);H_Matrix6((sensor_b6(2)-1)*Nt+1:(sensor_b6(2)-
1)*Nt+Nt,:);H_Matrix6((sensor_b6(3)-1)*Nt+1:(sensor_b6(3)-
1)*Nt+Nt,:);H_Matrix6((sensor_b6(4)-1)*Nt+1:(sensor_b6(4)-
1)*Nt+Nt,:)]];
H7=[H_Matrix6((sensor_b7(1)-1)*Nt+1:(sensor_b7(1)-
1)*Nt+Nt,:);H_Matrix6((sensor_b7(2)-1)*Nt+1:(sensor_b7(2)-
1)*Nt+Nt,:);H_Matrix6((sensor_b7(3)-1)*Nt+1:(sensor_b7(3)-
1)*Nt+Nt,:);H_Matrix6((sensor_b7(4)-1)*Nt+1:(sensor_b7(4)-
1)*Nt+Nt,:)]];
H8=[H_Matrix6((sensor_b8(1)-1)*Nt+1:(sensor_b8(1)-
1)*Nt+Nt,:);H_Matrix6((sensor_b8(2)-1)*Nt+1:(sensor_b8(2)-
1)*Nt+Nt,:);H_Matrix6((sensor_b8(3)-1)*Nt+1:(sensor_b8(3)-
1)*Nt+Nt,:)]];

H1=[H1;H2];
H2=[H3;H4];
H3=[H5;H6];
H4=[H7;H8];

```

```

[L,I]=size(H_Matrix6);

K=1;
N_B=4;    %Number of blocks that H is divided into
B_rows=L/N_B; %Number of rows in each block
BL=0;
if isinteger(B_rows)==0
    B_rows=(L-rem(L,N_B))/N_B;
    B_last=B_rows+rem(L,N_B);
    BL=1;
    increment=B_rows;
end

A=cell(N_B,1); %Define A initially as an empty cell array of 4
cells
A(:)={zeros(B_rows,N^2)}; %Each cell in A has a size of B_rows by
N^2, initially all elements in the blocks are zeros
y=cell(N_B,1); %Define y initially as an empty cell array of 4
cells
y(:)={zeros(B_rows,1)}; %Each cell is of size B_rows by 1

A{1}=[real(H1);imag(H1)];
A{2}=[real(H2);imag(H2)];
A{3}=[real(H3);imag(H3)];
A{4}=[real(H4);imag(H4)];
% A{5}=[real(H5);imag(H5)];
% A{6}=[real(H6);imag(H6)];
% A{7}=[real(H7);imag(H7)];
% A{8}=[real(H8);imag(H8)];

y{1}=A{1}*xi;
y{2}=A{2}*xi;
y{3}=A{3}*xi;
y{4}=A{4}*xi;
% y{5}=A{5}*xi;
% y{6}=A{6}*xi;
% y{7}=A{7}*xi;
% y{8}=A{8}*xi;

figure(1)
histogram(xi);
title('Histogram of Real Phantom');
xlabel('Initial Pressure Intensity');
ylabel('Redundancy of each intensity value');
figure(2),
plot(xi, 'r*')
hold on

% ADMM
%-----
rho=1;
alpha=1.3;
QUIET = 0;
MAX_ITER = 700;

```

```

[m ,n] = size(A{1});
u= cell(N_B,1);
load x_rand
xx=x;
x= cell(N_B,1);
x_relaxed=cell(N_B,1);
x(:) = {xx}; %Initialize the vector x
load z_rand
z_previous=z;
u(:) = {zeros(n,1)}; %Initialize the dual variable
sum_u=zeros(n,1);
sum_x=zeros(n,1);
for r=1:N_B
sum_u=u{r}+sum_u;
end

% Iterations update
At=cellfun(@transpose,A,'UniformOutput',false);
AAAt=cellfun(@(x,y)x*y,A,At,'UniformOutput',false); %this will result
in AAAt = A*A' but arranged in cells

t_start1 = tic; % start counting the time needed to run the whole
program
for e=1:N_B
POP{e}=At{e} * (AAAt{e} \ y{e});
PO{e}=eye(n) - (At{e} * (AAAt{e} \ A{e}));
end
time1=toc(t_start1);

if ~QUIET
fprintf('%3s\t%10s\t%10s\n', 'iter','r norm', 'objective');
end

time2=[];
time3=[];
time4=[];
u_avg=sum_u/N_B;
Links_Reduced=0;
for k = 1:MAX_ITER
selection= randperm(N_B);
% x-update
for j=1:N_B
x_prev=x{j};
t_start2=tic;
x{j} =PO{j}*(z - u{j}) + POP{j}; %projection onto Ax=b
time2=[time2;toc(t_start2)];
% XI=j==selection(1)||j==selection(2)||j==selection(3);
% XI=j==selection(1);
%
XI=j==selection(1)||j==selection(2)||j==selection(3)||j==selection(4
)||j==selection(5)||j==selection(6);
%
XI=j==selection(1)||j==selection(2)||j==selection(3)||j==selection(4
);
% XI=rand>0.3 ;
XI=norm(x{j}-x_prev,2)>=0.9;
% x{j}=(1-XI)*x{j}+XI*x_prev;

```

```

x{j}=XI*x{j}+(1-XI)*x_prev;

sum_x=x{j}+sum_x;
if XI==1
    Links_Reduced=Links_Reduced+1;
end
end

sum_x=x{j}+sum_x;
% z-update with over relaxation parameter
z_previous = z;
t_start3=tic;
x_avg=sum_x/N_B;
x_avg= alpha*x_avg + (1 - alpha)*z_previous; %The over
relaxation will speed up the convergence of iteration process
u_avg=sum_u/N_B;
u_avg= alpha*u_avg + (1 - alpha)*x_avg; %The over relaxation
will speed up the convergence of iteration process
z = max(0, (x_avg+u_avg)-(1/rho)) - max(0, -(x_avg+u_avg)-
(1/rho));
time3=[time3;toc(t_start3)];

sum_x=zeros(n,1);
sum_u=zeros(n,1);
%Dual Update
for j1=1:N_B
    u_prev=u{j1};
    t_start4=tic;
    u{j1} = u{j1} + (x{j1} - z);
    time4=[time4;toc(t_start4)];
%    XU=norm(u{j}-u_prev,1)/(norm(u{j},1)+0.00001)>=0.0001;
%    XU=j1==selection(1)||j1==selection(2)||j1==selection(3);
%    XU=j1==selection(1);
%
XU=j1==selection(1)||j1==selection(2)||j1==selection(3)||j1==selecti
on(4)||j1==selection(5)||j1==selection(6);
%
XU=j1==selection(1)||j1==selection(2)||j1==selection(3)||j1==selecti
on(4);
%    XU=norm(u{1j}-x_prev,2)>=5;

%    u{j1}=(1-XU)*u{j1}+XU*u_prev;
sum_u=sum_u+u{j1};
end

if ~QUIET
    fprintf('%3d\t%10.5f\t%10.5f\t\n', k, rho*norm(z-
z_previous), norm(x_avg,1));
end
if rho*norm(z-z_previous)<=0.01 %if norm 2 of (x-z)<=0.001
then terminate the iterations
    break;
end
end
end

x_hat=z;

```

```

%Comparing the image with the reconstructed one
%-----
plot(x_hat, 'go');
hold on
plot(xi, 'r*');
legend('Original image', 'Reconstructed image');
title('Original x vs Reconstructed x');
figure(3),
subplot(2,1,1);
imshow(x0, []);
title('Original Image');
subplot(2,1,2);
X_hat=reshape(x_hat, N,N);
imshow(X_hat, []);
title('Reconstructed Image (ADMM)');

% (1)Using Mean Square Error
%-----

MSE=mean((xi-x_hat(:)).^2);
fprintf('The MSE value is %0.15f.\n',MSE);

%(2)Similarity index
%-----
[ssimval, ssimmap] = ssim(X_hat,reshape(xi,N,N));

fprintf('The SSIM value is %0.4f.\n',ssimval);

%Total Time of reconstruction
%-----
total_time1=((sum(time2)+sum(time4))/N_B)+sum(time3);
total_time2=((time1+sum(time2)+sum(time4))/N_B)+sum(time3);
fprintf('The Total time of parallel reconstruction is
%0.4f.\n',total_time2);
fprintf('The Total time without pop and po is
%0.4f.\n',total_time1);

```

```

%-----
%Code 4:
Comparing measurements of kwave and H matrix
%-----

    clc;
clear all;
close all;

Nin=32;           %Imaging grid size Nin^2
Nout=128;        %K space grid size Nout^2
d=0.1e-3;        %Grid spacing
c=1500;          %sound speed [m/s]
max_freq=c/(2*d); %Maximum frequency
Fs=max_freq*2;   %Sampling frequency
t=5e-6;          %Time period of measuring acoustic waves
Nt=t*Fs;         %number of samples

%Sensor distribution
%-----
skip=4;
sensor_grid=38;
sensor.mask = zeros(sensor_grid, sensor_grid);
sensor.mask(1, 1:skip:sensor_grid) = 1;
sensor.mask(end, 1:skip:sensor_grid) = 1;
sensor.mask(1:skip:sensor_grid, 1) = 1;
sensor.mask(1:skip:sensor_grid, end) = 1;
Ns=nnz(sensor.mask); %total number of sensors
kgrid_sensor= kWaveGrid(sensor_grid, d, sensor_grid, d);
[cart_data, order_index]=grid2cart(kgrid_sensor,sensor.mask);
xs=cart_data(1,:);
ys=cart_data(2,:);
rs_vec=[xs(:),ys(:)];

%-----
%Initial Pressure Distribution Based on Realistic data
%-----

p0_magnitude = 2;
p0 = p0_magnitude * loadImage('EXAMPLE_source_two.bmp');
x0 = resize(p0, [Nin, Nin]);

%-----
%Constructing The Forward Matrix H
%-----

%Define the imaging grid of size Nin^2
kgrid= kWaveGrid(Nin, d, Nin, d);
x=kgrid.x;
y=kgrid.y;
r_vec=[x(:), y(:)];

%Define the kspace grid of size Nout^2
kkgrid=kWaveGrid(Nout, 1, Nout, 1);
u=kkgrid.x; v=kkgrid.y;

```



```

kx=( (2*pi)/(Nout*d))*u;
ky=( (2*pi)/(Nout*d))*v;
k_vec=[kx(:), ky(:)];
k_vec_k=sqrt(kx(:).^2+ky(:).^2);

%plot sensor distribution
figure(1)
plot(xs,ys,'o');
title('Sensors Distribution');xlabel('[m]');ylabel('[m]');
grid on
tic

%Find the forward discrete fourier transform
Wfwd=zeros(Nout^2,Nin^2);
for i=1:Nout^2
    for j=1:Nin^2
        Wfwd(i,j)=(1/Nout)*exp(-sqrt(-1)*dot(k_vec(i,:),r_vec(j,:)));
    end
end

%Initialization
Winv=zeros(Ns,Nout^2);
K=zeros(Ns*Nt,Nout^2);
Q=1:Ns;

%Find the inverse discrete fourier transform
for s=1:Ns
    for i=1:Nout^2
        Winv(s,i)=(1/Nout)*exp(sqrt(-1)*dot(k_vec(i,:),rs_vec(s,:)));
    end
end

%Construct K by stacking Kt_Matrix of all time samples.
Col_one=ones(Ns,1);
delta_t=t/Nt;
for sample=0:Nt-1
    kt=(cos(c*k_vec_k(:)*(sample*delta_t)));
    kt_matrix=Col_one*kt';
    Kt_Matrix=Winv.*kt_matrix;
    K(Q,:)=Kt_Matrix;
    Q=Q(end)+1:Q(end)+Ns;
end

%Find the H Matrix
H_Matrix=K*Wfwd;
%Find measurments vector y
y=H_Matrix*x0(:);

%-----
%Simulate the PA sensor measurements using kwave
%-----

% assign the grid size and create the computational grid
PML_size =45; % size of the PML in grid points at each
side of image
PMLAlpha=2;

```

```

Nin=32;
Nout=128;
Nx = Nout;    % number of grid points in the x direction
Ny = Nout;    % number of grid points in the y direction
dx = 0.1e-3; % grid point spacing in the x direction [m]
dy = 0.1e-3; % grid point spacing in the y direction [m]
% Ns=34;      %Total num of sensors
kgrid= kWaveGrid(Nx, dx, Ny, dx);

% resize the input image to the desired number of grid points
p0_inner = resize(p0, [Nin, Nin]);
p0_outer = zeros(Nout,Nout);
p0_outer(((Nout-Nin)/2)+1:Nout-((Nout-Nin)/2),((Nout-Nin)/2)+1:Nout-
((Nout-Nin)/2))=p0_inner;
p0=p0_outer;

% assign to the source structure
source.p0 = p0;

% define the properties of the propagation medium
medium.sound_speed = 1500; % [m/s]

%assign to sensor structure
sensor.mask = cart_data;

% create the time array
dt=1/Fs;
kgrid.setTime(Nt, dt);

% set the input options
input_args = {'Smooth', false, 'PMLInside',true,'PMLSize',PML_size,
'PMLAlpha',PMLAlpha, 'PlotPML', false};

% run the simulation
sensor_data = kspaceFirstOrder2D(kgrid, medium, source, sensor,
input_args{:});

% Plotting the y measurments vs the kwave measurments of each sensor
and
% find the MSE, RE and RMSE for comparison
y=y.';
y_Matrix=vec2mat(y,Ns);
save y_Matrix y_Matrix

for w=1:Ns
figure(w)
plot(real(y_Matrix(:,w)),'->g');
hold on,
plot(real(sensor_data(w,:)),'-*k');
grid on
title('Measurments of One Sensor');
xlabel('Time Sample');
ylabel('Pressure Magnitude [Pa]');
RMSE=sqrt(mean((sensor_data(w,:)-real(y_Matrix(:,w)))'.^2));
fprintf('Sensor # %d:\n',w);

```

```
fprintf('The RMSE value is %0.4f.\n',RMSE);
MSE=mean((sensor_data(w,:)-real(y_Matrix(:,w)))'.^2);
fprintf('The MSE value is %0.15f.\n',MSE);
RE = mean((sensor_data(w,:)-
real(y_Matrix(:,w)))'.^2)/mean(real(y_Matrix(:,w)).^2);
fprintf('The relative error is %.15f \n',RE);
end
```