Time-reversal imaging

by

Marcus Hofstad

THESIS
for the degree of
MASTER OF SCIENCE

(Master i Anvendt matematikk og mekanikk)

Faculty of Mathematics and Natural Sciences
University of Oslo

August 2008

Det matematisk- naturvitenskapelige fakultet
Universitetet i Oslo

Acknowledgements

I have written this thesis in order to fulfill my masters degree in Computational Science at the University of Oslo, applied mathematics division. I would like to thank both my supervisors Leiv J. Gelius and Andreas Austeng for seeing me through this. Especially my main supervisor Leiv J. Gelius for his effort in helping me realize the physical aspects involved. Thanks to my friends who has been very supporting and last but not least my parents for their love and support.

Marcus Hofstad

Contents

Introduction

We consider the problem of locating buried objects by generating an acoustic wavefield in the medium and using data from the detected waves. Making images that displays such targets can be done with little interfering of the medium under investigation. If there are reflecting features present, acoustic sources can be excited and measured outside the medium and the detecions are used to make an image of the interior. In this thesis we study two techniques named MUSIC and backpropagation. The reader should be aware of that although MUSIC appears frequently in areas of signal processing, it is not the traditional version which recognizes a signal through its frequencies when there is additional noise present. The technique discussed herein is referred to as MUSIC in other litterateur and it is not made an exception here. Both algorithms are built upon the same theory of time-reversing acoustic waves. In Chapter 1 some of the underlying theory on this subject is presented together with additional proposals of applications. This chapter also introduces central concepts for the following work. It is discussed in a somewhat simpler form but with support of mathematics and physics describing wave propagation in a convenient way. Chapter 2 contains a complete theoretical description with focus on deriving the two algorithms. The treatment of this is general but with an underlying thought of cases that are studied here. The model for emitting and detecting waves follows the geometry of a borehole-acquisition. Sources are then placed on the surface of a medium and the measurements are done in a well within it. MUSIC and backpropagation are tested on data extracted from simulating propagating waves. It is desired to obtain data that behaves like in physical experiments, but at the same time have good control over the generated data. The purpose is to investigate the results from employing these techniques to pointlike targets present in a homogeneous background medium. It is done in a $2D$ environment by numerically simulating waves according to the physical laws of propagation. Discussion about how this is done is given in Chapter 3. Similar work has been published and the article given in [1] has inspired some of the questioning for this thesis, both in underlying theory and when testing the imaging algorithms. It is tried to make a more complete description herein, this gains awareness about the validity of theoretics versus restrictions in experiments. Results from simulated experiments are shown and discussed in Chapter 4. They are compared with images and results from ideally generated data. It is focused mainly on monochromatic waves (signals with a single frequency), but under the treatment of ideal data the question of exploiting signal bandwidth is briefly studied. This question naturally comes to mind when considering the way data is generated. Under the same treatment we will see how extending the arrays can improve the accuracy of images. We will be using limited apertures in a certain bore-hole geometry. From this

it follows that the performance of the algorithms are tried with restricted access to data. In previous work there are strong indications of MUSIC being the better technique, but we do not exclude the other because it is expected that backpropagation also can be a well performing technique. MUSIC will be tested on several target geometries, also on cases that obviously violates assumptions from the theory. A usual problem when gathering data from measurements in real or experimental data-acquisitions is the presence of noise on output recordings. This defines the final test, where MUSIC is employed to simulated data that contains noise of different levels.

Chapter 1

Wave equations and time-reversal

Intro

It is possible to focus acoustic waves back towards the source of its origin. Because of the physical nature of waves, it can be accomplished to create waves that behaves as if they where propagating backwards in time. In this chapter we study principles of time-reversal, a technique that in theory shows highly promising practical usage and has already given exciting results through experiments. If the development of time-reversal continues it can applied in a wide range of areas. Examples are medicine, military operations and seismology. As long as there is some need of locating or focusing on an object of unknown origin and the object is able to reflect waves, we are within the boundaries of possible usage. The following text begins with an easy introduction to time-reversal and at the same time mention examples of applications in more detail. We then start working with the main focus of this chapter, the underlying theory for this technique. To do this, we will use the mathematical theory describing basic waves and use it explain principles. In doing so we rediscover results that are well known which hopefully gives a smooth transition to problems that might be less familiar. At the same time we will be touching the basis of tools used throughout the thesis. As mentioned we focus on principles and try to keep things simple, but some level of mathematical skills should be in place to fully comprehend this. Mathematics is a reliable tool, so we will see that this is not just wishful thinking and fairytale's. In theory the technique works for any waves that can be found as solutions of wave equations similar to those described here. But the underlying thought here is sound waves. At the end of the chapter we look at an example on how time reversal can be applied in an iterative mode. Using arrays to emit and receive signals described as a linear transform of source excitations. This introduces a matrix description that plays an important role in this thesis and will appear in most parts of the work.

1.1 Introduction to time-reversal

A gunshot is fired at night in an area with lots of tall concrete buildings. Soon afterward a location is pinpointed and displayed on screen at the emergency department of the nearest hospital and also in a patrol car near by. What has happened is that the sound from the gunshot is recognized by microphones placed on the surrounding buildings and in a computer simulation projected back to the source. This might seem like science fiction and it still is, but with proper development it might not continue to be. In 1994 in France, a professor named Mathias Fink and his student successfully carried out an experiment of time-reversing and make a seemingly chaotic signal focus back to its source [2]. What they did was emitting a short pulse through a forest of steel rods. The received signal was hundreds of times longer because of interactions in between the rods. When the measurement was time-reversed and re-emitted it arrived at the source location again as a short pulse. The property of waves that makes time-reversal possible, is the fact that the underlying physics does not care whether they expand through space or converges towards the origin, they still obey the same equations.[1] The experiment also indicates an additional property of time-reversal. Propagating waves when they appear in space at the same time they tend to reinforce or cancel each other depending on their phase. A short signal again arriving at the source tells us that there must have been cancellations of waves when they traveled through the forest of rods the second time. We will see later that time-reversing a signal also results in change of phase, that is, peaks are displaced relative to the initial signal. Figures 1.1, 1.2 and 1.3 illustrates an example of a converging wave. These might look more like some kind of ocean wave traveling backwards in time than a burst of sound converging. But if an acoustic wave was forced to expand through only two spatial dimensions this is how the pulse could look like. The images are simulated solutions of a wave equation in two space dimensions played backwards in time. Some ocean waves are also described by similar equations as sound, but it seems as a tough challenge creating a device that reverses waves of the ocean. Manipulating sound has probably been a tradition of man since the early days of our existence. The time-reversal mirror used in Fink's experiment forms a temporal signal by measuring waves in time at chosen spatial locations. Then what came last in is emitted first out. If there are features distorting the wavefield the process can be repeated to obtain better focus. Without any compensation method is it would depend on the source to be strong compared to the distortions.

[1] This has been a known theoretical property for a long time. Named backpropagation in seismic [3] and inverse diffraction in optics [4].

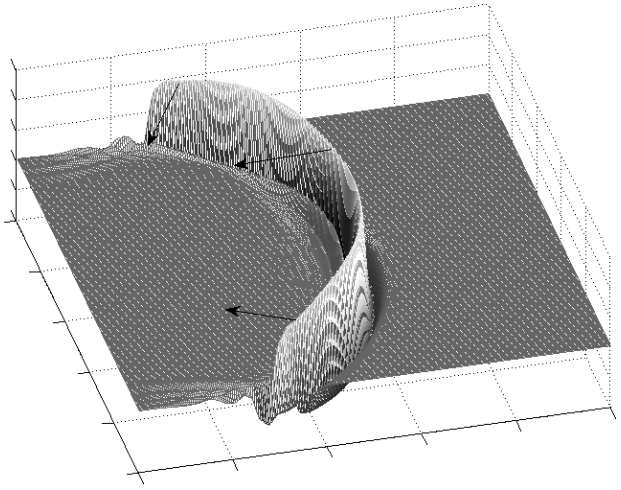

Figure 1.1: Illustration of a wavefield generated by a time-reversal operation on a measured signal. The field is pictured in space at a time-instant $t1$.

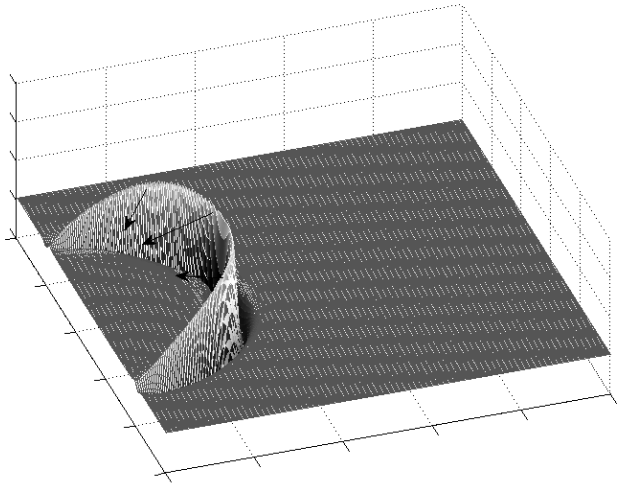

Figure 1.2: Time-instant $t2 > t1$.

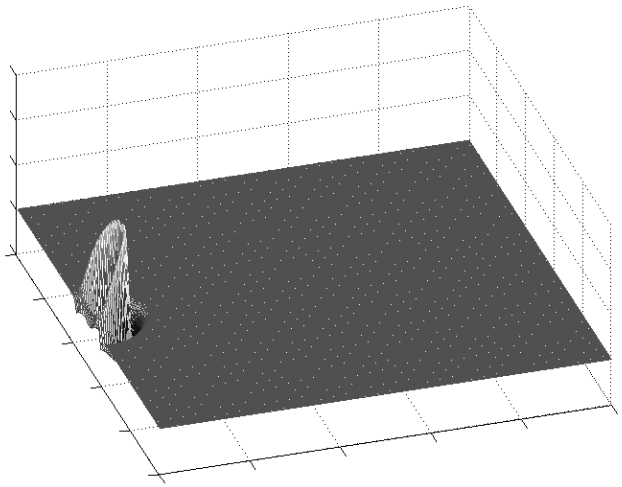

Figure 1.3: Time-instant $t3 > t2$, field is focusing in on the source.

1.2 Wave equations

To explain principles we approach the subject through a combined physical and mathematical description, this is inspired by [5]. Variants of a well known wave-equation are presented. We study simple solutions, and by adding physical interpretations we hopefully get a better intuition of how time-reversal works. The reason that different wave solutions are presented is because one is more familiar in many textbooks concerning basic wave problems. It represents plane wave solutions of a three or fewer dimensional scalar wave equation. This opens for a smooth transition to a maybe less familiar problem. When the first case is resolved it is then easier to discuss the other ones. These are simple versions of radially symmetrical equations describing waves that behaves much like sound waves expanding from a small source. They are more appropriate regarding our subject, especially when illustrating effects of time-reversal. To initialize this treatment we begin with stating the first equation and solutions to it.

1.2.1 Basic equation

$$\nabla^2 v - \frac{1}{c^2}\frac{\partial^2 v}{\partial t^2} = 0 \tag{1.1}$$
$$t \in \mathbb{R},\ 0 \leq t,\ \vec{x} \in \mathbb{R}^n$$

$\vec{x}$ denotes the spatial coordinate, t the time-variable and c is a real positive constant. Notice that we consider positive times only. Introduce another real constant w_0, and a vector of constant values:

$$\vec{k} = [k_{x_1}\ ...\ k_{x_n}]^T \tag{1.2}$$
$$\vec{k} \in \mathbb{R}^n$$

Monochromatic plane wave solutions to (1.1) can be found by separation of variables and has the form:

$$v = \psi(\vec{x}, t) = C_a e^{j(w_0 t - \vec{k}^T \vec{x})} \tag{1.3}$$

C_a is a complex constant and the following condition must be satisfied:

$$\|\vec{k}\| = \frac{w_0}{c} \tag{1.4}$$

w_0 will be referred to as the angular frequency and $\vec{k}$ denotes the wave number vector. We need to be aware of that the problem statement (1.1) and its solution could be given in any number n of spatial dimensions (in $\mathbb{R}^n$). Since we are looking at physical waves propagating through space it wouldn't make much

sense for $n > 3$ in this context. In section 1.2.3 we will make use of the one dimensional version but this is only solution oriented and the waves there expand through more than a single dimension. Observe that in (1.3) we there is a relation between $\vec{k}$ and $\vec{x}$. Notice that when they point in the same direction the vector product has its maximum and is zero when they are perpendicular. Rewrite $\psi(\vec{x}, t)$ as:

$$\psi(\vec{x}, t) = \psi_u(t - \frac{\vec{k}^T\vec{x}}{w_0}) = C_a e^{jw_0(t - \frac{\vec{k}^T\vec{x}}{w_0})} \tag{1.5}$$

The solution on the form of (1.5) makes the time and space dependency more visible. In addition ψ_u is a function of one variable when $u(\vec{x}, t) = t - \frac{\vec{k}^T\vec{x}}{w_0}$.

1.2.2 Propagation velocity

The relation of values in $\psi(\vec{x}, t)$ are given and we now proceed with the physics. Consider the constant c and the solution on the form in (1.5). Assume that the variable u doesn't vary in time. That is, we have a constant relationship between time and displacement, it is denoted a wave front. Thus the temporal derivative of u is:

$$\frac{\partial}{\partial t}u = 0 = 1 - \frac{1}{w_0}\frac{\partial(\vec{k}^T\vec{x})}{\partial t} \tag{1.6}$$

Take $\vec{x}$ to be in the same direction as $\vec{k}$, then (1.6) gives us:

$$\frac{\partial(\|\vec{k}\|\|\vec{x}\|)}{\partial t} = \|\vec{k}\|\|\frac{\partial\vec{x}}{\partial t}\| = w_0 \tag{1.7}$$

From the relation in (1.4) we find that the spatial displacement in time in the direction of $\vec{k}$ yields:

$$\|\frac{\partial\vec{x}}{\partial t}\| = c \tag{1.8}$$

Where c is the propagation velocity of the uniform medium (c is constant in all positions). Throughout this chapter solutions are evaluated only when $\vec{k}$ and $\vec{x}$ point in the same direction.

1.2.3 Oscillation and periodicity

The magnitude of $\psi_u(u) = C_a e^{jw_0u}$ is always $|C_a|$ so this gives the wave amplitude. Further more, the oscillating term e^{jw_0u} has the angular frequency w_0. If we consider the temporal behavior of $\psi_u(u)$ at a fixed point $\vec{x}_0$ in space, we have the time dependent term $e^{j(w_0t - \|\vec{k}\|\|\vec{x}_0\|)}$. Since $\|\vec{k}\|\|\vec{x}_0\|$ is just a

constant value, the time related frequency at any spatial coordinate is w_0. The wave given by $\psi_u(u)$ has periodicity 2π, thus its propagation will reappear in space after a temporal period T from any time instant t_0:

$$T = \frac{2\pi}{w_0} \tag{1.9}$$

Assume that the wave propagates from a position $\vec{x}$ to a new position $\vec{x} + \vec{\lambda}$ within a time T, then:

$$\psi_u(u(\vec{x} + \vec{\lambda}, t + \frac{2\pi}{w_0})) = \psi_u(u(\vec{x}, t))$$

Note that $\vec{\lambda}$ must also be in the same direction as $\vec{k}$. Writing out the above expression gives us:

$$e^{j(2\pi - \|\vec{k}\|\|\vec{\lambda}\|)} = 1 \tag{1.10}$$

We can find infinitely many solutions to (1.10) but the one with a meaningful physical interpretation is:

$$\lambda = \|\vec{\lambda}\| = \frac{2\pi}{\|\vec{k}\|} \tag{1.11}$$

λ is defined as the wavelength. One wavelength is inverse proportional the size of $\vec{k}$, thus the wave-number acts as a spatial frequency.

1.2.4 Symetrically radiating waves

Having physical values needed to describe basic plane-waves in a homogenous medium, we now move on to an analytical description of radially expanding waves. Radiation of sound may in several situations be thought of as such a radially expanding wavefield. If we are listening to music from a loudspeaker, there are usually small differences whether we stand in front or to the side a of the speaker. Nor would we hear very different sounds if we were above it, thus indicating that there are symmetrical properties involved. To keep things as simple as possible, we assume radial symmetry in any direction. The solutions behaves the same way anywhere on spheres or circles of equal radius. (1.1) may be transformed to spherical coordinates (r, θ, ϕ). r is the radius and the last two coordinates give horizontal and vertical angles. Under the symmetry assumption there is no dependency on angles and a solution must satisfy [5]:

$$\frac{1}{r^2}\frac{\partial^2}{\partial r^2}(r\psi_r(r,t)) - \frac{1}{c^2}\frac{\partial^2}{\partial t^2}(r\psi_r(r,t)) = 0 \tag{1.12}$$

$$r^2 = \|\vec{x}\|^2 = x^2 + y^2 + z^2$$

$$t \in \mathbb{R},\ 0 \leq t$$

ψ_r is just a function evaluation of v from (1.1) in spherical coordinates without angular dependency:

$$\psi_r(r,t) = v(\vec{x}(r), t) \tag{1.13}$$

If we consider the fact that in the solution discussed in the previous section, we only considered waves along $\vec{k}$. It could seem strange that we are able to use this to obtain a solution for waves propagating in all directions. It is not, it is enough to evaluate (1.12) in one direction. We are free to choose this to be along $\vec{k}$, we know that the behaviour will follow this pattern on any angle. It is only the size of the vector that will affect the solution. Let $v = r\psi_r(r,t)$, then the problem given in (1.12) is exactly the same as (1.1) in one spatial dimension. The solution is given in (1.3) and thus:

$$\psi_r(r,t) = \frac{C_a}{r}e^{j(w_0 t - \|\vec{k}\|r)} \tag{1.14}$$

Without discussing the details, it should be mentioned that we may well evaluate this wave solution on a plane or line in space. But it is not exactly the same as the solution that would be found using two spatial dimensions and a polar coordinate transformation in stead of spherical. In this case the spherical waves are replaced by cylindrical waves described by Hankel functions. In fact this problem is slightly harder to solve for a homogeneous medium due

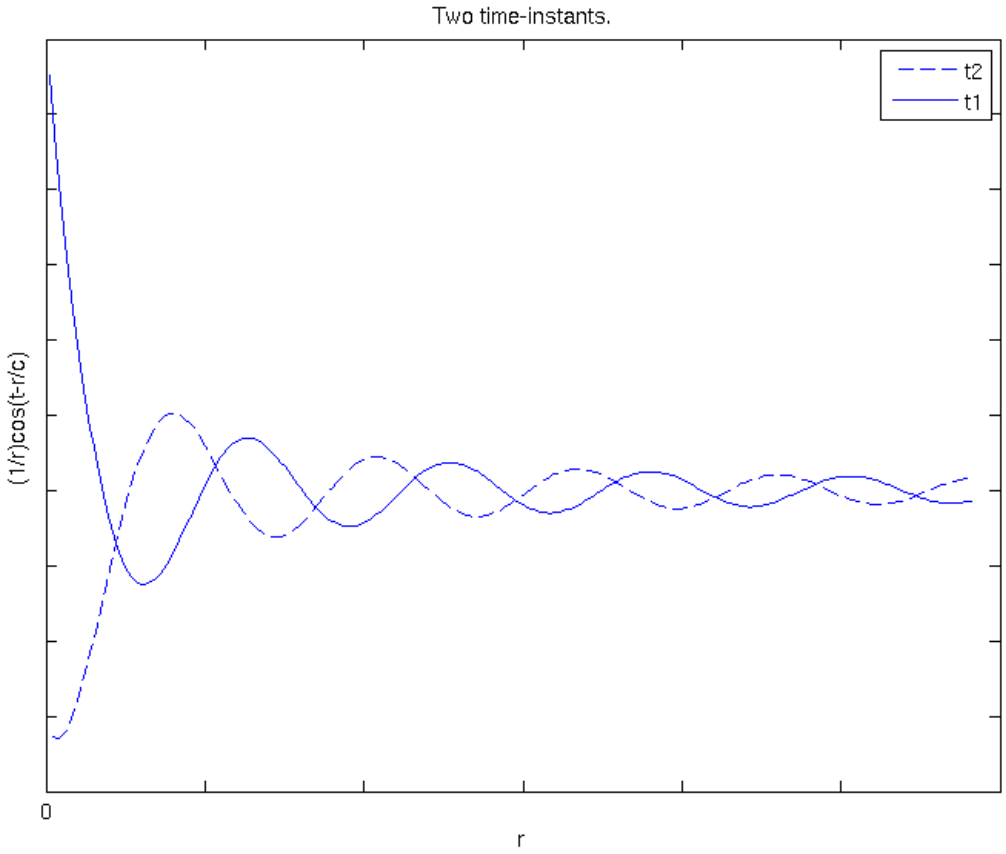

Figure 1.4: Illustrates two time-instant snapshots, $t1 < t2$ of $\frac{1}{r}\cos(t - \frac{r}{c})$.

to these Hankel-functions[2], but it has many similar features as in the three dimensional case. A common approximation with a short analytical term is given by:

$$\psi_r^{cyl}(r,t) = \frac{C_a}{\sqrt{r}} e^{j(w_0 t - \|\vec{k}\| r)} \tag{1.15}$$

For this expression to be a good approximation we need the following condition to be satisfied:

$$\frac{1}{r^2}\frac{\partial^2}{\partial r^2}(\sqrt{r}\psi_r^{cyl}(r,t)) - \frac{1}{c^2}\frac{\partial^2}{\partial t^2}(\sqrt{r}\psi_r^{cyl}(r,t)) \approx 0 \tag{1.16}$$

$$r^2 = x^2 + y^2$$

This means that we must evaluate (1.15) at a distances away from $r = 0$. When increasing the distance this converges to the exact solution for the $2D$ case. Since it is difficult to properly illustrate three-dimensional radiation, we use two-dimensional waves in the figures. And for simplicity we will use the approximated version to discuss waves and time-reversal. Results are completely analogous in $2D$ and $3D$ and if we think of the distance r as a large value the theory is well defined. Consider (1.14) and (1.15), these are damped one-variable versions of (1.3). All relations given in 1.2.2 and 1.2.3 are still in place. Notice especially that the temporal period and wavelength by definition remains the same. Because the waves are damped, they will never appear twice in space with the same amplitude but their propagations properties remains when considering only the oscillating term. As in the previous case we introduce a variable describing the phase front:

$$u_r(r,t) = t - \frac{r}{c} \tag{1.17}$$

And the one-variable expression is:

$$\psi_{u_r}^{cyl}(u_r) = \psi_{u_r}(t - \frac{r}{c}) = \frac{C_a}{\sqrt{r}} e^{jw_0 u_r} \tag{1.18}$$

Figure 1.5 display a monochromatic 2D radially expanding wave at three time instants. The figures clearly show radiation and radial symmetries. However, the amplitude decay in 2D is not so large as in 3D (asymptotically falls as $\sqrt{r}$ instead of r).

[2]The exact solution for the $2D$ wave equation in polar coordinates is given by the zero order Hankel-function of the second kind: $\psi_r^{cyl}(r,t) = H_0^{(2)}(\|\vec{k}\| r)$

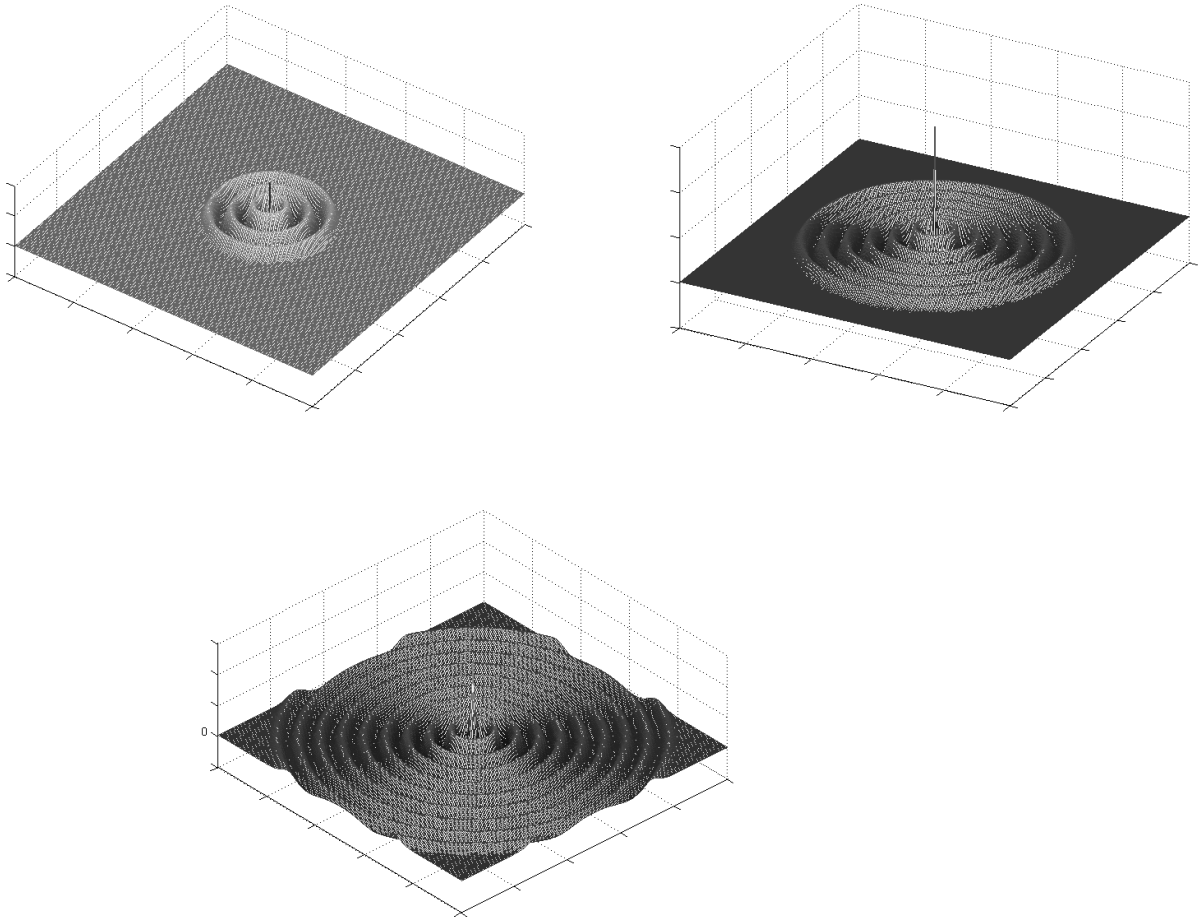

Figure 1.5: Increasing time instant snapshots at of a two dimensional wave solution. The images are results from a simulation of a wavefield that obeys the physical laws for propagation as result of a centered source in a homogeneous background.

1.3 Wave equations and time-reversal

In the solution (1.15) we have a wave radiating outwards from the origin. The propagation is initialized at $t = 0$ and the wave moves forward in time. We study time-reversal through wave-equations and the first step is to observe that if $\psi_{u_r}^{cyl}(t - cr)$ is considered a solution to a wave-equation, then $\psi_{u_r}^{cyl}(-(t + cr))$ is also a solution. It follows directly from the two times temporal derivative term. Thus any function that solves such wave-problems, will have the same feature. Again we make use of a variable having both time and place (negative phase-front):

$$u_r(r, -t) = -t - \frac{r}{c} \tag{1.19}$$

The solution with a negative phase-front is:

$$\psi_{u_r}^{cyl}(-t - \frac{r}{c}) = \frac{C_a}{\sqrt{r}} e^{-jw_0(t+\frac{r}{c})} \tag{1.20}$$

As before we evaluate the time derivative of (1.19) to find:

$$\frac{dr}{dt} = -c \tag{1.21}$$

Remember that due to causality, t can't go below zero and also that the velocity c is positive. Thus the distance r in the solution (1.20) always decays in time at the rate c. Since the source is at the origin it does not make much sense evaluating this from $t = r = 0$. It is still a radially radiating wave, but it propagates inwards. We must see this in connection with the first wave $\psi_{u_r}^{cyl}(t - \frac{r}{c})$. If we evaluate this wavefield in time on a radius $r = R$ around the origin. $\psi_{u_r}^{cyl}(-t - \frac{R}{c})$ represents waves propagating inwards with focus on the origin and with the same velocity and frequency as the outwards field. It behaves as $\psi_{u_r}^{cyl}(t - \frac{r}{c})$ propagating backwards in time. It is the time-reversed wavefield.

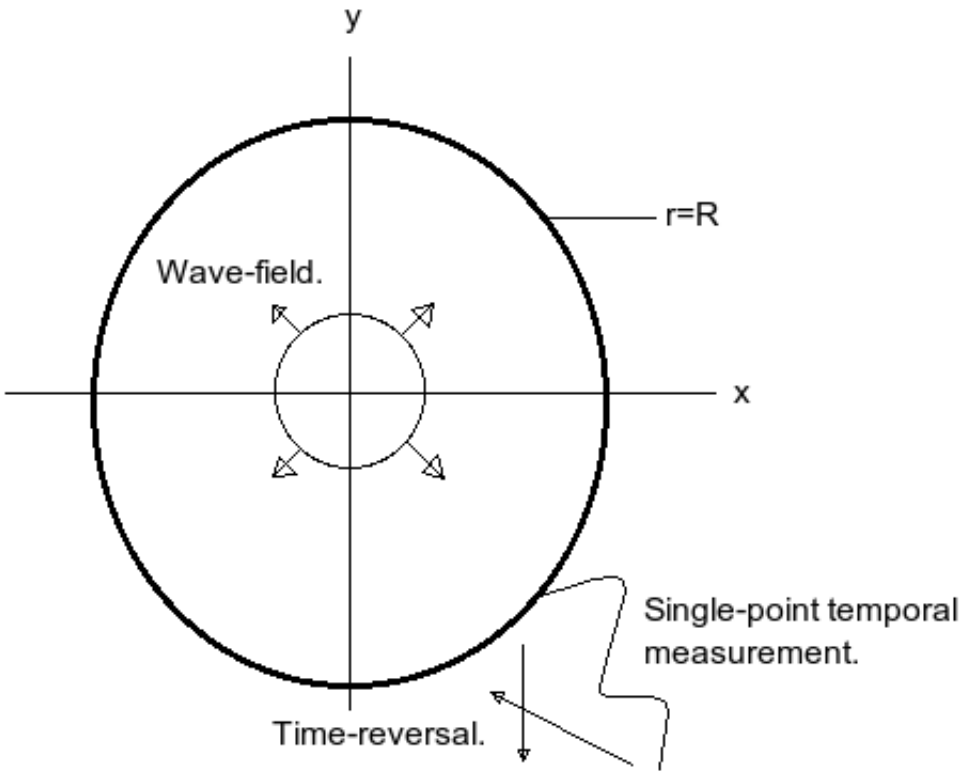

Figure 1.6: Plane evaluation of a wave radiating from the origin. A measurement is done throughout a circle of radius R. A temporal measurement at a single point $\vec{x}_0$ is illustrated. A time-reversal of the signal measured at this point would be done in a first in last out manner.

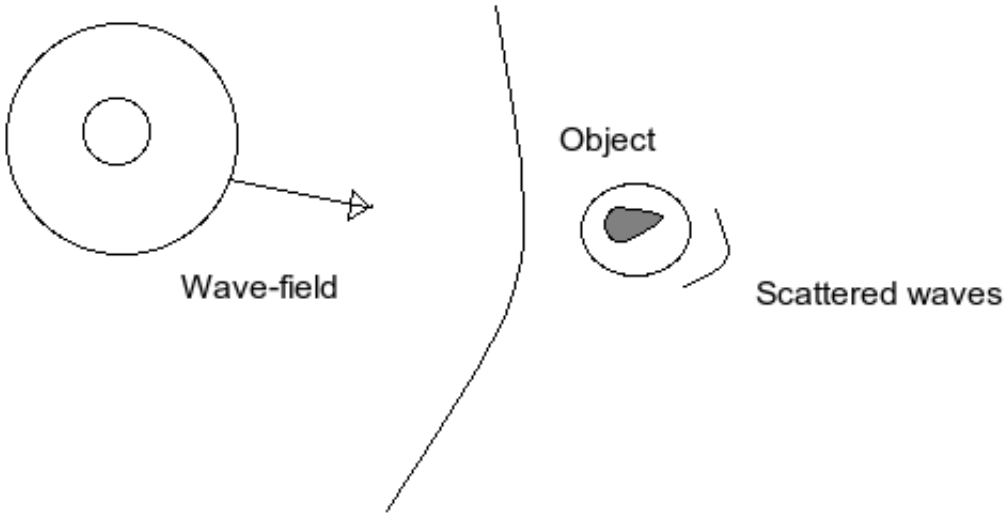

Figure 1.7: Wave-field in homogeneous medium with object of different propagation-velocity.

1.3.1 Scattering and secondary sources

In principle we have seen that time-reversal leads to refocus on sources. In applications we often wish to focus waves on objects other than the source that is physically excited. An important part, in several time-reversal applications, revolves around the fact that objects under the effect of waves, can acts as secondary sources them selves, see Figure 1.7. A well known example is the echo from a mountain wall. A propagating wave deviates from it initial way of radiating, it depends on the materials it travels through. This phenomena can occur in several ways, when these are reflections it is commonly known as scattering. A formal description can get complicated, in chapter 2 we study a special case. For now, we keep things simple and settle with the fact that, if a wave-front reach a material of different propagation velocity a reflected wavefield is initialized. In the situation with the echo a wall acts as a secondary source. Reflected waves carries information about the the reflecting medium, with time-reversal techniques we are able to create waves focusing back towards reflectors. Consider a solution to the one dimensional case of (1.1). Describe this as the incident field and denote it $\psi_I(t - \frac{x}{c})$. Assume it propagates towards a mountain wall that perfectly reflects it, see Figure 1.8. Denote the reflected waves $\psi_s(t + \frac{x}{c})$, then the total wave-solution includes both:

$$\psi(x,t) = \psi_I(t - \frac{x}{c}) + \psi_s(t + \frac{x}{c}) \tag{1.22}$$

A person standing between a friend and a reflecting wall will be able to hear both a shout form his friend and the echo. Also he should be able to separate

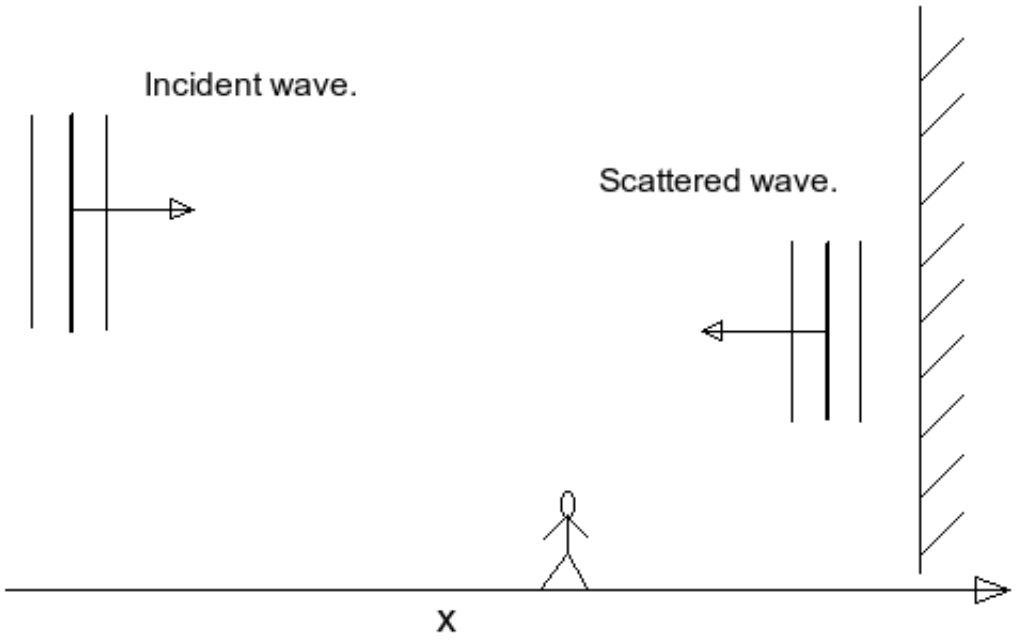

Figure 1.8: Plane-wave towards a perfectly reflecting wall.

between the two. A formal explanation of (1.22) is that both fields must meet the requirement of a wave-equation. Under the right conditions they are the only waves present and by the superposition principle the sum solves the equation.

1.3.2 Time-reversal in the frequency domain

In areas of applied mathematics making use of differential equations, especially wave solutions, a much used technique is the Fourier transform (F-T). It transforms the time dependent wave solution to an expression only depending on frequencies. It can always be reversed by an inverse F-T. The Fourier transform plays a central role also in applied time-reversal techniques. Denote the operator performing the F-T of a function $f(t)$ as $F\{f(t)\}$. It is defined by:

$$F\{f(t)\} = \int_{-\infty}^{\infty} f(t)e^{-jwt}dt \tag{1.23}$$

Where w denotes the angular frequency variable. (1.23) has many properties, three of them are essential when studying the time-reversal techniques herein. These are listed below:

- The Amplitude-spectrum is given by.

$$|F\{f(t)\}| \tag{1.24}$$

- Time reversing signals is a complex conjugation in the frequency-domain.

$$F\{f(-t)\} = F\{f(t)\}^* \tag{1.25}$$

- Convolution corresponds to multiplication in the frequency-domain.

$$F\{\int_{-\infty}^{\infty} f(t)s(t-\tau)d\tau\} = F(w)S(w) \tag{1.26}$$

The Fourier transformed signal is decomposed into its frequency components. The wave-signals we are working with in this chapter are monochromatic and displays this feature well. The amplitude spectrum of such waves peaks only at the frequency $w = w_0$. An example of a single frequency signal is given in Figure 1.9. An expression was given for the time-reversed cylindrical wave when we measured it on a radius R. In experiments we may not have enough knowledge about the source to give such an expression. We need to do a measurement within a window of time. One way to obtain the time-reversed wave is by applying (1.25), then perform an inverse transform and get the time-reversed signal. In the example of locating a gunshot where the sound is reflected and scattered several times over. We would need a map of the environment and perform simulations of the trajectory path. A convenient way to obtain the time-reversed signal in this case would be through the F-T, such a computer simulation would probably be done faster than operating in the time-domain. This brings us to the next property. Working with waves and describing their behavior often calls for a convolution. Propagation in

mediums can be described as a convolution of a mediums filtering abilities $f(t)$ and the source behavior $s(t)$. Considering (1.26) we see that this can be an intricate calculation. The result of a convolution at one time instant requires an integration over the total domain of $g(t)$. But in the frequency-domain it reduces to a simple multiplication. The F-T of measured signals is well developed an can be calculated fast on a computer. The Fourier transformed monochromatic cylindrical wave is:

$$\psi_r^{cyl}(w,r) = \int_{-\infty}^{\infty} \frac{C_a}{\sqrt{r}} e^{jw_0(t-\frac{r}{c})} e^{-jwt} dt$$

$$= \frac{C_a}{\sqrt{r}} e^{-jw_0\frac{r}{c}} \int_{-\infty}^{\infty} e^{jt(w_0-w)} dt = \frac{C_a}{\sqrt{r}} e^{-jw_0\frac{r}{c}} \delta(w_0 - w) \tag{1.27}$$

$\delta(w_0 - w)$ represents a delta distribution which peaks at $w = w_0$, see Figure 1.9. Evaluate (1.27) at a distance $r = R$ to obtain:

$$\psi_r^{cyl}(w,R) = Ae^{j\theta(w)} \tag{1.28}$$

Where the amplitude $A = |\psi_r^{cyl}(w,R)|$ and the phase is:

$$\theta(w) = \tan^{-1}\left(\frac{Im(\psi_r^{cyl}(w,R))}{Re(\psi_r^{cyl}(w,R))}\right) \tag{1.29}$$

In 1.1 it was mention that time-reversal alters the signal-phase. The time-reversed signal in terms of (1.28) is now given by the complex conjugation:

$$\psi_{TR}(w,R) = (\psi_r^{cyl}(w,R))^* = Ae^{-j\theta(w)} \tag{1.30}$$

The reversed signal of (1.28) is found by its negative phase.

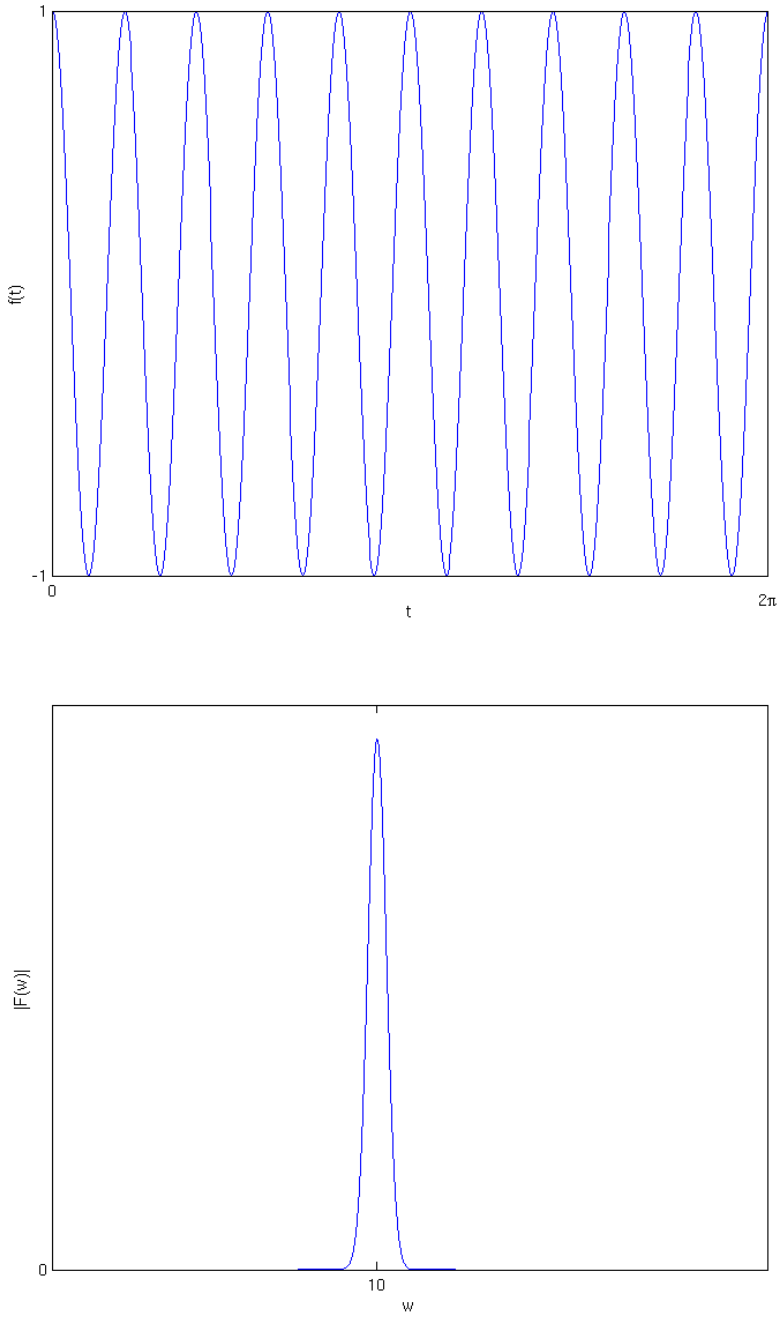

Figure 1.9: The upper figure is a Cosine with angular frequency $w_0 = 10$. Below the amplitude spectrum of the Fourier transformed signal is shown. This peaks towards infinity at $w_0 = 10$ and has zero values away from it.

1.3.3 Multiple evaluation and iterative focusing

Examples of continous evaluation of waves on a sphere or a circle has been given, it might not be possible or convenient to measure the field continuously in space. In applications it is common to use arrays of sensors, the geometries can be very different. As an example, in ultrasonics a probe with transducers is often used. Transducers can both send and receive signals. These probes come in a variety of different shapes. Some are small enough to be handheld an passed over different bodyparts. In seismic data-acquisition, large arrays of sensors can be placed inside a cable towed by a vessel, on the seafloor or inside boreholes. In any case of measuring acoustic waves the interelement spacing is usually half the wavelength of the signal. This is needed to ensure a precise reconstruction. The discussion now is about multiple source-receiver evaluation of signals. In this section our concern are the principles and we do not go far with details. Up til now we have seen a wavefield originating from the origin. This is easily extended by introducing a displaced position r_l such that:

$$r_l = \|\vec{x} - \vec{x}_l\| \tag{1.31}$$

The cylindrical wave can then radiate from from any point $\vec{x}_l$ in space:

$$\psi_{u_r}^{cyl}(t - \frac{r_l}{c}) \tag{1.32}$$

Measurements can be done at distances R_k away from a source or secondary source. Fink has proposed an experiment in ultrasonics and medicine [2]. A kidney-stone can be a highly reflective target (or scatterer) compared to its surroundings. An iterative time-reversal process may crush the stone. An array of transducers would be applied, which could send and receive signals in a time-reversed manner. Consider now signals through their frequency components. Define a vector with N source-receiver paired elements placed around the patient at positions:

$$R_1, \ \ldots \ , R_N$$

In principle, we could excite sources individually, starting with $s_1(w)$ at R_1. For each source fired we measure scattered waves at all positions to get $\vec{r}(w)$. This gives us N vectors of measurements. As mentioned in 1.3.1, propagation is medium dependent. Thus each receiver vector contains source permutations as the wave propagates from the source and is scattered back. See Figure 1.10. We may describe this as a linear transformation from sources to receivers:

$$\vec{r}(w) = \mathbf{K}\vec{s}(w) \tag{1.33}$$

K contains the information about how the signals are altered in all source-kidney-receive propagation's. Performing the time reversal on received

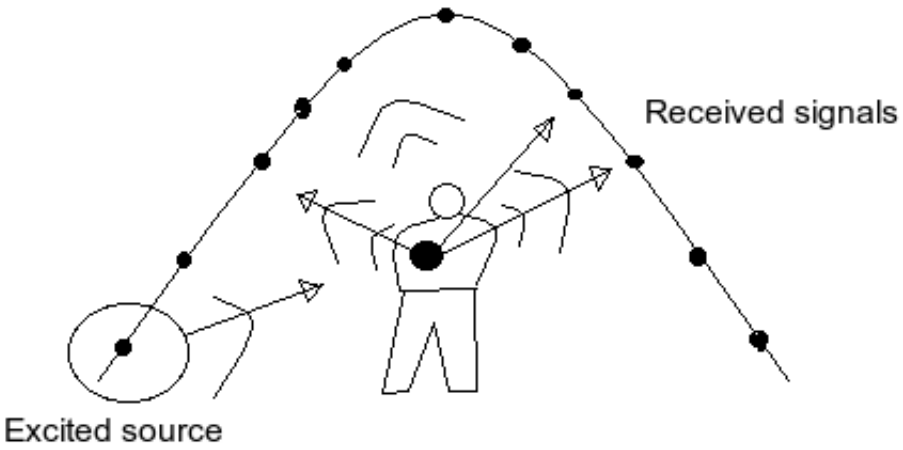

Figure 1.10: One source is fired, a reflected wavefield is generated by the kidney stone and measured in the array.

signals is done by conjunction:

$$\vec{r}^*(w) = \mathbf{K}^*\vec{s}^*(w) \tag{1.34}$$

Using this as new sources should focus back on the kidney-stone, and the received signals are now given by combining (1.33) and (1.34):

$$\vec{r}_2(w) = \mathbf{K}\mathbf{K}^*\vec{s}^*(w) \tag{1.35}$$

By further iteration, the energy will focus even better on the stone and it will eventually be crushed. However, this procedure relies on the main assumption of the kidney stone representing a strong scatter relative to its surroundings.

Chapter 2

Scattererd waves as a linear transformation and imaging

Intro

In cases of not only one, but several scattering objects it can be achieved to gain knowledge about the number of scatterers and even details about their reaction to incident waves. We will need to apply multiple sources and receivers, under given circumstances it is then possible to obtain a matrix description of the system where number of non zero eigenvalues correspond to number of scattering targets and also gives information about their scattering strengths. Rather than just stating results and building the theory from there, we will walk through the derivations of underlying theory. This gains awareness about validity and restrictions on the model, and although it may seem a bit theoretical, there are some nice results to be read for eyes belonging to the curious reader. We start very general and seek a description of the resulting wave field when a source is excited in a background medium with some object perturbing the incident field. When this is derived we will see that an approximation can be applied to get a useful expression for the resulting waves. Namely, a first order Born approximation. Having obtained this, we combine the theory with a configuration including several scatterers and arrays of sources and receivers. Which in turn, leads us to a matrix description of the complete system. The central part of this description is the transfermatrix which gives a mapping of source excitations to received data. It can also be used the other way around, this relies on the assumption of a reciprocal background. It is with this matrix we are able to extract the much wanted information about the scatterers. Having the transfermatrix we will decompose it with respect to its eigenvalues and their related eignvectors. With these we present two imaging techniques that exploits their orthogonal nature to form images that reveals the scattering targets locations in the background medium.

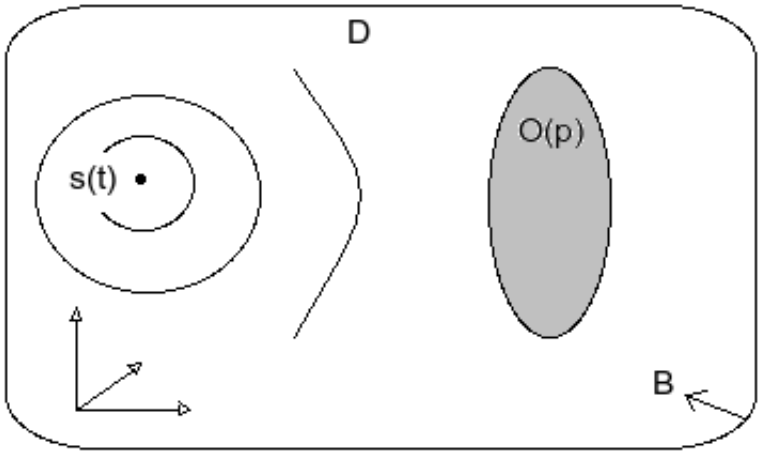

Figure 2.1: Wave field generated from a source $s(t)$ at $\vec{x}_s$, waves are considered inside an arbitrary domain D including a subregion $O(p)$.

2.1 Obtaining an expression for scattererd waves

Consider Figure 2.1, there are five quantities involved which will be used throughout the following derivation. Below a description is given:

- D: Total domain including a perturbing area.
- B: Boundary of the domain encapsulating D.
- $O(p)$: Area of interest where $p(\vec{x})$ is defined.
- $p(\vec{x})$: Spatial perturbation.
- wave field: radiating from the point source $s(t)$ at $\vec{x}_s$.

Note that the situation in Figure 2.1 could well describe a field onto more than one region. The domain $O(p)$ would then consist of sub-regions and $p(\vec{x})$ defined accordingly. We seek the field at all $\vec{x} \in D$, including $O(p)$ as the solution to a wave-equation. Denote the wave-field $\psi(\vec{x}, \vec{x}_s, t)$ and the problem can be stated:

$$(\nabla^2 - \frac{1}{v(\vec{x})^2}\frac{\partial^2}{\partial t^2})\psi(\vec{x}, \vec{x}_s, t) = -\delta(\vec{x} - \vec{x}_s)s(t) \tag{2.1}$$

$$\vec{x} \in D$$

This is the field generated from a temporal source at $\vec{x}_s$ with $v(\vec{x})$ as the space-dependent velocity profile of D. Applying a temporal Fourier transform gives us [6]:

$$(\nabla^2 + \frac{w^2}{v(\vec{x})^2})\psi(\vec{x}, \vec{x}_s, w) = -\delta(\vec{x} - \vec{x}_s)S(w) \tag{2.2}$$

$$\vec{x} \in D$$

w is the angular frequency and $S(w)$ represents the source spectrum. The the time derivative term in (2.1) is replaced by a multiplication with $-w^2$. From here on and out we will be working in the $(\vec{x}, w)$-domain. But since an inverse Fourier transform can be done in every step of the way, most of the physical interpretations is related to waves propagating forward in time. Let $c(\vec{x})$ be the velocity of the background when no perturbation is present and define $p(\vec{x})$ such that:

$$p(\vec{x}) = \begin{cases} 0 & \vec{x} \notin O(p) \\ c_s(\vec{x}) & \vec{x} \in O(p) \end{cases} \tag{2.3}$$

The following definition of $v(\vec{x})$ includes the total velocity profile in a convenient way:

$$v(\vec{x})^2 = c(\vec{x})^2 \frac{1}{1 + p(\vec{x})} \tag{2.4}$$

Using (2.4) as the expression for the velocity profile in (2.2) we obtain:

$$(\nabla^2 + \frac{w^2}{c(\vec{x})^2})\psi(\vec{x}, \vec{x}_s, w) = -\delta(\vec{x} - \vec{x}_s)S(w) - p(\vec{x})\frac{w^2}{c(\vec{x})^2}\psi(\vec{x}, \vec{x}_s, w) \tag{2.5}$$

The conveniense of defining $v(\vec{x})$ as in (2.4) can be seen in (2.5). If we consider the position vector at locations excluding $O(p)$ then $p(\vec{x}) = 0$ and (2.5) is an expression similar to that in (2.2). Only now the velocity profile concerns only the background $c(\vec{x})$. We may divide the problem by introducing the solution as a superposition of the background and the scattered soution:

$$\psi(\vec{x}, \vec{x}_s, w) = \psi_I(\vec{x}, \vec{x}_s, w) + \psi_s(\vec{x}, \vec{x}_s, w) \tag{2.6}$$

Solving (2.5) is equivalent to solve the system of differential equations:

$$(\nabla^2 + \frac{w^2}{c(\vec{x})^2})\psi_I(\vec{x}, \vec{x}_s, w) = -\delta(\vec{x} - \vec{x}_s)S(w) \tag{2.7}$$

$$(\nabla^2 + \frac{w^2}{c(\vec{x})^2})\psi_s(\vec{x}, \vec{x}_s, w) = -p(\vec{x})\frac{w^2}{c(\vec{x})^2}\psi(\vec{x}, \vec{x}_s, w) \tag{2.8}$$

The system has dependent solutions but (2.7) is the most familiar problem and we approach this first. This describes waves propagating in the background when no perturbations are present. In a homogeneous background ($c(\vec{x})$ is constant) the basic solution is a spherical wave for $\vec{x} \in \mathbb{R}^3$ and a cylindrical wave when $\vec{x} \in \mathbb{R}^2$. To solve for scattered waves as in (2.8), an integral solution can be obtained by employing Green's theorem as shown below. The Greens

function or impulse response of the background can be defined as the function $g(\vec{x}, \vec{y}, w)$ solving the equation [6]:

$$(\nabla^2 + \frac{w^2}{c(\vec{x})^2})g(\vec{x}, \vec{y}, w) = -\delta(\vec{x} - \vec{y}) \tag{2.9}$$

That is, the impulse response from a unit-source at $\vec{y}$ measured at a position $\vec{x}$. In a reciprocal medium we have the following spatial symmetry property for the Green's function:

$$g(\vec{y}, \vec{x}, \cdot) = g(\vec{x}, \vec{y}, \cdot) \tag{2.10}$$

Physically this means that there are no difference in the resulting field whether the unit pulse is sent from $\vec{x}$ to $\vec{y}$ or the other way around. This depends on the medium, so when we make the assumption of reciprocity, the validity of results obtained using $g(\vec{x}, \vec{y}, \cdot)$ will be restricted to such. By comparing (2.9) and (2.7), the background wavefield can be written as:

$$\psi_I(\vec{x}, \vec{x}_s, w) = S(w)g(\vec{x}, \vec{x}_s, w) \tag{2.11}$$

Let $u(\vec{x}, w)$ be any smooth scalar valued function defined on a compact domain D, then **Green's theorem** states that [6]:

$$\int_{\vec{x}\in D} \left[g(\vec{x}, \vec{y}, w)\nabla^2 u(\vec{x}, w) - u(\vec{x}, w)\nabla^2 g(\vec{x}, \vec{y}, w) \right] d\vec{x}$$

$$= \int_{\vec{x}\in B} [g(\vec{x}, \vec{y}, w)\nabla u(\vec{x}, w) - u(\vec{x}, w)\nabla g(\vec{x}, \vec{y}, w))] \cdot \vec{n} dS \tag{2.12}$$

$\vec{n}$ is the unit normal vector to the surface B enclosing D. In short terms, this states that integrating a function over a domain in space can be reduced to a surface integral defined by B [6]. Figure 2.2 illustrates the values involved. The Green's theorem can now be employed to solve for the scattered field by setting $u = \psi_s(\vec{x}, \vec{x}_s, w)$ in (2.12):

$$\int_{\vec{x}\in D} g(\vec{x}, \vec{y}, w)\nabla^2 \psi_s(\vec{x}, \vec{x}_s, w) - \psi_s(\vec{x}, \vec{x}_s, w)\nabla^2 g(\vec{x}, \vec{y}, w) d\vec{x}$$

$$= \int_{\vec{x}\in B} [g(\vec{x}, \vec{y}, w)\nabla \psi_s(\vec{x}, \vec{x}_s, w) - \psi_s(\vec{x}, \vec{x}_s, w)\nabla g(\vec{x}, \vec{y}, w)] \cdot \vec{n} dS \tag{2.13}$$

Consider the left side integral of (2.13), the last term of the integrand is known from (2.9) and $\nabla^2 \psi_s(\vec{x}, \vec{x}_s, w)$ can be obtained from (2.8), (the problem-statements it self). The only unknown is thus $\psi_s(\vec{x}, \vec{x}_s, w)$ and that is exactly what we are after. The right side integral however is not that straight forward. In order to get something useful from (2.13) we make use of the fact that the position vector is evaluated only at the surface B. The right side integral is the flux of $\vec{F} = g(\vec{x}, \vec{y}, w)\nabla u(\vec{x}, w) - u(\vec{x}, w)\nabla g(\vec{x}, \vec{y}, w))$ through the entire

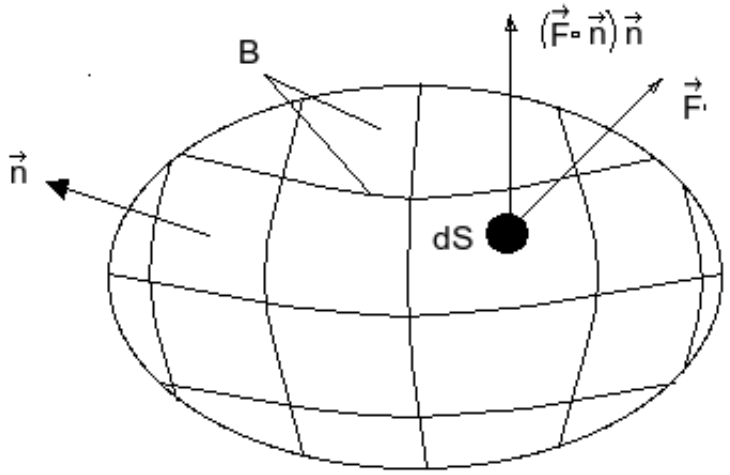

Figure 2.2: Arbitrary Domain D with boundary B. (Figure is for illustration-purposes only). Figure diplays flux of $\vec{F}$ through boundary-element dS on B. $\vec{F} = g(\vec{x},\vec{y},w)\nabla u(\vec{x},w) - u(\vec{x},w)\nabla g(\vec{x},\vec{y},w)$.

boundary. As long as the unit source and source coordinate, $\vec{y}$ and $\vec{x}_s$, are located away from the boundary they are excluded by the integration. Knowing this we may let the boundary be infinitely far away without loss of generality in the expression. From a physical point of view this is nice. Because in many situations of wave propagation evaluated at distances far from the source, the field tends to zero.

Far-field conditions
Assume that:

$$\lim_{\vec{x}\to\vec{z}} \psi(\vec{x},\vec{x}_s,w) = 0 \qquad (2.14)$$
$$\vec{z} \in B$$

That is, when $\vec{x}$ tends to positions on the vanishing boundary, the outgoing waves tends to zero. Remember now the dependency of the perturbed and the incident solution. Considering (2.6) and (2.11) it follows that if $\psi(\vec{x},\vec{x}_s,w) \to 0$ throughout B, then $g(\vec{x},\vec{y},w)$ and $\psi_s(\vec{x},\vec{x}_s,w)$ must follow the same behavior. Under the far field conditions the right side of (2.13) vanish. We are left with:

$$\int_{\vec{x}\in D} \left[\nabla^2\psi_s(\vec{x},\vec{x}_s,w)g(\vec{x},\vec{y},w) - \psi_s(\vec{x},\vec{x}_s,w)\nabla^2 g(\vec{x},\vec{y},w)\right] d\vec{x} = 0 \qquad (2.15)$$

Substituting (2.8) and the definition of $g(\vec{y},\vec{x},w)$, given in (2.9), in the above expression yields:

$$\int_{\vec{x}\in D} \left[\frac{w^2}{c(\vec{x})^2} p(\vec{x}) g(\vec{x},\vec{y},w)\psi(\vec{x},\vec{x}_s,w) + \Psi_s(\vec{x},\vec{x}_s,w)\delta(\vec{x}-\vec{y})\right] d\vec{x} = 0$$

Which can be recognized as:

$$\psi_s(\vec{y}, \vec{x}_s, w) = \int_{\vec{x} \in D} \frac{w^2}{c(\vec{x})^2} g(\vec{x}, \vec{y}, w) \psi(\vec{x}, \vec{x}_s, w) p(\vec{x}) d\vec{x} \tag{2.16}$$

Remember now that $p(\vec{x}) = 0$ for $\vec{x} \notin O(p)$ so the integrand will be zero outside $O(p)$. Another thing is, that when we want to study the perturbed field at a position $\vec{y}$, we need to evaluate the impulse response (of the background) from that coordinate to all positions $\vec{x} \in O(p)$. But when dealing with reciprocal backgrounds we may apply (2.10). The final solution for the perturbed waves yields:

$$\psi_s(\vec{y}, \vec{x}_s, w) = \int_{\vec{x} \in O(p)} \frac{w^2}{c(\vec{x})^2} g(\vec{y}, \vec{x}, w) \psi(\vec{x}, \vec{x}_s, w) p(\vec{x}) d\vec{x} \tag{2.17}$$

Which is the so called Lippman-Schwinger equation [7].

Born approximation

The perturbed field (2.17) involves the total wavefield $\Psi(\vec{x}, \vec{x}_s, w)$ evaluated at points inside our region $O(p)$. Consider (2.6), if $\Psi_s(\vec{x}, \vec{x}_s, w)$ makes a small contribution compared to the incident waves, a first order Born approximation may be sufficient [8]:

$$\psi_s(\vec{y}, \vec{x}_s, w) = \int_{\vec{x} \in O(p)} \frac{w^2}{c(\vec{x})^2} g(\vec{y}, \vec{x}, w) \psi_I(\vec{x}, \vec{x}_s, w) p(\vec{x}) d\vec{x} \tag{2.18}$$

Where $\Psi_I(\vec{x}, \vec{x}_s, w)$ substitutes the total field inside the integral. This equation is the foundation for most of the work from here on and out. How well the approximation works must be seen in context with the situation it is applied. In any case we should get better results as $\epsilon \to 1$ in:

$$\frac{|\psi_I(\vec{x}, \vec{x}_s, w) - \psi_s(\vec{x}, \vec{x}_s, w)|}{|\psi_I(\vec{x}, \vec{x}_s, w)|} = \epsilon \tag{2.19}$$

2.2 Point scatterers in an homogeneous background

2.2.1 Single scattering

Assume that the background medium is homogeneous with propagation velocity c and that $O(p)$ defines a region of different but constant propagation velocity c_s. The velocity profile from (2.4) takes the form:

$$v(\vec{x}) = \begin{cases} c & \vec{x} \notin O(p) \\ c_s & \vec{x} \in O(p) \end{cases} \tag{2.20}$$

Substitute (2.20) for $v(\vec{x})$ in (2.4):

$$p(\vec{x}) = \begin{cases} (\frac{c}{c_s})^2 - 1 & \vec{x} \in O(p) \\ 0 & \vec{x} \notin O(p) \end{cases} \tag{2.21}$$

Using this relation in (2.18) we get:

$$\psi_s(\vec{y}, \vec{x}_s, w) = \int_{\vec{x} \in O(p)} w^2 (\frac{1}{c_s^2} - \frac{1}{c^2}) g(\vec{y}, \vec{x}, w) \psi_I(\vec{x}, \vec{x}_s, w) d\vec{x} \tag{2.22}$$

Let $O(p)$ be only a single point, denote this $\vec{x}_i$ and its velocity $c_{s,i}$, we then have to redefine $p(\vec{x})$ by using a spatial delta-function at the singularity:

$$p_i(\vec{x}) = \delta(\vec{x} - \vec{x}_i)(\frac{c^2}{c_{s,i}^2} - 1) \tag{2.23}$$

We can now use (2.23) in (2.22) and integrate out the delta function to obtain:

$$\psi_s(\vec{y}, \vec{x}_s, w) = \tau_i(w) g(\vec{y}, \vec{x}_i, w) \psi_I(\vec{x}_i, \vec{x}_s, w) \tag{2.24}$$

$$\tau_i(w) = (\frac{1}{c_{s,i}^2} - \frac{1}{c^2}) w^2. \tag{2.25}$$

$\tau_i(w)$ will be referred to as the **scattering-potential** or **scattering-strength**.
If we look back at the solution for the incident problem (2.11) we can give an analogous physical interpretation of (2.24). The incident field generates a secondary source at $\vec{x}_i$ scaled with strength $\tau_i(w)$. Scattered waves that are generated propagates away according to the Greens-function. An illustration is given in Figure 2.3.

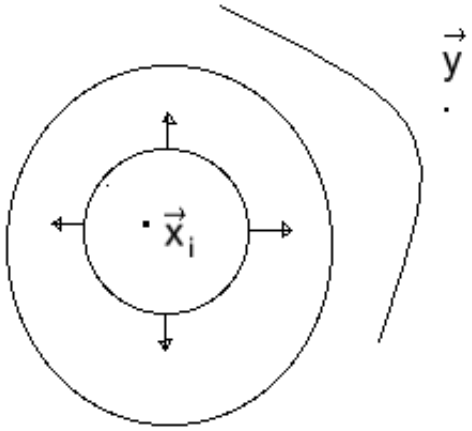

Figure 2.3: Field radiating from a point source: $\tau_i(w)\psi_I(\vec{x}_i, \vec{x}_s, w)$.

2.2.2 Born multiple scattering

In the beginning of 2.1 it was stated that $O(p)$ could well be made up of a set of sub regions and the equations describing the waves are still valid. Consider now M points of velocity perturbations in the homogeneous background. Mathematically we describe the situation by letting i take on M values and define: $O(p) = \{\vec{x}_i \in \mathbb{R}^3 : i = 1, 2, ..., M\}$. The perturbed waves under the Born approximation becomes [1]:

$$\begin{aligned}\psi_s(\vec{y}, \vec{x}_s, w) &= \tau_i(w) \int_{x \in O(p)} g(\vec{y}, \vec{x}, w)\psi_I(\vec{x}, \vec{x}_s, w)\delta(\vec{x} - \vec{x}_i)d\vec{x} \\ &= \sum_{i=1}^{M} \tau_i(w)g(\vec{y}, \vec{x}_i, w)\psi_I(\vec{x}_i, \vec{x}_s, w) \qquad (2.26)\end{aligned}$$

The expression in (2.26) states that the field at position $\vec{y}$ (for all times) is the sum of the individual interactions between the incident field and scatters. Notice that it does not include multiple scattering. Since cases with two or more targets must describe scattered waves impinging on each other these are excluded under Born approximation, see Figure 2.4. The error-expression (2.19) given together with the Born approximation states that if ϵ tends to one, we get close to an exact description. This does not imply that once ϵ increases the approximation goes bad. As mentioned we need to evaluate the circumstances. With appropriate geometries and temporal measurements we may get accurate results. As an example, consider Figure 2.5. Assume a temporal measurement within a window of time is performed. Multiple

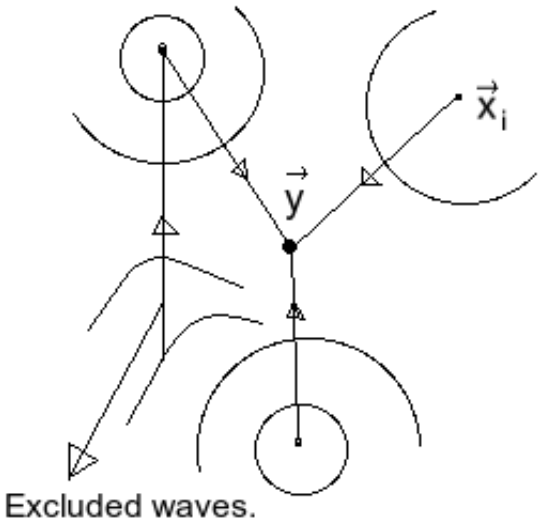

Figure 2.4: Only waves along the direct lines are accounted for at $\vec{y}$ in the solution when using Born approximation.

scattering can not be seen from the measuring point of view. If the purpose of the measurement was to record scattered waves from both targets we could achieve reliable data. We have the solution for the incident field given in (2.11). Finally, substitute this in (2.26) to obtain:

$$\psi_s(\vec{y}, \vec{x}_s, w) = \sum_{i=1}^{M} g(\vec{y}, \vec{x}, w)\tau_i(w)g(\vec{x}_i, \vec{x}_s, w)S(w) \tag{2.27}$$

The above expression is a main result and builds a foundation for the remaining discussion of this chapter.

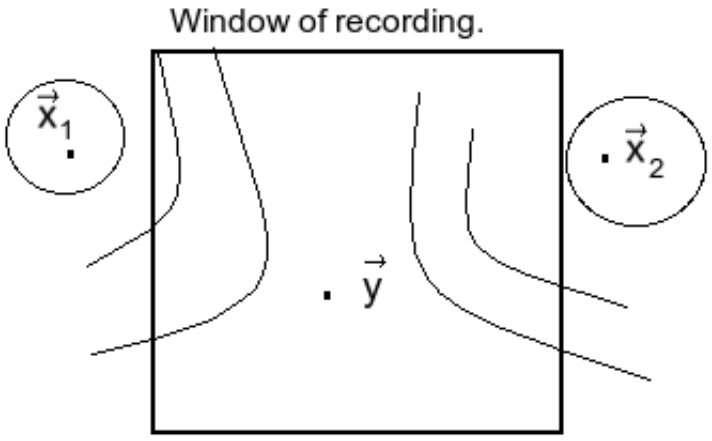

Figure 2.5: A wavefield is measured within a time window such that no multiple scattering is detected.

2.2.3 Born scattering and the transfermatrix

For the point scatterer case in the previous section we now develop a matrix formulation. The source coordinate was given as $\vec{x}_s$ and is arbitrary, but we now open the possibility for several positions in the same configuration. A source may be located at any position $\vec{x}_j^s$ for $j = 1, 2, \ ... \ , N$. The spectrum of the excited signal at $\vec{x}_j^s$ is denoted $S_j(w)$. In the same manner we may measure the scattered wave-field at $\vec{y}_l$ for $l = 1, 2, \ ... \ , L$. Using these position vectors in (2.27) we get:

$$\psi_s(\vec{y}_l, \vec{x}_j^s, w) = \sum_{i=1}^{M} g(\vec{y}_l, \vec{x}_i, w)\tau_i(w)g(\vec{x}_i, \vec{x}_j^s, w)S_j(w) \tag{2.28}$$

$$l = 1, 2, \ ... \ L.$$

$$j = 1, 2, \ ... \ N.$$

Considering (2.28), we find that impulse responses are either from or to scatterer locations. The ones to the left in the expression are to receiver coordinates and those to the right from source coordinates. Arrange the responses in matrices as defined below:

$$\mathbf{U} = \begin{bmatrix} g(\vec{y}_1, \vec{x}_1, w) & g(\vec{y}_1, \vec{x}_2, w) & ... & g(\vec{y}_1, \vec{x}_M, w) \\ g(\vec{y}_2, \vec{x}_1, w) & g(\vec{y}_2, \vec{x}_2, w) & ... & g(\vec{y}_2, \vec{x}_M, w) \\ ... & ... & ... & \\ g(\vec{y}_L, \vec{x}_1, w) & g(\vec{y}_L, \vec{x}_2, w) & ... & g(\vec{y}_L, \vec{x}_M, w) \end{bmatrix} \tag{2.29}$$

$$\mathbf{V}^T = \begin{bmatrix} g(\vec{x}_1, \vec{x}_1^s, w) & g(\vec{x}_1, \vec{x}_2^s, w) & ... & g(\vec{x}_1, \vec{x}_N^s, w) \\ g(\vec{x}_2, \vec{x}_1^s, w) & g(\vec{x}_2, \vec{x}_2^s, w) & ... & g(\vec{x}_2, \vec{x}_N^s, w) \\ ... & ... & ... & \\ g(\vec{x}_M, \vec{x}_1^s, w) & g(\vec{x}_M, \vec{x}_2^s, w) & ... & g(\vec{x}_M, \vec{x}_N^s, w) \end{bmatrix} \tag{2.30}$$

T denotes the matrix-transpose. Also arrange the scatterer strengths in a matrix:

$$\mathbf{\Sigma} = \begin{bmatrix} \tau_1(w) & 0 & 0 & ... \\ 0 & \tau_2(w) & 0 & ... \\ ... & & ... & ... \\ ... & & & \tau_M(w) \end{bmatrix} \tag{2.31}$$

$\mathbf{\Sigma}$ is a square $M \times M$ diagonal matrix. From (2.25) we know that these are all real valued. Let the monochromatic source vector $\vec{S}(w)$ of dimension N represent the transmitted signals:

$$\vec{S}(w) = [S_1(w) \ ... \ S_N(w)]^T \tag{2.32}$$

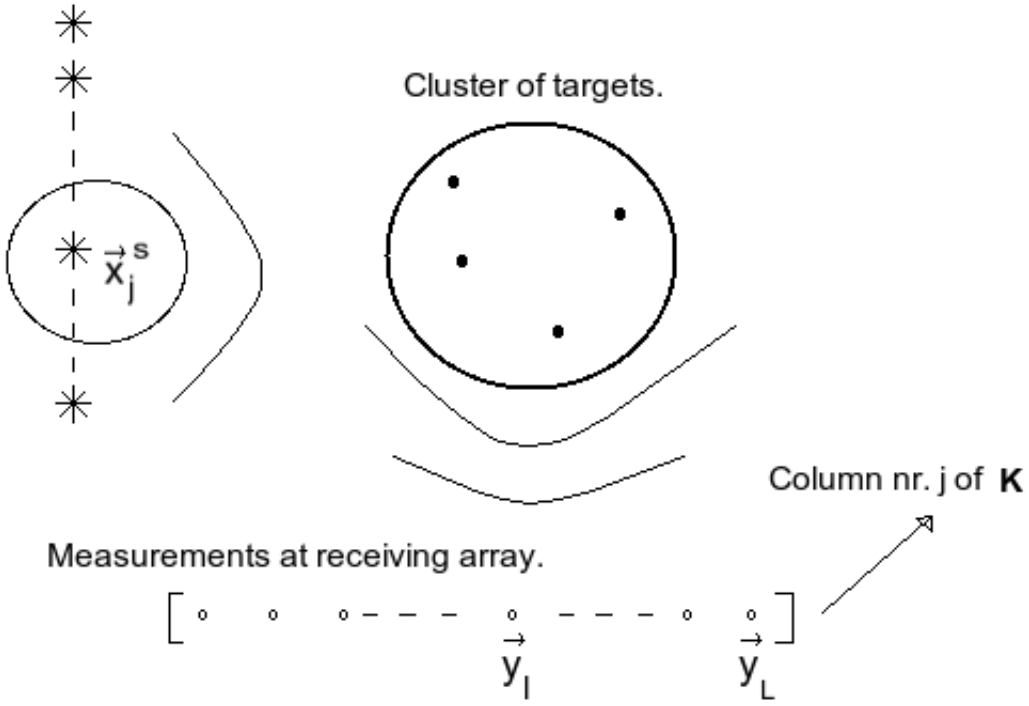

Figure 2.6: Single source excitation. The receiving output gives a column of **K**.

Alos let $\vec{R}(w)$ represent the data measured along the receiver array positions $\vec{y}_l$ (dimension L). Then (2.28) can be written as a matrix system [9]:

$$\vec{R}(w) = \mathbf{K}\vec{S}(w) \tag{2.33}$$

$$\mathbf{K} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \tag{2.34}$$

K is an $L \times N$ matrix projecting source vectors on to the receiver space. It is referred to as the **Transfermatrix**. Note that we have only considered the scattered waves at the receiver spaces. The total wavefield measurement would also need an incident wave vector but we now want to explore features of the scattered field and do not include this in the description. **K** projects the scattered field only. Columns of **K** are given by a one to one manner between single source excitations and receiver outputs, see Figure 2.6.[1] In (2.34), a singularvalue like decomposition of **K** is given. Column i of **U** corresponds to scatter-potential $\tau_i(w)$, **U** thus have all scatter-receiver relations. Analogously columns of **V** are related to the scattering potentials and have all source-scatter relations.

[1]Since detection in the receiver array follows from a single source excitation. Each source that emits a temporal signal will have a corresponding receiver vector. The transfermatrix **K** can thus easy be experimentally computed.

2.2.4 Singular value decomposition of the transfermatrix

Rewrite the matrix **U** on the form:

$$\mathbf{U} = [\vec{g}_1^r(w) \ ... \ \vec{g}_M^r(w)] \tag{2.35}$$

Where $\vec{g}_i^r(w)$ represents the collection of Greens functions that falls along column i:

$$\vec{g}_i^r(w) = [g(\vec{y}_1, \vec{x}_i, w) \ ... \ g(\vec{y}_L, \vec{x}_i, w)]^T \tag{2.36}$$

By analogy the matrix **V** can be rewritten as:

$$\mathbf{V} = [\vec{g}_1^s(w) \ ... \ \vec{g}_M^s(w)] \tag{2.37}$$

With:

$$\vec{g}_i^s(w) = [g(\vec{x}_i, \vec{x}_1^s, w) \ ... \ g(\vec{x}_i, \vec{x}_N^s, w)]^T \tag{2.38}$$

The underlying assumption of Born scattering is that the various scatterers do not interact. In addition it is now assumed that the targets are perfectly resolved. This implies ideal point spread functions with respect to both source and receiver arrays. Mathematically this is formulated [1]:

$$(\vec{g}_i^r(w))^\dagger(\vec{g}_j^r(w)) = 0 \tag{2.39}$$

$$i \neq j$$

and:

$$(\vec{g}_i^s(w))^\dagger(\vec{g}_j^s(w)) = 0 \tag{2.40}$$

$$i \neq j$$

By imposing the reciprocity principle of the Greens functions, (2.39) and (2.40) have the same interpretation. $\vec{g}_i^q(w)$ represents data measured along array $q \in \{r, s\}$ for a scatterer (impulse source) at position j. Multiplying with $(\vec{g}_i^q(w))^\dagger$ means backpropagating(focusing) these data to a possible scatterer position i. Introduce the row normalized versions of **U** and **V**:

$$\mathbf{U} = \hat{\mathbf{U}}\Delta_u \tag{2.41}$$

$$\mathbf{V} = \hat{\mathbf{V}}\Delta_v \tag{2.42}$$

Where Δ_u and Δ_v are diagonal matrices. From (2.41) and (2.41) it follows that:

$$\hat{\mathbf{U}}^\dagger\hat{\mathbf{U}} = \mathbf{I} \tag{2.43}$$

$$\hat{\mathbf{V}}^\dagger\hat{\mathbf{V}} = \mathbf{I} \tag{2.44}$$

With † denoting complex conjugate and transpose. Combining (2.34) with (2.41) and (2.42) gives:

$$\mathbf{K} = \mathbf{U}\mathbf{\Sigma}\mathbf{V} = \hat{\mathbf{U}}\Delta_u\mathbf{\Sigma}(\hat{\mathbf{V}}\Delta_v)^T$$

$$= \hat{\mathbf{U}}\Delta_u \mathbf{\Sigma} \Delta_v^T \hat{\mathbf{V}}^T = \hat{\mathbf{U}}\hat{\mathbf{\Sigma}}(\hat{\mathbf{V}}^*)^\dagger \tag{2.45}$$

Which represents a singularvalue decomposition (svd) of $\mathbf{K}$. Given the svd, the two following relations hold:

$$\mathbf{K}^\dagger \mathbf{K} = \hat{\mathbf{V}}^* \hat{\mathbf{\Sigma}}^2 (\hat{\mathbf{V}}^*)^\dagger \tag{2.46}$$

$$\mathbf{K}^* \mathbf{K}^T = \hat{\mathbf{U}}^* \hat{\mathbf{\Sigma}}^2 (\hat{\mathbf{U}}^*)^\dagger \tag{2.47}$$

In the following section it is demonstrated that these two relations can be interpreted as time-reversal operations.

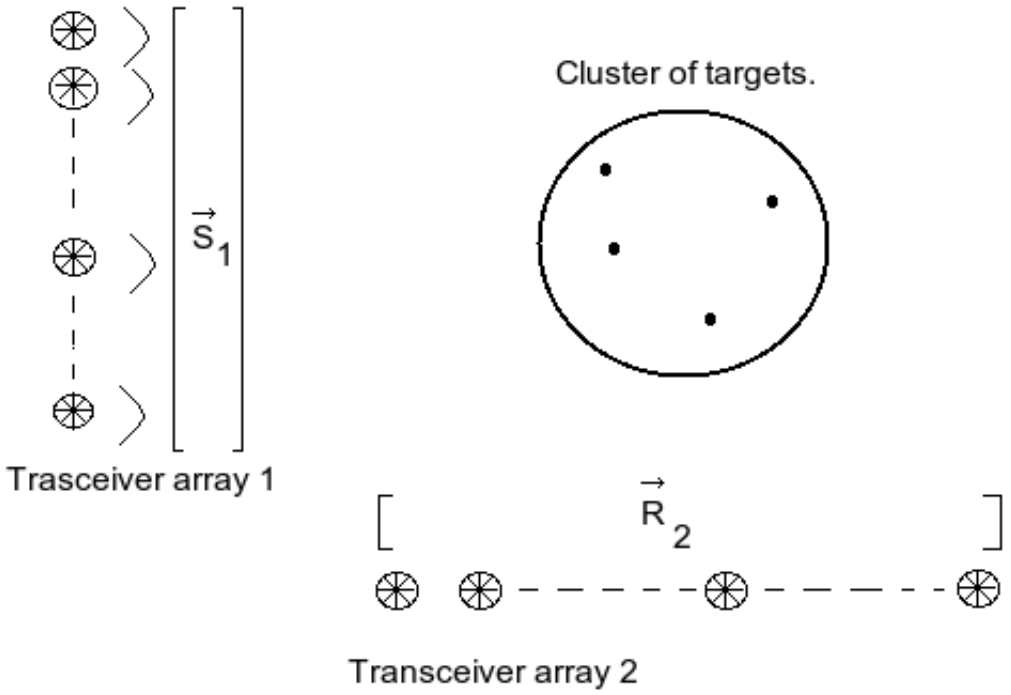

Figure 2.7:

2.2.5 Timereversal of the transfermatrix in case of separated arrays

The case of separate source and receiver arrays should be generalized to a two transceiver array case [10] and [11]. In the following the source array is represented by transceiver array 1 (T1) and the receiver array by transceiver array 2 (T2). Let $\vec{S}_1(w)$ be the transmitted signal vector from T1, See Figure 2.7. The data $\vec{R}_2(w)$ received at T2 is given by the relationship:

$$\vec{R}_2(w) = \mathbf{K}\vec{S}_1(w) \tag{2.48}$$

Time-reversal of these measurements and detection at T1 gives:

$$\vec{R}_1(w) = \mathbf{K}^T\vec{R}_2^*(w) = \mathbf{K}^T\mathbf{K}^*\vec{S}_1^*(w) \tag{2.49}$$

In (2.49) the transposed matrix $\mathbf{K}^T$ represents the transfermatrix from T2 to T1, this follows directly from the assumption of a reciprocal medium. Finally, complex conjugation of (2.49) gives the key result:

$$\vec{R}_1^*(w) = \mathbf{K}^\dagger\mathbf{K}\vec{S}_1(w) \tag{2.50}$$

This defines the **time-reversal matrix** with respect to transceiver array 1:

$$\mathbf{T}_1 = \mathbf{K}^\dagger \mathbf{K} \tag{2.51}$$

Let λ be an eigenvalue and $\vec{v}$ the associated eigenvector of $\mathbf{T}_1$, then the following relationship can be established:

$$\begin{aligned} \|\mathbf{K}\|^2 = (\mathbf{K}\vec{v})^\dagger(\mathbf{K}\vec{v}) = \vec{v}^\dagger \mathbf{K}^\dagger \mathbf{K}\vec{v} \\ = \lambda \|\vec{v}\|^2 \end{aligned} \tag{2.52}$$

Thus, eigenvalues of T_1 are positive. From (2.50) it follows that if the input signal is an eigenvector of the time-reversal matrix, then it is an invariant of the time-reversal process, that is, the received signal after one timereversal is proportional to the phase conjugate of the first input. Consider the alternative experiment, starting with an emitted signal vector $\vec{S}_2(w)$ from T2 and carry out one time-reversal procedure. This gives the counterpart to (2.50):

$$\vec{R}_2^*(w) = \mathbf{K}^* \mathbf{K}^T \vec{S}_2(w) \tag{2.53}$$

The time-reversal matrix with respect to T2 is defined:

$$\mathbf{T}_2 = \mathbf{K}^* \mathbf{K}^T \tag{2.54}$$

Again it follows that if the inputsignal is an eigenvector of $\mathbf{T}_2$, then it is an invariant of the second time-reversal process.

2.2.6 Eigenvector solutions of the timereversalmatrix

The row-normalized versions of **U** and **V** are given:

$$\hat{\mathbf{U}} = \left[\frac{\vec{g}_1^r(w)}{\|\vec{g}_1^r(w)\|} \ ... \ \frac{\vec{g}_M^r(w)}{\|\vec{g}_M^r(w)\|} \right] \tag{2.55}$$

$$\hat{\mathbf{V}} = \left[\frac{\vec{g}_1^s(w)}{\|\vec{g}_1^s(w)\|} \ ... \ \frac{\vec{g}_M^s(w)}{\|\vec{g}_M^s(w)\|} \right] \tag{2.56}$$

Direct use of the relation in (2.39) identifies the eigenvectors of $\mathbf{T}_1$ to be:

$$\frac{(\vec{g}_i^s(w))^*}{\|\vec{g}_i^s(w)\|} \tag{2.57}$$

$$i = 1, 2, ..., M.$$

Each eigevector can be interpreted as the time-reversed form of the signal that would be received on array 1 (source array) if target number i acted as the only secondary source, see Figure 2.8. Analogously we may apply (2.40) to find the eigen-vectors of $\mathbf{T}_2$:

$$\frac{(\vec{g}_i^r(w))^*}{\|\vec{g}_i^r(w)\|} \tag{2.58}$$

Again assuming a secondary source at target position i, the corresponding eigenvector represents the time-reversed version of detected signals in array 2 (receiver array). Eigenvalues are given:

$$\lambda_i = |\tau_i| \|\vec{g}_i^s(w)\| \|\vec{g}_i^r(w)\| \tag{2.59}$$

$$i = 1, 2, ..., M.$$

This demonstrates that eigenvalues are proportional to the scattering strengths. We should also notice that by the above description, that if array 1 represents an array of sources and array 2 the receiver array, it follows that the singular system:

$$\begin{aligned} \mathbf{K}\vec{e}_i &= \sigma_i \vec{v}_i \\ \mathbf{K}^\dagger \vec{v}_i &= \sigma_i \vec{e}_i \end{aligned} \tag{2.60}$$

is solved by the singularvectors:

$$\vec{e}_i = \frac{(\vec{g}_i^s(w))^*}{\|\vec{g}_i^s(w)\|} \tag{2.61}$$

$$\vec{v}_i = \frac{(\vec{g}_i^r(w))}{\|\vec{g}_i^r(w)\|} \tag{2.62}$$

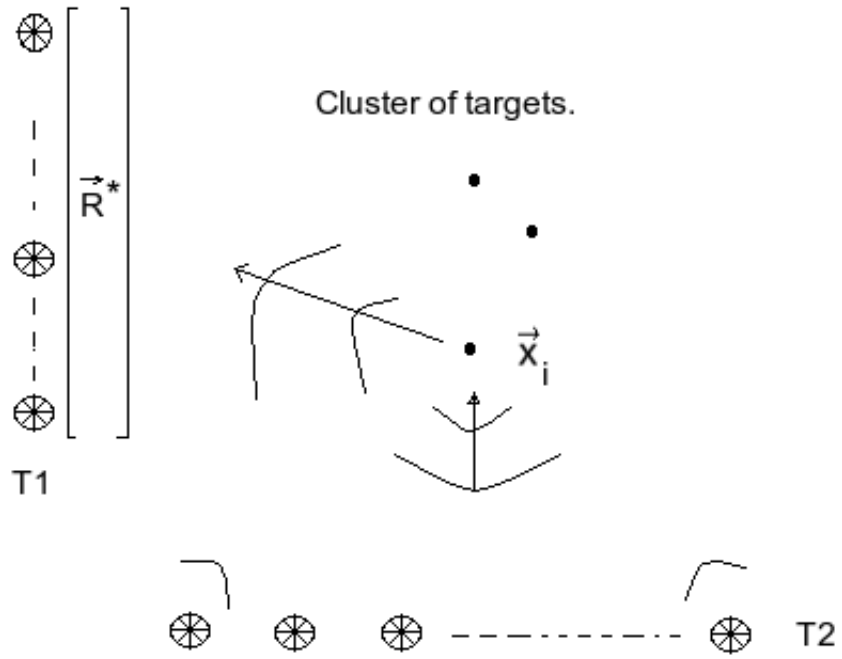

Figure 2.8: After measureing at T2 a time-reversal is performed. Signals are emitted from T2 and detected in T1. The detection in T1 is time-reversed.

We then have a projetion of singularvectors from the source array to the receiver array through the transfermatrix. And the opposite projection of singularvectors goes through the adjoint transfermatrix. In case of coincide source-receiver-arrays, the transfermatrix become quadratic and symmetric and the two time-reversal matrices become equal:

$$\mathbf{T}_1 = \mathbf{T}_2 = \mathbf{K}^*\mathbf{K} \tag{2.63}$$

Also the two sets of eigenvectors will coincide.

2.3 Imaging algorithms

In this section we look at two algorithms that makes us able to visualize buried scattering targets. The methods are backpropagation and MUSIC. These will be treated in light of the theory from **??**. Remember that theory assumes perfectly resolved point-targets under Born approximation of the scattered waves. In this case we can achieve highly accurate information about the positions of the scatterers and their strengths. Under these circumstances MUSIC is denoted a super-resolution imaging technique. The most important properties of the well resolved case herein are the orthogonality of the Green's function vectors from source and receiver arrays, and also that they are related to the eigenvectors of the time-reversal matrices. This is taken advantage of in both algorithms.

2.3.1 Backpropagation and MUSIC

We continue our work in the frequency-domain, more precisely, we present the theory within the $(\vec{x}, w)$-domain. Also, we will be considering the case of separated source and receiver arrays. From here on and out we assume that the transfermatrix **K** is already obtained. The Greens function vector from all receiver positions to an arbitrary coordinate $\vec{x}$ is:

$$\vec{g}^r(\vec{x}, w) = [g(\vec{x}, \vec{x}_1, w) \ ... \ g(\vec{x}, \vec{x}_L, w)]^T \tag{2.64}$$

The source-array related Green's function vector is given by:

$$\vec{g}^s(\vec{x}, w) = [g(\vec{x}, \vec{x}_1, w) \ ... \ g(\vec{x}, \vec{x}_N, w)]^T \tag{2.65}$$

As before, let $T_1 \in \mathbb{C}^{N,N}$ be the time-reversal matrix related to the source-array and $T_2 \in \mathbb{C}^{L,L}$ to the receiver-array. Define eigenvectors of **T1** to be $\vec{e}_j$ and denote eigenvectors of **T2** as $\vec{v}_l$. We have that the number of eigenvectors corresponding to non-zero eigenvalues are given by the conjugated and normalized Greens function vectors from (2.64) and (2.65) evaluated at $\vec{x}_i$. Which gives:

$$\vec{e}_i = \frac{(\vec{g}^s(\vec{x}_i, w))^*}{\|\vec{g}^s(\vec{x}_i, w)\|} = \frac{(\vec{g}_i^s(w))^*}{\|\vec{g}_i^s(w)\|} \tag{2.66}$$

$$\vec{v}_i = \frac{(\vec{g}^r(\vec{x}_i, w))^*}{\|\vec{g}^r(\vec{x}_i, w)\|} = \frac{(\vec{g}_i^r(w))^*}{\|\vec{g}_i^r(w)\|} \tag{2.67}$$

$$i = 1, 2, ..., M$$

Where M is the number of scatterers. In order to use both arrays when gathering scattered data we need the number of sources and receivers to be at least as many as number of targets. By definition all eigenvectors of a matrix are orthogonal and in particular we have that:

$$\vec{e}_{k1}^T \vec{g}_i^s(w) = 0 \tag{2.68}$$

$$\vec{v}_{k2}^T \vec{g}_i^r(w) = 0 \tag{2.69}$$

$$k_1 = M + 1, M + 2, ..., N - M$$

$$k_2 = M + 1, M + 2, ..., L - M$$

$$i = 1, 2, ..., M.$$

These two results focus on the fact that scatterer related and non-scatter related eigenvectors are still orthogonal.

2.3.2 Backpropagation

Rather than using a conventional backpropagation algorithm we take into account that we are working with time-reversed data. Backpropagating receiver related eigenvectors is done as followed:

$$P_{br}^{i}(\vec{x}, w) = \vec{g}^{r}(\vec{x}, w)^{T}\vec{v}_i \tag{2.70}$$

Notice that only eigenvectors corresponding to targets are included. Also define the backpropagation from the source array accordingly:

$$P_{bs}^{i}(\vec{x}, w) = \vec{g}^{s}(\vec{x}, w)^{T}\vec{e}_i \tag{2.71}$$

With these formulas we project vectors back into the space from which signals propagated. It may seem a little peculiar to perform a backpropagation from the source array. But we must remember the dualism of time-reversal theory in this case of separated source and receiver arrays. In (2.70) and (2.71) the important properties are [1,11]:

$$P_{br}^{i}(\vec{x}, w) = \begin{cases} \|\vec{g}_i^{r}(w)\| & \vec{x} = \vec{x}_i \\ 0 & \vec{x} \neq \vec{x}_i \end{cases} \tag{2.72}$$

and

$$P_{bs}^{i}(\vec{x}, w) = \begin{cases} \|\vec{g}_i^{s}(w)\| & \vec{x} = \vec{x}_i \\ 0 & \vec{x} \neq \vec{x}_i \end{cases} \tag{2.73}$$

Thus, each time we use a target coordinate as input in (2.72) and (2.73) the results differs from zero. How much the results differs from zero depends on the Green's function and locations of the scatterers relative to arrays.

2.3.3 MUSIC

The MUSIC algorithm exploits the results from (2.68) and (2.69) and goes as followed [1,11]:

$$P_M(\vec{x}, w) = \frac{1}{\sum_{k=M+1}^{min\{L,N\}} |\vec{e}_k^{T}\vec{g}^{s}(\vec{x}, w)| + |\vec{v}_k^{T}\vec{g}^{r}(\vec{x}, w)|} \tag{2.74}$$

Once we point $\vec{x}$ to a target, the Greens function vectors becomes eigenvectors. Hence, MUSIC can be interpreted as backpropagation of the zero solutions followed by an inversion of the obtained image. By zero solutions we mean eigenvectors that are related to zero eigenvalues. So the idea is that these vectors conclusively does not relate to the targets. But when they are pointing at these locations we exploit the feature of eigenvectors being orthogonal.

Since eigenvectors at target positions are given by the Green's function vectors we get the zero values and thus a strong peak. In other locations the Green's function vectors are not eigenvector solutions of the system and the backpropagation should be different from zero. As result we get small image values there.

Chapter 3

Simulating propagating waves and experimentally building the transfermatrix

Intro

Our concern now are principles on how we simulate the experimental data-acquisition. When using computers and discrete models to create data there usually are some limitations to consider. We will look at the main ingredients of the program and make awareness about some discrepancies. When performing numerical experiments we need to know that the simulations are reliable. It is desired that they imitate the physical world up to a point of satisfaction. Results from simulating wave-detection strongly depend on propagation and velocity accuracy. We need to know that our discretely propagating waves follows the given velocity profile. The work is done by modeling waves propagating in two spatial dimensions. Generating 3D data would demand a lot of computation. Because of similarities in all aspects of the theory for 2D and 3D waves it can be expected that results would be similar in a 3D simulation. An advantage with 2D is that it easy to visualize the outcome. This chapter begins with a brief description on how array detections are simulated when a source is excited. Then the mathematical model with the physics of the wave simulation is presented and this is verified through numerical experiments. The purpose of generating data is to obtain the transfermatrix. At the end of this chapter a description of how to experimentally build this matrix is given.

3.1 Simulating data recordings

We will perform computer simulations to create data and follow the common pattern:

Simulate $\rightarrow$ **Data** $\rightarrow$ **Usage**

In these experiments we will be using the two imaging algorithms MUSIC and backpropagation within the frames of time-reversal theory. All needed data is thus related to building the transfermatrix **K**. The core of the program is a simulation of propagating waves in a scattering environment. Measurements gives the transfermatrix, the basic program structure is shown in Figure 3.1. This structure assumes a case of monochromatic signals. If sources contain multiple frequencies the process can be repeated. To simulate wavepropagations we use a forward finite difference scheme, a mathematical description is given in the section 3.1. Consider Figure 3.1, the box numbered 1 shows three actions performed within a finite $2D$ grid. Time steps are given by $j = 1, ..., N$ and a source is defined at a given coordinate with a temporal behavior $s(t_j)$ in this position. The ingrid values away from this point is thus dependent of the source. These values (wavepropagation) are computed according to the $FD2D$ scheme and continues until $j = N$. While the loop runs, values at given locations are kept in every temporal step, as result we get a receiver vector $R(\vec{t}_j)$. This describes one source-detection simulation. If an array of sources are to be employed the complete set of data is obtained by repeating this process until all sources are accounted for. Single source excitement and array detection gives paired data, thus we have a distinct set of recorded data for every source (different locations and or different pulse).

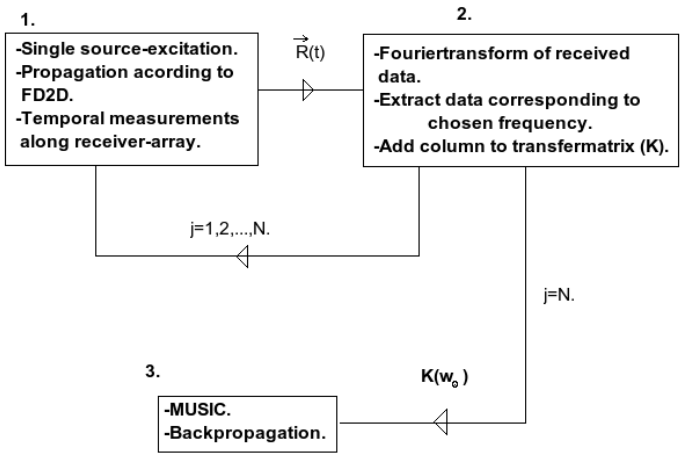

Figure 3.1: The program-core is wavesimulations according to a forward finite $2D$ scheme ($FD2D$), this may be repeated several times depending on the number of sources.

3.1.1 2D explicit finite difference scheme

As mentioned we record data by simulating 2D propagating waves. To perform the simulation we use an explicit scheme which means that we obtain the spatial solution in one temporal step forward based on solutions in previous steps. To derive this scheme we again look at the wave equation. From (2.1) in Chapter 2 we have the foundation for most work done herein, i.e the wave equation which theory is built upon. We now consider the two dimensional version of this equation:

$$(\frac{\partial^2}{\partial x^2}) + \frac{\partial^2}{\partial y^2})\psi(x,y,t) - \frac{1}{v(x,y)^2}\frac{\partial^2}{\partial t^2}\psi(x,y,t) = 0 \tag{3.1}$$

$$\psi(x_s, y_s, t) = s(t)$$

The source-coordinate is excluded in the solution due to notational convenience. To obtain the discrete solution we seek we start with two different Taylor expansions of the continuous wavesolution of (3.1):

$$\begin{aligned}\psi(x+\Delta x, y_0, t_0) =& \psi(x, y_0, t_0) + \Delta x\frac{d}{dx}\psi(x, y_0, t_0) \\ &+ \frac{\Delta x^2}{2}\frac{d^2}{dx^2}\psi(x, y_0, t_0) + \frac{\Delta x^3}{6}\frac{d^3}{dx^3}\psi(x, y_0, t_0) \\ &+ \frac{\Delta x^4}{24}\frac{d^4}{dx^4}\psi(x+c_1, y_0, t_0)\end{aligned}$$

and

$$\begin{aligned}\psi(x-\Delta x, y_0, t_0) =& \psi(x, y_0, t_0) - \Delta x\frac{d}{dx}\psi(x, y_0, t_0) \\ &+ \frac{\Delta x^2}{2}\frac{d^2}{dx^2}\psi(x, y_0, t_0) - \frac{\Delta x^3}{6}\frac{d^3}{dx^3}\psi(x, y_0, t_0) \\ &+ \frac{\Delta x^4}{24}\frac{d^4}{dx^4}\psi(x-c_1, y_0, t_0)\end{aligned}$$

Where $0 \leq c_1, c_2 \leq \Delta x$. Notice that we only look at variations in the x-direction, while viewing the solution from a locked position y_0 at a timeinstant t_0. If we solve both expressions with respect to the terms involving the third derivatives we obtain:

$$\begin{aligned}&-\psi(x-\Delta x, y_0, t_0) + \psi(x, y_0, t_0) + \frac{\Delta x^2}{2}\frac{d^2}{dx^2}\psi(x, y_0, t_0) + \frac{\Delta x^4}{24}\frac{d^4}{dx^4}\psi(x-c_2, y_0, t_0) \\ &= \psi(x+\Delta x, y_0, t_0) - \psi(x, y_0, t_0) - \frac{\Delta x^2}{2}\frac{d^2}{dx^2}\psi(x, y_0, t_0) - \frac{\Delta x^4}{24}\frac{d^4}{dx^4}\psi(x+c_1, y_0, t_0)\end{aligned}$$

This can be solved with respect to the second derivative term which gives us:

$$\begin{aligned}&\frac{d^2}{dx^2}\psi(x, y_0, t_0) = \\ &\frac{1}{\Delta x^2}\left[\psi(x+\Delta x, y_0, t_0) - 2\psi(x, y_0, t_0) + \psi(x-\Delta x, y_0, t_0)\right] + \\ &\frac{\Delta x^2}{24}\left[\frac{d^4}{dx^4}\psi(x+c_1, y_0, t_0) + \frac{d^4}{dx^4}\psi(x-c_2, y_0, t_0)\right]\end{aligned} \tag{3.2}$$

The last part of (3.2) is excluded and will be referred to as the error term. With this we define the operator which is of the Crank-Nicolson type [12]:

$$L_{y_0,t_0}(\psi)(x_i) = \frac{\psi(x_{i+1}, y_0, t_0) - 2\psi(x_i, y_0, t_0) + \psi(x_{i-1}, y_0, t_0)}{\Delta x^2} \tag{3.3}$$

$$\Delta x = x_i - x_{i-1}$$

Which satisfies:

$$|L_{y_0,t_0}(\psi)(x_i, y_0, t_0) - \frac{d^2}{dx^2}\psi(x_i, y_0, t_0)| \leq \frac{M_f \Delta x^2}{12} \tag{3.4}$$

Where M_ψ is the constant related to the behavior of $g(x) = \psi(x, y_0, t_0)$ by:

$$M_\psi = max_x\{|\frac{d^4 g(x)}{dx^4}|\} \tag{3.5}$$

M_ψ is nothing else than taking the absolutevalue of the upper limit of the **error term**. We thus have that:

$$\frac{d^2}{dx^2}\psi(x_i, y_0, t_0) = L_{y_0,t_0}(\psi)(x_i) + O(\Delta x^2) \tag{3.6}$$

The $O(\Delta x^2)$ indicates that the error is in the order of Δx^2 limited by (3.5). Thus the difference between the approximation and the exact value will never go beyond this size. Deriving this operator for the direction y and in time t are completely analogous, it only differs in notation of variables. We use the L-operator to obtain a discrete version of (3.1) and add the errors to ensure equality:

$$\begin{aligned}&L_{y_j,t_i}(\psi)(x_k) + L_{x_k,t_i}(\psi)(y_j) - \frac{1}{v_{k,j}^2}L_{x_k,y_j}(\psi)(t_i) \\ &+ O(\Delta x^2) + O(\Delta y^2) + O(\Delta t^2) = 0 \\ &\psi_{s,s}^i = s(t_i)\end{aligned} \tag{3.7}$$

The spatial steps may be of different sizes, but we choose $\Delta x = \Delta y$. Introduce the following notation for the discrete solution:

$$\psi(x_k, y_j, t_i) = \psi_{k,j}^i$$

Rearrange (3.7) to get:

$$\begin{aligned}\psi_{k,j}^{i+1} =& \frac{v_{k,j}^2 \Delta t^2}{\Delta x^2}\left[\psi_{k+1,j}^i - 4\psi_{k,j}^i + \psi_{k,j+1}^i + \psi_{k-1,j}^i + \psi_{k,j-1}^i\right] \\ &+ 2\psi_{k,j}^i - \psi_{k,j}^{i-1}\end{aligned} \tag{3.8}$$

$$\psi_{s,s}^i = s(t_i)$$

With an maximum error:

$$E = O(\Delta x^4)\Delta t^2 + O(\Delta t^4)\Delta x^2 \tag{3.9}$$

We may use a discrete delta on the source term and add it to the solution:

$$\begin{aligned}\psi_{k,j}^{i+1} = \frac{v_{k,j}^2 \Delta t^2}{\Delta x^2}&\left[\psi_{k+1,j}^i - 4\psi_{k,j}^i + \psi_{k,j+1}^i + \psi_{k-1,j}^i + \psi_{k,j-1}^i + \psi_{k,j}^i\right] \\ &+2\psi_{k,j}^i - \psi_{k,j}^{i-1} + \delta_{s,s}s^i\end{aligned} \tag{3.10}$$

Considering (3.9) we would like to choose the spatial and temporal steps small, but this can not be done arbitrary. To ensure stability in the discrete solution we need to furfill the Courant condition [13]. To minimize the effect of numerical dispersion due to discretization of the wave-equation, the spatial sampleinterval is chosen six times denser than the minimum requirement of half of a wavelength, i.e (f_0 denotes the center-frequency):

$$\sqrt{2}\Delta t \leq \frac{\Delta x}{max_{x,y}\{v(x,y)\}} \tag{3.11}$$

$$\Delta x = \frac{min_{x,y}\{v(x,y)\}}{12 f_0} \tag{3.12}$$

For every temporal step in the discrete solution we need two previous temporal computations and their related spatial values. An illustration is given in Figure 3.2. From this we see that the ingrid values will depend on the solution at the gridboundary. When we perform tests with simulations, we want to work as directly as possible with the data we seek to use. Therefore we apply absorbing boundary conditions together with the scheme [14]. The idea is that the solution should behave as if we look at waves thorugh a window, see Figure 3.3 (i.e. ideally no waves are reflected back from artificial boundaries). This ables us to measure the waves without having to apply filteringtechniques to get rid of unwanted data.

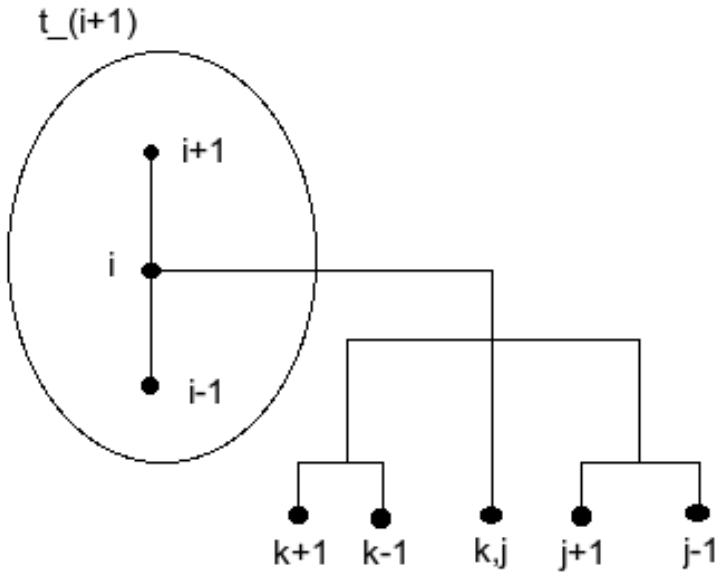

Figure 3.2: In one time-step forward we need 7 previous evaluations.

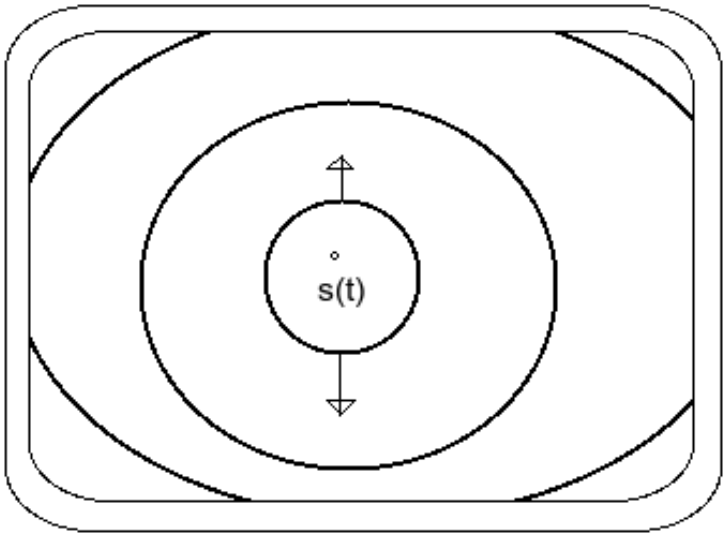

Figure 3.3: Radially propagating wave, a frame illustrates the gridboundary which ideally does not reflect waves.

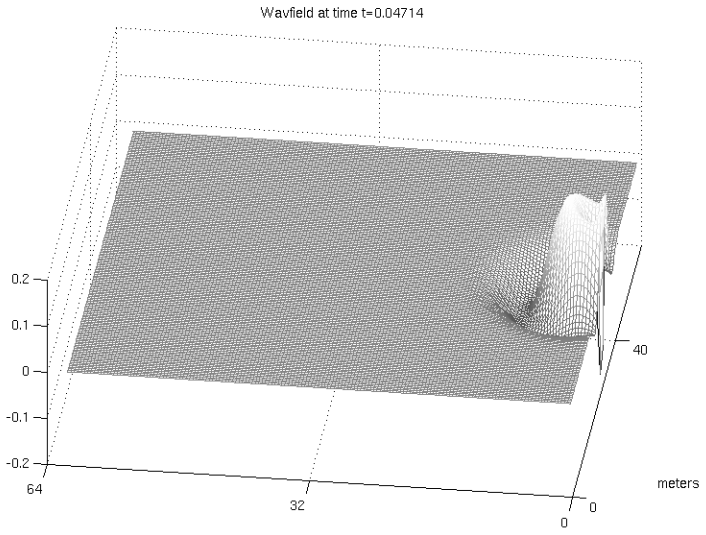

Figure 3.4: Snapshot of excitated source at $t = 0.04714s$. Wave-propagation velocity $c = 500\frac{m}{s}$.

3.2 Simulating propagating waves

Before we can use the program to create images we must ensure that wave-propgations are reliable. We need the discrete waves to behave as expected, and also to verify velocity versus arrivaltimes. Using vanishing boundary conditions results in some reflections, this will be considered in more detail. First, some images of the waves under consideration are given. Consider Figure 3.4 and 3.5. A source is excited in a homogeneous background. The wave propagates forward and in the center of the grid a pointscatterer is located. In Figure 3.6 the scattered field is displayed. We can see that the scatterer clearly acts as a secondary source. It creates a symmetrical radially expanding wave-field. Figure 3.7 shows the scattered wave at a later time.

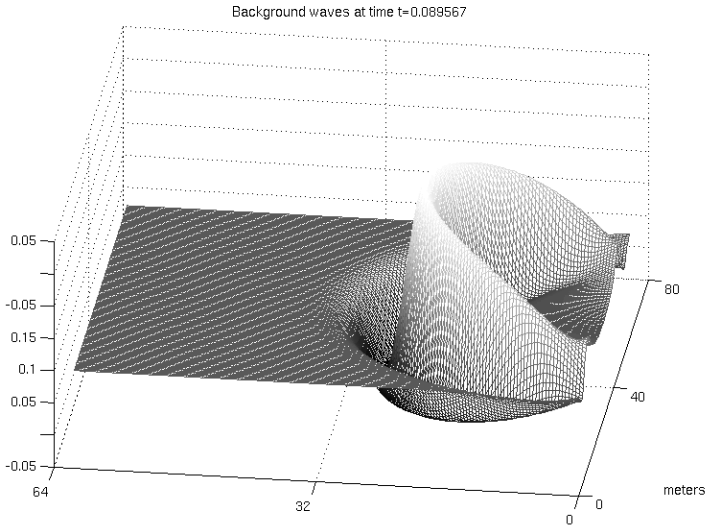

Figure 3.5: Snapshot just before the wave reaches the scatterer, $t = 0.089567s$.

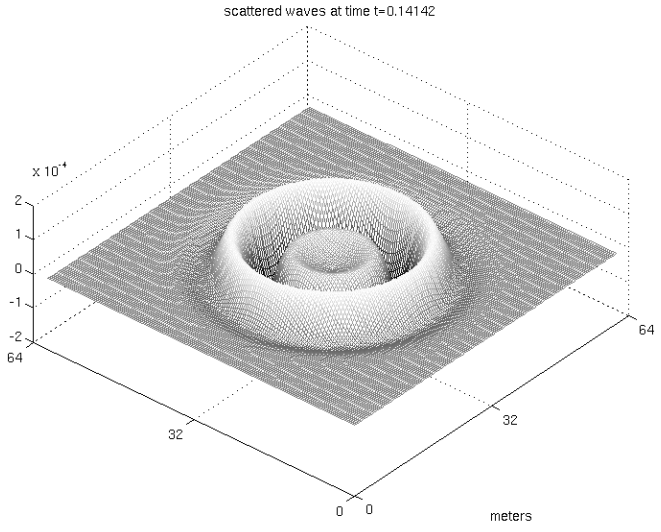

Figure 3.6: Scattered wave at $t = 0.14142s$.

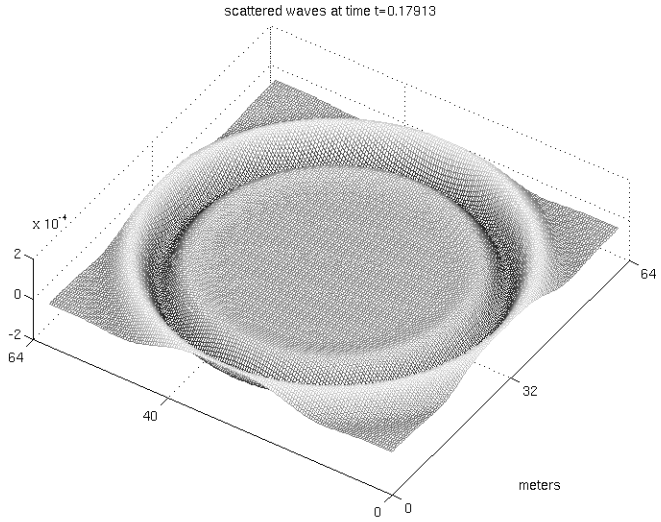

Figure 3.7: Scattered wave at $t = 0.17913s$, the wave has reached the grid-boundary.

3.2.1 Arrivaltimes

A wave is generated from different shotpoint locations and propagates towards a scatterer. In all situations a $30Hz$ Ricker-pulse is used as the inputsignal see Figure 3.8. The important feature here is when the source is excited and when it is detected. To check the consistency a scattered wave is measured at three locations. We will see the time of arrival at these locations coincide well with analytically calculated ones. Figures are used to gain a more detailed explanation. Two different scenarios of source-receiver configurations follows, see Figures 3.9 and 3.11. Consider Figure 3.9, the scatterer is now centered 30 meters away from the source. Receivers are placed at distances 30 and 42.4264 meters away from scatterer (center-side and upper-side). Total length of propagation is thus 60 and 72.4264 meters. The center wavelength is about $17m$. Waves propagating at constant velocity $500\frac{m}{s}$ should arrive at 120 and 144.9 ms. As illustrated in Figure 3.10 the program simulates this accurately. The configuration given in Figure 3.11 shows the following setup: distance between source and scatterer is 42.4264 meters, while the scatterer-receiver distance is 30 meters. Note that this is the same total length of propagation as in the previous case for the receivers located at each side. Waves are therefore expected to use the same amount of time to reach these receivers. Figure 3.12 shows us that this is the exactly the case. We have an accurate simulation but we should not forget that there are error-terms involved. We are not ensured such highly consistent propagation-times in any situation.

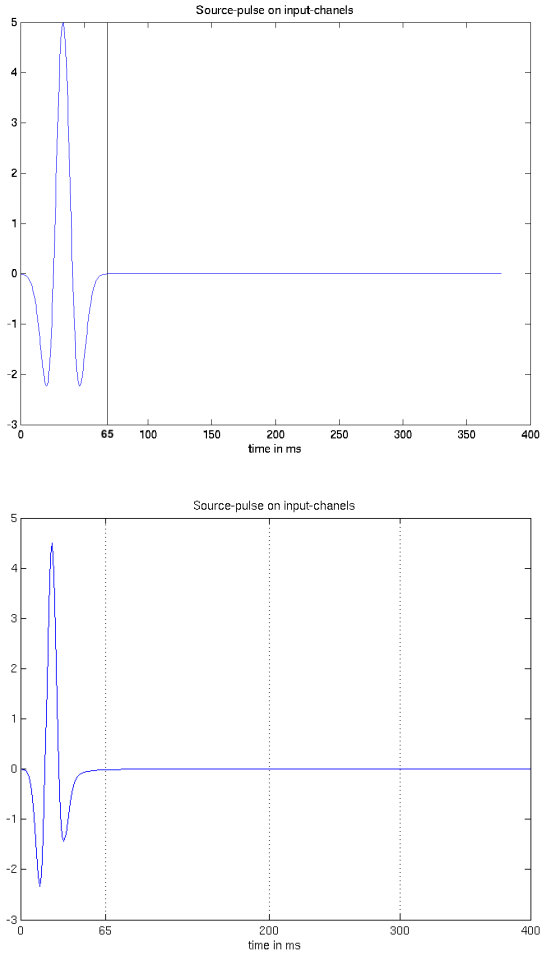

Figure 3.8: The analytical behvior of a Ricker-pulse with central frequency $30Hz$ is given in the upper figure. When this defines the pulse in the $2D$ modeling program the generated source is given as in the lower figure.

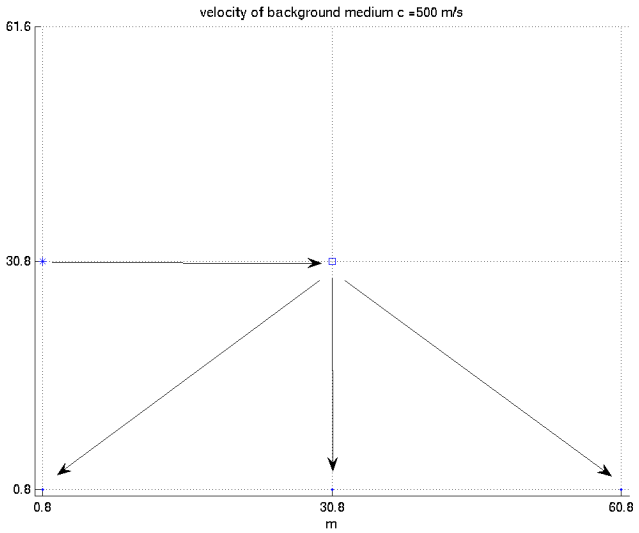

Figure 3.9: Source-coordinate $(0.8, 30.8)m$, scatterer-coordinate $(30.8, 30.8)m$ and three receivers are equally spaced at $(0.8, 0.8)m$, $(30.8, 0.8)m$ and $(60.8, 0.8)m$.

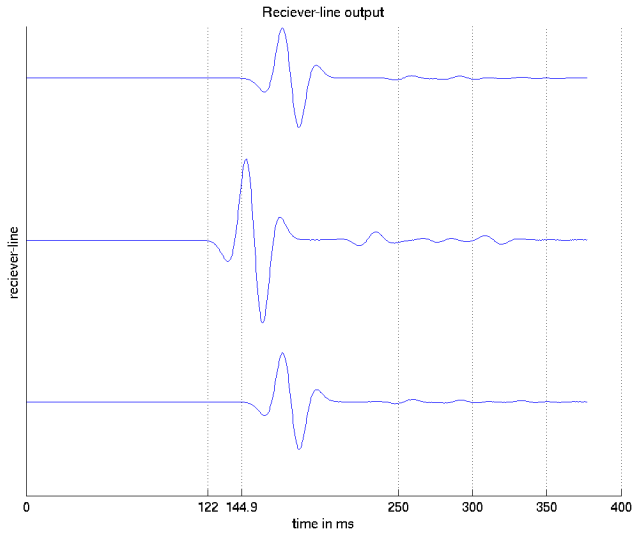

Figure 3.10: Receiver outputs in locations given in Figure 3.9. The times of arrival are $120ms$ and $144.9ms$.

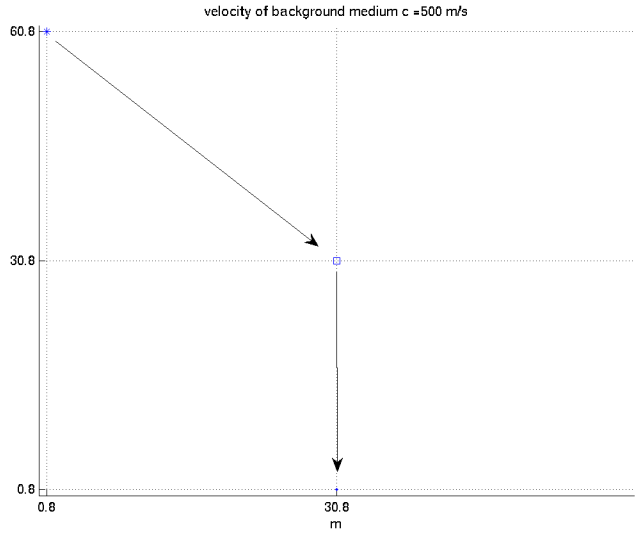

Figure 3.11: The source coordinate is $(0.8, 60.8)m$, the scatterer is located at $(30.8, 30.8)m$ and a single receiver is located at $(30.8, 0.8)m$.

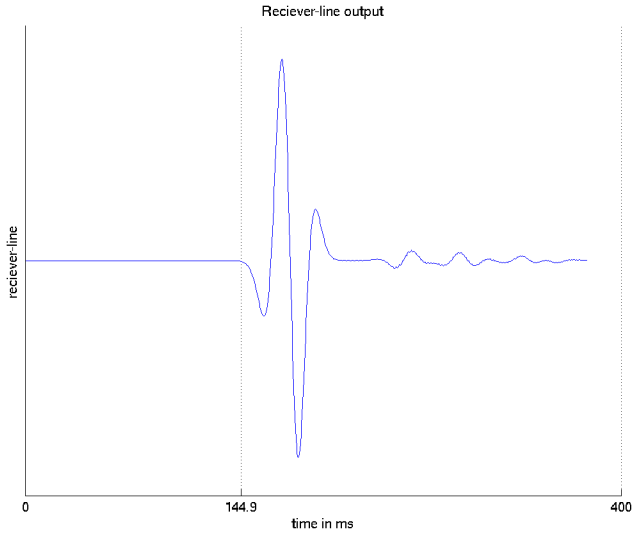

Figure 3.12: The time of arrival is $144.9ms$.

3.2.2 Effects from non-absorbed waves

Generating data with absorbing boundary conditions do not make waves vanish completely at the grid endpoints. There are some reflections of the boundary and the solution will be influenced by this without proper considerations. In Figure 3.13 a scattered wave propagates towards the gridboundary and Figure 3.14 shows the reflections that are created at a

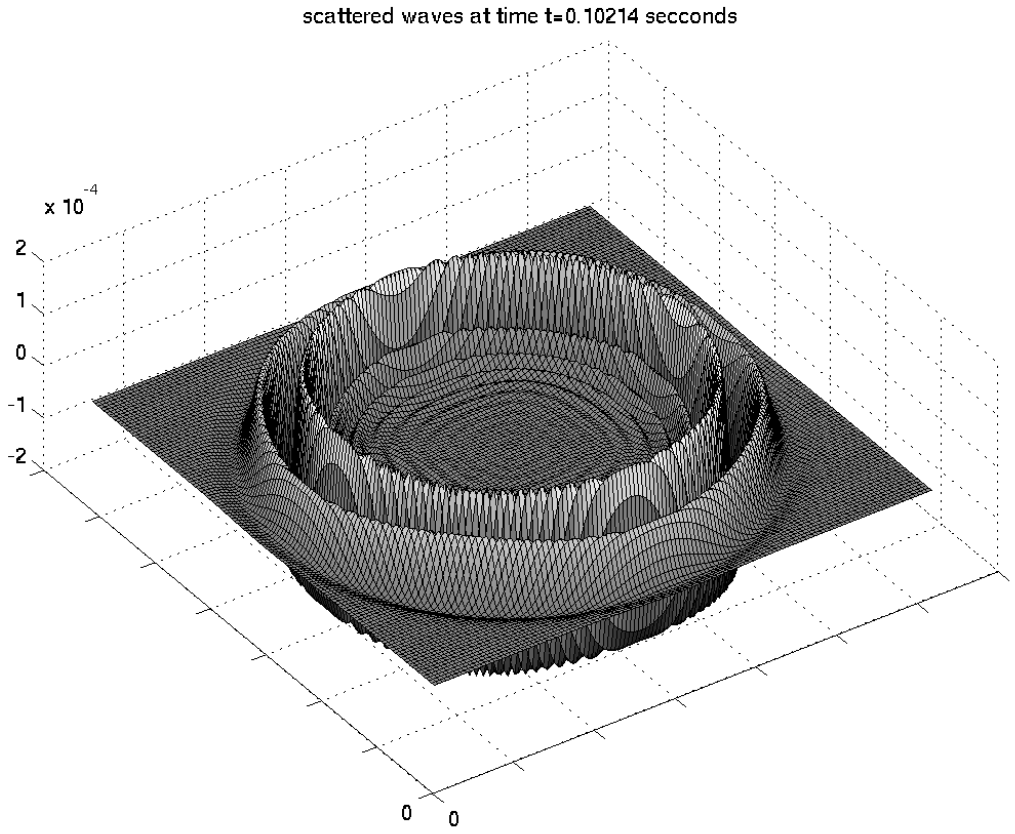

Figure 3.13: Scattered wave from a centered perturbation. The solution is shown at a timeinstant before the wave reaches the boundary.

later time instant. Imagine that some sensors measuring the waves are placed inside the grid. Boundaryreflections will eventually interact with the scatterer once more, and if measurements is carried out long enough the data will be distorted. Both from these inward propagating waves and the same waves scattered from the perturbation. So if the window of recording is to large, we may get data different from what is expected. Figure 3.15 displays a system with a corresponding long time measurement. The latter clearly

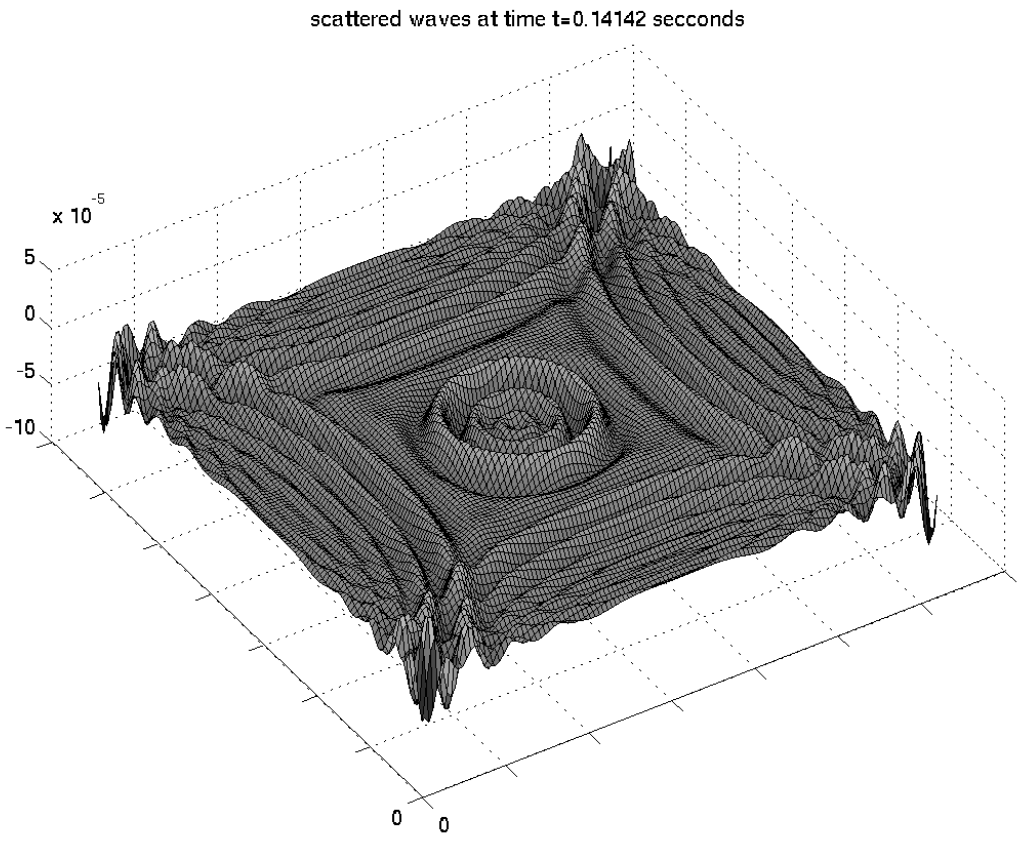

Figure 3.14: The solution is shown at a timeinstant after the wave has reached the boundary. Boundary reflections are clearly displayed.

illustrates the effects of the boundaries where values ideally should be zero at the receiving end. In this situation we could get rid of the unwanted data by recording within a timeframe just so the scattered wave has passed the receiver. However this is not that easy when dealing with multiple receivers and scatterers. Consider the system in Figure 3.16, if we stopped the measurements to early we would not get data from the far most distanced scatter. If we wait for signals to reach left and rightmost receivers the middle receiver records boundary reflections. One way, and maybe the most effective one would be to extend the grid and use only a small centered window with the elements involved. This could ensure that reflected waves did not reach receivers within this window of time. Performing calculations with large grids calls for a lot of computation. When we now start working with measured data, we extend the grid, but not so much that it takes unnecessary amounts of time to obtain accurate results. Instead, the geometrical layout of

perturbations, sources and receivers together with the period of recording are set to keep the measured reflections to a minimum.

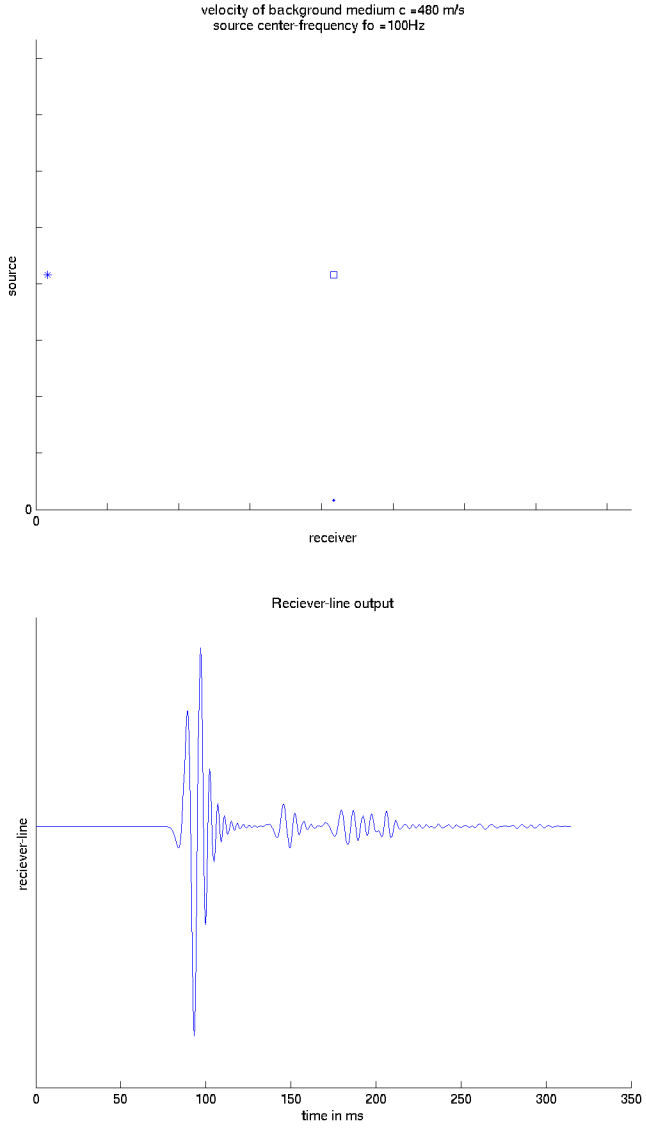

Figure 3.15: The geometric layout of a system with a source, a scatterer and a receiver is shown. The receiver is placed close to the boundary of the grid. Below a long time measurements with boundary reflections is given.

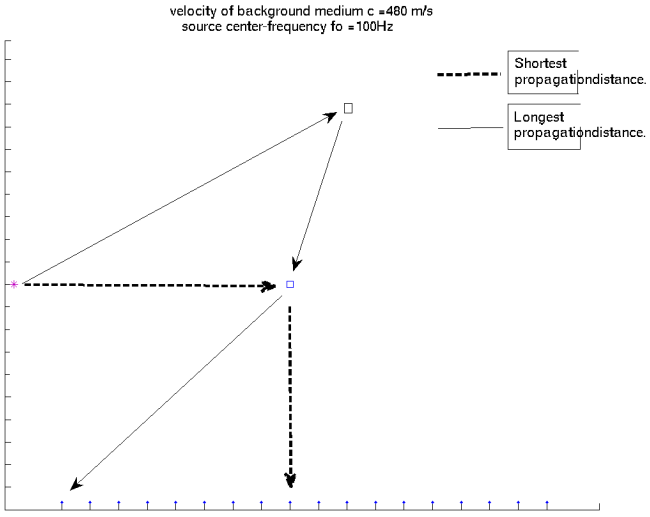

Figure 3.16: Scenario that could lead to unexpected data-measurements.

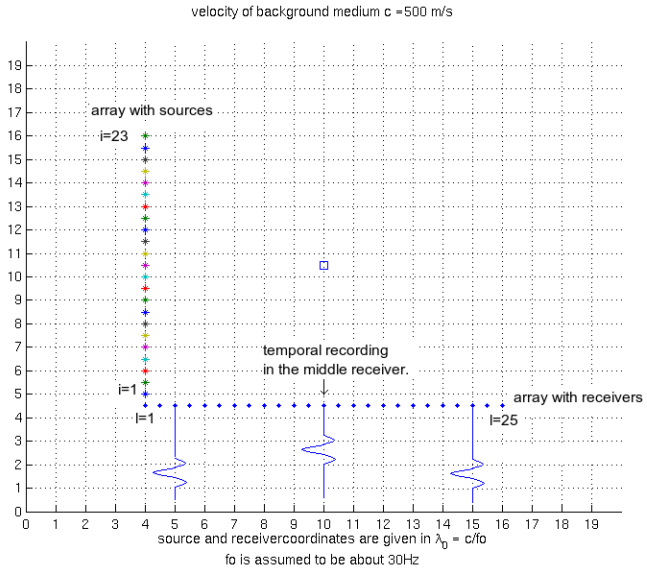

Figure 3.17: One scatter centered relative to sources and receivers.

3.3 Computing the transfermatrix

The Transfermatrix soon becomes the most important ingredient. So before we go ahead and make images, we take a look at how it is computed in a modeled scattering environment.

3.3.1 Transfermatrix

The characteristic pulse and its amplitude spectrum from all shotpoint locations is given in Figure 3.18, it is a delayed Ricker-pulse. One column in the transfermatrix is the vector containing measurements of the scattered field at each receiver when exciting one source $s_i(t)$. This is done from the bottom and up, as an example we may consider Figure 3.17. The source with coordinates $(4\lambda_o, 5\lambda_o)$ is first fired. Waves reach the perturbation and is scattered, we measure (the scattered field only) at all receiver locations $r_l(t)$. We now have simulated temporal data on each receiver element. An example of generated outputs is shown in Figure 3.19. The first element corresponds to the leftmost receiver, the second element to the right adjacent receiver and so on. A Fourier transform is performed on the source and all recorded signals. Denote the source spectrum $S_i(w)$ and the recorded spectrum's $R_l^i(w)$. The common index i indicate the relation between a source input and receiver outputs. To obtain one column of the transfermatrix we choose an angular frequency w_0 and arrange the corresponding data from each receiver in a vector $\vec{R}^i(w_0)$. We need

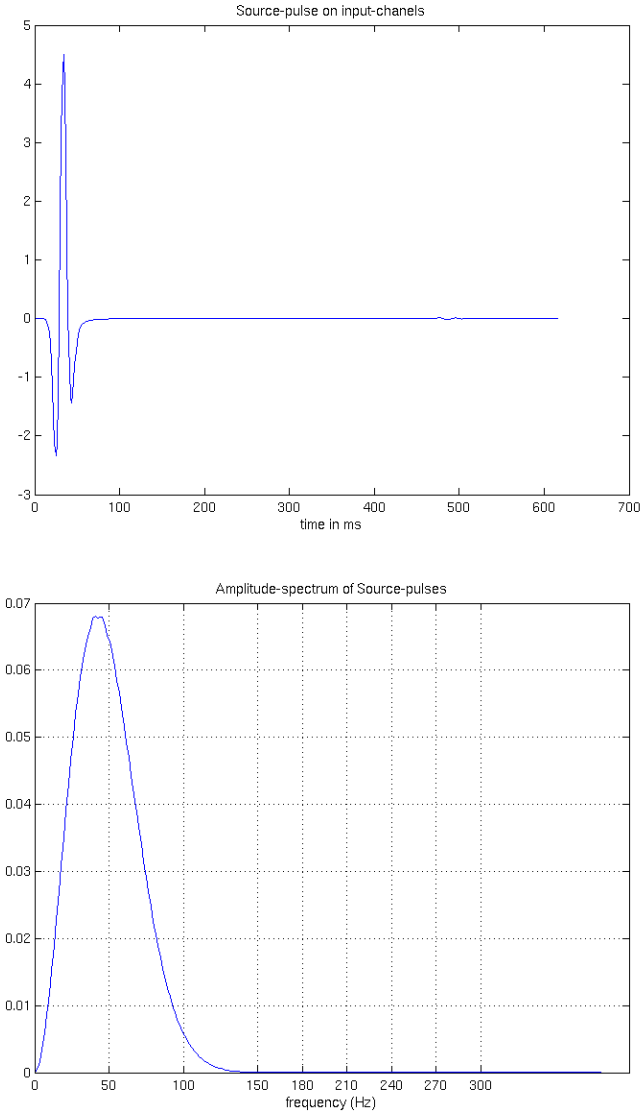

Figure 3.18: Ricker pulse and its amplitude spectrum.

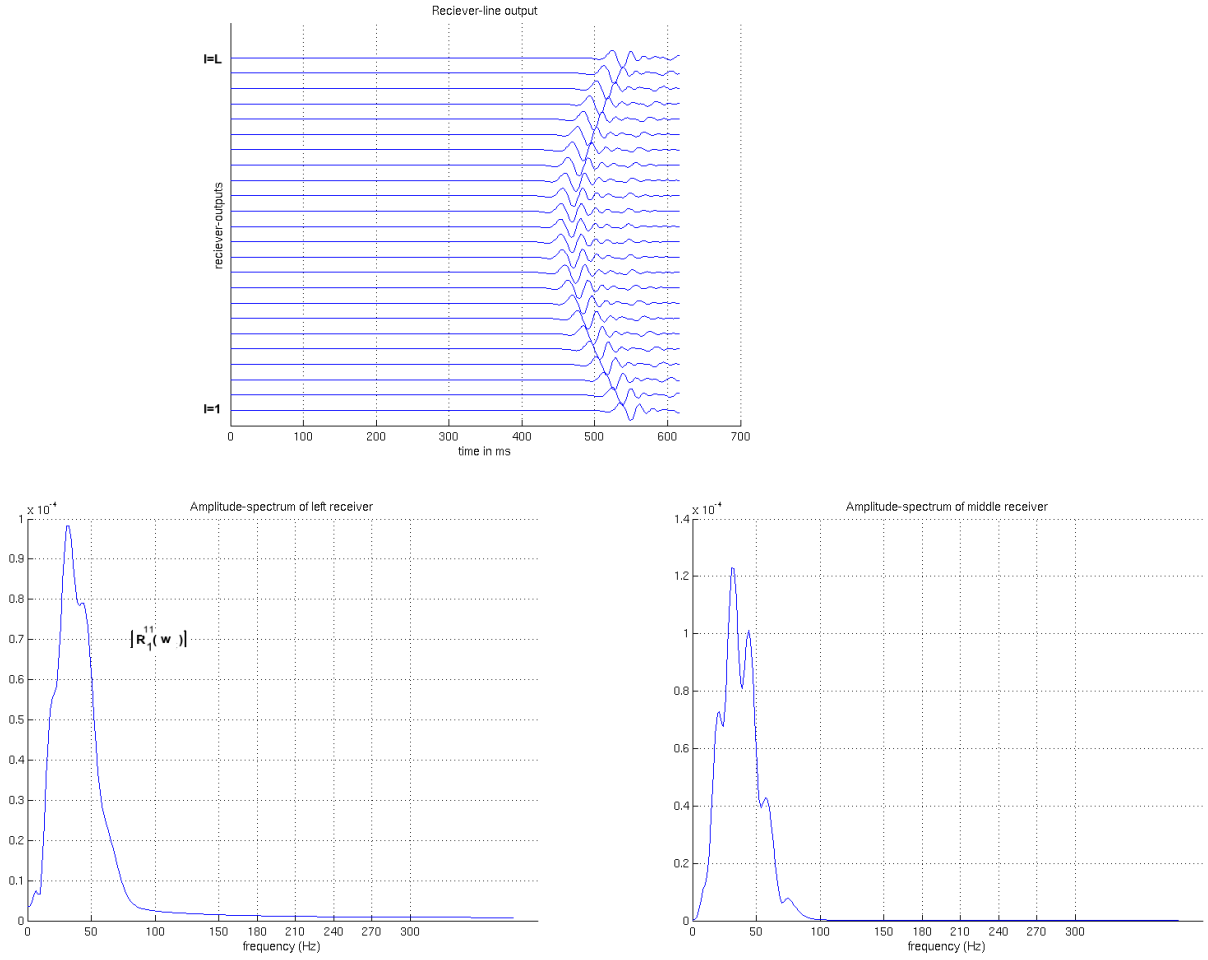

Figure 3.19: Temporal recordings when the middle source in Figure 3.17 is excited. The output on each receiver is shown in the upper figure. The amplitude spectrum of the left most and the middle receiver is given respectively in the lower left and right figure.

now to use the same frequency for each iteration and normalize this vector with the Fourier transformed source. A column from exciting source number i is thus given by:

$$\vec{K}_i(w_0) = \frac{\vec{R}^i(w_0)}{S_i(w_0)} \tag{3.13}$$

This process is repeated until all columns are computed. That is, when all sources are accounted for. Choosing a frequency can be done arbitrary within the common bandwidth of the source and receiver signals. If it is desired to use multiple frequencies this process can be repeated where the matrix from every computation is kept. Herein we will be working with a frequency of about $30Hz$ which is close to the peak-frequency in the amplitude spectrum of simulated outputs. We have now seen how to experimentally build the transfermatrix $\mathbf{K}(w_0)$. Only one thing remains to be able to employ the imaging techniques. The Green's function of the background medium. As a model we use a Hankel-function of the first kind, a $2D$ impulse response which is accurate when used properly. The expression for the function is given by:

$$G(r, k_0) = \frac{1}{\sqrt{k_0 r}} e^{j(k_0 r + \frac{\pi}{4})} \tag{3.14}$$

Where r is the distance from a unit source to an evaluationpoint and $k_0 = \frac{w_0}{c}$ is the wavenumber. It has asymptotic behavior for small distances so scattering targets must be at least a few wavelengths away from the arrays.

Chapter 4

Simulation and imaging

Intro

In this chapter we employ the two imaging algorithms MUSIC and backpropagation on data generated by computer simulations of propagating waves. The theory from Chapter 2 will be applied and tested in various configurations of scatterer locations relative to an array of sources and receivers. When the theory was presented we made assumptions which in general is rare or maybe impossible to achieve. Especially consering the ability to resolve scatterers independent of their geometry. In real physical experiments the Born data model might be close enough so that the theory on this subject is valid to a certain extent. But perfectly resolved point targets is a demand that is hard to meat in the physical world. The algorithms are used in situations that obviously violates that assumption but never the less, we will see that images computed based on this theory can give very specific information about scatterers. The first part is a verification of the imaging algorithms where we work with a model that ensures Born data. When this is done we will create images from data generated by simulating waves radiating through two spatial dimensions, this was described in Chapter 3. Both algorithms are employed to cases of scatterers with well resolved geometries. There are two subjects we are especially interested in this chapter. When scattering targets are closely spaced we look at how the performance of MUSIC is affected. In physical experiments there is often some kind of noise distorting the data, so we will add noise to the simulated data and employ MUSIC to check its robustness towards different noise ratios. The imaging techniques are employed in the manner as presented in section 2.3 but some adjustment are also considered. We will look at the outcome of manipulating MUSIC by considering alternative usage of the array related vectors. We have not given much thought to the array geometries previously. But this does have an effect on the results. This chapter starts by viewing how extending the arrays and using multiple frequencies affects images.

4.1 Ideal point spread

The ability to resolve point scattering targets depends both on the range of frequencies available and array geometries. To easier discuss the results of MUSIC and backpropagation later on we first look at these effects. Herein we obtain an ideal point spread (or ideal point source) by considering three factors:

- Ideal propagation
- Array range
- Frequency availability

The waves detected by the array contains only the direct wavefield measurement when a temporal delta acts as the source (or secondary source). Direct wavefield meaning that waves propagate according to the Greens function of the background medium only. No external factors are influencing the propagation. The point source then converges towards being perfectly resolved as the array range increases and more frequencies are used. In Figure 4.1 we have a model with three different configurations of transceiver arrays relative to a centered point scatterer. Here we consider transceivers as sources that are also able to detect the scattered waves. A homogeneous background model of velocity $500\frac{m}{s}$ is used where we start with one linear array and add more. All linear arrays have equally spaced transceivers separated with $8.33m$ which is half the wave length of a signal with frequency $f = 30Hz$. This sample spacing will not have any effect on the reconstruction of measured signals here since the data is set according to the ideal point spread. This wave length will be denoted:

$$\lambda_0 = 16.7m \tag{4.1}$$

It defines the coordinates in all images and configurations in this chapter (Parameters here are chosen similar to those we will be using later on when they need to be set according to the signals). When the images are computed, it is assumed that the source generated at the centered scatterer is a temporal delta $\delta(t)$. The delta is a signal containing all frequencies, such that in the w-domain we get:

$$F\{\delta(t)\} = S(w) = 1 \tag{4.2}$$

In a receiver we then measure only the mediums Green's function (or impulse response) evaluated from the scatterer to its position. To create an image the detected signals are complex conjugated (or time reversed) and propagated back into the medium from every location where it is detected. We then get complex values at each point in the working grid. Physically this can be interpreted as the behavior of these fields summed throughout the medium.

To view these images we consider the absolute values and images are given as contour plots. The images in Figure 4.2 are results when data from the scenarios given in Figure 4.1 is used. As expected we get a peak at the target location and values close to zero away from it. If we look at the upper image we have the result from when a single linear array is in place. We clearly see that locating the target is more difficult in the direction perpendicular to the array, the peak is stretched in this direction. When more arrays are added the following images displays how the stretch reduces as they recover more space around the scatterer. In Figure 4.3 we have a comparing of the images created when using a single frequency (left images) and by employing several frequencies (right images). In the upper ones we have data from the configuration of two arrays. The bottom images are created from data in the scenario of four arrays. When images from several frequencies are summed together (complex image values) the spatial stretch remains but the peak increases while the surrounding values remains low. With an infinite array the image will converge to a peak at the point source. If we have an infinite range of frequencies we can achieve a delta distribution at this point.

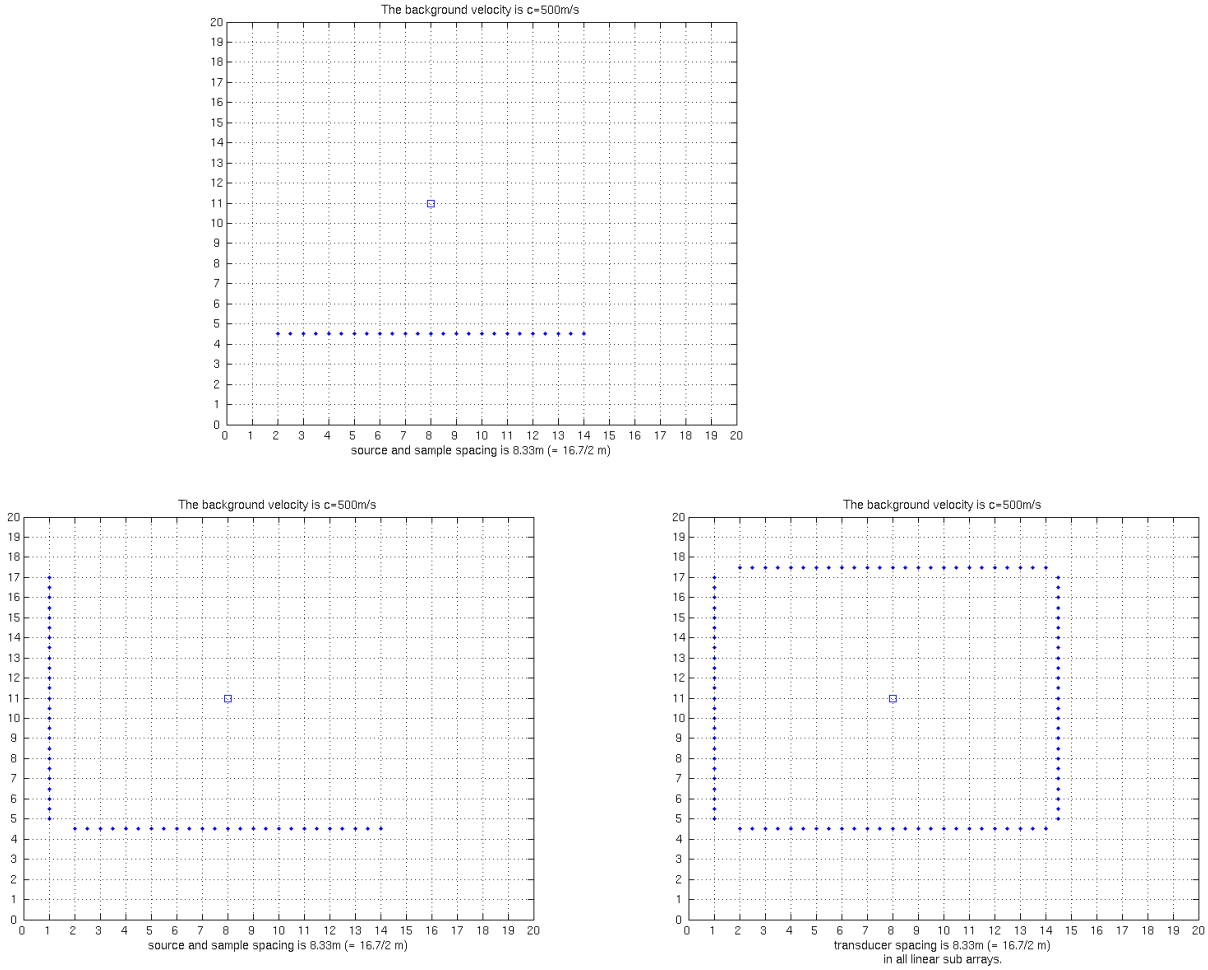

Figure 4.1: The upper left to lower right figures displays three linear transceiver array configurations relative to the same scatterer located in a homogeneous background of propagation velocity $500\frac{m}{s}$. From the scatterer position a temporal delta is excited. Coordinates are given in $\lambda_0 = 16.7m$ and the transceivers in the linear arrays are equally spaced with $\frac{\lambda_0}{2}$. From the top left to bottom right: one, two and four linear arrays encapsulates the target.

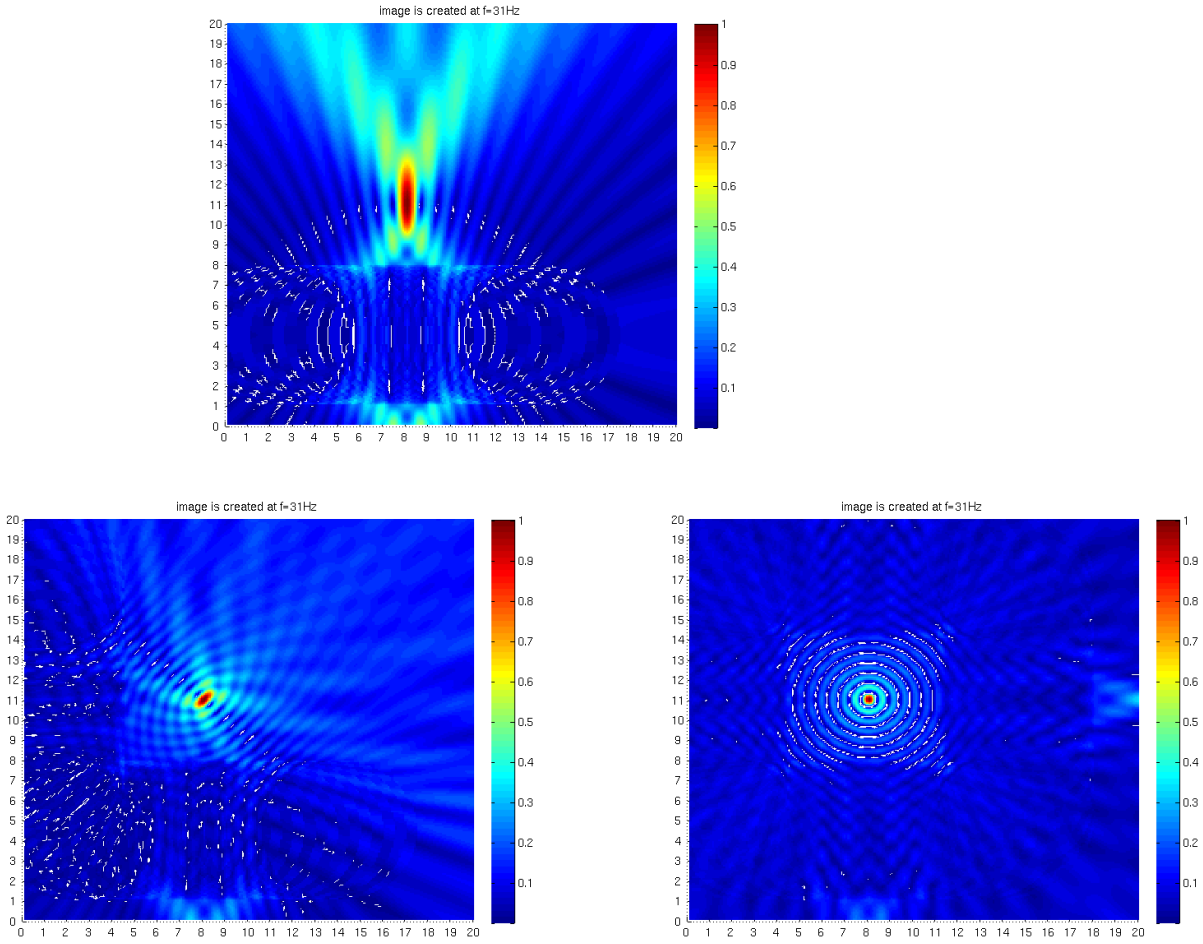

Figure 4.2: The three images are the results of backpropagating ideal signaldata from the three transceiver array configurations given in Figure 4.1. In the upper left image we have the image created when one linear array is used, the lower left and right images are results from using two and four linear arrays respectively. All images are created using the frequency $f = 31Hz$.

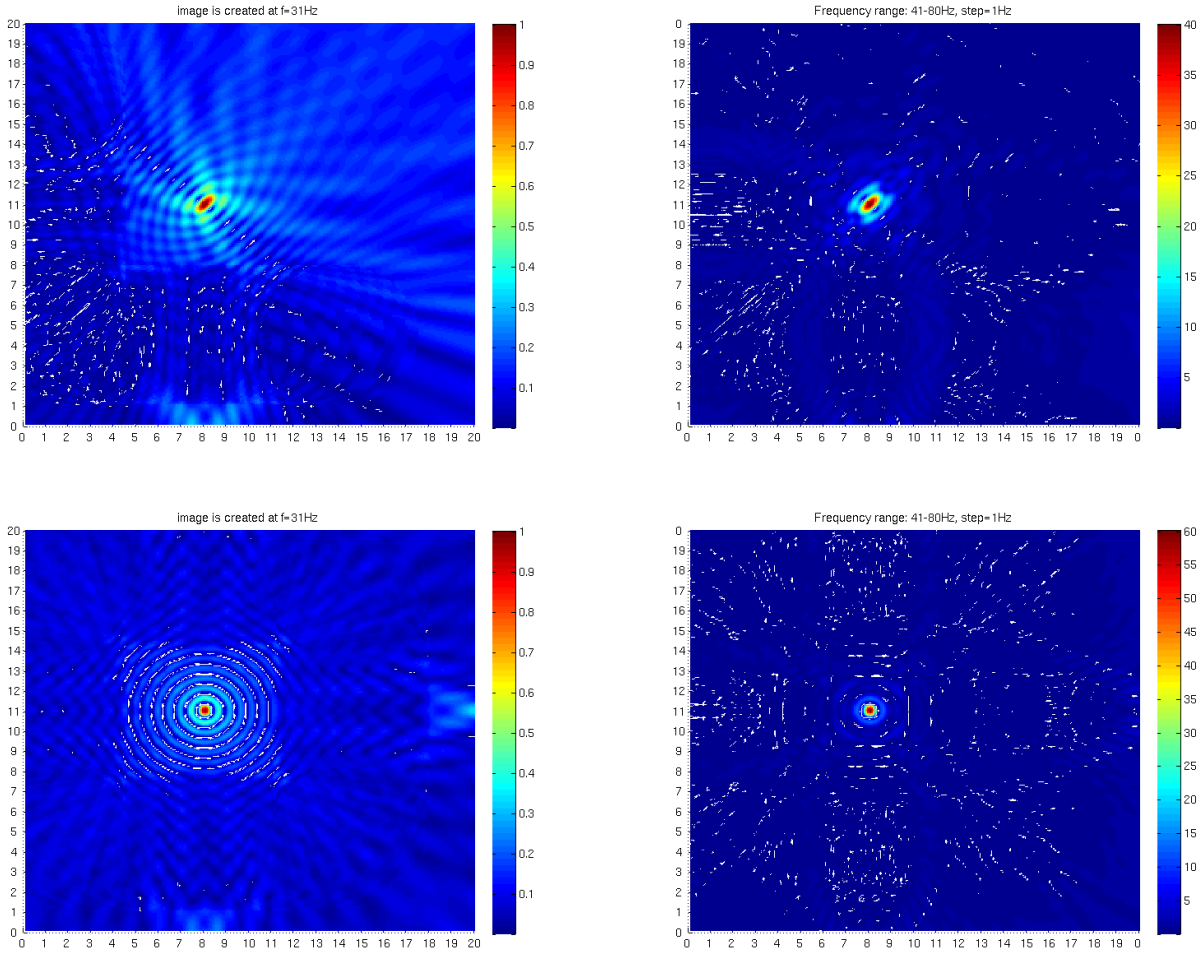

Figure 4.3: This figure compares images computed using a single frequency and the sum of images from several frequencies. To the left images are computed at $f = 31Hz$ with two and four linear transceiver arrays as in Figure 4.2. The right images are created with the same configurations but by using 40 frequencies in range $41 - 80Hz$.

To further explain what happens when we use multiple frequencies we consider Figure 4.4. Again we have used the configuration with two linear arrays but we only keep track of values in two positions. This is done at the location of the scatterer (blue) and at a position away from it (red). Displayed here is the real and imaginary part of of these measurements (bottom left figure) and the absolute values of the cumulative distribution. The cumulative distribution is given by the total sum for every added image, or every new frequency used (bottom right). If we compare the graph from the target location with the one from the other position, we see that it is constant 1 for all frequencies. While the other location has positive and negative values both in the real and the imaginary part. As result the cumulative distribution always increases where the scatterer is located. In the non-scatterer location we get cancellations that makes it stay below a constant value.

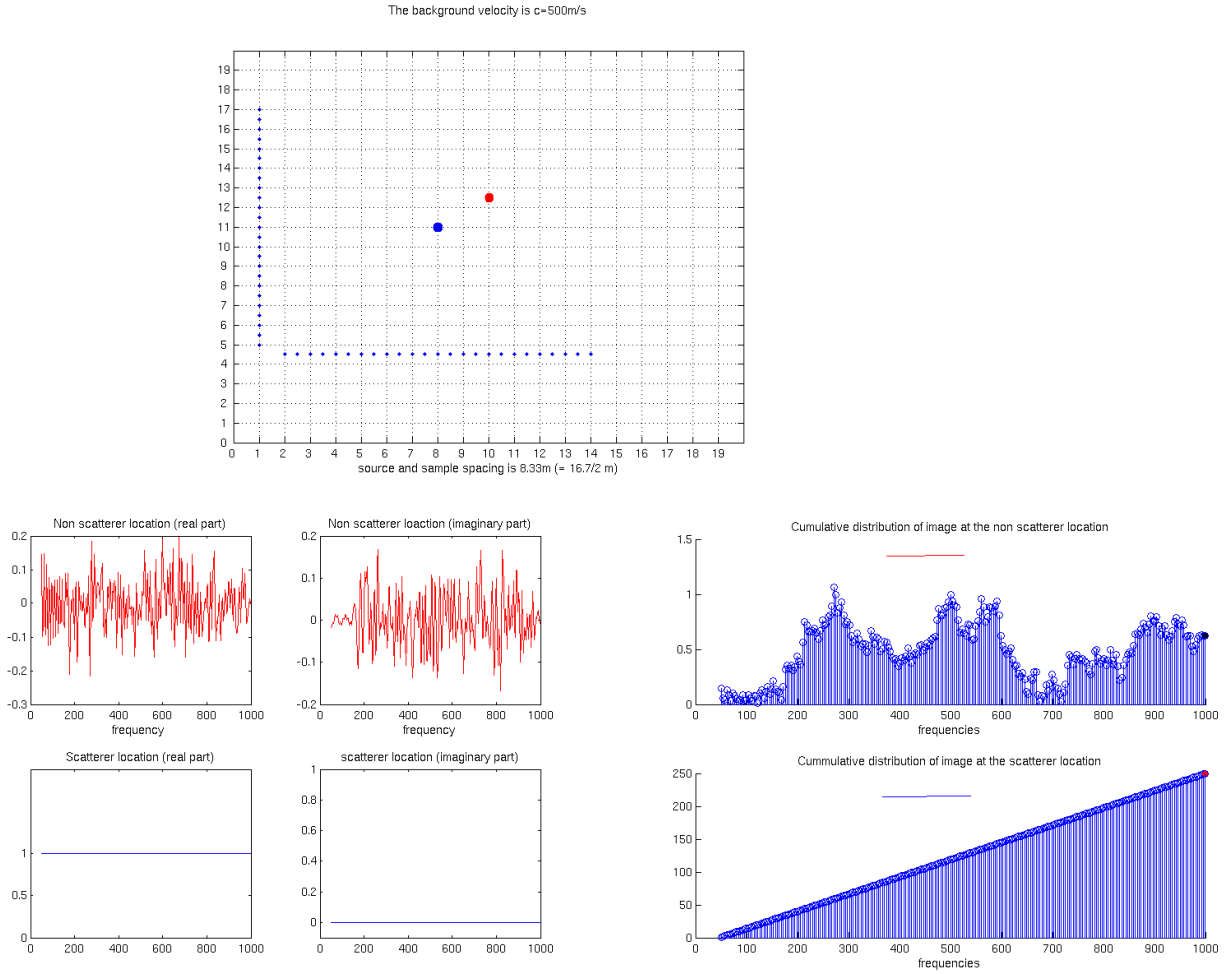

Figure 4.4: The upper figure shows a scatterer centered relative to two linear transiever arrays. The blue dot indicates the scatterer position and the red dot gives a point away from this. Ideal signaldata is backpropagated to both positions using 250 equally spaced frequencies in range 10-1000Hz. The lower left figure shows values at the imagepoints as function of frequency. The lower right gives the cumaltive value at both positions up to frequencies in the same range.

4.2 Simulations using Born data

The mathematical description of our two imaging algorithms includes an assumption of the ability to perfectly resolve targets. This assumption is made regardless of array versus scatterer geometry and frequencies available. Such an assumption is almost impossible to meet in physical experiments. But because of the underlying physics of time reversal and the fact that it makes waves refocus on sources. Assuming this leads to an easier mathematical formulation describing this behavior. In the previous section we saw how the resolution of a point scatterer will depend on both the arrays and the signal bandwidth available. In this section we will verify that the algorithms can be applied to data generated in a strict Born sense and give images that follows the desired behavior. Also we will check if it is useful to use multiple frequencies when employing the backpropagation algorithm. To simulate experiments with Born data, the transfermatrix is constructed by the model given by (2.34) of section 2.2.2:

$$\mathbf{K} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \tag{4.3}$$

This structure is imposed by including the scattering potentials $\tau(w)$ in the diagonal matrix $\mathbf{\Sigma}$. The Green's function vectors are computed between the arrays and the scatterers belonging and placed in $\mathbf{U}$ and $\mathbf{V}^T$. Further more, the Born data acquisition is simulated in a structure of transmission from one array and detection in another. Figure 4.5 shows the typical array geometry used from this point on. The vertical array displays source locations and the horizontal gives the receiver positions. A complete process of exciting all sources and detecting signals is given by the linear transformation:

$$\vec{R}(w) = \mathbf{K}(w)\vec{S}(w) \tag{4.4}$$

When working with Born data we assume that a temporal delta is the source input from all locations, such that the vector $\vec{S}(w)$ contains only ones. This gives us the opportunity to use any desired frequency when employing the algorithms. In both arrays the spacing is set to $\frac{\lambda_0}{2} = 8.33m$ and the medium velocity is $c = 500\frac{m}{s}$. These parameters will be used throughout this thesis and although we could set these arbitrary when generating Born data, they are chosen so that we have the same layout here as when we later compare results with those from data generated by simulating propagating waves. The element spacing is set to be about half the wave length of signals with a temporal frequency $f \approx 30Hz$. At the target positions the medium velocity is assumed to be $750\frac{m}{s}$ or 1.5 times the background velocity. If we consider the scattering potential at $30Hz$, calculated by the formula (2.25) of section 2.1, we get $|\tau(2\pi * 30)| = 0.079$. This potensial is set equal for all perturbing points.

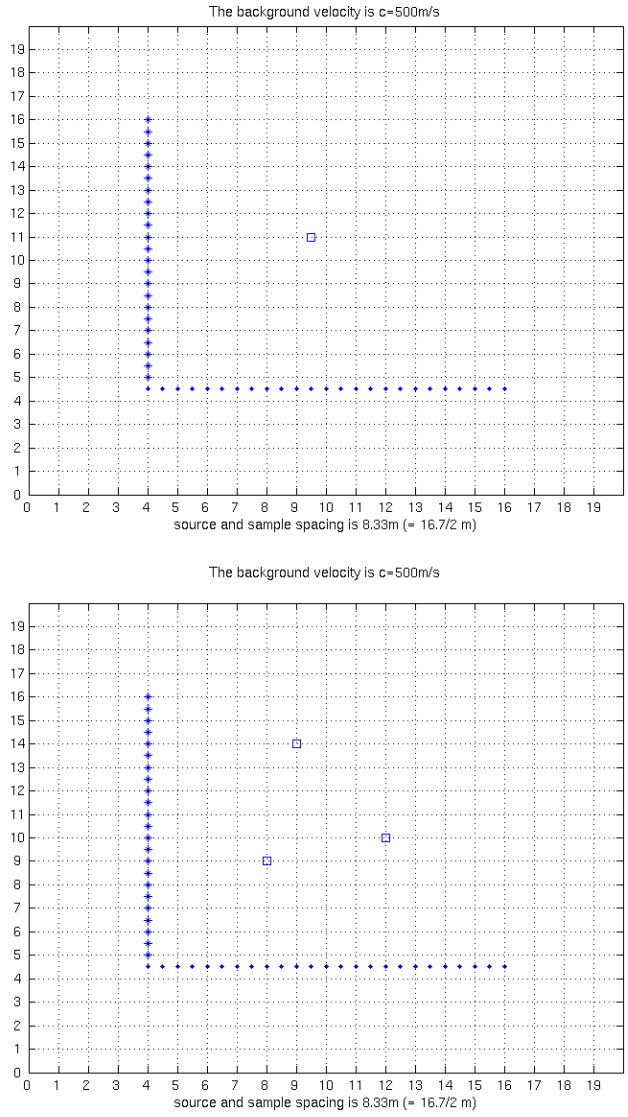

Figure 4.5: Both figures displays the same array geometry of sources(vertical) and receivers(horizontal), one with a single scatterer and the other with three scatterers. The background propagationvelocity is $500\frac{m}{s}$. The source and receiver spacing is $\frac{\lambda_0}{2} = 8.33m$ and thereby the model is constructed for wave signals with frequencies of about $30Hz$. All coordinates are given in number of $\lambda_0 = 16.7m$. At the scatterer positions the propagation velocity is set to be 1.5 times the backgroundvelocity and thus the scattering potentials are equally given by $|\tau(w)| = 0.079$.

4.2.1 Eigenvalues in case of well separated scatterers

We now study results of performing an eigenvalue decomposition of the Born time-reversal matrix (or singularvalue decomposition of the corresponding transfermatrix). The relations of these eigenvalues is described in sections 2.2.4 and 2.2.5 under the assumption of perfectly resolved targets. The two scenarios shown in Figure 4.5 display one case with a single scatterer and another with three scatterers. In the latter the smallest target spacing is $4.12m$ and thus they are separated so much that we should have a well resolved situation. Eigenvalues computed by using data from the two given configurations can be seen in Figure 4.6. They clearly indicate that the number of non-zero eigenvalues is equal to the number of scattering targets. In the theory it was stated ((2.59) of section 2.2.5) that the eigenvalues are proportional with the norm of the array related Green's function vectors and the scattering potentials. In Figure 4.6 the values from the time reversal matrix is compared with the values computed by this formula. In the single scatterer situation, eigenvalues from Born data and theory based calculations are almost identical. When there are more targets present we get some small deviations. We now have two situations where results from the theory are close to those computed by generating data. As conclution, the number of eigenvalues indeed corresponds to the number of scatterers, but we can not state that there is a strict one to one relation between them. This is further discussed at the end of the verification part in section 4.2.4.

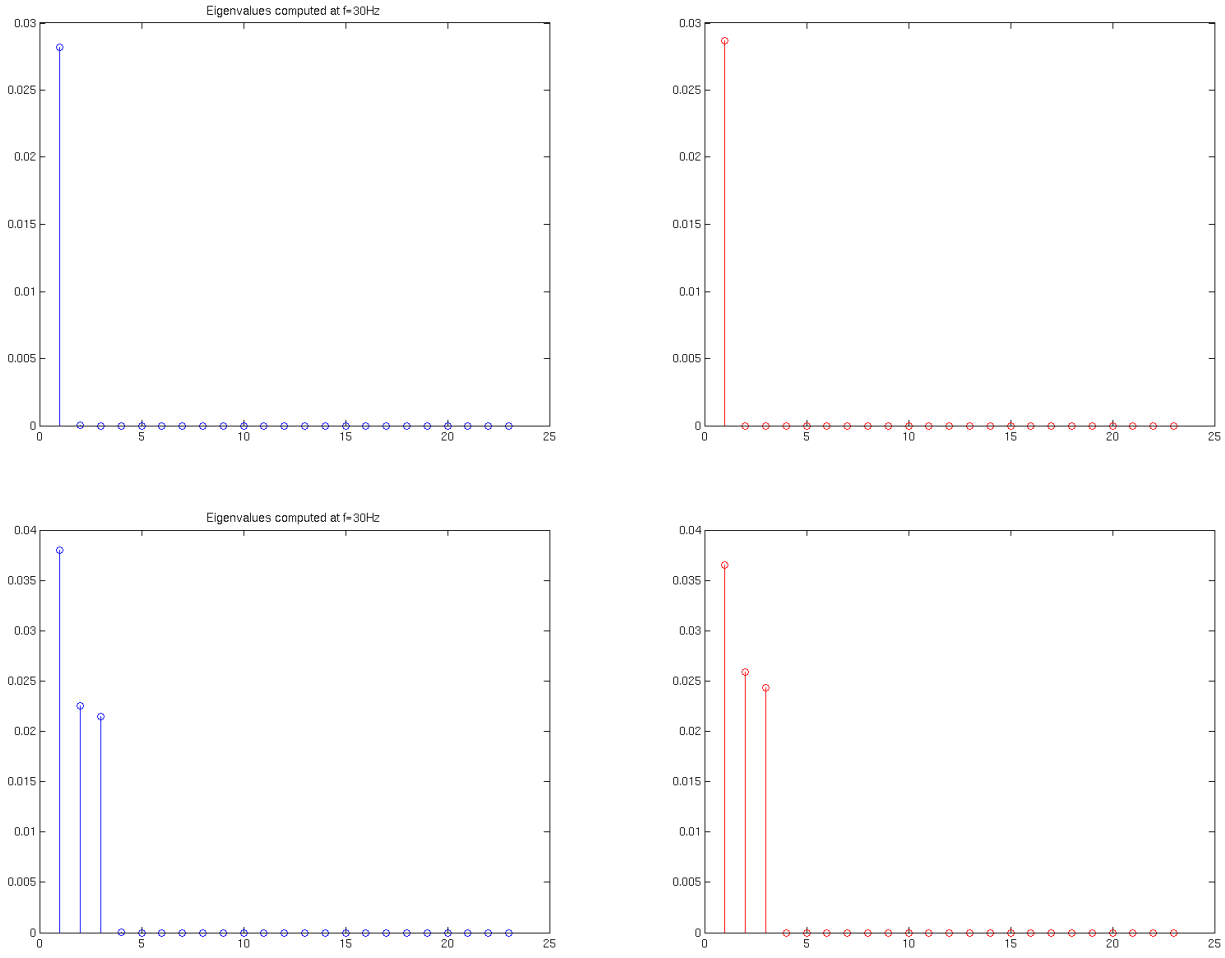

Figure 4.6: The two figures gives the resulting eigenvalues computed according to the experiments presented in Figure 4.5. (The upper figures display results from the single scatterer scenario and the lower gives the eigenvalues from the scenario with three scatterers). Eigenvalues calculated with Born-data are shown on the left and values calculated by using (2.59) of section 2.2.5 is displayed to the right (theory of perfectly resolved targets). All computations are carried out with $f = 30Hz$.

4.2.2 Backpropagation in case of well separated scatterers

The images in Figures 4.7 and 4.8 are computed by employing the backpropagation algorithm described in section section 2.3.1. From (2.70) and (2.71) we have the two functions $P^i_{br}(\vec{x}, w)$ and $P^i_{bs}(\vec{x}, w)$. These gives us the image distribution with respect to the receiver array and the source array for any position $\vec{x}$ at the angular frequency w. Where i denotes the eigenvector we are using and thus which eigenvalue it correspond to. From this point on we restrict the positions $\vec{x}$ in both directions to be limited by the source and receiver array lengths. The image computed from one set of eigenvectors is given by the sum:

$$|P^i_b(\vec{x}, w)| = |P^i_{br}(\vec{x}, w) + P^i_{bs}(\vec{x}, w)| \tag{4.5}$$

Generally the algorithm depend on the ability to determine the number of non-zero eigenvalues and we may evaluate images individually (one image per eigenvector) as in (4.5), or we can view the total image by summing the distributions as followed:

$$|P_b(\vec{x}, w)| = |P^1_b(\vec{x}, w) + ... + P^M_b(\vec{x}, w)| \tag{4.6}$$

Here M gives us the number of non-zero eigenvalues considered. The data used is generated from the configurations given in Figure 4.5 at $w = 2\pi * 30Hz$. All images but the one given at the bottom right in Figure 4.8 are formed by employing (4.5). They have the strongest peak very close to the target positions. The white lines gives the accurate ones. In addition there are lobes that increases towards the targets, in Figure 4.7 a $3D$ image from the case with a single scatterer is shown. Here we see these lobes more clearly. One reason for the disturbances around the peak and small displacements is explained by array geometries. $P^i_{br}(\vec{x}, w)$ and $P^i_{bs}(\vec{x}, w)$ are not completely synchronized in locating the targets. It is expected that these distortions will decrease if arrays are extended. Despite these additional peaks we are completely able to determine the correct number of scatterers and also get a good indication on where they are located. In the bottom right image of Figure 4.8 it is also included a result where the backpropagation is carried out as the total sum given by (4.6). What happens here is that one peak (leftmost target) gains a higher value, but the others are suppressed. So when using this algorithm as herein one could expect better images by using the algorithm with each vector individually. This is also how images will be computed when performing backpropagations from here on and out.

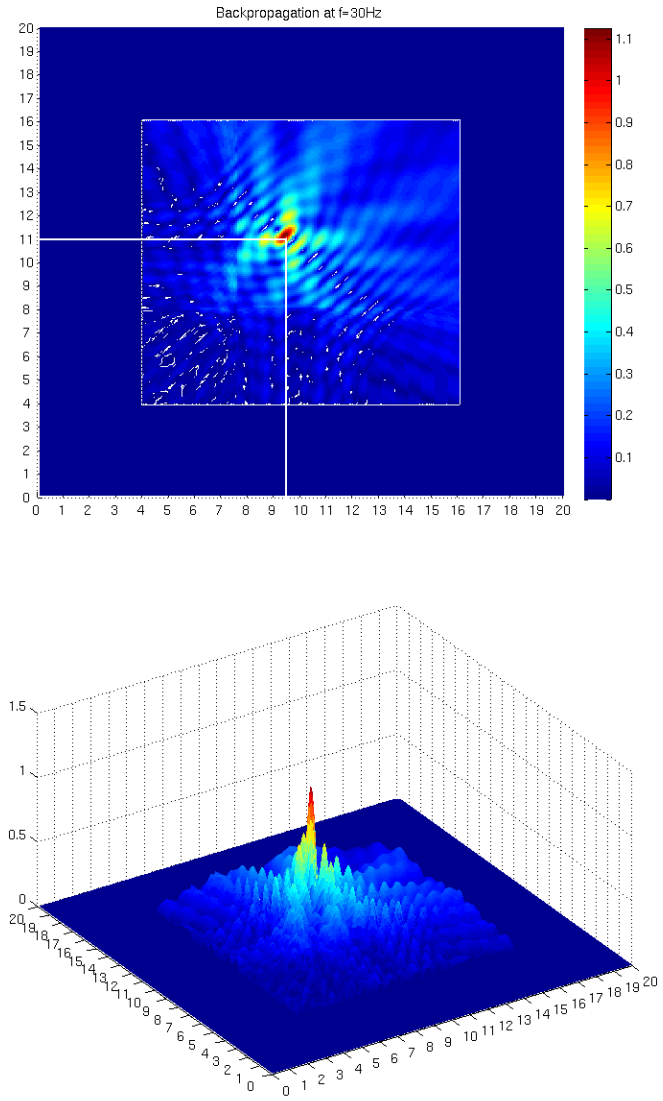

Figure 4.7: Result of employing backpropagtion using data from the experiment with a single scatterer given in Figure 4.5. A contour and a 3D image is shown.

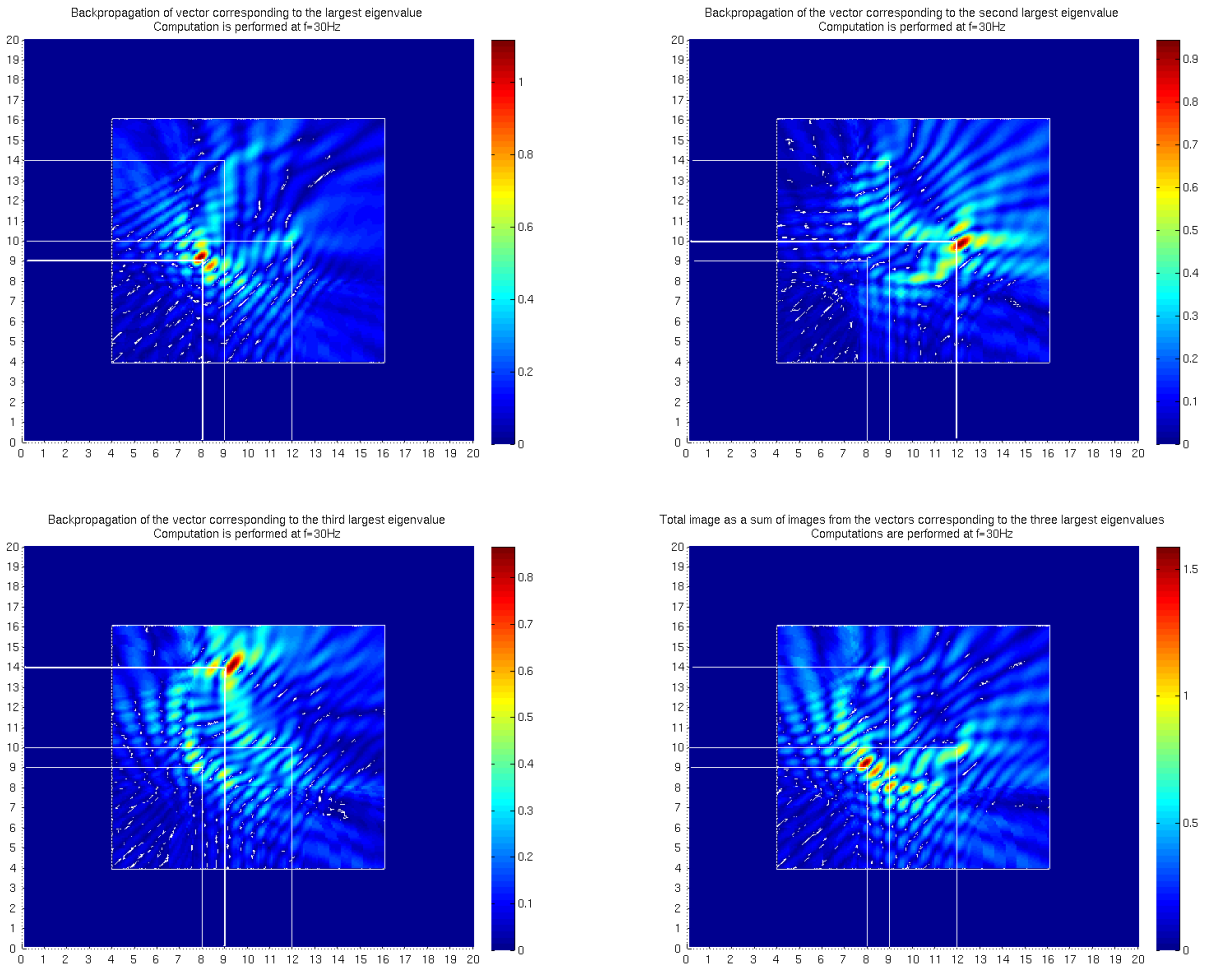

Figure 4.8: Computations are done with data as in the experiment with three scatterers, see Figure 4.5. Figures shows individual images as result of backpropagating one and one eigenvector. Vectors with correspondence from the largest to the third largest are used and are displayed respectively from upper left to lower left. The lower right image is the total image as a sum of images from the individual backpropagations.

4.2.3 Backpropagation using multiple frequencies

In section 4 we saw how to obtain better images by combining images from several frequencies. This is tested with data from the configuration with a single scatterer in Figure 4.5. In Figure 4.9 images from twenty different frequencies are summed, the result is compared with the one from a single frequency. Multi frequency images are given by the cumulative distribution:

$$|P_b(\vec{x})| = |P_b(\vec{x}, w_1) + ... + P_b(\vec{x}, w_{Nf})| \tag{4.7}$$

Where Nf gives the total number of frequencies employed. The scatterer is resolved better in the sense of achieving a more narrow peak area, but the image is distorted by high values away from this location. To get more insight in how employing multiple frequencies evolves, consider Figure 4.10. Image values in two locations are computed for 200 equally spaced frequencies in range $10 - 1000Hz$. The evaluated points are the scatterer position(blue) and a location (red) with coordinates $(15\lambda_0, 10\lambda_0)$. Here we have additional high values in the image from Figure 4.9. Remember now the results from when the ideal point spread data was used. At the scatterer position we had a constant real value and a zero imaginary part for all frequencies. Here we get values that oscillates around zero in the real part and various values in the imaginary part which diverges by swapping between negative and positive values. The resulting cumulative distribution at this point is maintained at about 1. We thus get the same behavior at the scatterer position as in the non-target location, the values stabilizes. So although we will get a focus on the targets in every single frequency image, we do not achieve the same as with the ideal point spread data. With the data available from this borehole geometry. We know that target locations aren't completely synchronized from the two arrays. For the multiple frequencies backpropagation algorithm to work, we should have this demand fulfilled. Since images very much depend on the array geometry we can expect this to be improved if arrays are extended. To obtain a synchronization we would need to study geometries more closely but we are here interested in results from the data already available. The backpropagation algorithms ability to image targets is thus reduced to the single frequency case herein.

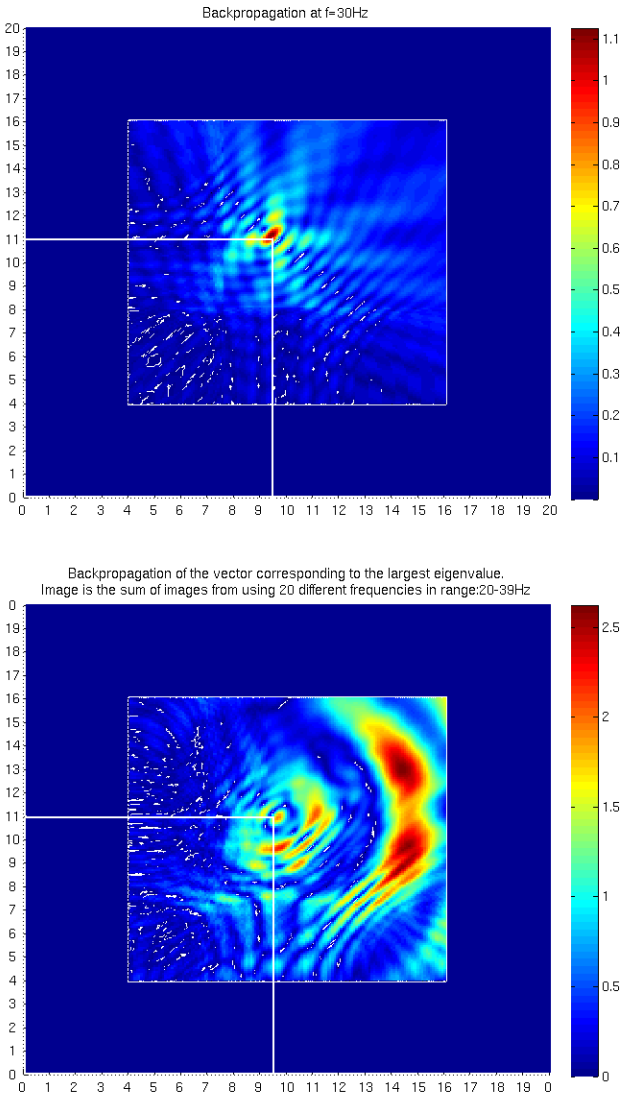

Figure 4.9: Images are created using data from the configuration given in Figure 4.5 with a single scatterer. The upper image is computed at $f = 30Hz$ and the lower one is a sum of images from 20 different frequencies. These are in range $20 - 39Hz$.

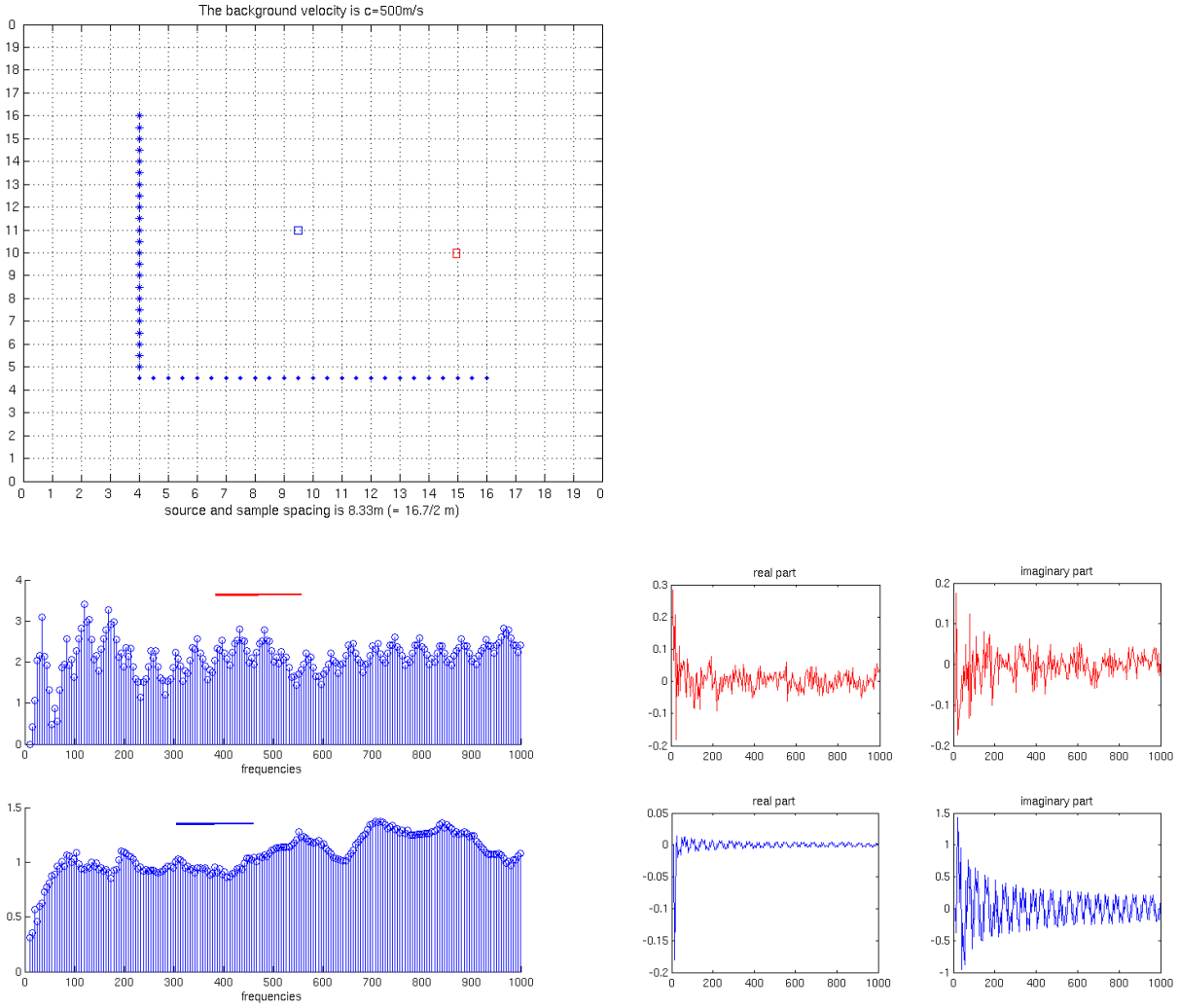

Figure 4.10: The system given in the upper figure shows the single scatterer scenario as used previously, see Figure 4.5. The points of consideration is the scatterer position(blue) and a point where we got additional high image values when several frequencies where used(red), see Figure 4.9. The bottom left figures gives the cumulative values as function of frequency both for the scatterer location and the non target position (absolute values). In the bottom right figures the image value at these points are given as functions of frequency, (both the real and imaginary part of the values are shown). The frequency range is $10 - 1000Hz$ with 200 different and equally spaced frequencies.

4.2.4 MUSIC

We now verify that the MUSIC algorithm can be used to image the scatterers when employed to Born data. In the same manner as with the backpropagation we look at images from taking the absolute value of the distribution function $P_M(\vec{x}, w)$ from (2.74) in section 2.3.2. Employing MUSIC to the well separated cases with Born data proves to be successful to the extent that a small additional value has been added to the formula. In the modified version $\sigma = 10^{-5}$ is included such that:

$$P_M(\vec{x}, w) = \frac{1}{\sum_{k=M+1}^{min\{L,N\}} |\vec{e}_k^T \vec{g}^s(\vec{x}, w)| + |\vec{v}_k^T \vec{g}^r(\vec{x}, w)| + \sigma} \tag{4.8}$$

Without the adjustment the images shows peaks that are so high and narrow that they are hard to spot. (4.8) includes the eigenvectors related both to the receiver and the source array but now corresponding to the zero eigenvalues. So when using this imaging technique it also depend on the ability to separate values related to scatterers and those that are not. In Figure 4.11 we again have the data from Figure 4.5, the two well separated scatterer configurations (single and multiple targets). Images are computed using the angular frequency $w = 2\pi * 30$. Without discussion this imaging technique gives very good results. The values are very high in targets locations and compared with the images from the backpropagation, the peak areas are small and close to exactly positioned. White lines gives the correct scatterer locations. The problem we met when employing the backpropagation technique, that arrays did not completely agree on locations, is not an issue here.

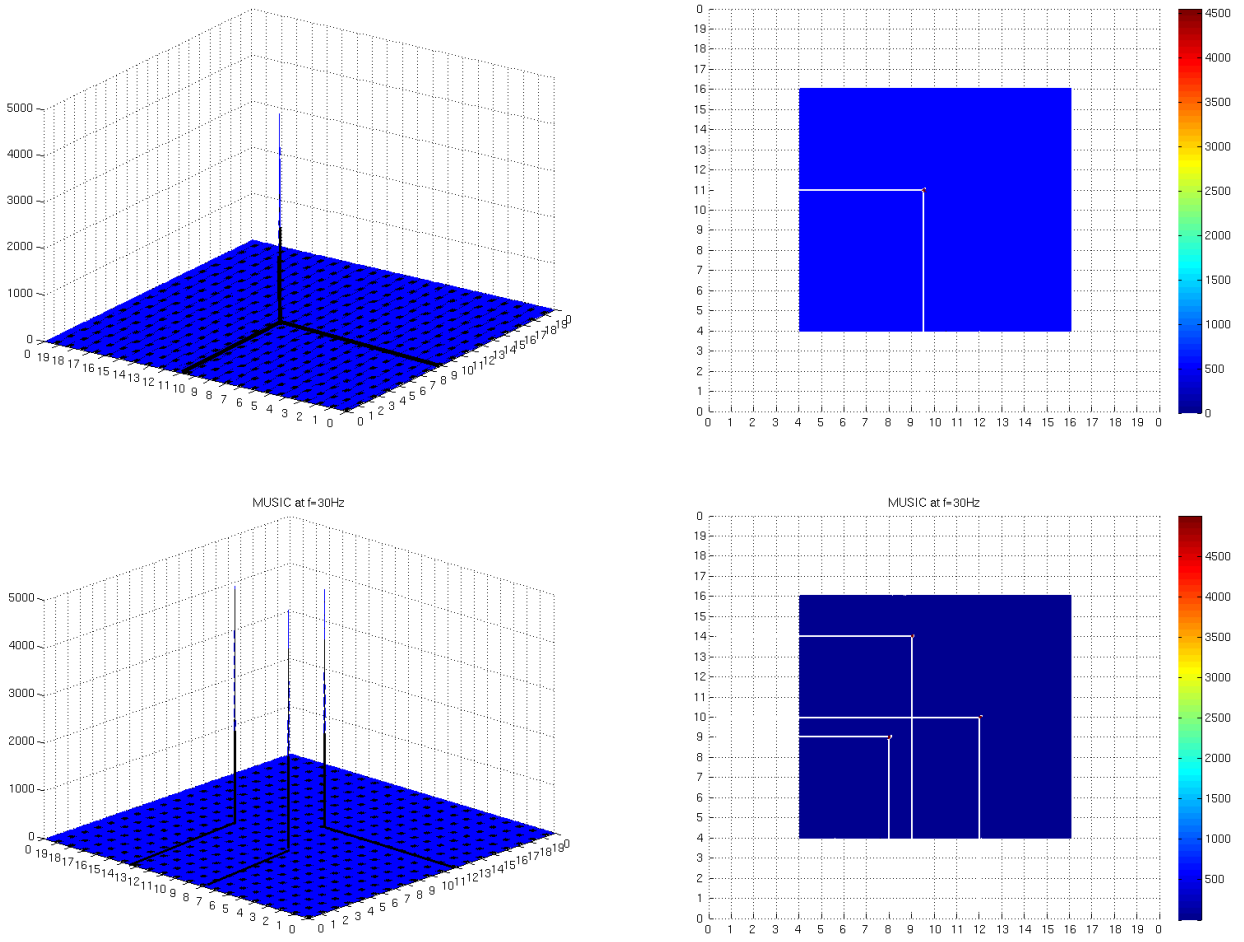

Figure 4.11: Results of employing MUSIC to data from the experiments with a single scatterer and three scatterers given in Figure 4.5. Contour and 3D images of the same results are displayed. The upper images are computed when there is a single scatterer present and the lower when there are three scatterers. All images are results of a computational frequency $f = 30Hz$.

4.2.5 Relations between eigenvectors and scatterers

With Born data we are ensured that no multiple scattering takes place. That is, simulated recordings will not contain waves that has been reflected between one scatterer and another. In the first part of this chapter we found that the eigenvalues came close to those from theory. But we used Born data with cases of well separated targets and there are still some indications of eigenvectors not having a strict one to one relation with the scatterers. Consider the two configurations with the same array geometries given in Figure 4.12. The related eigenvalues are shown on the right hand. First we have a situation with two scatterers and next one of them is removed. If there where a one to one relation between targets and eigenvalues the remaining scatterer should result in the same value. But in this case the value has been reduced some. When both scatterers are present the eigenvalue is larger and has gained size because of the additional target. Since we have that eigenvalues and eigenvectors always appear in pairs, we can conclude that the vector related to the largest value has a relation with the second target. To show the effect of this in images we again use data from the scenario with three well spaced targets. Consider Figure 4.13, there are performed six backpropagation's. In these images we are more interested in values at non-scatterer locations[1]. In images to the left the data are taken from when all three targets are present at the same time. To the right the data includes the same target locations but only one is present in each scenario. Despite the fact that we obtained the exact number of non-zero eigenvalues we can see some differences. These effects are shown best in the four first images. (Remember that when we employ backpropagation we use one vector for each image.) Comparing these results we find that multiple target data gives lower peaks and higher image values where the other targets are located. We should be aware of that this might be in addition to recordings of multiple scattering in experiments. Especially when scatterers are closely spaced.

[1]The right bottom image in Figure 4.13 display a divided peak. This is due to the problem of array focusing. If the arrays are increased this focus will move towards an equal position.

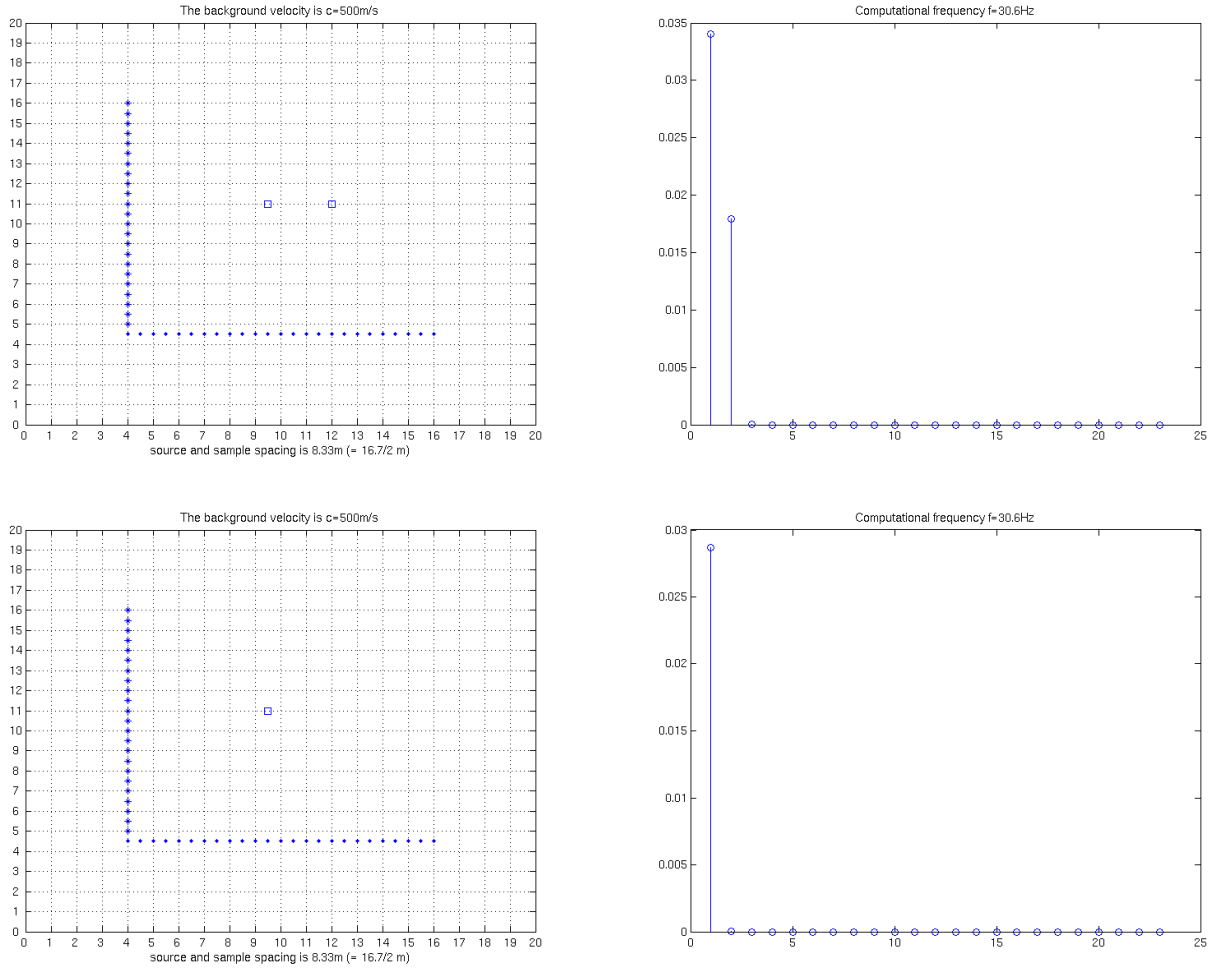

Figure 4.12: Two scenarios are given both with the same array geometries. In the upper figures the system includes two scatterers with the computed eigenvalues. In the second scenario one scatterer is removed and the corresponding eigenvalues are computed. Both results are computed by using the same frequency $f = 30.6Hz$.

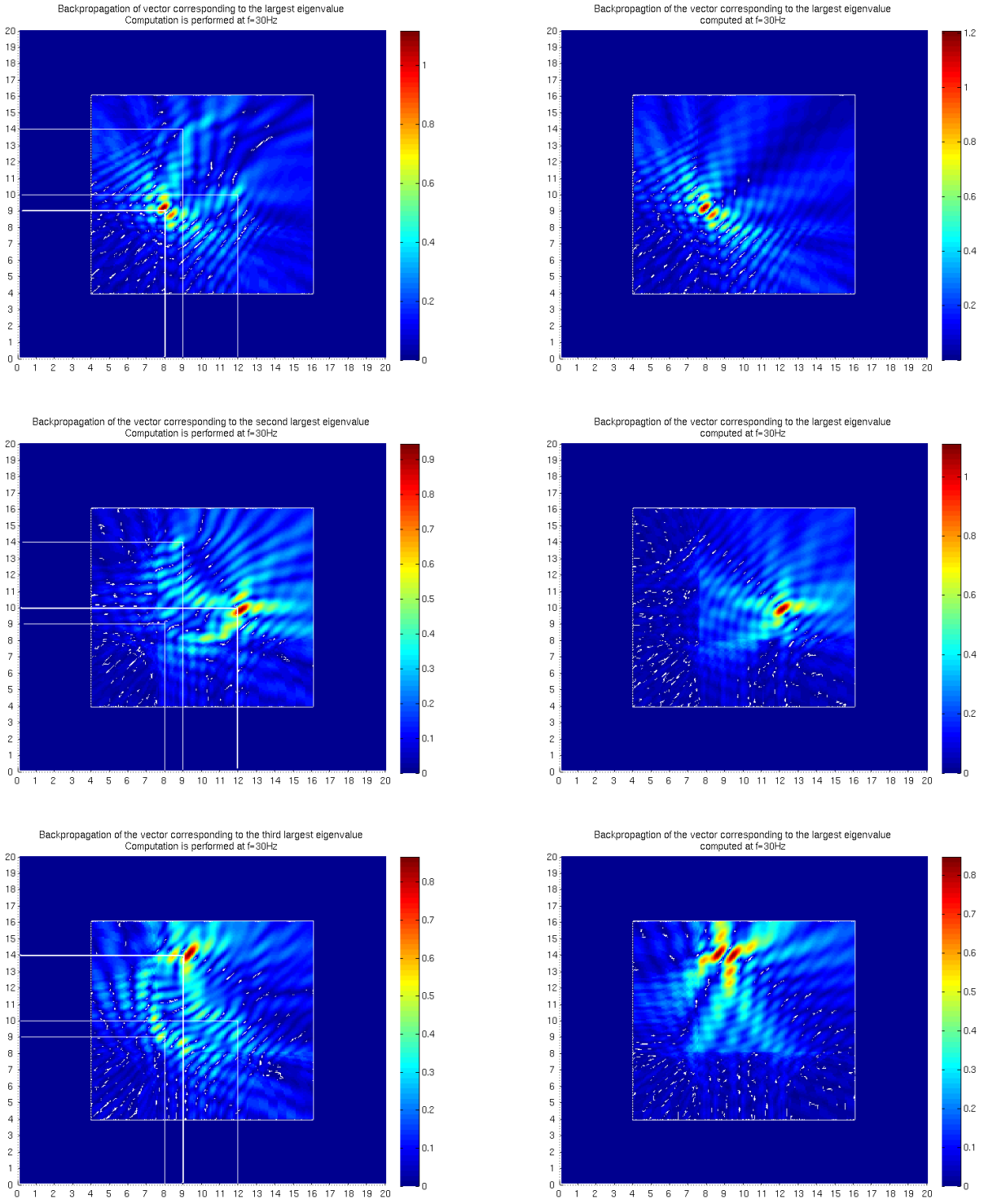

Figure 4.13: The left images displays the same results as in Figure 4.8, the three scatterers experiment. On the right side only one scatterer is present in each scenario and located at the same positions.

4.3 Physical simulations and imaging

We now start working with data generated by employing the 2D modeling program, a description of how this is given in chapter 3. The imaging algorithms are based on theory of perfectly resolved targets. In the previous section we discussed how this assumption is practically impossible to meet when working with data from the timereversal matrix as herein. But we will work with simulations where the Born approximation should be satisfying up to a certain point. In the first part we test the techniques on the same configurations as with Born-data. Array layouts and parameters concerning scattering strengths and propagation velocities remains the same. Input signals and output measurements generated can be viewed in section 3.2.2. Remember that since discrete Fourier transforms are performed on signals we might not be able to compute with the exact frequency of $f = 30Hz$ as before. But we will be using frequencies close to this and the effects from a small difference in the computational frequency makes no noticeable trace in the results. Since MUSIC seems to be stronger algorithm the main focus will be on this technique.

4.3.1 Eigenvalues in case of well separated scatterers

In Figure 4.14 the eigenvalues computed according to the experiments with well separated scatterers are shown. The largest values from the modeling program are reduced compared to the ideal Borne case. This reduction is to about half the size of eigenvalues from Born simulations. We also have some small deviations where values that ideally should be zero are not when there are multiple targets present. The relative relation between "non-zero" values is maintained and the figures clearly suggest a connection between the number of targets and large eigenvalues. In the case with a single scatterer it is strictly obeyed, in this situation no multiple reflections can be generated. With a single point scattering target the Born model should be very close to the exact description. A multiple scattering situation includes data that is not accounted for and this is reflected in the eigenvalues.

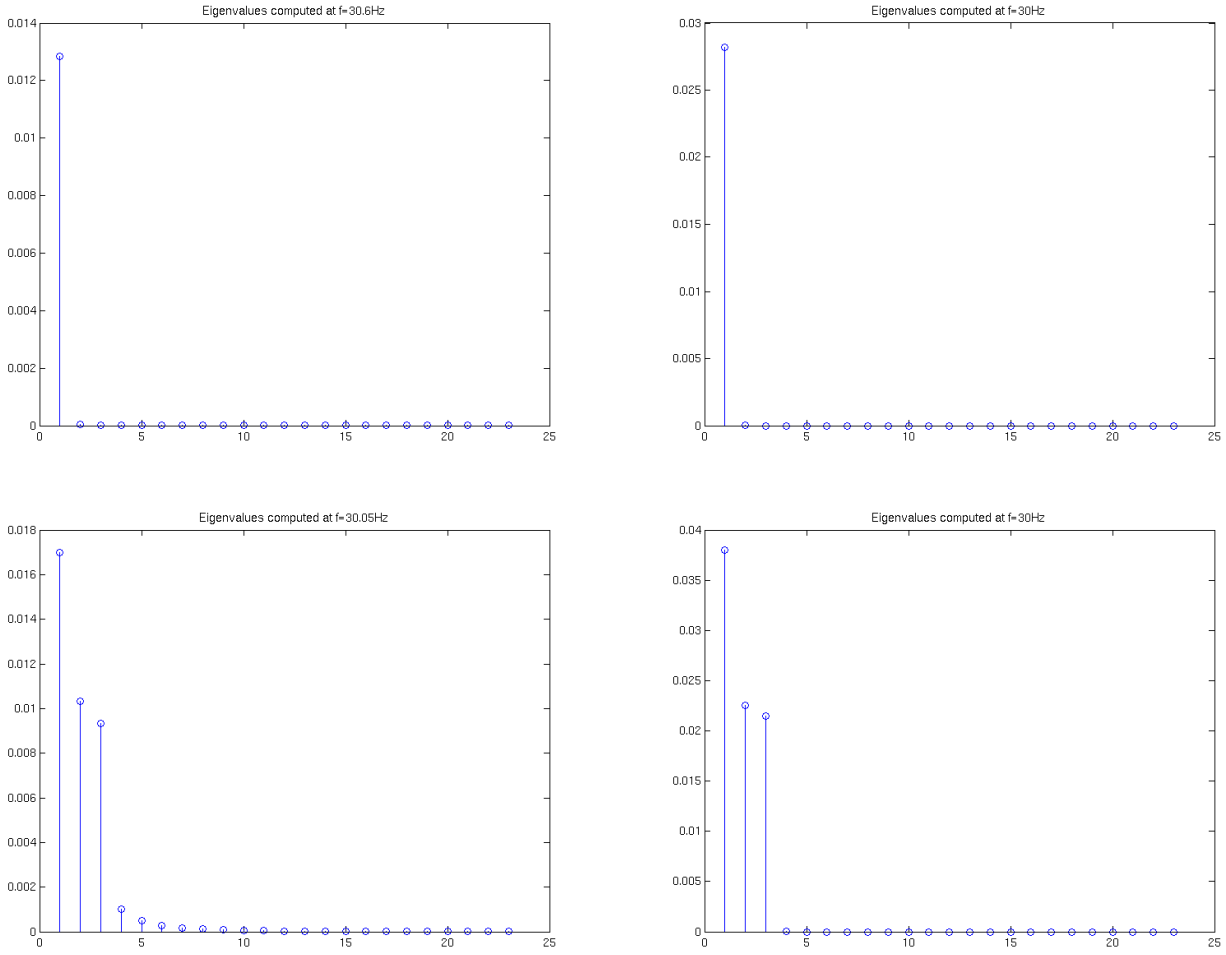

Figure 4.14: Eigenvalues of the timereversal matrix associated with the configurations presented in Figure 4.5, the single and three scatterers experiments. The time reversal matrix is generated by the $2D$ modeling program and with Born-data. Results from the modeling program are displayed to the left and results from Born simulations are given to the right.

4.3.2 Backpropagation in the case of well separated scatterers

In Figure 4.15 a backpropagation is done with data from the experiment with a single scattering target. The image here is very similar to the one created using Born-data. This is also expected since the Born model should be completely valid in this scenario. The non-zero eigenvalue for this configuration was reduced compared with the Born generated one, but it has no impact on the image, in fact the peak is slightly higher here. If we consider the images in Figure 4.16, the case with three scatterers, we again have results that are very similar to those from the Borne cases. The feature of a small displacement in target positions is repeated and as in case of a single target we have higher peaks here compared with Born-data. The algorithm works well both in determining the number of targets and their positions. Achieving better results with data from modeling wave propagation than with Born-data can not be expected in any situation. But these improvements are very small so we can conclude that as long as we have cases of well separated scatterers in an environment where the Born model is sufficient we should get similar results.

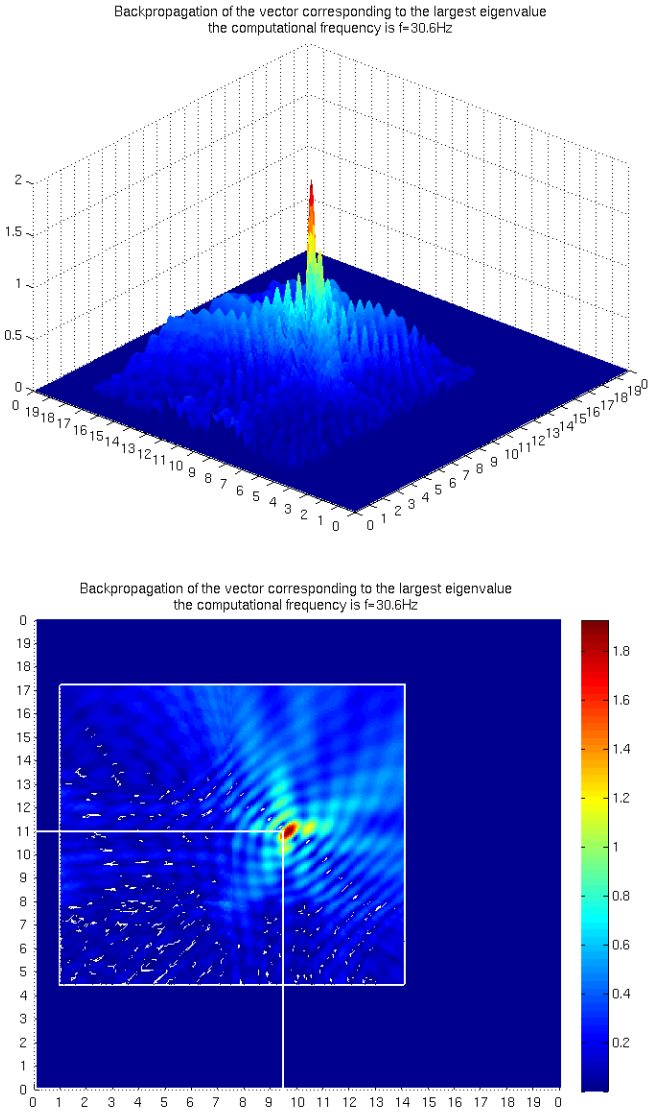

Figure 4.15: Backpropagation of the vector corresponding to the largest eigenvalue with data generated from the case of a single scatterer as given in Figure 4.5. The images displays the same results, the above image is given in 3D and below a contourplot is displayed. The computational frequency is f=30.6Hz.

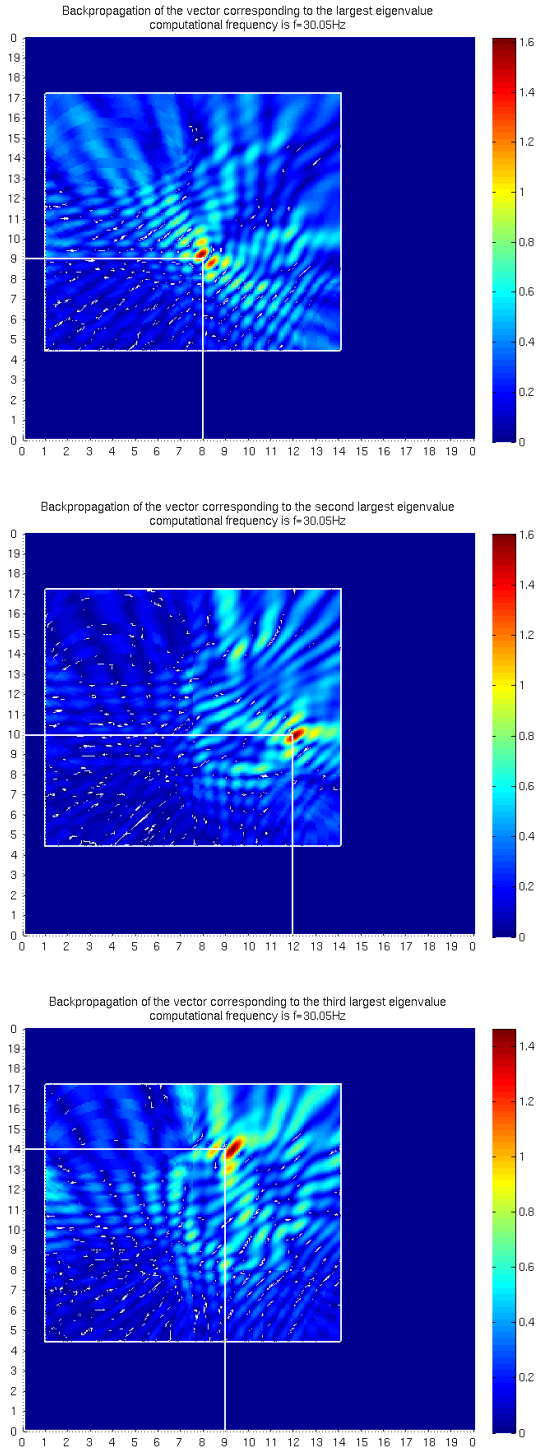

Figure 4.16: Backpropagation of the vectors corresponding to the three largest eigenvalues with data generated from the case of three scatterers as given in Figure 4.5. The computational frequency is f=30.05Hz.

4.3.3 MUSIC in case of well separated scatterers

MUSIC is now employed to the two situations of well separated targets given in Figure 4.5. The performance of this imaging algorithm is highly reduced compared with the Born cases. The images in Figure 4.17 are computed using the formula given by (2.74) in section 2.3. Remember that in the Born simulations we had to modify the formula because MUSIC achieved so well that it would be hard to spot targets without doing so. This modification is not applied here. In both scenarios we now have a much lower peak relative to bottom values and their areas are wider. MUSIC still performs very well. Locations are pinpointed almost exact and compared to backpropagation, we have more narrow and higher peaks.

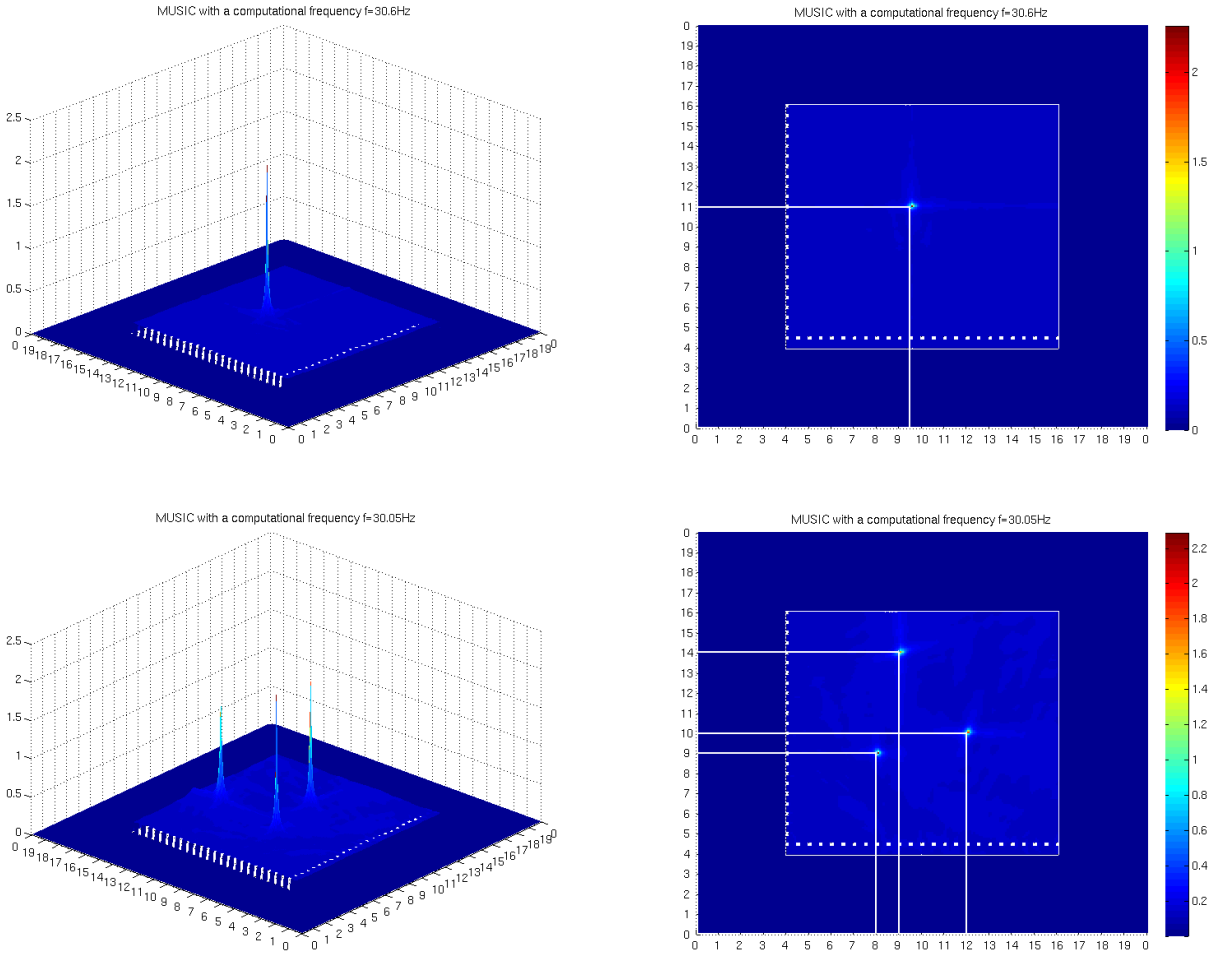

Figure 4.17: Results from employing MUSIC to the cases given in Figure 4.5, the single scatterer scenario and the scenario with three scatterers. The left images displays the results in 3D and to the right the same images are given as contour plots. The computational frequency is f=30.6Hz in the upper images and f=30.05Hz in the lower images.

4.4 Resolution and scatterers in chains using MUSIC

4.4.1 Resolution

To study resolution we will use two closely separated scatterers. It is difficult to properly define a resolution limit since the results here will vary with locations versus array geometries. We will therefore consider resolution by discussing the algorithms ability to separate two targets by their peaks. In Figure 4.18 scatterers are separated by a $\frac{\lambda_0}{2}$ and positioned perpendicular to the source array. The related eigenvalues, a $3D$ image and a zoomed contour plot from the same scenario is shown. In Figure 4.19 the same computations where performed but now with a separation of $\frac{\lambda_0}{3}$. In both cases the images are computed by assuming that the two largest eigenvalues are related to targets. Considering the values from both configurations, we find that in the latter case, it is easier to check that we have only two scatterers. But this does not result in a better resolution, in the case with $\frac{\lambda_0}{2}$ spacing, the scatterers are resolved better because values between the peaks are lower. With this data MUSIC is able to resolve two targets down to a spacing of about $\frac{\lambda_0}{3}$.

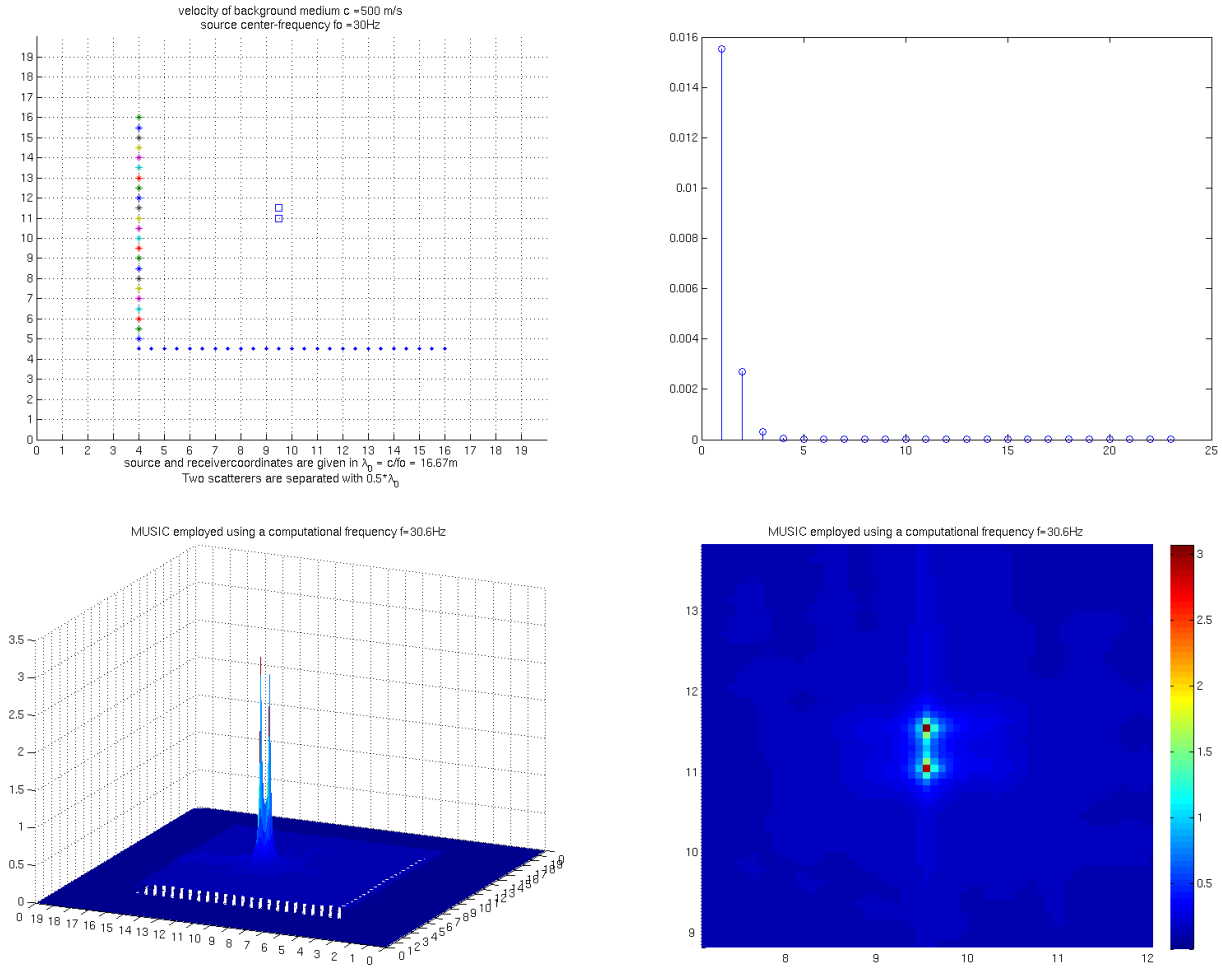

Figure 4.18: Two scatterers are present and spaced by $\frac{\lambda_0}{2}$, the system configuration is given in the upper left figure. The computational frequency is f=30.6Hz and in the upper right figure the resulting eigenvalues are displayed. Images shown in the bottom figures are the same results, to the left a $3D$ image and to the right a zoomed contour plot.

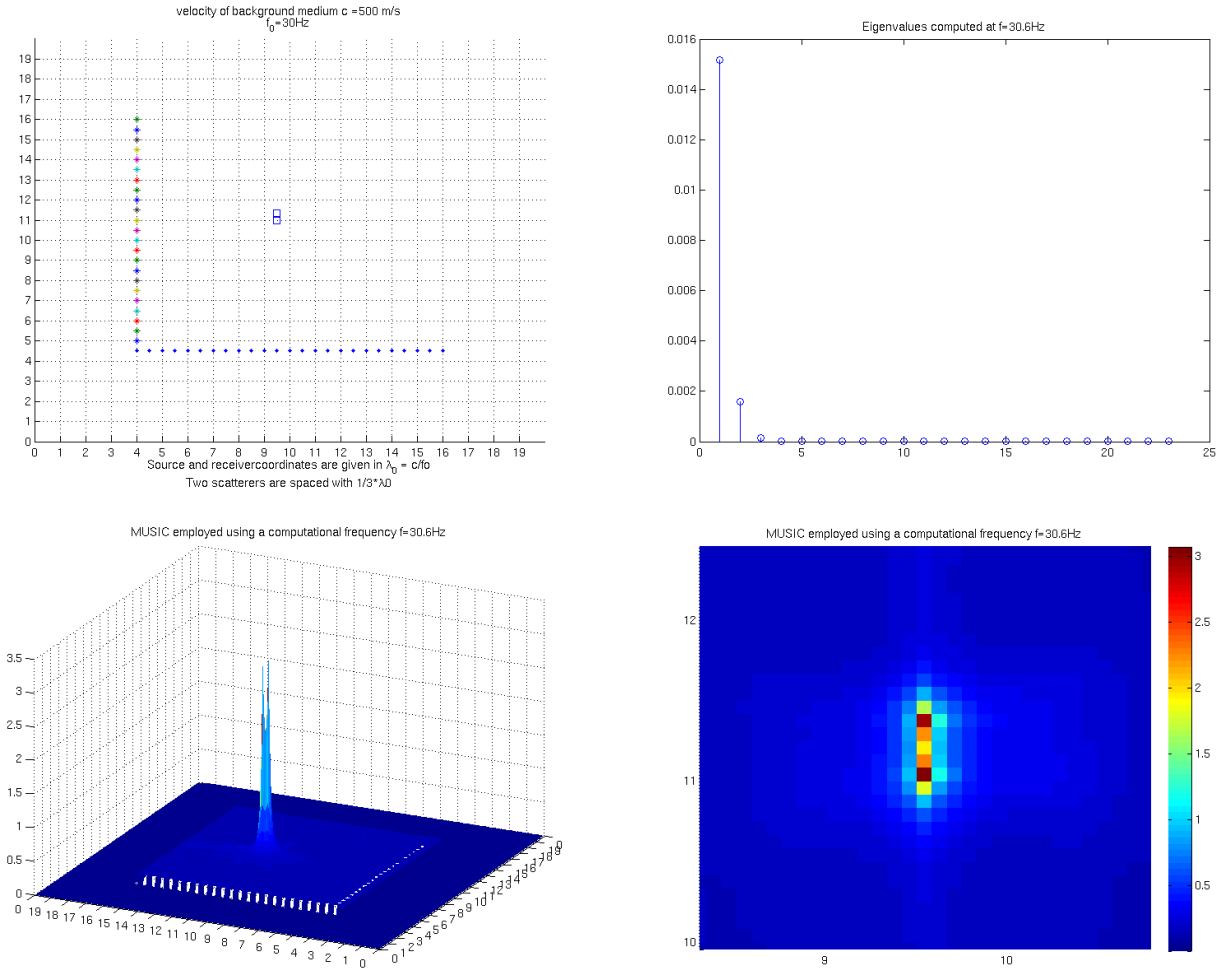

Figure 4.19: Two scatterers are present and spaced by $\frac{\lambda_0}{3}$, the system configuration is given in the upper left figure. The computational frequency is f=30.6Hz and in the upper right figure the resulting eigenvalues are displayed. Images shown in the bottom figures are the same results, to the left a $3D$ image and to the right a zoomed contour plot.

4.5 Scatterers in chains

MUSIC is now employed to cases of multiple targets in chains. Figures 4.20 to 4.22 shows experiments with an increasing number of targets perpendicular to the source array. Images computed in these figures are done by assuming the correct number of scatterers. First three scatterers are separated with $\frac{\lambda_0}{2}$, see Figure 4.20. Next we have a chain of four targets spaced with $\frac{5\lambda_0}{6}$. These cases are down towards the limit where MUSIC can no longer differentiate between present targets. Although it can be difficult to determine the number of target related eigenvalues, images displays each location. In Figure 4.22 five targets are present, the spacing is increased to $\frac{3\lambda_0}{2}$. As number scattering targets increases the spacing must be increased for MUSIC to resolve them. Since the eigenvalues can be misleading we should look at how an image can be affected if the wrong number is guessed. The scenario including three scatterers gives eigenvalues where it is impossible to determine the actual number of targets. Figure 4.23 compares the image computed by guessing three as before and an image where four eigenvalues are assumed to be target related. In the later we find that this gives gain to the peaks, especially to the middle one. Concerning the resolution it has no effect. Note that when we looked at the relation between these values and how vectors was connected to the targets we did not find a one to one relation. When several targets are closely spaced we are in a situation where assumptions from the theory is no longer valid. The biggest impact is on the eigenvalues, we now get more non-zero values and thus vectors connected to scatterers. But when we apply the same technique it still works. To explain this we must consider the fact that although vectors do not obey the orthogonal properties as in the theory. We clearly have vectors related to the zero eigenvalues that still is close to being orthogonal with the targets Green's function vectors. As result we get relatively higher peaks in scatterer locations if vectors connected to all non-zero values are excluded from the computation.

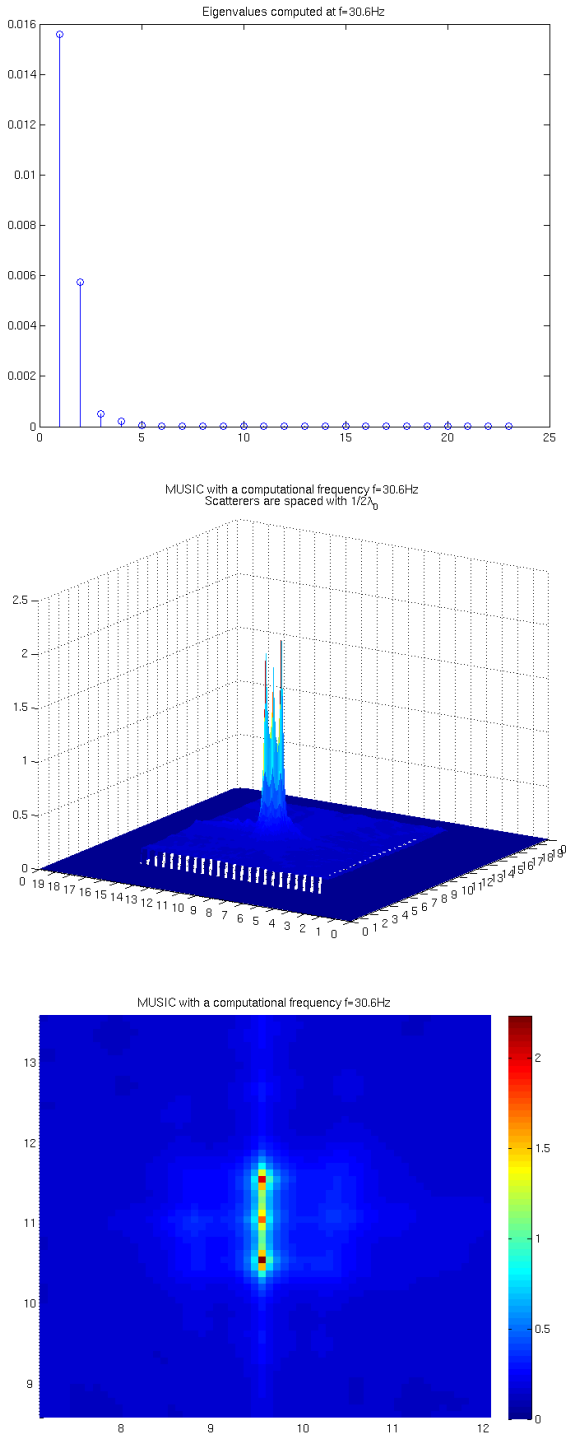

Figure 4.20: Three scatterers in chain perpendicular to the source-array. Target spacing is $\frac{\lambda_0}{2}$. Eigenvalues, a 3D image and a zoomed contourplot is displayed from top to bottom. In computations eigenvectors excluded are corresponding to the three largest eigenvalues(assuming three scatterers). A computational frequency of $f = 30.6Hz$ is used.

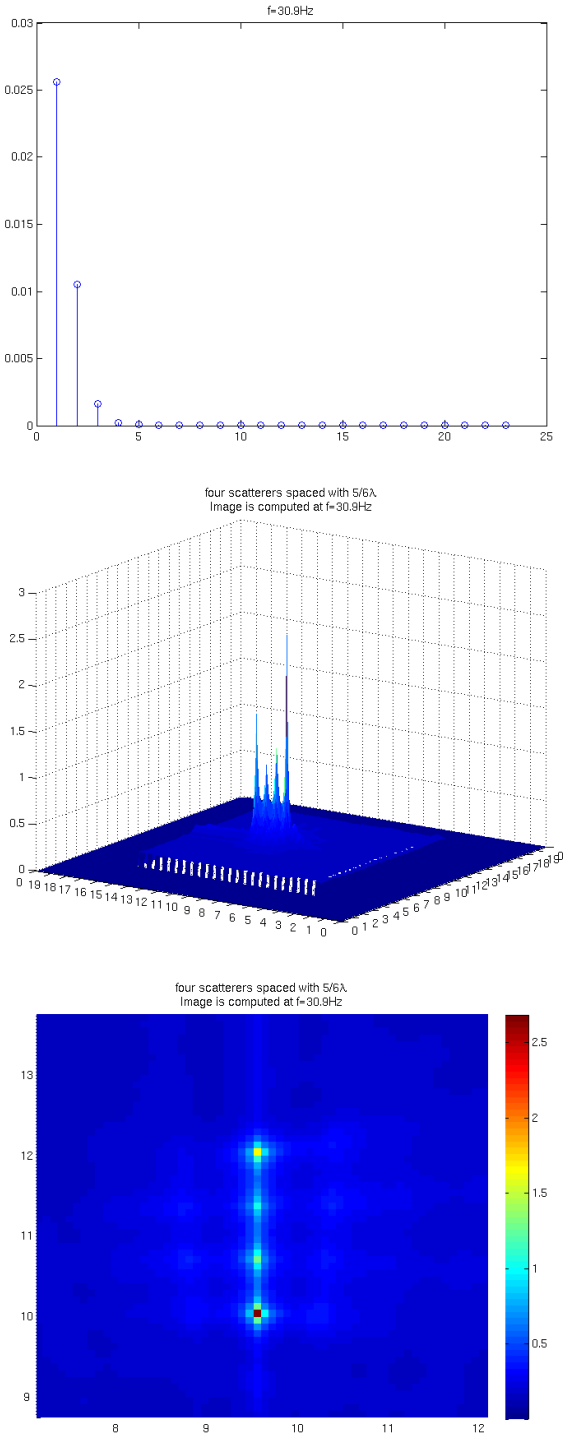

Figure 4.21: Four scatterers in chain with spacing $\frac{5}{6}\lambda_0$. MUSIC is employed by including vectors corresponding to the eigenvalues ranging from the fifth largest and out. The figures are results computed at $f = 30.9Hz$.

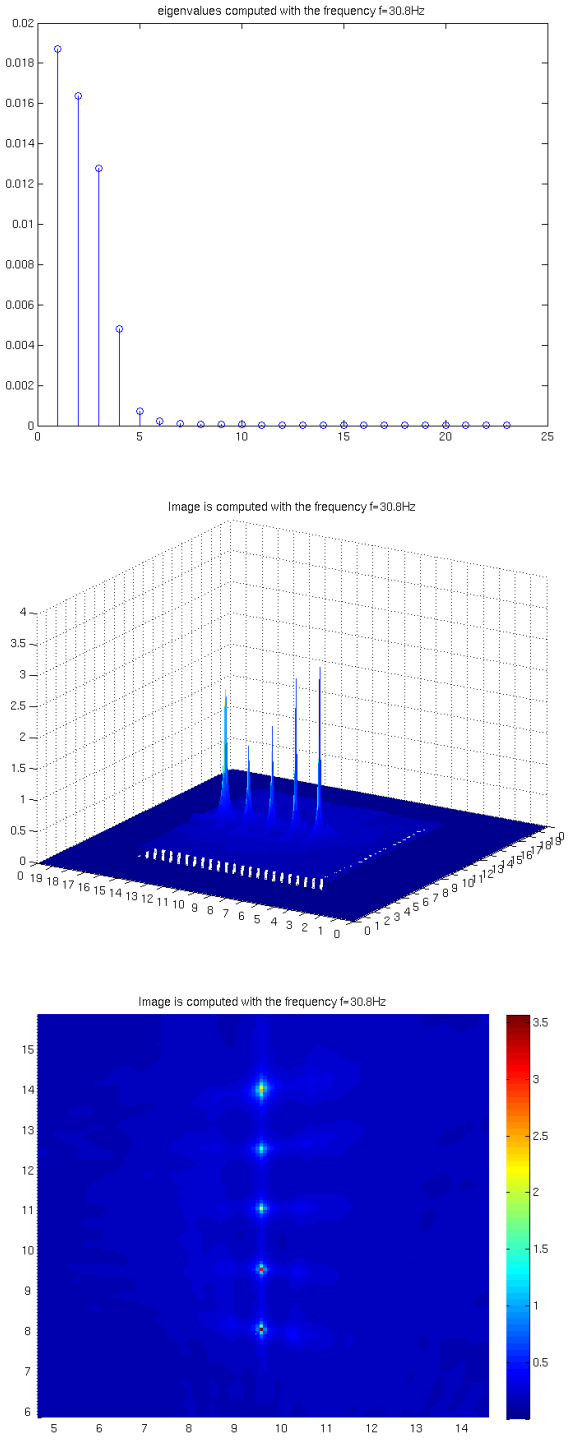

Figure 4.22: Five scatterers in chain. The target spacing is $\frac{3}{2}\lambda_0$ and the results are computed at $f = 30.8Hz$. MUSIC is employed by including vectors corresponding to the sixth largest eigenvalue and out which is an assumption of five scatterers.

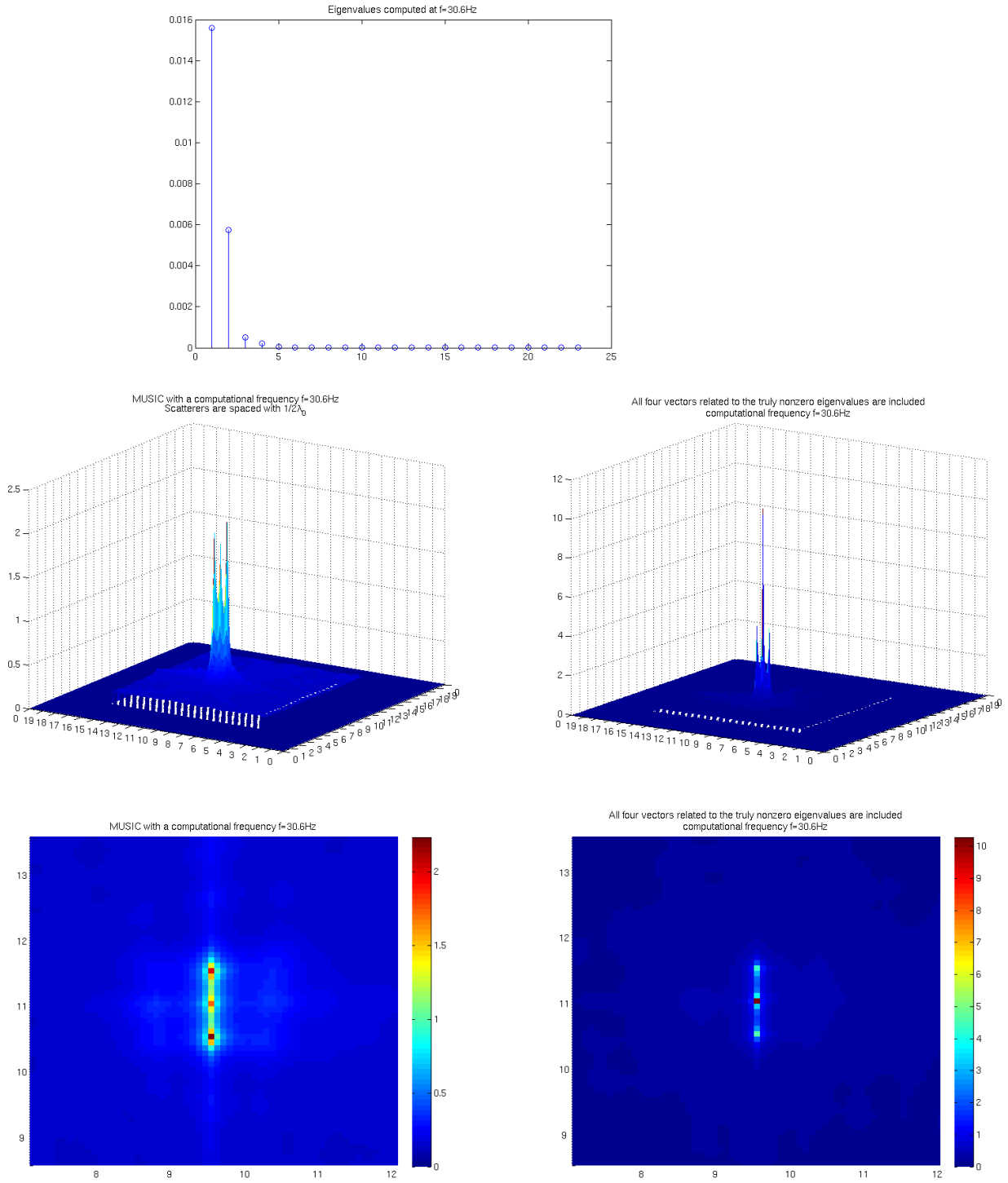

Figure 4.23: The case of three scatterers in chain with spacing $\frac{\lambda_0}{2}$. To the left the same results as in Figure 4.20 is given. It is assumed the correct number of scatterers for these. To the right the computations are done considering 4 target related eigenvalues. The computational frequency is $f = 30.6Hz$.

4.5.1 MUSIC and onesided imaging

Up til now we have been viewing images as result from using (2.74) of section 2.3.2 in a straight forward fashion. This formula includes the set of vectors related to both the source and the receiver array. In this section we consider images made by applying the same algorithm but with only one set at a time. The two expressions are then given by:

$$P_{Ms}(\vec{x}, w) = \frac{1}{\sum_{k=M+1}^{min\{L,N\}} |\vec{e}_k^T \vec{g}^s(\vec{x}, w)|} \tag{4.9}$$

$$P_{Mr}(\vec{x}, w) = \frac{1}{\sum_{k=M+1}^{min\{L,N\}} |\vec{v}_k^T \vec{g}^r(\vec{x}, w)|} \tag{4.10}$$

P_{Ms} in (4.9) gives the image when considering vectors belonging to the source array and P_{Mr} in (4.10) gives the image from the receiver array. These are tested on the case of three well spaced targets and the situation of three scatterers in chain. Layouts are shown in the Figures 4.5 and 4.20. In all computations the number of eigenvectors assumed to be related to scatterers is set equal to the actual number of the present ones. Results from the situation with well spaced targets are displayed in Figure 4.24. These are compared with the previous image, when both set of vectors was used at the same time. These are given in the upper images. The following images are onesided computations, the two middle figures shows the result from employing (4.9) and in the last two (4.10) is used. When evaluating these we find that with onesided imaging we get stronger peaks, especially in the location furthest away from the array whose vectors are being used. There are some stretch in the peak areas perpendicular the arrays but these have low values relative to the tops. In Figure 4.25 the same results as before are given but viewed only in directions perpendicular to the arrays. The receiver sided MUSIC gives the highest gain where the lowest peak reaches 4.5 and the highest about 17. In the two sided we have 1.5 and 2.3 respectively. Considering these figures we should notice that in the onesided images the accuracy is better in the direction corresponding to the set of vectors used and reduced in the other. This is expected since the array vectors are projected from one array only. We now end this section by studying the outcome of onsided MUSIC on a situation where the twosided fails to resolve targets. Three scatterers are separated equally by $\frac{\lambda_0}{3}$, they are positioned perpendicular to the receiver array. Results are given in Figure 4.26. Here we find that employing the twosided and the receiversided MUSIC performs better in narrowing down the peak areas. But they can not resolve the targets. The source sided computation shows a number of peaks corresponding to the number of scatterers but the position of the middle target is misplaced. By evaluating the images from all three techniques we have gained more precise information about positions and number of targets.

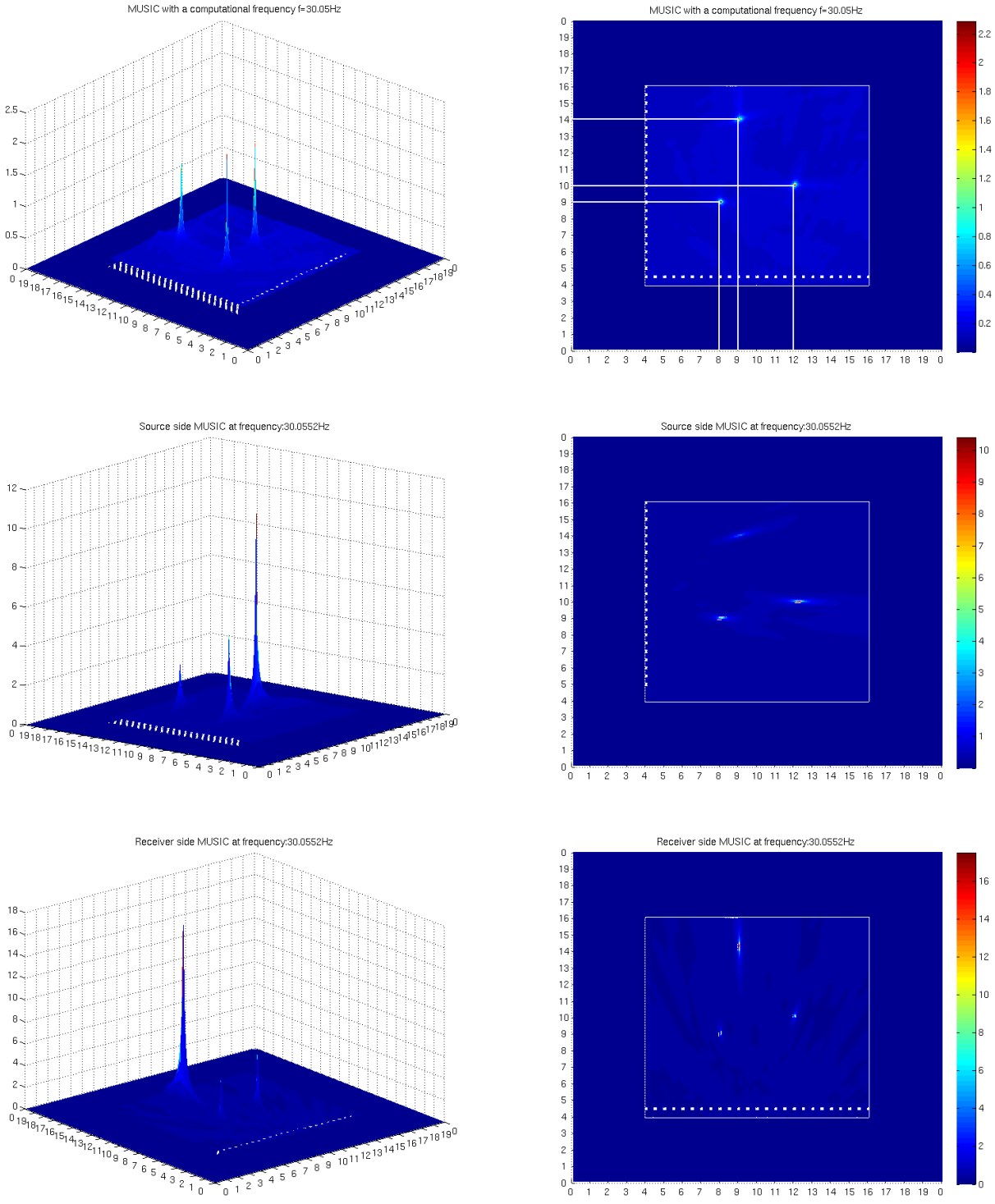

Figure 4.24: Comparison of images computed as both two sided and one sided projections. Images are compared when data from the case of three scatterers given in Figure 4.5 is used. The upper images shows a 3D and a contour plot of the result when vectors related to both the receiver array and the source array are included (shown previously). The following images are computed by using only the source related vectors(middle) and the receiver related vectors(bottom). The computational frequency is $f = 30.05Hz$.

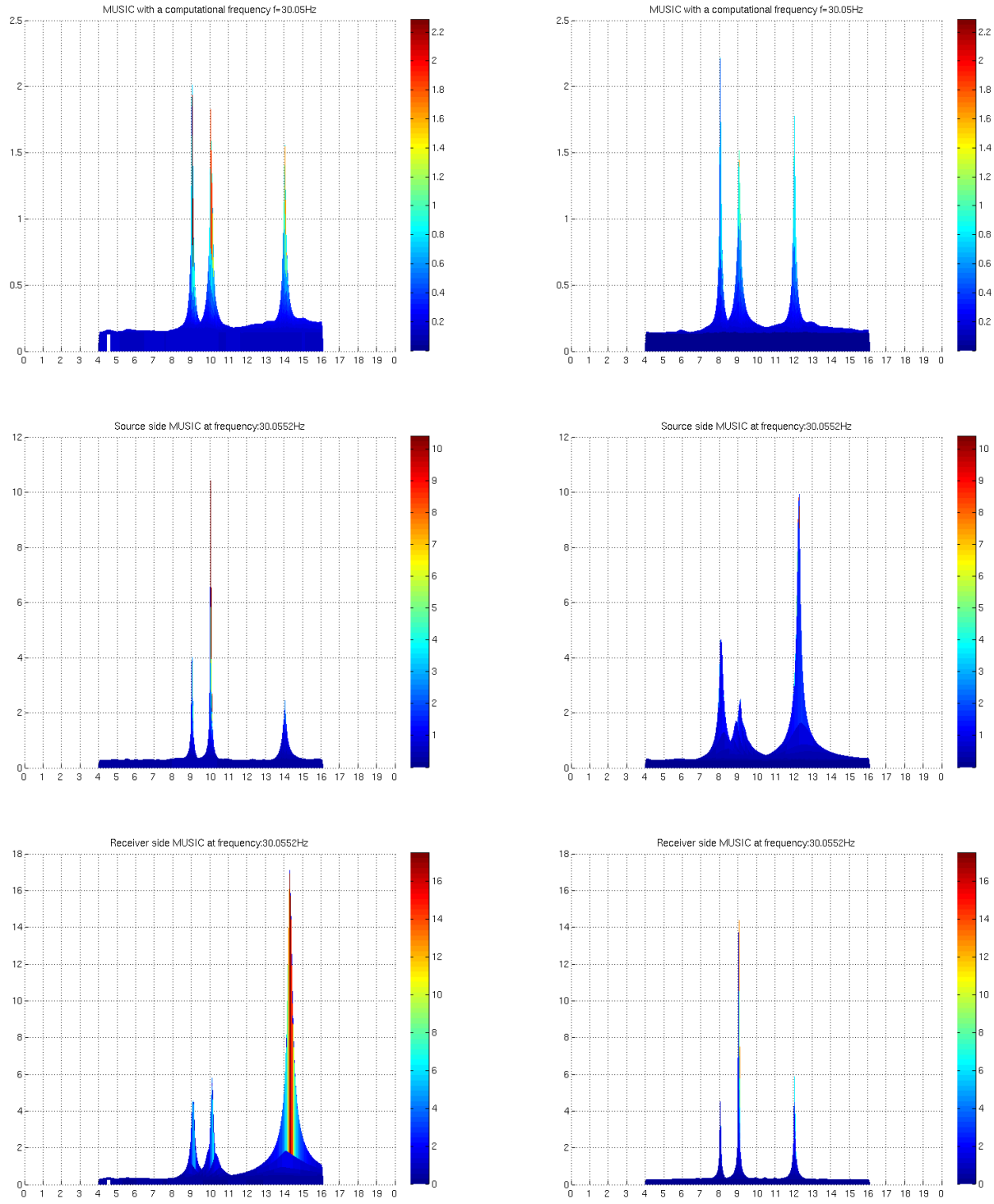

Figure 4.25: The figures displays the same results as in Figure 4.24. Here images are displayed by a view from the source array(left) and as looking down the receiver array(right).

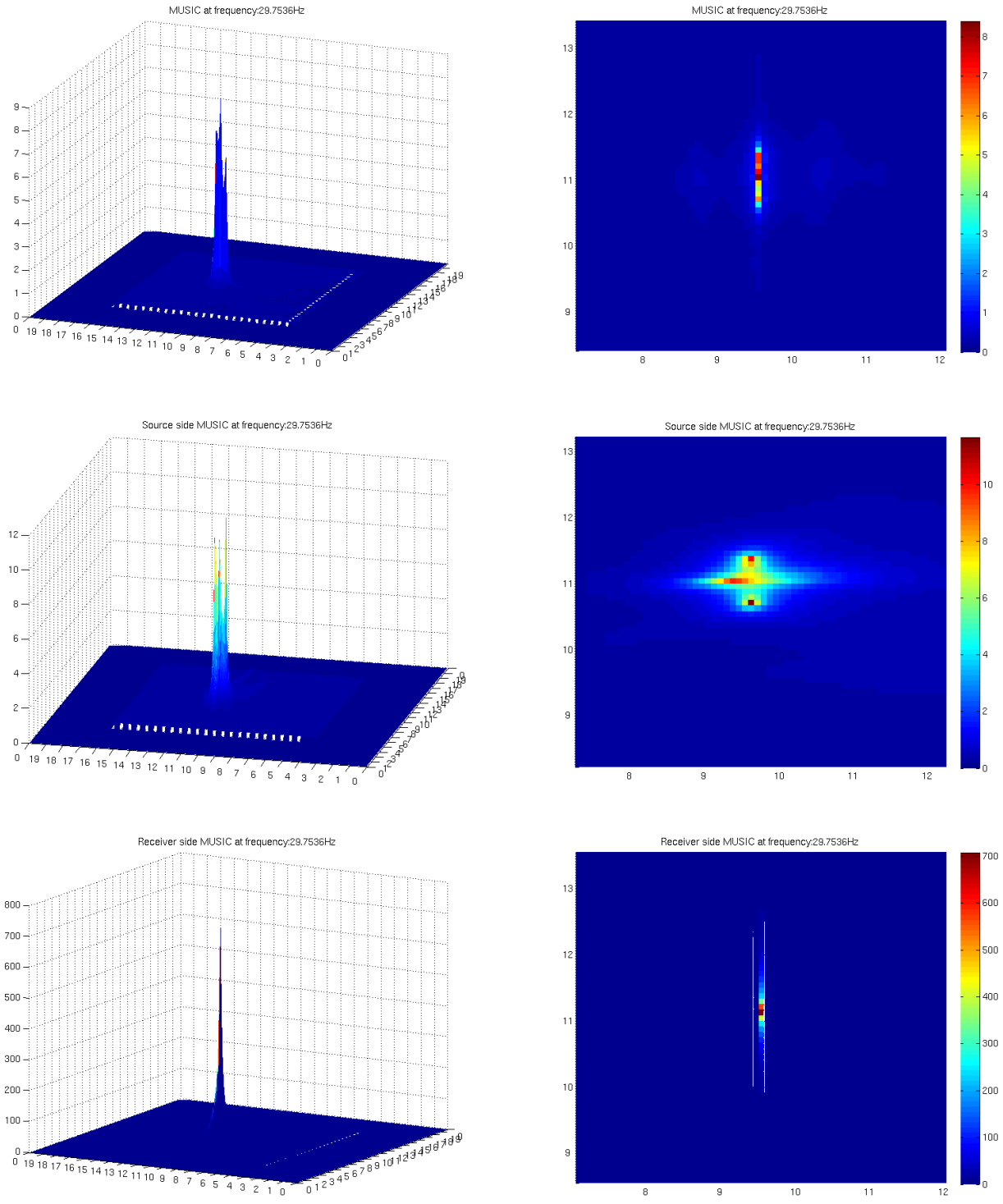

Figure 4.26: Three scatterers are located perpendicular to the source array. The spacing is $\frac{\lambda_0}{3}$. Images are 3D images(left) and zoomed contour plots(right) of MUSIC by using vectors related to both the receiver and the source array simultaneously, the source array only and the receiver array only. Results are given respectively from top to bottom. The images are computed with the frequency $f = 29.8Hz$.

4.6 MUSIC employed in a noisy multiple scattering environment

We have seen how MUSIC is able to give precise information on target locations if there are no external interferences in propagating waves. It was employed on data generated such that recordings only contained the scattered signals. Now we test how MUSIC is able to suppress noise in the recorded data. White Gaussian noise is added to the output signals. A level of low and a level of moderate signal to noise ratio (SNR) is treated. On each output a temporal signal containing only Gaussian noise is added. The signal noise is computed as followed:
The time dependent signal energy is found by:

$$e(t) = \frac{r(t)^2}{T} \tag{4.11}$$

The energy is considered over the period T starting from when measurements first has a non-zero amplitudes[2]. T is set equal to one period of the input signal (about $65ms$, see Figure 3.18 in section 3.2.2). Then the Gaussian distribution $N(0, \sigma)$ is generated individually for all outputs. Zero gives the expected value and σ the standard deviation and is dependent of the SNR in the following manner:

$$\sigma^2 = \frac{e(t)}{SNR} \tag{4.12}$$

Examples of output signals with noise are shown in Figure 4.27. As usual we first consider the well resolved case. When this is done we look at how the resolution is affected.

[2]When scatterers are well spaced the signals might contain more energy than accounted for in the period T. But since we divide by the the period, signals that have shorter duration of non-zero amplitudes gets reduced energy when T is increased.

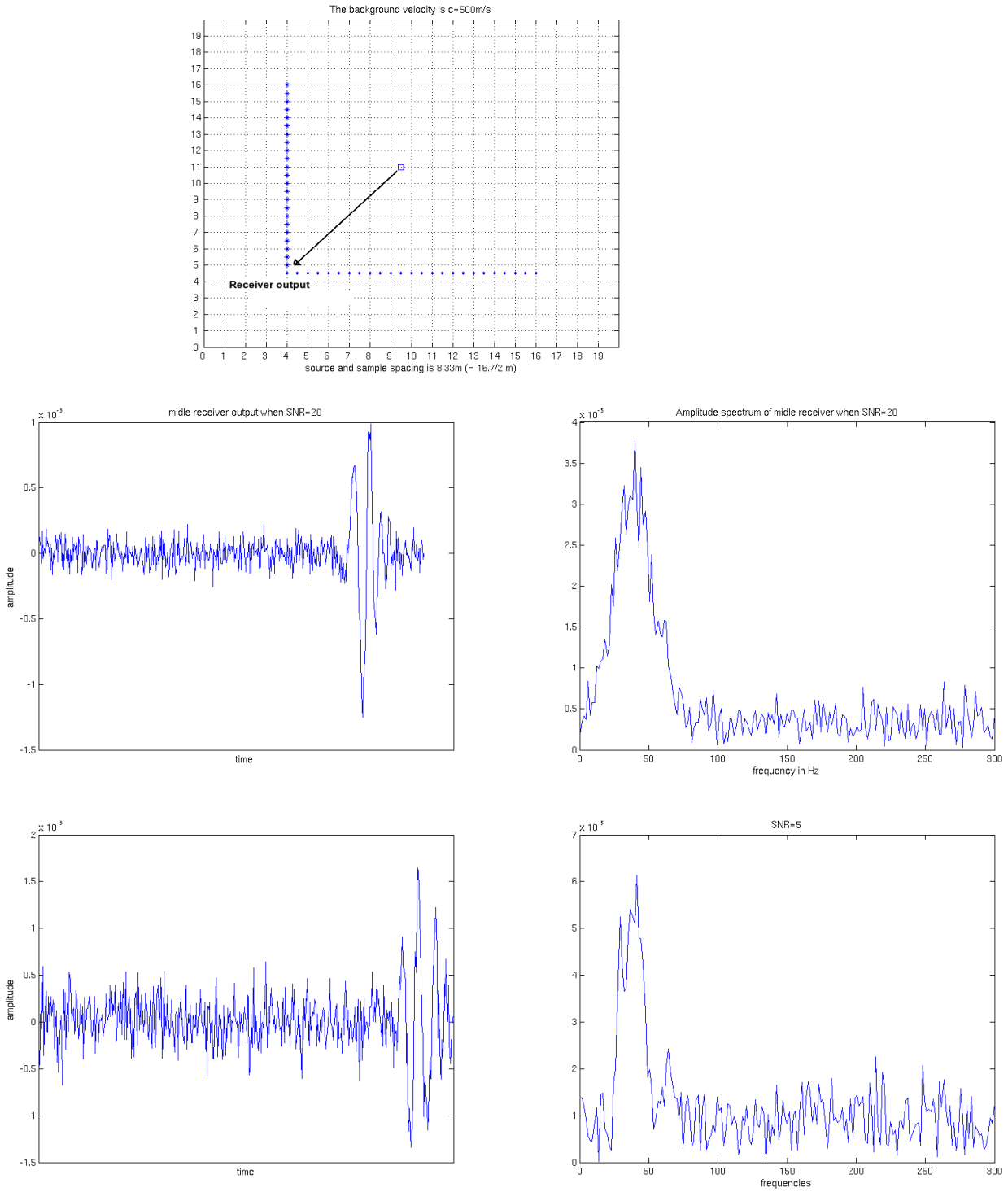

Figure 4.27: The images shows two receiver element outputs, both their measured signals(right) and their amplitude spectrums(right) are shown. These signals are detected in the left most receivers with white Gaussian noise added. In the upper image the $SNR = 20$ and in the bottom image this ratio is given with a $SNR = 5$.

4.6.1 Well separated scatterers

Eigenvalues and images from employing MUSIC to the case with three well separated scatterers (Figure 4.5) are given in Figures 4.28 and 4.29. In the first set of figures there is added a moderate level of noise where the $SNR = 20$ and in the latter the $SNR = 5$. These are compared with results where there are no noise in the outputs. Consider the eigenvalues from both set of figures, the relation and sizes of the three largest values are maintained. As the noise ratio is increased we get higher values in more of those who ideally should be zero. But there still is a strong indication of three targets being present and the images are computed by using the vectors related to these. Images shows a reduction in peaks depending on the SNR. As this reduces the peaks in target locations are reduced. As result the contour images shows wider peak areas. But these effects are small and the algorithms locational properties remains good.

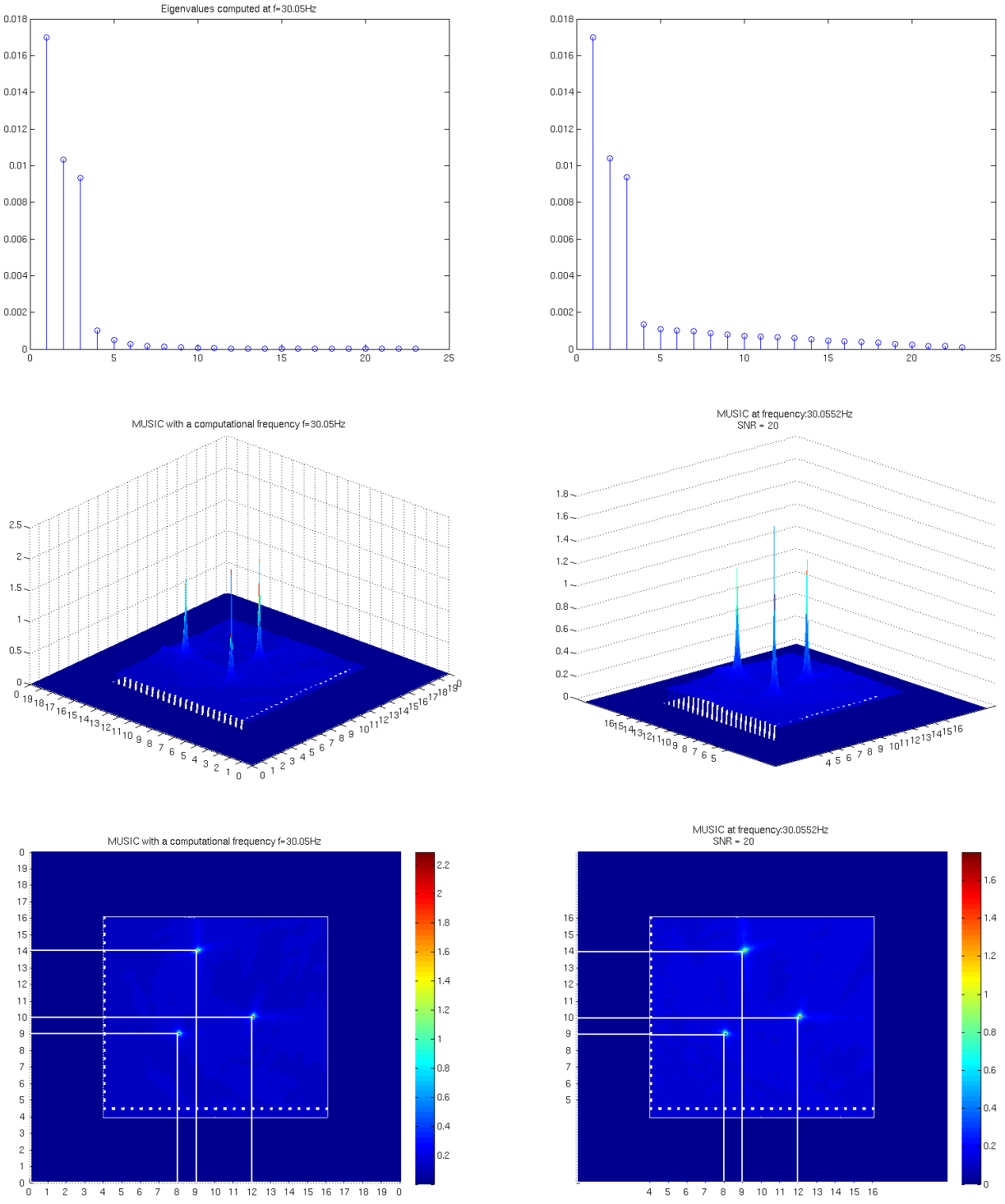

Figure 4.28: Results here are computed with data from the well resolved case of three scatterers given in Figure 4.5. The left side images are computed without any noise present. On the right side images are created with noise added to the receiver elements where $SNR = 20$. All results are computed at the frequency $f = 30.05Hz$.

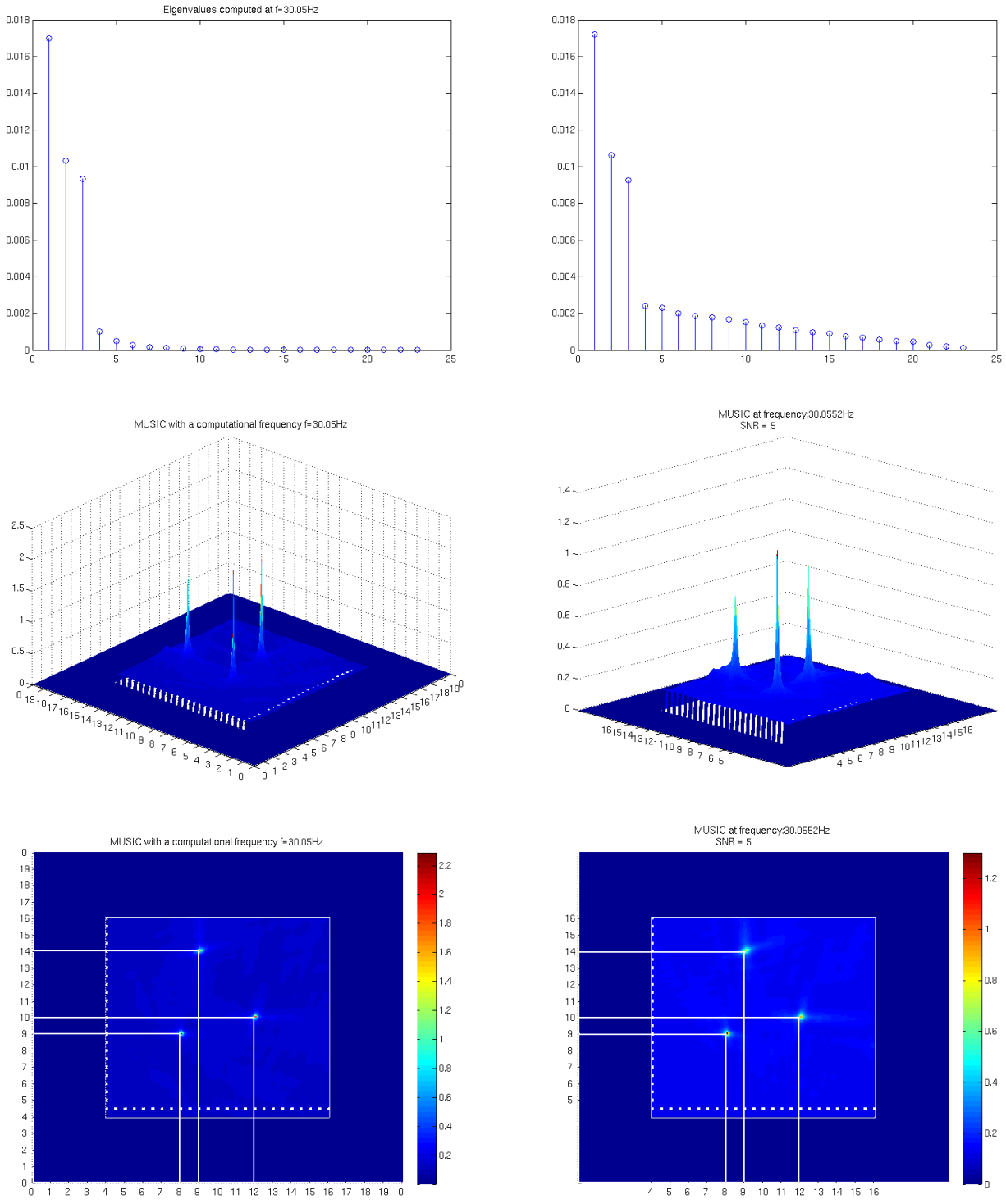

Figure 4.29: Images shows analogous results as in Figure 4.28 but now the $SNR = 5$.

4.6.2 MUSIC and resolution

To study the robustness towards noise in MUSIC's ability to resolve two scatterers, consider Figures 4.30 and 4.31. The targets are separated by $\frac{\lambda_0}{2}$ and the algorithm is employed by using the eigenvectors corresponding to the two largest values. In the previous section we saw that the image values in target locations became wider when influenced by noise. This should reduce the resolution but as shown in the images we are still able to locate two targets for both levels of SNR. As in the case of well spaced scatterers we have a reduction of peaks and in the image where $SNR = 5$ (Figure 4.31) it is clearly displayed how the areas are widened. Considering the eigenvalues here, it is impossible to determine the number of targets. In Figure 4.32 MUSIC is used by excluding vectors related to the 20 largest eigenvalues and results are compared to the previous images (two targets assumed). It gives divided results, we get higher peaks over a more narrow area, but the image value between the targets has increased relative to the tops. But still, the image from using vectors related to all non-zero eigenvalues can be considered to give the better image. This was also discussed at the end of section 4.4.1 without noise. The performance of MUSIC reduces as the noise ratio increases. But this reduction is low enough so that we may indeed consider the algorithm to be robust.

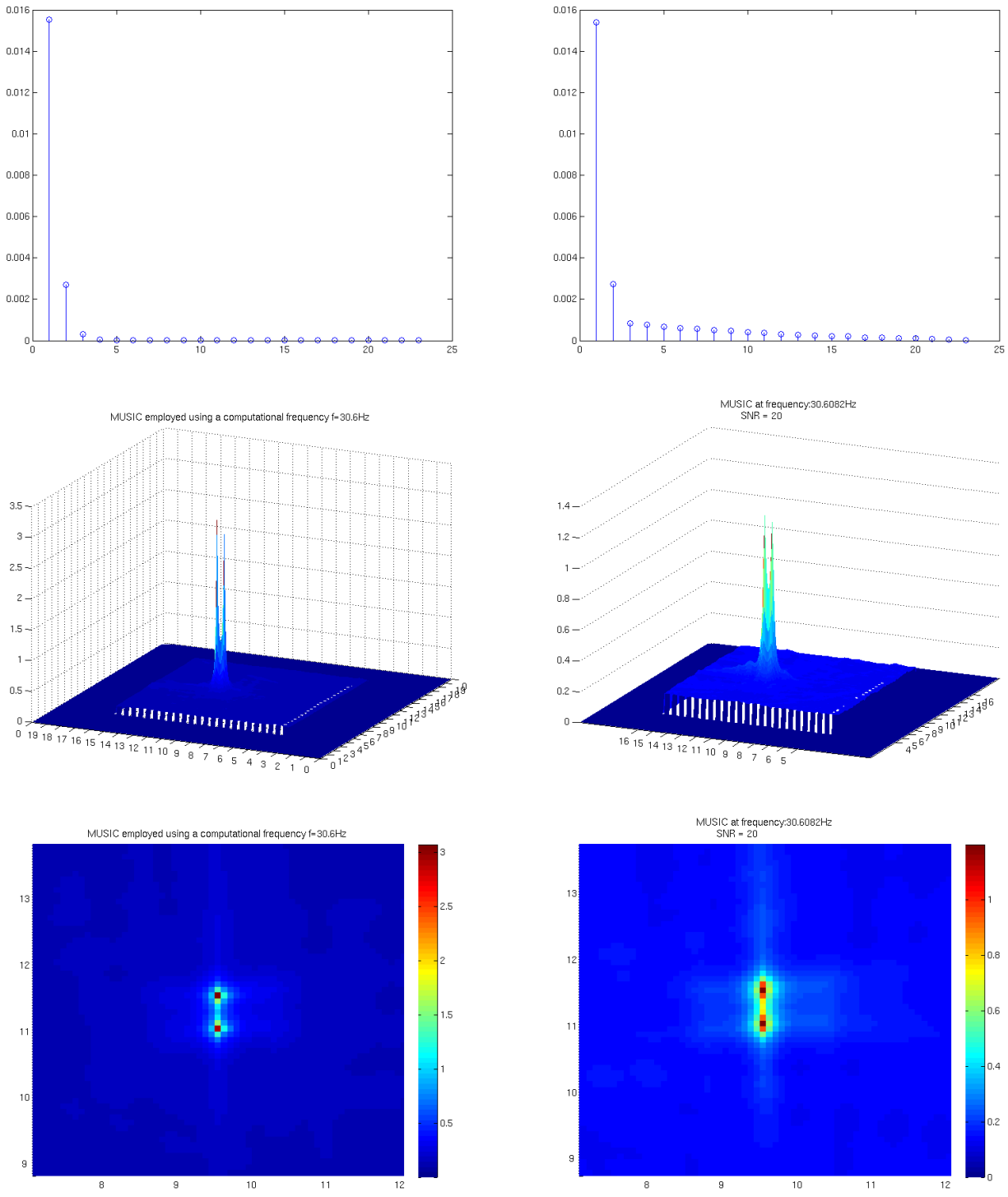

Figure 4.30: Images on the left side shows results from using MUSIC in the situation where two scatterers are present spaced by $\frac{\lambda}{2}$. To the right MUSIC is employed to the same configuration but there is noise added on the recorded signals. The $SNR = 20$ and all computations are performed at $f = 30.6Hz$ by using the vectors related to the two largest eigenvalues.

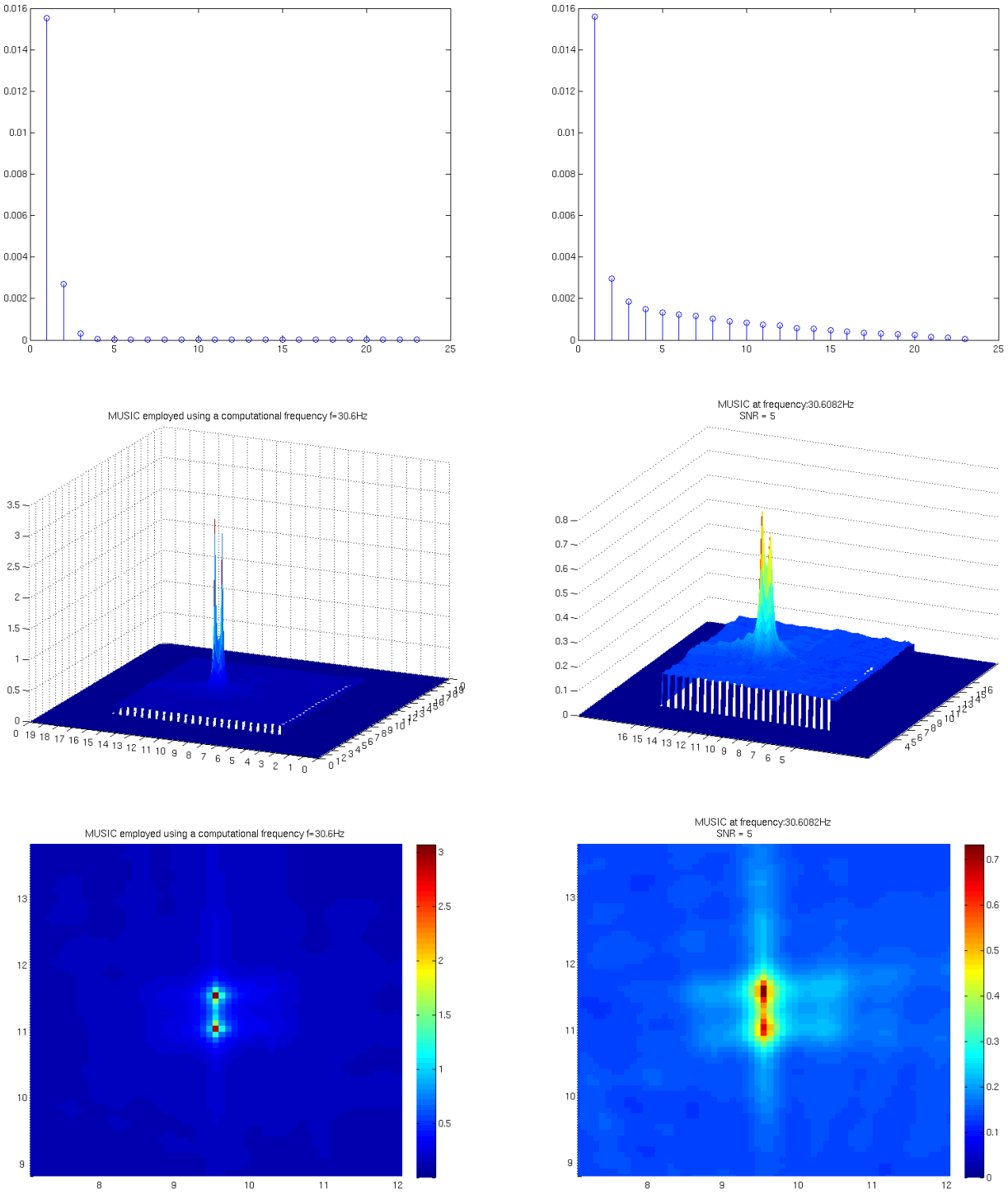

Figure 4.31: Images displays analogous results as in Figure 4.30 but here the noiseless results are compared to the results when $SNR = 5$.

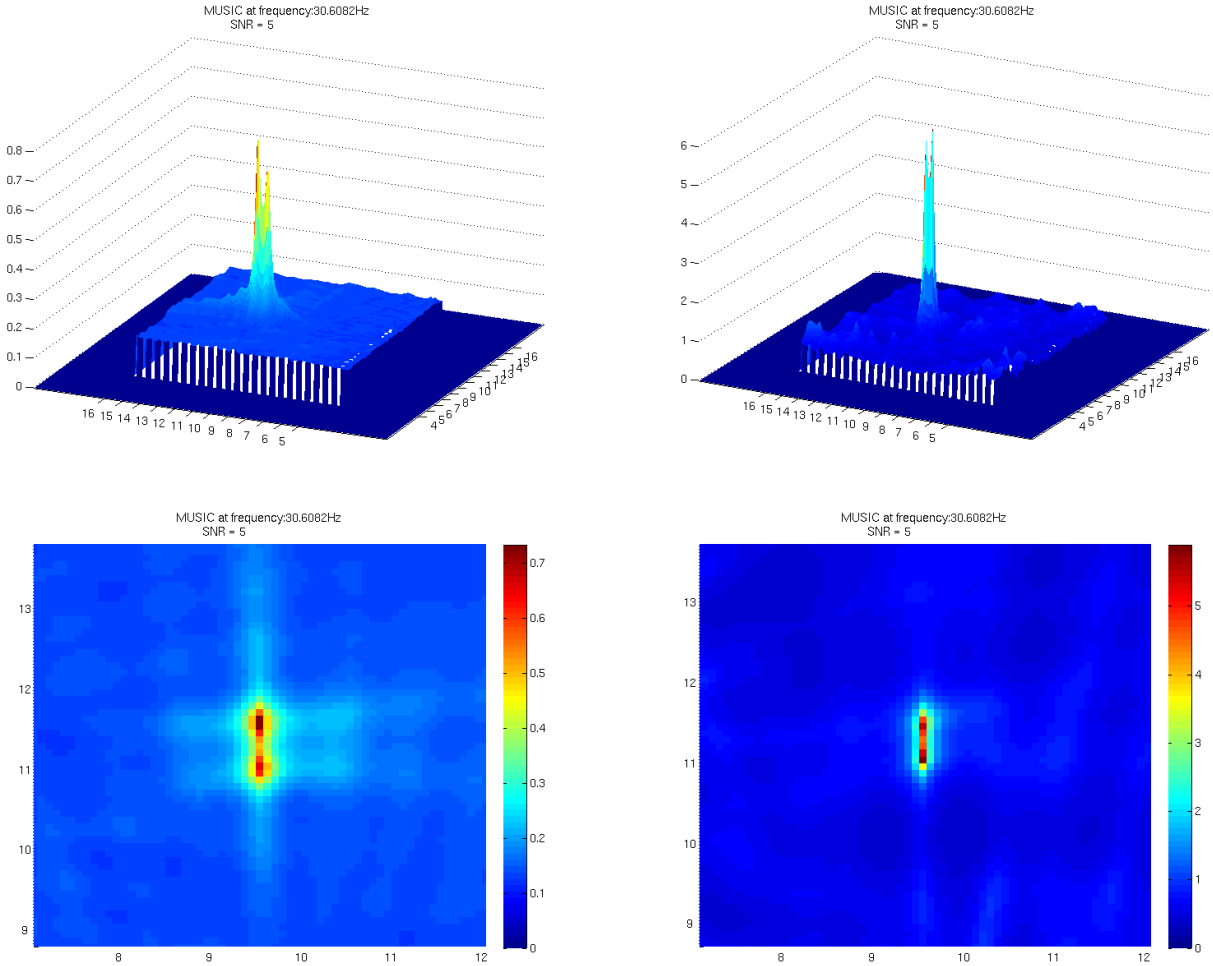

Figure 4.32: All images are computed with $SNR = 5$, images on the right side are computed by using MUSIC and excluding vectors related to several of the non zero eigenvalues, 20 vectors are excluded. The left images are the same as in Figure 4.31 (assuming two scatterers).

Chapter 5

Summary and conclusions

5.1 Summary

The problem statements in short terms was to make images of reflecting objects buried in a medium. It was limited to the case of point scatterers in a homogeneous background. This was done in a $2D$ model by considering data as if acoustic waves where emitted and detected in a bore-hole geometry. Source excitations and measurements was treated under the theory of time-reversal. In doing so we could represent this data in a system containing information about the scatterers, this system includes the important time-reversal matrix. By performing an eigenvalue decomposition we where able to derive the two imaging algorithms, MUSIC and backpropagation. Both are dependent of eigenvectors, related to ideally non-zero eigenvalues, being linked to the scattering targets. They also rely on target related Green's function vectors to be close to eigenvectors and the fact that such vectors are orthogonal. Where the backpropagation algorithm uses this directly through vectors belonging to non-zero values. MUSIC on the other hand, excludes the same vectors. It exploits that eigenvectors related to zero eigenvalues still should be close to orthogonal to Green's function vectors in target locations. When we walked through the theory it was made a somewhat extreme assumption of the ability to perfectly resolve scatterers. It is possible to come close to this situation, but in experiments it is hard to fulfill. We have seen that it is dependent on the range of the data-acquisition and the available frequencies in the detected signals. The techniques where first tested on situations where the theory should valid to a certain extent. We used a strict Born model to generate data from targets that where well separated. Since MUSIC seemed as the better choice when making images in a model as herein, we have mainly concentrated on this technique. Both algorithms was employed to data generated by simulating propagating waves. The simulation was done according to a forward finite difference scheme, which

is derived from a discrete version of the $2D$ wave equation. MUSIC was tested thoroughly by studying resolution and cases of closely spaced scatterers in chains. We tried both one and two sided imaging. Two sided is using the set of vectors related to both the receiver and the source array at the same time. One sided is applying only one set to make one image. Finally we checked MUSIC's robustness towards noise on the detected output channels. A low and moderate level of noise was considered.

5.2 Conclusions

When making images of point like targets we seek to obtain strong peaks where scatterers are located and relatively low image values away from their positions. The locational performance depend on the peaks to be located exactly over a narrow area. We found that making images with the two techniques can be successful especially when targets are well away from each other. When employing them to Born data we got good results from both algorithms. Especially from MUSIC, it achieved so well that a distorting value needed to be included. In images from backpropagations we got some small inaccuracies in the target locations. One reason for this was that we had one set of eigenvectors related to the source array and another to the receiver array. With the data available they couldn't locate the targets at the exact same positions. This discrepancy is related to the range of the arrays and can be improved if they are extended. However we are interested in comparing the two techniques also by considering the performance with limited data. So the extended array scenario was not treated. In the case with Born data we checked if images made by the backpropagation could be improved by using multiple frequencies. Compared to ideal multiple frequency imaging this failed. Again, a part of the problem is related to the arrays locational differences. But from the work done in here we can not deduce that we are guaranteed a lot of improvement with larger arrays. Generating data with the $2D$ wave modeling programs was done in a manner such that simulations came close to the Born model. Both algorithms where tested on cases with well separated scatterers. The backpropagation technique gave very similar results as in the case of strict Born data. MUSIC's performance was highly reduced compared to the Born simulations. But taking in to account that it worked extremely well in those cases, we still got very good results. It's locational properties was maintained but with small extensions in peak areas. The biggest impact was on the peak height relative to bottom values, these where strongly reduced. If we compare MUSIC and backpropagation we can deduce an important result. With limited data we can expect to get a much better performance by employing MUSIC. But when assuming this, we must remember that MUSIC was more affected in the transition from Born data. So

we only know that it is the leading algorithm in situations close to the Born model. The main reason that MUSIC performed better than backpropagation must be related to the eigenvectors of the time-reversal matrix, and their correspondence to Green's function vectors. Where eigenvectors belonging to the zero eigenvalues are closer to being orthogonal to the targets Green's function vectors. Further testing of MUSIC was done in cases of scatterers being closely spaced. With two targets we saw that the resolution came down to about $\frac{\lambda_0}{3}$. Where λ_0 is the wave length according to the center frequency of the modeled waves. As the number of scattering targets increased the resolution decreased. By employing the one sided versions we could get improved images. Higher peaks and narrowed areas was the results. But we also got some displacement so a one sided computation should be seen in addition to results from the original algorithm (two sided). MUSIC has also proven to be robust towards noise on the output signals. We did get a reduction of the resolution, but with the $SNR = 5$ (signal to noise ratio) it was able to resolve two scatterers separated by $\frac{\lambda_0}{2}$. The biggest impact from noise in outputs where on the eigenvalues. In cases of well resolved targets the ideally zero values where no longer so close to zero. The target related values where relatively much higher so it wasn't difficult to guess the correct number of scatterers present. This goes for both noise ratio's, $SNR = 20$ and $SNR = 5$. In the situation when targets were closely separated, it was more of a problem to determine the correct number by the eigenvalues. To check if this is a problem, we considered the case with two targets spaced by $\frac{\lambda_0}{2}$ and the $SNR = 5$. We computed an image where we excluded vectors corresponding to all non-zero eigenvalues. What we got was an image with higher peaks over a narrowed area. It became a little difficult to separate them but it was still a more accurate image.

5.3 Future work

5.3.1 Inhomogeneous background

For simplicity we limited the problem to a homogeneous background. It is easily extended to the case of a medium with smooth velocity changes. The only difference will be made in the Green's function model. In an homogeneous background we have a constant velocity and it is thus valid in a very restricted area. Most mediums are inhomogeneous with a varying velocity profile. This case was considered until the treatment of point scattering perturbations (section 2.1). We then considered the profile to be constant except for some non-smooth transitions. Which in turn resulted in a discription of the point scatterers. We could just as easily included an inhomogeneous background with a smoothly varying velocity.

5.3.2 Extended scatterers

The array geometries and signal parameters used herein relates to those used in seismic imaging. In this area it is of special interest to know about reflecting segments. We have studied the results from closely spaced scatterers and know that if they are close enough the images from MUSIC will display them as lines. But the theory does not include such scenarios. The segments should be considered as perturbed areas. To obtain such a description, the targets must be considered to be more than just point like objects. The theory from section 2.1 and throughout Chapter 2 then needs to be modified. The challenge here is to obtain a transfermatrix in case of extended objects.

5.3.3 Modeling

The choice of two spatial dimensions was made because it reduces the processing time dramatically when computing images. Much of the challenge with this extension lies in how images should be presented. Because of similarities in all aspects of the theory it can be expected that results will reflects those in this thesis. It is needed little effort to extend the formulations in Chapter 2 and 3 to include three spatial dimensions. The modeling program used here is built upon an explicit scheme. To be ensured stability in computations this scheme relates time and spatial computations through a constant value, see (3.11) and (3.12) in section 3.1. Using implicit schemes often leads to numerical solutions which avoid such stability demands. Without a stability demand we get more fredom in model parameters. It is also expected that an implicit scheme would reduce the processing time because it reduces to solving a matrix system. Matrix solvers are fast and well developed. This is recommended if a $3D$ model is used to simulate data.

Chapter 6

Appendix

The following programs contains the matlab code used in chapter 3 and 4. Each section is a package of program files to perform a stand alone computation. Some files have the same or almost the same content and might appear in more than one programpackage but only once in each package. All packages makes use of the same programs for reading data from files and drawing system configurations, these are included in an individual section. Examples of data file formats are given in the last section. Some parameters are not read from file and is changed manulay in the program code. These may not reflect those given in the text.

6.1 Program package for simulating propagating waves and signal parameters in chapter 3

```
function [f] = program()
% Plots inputsignal and its amplitude-spectrum
% Plot reciever line outputs and its amplitude-spectrum.

[Rec,xrec,yrec,dt,fo,Nt,analytic_source,dx, velMatrix_B,c,Nx,Ny] = solve2D;
source = solve2Dsource(velMatrix_B,c,fo,Nt,Nx,Ny);

Nr=length(Rec(:,1));
M=ceil(Nr/2);
space = max(Rec(M,:));

%time in ms
t=linspace(0,dt*Nt*1000,Nt);

% source on input-chanels
```

```
figure
plot(t,source);
xlabel('time in ms');
title('Source-pulse on input-chanels');

%amplitude-spectrums
[S,f_s]=myFFT(source,Nt,dt);
Nl=floor(length(f_s)/2);

figure
plot(f_s(1:Nl),abs(S(1:Nl)));
xlabel('frequency (Hz)')
title('Amplitude-spectrum of Source-pulses');
set(gca,'XTick',0:30:300)
grid on

figure
hold
for i=1:Nr-1
    plot(t(1:Nt),Rec(i,:)+2*i*space);
end

set(gca,'YTick',xrec);

text1 = 'reciever-outputs';
title('Reciever-line output');
ylabel(text1);
xlabel('time in ms')
grid on

%DFT of outputs
for i=1:Nr
[R_fft,f]=myFFT(Rec,Nt,dt);
end

figure
hold
%plot reciever-line amplitudespectrum
plot(f(1:Nl),abs(R_fft(1,1:Nl)));
title('Amplitude-spectrum of side receivers');
xlabel('frequency (Hz)')
set(gca,'XTick',0:30:300)
grid on
```

```
figure
hold
%plot reciever-line amplitudespectrum
plot(f(1:Nl),abs(R_fft(Nr,1:Nl)));
title('Amplitude-spectrum of left receiver');
xlabel('frequency (Hz)')
set(gca,'XTick',0:30:300)
grid on

figure
hold
%plot reciever-line amplitudespectrum
plot(f(1:Nl),abs(R_fft(M,1:Nl)));
title('Amplitude-spectrum of middle receiver');
xlabel('frequency (Hz)')
set(gca,'XTick',0:30:300)
grid on

function [Rec,xrec,yrec,dt,fo,Nt,analytic_source,dx, velMatrix_B,c,Nx,Ny] = solve2D

%{
Rec is a mtrix with with measured data of the scattered field.
One column for each timestep.
%}

% Values to be read from file:
%{
f0           center frequency
Nx           # of x gridpoints
Ny           # of y gridpoints

Nsnap        # of snapshots
Nt           # of time samples
twindow      window of timesteps when the scattered field is measured
xrec,yrec    vectors with arraymeasure-coordinates(stepvalues).

velMatrix_B     Matrix with mediumvelocity (Backgroundmedium)
velMatrix_P     Matrix with mediumvelocity and pertubations
sourceMatrix Matrix with source-coordinates

%}
```

```
paramfile = 'parameters.txt'; %file with parameterdata
velfile = 'velocityData.txt'; %file with velocitydata of background
sourcefile = 'sourceData.txt';
pertfile = 'scatterData.txt';
resfile = 'recData.txt';

[fo,Nx, Ny, snapshots, Nt] = readParameters(paramfile);
[xp,yp] = readPertCoords(pertfile); %pertubation-coordinates(numer of steps)
[velMatrix_B,velMatrix_P,c,cmax] = readVel(velfile,Nx,Ny,xp,yp);
sourceMatrix = readSource(sourcefile,Nx,Ny); %locations of sources(steps)
[xrec,yrec,Nr] = readRec(resfile); %measure-koordinates (# steps)


%stability according to pertubated field
dx = c/(12*fo);
dt=dx/(cmax*sqrt(2));
dtdx=dt/dx; %constant for all gridvalues

%scetch the system
drawSystem(xrec,yrec,sourceMatrix,xp,yp,c,fo,dx);

C_B=velMatrix_B*dtdx;
C_B=C_B.^2;

C_P=velMatrix_P*dtdx;
C_P=C_P.^2;

%time-step solution-matrices:

%background solution
u_koB = zeros(Nx,Ny); % time-step k-1
u_kB = zeros(Nx,Ny); % time-step k
u_kkB = zeros(Nx,Ny); % time-step k+1

% with perturbation
u_koP = zeros(Nx,Ny); % time-step k-1
u_kP = zeros(Nx,Ny); % time-step k
u_kkP = zeros(Nx,Ny); % time-step k+1


%.............Start computations.................

%snapcntr=1;
```

```
to = 1.5321/fo;
wo=2*pi*fo;
source = zeros(1,Nt);
Rec = zeros(length(xrec),Nt); % recordingmatrix
amp=3;

%time-loop:
for t_k = 1:Nt

    t=(t_k-1)*dt-to;
    %Rickerpulse
    %t=(t_k-1)*dt-delay;
    analytic_source(t_k)=...
        amp*exp(-pi*pi*fo*fo*t*t)*(1.0-2.0*pi*pi*fo*fo*t*t);

    %DOG
    %source(t_k)=-exp(0.5)*t*exp(-0.5*(wo*t)^2);

   %space-loops:
    for xi = 2:Nx-1
    for yi = 2:Ny-1
        u_kkB(xi,yi) =  C_B(xi,yi)*( u_kB(xi+1,yi) - ...
            4*u_kB(xi,yi) + u_kB(xi,yi+1) + u_kB(xi-1,yi) + u_kB(xi,yi-1));
        u_kkB(xi,yi) = u_kkB(xi,yi) + 2*u_kB(xi,yi)  - u_koB(xi,yi);
        u_kkB(xi,yi) = u_kkB(xi,yi) + ...
            analytic_source(t_k)*sourceMatrix(xi,yi)*(velMatrix_B(xi,yi)*dt)^2;

        u_kkP(xi,yi) =  C_P(xi,yi)*( u_kP(xi+1,yi) - ...
            4*u_kP(xi,yi) + u_kP(xi,yi+1) + u_kP(xi-1,yi) + u_kP(xi,yi-1));
        u_kkP(xi,yi) = u_kkP(xi,yi) + 2*u_kP(xi,yi)  - u_koP(xi,yi);
        u_kkP(xi,yi) = u_kkP(xi,yi) + ...
            analytic_source(t_k)*sourceMatrix(xi,yi)*(velMatrix_P(xi,yi)*dt)^2;

    end
    end

    %absorbing boundary:

    % (surface) u(0,y,t_k+1):
    for iy = 1:Ny
        kB=-dtdx*velMatrix_B(1,iy);
        u_kkB(1,iy) = u_kB(1,iy) - kB*(u_kB(2,iy)-u_kB(1,iy));
```

```
    kP=-dtdx*velMatrix_P(1,iy);
    u_kkP(1,iy) = u_kP(1,iy) - kP*(u_kP(2,iy)-u_kP(1,iy));

end

%(bottom) u(Nx,y,t_k+1):
for iy = 1:Ny
    kB=dtdx*velMatrix_B(Nx,iy);
    u_kkB(Nx,iy)=u_kB(Nx,iy)-kB*(u_kB(Nx,iy)-u_kB(Nx-1,iy));

    kP=dtdx*velMatrix_P(Nx,iy);
    u_kkP(Nx,iy)=u_kP(Nx,iy)-kP*(u_kP(Nx,iy)-u_kP(Nx-1,iy));
end


% side (left) u(x,0,t_k+1):
for ix=2:Nx-1
    kB=-dtdx*velMatrix_B(ix,1);
    u_kkB(ix,1)=u_kB(ix,1) - kB*(u_kB(ix,2)-u_kB(ix,1));

    kP=-dtdx*velMatrix_P(ix,1);
    u_kkP(ix,1)=u_kP(ix,1) - kP*(u_kP(ix,2)-u_kP(ix,1));
end

% side (right) u(x,Ny,t_k+1):
for ix=2:Nx-1
    kB=dtdx*velMatrix_B(ix,Ny);
    u_kkB(ix,Ny)=u_kB(ix,Ny) - kB*(u_kB(ix,Ny)-u_kB(ix,Ny-1));

     kP=dtdx*velMatrix_P(ix,Ny);
    u_kkP(ix,Ny)=u_kP(ix,Ny) - kP*(u_kP(ix,Ny)-u_kP(ix,Ny-1));
end


%time-step update:
u_koB = u_kB;
u_kB = u_kkB;

u_koP=u_kP;
u_kP=u_kkP;

%scattered field
u_kk = u_kkP-u_kkB;
```

```
            %measure the field
            for r=1:Nr
              Rec(r,t_k)=u_kk(xrec(r),yrec(r));
            end

        %snapshots
        %if snapcntr<=length(snapshots) && t_k==snapshots(snapcntr)
           % snap(u_kkB,u_kk,t_k,dt);
            %snapcntr = snapcntr+1;
        %end

end %end time loop



function [source] = solve2Dsource(velMatrix_B,c,fo,Nt,Nx,Ny)

%{
Rec is a mtrix with with measured data of the scattered field.
One column for each timestep.
%}

% Values to be read from file:
%{
f0           center frequency
Nx           # of x gridpoints
Ny           # of y gridpoints

Nsnap        # of snapshots
Nt           # of time samples
twindow      window of timesteps when the scattered field is measured
xrec,yrec    vectors with arraymeasure-coordinates(stepvalues).

velMatrix_B     Matrix with mediumvelocity (Backgroundmedium)
velMatrix_P     Matrix with mediumvelocity and pertubations
sourceMatrix Matrix with source-coordinates

%}
Xmid=ceil(Nx/2);
Ymid=ceil(Ny/2);
```

```
sCoordinateMatrix=zeros(Nx,Ny);
sCoordinateMatrix(Xmid,Ymid)=1;

%stability according to pertubated field
dx=c/(12*fo);
dt=dx/(c*sqrt(2));
dtdx=dt/dx; %constant for all gridvalues

C_B=velMatrix_B*dtdx;
C_B=C_B.^2;

%time-step solution-matrices:

%background solution
u_koB = zeros(Nx,Ny); % time-step k-1
u_kB = zeros(Nx,Ny); % time-step k
u_kkB = zeros(Nx,Ny); % time-step k+1

%.............Start computations.................

delay = 1.5/fo;

%time-loop:
for t_k = 1:Nt

    %sources:
    t=(t_k-1)*dt-delay;
    amp=5;
    %Rickerpulse
    A=amp*exp(-pi*pi*fo*fo*t*t)*(1.0-2.0*pi*pi*fo*fo*t*t);


    %space-loops:
    for xi = 2:Nx-1
    for yi = 2:Ny-1
        u_kkB(xi,yi) =  C_B(xi,yi)*( u_kB(xi+1,yi) - 4*u_kB(xi,yi) +...
            u_kB(xi,yi+1) + u_kB(xi-1,yi) + u_kB(xi,yi-1));
        u_kkB(xi,yi) = u_kkB(xi,yi) + 2*u_kB(xi,yi)  - u_koB(xi,yi);
        u_kkB(xi,yi) = u_kkB(xi,yi) +...
            A*sCoordinateMatrix(xi,yi)*(velMatrix_B(xi,yi)*dt)^2;

    end
    end
```

```
    %absorbing boundary:

    % (surface) u(0,y,t_k+1):
    for iy = 1:Ny
        kB=-dtdx*velMatrix_B(1,iy);
        u_kkB(1,iy) = u_kB(1,iy) - kB*(u_kB(2,iy)-u_kB(1,iy));
    end

    %(bottom) u(Nx,y,t_k+1):
    for iy = 1:Ny
        kB=dtdx*velMatrix_B(Nx,iy);
        u_kkB(Nx,iy)=u_kB(Nx,iy)-kB*(u_kB(Nx,iy)-u_kB(Nx-1,iy));
    end


    % side (left) u(x,0,t_k+1):
    for ix=2:Nx-1
        kB=-dtdx*velMatrix_B(ix,1);
        u_kkB(ix,1)=u_kB(ix,1) - kB*(u_kB(ix,2)-u_kB(ix,1));
    end

    % side (right) u(x,Ny,t_k+1):
    for ix=2:Nx-1
        kB=dtdx*velMatrix_B(ix,Ny);
        u_kkB(ix,Ny)=u_kB(ix,Ny) - kB*(u_kB(ix,Ny)-u_kB(ix,Ny-1));
    end


    %time-step update:
    u_koB = u_kB;
    u_kB = u_kkB;

   %measure the generated sourcefield
   source(t_k)=u_kkB(Xmid,Ymid);

 end %end time loop

function [] = snap(u_B,u,t_k,dt)
%SNAP Summary of this function goes here
%   Detailed explanation goes here
t=t_k*dt;
figure
```

```
surf(u_B);
title(strcat('Background waves at time t= ',num2str(t),' secconds'));

figure
surf(u);
title(strcat('scattered waves at time t= ',num2str(t),' secconds'));


function [Y,f] = myFFT(signalM,L,dt)

Nr=length(signalM(:,1));
Fs = 1/dt;     % Sampling frequency

Y=zeros(Nr,L);
for i=1:Nr
    Yf = fft(signalM(i,:),L)/L;
    Y(i,:)=Yf;
end
f = Fs*linspace(0,1,L);
```

6.2 Program package for simulating ideal point spread in section 4.1

```
function [] = program(fo,Nf,fstep)
% Imaging by ideally recording a single point spread.
% The array has coincide sources and receivers.

Nx=240;
Ny=Nx;
P=zeros(Nx,Ny);
% read data
[xrec,yrec,Nr]=readData('recData.txt');
[vel,pertVel] = readVelocity('velocityData.txt');
[xscatter,yscatter,Np] = readScatterer('scatterData.txt');

% systemconfiguration plot
dx=vel/(12*fo);
dy=dx;
lambda = vel/fo;
```

```
% Project the ideal recieving vector when a delta is reflected from a
% point spread:

Lf = fo - Nf*fstep/2+fstep; %lowest frequency
Uf = fo + Nf*fstep/2; %highest
w=2*pi*[Lf:fstep:Uf];%angular frequencies
drawSystem(P,dx,lambda,w,fstep,vel, xrec, yrec, xscatter, yscatter,Nr,Np);

for p=1:length(w)
    s = vecG(w(p),vel,dx,xrec,yrec,xscatter,yscatter); %ideal array
    for i=1:Nx                                          %recording
        for j=1:Ny
            P(i,j) = P(i,j) + s'*vecG(w(p),vel,dx,xrec,yrec,i,j);
        end
    end
end

csvwrite('P',P);
figure
titl = strvcat('Pointspread', strcat('frequency range: ', num2str(1), '-'...
       ,num2str(3), 'Hz'))
surf(abs(transpose(P)))
[Nx,Ny]=size(P);
xlim([0 Nx])
ylim([0 Ny])
set(gca,'XTick',0:12:Nx)
set(gca,'XTickLabel',0:Nx*dx/lambda)
set(gca,'YTick',0:12:Ny)
set(gca,'YTickLabel',0:Ny*dx/lambda)
grid on

function [] = program(Lf,Uf,Nstep)
% Imaging by ideally recording a single point spread.
% The array has coincide sources and receivers.

Nx=240;
Ny=Nx;
P1=zeros(Nx,Ny);
P_target=P1;

% read data
[xrec,yrec,Nr]=readData('recData.txt');
[vel,pertVel] = readVelocity('velocityData.txt');
```

```
[xscatter,yscatter,Np] = readScatterer('scatterData.txt');

% systemconfiguration plot
fo=30;
dx=vel/(12*fo);
dy=dx;
lambda = vel/fo;


% Project the ideal recieving vector when a delta is reflected from a
% point spread:


w=2*pi*linspace(Lf,Uf,Nstep);%angular frequencies
%drawSystem(P1,dx,lambda,w,fstep,vel, xrec, yrec, xscatter, yscatter...
%,Nr,Np);

p1=zeros(length(w),1);
p_target=p1;


figure
f=w/(2*pi);
titl = strvcat('Pointspread', strcat('frequency range: ', num2str(Lf), '-'...
    ,num2str(Uf), 'Hz'));
xlabel('frequencies')

x1=120;
y1=150;

for p=1:length(w)
    s = vecG(w(p),vel,dx,xrec,yrec,xscatter,yscatter); %ideal array recording
    p1(p) = s'*vecG(w(p),vel,dx,xrec,yrec,x1,y1);
    p_target(p) = s'*vecG(w(p),vel,dx,xrec,yrec,xscatter,yscatter);

    P1(x1,y1) = P1(x1,y1) + p1(p);
    P_target(xscatter,yscatter) = P_target(xscatter,yscatter) + p_target(p);

    subplot(2,1,1)
    hold on
    h1 =  stem(f(p),abs(P1(x1,y1)));
    subplot(2,1,2)
    hold on
```

```
htarget=    stem(f(p),abs(P_target(xscatter,yscatter)));
end
set(h1,'MarkerFaceColor','black')
set(htarget,'MarkerFaceColor','red')
hold off

figure
subplot(2,2,1)
title('measurement as function of frequency')
xlabel(strvcat('Upper figures shows adjacent point and lower the scatterer position
'Figures to the left gives the real part and to the right the imaginary part is giv
plot(f,real(p1),'black')
subplot(2,2,2)
plot(f,imag(p1),'black')
subplot(2,2,3)
plot(f,real(p_target),'red')
subplot(2,2,4)
plot(f,imag(p_target),'red')



function [g] = response(w,dx,vel,x1,y1,x2,y2)
% 2D impulse response from (x2,y2) to (x1,y1)

k=w/vel;

x1=dx*x1;
y1=dx*y1;
x2=dx*x2;
y2=dx*y2;

r=sqrt((x2-x1)^2 + (y2-y1)^2);
y=k*r;

f=2*pi/w;
f=1/f;
wl=vel/f;

% far field calculations only
if r < 3.5*wl
   g=0;
else
```

```
g=sqrt(2/(pi*y))*exp(i*(y-pi/4));
end

function [gVec] = vecG(w,vel,dx,xVec,yVec,xCoordinate,yCoordinate)
Nimp = length(xVec);
gVec=zeros(Nimp,1);

for k=1:Nimp
    gVec(k) = response(w,dx,vel,xVec(k),yVec(k),xCoordinate,yCoordinate);
end
gVec=gVec/norm(gVec);
```

6.3 Program package for simulating Born-data experiments in section 4.2 (backpropagation and MUSIC)

```
function [f_vector] = program1(Lf,Uf,Nf)
% Backpropagation including only a scatterer location and and a non
% scatterer position.

Nx=240;
Ny=Nx;
P1=zeros(Nx,Ny);


% read data
[xrec,yrec,Nr]=readData('recData.txt');
[xsource,ysource,Ns] = readData('sourceData.txt');
[vel,pertVel] = readVelocity('velocityData.txt');
[xscatter,yscatter,Np] = readScatterer('scatterData.txt');

% systemconfiguration plot
fo=30;
dx=vel/(12*fo);
dy=dx;
lambda = 500/30;

drawSystem(P1,dx,lambda,fo,vel, xrec, yrec, xsource, ysource, xscatter,...
    yscatter,Nr,Ns,Np);
%backpropagation
```

```
w=2*pi*linspace(Lf,Uf,Nf);%angular frequencies
f_vector = w/(2*pi);


Ps=zeros(length(w),1);
P2=Ps;
PsCum=0;
P2Cum=0;

figure
for f=1:length(w)
tau = ((1/pertVel)^2 - (1/vel)^2)*w(f)^2; %scattering potensial
K=transfermatrix(w(f),dx,vel,xrec,yrec,xsource,ysource,xscatter,yscatter,...
    tau,Nr,Ns,Np); %compute transfermatrix

[U,D,V]=svd(K);


        %receiver side
        v1=U(:,1); % singular vector corresponding to largest value

        gr = vecG(w(f),vel,dx,xrec,yrec,114,132);%scatterer Pos
        gr2 = vecG(w(f),vel,dx,xrec,yrec,180,120);%non scatterer Pos

        Ps(f) = v1'*gr;
        P2(f) = v1'*gr2;


        %source side
        e1=V(:,1);
        %e2=V(:,2);
        %e3=V(:,3);

        gs = vecG(w(f),vel,dx,xsource,ysource,114,132);
        gs2 = vecG(w(f),vel,dx,xsource,ysource,180,120);

        Ps(f)=Ps(f)+transpose(e1)*gs;
        P2(f)=P2(f)+transpose(e1)*gs2;

        PsCum = PsCum+Ps(f);
        P2Cum = P2Cum+P2(f);
```

```
subplot(2,1,2)
hold on
stem(f_vector(f),abs(P2Cum))
subplot(2,1,1)
hold on
stem(f_vector(f),abs(PsCum))
end

figure
subplot(2,2,1)
plot(f_vector,real(Ps))
subplot(2,2,2)
plot(f_vector,imag(Ps))
subplot(2,2,3)
plot(f_vector,real(P2),'red')
subplot(2,2,4)
plot(f_vector,imag(P2),'red')
```

```
function [] = program(fo)
% Imaging by employing MUSIC on Born-data.
% w gives the angular frequency. The input signal is a delta but
%the spatial sampling model assumes wave lengths about 17m
Nx=240;
Ny=Nx;
P=zeros(Nx,Ny);
% read data
[xrec,yrec,Nr]=readData('recData.txt');
[xsource,ysource,Ns] = readData('sourceData.txt');
[vel,pertVel] = readVelocity('velocityData.txt');
[xscatter,yscatter,Np] = readScatterer('scatterData.txt');

% systemconfiguration plot
dx=vel/(12*fo);
dy=dx;
lambda = vel/fo;
drawSystem(P,dx,lambda,fo,vel, xrec, yrec, xsource, ysource, xscatter,...
    yscatter,Nr,Ns,Np);
```

```
% compute transfermatrix
w=2*pi*fo;
tau = ((1/pertVel)^2 - (1/vel)^2)*w^2; %scattering potensial
K=transfermatrix(w,dx,vel,xrec,yrec,xsource,ysource,xscatter,yscatter,tau,...
    Nr,Ns,Np); %compute transfermatrix

% MUSIC
[U,D,V]=svd(K);

%singularvalues
figure
stem(diag(D));

for i=48:192
    for j=48:192
       Ps=0;
       for l=Np+1:min(Ns,Nr)
       v=U(:,l);
       e=V(:,l);
       gs = vecG(w,vel,dx,xsource,ysource,i,j);
       gr = vecG(w,vel,dx,xrec,yrec,i,j);
       Ps = Ps + abs(transpose(e)*gs) + abs(v'*gr) + 10^-5;
       end
       P(i,j)=1/Ps;
    end
end

csvwrite('P',P);

figure
title(strcat(['MUSIC at frequency:' num2str(fo) 'Hz']));
surf(abs(transpose(P)))
[Nx,Ny]=size(P);
xlim([0 Nx])
ylim([0 Ny])
set(gca,'XTick',0:12:Nx)
set(gca,'XTickLabel',0:Nx*dx/lambda)
set(gca,'YTick',0:12:Ny)
set(gca,'YTickLabel',0:Ny*dx/lambda)

function [g] = response(w,dx,vel,x1,y1,x2,y2)
% 2D impulse response from (x2,y2) to (x1,y1)
```

```
k=w/vel;

x1=dx*x1;
y1=dx*y1;
x2=dx*x2;
y2=dx*y2;

r=sqrt((x2-x1)^2 + (y2-y1)^2);
y=k*r;

f=2*pi/w;
f=1/f;
wl=vel/f;

% far field calculations only
if r < 3.5*wl
   g=0;
else
g=sqrt(2/(pi*y))*exp(i*(y-pi/4));
end

function [K] = transfermatrix(w,dx,vel,xrec,yrec,xsource,ysource,...
    xscatter,yscatter,tau,Nr,Ns,Np)
% K is the transfermatrix build by using a Born-data assumption.
U = zeros(Nr,Np);
V_T = zeros(Np,Ns);
for i=1:Nr
    for j=1:Np
        U(i,j) = response(w,dx,vel,xrec(i),yrec(i),xscatter(j),yscatter(j));
    end
end
for i=1:Np
    for j=1:Ns
         V_T(i,j) = response(w,dx,vel,xscatter(i),yscatter(i),xsource(j),...
             ysource(j));
    end
end
sigmas = tau*ones(Np,1);
Sigma = diag(sigmas);
K=U*Sigma*V_T;

function [gVec] = vecG(w,vel,dx,xVec,yVec,xCoordinate,yCoordinate)
```

```
Nimp = length(xVec);
gVec=zeros(Nimp,1);

for k=1:Nimp
    gVec(k) = response(w,dx,vel,xCoordinate,yCoordinate,xVec(k),yVec(k));
end
```

6.4 Program package for simulating propagating waves experiments in sections 4.3, 4.4 and 4.5 (backpropagation and MUSIC)

```
function [P] = program(frequencies)
% Backpropagation shown as a sum of imagesMatrices, each image is
%computed at a
% given frequency. Frequency-indicies are specified in frequencies.
%(Use the program in velocityanalysis to get the wanted frequency
%indices)
% P is a matrix where the projection is computed.

minR=3.5*500/30;

P1=zeros(240,240);
P2=P1;
P3=P1;

[nx,ny]=size(P1);
for i=1:length(frequencies)
[K,f_vector,c,xrec,yrec,sourcex,sourcey,dx] = ...
    TransferMatrix(frequencies(i)); % K(f=frequencies(i))
[u,s,v] = svd(K);%singular-value decomposition

w = 2*pi*f_vector(frequencies(i));% angular frequency
Rvector1= u(:,1);
%Rvector2= u(:,2);
%Rvector3= u(:,3);

P1 = P1 + projectR(Rvector1,w,c,xrec,yrec,nx,ny,dx,minR);
%P2 = P2 + projectR(Rvector2,w,c,xrec,yrec,nx,ny,dx,minR);
%P3 = P3 + projectR(Rvector3,w,c,xrec,yrec,nx,ny,dx,minR);

Svector1 = v(:,1);
```

```
%Svector2 = v(:,2);
%Svector3 = v(:,3);

P1 = P1 + projectS(Svector1,w,c,sourcex,sourcey,nx,ny,dx,minR);
%P2 = P2 + projectS(Svector2,w,c,sourcex,sourcey,nx,ny,dx,minR);
%P3 = P3 + projectS(Svector3,w,c,sourcex,sourcey,nx,ny,dx,minR);
end
P=P1+P2+P3;
csvwrite('P',P);
csvwrite('P1',P1);
csvwrite('P2',P2);
csvwrite('P3',P3);


f_start = num2str(f_vector(frequencies(1)));
f_end = num2str(f_vector(frequencies(length(frequencies))));
str1 = ['Twosided projection at frequency: ' f_start 'Hz'];
titl = strvcat(str1);

[Nx,Ny]=size(P);
figure
lambda=500/30; %Note that this is not neseseraly the signal wavelength!!
title(titl);
surf(abs(transpose(P)))
xlim([0 Nx])
ylim([0 Ny])
set(gca,'XTick',0:12:Nx)
set(gca,'XTickLabel',0:Nx*dx/lambda)
set(gca,'YTick',0:12:Ny)
set(gca,'YTickLabel',0:Ny*dx/lambda)


figure
lambda=500/30; %Note that this is not neseseraly the signal wavelength!!
title(titl);
surf(abs(transpose(P1)))
xlim([0 Nx])
ylim([0 Ny])
set(gca,'XTick',0:12:Nx)
set(gca,'XTickLabel',0:Nx*dx/lambda)
set(gca,'YTick',0:12:Ny)
set(gca,'YTickLabel',0:Ny*dx/lambda)
```

```
figure
lambda=500/30; %Note that this is not neseseraly the signal wavelength!!
title(titl);
surf(abs(transpose(P2)))
xlim([0 Nx])
ylim([0 Ny])
set(gca,'XTick',0:12:Nx)
set(gca,'XTickLabel',0:Nx*dx/lambda)
set(gca,'YTick',0:12:Ny)
set(gca,'YTickLabel',0:Ny*dx/lambda)


figure
lambda=500/30; %Note that this is not neseseraly the signal wavelength!!
title(titl);
surf(abs(transpose(P3)))
xlim([0 Nx])
ylim([0 Ny])
set(gca,'XTick',0:12:Nx)
set(gca,'XTickLabel',0:Nx*dx/lambda)
set(gca,'YTick',0:12:Ny)
set(gca,'YTickLabel',0:Ny*dx/lambda)

function [K,f_vector,c,xrec,yrec,sourcex,sourcey,dx] =...
    TransferMatrix(f_index)
%Transfermatrix at f(f_index)

paramfile = 'parameters.txt'; %file with parameterdata
velfile = 'velocityData.txt'; %file with velocitydata of background
sourcefile = 'sourceData.txt';
pertfile = 'scatterData.txt';
resfile = 'recData.txt';

[fo, Nx, Ny, snapshots, Nt] = readParameters(paramfile);
[xp,yp] = readPertCoords(pertfile);
[velMatrix_B,velMatrix_P,c, cmax] = readVel(velfile,Nx,Ny,xp,yp);
[sourceMatrix,Ns,sourcex,sourcey] = readSource(sourcefile,Nx,Ny);
[xrec,yrec,Nr] = readRec(resfile);

source = solve2Dsource(velMatrix_B,c,fo,Nt,Nx,Ny);
%*****compute Transfermatrix********
K=zeros(Nr,Ns);
```

```
%source input
[Rec,dt,dx]=solve2D(xrec, yrec, zeros(Nx,Ny), fo, Nx, Ny, Nt,...
    velMatrix_B, velMatrix_P, c, cmax);
%drawSystem(xrec,yrec,sourceMatrix,xp,yp,c,fo,dx);
%Fouriertransform source
[S,f_vector] = myFFT(source,Nt,dt);

norm = 1/S(f_index);
%Compute columns of K
for i=1:Ns
sCoordinateMatrix=zeros(Nx,Ny);%Matrix with one source for each iteration
sCoordinateMatrix(sourcex(i),sourcey(i))=1;
Rec=solve2D(xrec, yrec, sCoordinateMatrix, fo, Nx, Ny, Nt,...
    velMatrix_B, velMatrix_P, c, cmax);

%Fourier transform
Rec = myFFT(Rec,Nt,dt);
K(:,i)=norm*Rec(:,f_index);
end

function [Rec,dt,dx] = ...
    solve2D(xrec, yrec, sCoordinateMatrix, fo, Nx, Ny, Nt,...
    velMatrix_B, velMatrix_P, c, cmax)

%{
Rec is a mtrix with with measured data of the scattered field.
One column for each timestep.
%}

% Values to be read from file:
%{
f0           center frequency
Nx           # of x gridpoints
Ny           # of y gridpoints

Nsnap        # of snapshots
Nt           # of time samples
twindow      window of timesteps when the scattered field is measured
xrec,yrec    vectors with arraymeasure-coordinates(stepvalues).

velMatrix_B     Matrix with mediumvelocity (Backgroundmedium)
velMatrix_P     Matrix with mediumvelocity and pertubations
sourceMatrix Matrix with source-coordinates
```

```
%}



%stability according to pertubated field
dx=c/(12*fo);
dt=dx/(cmax*sqrt(2));
dtdx=dt/dx; %constant for all gridvalues



C_B=velMatrix_B*dtdx;
C_B=C_B.^2;

C_P=velMatrix_P*dtdx;
C_P=C_P.^2;

%time-step solution-matrices:

%background solution
u_koB = zeros(Nx,Ny); % time-step k-1
u_kB = zeros(Nx,Ny); % time-step k
u_kkB = zeros(Nx,Ny); % time-step k+1

% with perturbation
u_koP = zeros(Nx,Ny); % time-step k-1
u_kP = zeros(Nx,Ny); % time-step k
u_kkP = zeros(Nx,Ny); % time-step k+1


%.............Start computations.................



to = 1.5/fo;
source = zeros(1,Nt);
Rec = zeros(length(xrec),Nt); % recordingmatrix
amp=3;
%time-loop:
for t_k = 1:Nt
```

```
%sources:
t=(t_k-1)*dt-to;

%Rickerpulse
A=amp*exp(-pi*pi*fo*fo*t*t)*(1.0-2.0*pi*pi*fo*fo*t*t);


source(t_k)=A;
%space-loops:
for xi = 2:Nx-1
for yi = 2:Ny-1
    u_kkB(xi,yi) =  C_B(xi,yi)*( u_kB(xi+1,yi) -...
        4*u_kB(xi,yi) + u_kB(xi,yi+1) + u_kB(xi-1,yi) + u_kB(xi,yi-1));
    u_kkB(xi,yi) = u_kkB(xi,yi) + 2*u_kB(xi,yi)  - u_koB(xi,yi);
    u_kkB(xi,yi) = u_kkB(xi,yi) + ...
        A*sCoordinateMatrix(xi,yi)*(velMatrix_B(xi,yi)*dt)^2;

    u_kkP(xi,yi) =  C_P(xi,yi)*( u_kP(xi+1,yi) - ...
        4*u_kP(xi,yi) + u_kP(xi,yi+1) + u_kP(xi-1,yi) + u_kP(xi,yi-1));
    u_kkP(xi,yi) = u_kkP(xi,yi) + 2*u_kP(xi,yi)  - u_koP(xi,yi);
    u_kkP(xi,yi) = u_kkP(xi,yi) +...
        A*sCoordinateMatrix(xi,yi)*(velMatrix_P(xi,yi)*dt)^2;

end
end

%absorbing boundary:

% (surface) u(0,y,t_k+1):
for iy = 1:Ny
    kB=-dtdx*velMatrix_B(1,iy);
    u_kkB(1,iy) = u_kB(1,iy) - kB*(u_kB(2,iy)-u_kB(1,iy));

    kP=-dtdx*velMatrix_P(1,iy);
    u_kkP(1,iy) = u_kP(1,iy) - kP*(u_kP(2,iy)-u_kP(1,iy));

end

%(bottom) u(Nx,y,t_k+1):
for iy = 1:Ny
    kB=dtdx*velMatrix_B(Nx,iy);
    u_kkB(Nx,iy)=u_kB(Nx,iy)-kB*(u_kB(Nx,iy)-u_kB(Nx-1,iy));
```

```
        kP=dtdx*velMatrix_P(Nx,iy);
        u_kkP(Nx,iy)=u_kP(Nx,iy)-kP*(u_kP(Nx,iy)-u_kP(Nx-1,iy));
    end


    % side (left) u(x,0,t_k+1):
    for ix=2:Nx-1
        kB=-dtdx*velMatrix_B(ix,1);
        u_kkB(ix,1)=u_kB(ix,1) - kB*(u_kB(ix,2)-u_kB(ix,1));

        kP=-dtdx*velMatrix_P(ix,1);
        u_kkP(ix,1)=u_kP(ix,1) - kP*(u_kP(ix,2)-u_kP(ix,1));
    end

    % side (right) u(x,Ny,t_k+1):
    for ix=2:Nx-1
        kB=dtdx*velMatrix_B(ix,Ny);
        u_kkB(ix,Ny)=u_kB(ix,Ny) - kB*(u_kB(ix,Ny)-u_kB(ix,Ny-1));

         kP=dtdx*velMatrix_P(ix,Ny);
        u_kkP(ix,Ny)=u_kP(ix,Ny) - kP*(u_kP(ix,Ny)-u_kP(ix,Ny-1));
    end


    %time-step update:
    u_koB = u_kB;
    u_kB = u_kkB;

    u_koP=u_kP;
    u_kP=u_kkP;

    %scattered field
    u_kk = u_kkP-u_kkB;

          %measure the field
          for r=1:length(xrec)
            Rec(r,t_k) = u_kk(xrec(r),yrec(r));
          end



end %end time loop
```

```
function [A] = projectR(vector,w,c,xrec,yrec,nx,ny,dx,minR)
% projection of vector on to points in a grid defined input params.
% projections are set to a size eps at distances less than minR.

% xrec,yrec are dicrete positionsvalues where projection is computed from.
% dx is the phycal-distance cofficient (dx*xrec)
%nx,ny arenumber of x and y positions in projection-grid.
%w is the angular frequency used in impulse-response calculations.
% vector contains output sutch that (xrec(i),yrec(i)) has output = vector(i)
xstart = 12;
ystart = 54;
yend = 206;
xend = 168;

A=zeros(nx,ny);
xr = dx*xrec;
yr = dx*yrec;
for i=xstart:xend
    for j = ystart:yend
        gvec = vecG(w,c,xr,yr,i*dx,j*dx,minR);
        value = transpose(vector)*gvec;
        A(i,j) = value;
    end
end


function [A] = projectS(vector,w,c,xsource,ysource,nx,ny,dx,minR)
% projection of vector on to points in a grid defined input params.
% projections are set to a size eps at distances less than minR.

% xrec,yrec are dicrete positionsvalues where projection is computed from.
% dx is the phycal-distance cofficient (dx*xrec)
%nx,ny arenumber of x and y positions in projection-grid.
%w is the angular frequency used in impulse-response calculations.
% vector contains output sutch that (xrec(i),yrec(i)) has output = vector(i)
xstart = 12;
ystart = 54;
yend = 206;
xend = 168;
A=zeros(nx,ny);
xs = dx*xsource;
ys = dx*ysource;
for i=xstart:xend
```

```
    for j=ystart:yend
        gvec = vecG(w,c,xs,ys,i*dx,j*dx,minR);
        value = vector'*gvec;
        A(i,j) = value;
    end
end

function [Projection,freq] = program(f_index)
% Projection is a matrix with values evaluated for each point in a grid
% Evaluation is done using MUSIC at frequency f=freq(f_index)).
% (exact frequency is found by running velocityanalysis.)
% vecG_s and vecG_r are vectors with impulse responses from the
% transmitter and receiver-coordinates to a grid posision.
% These are computed for each position in the grid.


[K,Np,Ns,xsource,ysource,Nr,xrec,yrec,freq,c,Nx,Ny,dx,dy] =...
    TransferMatrix(f_index);
[U,D,V] = svd(K);   % U is connected to receivers
                    % V to sources
                    % D to scatters
w=2*pi*freq;
Projection = zeros(Nx,Ny);

%does not include coordinates outside array-window.
for i=48:192
    for j=48:192
        %vecG_t and vecG_r is calculated for all grid points.
        vecG_s = vecG(xsource,ysource,i,j,dx,dy,w,c);
        vecG_r = vecG(xrec,yrec,i,j,dx,dy,w,c);
        % compute innerproducts:
        Ps=0;
        for m=Np+2:Ns
           Ps = Ps + abs(transpose(U(:,m))*vecG_r) + abs(V(:,m)'*vecG_s);
        end
        Projection(i,j) = 1/Ps;
    end
end


lambda=500/30;
figure
titl = strcat(['MUSIC at frequency:' num2str(freq) 'Hz']);
```

```
surf(abs(transpose(Projection)))
xlim([0 Nx])
ylim([0 Ny])
set(gca,'XTick',0:12:Nx)
set(gca,'XTickLabel',0:Nx*dx/lambda)
set(gca,'YTick',0:12:Ny)
set(gca,'YTickLabel',0:Ny*dx/lambda)

figure
title(strcat(['singularvalues computed at frequency:' num2str(freq) 'Hz']));
D=svd(K);
stem(D);

function [Projection,freq] = program1(f_index)
% Projection is a matrix with values evaluated for each point in a grid
% Evaluation is done using MUSIC at frequency f=freq(f_index)).
% (exact frequency is found by running velocityanalysis.)
% vecG_s and vecG_r are vectors with impulse responses from the
% transmitter and receiver-coordinates to a grid posision.
% These are computed for each position in the grid.
%This program computes  both two sided and one sided images.


[K,Np,Ns,xsource,ysource,Nr,xrec,yrec,freq,c,Nx,Ny,dx,dy] = ...
    TransferMatrix(f_index);
[U,D,V] = svd(K);   % U is connected to receivers
                    % V to sources
                    % D to scatters
w=2*pi*freq;
ProjectionR = zeros(Nx,Ny);%Receiver array image
ProjectionS = zeros(Nx,Ny);%source array image
Projection = zeros(Nx,Ny); %two sided image

%MUSIC
%does not include coordinates outside array-window.
for i=48:192
    for j=48:192
        %vecG_t and vecG_r is calculated for all grid points.
        vecG_s = vecG(xsource,ysource,i,j,dx,dy,w,c);
        vecG_r = vecG(xrec,yrec,i,j,dx,dy,w,c);
        % compute innerproducts:
        PsR=0;
        PsS=0;
```

```
        Ps=0;
        for m=Np+1:Ns
           PsR = PsR + abs(transpose(U(:,m))*vecG_r);
           PsS = PsS + abs(V(:,m)'*vecG_s);
           Ps = Ps +  abs(V(:,m)'*vecG_s) + abs(transpose(U(:,m))*vecG_r);
        end
        ProjectionR(i,j) = 1/PsR;
        ProjectionS(i,j) = 1/PsS;
        Projection(i,j) = 1/Ps;
    end
end


lambda=500/30;
figure
titl = strcat(['MUSIC at frequency:' num2str(freq) 'Hz']);
surf(abs(transpose(Projection)))
xlim([0 Nx])
ylim([0 Ny])
set(gca,'XTick',0:12:Nx)
set(gca,'XTickLabel',0:Nx*dx/lambda)
set(gca,'YTick',0:12:Ny)
set(gca,'YTickLabel',0:Ny*dx/lambda)
title(titl)


figure
titl = strcat(['Receiver side MUSIC at frequency:' num2str(freq) 'Hz']);
surf(abs(transpose(ProjectionR)))
xlim([0 Nx])
ylim([0 Ny])
set(gca,'XTick',0:12:Nx)
set(gca,'XTickLabel',0:Nx*dx/lambda)
set(gca,'YTick',0:12:Ny)
set(gca,'YTickLabel',0:Ny*dx/lambda)
title(titl)

figure
titl = strcat(['Source side MUSIC at frequency:' num2str(freq) 'Hz']);
surf(abs(transpose(ProjectionS)))
xlim([0 Nx])
ylim([0 Ny])
set(gca,'XTick',0:12:Nx)
```

```
set(gca,'XTickLabel',0:Nx*dx/lambda)
set(gca,'YTick',0:12:Ny)
set(gca,'YTickLabel',0:Ny*dx/lambda)
title(titl)

function [Rec,dt,dx] = ...
    solve2D(xrec, yrec, sCoordinateMatrix, fo, Nx, Ny, Nt,...
    velMatrix_B, velMatrix_P, c, cmax)

%{
Rec is a mtrix with with measured data of the scattered field.
One column for each timestep.
%}

% Values to be read from file:
%{
f0           center frequency
Nx           # of x gridpoints
Ny           # of y gridpoints

Nsnap        # of snapshots
Nt           # of time samples
twindow      window of timesteps when the scattered field is measured
xrec,yrec    vectors with arraymeasure-coordinates(stepvalues).

velMatrix_B     Matrix with mediumvelocity (Backgroundmedium)
velMatrix_P     Matrix with mediumvelocity and pertubations
sourceMatrix Matrix with source-coordinates

%}



%stability according to pertubated field
dx=c/(12*fo);
dt=dx/(cmax*sqrt(2));
dtdx=dt/dx; %constant for all gridvalues

C_B=velMatrix_B*dtdx;
C_B=C_B.^2;

C_P=velMatrix_P*dtdx;
```

```
C_P=C_P.^2;

%time-step solution-matrices:

%background solution
u_koB = zeros(Nx,Ny); % time-step k-1
u_kB = zeros(Nx,Ny); % time-step k
u_kkB = zeros(Nx,Ny); % time-step k+1

% with perturbation
u_koP = zeros(Nx,Ny); % time-step k-1
u_kP = zeros(Nx,Ny); % time-step k
u_kkP = zeros(Nx,Ny); % time-step k+1


%.............Start computations.................

delay = 1.5/fo;
Rec = zeros(length(xrec),Nt); % recordingmatrix
source = zeros(Nt,1);
%time-loop:
for t_k = 1:Nt

    %sources:

    t=(t_k-1)*dt-delay;
    amp=5;
    %Rickerpulse
    A=amp*exp(-pi*pi*fo*fo*t*t)*(1.0-2.0*pi*pi*fo*fo*t*t);

    %sine-wave
    %A=amp*sin(2*pi*fo*t);

    source(t_k)=A;
    %space-loops:
    for xi = 2:Nx-1
    for yi = 2:Ny-1
        u_kkB(xi,yi) =  C_B(xi,yi)*( u_kB(xi+1,yi) - ...
            4*u_kB(xi,yi) + u_kB(xi,yi+1) + u_kB(xi-1,yi) + u_kB(xi,yi-1));
        u_kkB(xi,yi) = u_kkB(xi,yi) + 2*u_kB(xi,yi)  - u_koB(xi,yi);
        u_kkB(xi,yi) = u_kkB(xi,yi) + ...
            A*sCoordinateMatrix(xi,yi)*(velMatrix_B(xi,yi)*dt)^2;
```

```
    u_kkP(xi,yi) =  C_P(xi,yi)*( u_kP(xi+1,yi) - ...
        4*u_kP(xi,yi) + u_kP(xi,yi+1) + u_kP(xi-1,yi) + u_kP(xi,yi-1));
    u_kkP(xi,yi) = u_kkP(xi,yi) + 2*u_kP(xi,yi)  - u_koP(xi,yi);
    u_kkP(xi,yi) = u_kkP(xi,yi) +...
        A*sCoordinateMatrix(xi,yi)*(velMatrix_P(xi,yi)*dt)^2;

end
end

%absorbing boundary:

% (surface) u(0,y,t_k+1):
for iy = 1:Ny
    kB=-dtdx*velMatrix_B(1,iy);
    u_kkB(1,iy) = u_kB(1,iy) - kB*(u_kB(2,iy)-u_kB(1,iy));

    kP=-dtdx*velMatrix_P(1,iy);
    u_kkP(1,iy) = u_kP(1,iy) - kP*(u_kP(2,iy)-u_kP(1,iy));

end

%(bottom) u(Nx,y,t_k+1):
for iy = 1:Ny
    kB=dtdx*velMatrix_B(Nx,iy);
    u_kkB(Nx,iy)=u_kB(Nx,iy)-kB*(u_kB(Nx,iy)-u_kB(Nx-1,iy));

    kP=dtdx*velMatrix_P(Nx,iy);
    u_kkP(Nx,iy)=u_kP(Nx,iy)-kP*(u_kP(Nx,iy)-u_kP(Nx-1,iy));
end


% side (left) u(x,0,t_k+1):
for ix=2:Nx-1
    kB=-dtdx*velMatrix_B(ix,1);
    u_kkB(ix,1)=u_kB(ix,1) - kB*(u_kB(ix,2)-u_kB(ix,1));

    kP=-dtdx*velMatrix_P(ix,1);
    u_kkP(ix,1)=u_kP(ix,1) - kP*(u_kP(ix,2)-u_kP(ix,1));
end

% side (right) u(x,Ny,t_k+1):
for ix=2:Nx-1
    kB=dtdx*velMatrix_B(ix,Ny);
```

```
        u_kkB(ix,Ny)=u_kB(ix,Ny) - kB*(u_kB(ix,Ny)-u_kB(ix,Ny-1));

         kP=dtdx*velMatrix_P(ix,Ny);
        u_kkP(ix,Ny)=u_kP(ix,Ny) - kP*(u_kP(ix,Ny)-u_kP(ix,Ny-1));
    end


    %time-step update:
    u_koB = u_kB;
    u_kB = u_kkB;

    u_koP=u_kP;
    u_kP=u_kkP;

    %scattered field
    u_kk = u_kkP-u_kkB;

           %measure the field
          for r=1:length(xrec)
          Rec(r,t_k)=u_kk(xrec(r),yrec(r));
             end



end %end time loop


function [K,Np,Ns,sourcex,sourcey,Nr,xrec,yrec,freq,c,Nx,Ny,dx,dy] = ...
    TransferMatrix(f_index)
%Transfermatrix at f(f_index)

paramfile = 'parameters.txt'; %file with parameterdata
velfile = 'velocityData.txt'; %file with velocitydata of background
sourcefile = 'sourceData.txt';
pertfile = 'scatterData.txt';
resfile = 'recData.txt';

[fo, Nx, Ny, snapshots, Nt] = readParameters(paramfile);
[xp,yp,Np] = readPertCoords(pertfile); %pertubation
                                       %-coordinates(numer of steps)
[velMatrix_B,velMatrix_P,c, cmax] = readVel(velfile,Nx,Ny,xp,yp);
[sourceMatrix,Ns,sourcex,sourcey] = readSource(sourcefile,Nx,Ny); %locat
                                              %-ions of sources(steps)
```

```
[xrec,yrec,Nr] = readRec(resfile); %measure-koordinates (# steps)

% Generate source by measuring at an ingrid position
source = solve2Dsource(velMatrix_B,c,fo,Nt,Nx,Ny);

%*****compute Transfermatrix********
K=zeros(Nr,Ns);
%source input
[Rec,dt,dx]=solve2D(xrec, yrec, sourceMatrix, fo, Nx, Ny, Nt,...
    velMatrix_B, velMatrix_P, c, cmax);
%plot configuration.
drawSystem(xrec,yrec,sourceMatrix,xp,yp,c,fo,dx);
%Fourier transform source
[S,f_vector] = myFFT(source,Nt,dt);

norm = 1/S(f_index);
%Compute columns of K
for i=1:Ns
sCoordinateMatrix=zeros(Nx,Ny);%Matrix with one source for each iteration
sCoordinateMatrix(sourcex(i),sourcey(i))=1;
Rec=solve2D(xrec, yrec, sCoordinateMatrix, fo, Nx, Ny, Nt, ...
    velMatrix_B, velMatrix_P, c, cmax);

%Fourier transform
Rec = myFFT(Rec,Nt,dt);
K(:,i)=norm*Rec(:,f_index);
end
[U,D,V] = svd(K);
figure
stem(diag(D));

freq = f_vector(f_index);
dy=dx;
```

6.5 Program package for simulating propagating waves experiments with noise on receivers in section 4.6 (MUSIC)

```
function [Projection,freq] = program(f_index,SNR)
% Projection is a matrix with values evaluated for each point in a grid
% from Transfermatrix. Evaluation is done using MUSIC at frequency f=freq.
```

```
% program assumes that number of source-coordinates
% <= #receiver-coordinates

% vecG_s and vecG_r are vectors with impulse responses from the
% transmitter and receiver-coordinates to a grid position.
% These are computed for each position in the grid.

% (Number of perturbations should be known. They can be found using
% the program singularvalues.m with the same parameterfiles.)

[K,Np,Ns,xsource,ysource,Nr,xrec,yrec,freq,c,Nx,Ny,dx,dy] =...
    TransferMatrix(f_index,SNR);
[U,D,V] = svd(K); % U is connected to receivers
                  % V to sources
                  % D to scatters
w=2*pi*freq;
wl=c/30; %wave length of input source when fo=30
Projection = zeros(Nx,Ny);


%does not include coordinates outside source-receiver-window.
for i=48:192
    for j=48:192
    %vecG_t and vecG_r is calculated for all grid points.
    vecG_s = vecG(xsource,ysource,i,j,dx,dy,w,c);
    vecG_r = vecG(xrec,yrec,i,j,dx,dy,w,c);

        Ps=0;
        for m=Np+1:Ns
           Ps = Ps + abs(V(:,m)'*vecG_s)+ abs(conj(U(:,m)')*vecG_r);
        end
        Projection(i,j) = 1/Ps;
    end
end
csvwrite('P',Projection);

figure
D=svd(K);
stem(D);

titl = strvcat(strcat(['MUSIC at frequency:' num2str(freq) 'Hz']),...
    ['SNR = ' num2str(SNR)]);
figure
```

```
hold
title(titl);
surf(abs(transpose(Projection)))
wl=c/30;
set(gca,'Xtick',xrec(1:2:length(xrec)));
set(gca,'XtickLabel',xrec(1:2:length(xrec))*dx/wl);
set(gca,'Ytick',ysource(1:2:length(ysource)));
set(gca,'YtickLabel',ysource(1:2:length(ysource))*dx/wl);


function [K,Np,Ns,sourcex,sourcey,Nr,xrec,yrec,freq,c,Nx,Ny,dx,dy] = ...
    TransferMatrix(f_index,SNR)
%Transfermatrix at f(f_index)

paramfile = 'parameters.txt'; %file with parameterdata
velfile = 'velocityData.txt'; %file with velocitydata of background
sourcefile = 'sourceData.txt';
pertfile = 'scatterData.txt';
resfile = 'recData.txt';
Nsamples = 65;%Number of samples included in computing signal power
              %This is set equal to the duritation of the input signal

[fo, Nx, Ny, snapshots, Nt] = readParameters(paramfile);
[xp,yp,Np] = readPertCoords(pertfile);
[velMatrix_B,velMatrix_P,c, cmax] = readVel(velfile,Nx,Ny,xp,yp);
[sourceMatrix,Ns,sourcex,sourcey] = readSource(sourcefile,Nx,Ny);
[xrec,yrec,Nr] = readRec(resfile);


source = solve2Dsource(velMatrix_B,c,fo,Nt,Nx,Ny);

%*****compute Transfermatrix********
K=zeros(Nr,Ns);
[Rec,dt,dx] = solve2D(xrec, yrec, sourceMatrix, fo, Nx, Ny, Nt,...
    velMatrix_B, velMatrix_P, c, cmax);
%Fourier transform source
[S,f_vector] = myFFT(source,Nt,dt);
norm = 1/S(f_index);
%Compute columns of K
for i=1:Ns
sCoordinateMatrix=zeros(Nx,Ny);%Matrix with one source for each iteration
sCoordinateMatrix(sourcex(i),sourcey(i))=1;
Rec=solve2D(xrec, yrec, sCoordinateMatrix, fo, Nx, Ny, Nt,...
```

```
    velMatrix_B, velMatrix_P, c, cmax);

%Noise:
Rec_Var = signalPower(Rec,Nr,Nsamples); %estimated receiver output-power
Noise_Var = NoisePower(Rec_Var,Nr,SNR); % Noise-variance according to SNR
Noise = gaussian(Noise_Var,Nr,Nt); % Matrix with white Noise in each row
Rec = Rec + Noise; %add Noise

%Fourier transform
Rec = myFFT(Rec,Nt,dt);
K(:,i)=norm*Rec(:,f_index);
end


freq = f_vector(f_index);
dy=dx;

function [Rec,dt,dx] = solve2D(xrec, yrec, sCoordinateMatrix, fo, Nx, Ny, Nt, velMa

%{
Rec is a mtrix with with measured data of the scattered field.
One column for each timestep.
%}

% Values to be read from file:
%{
f0           center frequency
Nx           # of x gridpoints
Ny           # of y gridpoints

Nsnap        # of snapshots
Nt           # of time samples
twindow      window of timesteps when the scattered field is measured
xrec,yrec    vectors with arraymeasure-coordinates(stepvalues).

velMatrix_B     Matrix with mediumvelocity (Backgroundmedium)
velMatrix_P     Matrix with mediumvelocity and pertubations
sourceMatrix Matrix with source-coordinates

%}
```

```
%stability according to pertubated field
dx=c/(12*fo);
dt=dx/(cmax*sqrt(2));
dtdx=dt/dx; %constant for all gridvalues



C_B=velMatrix_B*dtdx;
C_B=C_B.^2;

C_P=velMatrix_P*dtdx;
C_P=C_P.^2;

%time-step solution-matrices:

%background solution
u_koB = zeros(Nx,Ny); % time-step k-1
u_kB = zeros(Nx,Ny); % time-step k
u_kkB = zeros(Nx,Ny); % time-step k+1

% with perturbation
u_koP = zeros(Nx,Ny); % time-step k-1
u_kP = zeros(Nx,Ny); % time-step k
u_kkP = zeros(Nx,Ny); % time-step k+1


%.............Start computations.................


source = zeros(1,Nt);
delay = 1.5/fo;
Rec = zeros(length(xrec),Nt); % recordingmatrix

%time-loop:
for t_k = 1:Nt

    %sources:

    t=(t_k-1)*dt-delay;
    amp=5;
    %Rickerpulse
    A=amp*exp(-pi*pi*fo*fo*t*t)*(1.0-2.0*pi*pi*fo*fo*t*t);
```

```
%sine-wave
%A=amp*sin(2*pi*fo*t);

source(t_k)=A;
%space-loops:
for xi = 2:Nx-1
for yi = 2:Ny-1
    u_kkB(xi,yi) =  C_B(xi,yi)*( u_kB(xi+1,yi) - ...
        4*u_kB(xi,yi) + u_kB(xi,yi+1) + u_kB(xi-1,yi) + u_kB(xi,yi-1));
    u_kkB(xi,yi) = u_kkB(xi,yi) + 2*u_kB(xi,yi)  - u_koB(xi,yi);
    u_kkB(xi,yi) = u_kkB(xi,yi) +...
        A*sCoordinateMatrix(xi,yi)*(velMatrix_B(xi,yi)*dt)^2;

    u_kkP(xi,yi) =  C_P(xi,yi)*( u_kP(xi+1,yi) - ...
        4*u_kP(xi,yi) + u_kP(xi,yi+1) + u_kP(xi-1,yi) + u_kP(xi,yi-1));
    u_kkP(xi,yi) = u_kkP(xi,yi) + 2*u_kP(xi,yi)  - u_koP(xi,yi);
    u_kkP(xi,yi) = u_kkP(xi,yi) + ...
        A*sCoordinateMatrix(xi,yi)*(velMatrix_P(xi,yi)*dt)^2;

end
end

%absorbing boundary:

% (surface) u(0,y,t_k+1):
for iy = 1:Ny
    kB=-dtdx*velMatrix_B(1,iy);
    u_kkB(1,iy) = u_kB(1,iy) - kB*(u_kB(2,iy)-u_kB(1,iy));

    kP=-dtdx*velMatrix_P(1,iy);
    u_kkP(1,iy) = u_kP(1,iy) - kP*(u_kP(2,iy)-u_kP(1,iy));

end

%(bottom) u(Nx,y,t_k+1):
for iy = 1:Ny
    kB=dtdx*velMatrix_B(Nx,iy);
    u_kkB(Nx,iy)=u_kB(Nx,iy)-kB*(u_kB(Nx,iy)-u_kB(Nx-1,iy));

    kP=dtdx*velMatrix_P(Nx,iy);
    u_kkP(Nx,iy)=u_kP(Nx,iy)-kP*(u_kP(Nx,iy)-u_kP(Nx-1,iy));
end
```

```
    % side (left) u(x,0,t_k+1):
    for ix=2:Nx-1
        kB=-dtdx*velMatrix_B(ix,1);
        u_kkB(ix,1)=u_kB(ix,1) - kB*(u_kB(ix,2)-u_kB(ix,1));

        kP=-dtdx*velMatrix_P(ix,1);
        u_kkP(ix,1)=u_kP(ix,1) - kP*(u_kP(ix,2)-u_kP(ix,1));
    end

    % side (right) u(x,Ny,t_k+1):
    for ix=2:Nx-1
        kB=dtdx*velMatrix_B(ix,Ny);
        u_kkB(ix,Ny)=u_kB(ix,Ny) - kB*(u_kB(ix,Ny)-u_kB(ix,Ny-1));

         kP=dtdx*velMatrix_P(ix,Ny);
        u_kkP(ix,Ny)=u_kP(ix,Ny) - kP*(u_kP(ix,Ny)-u_kP(ix,Ny-1));
    end


    %time-step update:
    u_koB = u_kB;
    u_kB = u_kkB;

    u_koP=u_kP;
    u_kP=u_kkP;

    %scattered field
    u_kk = u_kkP-u_kkB;

           %measure the field
          for r=1:length(xrec)
          Rec(r,t_k)=u_kk(xrec(r),yrec(r));
             end



end %end time loop


function [Rec_P] = signalPower(Rec,Nr,Ls)
% Compute energy of all signals in Rec
```

```
% Ls is the assumed signal-duretation as number of steps.

Rec_P = zeros(Nr,1);
for r=1:Nr
    signal = Rec(r,:);
    l = find(abs(signal)>10^-10,1,'first'); %Ricker-pulse settings
    % Power is computed individually for each output
    for i=l:Ls+l
    Rec_P(r) = Rec_P(r) + signal(i)^2;
    end
    Rec_P(r) = Rec_P(r)/Ls;
end

function [Noise_Var] = NoisePower(Rec_Var,Nr,SNR)
%Set values on individual outputs according to SNR
Noise_Var = zeros(Nr,1);
for r=1:Nr
    Noise_Var(r) = Rec_Var(r)/SNR;
end



function [Y] = gaussian(Noise_Var,Nr,Nt)
% Y is a matrix with a gaussian distribution in every row.
% The distribution has a mean value of zero and Noise_varianse as varianse.
% Nt gives the number of samples in the ditributions
Y = zeros(Nr,Nt);
for r=1:Nr
   rand('seed',r);
   Y(r,:) =random('norm',0,sqrt(Noise_Var(r)),1,Nt);%mean = 0
end
```

6.5.1 Common programs needed in both algorithms with and without noise

```
function [gvec] = vecG(w,c,xvec,yvec,x,y,minR)
% gvec is a vector with impulse responses from
% positions (xvec,yvec) to a single point (x,y)
Nx = length(xvec);
gvec = zeros(Nx,1);
for k=1:Nx
    gvec(k) = G(w,c,xvec(k),yvec(k),x,y,minR);
end
```

```
gvec = gvec/norm(gvec);

function [g] = G(w,c,x1,y1,x2,y2,minR)
% Impulserespons, evaluates the response from (x1,y1) to (x2,y2).

k=w/c;
r=sqrt((x2-x1)^2 + (y2-y1)^2);
if r>minR
y=k*r;
g=sqrt(2/(pi*y))*exp(i*(y-pi/4));
else
    g=0;
end

function [Y,f] = myFFT(signalM,L,dt)

Nr=length(signalM(:,1));
Fs = 1/dt;     % Sampling frequency

Y=zeros(Nr,L);
for i=1:Nr
    Yf = fft(signalM(i,:),L)/L;
    Y(i,:)=Yf;
end
f = Fs*linspace(0,1,L);


function [source] = solve2Dsource(velMatrix_B,c,fo,Nt,Nx,Ny)

%{
Rec is a mtrix with with measured data of the scattered field.
One column for each timestep.
%}

% Values to be read from file:
%{
f0           center frequency
Nx           # of x gridpoints
Ny           # of y gridpoints

Nsnap        # of snapshots
Nt           # of time samples
twindow      window of timesteps when the scattered field is measured
```

```
xrec,yrec    vectors with arraymeasure-coordinates(stepvalues).

velMatrix_B     Matrix with mediumvelocity (Backgroundmedium)
velMatrix_P     Matrix with mediumvelocity and pertubations
sourceMatrix Matrix with source-coordinates

%}
Xmid=ceil(Nx/2);
Ymid=ceil(Ny/2);
sCoordinateMatrix=zeros(Nx,Ny);
sCoordinateMatrix(Xmid,Ymid)=1;

%stability according to pertubated field
dx=c/(12*fo);
dt=dx/(c*sqrt(2));
dtdx=dt/dx; %constant for all gridvalues

C_B=velMatrix_B*dtdx;
C_B=C_B.^2;

%time-step solution-matrices:

%background solution
u_koB = zeros(Nx,Ny); % time-step k-1
u_kB = zeros(Nx,Ny); % time-step k
u_kkB = zeros(Nx,Ny); % time-step k+1

%.............Start computations.................

delay = 1.5/fo;

%time-loop:
for t_k = 1:Nt

    %sources:
    t=(t_k-1)*dt-delay;
    amp=5;
    %Rickerpulse
    A=amp*exp(-pi*pi*fo*fo*t*t)*(1.0-2.0*pi*pi*fo*fo*t*t);


    %space-loops:
    for xi = 2:Nx-1
```

```
for yi = 2:Ny-1
    u_kkB(xi,yi) =  C_B(xi,yi)*( u_kB(xi+1,yi) - ...
        4*u_kB(xi,yi) + u_kB(xi,yi+1) + u_kB(xi-1,yi) + u_kB(xi,yi-1));
    u_kkB(xi,yi) = u_kkB(xi,yi) + 2*u_kB(xi,yi)  - u_koB(xi,yi);
    u_kkB(xi,yi) = u_kkB(xi,yi) + ...
        A*sCoordinateMatrix(xi,yi)*(velMatrix_B(xi,yi)*dt)^2;

end
end

%absorbing boundary:

% (surface) u(0,y,t_k+1):
for iy = 1:Ny
    kB=-dtdx*velMatrix_B(1,iy);
    u_kkB(1,iy) = u_kB(1,iy) - kB*(u_kB(2,iy)-u_kB(1,iy));
end

%(bottom) u(Nx,y,t_k+1):
for iy = 1:Ny
    kB=dtdx*velMatrix_B(Nx,iy);
    u_kkB(Nx,iy)=u_kB(Nx,iy)-kB*(u_kB(Nx,iy)-u_kB(Nx-1,iy));
end


% side (left) u(x,0,t_k+1):
for ix=2:Nx-1
    kB=-dtdx*velMatrix_B(ix,1);
    u_kkB(ix,1)=u_kB(ix,1) - kB*(u_kB(ix,2)-u_kB(ix,1));
end

% side (right) u(x,Ny,t_k+1):
for ix=2:Nx-1
    kB=dtdx*velMatrix_B(ix,Ny);
    u_kkB(ix,Ny)=u_kB(ix,Ny) - kB*(u_kB(ix,Ny)-u_kB(ix,Ny-1));
end


%time-step update:
u_koB = u_kB;
u_kB = u_kkB;

%measure the generated sourcefield
```

```
   source(t_k)=u_kkB(Xmid,Ymid);

 end %end time loop
```

6.5.2 Programs for reading data files

```
function [x,y,N] = readData(filename)

file = fopen(filename,'r');
%number of recievers
line = fgetl(file); %first line in file
N = str2double(line);
x = zeros(N,1);
y = zeros(N,1);
for n=1:N
   line = fgetl(file);
   xy = split(line,' ');
   x(n) = str2double(xy(1));
   y(n) = str2double(xy(2));

end
fclose(file);


function [x,y,N] = readScatterer(file)
%READPERTCOORDS Summary of this function goes here
%   Detailed explanation goes here
file = fopen(file,'r');

%pertubations
line = fgetl(file); %first line in file
N = str2double(line);

x= zeros(N,1);
y= zeros(N,1);
for n=1:N
   line = fgetl(file);
   xy = split(line,' ');
   x(n) = str2double(xy(1));
   y(n) = str2double(xy(2));
end

fclose(file);
```

```
function [xp,yp] = readPertCoords(file)
%READPERTCOORDS Summary of this function goes here
%   Detailed explanation goes here
file = fopen(file,'r');

%pertubations
line = fgetl(file); %first line in file
Np = str2double(line);

xp= zeros(Np,1);
yp= zeros(Np,1);


for n=1:Np
   line = fgetl(file);
   xy = split(line,' ');
   xp(n) = str2double(xy(1));
   yp(n) = str2double(xy(2));
end

fclose(file);

function [fo,Nx, Ny, snapshots, Nt] = readParameters(paramfile)
%open file for reading
file = fopen(paramfile,'r');

%read each line and extract value
line = fgetl(file);
fo = str2double(line);
line = fgetl(file);
Nx = str2double(line);
line = fgetl(file);
Ny = str2double(line);
line = fgetl(file);
Nt = str2double(line);

line = fgetl(file);
line = fgetl(file);
Nsnap = str2double(line);

snapshots = zeros(1,Nsnap);

for n=1:Nsnap
```

```
    line = fgetl(file);
    snapshots(n)=ceil(str2num(line));%rounds up to integervalue
end                                  %if nessecary
fclose(file);

function [sourceMatrix,Ns,sourcex,sourcey] = readSource(sourcefile,Nx,Ny)
% matrix has positions (i,j) defined by number of steps
% (i*dx,j*dy)
%
% Note that the Matrix has step-coordinates only, not Source-pulse!!

sourceMatrix = zeros(Nx,Ny);


file = fopen(sourcefile,'r');

%number of sources
line = fgetl(file); %first line in file
Ns = str2double(line);

sourcex = zeros(Ns,1);
sourcey = sourcex;
for n=1:Ns
   line = fgetl(file);
   xy = split(line,' ');
   i = str2double(xy(1));
   j = str2double(xy(2));
   sourceMatrix(i,j) = 1;
   sourcex(n)=i;
   sourcey(n)=j;
end

fclose(file);

function [xrec,yrec,Nrec] = readRecievers(filename)
%READRECIEVERS Summary of this function goes here
%   Detailed explanation goes here

file = fopen(filename,'r');

%number of recievers
line = fgetl(file); %first line in file
Nrec = str2double(line);
```

```
xrec = zeros(Nrec,1);
yrec = zeros(Nrec,1);

for n=1:Nrec
   line = fgetl(file);
   xy = split(line,' ');
   xrec(n) = str2double(xy(1));
   yrec(n) = str2double(xy(2));

end

fclose(file);
```

6.5.3 Examples of data-files

```
Parameter file:
30
240
240
500

0
60
80
100
140
180
200
220
240
280
300
350
191
383

DISCRIPTION of the above:

#centralfrequency of source
# size of length-unit (dx)
# of x-coordinates
# of y-coordinates
```

```
# of timesteps

# of snapshots (Nsnap)
n1 in snapshot 1 (n1*dt)
n2 in snapshot 2 (n2*dt)
...
...
nNsnap in nNsnap*dt

scatterer data-file:
1
114 132


Description:

number of pertubations
number of x-steps  number of y-steps

receiver data-file:
25
48 54
54 54
60 54
66 54
72 54
78 54
84 54
90 54
96 54
102 54
108 54
114 54
120 54
126 54
132 54
138 54
144 54
150 54
156 54
162 54
168 54
174 54
```

```
180 54
186 54
192 54

Description:
# of recievers
# n m  in number of steps n*dx m*dy


source data-file:
23
48 60
48 66
48 72
48 78
48 84
48 90
48 96
48 102
48 108
48 114
48 120
48 126
48 132
48 138
48 144
48 150
48 156
48 162
48 168
48 174
48 180
48 186
48 192


first integer defines number of coordinates.

Following integers are x,y positions, thiese are not
the actual coordinate, but number of steps in x and y directions.
Sources cannot be placed in grid for values < 2

velocity data-file:
```

```
500
1.5

velocity of background
factor(pT) of perturbation (pT*backgorundvelocity)
```

References

[1] Devaney A.J and Lehman S.K., 2003. Transmission mode time-reversal super-resolution imaging, J. Acoust. Soc. Am **113** (5), 2742-2753.

[2] Fink M., November 1999. Time-reversed acoustics." Scientific American Magazine.

[3] Berkhout A.J., 1982. Seismic migration. Imaging of acoustic energy by wave field extrapolation. A. Theoretical aspects. Developments in Solid Earth Geophysics 14A. Elsevier Scientific Publishing Company.

[4] Shewell J.R. and Wolf E., 1968. Inverse Diffraction and a New Reciprocity Theorem, J. Opt. Soc. Am. **58**, 1596-

[5] Johnson D.H. and Dudgeon D.E., 1993. Array signal processing. 1^{st} edition, Prentice Hall signal processing series.

[6] Bleistein N.B., 1984. Mathematical methods for wave phenomena. Academic Press.

[7] Newton R.G., 1982. Scattering Theory of Waves and Particles. 2^{nd} edition, Springer, New York.

[8] Born M., 1933. Optics. Springer Verlag, New York.

[9] Prada P., Thomas J-L. and Fink M., 1995. The iterative time reversal process: Analysis of the convergence, J. Acoust. Soc. Am **97** (1), 62-71.

[10] Gelius, L.J., 2008. Personal communications.

[11] Gelius, L.J., 2007. Time-reversal in case of two transceiver arrays - a generalized D.O.R.T. method. Unpublished note.

[12] Claerbout, J.F., 1985. Imaging the Earth's Interior. Blackwell Scientific Publications.

[13] Kelly K.R., Ward, R.W., Treitel S. and Alford R.M., 1976. Synthetic seismograms: A finite-difference approach. Geophysics **41**, 2-27.

[14] Engquist B. and Majda A., 1979. Radiation boundary conditions for acoustic and elastic waves. Communications on Pure and Applied Mathematics **32**, 313-320.