NTNU – Trondheim
Norwegian University of
Science and Technology

Fast Algorithm for Simulation of Signals in Medical Ultrasound Blood Flow Imaging

You Zhou

Medical Technology
Submission date: June 2012
Supervisor: Ilangko Balasingham, IET

Norwegian University of Science and Technology
Department of Electronics and Telecommunications

Fast Algorithm for Simulation of Signals in Medical Ultrasound Blood Flow Imaging

Master Thesis
You Zhou

Department of Circulation and Medical Imaging,
Norwegian University of Science and Technology (NTNU)
14/June/2012

Preface

This master thesis documents the project work in my last semester. It is part of my master degree in Medical Technology with Specialization in Signal Processing and Imaging at NTNU. The work has been carried out at the Department of Circulation and Medical Imaging.

I would like to thank my supervisors Prof. Hans Torp, Ph. D student Tonje D. Fredriksen and Ph. D student Bastien Denarie. During this semester, they have dedicated a lot of time to helping me in every way they can. I did learn a lot from them. I also appreciate the company SONOWAND AS for letting me borrow their string phantom.

Abstract

The commonly used simulation method Field II, which is based on the spatial impulse response approach, has excellent accuracy in linear domain. However the computational time can be up to many days for one simulation. One of the solutions to this problem is a convolution-based methodology called COLE. It is much faster than Field II and has very good approximation. It generates the data by reducing multi-dimensional convolution model to multiple single-dimensional convolutions.

This thesis is about implementing COLE on the FieldSim 3 platform and using it for blood flow imaging. This platform is written in MATLAB with object-oriented programming and it is now under development at department of circulation and medical imaging.

Both Field II and real scanner have been used to compare with COLE. The simulated phantom for both simulators was a straight tube with scatterers moving inside, whereas a string phantom was used to get the data from the scanner. The computational time of COLE with 2D Doppler mode scan in FieldSim 3 achieved 85 times faster than Field II. The plotted PW Doppler spectra and the 2D power spectra showed that the velocity resolutions of both simulators were at the same level. COLE had higher noise floor than Field II and scanner in Doppler mode scan. COLE had relatively high sampling frequency requirement compared with Field II. If the sampling frequency was not high enough, COLE would produce side lobes in the PW Doppler spectra.

Table of contents

1 Introduction

Ultrasound has been used in medical diagnosis for about half a century. It has now become one of the most important and popular methods to image the human body. Compared with the other imaging modalities (Computed tomography (CT) and Magnetic Resonance Imaging (MRI) etc.) in medical diagnosis, its advantages are that it provides real-time imaging and it is safe for the patients. The modern ultrasound machines are sophisticated signal processing machines. The prices of the medical ultrasound machines are usually much less expensive than the other imaging machines mentioned above, because the way that they used to image are mainly electronic.

In the research and study of medical ultrasound, computer simulations are widely used. They are usually written in different languages by the researchers for different purposes. Some of them are powerful at some aspects, but weak at others. This is the reason why different simulation tools are developed.

FieldSim 3 is a unified ultrasound simulation platform which is under development at Department of Circulation and Medical Imaging at NTNU. It is written in MATLAB with Object-Oriented Programming and contains default setups for scanner configurations and typical ultrasound probes which can be found in modern ultrasound systems. One of the advantages of FieldSim 3 is that different simulators can be used in it with the same setup. In this project, two different simulators COLE and Field II are implemented in this platform to generate the simulated data for simple flow examples (e.g. uniform velocity field). The simulated phantoms are modeled as point scatterers flowing in a straight tube. The simulations are also verified by in vitro experiments with ultrasound machine.

Field II is a commonly used simulator which is based on the spatial impulse response method. It has typically been used as the ‘gold standard’ due to its perfect accuracy in linear domain [1]. However, the computation time is a problem when the number of scatterers becomes very large. COLE is one of the solutions to this problem. It is a new convolution based simulator, which is proposed by H. Gao et al. for simulating 2-D/3-D cardiac ultrasound images [2]. It is much faster than the spatial impulse response method and has good approximation to it. The detailed comparison is given in the following chapters.

The cardiovascular system is different from the other parts of the human body, since the heart and blood are moving all the time. The Doppler Effect is used to solve this problem. The most frequently used techniques are Continuous Wave (CW) Doppler, Pulsed Wave (PW) Doppler and Color Flow (CF) Doppler. They are all used to measure the blood flow by estimating the blood velocity. However, they are used for different purposes. The first two are usually used to get the velocity spectrum. It provides a detailed relationship between time and velocity. The last one gives better visualization of blood flow in heart or blood vessels. In practice, these methods are supplementary to each other.

In this report, the detailed theory is explained in the coming chapter. In chapter 3, the methods that have been used in the project are presented. The simulation and vitro test results are shown in the results chapter. In discussion, the two different simulation methods and experiment with their results are analyzed and compared. Finally, the conclusion and the future work suggestion are given in the last chapter.

2 Theory

2.1 Ultrasound imaging

As the ultrasound systems are getting more complicated and powerful, machines made by different companies are usually somehow different. However the basic techniques are still similar. Most ultrasound machines provide B-mode imaging, M-mode imaging and Doppler imaging.

2.1.1 Basic ultrasound theory

The ultrasound machines usually generate sound waves with frequency from 2 MHz to 15 MHz, which have a speed around 1540m/s in the human body. Some modern machines use higher frequencies for different purposes. With higher frequencies the range resolution gets better, while the attenuation becomes larger. This is because the attenuation in human tissue is directly proportional to the transmit frequency. Due to this trade off, frequencies in the range of 2.5 – 5 MHz is suitable for adult cardiology and imaging of deep organs [3].

Usually the machine has different kinds of probes connected. They have different properties and are used to image different parts of the body. There are linear arrays, curvilinear arrays, phased arrays and annular arrays. For 3D imaging, the two dimensional matrix arrays are used. To image the heart the phased array is more suitable while the linear array is better to measure blood flow in the blood vessels near the surface. In project, the linear array was used. It is shown in Fig. 2.1 [4]. The linear array acquires a rectangular image. The image is constructed by moving the beam over the imaging region. Each beam is generated by firing pulses from several elements and using electronic focusing as shown in Fig. 2.2 [4].

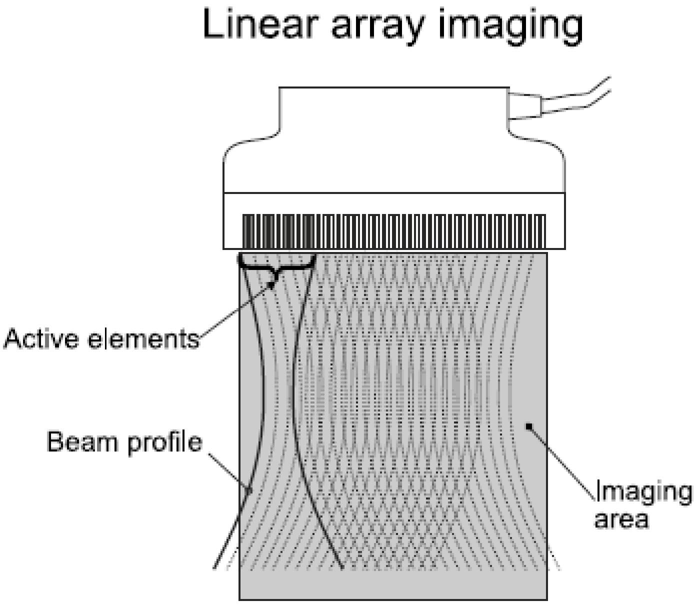


Fig. 2.1: Illustration of scan sequence in a linear array transducer. Each beam is generated be a number of active elements.

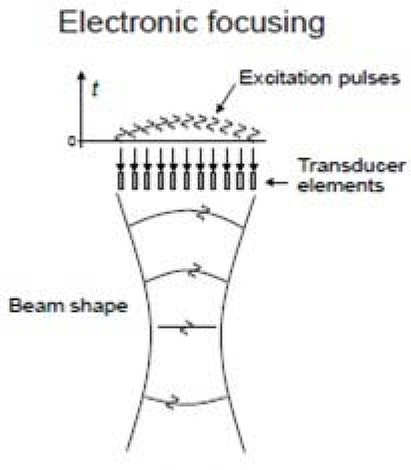


Fig. 2.2: Illustration of electronic focusing in a linear array transducer. By adding delays to each pulse, the plane wave can be achieved at the focus.

The probe is used for both transmitting and receiving ultrasound waves. In tissue imaging, the transducer sends short pulses through the tissue. At each interface of different tissues, part of the pulse keeps going forward and another part gets reflected, as shown in Fig. 2.3. By measuring the time that it takes for the pulse to propagate from transmit to receive, the tissue can be imaged. There are a lot of effects that will introduce some noise during imaging, for instance reverberation, aberration, grating lobes and side lobes. They should be avoided or reduced during the tissue imaging. The range resolution, which is the ability that one can resolve two close targets, is determined by the transmit pulse length in equation (2.1)

$$\Delta \mathrm{r} = \frac{1}{2} \mathrm{c} \cdot T_p \tag{2.1}$$

where c is the wave velocity and T_p is the pulse length.

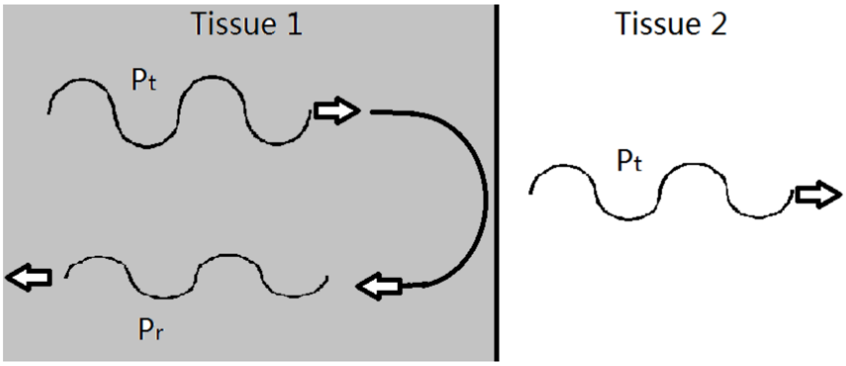


Fig. 2.3 [3]: Wave propagation at the interface of two different tissues. Part of the transmitted wave keeps transmitting through the boundary and part of it is reflected at the boundary.

The F#, given in equation (2.2), is an important value for calculating the beam width,

$$\mathrm{F\#} = \frac{\mathrm{F}}{\mathrm{D}} \tag{2.2}$$

where F is the focal depth, D is the aperture diameter, shown in Fig. 2.4 The beam width in focus is equal to

$$\mathrm{D_F} = \mathrm{F\#} \cdot \lambda \tag{2.3}$$

where λis the wavelength.

In the focal plane, the Fourier transform of the aperture can be used to estimate the lateral resolution. For example, the Fourier transform of a rectangular aperture gives a sinc function.

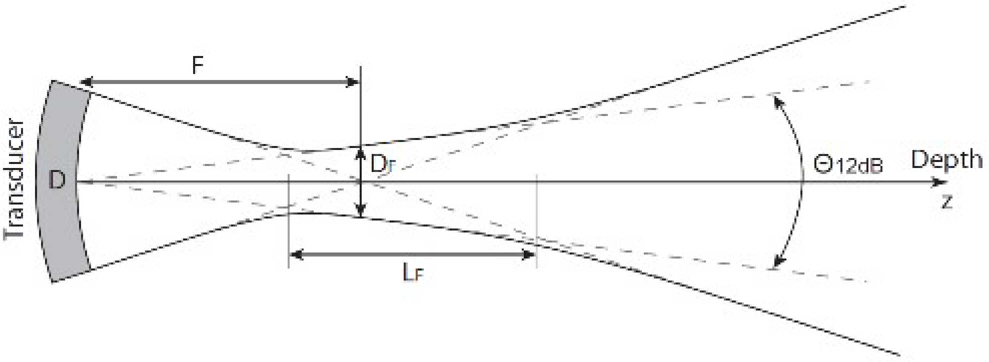


Fig. 2.4: The focused beam profile with aperture diameter D, focal depth F. The figure is taken from a presentation by Lasse Løvstakken and Hans Torp.

2.2 *Ultrasound fields and Field II*

In this section, the linear ultrasound field description is given by using the spatial impulse response method. It is also the method that Field II [5] [6] uses to generate the simulation data.

Field II is a program that consists of a C program and a number of MATLAB m-functions that calls this program. All calculations are performed by the C program, and all data is kept by the C program. Three types of m-functions are found. They are used for initializing the program, defining and manipulating transducers and performing calculations [7]. The theory presented in this section is based on notes by J. A. Jensen [4].

2.2.1 Linear acoustic model

In electrical engineering, a linear system can be characterized by its impulse response

$$y(t) = \int_{-\infty}^{+\infty} h(\tau)\, x(t-\tau)d\tau \tag{2.4}$$

where h(t) is the impulse response of the system, and y(t) and x(t) are the output and input signals of the system respectively as shown in Fig. 2.5.

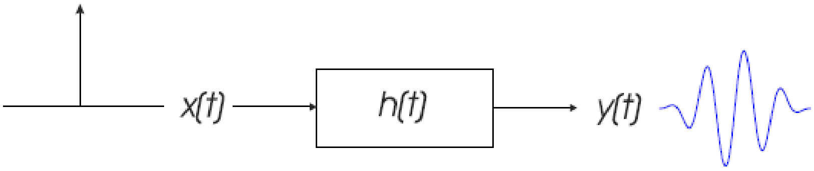


Fig. 2.5: The impulse response of a linear system with a delta input function [4].

Similarly it can be applied to a linear acoustic system as shown in Fig. 2.6. The transducer on the left can be seen as an infinite rigid baffle at position $\vec{r_2}$. Assume that the homogeneous medium with density ρ_0 is a linear system. The transducer generates the sound wave with a constant speed c. A hydrophone placed on the right at position $\vec{r_1}$ is used to measure the acoustic pressure from the transducer. If the transducer generates a delta function, the measured pressure at $\vec{r_1}$ is the acoustic impulse response of this specific system. The impulse response changes when the transducer or the hydrophone is moved spatially to the other place. Therefore, the impulse response is called spatial impulse response and it depends on the relative position of both transducer and hydrophone $(\vec{r_2} - \vec{r_1})$.

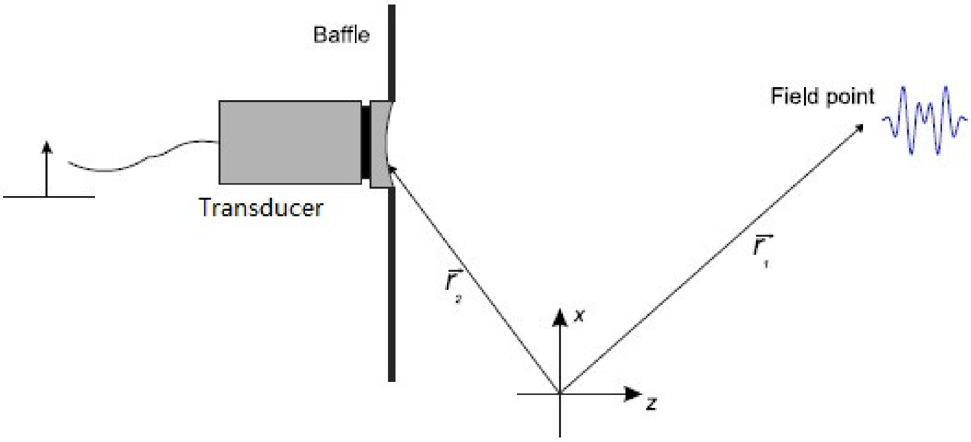


Fig. 2.6: A linear acoustic system. The system is defined by the impulse response and the generated electric pulse. The figure is taken from [4].

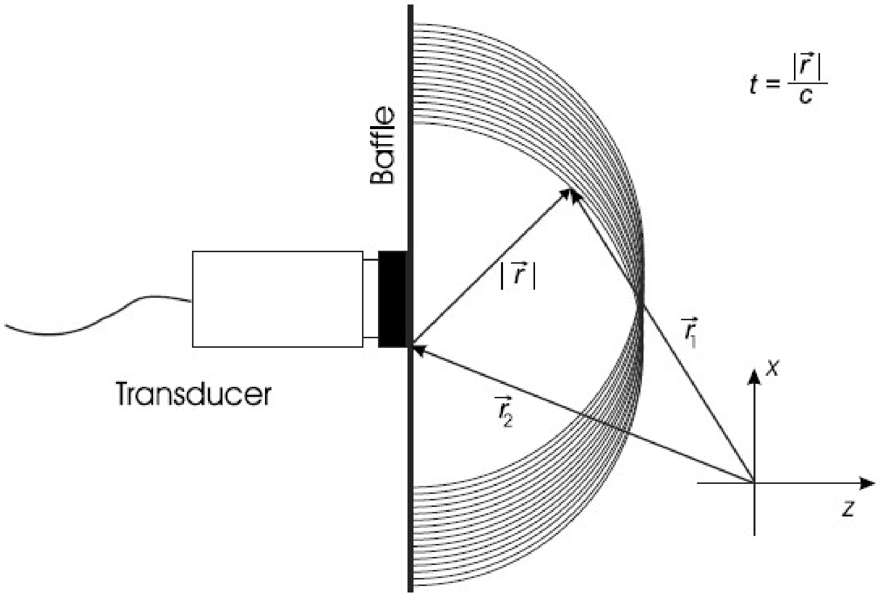


Fig. 2.7: Illustration of Huygens' principle for a fixed time instance. A spherical wave with a radius of $|\vec{r}|= ct$ is radiated from each point on the aperture. The figure is taken from [4].

As illustrated in Fig. 2.7, by applying Huygens' principle in which every point on the radiating surface is the origin of an outgoing spherical wave, a perception of the sound field for a fixed time instance can be obtained. Each of the outgoing spherical waves is given by

$$p_s(\vec{r_1}, t) = \delta\left(t - \frac{|\vec{r_2} - \vec{r_1}|}{c}\right) = \delta\left(t - \frac{|\vec{r}|}{c}\right) \tag{2.5}$$

where $\vec{r_1}$ is the point in space, $\vec{r_2}$ is the point on the transducer surface and t is the propagation time. Then the spatial impulse response can be found by summing up all the spherical waves passing the observation point with distance $|\vec{r}|$.

2.2.2 *Spatial impulse response*

Assume that a triangular shaped aperture is placed in an infinite rigid baffle, on which the velocity normal to the plane (not including the aperture) is zero. The pressure field generated by the aperture at $|\vec{r_1}|$ can be found by the Rayleigh integral [8]

$$p(\overrightarrow{r_1},t) = \frac{\rho_0}{2\pi}\int_S \frac{\frac{\partial v_n(\overrightarrow{r_2}, t - \frac{|\overrightarrow{r_2} - \overrightarrow{r_1}|}{c})}{\partial t}}{|\overrightarrow{r_2} - \overrightarrow{r_1}|} dS \qquad (2.6)$$

where v_n is the velocity normal to the transducer surface. In this integral, the medium is assumed to be linear, homogeneous and without propagation attenuation.

In order to avoid reverberation in further derivation, the aperture is assumed to be flat. The integral in (2.6) can be written as

$$p(\overrightarrow{r_1},t) = \frac{\rho_0}{2\pi} \frac{\partial \int_S \frac{v_n\left(\overrightarrow{r_2}, t - \frac{|\overrightarrow{r_2} - \overrightarrow{r_1}|}{c}\right)}{|\overrightarrow{r_2} - \overrightarrow{r_1}|} dS}{\partial t} \qquad (2.7)$$

The relationship between the pressure p and velocity potential are shown as follows [3]

$$\vec{v}(\vec{r},t) = -\nabla\psi(\vec{r},t)$$
$$p(\vec{r},t) = \rho_0 \frac{\partial\psi(\vec{r},t)}{\partial t} \qquad (2.8)$$

By using this relationship, (2.7) can be written as

$$\psi(\overrightarrow{r_1},t) = \int_S \frac{v_n(\overrightarrow{r_2}, t - \frac{|\overrightarrow{r_2} - \overrightarrow{r_1}|}{c})}{2\pi|\overrightarrow{r_2} - \overrightarrow{r_1}|} dS \qquad (2.9)$$

Since the generated pulse can be written in a time convolution with a delta function as in Equation (2.4), the Equation (2.9) can be written as

$$\psi(\overrightarrow{r_1},t) = \int_S \int_T \frac{v_n(\overrightarrow{r_2},\tau)\delta(t - \tau - \frac{|\overrightarrow{r_2} - \overrightarrow{r_1}|}{c})}{2\pi|\overrightarrow{r_2} - \overrightarrow{r_1}|} d\tau dS \qquad (2.10)$$

Now assume that the velocity is the same over the aperture, i.e. independent of $\overrightarrow{r_2}$, then (2.10) can be further derived to

$$\psi(\overrightarrow{r_1},t) = v_n(t) * \int_S \frac{\delta(t - \frac{|\overrightarrow{r_2} - \overrightarrow{r_1}|}{c})}{2\pi|\overrightarrow{r_2} - \overrightarrow{r_1}|} dS \qquad (2.11)$$

Let

$$h(\vec{r_1},t) = \int_S \frac{\delta(t - \frac{|\vec{r_2} - \vec{r_1}|}{c})}{2\pi|\vec{r_2} - \vec{r_1}|} dS \qquad (2.12)$$

Then

$$\psi(\vec{r_1},t) = v_n(t) * h(\vec{r_1},t) \qquad (2.13)$$

$*$ denotes the convolution and (2.12) is called spatial impulse response.

The pressure field can now be written as

$$p(\vec{r},t) = \rho_0 \frac{\partial v_n(t)}{\partial t} * h(\vec{r_1},t) \qquad (2.14)$$

It equals the emitted pulse pressure for all kinds of surface vibration $v_n(t)$, which makes it possible to find all ultrasound fields of interest by using the derivation presented in this section.

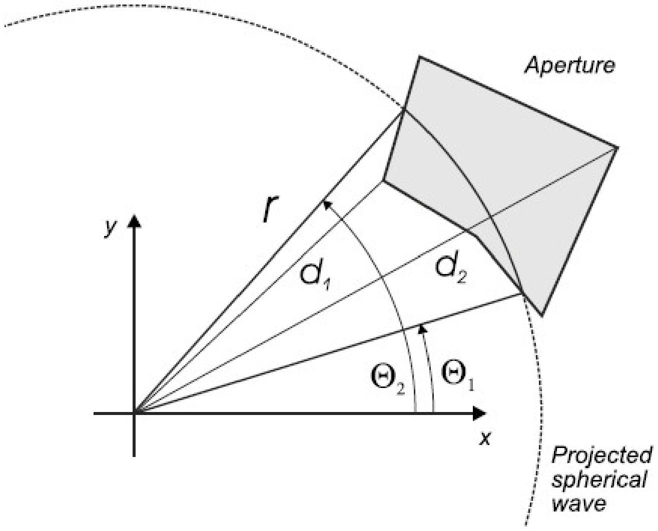


Fig. 2.8: Distances and angles in the aperture plane for evaluating the Rayleigh integral. d1 and d2 are the closest and farthest projected distances determined by the aperture. Θ_1 and Θ_2 are the corresponding angles for a given time. r is the radius of the projected circle. The figure is taken from [4].

The calculation of the impulse responses can be done by projecting the field point onto the plane which has an intersection with the aperture as, shown in Fig.

2.8, then find the intersection.

The spatial impulse response in (2.12) can be written with polar coordinates as

$$h(\overrightarrow{r_1}, t) = \int_{\Theta_1}^{\Theta_2} \int_{d_1}^{d_2} \frac{\delta(t - \frac{R}{c})}{2\pi R} r\, dr\, d\Theta \qquad (2.15)$$

where R is the distance from the field point to the aperture, which is $|\vec{r}|$ in the previous figure.

Substitute t' $= R/c$ and $RdR = rdr$, the Equation (2.15) can be written as

$$h(\overrightarrow{r_1}, t) = \frac{c}{2\pi} \int_{\Theta_1}^{\Theta_2} \int_{d_1}^{d_2} \delta(t - t^{'}) dt^{'} d\Theta \qquad (2.16)$$

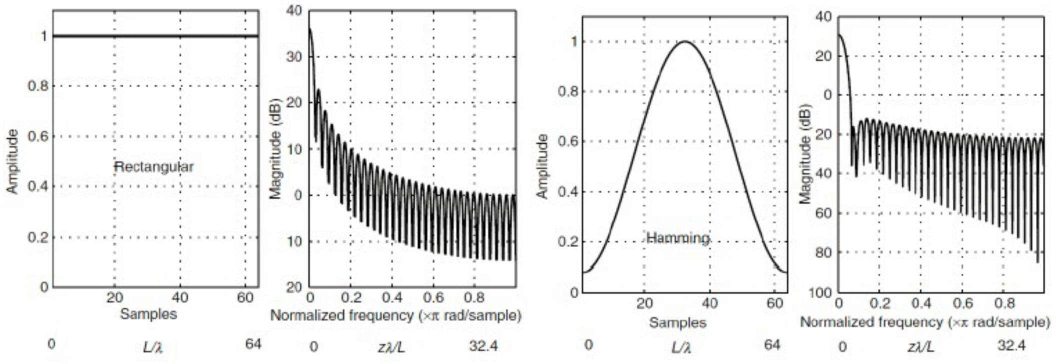
By giving a specific time instance, the contribution along the arc is a constant value and the integral gives

$$h(\overrightarrow{r_1}, t) = \frac{c(\Theta_2 - \Theta_1)}{2\pi} \qquad (2.17)$$

When no apodization is used, the spatial impulse response can be found by keeping track of the intersections as a function of time. The detailed calculation procedure and specific solution cases can be found in [4].

2.2.3 Apodization application

Apodization technique refers to amplitude weighting of the velocity distribution over the transducer aperture. It is used to reduce side lobes in the emitted sound field meanwhile the main lobe becomes wider. Fig. 2.9 (a) and (b) shows the same rectangular aperture and its frequency response before and after using the Hamming window apozidation respectively.



(a)

(b)

Fig. 2.9: Apodization technique applied on a rectangular aperture. (a) shows the rectangular aperture without using apodization, (b) shows the Hamming window apodization used on the same aperture. The figure is taken from a presentation by Lasse Løvstakken and Hans Torp.

The apodization function $a(r, \Theta)$ over the aperture can be introduced to Equation (2.15)

$$h(\overrightarrow{r_1}, t) = \int_{\Theta_1}^{\Theta_2}\int_{d_1}^{d_2} a(r, \Theta)\frac{\delta(t - \frac{R}{c})}{2\pi R} r\, dr\, d\Theta \tag{2.18}$$

Then the same substitution as previous subsection can be used, and Equation (2.18) can be derived to

$$h(\overrightarrow{r_1}, t) = \frac{c}{2\pi}\int_{\Theta_1}^{\Theta_2} a_1(t, \Theta) d\Theta \tag{2.19}$$

where $a_1(t', \Theta) = a(\sqrt{(ct')^2 - z_p^2}, \Theta)$, t' is the same as in Equation (2.16) and z_p is the field point height above the x-y plane of the aperture shown in Fig. 2.8.

2.3 Convolution models and COLE

An easy and fast methodology, that has been used extensively, is the so-called convolution model [9]–[11]. It is based on the assumption of a space-invariant point spread function and typically results in a linear image [2]. In 1-D implementations the point spread function of the image is spatially convolved with a Dirac-train which represents the position and the reflectivity of discrete scatterers along the image line [2]. 2-D or 3-D simulations can be done by extending the convolution to 2D or 3D respectively. Fig. 2.10 shows a conventional 2-D convolution model.

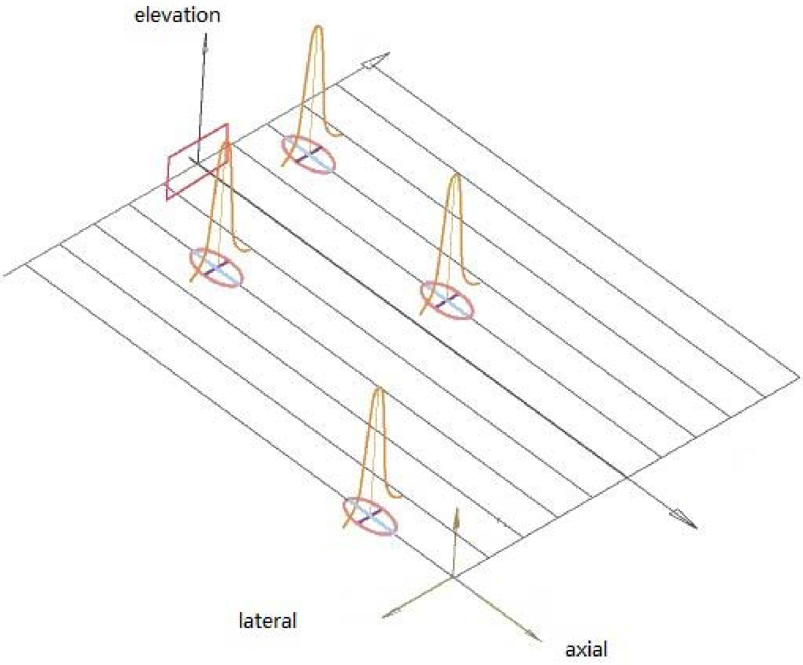


Fig. 2.10: A 2-D convolution model applied in the rectangular scan sequence. The figure is taken from a presentation of Jan D’hooge.

COLE is a fast convolution-based methodology, proposed by H. Gao et al [2], which uses multiple 1-D convolutions instead of a conventional 2-D or 3-D convolution. It produces simulated data sets by convolving the transmitted ultrasound pulse with the projected amplitudes of all the scatterers along one image line in the spatial domain [12]. For example, the 3-D RF signals have a pyramidal format $\mathrm{I}(\mathrm{r}, \theta, \varphi)$ in polar coordinates. Using the new convolution model each individual image line, the RF signal, is calculated as a 1-D convolution and has format $\mathrm{I}(\mathrm{r}, \theta_{\mathrm{m}}, \varphi_{\mathrm{n}})$, as

$$\mathrm{I}\left(\mathrm{r}, \theta_{\mathrm{m}}, \varphi_{\mathrm{n}}\right) = \mathrm{H}(\mathrm{r}) * \mathrm{T}\left(\mathrm{r}, \theta_{\mathrm{m}}, \varphi_{\mathrm{n}}\right) \tag{2.20}$$

and

$$\mathrm{T}\left(\mathrm{r}, \theta_{\mathrm{m}}, \varphi_{\mathrm{n}}\right) = \sum_{\mathrm{q}=1}^{\mathrm{N}} \omega_{\mathrm{q}} \mathrm{a}_{\mathrm{q}} \delta(\mathrm{r} - \mathrm{r}_{\mathrm{q}}) \tag{2.21}$$

where θ_m and φ_n indicate the lateral location and the elevation location respectively, $H(r)$ is the axial point spread function, $T(r, \theta_m, \varphi_n)$ is the scatterer distribution function, ω_q is the projection factor for the qth scatterer of total number N and a_q is the echogenicity of the qth scatterer.

Assume that there are M axial samples along the image line. Each projection r_q should be attributed to the closest sample r_i taken along the image line, where i=1, 2, …, M. Define the voxel V_i as the spatial interval between 2 subsequent radial samples r_{i-1} and r_i. If multiple projections fall within the same voxel, the summation of these amplitudes is attributed to the radial sample. Then the final sampled version $T(r_i, \theta_m, \varphi_n)$ of $T(r, \theta_m, \varphi_n)$ is

$$T(r_i, \theta_m, \varphi_n) = \sum_{q=1}^{N} \omega_q a_q \delta(r - r_q)|_{r_q \in V_i} \tag{2.22}$$

Equation (2.22) can be represented by convolution of the scatterer distribution function with a bloc-function R(Δr) before sampling

$$T(r_i, \theta_m, \varphi_n) = \left[T(r, \theta_m, \varphi_n) * R(\Delta r)\right] * \sum_{i=1}^{M} \delta(r - r_i) \tag{2.23}$$

where Δr the spatial sampling distance alone the image line.

When Δr is sufficiently small, $T(r, \theta_m, \varphi_n)$ in Equation (2.20) can be replaced by $T(r_i, \theta_m, \varphi_n)$.

The process described above can be repeated to each image line to get the RF signal of the image.

In order to see how COLE weights the contribution of the scatterers with the distance from the image line, Fig. 2.11 uses a 2-D rectangular scan sequence to demonstrate. In this case the image line is in the middle. It shows that the closer the scatterer is to the image line, the bigger will the amplitude of the projection be on the line.

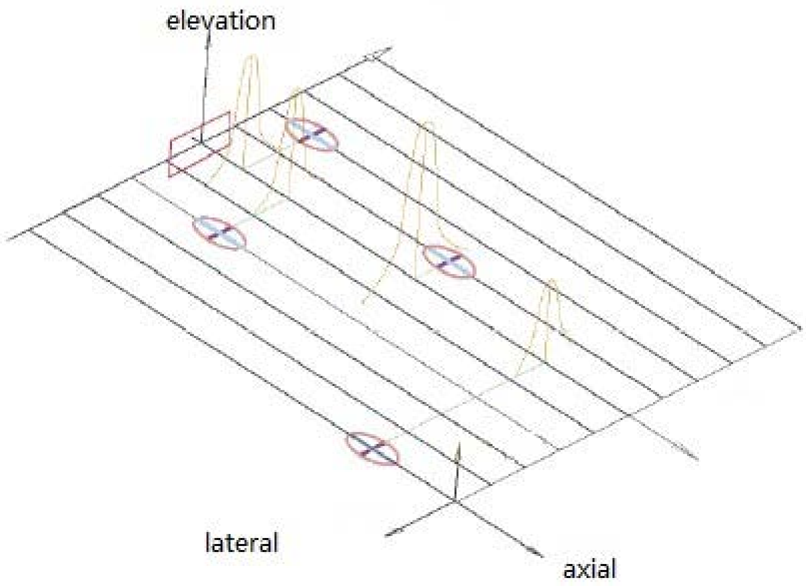


Fig. 2.11: A 2-D rectangular scan sequence with scatterers with different distance from the image line and project on the line with the Cole method. The figure is taken from a presentation of Jan D’hooge.

There are 3 ways to define the weighting of ω_q:

1. From the Gaussian point spread function of the beam profile, which is defined as:

$$\omega_q = \exp\left\{-\frac{1}{2}\left[\frac{r_q^2(\theta_q-\theta_m)^2}{\sigma_L^2}+\frac{r_q^2\left(\varphi_q-\varphi_n\right)^2}{\sigma_E^2}\right]\right\} \qquad (2.24)$$

2. From a simulated beam profile look up table (LUT) by its axial distance, its lateral angle and its elevation angle.
3. From a measured beam profile LUT by the same indices as the second one.

2.4 Doppler mode imaging

The Doppler mode is used to measure the blood flow in the body. It utilizes the property that the blood is moving. If the scatterers are moving, the frequency of the back scattered signal will be altered from the transmitted frequency. This change in frequency is a phenomenon called the Doppler Effect, and can be used to measure the velocity of the moving scatterers [13]. The Doppler shift f_d can be measured by

$$f_d = -2f_0 \frac{v \cdot \cos\varphi}{c} \tag{2.25}$$

where f_0 is the transmit pulse frequency,v is the blood velocity and φ is the angle between the blood velocity vector and the transmitted beam.

In PW Doppler mode, the transducer transmits an ultrasound pulse. Therefore the range resolution can be achieved by measuring the pulse echo time. The drawback of PW Doppler is the maximum velocity limit caused by frequency aliasing. The Doppler signal is actually sampled once for every pulse transmission, and the sampling frequency is hence equal to the pulse repetition frequency (PRF) of the ultrasound machine [3]. The highest velocity that can be measured, which is the Nyquist limit, is

$$v_{Nyq} = \frac{c \cdot PRF}{4f_0} \tag{2.26}$$

where c is the speed of the ultrasound wave, PRF is the pulse repetition frequency of the transmitted pulse and f_0 is the transmitted pulse frequency.

The ultrasound waves are received by the probe. The mechanical energy is converted to electric by piezoelectric material on the surface of the probe. After receive beamforming, the radio frequency (RF) signal is IQ demodulated. Doppler processing is one of the operations in the Mid- End processing shown in Fig. 2.12.

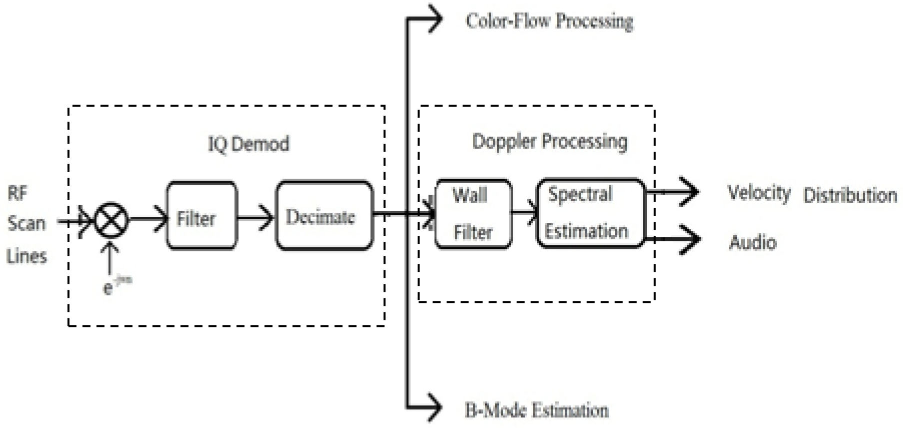


Fig. 2.12[14]: Mid-End processing of ultrasound system. Both IQ demodulation and Doppler signal processing are done in this stage.

2.4.1 Window function and filter

In the whole Mid-End processing stage, different window functions are applied. The window function is a mathematical function which is often used in signal processing to do spectral analysis, filter design, and beam forming. Two important parameters should be concerned when choosing a window, the width of the main lobe, the suppression of the side lobes. In Doppler estimation, windows with lower side lobes are usually used, like the Hamming window.

Except for window functions, digital filters are also commonly used in this stage. The clutter filter is a typical example. The Doppler spectrum usually contains clutter signal, blood signal and thermal noise, as shown in Fig. 2.13 (a) on next page. The clutter shown in the figure is signal from surrounding tissue due to beam side lobes and reverberations. It usually has much higher power than the blood signal, around 40-80 dB. But it has lower velocity than blood, in other words it has smaller Doppler shifts. Then by introducing a High Pass (HP) filter which is the clutter filter in Fig. 2.13 (b) on next page, the clutter component can be easily removed. Most of the filters used in Doppler processing are digital FIR (Finite-duration Impulse Response) filters. The FIR filter has a certain number of zeros and none poles in the transfer function. They are always stable and it is possible for them to have linear phase. An FIR filter of length M with filter coefficients b_k, input $x(n)$ and output $y(n)$ can be described by the difference equation as follows.

$$y(n) = \sum_{k=0}^{M-1} b_k x(n-k) \tag{2.27}$$

2.4.2 In-phase Quadrature (IQ) demodulation

In the ultrasound field, RF has a different meaning than in communication. Here it means the unprocessed data that come out from the beamformer. It is constructed by adding the signals from all the channels of the probe. IQ demodulation of the RF signal involves 3 steps. They are down mixing, LP filtering and decimation. The received RF signal from the transducer is a band pass signal. In Fig. 2.14 a transmit frequency of 2.5 MHz and a sampling frequency of 20 MHz is assumed. The negative part of the spectrum is a replica of the positive frequency. The parts are symmetric about zero frequency. This is because the RF signal is real valued. The Nyquist frequency indicates the upper limit of the spectrum.

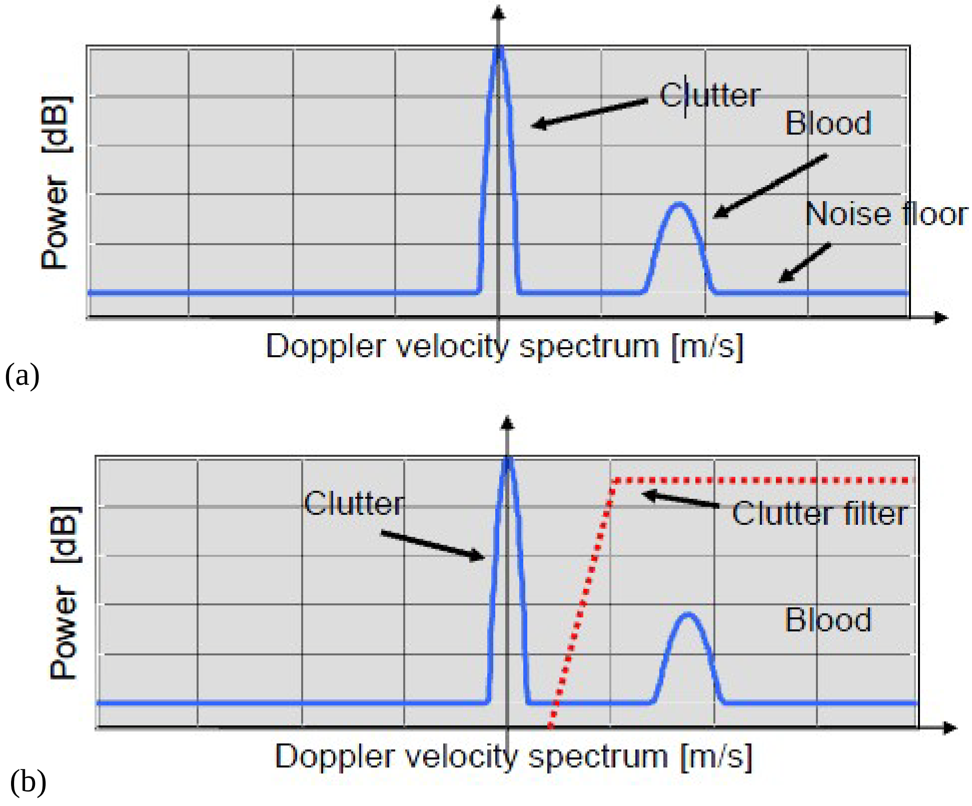


Fig. 2.13: Doppler signal components and clutter filter. This figure is taken from Hans Torp and Lasse Løvstakken’s presentation.

The first step (down mixing) is done by multiplying $e^{-j\omega t}$ with the RF signal. It makes the signal complex and gives signal information of velocity direction. As a consequence, the spectrum is now no longer symmetric about zero and both parts of the spectrum move down so that the central frequency of the positive part moves to zero. See Fig. 2.14 (b). The second step (low pass filtering) is shown in Fig. 2.14 (c). By applying a LP filter, the negative frequency part and the noise outside the bandwidth of interest can be removed. However, half of the energy of the RF signal is also removed. Then a factor $\sqrt{2}$ should be multiplied with the signal after LP filtering. Since the signal is complex, its bandwidth is equal to the complex sampling frequency. This makes it possible to reduce the sampling frequency. As shown in Fig. 2.14 (d), the cutoff frequency is ± 1.5 MHz. According to the Nyquist sampling theorem, the sampling frequency should be larger than twice of the largest frequency component, which is 1.5 MHz.

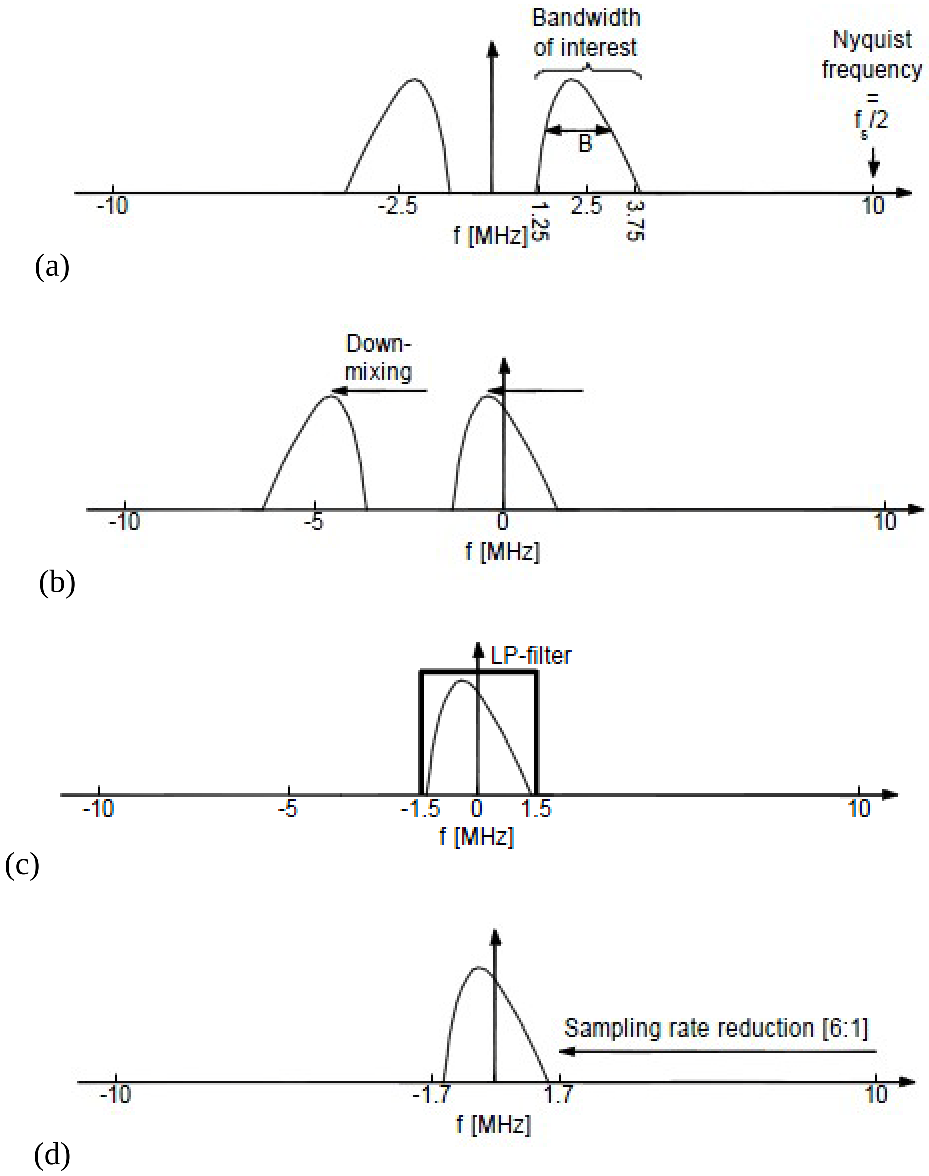


Fig. 2.14 [15]: Spectrum variation during IQ demodulation. The spectrum of RF signal is shown in (a), down-mixing of the RF signal is shown in (b), low pass filtered spectrum is shown in (c) and the IQ demodulated is given in (d).

2.4.3 Power spectrum estimation

The blood contains a large number of red blood cells. These cells moving with the blood can be considered as a large number of independent scatterers. The received RF signal, which is the sum of all the echoes from the scatterers, is a Gaussian bandpass process. Its complex envelop is complex Gaussian. Therefore, it is convenient to model the Doppler signal from moving blood as a complex

Gaussian random process, denoted as z(n), $\mathrm{n} \in (-\infty, +\infty)$. If the process is stationary, its autocorrelation function is

$$\mathrm{R(m)} =< \mathrm{z(n+m)^{*}z(n)} > \tag{2.28}$$

where $\mathrm{m} = 0, \pm 1, \pm 2, \ldots$ and *means complex conjugate.

The power spectrum of a stationary process z(n) (shown in Fig. 2.14 (c)) is defined as the Fourier transform of the autocorrelation function [3]

$$\mathrm{G}(\omega) = \sum_{1}^{m} R(m)e^{-j\omega m} \tag{2.29}$$

where $-\pi < \omega < \pi$.

The autocorrelation function equals to coefficients in Fourier series of $\mathrm{G}(\omega)$.

$$\mathrm{R(m)} = \frac{1}{2\pi}\int_{-\pi}^{\pi} G(\omega)e^{j\omega m}\, d\omega \tag{2.30}$$

The power spectrum of the Doppler signal represents the distribution of velocities within the blood vessel.

For a finite discrete signals z(k), k=$1,2, \ldots, \mathrm{N}$, the power spectrum $\mathrm{G}_{N}(\omega)$ where $-\pi < \omega < \pi$, can be expressed as

$$\mathrm{G}_{N}(\omega) = \frac{1}{N}|\mathrm{Z}_{N}(\omega)|^{2} \tag{2.31}$$

$$\mathrm{Z}_{N}(\omega) = \sum_{1}^{N} z(k)e^{-j\omega k} \tag{2.32}$$

The spectrum estimate $\mathrm{G}_{N}(\omega)$is called the periodogram. Introducing a window function $w_{N}(k)$ to the above equation (2.31) and (2.32), one can get

$$\mathrm{Z}_{N}(\omega) = \sum_{1}^{N} w_{N}(k)z(k)e^{-j\omega k} \tag{2.33}$$

The expected value is

$$< \mathrm{G}_{N}(\omega) >= \frac{1}{2\pi}\int_{-\pi}^{\pi} G(\omega - \varphi)|\mathrm{W}_{N}(\varphi)|^{2}d\varphi \tag{2.34}$$

$$\mathrm{W}_{N}(\varphi) = \sum_{1}^{N} w_{N}(k)e^{-j\omega k} \tag{2.35}$$

As explained in the previous section, a smooth window is usually used in this case, for example, a Hamming window.

2.4.4 Spectral analysis of Doppler signals

In PW Doppler, the spectral analysis is done on a 2D signal model by using the Fast Fourier Transform (FFT). See Fig. 2.15. The fast time indicates the range direction, while the slow time indicates the beam direction. FFT-based Doppler analysis has been used extensively for assessing various circulatory systems, and it is the current industry standard for Doppler ultrasound devices used for medical diagnostic applications [16]. Applying FFT on fast time gives the spectrum of the transmit signal, whereas FFT on slow time gives the Doppler shift, which can be used to calculate the velocity.

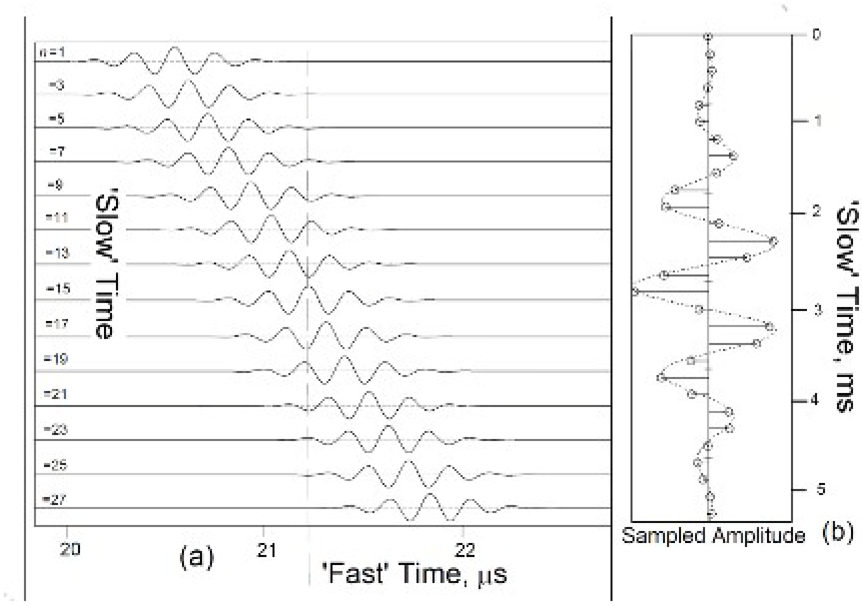


Fig. 2.15: 2D Doppler signal model. To the right is the sampled amplitude taken along the vertical line in left model. This figure is taken from Hans Torp and Lasse Løvstakken’s presentation.

2.4.5 2D power spectrum analysis

The algorithm of conventional PW Doppler requires only one range sample. In order to verify the simulation algorithm, it is necessary to see the correlation in the range direction. The 2D power spectrum is a good method to get this information.

The mathematical model described here is taken from a previous work by H.Torp et al. 1994 [17]. The received signal after IQ demodulation is described as a 2D Gaussian process $\mathrm{x(t,k)}$. According to the precious section, its complex pre-envelop signal is

$$\mathrm{x}_p(\mathrm{t,k}) = \mathrm{x(t,k)} \cdot e^{j\omega_0 t} \tag{2.36}$$

The signal is considered as a continuous function. Here, t is the pulse transmission time and k is the corresponding range depth $\mathrm{c}/2 * \mathrm{t}$.

The autocorrelation of the complex pre-envelop signal is

$$R_{x_p}(\tau, m) =< \mathrm{x}_p(\mathrm{t,k}) * \mathrm{x}_p(\mathrm{t+\tau,k+m}) > \tag{2.37}$$

where < > denotes the expected value.
The 2D power spectrum is the Fourier transform of the autocorrelation function

$$\mathrm{G}(\omega_1, \omega_2) = \sum_m \int R_{x_p}(\tau, m) e^{j\omega_1 T} e^{j\omega_2 mT} d\tau \tag{2.38}$$

where ω_1 and ω_2 are angular frequencies in the Fourier transform of fast time and slow time respectively.

Fig. 2.16 shows the 2D power spectrum. According to the theory explained before, ω_2 gives the Doppler shift and $\omega_1$is the frequency of transmitted pulse. ω_1 and ω_2 can be seen as f_0 and f_d in equation (2.25) respectively. Thus the equation (2.25) can be written as

$$\omega_2 = \alpha\omega_1 \tag{2.39}$$

where $\alpha = -2\frac{v \cdot \cos\varphi}{c}$.

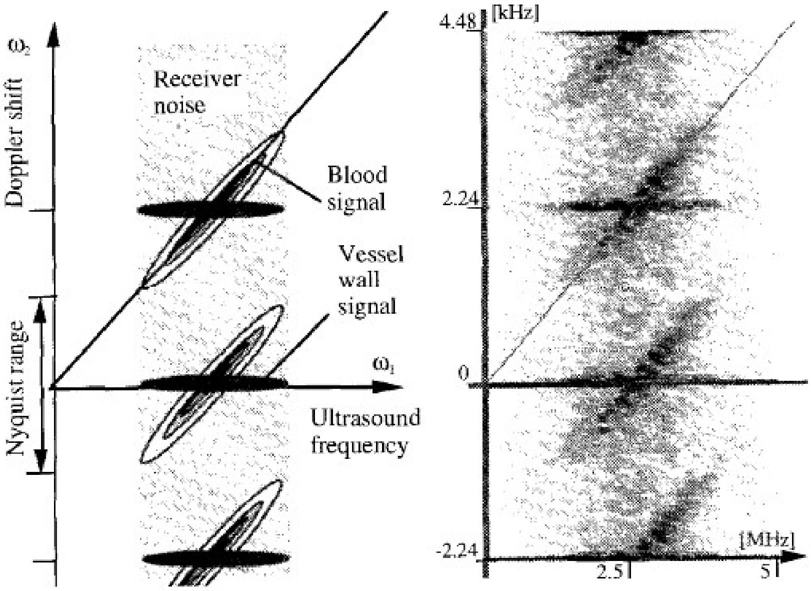


Fig. 2.16 [18]: 2D spectrum of a blood flow. To the left is an ideal model of the spectrum, to the right is a measured spectrum of a human artery.

The factor α indicates the slope of the line plotted through the spectrum in the figure. The bandwidth of ω_1 indicates the distribution of frequency components of the transmit pulse. It is inversely proportional to the pulse length and the sample volume size.

3 Simulations and experiments

To evaluate the convolution-based method, COLE simulator, both computer based simulations and string phantom experiments were used. 3 different constant velocities were used in both simulations and phantom experiments. Therefore, each velocity was simulated by both the Field II, and the COLE simulator and applied in the phantom experiment, where it was estimated by conventional PW Doppler. The computation times of the simulated data generated from both COLE and Field II were compared.

The string phantom experiments will be described in section 3.1. The experiment setup and the functions of the phantom will be shown and together with a description of the ultrasound scanner. Section 3.2 will describe the simulation approach. The FieldSim 3 platform will be shown as well as the simulators and the simulated phantom settings.

3.1 String phantom experiment

In order to verify the performance of COLE in the FieldSim 3 platform, a string phantom experiment was applied. The experimental setup is shown in Fig 3.1. A Vivid E9 GE Vingmed Ultrasound scanner and a GE linear array probe, 9L, were used in the experiment. The probe fixed straight downwards to the water. The azimuth direction of the probe was placed along the string. The string was placed in the water with a fixed angle to the beam direction. It moved with the wheels according to the motion of the electric motor. The string velocity had ± 1% accuracy of the stated velocity [19]. In this project three different constant velocities were recorded. The constant velocities were 0.15 m/s, 0.25 m/s and 0.50 m/s The ultrasound scan parameters are shown in Table 3.1

Table 3.1: The ultrasound scan parameters of the string phantom experiment

Transducer frequency	5MHz
IQ Sampling frequency	3.125MHz
Number of pulse period	6.5
PRF	5000
Start depth	15mm
End depth	30mm
Packet size	64
Number of beams	102
Ultrasound velocity	1540

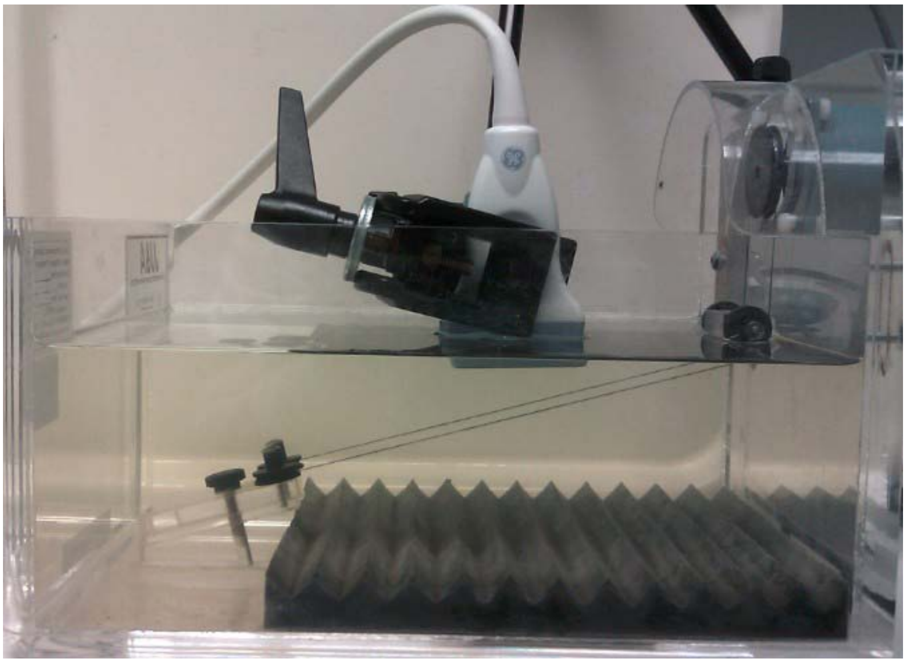


Fig. 3.1: The string phantom experiment setup. The probe is the GE linear array probe, 9L.

3.2 Computer based simulation approach

Simulation is a good way to test some new ideas or methods before achieving them on a real scanner. It is also wise to test some methods that are well developed theoretically, but difficult to be implemented on hardware. In this project, all simulations were implemented using MATLAB in the FieldSim 3 platform.

3.2.1 MATLAB Object-oriented programming and the FieldSim 3 platform

MATLAB (MATrix LABoratory) is a general-purpose scripting language particularly well suited for mathematical programming [20]. The data in MATLAB is normally stored in matrices. There are many toolboxes that handle different functions in different areas in MATLAB.

Object-oriented programming is one the most popular programming approaches. It makes it easier to develop and maintain a very large application. The software package FieldSim 3 was used in the project to generate the simulation data. It is written in MATLAB with object-oriented programming. Therefore it is easily configurable. The following block diagram shows the architecture of FieldSim 3. It is mainly constructed by 6 blocks: front-end, scan definition, scan geometry, the biological setup, the simulator and the post-processing and display.

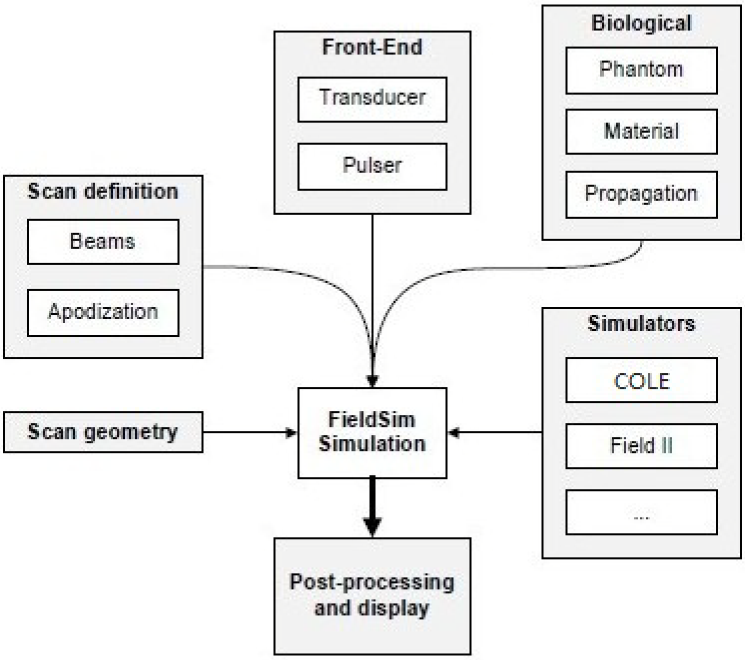


Fig. 3.2: Block diagram of the FieldSim 3 architecture. The diagram is based on Ole Bakstad’s project report [21].

Code list I: Color flow simulation example

```
1)  sim          = FieldSim.Simulation();
2)  sim.name      = 'CF-9L';
3)  % Select the basic scan configuration
4)  sim.selectMode('ColorFlow');
5)  sim.probe = '9L';
6)  sim.simulator = 'Cole';
7)  % Phantom Setting
8)  sim.phantom = 'PhantomTube';
9)  sim.phantom.string_length = 0.1;
10) sim.phantom.string_radius = 0.001;
11) sim.phantom.string_resScat = 10;
12) sim.phantom.string_vMax = 0.1;
13) sim.phantom.tilt=[0 0 pi/4];
14) % Configure the Tx/Rx layout
15) sim.scan.txPulser.f0 = 5e6;
16) sim.scan.txPulser.noPeriodsExcitation = 6.5;
17) sim.scan.txGenericBeam.focus   = 20e-3;
18) sim.scan.txGenericBeam.fNumber = 1.5;
19) sim.scan.rxGenericBeam.focus   = 'dynamic';
20) sim.scan.rxGenericBeam.fNumber = 1.1;
21) sim.scan.scanShape.shape_az      = 'rectangle';
22) sim.scan.scanShape.openingSize(1) = 3e-3;
23) sim.scan.scanShape.range_min     = 15e-3;
24) sim.scan.scanShape.range_max     = 30e-3;
25) sim.scan.PRFmax = sim.propagation.c / (2 * sim.scan.scanShape.range_max);
26) sim.scan.PRF    = 5000;
27) sim.scan.noPacket  = 64;
28) sim.scan.interleaveMode = 'auto';
29) figure(2)
30) plot(sim.scan);
31) % Simulation and post processing
32) sim.doScan();
33) sim.postProcessing.clear();
34) sim.postProcessing.addFilter('iqDemodulation', struct('f0', sim.probe.f0));
35) [iq_data metadata] =
    sim.postProcessing.apply(sim.data.rf_data, sim.data.meta);
```

In code list I, a color flow simulation example is shown with a straight tube phantom and the COLE simulator. Each block in Fig. 3.2 is set in this simulation code example. The code starts with creating an object in the ‘Simulation’ class. Then it sets different parameters in the object.

In the front-end, a certain probe needs to be chosen for the simulation. In line 5 of the code example the probe was defined. Each available probe contains many default settings. They are stored in a XML file. The frond-end contains different typical probes. Users can choose them according to different simulations. In this project, a linear array probe (GE 9L) was always applied in the simulations, since the linear probe is suitable for blood vessel imaging. Line 15 and 16 sets the transmit pulse. The impulse response of the 9L probe with the transmit pulse is shown in both time and frequency domain in Fig. 3.3. It shows that the frequency impulse response of the probe is a bandpass signal. Therefore, in order to get a suitable pulse, most of the power of the transmit pulse has to be inside the frequency impulse response of the probe.

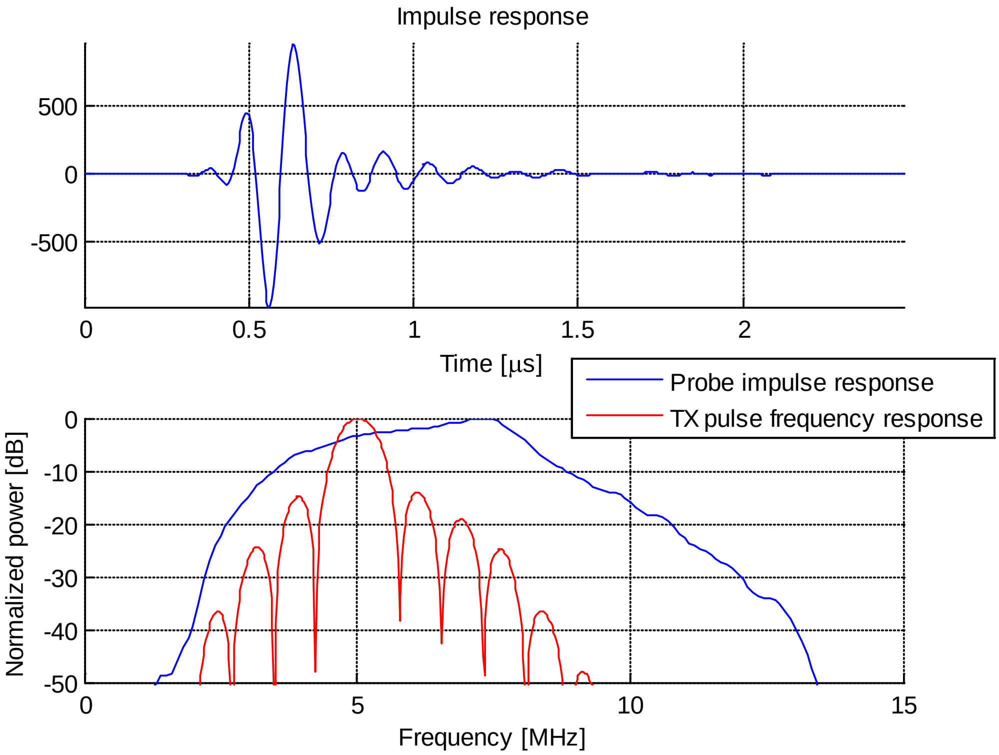


Fig 3.3: Impulse response of the linear probe 9L and the simulated transmit pulse frequency response. The upper figure shows the time domain response and the lower figure shows the frequency domain impulse response with the transmit pulse response.

In the scan definition, the users can choose different scan modes based on their simulation requirements. So far, it has B-mode, beam profile mode and color flow mode. The color flow mode was applied in the project. With a certain scan mode, detailed parameters can be set to fit the specific scan requirement as shown in the code. The scan geometry contains 3 types of scan shapes. They are rectangular, sector and curvilinear. The rectangular scan shape was used in this project. Fig. 3.4 shows a rectangular scan shape generated by the example code.

The transmit beams focused at 20 mm and the imaging range is from 15 mm to 30 mm in depth. Single line acquisition is applied.

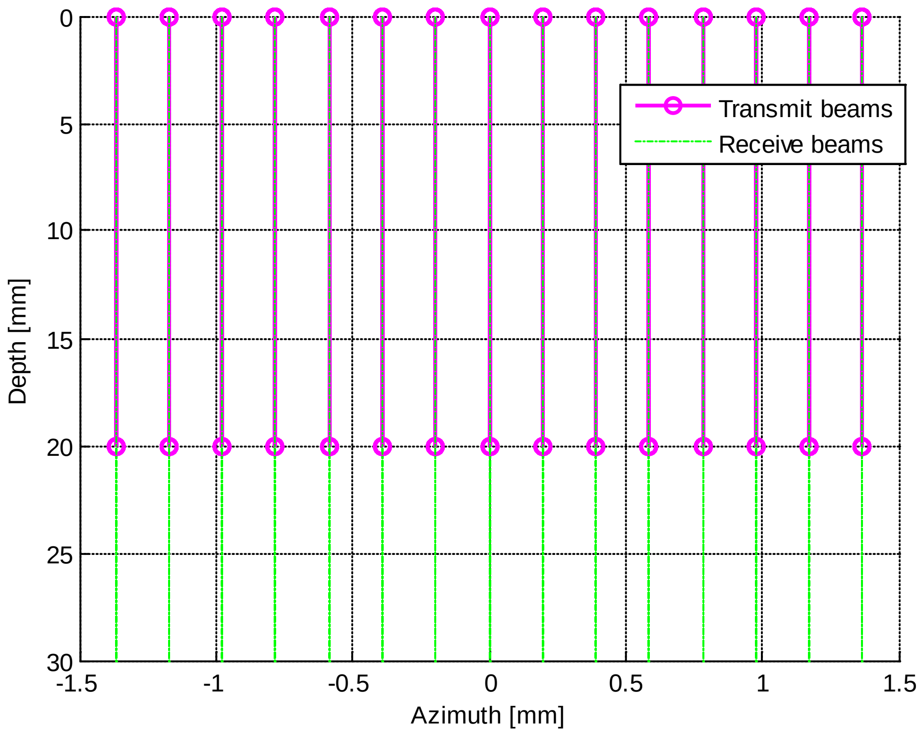


Fig. 3.4: Rectangular scan shape constructed by the transmit beams and receive beams. The transmit beams focused at 20 mm in depth.

In the post-processing and display block, different filters can be applied to the simulation data. Here the filter called “iqDemodulation” was applied to get the IQ data for further Doppler processing. The simulated data can be displayed by the predefined plot function in the FieldSim 3. The simulator and biological blocks will be described in the following section with a simulation code example.

3.2.2 Simulators and phantom settings

FieldSim 3 makes it very convenient to use different simulators. As shown in the code list I, the simulators can be changed by just changing the simulator name in the created “sim” object. By doing this the user can achieve the same simulation settings with different simulators. So far COLE, Field II, Abersim and Propose are available in the FieldSim 3 platform. In this project, only COLE and Field II were used.

Since COLE does not have the ability to generate beam profiles, Field II was used to generate the beam profile for COLE as its LUT. The 2D transmit, receive and two-way beam profiles generated by code list I are shown in Fig. 3.5. The LUT is constructed by generating a grid on the beamprofile with certain lateral width and axial distance. The scatterers that are located outside the LUT will not be considered. The scatterers are weighted at each depth according to its distance to the imaging line and summed up to attribute to the corresponding radial sample. Finally, by taking the convolution with the generated pulse, the signal along one image line is done. The 2D image can be generated by repeating this procedure.

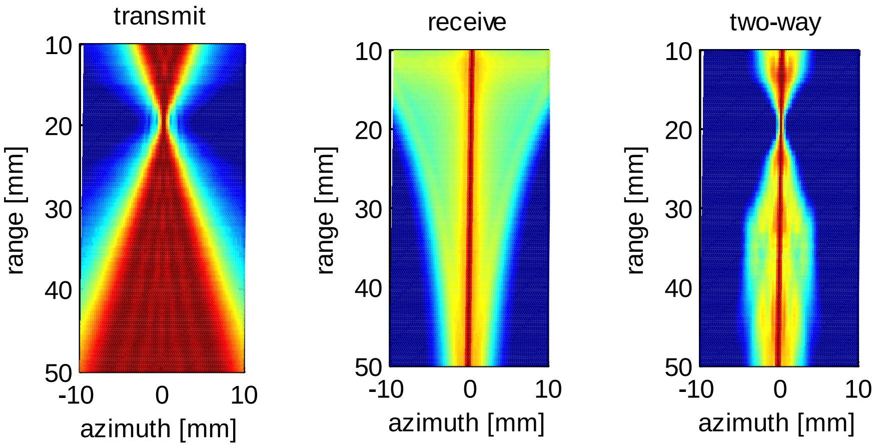


Fig. 3.5: The transmit, receive and two-way beam profiles generated from Field II.

The linear probe 9L was used in the simulation because it is suitable for vascular Doppler imaging and the phantom was simulated as a straight tube full of moving scatterers. Fig. 3.6 is the 3D plot of the simulated point scatterers distributed in the tube. It has 1 mm radius and 10 cm length. These point scatterers behave like blood cells in the vessel. The scatterer density can be set by the parameter called “string_resScat”. It indicates how many scatterers in one sampling volume, which equals to the spatial sampling frequency multiplies by the beam widths in both elevation and azimuth directions. The phantom parameters can be found in code list I. Note that for different blood velocities the length of the tube should be enough to make sure that the scatterers always appear in the measurement area. Equation (3.1) gives the minimum length of the tube.

$$L = L_1 + L_2 = \frac{packetsize}{PRF} \times v + L_2 \tag{3.1}$$

where L_1 is the length that the scatterers will move, L_2 is the length of measurement area.

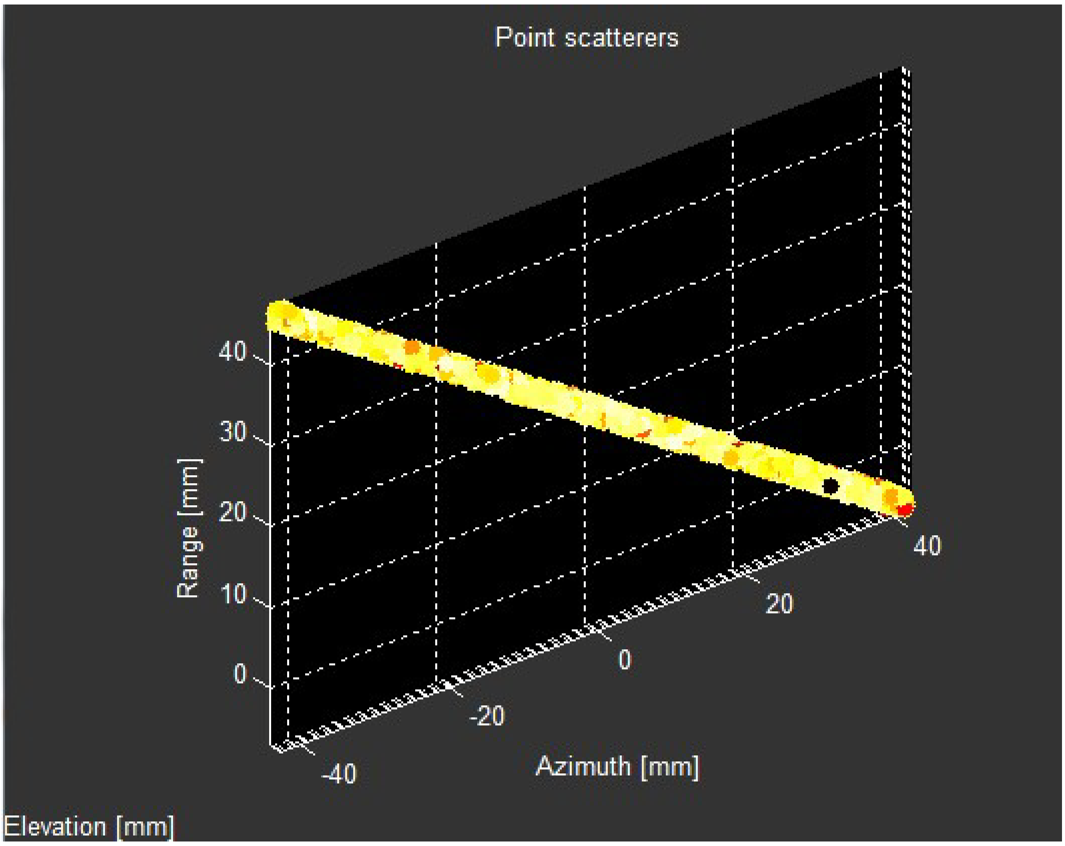


Fig. 3.6: Simulated straight tube phantom with point scatterers distributed in the tube as blood cells. The radius is 1 mm and the length is 10 cm.

4 Results

In this chapter, the results achieved in the project will be presented. The parameters used to investigate the computational time are listed, the plotted Doppler spectra and the 2D power spectra of both simulation data and recorded IQ data from the ultrasound scanner will be presented in section 4.2 and 4.3 respectively, so that the performance of the COLE simulator in the FieldSim 3 platform can be evaluated and compared with the other methods. The 2D power spectra were also plotted from the simulation data with the same settings but with different pulse length, so that the correlation in range direction can be shown.

4.1 Computational time of COLE and Field II

In order to compare the computational time of both COLE and Field II in the FieldSim 3 platform, different parameters were used in the simulation so that we can see the influence of the parameters on the efficiency of the simulator COLE. These parameters were packet size and scatterer resolution. The parameters in Table 4.1 were the fixed parameters. All simulations were recorded 5 times but only their mean computational time values are presented. The recording was done on a PC with an Intel Xeon CPU E 31270 with 4 cores 3.4GHz and 8GB RAM. The FieldSim 3 was run on MATLAB 2011b on Windows 7.

Table 4.1: Simulation parameters for investigating the computational time

Transducer frequency	5MHz
Sampling frequency	200MHz
Ultrasound velocity	1540m/s
PRF	4000
Start depth	15mm
End depth	30mm
Tube length	0.1m
Tube radius	2.5mm
Scatterer velocity	0.1m/s
Size of scan range	20mm

With a fixed number of scatterers, the computational time increased with increasing the packet size. In the following case the scatterer resolution was set to 1, i.e. 1246 scatterers in the simulated straight tube phantom. The mean CPU times for both simulators are presented in Table 4.2. The ratio of computational time of Field II to computational time of COLE is shown in Fig. 4.1.

Table 4.2: The mean CPU time for both simulators

Packet size	1	2	3	5	10	20
COLE (s)	9.17	18.9	28.1	48	101	199
Field II (s)	699	1420	2090	3531	6781	13101

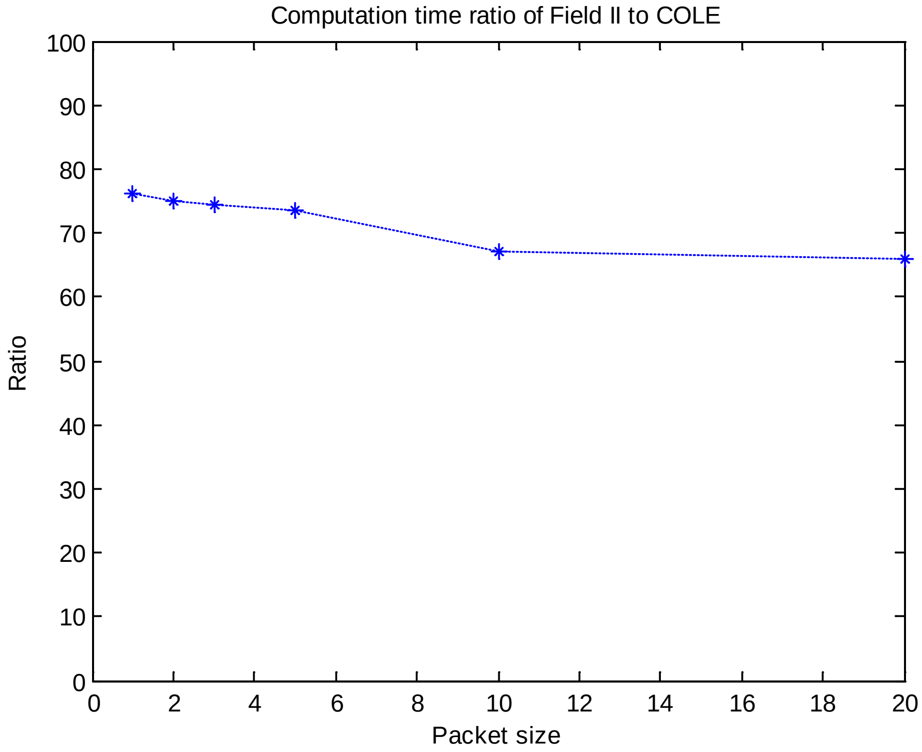


Fig. 4.1: Relationship between computational time ratio of Field II to COLE and scatterer resolution.

With a fixed packet size, the computational time increased with increasing the number of scatterers in the simulated straight tube phantom. In the following case the packet size was set to 1. The mean CPU time for both simulators is presented in Table 4.3 with different scatterer resolutions. The ratio of computational time of Field II to computational time of COLE is shown in Fig. 4.2.

Table 4.3: The mean CPU time for both simulators

Scat. Res.	0.1	0.5	0.7	1	3	5	7	10
COLE (s)	2.16	5.23	6.81	9.17	24.1	41.4	56.4	82.1
Field II (s)	72.4	361	481	699	2030	3460	4707	6883

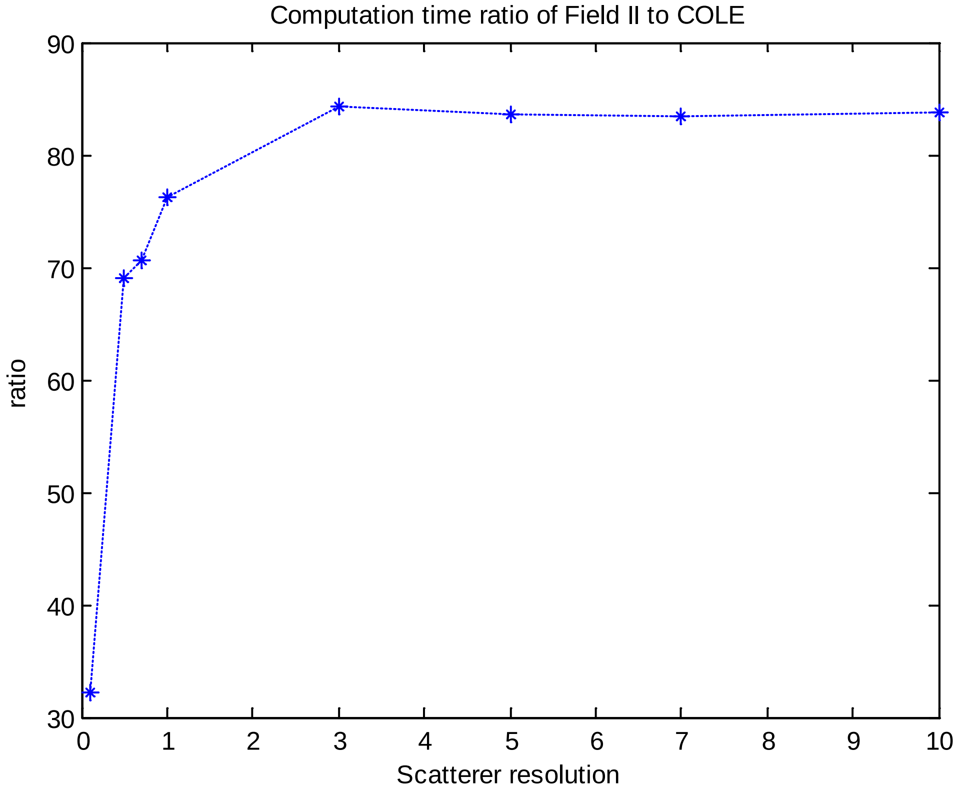


Fig. 4.2: Computational time ratio of Field II to COLE with varying scatterer resolution.

4.2 Doppler spectra from scanner recordings and simulations

In this section, the PW Doppler spectra are shown. The first 3 figures are shown to compare the performance of velocity estimation in COLE, Field II and the scanner. The simulations were done using the same settings as shown in Table 4.4. The velocities were 0.15 m/s, 0.25 m/s, 0.50 m/s. Each spectrum was done by taking the FFT with a 64 point Hamming window to one packet and then averaging over 10 estimations to reduce the variance. The velocity axis in each figure is already angle corrected, so that the true velocities are presented. Fig. 4.6 shows the performance of the velocity estimation in the COLE and Field II simulators with different sampling frequencies.

Table 4.4: Simulation parameters

Transducer frequency	5MHz
Sampling frequency	200MHz
Ultrasound velocity	1540m/s
No. of period	6.5
Start depth	15mm
End depth	30mm
Tube length	0.1m
Doppler angle	58 degree
PRF	5000
Packet size	64

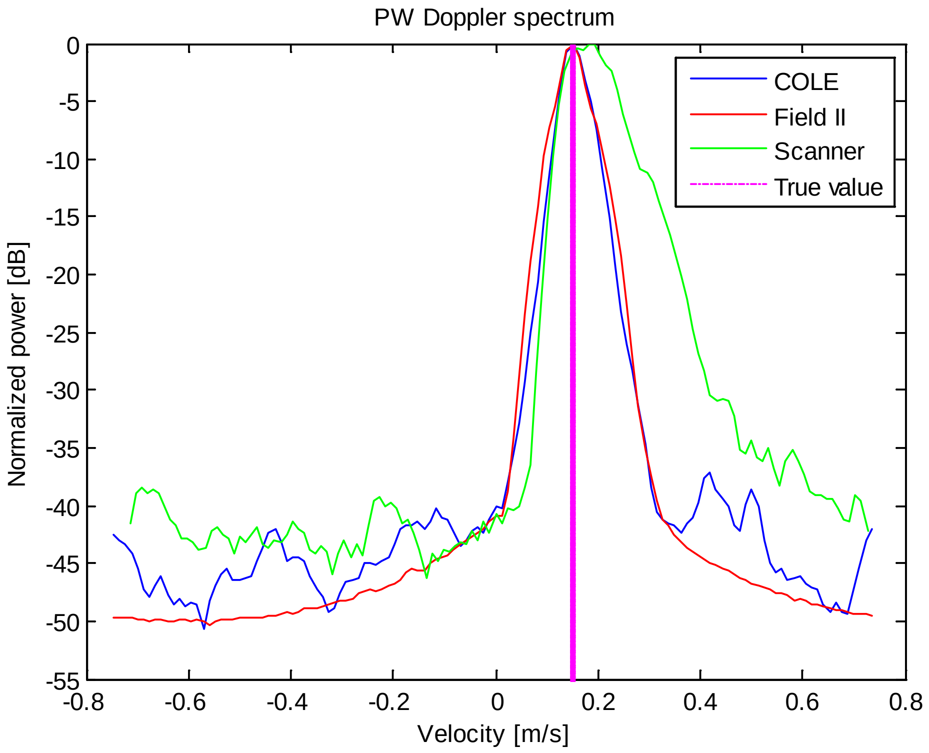


Fig. 4.3: PW Doppler spectra generated from COLE (blue curve), Field II (red curve) and from the scanner IQ data (green curve) respectively. The true velocity value is 0.15 m/s.

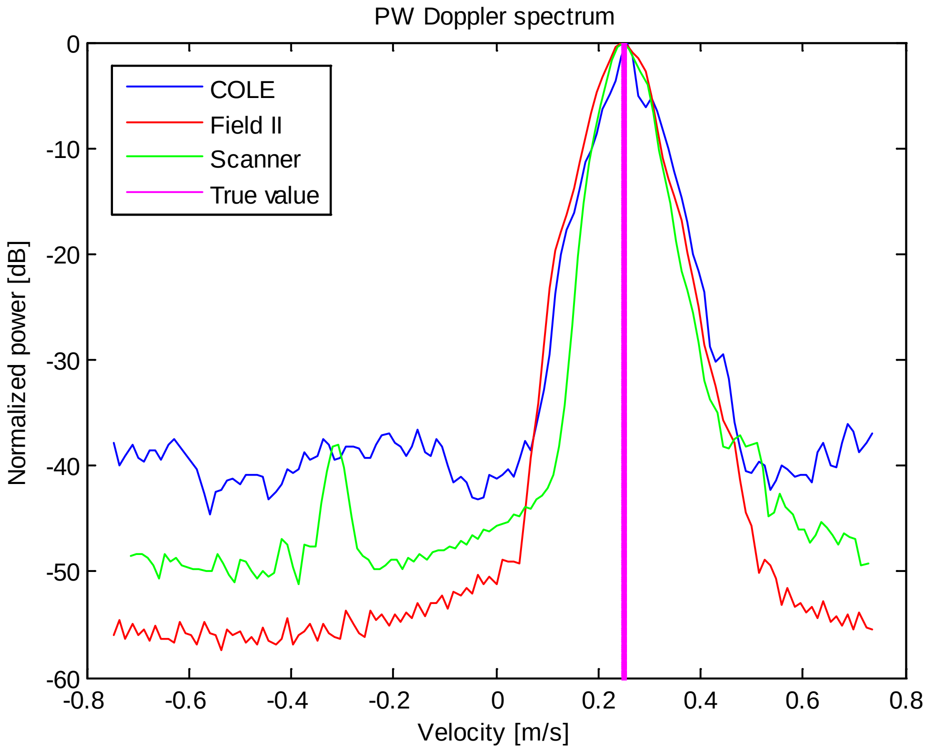


Fig. 4.4: PW Doppler spectra generated from COLE (blue curve), Field II (red curve) and the scanner IQ data (green curve) respectively. The true velocity value is 0.25 m/s.

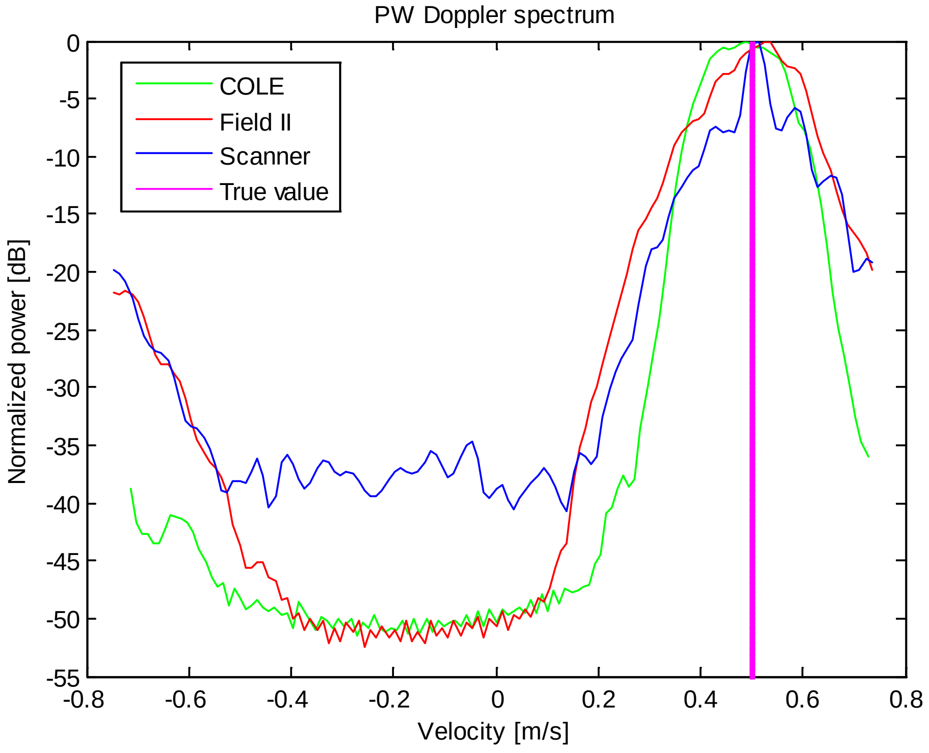


Fig. 4.5: PW Doppler spectra generated from COLE (blue curve), Field II (red curve) and the scanner IQ data (green curve) respectively. The true velocity value is 0.50 m/s.

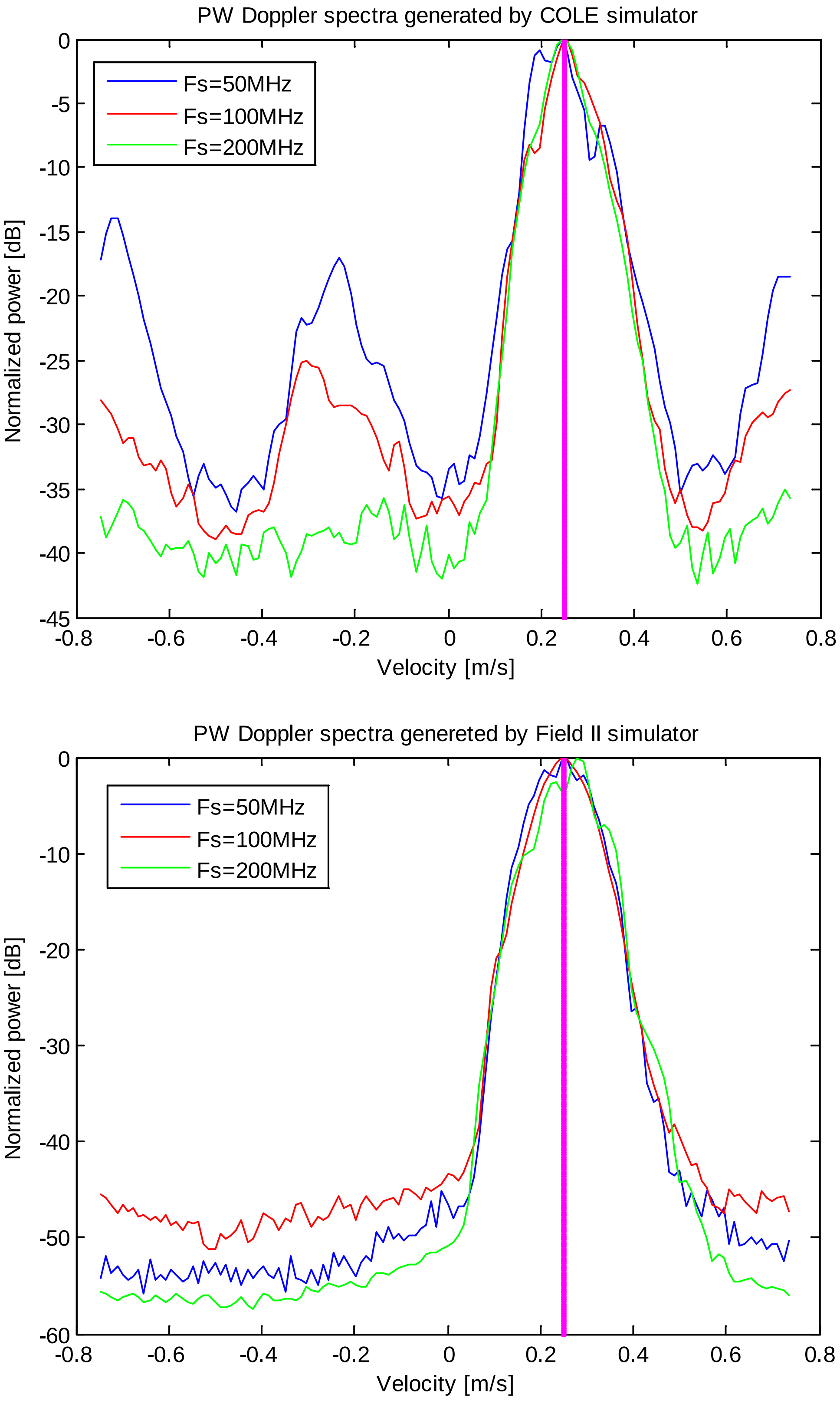


Fig. 4.6: PW Doppler spectra generated from COLE and Field II with different sampling frequencies. The blue curve has sampling frequency of 50 MHz. The red curve has sampling frequency of 100 MHz. The green curve has sampling frequency of 200 MHz. The true velocity values are 0.25m/s.

4.3 2D power spectra generated from the scanner and the simulation

The 2D power spectra are plotted using different data with different velocities and are shown in Fig. 4.7-4.15. They are the corresponding 2D power spectra of the Doppler spectra shown in Fig. 4.3-4.5. The same settings as shown in Table 4.4 were used. The slope of the red line plotted on each 2D power spectrum indicates the factor in equation (2.39). Both horizontal bandwidth and vertical bandwidth for each spectrum are listed in Table 4.5. The vertical axis in each figure shows the Doppler shift, whereas the horizontal axis gives the transmit pulse frequency. The sampling frequency for the IQ data is 3.125 MHz.

Table 4.5: Bandwidths of the 2D power spectra.

Fig. Band-width	4.7	4.8	4.9	4.10	4.11	4.12	4.13	4.14	4.15
Velocity (m/s)	0.50			0.25			0.15		
Horizontal (MHz)	1.52	1.50	1.51	1.51	1.50	1.53	1.48	1.50	1.52
Vertical (KHz)	3.25	3.57	3.36	2.01	2.14	2.00	1.29	1.31	1.78

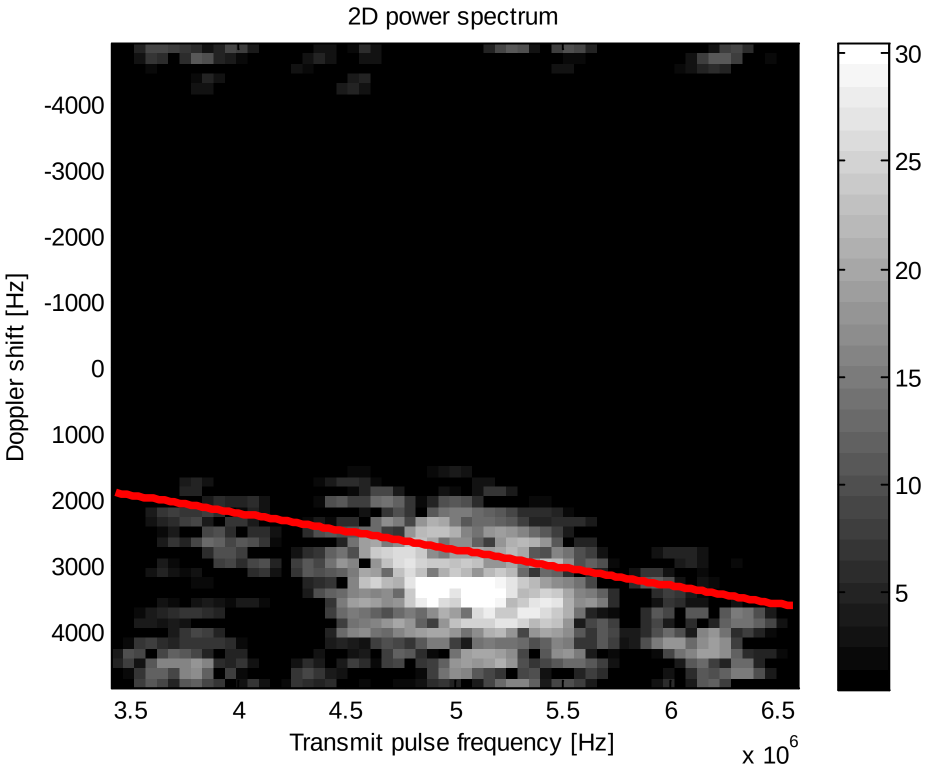


Fig. 4.7: 2D power spectrum generated from the simulated data of COLE. The velocity of the simulated phantom is 0.50 m/s.

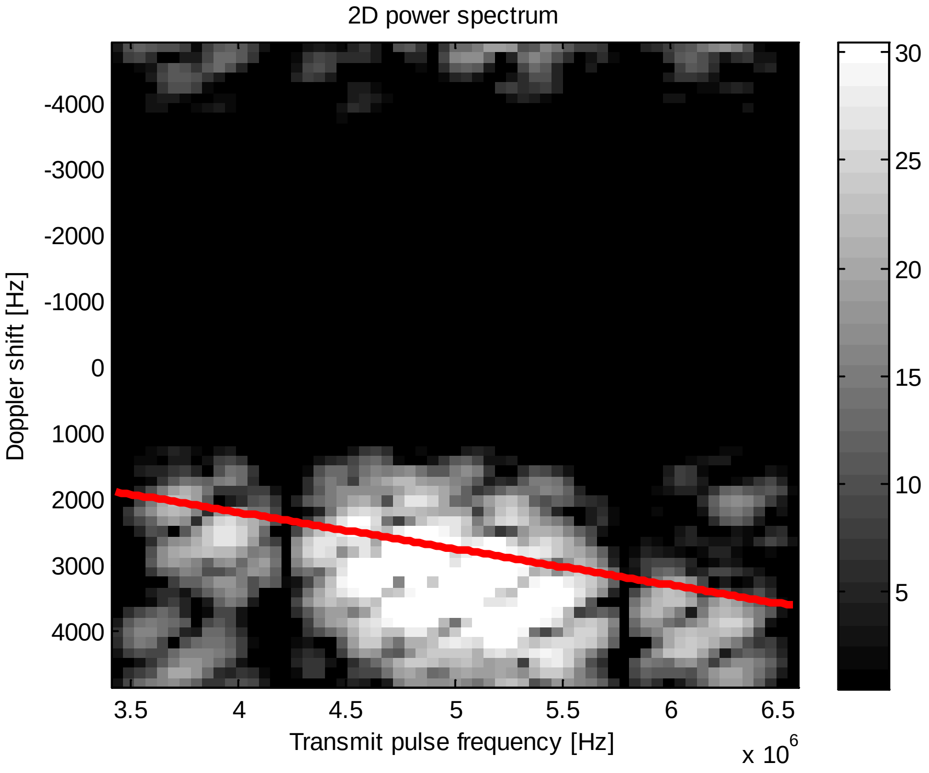


Fig. 4.8: 2D power spectrum generated from the simulated data of Field II. The velocity of the simulated phantom is 0.50 m/s.

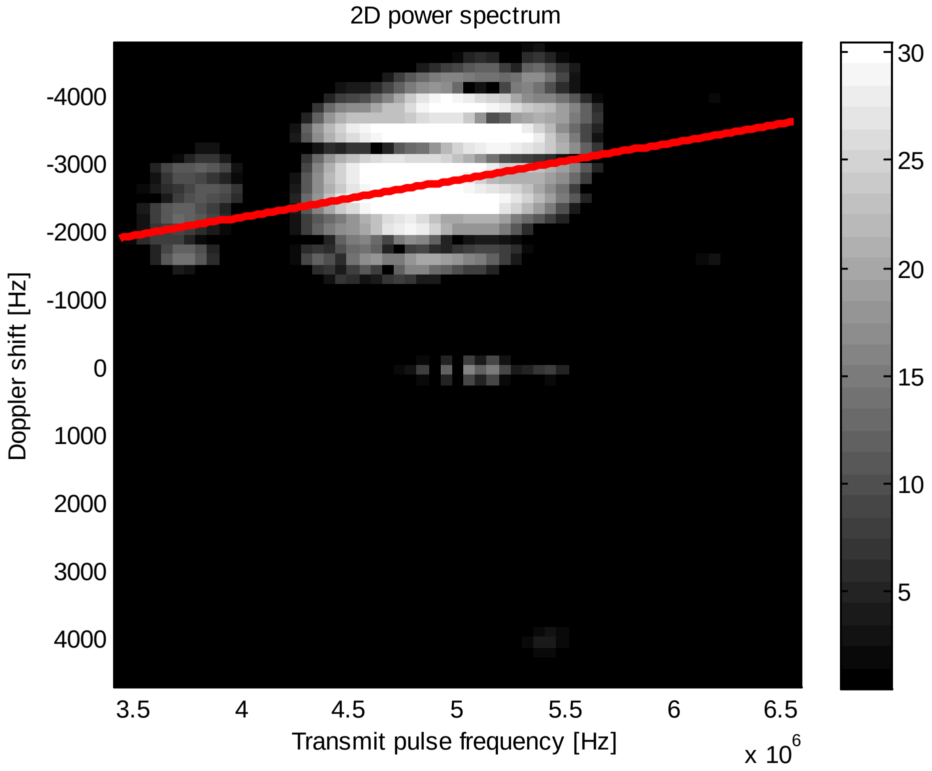


Fig. 4.9: 2D power spectrum generated from the IQ data from the scanner. The velocity of the string phantom is 0.50 m/s.

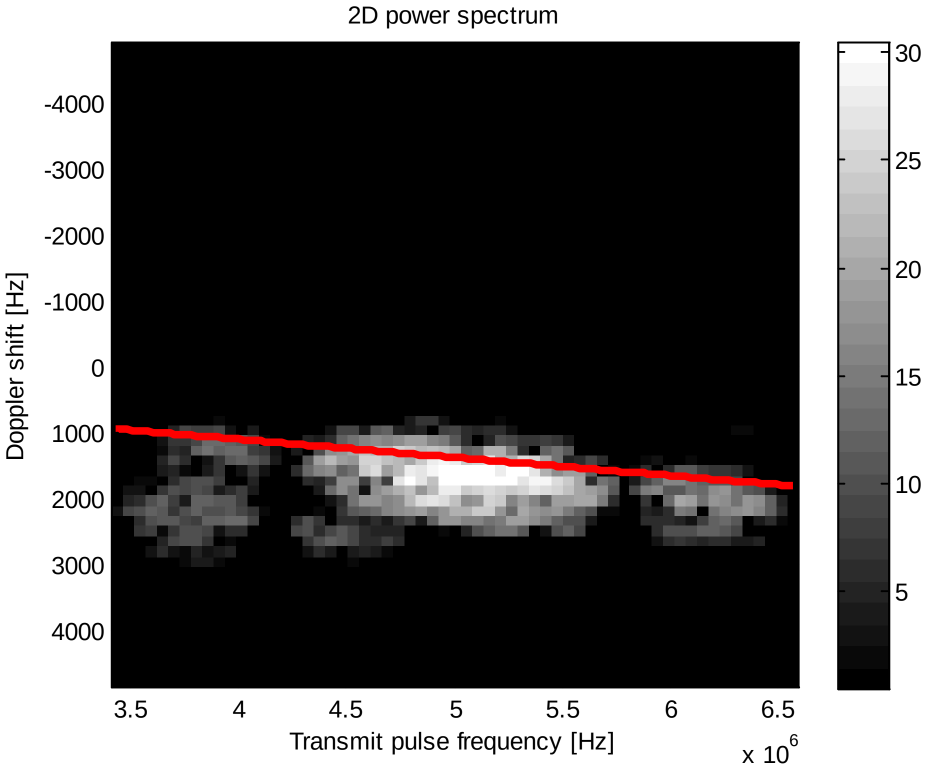


Fig. 4.10: 2D power spectrum generated from the simulated data of COLE. The velocity of the simulated phantom is 0.25 m/s.

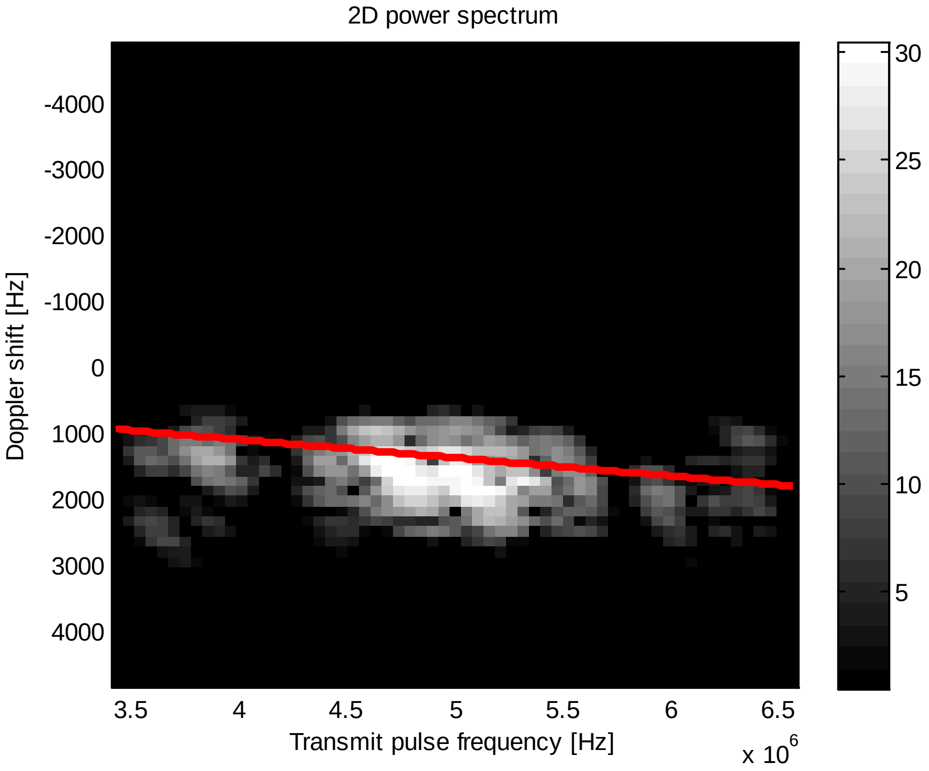


Fig. 4.11: 2D power spectrum generated from the simulated data of Field II. The velocity of the simulated phantom is 0.25 m/s.

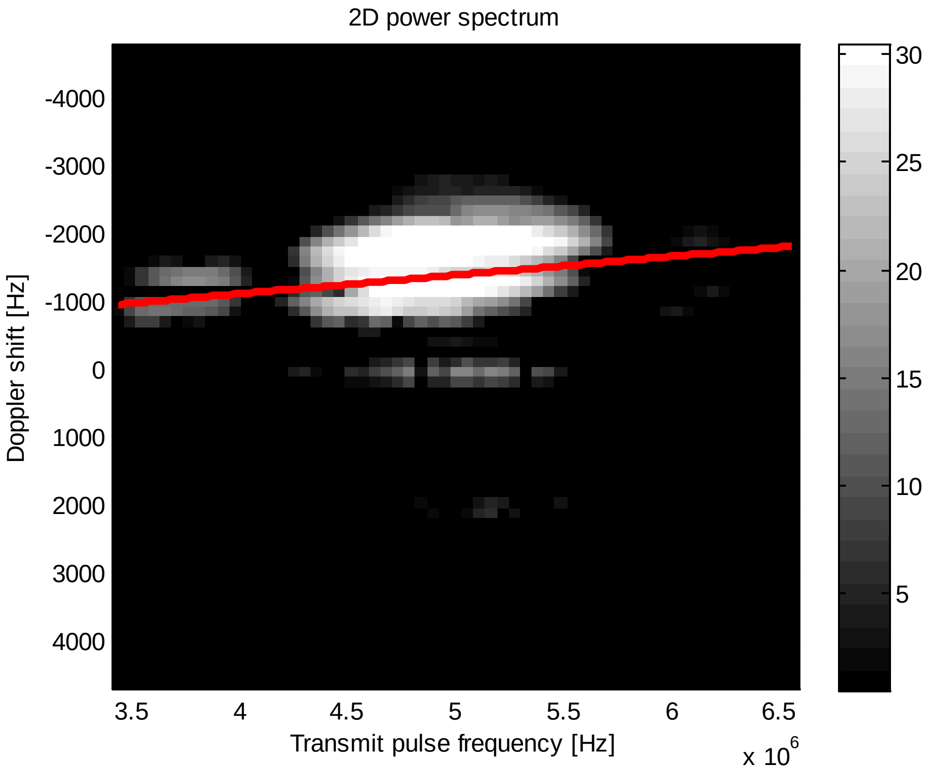


Fig. 4.12: 2D power spectrum generated from the IQ data from the scanner. The velocity of the string phantom is 0.25 m/s.

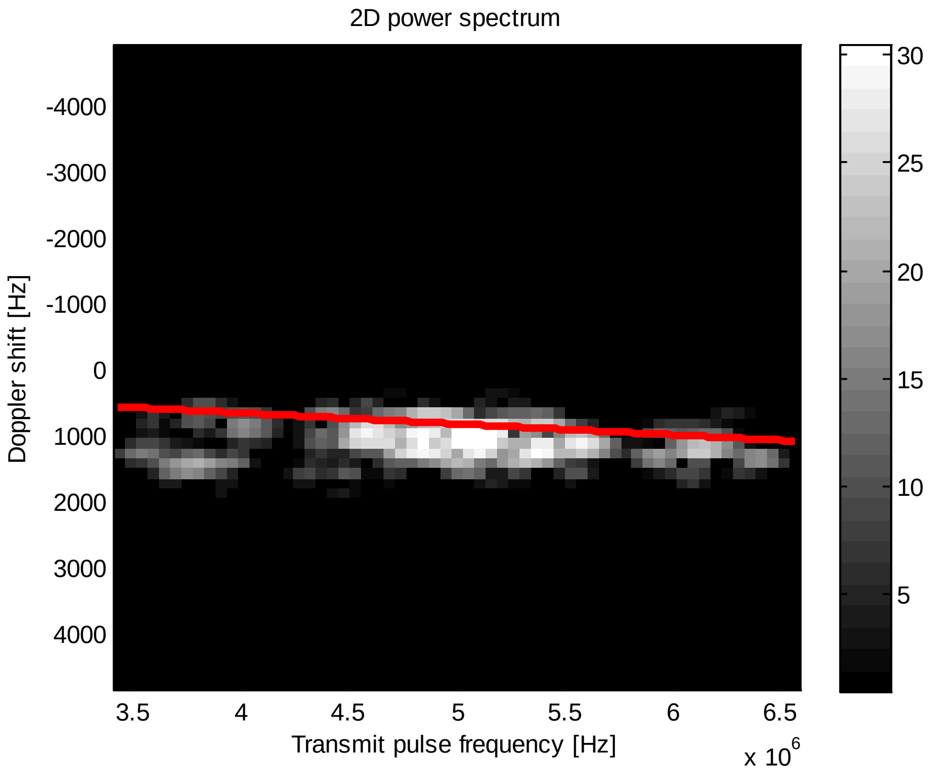


Fig. 4.13: 2D power spectrum generated from the simulated data of COLE. The velocity of the simulated phantom is 0.15 m/s.

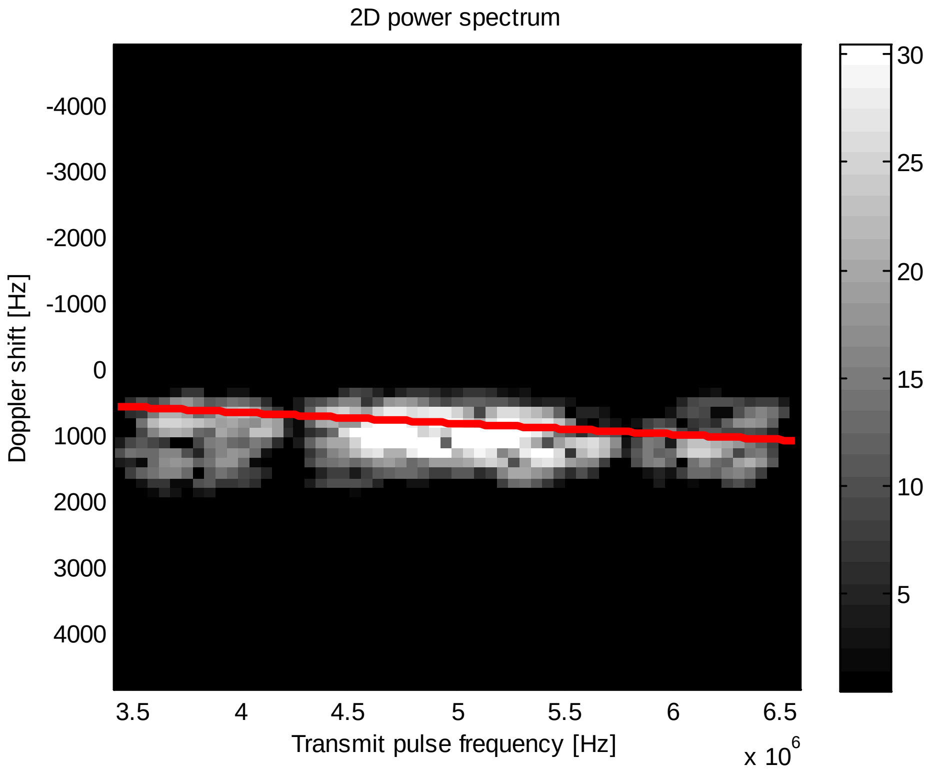


Fig. 4.14: 2D power spectrum generated from the simulated data of Field II. The velocity of the simulated phantom is 0.15 m/s.

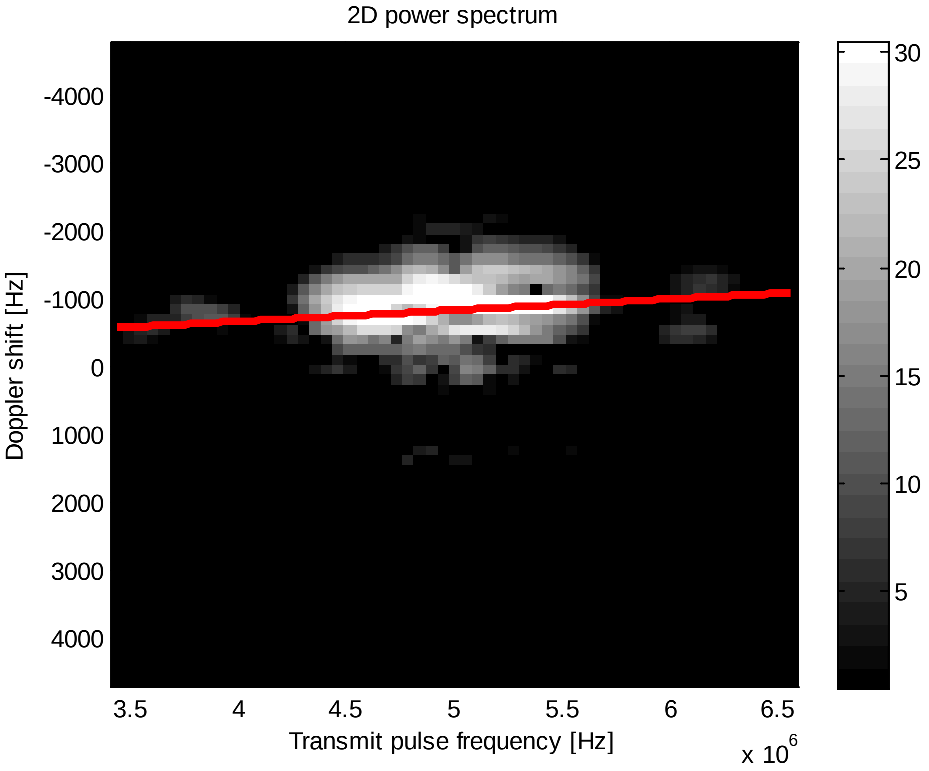


Fig. 4.15: 2D power spectrum generated from the IQ data from the scanner. The velocity of the string phantom is 0.15 m/s.

Fig. 4.16 4.17 and 4.18 are 2D power spectra generated from data simulated by COLE with the same velocity, 0.25 m/s, and different numbers of pulse periods. In this case the horizontal bandwidths change and the correlation in range direction can be seen clearly. With a pulse of 3 periods, the horizontal bandwidth is 2.97 MHz as shown in Fig. 4.16. With a pulse of 6 periods, the horizontal bandwidth is 1.51 MHz as shown in Fig 4.17. With a pulse of 9 periods, the horizontal bandwidth is 1.01 MHz as shown in Fig 4.18.

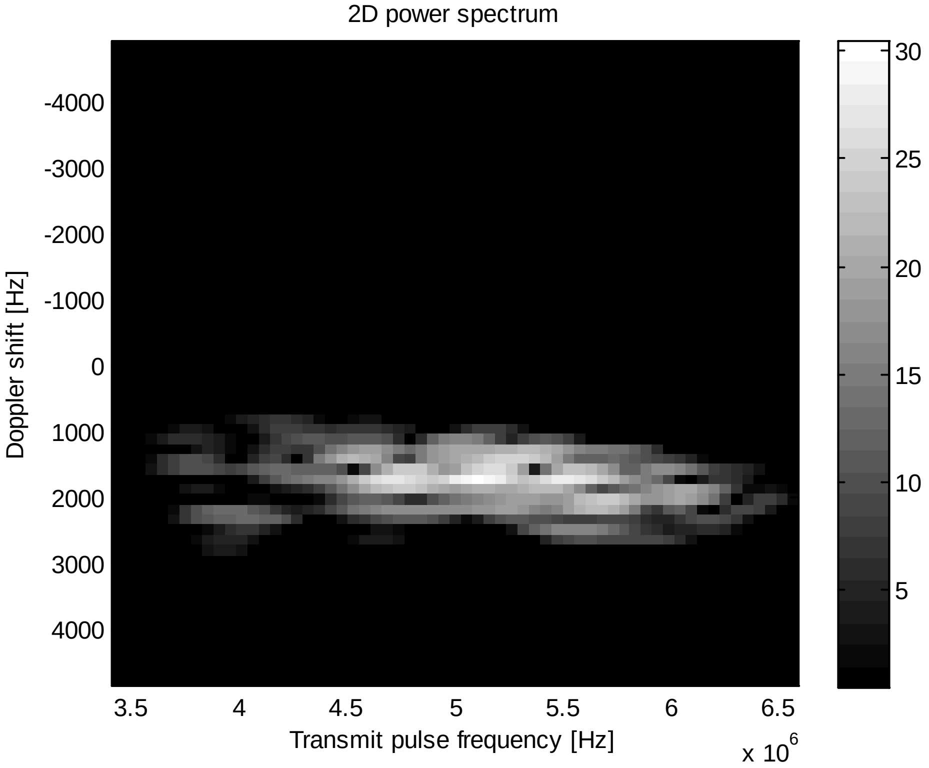


Fig. 4.16: 2D power spectrum generated from the simulated data of COLE with 3 pulse periods. The velocity of the simulated phantom is 0.25 m/s.

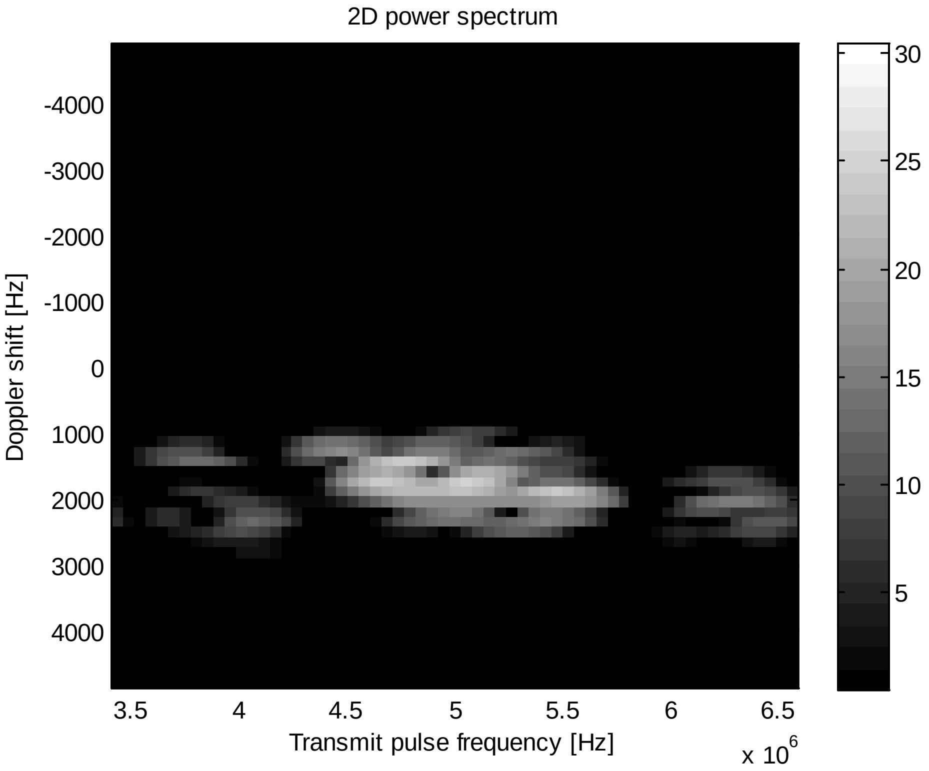


Fig. 4.17: 2D power spectrum generated from the simulated data of COLE with 6 pulse periods. The velocity of the simulated phantom is 0.25 m/s.

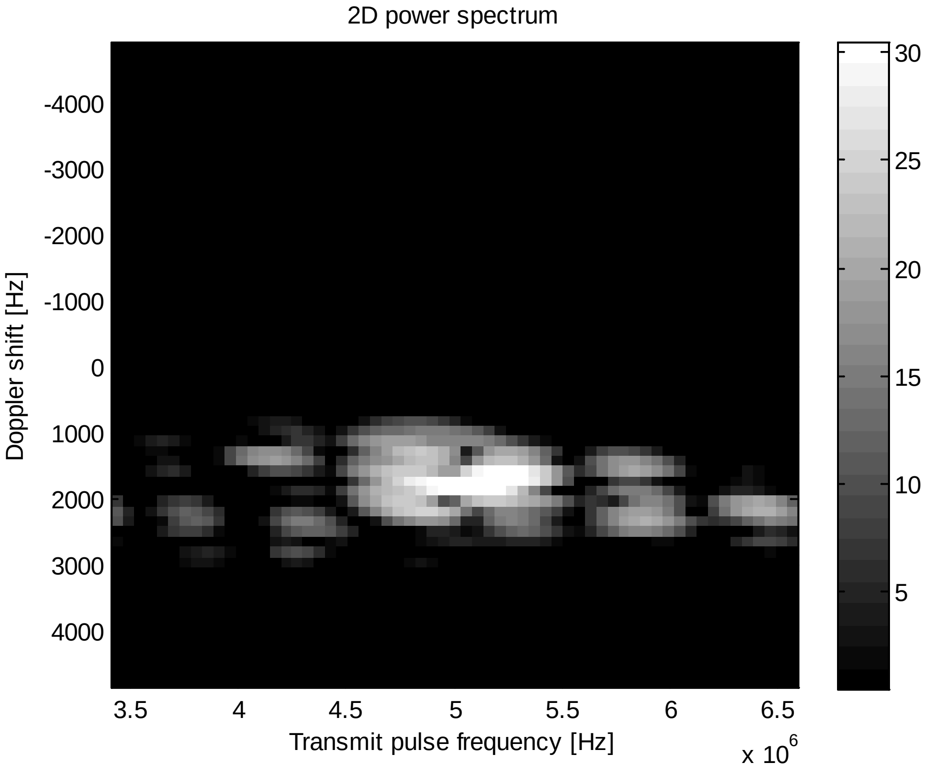


Fig. 4.18: 2D power spectrum generated from the simulated data of COLE with 9 pulse periods. The velocity of the simulated phantom is 0.25 m/s.

5 Discussion

In this chapter, all of the results presented in the previous chapter will be discussed and analyzed. Both advantages and disadvantages of COLE, compared with Field II and the scanner, will be shown by taking a closer look at the computational time and the spectra.

In section 5.1, the computational time of both simulators as a function of packet size and number of scatterers will be compared. In section 5.2, the estimation performance of the 3 methods will be discussed by comparing the bandwidth of their spectra in both the fast-time direction and the slow-time direction.

5.1 Computational time

The most important advantage of COLE is that its computational time is short. In the previous work published by H. Gao et al. [2], the computational time achieved was 1200 times faster than Field II in 3D B-mode imaging. We have looked at the 2D Doppler mode scan in the FieldSim 3 platform.

With a fixed number of scatterers, the computational time of both simulators increased when the packet size increased. However the computational time ratio of Field II to COLE did not change too much, because by increasing the packet size only the repetitions of the transmit pulse increased and the averaging simulation time for each transmit should be same theoretically. Therefore, the same theory can be applied to the number of beams and frames. The COLE simulator was about 70 times faster than the Field II simulator in the FieldSim 3 platform.

With a fixed packet size, the computational time of both simulators increased when the number of scatterers increased. The computational time ratio of Field II to COLE changed a lot when the number of scatterers increased, until the scatterer density reaching a certain value. The ratio plotted in Fig.4.2 is similar with an exponential function. When the scatterer resolution was smaller than 1, the ratio changed a lot. When the scatterer resolution was larger than 3 the ratio became almost a constant. The reason is that there are a lot of functions were used in the simulation and when the number of scatterers was small; the computational time of those functions took relatively high proportion of the total time. Howener, when the number of scatterers was large enough, the computational time of those functions can be neglected. The COLE simulator achieved a computation time that was about 85 times faster than Field II.

5.2 PW Doppler estimation performance

In order to evaluate the performance of the COLE simulator in the FieldSim 3 platform, both Doppler spectra and 2D power spectra were generated. The Field II simulator and the string phantom experiment were used for comparison. Since Field II has perfect accuracy, theoretically, its performance should be equivalent to the scanner. Constant velocities of 0.15 m/s, 0.25 m/s and 0.50 m/s were used in both the simulations and in the in vitro phantom. The spectra plotted from the string phantom experiment have relatively low accuracy, due to the mechanical error of the string motion and the measurement system error.

In the plotted Doppler spectra, i.e. Fig. 4.3-4.5, the same settings are applied except the velocity. Generally, the simulated data from the Field II simulator gave better performance than the other 2 methods. Compared with Field II, COLE had higher noise floor. The biases of these three methods are almost zero. The spectra plotted from scanner IQ data always have a side lobe around the negative velocity. That is because the string was moving in a round trip and they were not that far away from each other, see Fig. 3.1. The probe can detect a weak signal from the water moving along with the string on the other side of the phantom.

Fig. 4.3 shows that both simulators have a narrower bandwidth than the scanner. Theoretically it should have similar bandwidth as Field II. Here it caused by some unknown reasons. It can be caused by a mechanical effect. Since the string moves in the water, it “forces” the water to flow with it. As a consequence, there might be some bubbles in the water around the string contributes to the velocity band. The peak velocities of all three methods are approximately 40dB stronger than the noise floor. In this case with low velocity, the noise floor of the spectrum from COLE is almost at the same level as Field II.

In Fig. 4.4, both simulators have slightly worse velocity resolution than the scanner. The noise floor of COLE is higher than in the previous situation. The negative velocity component of the scanner spectrum becomes stronger, due to that the power of the water around the string is increased.

In Fig. 4.5, all three spectra have worse velocity resolution and aliasing appears due to the velocity band exceeds the Nyquist limit. The spectrum from the scanner has better velocity resolution than the simulators. The noise floor of COLE becomes higher.

Fig. 4.6 is plotted to show how the sampling frequencies affect spectra generated from COLE and Field II simulated data. In the spectra generated by the data from COLE, as the sampling frequency increases the side lobes of the spectra are suppressed. At 200 MHz the side lobes almost disappear. Meanwhile, the variance around the peak of the spectra becomes lower. In the spectra generated by the data from Field II, the side lobes do not appear in all the tested sampling frequencies.

In the plotted 2D power spectra, i.e. Fig. 4.7-4.15, the same settings are applied except the velocity. Generally, as shown in Table 4.5, the horizontal bandwidths are approximately the same, because the transmit pulse lengths are the same. The measured bandwidths of the 2D power spectra have relatively low accuracy, because they were measured manually and the gain used in each method was different. The side lobes of the transmitted frequency appear in each figure. The vertical axis gives the Doppler shift. The bandwidths along the vertical axis correspond to the bandwidths in the Doppler spectra. The locations of the main lobe of the spectra in vertical direction indicate the Doppler shifts, which can be used to calculate to the velocities. The red line plotted on each figure along the spectrum was generated using equation (2.39). The slope of the line is the factor α in that equation and, theoretically, it should fit the spectrum. The slope of the line from the scanner spectra equals to the negative slope of the line from the simulation. It means that the velocity directions are different. The string phantom has the same velocity direction as the beam, whereas the simulated scatterers in the tube have opposite direction of the beam. This can be defined by the negative sign in equation (2.25).

In the 2D power spectra generated from scanner IQ data, every spectrum contains a zero Doppler shift component. It appears because of the clutter signal. In the 2D power spectra generated from the simulators, both simulators have batter velocity resolution than the scanner, when the true velocity equals to 0.15 m/s. When the velocities equal to 0.25 m/s and 0.5 m/s, the velocity resolutions from all three methods are at the same level. This can be verified by comparing the Doppler spectra presented in section 4.2. The vertical bandwidths of the 2D spectra should theoretically equal to the velocity spectra bandwidths. However a little difference can be found because the bandwidths of the 2D spectra can be affected by the gain of the images.

In Fig. 4.16-4.18, the bandwidth of the transmit frequency increases when the number of transmit pulse periods is reduced. From the measured data in the previous chapter, it can easily be seen that the bandwidths are inversely proportional to the pulse lengths. This means that the simulated data from COLE has correct correlation in range direction.

6 Conclusion

This thesis work can be divided into two parts. One is to implement the COLE simulator on the FieldSim 3 platform, the other is to test the simulator with PW Doppler scan mode and compare the computational time and the performance with the Field II simulator and data from a real scanner.

As a convolution-based methodology, COLE produces ultrasound data sets by convolving the transmitted pulse with the projected amplitudes of all the scatterers along one image line in the spatial domain [2]. Since the COLE simulator does not have the ability to generate a beamprofile, Field II was used to generate the beamprofile as its LUT. In this way, the conventional 2D/3D convolution problem reduces to multiple 1D convolution (one for each line) [22]. COLE has successfully become one of the simulators available in the FieldSim 3 platform.

The most important three parameters to evaluate the simulator COLE are the computational time, the velocity resolution and the noise floor level. The first one is the advantage of COLE and the last two mostly decid the performance of COLE in Doppler scan mode.

The computational time of COLE in the FieldSim 3 platform has been measured in 2D Doppler scan mode. With a fixed number of scatterers, the ratio of Field II to COLE did not change too much for different packet sizes. With enough number of scatterers, which means to set the scatterer resolution larger than 3 in the straight tube phantom, the computational time that COLE could achieve about 85 times faster than the Field II simulator.

The performance of COLE was compared with both Field II and an ultrasound scanner using PW Doppler spectra and 2D power spectra. From the PW Doppler spectra, it was seen that in general COLE had worse performance than Field II. Especially when the velocity increased, the noise floor of COLE became higher than the other two methods. The velocity resolution at the true velocity of COLE was almost the same as Field II. The bias of COLE was in the same level as Field II and the scanner, which was almost zero. COLE had relatively high sampling frequency requirement compared with Field II. In order to avoid the side lobes of the spectra, the sampling frequency should be high enough. From the 2D power spectra, the plotted skewed lines fitted the spectra well. The Doppler shift bandwidths had correct value based on the PW velocity spectra; the transmit frequency bandwidths were inversely proportional to the transmit pulse length, which meant the correlation in range direction was correct as well.

In words, COLE can provide the velocity resolution at the same level as Field II. Even though it had higher noise floor, 40dB is enough to avoid the noise contribution. The computational time is much faster than Field II. Considering these, COLE is a good simulator choice in the FieldSim 3 platform for the 2D Doppler mode scan.

6.1 Future work

The future work can be divided into two directions. One is continuously working with developing the FieldSim 3 platform. The other one is to test the performance of COLE continuously.

So far COLE can only use Field II to generate the beamprofile. It will be good if other simulators become available to generate beamprofiles for COLE. Then the user will have more choices. It is also possible to optimize some functions used in COLE, so that the computational time can be further reduced.

COLE has just been tested with 2D Doppler scan mode with velocities below the Nyquist limit. It will be interesting to have simulated data generated from COLE with velocities higher than the Nyquist limit and compare the performance with Field II. And also it can be used to simulate 3D/4D Doppler simulation. In 3D/4D Doppler simulation, the computational time becomes much higher than in 2D. Then the advantage of COLE will become significant.

Reference list

[1] J. A. Jensen, “A Model for the Propagation and Scattering of Ultrasound in Tissue,” *J. Acoust. Soc. Am.*, vol. 89, pp.182-191, 1991.

[2] H. Gao et al., “A Fast Convolution-Based Methodology to Simulate 2-D/3-D Cardiac Ultrasound Images,” *IEEE Trans., Ultrason.Ferroelectr.Freq. Control,* vol. 56, no. 2, pp. 404-409, Feb. 2009.

[3] B. Angelsen and H. Torp, *Ultrasound Imaging - Waves, Signals and Signal Processing in Medical imaging*, vol. 1 and 2. Emantec AS, 2000.

[4] J. A. Jensen, Linear description of ultrasound imaging systems, notes for the international summer school on advanced ultrasound imaging," tech. rep., Technical University of Denmark, 1999.

[5] J. A. Jensen, “Field: A Program for Simulating Ultrasound Systems,” *Paper presented at the 10th Nordic-Baltic Conference on Biomedical Imaging Published in Medical & Biological Engineering & Computing*, pp. 351-353, Volume 34, Supplement 1, Part 1, 1996.

[6] J. A. Jensen and N. B. Svendsen, “Calculation of pressure fields from arbitrarily shaped, apodized, and excited ultrasound transducers,” *IEEE Trans. Ultrason., Ferroelec., Freq. Contr.,* 39, pp. 262-267, 1992.

[7] J. A. Jensen, *Users’ guide for the Field II program*, Technical University of Denmark, 2011

[8] A. D. Pierce. “Acoustics, An Introduction to Physical Principles and Applications,” *Acoustical Society of America*, New York, 1989.

[9] J. C. Bamber and R. J. Dickinson, “Ultrasonic B-scanning: A computer simulation,” *Phys. Med. Biol.*, vol. 25, no. 3, pp. 463–479, May 1980.

[10] J. Meunier, “Tissue motion assessment from 3D echographic speckle tracking,” *Phys. Med. Biol.*, vol. 43, no. 5, pp. 1241–1254, May 1998.

[11] J. Meunier and M. Bertrand, “Echographic image mean gray level changes with tissue dynamics: A system-based model study,” *IEEE Trans. Biomed. Eng.*, vol. 42, no. 4, pp. 403–410, Apr. 1995.

[12] T. Hergum, S. Langeland, E. W. Remme, H. Torp, “Fast ultrasound imaging simulation in K-space,” *IEEE Trans Ultrason Ferroelectr Freq. Control,* vol. 56, no. 6, pp. 1159-1167, 2009.

[13] T. L. Szabo, *Diagnostic Ultrasound Imaging - Inside Out.* Elsevier, 2004.

[14] R. Pailoor and D. Pradhan, *Digital Signal Processing (DSP) for Portable Ultrasound,* Texas Instruments, Dec. 2008.

[15] J. Kirkhorn, Introduction to IQ-demodulation of RF-data. 1999.

[16] D. Maulik, *Doppler Ultrasound in Obstetrics and Gynecology,* Springer, 2005.

[17] H. Torp and K. Kristoffersen et al., “Autocorrelation techniques in color flow imaging. Signal model and statistical properties-of the autocorrelation estimates,” *IEEE Trans. Ultrasound Ferroelec. Freq. Control,* vol. 41, no. 5, pp. 604-612, 1994.

[18] H. Torp and K. Kristoffersen, “Velocity matched spectrum analysis: A new method for suppressing velocity ambiguity in pulsed-wave doppler”, *Ultrasound in Medicine and Biology*, vol. 21, no. 7, pp. 937 - 944, 1995.

[19] Online manual script of the Doppler string phantom, http://www.cirsinc.com/file/Products/043/043_CP.pdf

[20] The Math Works Inc. *Matlab - The language of technical computing*.

[21] O. Bakstad, “A unified simulation framework for medical ultrasound imaging Incorporation and validation of non-linear wave propagation using Abersim,” *Project Report,* Feb., 2012

[22] H. Gao, J. D’hooge, T. Hergum and H. Torp, “Comparison of the Performance of Different Tools for Fast Simulation of Ultrasound Data”, *IEEE International Ultrasonics Symposium Proceedings,* 2008.

Appendix A

Simulated straight tube phantom code

```
classdef PhantomTube < FieldSim.Medium.Phantom

   properties

      % define the tube for the string
      string_length = 0.05;
      string_radius = 0.0025;

      % quantity of scatterers
      string_resScat = 10;

      % velocity for Poseuille flow
      string_vMax       = 0.5;
      string_vExponent = 2;
      PoiseuilleFlow=1; %0 for uniform flow

   end % properties

   properties (Hidden)

      scat_velocities = []; % the velocities for all initial
                                  scatterers
   end % properties (Hidden)
```

```
    methods

        function obj = PhantomTube (sim)
            obj = obj@FieldSim.Medium.Phantom(sim);
            sim.phantom.reset();
            obj.origo = [0 0 20e-3];
            obj.tilt  = [0 0 pi/9];
        end % PhantomTube

        function copy(obj, phantom)
            if ~isa(phantom, 'FieldSim.Medium.Phantom')
                error('The objects to copy must be of the same
                      class');
            end
            copy@FieldSim.Medium.Phantom(obj, phantom);
            if isa(phantom, 'FieldSim.Medium.PhantomTube')
                obj.string_length     = phantom.string_length;
                obj.string_radius     = phantom.string_radius;
                obj.string_resScat    = phantom.string_resScat;
                obj.string_vMax       = phantom.string_vMax;
                obj.string_vExponent = phantom.string_vExponent;
                obj.scat_velocities  = phantom.scat_velocities;
            end
        end % copy

        function [positions, amplitudes] =
                                           generateInitalScatterers(obj)

            obj.displacement = @obj.displacementCallback;

            total_volume = obj.string_length * 2*obj.string_radius
* 2*obj.string_radius;
```

```
        % Ultrasound-related variables
        Deltax = obj.sim.scan.txGenericBeam.fNumber(1) *
              obj.sim.scan.txPulser.getLambda(obj.sim);
        Deltay = Deltax;
        Deltaz = obj.sim.propagation.c / 2 *
              obj.sim.scan.txPulser.noPeriodsExcitation /
              obj.sim.scan.txPulser.f0;
        number_scat = ceil( obj.string_resScat * (total_volume)
/ (Deltax*Deltay*Deltaz) );

        % New creation of random scatterers
        % Creation of random scatterers in a bounding box          surrounding
the tube
        x = -obj.string_length/2 + obj.string_length *
rand(number_scat,1);
        y = -obj.string_radius + 2*obj.string_radius *
rand(number_scat,1);
        z = -obj.string_radius +
2*obj.string_radius*rand(number_scat,1);

        % Find which scatterers are within the blood vessel
        r = (y.^2+z.^2).^(0.5);
        limInd=find(r > obj.string_radius);
        x(limInd)=[];
        y(limInd)=[];
        z(limInd)=[];
        r(limInd)=[];

        if obj.PoiseuilleFlow
           velocity = obj.string_vMax * (1-
              (r/obj.string_radius).^obj.string_vExponent);%        Poiseuille
formula
        else
           velocity = obj.string_vMax;
```

```
      end

      % Store the velocities along the x-axis for each of          the initial
scatterers positions:
      obj.scat_velocities = velocity;

      % Fill the initial scatterers position:
      positions  = [x, y, z];
      amplitudes = randn(size(x,1),1);

    end % generateInitalScatterers

    function [positions amplitudes] = displacementCallback(obj,   time,
init_positions, init_amplitudes)

      % Move the tube scatterers along the x axis with the          flow velocity
      translation_flow = obj.scat_velocities * time;
      obj.positions      = init_positions;
      obj.positions(:,1) = obj.positions(:,1) +           translation_flow;

      % amplitudes do not change
      obj.amplitudes = init_amplitudes;

      % Translate and rotate the tube in the desired  positions:
      translation = obj.origo;
      rotations   = obj.tilt;
      if max(abs(translation)) > 0 || max(abs(rotations))>0
        obj.rotate(rotations);
        obj.translate(translation);
      end
      positions  = obj.positions;
      amplitudes = obj.amplitudes;

    end % displacementCallback
```

```
        function plot(obj, time, scanShape, txEvent)
            args = {};
            if nargin > 1
                args{1} = time;
            end
            if nargin > 2
                args{2} = scanShape;
            end
            if nargin > 3
                args{3} = txEvent;
            end
            obj.generate(args{:});
            figure(gcf), clf
            colordef(gcf, 'black')
            scatter3(obj.positions(:,1)*1e3, ...
                   obj.positions(:,2)*1e3, ...
                   obj.positions(:,3)*1e3, ...
                   50, ...
                   20*log10(abs(obj.amplitudes)), ...
                   'filled' );
            axis equal; axis ij;
            title('Point scatterers');
            xlabel('Azimuth [mm]');
            ylabel('Elevation [mm]')
            zlabel('Range [mm]')
            colormap(hot);
        end % plot

    end

end
```

Appendix B

Code for plotting spectrum

```
clear all;
load('test6.mat');
addpath('C:\Users\zhou\Documents\MATLAB\functions') %get the logabs function used to plot the 2D spectrum
iq=iq_data;
[Nr,Nb,Nfr,Np]=size(iq);

figure(23); imagesc(squeeze(abs(iq(:,:,1,1))));
[x,y]=ginput(1); %get a point in the space
bb = round(x);%30; %Beam
fr=1; %frame
r =round(y); %round(Nr/5); %range
iq1=squeeze(iq(:,:,fr,:));
c=sim.propagation.material.c0;

%% velocity spectrum analysis
for i=bb:10+bb % velocity spectrum analysis
data1=iq1(r,i,:);
data1=squeeze(data1);
Nfft=128;
Nw=64;
w=hamming(Nw);
    q=data1;
    q = q.*w;
    q=abs(fftshift(fft(q,Nfft))).^2;
    a(:,i)=q;
end
Pc=sum(a,2)./10;

f1 = ([-0.5*Nfft:0.5*Nfft-      1]/Nfft )*(sim.scan.PRF)*c/(2*sim.probe.f0);
f2= ([-0.5*Nfft:0.5*Nfft-1]/Nfft )*(sim.scan.PRF)
t = linspace(0,size(Pc,2)/sim.scan.PRF,size(Pc,2));

%% plot velocity spectrum
figure(144);
Pc= Pc(:,round(size(Pc,2)/2));
Pc1=squeeze(10*log10(Pc/max(Pc)));
plot(f1./sin(sim.phantom.tilt(3)),Pc1,'r');ylabel('Normalized power
[dB]');xlabel('Velocity [m/s]');

%plot the 2D spectrum
figure(111);
gain=-215;
dyn=30;
freq = linspace(-3.125e6/2+5e6, 3.125e6/2+5e6,Nfft);
iq1=squeeze(iq(:,bb,fr,:));
image(freq,f2/sin(sim.phantom.tilt(3)),dyn*logabs(fftshift(abs(fft2(hamming(size(
iq1.',1))*hamming(size(iq1.',2)).'.*iq1.'))),gain,dyn));
colormap(gray(dyn));
colorbar;
hold on;
plot(freq,(2*0.25/1540*cos(32/180*pi)*freq), 'r','LineWidth',3); %0.25 is the
phantom velocity
title('2D power spectrum');xlabel('Transmit pulse frequency [Hz]');ylabel('Doppler
shift [Hz]');
[xx,yy]=ginput(2); %measure the bandwidths
```

logabs function used for 2D spectrum

```
function y=logabs(x,gain,dyn);
%LOGABS   Logaritmisk kompresjon av absoluttverdi

   y=abs(x); %Ta absoluttverdi
      y=y+1e-30;
      y=20*log10(y); %Gjer amplitude om til dB
   y=y+gain; %Legg til gain (i dB)
   y=y/dyn;  %Skaler dynamisk område
   y=max(0,y); %Sett negative verdiar lik 0
   y=min(1,y); %sett verdiar >1  lik 1
```