8-2012

Flow and pressure measurement using phase-contrast MRI : experiments in stenotic phantom models.

Iman Khodarahmi
University of Louisville

FLOWAND PRESSURE MEASUREMENT USING PHASE-CONTRAST MRI: EXPERIMENTS IN STENOTIC PHANTOM MODELS

By

Iman Khodarahmi
B.S., University of Tehran, 2006
M.S., University of Tehran, 2007
M.D., University of Tehran, 2007

A Dissertation
Submitted to the Faculty of the
J. B. Speed School of Engineering of the University of Louisville
in Partial Fulfillment of the Requirements
for the Degree of

Doctor of Philosophy

Department of Electrical and Computer Engineering
University of Louisville
Louisville, Kentucky
August 2012

FLOW AND PRESSURE MEASUREMENT USING PHASE-CONTRAST MRI: EXPERIMENTS IN PHANTOM MODELS

By

Iman Khodarahmi Qahnavieh

B.S., University of Tehran, 2006

M.S., University of Tehran, 2007

M.D., University of Tehran, 2007

A Dissertation Approved on

May 29, 2012

By the following Dissertation Committee

Dr. Amir Amini, Co-Director ______________________

Dr. Keith Sharp, Co-Director ______________________

Dr. John Naber ______________________

Dr. Aly Farag ______________________

Dr. Tamer Inanc ______________________

Dr. Pamela Woodard ______________________

DEDICATION

This dissertation is dedicated to my parents, who taught me that even the largest task can be accomplished by hard work and perseverance.

ACKNOWLEDGEMENT

This dissertation project would not have been possible without the support of many people. The author wishes to express his gratitude to his co-advisors, Dr. Amir Amini and Dr. Keith Sharp, who were abundantly helpful and offered invaluable assistance, support and guidance. Deepest gratitude is also due to the members of my dissertation committee, Dr. John Naber, Dr. Aly Farag, Dr. Tamer Inanc, and Dr. Pamela Woodard, without whose knowledge and assistance, this study would not have been successful. Special thanks also to my friend over the past four years, Dr. Mostafa Shakeri, whose experience and expertise was crucial for all steps of this research.

The author wishes to express his love and gratitude to his beloved family; for their understanding and endless love, through the duration of his studies.

ABSTRACT

FLOWAND PRESSURE MEASUREMENT USING PHASE-CONTRAST MRI: EXPERIMENTS IN STENOTIC PHANTOM MODELS

Iman Khodarahmi

May 29, 2012

Peripheral Arterial Disease (PAD) is a progressive atherosclerotic disorder which is defined as any pathologic process obstructing the blood flow of the arteries supplying the lower extremities. Moderate stenoses may or may not be hemodynamically significant, and intravascular pressure measurements have been recommended to evaluate whether these lesions are clinically significant.

Phase-contrast MRI (PC-MRI) provides a powerful and non-invasive method to acquire spatially registered blood velocity. The velocity field, then, can be used to derive other clinically useful hemodynamic parameters, such as blood flow and blood pressure gradients. Herein, a series of detailed experiments are reported for the validation of MR measurements of steady and pulsatile flows with stereoscopic particle image velocimetry (SPIV).

Agreement between PC-MRI and SPIV was demonstrated for both steady and pulsatile flow measurements at the inlet by evaluating the linear regression between the two methods, which showed a correlation coefficient of > 0.99 and > 0.96 for steady and pulsatile flows, respectively. Experiments revealed that the most accurate measures of flow by PC-MRI are found at the throat of the stenosis (error $< 5\%$ for both steady and pulsatile mean flows). The flow rate error distal to the stenosis was shown to be a function of narrowing severity.

Furthermore, pressure differences across an axisymmetric stenotic phantom model were estimated by solving the pressure-Poisson equation (iterative method) and a non-iterative method based on harmonics-based orthogonal projection using PC-MRI velocity data. Results were compared with the values obtained from other techniques including SPIV, computational fluid dynamic (CFD) simulations, and direct pressure measurements.

Using the pressure obtained from CFD as the ground truth and PC-MRI velocity data as the input, the relative error in pressure drop for iterative and non-iterative techniques were 13.1% and 12.5% for steady flow, 4.0% and 22.1% for pulsatile flow at peak-systole, and 194.5% and 155.2% at end-diastole, respectively. It was concluded that pressure drop calculation using PC-MRI is more promising for steady cases and pulsatile cases at peak-systole compared to pulsatile flow cases at end-diastole.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION AND MOTIVATION

In 2000, one in 16 adults aged 40 years and older in the United States was found to suffer from peripheral arterial disease (PAD). The age adjusted prevalence of PAD was determined to be 4% to 15%, and in the presence of cardiovascular risk factors it increases to 30%, totally affecting 5 million adults in the United Stated (Allison, Ho et al., 2007; Kasapis and Gurm, 2009).

PAD refers to the lipid deposition and the resulting inflammation in the endothelium of the arteries outside the heart; mainly the arteries supplying the lower extremities. The initiation and progression of atherosclerosis in PAD involves multiple factors such as platelet activation, thrombosis, endothelial dysfunction and vascular smooth muscle activation (Faxon, Fuster et al., 2004). Atherosclerotic plaque formation causes narrowing of the vessel lumen and blocks the circulation to the leg muscles and feet. Although PAD is a systemic process with high morbidity and mortality, the iliac, femoral, and popliteal arteries are more commonly affected (Allison, Ho et al., 2007; Kasapis and Gurm, 2009). Less than 20% of patients with PAD have typical symptoms of intermittent claudication i.e., leg muscle discomfort on exertion that is relieved by rest, or rest pain, ulceration or gangrene (Leng, Lee et al., 1996; Hirsch, Criqui et al., 2001), whereas another third have atypical exertional leg complaints (McDermott, Mehta et al., 1999).

Contrast angiography is considered as the gold standard to evaluate patients with PAD of lower limbs and provides thorough information about the arterial "anatomy". Therefore, it is generally recommended before any revascularization procedure. Nonetheless, single-plane angiography and, to some extent, triplane angiography are proved to be inaccurate in evaluating the hemodynamic significance of stenoses (Kinney and Rose, 1996).

Stenoses of 50% to 75% diameter determined by angiography may or may not be hemodynamically significant, and consequently, intravascular pressure measurements have been recommended to check whether these lesions are significant and to predict if the patient will benefit from a revascularization procedure (Udoff, Barth et al., 1979; Tetteroo, van Engelen et al., 1996; Kasapis and Gurm, 2009). According to the guidelines of the American College of Cardiology (ACC) and American Heart Association (AHA), a mean difference of 10 mmHg before or after vasodilators; or a mean difference of 5 mmHg and peak systolic difference of 10, 15 or 20 mmHg; or a peak systolic pressure difference of 15% (of peak systolic pressure) after administration of a vasodilator is considered hemodynamically significant (Hirsch, Haskal et al., 2006).

Angiography with a concomitant intravascular pressure measurement is an invasive procedure requiring arterial puncture, intravenous sedation, and close monitoring of the procedure to avoid serious complications, such as hemorrhage, infection, atheroembolism, dissection, pseudoaneurysm, hematoma formation, and arteriovenous fistula (Khodarahmi, Shakeri et al., 2010).

Hence, a non-invasive measurement of relative pressures from velocity-based methods, such as phase contrast MRI (PC-MRI) and Doppler ultrasound, has received much attention in the literature. Pressure gradients have been calculated by applying the Navier-Stokes equations to the velocity data obtained by PC-MRI and using iterative algorithms in both phantom (Yang, Kilner et al., 1996; Tyszka, Laidlaw et al., 2000; Nasiraei-Moghaddam, Behrens et al., 2004; Khodarahmi, Shakeri et al., 2010) and in-vivo models (Thompson and McVeigh, 2003; Lum, Johnson et al., 2007).

In this research, the accuracy of the PC-MRI derived velocity and flow data has been evaluated against other experimental and computational techniques in stenotic phantom models of the human iliac artery. In addition, pressure gradients have been calculated using an iterative method as well as a novel, faster non-iterative method, with the results compared against direct pressure measurements.

2 FLUID FLOW

A fluid, gas or liquid, is defined as a substance that has no shape and deforms easily under any external force tangential to it. Such a tangential force is called a shear force, and the shear stress is defined as the ratio of the so called shear force to the area on which it acts (Massey, Ward-Smith et al., 2006). Several of the more common fluid properties are introduced here:

Compressibility

Certain fluids under static conditions undergo very little change in density despite the existence of large pressures. These fluids are invariably in the liquid state for such behavior. Under such circumstances, the fluid is termed incompressible, and it is assumed during computations that the density is constant (Shames, 2003).

Viscosity

Viscosity is a fluid property which is responsible for the resistance of the fluid layers to move over each other. Viscosity is generally of high importance near solid boundaries because of the presence of a thin layer of high strain rate which is known as boundary layer. The fluid immediately adjacent to the boundary must move at the same speed as the boundary, which is called "no-slip" condition in fluid mechanics (Massey, Ward-Smith et al., 2006).

Steady vs. unsteady flow

Steady flow is defined as that in which the various parameters at any point do not change with time. In other words, when all the time derivatives of a flow field are zero, the flow is considered to be a steady flow. Flow in which a temporal change exists is called non-steady or unsteady. In practice, many problems may be simplified and studied effectively by assuming that the flow is steady. Such flows are called quasi-steady. By definition, turbulent flows are unsteady (Massey, Ward-Smith et al., 2006).

Laminar vs. turbulent flow

The flow of a fluid has been categorized into two different kinds from about 1840: laminar and turbulent. In laminar flow which is also called streamline flow, the particles of the flow in a straight pipe always follow the same straight lines in fully developed flow. The velocity of the particles moving along one line (so-called streamline) is not necessarily the same as that along other lines. Such fluid is considered to move in layers, or laminae, therefore, is called laminar flow. Compared to turbulent flow, this kind of flow occurs at the lower velocities.

In contrast, in turbulent flow, the paths of fluid particles are no longer straight everywhere but are intertwining and crossing one another in an irregular manner and as a result a complete mixing of the fluid occurs. From a fluid mechanics perspective, irregular, countless and haphazard secondary components are superimposed on the principal motion of the fluid in turbulent flow (Massey, Ward-Smith et al., 2006).

Newtonian vs. non-Newtonian fluids

A fluid is called Newtonian if the shear stress of the fluid is directly proportional to the velocity gradient. Many common fluids, such as air, water and oil are Newtonian. In contrast, whenever the relationship between shear stress and strain rate is non-linear the fluid is called Non-Newtonian. Non-Newtonian fluids usually have a complex molecular composition. Examples of non-Newtonian fluids are liquid plastics and blood (Potter and Wiggert, 2009).

Reynolds number

A dimensionless number, the Reynolds number (Re) is defined as:

$$\mathrm{Re} = \frac{\rho VD}{\mu} = \frac{VD}{\nu} \qquad [2.1]$$

where V, D, μ, ρ, and υ show the representative velocity, the characteristic length, the dynamic viscosity, the density, and the kinematic viscosity ($\upsilon = \frac{\mu}{\rho}$) of the fluid, respectively (Potter and Wiggert, 2009).

In evaluating the Reynolds number of a flow in a channel with circular cross section, the characteristic length is conventionally taken as the channel diameter and the representative velocity is the mean velocity. Usually for fluid flowing through pipes, a Re less than 2000 may be considered laminar and Re greater than 4000 may be regarded as turbulent. In the interval between 2000 and 4000, depending upon other factors, such as flow uniformity and pipe roughness, both laminar and turbulent flows are possible ("transitional" flows) (Massey, Ward-Smith et al., 2006).

Womersley number

The Womersley number is another dimensionless parameter in fluid mechanics that is generally considered as the pulsatile version of Reynolds number. It is a dimensionless expression of the pulsatile inertia of the flow in relation to viscous effects. The Womersley number, α, can be expressed as

$$\alpha = \mathrm{R}\left(\frac{\omega\rho}{\mu}\right)^{1/2} = \mathrm{R}\left(\frac{\omega}{\upsilon}\right)^{1/2} \qquad [2.2]$$

where:

R: a characteristic length scale (radius for a of a pipe),

ω: angular frequency of the oscillations,

υ: kinematic viscosity,

ρ: density,

μ: dynamic viscosity of the fluid (Gudbjartsson and Patz, 1995).

Conservation laws and Navier–Stokes equations

In the range of engineering precision, experiments have proved three basic laws for any fluid known as the conservation laws:

- Conservation of matter (continuity equation) which states that matter is indestructible. For the rectangular Cartesian coordinate system, with coordinates x, y, z, and corresponding velocity components u, v, w the continuity equation is

$$\frac{\partial \rho}{\partial \mathrm{t}} + \frac{\partial(\rho u)}{\partial x} + \frac{\partial(\rho v)}{\partial y} + \frac{\partial(\rho w)}{\partial z} = 0 \qquad [2.3]$$

where ρ is the fluid density.

- Conservation of momentum (Newton's second law) which states that if no external forces are acting on the system, the momentum of a system remains constant. The three components of the momentum equation are obtained after applying Newton's second law to an infinitesimal volume of fluid. The resulting three equations for the Cartesian coordinate system are

$$\begin{aligned}\frac{\partial(\rho u)}{\partial t}+\frac{\partial(\rho u^2)}{\partial x}+\frac{\partial(\rho uv)}{\partial y}+\frac{\partial(\rho uw)}{\partial z}&\\ =\rho f_x-\frac{\partial p}{\partial x}&\\ +\frac{\partial}{\partial x}\left[\mu\left(2\frac{\partial u}{\partial x}-\frac{2}{3}\left(\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}+\frac{\partial w}{\partial z}\right)\right)\right]&\qquad [2.4]\\ +\frac{\partial}{\partial y}\left[\mu\left(\frac{\partial u}{\partial y}+\frac{\partial v}{\partial x}\right)\right]+\frac{\partial}{\partial z}\left[\mu\left(\frac{\partial w}{\partial x}+\frac{\partial u}{\partial z}\right)\right]&\end{aligned}$$

$$\begin{aligned}\frac{\partial(\rho v)}{\partial t}+\frac{\partial(\rho uv)}{\partial x}+\frac{\partial(\rho v^2)}{\partial y}+\frac{\partial(\rho vw)}{\partial z}&\\ =\rho f_y-\frac{\partial p}{\partial y}&\\ +\frac{\partial}{\partial y}\left[\mu\left(2\frac{\partial v}{\partial y}-\frac{2}{3}\left(\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}+\frac{\partial w}{\partial z}\right)\right)\right]&\qquad [2.5]\\ +\frac{\partial}{\partial z}\left[\mu\left(\frac{\partial v}{\partial z}+\frac{\partial w}{\partial y}\right)\right]+\frac{\partial}{\partial x}\left[\mu\left(\frac{\partial u}{\partial y}+\frac{\partial v}{\partial x}\right)\right]&\end{aligned}$$

$$\begin{aligned}\frac{\partial(\rho w)}{\partial t}+\frac{\partial(\rho uw)}{\partial x}+\frac{\partial(\rho vw)}{\partial y}+\frac{\partial(\rho w^2)}{\partial z}&\\ &=\rho f_z-\frac{\partial p}{\partial z}\\ &+\frac{\partial}{\partial z}\left[\mu\left(2\frac{\partial w}{\partial y}-\frac{2}{3}\left(\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}+\frac{\partial w}{\partial z}\right)\right)\right]\\ &+\frac{\partial}{\partial x}\left[\mu\left(\frac{\partial w}{\partial x}+\frac{\partial u}{\partial z}\right)\right]+\frac{\partial}{\partial y}\left[\mu\left(\frac{\partial v}{\partial z}+\frac{\partial w}{\partial y}\right)\right]\end{aligned} \qquad [2.6]$$

where p is the hydrostatic pressure, μ is the dynamic viscosity, and f_x, f_y, f_z show the Cartesian components of external forces.

- Conservation of energy i.e. first law of thermodynamics, which states that the total internal energy of an isolated system remains unchanged.

$$\begin{aligned}\frac{\partial(\rho e)}{\partial t}+\frac{\partial(\rho ue)}{\partial x}+\frac{\partial(\rho ve)}{\partial y}+\frac{\partial(\rho we)}{\partial z}&\\ &=\rho q+\frac{\partial}{\partial x}\left(k\frac{\partial T}{\partial x}\right)+\frac{\partial}{\partial y}\left(k\frac{\partial T}{\partial y}\right)+\frac{\partial}{\partial z}\left(k\frac{\partial T}{\partial z}\right)\\ &-p\left(\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}+\frac{\partial w}{\partial z}\right)-\lambda\left(\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}+\frac{\partial w}{\partial z}\right)^2\\ &+\mu\left\{2\left[\left(\frac{\partial u}{\partial x}\right)^2+\left(\frac{\partial v}{\partial y}\right)^2+\left(\frac{\partial w}{\partial z}\right)^2\right]\right.\\ &\left.+\left(\frac{\partial v}{\partial x}+\frac{\partial u}{\partial y}\right)^2+\left(\frac{\partial w}{\partial y}+\frac{\partial v}{\partial z}\right)^2+(\frac{\partial u}{\partial z}+\frac{\partial w}{\partial x})^2\right\}\end{aligned} \qquad [2.7]$$

where λ is the bulk viscosity, e is the internal energy, q is the heat addition per unit mass and k is the thermal conductivity. Full derivation of these equations can be found in (Shames, 2003; Massey, Ward-Smith et al., 2006; Potter and Wiggert, 2009).

The momentum conservation equations derived by Claude-Louis Navier (1822) and George Gabriel Stokes (1842), are referred to as the Navier–Stokes equations. For an incompressible Newtonian fluid with hydrostatic pressure p, velocity vector field **v**, density ρ, dynamic viscosity μ, and body forces **f** (such as gravity), if temperature effects are neglected, continuity and Navier-Stokes equations take the following form (Massey, Ward-Smith et al., 2006):

$$\nabla.\mathbf{v} = 0$$

$$\nabla \mathrm{p} = -\rho\frac{\partial \mathbf{v}}{\partial \mathrm{t}} - \rho(\mathbf{v}\cdot\nabla)\mathbf{v} + \mu\nabla^2\mathbf{v} + \rho\mathbf{f} \qquad [2.8]$$

3 VELOCITY MEASUREMENT TECHNIQUES

There are several methods of flow visualization and quantitative velocity measurements in experimental fluid dynamics:

- **Tracer methods:** Particle tracking is the most common fluid velocity measurement technique. It involves inferring the velocity of fluid at a particular point and time from measurement of the motion of small particles mixed with fluid. A particle tracking measurement system consists of three components, namely, an illumination source, tracing particles, and an observation system. Assuming that the particles faithfully follow the streamlines of the flow, flow velocity could be measured. Laser Doppler Velocimetry (LDV) and Particle Image Velocimetry (PIV) are two generic types of tracer methods (Emrich, 1981).
- **Probe methods:** Examples of these methods are Pitot probe, propeller and vane anemometer and hot-wire and hot-film anemometers. These methods in contrast to tracer methods are invasive and less sensitive at low fluid velocities (Emrich, 1981).
- **Doppler based methods:** The principle of the Doppler effect is based upon transmitting a ultrasound beam with a certain frequency and a well-known angle through a liquid. Solid particles or gas bubbles carried in the liquid reflect a part of the ultrasound energy. A frequency shift is observed in the reflective beam due to the movement of the particles. This frequency shift corresponds to the velocity

of the particles. Doppler ultrasound (Niederer, 2010), laser Doppler flowmetry (Rajan, Varghese et al., 2009), optical Doppler tomography (Chen, Milner et al., 1997) are examples of such methods.

- **Phase-Contrast Magnetic Resonance Imaging:** MRI is very prone to motion such that motion artifacts are the most common causes of image degradation. Blood and CSF flows are among the involuntary movements, resulting in a variety of flow effects. Although methods such as "flow compensation" are used to eliminate the flow artifacts, the same concept can be used advantageously to develop non-invasive techniques to image the vascular anatomy and/or to measure the blood velocity. This technique is called Magnetic Resonance Angiography (MRA) and are classified into three major categories: Contrast enhanced angiography (anatomy), Time-of-flight (anatomy and flow velocity), and Phase contrast MR angiography (PC-MRI) (anatomy and flow velocity) (Hornak, 2008). Compared to nuclear medicine and radiographic techniques, MR based flow quantification does not involve use of ionizing radiation. Furthermore, contrast mechanisms independent of contrast agents are available for MRA. Compared to Doppler based methods, on the other hand, MRI can be used to measure the blood velocity using arbitrary image plane orientations without restrictions such as acoustic windows. More importantly, MR based flow quantification techniques can provide all three components of the velocity vector, compared to the single component (in the direction of insonification) obtained from Doppler ultrasound.

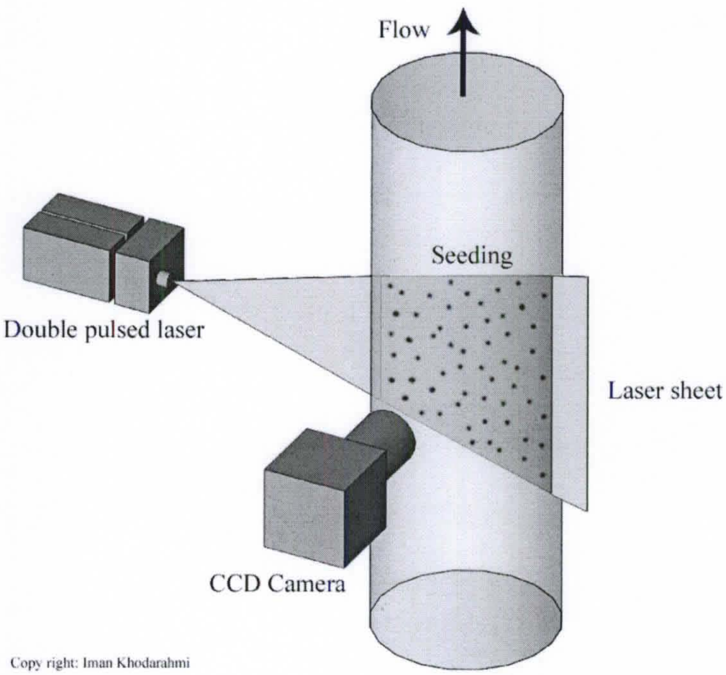
From the aforementioned techniques, PIV and PC-MRI are chosen for flow measurement in this research and will be discussed further.

3.1 Particle Image Velocimetry

Particle Image Velocimetry (PIV) is a noninvasive tracer method for measuring flow velocity in a fluid field. In contrast to single point measurement techniques, PIV can concurrently acquire two-dimensional velocity information across an entire plane making it possible to detect in-plane two-dimensional flow structures with excellent spatial and temporal resolution. This makes it particularly valuable for time-dependent flows. The liquid is seeded with tracer particles which, because of their small size, are assumed to faithfully follow the flow streamlines. The fluid with entrained particles is illuminated usually by a laser light source in any desired plane so that particles are visible. The displacement of the particles is used to calculate speed and direction of the flow (Adrian, 2005).

3.1.1 Principles of two-component PIV (2C-PIV)

Based on the definition of velocity which is the first time derivative of position, PIV technique measures the displacement of the fluid (or particles which faithfully follow the streamlines) over a known time interval to derive the velocity. The displacement of the fluid elements is imaged through the light scattered by liquid or solid fluorescent particles illuminated by a laser light sheet. Such particles are not usually present in the fluid; therefore, the liquid has to be seeded with such tracer particles. These particles should be small enough and have the same density of the fluid to follow the local flow velocity patterns (Brossard, Monnier et al., 2009). Figure 3-1 illustrates a typical standard two-component PIV (2C-PIV) system.



Copy right: Iman Khodarahmi

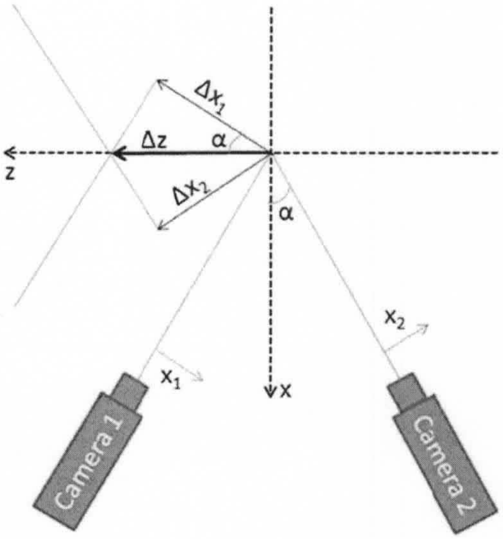
Figure 3-1. Standard PIV system.

Different components of a standard PIV system are:

- **Light source:** Light source is composed of two independent laser cavities (most commonly pulsed Nd:YAG laser), but the laser beams should be superimposed in both the near and far fields so that the two laser sheets illuminate the exact same location.
- **Imaging device:** Digital photography is performed via a Charge-Coupled Device (CCD) sensor which based on the photoelectric effect converts photons to an electric charge. The CCD sensor consists of many individual sensors in a rectangular array arrangement. The plane of interest within the flow is illuminated twice by the two laser light sheets and the light scattered by the particles is images by the CCD camera sensor on two separate frames.
- **Processing:** Each digital PIV image is divided in small rectangular areas called "interrogation windows". Then using a spatially statistical cross-correlation function, the local two component (2C) displacement vector of the particle images between the two illuminations is determined for each interrogation mask. Knowing the time interval between the two laser pulses and the image magnification obtained from a calibration process, the "projection" of the local flow velocity vector onto the plane of the light sheet can then be deduced. This projection accounts for the name two-component in this standard PIV method. Further details of the PIV technique can be found in (Brossard, Monnier et al., 2009).

3.1.2 Principles of stereoscopic PIV (3C-PIV)

Out-of-plane component of the velocity vector, which is the component perpendicular to the laser light sheet can measured by adding a second camera to the system, and also arranging both cameras at different viewing angles. The resulting system is called three-component (3C) or Stereoscopic-PIV (SPIV). SPIV is a well-known technique to measure all three-components (3C) of the velocity vector in the plane of the laser light sheet. Similar to depth perception in human vision, SPIV uses two cameras that look to the laser sheet at different angles, each camera measuring the displacement of the seeding particles perpendicular to its viewing angle. These two different projections of the velocity, one from each camera, then can be combined to reconstruct the 3C velocity vector. This is illustrated in Figure 3-2, where an arbitrary local coordinate system is used. Following the notation of van Doorne and Westerweel (van Doorne and Westerweel, 2007), the x- and z- axes lay in the plane defined by the measurement point and the two cameras, and the x-axis divides the angle (2α) between the two cameras in two equal halves. The y_1- and y_2-axes of cameras 1 and 2 respectively are chosen to be parallel to the y-axis of the above defined coordinate system.



Copy right: Iman Khodarahmi

Figure 3-2. Illustration of the principle of SPIV.

If cameras 1 and 2 measure the projected displacements $(\Delta x_1, \Delta y_1)$ and $(\Delta x_2, \Delta y_2)$ respectively, then the real displacement vector $(\Delta\mathrm{x}, \Delta\mathrm{y}, \Delta\mathrm{z})$ can be calculated if the projection angle is known. When the particle displacements are much smaller compared to the distance to the camera lenses, (which is called a paraxial approximation), the displacement vector (Δx, Δy, Δz) reconstructed to be:

$$\Delta x = \frac{\Delta x_1 - \Delta x_2}{2\sin\alpha}$$

$$\Delta y = \frac{\Delta y_1 + \Delta y_2}{2} \qquad [3.1]$$

$$\Delta z = \frac{-\Delta x_1 - \Delta x_2}{2\cos\alpha}$$

In practice, first the two-component (2C) vector fields of the particle displacements observed by each camera are evaluated by standard PIV cross-correlation methods. Then, for the calculation of the 3C-vector fields, the 2C vector fields must be mapped (dewarped) from the image planes onto the real-world plane of the light sheet and interpolated on a rectangular grid. Then the displacement vectors from both cameras are combined to reconstruct the three components of the particle displacement. The dewarping and reconstruction can in principle be based on the exact knowledge of the geometry of the setup, but most often they are based on a calibration procedure using a calibration target with known geometry. A comprehensive description of the principles of SPIV can be found in (Prasad, 2000).

3.2 Phase Contrast MRI

Phase contrast (PC) methods rely on the fact that moving spins experience different magnetic field gradients compared to static spins. Consequently, their accumulated phase would be different from static spins. As explained below, the phase shift is proportional to the product of flow velocity and the first moment of the magnetic field gradient in the direction of flow. Phase-based methods are particularly suited to applications requiring flow quantification. The velocity field obtained from PC-MRI can then be used to derive other clinically useful hemodynamic parameters, such as wall shear stress and blood pressure gradient (Nasiraei-Moghaddam, Behrens et al., 2004; Frydrychowicz, Stalder et al., 2009; Harloff, Nussbaumer et al., 2010). PC-MRI has been applied in several clinical scenarios, such as evaluation of aortic coarctation and dissection, valvular heart abnormalities, peripheral arterial diseases and congenital shunt lesions, as well as quantification of cardiac function (Szolar, Sakuma et al., 1996; Srichai, Lim et al., 2009).

3.2.1 Physical Principles of PC-MRI

The imaged quantity in MRI is the effective spin density which is proportional to transverse magnetization in each voxel. This transverse magnetization due to some factors such as field inhomogeneity, and tissue magnetic susceptibility is a complex quantity having both magnitude and phase in the rotating frame of reference. MR images usually contribute only the magnitude of the magnetization (magnitude image). However, phase images can contain information about motion, and it is the phase that provides the velocity dependent signal in PC MRI. Detailed desription of PC-MRI can be found in (Pelc, Sommer et al., 1994).

For an arbitrary gradient $G(t)$ and position $r(t)$ in the direction of G, the frequency offset from the resonance frequency at any moment is $\gamma G(t)r(t)$, and the accumulated phase at the echo delay time (TE) is

$$\varphi = \gamma \int_0^{TE} G(t)r(t)dt. \qquad [3.2]$$

where γ is the magnetogyric ratio. In PC-MRI this arbitrary gradient is usually a bipolar gradient lobe. The position vector can be described by a Taylor expansion about initial position:

$$r(t) = r_0 + vt + a\frac{t^2}{2} + \cdots \qquad [3.3]$$

If the gradient waveforms are constructed to provide no phase shifts from initial position (r_0) (which is the case for bipolar lobes) and if phase shifts due to higher order terms are small, Eq [3.2] becomes:

$$\varphi = v(\gamma M_1), \qquad [3.4]$$

where M_1 represents the first moment of the gradient:

$$M_1 = \int_0^{TE} G(t)t\, dt. \qquad [3.5]$$

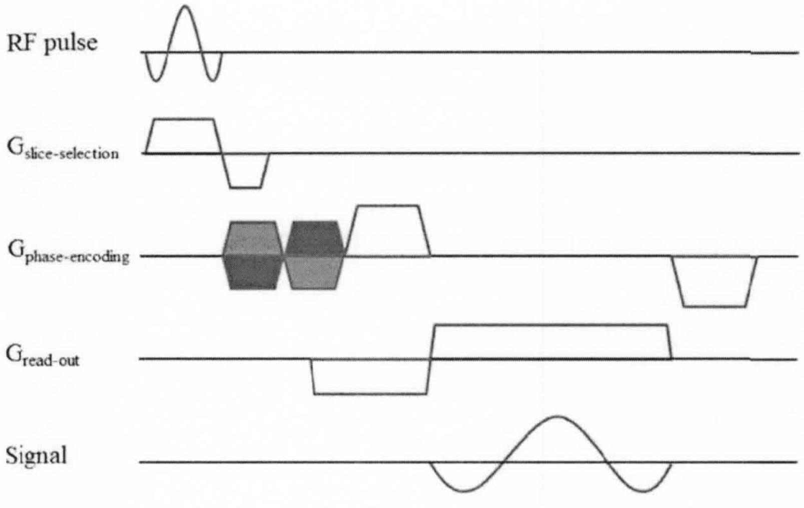
Equation [3.4] shows that the motion-induced phase accumulation is proportional to velocity and that the proportionality constant is determined by the first moment of the gradient waveform. This relationship is the foundation for not only phase contrast imaging, but also Fourier velocity encoding method and artifact reduction by first moment nulling as well (known as flow compensation). An important assumption in deriving this expression is that the velocity is constant during the echo time (TE).

As noted before, conventional MRI produces images of the magnitude of the effective spin density (transverse magnetization) in each voxel. Equation [3.4] suggests that images of the phase of magnetization could provide some information about spin velocity. Unfortunately, the phase offset can be altered by many other phenomena, including tissue magnetic susceptibility, magnetic field inhomogeneity, and motion in directions other than the interested one. Therefore, to precisely extract the effect of motion, phase shifts due to these sources have to be eliminated. In PC acquisitions this elimination is performed by performing a dual measurement with a different M_1 (and therefore different velocity-induced phase shifts), but with the same phase effects due to other sources, and then subtracting the results. If two complete acquisitions with different M_1 values are performed, the phase difference in each voxel corresponds to the spin velocity as:

$$\Delta\varphi = v(\gamma \Delta M_1), \qquad [3.6]$$

where ΔM_1 is the change in first moment. The image produced by one acquisition is taken as phase reference, and is used to correct for undesired phase offsets of the second image acquired with modified flow sensitivity (Haacke, 1999).

Measurement of velocity in each direction requires two pulse sequences with different gradient moments in that direction. An example of a PC gradient echo pulse sequence is shown in Figure 3-3.



Copy right: Iman Khodarahmi

Figure 3-3. Phase-contrast pulse sequence.

In the above figure, the bipolar lobes (blue) are added to the phase-encoding direction to encode velocity in this direction. The same sequence is repeated with the reverse bipolar lobes (red), and then the phase offsets are subtracted from each other. To measure velocity in other directions, the same procedure should be repeated in other directions i.e. slice-selection and read-out directions.

Figure 3-4 shows the magnitude (a) and phase images (b) at a section through the abdomen. In the phase image, the gray intensity of each pixel represents the velocity value in that pixel. White values show flow away from the viewer, whereas black values show flow toward the viewer (abdominal aorta in this case) (Lotz, Meier et al., 2002). Sensitivity to flow in all x, y, and z directions requires at least four measurements (Pelc, Bernstein et al., 1991; Haacke, 1999).

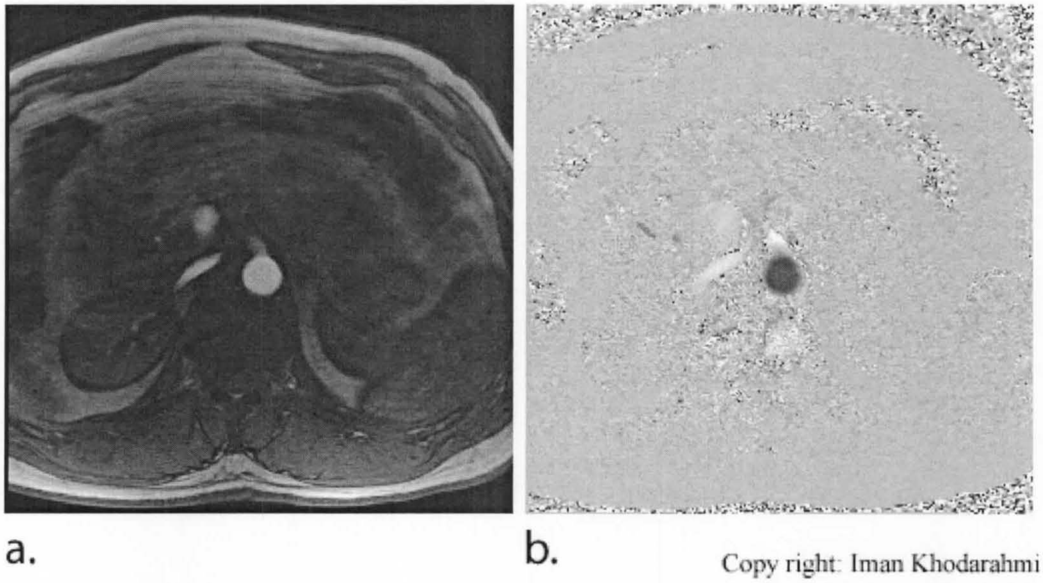


Figure 3-4. Sample PC-MRI image.

An important parameter in PC-MRI is the strength of the flow encoding which is determined by the first moment of the gradient used in the pulse sequence (ΔM_1) and controls the amount of phase shift produced by a given velocity. A commonly used parameter is the velocity that produces a phase shift of π radians (180°). This velocity is often referred to as the encoding velocity v_{enc}:

$$v_{enc} = \frac{\pi}{\gamma \Delta M_1}, \qquad [3.7]$$

and a phase shift of $\Delta\varphi$ is converted to velocity as

$$v = \Delta\varphi \left(\frac{v_{enc}}{\pi}\right). \qquad [3.8]$$

The velocity encoding (v_{enc}) defines the velocity range that is free of aliasing (wrap around artifact) and the noise in the velocity measurements. Three dimensional

PC-MRI is a natural extension, which requires only adding a spatial phase encoding gradient to the slice select axis.

3.2.2 Challenges in Phase Contrast Imaging

- *Aliasing*

As equation [3.6] shows a phase image is linear mapping between velocity and phase shift. However, since phase shifts are to be in the interval of (-180,180), velocities higher than v_{enc} that produce phase shifts greater than 180 will be mapped to incorrect phases. Thus, for example, a phase change of 210° (velocity = 1.3 v_{enc}) is indistinguishable from one of -150° (velocity = -0.7 v_{enc}), a phenomenon called velocity aliasing. Although, if the multiple of 360° or 2 v_{enc} is known, the effect can be corrected at a post-processing step, this is generally not the case.

Velocity aliasing might be corrected by use of algorithms that use spatial continuity to resolve the ambiguity (Moon-Ho Song, Napel et al., 1995).

- *Eddy currents*

Because of difference schema used in PC MRI, and the fact that most of the gradient lobes are common between reference and sensitivity measurements, eddy current induced phase shifts are restricted to bipolar lobes. However, they could be sources of inaccuracy in velocity measurements. Shielded gradient coils and algorithms to compensate eddy current-induced errors has been worked well to decrease this source of error (Pelc, Sommer et al., 1994).

- *Reproducibility*

The reproducibility of in vivo velocity measurements can be restricted by a variety of factors. Apart from some errors beyond the control of PC MRI sequence (e.g., physiological variation and patient positioning), reproducibility is inversely proportional to the variance of the measured velocity. The noise in the measured velocity depends on the noise in the phase measurements and on the velocity encoding. The standard deviation in the measured velocity, σ_v, is:

$$\sigma_v = \left(\frac{\sqrt{2}}{\pi}\right)\frac{v_{enc}}{SNR} \qquad [3.9]$$

where SNR is the signal-to-noise ratio in a single image. Therefore, reproducibility can be improved by increasing the SNR and decreasing v_{enc} (Haacke, 1999).

- *Acceleration*

The fundamental relationship between phase shift and velocity in equation [3.4] was derived assuming that terms of 2^{nd} order and higher in the Taylor expansion of position could be ignored. As a result, significant acceleration, jerk and higher order motions could be considered as potential sources of error. For a general case of bipolar lobes, equation [3.4] should reflect all of the terms in the Taylor series expansion of position and is changed to:

$$\varphi = v(\gamma M_1) + a\left(\frac{1}{2}\gamma M_2\right) + \cdots \qquad [3.10]$$

where M_1, M_2, v, and a are the first moment of the gradient waveform, the second moments of the gradient waveform, the velocity and the acceleration, respectively.

Assuming no unwanted change in two successive signal acquisitions, with subtracting the two measured phases, equation. [3.10] changes to:

$$\Delta\varphi = v(\gamma\Delta M_1) + a\left(\frac{1}{2}\gamma\Delta M_2\right) + \cdots \qquad [3.11]$$

Therefore, the measured velocity is

$$\hat{\mathrm{v}} = \mathrm{v} + \mathrm{a}\left(\frac{\Delta \mathrm{M}_2}{2\Delta \mathrm{M}_1}\right) + \cdots \qquad [3.12]$$

which is equal to the actual velocity only if higher order terms are minute. As shown, acceleration appears as a velocity error proportional to the change in second moment divided by the change in first moment. Equation [3.12] can be rewritten as:

$$\hat{\mathrm{v}} = \mathrm{v} + \mathrm{a}\bar{\mathrm{t}} \qquad [3.13]$$

where $\bar{\mathrm{t}}$ depends upon the gradient waveform geometry and can be shown to be essentially the time interval between RF pulse and flow encoding lobes. Hence, the effect of acceleration is similar to the effects due to the fact that slice encoding, phase encoding, and frequency encoding gradients are applied at different times in the pulse sequence. In fact, since the effective flow encoding time ($\bar{\mathrm{t}}$) is less than the echo time (TE), it can be concluded that the acceleration effect on flow encoding is less than the corresponding spatial mismapping between slice selection and frequency encoding gradients (Pelc, Sommer et al., 1994).

Acceleration in general (Lagrangian description) can be divided into two components:

$$\frac{dV}{dt} = \frac{\partial V}{\partial t} + (V.\nabla)V \qquad [3.14]$$

First component $(\frac{\partial V}{\partial t})$ is the rate of change of velocity at a fixed point in space, which is called time-dependent acceleration and can be thought of as pulsatility (Eulerian description). It has been showed that this acceleration is too small to cause significant errors for physiological flow waveforms (Firmin, Nayler et al., 1990). Furthermore, because of periodicity of the flow waveform due to cardiac cycles, the average value of the time-dependent acceleration is zero. Consequently, although errors are introduced at individual time frame velocities, the error in the measured time-averaged flow is zero.

The second component $((V.\nabla)V)$ of acceleration reflects the changing velocity of spins as they move through the vasculature and is called convective acceleration. Therefore, even for steady flow, convective acceleration can be nonzero. It should be emphasized that PC MRI represents a Lagrangian measurement of the velocity map. Therefore, the effect of convective acceleration can be prominent as spins move through sharp bends or stenoses. This error leads to some under/over estimations in flow measurements of stenotic vessels (Siegel, Oshinski et al., 1996).

- *Partial volume effect*

Partial volume effects are perhaps the most serious limitations to the accuracy of PC measurements. Consider a voxel of the image in which half of the spins are static while the other half move at a uniform speed. The MR signal measured for this voxel is the vector sum of the signals from stationary and moving spins and, as shown in (Pelc, Sommer et al., 1994) is the intermediate between the phase shift of moving spins and zero (the phase shift of the static spins). The exact value of the measured signal depends on the relative signal intensities of the static and moving spins.

If the goal is the measurement of the velocity of the moving spins regardless of the presence of static tissue in the voxel, the signal intensity from static tissue should be kept as low as possible. This could be possible by applying chemical shift saturation to suppress the signal from static lipids.

However, if the goal is to measure the volume flow, static and moving spins are desired to contribute to the measured phase shift in proportion to their volume fraction. If the stationary and moving spins generate equal MR signal intensity relative to their volume fraction, the situation is the best for flow measurement. However, in practice, low signal in the vessel wall and surrounding tissues, and flow-related enhancement causes flowing spins to produce a fraction of the total signal that is much greater than their volume fraction (Pelc, Sommer et al., 1994).

- *Pulsatility*

Errors evoked by pulsatility, that is the variable flow throughout the scan, can be viewed as being a partial volume effects in time. The reconstructed image at the pixel is proportional to the average magnetization at the corresponding voxel throughout the entire scan time. The voxel can be considered as containing spins at multiple velocities in which each velocity value reflects the velocity of the voxel at a certain time point in the pulse sequence. Therefore, the effect of velocity variation during the repetition time (TR) is a partial volume effect in the temporal direction. Similar to spatial partial volume effects, overestimation of the average velocity may occur if the signal magnitude during periods of high velocity is higher than normal due to flow-related enhancement. On the other hand, the average velocity may be underestimated if the high velocity periods are associated with signal loss due to intravoxel dephasing (Hangiandreou, Rossman et al., 1993; Pelc, Sommer et al., 1994).

- *Chemical Shift*

If the imaged vessel is surrounded by adipose tissue, the signal from fat can be displaced in the frequency-encoded direction because of its chemical shift and overlap the vessel lumen. This situation aggravates especially in rapid imaging methods in which short slice-selection excitations are applied (Nayak, Hu et al., 2003). Though physically in two different locations, signal from static lipid would contribute to that of flowing blood. This chemical shift induced interference leads to some underestimation in the measured velocity and flow (Pelc, Sommer et al., 1994).

4 EXPERIMENTAL TECHNIQUES

4.1 Flow System

The experiments were carried out in a closed-loop flow system. A schematic diagram of the flow apparatus is shown in Figure 4-1. The flow system includes a CardioFlow 1000 programmable pump (Shelley Medical Imaging Technologies, London, Ontario, Canada) capable of producing flows with a range of waveforms with less than 1% variance (Holdsworth, Rickey et al., 1991; Frayne, Holdsworth et al., 1992).

Idealized rigid models of axisymmetric Gaussian shape were machined from transparent acrylic, initially specified at 50%, 75%, and 90% area occlusion. Later, the exact geometry was measured with high resolution CT scans (see next section) and the area occlusions were found to be 50%, 74%, and 87%, respectively.

The inlet diameter for all three models was 1.000 inch (Figure 4-2). To ensure a fully developed laminar flow at the entrance of the model, a 75-cm long straight rigid acrylic tube was placed upstream of the test section. To reduce optical refraction mismatches between the fluid and acrylic, the stenosis model was contained in an enclosure filled with the same index of refraction-matched fluid used inside the flow loop. The water-bath enclosure was machined to provide several viewing windows with flat surfaces to reduce optical distortion due to phantom curvatures. The experimental system was designed to allow easy replacement of the model stenosis with other vascular models of matching diameter at entrance and exit inside the water-bath enclosure.

A solution of glycerol and water (60:40, w/w) was prepared and then sodium iodide was added to match its refractive index with that of acrylic. The viscosity of the final solution was measured using a LVT Cone-Plate viscometer (Brookfield Labs., Stoughton, MA, USA) to be 0.022 Pa.s at 68°F. The final density of the solution was 1600 kg/m^3. These values are to be compared to 1060 kg/m3 and 0.003-0.004 Pa.s for blood (Brunette, Mongrain et al., 2008). The T1 and T2 values of the final solution were 500 and 45 ms, respectively at 3 T.

For steady flows, inlet Reynolds numbers (Re) of 190, 160 and 130 corresponding to the range of Reynolds number typically encountered in the human common iliac artery (mean ± 20%) were used for both sagittal and axial studies.

Flow in the human common iliac artery was modeled, since it is one of the most common sites for development of atherosclerotic lesions. Furthermore, another steady flow with inlet Reynolds number of 550 (mimicking the human renal artery) was used in axial studies to study the validity of PC-MRI at high velocities (Table 4-1).

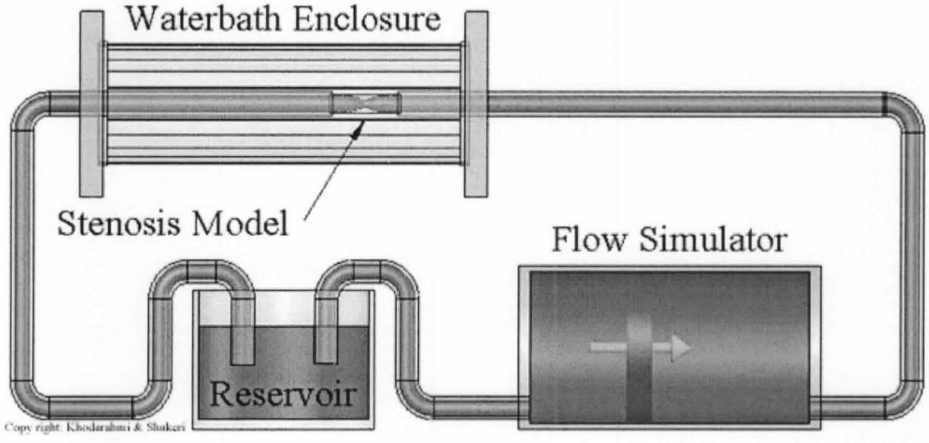
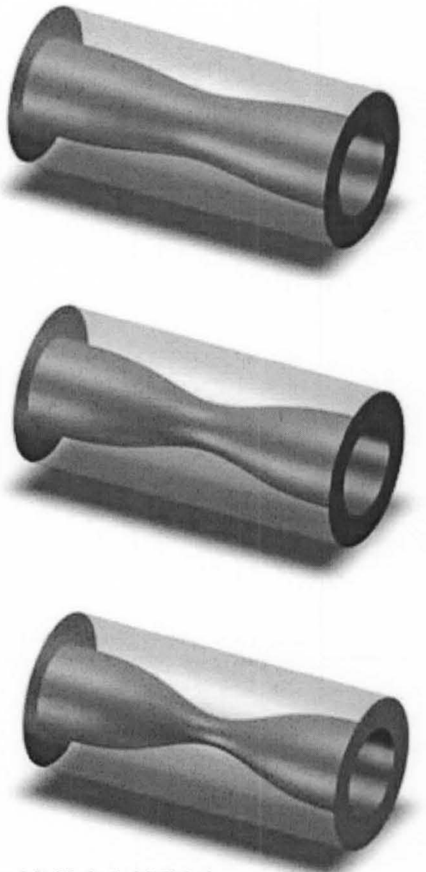


Figure 4-1. Schematic diagram of the flow apparatus.



Copy right: Khodarahmi & Shakeri

Figure 4-2. Solid model of the 50%, 74%, and 87% stenoses.

In addition to steady flows, three biphasic pulsatile flows (see results for the shape) with the same mean Reynolds numbers (i.e., 190, 160 and 130) with peak Reynolds number of 360, 300 and 250 were also studied (Table 4-1). At a frequency of 60 beats/min, the Womersley number was 8.6, similar to that in the human common iliac artery. By matching both the Reynolds and the Womersley numbers to those of the human iliac artery, similarity with physiologic flows is ensured.

Table 4-1
Steady and pulsatile flow cases with the corresponding Reynolds numbers (Re).

Steady flows			**Pulsatile flows, Womersley number 8.6**			
Steady flow	**Flow rate (ml/s)**	**Inlet Re**	**Pulsatile flow**	**Mean inlet flow rate (ml/s)**	**Mean inlet Re**	**Peak inlet Re**
SF-0	160	550				
SF-1	56.3	190	PF-1	56.3	190	360
SF-2	46.9	160	PF-2	46.9	160	300
SF-3	37.5	130	PF-3	35	130	250

4.2 CT Studies

To extract the exact geometry of the stenotic models for fluid dynamic simulations and also to determine the exact degree of the stenosis of each phantom, Computed Tomography (CT) scans of all phantom models were performed on a 64-slice multi-detector CT scanner (LightSpeed VCT scanner, GE Medical Systems, Milwaukee, WI, USA). Imaging parameters are summarized in Table 4-2:

Table 4-2
Imaging parameters for CT studies.

Scan option / Spiral pitch	Helical mode / 0.52
Slice thickness / Pixel Spacing	0.625 mm / 0.22 mm × 0.22 mm
Image size	512 × 512
Coverage	1400 mm
kVp/Exposure	140/3mAs
Tube current	295 mA
Exposure time	730 ms

The exact stenosis for 50%, 75%, and 90% models are 50.44 ± 5.94%, 74.06 ± 3.96%, and 87.46 ± 2.56%, respectively.

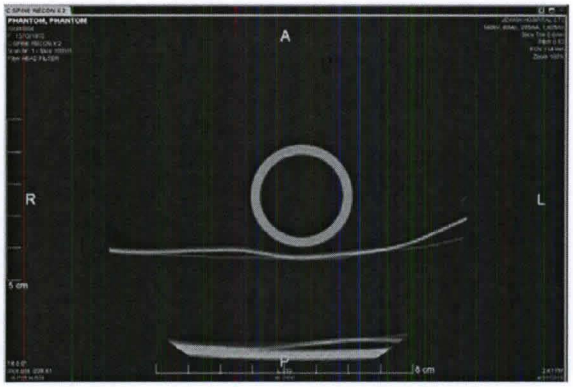


Figure 4-3. A sample CT image of the 90% phantom at the stenosis throat showing the circular cross section of the phantom.

4.2.1 Uncertainty in CT measurements

Canny edge detector was used on axial CT images to determine the radius at each cross section. The stenosis severity was then calculated using:

$$S\,(\%) = 100 \times (1 - \frac{r_1^{\,2}}{r_2^{\,2}}) \qquad [4.1]$$

where r_1 and r_2 are the radii at the throat and inlet sections, respectively. Propagation of uncertainty in radius measurement causes an error in the stenosis measurement according to:

$$\Delta S\,(\%) = 100 \times \sqrt{\frac{4r_1^{\,2}}{r_2^{\,4}}(\Delta r_1)^2 + \frac{4r_1^{\,4}}{r_2^{\,6}}(\Delta r_2)^2} \qquad [4.2]$$

Based on the above calculations, the uncertainty in the measurement of stenosis severity is as follows: 50.44 ± 5.94% for 50% model, 74.06 ± 3.96% for 75% model, and 87.46 ± 2.56% for 90% model.

4.3 MRI Studies

Phase-Contrast MRI measurements were conducted in a 3 T TX MRI scanner (Achieva R3.2.1, Philips Healthcare, Best, Netherlands) with a maximum gradient strength of 80 mT/m and a slew rate of 200 T/m/s using an eight-channel knee coil. The imaging pulse sequence (Figure 4-4) consisted of a slice-selective excitation, phase encoding gradients, a bipolar gradient, a Cartesian readout, flow compensation gradients for all three directions, and gradient spoiler. ECG output of the pump with frequency of 1 Hz (heart rate of 60 beats/min) was used to retrospectively gate for 40 cardiac phase acquisitions, corresponding to a temporal resolution of 25 milliseconds.

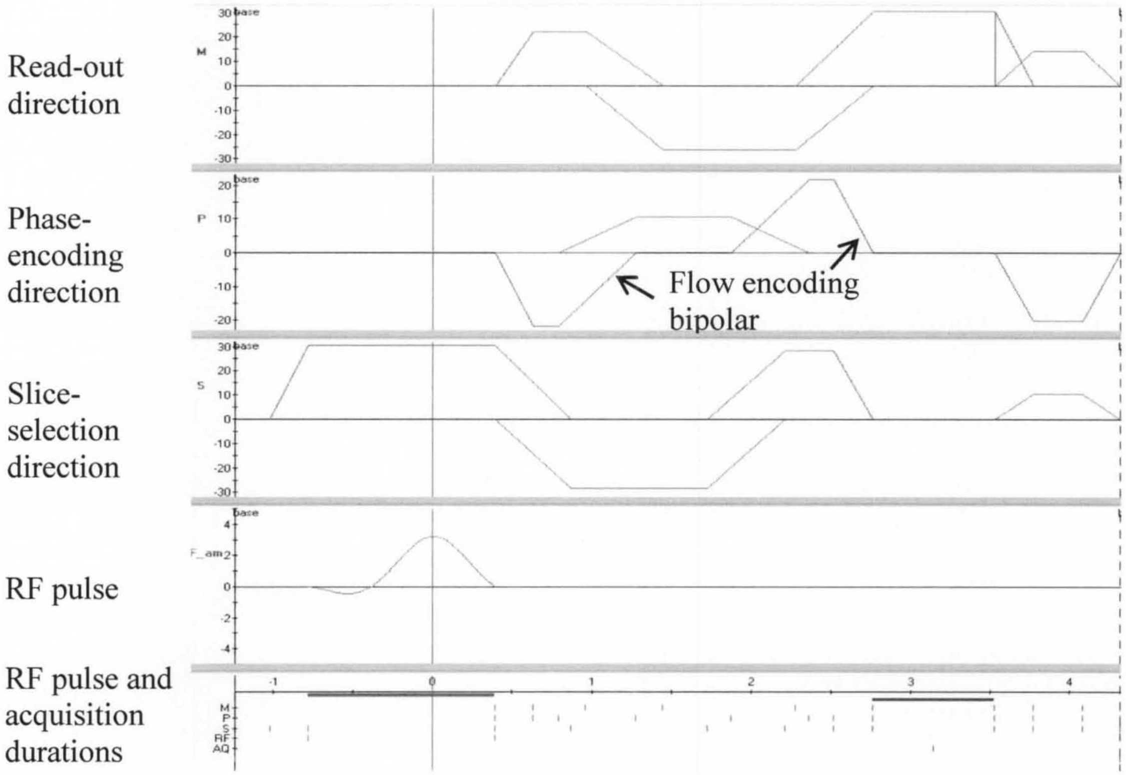


Figure 4-4. Pulse sequence diagram for phase contrast images.

Two sets of experiments were performed for axial and sagittal directions:

4.3.1 Sagittal imaging

Sagittal imaging was done for four different zones to cover the whole length of the phantom with the same encoding-velocity (Venc) (Figure 4-5). Field of View (FOV) was 192 by 64 mm in readout and phase-encoding directions, respectively. The linear part of the coil was determined to be 100 mm, therefore, an overlap of about 45 mm was chosen between consecutive images. Registration of the images was performed using a narrow tube filled with contrast material (gadolinium) rolled around the phantom. Other parameters were as follow:

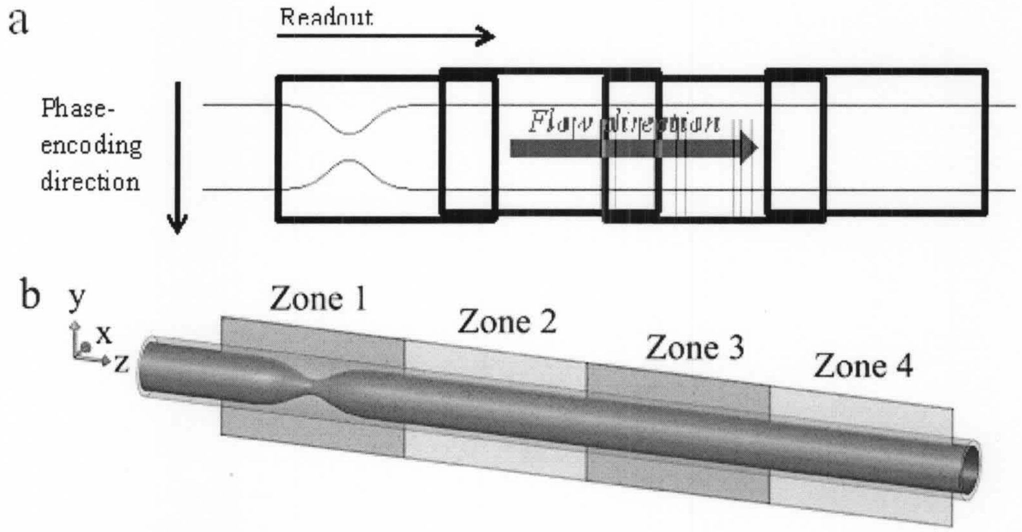
Echo time (TE) = 3.0 ms, Repetition time (TR) = 5.0 ms, Slice-thickness = 2 mm, Flip angle = 20°, Resolution = 1 × 1 mm. Number of signal averages (NSA) was 10 and 2 for steady and pulsatile flows, respectively. Venc values are summarized in Table 4-3.

Table 4-3

Venc values for different flow regimes in sagittal imaging.

Steady flow regime	Flow rate	Venc Read-out direction	Venc Phase-encoding direction	Venc Slice-selection direction
SF-1	56.3	120	20	20
SF-2	46.9	120	20	20
SF-2	37.5	120	20	20
Pulsatile flow regime	**Max flow rate**	**Venc Read-out direction**	**Venc Phase-encoding direction**	**Venc Slice-selection direction**
PF-1	184	200	35	-
PF-2	153	200	35	-
PF-3	122	200	35	-

The Venc was chosen to be roughly 10–20% above the expected peak velocity to avoid aliasing while providing maximum sensitivity. All PC velocity images were corrected for background phase errors resulted from concomitant gradient terms and eddy currents. To do this, first degree polynomials were fitted to the no flow images with the pump turned off, collected at the same locations with same parameters. Later, the resulting polynomial surface fits were subtracted from the corresponding velocity images.



Copy right: Iman Khodarahmi

Figure 4-5. Schematic diagram of the sagittal MR imaging.

4.3.2 Axial imaging

A standard 2D phase contrast sequence with Cartesian read-out, through-plane velocity encoding, and velocity compensation in all three directions was used to image three different sections perpendicular to the long axis of the phantom: 2 diameters proximal to the stenosis (inlet), at the stenosis (throat), and 2 diameters distal to the stenosis (outlet) (Figure 4-6). The throat of the stenosis was placed at the iso-center of the magnet, with the long axis of the phantom parallel to the B_0 magnetic field.

Imaging parameters were as follow: Repetition time (TR) = 4.0 ms, Echo time (TE) = 3.0 ms, Slice-thickness = 4 mm, Flip angle = 20°, Field of view (FOV) = 64 × 64 mm, Resolution = 1 × 1 mm. Number of signal averages (NSA) was 10 and 2 for steady and pulsatile flows, respectively. Since flow measurement was the purpose of axial imaging studies, velocity encoding was performed only in slice-selection direction. Venc values are given in Table 4-4. Venc was chosen to be roughly 10–20% above the expected peak velocity to avoid aliasing while providing maximum sensitivity.

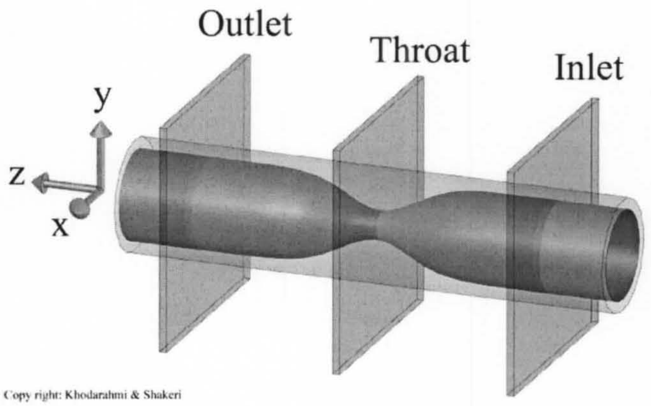
Similar to sagittal images, here, all axial PC velocity images were corrected for background phase errors with no-flow images collected (with same parameters) at the same locations.

To remove flow calculation errors due to variations in the area, regions of interest at every slice were defined using a circular mask at phantom contours on the basis of the a priori knowledge of the phantom geometry from CT images. The resolution of the velocity images was artificially increased by a factor of 10 before fitting the circular mask to minimize the partial volume effect, and then pixels with more than 50% of their area contained within the contour were included in the mask.

Table 4-4

Venc values for different flow regimes in axial imaging.

Steady flow regime	Flow rate	Venc Slice-selection direction (87%) Inlet/Throat/Outlet	Venc Slice-selection direction (74%) Inlet/Throat/Outlet	Venc Slice-selection direction (50%) Inlet/Throat/Outlet
SF-0	160	75/300/300	75/160/150	75/100/90
SF-1	56.3	30/120/120	30/60/60	30/30/30
SF-2	46.9	30/120/120	30/60/60	30/30/30
SF-2	37.5	30/120/120	30/60/60	30/30/30
Pulsatile flow regime	**Max flow rate**	**Venc Slice-selection direction (87%) Inlet/Throat/Outlet**	**Venc Slice-selection direction (74%) Inlet/Throat/Outlet**	**Venc Slice-selection direction (50%) Inlet/Throat/Outlet**
PF-1	184	50/200/200	50/100/80	50/75/60
PF-2	153	50/200/200	50/100/80	50/75/60
PF-3	122	50/200/200	50/100/80	50/75/60



Copy right: Khodarahmi & Shakeri

Figure 4-6. Schematic diagram of the axial MR imaging.

4.3.3 Uncertainty in PC-MRI measurements

MRI signal is measured through a quadrature receiver coil that reads the real and the imaginary signals. It is generally assumed that the noise in each signal has a Gaussian distribution with zero mean contaminated by a white noise. In an extensive study by Gudbjartsson and Patz, it has been shown that for SNR > 2, noise distribution is nearly Gaussian for both magnitude and phase images (Gudbjartsson and Patz, 1995). As shown in previous sections, the noise in the measured velocity depends on the noise in the phase measurements and on the velocity encoding. Therefore, the standard deviation in the measured velocity, σ_v, is:

$$\sigma_v = \left(\frac{\sqrt{2}}{\pi}\right)\frac{v_{enc}}{SNR} \qquad [4.3]$$

where signal to noise ratio of a magnitude image is represented with SNR. In our studies, SNR was calculated using an in-house MATLAB program to be 5.6. Hence, noise in the measured velocity, depending on the velocity encoding, can be determined as:

$$\sigma_v = 0.08\, v_{enc} \qquad [4.4]$$

4.4 Stereoscopic PIV Technique (joint work with Mostafa Shakeri, Ph.D.)

4.4.1 Sagittal configuration

A Stereoscopic PIV (SPIV) system (Powerview from TSI Inc., MN, USA) was utilized to measure three components of the velocity in the mid-plane of the axisymmetric stenosis model. The SPIV system (TSI Inc., MN, USA) consisted of a double-pulsed Nd-YAG laser with an energy of 120 mJ/pulse and a maximum repetition rate of 15 Hz (Khodarahmi, Shakeri et al., 2010; Shakeri, Khodarahmi et al., 2010). A number of optical components including three mirrors to redirect the laser beam to the imaging site, one concave cylindrical lens to convert the circular laser beam into a laser sheet, and two spherical lenses to control the thickness of the laser sheet were used (Figure 4-7).

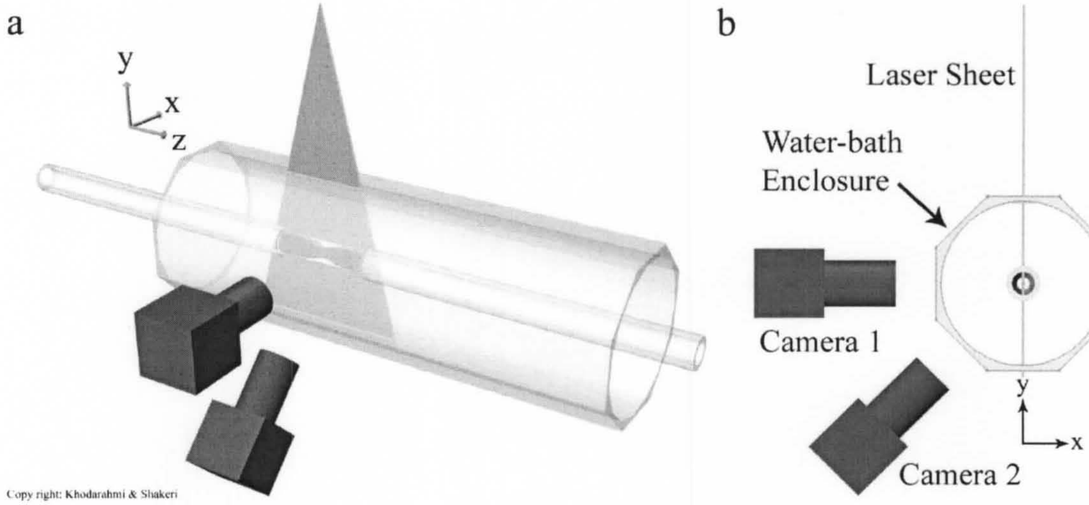


Figure 4-7. Stereoscopic PIV system (a) isometric view and (b) end view.

The images were recorded using two high-resolution CCD cameras (PowerView Plus 4MP) with 2048 × 2048 pixel resolution and 12 bits depth. The cameras allowed acquisition of 7.25 image pairs per second and frame straddling allowed a minimum of 200 ns between frames. Each camera was equipped with a 60-mm Mikro Nikkor lens with a minimum aperture of f-2.8. The two cameras looked at an angle of 0° and 45° relative to the normal to the light sheet and. This configuration allows for easy detection of the actual boundaries of the phantom from the normal looking camera. The phantom boundaries are needed for calculation of other hemodynamic parameters such as shear stress from the velocity data. More details of the experimental setup have been described in (Shakeri, Khodarahmi et al., 2010).

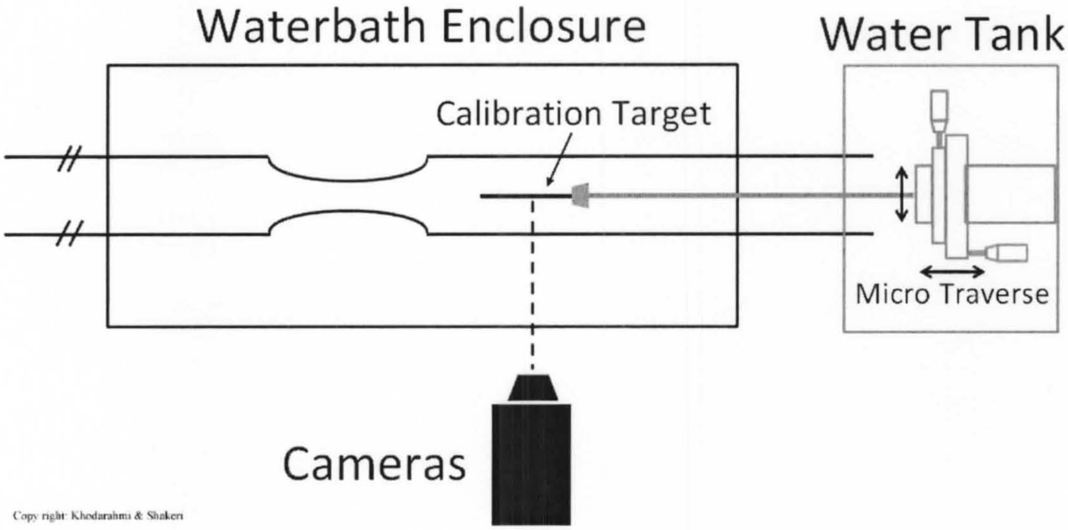
To reduce optical refraction mismatches between the fluid and acrylic, the stenosis model was contained in an enclosure filled with the same index of refraction-matched fluid used inside the flow loop. The water-bath enclosure was machined to provide several viewing windows with flat surfaces with sufficient accuracy and smoothness to eliminate optical distortion from this source. Neutrally buoyant fluorescent polymeric particles (Dantec Dynamics, Denmark) of diameter 1–20 μm (mean diameter 10 μm) were used in combination with an orange filter (Melles Griot Inc., CA, USA) to eliminate reflections from the model surface at the laser excitation wavelength (532 nm).

In these experiments, a 300-1000 μs time interval separated the two images for each acquisition (image pair) from which velocity was estimated. It should be noted that the appropriate time interval between image pairs is dependent upon the velocity of the flow. Recursive window size was used with the coarse and fine window sizes of 96 × 96 and 24 × 24, respectively. The pulse separation rates were chosen such that the maximum pixel displacement of the PIV particles corresponded to the dynamic range allowable for the coarse interrogation window. To improve accuracy, each spatial velocimetry assessment was generated with 40 pairs of images, therefore providing a more robust velocity estimation that can overcome potential artifacts like small equipment vibrations, transient bubble passages, or slight flow variations.

The entire study section was 460 mm long. Therefore, to record the images over this long distance, a light steering system was used and the cameras were mounted on a motorized traversing stage to allow for convenient collection of the images along the model. Flow images were collected over the entire flow domain from about 3 diameters upstream of the stenosis to about 14 diameters downstream.

Insight3G software (TSI Inc., MN, USA) was used to control the operation of the SPIV system. This software enables the manual or automatic control of the laser pulse separation time, independent control of the laser power output by adjusting the Q-switch delay time, as well as other selectable parameters such as image acquisition rate, camera delay time and the number of frames in a sequence. This software also allows control of the three-dimensional traversing system.

For the calibration purpose, an opaque glass calibration target was fabricated with a grid of 150 μm diameter dots spaced 400 μm apart. Before any set of experiments, the calibration target was placed in the plane of the laser light sheet and a picture of the target was taken with each camera (Figure 4-8).



Copy right: Khodarahmi & Shakeri

Figure 4-8. Schematic diagram of the calibration process.

The grid was clamped to one end of a stainless steel rod and the metal rod formed a solid support for the grid. To insert the calibration grid into the pipe, an open tank was placed behind the test section Figure 4-8. Five sets of images were recorded at different locations in the direction normal to the plane of the light sheet. For each set of new calibration images, the target was displaced 0.25 mm towards the camera and normal to the light sheet. The calibration grid was translated by a mirco-stage with an accuracy of 10 μm. These calibration images were then processed and used to transform points in the image plane into the physical plane (plane of the laser sheet) (Figure 4-9). Accordingly, the spatial resolution of the system was determined to be 14.0 μm/pixel.

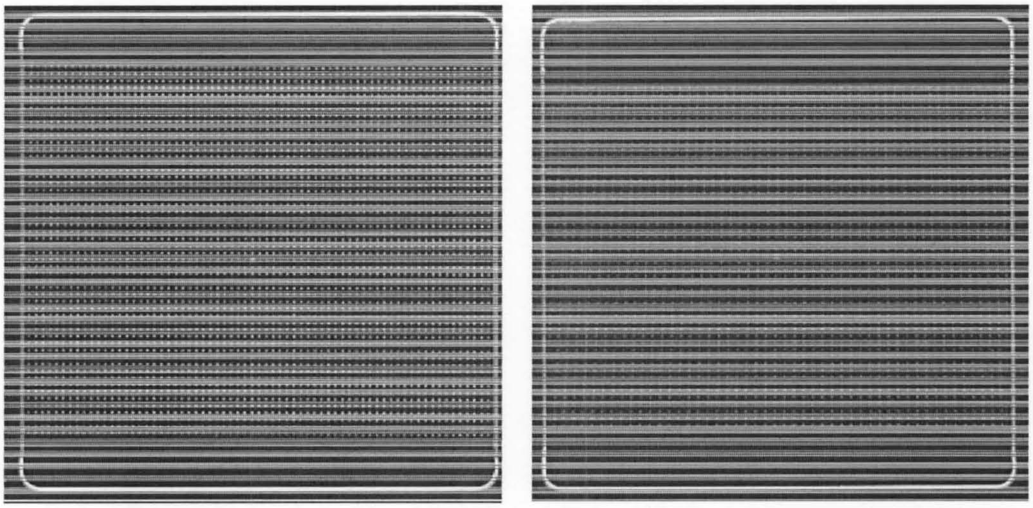


Figure 4-9. Images of the calibration target. Right and left images are from the bottom and top cameras, respectively.

Each camera used consecutive images to measure displacement of the seeding particles perpendicular to its viewing angle. These two different projections of the velocity, one from each camera, can be combined to reconstruct the three components of velocities. In practice, the two-component particle displacement fields observed by each camera were first evaluated by standard PIV cross-correlation methods. Subsequently, the particle displacement fields were mapped (dewarped) from the image planes onto the real-world plane of the light sheet and interpolated on a rectangular grid. Finally, the displacement vectors from both cameras were combined to reconstruct the three components of particle displacements based on the aforementioned calibration procedure. The final spatial resolution for velocity data in the mid-sagittal plane was calculated as:

$$SR_{velocity} = \frac{W}{2} \times Spatial\ resolution = \frac{24}{2} \times 0.014 \qquad [4.5]$$

$$= 0.168\ mm$$

where W is the interrogation window size in pixels and $SR_{velocity}$ is the spatial resolution for the velocity data.

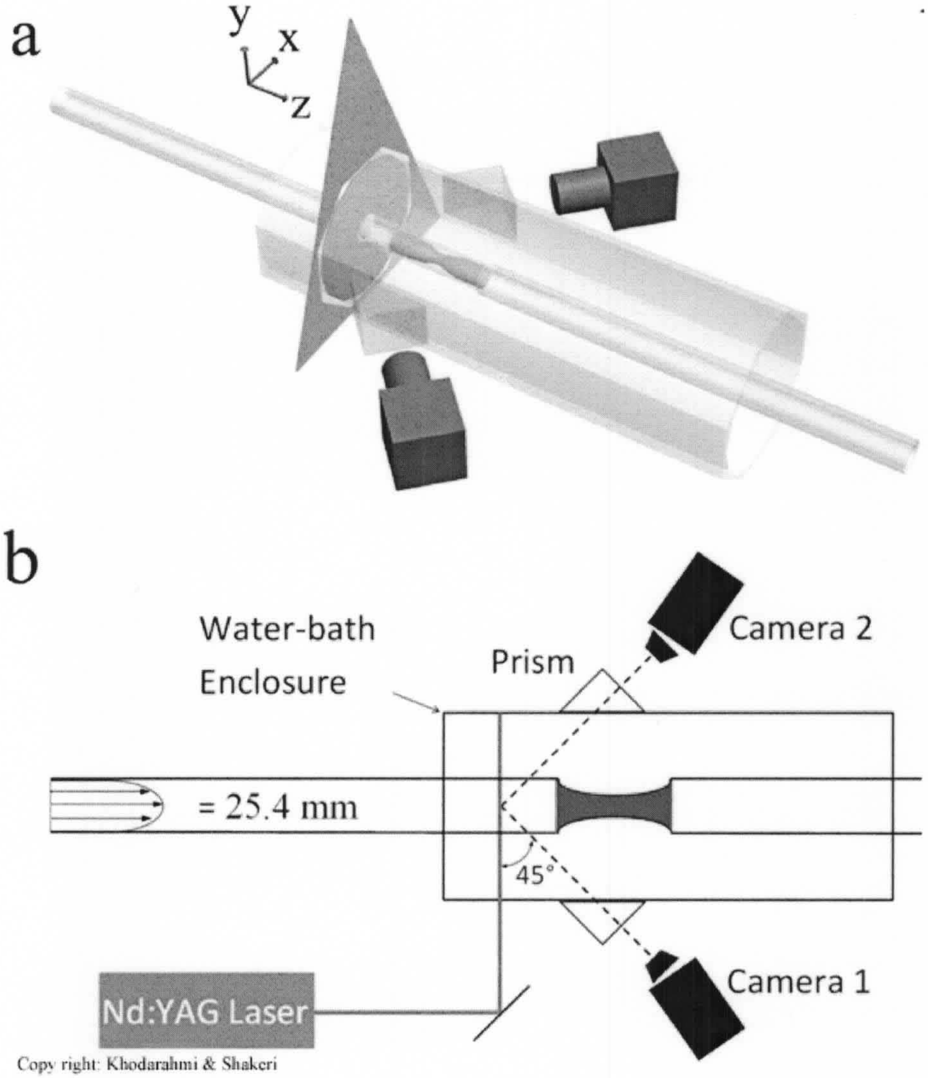
4.4.2 Axial configuration

The same PIV system with different configuration was applied for axial measurements (Figure 4-10). For this purpose cameras were placed on the two sides of the phantom, looking at the laser sheet at 45° angles, making a stereotypic view of the flow, to resolve the velocity vector for a cross section of the tube. This configuration is suitable for flow measurements through the system. To compensate for the optical refractions, two 45° prisms were built from transparent acrylic, filled with the same fluid of the water-bath enclosure and placed in front of each camera (Figure 4-10).

The calibration of cameras was repeated for this configuration. The field of view of each camera was approximately 20 mm × 30 mm, yielding a nominal spatial resolution of 205 μm for the velocity data points.

Here, the time difference between two consecutive image was 300-1200 μs, depending on the flow regime.

To improve accuracy, each spatial velocimetry assessment was generated with 30 pairs of images for steady flows and 12 pairs of images for pulsatile flows. The temporal resolution of pulsatile flow measurements was 25 milliseconds, corresponding to 40 measurements per second.



Copy right: Khodarahmi & Shakeri

Figure 4-10. Isometric view (a) and schematic top view (b) of the SPIV apparatus.

4.4.3 Uncertainty in SPIV measurements

Different sources of error contribute to the uncertainty in SPIV measurements: errors due to optics and cameras, errors of refractive index mismatch, correlation noise, systematic errors including bias and peak-locking, and misregistration errors (van Doorne and Westerweel, 2007). Signal-to-noise ratio in PIV experiments is generally 100-600 depending on the laser-pulse delay time and laser thickness. With proper parameter selection, as in the proposed experiments, correlation noise is estimated to be 0.1 pixels (Huang, Dabiri et al., 1997; van Doorne and Westerweel, 2007).

Although numerous papers have already discussed the uncertainty in PIV measurements, since each of these papers focus on one particular cause of uncertainty, a reliable quantification of the uncertainties in PIV is still lacking (Brossard, Monnier et al., 2009). Preliminary results of a study by Onera have provided orders of magnitude for the relative significance of sources of uncertainty in the case of 2C-PIV: determination of the displacement in pixels from raw images (90% of the total uncertainty), camera calibration (10% of the total uncertainty), and negligible error from the uncertainty on the time interval between the two laser pulses (Brossard, Monnier et al., 2009).

Registration errors, resulting from a small misalignment between the laser sheet and the calibration target, are the potential predominant errors in SPIV measurements. Using direct numerical simulations it has been found that the accuracy of SPIV measurements for both laminar and turbulent flows is better than 1% of the mean axial velocity which is sufficient to resolve the secondary flow patterns in transitional pipe flow (van Doorne and Westerweel, 2007).

4.5 Catheter Assisted Pressure Measurement (joint work with M. Shakeri, Ph.D.)

Two Mikro-tip pressure catheter transducers (SPR-320, Millar Instruments Inc. Houston, TX, USA) along with a pressure control unit (PCU-2000, Millar Instruments Inc. Houston, TX, USA) were used to measure the pressure proximal and distal to the stenosis. Catheters made of polyurethane, were 140 cm in length and 2 French in size. The catheters provide accurate high fidelity pressure monitoring with insignificant damping. Pressure data was acquired at a sampling rate of 500 Hz, digitized with a data acquisition system (model PCI-6052E, National Instruments Inc., Austin, TX, USA), and recorded on a computer using LabView (National Instruments Inc. , Austin, TX, USA) software. The pressure control unit was a two-channel, patient-isolated, line-powered amplifier/interface unit for connecting two catheters to a pressure monitor and data acquisition system (Khodarahmi, Shakeri et al., 2010).

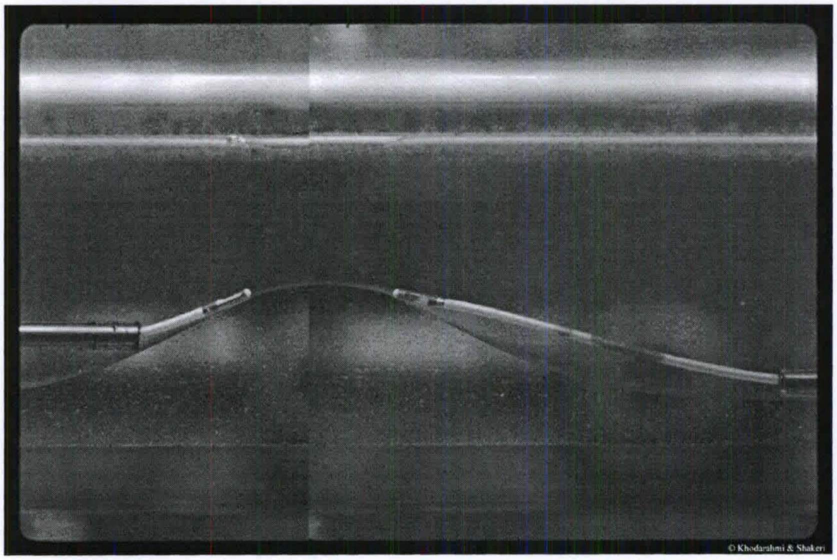

© Khodarahmi & Shakeri

Figure 4-11. Simultaneous pressure measurement.

One of the catheters was placed proximal to the stenosis and the other one distal to the stenosis. Keeping the proximal catheter in place, the distal one was pulled back to measure the pressure at different locations. The position of the tip of the catheter was recorded using a camera attached to a mobile traverse system. Later, images and hence position of the tip of the catheters, were registered using the coordinate system of the traverse. In order to ensure measuring the correct static pressure, to the extent possible, tip of each catheter was placed adjacent to the wall of the phantom.

Five hundred data points within each cardiac cycle were recorded and this procedure was repeated for 10 cardiac cycles to increase the reliability of the pressure measurements. This data was compared against relative pressures calculated from the iterative and non-iterative pressure calculation method.

4.5.1 Uncertainty in catheter-assisted pressure measurements

The sensitivity of the pressure catheter is 5 µV/V/mmHg (< 0.4 Pascal). The pressure sensor is sensitive to all components of the pressure including the dynamic pressure. The contribution of the dynamic pressure (expressed as $\frac{1}{2}\rho v^2$) could reach to ~ 650 Pa at the throat of the stenosis for steady flow and even higher for pulsatile cases. Therefore, to measure the correct static pressure, tip of the catheter was placed in a tiny cap to minimize the dynamic pressure contribution to the measured value. Also, tip of the catheters were always placed as close as possible to the wall of the phantom. Furthermore, number of pressure readings at each point was increased to > 7000 for steady and > 24 for pulsatile flow regimes.

5 PRESSURE CALCULATION

5.1 CFD Simulation (joint work with Mostafa Shakeri, Ph.D.)

The flows were numerically simulated for the same flow rates, geometry, and fluid properties as for the PC-MRI and SPIV experiments. The geometry of the simulated stenosis was set to match scans obtained on a 64-slice multi-detector CT scanner (LightSpeed VCT scanner, GE Medical Systems, Milwaukee, WI, USA) with slice thickness of 0.625 mm and pixel spacing of 0.22 mm × 0.22 mm.

The geometry was created in the preprocessor, GAMBIT 2.4.6 (ANSYS, Inc., Canonsburg, PA, USA). Explicit grid refinements were performed until negligible changes in the final results were observed. The final mesh used in this study had 1,200,000 pyramid/wedge shaped computational cells. To better resolve the flow near the wall and in the neighborhood of the stenosis throat, the computational domain was meshed with a successive ratio of 1.2 from the wall towards the centerline of the phantom, and with a bell-shaped distribution in the streamwise direction centered at the stenosis. The length of the computational domain was 460 mm (Figure 5-1).

Velocity distribution over the phantom volume was computed by solving the 3D Navier-Stokes equations for an incompressible Newtonian fluid, with rigid walls using the CFD software package Fluent 12.1 (ANSYS, Inc., Canonsburg, PA, USA) based on a finite volume scheme.

In order to match the experimental flow conditions, the axial SPIV velocity measurements acquired at the phantom inlet was applied as the inlet boundary condition. For the upstream boundary conditions, measured velocities at the model inlet were processed and smoothed before being mapped onto the computational grid inlet. Linear interpolation was performed in space and time between the measurement points. Pulsatile flow boundary conditions were prescribed at the inlet of the stenosis using the Womersley velocity profile (Cebral, Putman et al., 2009). For the downstream boundary conditions, outflow conditions were assumed. The computations were performed using a second-order upwind differencing scheme in space. Three cycles of computation would be sufficient to reach a cyclically repeated solution (Marshall, Zhao et al., 2004).

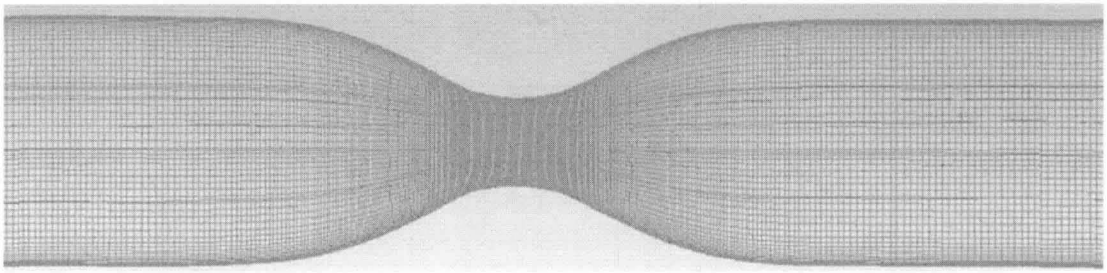


Figure 5-1. Mesh close up at the throat.

5.2 Iterative Method

Fluid motion is governed by the Navier-Stokes equations, which for an incompressible Newtonian fluid, takes the following form:

$$\nabla \mathrm{P} = -\rho \frac{\partial \mathbf{u}}{\partial t} - \rho(\mathbf{u} \cdot \nabla)\mathbf{u} + \mu \nabla^2 \mathbf{u} + \rho \mathbf{f} \qquad [5.1]$$

where $\mathrm{P}, \mathbf{u}, \rho, \mu$ and $\mathbf{f}$ denote the hydrostatic pressure, velocity vector field, density of the fluid, dynamic viscosity, and external body forces, respectively.

Gradient of pressure for steady flow in axisymmetric coordinates can be written as (Nasiraei-Moghaddam, Behrens et al., 2004; Khodarahmi, Shakeri et al., 2010):

$$\begin{aligned} \hat{P}_r = \frac{\partial \mathrm{P}}{\partial \mathrm{r}} = & -\rho\left(\mathrm{u_r}\frac{\partial \mathrm{u_r}}{\partial \mathrm{r}} + \mathrm{u_z}\frac{\partial \mathrm{u_r}}{\partial \mathrm{z}}\right) \\ & + \mu\left(\frac{1}{\mathrm{r}}\frac{\partial}{\partial \mathrm{r}}\left(\mathrm{r}\frac{\partial \mathrm{u_r}}{\partial \mathrm{r}}\right) + \frac{\partial^2 \mathrm{u_r}}{\partial \mathrm{z}^2} - \frac{\mathrm{u_r}}{\mathrm{r}^2}\right) \qquad [5.2] \\ \hat{P}_z = \frac{\partial \mathrm{P}}{\partial \mathrm{z}} = & -\rho\left(\mathrm{u_r}\frac{\partial \mathrm{u_z}}{\partial \mathrm{r}} + \mathrm{u_z}\frac{\partial \mathrm{u_z}}{\partial \mathrm{z}}\right) + \mu\left(\frac{1}{\mathrm{r}}\frac{\partial}{\partial \mathrm{r}}\left(\mathrm{r}\frac{\partial \mathrm{u_z}}{\partial \mathrm{r}}\right) + \frac{\partial^2 \mathrm{u_z}}{\partial \mathrm{z}^2}\right) \end{aligned}$$

where $\hat{P}_r$ and $\hat{P}_z$ denote the r and z components, respectively, of the gradient of pressure as calculated from velocity data. Due to the noisy nature of the velocity measurement techniques, the vector field ($\hat{P}_r$,$\hat{P}_z$) is not integrable (not curl-free), and as a result it cannot be integrated to produce the correct pressure. To find a pressure $\mathrm{P_c}$ such that $\nabla \mathrm{P_c}$ is the projection of ($\hat{P}_r$,$\hat{P}_z$) onto the curl-free subspace of integrable vector fields, a scalar function is desired such that the final relative error (FRE) is minimized:

$$\mathrm{FRE} = \frac{\left\|(\hat{P}_r,\hat{P}_z) - \nabla \mathrm{P_c}\right\|^2}{\left\|(\hat{P}_r,\hat{P}_z)\right\|^2}$$

$$= \frac{\iint_{\Omega_\mathrm{F}}\left[\left(\frac{\partial \mathrm{P_c}}{\partial \mathrm{r}} - \hat{P}_r\right)^2 + \left(\frac{\partial \mathrm{P_c}}{\partial \mathrm{z}} - \hat{P}_z\right)^2\right]\mathrm{r\,dr\,dz}}{\iint_{\Omega_\mathrm{F}}\left[\hat{P}_r^{\,2} + \hat{P}_z^{\,2}\right]\mathrm{r\,dr\,dz}} \qquad [5.3]$$

where Ω_F represents the flow domain. Since the denominator of equation [5.3] is constant for each measured velocity field, minimization the numerator of equation [5.3] would be sufficient. Using the calculus of variations, the well-known Pressure-Poisson equation for $\mathrm{P_c}$ can be obtained, which is a necessary condition for optimality (Song, Leahy et al., 1994; Nasiraei-Moghaddam, Behrens et al., 2004):

$$\nabla^2 \mathrm{P_c} = \nabla \cdot (\hat{P}_r,\hat{P}_z) \qquad [5.4]$$

At the boundaries, minimization of FRE produces the natural boundary condition:

$$\nabla \mathrm{P_c} \cdot \hat{\mathrm{n}} = (\hat{P}_r,\hat{P}_z) \cdot \hat{\mathrm{n}} \qquad [5.5]$$

where $\hat{\mathrm{n}}$ is the unit vector normal to the boundary of the flow domain. The above equations were solved over the stenotic region using the Gauss-Seidel iterative method. Natural boundary conditions were applied by first fitting a surface to the boundary and then finding the normal vector to that surface. The acceptable average error of the pressure for each point during each iteration was set to 5e-10 (convergence criteria).

5.3 Non-iterative Harmonics-based Orthogonal Projection

In the case of axisymmetric geometries, following Wang and Amini (Wang and Amini, 2005), for the vector field $\hat{P}_r$ and $\hat{P}_z$ to be integrable, their integral needs to be the path independent or they should be the gradient of a scalar function. So,

$$\frac{\partial \hat{P}_r}{\partial z} = \frac{\partial \hat{P}_z}{\partial r} \qquad [5.6]$$

The projection of $\widehat{\nabla P}(r,z) = (\hat{P}_r, \hat{P}_z)$ onto an integrable subspace translates into minimizing the following integral:

$$D = \iint_{\Omega} |\tilde{P}_r - \hat{P}_r|^2 + |\tilde{P}_z - \hat{P}_z|^2 \; drdz \qquad [5.7]$$

where $(\tilde{P}_r,\ \tilde{P}_z)$ is the projection onto the desired integrable subspace, which minimizes D in equation [5.7].

Using some orthogonal integrable basis function $\varphi(r,z,\underline{\omega})$, with $\underline{\omega}$ as the vector (ω_r, ω_z) of spatial frequencies, the pressure $\tilde{\boldsymbol{P}}$ could be expanded as:

$$\tilde{\boldsymbol{P}} = \sum \tilde{C}(\underline{\omega})\, \varphi(r,z,\underline{\omega}) \qquad [5.8]$$

Hence, its gradient are given as:

$$\begin{aligned} \tilde{P}_r &= \sum \tilde{C}(\underline{\omega})\, \varphi_r(r,z,\underline{\omega}) \\ \tilde{P}_z &= \sum \tilde{C}(\underline{\omega})\, \varphi_z(r,z,\underline{\omega}) \end{aligned} \qquad [5.9]$$

where $\varphi_r = \frac{\partial\varphi}{\partial r}$ and $\varphi_z = \frac{\partial\varphi}{\partial z}$.

The measured gradient can also be expanded using same basis functions:

$$\hat{P}_r = \sum \hat{C}_1(\underline{\omega})\, \varphi_r(r, z, \underline{\omega})$$
$$\hat{P}_z = \sum \hat{C}_2(\underline{\omega})\, \varphi_z(r, z, \underline{\omega}) \qquad [5.10]$$

Applying the demonstration of Frankot and Chellappa (Frankot and Chellappa, 1988), the coefficient of expansion of the projected pressure $\widetilde{\boldsymbol{P}}$ in the integrable subspace, is related to $\hat{C}_1$ and $\hat{C}_2$ by:

$$\tilde{C} = \frac{\hat{C}_1 T_r + \hat{C}_2 T_z}{T_r + T_z} \qquad [5.11]$$

where $T_r = \int_{R^2} |\varphi_r|^2 drdz$ and $T_z = \int_{R^2} |\varphi_z|^2 drdz$.

Therefore, by substituting $\tilde{C}(\underline{\omega})$ from equation [5.11] into Eqs. [5.8] and [5.9], integrable pressure gradients will correctly be calculated. Integration of these path-independent gradients will produce the true pressure. Note that the pressure field is calculated in a single calculation with no iterations, using all of the available information contained in $\hat{P}_r$ and $\hat{P}_z$.

Fourier basis functions can be chosen for $\varphi(\underline{\omega})$ for convenience of computations using the fast implementation of the Discrete Fourier Transform (FFT). The Fourier basis functions are expressed as:

$$\varphi(r, z, \underline{\omega}) = e^{j\omega_r r + j\omega_z z} \qquad [5.12]$$

Using the Fourier basis functions equation [5.11] changes to:

$$\tilde{C}(\underline{\omega}) = \frac{\omega_r^2 \hat{C}_1(\underline{\omega}) + \omega_z^2 \hat{C}_2(\underline{\omega})}{\omega_r^2 + \omega_z^2} \qquad [5.13]$$

As explained thoroughly in (Wang and Amini, 2005), two kinds of discontinuities exist that affect the non-iterative technique. First, because of the non-rectangular geometry of the vessel, in addition to the inside area, area outside of the vessel will be included in the projection. This discontinuity can be avoided by defining the gradients as:

$$\begin{aligned} \hat{P}_r(r,z) &= \begin{cases} \hat{P}_r(r_b,z), & r_b(z) < r \leq r_L \\ \hat{P}_r(r,z), & r \leq r_b(z) \end{cases} \\ \hat{P}_z(r,z) &= \begin{cases} \hat{P}_z(r_b,z), & r_b(z) < r \leq r_L \\ \hat{P}_z(r,z), & r \leq r_b(z) \end{cases} \end{aligned} \qquad [5.14]$$

where r_L is the boundary of the rectangular region. The second discontinuity occurs at locations r = 0, r = r_L, z = 0, and z = z_L since the Discrete Fourier Transform applies a periodic assumption at the boundaries. The latter discontinuity is avoided by a symmetric expansion of the pressure (Wang and Amini, 2005):

$$\hat{p}(r,z) = \hat{p}(r,-z) = \hat{p}(-r,z) = \hat{p}(-r,-z) \qquad [5.15]$$

6 RESULTS

6.1 Velocity Comparison

Figure 6-1, Figure 6-2, and Figure 6-3 show the axial velocity contour from PC-MRI, SPIV and CFD for steady flow cases SF-1, SF-2, and SF-3, respectively. Figure 6-4 through Figure 6-9 show the axial velocity contours from PC-MRI and CFD for pulsatile flows PF-1, PF-2, and PF-3 at peak-systole (t = 400 ms on flow waveform curve in Figure 6-10), and end-diastole (t = 275 ms on flow waveform curve in Figure 6-10).

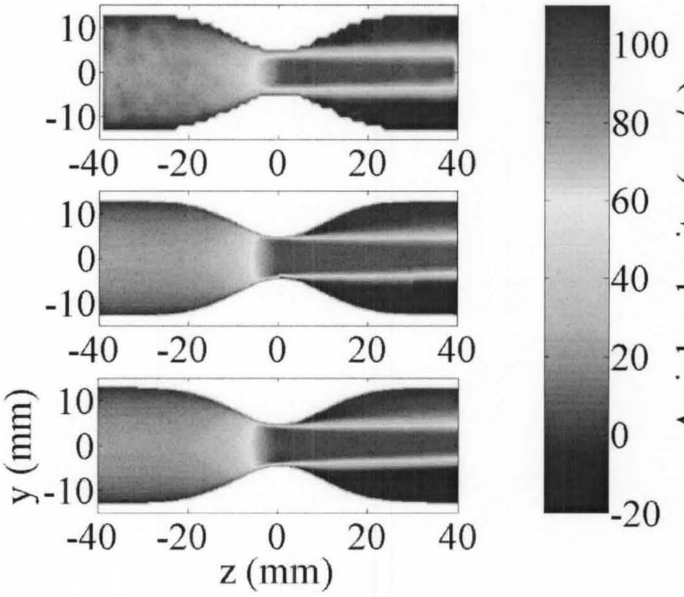


Figure 6-1. Axial velocity contours for steady flow SF-1 from PC-MRI (top), SPIV (middle), and CFD (bottom).

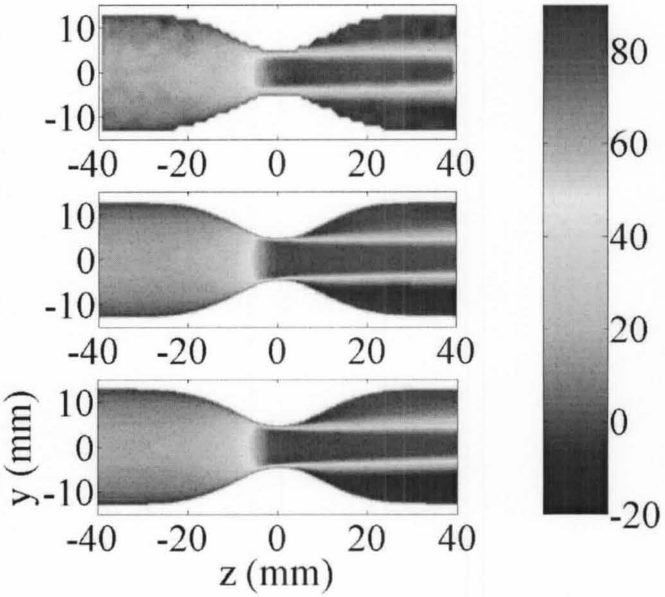


Figure 6-2. Axial velocity contours for steady flow SF-2 from PC-MRI (top), SPIV (middle), and CFD (bottom).

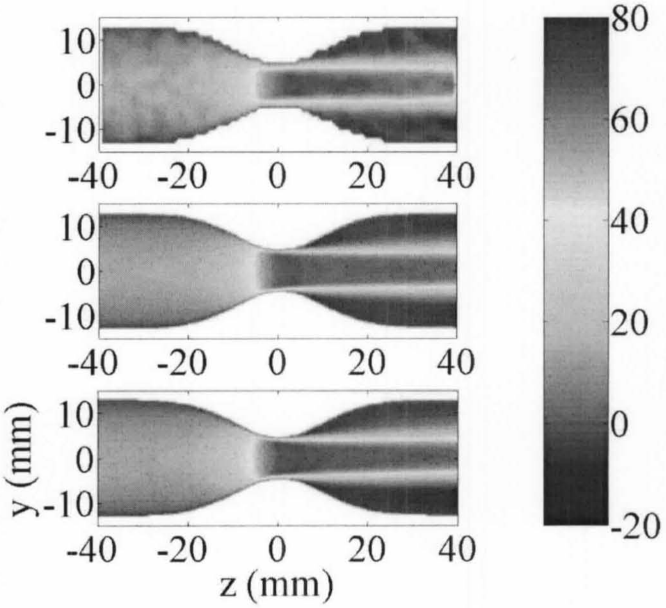


Figure 6-3. Axial velocity contours for steady flow SF-3 from PC-MRI (top), SPIV (middle), and CFD (bottom).

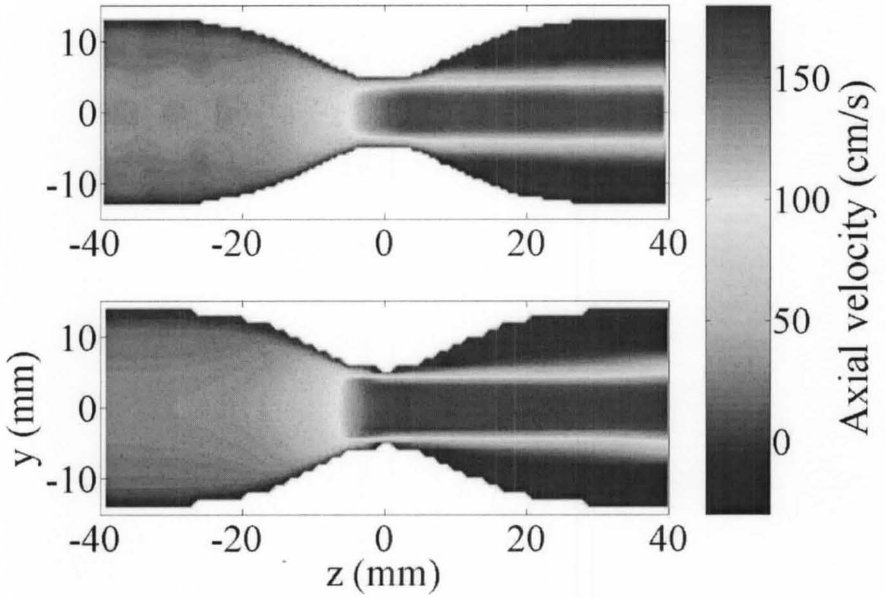


Figure 6-4. Axial velocity contours for pulsatile flow PF-1 at peak systole (t = 400 ms on flow waveform curve in Figure 6-10) from PC-MRI (top) and CFD (bottom).

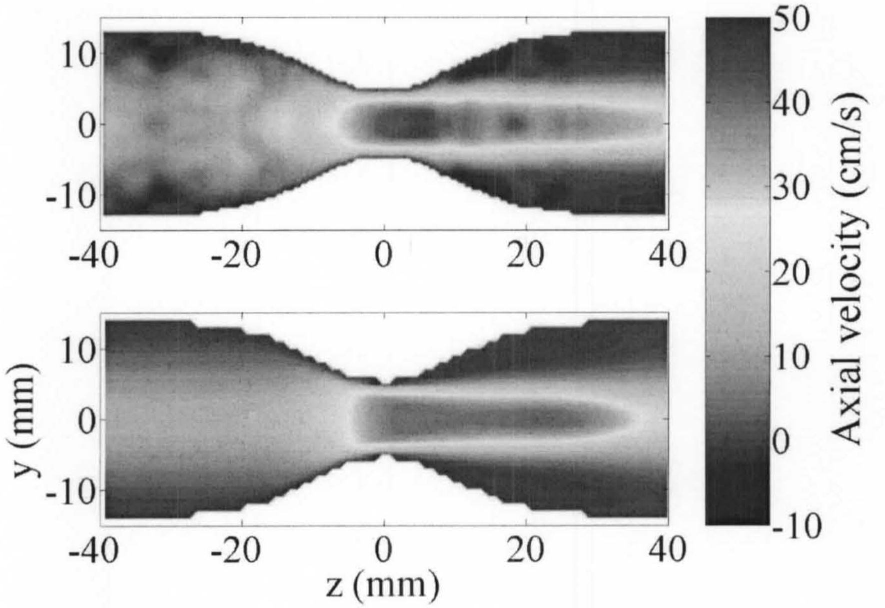


Figure 6-5. Axial velocity contours for pulsatile flow PF-1 at end diastole (t = 275 ms on flow waveform curve in Figure 6-10) from PC-MRI (top) and CFD (bottom).

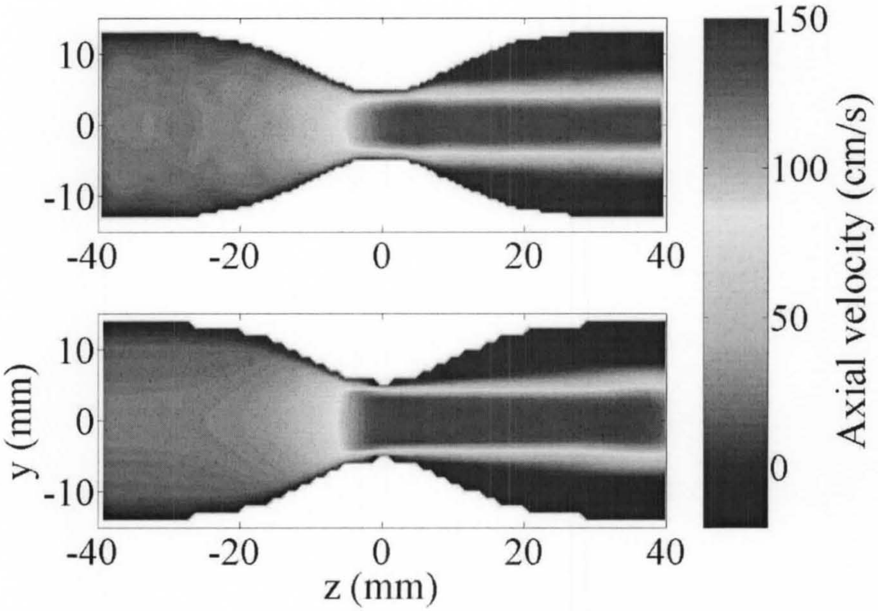


Figure 6-6. Axial velocity contours for pulsatile flow PF-2 at peak systole (t = 400 ms on flow waveform curve in Figure 6-10) from PC-MRI (top) and CFD (bottom).

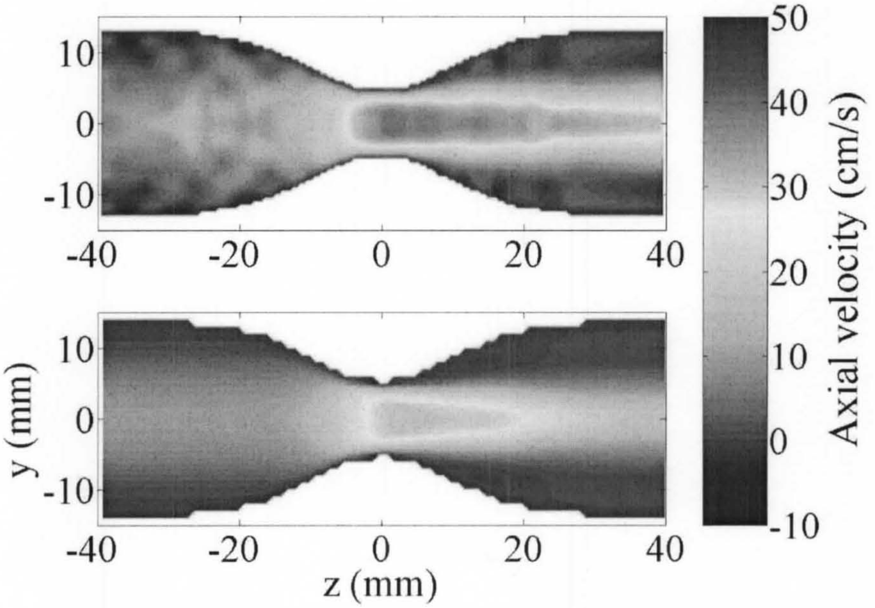


Figure 6-7. Axial velocity contours for pulsatile flow PF-2 at end diastole (t = 275 ms on flow waveform curve in Figure 6-10) from PC-MRI (top) and CFD (bottom).

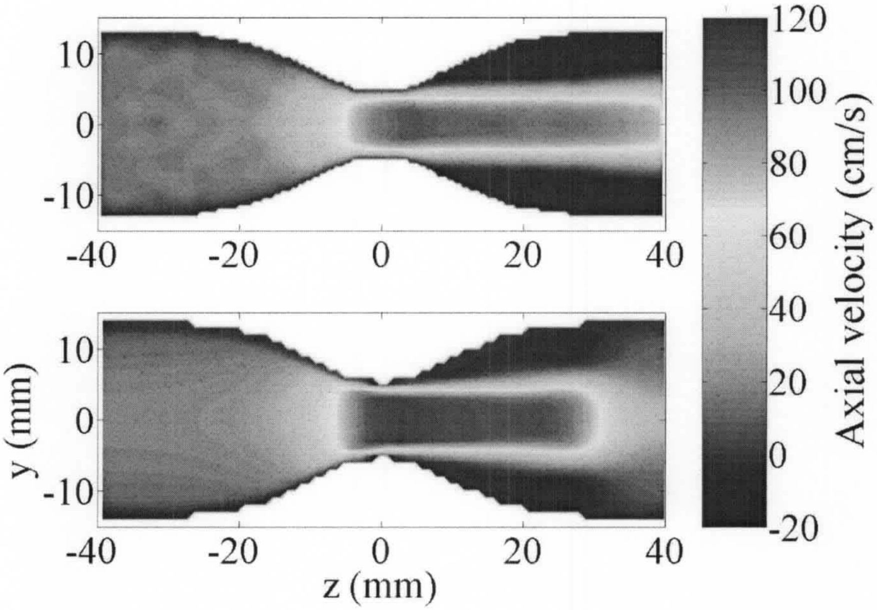


Figure 6-8. Axial velocity contours for pulsatile flow PF-3 at peak systole (t = 400 ms on flow waveform curve in Figure 6-10) from PC-MRI (top) and CFD (bottom).

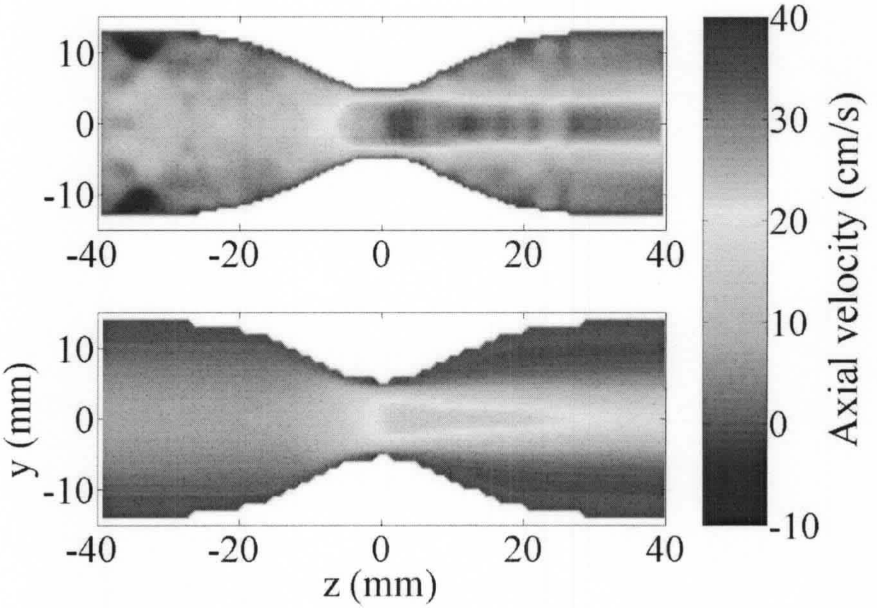


Figure 6-9. Axial velocity contours for pulsatile flow PF-3 at end diastole (t = 275 ms on flow waveform curve in Figure 6-10) from PC-MRI (top) and CFD (bottom).

6.2 Flow Measurement

6.2.1 SPIV Reproducibility

SPIV provided a robust measurement of flow for both steady and pulsatile cases. In repeat experiments the coefficient of variation, defined as the standard deviation divided by the mean, was less than 0.5% for steady flows and less than 1.5% for every time-point measurement in pulsatile flows determined over multiple cardiac cycles. In pulsatile flows, the programmed and measured flow waveforms differed significantly due to the damping caused by the elastic tube connections. This leads to a delay and smoothing of the input waveform at the inlet of the phantom (Figure 6-10). However, the total actual flow per cycle per minute remained unchanged.

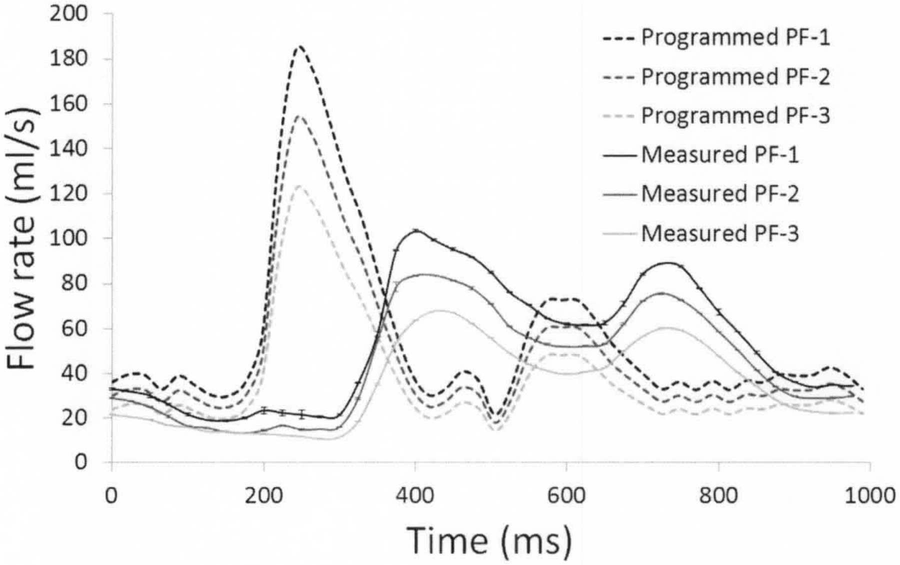


Figure 6-10. Programmed and measured flow waveforms. Dashed curves are applied by the computer controlled pump and solid curves are the measured flow using SPIV, each curve with 40 point measurements. Error bars on the measured flow waveforms show the standard deviation of the SPIV measurements.

6.2.2 Correlation of SPIV and PC-MRI Measurements at Inlet

Comparison of the flow measured by PC-MRI and SPIV at the inlet of three phantoms, where flow was fully developed, gives an estimate of the validity of PC-MRI for laminar flows. Results show good agreement between the two methods. The regression line approaches the line of identity with $f_{MRI} = 1.01\ f_{SPIV} - 2.26$ ($R > 0.99$) for steady flows (Figure 6-11a). In pulsatile flows (Figure 6-11b), this correlation can be expressed as: $f_{MRI} = 0.87\ f_{SPIV} + 2.37$ ($R > 0.96$) (with 40 samples per waveform).

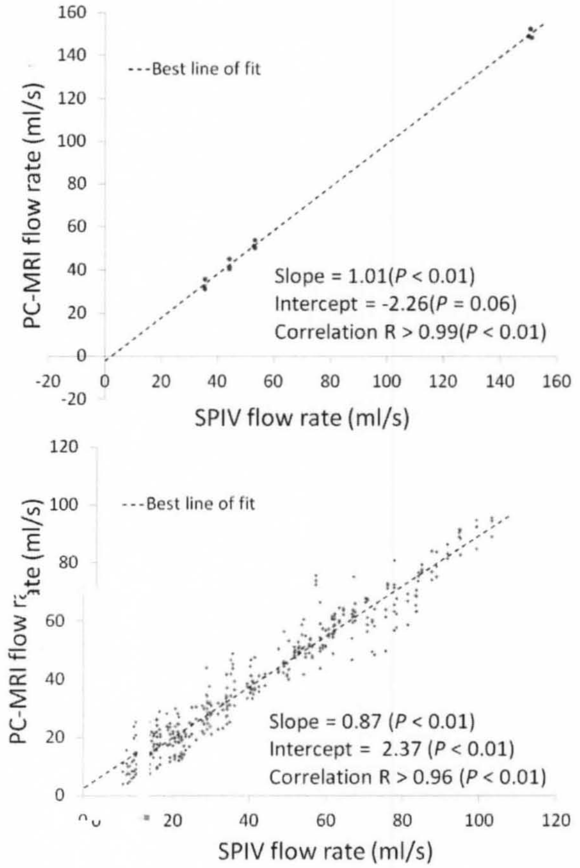


Figure 6-11. Scatter diagrams comparing PC-MRI with SPIV flow measurements. Data for all flow regimes at the inlet of the three phantoms are combined in this plot. a: Steady flow (N = 12). b: Pulsatile flow (N = 360).

6.2.3 Steady PC-MRI Measurement at Three Sections

Errors in mean flow for steady flow measurements using PC-MRI are reported in Table 6-1. All comparisons are based on SPIV as ground truth. For SF-0 and SF-1 flows (see Table 4-1 for definition), which benefit from suitable Venc selection, the error at the inlet is less than 5%. Higher errors in SF-2 and SF-3 show the importance of choosing a proper Venc (Gudbjartsson and Patz, 1995; Nasiraei-Moghaddam, Behrens et al., 2004) .

Flow rate errors show that for the 87% and 74% stenoses, PC-MRI underestimated flow at the outlet. For the same phantoms, flow measurements at both the inlet and throat show very good agreement between the two methods. Through-plane velocity contours and profiles at the inlet for the 87% area occlusion phantom for flow SF-1 are shown in Figure 6-12a. Both contour plots and velocity profiles show good agreement between the two methods.

Table 6-1

Errors of the steady flow as measured with PC-MRI compared to SPIV.

		Steady flow rate error (%)			
		SF-0	**SF-1**	**SF-2**	**SF-3**
87% stenosis	**Inlet**	-1.6	-4.7	-7.6	-11.6
	Throat	-0.8	1.0	2.0	0.7
	Outlet	-4.6	-16.5	-22.2	-30.6
74% stenosis	**Inlet**	-0.4	-2.8	-6.0	-7.8
	Throat	4.9	4.5	3.7	3.3
	Outlet	-3.4	-7.7	-10.0	-13.1
50% stenosis	**Inlet**	1.4	1.8	2.4	1.2
	Throat	3.8	3.2	3.3	2.1
	Outlet	1.4	1.5	1.5	-1.3

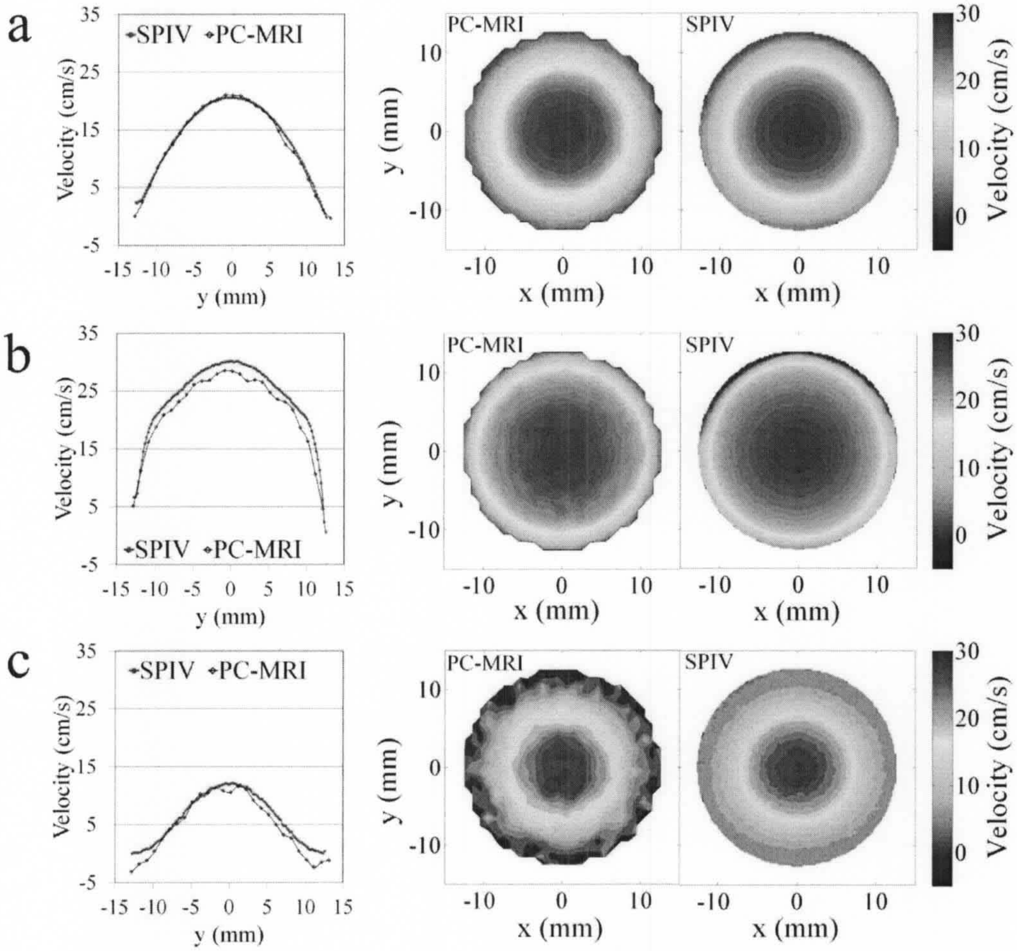


Figure 6-12. Through-plane velocity contours and profiles for steady and pulsatile flows at the inlet in the 87% area occlusion phantom: a: SF-1, b: PF-1 (t = 400 ms; maximum flow), c: PF-1 (t = 275 ms; minimum flow). Profiles are depicted for the cross line x = 0.

6.2.4 Pulsatile PC-MRI Measurement at Three Sections

Flow rate error relative to SPIV measurements was calculated for each of the three phantom locations. Normalized root mean square error (NRMSE) was calculated according to

$$NRMSE = \sqrt{\frac{\sum_t \left(f_{MRI}(t) - f_{SPIV}(t)\right)^2}{\sum_t \left(f_{SPIV}(t)\right)^2}}$$

where f is the flow measurement by either method and the summation is performed over different time points (40 measurements for pulsatile flows).

Flow waveforms measured by SPIV and PC-MRI at three sections for 50%, 74% and 87% phantoms and flows PF-1, PF-2 and PF-3 are shown in Figure 6-13. The corresponding mean and peak flow errors as well as NRMSE are summarized in Table 6-2. While the mean flow measurement for 50% stenosis phantom was relatively accurate at all three sections, for the other two phantoms the best accuracy was obtained at the throat. Also, PC-MRI generally detected mean flow better than peak flow as evidenced by the smaller errors.

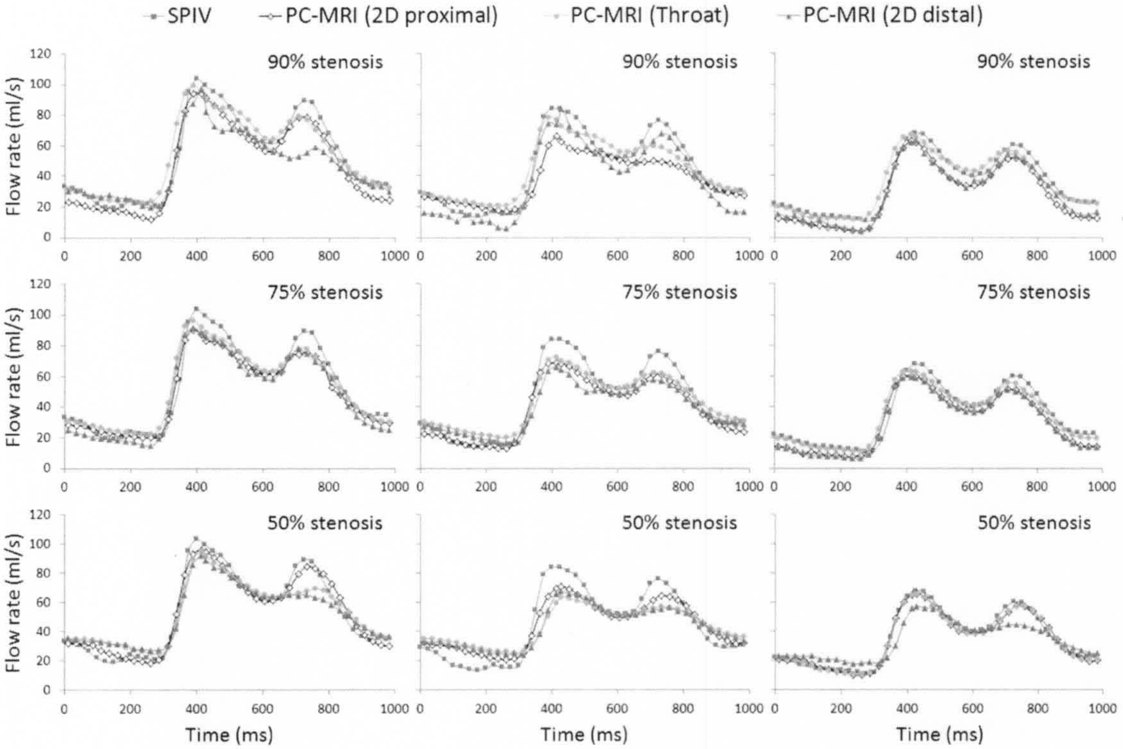


Figure 6-13. Flow waveforms measured by SPIV and PC-MRI at inlet, throat and outlet. Top, middle, and bottom rows show results for 87%, 74% and 50% stenosis phantoms, respectively. Left, middle, and right columns show pulsatile flows PF-1, PF-2 and PF-3, respectively.

Table 6-2

Errors of the mean and peak flow measurements with PC-MRI for all pulsatile flows. Flow rates are compared against SPIV as the gold standard. Normalized root mean squared errors (NRMSE) are also reported.

Stenosis		Mean flow rate error (%)			Peak flow rate error (%)			Normalized root mean square error (%)		
		PF-1	PF-2	PF-3	PF-1	PF-2	PF-3	PF-1	PF-2	PF-3
87%	Inlet	-10.4	-14.1	-19.2	-9.1	-21.7	-7.0	11.9	23.5	17.3
	Throat	2.3	1.16	1.5	-4.1	-6.7	-0.6	11.7	17.2	8.3
	Outlet	-10.8	-15.3	-15.3	-6.5	-0.2	-7.3	21.9	15.8	14.8
74%	Inlet	-6.0	-11.4	-13.4	-12.7	-18.0	-11.3	11.5	16.5	14.3
	Throat	0.42	-0.8	-1.0	-6.9	-13.8	-6.8	11.4	14.5	8.8
	Outlet	-8.8	-11.4	-15.5	-11.7	-21.0	-12.4	11.4	20.2	15.8
50%	Inlet	0.3	-0.6	-0.9	-7.5	-15.7	-2.8	9.3	16.1	8.8
	Throat	0.9	-1.6	-1.4	-8.3	-24.3	-2.9	13.8	25.4	8.0
	Outlet	-1.6	-2.2	-2.9	-11.0	-19.5	-15.8	15.7	22.7	17.8

Through-plane velocity contours and profiles of pulsatile flow PF-1 for two time points t = 400 ms (corresponding to the maximum flow rate) and t = 275 ms (corresponding to the minimum flow rate) at the inlet are shown in Figure 6-12 (b and c) for the 87% area occlusion phantom. Both contour plots and velocity profiles show good agreement between the methods. However, the near-wall velocity at t = 275 ms (Figure 6-12c) was noisy, because Venc is set for the maximum flow rate.

As evidenced by the contour plots and profiles in Figure 6-12, near-wall velocity at cardiac phases with low flow rates (e.g., t = 275 ms), is noisy, leading to underestimation of the flow for those cardiac phases. This occurred because in pulsatile studies, Venc is generally set corresponding to the peak flow rate and is constant over cardiac phases. MR sequences with the capability to automatically optimize Venc for individual heart phases in pulsatile flow could potentially prevent this underestimation (Ringgaard, Oyre et al., 2004).

Scatter diagrams comparing pulsatile PC-MRI with SPIV flow measurements at three different sections for three different phantoms are shown in Figure 6-14. Data for all flows PF-1, PF-2 and PF-3 are combined for each section/phantom to evaluate the accuracy of flow measurement as a function of imaging section. Correlation coefficients, slopes, and intercepts are summarized in Table 6-3, showing good agreement between MRI and gold standard flow from SPIV. Normalized root mean square error (NRMSE) for all phantoms, flows and sections were higher compared to errors in mean flow measurement of the corresponding values in all cases. This may suggest decreased ability of PC-MRI to detect the high frequency components of the flow waveform, because of the averaging that occurs during data collection for different k-space lines in PC-MRI.

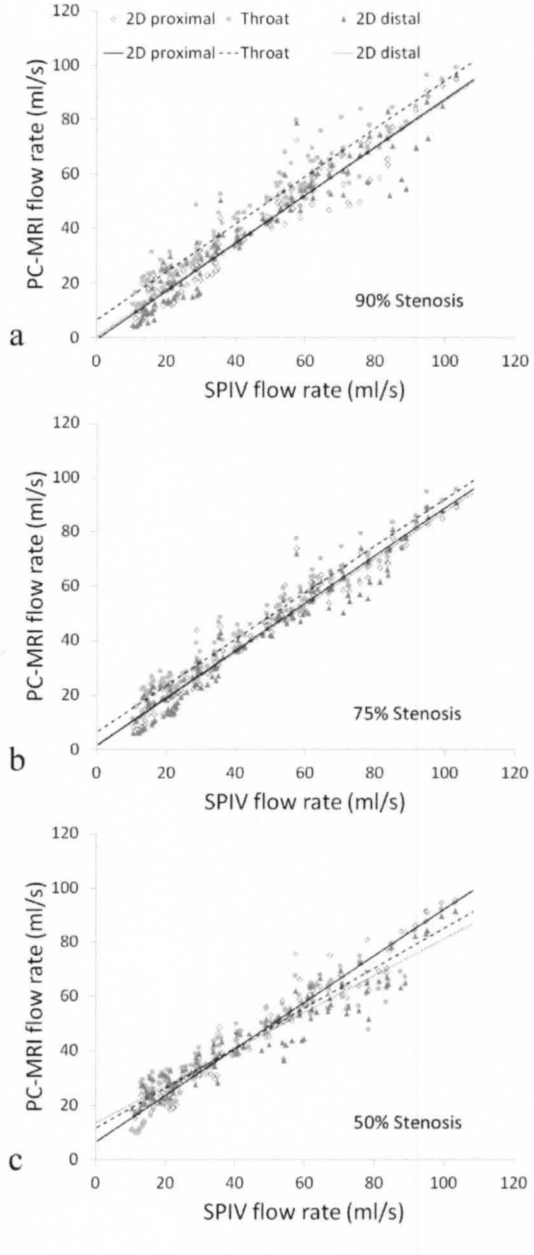


Figure 6-14. Scatter diagrams comparing pulsatile PC-MRI with SPIV flow. Data for all flows PF-1, PF-2 and PF-3 are combined for each section/phantom (N = 120 for each section in each phantom). Solid, dashed and dotted lines represent the best line of fit for inlet, throat and outlet sections, respectively. Corresponding slopes, intercepts and correlation coefficients are reported in Table 6-3.

Table 6-3

Linear regression of the flow measured by PC-MRI against SPIV.

Stenosis		Linear correlation for all flows and (p-values)		
		Slope	**Intercept**	**R**
87%	**Inlet**	0.88 (< 0.01)	-0.71 (0.52)	> 0.96 (< 0.01)
	Throat	0.87 (< 0.01)	6.39 (< 0.01)	> 0.96 (< 0.01)
	Outlet	0.86 (< 0.01)	0.32 (0.80)	> 0.95 (< 0.01)
74%	**Inlet**	0.87 (< 0.01)	1.52 (0.09)	> 0.97 (< 0.01)
	Throat	0.86 (< 0.01)	6.41 (< 0.01)	> 0.97 (< 0.01)
	Outlet	0.86 (< 0.01)	1.36 (0.17)	> 0.97 (< 0.01)
50%	**Inlet**	0.86 (< 0.01)	6.31 (< 0.01)	> 0.97 (< 0.01)
	Throat	0.74 (< 0.01)	11.50 (< 0.01)	> 0.94 (< 0.01)
	Outlet	0.68 (< 0.01)	13.50 (< 0.01)	> 0.95 (< 0.01)

6.3 Pressure Measurement

6.3.1 Comparison of the iterative and non-iterative techniques

Iterative and non-iterative techniques are compared with each other by applying both methods to the noise-free velocity data obtained from CFD simulations on ideal stenoses for steady flow SF-2. In order to do this, velocity data were first calculated by CFD simulations, and were then regridded to rectangular meshes with different resolutions. Mesh sizes of 1.88, 1.57, 1.25, 0.94, 0.63, 0.31, 0.15, and 0.10 mm were tested. These values correspond to 7.4%, 6.2%, 4.9%, 3.7%, 2.5%, 1.2%, 0.6%, and 0.4% of the inlet diameter, respectively. Pressure gradients were calculated using second order central differencing scheme in the fluid domain and forward/backward differencing scheme at the boundaries.

After regridding, both iterative and non-iterative methods were applied to the regridded velocity data (Figure 6-15 through Figure 6-22). Results of the two methods were compared in terms of the CPU usage time and normalized root mean square error (NRMSE) over the entire volume. The normalized root mean square error (NRMSE) was calculated according to

$$NRMSE = \sqrt{\frac{\iint \left(P_c - P_f\right)^2 r\, dr\, dz}{\iint \left(P_f\right)^2 r\, dr\, dz}}$$

where P_f is the pressure calculated by Fluent software, and P_c is the pressure calculated by either iterative or non-iterative methods. Results are represented in Table 6-4.

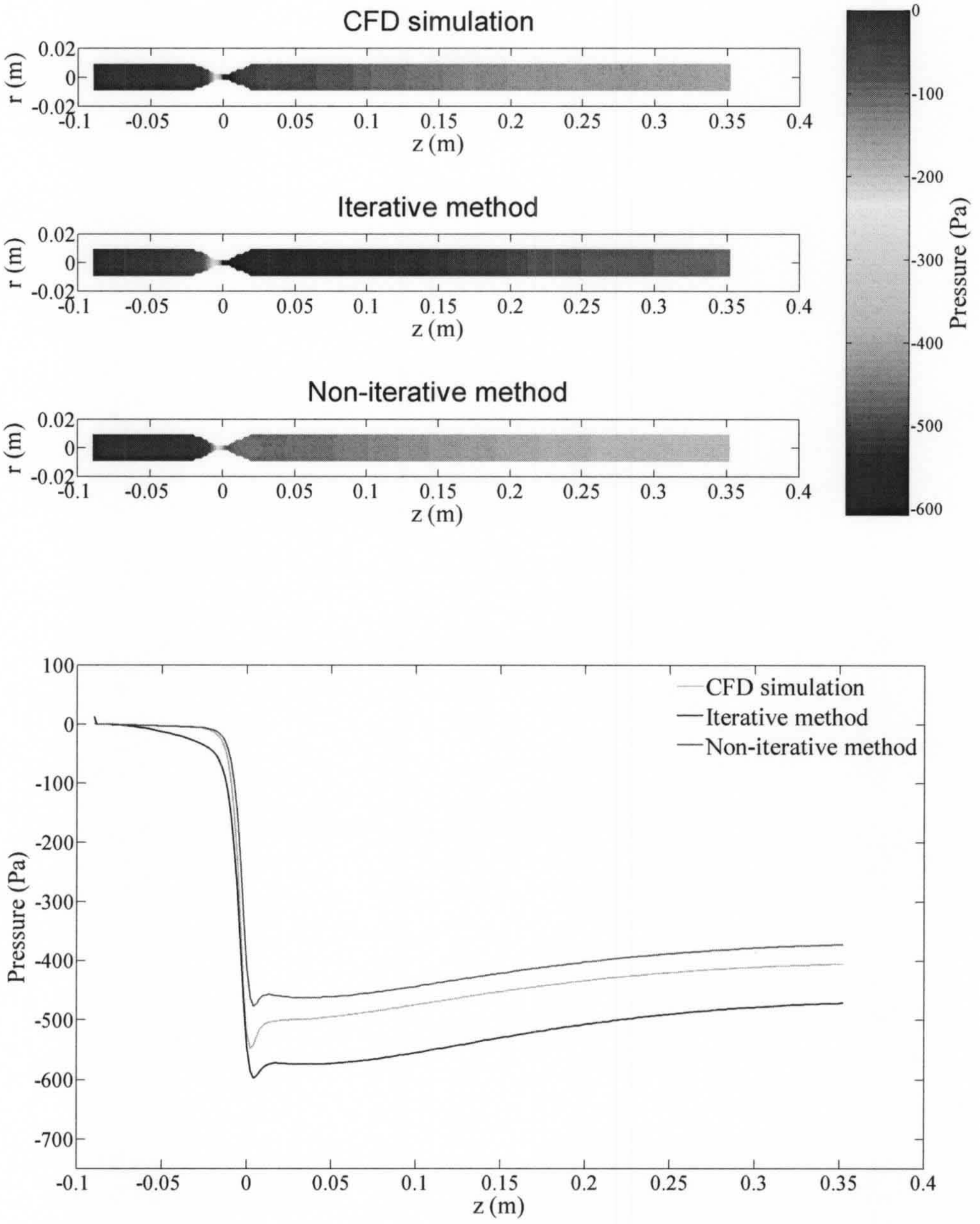


Figure 6-15. Pressure contour plot (top) and pressure profile along the centerline (bottom) obtained directly from CFD simulations, and calculated using iterative method and non-iterative methods on the velocity data regridded to a rectangular mesh at resolution 1.88 mm.

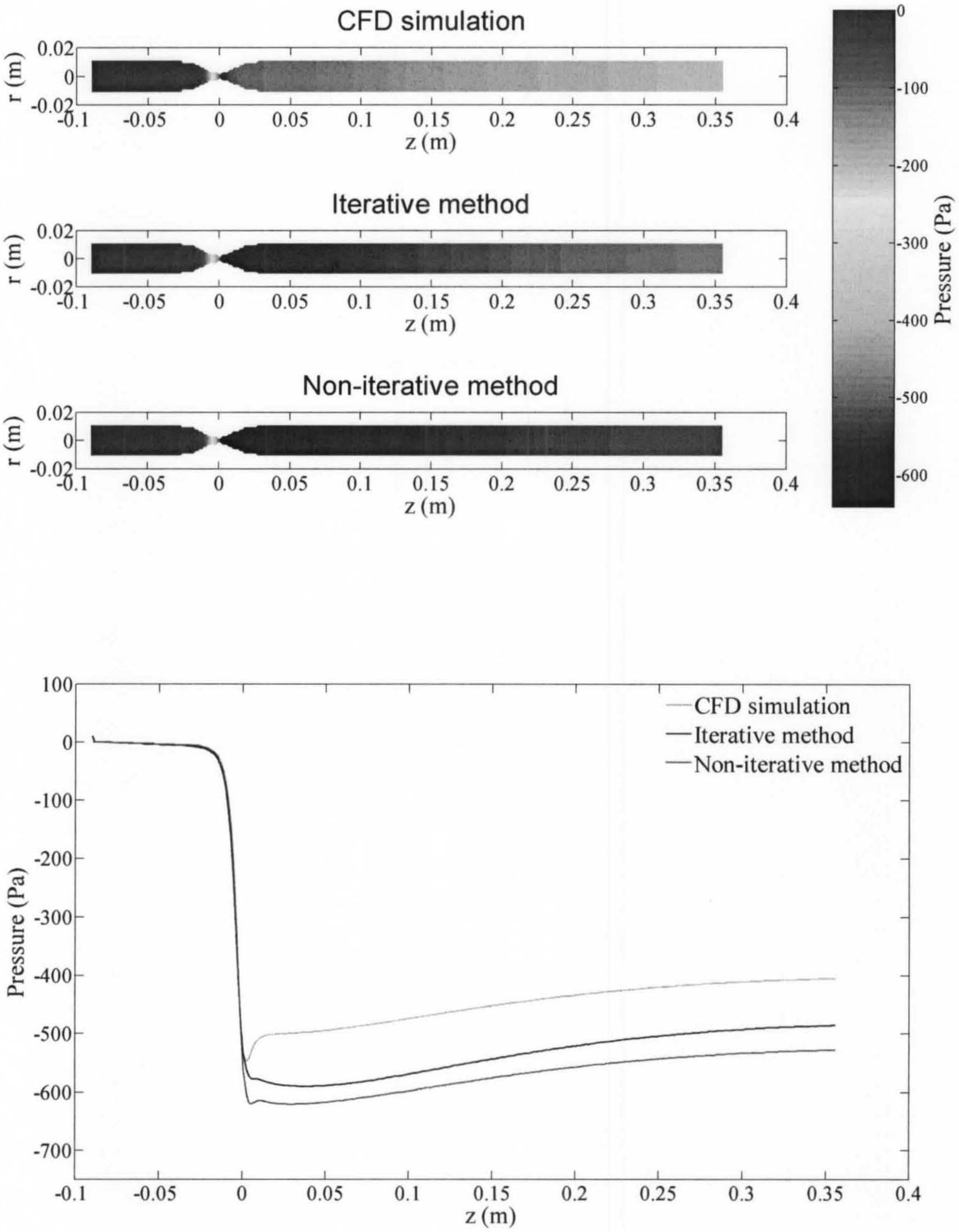


Figure 6-16. Pressure contour plot (top) and pressure profile along the centerline (bottom) obtained directly from CFD simulations, and calculated using iterative method and non-iterative methods on the velocity data regridded to a rectangular mesh at resolution 1.57 mm.

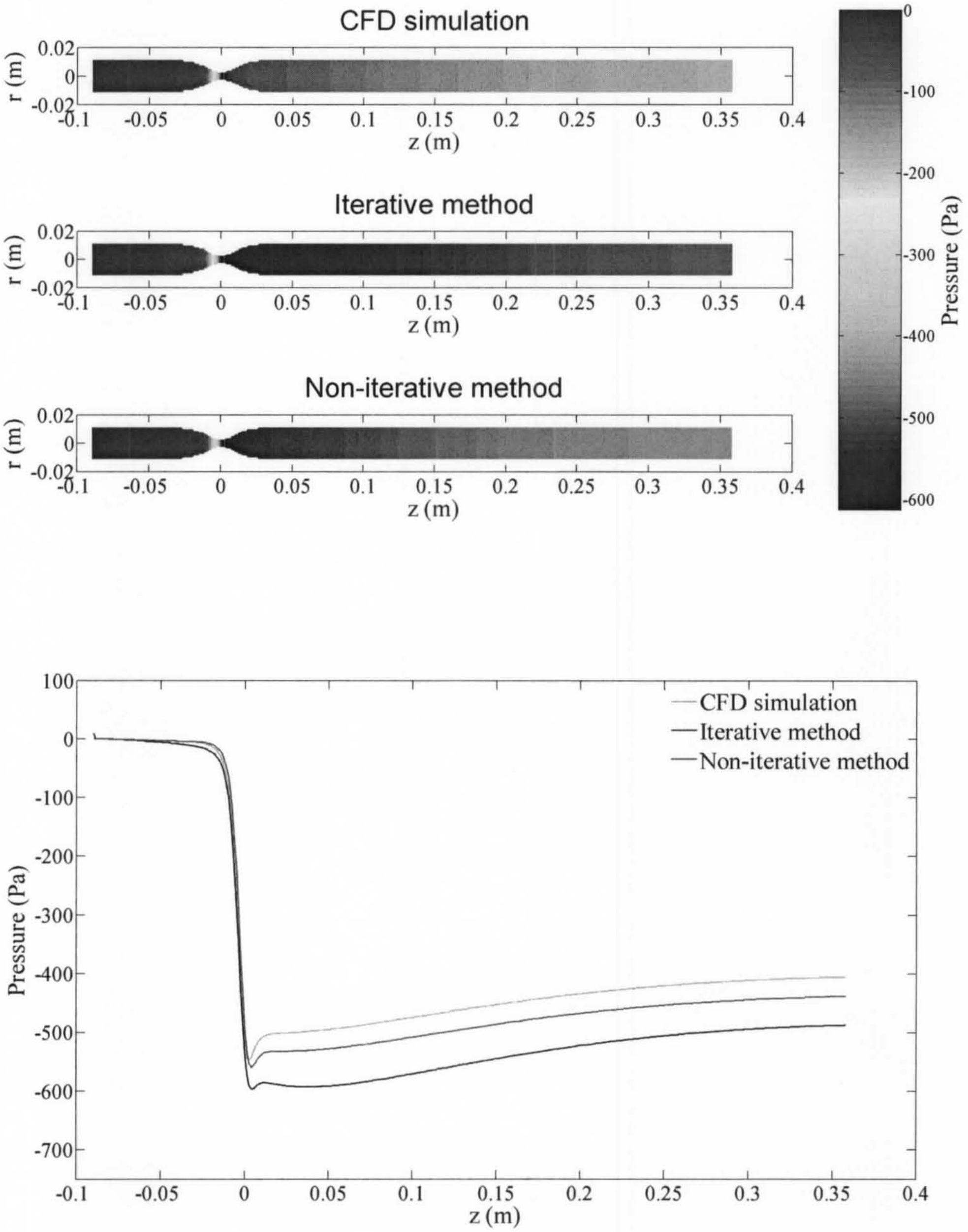


Figure 6-17. Pressure contour plot (top) and pressure profile along the centerline (bottom) obtained directly from CFD simulations, and calculated using iterative method and non-iterative methods on the velocity data regridded to a rectangular mesh at resolution 1.25 mm.

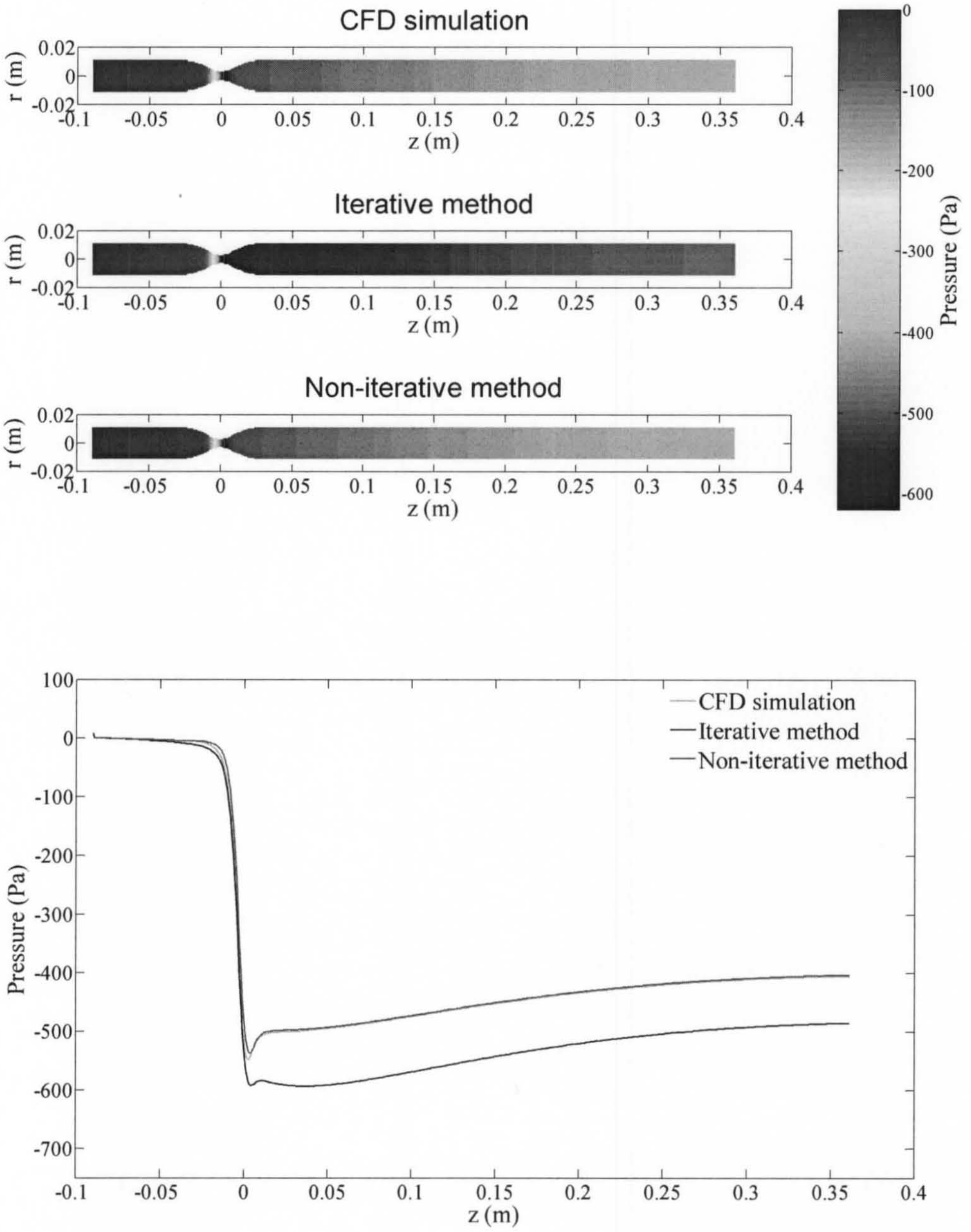


Figure 6-18. Pressure contour plot (top) and pressure profile along the centerline (bottom) obtained directly from CFD simulations, and calculated using iterative method and non-iterative methods on the velocity data regridded to a rectangular mesh at resolution 0.94 mm.

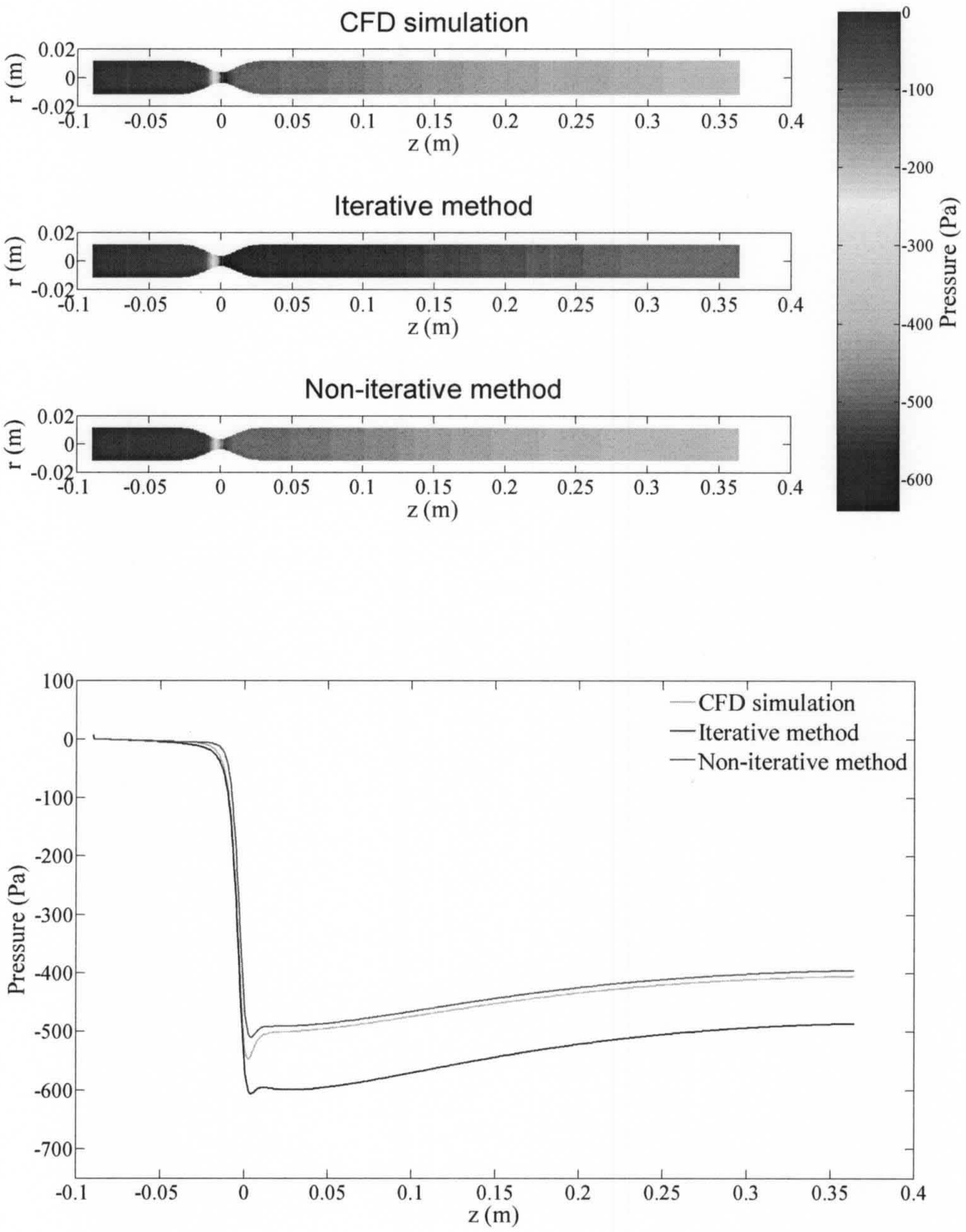


Figure 6-19. Pressure contour plot (top) and pressure profile along the centerline (bottom) obtained directly from CFD simulations, and calculated using iterative method and non-iterative methods on the velocity data regridded to a rectangular mesh at resolution 0.63 mm.

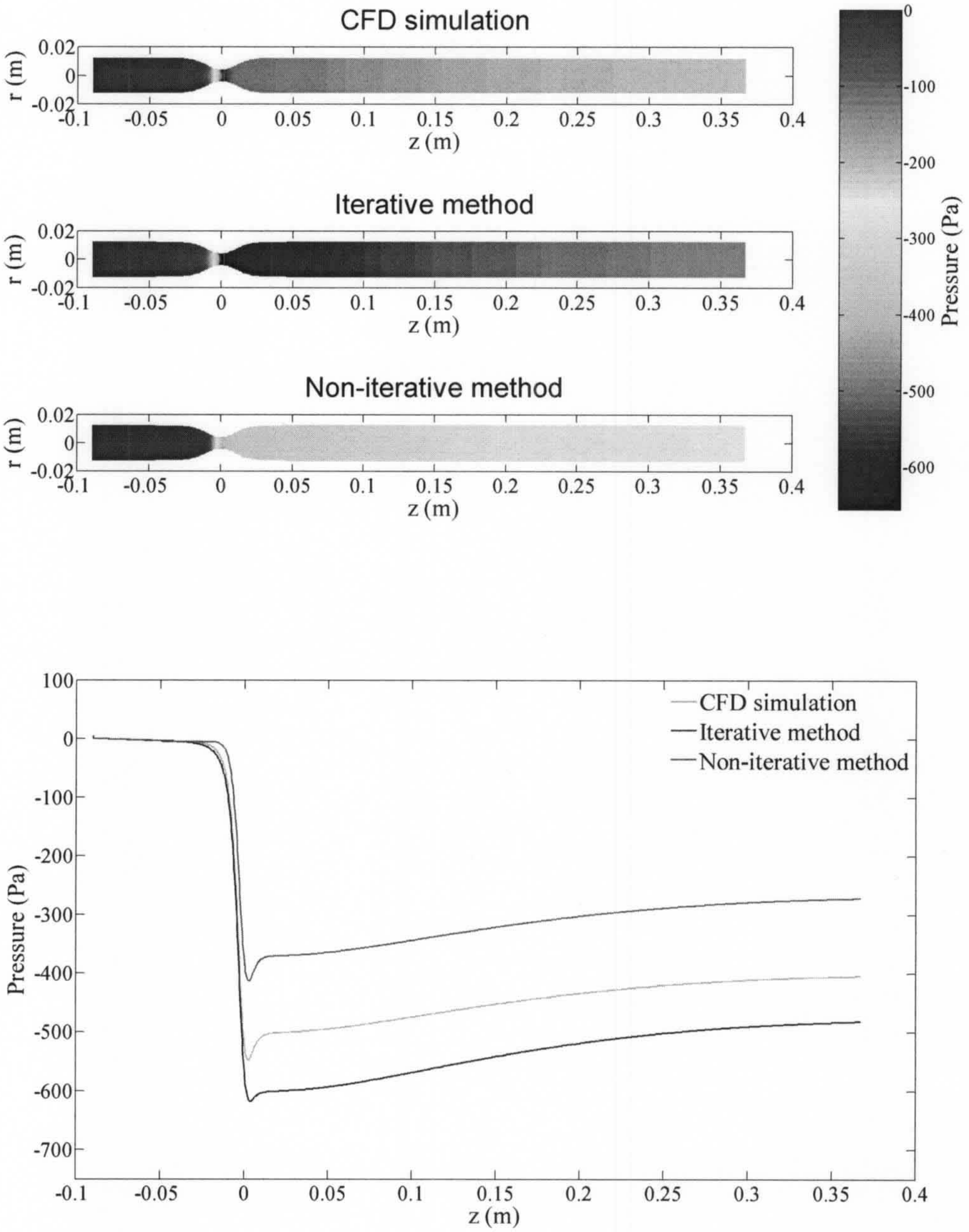


Figure 6-20. Pressure contour plot (top) and pressure profile along the centerline (bottom) obtained directly from CFD simulations, and calculated using iterative method and non-iterative methods on the velocity data regridded to a rectangular mesh at resolution 0.31 mm.

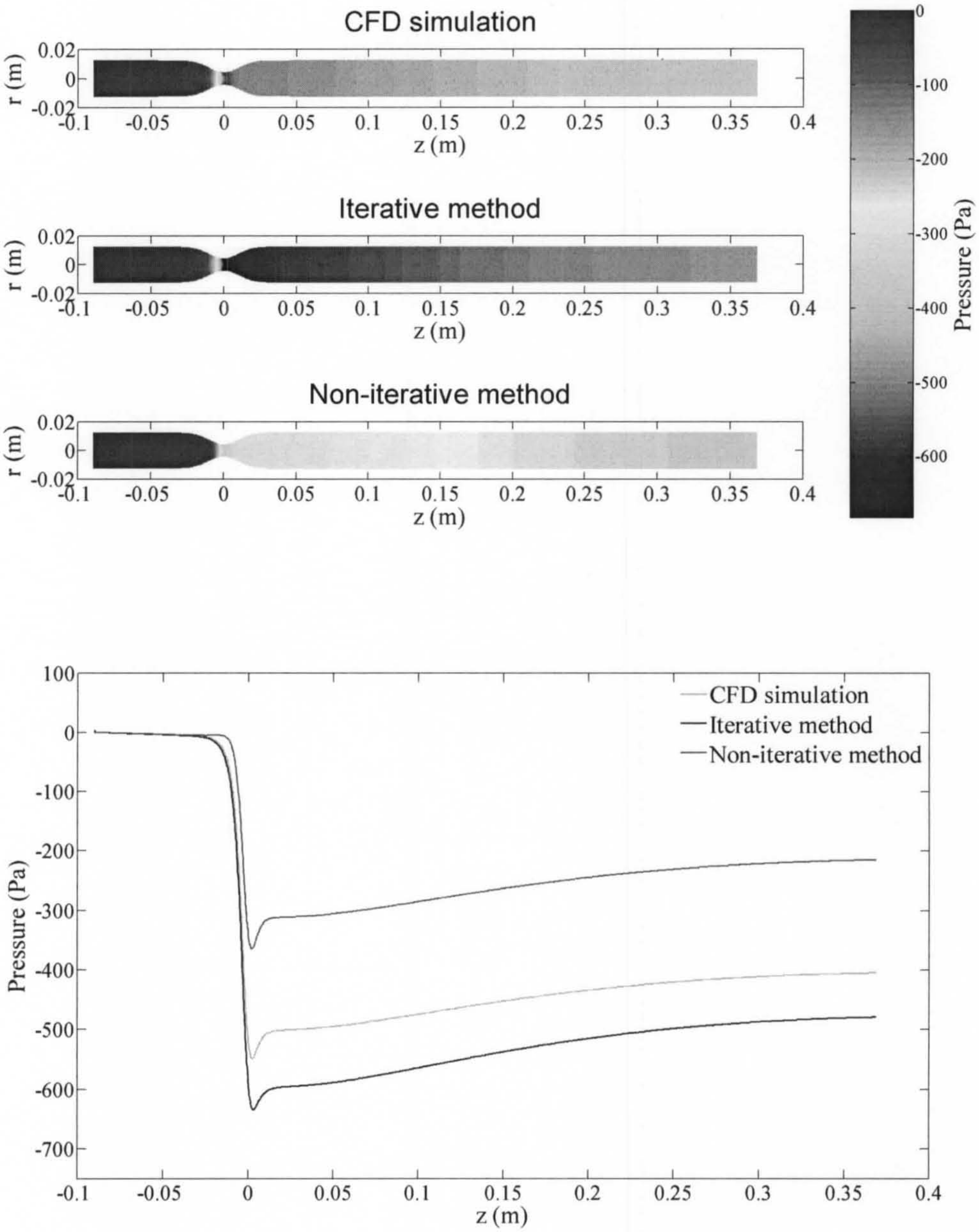


Figure 6-21. Pressure contour plot (top) and pressure profile along the centerline (bottom) obtained directly from CFD simulations, and calculated using iterative method and non-iterative methods on the velocity data regridded to a rectangular mesh at resolution 0.15 mm.

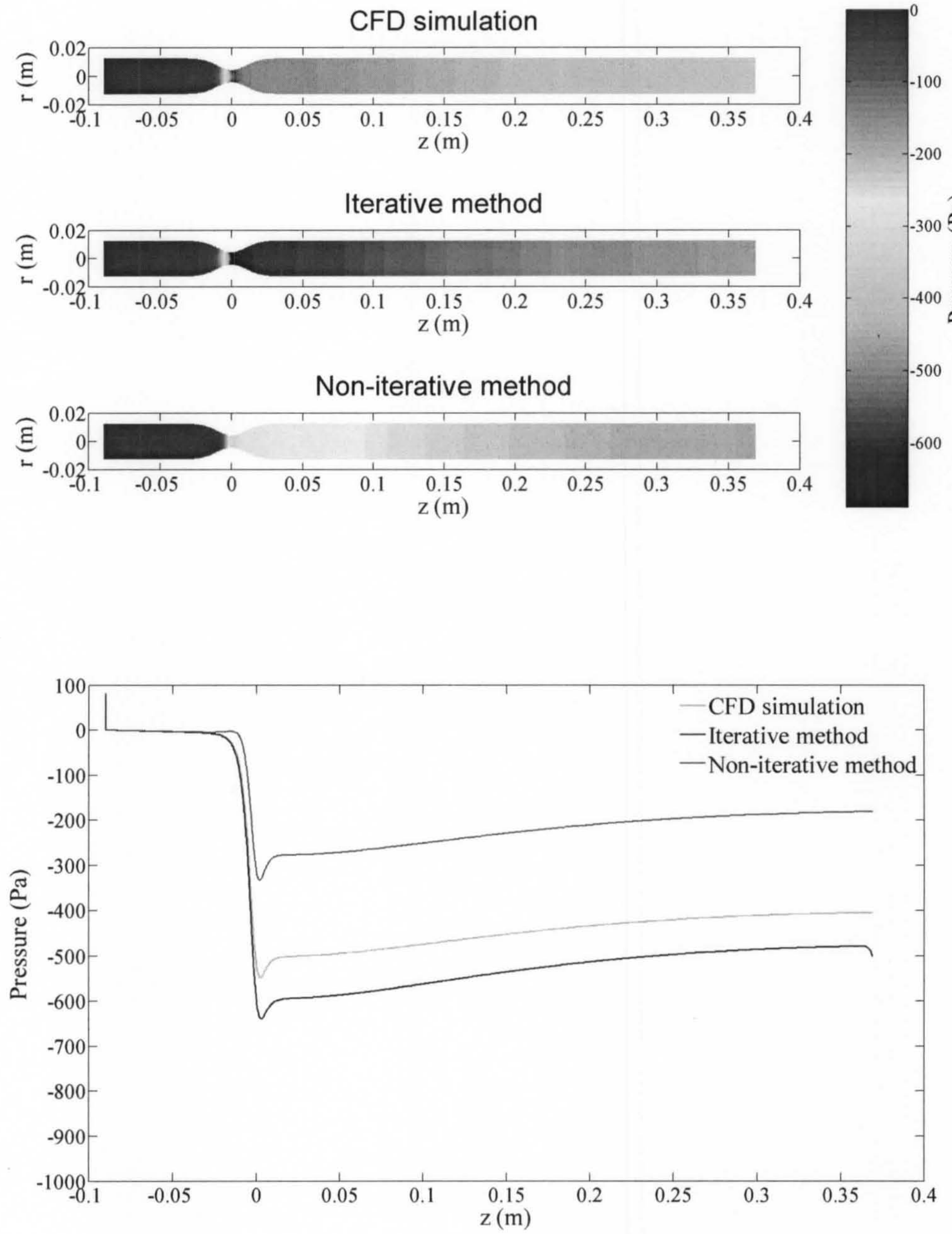


Figure 6-22. Pressure contour plot (top) and pressure profile along the centerline (bottom) obtained directly from CFD simulations, and calculated using iterative method and non-iterative methods on the velocity data regridded to a rectangular mesh at resolution 0.10 mm.

Table 6-4

Normalized root mean square error (NRMSE) and CPU time usage of the pressure calculation using iterative and non-iterative techniques using CFD simulation as the basis for comparison.

Resolution (mm)	NRMSE (%)		CPU time (sec)	
	Iterative	Non-iterative	Iterative	Non-iterative
1.88	16.6	7.3	3.78	0.11
1.57	19.7	27.5	4.82	0.14
1.25	20.0	7.4	8.31	0.19
0.94	19.6	0.9	18.31	0.23
0.63	20.1	2.4	53.69	0.67
0.31	19.5	29.5	234.98	2.17
0.15	18.7	42.6	1318.37	12.09
0.10	18.3	50.0	10852.18	34.68

6.3.2 Catheter Assisted Pressure Measurement

Pressure drop along the centerline of the phantom measured directly using pressure catheters are depicted in Figure 6-23-Figure 6-25. Here, the inlet pressure at 8 cm upstream of the stenosis throat was chosen as the reference point and set to zero for all measurements. Figure 6-23 shows the mean and standard deviation (SD) of the direct pressure (catheter) for N > 7000 readings at several axial locations for steady flows SF-1, SF-2 and SF-3. Figure 6-24 and Figure 6-25 show the mean and SD of the direct pressure measurement for N > 24 readings at several locations for pulsatile flows PF-1, PF-2 and PF-3 at end-diastole (Figure 6-10, t = 275 ms) and peak-systole (Figure 6-10, t = 400 ms), respectively.

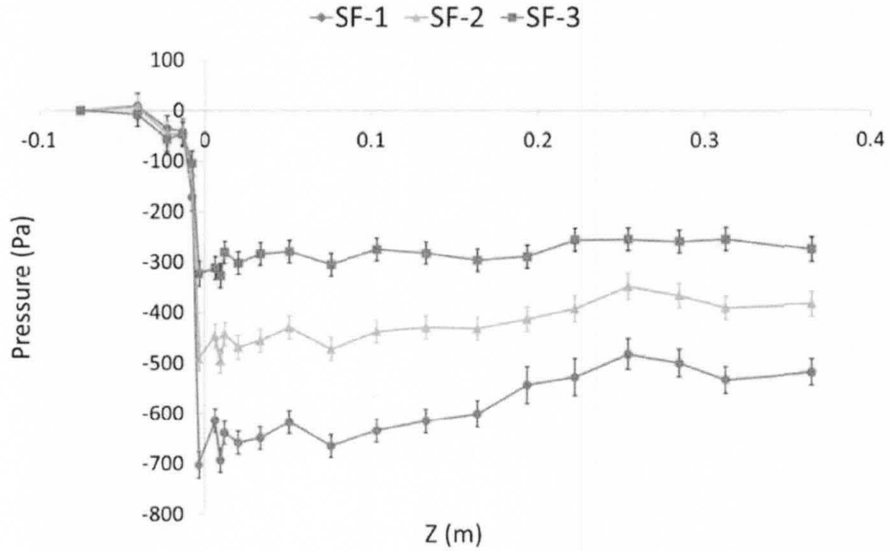


Figure 6-23. Pressure profile along the centerline of the 87% area stenosis phantom with steady flow rates of SF-1 (56.3 ml/s), SF-1 (46.9 ml/s), and SF-3 (37.5ml/s). Error bars show the standard deviation of the measured pressure for N > 7000 readings.

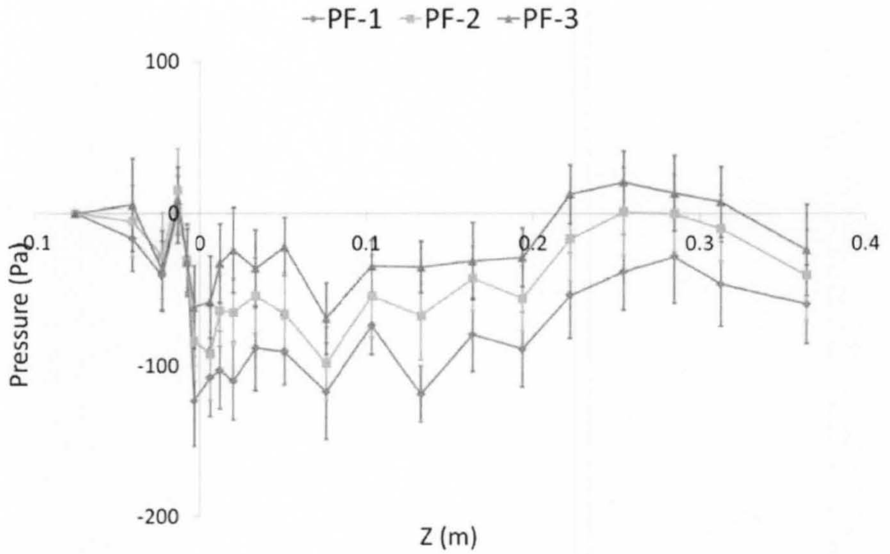


Figure 6-24. Pressure profile along the centerline of the 87% area stenosis phantom with pulsatile flow rates of PF-1 (184 ml/s), PF-1 (153 ml/s), and PF-3 (122 ml/s) at end-diastole (t = 275 ms on flow waveform curve). Error bars show the standard deviation of the measured pressure for N > 24 readings.

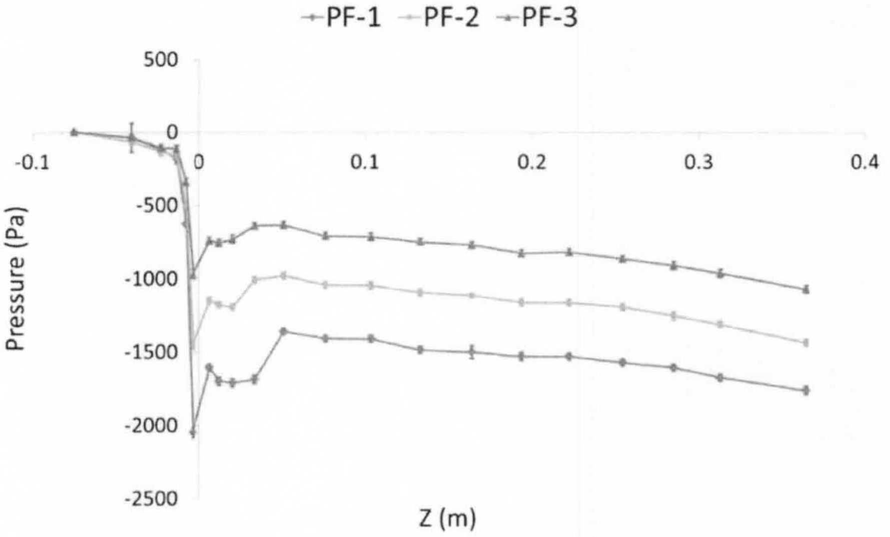


Figure 6-25. Pressure profile along the centerline of the 87% area stenosis phantom with pulsatile flow rates of PF-1 (184 ml/s), PF-1 (153 ml/s), and PF-3 (122 ml/s) at peak-systole (t = 400 ms on flow waveform curve). Error bars show the standard deviation of the measured pressure for N > 24 readings.

6.3.3 CFD simulation

Pressure drop along the centerline of the phantom was obtained directly from CFD simulations for steady flows (Figure 6-26), for pulsatile flows at end-diastole (Figure 6-27), and for pulsatile flows at peak-systole (Figure 6-28). The inlet pressure at 8 cm upstream of the stenosis throat was chosen as the reference point and set to zero for all measurements.

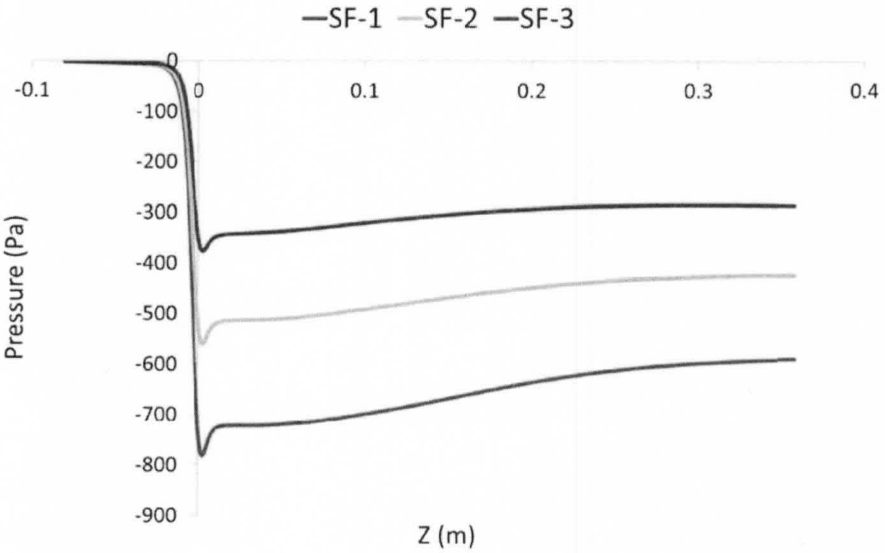


Figure 6-26. Pressure profile along the centerline of the 87% area stenosis phantom with steady flow rates of SF-1 (56.3 ml/s), SF-1 (46.9 ml/s), and SF-3 (37.5ml/s) obtained directly from CFD simulation.

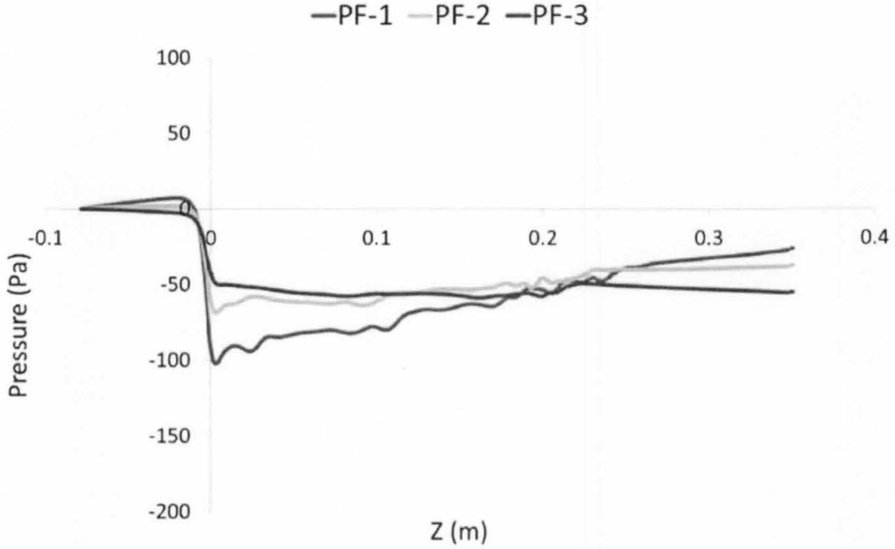


Figure 6-27. Pressure profile along the centerline of the 87% area stenosis phantom with pulsatile flow rates of PF-1 (184 ml/s), PF-1 (153 ml/s), and PF-3 (122 ml/s) at end-diastole (t = 275 ms on flow waveform curve) obtained directly from CFD simulation obtained directly from CFD simulation.

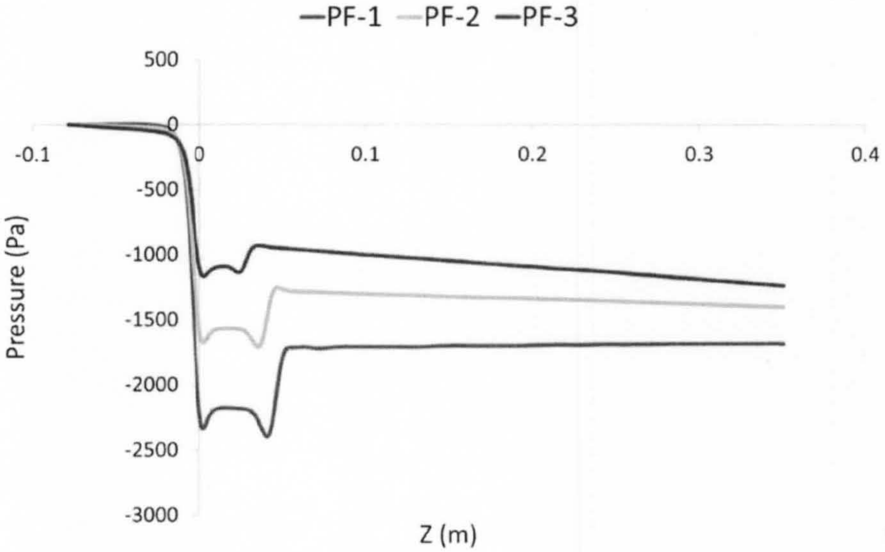


Figure 6-28. Pressure profile along the centerline of the 87% area stenosis phantom with pulsatile flow rates of PF-1 (184 ml/s), PF-1 (153 ml/s), and PF-3 (122 ml/s) at peak-systole (t = 400 ms on flow waveform curve) obtained directly from CFD simulation obtained directly from CFD simulation.

6.3.4 CFD velocity data using iterative method

Pressure drop along the centerline of the phantom was calculated from CFD velocity data using iterative method for steady flows (Figure 6-29), for pulsatile flows at end-diastole (Figure 6-30), and for pulsatile flows at peak-systole (Figure 6-31). The inlet pressure at 8 cm upstream of the stenosis throat was chosen as the reference point and set to zero for all measurements.

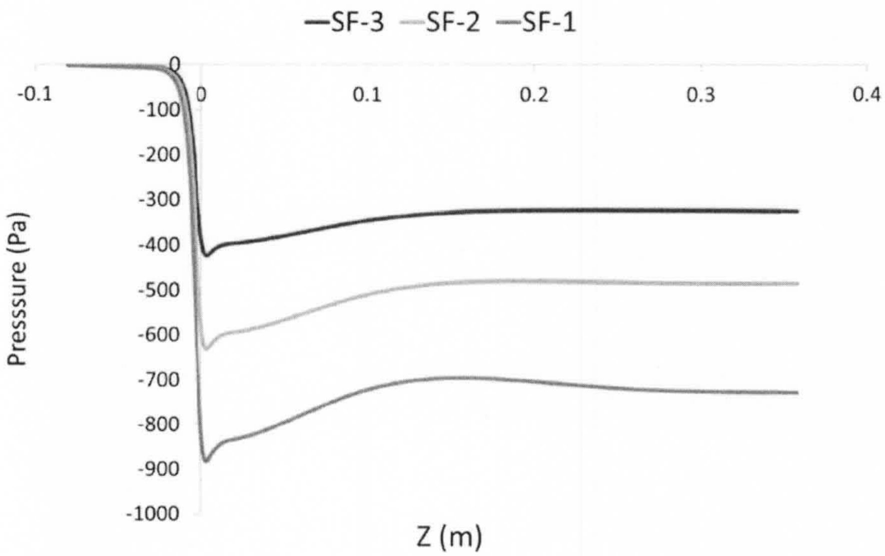


Figure 6-29. Pressure profile along the centerline of the 87% area stenosis phantom with steady flow rates of SF-1 (56.3 ml/s), SF-1 (46.9 ml/s), and SF-3 (37.5ml/s) calculated from CFD velocity data using iterative method.

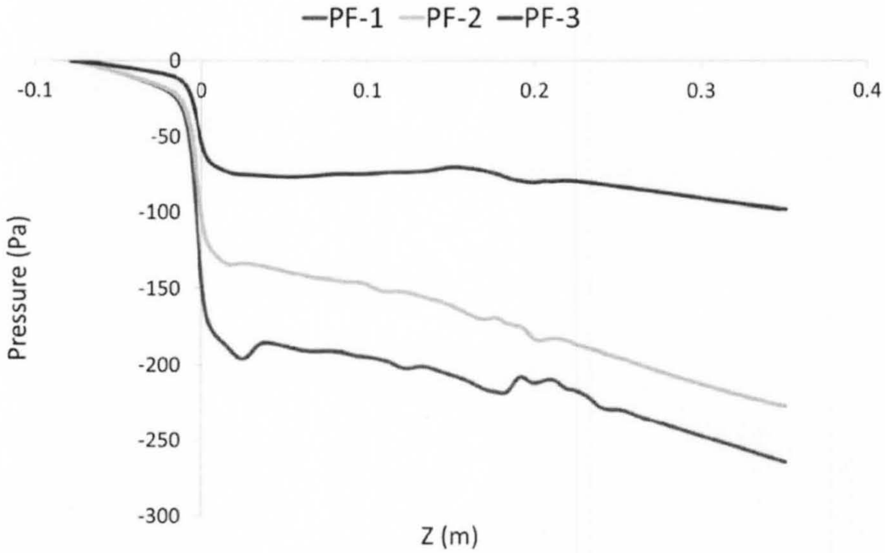


Figure 6-30. Pressure profile along the centerline of the 87% area stenosis phantom with steady flow rates of PF-1 (184 ml/s), PF-1 (153 ml/s), and PF-3 (122 ml/s) at end-diastole (t = 275 ms on flow waveform curve) calculated from CFD velocity data using iterative method.

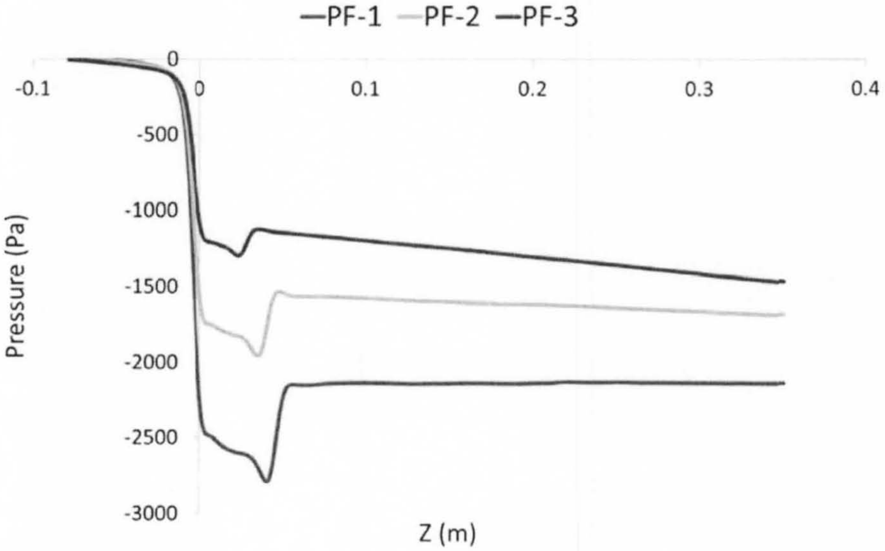


Figure 6-31. Pressure profile along the centerline of the 87% area stenosis phantom with steady flow rates of PF-1 (184 ml/s), PF-1 (153 ml/s), and PF-3 (122 ml/s) at peak-systole (t = 400 ms on flow waveform curve) calculated from CFD velocity data using iterative method.

6.3.5 CFD velocity data using non-iterative method

Pressure drop along the centerline of the phantom was calculated from CFD velocity data using non-iterative method for steady flows (Figure 6-32), for pulsatile flows at end-diastole (Figure 6-33), and for pulsatile flows at peak-systole (Figure 6-34). The inlet pressure at 8 cm upstream of the stenosis throat was chosen as the reference point and set to zero for all measurements.

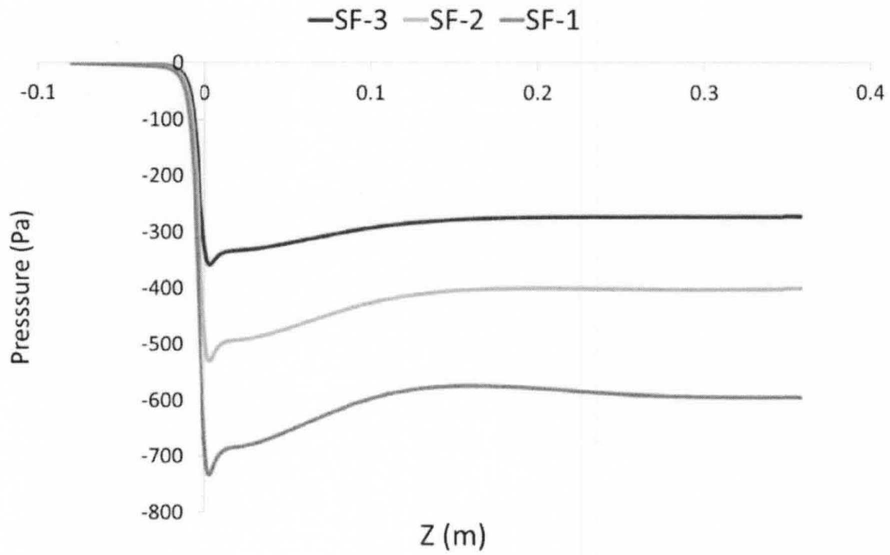


Figure 6-32. Pressure profile along the centerline of the 87% area stenosis phantom with steady flow rates of SF-1 (56.3 ml/s), SF-1 (46.9 ml/s), and SF-3 (37.5ml/s) calculated from CFD velocity data using non-iterative method.

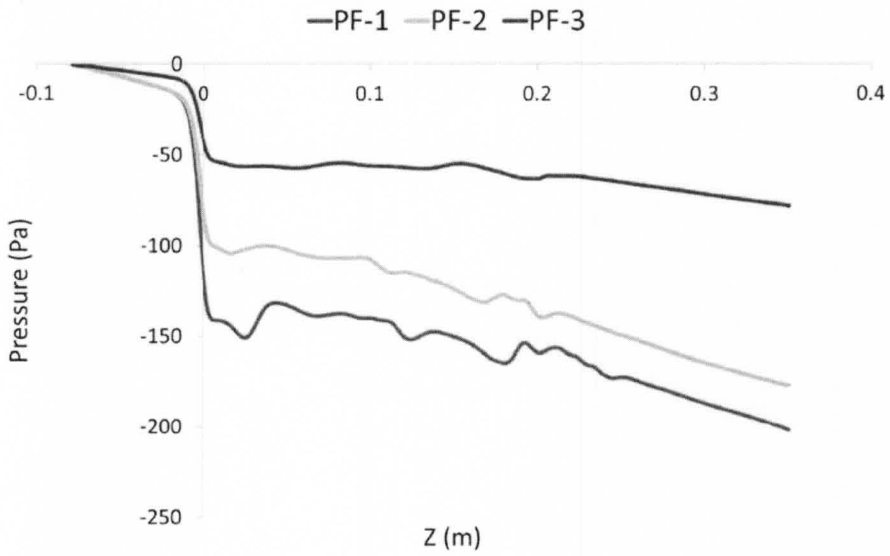


Figure 6-33. Pressure profile along the centerline of the 87% area stenosis phantom with steady flow rates of PF-1 (184 ml/s), PF-1 (153 ml/s), and PF-3 (122 ml/s) at end-diastole (t = 275 ms on flow waveform curve) calculated from CFD velocity data using non-iterative method.

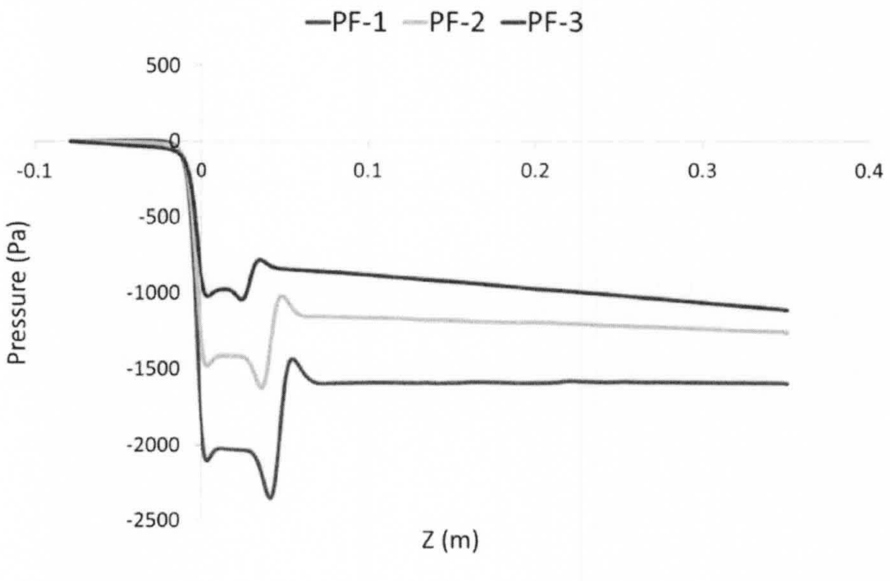


Figure 6-34. Pressure profile along the centerline of the 87% area stenosis phantom with steady flow rates of PF-1 (184 ml/s), PF-1 (153 ml/s), and PF-3 (122 ml/s) at peak-systole (t = 400 ms on flow waveform curve) calculated from CFD velocity data using non-iterative method.

Results of the two iterative (previous section) and non-iterative methods applied on the CFD velocity data were compared in terms of CPU usage time and normalized root mean square error (NRMSE) over the entire volume. Normalized root mean square error (NRMSE) was calculated, as discussed before. Results are represented in Table 6-5.

Table 6-5

Normalized root mean square error (NRMSE) and CPU time usage of the pressure calculation using iterative and non-iterative techniques applied on the CFD velocity data using CFD simulated pressures as the basis for comparison (Resolution = 1 mm).

Flow case	NRMSE (%)		CPU time (sec)	
	Iterative	Non-iterative	Iterative	Non-iterative
SF-1	13.69	9.53	681.35	10.80
SF-2	9.96	9.68	628.13	10.28
SF-3	11.36	6.28	678.03	9.61
PF-1 (peak-systole)	24.43	6.68	162.10	0.69
PF-2 (peak-systole)	20.38	10.35	67.65	0.25
PF-3 (peak-systole)	19.14	11.69	45.38	0.39
PF-1 (end-diastole)	264.98	177.70	13.14	0.28
PF-2 (end-diastole)	257.24	175.50	10.08	0.25
PF-3 (end-diastole)	50.19	20.75	3.13	0.55

6.3.6 PC-MRI velocity data using iterative method

Pressure drop along the centerline of the phantom was calculated from PC-MRI velocity data using iterative method for steady flows (Figure 6-35), for pulsatile flows at end-diastole (Figure 6-36), and for pulsatile flows at peak-systole (Figure 6-37). The inlet pressure at 8 cm upstream of the stenosis throat was chosen as the reference point and set to zero for all measurements.

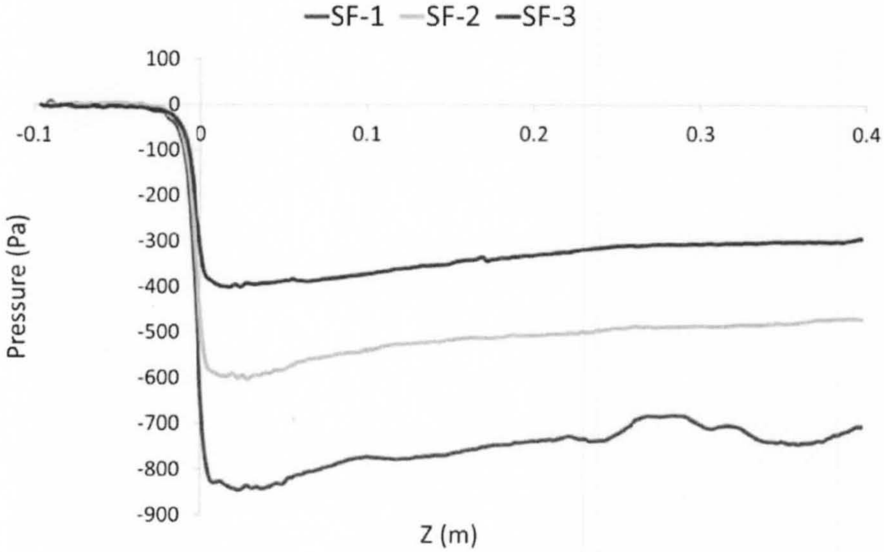


Figure 6-35. Pressure profile along the centerline of the 87% area stenosis phantom with steady flow rates of SF-1 (56.3 ml/s), SF-1 (46.9 ml/s), and SF-3 (37.5ml/s) calculated from PC-MRI velocity data using iterative method.

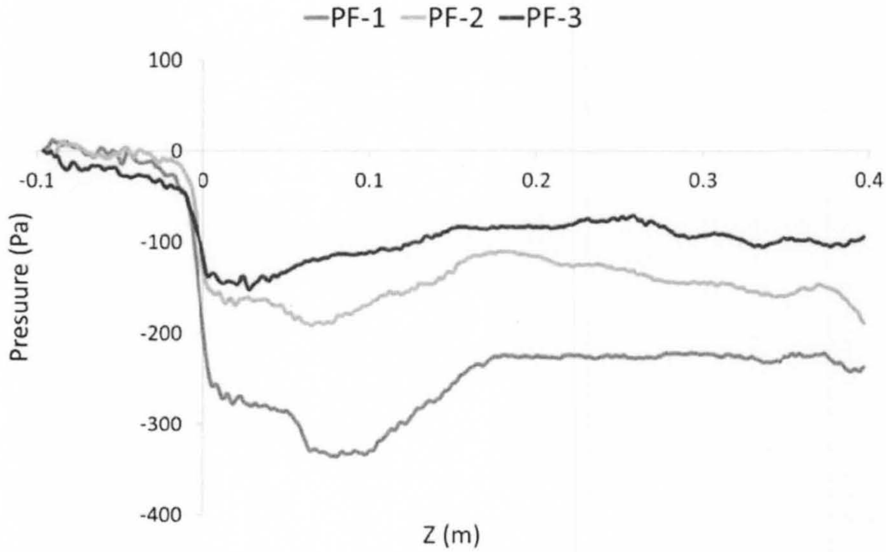


Figure 6-36. Pressure profile along the centerline of the 87% area stenosis phantom with steady flow rates of PF-1 (184 ml/s), PF-1 (153 ml/s), and PF-3 (122 ml/s) at end-diastole (t = 275 ms on flow waveform curve) calculated from PC-MRI velocity data using iterative method.

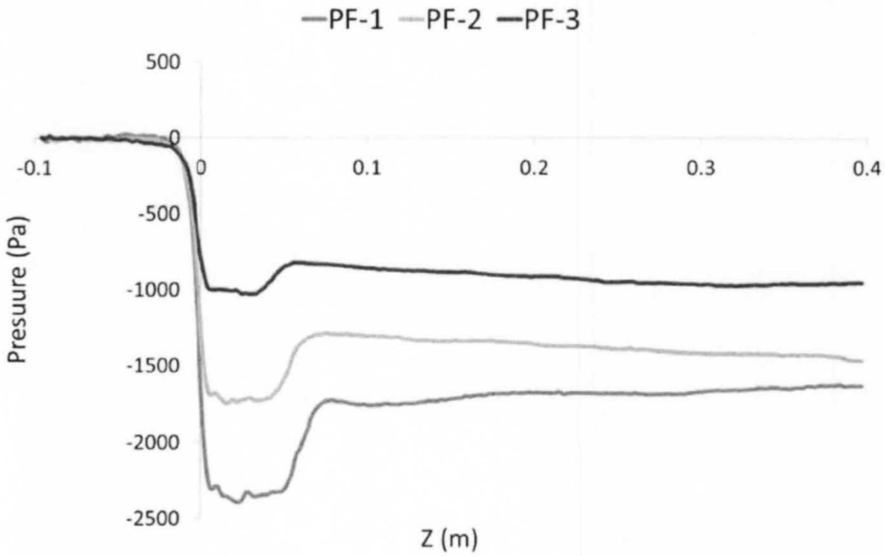


Figure 6-37. Pressure profile along the centerline of the 87% area stenosis phantom with steady flow rates of PF-1 (184 ml/s), PF-1 (153 ml/s), and PF-3 (122 ml/s) at peak-systole (t = 400 ms on flow waveform curve) calculated from PC-MRI velocity data using iterative method.

6.3.7 PC-MRI velocity data using non-iterative method

Pressure drop along the centerline of the phantom was calculated from PC-MRI velocity data using non-iterative method for steady flows (Figure 6-38), for pulsatile flows at end-diastole (Figure 6-39), and for pulsatile flows at peak-systole (Figure 6-40). The inlet pressure at 8 cm upstream of the stenosis throat was chosen as the reference point and set to zero for all measurements.

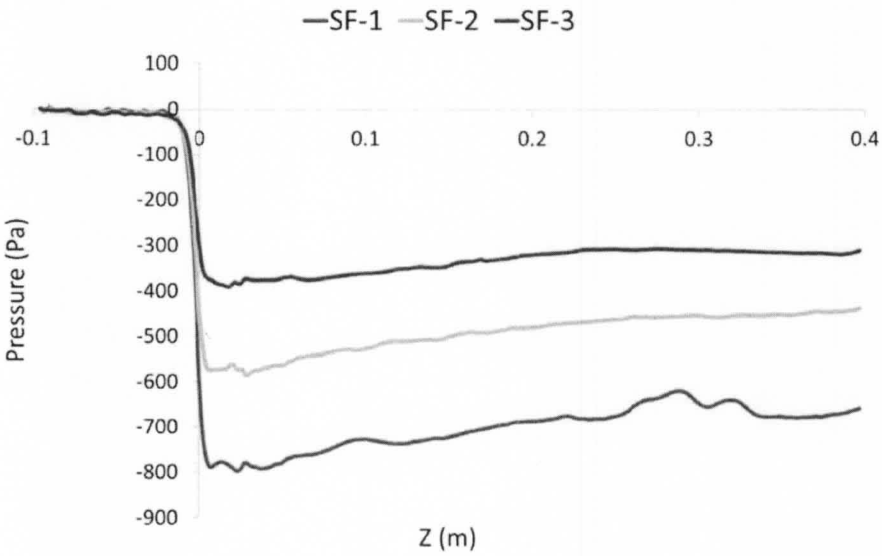


Figure 6-38. Pressure profile along the centerline of the 87% area stenosis phantom with steady flow rates of SF-1 (56.3 ml/s), SF-1 (46.9 ml/s), and SF-3 (37.5ml/s) calculated from PC-MRI velocity data using non-iterative method.

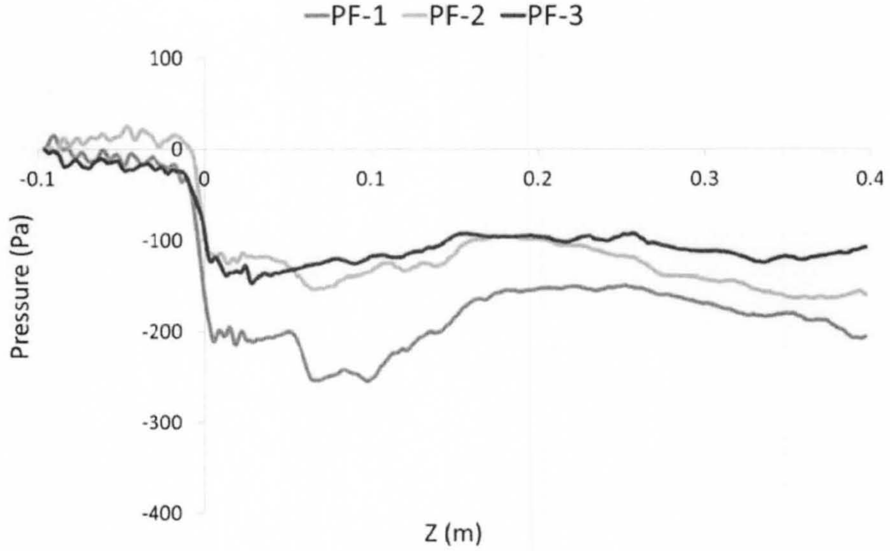


Figure 6-39. Pressure profile along the centerline of the 87% area stenosis phantom with steady flow rates of PF-1 (184 ml/s), PF-1 (153 ml/s), and PF-3 (122 ml/s) at end-diastole (t = 275 ms on flow waveform curve) calculated from PC-MRI velocity data using non-iterative method.

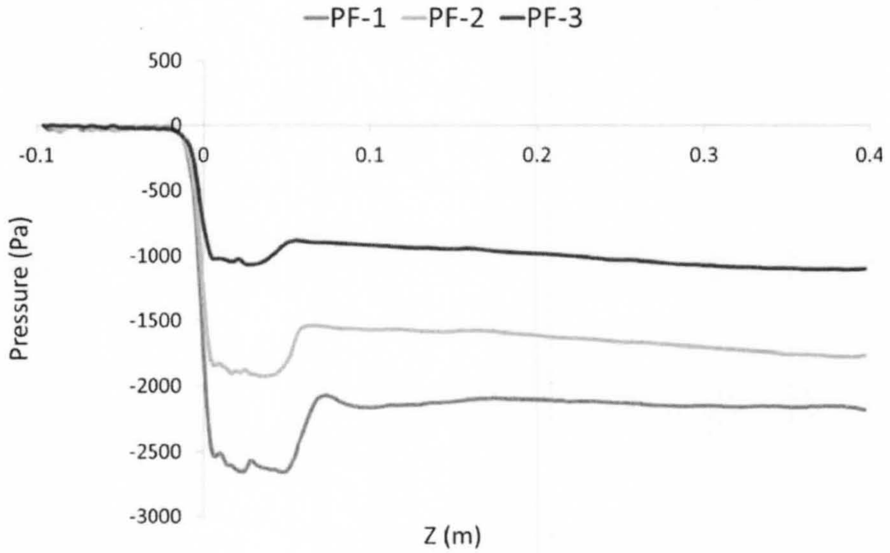


Figure 6-40. Pressure profile along the centerline of the 87% area stenosis phantom with steady flow rates of PF-1 (184 ml/s), PF-1 (153 ml/s), and PF-3 (122 ml/s) at peak-systole (t = 400 ms on flow waveform curve) calculated from PC-MRI velocity data using non-iterative method.

Results of the two iterative (previous section) and non-iterative methods applied on the PC-MRI velocity data were compared in terms of CPU usage time. Results are represented in Table 6-6.

Table 6-6

CPU time usage of the pressure calculation using iterative and non-iterative techniques applied on the PC-MRI velocity data.

Flow case	CPU time (sec)	
	Iterative	Non-iterative
SF-1	189.23	0.90
SF-2	114.70	1.59
SF-3	302.59	1.81
PF-1 (peak-systole)	850.35	0.81
PF-2 (peak-systole)	637.63	1.03
PF-3 (peak-systole)	318.64	0.95
PF-1 (end-diastole)	203.53	1.09
PF-2 (end-diastole)	139.92	0.97
PF-3 (end-diastole)	119.01	1.05
PF-1 (whole cardiac cycle)	~ 22K	43.1
PF-2 (whole cardiac cycle)	~ 17K	47.9
PF-3 (whole cardiac cycle)	~ 10K	53.8

6.3.8 SPIV velocity data using iterative method

Pressure drop along the centerline of the phantom was calculated from SPIV velocity data using iterative method for steady flows (Figure 6-41). The inlet pressure at 8 cm upstream of the stenosis throat was chosen as the reference point and set to zero for all measurements.

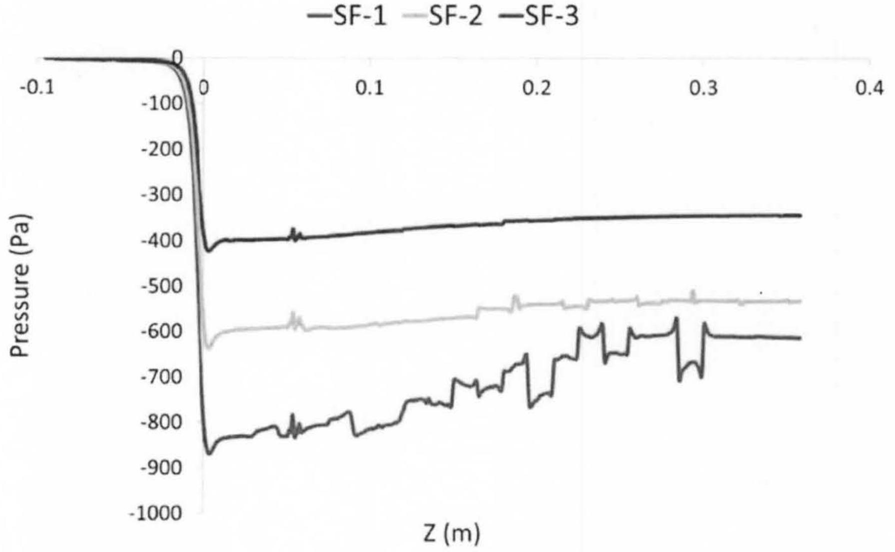


Figure 6-41. Pressure profile along the centerline of the 87% area stenosis phantom with steady flow rates of SF-1 (56.3 ml/s), SF-1 (46.9 ml/s), and SF-3 (37.5ml/s) calculated from SPIV velocity data using iterative method.

6.3.9 SPIV velocity data using non-iterative method

Pressure drop along the centerline of the phantom was calculated from SPIV velocity data using non-iterative method for steady flows (Figure 6-42). The inlet pressure at 8 cm upstream of the stenosis throat was chosen as the reference point and set to zero for all measurements.

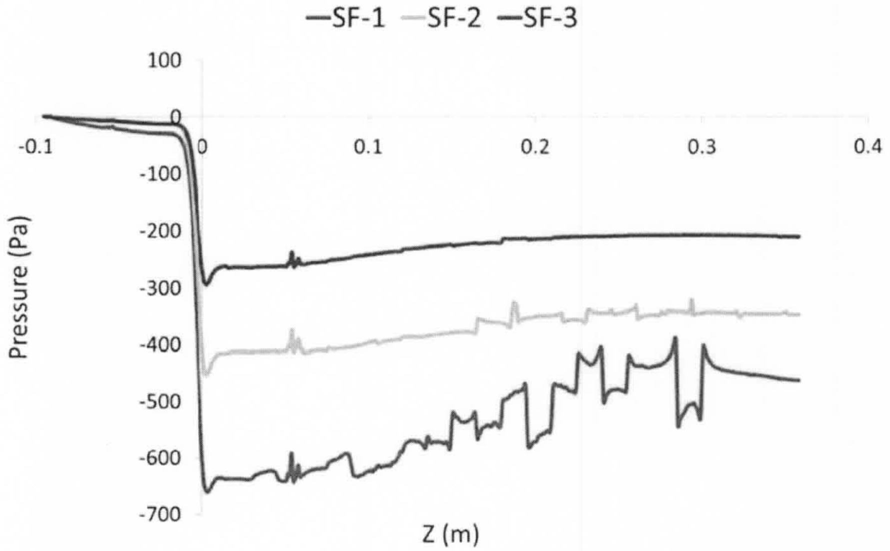


Figure 6-42. Pressure profile along the centerline of the 87% area stenosis phantom with steady flow rates of SF-1 (56.3 ml/s), SF-1 (46.9 ml/s), and SF-3 (37.5ml/s) calculated from SPIV velocity data using non-iterative method.

Results of the two iterative (previous section) and non-iterative methods applied on the SPIV velocity data were compared in terms of CPU usage time. Results are represented in Table 6-7.

Table 6-7

CPU time usage of the pressure calculation using iterative and non-iterative techniques applied on the SPIV velocity data.

Flow case	CPU time (sec)	
	Iterative	Non-iterative
SF-1	25872.26	10.83
SF-2	7959.46	10.76
SF-3	3953.21	10.08

6.3.10 Multimodality comparison

The pressure drop along the centerline of the phantom is shown in Figure 6-43 for steady flow SF-2 with different methods. Here, the pressure 8 cm upstream of the stenosis throat was chosen as the reference point (Pref) and set to zero for the pressure calculated from other techniques, including PC-MRI, SPIV, CFD, Fluent, and Catheter.

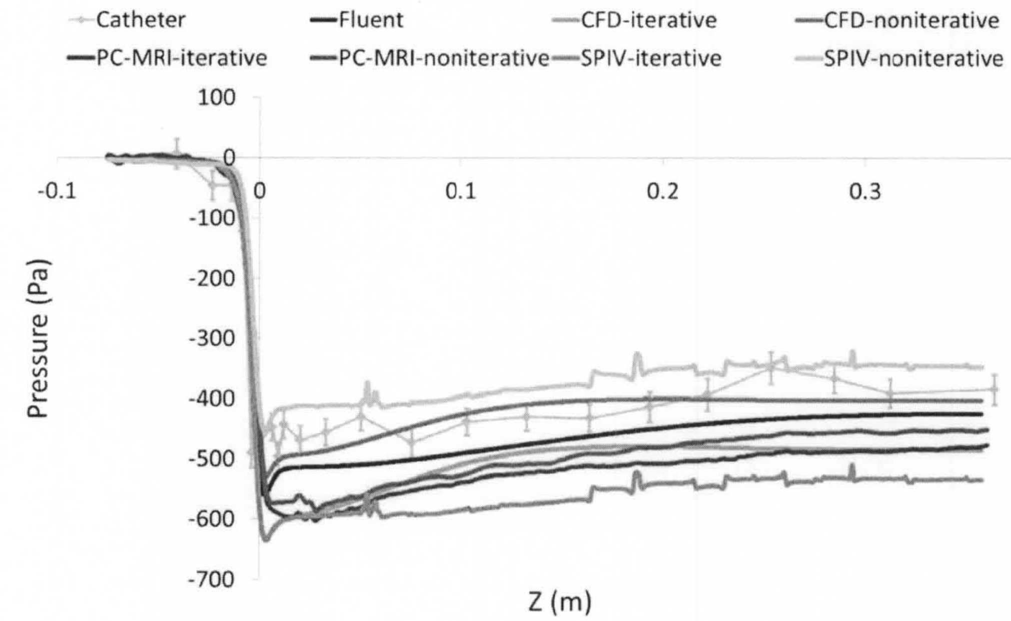


Figure 6-43. Pressure profile along the centerline of the phantom obtained from different methods for steady flow SF-2. Error bars show the standard deviation of the measured pressure for N > 7000 readings. The inlet pressure at 8 cm upstream of the stenosis was set to the same value for all methods.

The pressure drop for each method was calculated using the reference point as: $\mathrm{PD} = \mathrm{P} - \mathrm{P}_{\mathrm{ref}}$. The relative error in the pressure drop was then defined as $\sum\left|\mathrm{PD}_{\mathrm{i}} - \mathrm{PD}_{\mathrm{i}}^{\mathrm{fluent}}\right| / \sum\left|\mathrm{PD}_{\mathrm{i}}^{\mathrm{fluent}}\right|$ for each modality and/or technique, where summation ("*i*") is over all data points along the centerline of the phantom for each method. Fluent pressure was chosen as the basis for comparison since it showed better results compared to that of the catheter at end-diastole (compare Figure 6-24 with Figure 6-27). The relative error in pressure drop for catheter, PC-MRI (iterative), PC-MRI (non-iterative), CFD (iterative), CFD (non-iterative), SPIV (iterative), and SPIV (non-iterative) methods were 13.1%, 12.5%, 7.8%, 9.3%, 9.0%, 20.5%, and 20.1%, respectively.

Figure 6-44 shows the pressure drop along the centerline of the phantom for pulsatile flow PF-2 at peak-systole (t = 400 ms on flow waveform curve) with different methods. Using the Fluent pressure as the ground truth, the relative error in pressure drop for catheter, PC-MRI (iterative), PC-MRI (non-iterative), CFD (iterative), and CFD (non-iterative) methods were 20.3%, 4.0%, 22.1%, 19.8%, and 10.7%, respectively.

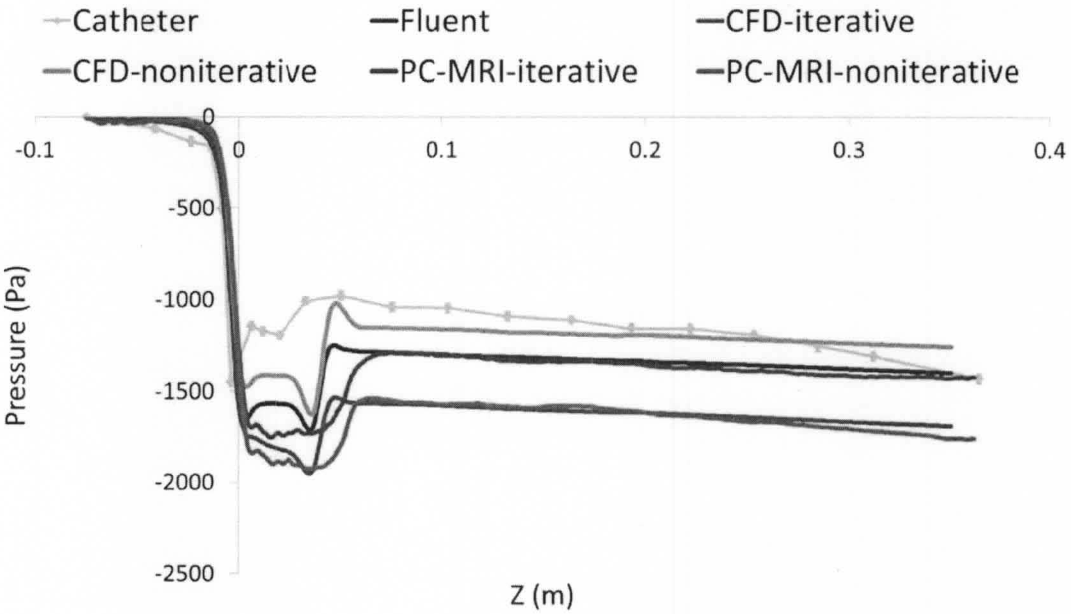


Figure 6-44. Pressure profile along the centerline of the phantom obtained from different methods at peak-systole (t = 400 ms on flow waveform curve) for pulsatile flow PF-2. Error bars show the standard deviation of the measured pressure for N > 24 readings. The inlet pressure at 8 cm upstream of the stenosis was set to the same value for all methods.

Figure 6-45 shows the pressure drop along the centerline of the phantom for pulsatile flow PF-2 at end-diastole (t = 275 ms on flow waveform curve) with different methods. Using the pressure obtained from Fluent as the ground truth, the relative error in pressure drop for catheter, PC-MRI (iterative), PC-MRI (non-iterative), CFD (iterative), and CFD (non-iterative) methods were 44.1%, 194.5%, 155.2%, 245.2%, and 162.8%, respectively.

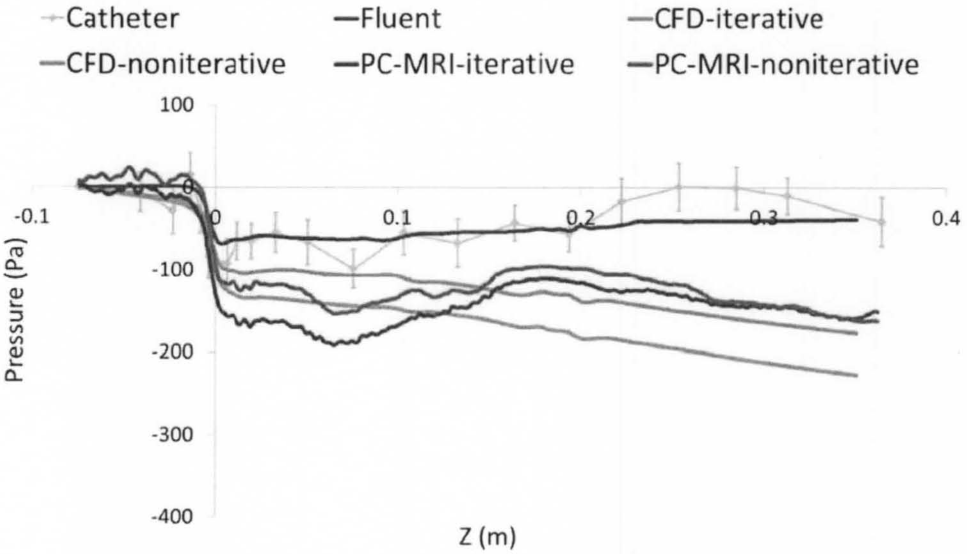


Figure 6-45. Pressure profile along the centerline of the phantom obtained from different methods at end-diastole (t = 275 ms on flow waveform curve) for pulsatile flow PF-2. Error bars show the standard deviation of the measured pressure for N > 24 readings. The inlet pressure at 8 cm upstream of the stenosis was set to the same value for all methods.

7 DISCUSSION

7.1 Flow measurement

To avoid long scan time and low signal-to-noise ratio, phase contrast measurements are usually performed at relatively low resolutions (around 1 mm^3). This leads to a loss of flow information because of averaging of the velocity in a voxel, increased intravoxel dephasing, signal loss near vessel edges and inaccurate vessel wall definition as a result of partial volume effects. Furthermore, the phase velocity mapping technique assumes that velocity is constant over the measurement time, which introduces other sources of error to the measurements. Taking all these challenges into account before any clinical application, the accuracy of PC-MRI measurements, needs to be evaluated in phantom models for different pathologic conditions.

This research provides an assessment of the accuracy of PC-MRI flow measurements using an independent modality in a stenotic phantom of common iliac artery under both steady and pulsatile flow conditions, and presents a quantitative comparison between the two modalities. In this study, we have shown that good qualitative and quantitative agreement exists between PC-MRI and SPIV measurements of flow patterns in phantom models of common iliac artery stenosis. Agreement was demonstrated for both steady and pulsatile measurements by evaluating the linear regression between the two methods, which showed a correlation coefficient of > 0.99 and > 0.96 for steady and pulsatile flows, respectively.

The difference between SPIV and PC-MRI measurements for steady flows with proper Venc selection (SF-0 and SF-1) was less than 5% for both inlet and throat and showed good agreement in all cases. The agreement, however, was weaker at the outlet especially for the 87% stenosis. CFD simulations in Figure 7-1 have been used to spatially localize the source of error for steady flow SF-1. Comparison of the velocity contours at the outlet section fails to show any particular pattern explaining the errors associated with the higher flow rate. However, PC-MRI velocity contours at the outlet are noisier compared to the other two sections. This may be explained by the fact that flow distal to the stenosis can be unsteady even in the constant flow cases (see for example (Griffith, Leweke et al., 2008)). Flow measurement by each method assumes constant velocity over a short period of time. In the case of SF-1, this time period was 200 μs for SPIV (time difference between two consecutive images) and 256 ms for PC-MRI (time to collect data for the whole k-space). This longer time duration in PC-MRI leads to a signal dephasing due to more incoherent spin motion in the unsteady regions.

The same mechanism leads to errors in flow measurements of increasing magnitude in the distal section of more severe stenoses where flow unsteadiness is accentuated. Previous studies have shown an overestimation of flow by over 100% immediately upstream and downstream of the stenosis throat (Siegel, Oshinski et al., 1996). However, compared to those studies, a shorter echo time was used here to alleviate some of the problems associated with flow unsteadiness.

The steady flow errors for the SF-2 and SF-3 followed a similar pattern, though the errors were generally larger than for SF-0 and SF-1. This originated from the well-known fact that velocity-to-noise ratio is proportional to Venc (Gudbjartsson and Patz, 1995; Nasiraei-Moghaddam, Behrens et al., 2004) and proper Venc selection in in-vivo studies plays a crucial role in the accuracy of the measurements.

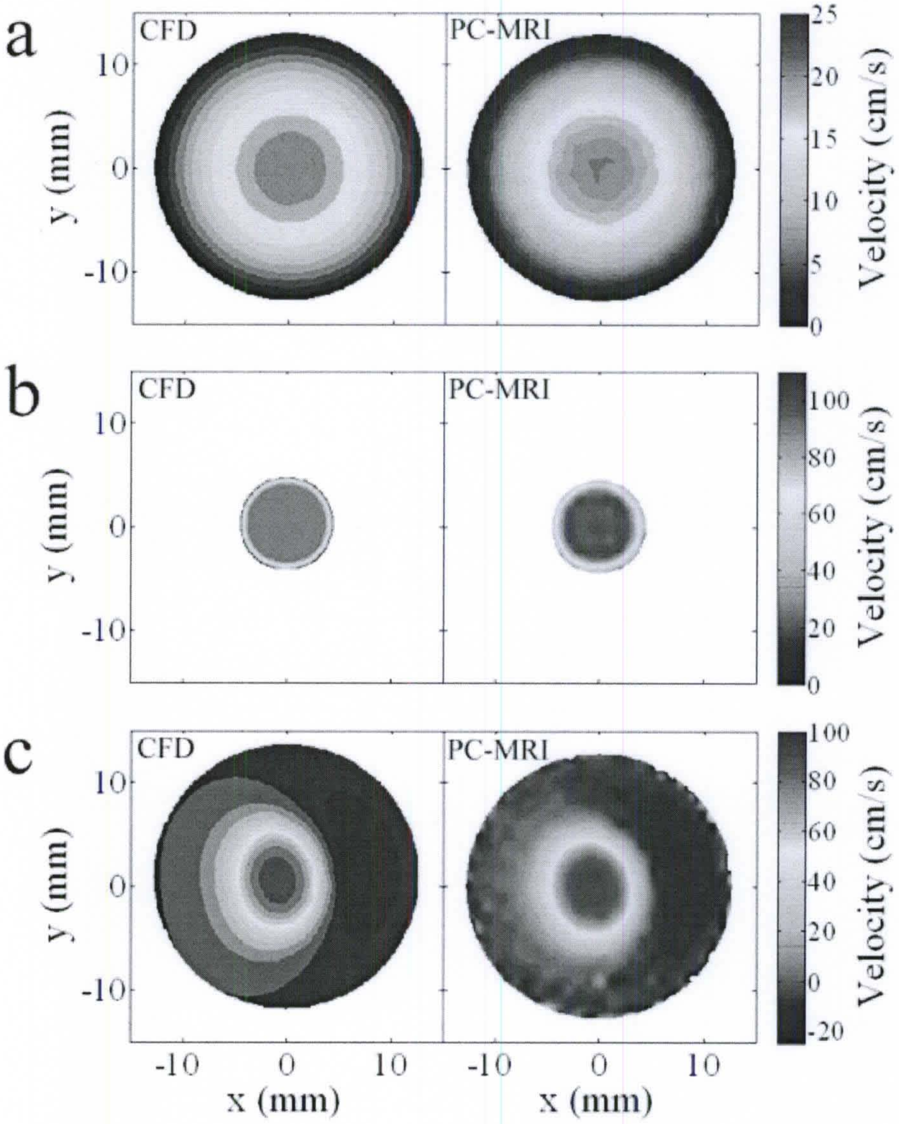


Figure 7-1. Through-plane velocity contours at three sections: a: Inlet, b: Throat and c: Outlet for steady flow SF-1.

The mean and peak flow rate errors, and NRMSE for all pulsatile flow regimes are reported in Table 6-2. For pulsatile flows, mean and peak flow rate errors were lowest at the throat section. This pattern, though less evident, was also observed in NRMSE. NRMSE was higher than error in the mean flow rate. Also, NRMSE for all phantoms, flow regimes and sections was found to be higher compared to steady flow in all cases, suggesting the decreased ability of PC-MRI to detect high frequency components of the flow waveform.

As evidenced by the contour plots and profiles in Fig.5b and c, near-wall velocity at cardiac phases with low flow rates (e.g., t = 275 ms), is noisy, leading to underestimation of the flow for those cardiac phases. This occurred because in pulsatile studies, Venc is generally set corresponding to the peak flow rate and is constant over cardiac phases. MR sequences with the capability to automatically optimize Venc for individual heart phases in pulsatile flow could potentially prevent this underestimation (Ringgaard, Oyre et al., 2004).

7.2 Pressure measurement

Understanding and predicting the hemodynamics of the stenosis has the potential to aid in efficient and safe treatment by offering crucial information for therapeutic decision-making and patient management. Intravascular catheterization is an invasive procedure which provides at best a one-dimensional pressure measurement.

In clinical applications, a modification of Bernoulli's equation is commonly used to estimate the pressure difference across a vessel stenosis or a restricted valve orifice. Despite its wide clinical applications, this estimate depends upon many assumptions about the nature of the flow in stenosis and provides no information about the temporal or spatial variation of the pressure. These limitations produce the motivation to approach the problem through the Navier-Stokes equations, which completely define the behavior of simple fluids.

In this research, two separate methods for integrating the noisy pressure gradients obtained from Navier-Stokes equations are presented. The results of these two methods, i.e. iterative and non-iterative methods, are compared with each other and with other independent methods of pressure measurement.

To better compare the results of iterative and non-iterative methods, these methods have been applied on simulated velocity data (Figure 7-2). Based on the results on steady flow SF-2, the iterative method as shown in Figure 7-2(a) and Table 6-4, produces robust results with a relatively constant accuracy when applied to velocity fields with different resolutions. Some studies such as (Nasiraei-Moghaddam, Behrens et al., 2004) state that resolution is a significant determinant of accuracy for the iterative technique. However, this discrepancy could be due to the different input source that they have used for pressure calculation. While here the "ideal" CFD velocity data has been used to calculate the pressure, those studies have used the in-vitro PC-MRI data to solve the pressure-Poisson equation. Since the signal-to-noise ratio in PC-MRI is a function of the resolution, their results expressed a mixed effect of both resolution and signal-to-noise ratio rather than the pure effect of the resolution.

On the other hand, based on the results on steady flow SF-2, the accuracy of the non-iterative method depends on the resolution of the velocity field (Figure 7-2 (b) and Table 6-4). As explained in the previous chapters, to avoid the discontinuity at the vessel boundaries, the gradients at the boundaries were extended to a rectangular area outside the vessel. During discretization and digitalization of the boundaries at each resolution, the calculated pressure gradients become noisy. This noise is spread to the whole computational domain after extending the pressure gradients to the area outside the vessel. This resolution-dependency could be the most limiting factor in potential clinical applications for the non-iterative technique.

Further simulations on different flow regimes may be needed to draw a general conclusion regarding the sensitivity of the aforementioned methods to the grid resolutions.

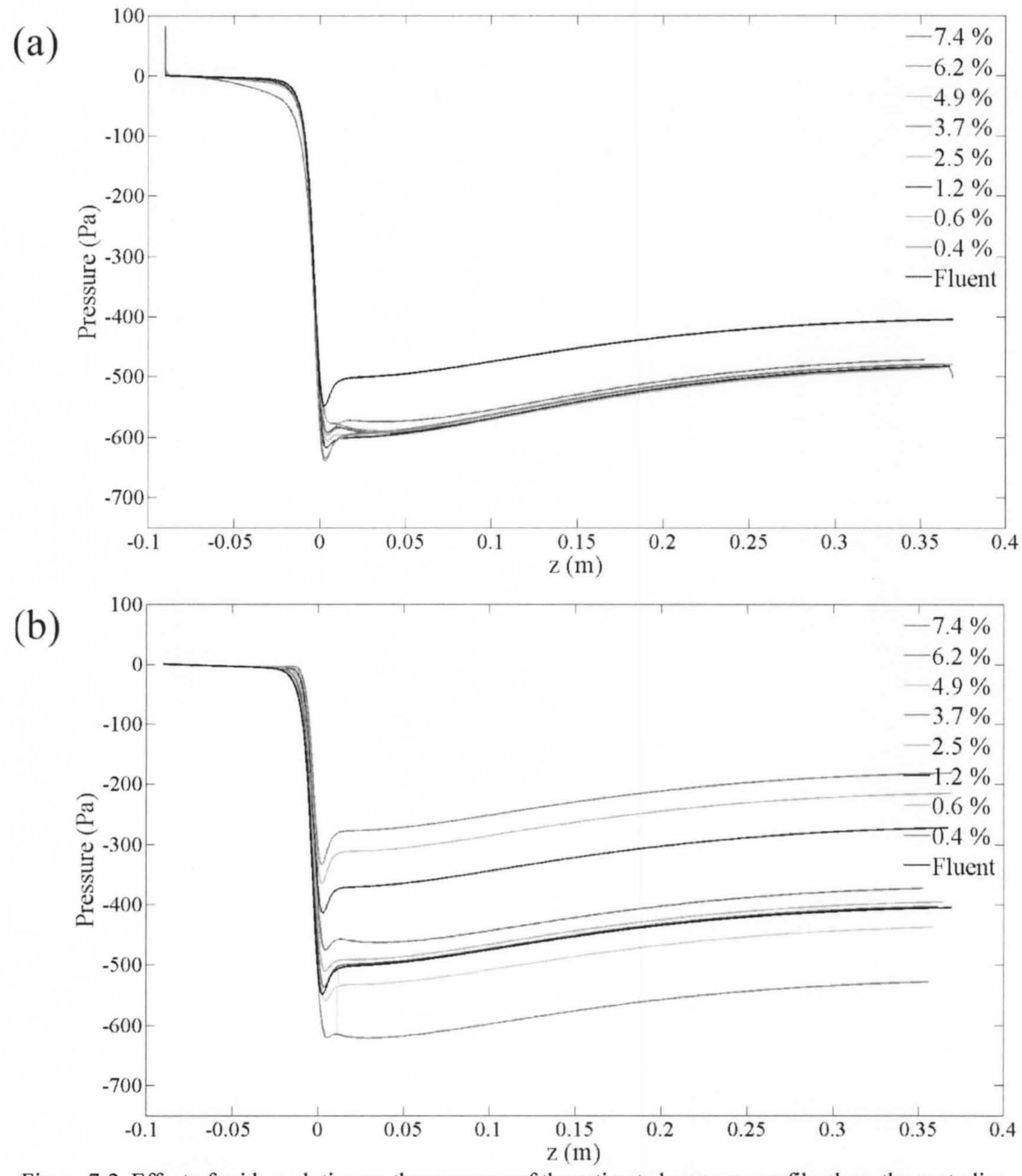


Figure 7-2. Effect of grid resolution on the accuracy of the estimated pressure profile along the centerline for flow regime SF-2. Grid resolution was normalized by the inlet diameter = 25.73 mm. a: Iterative technique. b: Non-iterative technique.

Time wise, as shown in Table 6-4, the non-iterative method, depending on the resolution, is more than one to two orders of magnitude faster than the iterative method, which makes it favorable for clinical usage. Applying the iterative method on three-dimensional, time-resolved (4D) patient velocity data at clinical resolutions of about 1 mm would be on the order of hours, which makes it less interesting for clinical applications.

In the multimodality comparison, a relatively good agreement was found between pressures calculated from different methods. For steady flow, using the pressure obtained from Fluent as the ground truth, the relative error in pressure drop for catheter, PC-MRI (iterative), PC-MRI (non-iterative), CFD (iterative), CFD (non-iterative), SPIV (iterative), and SPIV (non-iterative) methods were 13.1%, 12.5%, 7.8%, 9.3%, 9.0%, 20.5%, and 20.1%, respectively.

For pulsating flow at peak-systole, the relative error in pressure drop for catheter, PC-MRI (iterative), PC-MRI (non-iterative), CFD (iterative), and CFD (non-iterative) methods were 20.3%, 4.0%, 22.1%, 19.8%, and 10.7%, respectively. At end-diastole, the relative error in pressure drop for catheter, PC-MRI (iterative), PC-MRI (non-iterative), CFD (iterative), and CFD (non-iterative) methods were 44.1%, 194.5%, 155.2%, 245.2%, and 162.8%, respectively.

The difference between the pressure from Fluent, on one hand, and iterative/non-iterative techniques on the other hand, could be because of three main factors: First, due to the uniform distribution of the regular grid in iterative/non-iterative techniques, it is not possible to increase the resolution selectively at the stenosis throat. Computationally, it is extremely costly to increase the grid resolution of the entire flow domain to a level comparable to that of Fluent at the throat. Second, unlike Fluent, which uses a finite volume scheme to solve the Navier-Stokes equation, we employed a finite difference algorithm to compute the pressure. Finally, unlike Fluent, which solves the flow equations in 3D space, other computational methods have been solved in 2D axi-symmetric coordinate systems. Although the phantom model was fabricated with much effort to be axi-symmetric, a minimal deviation from axi-symmetry could cause significant changes in flow behavior.

Another point is that the boundary condition for pressure on CFD, PC-MRI and SPIV images is not well defined. However, this issue is not seen when using computational simulation to solve the pressure field since the boundaries are precisely defined (Fujisawa, Tanahashi et al., 2005; Murai, Nakada et al., 2007).

Immediately after the throat of the stenosis, a negative spike could be observed in the pressure calculated by SPIV and CFD data (Figure 6-43). This negative spike, however, was not observed in the pressure profile calculated from PC-MRI data. As shown in Figure 7-2 the pressure profile calculated from CFD data regridded to a coarse mesh fails to detect this negative spike. Likewise, the inability of the PC-MRI to reveal this spike in Figure 6-43 could be due to its lower resolution when compared to SPIV and CFD.

A practical issue in direct pressure measurements is that the pressure measured by catheter suffers from some degree of deviation from the true pressure; this is because first, the catheter disturbs the flow and second, putting the catheter sensor in a cap, which is necessary to minimize dynamic pressure contribution, may disturb the true reading by prolonging the response time of the catheter. This deviation was higher in the pulsatile flow when compared to steady flows. This could be due to the relatively long response time of the catheter (when placed in the cap) in pulsatile flows where the catheter experiences a wide range of pressure over the cardiac cycle.

Finally, it should be noted that although CFD produces more accurate relative pressures in comparison with the iterative and non-iterative solution of PPE on velocity data, it requires larger CPU times (one order of magnitude longer than the iterative technique and three orders of magnitude longer than the non-iterative technique for two-dimensional steady flows). In clinical practice, where the blood flow is pulsatile and three-dimensional, the difference in CPU times is even larger, making CFD less practical.

8 CONCLUSION AND RECOMMENDATIONS

In conclusion, SPIV is a robust and accurate in-vitro method for three-dimensional velocity and flow measurement with the ability to serve as the ground truth for PC-MRI velocity measurements. In particular, the approach taken here could serve as the basis for validation and optimization of new and established MR velocimetry techniques. As shown by measurements and statistical analyses, the flow measurements at different sites in the stenotic model and at different time instants show a good correspondence between standard spin-warp PC-MRI and the reference SPIV method. The results further support the use of PC-MRI in validating hemodynamic information for numerical simulations to diagnose stenotic flow behavior and assist in treatment planning.

The pressure Poisson equation was solved based on an iterative method to estimate the pressure gradient across a stenosis. In general, the noise in the calculated pressure from all velocity-based pressure measurement techniques such as SPIV or PC-MRI is amplified since the pressure Poisson equation contains terms that have spatial derivatives. However, this drawback can be partially overcome by smoothing the data before rendering any pressure calculation algorithm.

In another approach, the Navier-Stokes equation was integrated based on an optimization technique proposed by (Frankot and Chellappa, 1988) and results were compared against previous methods.

The results of this non-iterative technique were in good agreement with other methods (relative error of < 22% for steady flow and pulsatile flow at peak-systole), though the accuracy of this method depended on the resolution of the data. However, being extremely time efficient, this method, if optimized, has the potential to serve in clinical applications.

The current work proposed in this research can be considered as an initial start of a multimodality approach to stenotic flow behavior. Future directions for research include:

1- Using larger coils in PC-MRI experiments to eliminate the errors caused by image registration.

2- Performing 4D PC-MRI imaging to increase the signal-to-noise ratio and using SENSE to reduce the imaging time.

3- Study of the flow at the throat and distal to the stenosis using stereoscopic particle image velocimetry to accurately localize the sources of error in PC-MRI flow measurements.

4- Study of the flow measurement on non-axisymmetric geometries where turbulence distal to the stenosis causes high over/underestimation of the flow.

5- Three dimensional study of the stenotic flow in both SPIV and PC-MRI studies to evaluate the role of possible out of plane components of the velocity which are neglected in this study.

6- Performing a similar study in more severe stenoses and in non-axisymmetric geometries to evaluate the accuracy of the proposed pressure calculation techniques.

7- Simulations of different flow regimes to draw a general conclusion regarding the sensitivity of the iterative and non-iterative techniques to the grid resolutions.

REFERENCES

Adrian, R. J. (2005). Twenty years of particle image velocimetry. *Experiments in Fluids* 39(2): 159-169.

Allison, M. A., E. Ho, J. O. Denenberg, R. D. Langer, A. B. Newman, R. R. Fabsitz and M. H. Criqui (2007). Ethnic-specific prevalence of peripheral arterial disease in the United States. *Am J Prev Med* 32(4): 328-333.

Brossard, C., J. C. Monnier, P. Barricau, F. X. Vandernoot, Y. Le Sant, F. Champagnat and L. B. G. (2009) Principles and Applications of Particle Image Velocimetry. *Optical Diagnostics of Flows*, 1-11.

Brunette, J., R. Mongrain, J. Laurier, R. Galaz and J. C. Tardif (2008). 3D flow study in a mildly stenotic coronary artery phantom using a whole volume PIV method. *Med Eng Phys* 30(9): 1193-1200.

Cebral, J. R., C. M. Putman, M. T. Alley, T. Hope, R. Bammer and F. Calamante (2009). Hemodynamics in Normal Cerebral Arteries: Qualitative Comparison of 4D Phase-Contrast Magnetic Resonance and Image-Based Computational Fluid Dynamics. *J Eng Math* 64(4): 367-378.

Chen, Z., T. E. Milner, S. Srinivas, X. Wang, A. Malekafzali, M. J. van Gemert and J. S. Nelson (1997). Noninvasive imaging of in vivo blood flow velocity using optical Doppler tomography. *Optics letters* 22(14): 1119-1121.

Emrich, R. J. (1981). *Fluid Dynamics*, Academic Press.

Faxon, D. P., V. Fuster, P. Libby, J. A. Beckman, W. R. Hiatt, R. W. Thompson, J. N. Topper, B. H. Annex, J. H. Rundback, R. P. Fabunmi, R. M. Robertson and J. Loscalzo (2004). Atherosclerotic Vascular Disease Conference: Writing Group III: pathophysiology. *Circulation* 109(21): 2617-2625.

Firmin, D. N., G. L. Nayler, P. J. Kilner and D. B. Longmore (1990). The application of phase shifts in NMR for flow measurement. *Magn Reson Med* 14(2): 230-241.

Frankot, R. T. and R. Chellappa (1988). A method for enforcing integrability in shape from shading algorithms. *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 10(4): 439-451.

Frayne, R., D. W. Holdsworth, L. M. Gowman, D. W. Rickey, M. Drangova, A. Fenster and B. K. Rutt (1992). Computer-controlled flow simulator for MR flow studies. *J Magn Reson Imaging* 2(5): 605-612.

Frydrychowicz, A., A. F. Stalder, M. F. Russe, J. Bock, S. Bauer, A. Harloff, A. Berger, M. Langer, J. Hennig and M. Markl (2009). Three-dimensional analysis of segmental wall shear stress in the aorta by flow-sensitive four-dimensional-MRI. *J Magn Reson Imaging* 30(1): 77-84.

Fujisawa, N., S. Tanahashi and K. Srinivas (2005). Evaluation of pressure field and fluid forces on a circular cylinder with and without rotational oscillation using velocity data from PIV measurement. *Measurement Science and Technology* 16(4): 989.

Griffith, M. D., T. Leweke, M. C. Thompson and K. Hourigan (2008). Steady inlet flow in stenotic geometries: convective and absolute instabilities. *Journal of Fluid Mechanics* 616: 111-133.

Gudbjartsson, H. and S. Patz (1995). The Rician distribution of noisy MRI data. *Magn Reson Med* 34(6): 910-914.

Haacke, E. M. (1999). *Magnetic resonance imaging: physical principles and sequence design*, J. Wiley & Sons.

Hangiandreou, N. J., P. J. Rossman and S. J. Riederer (1993). Analysis of MR phase-contrast measurements of pulsatile velocity waveforms. *J Magn Reson Imaging* 3(2): 387-394.

Harloff, A., A. Nussbaumer, S. Bauer, A. F. Stalder, A. Frydrychowicz, C. Weiller, J. Hennig and M. Markl (2010). In vivo assessment of wall shear stress in the atherosclerotic aorta using flow-sensitive 4D MRI. *Magn Reson Med* 63(6): 1529-1536.

Hirsch, A. T., M. H. Criqui, D. Treat-Jacobson, J. G. Regensteiner, M. A. Creager, J. W. Olin, S. H. Krook, D. B. Hunninghake, A. J. Comerota, M. E. Walsh, M. M. McDermott and W. R. Hiatt (2001). Peripheral arterial disease detection, awareness, and treatment in primary care. *Jama* 286(11): 1317-1324.

Hirsch, A. T., Z. J. Haskal, N. R. Hertzer, C. W. Bakal, M. A. Creager, J. L. Halperin, L. F. Hiratzka, W. R. Murphy, J. W. Olin, J. B. Puschett, K. A. Rosenfield, D. Sacks, J. C. Stanley, L. M. Taylor, Jr., C. J. White, J. White, R. A. White, E. M. Antman, S. C. Smith, Jr., C. D. Adams, J. L. Anderson, D. P. Faxon, V. Fuster, R. J. Gibbons, S. A. Hunt, A. K. Jacobs, R. Nishimura, J. P. Ornato, R. L. Page and B. Riegel (2006). ACC/AHA 2005 Practice Guidelines for the management of patients with peripheral arterial disease (lower extremity, renal, mesenteric, and abdominal aortic): a collaborative report from the American Association for

Vascular Surgery/Society for Vascular Surgery, Society for Cardiovascular Angiography and Interventions, Society for Vascular Medicine and Biology, Society of Interventional Radiology, and the ACC/AHA Task Force on Practice Guidelines (Writing Committee to Develop Guidelines for the Management of Patients With Peripheral Arterial Disease): endorsed by the American Association of Cardiovascular and Pulmonary Rehabilitation; National Heart, Lung, and Blood Institute; Society for Vascular Nursing; TransAtlantic Inter-Society Consensus; and Vascular Disease Foundation. *Circulation* 113(11): e463-654.

Holdsworth, D. W., D. W. Rickey, M. Drangova, D. J. Miller and A. Fenster (1991). Computer-controlled positive displacement pump for physiological flow simulation. *Med Biol Eng Comput* 29(6): 565-570.

Hornak, J. P. (2008). The Basics of MRI.

Huang, H., D. Dabiri and M. Gharib (1997). On errors of digital particle image velocimetry. *Measurement Science & Technology* 8(12): 1427-1440.

Kasapis, C. and H. S. Gurm (2009). Current approach to the diagnosis and treatment of femoral-popliteal arterial disease. A systematic review. *Curr Cardiol Rev* 5(4): 296-311.

Khodarahmi, I., M. Shakeri, M. Sharp and A. A. Amini (2010). Using PIV to determine relative pressures in a stenotic phantom under steady flow based on the Pressure-Poisson equation. *Conf Proc IEEE Eng Med Biol Soc* 1: 2594-2597.

Kinney, T. B. and S. C. Rose (1996). Intraarterial pressure measurements during angiographic evaluation of peripheral vascular disease: techniques, interpretation, applications, and limitations. *AJR Am J Roentgenol* 166(2): 277-284.

Leng, G. C., A. J. Lee, F. G. Fowkes, M. Whiteman, J. Dunbar, E. Housley and C. V. Ruckley (1996). Incidence, natural history and cardiovascular events in symptomatic and asymptomatic peripheral arterial disease in the general population. *Int J Epidemiol* 25(6): 1172-1181.

Lotz, J., C. Meier, A. Leppert and M. Galanski (2002). Cardiovascular flow measurement with phase-contrast MR imaging: basic facts and implementation. *Radiographics* 22(3): 651-671.

Lum, D. P., K. M. Johnson, R. K. Paul, A. S. Turk, D. W. Consigny, J. R. Grinde, C. A. Mistretta and T. M. Grist (2007). Transstenotic pressure gradients: measurement in swine--retrospectively ECG-gated 3D phase-contrast MR angiography versus endovascular pressure-sensing guidewires. *Radiology* 245(3): 751-760.

Marshall, I., S. Zhao, P. Papathanasopoulou, P. Hoskins and Y. Xu (2004). MRI and CFD studies of pulsatile flow in healthy and stenosed carotid bifurcation models. *J Biomech* 37(5): 679-687.

Massey, B. S., J. Ward-Smith and A. J. Ward-Smith (2006). *Mechanics of fluids*, Taylor & Francis.

McDermott, M. M., S. Mehta and P. Greenland (1999). Exertional leg symptoms other than intermittent claudication are common in peripheral arterial disease. *Arch Intern Med* 159(4): 387-392.

Moon-Ho Song, S., S. Napel, N. J. Pelc and G. H. Glover (1995). Phase unwrapping of MR phase images using Poisson equation. *IEEE Trans Image Process* 4(5): 667-676.

Murai, Y., T. Nakada, T. Suzuki and F. Yamamoto (2007). Particle tracking velocimetry applied to estimate the pressure field around a Savonius turbine. *Measurement Science and Technology* 18(8): 2491.

Nasiraei-Moghaddam, A., G. Behrens, N. Fatouraee, R. Agarwal, E. T. Choi and A. A. Amini (2004). Factors affecting the accuracy of pressure measurements in vascular stenoses from phase-contrast MRI. *Magn Reson Med* 52(2): 300-309.

Nayak, K. S., B. S. Hu and D. G. Nishimura (2003). Rapid quantitation of high-speed flow jets. *Magn Reson Med* 50(2): 366-372.

Niederer, P. F. (2010). Ultrasound imaging and Doppler flow velocity measurement. *Technology and health care : official journal of the European Society for Engineering and Medicine* 18(3): 245-265.

Pelc, N. J., M. A. Bernstein, A. Shimakawa and G. H. Glover (1991). Encoding strategies for three-direction phase-contrast MR imaging of flow. *J Magn Reson Imaging* 1(4): 405-413.

Pelc, N. J., F. G. Sommer, K. C. Li, T. J. Brosnan, R. J. Herfkens and D. R. Enzmann (1994). Quantitative magnetic resonance flow imaging. *Magn Reson Q* 10(3): 125-147.

Potter, M. C. and D. C. Wiggert (2009). *Mechanics of Fluids, SI Edition*, Cengage Learning.

Prasad, A. K. (2000). Stereoscopic particle image velocimetry. *Experiments in Fluids* 29(2): 103-116.

Rajan, V., B. Varghese, T. G. van Leeuwen and W. Steenbergen (2009). Review of methodological developments in laser Doppler flowmetry. *Lasers in medical science* 24(2): 269-283.

Ringgaard, S., S. A. Oyre and E. M. Pedersen (2004). Arterial MR imaging phase-contrast flow measurement: improvements with varying velocity sensitivity during cardiac cycle. *Radiology* 232(1): 289-294.

Shakeri, M., I. Khodarahmi, M. K. Sharp and A. A. Amini (2010). Optical imaging of steady flow in a phantom model of iliac artery stenosis: comparison of CFD simulations with PIV measurements. *Proc. SPIE.* San Diego, California, USA, The International Society for Optical Engineering. 76269, (76260L).

Shames, I. H. (2003). *Mechanics of fluids*, McGraw-Hill.

Siegel, J. M., Jr., J. N. Oshinski, R. I. Pettigrew and D. N. Ku (1996). The accuracy of magnetic resonance phase velocity measurements in stenotic flow. *J Biomech* 29(12): 1665-1672.

Song, S. M., R. M. Leahy, D. P. Boyd, B. H. Brundage and S. Napel (1994). Determining cardiac velocity fields and intraventricular pressure distribution from a sequence of ultrafast CT cardiac images. *IEEE Trans Med Imaging* 13(2): 386-397.

Srichai, M. B., R. P. Lim, S. Wong and V. S. Lee (2009). Cardiovascular applications of phase-contrast MRI. *AJR Am J Roentgenol* 192(3): 662-675.

Szolar, D. H., H. Sakuma and C. B. Higgins (1996). Cardiovascular applications of magnetic resonance flow and velocity measurements. *J Magn Reson Imaging* 6(1): 78-89.

Tetteroo, E., A. D. van Engelen, J. H. Spithoven, A. V. Tielbeek, Y. van der Graaf and W. P. Mali (1996). Stent placement after iliac angioplasty: comparison of hemodynamic and angiographic criteria. Dutch Iliac Stent Trial Study Group. *Radiology* 201(1): 155-159.

Thompson, R. B. and E. R. McVeigh (2003). Fast measurement of intracardiac pressure differences with 2D breath-hold phase-contrast MRI. *Magn Reson Med* 49(6): 1056-1066.

Tyszka, J. M., D. H. Laidlaw, J. W. Asa and J. M. Silverman (2000). Three-dimensional, time-resolved (4D) relative pressure mapping using magnetic resonance imaging. *J Magn Reson Imaging* 12(2): 321-329.

Udoff, E. J., K. H. Barth, D. P. Harrington, S. L. Kaufman and R. I. White (1979). Hemodynamic significance of iliac artery stenosis: pressure measurements during angiography. *Radiology* 132(2): 289-293.

van Doorne, C. and J. Westerweel (2007). Measurement of laminar, transitional and turbulent pipe flow using Stereoscopic-PIV. *Experiments in Fluids* 42(2): 259-279.

Wang, Y. and A. A. Amini (2005). Integrable Pressure Gradients via Harmonics-Based Orthogonal Projection. *Information Processing in Medical Imaging*. G. Christensen and M. Sonka, Springer Berlin / Heidelberg. 3565: 431-442.

Yang, G. Z., P. J. Kilner, N. B. Wood, S. R. Underwood and D. N. Firmin (1996). Computation of flow pressure fields from magnetic resonance velocity mapping. *Magn Reson Med* 36(4): 520-526.

APPENDICES

Appendix A: C++ code for iterative technique

```
?/*********************************************************************************
// Pressure_Poisson.cpp : Defines the entry point for the console application.
// Developed by Iman Khodarahmi in C++/Windows environment from 2008-2012
//*********************************************************************************


#include "stdafx.h"
#include <iostream>
#include <fstream>
#include <cstring>
#include <cmath>
#include <iomanip>
#include <sstream>
#include <string>
#include <windows.h>
#include <conio.h>
using namespace std;
//--------constant (Physical or mathematical) parameters -------------------
const double PI = 3.141592653589793238;
const double RHO = 1600;
const double INITPRES = 300;
const double MU = 0.02;
const double fr = 0;
const double fz = 0;
//-------------------------Problem specific constants------
const int MAXSLICE = 4400; //270;
const int RESOLUTION = 1300;
const int MAXPOINTS = 9000; //550
const int ponum = 8;
const double SIGMA2 = 1.0;         // used in covariance calculation.
const int HUGEIG = 100000; // more than any probable eigenvalue
const int NR_END = 1;
const double PointEr = 5e-10;      // acceptable Average Error in each point.
const double epsilon = 0.0001;    // this value should be less than RCMS and RCMZ
const double hugdis = 10000;      // this value should be greater than system
dimensions
#define FREE_ARG char*
#define sqr(k) ((k) * (k))
template <class Any>
inline void SWAP (Any &a, Any &b)
{
	Any temp = a;
	a = b;
	b =temp;}
```

```
struct point
{
        double r;
        double z;};

//--------------------GLOBAL VARIABLES-----------------------------------------
------------------
point vfilt[RESOLUTION][MAXSLICE];
point v[RESOLUTION][MAXSLICE];
point v_bak[RESOLUTION][MAXSLICE];
point v_bak1[RESOLUTION][MAXSLICE];
unsigned short bound[RESOLUTION][MAXSLICE];
point contour[MAXPOINTS];
point n[RESOLUTION][MAXSLICE];
double vdiv[RESOLUTION][MAXSLICE];
point vdelv[RESOLUTION][MAXSLICE];
point lapv[RESOLUTION][MAXSLICE];
point Pgr[RESOLUTION][MAXSLICE];
double dpn[RESOLUTION][MAXSLICE];
double Pc[RESOLUTION][MAXSLICE];
double Plap[RESOLUTION][MAXSLICE];
point Pcgr[RESOLUTION][MAXSLICE];
//---------------------------------------------------------------------------
--------------------
void tecread(int* j00,int* imax, const char* filename, int* nslice, double RCMS,
double ZCMS);
double gasdev(int*);
void nrerror(char error_text[]);
int
find_nclosest(point[],int,int,int,double[],int,double[],int,double,double,int);
void Fitlane(double*,double*,double[],int,double[],int,int);
double distance(double,double,double,double);
double innerprod(double,double,double,double,double,double);
int findxmaller(double[],int);
void filter(point[][MAXSLICE],int,int,double,double,double,double);
void diver(point[][MAXSLICE], double[][MAXSLICE],int,int,double,double);
void cal_lapv(int i,int j,int nslice, double RCMS, double ZCMS);
void cal_vdelv(int i,int j,int nslice, double RCMS, double ZCMS);
void gradientns(int i,int j);
void estim_b_gradient(int,int,int,double*,double*);
double inner_prod(point[][MAXSLICE],point[][MAXSLICE],int,int);
void ludcmp(double[][7],int,int*,double*);
void lubksb(double[][7],int, int*, double[]);
void gradient(double [][MAXSLICE], point [][MAXSLICE],int,int,double,double);
//----------------------------------------------------------------
int main(int argc,char **argv)
{
        ifstream fin;
        ofstream fout;
        double
RCMS,ZCMS,Vencr,Vencz,phasenoise,gsigr,gsigz,gsigvdvz,noisPowz,noisPowr,noise,a,b,
r[MAXSLICE],z[MAXSLICE],flow[MAXSLICE];
        double
sig1,sig2,tErr,Err,tErz,Erz,temp,cnt,gradmag,Lambda1,Lambda2,ar,az,lError,FError,E
rror,lFError,newPc,om1,om2;
        int i,j,j00,imax,nslice,length,nupo,ii,jj,cnt1,loop,j1,j2;
        double Pgrinit[MAXSLICE],third,gz,gr,meanPc,temp2;
        string str, fname, path;
```

```
       fname = argv[1];
       const char* ffname = fname.c_str();
       RCMS = atof(argv[2]);
       ZCMS = atof(argv[3]);
       Vencr = atof(argv[4]);
       Vencz = atof(argv[5]);
       phasenoise = atof(argv[6]);
       gsigr = atof(argv[7]);
       gsigz = atof(argv[8]);
       gsigvdvz = atof(argv[9]);
       if (phasenoise > 0.2 | phasenoise < 0.0)
       {
              phasenoise = 0;
           cout<< "\n phasenoise can't be out of [0 0.2]. It is zero now \nPress
any key to continue";}
       if (gsigr > 0.10 | gsigr < 0.001)
       {
           gsigr = 0.001;
           cout<<"\ngsigr can't be out of [0.001 0.1]. It is 0.001 now \nPress any
key to continue";}
       if (gsigz > 0.4 | gsigz < 0.001)
       {
           gsigz = 0.001;
           cout<<"\n gsigz can't be out of [0.001 0.4]. It is 0.001 now \nPress
any key to continue";}
       if (gsigvdvz > 0.2 | gsigvdvz < 0.001)
       {
           gsigvdvz = 0.001;
           cout<<"\n gsigvdvz can't be out of [0.001 0.2]. It is 0.001 now \npress
any key to continue";}
       tecread(&j00, &imax, ffname, &nslice, RCMS, ZCMS);
       noisPowz = Vencz*phasenoise/PI;
       noisPowr = Vencr*phasenoise/PI;
    for(i=0; i< imax +1;i++)
       {
              bound[i][0] = bound[i][1];
              v[i][0].r = 0;
              v[i][0].z = v[i][1].z;
              bound[i][nslice+1] = bound[i][nslice];
              v[i][nslice+1].r = 0;
              v[i][nslice+1].z = v[i][nslice].z;}
       int neg_intz = -5000; /// negative integer input to gasdev funtion for z
direction
       int neg_intr = -2000; /// negative integer input to gasdev funtion for r
direction
       for (j=0; j<nslice+2 ;j++)
       {
              for (i =0; i < imax+1; ++i)
              {
                     v_bak[i][j].r = v[i][j].r;
                     v_bak[i][j].z = v[i][j].z;
                     noise=noisPowz*gasdev(&neg_intz);
                     v[i][j].z +=noise;
                     v[i][j].r += noisPowr*gasdev(&neg_intr);}
              v[0][j].r = -v[2][j].r;
              v[0][j].z = v[2][j].z;
              v[1][j].r = 0;
              bound[0][j] = bound[2][j];
              bound[1][j] = 0;}
       length=0;
    for (j =1; j < nslice+1; ++j){
              if (bound[imax][j]==1 && bound[imax-1][j]==0){
                    cout<<"\nwarning for small imax   j=" <<j;
                    cin.get();
                    for (i =2; i < imax; ++i){
                           if (bound[i][j]==0 &&( bound[i-1][j]==1 |
bound[i+1][j]==1 | bound[i][j-1]==1 | bound[i][j+1]==1)){
                                  bound[i][j] = 2;
                                  length++;
                                  contour[length].r = (i - 1) * RCMS;
                                  contour[length].z = j * ZCMS;}}}
              else          {
                    for (i =1; i < imax; ++i){
                           if (bound[i][j]==0 &&( bound[i-1][j]==1 |
bound[i+1][j]==1 | bound[i][j-1]==1 | bound[i][j+1]==1)){
                                  bound[i][j] = 2;
                                  length++;
                                  contour[length].r = (i - 1) * RCMS; //WHY i-1?
                                  contour[length].z = j * ZCMS;}}}}
       for (i = 1; i < length+1; ++i){
              nupo = find_nclosest(contour,MAXPOINTS ,nslice
,i,r,MAXPOINTS,z,MAXPOINTS,RCMS,ZCMS,length);
              ii = (int)(floor)(0.5 + contour[i].r / RCMS) +1;//WHY now correcting
it?
              jj = (int)(floor)(0.5 + contour[i].z / ZCMS);
              Fitlane(&a,&b,r,MAXPOINTS,z,MAXPOINTS,nupo);
              if(bound[ii+(int)(2*a)][jj+(int)(2*b)]== 0 )
////////////////////////////// This 2 is completely dependent upon geometry
!!!!!!!!!!
              {a = -a; b= -b;}
              n[ii][jj].r = (double)(a);  n[ii][jj].z = (double)(b);}
    for (i =0; i <= imax; ++i)    {
              n[i][0].r = n[i][1].r;
              n[i][0].z = n[i][1].z;
              n[i][nslice+1].r = n[i][nslice].r;
              n[i][nslice+1].z = n[i][nslice].z;}
    for (i =1; i < imax; ++i){
              if (bound[i][0]==0 &&( bound[i-1][0]+bound[i+1][0]==1 ))
bound[i][0] = 2;
              if (bound[i][nslice+1]==0 &&( bound[i-
1][nslice+1]+bound[i+1][nslice+1]==1 ))  bound[i][nslice+1] = 2;}
       for (j =0; j < nslice+2; ++j){
              v[0][j].z = .5*(v[0][j].z+v[2][j].z); v[2][j].z = v[0][j].z;
              v[1][j].r = 0;v[0][j].r = .5*(v[0][j].r-v[2][j].r);    v[2][j].r = -
v[0][j].r;
           bound[0][j] = bound[2][j];
           n[0][j].r = -n[2][j].r;
           n[0][j].z = n[2][j].z;}
       for (j=0; j< nslice+1;j++){
              flow[j] = v[1][j].z*sqr(RCMS)*PI/4; //for a srip with width RCMS
around row i, velocity is considered v[i], i=1 being the symmetry axis.
              for (i=2;i< imax+1;i++){
                    flow[j] += v[i][j].z*2*PI*(i-1)*sqr(RCMS); }}
       for (j=0; j<nslice+2 ;j++)
       {for (i =0; i < imax+1; ++i){
                    v_bak1[i][j].r=v[i][j].r;
```

```
                        v_bak1[i][j].z=v[i][j].z;}}
        sig1 = gsigr; sig2 = gsigz;
        filter(v,nslice,imax, sig1,sig2,RCMS,ZCMS);
        diver(v,vdiv,nslice,imax,RCMS,ZCMS);
cnt1 = 0;
        for(j = 1; j < nslice+1; ++j) {
                for (i = 1; i < imax+1; ++i){
                        if (bound[i][j] == 0)                   {
                                cal_vdelv(i,j,nslice,RCMS,ZCMS);
                                cal_lapv(i,j,nslice,RCMS,ZCMS);}}
            vdelv[0][j].r = -vdelv[2][j].r;
            vdelv[0][j].z = vdelv[2][j].z;
            lapv[0][j].r = -lapv[2][j].r;
            lapv[0][j].z = lapv[2][j].z;}
        sig1 = 0.0; //gsigr;
        sig2 = gsigvdvz;
        filter(vdelv,nslice,imax, sig1,sig2,RCMS,ZCMS);
        for(j = 1; j < nslice+1; ++j)      // calculated values for first and last
slices can NOT be accurate. to be calculated later.
        {         for (i = 1; i < imax+1; ++i){
                  if (bound[i][j] == 0){
                          if(j>1 & j< nslice)cnt1++;
                          gradientns(i,j);}}
           Pgr[0][j].r = -Pgr[2][j].r;
           Pgr[0][j].z = Pgr[2][j].z;}
        tErr = 0; tErz = 0; cnt1 = 0;
        for(j = 2; j < nslice; ++j)
        {       for (i =1; i < imax+1; ++i)
                {if (bound[i][j] == 2){
                                cnt1++;
                                estim_b_gradient(i,j,nslice,&Err,&Erz);
                                tErr += Err; tErz += Erz;
                                dpn[i][j] = inner_prod(Pgr,n,i,j);}}
                Pgr[0][j].r = -Pgr[2][j].r;
            Pgr[0][j].z = Pgr[2][j].z;
            dpn[0][j] = dpn[2][j];}
        for (j=1;j<nslice+1;j++){
                temp = 0; cnt = 0;
                for (i =1; i <= imax; ++i){
                        if (bound[i][j] + bound[i+1][j] + bound[i-1][j] == 0){
                                temp += Pgr[i][j].z;
                                cnt ++;}             }
                if (cnt != 0) Pgrinit[j] = 1.2*temp/cnt;else Pgrinit[j] = 0;}
        temp=0;       cnt=0; i=0;
        do {//averaging Pgr.z for j=4 according to j=3,4,5
                i++;
                temp +=(Pgr[i][5].z + Pgr[i][4].z + Pgr[i][3].z);
                cnt += 3;
        }while(bound[i][4]==0);
        temp /= cnt;
        for (i=0 ; i < imax-1 ; i++)
        {             if(bound[i][4]==0)   Pgr[i][4].z=temp;}
        temp=0;       cnt=0; i=0;
        do {//averaging Pgr.z for j=nslice-3 according to j=nslice-2, nslice-3,
nslice-4
                i++;
                temp +=(Pgr[i][nslice-4].z + Pgr[i][nslice-3].z + Pgr[i][nslice-
2].z);
```

```
                cnt += 3;
        }while(bound[i][nslice-3]==0);
        temp /= cnt;
        for (i=0 ; i < imax-1 ; i++)
        {               if(bound[i][nslice-3]==0)   Pgr[i][nslice-3].z=temp;}
for(i=1; i <= imax; ++i){               Pgr[i][1].z = 0;
                if(bound[i][1]!=1){
                        Pgr[i][1].z = 2*Pgr[i][3].z - Pgr[i][5].z;
                        if(bound[i][1]==2)
                                dpn[i][1] = dpn[i][2];}
                Pgr[i][nslice].z = 0;
                if(bound[i][nslice]!=1)
                {                       Pgr[i][nslice].z = 2*Pgr[i][nslice-2].z -
Pgr[i][nslice-4].z;
                        if(bound[i][nslice]==2)
                                dpn[i][nslice] = dpn[i][nslice-1];}       }
        for (i =1; i <= imax; ++i)
        {               Pgr[i][1].r = 0;
                if(bound[i][1]!=1 & bound[i+1][1]!=1 & bound[i-1][1]!=1){
         //      Pgr[i][1].r = Pgr[i][2].r +(Pgr[i-1][1].z+Pgr[i-1][2].z-
Pgr[i+1][1].z-Pgr[i+1][2].z)*ZCMS/RCMS/4;//curl free
                        Pgr[i][1].r = Pgr[i][2].r; }
                Pgr[i][nslice].r = 0;
                if(bound[i][nslice]!=1 &bound[i-1][nslice]!=1
&bound[i+1][nslice]!=1)
                {       //      Pgr[i][nslice].r = Pgr[i][nslice-1].r +(Pgr[i-
1][nslice].z+Pgr[i-1][nslice-1].z-Pgr[i+1][nslice].z-Pgr[i+1][nslice-
1].z)*ZCMS/RCMS/4;
                        Pgr[i][nslice].r = Pgr[i][nslice-1].r;}}
Pc[1][1]= INITPRES+4;                   ///4
        for(j = 2; j < nslice+1; ++j)
        {               Pc[1][j]= Pc[1][j-1]+(Pgrinit[j-1]+Pgrinit[j])*ZCMS/2;}
        for(j = 1; j < nslice+1; ++j)
        {               for (i = 2; i < imax+1; ++i){
                        if (bound[i][j] != 1 & j>3 & j<nslice-1)
                        {                               Pc[i][j] = Pc[1][j];
//                     Pc[i][j] = Pc[i-1][j]+(Pgr[i-1][j].r+Pgr[i][j].r)*RCMS/2;
                        }else Pc[i][j] = INITPRES;}
                Pc[0][j]=Pc[2][j];}
        for (i = 0; i < imax+1; ++i)
        {               if (bound[i][0]==0)
                        Pc[i][0]= INITPRES+4;
                else                    Pc[i][0]= INITPRES;
                if (bound[i][nslice+1]==0)
                        Pc[i][nslice+1]= INITPRES+4;
                else                    Pc[i][nslice+1]= INITPRES;
                if (bound[i][0] != 1)
                        Pc[i][0]= Pc[i][3]-Pgr[i][2].z*3*ZCMS;
                if (bound[i][nslice+1] != 1)
                        Pc[i][nslice+1]= Pc[i][nslice-2]+Pgr[i][nslice-1].z*3*ZCMS;
                for(j = 0; j < nslice+2; ++j)
                {                       if(bound[i][j]==1)
                                Pc[i][j]=INITPRES;}}
diver(Pgr,Plap,nslice,imax,RCMS,ZCMS);          // Plap is not calculated for
first and last slices(i.e.#1 & #nslice)
                        gradient(Pc , Pcgr ,nslice,imax,RCMS,ZCMS);
                gradmag = 0;
        for (j=4;j < nslice-2;++j)
```

```
       {             for (i =1; i < imax+1; ++i){
                     if (bound[i][j] != 1)
                     {                            gradmag += (sqr(Pgr[i][j].r) +
sqr(Pgr[i][j].z))*(i-1)*RCMS;
                            cnt += i-1;}         }}
       gradmag /= cnt;            //gradmag= sqrt(gradmag);
       om1 = 1.9; om2 = 1.4;
       Lambda1 = (2/sqr(RCMS) + 15/sqr(ZCMS)/6);
       Lambda2 = (2/sqr(RCMS) + 2/sqr(ZCMS));
       ar=0; az=0;
       loop=0; Error = 0;
       lError = 0; lFError = 1; FError = 1;
              do {
              loop ++;
           if (lFError > FError) lFError = FError;
           lError = Error;
           Error = 0;              cnt = 0;
           for (i =1; i < imax+1; ++i)
              {                    if( bound[i][2] != 1){
                            newPc = Pc[i][6]-Pgr[i][4].z*4*ZCMS;//first order
difference at 4 but why using
Pgr[i][4]??????????????????????????????????????????????????????????????
                            if (loop > 4000 & fabs(newPc-Pc[i][2])>.05)
                            {                                    cout<< " Error= "<<
Error << " change= "<<      newPc-Pc[i][2] << " new= "<<newPc;}
                            Error += fabs(newPc-Pc[i][2]);
                            temp += om2*(newPc-Pc[i][2]);
                            cnt++;
                            Pc[i][2] = om2*newPc + (1-om2)*Pc[i][2];}
                                          if( bound[i][3] != 1)
                     {                            newPc = (8*Pc[i][5]-Pc[i][6] +
Pc[i][2] - 12*ZCMS*Pgr[i][4].z)/8; //centered difference approximation with fourth
order error
                            Error += fabs(newPc-Pc[i][3]);
                            Pc[i][3] = om2*newPc + (1-om2)*Pc[i][3];}       }
              if (cnt != 0) temp /= cnt; else temp = 0;
              for (i = 1; i < imax+1; ++i)
              {                    for (j = 2;j < nslice;++j){
                            if( bound[i][j] != 1) Pc[i][j] -= temp;}}
              Pc[0][2]=Pc[2][2];
              Pc[0][3]=Pc[2][3];
              cnt = 0;
           for (i =1; i < imax+1; ++i)
              {                    if( bound[i][nslice-1] != 1){
                            newPc = Pc[i][nslice-5] + Pgr[i][nslice-3].z*4*ZCMS;
                            Error += fabs(newPc-Pc[i][nslice-1]);
                            if (Error/(imax*nslice) > 0.4){
                                   cout<<i<<",nslice-1"<<" Error= "<<Error<<"
change= "<<newPc-Pc[i][nslice-1]<<"       new= "<<newPc;             }
                            Pc[i][nslice-1] = om2*newPc + (1-om2)*Pc[i][nslice-1];}
                 if( bound[i][nslice-2] != 1)
                 {                    newPc = (8*Pc[i][nslice-4]-Pc[i][nslice-5] +
Pc[i][nslice-1] + 12*ZCMS*Pgr[i][nslice-3].z)/8;
                        Error += fabs(newPc-Pc[i][nslice-2]);
                        if (Error/(imax*nslice) > 0.5){
                               cout<<i<<",nslice-2"<<" Error= "<<Error<<" change=
"<<newPc-Pc[i][nslice-2]<<" new= "<<newPc;}
```

```
                        Pc[i][nslice-2] = om2*newPc + (1-om2)*Pc[i][nslice-2];}
       }
                            Pc[0][nslice-2]=Pc[2][nslice-2];
              Pc[0][nslice-1]=Pc[2][nslice-1];
                         Error = 0;
           for (j=4;j < nslice-2;++j){
                     for (i =1; i < imax+1; ++i){
                            newPc = Pc[i][j];
                            if (bound[i][j] == 0)
                            {      if(i==1) third = (Pc[i+1][j] + Pc[i-
1][j])/sqr(RCMS); // Pc[i][j] term exists.
                                   else third= (Pc[i+1][j] - Pc[i-1][j])/((i-
1)*RCMS*2*RCMS); // No Pc[i][j] term
                                   if(bound[i][j+2]!=1 & bound[i][j-2]!=1){
                                          meanPc = (Pc[i-
1][j]+Pc[i+1][j])/sqr(RCMS) + (16*(Pc[i][j-1]+Pc[i][j+1])-(Pc[i][j-
2]+Pc[i][j+2]))/sqr(ZCMS)/12+third; //O(h4) is used for second derivative in z
direction
                                          if(i==1) newPc = (meanPc -
Plap[i][j])/(Lambda1+2/sqr(RCMS));
                                          else newPc = (meanPc -
Plap[i][j])/Lambda1;}
              else{
                                          meanPc = (Pc[i-
1][j]+Pc[i+1][j])/sqr(RCMS) + (Pc[i][j-1]+Pc[i][j+1])/sqr(ZCMS)+third;
                                          newPc = (meanPc - Plap[i][j])/Lambda2;}}
                                                 else if (bound[i][j] == 2){
                                   if (bound[i+1][j] != 0     & bound[i-1][j] == 0)
{
                                          gr = (- 2*Pc[i-1][j] + 0.5*Pc[i-
2][j])/RCMS;  //O(hr)
                                          ar = 1.5/RCMS;  }
                                   else if(bound[i-1][j] != 0 & bound[i+1][j] == 0)
                                   {                                        gr = -
(- 2*Pc[i+1][j] + 0.5*Pc[i+2][j])/RCMS;   //O(hr)
                                          ar = -1.5/RCMS;}
                                   else if(bound[i+2][j] == 1){
                                          gr = (- 2*Pc[i-1][j] + 0.5*Pc[i-
2][j])/RCMS;  //O(hr)
                                          ar = 1.5/RCMS;}
                                   else if(bound[i-2][j] == 1)
                                   {                                        gr = -
(- 2*Pc[i+1][j] + 0.5*Pc[i+2][j])/RCMS;   //O(hr)
                                          ar = -1.5/RCMS;}
       else
                                   {                                        gr =
(Pc[i+1][j] - Pc[i-1][j])/(2*ZCMS);       //O(hr)
                                          ar = 0;}

                                   gz = (8*Pc[i][j+1] - 8*Pc[i][j-1] - Pc[i][j+2] +
Pc[i][j-2])/12/ZCMS;//O(hz^3)
                                   az = 0;
                                   if(j==3)
                                   {      gz = -(2*Pc[i][j-1] - 6*Pc[i][j+1] +
Pc[i][j+2])/(6*ZCMS);   //O(hz^2)
                                          az = -.5/ZCMS;}
                                   else if(j== nslice-2)
```

```
                                    {                           gz = (2*Pc[i][j+1] -
6*Pc[i][j-1] + Pc[i][j-2])/(6*ZCMS);    //O(hz^2)
                                          az = .5/ZCMS;}
       if (bound[i][j+1] == 2 & bound[i][j+2] == 1)
       {                    gz = (2*Pc[i][j+1] - 6*Pc[i][j-1] + Pc[i][j-
2])/(6*ZCMS);   //O(hz^2)
                                          az = .5/ZCMS;}
       else if(bound[i][j-1] == 2& bound[i][j-2] == 1)
                                    {                                  gz = -
(2*Pc[i][j-1] - 6*Pc[i][j+1] + Pc[i][j+2])/(6*ZCMS);   //O(hz^2)
                                          az = -.5/ZCMS;
       }
                                          if (bound[i][j+1] == 1){
                                          if(j >2)
                                          {      gz = (- 2*Pc[i][j-1] +
0.5*Pc[i][j-2])/ZCMS;//can be more accurate.O(hz)
                                                 az = 1.5/ZCMS;}
                                                               else{
                                                 gz = - Pc[i][j-1]/ZCMS;// This
condition hardly occures.  O(1)!!!
                                                 az = 1/ZCMS;  }       }
                                    else if(bound[i][j-1] == 1){
                                          if(j< nslice-1)
                                          {      gz = -(- 2*Pc[i][j+1] +
0.5*Pc[i][j+2])/ZCMS;//can be more accurate.O(hz)
                                                 az = -1.5/ZCMS;}
                                          else
                                          {                     gz =
Pc[i][j+1]/ZCMS;// This condition hardly occures.  O(1)!!!
                                                 az = -1/ZCMS;
                                                 cout<<"\n****";}}
                                                                      newPc =
(dpn[i][j]-n[i][j].r*gr-n[i][j].z*gz)/(n[i][j].r*ar+n[i][j].z*az);   }
                                                        if (bound[i][j] != 1){
                                    Error += fabs(newPc-Pc[i][j]);
                                    if (Error > 1000) cout<<"\n("<<i<<","<<j<<")"<<"
Error= " <<Error<<" change= "<<newPc-Pc[i][j]<<"new= "<<newPc;
                                    if(bound[i][j] == 2)Pc[i][j] = om2*newPc + (1-
om2)*Pc[i][j];
                                    else Pc[i][j] = om1*newPc + (1-om1)*Pc[i][j];
                                    Pc[i][j] = newPc;}              }
                     Pc[0][j]=Pc[2][j];}
                            gradient(Pc , Pcgr ,nslice ,imax,RCMS,ZCMS);
                            if(10*floor(loop/10.0) == loop){
                 FError = 0;
                 cnt =0;
                 for (j=4;j < nslice-2;++j)
                 {                  for (i =1; i < imax+1; ++i)
                         {                            if (bound[i][j] == 0)
                               {
                                          FError += (sqr(Pcgr[i][j].r- Pgr[i][j].r)
+ sqr(Pcgr[i][j].z- Pgr[i][j].z))*(i-1)*RCMS;
                                          cnt += i-1;}                 }
}
                     FError /= cnt;       }
       }while ((Error > PointEr*cnt & loop < 25000 & FError <= 1.1*lFError &
fabs(lError/Error-1) > 1.0e-20 ) |loop < 1000 );
                     gradient(Pc , Pcgr ,nslice,imax,RCMS,ZCMS);
```

```
                FError = 0;cnt1 = 0; temp=0; temp2=0;
        for (j=4;j < nslice-2;++j){       for (i =1; i < imax+1; ++i){
                        if (bound[i][j] == 0)
                        {                            FError += (sqr(Pcgr[i][j].r-
Pgr[i][j].r) + sqr(Pcgr[i][j].z- Pgr[i][j].z))*(i-1)*RCMS;
                                temp += Pc[i][j]*(i-1);
                                cnt1 += i-1;}       }      }
        FError /= cnt1;
        j=1;i=2;
        while (j < nslice &(fabs(n[i][j].z)<.02 | bound[i][j] != 2))
        {               if (i < imax) i++;
                else {j++;i=1;}}
        j1=j; j++;
        while (j < nslice &(fabs(n[i][j].z)>=.02 | bound[i][j] != 2))
        {               if (i < imax) i++;
                else {j++;i=1;}}
        j++;
        while (j < nslice &(fabs(n[i][j].z)<.02 | bound[i][j] != 2))
        {               if (i < imax) i++;
                else {j++;i=1;}}
        j++;
        while (j < nslice &(fabs(n[i][j].z)>=.02 | bound[i][j] != 2))
        {               if (i < imax) i++;
                else {j++;i=1;}}
        j2=j;
        if (j2 > nslice)cout<<"\n Region of interest is not determined properly and
must be fixed manually.";
        temp=0; temp2 = 0; cnt1 = 0;
        for (j=j1+6; j < j2-4 ; j++){
                for (i=1; i<imax+1; i++){
                        if (bound[i][j] != 1){
                                temp += Pc[i][j]*(i-1);
                                cnt1 += i-1;}       }      }
                path = "enter path here" + str;
        fout.open(path.c_str(),ios_base::out);
                temp = 0;
        for(j= 0 ; j< nslice +2; j++){
                for(i= 0; i<imax+1; i++){
                        fout<< Pc[i][j] <<"\n";}
                if(j>2 & j<nslice-1)temp += Pcgr[12][j].z;}
        temp /= (nslice-4);
                return 0;}
                void tecread(int* j00,int* imax, const char* filename, int* nslice,
double RCMS, double ZCMS)
{       int i,j,jp,ip,temp5;
        char header[30];
        double temp1,temp2,temp3,temp4,temp1p,temp2p;
        ifstream fin;
        fin.open(filename, ios_base::in);
        fin.getline(header,29);
        fin>> temp1 >> temp2 >> temp3 >> temp4 >> temp5;
        if (temp2 < 0) {temp2 = -temp2;}
        *imax = (int) (floor) (0.5 + temp2/RCMS) +1;
        *j00  = (int) (floor) (0.5 + temp1/ZCMS) -1;
    bound[*imax][1] = temp5;
    v[*imax][1].r = temp4;
    v[*imax][1].z = temp3;
        jp=1; temp2p=temp2; temp1p=temp1;
```

```
       fin>> temp1 >> temp2 >> temp3 >> temp4 >> temp5;
       i = (int) (floor) (0.5 + temp2/RCMS) + 1;
       j = (int) (floor) (0.5 + temp1/ZCMS) - *j00;
               bound[i][j] = temp5;
    v[i][j].r = temp4;
    v[i][j].z = temp3;
       ip=i; jp=j;
       do     {
              fin>> temp1 >> temp2 >> temp3 >> temp4 >> temp5;
              ip=i; jp=j;
              i = (int) (floor) (0.5 + temp2/RCMS) + 1;
              j = (int) (floor) (0.5 + temp1/ZCMS) - *j00;
              if(i >= 1)      {
                     bound[i][j] = temp5;
                     v[i][j].r = temp4;
                     v[i][j].z = temp3;}
              if(i==1) v[i][j].r = 0;
       }while(j != jp | i != ip);
       *nslice = jp;
       for(j=1;j< *nslice +1;j++){
                            bound[0][j] = bound[2][j];
              v[0][j].r = - v[2][j].r;
              v[1][j].r = 0;
              v[0][j].z = v[2][j].z;}}
double gasdev(int *idum){
   static int iset=0;
   static double gset;
   double fac, r,v1,v2;
   double ran1(int *);
      if (iset == 0) {
      do {
        v1=2.0*ran1(idum)-1.0;
              v2=2.0*ran1(idum)-1.0;
              r=v1*v1+v2*v2;
       } while(r >= 1.0);
      fac = sqrt(-2.0*log(r)/r);
      gset = v1*fac;
      iset = 1;
      return v2*fac;
   } else {
      iset = 0;
      return gset;}}
#define M1 259200
#define IA1 7141
#define IC1 54773
#define RM1 (1.0/M1)
#define M2 134456
#define IA2 8121
#define IC2 28411
#define RM2 (1.0/M2)
#define M3 243000
#define IA3 4561
#define IC3 51349
double ran1(int * idum){
   static long ix1,ix2,ix3;
   static double r[98];
   double temp;
   static int iff=0;
   int j;
   if (*idum < 0 || iff ==0) {
      iff = 1;
      ix1 = (IC1-(*idum)) % M1;
      ix1 = (IA1*ix1+IC1) % M1;
      ix2 = ix1 % M2;
      ix1 = (IA1*ix1+IC1) % M1;
      ix3 = ix1 % M3;
      for (j=1;j<=97;j++) {
           ix1 = (IA1*ix1+IC1) % M1;
           ix2 = (IA2*ix2+IC2) % M2;
          r[j]= (ix1+ix2*RM2)*RM1;}
       *idum=1;}
    ix1 = (IA1*ix1+IC1) % M1;
    ix2 = (IA2*ix2+IC2) % M2;
    ix2 = (IA3*ix3+IC3) % M3;
    j=1 +((97*ix3)/M3);
    if (j > 97 || j < 1) nrerror("RAN1: This can not happen.");
    temp=r[j];
    r[j]=(ix1+ix2*RM2)*RM1;
    return temp;}
void nrerror(char error_text[])
{        fprintf(stderr,"Numerical Recipes run-time error...\n");
        fprintf(stderr,"%s\n",error_text);
        fprintf(stderr,"...now exiting to system...\n");
        exit(1);}
int find_nclosest(point contour[],int contour_size,int nslice,int i ,double
x[],int x_size, double y[], int y_size, double RCMS, double ZCMS, int length)
{      int r,m,k,up,dn;
       int* index = new int[length+1];
       double* dist = new double[length+1];
           for(k=1;k<length+1;++k){
              dist[k] =
distance(contour[i].r,contour[i].z,contour[k].r,contour[k].z);;}
       for (k = 0; k < 2*ponum+1; k++){
              index[k] = findxmaller(dist,length);
              m = index[k];
              dist[m] = dist[m] + hugdis;}
       up=0; dn=0;  k=0;
       do{
              m = index[k];
              dist[m] = dist[m] - hugdis;
              x[k] = contour[m].r;
              y[k] = contour[m].z;
              if ((y[k]-contour[i].z) > epsilon)
              {                    up=(int)(0.5+(y[k]-contour[i].z)/ZCMS);
       }
              if ((y[k]-contour[i].z) < -epsilon){
                     dn=(int)(0.5+(contour[i].z-y[k])/ZCMS);}
                            k++;
       }while(k <= ponum | up+dn < 2);
              return(k);}
void Fitlane(double* a,double* b,double x[],int n,double y[],int m,int size)
{
       #define DEBug 0
       double sxx, sxy, sxz, syy, syz, szz,sx,sy,sz,mag,c;
       double B[4][4],d[4];
       int i,j,k;
       void jacobi(double a[][4], int n, double d[], double v[][4]);
       double* z = new double[2*size];
       for (k = 0; k < size; k++)
       {             x[k+size] = x[k];
              y[k+size] = y[k];
              z[k] = -1;
              z[k+size] = 1;     }
       size *=2;
    sz = sx = sy = sxx = sxy = sxz = syy = syz = szz = 0.0;
       for (i = 0; i < size; ++i)
       {             sx = sx + (double)(x[i])/SIGMA2;
              sy = sy + (double)(y[i])/SIGMA2;
              sz = sz + (double)(z[i])/SIGMA2;
              sxx = sxx + (double)(x[i] * x[i])/SIGMA2;
              sxy = sxy + (double)(x[i] * y[i])/SIGMA2;
              sxz = sxz + (double)(x[i] * z[i])/SIGMA2;
              syy = syy + (double)(y[i] * y[i])/SIGMA2;
              syz = syz + (double)(y[i] * z[i])/SIGMA2;
              szz = szz + (double)(z[i] * z[i])/SIGMA2;   }
              double A[4][4];
       A[1][1] = sxx-sx*sx/size; A[1][2] = sxy-sx*sy/size; A[1][3] = sxz-
sx*sz/size;
       A[2][1] = A[1][2];    A[2][2] = syy-sy*sy/size; A[2][3] = syz-sy*sz/size;
       A[3][1] = A[1][3];    A[3][2] = A[2][3];    A[3][3] = szz-sz*sz/size;
              jacobi(A,3,d,B);
       d[0]= HUGEIG; //Is it necessary?
       i = findxmaller(d,3);
       *a = B[1][i];
       *b = B[2][i];
       c = B[3][i];
       mag = sqrt(sqr(*a) + sqr(*b));
       *a= *a/mag;
       *b= *b/mag;}
double distance(double x1,double y1,double x2,double y2)
{   double d;
   d = sqrt((double)(sqr(x1-x2)+sqr(y1-y2)));
   return(d);}
int findxmaller(double ra[],int ra_size)
{      int l,ii;
       double s;
       s = ra[1]+hugdis;
       for (l=1; l<ra_size+1;l++)
       {             if(ra[l] < s)
              {                    s = ra[l];
                     ii = l;}}
       return (ii);
}
double innerprod(double a1,double b1, double c1,double a2,double b2, double c2)
{      double a;
       a=a1*a2 + b1*b2 +c1*c2;
       return (a);}
#define ROTATE(a,i,j,k,l) g=a[i][j];h=a[k][l];a[i][j]=g-
s*(h+g*tau);a[k][l]=h+s*(g-h*tau);
void jacobi(double a[][4], int n, double d[], double v[][4])
{    int j,iq,ip,i,ii,jj;
   double tresh,theta,tau,t,sm,s,h,g,c,b[4],z[4];
   for (ip=1;ip<=n;ip++) {          //Initialize to the identity matrix.
        for (iq=1;iq<=n;iq++) v[ip][iq]=0.0;
       v[ip][ip]=1.0;
     }
   for (ip=1;ip<=n;ip++) {     //Initialize b and d to the diagonal of a.
       b[ip]=d[ip]=a[ip][ip];
       z[ip]=0.0;          //This vector will accumulate terms of the form tapq as
in equa- tion (11.1.14).a'pp = app - tapq
     }
   for (i=1;i<=50;i++) {
      sm=0.0;
      for (ip=1;ip<=n-1;ip++) {    //Sum off-diagonal elements.
         for (iq=ip+1;iq<=n;iq++)
          sm += fabs(a[ip][iq]); }
//      printf ("sum of Off Diagonal elements = %g\n",sm);     //
      if (sm == 0.0) {             // The normal return, which relies on quadratic
convergence to machine underflow.
       return;}
      if (i < 4)
           tresh=0.2*sm/(n*n);     //...on the  first three sweeps.
      else
           tresh=0.0;              //...thereafter.
      for (ip=1;ip<=n-1;ip++) {
          for (iq=ip+1;iq<=n;iq++) {
             g=100.0*fabs(a[ip][iq]);

             if(i > 4 &&(double)(fabs(d[ip])+g) == (double)fabs(d[ip])
              && (double)(fabs(d[iq])+g) == (double)fabs(d[iq]))
                a[ip][iq]=0.0;
             else if (fabs(a[ip][iq]) > tresh) {
                  h=d[iq]-d[ip];
                 if ((double)(fabs(h)+g) == (double)fabs(h))
                     t=(a[ip][iq])/h; //t = 1/(2*teta )
                 else {
                     theta=0.5*h/(a[ip][iq]); //Equation (11.1.10).
                     t=1.0/(fabs(theta)+sqrt(1.0+theta*theta));
                     if (theta < 0.0) t = -t; }
                 c=1.0/sqrt(1+t*t);
                 s=t*c;
                 tau=s/(1.0+c);
                 h=t*a[ip][iq];
                 z[ip] -= h;
                 z[iq] += h;
                 d[ip] -= h;
                 d[iq] += h;
                 a[ip][iq]=0.0;
                 for (j=1;j<=ip-1;j++) {  // Case of rotations 1 <= j <p.
                    ROTATE(a,j,ip,j,iq)}
                 for (j=ip+1;j<=iq-1;j++) {      // Case of rotations p <j< q.
                    ROTATE(a,ip,j,j,iq)}
                 for (j=iq+1;j<=n;j++) {  // Case of rotations q <j <= n.
                    ROTATE(a,ip,j,iq,j) }
                 for (j=1;j<=n;j++) {
                    ROTATE(v,j,ip,j,iq)}}}}
       for (ip=1;ip<=n;ip++) {
           b[ip] += z[ip];
          d[ip]=b[ip];      // Update d with the sum of tapq,
          z[ip]=0.0;        // and reinitialize z.
        } } nrerror("Too many iterations in routine jacobi"); }
```

```
void filter(point A[][MAXSLICE],int nslice,int imax,double sig1,double sig2,
double RCMS, double ZCMS)
{      int i,j;
       double tmpz1,tmpz2,tmpz3,tmpz4,tmpr1,tmpr2,tmpr3,tmpr4;
       double m, temp,a0,a1,a2,a3;
       for (i=0;i<imax+2;i++)
              for(j=0;j<nslice+2;j++)
              {
                     vfilt[i][j].z = A[i][j].z;
                     vfilt[i][j].r = A[i][j].r;
              }
       m= -.5*sqr(RCMS/sig1);
       a0=1; a1=exp(m); a2=exp(4*m); a3=exp(9*m);
       temp = 1+2*(a1+a2+a3);
       a0=a0/temp; a1=a1/temp; a2=a2/temp; a3=a3/temp;
       for(j=0; j<nslice+2;j++)
       {////////////////this filter does not do filtering for marginal upper rows,
but does it for lower rows using symmetry.
              for(i=1;i< imax-2;i++)
              {                    if( i==1 ){
                            vfilt[i][j].z = a0*A[i][j].z + a1*(A[i-
1][j].z+A[i+1][j].z) + a2*(A[3][j].z+A[i+2][j].z) + a3*(A[4][j].z+A[i+3][j].z);
                            vfilt[i][j].r = a0*A[i][j].r + a1*(A[i-
1][j].r+A[i+1][j].r) + a2*(-A[3][j].r+A[i+2][j].r) + a3*(-A[4][j].r+A[i+3][j].r);
                     } else if( i==2 ){
                            vfilt[i][j].z = a0*A[i][j].z + a1*(A[i-
1][j].z+A[i+1][j].z) + a2*(A[i-2][j].z+A[i+2][j].z) + a3*(A[3][j].z+A[i+3][j].z);
                            vfilt[i][j].r = a0*A[i][j].r + a1*(A[i-
1][j].r+A[i+1][j].r) + a2*(A[i-2][j].r+A[i+2][j].r) + a3*(-A[3][j].r+A[i+3][j].r);
                     } else if( i > 2 ){
                            if (bound[i][j] != 1){
                                   if (bound[i+1][j] != 1){
                                          if (bound[i+2][j] != 1){
                                                 vfilt[i][j].z = a0*A[i][j].z +
a1*(A[i-1][j].z+A[i+1][j].z) + a2*(A[i-2][j].z+A[i+2][j].z) + a3*(A[i-
3][j].z+A[i+3][j].z);
                                                 vfilt[i][j].r = a0*A[i][j].r +
a1*(A[i-1][j].r+A[i+1][j].r) + a2*(A[i-2][j].r+A[i+2][j].r) + a3*(A[i-
3][j].r+A[i+3][j].r);
                                          } else
       {
                                                 vfilt[i][j].z = (a0*A[i][j].z +
       a1*(A[i-1][j].z+A[i+1][j].z)+ a2*(A[i-2][j].z+A[i+2][j].z))/(a0+2*a1+2*a2);
                                                 vfilt[i][j].r = (a0*A[i][j].r +
a1*(A[i-1][j].r+A[i+1][j].r)+ a2*(A[i-2][j].r+A[i+2][j].r))/(a0+2*a1+2*a2);}
                                   } else        {
                                          vfilt[i][j].z = (a0*A[i][j].z +
       a1*(A[i-1][j].z+A[i+1][j].z))/(a0+2*a1);
                                          vfilt[i][j].r = (a0*A[i][j].r + a1*(A[i-
1][j].r+A[i+1][j].r))/(a0+2*a1);}
                            }else                {
                                   vfilt[i][j].z = A[i][j].z; vfilt[i][j].r =
A[i][j].r;}}         }
              vfilt[0][j].z = vfilt[2][j].z; vfilt[0][j].r = -vfilt[2][j].r;}
       m= -.5*sqr(ZCMS/sig2);
       a0=1; a1=exp(m); a2=exp(4*m); a3=exp(9*m);
       temp = 1+2*(a1+a2+a3);
       a0=a0/temp; a1=a1/temp; a2=a2/temp; a3=a3/temp;
```

```
    for(i=1;i< imax;i++)
    {           tmpz1=A[i][0].z; tmpz2=A[i][1].z; tmpz3=A[i][2].z;
    tmpr1=A[i][0].r; tmpr2=A[i][1].r; tmpr3=A[i][2].r;
        for(j=1; j<nslice+1;j++)
        {                   if (bound[i][j] != 1){
                            if (bound[i][j+1] != 1 & bound[i][j-1] != 1 & j
> 1 & j < nslice){
                                if (bound[i][j+2] != 1 & bound[i][j-2] !=
1 & j > 2 & j < nslice - 1)
                                {
    vfilt[i][j].z = a0*A[i][j].z + a1*(A[i][j-1].z+A[i][j+1].z) + a2*(A[i][j-
2].z+A[i][j+2].z) + a3*(A[i][j-3].z+A[i][j+3].z);
                                    vfilt[i][j].r = a0*A[i][j].r +
a1*(A[i][j-1].r+A[i][j+1].r) + a2*(A[i][j-2].r+A[i][j+2].r) + a3*(A[i][j-
3].r+A[i][j+3].r);
                                } else{
                                    vfilt[i][j].z = (a0*A[i][j].z +
a1*(A[i][j-1].z+A[i][j+1].z)+ a2*(A[i][j-2].z+A[i][j+2].z))/(a0+2*a1+2*a2);
                                    vfilt[i][j].r = (a0*A[i][j].r +
a1*(A[i][j-1].r+A[i][j+1].r)+ a2*(A[i][j-2].r+A[i][j+2].r))/(a0+2*a1+2*a2);
                                }} else{
                                vfilt[i][j].z = (a0*A[i][j].z +
a1*(A[i][j-1].z+A[i][j+1].z))/(a0+2*a1);
                                vfilt[i][j].r = (a0*A[i][j].r +
a1*(A[i][j-1].r+A[i][j+1].r))/(a0+2*a1);}
                }else           {
                    vfilt[i][j].z = A[i][j].z; vfilt[i][j].r = A[i][j].r;}}
    }
  for( j = 2; j < nslice ; j++) {
    vfilt[0][j].z = vfilt[2][j].z; vfilt[0][j].r = -vfilt[2][j].r;}
    for (i=0;i<imax+2;i++)
        for(j=0;j<nslice+2;j++)
        {                   A[i][j].z = vfilt[i][j].z;
                A[i][j].r = vfilt[i][j].r;}}
void diver(point A[][MAXSLICE], double B[][MAXSLICE],int nslice , int imax, double
RCMS, double ZCMS)
{   int i,j,k;
    double roundz2,roundr1,third;
    for (j=2;j < nslice;++j)
    {           for (i =1; i < imax+1; ++i){
                if (bound[i][j] ==0){
                    roundr1 = (A[i+1][j].r - A[i-1][j].r)/2/RCMS;
                    if(j > 3 & j < nslice-2 & bound[i][j+2] == 0 &
bound[i][j-2] == 0)
                        roundz2 = (8*A[i][j+1].z - 8*A[i][j-1].z -
A[i][j+2].z + A[i][j-2].z)/12/ZCMS;//  O(hz^3)
                    else if(j > 2 & j < nslice-1 & bound[i][j+1] == 0 &
bound[i][j-1] == 0)
                        roundz2 = (A[i][j+1].z - A[i][j-1].z)/2/ZCMS;
// O(hz)
                    else if(j == 2 | bound[i][j-1] != 0)
                        roundz2 = -(2*A[i][j-1].z + 3*A[i][j].z -
6*A[i][j+1].z + A[i][j+2].z)/6/ZCMS;
                    else if(j == nslice-1 | bound[i][j+1] != 0)
                        roundz2 = (2*A[i][j+1].z + 3*A[i][j].z -
6*A[i][j-1].z + A[i][j-2].z)/6/ZCMS;
                    if(i==1)third = roundr1;
                    else third= A[i][j].r/(i-1)/RCMS;
```

```
                                B[i][j] = roundr1+roundz2+third;
                        }else{
                                B[i][j] = 0;}           }
                B[0][j]=B[2][j];}}
void cal_vdelv(int i,int j,int nslice, double RCMS, double ZCMS)
{       double roundr1,roundr2,roundz1,roundz2;
                roundr1 = (v[i+1][j].r - v[i-1][j].r)/2/RCMS;
        roundr2 = (v[i+1][j].z - v[i-1][j].z)/2/RCMS;
        if(j>1 & j<nslice & bound[i][j+1]==0 & bound[i][j-1]==0)
        {               roundz1 = (8*v[i][j+1].r - 8*v[i][j-1].r - v[i][j+2].r +
v[i][j-2].r)/12/ZCMS;//round of V.r over round of z
                roundz2 = (8*v[i][j+1].z - 8*v[i][j-1].z - v[i][j+2].z + v[i][j-
2].z)/12/ZCMS;}
        else    {
                roundz1 = (v[i][j+1].r - v[i][j-1].r)/2/ZCMS;
                roundz2 = (v[i][j+1].z - v[i][j-1].z)/2/ZCMS;}
        vdelv[i][j].r = v[i][j].r*roundr1 + v[i][j].z*roundz1 ;
        vdelv[i][j].z = v[i][j].r*roundr2 + v[i][j].z*roundz2 ;}
void cal_lapv(int i,int j,int nslice, double RCMS, double ZCMS)
{       double
roundr1,roundr2,round2r1,round2r2,round2z1,round2z2,urr2,r,thirdr,thirdz;
        roundr1 = (v[i+1][j].r - v[i-1][j].r)/2/RCMS;
        roundr2 = (v[i+1][j].z - v[i-1][j].z)/2/RCMS;
        round2r1 = (v[i+1][j].r - 2*v[i][j].r + v[i-1][j].r)/RCMS/RCMS;
        round2r2 = (v[i+1][j].z - 2*v[i][j].z + v[i-1][j].z)/RCMS/RCMS;// O(hr)
        if(j>1 & j<nslice & bound[i][j+1]==0 & bound[i][j-1]==0)
        {                       round2z1 = (16*v[i][j+1].r +16*v[i][j-1].r -
v[i][j+2].r - v[i][j-2].r - 30*v[i][j].r)/12/ZCMS/ZCMS;
                round2z2 = (16*v[i][j+1].z +16*v[i][j-1].z - v[i][j+2].z - v[i][j-
2].z - 30*v[i][j].z)/12/ZCMS/ZCMS;}
        else{
                round2z1 = (v[i][j+1].r - 2*v[i][j].r + v[i][j-1].r)/ZCMS/ZCMS;//
O(hz)
                round2z2 = (v[i][j+1].z - 2*v[i][j].z + v[i][j-1].z)/ZCMS/ZCMS;}
                r = (i-1)*RCMS;
        if (i!=1)
        {//                     i<3???????
                urr2 = v[i][j].r/r/r;
                thirdr = roundr1/r;
                thirdz = roundr2/r;
        }else{
                urr2 = round2r1;
                thirdr = round2r1;
                thirdz = round2r2;}
        lapv[i][j].r = round2r1 + round2z1 + thirdr -urr2;
        lapv[i][j].z = round2r2 + round2z2 + thirdz;}
void gradientns(int i,int j)
{       Pgr[i][j].r = RHO*(fr - vdelv[i][j].r) + MU*lapv[i][j].r;
        Pgr[i][j].z = RHO*(fz - vdelv[i][j].z) + MU*lapv[i][j].z;}
void estim_b_gradient(int i,int j,int nslice,double* Err,double* Erz)
#define N 6
{       double ab[N+1][4],zt[N+1][4],curlEr1;
        double A[200][N+1],z[200][4],a[N+1][N+1],ya[N+1][N+1],d,col[N+1];
        int ii,jj,kk,cnt=0,indx[N+1];
        for (ii=i-3; ii < i+4; ii++)
                for (jj=j-3; jj < j+4; jj++)
                if (bound[ii][jj] == 0 & sqr(ii-i)+sqr(jj-j) < 16){
                                cnt++;
```

```
                            A[cnt][1] = ii*ii; A[cnt][2] = jj*jj; A[cnt][3] =
ii*jj; A[cnt][4]= ii; A[cnt][5] = jj; A[cnt][6] = 1;
                            z[cnt][1] = Pgr[ii][jj].r; z[cnt][2] = Pgr[ii][jj].z;
z[cnt][3]=z[cnt][4]=0;}
                     for (ii = 1; ii< N+1; ii++)
              for (jj = 1; jj< N+1; jj++)
                     a[ii][jj] = 0;
                            for (ii = 1; ii< N+1; ii++) {
              for (jj = 1; jj< N+1; jj++){
                     for (kk = 1; kk < cnt+1; kk++)
                                                        a[ii][jj] +=
A[kk][ii]*A[kk][jj] ;}}
                                   for (ii = 1; ii< N+1; ii++)
              for (jj = 1; jj< 4; jj++)
                     zt[ii][jj] = 0;
                                          for (ii = 1; ii< N+1; ii++) {
              for (jj = 1; jj< 4; jj++)
              {                    for (kk = 1; kk < cnt+1; kk++)
                                          zt[ii][jj] += A[kk][ii]*z[kk][jj];}
       }
                                                 ludcmp(a,N,indx,&d); //Decompose
the matrix just once.
       for(jj=1;jj<=N;jj++)
       {   for(ii=1;ii<=N;ii++) col[ii]=0.0; // col is unity matrix.
              col[jj]=1.0;
              lubksb(a,N,indx,col);
              for(ii=1;ii<=N;ii++)
              {      ya[ii][jj]=col[ii];}}
       for (ii = 1; ii< N+1; ii++)
              for (jj = 1; jj< 4; jj++)
                     ab[ii][jj] = 0;
              for (ii = 1; ii< N+1; ii++) // ab = a'*zt = inv(A'A)*A'z
       {             for (jj = 1; jj< 4; jj++){
                     for (kk = 1; kk < N+1; kk++)
                            ab[ii][jj] += ya[ii][kk]*zt[kk][jj];}     }
       Pgr[i][j].r = i*i*ab[1][1] + j*j*ab[2][1] + i*j*ab[3][1] + i*ab[4][1] +
j*ab[5][1] + ab[6][1];
       Pgr[i][j].z = i*i*ab[1][2] + j*j*ab[2][2] + i*j*ab[3][2] + i*ab[4][2] +
j*ab[5][2] + ab[6][2];
              *Err = 0; *Erz = 0;
       cnt = 0;
       for (ii=i-3; ii < i+4; ii++)
              for (jj=j-3; jj < j+4; jj++){
                     if (bound[ii][jj] == 0 & sqr(ii-i)+sqr(jj-j) < 16){
                            cnt++;
                            *Err += sqr(ii*ii*ab[1][1] + jj*jj*ab[2][1] +
ii*jj*ab[3][1] + ii*ab[4][1] + jj*ab[5][1] + ab[6][1] -Pgr[ii][jj].r);
                            *Erz += sqr(ii*ii*ab[1][2] + jj*jj*ab[2][2] +
ii*jj*ab[3][2] + ii*ab[4][2] + jj*ab[5][2] + ab[6][2] -Pgr[ii][jj].z);}
                     if (bound[ii][j] == 0 & bound[ii+1][j] != 0) // Recalculation
of pressure gradient for nearest internal points to the boundary
                     {                           Pgr[ii][j].r = ii*ii*ab[1][1] +
j*j*ab[2][1] + ii*j*ab[3][1] + ii*ab[4][1] + j*ab[5][1] + ab[6][1];
                            Pgr[ii][j].z = ii*ii*ab[1][2] + j*j*ab[2][2] +
ii*j*ab[3][2] + ii*ab[4][2] + j*ab[5][2] + ab[6][2];}          }
       *Err /= cnt; *Erz /= cnt;
       *Err = sqrt(*Err); *Erz = sqrt(*Erz); //variance of the error due to 2nd
order approximation
```

```
              curlEr1 = 2*j*ab[2][1] + i*ab[3][1] + ab[5][1] - (2*i*ab[1][2] +
j*ab[3][2] + ab[5][2]);}
double inner_prod(point A[][MAXSLICE],point B[][MAXSLICE],int i,int j)
{      double a;
       a=A[i][j].r*B[i][j].r + A[i][j].z*B[i][j].z;
       return (a);}
#define TINY 1.0e-20   // A small number.
void ludcmp(double a[N+1][N+1], int n, int *indx, double *d)
{
   int i,imax,j,k;
   double big,dum,sum,temp;
   double *vv = new double[n+1];
   *d=1.0;           // No row interchanges yet.
   for (i=1;i<=n;i++) { // Loop over rows to get the implicit scaling information.
      big=0.0;
      for (j=1;j<=n;j++)
       if ((temp=fabs(a[i][j])) > big) big=temp;
      if (big == 0.0) nrerror("Singular matrix in routine ludcmp");
      vv[i]=1.0/big; }
    for (j=1;j<=n;j++) {    // This is the loop over columns of Crout s method.
              for (i=1;i<j;i++) {  // This is equation (2.3.12) except for i = j.
          sum=a[i][j];
         for (k=1;k<i;k++) sum -= a[i][k]*a[k][j];
         a[i][j]=sum; }
       big=0.0;  //Initialize for the search for largest pivot element.
       for (i=j;i<=n;i++) {        //This is i = j of equation (2.3.12) and i =
j+1. . .N of equation (2.3.13).
          sum=a[i][j];
          for (k=1;k<j;k++)
              sum -= a[i][k]*a[k][j];
          a[i][j]=sum;
          if ( (dum=vv[i]*fabs(sum)) >= big) {
              big=dum;
              imax=i; }     }
       if (j != imax) {     // Do we need to interchange rows?
          for (k=1;k<=n;k++) { // Yes, do so...
             dum=a[imax][k];
             a[imax][k]=a[j][k];
             a[j][k]=dum;}
           *d = -(*d);      //  ...and change the parity of d.
           vv[imax]=vv[j];}
       indx[j]=imax;
       if (a[j][j]== 0.0) a[j][j]=TINY;
       if (j != n) {        //Now, finally, divide by the pivot element.
           dum=1.0/(a[j][j]);
           for (i=j+1;i<=n;i++) a[i][j] *= dum;}     }
       delete []vv;}
void lubksb(double a[N+1][N+1], int n, int *indx, double b[])
{  int i,ii=0,ip,j;
  double sum;
    for (i=1;i<=n;i++) {
     ip=indx[i];
     sum=b[ip];
     b[ip]=b[i];
     if (ii)
       for (j=ii;j<=i-1;j++) sum -= a[i][j]*b[j];
     else if (sum) ii=i;
     b[i]=sum;}
  for (i=n;i>=1;i--) {  // Now we do the backsubstitution, equation (2.3.7).
     sum=b[i];
     for (j=i+1;j<=n;j++) sum -= a[i][j]*b[j];
     b[i]=sum/a[i][i]; }}
void gradient(double A[][MAXSLICE], point B[][MAXSLICE],int nslice,int imax,
double RCMS, double ZCMS)
{ int i, j;
  for (j=3;j < nslice-1;++j){
       for (i = 1; i < imax; ++i){
          if (bound[i][j] != 1){
                    B[i][j].r = (A[i+1][j] - A[i-1][j])/2/RCMS;      //O(hr)
                    B[i][j].z = (8*A[i][j+1] - 8*A[i][j-1] - A[i][j+2] + A[i][j-
2])/12/ZCMS;//O(hz^3)
                    if(j<4) B[i][j].z = -(2*A[i][j-1] + 3*A[i][j] - 6*A[i][j+1] +
A[i][j+2])/6/ZCMS;   //O(hz^2)
                    else if(j> nslice-3) B[i][j].z = (2*A[i][j+1] + 3*A[i][j] -
6*A[i][j-1] + A[i][j-2])/6/ZCMS;    //O(hz^2)
                    }else {
                    B[i][j].r = 0;
                    B[i][j].z = 0;}
           if(bound[i][j] == 2){
                    if (bound[i+1][j] == 1)
                         B[i][j].r = (1.5*A[i][j] - 2*A[i-1][j] + 0.5*A[i-
2][j])/RCMS;  //O(hr)
                    else if(bound[i-1][j] == 1)
                         B[i][j].r = -(1.5*A[i][j] - 2*A[i+1][j] +
0.5*A[i+2][j])/RCMS;}
           if (bound[i][j+1] == 2 & bound[i][j] == 0)
                     B[i][j].z = (2*A[i][j+1] + 3*A[i][j] - 6*A[i][j-1] + A[i][j-
2])/6/ZCMS;   //O(hz^2)
           else if(bound[i][j-1] == 2& bound[i][j] == 0)
                      B[i][j].z = -(2*A[i][j-1] + 3*A[i][j] - 6*A[i][j+1] +
A[i][j+2])/6/ZCMS;   //O(hz^2)
           if(bound[i][j] == 2){
                    if (bound[i][j+1] == 1)
                         if(j >3) B[i][j].z = (1.5*A[i][j] - 2*A[i][j-1] +
0.5*A[i][j-2])/ZCMS;//can be more accurate.O(hz)
                        else B[i][j].z = (A[i][j] - A[i][j-1])/ZCMS;// This
condition hardly occures.  O(1)!!!
                    else if(bound[i][j-1] == 1)
                         if(j< nslice-2) B[i][j].z = -(1.5*A[i][j] - 2*A[i][j+1] +
0.5*A[i][j+2])/ZCMS;//can be more accurate.O(hz)
                        else B[i][j].z = (A[i][j+1] - A[i][j])/ZCMS;// This
condition hardly occures.  O(1)!!!
                    else if (bound[i][j+2] == 1)
                         if(j >3) B[i][j].z = (2*A[i][j+1] + 3*A[i][j] - 6*A[i][j-
1] + A[i][j-2])/6/ZCMS;//  O(hz^2)
                        else B[i][j].z = (A[i][j+1] - A[i][j-1])/2/ZCMS;// This
condition hardly occures.  O(hz)
                    else if(bound[i][j-2] == 1)
                         if(j< nslice-2) B[i][j].z = -(2*A[i][j-1] + 3*A[i][j] -
6*A[i][j+1] + A[i][j+2])/6/ZCMS;   //O(hz^2)
                        else B[i][j].z = (A[i][j+1] - A[i][j-1])/2/ZCMS;}     }
         if (A[0][j] == A[2][j]) {B[0][j].z = B[2][j].z; B[0][j].r = -B[2][j].r;}
         else if (A[0][j] == -A[2][j]) {B[0][j].z = -B[2][j].z; B[0][j].r =
B[2][j].r;}}}
```

Appendix B: MATLAB code for non-iterative technique

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Non-iterative technique
% Direct matlab implementation of the algorithms presented in papers:
% IEEE-TPAMI 1998(10):439-51 and LNCS 2005(3565): 431-42
% By Iman Khodarahmi 2008-2012
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

fid1 = fopen('C:\Boundary.dat');
fid2 = fopen('C:\Gr.dat');
fid3 = fopen('C:\Gz.dat');
B = fscanf(fid1,'%g');
Gr = fscanf(fid2,'%g');
Gz = fscanf(fid3,'%g');
fclose all;
rRES = 1;
zRES = 1;
B = reshape(B,20,100);
Gr = reshape(Gr,20,100);
Gz = reshape(Gz,20,100);
for i=1:size(Gr,2)
    tempGr = Gr(:,i); tempGz = Gz(:,i); tempB = B(:,i);
    if ((i~=1) && (i~=size(Gr,2)))
        tempder = (find(B(:,i+1) == 2) - find(B(:,i-1) == 2))/2;
    elseif (i==1)
        tempder = find(B(:,i+1) == 2) - find(B(:,i) == 2);
    else
        tempder = find(B(:,i) == 2) - find(B(:,i-1) == 2);
    end;
    tempGr(tempB == 1)= 0; ind = find(tempB == 0);
    tempGz(tempB == 1)= tempGz(tempB == 2) + tempGr(tempB == 2)*
tempder;
    Gr(:,i) = tempGr; Gz(:,i) = tempGz;
end;
Gre = [fliplr([-flipud(Gr); Gr]) [-flipud(Gr); Gr]]; GRE = fft2(Gre);
Gze = [-fliplr([flipud(Gz); Gz]) [flipud(Gz); Gz]]; GZE = fft2(Gze);
% Variable definitions
vec_r = complex(0,sin(2*pi*(0:size(Gre,1)-1)/size(Gre,1)))/rRES;
vec_z = complex(0,sin(2*pi*(0:size(Gre,2)-1)/size(Gre,2)))/zRES;
for i=1:size(Gre,1)
    for j=1:size(Gre,2)
        PE(i,j)=(conj(vec_r(i))*GRE(i,j) + conj(vec_z(j))*GZE(i,j)) /
(abs(vec_r(i))*abs(vec_r(i))+abs(vec_z(j))*abs(vec_z(j)));
    end
end
PE(1,1) = 0; pe = ifft2(PE);
p = real(pe(1+size(Gr,1):size(Gre,1),size(Gr,2)+1:size(Gre,2)));
p(B ~= 0) = NaN; contourf(p,50)
```

CURRICULUM VITAE

Iman Khodarahmi

Contact Information

Work Address: Department of Electrical and Computer Engineering, J.B. Speed School of Engineering
University of Louisville, Louisville, KY 40292, USA
Work Phone: 502-852-0354
E-mail: i0khod01@louisville.edu
imankhodarahmi@yahoo.com

Education

Ph.D., Electrical Engineering, 2012
Department of Electrical and Computer Engineering, University of Louisville, Louisville, KY

M.D., Medicine, 2007
School of Medicine, University of Tehran, Tehran, Iran

M.S., Biophysics, 2007
Institute of Biochemistry and Biophysics, University of Tehran, Tehran, Iran

B.S., Solid State Physics, 2006
Department of Physics, University of Tehran, Tehran, Iran

Honors and Awards

2008-2012: Golden Key International Honor Society
2010: Honorable Mention Prize, Graduate Research Symposium, University of Louisville.
2010-2011: University of Louisville Fellows and Graduate Award.
2009: Ranked first among participants in the PhD Applicancy Qualifying Exam (AQE), Department of Electrical and Computer Engineering, University of Louisville, Jan 2009.
2006: Ranked first among all students participating in the university entrance exam for M.S. in biophysics
2006: Ranked first among bachelor students of the department of physics, University of Tehran, Tehran, Iran
2001-2006: University of Tehran scholarship for academic achievements, awarded by exceptional talent committee, University of Tehran, Tehran, Iran
2001: Permission to double-major (studied physics in addition to medicine) according to academic performance, awarded by exceptional talent committee of University of Tehran, Tehran, Iran (for the first time in Iran)
1998: Ranked 26th in Iran national exam of university entrance among more than 500,000 applicants
1997: National Physics Olympiad Award, Silver medal among more than 15,000 applicants, Ministry of Education, Tehran, Iran

Publications

Book chapter:

Khodarahmi I, Rashidi A. 2006, A careful review of basic assumptions in modeling atrial fibrillation and suggesting new ideas for research. In: Peeyush Chandra, B. V. Rathish Kumar (Eds). Mathematical Biology: Recent trends. Anamaya Publishers, New Delhi, India. pp. 305-311.

Refereed journal publications:

In-Vitro Validation of Flow Measurement with Phase Contrast MRI at 3T Using Stereoscopic Particle Image Velocimetry. **Khodarahmi I**, Shakeri M, Kotys-Traughber M, Fischer S, Sharp M, Amini AA. [J Magn Reson Med. In revision with high priority for publication]

The effect of 2.1 T static magnetic field on astrocyte viability and morphology. **Khodarahmi I**, Mobasheri H, Firouzi M. Magn Reson Imaging. 2010 Apr 22;28(6): 903-9. PMID: 20456893

Relationship between ABO blood groups and seroprevalence of Helicobacter pylori. **Khodarahmi I**, Rashidi A, Khodarahmi P. Indian J Gastroenterol. 2008 Mar-Apr;27(2):82-3. PMID: 18695311

Immunogenicity of recombinant hepatitis B virus vaccine in patients with and without chronic hepatitis C virus infection: a case-control study. Daryani NE, Nassiri-Toosi M, Rashidi A, **Khodarahmi I**. World J Gastroenterol. 2007 Jan 14;13(2):294-8. PMID: 17226912

Anxiolytic effect of noscapine in mice. Khodarahmi P, Rostami P, Rashidi A, **Khodarahmi I**. Pharmacol Rep. 2006 Jul-Aug;58(4):568-70. PMID: 16963805

Mathematical modelling of the course of chronic renal failure. Rashidi A, **Khodarahmi I**. Nephrol Dial Transplant. 2006 Nov;21(11):3344-5. PMID: 16887853

Mathematical modeling of the course and prognosis of factitious disorders: a game-theoretic approach. Rashidi A, **Khodarahmi I**, Feldman MD. J Theor Biol. 2006 May 7;240(1):48-53. PMID: 16207492

Nonlinear modeling of the atrioventricular node physiology in atrial fibrillation. Rashidi A, **Khodarahmi I**. J Theor Biol. 2005 Feb 21;232(4):545-9. PMID: 15588634

Refereed conference proceeding:

Preliminary Imaging of Red Blood Cells in Turbulent Flow, M. Shakeri, **I. Khodarahmi**, M. K. Sharp, Proceedings of the ASME 2012 Summer Bioengineering Conference SBC, June 2012-80416, Farjardo, Puerto Rico, USA. [Accepted]

Comparison of Relative Pressures Calculated from PC-MRI and SPIV with Catheter-based Pressure Measurements in a Stenotic Phantom Model, **I. Khodarahmi**, M. Shakeri, M. Kotys-Traughber, M.K. Sharp, and A.A. Amini, Proceedings of the SPIE Conference on Biomedical Applications in Molecular, Structural, and Functional Imaging, Feb 2012, San Diego, CA.

Pressure gradients calculated from PC-MRI, SPIV and CFD velocity data in a phantom model: comparison with catheter-based pressure measurement, **I. Khodarahmi**, M. Shakeri, M. Kotys-Traughber, S. Fischer, K. Sharp, A. Amini, Journal of Cardiovascular

Magnetic Resonance 2012, 14(Suppl 1):W34. Presented at 15th Annual SCMR Scientific Sessions, Feb 2012, Orlando, FL.

Accuracy of flow measurement with phase contrast MRI in a stenotic phantom: where should flow be measured? **I. Khodarahmi**, M. Shakeri, M. Kotys-Traughber, S. Fischer, K. Sharp, A. Amini, Journal of Cardiovascular Magnetic Resonance 2012, 14(Suppl 1):P219. Presented at 15th Annual SCMR Scientific Sessions, Feb 2012, Orlando, FL.

Using PIV to determine relative pressures in a stenotic phantom under steady flow based on the pressure-poisson equation. **Khodarahmi I**, Shakeri M, Sharp M, Amini AA. Conf Proc IEEE Eng Med Biol Soc. 2010; 2010:2594-7. PMID: 21096448

In-Vitro Validation of Phase Contrast MRI at 7T in a Stenotic Phantom Under Steady Flow Using PIV, **I Khodarahmi**, M. Shakeri, R. Pratt, J. Wansapura, M. K. Sharp, and Amir A. Amini, International Society of Magnetic Resonance in Medicine Annual Meeting, May 2010, Stockholm, Sweden.

Optical Imaging of Steady Flow in a Phantom Model of Iliac Artery Stenosis: Comparison of CFD Simulations with PIV Measurements, M. Shakeri, **I. Khodarahmi**, M. Sharp, and Amir A. Amini, Proceedings of the SPIE Conference on Biomedical Applications in Molecular, Structural, and Functional Imaging, Feb 2010, San Diego, CA.

Biophysical Studies of the Effects of 2.1 Tesla Static Magnetic Field on the Neural Cell Growth. **I. Khodarahmi**, H. Mobasheri, M. Firouzi. Biophysical Journal. 94(2, Supp 1): 1067-1076, Feb 2008, DOI: 10.1016/S0006-3495(08)79191-4

Khodarahmi I., Rashidi A., A careful review of basic assumptions in modeling atrial fibrillation and suggesting new ideas for research. International Conference on Mathematical Biology (ICMB) 2004, Kanpur, India.

Presentations

Optical Imaging of Steady Flow in a Phantom Model of Iliac Artery Stenosis, **I. Khodarahmi**, M. Shakeri, M. Sharp, A. Amini, Graduate Research Symposium, March 2010, University of Louisville, Louisville, KY, USA.

Pressure calculation in a Phantom Model of Iliac Artery Stenosis based on MRI and PIV data. **I. Khodarahmi**, M. Shakeri, K. Sharp, A. Amini, Research!Louisville, Oct 2009, University of Louisville, Louisville, KY, USA.

Khodarahmi I, Mobasheri H, Linear Assembles of Proteins at Strong Magnetic Fields: a theoretical approach. Protein Assembly in Materials, Biology and Medicine: Direct Impact on Biological Nanoscience, 2007, Crete, Greece.

H. Mobasheri, **I. Khodarahmi**, H. Khoskhou Delshad, M. Firouzi, Biophysical manipulation of neural cell's growth in culture by means of electric and magnetic fields . European Life Scientist Organization (ELSO), 2007, Dresden, Germany.

Khodarahmi I, Mobasheri H, Application of static magnetic fields to guide axonal growth, a biophysical preliminary report. Forth National Congress of Spinal Cord Injuries, 2007, Tehran, Iran.

Aledavood A, **Khodarahmi I**, Salmasian H, Joneidi A, Low level laser therapy (830 nm) in lateral epicondylitis: a case report. 18th International Medical Sciences Student Congress, 2003, Istanbul, Turkey.

Work and research interests

Cardiovascular MRI
Flow quantification methods
MRI sequence programming
Mathematical simulations

Professional Affiliations

SCBT-MR (The Society of Computed Body Tomography & Magnetic Resonance)
ISMRM (The International Society for Magnetic Resonance in Medicine)

Computer Skills

Programming Languages: Turbo Pascal, C, C++
Mathematical Software: MATLAB, MATCAD
Engineering Software: Solidworks, Gambit, Fluent
Statistical Analysis Software: MINITAB, SPSS
General Purpose Software: Microsoft Office (MS-Excel, MS-Word, MS-PowerPoint)

Certifications

Educational Commission for Foreign Medical Graduates (ECFMG)
IDEA Sequence Design Environment, Siemens Medical
Sequence Design Environment, Philips Medical

Language Skills

English: fluent
Persian: Native fluent

References will be furnished upon request.