BRNO UNIVERSITY OF TECHNOLOGY
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF INFORMATION TECHNOLOGY
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

ON-THE-FLY COMPRESSION IN TIME-DOMAIN ULTRASOUND SIMULATIONS

PRŮBĚŽNÁ KOMPRESE V ULTRAZVUKOVÝCH SIMULACÍCH V ČASOVÉ OBLASTI

PHD THESIS
DISERTAČNÍ PRÁCE

AUTHOR
AUTOR PRÁCE

Ing. PETR KLEPÁRNÍK

SUPERVISOR
ŠKOLITEL

prof. Dr. Ing. PAVEL ZEMČÍK, dr. h. c.

BRNO 2022

Abstract

This work proposes a new compression method and its application in the framework of time-domain ultrasound simulations, specializing in high-intensity focused ultrasound (HIFU). Large-scale numerical simulations of HIFU, important for model-based treatment planning, generate large amounts of data. A simulation typically requires hundreds of gigabytes of storage. The goal of using this method is to significantly save computing resources while maintaining sufficient quality of the simulation outputs. At the core of this work, experimental simulations are presented, which show that the proposed compression method and its use for on-the-fly calculation of the average acoustic intensity during the simulation bring significant improvements. The main advantage is to a large extent (up to 99%) reduced consumption of precious disk space and approximately the same requirement for operational memory during simulation, which can significantly reduce the price of the computing platform. Compression does not adversely affect the overall simulation time. The accuracy of the new method was evaluated using thermal simulations. Using the new method, the same results are achieved in ablated tissue determination as in other approaches.

Abstrakt

Tato práce navrhuje novou kompresní metodu a její aplikaci v rámci ultrazvukových simulací v časové oblasti se specializací na cílený ultrazvuk o vysoké intenzitě (HIFU). Rozsáhlé numerické simulace HIFU, důležité pro plánování léčby založené na modelu, generují velké množství dat. Při simulaci je obvykle nutné ukládat stovky gigabajtů. Cílem použití této metody je významná úspora výpočetních prostředků při zachování dostatečné kvality simulačních výstupů. V jádru této práce jsou prezentovány experimentální simulace, které ukazují, že navržená kompresní metoda a její využití pro on-the-fly výpočet průměrné akustické intenzity během simulace přináší významné vylepšení. Hlavní výhodou je do značné míry (až 99 %) snížená spotřeba vzácného místa na disku a přibližně stejný nárok na operační paměť během simulace, což může výrazně snížit cenu výpočetní platformy. Komprese neovlivňuje nepříznivě celkovou dobu simulace. Přesnost nové metody byla vyhodnocena prostřednictvím tepelných simulací. Pomocí nové metody je dosaženo v podstatě stejných výsledků při stanovení ablatované tkáně jako u jiných přístupů.

Keywords

Ultrasound simulation, compression, high-intensity focused ultrasound, average acoustic intensity, k-Wave toolbox, k-space pseudospectral method, high-performance computing

Klíčová slova

Ultrazvukové simulace, komprese, cílený ultrazvuk o vysoké intenzitě, průměrná akustická intenzita, k-Wave toolbox, k-space pseudospektrální metoda, vysoce výkonné výpočty

Reference

KLEPÁRNÍK, Petr. *On-the-fly compression in time-domain ultrasound simulations.* Brno, 2022. PhD thesis. Brno University of Technology, Faculty of Information Technology. Supervisor prof. Dr. Ing. Pavel Zemčík dr. h. c..

On-the-fly compression in time-domain ultrasound simulations

Declaration

I declare that this dissertation thesis is my original work under the supervision of prof. Dr. Ing. Pavel Zemčík dr. h. c. I have listed all the literary sources and publications that have been used for the preparation of this thesis.

. . . .
Petr Klepárník
November 10, 2022

Acknowledgements

I would like to thank my supervisor prof. Pavel Zemčík for his professional guidance, consultation, and support during the preparation of this work. I also thank my colleagues for their cooperation in the creation of scientific publications, I especially thank doc. Jiří Jaroš and dr. Bradley E. Treeby.

Contents

List of Figures

List of Tables

Acronyms

1-D One-dimensional. 2, 13, 31, 33, 43, 44, 47, 50, 53, 54, 57, 58, 78

2-D Two-dimensional. 26, 33, 37, 39, 40, 79, 81

3-D Three-dimensional. 10, 12–17, 19, 23, 25, 26, 29, 31, 33, 34, 39, 40, 44, 47, 49, 50, 62, 63, 68, 70, 79, 81

4-D Four-dimensional. 2, 7, 17, 31, 50, 54, 57, 58, 62, 64, 68

AAC Advanced audio coding. 28, 29, 37, 51, 53, 54

AC-3 Dolby digital audio codec. 28, 29, 51, 53, 54

ADPCM Adaptive differential pulse-code modulation. 28, 51–54

ALAC Apple lossless audio codec. 29, 51, 53, 54

AMI Average mutual information. 37

APCA Adaptive piecewise constant approximation. 30

CEM43 Cumulative equivalent minutes relative to $T = 43$ %. 69, 70, 72, 78

CFL Courant–Friedrichs–Lewy. 62

CHEB Chebyshev approximations. 30

CPU Central processing unit. 24, 26, 31, 37, 42, 69

CT Computed tomography. 11, 12, 17

CUDA Compute unified device architecture. 24, 50, 66, 69

DCT Discrete cosine transform. 26, 31

DPCA Distributed principal component analysis. 29

DWT Discrete wavelet transform. 26

EBRT External beam radiation therapy. 22

ECG Electrocardiogram. 29

FDTD Finite difference time domain. 11

FFT Fast Fourier transform. 12, 13, 24, 26, 64, 69

FFTW Fastest Fourier transform in the west. 24

FLAC Free lossless audio codec. 29, 51, 53, 54

GAMPS Grouping and amplitude scaling. 30

GPU Graphic processing unit. 10, 14, 24, 31, 33, 37, 69

HCM HIFU compression method. 2, 51–54

HDF5 Hierarchical data format 5. 13, 17, 24, 50

HIFU High-intensity focused ultrasound. 2, 7, 9–11, 19–22, 30–33, 42, 43, 47, 51–55, 61, 62, 66, 78–80

HLR Hierarchical low-rank. 34

HPC High-performance computing. 8, 25, 33, 34, 43

IFFT Inverse fast Fourier transform. 69

JPEG Joint photographic experts group. 26, 33, 37

LAVA Largest variation. 33

LPC Linear predictive coding. 29, 33

LTC Lightweight temporal compression. 30

MDCT Modified discrete cosine transform. 28, 29

MGARD Multigrid adaptive reduction of data. 34

MOS Multiple of overlap size. 2, 48, 52–55, 57, 58

MP3 MPEG-1 or MPEG-2 audio layer III. 28

MPEG Motion picture experts group. 28, 33

MPI Message passing interface. 10, 13, 14, 24, 65, 81

MRI Magnetic resonance imaging. 11

MSE Mean squared error. 36, 37

NMI Normalized mutual information. 37

OpenMP Open multi-processing. 13, 24, 50, 65, 66

PAA Piecewise aggregate approximation. 30

PBVR particle-based volume rendering. 31

PCA Piecewise constant approximation. 30

PCH Piecewise constant histogram. 30

PCM Pulse-code modulation. 51–54

PG Points per wavelength. 33

PLA Piecewise linear approximation. 30

PPW Points per wavelength. 62

PSNR Peak-signal-to-noise ratio. 2, 36, 43, 50, 52–58

PSTD Pseudospectral time domain. 7, 11, 15, 80

PWLH Piecewise linear histogram. 30

RAM Random-access memory. 2, 3, 26, 37, 43, 44, 59, 61, 63–70, 80

RF radio frequency. 33

RIDA Robust information-driven data compression architecture. 30

RMS Root mean square. 17

SBR Self-based regression. 30

SF Slide/swing filters. 30

SIMD Single instruction, multiple data. 24

SNR Signal-to-noise ratio. 36, 37

SSIM Structure similarity measure. 37

SWAB Sliding window and bottom-up. 30

TC Trace compression. 33

TMM Tissue-mimicking material. 22

USCT Ultrasound computed tomography. 31, 33

UST Ultrasound tomography. 9, 22

WHT Walsh-Hadamard transform. 31

WMA Windows media audio. 28

Chapter 1

Introduction

Simulations of ultrasound wave propagation have a wide range of applications. Especially simulations of ultrasound propagation through biological tissues are very useful for planning ultrasound treatments. High-intensity focused ultrasound (HIFU) presents a typical early stage technology that most probably will be widely used in the future for cancer treatment. It is a non-invasive therapy, during which a narrowly focused beam of ultrasound is used to rapidly heat the tissue in the selected area and the cells are destroyed by this. The most important effect of ultrasound is a precisely localized, well-timed thermal effect. HIFU simulations can significantly help surgeons and researchers advance this technology and make treatments more precise and successful.

At this time, to perform accurate ultrasound simulations, extensive computational power and a large amount of storage space are required. Typically, it is necessary to save hundreds of gigabytes or tens of terabytes during and after the simulation running on several thousands of processor cores.

The topic of this thesis falls mainly into two categories of computer technology. The first category covers large-scale ultrasound simulations and the second is data compression.

This work is focused on the compression of such large specific ultrasound simulations data. The work proposes a new on-the-fly compression method and its application within time-domain ultrasound simulations with a specialization in HIFU. The goal of the new compression method is to significantly save computing resources while maintaining sufficient quality of simulation outputs.

In the core of this work, experimental simulations are presented, showing that the proposed compression method and its use for the on-the-fly calculation of the average acoustic intensity during the simulation (which uses a staggered-grid pseudospectral time domain (PSTD) method) bring significant improvements. The main advantage is to a large extent (up to 99 %) reduced consumption of precious disk space and approximately the same requirement for operational memory during simulation, which can significantly reduce the price of the computing platform. Compression does not adversely affect the overall simulation time. The accuracy of the new method was evaluated using thermal simulations. Using the new method, the same results are achieved in ablated tissue determination as with other approaches.

The use of a new method could also lead to, for example, more efficient treatment planning, fast four-dimensional (4-D) volume data visualizations, and parallel data processing on-the-fly during simulations. Saving computing resources increases the chances of making effective use of acoustic simulations in practice. The method can be applied to signals of similar character, for example, electromagnetic radio waves.

The consequent Chapter 2 provides information on state-of-the-art ultrasound simulations - tasks, outputs, applications, and technologies. Chapter 3 is focused on methods for data compression. Basic information on general data compression is presented. The known and powerful state-of-the-art image, video, and audio compression methods are presented in more detail. In addition, some methods or techniques that are more relevant to characteristic ultrasound simulation data and high-performance computing (HPC) are discussed. The evaluation measures for compression efficiency are mentioned in Chapter 4. First, the typical and known general measures for compression are presented, and then some possible approaches related to the main topic of this thesis are discussed. The core of this work is contained in Chapter 5. Section 5.1 contains the formulation of the hypothesis and the specification of the verification of this hypothesis. Sections 5.3 and 5.4 present core publications with experiments and work results. These are followed by Section 5.5, which validates the results and the scientific contribution. Section 5.6 is focused on applications and future work and Chapter 6 concludes the thesis.

Chapter 2

Ultrasound simulations

The purpose of this chapter is to present the state-of-the-art in the field of ultrasound simulations. This chapter is focused on ultrasound simulation tasks, outputs, applications, and used technologies. The topic is very closely related to the k-Wave acoustic toolbox and the HIFU simulations [33, 78] which are state-of-the-art technologies.

2.1 Tasks of ultrasound simulations

Ultrasound is a compression-dilatation wave with high frequencies (1–20 MHz in medical ultrasound compared to 20 Hz–20 kHz for audible sound). The pressure wave is composed of phases with compression and dilatation which propagate in the medium. Thus, sound and ultrasound cannot propagate in a vacuum. The average value of the sound speed in the biological environment is 1,540 $\mathrm{m\,s^{-1}}$, the corresponding wavelengths for ultrasound are usually between 0.08 mm to 1.54 mm [82].

Ultrasound simulations have various uses. For example, simulations are needed to perform transcranial focused ultrasound safely [4] or have become a widely used technique to evaluate ultrasound interactions in bone [38]. Another use is, for example, pelvic ultrasound simulation training among residents, sonographers, and general practitioners [5]. Significant use of simulations is also made in the field of ultrasound tomography (UST) - here a full time-domain waveform inversion is performed as a UST image formation technique [52, 57]. This work focuses mainly on HIFU simulations.

HIFU is one of the modern technologies for cancer treatment. It is an emerging non-invasive therapeutic technique that uses ultrasound waves to destroy tissue, such as tumors inside the human body. A beam of ultrasound energy is sent into the tissue using a focused transducer. The focused region is rapidly heated, resulting in irreversible tissue damage, while surrounding tissue is not affected (illustrated in Figure 2.1) [12, 20, 33, 73, 89].

In recent years, many HIFU clinical trials have been conducted for the treatment of tumors in the prostate, kidney, liver, breast or brain, but the most important issue and challenge is the precise placement of the ultrasound focus. There are some tissue properties that can significantly distort ultrasound distribution, for example the skull. That is the main reason why precise simulations are needed [33].

However, there are two main challenges. First, it is necessary to create an acoustic and thermal model that is physically complex, due to a heterogeneous medium and nonlinear wave propagation. Second, the simulations are computationally very intensive and expensive, as they must be executed on large domains with billions of grid points. An important

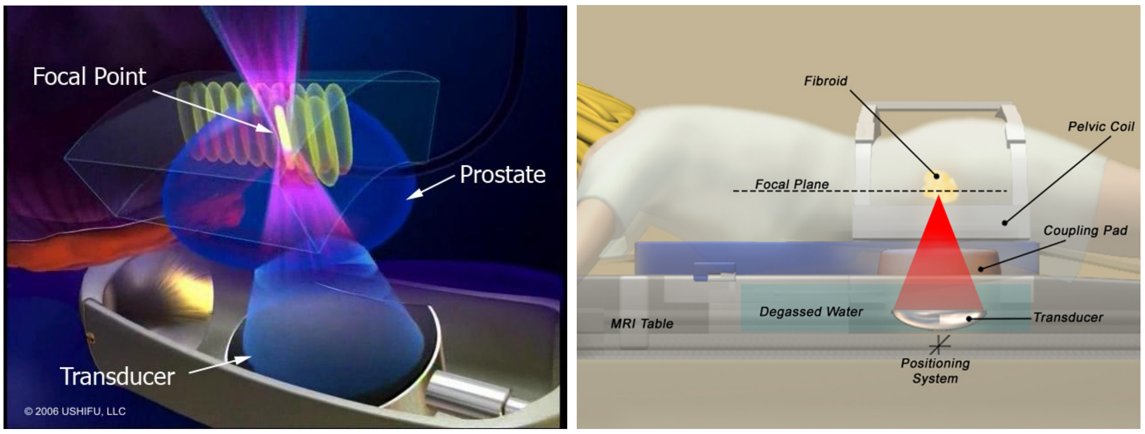


Figure 2.1: Schematic illustration of the HIFU device[1]and therapy[2].

factor is the time required for realistic and useful simulations. Now, some sophisticated parallel implementations in distributed clusters have been developed using message passing interface (MPI) technology or graphic processing units (GPU). One problem is the time required for the simulations, but the next big issue is the storage space required to save the simulation outputs [33, 35, 72].

Due to the large distances traveled by the ultrasound waves relative to the wavelength of the highest frequency harmonic and for precise results and applications in medical treatments, very large simulations have to be performed. Currently, the resolutions of the ultrasound simulation grid reach up to $4096 \times 2048 \times 2048$ samples in three-dimensional (3-D) space. Usually, the limited number of time steps in a small area encompassing the focus (sensor mask) or a discontinuous field of point sensors have to be sampled and saved for further processing, for example, in thermal modeling. The typical size of the sampled data for a clinically applicable simulation reaches 0.5 TB. If the entire simulation datasets had to be stored, for example, for 3000 simulation steps, ca 68 TB of storage space would be needed. To obtain a clinically relevant simulation, the grid sizes of 4096^3 to 8192^3 must be defined at least for 50 thousand simulation time steps [33, 72].

Typically, a simulation of wave propagation in a heterogeneous material is nonlinear. Thus, some higher harmonic frequencies of the source frequency are generated. The source frequency is usually between 0.5 and 2 MHz. Differences between harmonic frequencies generated in water and tissue are shown in Figure 2.2. At low focused locations, it is possible to get up to 10 harmonic frequencies, but at highly focused locations. Up to 600 harmonics would be useful for accurate modeling of heat propagation. For example, for histotripsy, up to 50 harmonics may be needed [33, 72].

In a real situation, for example, it is necessary to carry out a simulation of ultrasound propagation in the real domain with a size of 5 cm^3 (uniform Cartesian grid), the output data of the simulation must be saved using the maximum frequency 20 MHz and the speed of sound is assumed to be 1,500 $\text{m}\,\text{s}^{-1}$ (in water). So, with respect to the Nyquist theorem, a simulation grid size of 2667^3 is needed and this means 71 GB space for one matrix (one step in time) [33, 72].

To avoid the computational complexity of solving nonlinear acoustic equations in 3-D, there are one-way models, which have been successfully solved and simulated in homoge-

[1]http://miamiurologyconsultants.com/images/HIFU_lesions_labeled_C.jpg
[2]http://www.stargen-eu.cz/ostatni/medicina-2/fokusovany-ultrazvuk

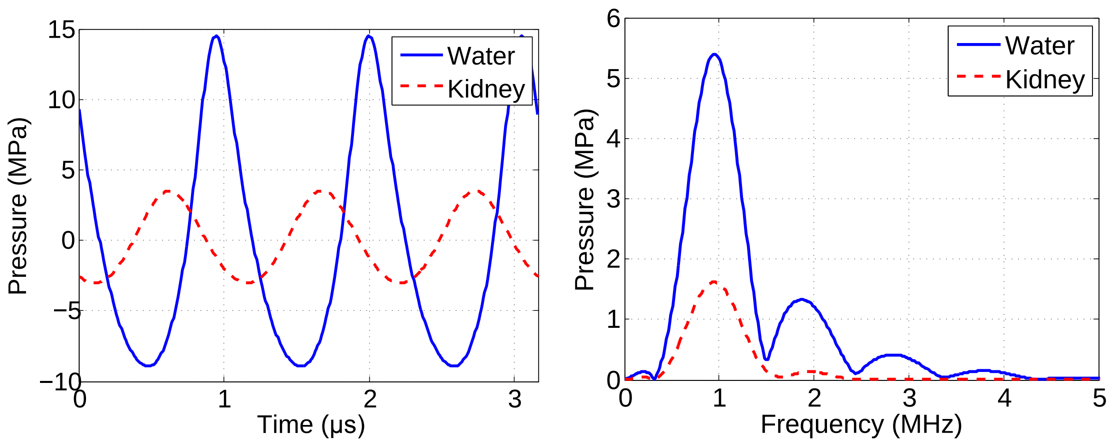


Figure 2.2: Time domain waveforms at the maximum peak pressure location in water and kidney and windowed harmonic frequency spectrum of the same wave forms[3].

neous media. However, for heterogeneous simulation of HIFU beams, the use of full-wave nonlinear models has been reported. These models are based on finite difference time domain (FDTD) methods (e.g. SimSonic[4] software). The computation has excellent weak scaling for properties for given grid sizes, but for large grids, there is big accumulation of numerical dispersion. The possible way to eliminate numerical dispersion is using the Fourier pseudospectral method and a more efficient k-space pseudospectral method that uses a time-staggered PSTD (e.g. software, such as k-Wave [78])[33].

Special methods for nonlinear ultrasound simulations in an axisymmetric coordinate system were also developed. The assumption of axisymmetry allows simulations with dense computational grids to model the propagation of nonlinear fields over large domains [81].

Typical HIFU treatment planning should consist of several steps:

1. The first step is computed tomography (CT) or magnetic resonance imaging (MRI) scan of the patient. On the scanned data a segmentation of bones, fat, skin, etc. is performed and the medium properties are gained (density, the speed of sound). A model for an acoustic simulation is created from these parameters [78].

2. The next step is the acoustic simulation. During the simulation HIFU, the source ultrasound signal is emitted from a transducer into the tissue with which it interacts. Multiple intersecting ultrasound beams are concentrated on the target (focus point). Some phenomena, such as attenuation, time delay, scattering, or nonlinear distortion, may occur during ultrasound propagation [7, 24, 73].

3. The output of the simulation can be in various quantities. This is usually acoustic pressure, acoustic particle velocity, or time-averaged acoustic intensity. Based on these data, for example, parameters for thermal simulations (the volume rate of heat deposition) can then be calculated [78].

4. Finally, the thermal simulation is executed to calculate the heat deposition. The result of the thermal simulation is information about the temperature in the target region

[3]The illustration taken from [72].
[4]http://www.simsonic.fr

after heating and cooling, the thermal dose, and the lesion size and an accurate focus position and aberration correction can be obtained. The final step is the application of ultrasound treatment [12, 33, 51, 55, 73].

2.2 Acoustic simulations

To model the effects of nonlinearity, acoustic absorption, and heterogeneities in the material properties, the final equation can be composed of three-coupled first-order partial differential equations [33, 73, 78] derived from conservation laws and a Taylor series expansion for pressure about density and entropy - momentum conservation Equation (2.1), mass conservation Equation (2.2) and pressure-density relation Equation (2.3)

$$\frac{\partial \mathbf{u}}{\partial t} = -\frac{1}{\rho_0}\nabla p + \mathbf{F}, \tag{2.1}$$

$$\frac{\partial \rho}{\partial t} = -\rho_0 \nabla \cdot \mathbf{u} - \mathbf{u} \cdot \nabla \rho_0 - 2\rho \nabla \cdot \mathbf{u} + \mathrm{M}, \tag{2.2}$$

$$p = c_0^2\left(\rho + \mathbf{d} \cdot \nabla \rho_0 + \frac{B}{2A}\frac{\rho^2}{\rho_0} - \mathrm{L}\rho\right) \tag{2.3}$$

where $\mathbf{u}$ is the acoustic particle velocity, p is the acoustic pressure, $\mathbf{F}$ is a force source term in $\mathrm{N\,kg^{-1}}$, M is a mass source term in units of $\mathrm{kg\,m^{-3}\,s^{-1}}$, $\mathbf{d}$ is the acoustic particle displacement, ρ is the acoustic density, ρ_0 is the ambient density, c_0 is the isentropic speed of sound, B/A is the nonlinearity parameter and L is a loss operator accounting for acoustic absorption and dispersion that follows a frequency power law.

The equations can be discretized using k-space pseudospectral method, where spatial gradients are calculated using the Fourier collocation spectral method, and time integration is performed using an explicit dispersion-corrected finite difference scheme [33, 73].

It is crucial and also somewhat disadvantageous that this method uses a staggered spatial and temporal grid for the calculation of the simulation step (the solution of the coupled first-order equations). In practice, this means that after one simulation step, the acoustic particle velocity is shifted relative to the acoustic pressure by half the time step and also by half the grid point spacing. Details are explained, for example, in [33].

It is important that the computation required 6 forward 3-D fast Fourier transforms (FFT), 8 inverse 3-D (FFTs) and overall it is about 100 element wise matrix operations. The development of an efficient numerical implementation that partitions the computational cost and memory usage across a large-scale parallel computer is desired.

The execution of the k-space pseudospectral method can be divided into three phases: preprocessing, simulation, and postprocessing [33, 78]. During the preprocessing, the input data for the simulation is generated - defining the domain discretization based on the size of the physical domain and the maximum frequency of interest, defining the spatially varying material properties (e.g., using a CT), defining the properties of the ultrasound transducer (e.g., the aperture diameter of the transducer bowl and the radius of curvature) and the drive signal, and defining the desired output data. The source ultrasound signals are always defined by a known harmonic function of a given frequency, usually a pressure sinusoid of frequency ranging from 0.5–1.5 MHz [73]. These parameters are defined in hierarchical data format (HDF5[5]) files.

The simulation phase is executed with the input data and with the discrete equations. The output data are stored. The postprocessing phase contains the analysis of the outputs and converts these data into human-readable form.

Briefly, during the simulation phase, the complete set of data stored in the memory contains:

- 21× real 3-D matrices in the spatial domain, and 3× real and 3× complex 3-D matrices in the Fourier domain (medium properties, time-varying acoustic quantities, derivative and absorption operators and temporary storage),
- 20× one-dimensional (1-D) size variable real vectors and
- approximately fifty scalar values defining, e.g., the domain size, grid spacing, number of simulation time steps, etc.

For computing 3-D FFT with MPI, two main ways of domain decomposition were developed. The first of them is 1-D domain decomposition [33], where 3-D matrices are partitioned along the z dimension and distributed across the MPI processes. 1-D vectors oriented along the dimensions x and y are broadcast. All scalar variables are broadcast and replicated on each process. An illustration of this approach is shown in Figure 2.3.

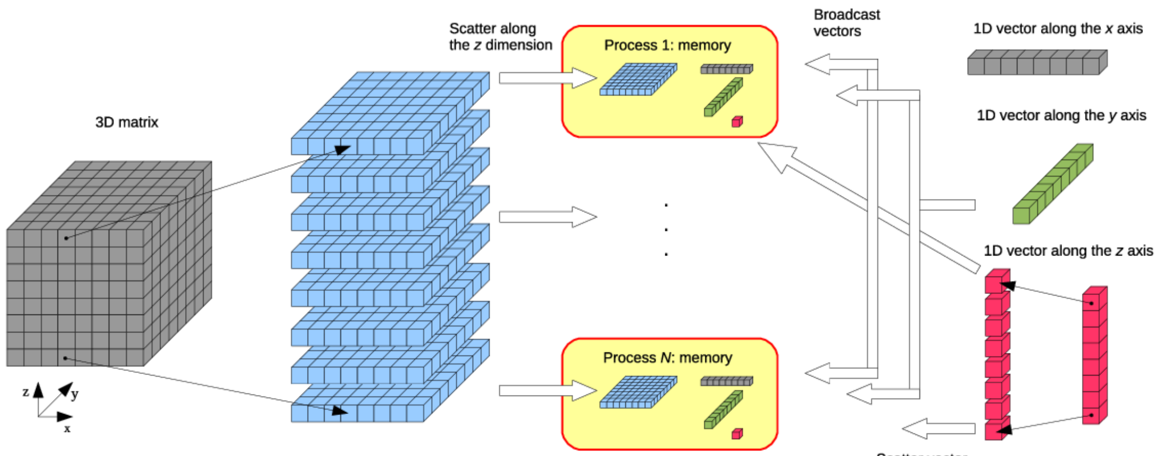

Scatter along the *z* dimension
Process 1: memory
Broadcast vectors
1D vector along the *x* axis
3D matrix
1D vector along the *y* axis
1D vector along the *z* axis
Process *N*: memory
Scatter vector
z
y
x

Figure 2.3: 1-D slab decomposition used to partition the 3-D domain within a distributed computing environment[6].

The second approach is hybrid open multiprocessing (OpenMP) / MPI decomposition [32] which tries to alleviate the disadvantages of the pure MPI decomposition by introducing a second level of decomposition and further breaking the 1-D slabs into pencils.

Another approach to compute the simulations is to use the local Fourier basis decomposition. By reducing the communication overhead and accepting a small numerical inaccuracy, it is possible to improve scaling from 512 to 8192 cores while reducing the simulation time

[5]https://www.hdfgroup.org/solutions/hdf5/
[6]The illustration taken from [33].

by a factor of 8.55. This approach eliminates the necessity of all-to-all communications by replacing them with local nearest-neighbor communication patterns[35] (Figure 2.4). More recent work [77], dedicated to local domain decomposition analysis of performance and accuracy on a single server containing eight NVIDIA P40 graphical processing units (GPUs), and compares it with the MPI version. Another recent work describes the implementation on the Salomon cluster equipped with 864 Intel Xeon Phi (Knight's Corner) accelerators [83].

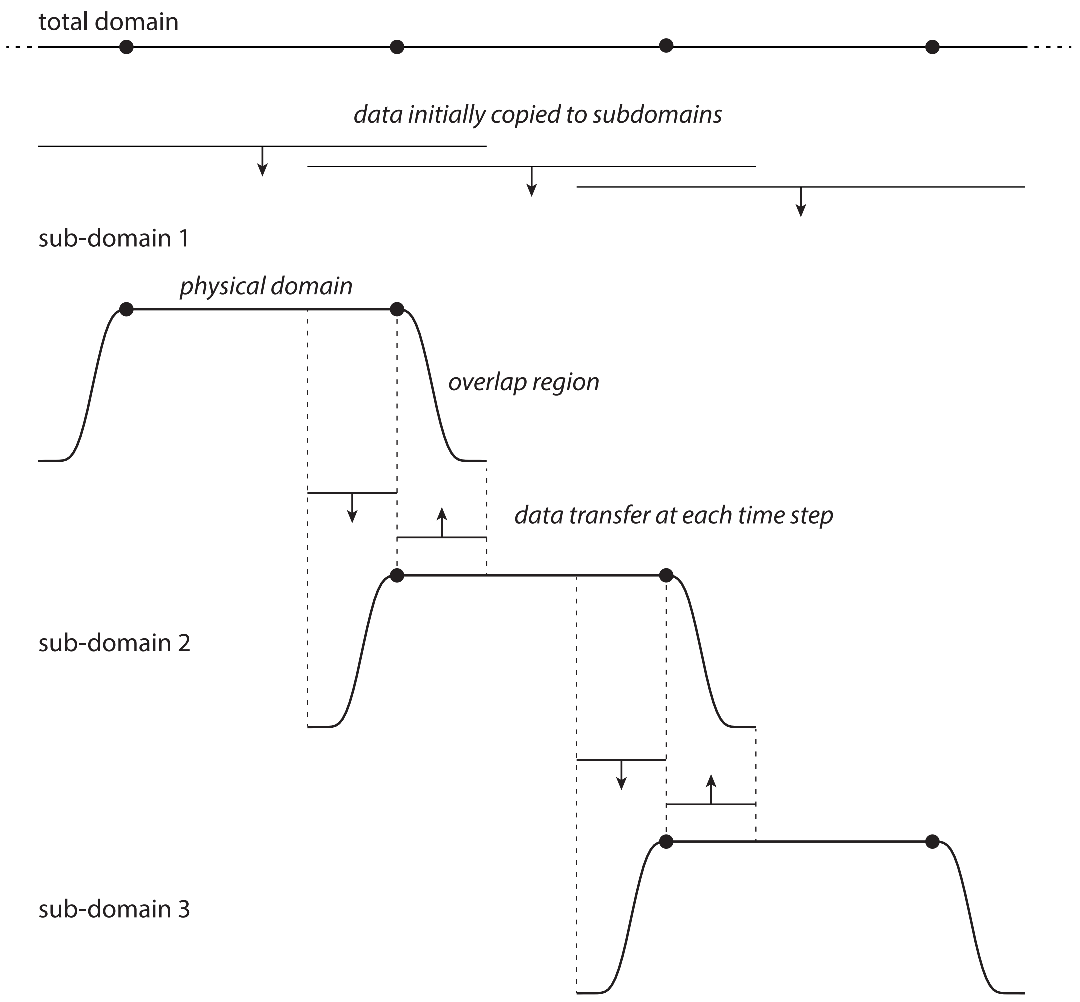


Figure 2.4: Schematic showing domain decomposition using local Fourier basis[7].

The ratio between total simulation time with and without saving the output data depends besides from the size of the sensor mask also on the type of implementation. The version with MPI-I/O technology provides several times faster parallel storing of the data. Generally, the time of storing the data range between 10 % and 70 % of the total simulation time. The minimal memory consumption of a typical simulation roughly corresponds to the size of the 3-D simulation domain times 30. But the MPI version with lots of compute nodes has a global memory peak many times higher (terabytes). With double enlargement

[7]The illustration taken from [35].

of the simulation domain, the time required for the simulation is about ten times larger. These values and claims were obtained by personal consultation and custom experiments.

2.3 Average intensity calculation

To calculate the thermal ablation in the tissue, an acoustic simulation is first performed. When using a time-staggered PSTD method, the results are the time-varying acoustic pressure and particle velocity (spatially and temporally staggered, a vector field) which can be used to calculate the time-averaged vector intensity. The average intensity can be used to calculate the inputs to the thermal simulations [80].

During the execution of the acoustic simulation, the acoustic pressure and the time-staggered particle velocity are often stored within the entire 3-D simulation domain. The reason why the entire domain is stored rather than a small area around the focus is the aliasing that arises when calculating the divergence of the average intensity as the input to the thermal simulations, and the accuracy of the input is the critical parameter of usability/precision of the thermal simulations. To calculate the average intensity, it is necessary to sample a signal with a duration of at least one period (T), which is given by the fundamental frequency of the ultrasound signal.

In each simulation (sampling) step, the current acoustic pressure and the time-staggered particle velocity are available. The use of staggered temporal (and also spatial) grids in the simulation calculations is related to discretization. In the case of discretization, their use brings us additional accuracy and stability [23]. Importantly, Fourier interpolation, which is typically used to accurately recalculate the particle velocity time shift, requires entire time series. Therefore, after the end of the simulation phase, the calculation of the average intensity vector $\mathbf{I}_{\text{avg}}$ is performed in the postprocessing phase according to

$$\mathbf{I}_{\text{avg}} = \frac{1}{T} \int_0^T p(t)\mathbf{u}(t)\mathrm{d}t \tag{2.4}$$

or

$$\mathbf{I}_{\text{avg}} = \frac{1}{N} \sum_{n=0}^{N-1} p(n)\mathbf{u}(n) \tag{2.5}$$

where

$$\mathbf{u}(n) = \mathbf{u}_{\text{staggered}}(n + 0.5) \tag{2.6}$$

and n or t is the simulation time step or time, respectively, $p(t)$ is acoustic pressure and $\mathbf{u}(t)$ is the vector acoustic particle velocity, T is the acoustic period of the fundamental frequency of the ultrasound signal. The evaluation of this equation is performed through numerical integration. N is the number of discrete signal samples taken within the period of T (it is assumed that T can be divided exactly into N sampling periods ($1/f_s$), $\Delta t = T/N$ (so that $N\Delta t = T$), $\mathbf{u}_{\text{staggered}}(n)$ is the time-staggered particle velocity output from the simulation, and $\mathbf{u}(n)$ is the velocity shifted half a step forward in time, typically using Fourier interpolation. For the calculation of the average intensity, the pressure and velocity data (vector field for the x, y, and z axes) must be read from the output file so that they are continuous over time [26, 78, 86].

A key bottleneck in this procedure is the fact that the average intensity must be calculated after the end of the acoustic simulation from the stored time-varying pressure and velocity data. The reason is the time shift of the particle velocity with respect to the acoustic pressure, which results from the time grid staggering in the k-space pseudospectral simulation method [33, 78]. This procedure requires reading the large stored time-varying simulation data from the files, temporally shifting the velocity data by half a time step, e.g., using Fourier interpolation, and calculating the average intensity by multiplying the velocity and pressure, and averaging. For large simulations, this means a large disk and memory consumption, in the order of terabytes, while the result should be a relatively small 3-D matrix with average intensity [26, 33, 73, 78].

2.4 Thermal simulations

As already mentioned in the Section 2.1, to obtain information about the temperature in focused regions, the thermal simulations can be performed based on the outputs of acoustic simulations. One of the ways to calculate thermal simulation is the Pennes' bio-heat equation [22, 26, 63, 73, 86] that uses, due to ultrasound absorption, the volume rate of the heat deposition term Q as an input quantity and also includes heat loss due to tissue perfusion (tissue blood flow) [78].

$$\rho_\mathrm{t} C_\mathrm{t} \frac{\partial T}{\partial t} = K_\mathrm{t} \nabla^2 T - \rho_\mathrm{b} W_\mathrm{b} C_\mathrm{b} (T - T_\mathrm{b}) + Q \tag{2.7}$$

where ρ_t is the tissue density in $\mathrm{kg\,m^{-3}}$, C_t is the tissue specific heat capacity in $\mathrm{J\,kg^{-1}\,K^{-1}}$, T is the total temperature in K, K_t is the tissue thermal conductivity in $\mathrm{W\,m^{-1}\,K^{-1}}$, ρ_b is the blood density in $\mathrm{kg\,m^{-3}}$, W_b is the blood perfusion rate in $\mathrm{s^{-1}}$, C_b is the blood specific heat capacity in $\mathrm{J\,kg^{-1}\,K^{-1}}$, T_b is the blood arterial temperature in K (initial temperature), and Q is the volume rate of heat deposition in $\mathrm{W\,m^{-3}}$. The subscripts 't' and 'b' refers to tissue and blood, resp. An example implementation (the time-domain solution) of this function is the `kWaveDiffusion` MATLAB function from k-Wave toolbox [78].

The calculation of the Q term is performed as soon as the acoustic simulation reaches a steady state. In the general case [26], the Q term can be calculated from the divergence of the time-averaged intensity according to

$$Q = -\mathrm{div}(\mathbf{I}_\mathrm{avg}) \tag{2.8}$$

where the divergence is calculated as the sum of the gradients for each axis. Another way to calculate the Q term is by approximating the plane wave relationship [73]

$$Q = \frac{1}{c_\mathrm{t} \rho_\mathrm{t}} \sum_{n=1}^{N} \alpha_\mathrm{t}(n f_0) |P_n|^2 \tag{2.9}$$

where c_t is the sound speed in the tissue, ρ_t is the frequency dependent attenuation in the tissue, f_0 is the fundamental frequency of the ultrasound signal, P_h is the pressure of the harmonic component h and H is the number of harmonics. The pressure values of each harmonic component can be obtained e.g. using the discrete Fourier transform of the time-domain ultrasound waveforms at each spatial location.

2.5 Simulation outputs

The simulations outputs are usually saved in HDF5 files [33, 78]. This file format is built for heterogeneous data, fast I/O processing and storage, and is composed of two primary types of objects: groups and datasets, which may also contain attributes with some extra information. By the way, this format also provides parallel writing to files. For postprocessing and visualization, there are two basic types of datasets: 3-D datasets and 4-D datasets.

In case of time-variable 4-D datasets, acoustic pressure and acoustic particle velocity can be stored across the defined area - sensor mask. Important is that the sensor mask can be an arbitrary and sparse set of locations (e.g. part of the ball shape or selected points corresponding to the skull bone illustrated in Figure 2.5) and the distribution of points in the 3-D space can not be predicted. A typical pressure or velocity signal recorded at a single grid point can be subdivided into three stages according to the magnitude of the amplitude and its changes. During the first stage, the signal resembles noise with an amplitude close to zero. The length (in samples) of this stage depends on the distance of the grid point from the transducer. The second stage can be characterized by a large increase in amplitude (a leading edge). This transient stage is usually very short, for example, around 5 % of the total simulation length. The third stage carries a relatively stable amplitude. This part of the signal is the most important part of the calculation of a heat deposition [73]. The stable stage usually starts at the moment when all waves emitted from the transducer arrive at the focus point. A typical sampled pressure signal recorded at one grid point is shown in Figure 2.6.

In 3-D datasets, aggregated values of acoustic pressure and acoustic particle velocity are stored (minimum, maximum and root mean square (RMS)) or its final, minimum and maximum values across the whole simulation domain. An example of volume rendered maximum aggregated acoustic pressure and CT scan of a kidney is shown in Figure 2.7. The figure is composed of two 3-D datasets. The blue cone with the small red area represents the simulated acoustic pressure. Other blended colors correspond to different tissues in CT data.

The data type of the computed values is float (32-bit). The datasets storage layout is simple, that is, usually contiguous linearly saved values, which are less suitable, e.g., for fast visualizations or postprocessing [40]. The format of the output data should be more analyzed and improved, and this is also related to the possible research topic, the topic of this thesis, and thus simulation data compression (more in Chapter 3). Everything depends on the application of the simulation outputs. One thing is certain, the outputs are very large and this fact must be considered.

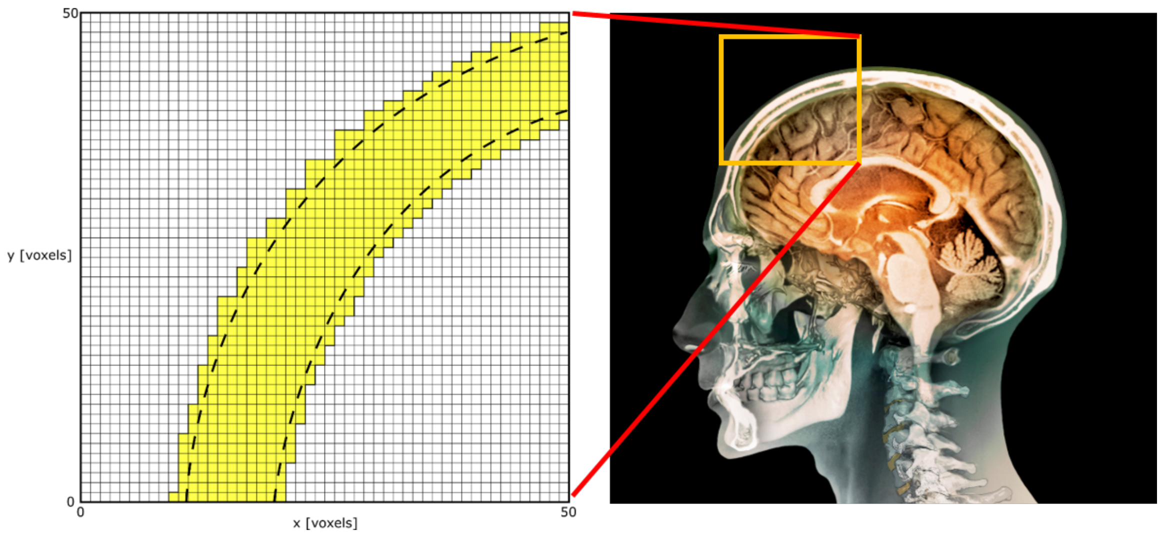


Figure 2.5: An example of a possible sensor mask definition for a skull[8].

[8]https://www.bbc.com/news/science-environment-37276219

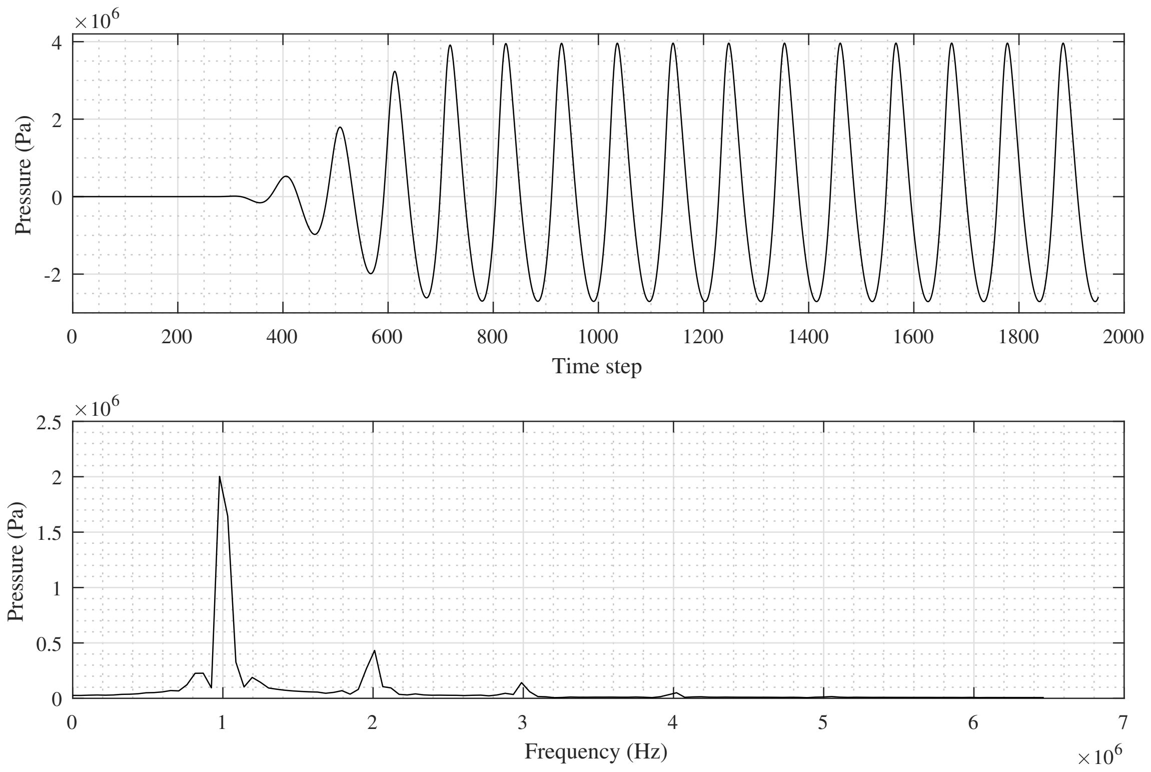


Figure 2.6: Illustration of a HIFU simulation time-variable signal (acoustic pressure) at one point in 3-D space including harmonic frequencies[9].

[9]The illustration taken from [41].

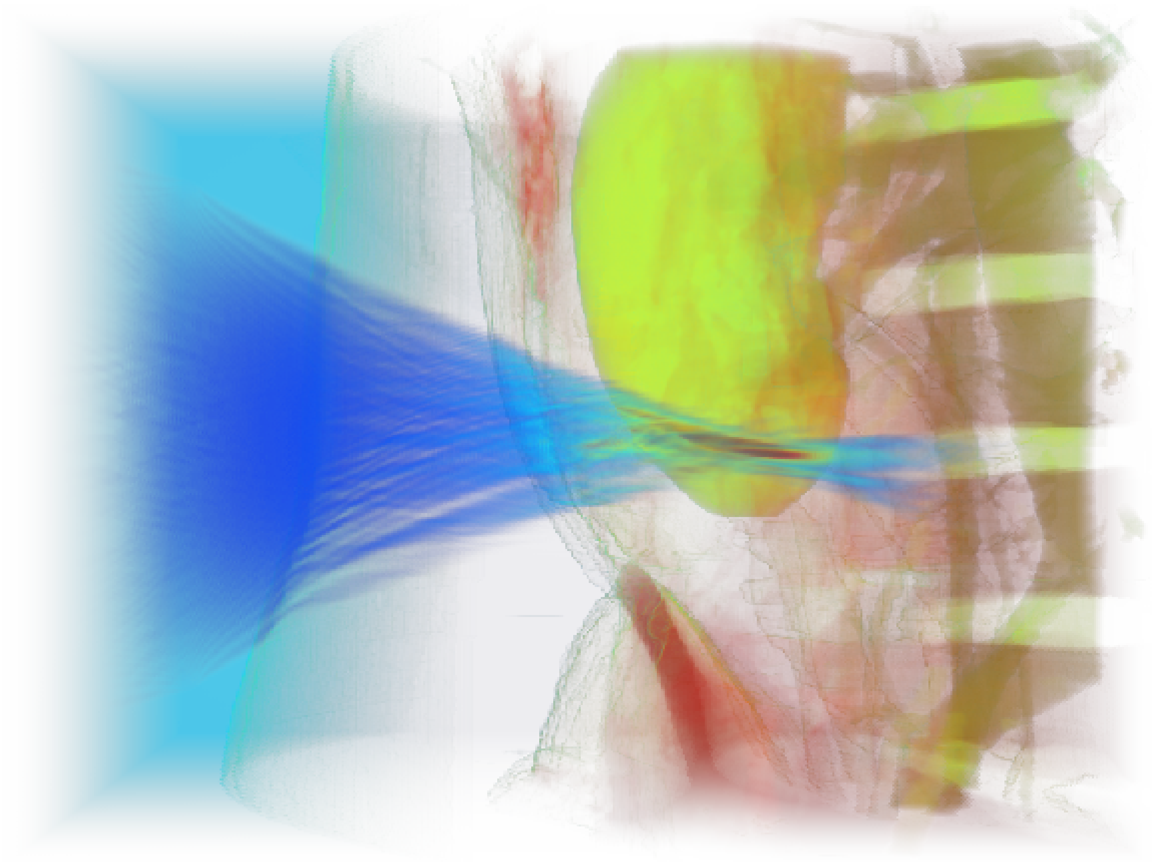


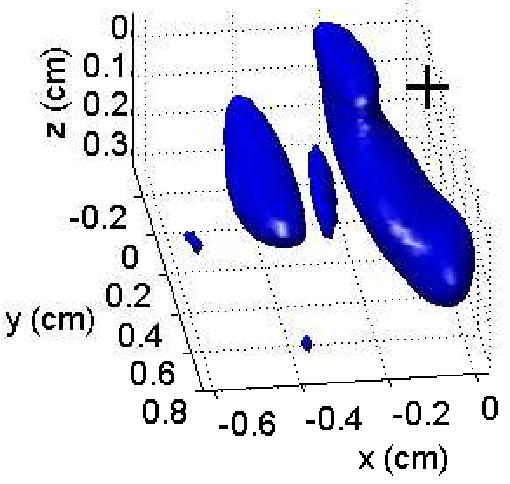
Figure 2.7: Example of volume rendered HIFU simulation output data[10]. The blue cone represents the maximum acoustic pressure, other colors visualize different human tissues. For example, the kidney is yellow.

[10]The illustration taken from [40].

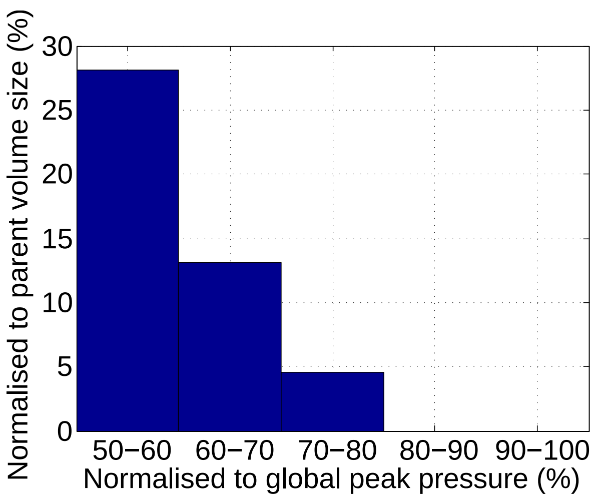
2.6 Application of simulation outputs

The applications of outputs are still in the development stage but can be, for example, treatment planning, exposimetry, and medical equipment design. The output values are also compared to the values measured from laboratory experiments, that is, in pure water [72].

One of the few published texts [72] concerns the use of a nonlinear ultrasound simulation model to study the effect of attenuation, refraction, and reflection due to different types of tissue on HIFU kidney therapy. Attenuation and beam splitting due to refraction were found to be the most significant factors that reduce the intensity of the ultrasound field (see Figure 2.8). Reflections from the rib cage could possibly cause significant losses, but this can be avoided by optimal positioning of the transducer. The reflections due to tissue interfaces were found to be negligible.



(a)



(b)

Figure 2.8: (a) The target focal point is marked with a black cross. It is possible to see the shifting and splitting of the focal point into one parent and child focal volumes. (b) Histogram showing the size of the child volumes relative to the parent focal volume[11].

Other results [73] show that the efficacy of HIFU therapy in the kidney could be improved with aberration correction. A method is proposed by which patient specific pretreatment calculations could be used to overcome the aberration and, therefore, make ultrasound treatment possible.

The next possible application is a “Convergence testing of a *k*-space pseudospectral scheme for transcranial time-reversal focusing” [34]. The HIFU technology is used for the treatment of essential tremors by ablation of the thalamus. The skull is an important obstacle to efficient transcranial transmission of ultrasound. Ultrasound simulations are used for time-reversal focusing.

Early-stage prostate cancer is often treated using external beam radiation therapy. This procedure usually involves implanting a small number of gold fiducial markers into the prostate to verify the position of the prostate gland between treatments. The objective

[11]The illustration taken from [72].

of the work “Modelling beam distortion during focused ultrasound surgery in the prostate using k-Wave” [24] was to systematically investigate, using computational simulations, how the fiducial markers affect the delivery of HIFU treatment (see Figure 2.9).

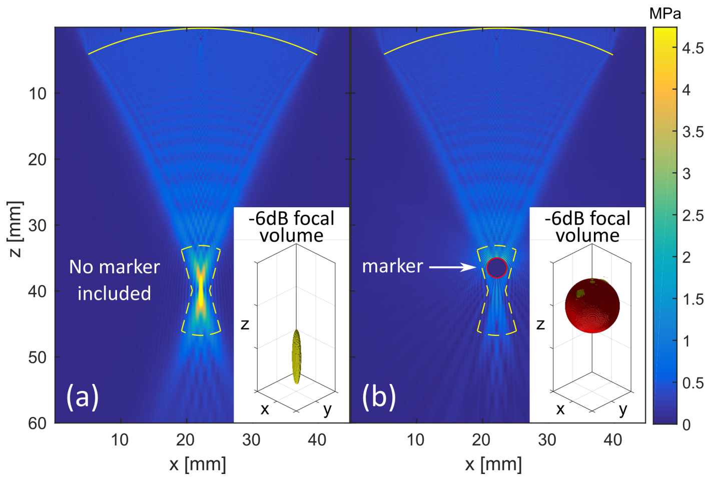


Figure 2.9: Maximum pressure field (b) due to the presence of a single spherical marker compared to (a) the a homogeneus simulation without a marker[12].

Another publication that deals with the topic [7] experimentally investigates the effect of a single external beam radiation therapy (EBRT) fiducial marker on the efficacy of HIFU treatment delivery using a tissue-mimicking material (TMM).

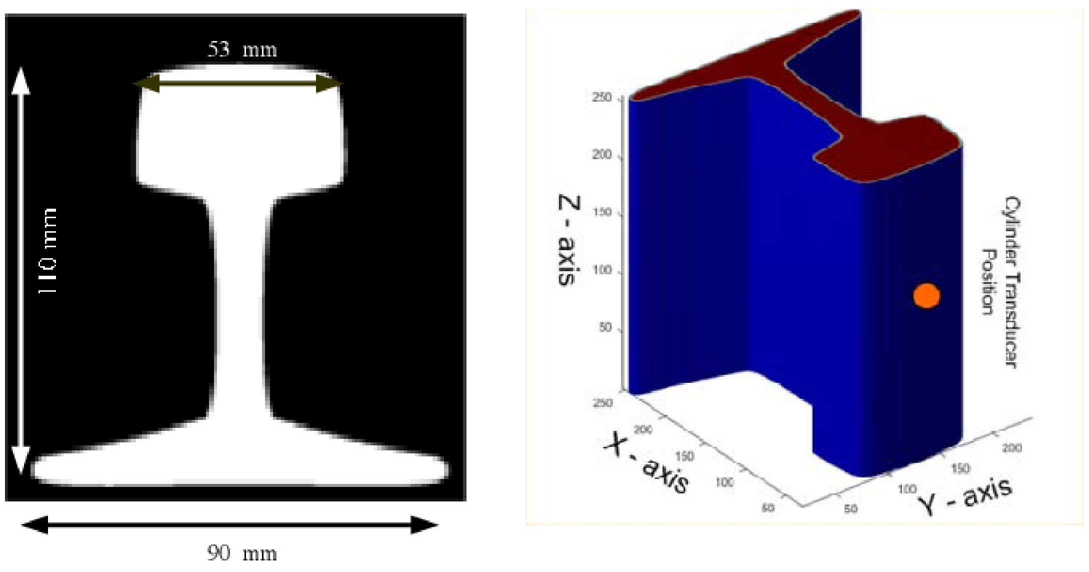
Another study deals with transurethral ultrasound therapy and uses ultrasound simulations to find out how prostatic calcifications affect therapeutic efficacy and to identify the best sonication strategy when calcifications are present [74].

Significant applications of simulation outputs that are not directly related to HIFU are in the field of UST. E.g., approach based on a second-order iterative minimization of the difference between the measurements and a model based on a ray approximation to the heterogeneous Green’s function [36] or also applications in the field of photoacoustic microscopy [69, 70].

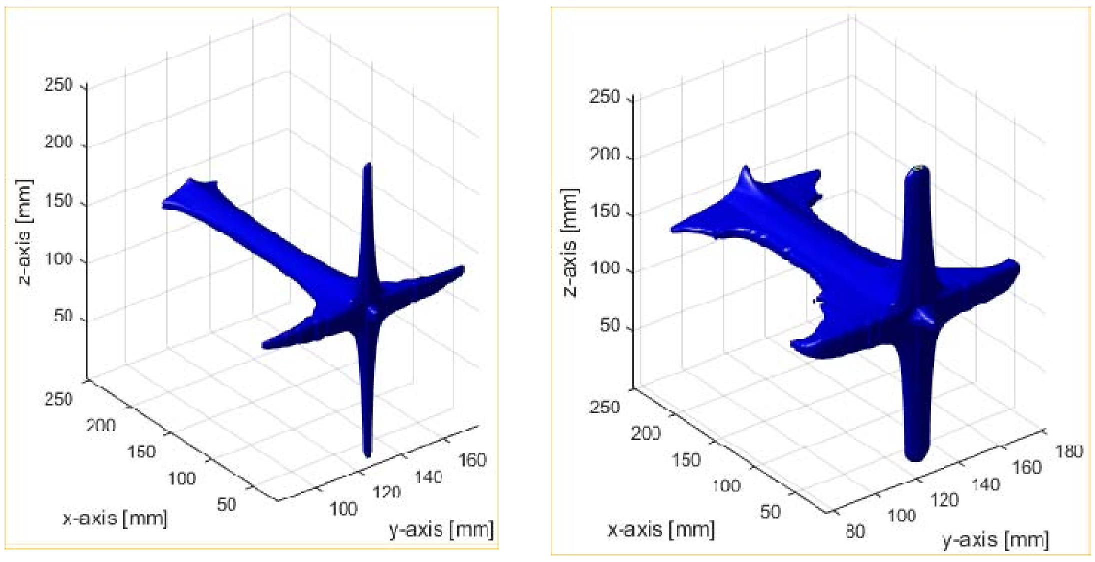
Ultrasonic simulations are also used in, for example, the inspection of railway track infrastructure. An inspection based on nondestructive testing was developed. Ultrasonic testing is one powerful method for nondestructive testing of internal railway crack, e.g. detect internal defects. It is possible to simulate nondestructive testing processes that are used to determine the parameters of ultrasonic testing [75](see Figure 2.10).

An interesting application of the outputs of HIFU simulations that are not directly related to the k-Wave toolbox is an alternative solution to performing many simulations - replacing with a surrogate model built from a database of simulation results. An interpolator is used, which enables the generation of results in near real time for configurations covered by the database range. This procedure and related tools have been implemented in CIVA-HealthCare [13].

[12]The illustration taken from [24].



(a)



(b)

Figure 2.10: (a) The cross section and 3-D computation domain of railway geometry. (b) The 3-D beam pattern of RMS pressure for selected iteration steps for the value of pressure ≥ 3 Pa[13].

[13]The illustration taken from [75].

2.7 Technologies

The implementation of the discrete equations was originally written as an open-source k-Wave acoustic toolbox for MATLAB. C++ simulation codes written as an extension to this toolbox try to maximize computational performance for large simulations. The k-Wave toolbox can be used for general time-domain acoustic and ultrasound simulations in complex and tissue-realistic media. Among other things, the toolbox also includes the option to use the forward model as a flexible time-reversal image reconstruction algorithm for photoacoustic tomography with an arbitrary measurement surface and includes a fast, one-step photoacoustic image reconstruction algorithm for recorded data. The HDF5 library, which also has a serial and parallel version (MPI-I/O) is used for storing simulation outputs and inputs. There are 4 basic versions of implementations [78, 79]:

- The MATLAB version - it is not suitable for large simulations, it is not accelerated, but it contains the most features, e.g. includes functions for thermal simulations.
- The C++ OpenMP central processing unit (CPU) version - accelerated version suitable for larger simulations. The feasibility of the simulation is determined by the amount of available operating memory, and the simulation speed is mainly influenced by the performance of the processor cores.
- The C++ GPU version - using CUDA to compute accelerated simulations on NVIDIA graphics cards. There is a significant acceleration of the simulation calculation compared to the OpenMP version, but the main limitation is the small amount of memory on the GPU

Versions that have not yet been officially released are also being developed:

- The C++ MPI CPU version - used for the largest simulations, is written in C++ with MPI, FFTW[14], single instruction, multiple data (SIMD) and is executed on distributed clusters. Storing is performed via a parallel I/O module based on the HDF5 library and Luster file system. The unavailability of large powerful computing clusters for ordinary users may be limiting here [33].
- The C++ GPU version using the local Fourier basis decomposition - implementation of the approach where global all-to-all FFT communications are replaced by direct neighbor exchanges. At the expense of a slight reduction in accuracy, communication can be significantly reduced. [35, 77, 83]

In general, the goal is to use the most modern technologies using parallel computing to enable the fastest possible simulations.

[14] www.fftw.org

Chapter 3

Compression methods

Since the goal of this work is to create a compression method for ultrasound simulation data, this chapter is focused on the data compression methods which are most relevant to the character of ultrasound simulation outputs. The point of the chapter is the introduction of methods that could be used to create a benchmark. First, a general introduction to the topic of data compression is presented. Then, the state-of-the-art compression methods of binary executables, natural language text, images, video, and audio are mentioned. Next, model-based coding is discussed. Some current papers and methods are described, which are oriented to 3-D time-varying data. More relevant information is obtained from two dissertations about ultrasonic signals compression. Finally, some methods from the HPC field in connection with big data compression are presented.

3.1 Data compression

In brief, data compression is the art or science of representing information in a compact form. These compact representations are created by identifying and using the structures that exist in the data. The compression algorithm takes an input X and generates a representation X_c that requires fewer bits, and there is a reconstruction algorithm that operates on the compressed representation X_c to generate the reconstruction Y. Based on reconstruction requirements, data compression schemes can be divided into two broad classes: lossless compression schemes, in which Y is identical to X, and lossy compression schemes, which generally provide much higher compression ratio than lossless compression, but allow Y to be different from X. Depending on the quality required from the reconstructed signal, varying amounts of loss of information about the value of each sample can be tolerated [67].

A compression algorithm can be measured in different ways - e.g. relative complexity, required memory, speed, compression ratio. In lossy compression, there needs to be some way to quantify the difference - often called distortion. In the compression of speech and video, the final quality arbitrator is the human [67]. In the case of ultrasound signal compression, the evaluation measures depend on the methods used by physicians or scientists (more about measures in Chapter 4).

The development of data compression algorithms for a variety of data can be divided into two phases - modeling (extraction of information about any redundancy and creation of a model) and coding (a description of the model and how the data differ from the model are encoded) [67].

The principle of compression (especially lossless) is based on information theory (Shannon, probability, self-information, entropy) which are not described here due to lack of space. Many approaches are based on this theory - Huffman coding, arithmetic coding, dictionary techniques and their applications. With lossy compression some possibilities to reduce data e.g. with vector and scalar quantization or differential encoding exist. Compression of multimedia signals is often connected with transform coding (fast Fourier transforms (FFT), discrete cosine transform (DCT), discrete wavelet transform (DWT)), which transforms the input data into a form more suited for compression (usually lossy) [8, 9, 67].

3.2 State-of-the-art compression methods

Generally, it is challenging to find and check every compression method/tool. Usually, for many different types of data, there are different sophisticated compression methods. All data about compression quality, compression ratios and computing performance, specifically for example memory consumption per signal sample in time, often cannot be read directly from the available information. To measure the exact memory requirement of each method, it is necessary to apply it to the required data, which is possible but very demanding. The most known and state-of-the-art methods for common types of signals are mentioned bellow.

Image compression - JPEG 2000

JPEG 2000[1] is an image coding system that uses state-of-the-art (lossy and lossless) compression techniques based on wavelet technology. It has a wide range of uses, from portable digital cameras to medical imaging, and other key sectors. Thus, this compression could be potentially applicable to the ultrasound simulations data. Of course, data compression in 2-D blocks, or its expansion into 3-D, is assumed here. Lossy compression of 3-D statistical shape and intensity models of femoral bones is an example where the JPEG 2000 image coding system is applied [44].

CMIX

CMIX[2] is a lossless data compression program aimed at optimizing the compression ratio at the cost of high CPU/memory usage. For new, this is the best known tool for large text compression[3]. CMIX uses three main components: preprocessing, model predictor (arithmetic coding), and context mixing. CMIX works with binary executables, natural text language, and images. The main disadvantages in connection with potential ultrasound simulation data compression are the preprocessing stage, where the complete input data must be transformed into another form and at least 32 GB of random-access memory (RAM).

OptimFROG

OptimFROG[4] is a lossless audio compression program. It is similar to ZIP compression, but it is highly specialized in compressing audio data. The compression ratios that can be obtained with OptimFROG are generally ranging between 25 % and 70 %. This method is

[1]http://jpeg.org/jpeg2000/index.html
[2]http://www.byronknoll.com/cmix.html
[3]http://mattmahoney.net/dc/text.html
[4]http://losslessaudio.org/

specialized for optimized high compression ratios at the expense of encoding and decoding speed. A comparison of CPU usage versus file size is shown in the Figure 3.1.

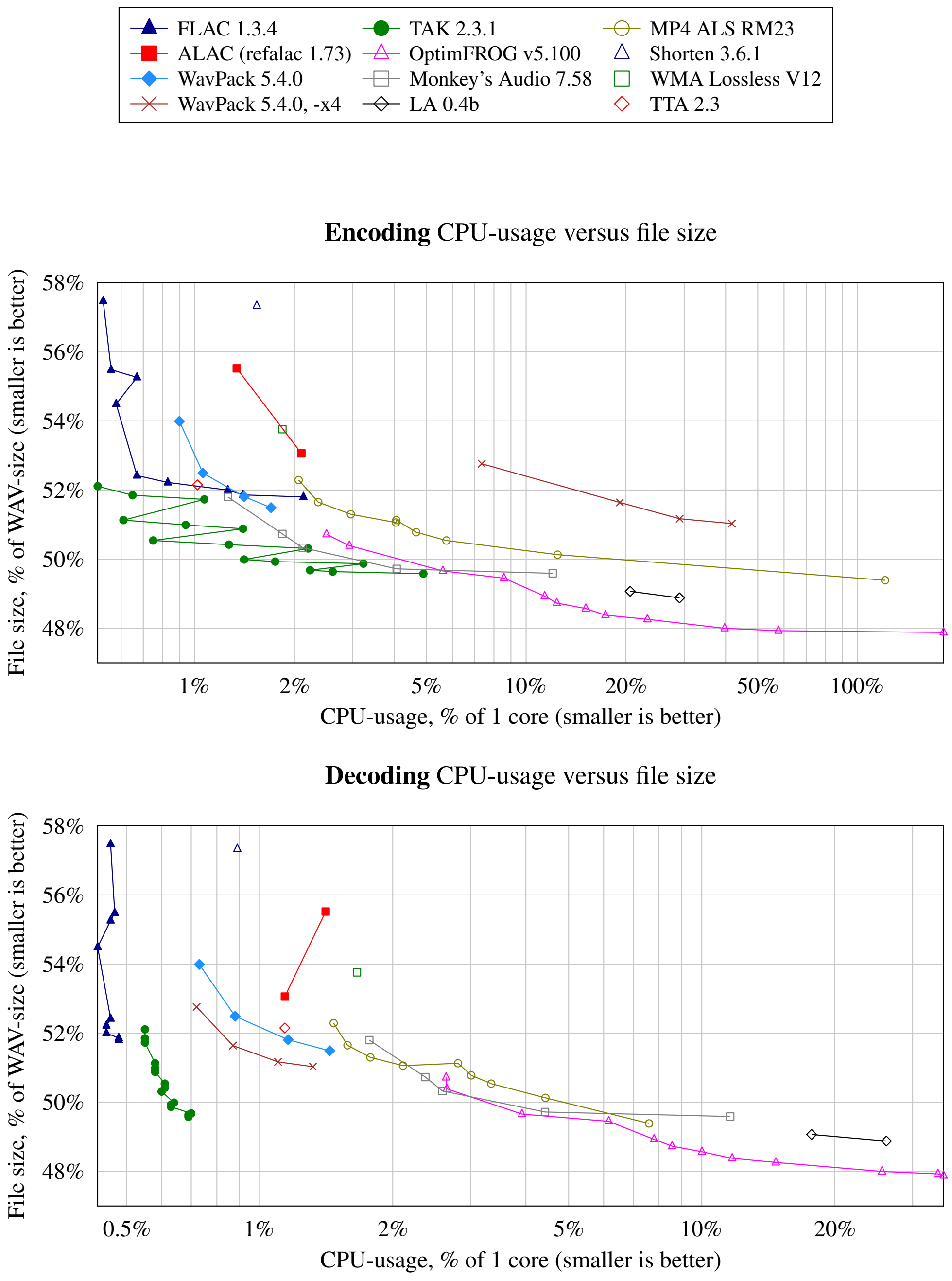


Figure 3.1: Lossless audio codec comparison. CPU-usage versus file size, average of all CDDA sources.[5].

[5]These graphs are part of a report which can be found on http://www.audiograaf.nl/losslesstest/Lossless%20audio%20codec%20comparison%20-%20revision%205%20-%20cdda.html.

Audio compression

In addition to the already mentioned OptimFROG, which is among the codecs with the highest lossless compression, plenty of coding methods have been developed to deal with the compression of audio and speech signals. Motion picture experts group 1 (MPEG-1) or MPEG-2 audio layer III (MP3), MPEG-2 advanced audio coding (AAC), Ogg Vorbis, Windows media audio (WMA) are the most known lossy audio compression technologies. Lossy compression methods provide higher compression ratios at the cost of fidelity. These methods rely on human psychoacoustic models to reduce the fidelity of less audible sounds. In contrast to this, lossless methods produce a representation of the input signal that decompresses to an exact duplicate of the original signal. However, the compression ratios are around 50–75 % of the original size. There is a possibility to apply some of these techniques to ultrasound simulation data, but general audio signals differ from ultrasound simulation signals - e.g., by number of harmonic frequencies (see Chapter 2). Furthermore, since the existing audio coding methods were not designed for use in extremely memory-limited environments, they consume the signal samples in frames typically containing thousands of samples. The size of these frames directly affects RAM consumption. These high memory requirements probably bring disadvantages for use in acoustic simulations. The following is a description of several methods, implemented in the FFmpeg[6] framework, that could be used for comparison with possible ultrasound signal compression [10, 19].

AAC is an international standard for lossy audio compression. It was designed to be the successor of the ubiquitous MP3 format. Because AAC generally achieves better sound quality than MP3 at the same bit rate. The compression process can be described as follows. First, a modified discrete cosine transform (MDCT) based filter bank is used to decompose the input signal into multiple resolutions. Afterward, a psychoacoustic model is used in the quantization stage to minimize the audible distortion. The time-domain prediction can be employed to take advantage of correlations between multiple resolutions. The format supports arbitrary bit rates. The minimum frame size is 1024 samples.

Dolby AC-3 (also Dolby digital, ATSC A/52) is a lossy audio compression standard developed by Dolby Laboratories. The decoder has the ability to reproduce various channel configurations from 1 to 5.1 from the common bitstream. During compression, the input signal is grouped into blocks of 512 samples. However, depending on the nature of the signal, an appropriate length of the subsequent MDCT-based filter bank is selected. The format supports selected bit rates ranging between 32 and 640 kbit s^{-1}. The minimum frame size is 1536 samples.

Opus is an open lossy audio compression standard developed by the Xiph.Org Foundation. It is the successor of the older Vorbis and Speex methods. The codec is composed of a layer based on linear prediction and a layer based on the MDCT, but both layers can be used at the same time (hybrid mode). Opus supports all bit rates from 6 to 510 kbit s^{-1}. The minimum frame size is 120 samples. This compression requires input values in 16-bit PCM format.

ADPCM is a lossy compression format with a single fixed compression ratio of approximately 4:1. ADPCM compression works by separating the input signal into blocks and predicting its samples on the basis of the previous sample. The predicted value is then adaptively quantized and encoded in the nibbles, giving rise to the 4:1 compression ratio (192 kbit s^{-1} for 48 kHz sampling rate). A psychoacoustic model is not used at all. The minimum frame size is 2036 samples.

[6] https://ffmpeg.org/

FLAC is an open lossless audio format now covered by the Xiph.org Foundation. FLAC uses linear prediction followed by coding of the residual (Golomb-Rice coding). For music, various benchmarks reported that FLAC generally reduces the size of the input signal size to 50–75 % of the original size. The minimum frame size is 16 samples, which is not so much, but on the other hand, the compression ratios are not very high.

ALAC is the open lossless audio format by Apple. Similarly to FLAC, ALAC uses linear prediction with Golomb-Rice coding of the residual. Furthermore, the achievable compression performance is similar to that of FLAC. The minimum frame size is 4096 samples, which is very high.

As mentioned earlier, the typical size of the ultrasound data to be compressed reaches about 0.5 MHz. The frame sizes can be set to the minimum supported size of a given compression method, but still typically require thousands of bytes of memory for one processed signal at a time. In conjunction with the previous point, such large frame sizes make the common audio codecs unusable for compression of the ultrasound data. Another issue to deal with is the sampling rate. Because the human hearing range is commonly given as 20 to 20,000 Hz, the common audio codecs were designed to process signals of comparable rates (e.g., 44.1, 48, 96, or 192 kHz). However, ultrasound signals are acquired (simulated) at a much higher sampling rate. Another problem is that many audio compression techniques require a normalized signal at the input, for example, to the $\langle 0, 1 \rangle$ interval. The outputs of ultrasound simulations are difficult to normalize, as the range of values is not known in advance.

3.3 Model-based coding

The idea of a model-based compression coding scheme is to characterize the source data in terms of some strong underlying model. Thus, some model parameters exist that define our signal [9]. It can be combined with transform coding - models created with transformed form of input signal.

A typical example of the model-based approach is presented in a paper named "Model-based filtering, compression and classification of the ECG" [17]. It is based on a realistic nonlinear electrocardiogram model (ECG) to account for T-wave asymmetry. The fitting procedure using a nonlinear optimization allows filtering (removing noise), efficient compression, and classification of the ECG. Compression based on the ECG signal model based on the hybridization technique is also addressed in [2].

There are also techniques for model-based coding of 3-D head sequences [25]. 3-D frames are analyzed and registered using a 3-D face model. The result is efficient compression.

In the voice codecs section, a model-based coding algorithm of MDCT spectral coefficients is used[71] in AAC, Dolby AC-3 and Opus.

Model-based compression scheme for seismic data models seismic traces as multitone sinusoidal waves superposition. Sinusoidal waves are represented by a set of distinct parameters. Here, a parameter estimation algorithm takes place. The performance of the method was shown to be better than that of linear predictive coding (LPC) and distributed principal component analysis (DPCA) [56].

An evaluation of model-based approaches to sensor data compression is presented [29]. This publication provides in-depth analysis of the benchmark results, obtained using 11 different real data sets consisting of 346 heterogeneous sensor data signals (see Figure 3.2). Compression approaches are classified into 4 categories: constant models, linear models, nonlinear models, and correlation models. The constant models category includes, e.g., piecewise aggregate approximation (PAA), piecewise constant approximation (PCA), adaptive piecewise constant approximation (APCA), and piecewise constant histogram (PCH). Linear models include, for example, piecewise linear approximation (PLA), piecewise linear histogram (PWLH), sliding window and bottom-up (SWAB), lightweight temporal compression (LTC), and slide/swing filters (SF). The category of nonlinear models includes Chebyshev approximations (CHEB). Correlation models include self-based regression (SBR), robust information-driven data compression architecture RIDA, and grouping and amplitude scaling (GAMPS). Six approaches were selected for experimental evaluation: PCA, APCA, PWLH, SF, CHEB, and GAMPS. The datasets contain various physical variables, e.g. moisture, humidity, voltage, radiation, or pressure. The interesting thing is that the compression ratio with the error tolerance set to 1 and 5 % is usually significantly better for the pressure dataset than for the other signals. From this it can be concluded that model-based coding could also be suitable for our HIFU simulation signals.

dataset (abbreviation)	number of signals	total number of readings	sampling rate (avg.)	min	max	mean	standard deviation	short-term fluctuation
moisture (mois)	20	442,313	300 sec.	16.18	30.50	18.20	1.09	medium
pressure (pres)	14	161,004	2 sec.	18.20	796.90	268.43	222.96	low
humidity (humi)	34	2,031,732	60 sec.	19.88	93.02	72.89	14.27	meium
voltage (volt)	16	523,877	600 sec.	3.29	3.49	3.41	0.05	medium
lysimeter (lysi)	24	232,746	300 sec.	0	3.81	0.11	0.35	medium
snow-height (snow)	29	349,557	600 sec.	-662.10	370.00	-18.17	155.00	low
temperature (temp)	78	4,846,162	60 sec.	-29.76	49.05	-4.57	9.78	medium
CO_2 (CO_2)	16	3,088,233	16 sec.	0	2000	558.00	219.4	high
radiation (radi)	34	3,955,268	600 sec.	137.5	798.40	261.02	46.86	medium
wind-speed (wspd)	46	2,376,156	60 sec.	0	14.37	2.84	2.31	high
wind-direction (wdir)	35	2,084,943	60 sec.	1.79	357.47	232.65	84.40	high

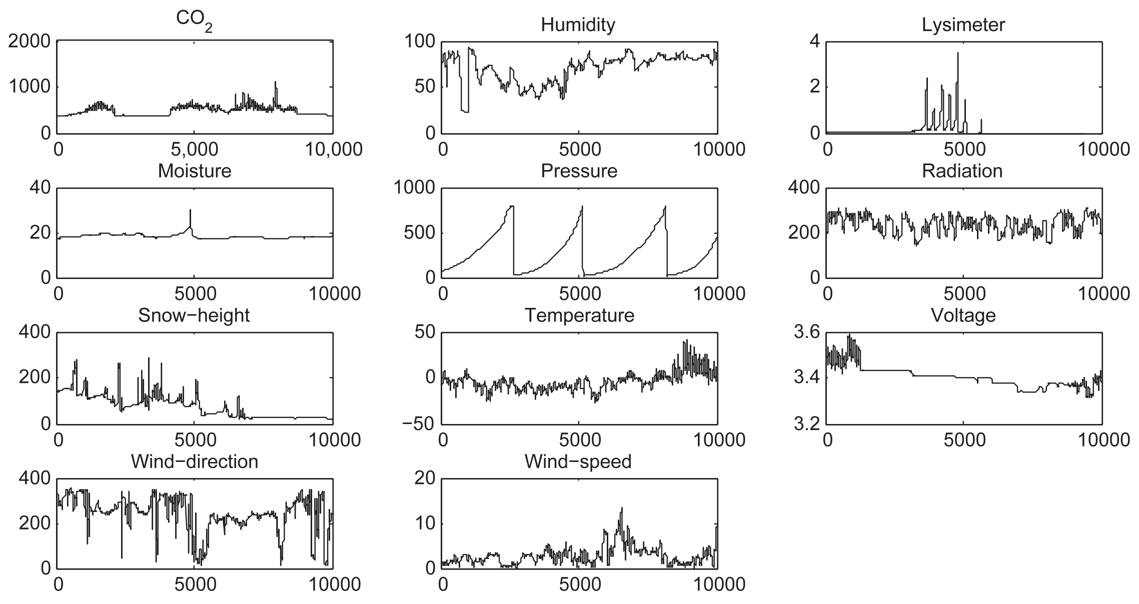


Figure 3.2: Real data sets used in the benchmark[7].

[7]The illustration taken from [29].

3.4 3-D time-varying data compression

The next point of view, which is connected to the topic compression and ultrasound simulation data, is 3-D time-varying data compression. Some works in the medical data imaging section, which are focused on 4-D (usually volume) medical data, GPU, visualization and also compression. None of the bellow mentioned techniques is focused directly on ultrasound.

For example, there is a framework for 4-D medical data compression architecture [87]. It is based on the detection of spatial and time redundancy in recorded 4-D medical data. The motion that produces input parameters for the neural network is analyzed. It is a combination of segmentation, block matching, motion field prediction, and wavelet packets. However, this framework is designed for general medical data and from the perspective of parallel processing, memory requirements and large-scale simulations it is not suitable for large ultrasound simulation data.

Compilation is usually related to rendering (out-of-core streaming techniques, GPU). Some streaming compression techniques have been developed for interactive visualization of time-varying volume data [62]. The rendering scheme combines a temporal prediction model and variable-length coding with a fast block compression algorithm. However, similar to other methods, 4-D data must be stored somewhere and are not applicable to on-the-fly ultrasound simulations.

For large-scale scientific simulations, which typically run on a cluster of CPU and the time steps are stored in the file system, compression of floating-point values was proposed [54]. Most of the compression schemes are designed to operate offline, but this proposed lossless scheme works online and achieves state-of-the-art compression rates and speeds. This method is also included in the HPC compression category. It is probably possible to apply these methods to ultrasound simulations as well, but the average compression rates of around 30 % that the method achieves may not be so beneficial for the simulations.

Another important thing is a fast decompression process. A compression scheme for large-scale time-varying volume data using spatio-temporal features with low-cost and fast decompression process was proposed [88]. This compression scheme contains two compression processes: spatial domain compression and temporal domain compression, which utilize spatial features and temporal features and are also connected to the fast rendering process (specialized particle-based volume rendering (PBVR)). However, this case of compression and decompression probably is not designed to work with really large-scale datasets. The presented experiments were carried out with only about 19 GB of data for all time steps and are not focused on on-the-fly parallel processing with a cluster of CPUs.

3.5 Ultrasound signals compression

Some works have been developed focused directly on the ultrasound signals compression. Many techniques have been applied to process 1-D ultrasonic signals (e.g., matching pursuit, 1-D discrete cosine transform (DCT), Walsh-Hadamard transform (WHT)) [1, 9]. They specialize in the topics of general ultrasonic imaging [11, 39, 64], 3-D ultrasound computed tomography (USCT) [66] or fetal Doppler ultrasound audio signals [76].

The last publication [76] is older, but the character of Doppler ultrasound signals is similar to HIFU simulation signals. A method was proposed to reduce the data rate for transmitting signals to cardiotocographs. The method involves splitting the signal into amplitude and frequency components. The amplitude is represented by samples of the signal envelope, and the frequency information is represented by the number of zero-crossings within fixed intervals (windows). They claim that their method can be used to reproduce the signal with no audible difference compared to the original waveform. A 15: 1 reduction in the data rate is achieved. However, the purpose of Doppler ultrasound signals (e.g. to determine the fetal heart) is different from HIFU signals and better precision than "no audiable difference" is needed for HIFU simulations (Figure 3.3).

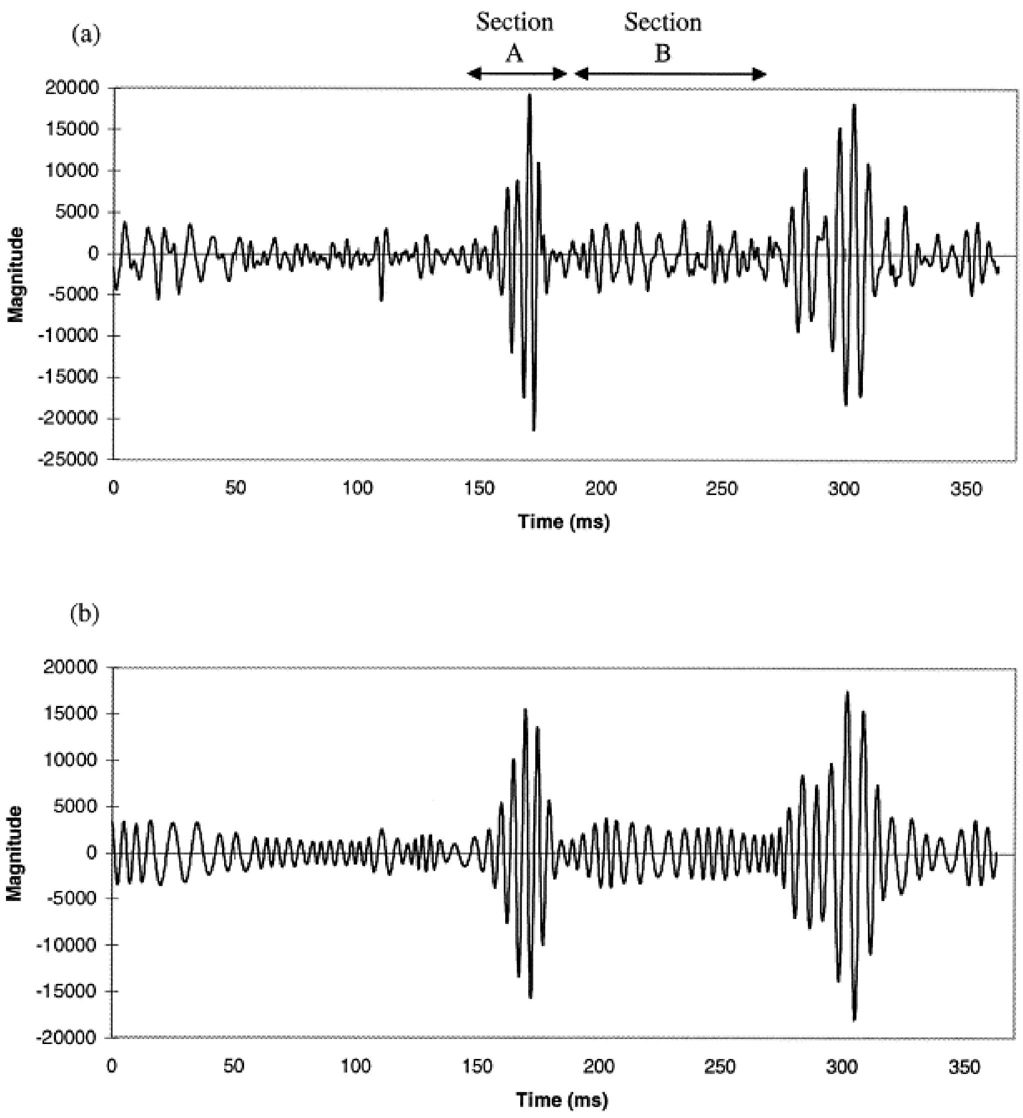


Figure 3.3: A fetal Doppler ultrasound audio signal shown over one heart cycle: (a) before compression; (b) after reconstruction using 5 ms windows and zero-crossing averaging[8].

The dissertation "Compression, estimation, and analysis of ultrasonic signals" [11] deals with different signal processing techniques to compress and denoise ultrasonic signals. The focus is on ultrasonic imaging, so the processing of data gained by transmitting acoustic waves into the specimen using an ultrasound transducer. The compression of this type of

[8]The illustration taken from [76].

data is also handled, for example, by publication [39]. Other work addresses this problem using deep neural networks [64]. The type of data in ultrasonic imaging is different from HIFU.

The other dissertation named "Data Compression in Ultrasound Computed Tomography" [66] is focused on the reduction of 3-D USCT data. USCT is developed at the Karlsruhe Institute of Technology aiming at a new medical imaging system for early detection of breast cancer. A discrete wavelet based data compression method at a compression ratio of 15 was suggested for compression of USCT datasets.

There are also other compression algorithms for radio frequency (RF) ultrasound signals [1, 15, 37]. These images are compressed after analog to digital conversion and before beamforming. The most used techniques are peak gates (PG), trace compression (TC), largest variation (LAVA), linear predictive coding (LPC), and also the well-known ZIP, JPEG or MPEG. In context with HIFU simulation signals, these data have some similar properties, i.e., temporal redundancy between adjacent frames.

In the area of GPU-based simulations, a paper focused on acceleration of GPU-based ultrasound simulation via data compression [27] was published. They have demonstrated a speedup of 1.5 times on a simulation that compresses single-precision floating-point values into 3 bytes. Unfortunately, this approach was developed only for small two-dimensional (2-D) simulations on GPU.

3.6 High-performance computing data compression

With exascale computing era, big problems are approaching - transferring and storing really big data. It is quite challenging to design a generic error-bounded lossy compressor with a very high compression ratio for HPC applications [19].

There is some comparison of the most known lossy and lossless compression schemes [68]. The lossless compression schemes such like FPC, ISOBAR, PRIMACY, ALACRITY, CC or IOFSL usually have the maximum achievable compression ratio just above 2×. Some of them use well-known algorithms like zlib or gzip. The typical lossy compression schemes for HPC are ISABELA, FPZIP or APAX. They use approximation algorithms similar to B-spline on sorting data or Lorenzo predictor with mapping values to integers and encoding residuals. The typical compression ratio, for example, for ISABELA is up to 5× with error less than 1 %. With lossy compressions several future challenges remain - e.g reducing the memory requirements to perform in-situ compression and performing compression in parallel.

ZFP and FPZIP are floating-point compressors developed by Peter Lindstrom et al.[9] These libraries were designed for compressing 1-D, 2-D or 3-D arrays of floating-point precision numbers. FPZIP is the floating-point analogue to PNG image compression, and ZFP is an advanced JPEG for floating-point arrays. ZFP [53] is lossy but optionally error-bounded compression, FPZIP [54] is a (older) library for lossless or lossy compression of 2-D or 3-D floating-point scalar fields.

SZ (Squeeze) [19] is a lossy compression scheme with strictly bounded errors and low overheads. In an extreme-scale use case, experiments show that the compression ratio (3.3–436) of SZ exceeds that of ZFP by 80 %. The key idea is to check each data point to see if it can be approximated by some bestfit curve fitting model and replace it by using a two-bit code indicating the model type if the approximation is within user-specified error bound.

[9] http://computation.llnl.gov/projects/floating-point-compression

Another example of a research area where it is necessary to solve data compression for HPC is astronomy. Research astronomy facilities can generate terabytes of data per day at a constant rate. Lossless compression algorithms were developed for these astronomical radio data, which reduce their size to 28 % of its original size and decompression with a throughput greater than $1\,\mathrm{GB\,s^{-1}}$ on a single core [61].

An example of an efficient compression algorithm that falls into the category of HPC, simulations, and also into Section 3.4 (3-D time-varying data compression) is named HLR-compress [50]. It is spatial data compression. It is based on hierarchical low-rank HLR methods combined with floating-point number compression schemes. On average, a greater than 100-fold compression of the original size of the datasets is achieved.

A similar lossy compression method, based on the theory of multigrid methods, is multigrid adaptive reduction of data (MGARD) [3, 14]. A specific feature of this method is the provision of guaranteed, computable bounds on the loss incurred by the reduction of the data.

Some of the methods from that section could be applied offline to ultrasound simulation outputs (ISABELA, ZFP, FPZIP); here is the question of compression error with regard to the application of these data. Other methods could also be applied to the data during the simulation calculation, e.g. MGARD or HLRcompres. However, these methods are focused on spatial compression.

Chapter 4

Evaluation measures

This chapter focuses mainly on compression quality metrics and evaluation methods. In general, compression evaluation is a complex problem due to the lack of measures to universally evaluate compression methods. It is hard to claim that one compression algorithm is better than the other. The first one can have a higher compression ratio with a smaller error, but the second one can be faster and less memory-intensive. Another thing is utilization of the compressed data, e.g. an audio stream vs. an image. For some cases, mathematical metrics can be used, but especially video, audio, and image compression quality measurements have to be tested on humans - algorithms are based on auditory perception in a human visual system [67]. Thus, it is necessary to set some evaluation methods for our case of data compression, including, for example, in terms of the application of outputs or the consumption of computing resources.

4.1 General measures

Several basic general evaluation measures exist [67]. A very basic and very logical way to measure how well a compression algorithm compresses is to look at the ratio of the number of bits required to represent the data before compression to the number of bits required to represent the data after compression - *compression ratio*. The compression ratio can also be expressed as a percentage, i.e. the reduction of the amount of data as a percentage of the size of the original data. Another measure is *rate* - e.g. the average number of bits required to represent a single sample.

Several ways to quantify the difference between the original and compressed signal in lossy compression exists - often called the *distortion*. Lossy techniques are generally used for the compression of data that originate as analog signals, such as speech and video. Here, the final arbiter of quality is a human.

Two popular measures of distortion or difference between the original and reconstructed sequences are the squared error measure and the absolute difference measure. These are called *difference distortion measures*. If $\{x_n\}$ is the source output and $\{y_n\}$ is the reconstructed sequence, then the squared error measure is given by

$$d_{\mathrm{se}}(x_n, y_n) = (x_n - y_n)^2 \tag{4.1}$$

and the absolute difference measure is given by

$$d_{\mathrm{ad}}(x_n, y_n) = |x_n - y_n|. \tag{4.2}$$

Several average measures are used to summarize the information in the difference sequence. The most used average measure is the average of the squared error measure - *mean squared error* (MSE). The mean squared error can be defined as

$$D = E\left\{\frac{1}{N}\sum_{n=1}^{N}(X_n - Y_n)^2\right\} = \frac{1}{N}\sum_{n=1}^{N}E\left[(X_n - Y_n)^2\right] \tag{4.3}$$

where the information sequence is a sequence of random variables $\{X_n\}$ and the reconstruction sequence is the sequence of random variables $\{Y_n\}$, for a sequence of length N, where the expectation is with respect to the joint distribution of X_n and Y_n. In practice (ergodic sequences), the ensemble averages in Equation (4.3) are replaced with the time averages to obtain

$$\sigma_d^2 = \frac{1}{N}\sum_{n=1}^{N}(x_n - y_n)^2. \tag{4.4}$$

Symbol σ_d^2 is used for MSE and implies that the variance of the distortion sequence $d(x_n, y_n)$ is equal to the second moment or that the distortion sequence is zero mean. For the size of the error relative to the signal, it is possible to calculate the ratio of the average squared value of the source output and the MSE - *signal-to-noise ratio* (SNR):

$$\text{SNR} = \frac{\sigma_x^2}{\sigma_d^2} \tag{4.5}$$

where σ_x^2 is the average squared value of the source output. The SNR can be measured on a logarithmic scale in decibels (dB) units. If the interest is in the size of the error relative to the peak value of the signal x_{peak}, it is possible to use *peak-signal-to-noise ratio* (PSNR, in dB):

$$\text{PSNR(dB)} = 10\log_{10}\frac{x_{\text{peak}}^2}{\sigma_d^2}. \tag{4.6}$$

A simple difference distortion measure, that is, the average of the absolute difference (based on the ℓ_1-norm [21]), is also often used:

$$d_1 = \frac{1}{N}\sum_{n=1}^{N}|x_n - y_n|. \tag{4.7}$$

To evaluate errors in some applications, the maximum value of the error magnitude may also be of interest (ℓ_∞-norm, ℓ_∞-error [21]):

$$d_\infty = \max_n |x_n - y_n| \tag{4.8}$$

or its relative version [6]:

$$\text{Relative } d_\infty = \frac{\max\limits_n |x_n - y_n|}{\max\limits_n (x_n)}. \tag{4.9}$$

And similarly the distortion measure based on ℓ_2-norm (ℓ_2-error) [6]:

$$d_2 = \sqrt{\sum_{n=1}^{N} (x_n - y_n)^2} \tag{4.10}$$

and its relative version:

$$\text{Relative } d_2 = \sqrt{\frac{\sum_{n=1}^{N} (x_n - y_n)^2}{\sum_{n=1}^{N} x_n^2}}. \tag{4.11}$$

For 2-D images compressed with JPEG or JPEG2000, structure similarity measure (SSIM) is often used. The average mutual information (AMI) and normalized mutual information (NMI) are widely used for image registration, the entropies are calculated to show the quality of distorted images. Other metrics are homogeneity-based measure or gradient vector flow [48, 66, 67].

4.2 Evaluating of compression quality of ultrasound simulation data

For our case of data compression, the quality of compression can be measured, for example, in three ways.

1. In the first case, the original and compressed data can be obtained and compared. It is possible to use known quality metrics (MSE, SNR, ℓ_∞ error, ...) and compare the measured numbers (e.g. the compression ratio) of the new method with other state-of-the-art compression methods designed for various data types (e.g. JPEG2000, AAC, CMIX, ISABELA, SZ, ZFP). In this case, many algorithms can be tested because the parallel environment where the simulations are performed and their applications are not taken into account

2. The next possible measurement is from the point of view of the application of the simulated data. It is meant as the utilization of the data by scientists or medical doctors - e.g. visualization, related calculations, more efficient treatment planning. It is possible to conduct experiments and measure how the compression method affects the quality of the application. It is possible to compare the application with and without compression. If a new compression can be shown to facilitate the work of doctors or scientific research, the work can be said to have a scientific benefit. Carrying out experiments is, however, demanding and complex, because the applications of the outputs are still in the development phase. For example, a comparison of temperature doses after thermal simulation with and without compression is offered here.

3. A third possible way of evaluating compression is in the area of computer simulation performance. That is, how the new compression method affects computing resources (RAM, a storage space, network bands, computing speed, the number of CPU cores within a supercomputer, GPU memory), e.g., from an economic point of view, if the target quality is set as satisfactory for applications and the goal is to compress data on-the-fly, during simulations. However, this measurement case depends on the previous case.

Some combinations of approaches between these measures can be tested and evaluated. It is ideal to carry out an evaluation using a combination of all three approaches. First, to perform a potential new method of compressing ultrasound simulation data with other methods that compress signals of a similar type, and then to evaluate this method from the point of view of application, comparing the quality of the outputs and the use of computing resources with and without compression. However, it is necessary to determine the acceptable quality of the simulation outputs and their application.

4.3 Acceptable errors in ultrasound simulation outputs

It is very difficult to determine which output quality is satisfactory and which is not. Unfortunately, there are almost no applications that determine specific and clear values of acceptable errors.

The publication from which these values can be derived deals with intercomparison of compressional wave models within transcranial ultrasound simulation [6]. Nine numerical benchmarks are defined for increasing geometric complexity (e.g. different types of bones, see Table 4.1 and Figures 4.1 and 4.2).

Table 4.1: Summary of benchmarks of the intercomparison. SC1 corresponds to the focused bowl transducer and SC2 to the plane piston transducer. Outputs are resampled to a regular Cartesian mesh with a grid spacing of 0.5 mm. Simulation layouts are shown in Figures 4.1 and 4.2), gp = grid points [6].

Label	Description	Output grid size
PH1-BM1-SC1/2	Water (lossless)	120×70mm (241×141 gp)
PH1-BM2-SC1/2	Water (artificial absorption of 1 dB/cm at 500 kHz)	120×70mm (241×141 gp)
PH1-BM3-SC1/2	Flat, single-layer skull (cortical bone) in water	120×70mm (241×141 gp)
PH1-BM4-SC1/2	Flat, skin, three-layered skull, and brain	120×70mm (241×141 gp)
PH1-BM5-SC1/2	Curved, single-layer skull (cortical bone) in water	120×70mm (241×141 gp)
PH1-BM6-SC1/2	Curved, skin, three-layered skull, and brain	120×70mm (241×141 gp)
PH1-BM7-SC1/2	Truncated skull mesh in water, target in visual cortex	120×70×70mm (241×141×141 gp)
PH1-BM8-SC1/2	Whole skull mesh, target in visual cortex	225×170×190mm (451×341×381 gp)
PH1-BM9-SC1/2	Whole skull mesh, target in motor cortex	212×224×184mm (425×449×369 gp)

Eleven different modeling tools are used to compare the results. Modeling tools include, in addition to the pseudospectral method, other numerical techniques, such as the finite difference method in the time domain, the angular spectrum method, the boundary element method, and the spectral element method. Difference metrics used for the intercomparison are defined in Equations (4.12) to (4.15):

$$\text{Relative } \ell_2 = \sqrt{\frac{\sum (p_1 - p_2)^2}{\sum p_1^2}}, \tag{4.12}$$

$$\text{Relative } \ell_\infty = \frac{\max|p_1 - p_2|}{\max(p_1)}, \tag{4.13}$$

$$\text{Focal (peak) pressure} = \frac{|\max(p_1) - \max(p_2)|}{\max(p_1)}, \tag{4.14}$$

$$\text{Focal position} = ||\text{pos } \max(p_1) - \text{pos } \max(p_2)||_2. \tag{4.15}$$

Here, p_1 and p_2 are amplitudes of the pressure in the 2-D or 3-D comparison domains, for reference and comparison field, resp. The sums and maximum values are over the entire domain. The focal values are taken from inside the brain region only [6].

When comparing the results in a cross-comparison, the mean values for each benchmark for the difference in focal pressure and position are less than 10 % and 1 mm, respectively. It can be read from the results that the values of difference in focal pressure are below 1 % only exceptionally, usually during simulations only in water, without bones, skin and other more complex properties of the medium. Similar are the results of relative ℓ_∞ and ℓ_2 errors in cases for the entire field (simulation domain). Based on the results of these benchmarks, where the errors of different modeling tools are evaluated, it can be concluded that the maximum acceptable relative errors caused by the application of potential compression should be in units of percent [6].

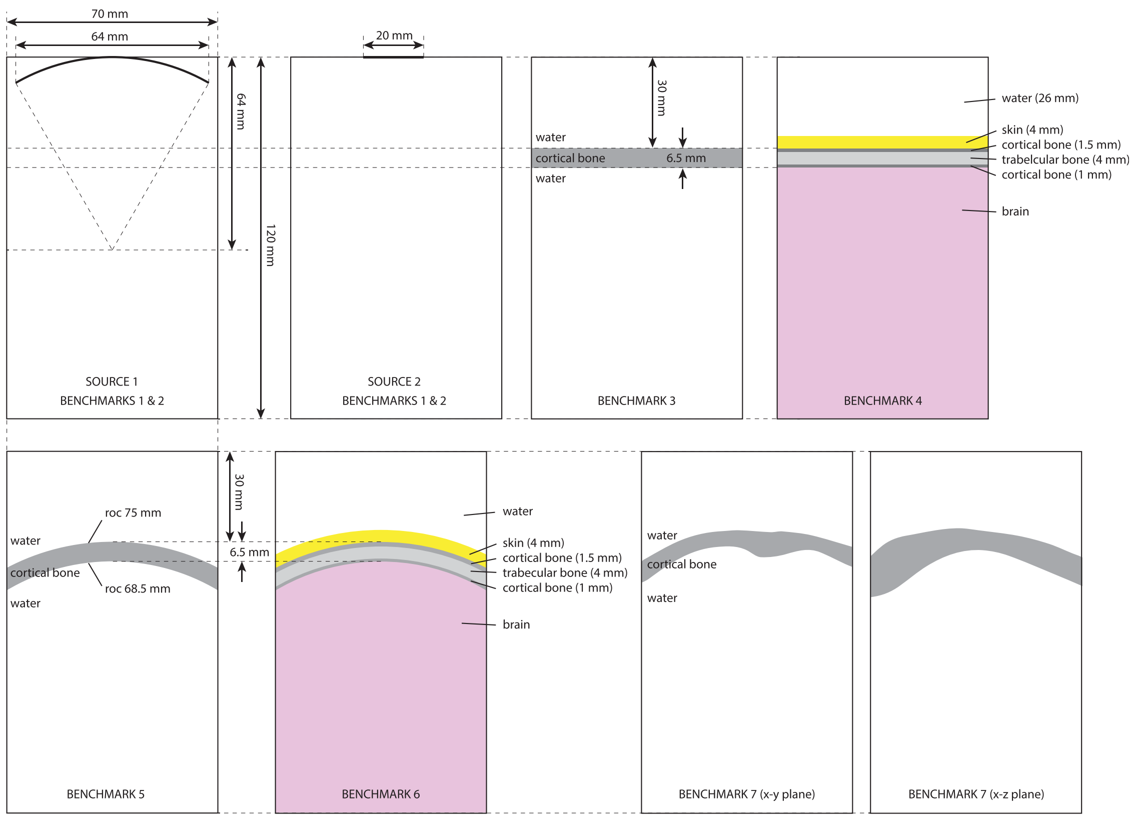


Figure 4.1: Transducer definitions and simulation layouts for benchmarks 1–7. Benchmarks 1–6 use a 2-D comparison domain, benchmark 7 uses a 3-D comparison domain (description in Figure 4.1)[1].

[1]The illustration taken from [6].

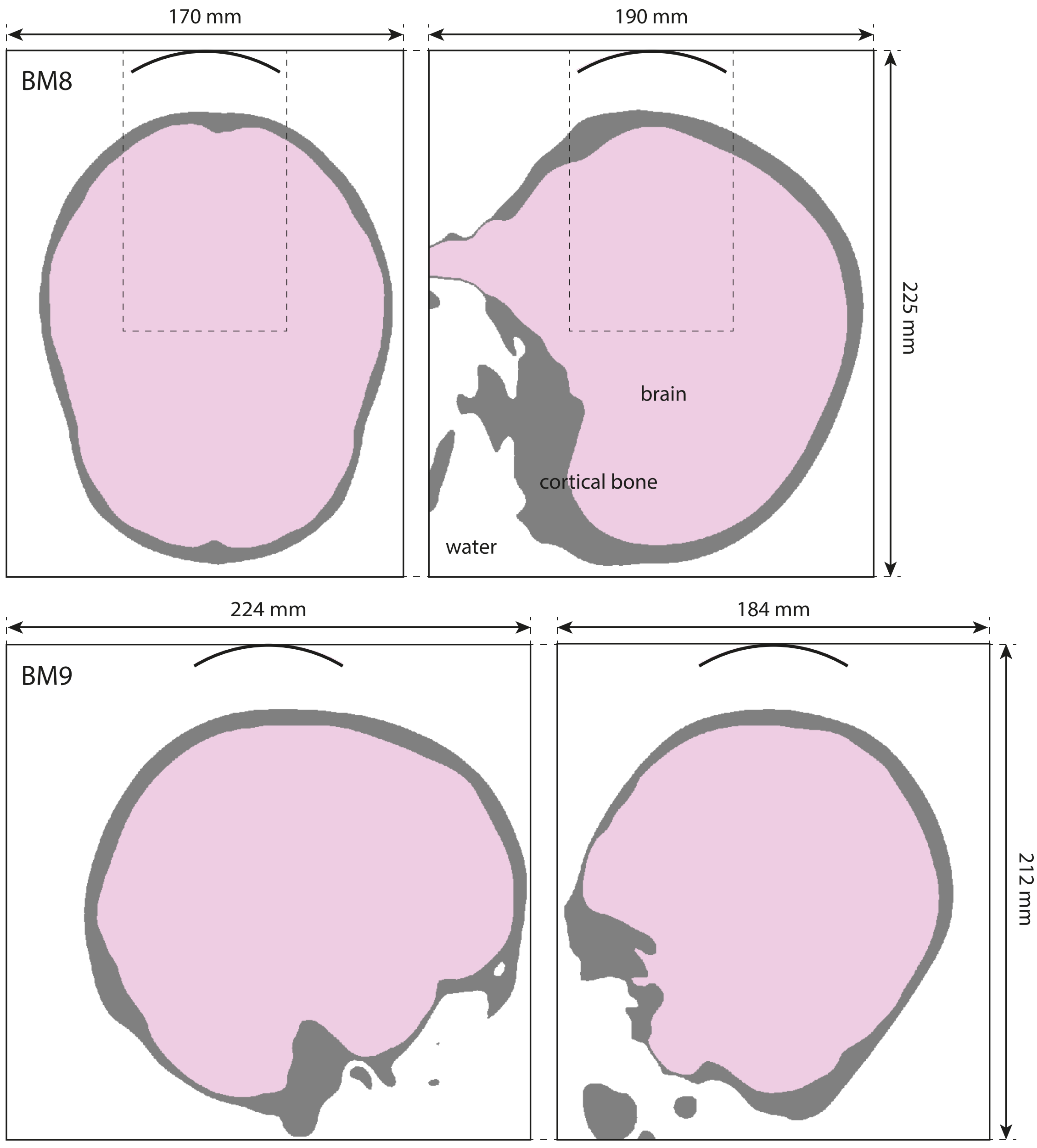


Figure 4.2: Simulation layouts for benchmarks 8 (top row) and 9 (bottom row) showing the central x-y and x-z slices (description in Figure 4.1)[2].

[2]The illustration taken from [6].

Chapter 5

Proposal of on-the-fly compression in ultrasound simulations

This chapter covers the core of the work - proposal of on-the-fly compression in ultrasound simulations. First, in Section 5.1, the objectives of the work are presented, that is, the hypothesis is formulated and the way of verifying it. In the next Section 5.2, the potential of the new compression method is shown on hypothetical application examples, and the justification of this proposal is presented.

In the following Section 5.3, the proposal of the new compression method is detailed. Experiments with this method and their results follow. Specifically, a comparison of the new method with other state-of-the-art methods that have a similar character is presented. Since there are a large number of compression methods for data with partially similar character, as mentioned, among others, in Chapter 3, and it is not possible to make a comparison with all of them, selected audio compression algorithms were used for this purpose. However, these experiments are primarily intended to verify that the new method has comparable properties in terms of output quality according to general evaluation measures, such as compression ratio or PSNR.

The next Section 5.4 presents probably the most important part of this work, namely the application of the compression method for on-the-fly calculation of the average acoustic intensity in time-domain ultrasound simulations. The section first describes a new, more efficient way to calculate the acoustic intensity using the already introduced compression method. Furthermore, an experimental evaluation, where there is a comparison of the use and not use of the compression with regard to the consumption of computing resources and numerical errors.

5.1 Goal of the thesis

The area of interest of my work is the compression of HIFU large-scale simulation data. No standard compression scheme exists for this type of simulation data. The goal was to develop new methods for simulation data reduction that will be efficient and applicable within large-scale HIFU simulations. Compression is assumed to be performed in parallel and on-the-fly during simulations on large CPU clusters. From the point of view of scientific contribution, the goal is primarily to find a significant application of this compression. The scientific contribution of my Ph.D. work consists in proving the following hypothesis:

It is possible to develop a novel on-the-fly compression method for 1-D time-varying data series - HIFU simulation outputs, with a compression-ratio comparable the other state-of-the-art methods, with the quality of application outputs from the users (scientists or medical doctors) point of view using the compressed data will be comparable to the quality achieved when the original uncompressed data or data compressed by other state-of-the-art methods are used, and the novel method will lead in significant improvement of the HPC simulation calculation through saving disk storage space by more than 90 % while the demands on other computing resources, such as RAM and and computational time will remain approximately the same.

The hypothesis is proved experimentally in the following Sections 5.3 and 5.4. At least 3 types of simulation datasets for the experiments were obtained from scientists in the field of HIFU simulations. The datasets contain typical data for clinical applications. The experimental evaluation of quality of outputs applications is performed automatically, e.g. calculation of heat propagation. Comparisons with other state-of-the-art compression methods is made using general evaluation measures such as PSNR or the compression ratio. The utilization of the computer resources is measured on HPC platforms typical for clinical simulations.

5.2 Justification of the design of the novel method properties

Based on the study of the state-of-the-art in the field of ultrasound simulations, specifically the k-space pseudospectral method that uses a time-staggered PSTD, it was generally found that:

- the simulations generate really large data, continuously, during simulation calculations, and this data should be compressed already during the simulations,
- computations are performed primarily in a parallel environment and it is expensive to exchange data between neighboring nodes, blocks or points,
- calculations have a high computational time and operational memory requirements and since a large number of nodes and large matrices are used for calculations, an increase in memory requirements for each point is undesirable,
- the selected sampled points can be often sparsely located in the space of the simulation domain, multidimensional compression cannot be reasonably used,
- the outputs of the simulations have a harmonic character, while the input fundamental frequency is known, and this information can be used in compression.

Furthermore, various compression methods were investigated and it was found that no state-of-the-art method exactly matches for this type of data.

From these facts, it was decided that it would be appropriate to design a novel compression method that would ideally have the following properties:

- design for parallel environment, so that data does not have to be sent between nodes, blocks or points,
- on-the-fly simulation outputs processing, which allows saving compressed data during the simulation,

- low memory complexity, due to the high memory requirements for each point in the simulation domain,
- high computational speed, which does not slow down the demanding calculation of the simulation
- design for 1-D, due to the arbitrary shapes of the masks used in ultrasound simulations,
- model-based coding, which uses the known input fundamental frequency and harmonic character of the signal.

It is assumed that the novel method does not necessarily need to use spatial coherence, as the individual points in space can be examined independently and sensor mask (selected points to be sampled and stored) can be an arbitrary and sparse set of locations. Every grid point of interest in 3-D space is processed separately. Moreover, by treating every spatial grid point independently, no additional communication is needed on distributed clusters (in the case of on-the-fly processing).

Suppose that some already existing state-of-the-art compression method was hypothetically applied to the simulation data. Due to the nature of the processed ultrasound signal, audio compression methods can be used, for example. Let us assume that the selected compression method would have compression ratio 20:1. As discussed in the state-of-the-art (Section 3.2), these methods process the signal by frames, which typically include thousands of samples. Based on this, it can be assumed that the selected state-of-the-art compression method could have a memory consumption per voxel (1 sampled point in the domain) of 1,024 bytes. Furthermore, let us assume that the novel compression method has a slightly worse compression ratio of 10:1, but significantly less memory consumption of 16 bytes per voxel for each harmonic component in the signal.

Several arguments why the development of a novel compression method defined in this way makes sense are shown in Tables 5.1 to 5.4. Memory consumption in these examples refers only to RAM, disk memory is not taken into account.

Let us define the simulation parameters as follows (close to the real life simulations): The simulation domain size (number of voxels) N_{all} is computed as $N_{\mathrm{all}} = N_x \times N_y \times N_z$, where N_x, N_y, and N_z are simulation domain sizes for the x-, y- and z-axes. The total memory consumption is calculates as $30 \times N_{\mathrm{all}} \times 4$ (ca 30 temporary matrices, 4 bytes per voxel, see Section 2.2). The sensor mask size N_{sensor} is the number of voxels sampled in one time step and is calculated as $N_{\mathrm{sensor}} = N_{\mathrm{all}} \times R_{\mathrm{sensor}}$, since the R_{sensor} is a ratio that indicates the number of sampled points of the domain size. The number of bytes stored in one step is calculated as $N_{\mathrm{sensor}} \times 4 \times 4$ since 4 time-variable quantities are stored with 4 bytes per voxel. The simulation speed is hypothetical simulation speed given in steps per second, adjusted for simulation size and computing environment.

In the first Table 5.1, another hypothetical (also realistic) example of simulation parameters is given. The simulation domain size is 768^3 and in the each simulation step 10 % of the simulation size is stored. Thus, the number of bytes stored in one simulation step (sensor mask size) is ca 724 MB. The total memory consumption (RAM) is ca 54 GB ($30 \times 768^3 \times 4$). We can assume that state-of-the-art compression method has a slightly better compression ratio (20:1), but much bigger memory consumption per voxel 1,024. The novel compression would require, for example, 48 bytes per voxel for 3 harmonic frequencies (16×3). Data flow and increased memory consumption are shown in Table 5.2. Without any compression there is excessive data flow (ca 760 MB s^{-1}), and with the use of some state-of-the-art compression method with a good compression ratio, the increase in memory consumption is enormous (185 GB). The novel compression method is a compromise, ca 8.6 GB is not so much memory due to 54 GB and the data flow 76 MB s^{-1} is acceptable.

Table 5.1: An example of simulation parameters for Table 5.2. The domain size: 768^3, 10 % of the size sampled for 4 time-variable quantities (acoustic pressure and the particle velocity vector)

Simulation parameters	
Total memory consumption	54 GB
Simulation domain size (number of voxels)	452,984,832
Sensor mask size (number of sampled voxels in one time step)	45,298,483
Number of bytes stored in one step	724,775,731
Simulation speed (steps per second)	1.05
Bytes/voxel - novel compression method	48
Bytes/voxel - state-of-the-art compression method	1,024
Novel compression ratio	10:1
State-of-the-art compression ratio	20:1

Table 5.2: An hypothetical example of the comparison of approaches without compression, with novel compression, and with state-of-the-art compression. The values are computed from the parameters in Table 5.1.

	No compression	Novel compression	State-of-the-art compression
Mega samples/second	190	19	10
Megabytes/second	761	76	38
Increase in memory consumption	0 GB	9 GB	186 GB
	Excessive data flow	**OK**	**Excessive increase in memory consumption**

The next hypothetical example of simulation parameters and the corresponding data flow and the increase in memory consumption for larger simulations is shown in Tables 5.3 and 5.4. The simulation domain size is 1152^3 and ca 6.6 % of the size is sampled. Total memory consumption is much higher due to large-scale simulation on distributed clusters (more than 700 GB). We assume that the novel method is model-based and has higher unit memory requirements for larger simulations, due to the occurrence of six harmonic frequencies; therefore, the number of bytes per voxel of the novel compression method is larger (96 bytes, 16×6). Due to the calculation on distributed clusters, the simulation itself could be about twice faster. It can be seen an enormous increase in memory consumption (more than 500 GB) with the state-of-the-art compression method and large data flow without any compression in Table 5.4.

Table 5.3: An example of simulation parameters for Table 5.4. The domain size: 1152^3, ca 6.6 % of the size sampled for 4 time-variable quantities (acoustic pressure and the particle velocity vector)

Simulation parameters	
Total memory consumption	715 GB
Simulation domain size (number of voxels)	2,038,431,744
Sensor mask size (number of sampled voxels in one step)	134,217,728
Number of bytes stored in one step	2,147,483,648
Simulation speed (steps per second)	2.19
Bytes/voxel - novel compression method	96
Bytes/voxel - state-of-the-art compression method	1,024
Novel compression ratio	10:1
State-of-the-art compression ratio	20:1

Table 5.4: An hypothetical example of the comparison of approaches without compression, with novel compression, and with state-of-the-art compression. The values are computed from the parameters in Table 5.3.

	No compression	Novel compression	State-of-the-art compression
Mega samples/second	1,176	118	59
Megabytes/second	4,703	470	235
Increase in memory consumption	0 GB	52 GB	550 GB
	Excessive data flow	**OK**	**Excessive increase in memory consumption**

The presented hypothetical examples show that it is really desirable to develop a new compression method with the properties listed at the beginning of this subsection.

5.3 Efficient compression of HIFU data

The subject of this section is the presentation of a new compression method for ultrasound simulations. The core compression method for time varying HIFU simulation data I published in [42], Efficient Lossy Compression of Ultrasound Data, with my colleagues Pavel Zemčík, and Jiří Jaroš, and in [41], Efficient Low-Resource Compression of HIFU Data, with my colleagues David Bařina, Pavel Zemčík, and Jiří Jaroš. Both publications contain, in addition to the definition of the method itself, a comparison with audio codec - state-of-the-art methods for on-the-fly data compression. The second publication is an extended journal version of the first publication.

The compression algorithm is able to compress 3-D pressure time series of linear and nonlinear simulations with very acceptable compression ratios and errors (over 80 % of the space can be saved with an acceptable error). The proposed compression enables significant reduction of resources, such as storage space, network bandwidth, CPU time, and so forth, enabling better treatment planning using fast volume data visualizations. This section is based on the revised texts of the journal paper [41]. Specifically it contains a description of the new method for compressing ultrasound simulation data, and also experiments and comparison results with other compression methods. The presentation of this method here is key to the Section 5.4 with its application to the calculation of time-averaged acoustic intensity.

5.3.1 Compression method

Our compression method is focused on time-varying signals (on-the-fly data compression during simulations) and is designed for 1-D data. Because the shape of the sensor mask can be an arbitrary and sparse set of locations, we decided not to deal with spatial coherence. Furthermore, by treating every spatial grid point independently, there was no need for additional communication on the distributed clusters. The goals of the compression method are low memory complexity and high processing speed; negligible data distortion is acceptable. These intentions differentiate our method from other state-of-the-art compression techniques.

The source ultrasound signal is defined by a known harmonic function of a given frequency, usually a pressure sinusoid. We can assume that the time-varying quantities, such as pressure and velocity at every grid point, also have a harmonic character with only small amplitude and phase deviations. These quantities are usually amplitude-modulated and can be composed of the fundamental and several harmonic frequencies. The number of harmonics depends on the size of the simulation grid and a factor of nonlinearity. Currently, we use up to five or six harmonics for real simulations, which corresponds to the HIFU in thermal mode.

The proposed approach is to model an output signal, such as the decomposition of a 1-D signal (one point in the 3-D space), as a sum of overlapped exponential bases multiplied by a window function. We decided to use complex exponential bases, because such bases can represent fundamental harmonic functions well, including phase changes and higher harmonics, as well as window functions whose sum is constant if they are half-overlapped, because of on-the-fly processing by parts of the signal. It is important that the input frequency emitted by the transducer is known and that it can be used by the compression algorithm.

Each base is defined by its complex coefficients (amplitude and phase) and a harmonic frequency (wavenumber). A shifted window function w is defined as

$$w(t, m, d) = \begin{cases} 0 & (m+2)dT \leq t < mdT \\ w_0(t - mdT) & \text{otherwise} \end{cases} \tag{5.1}$$

or

$$w(n, m, d) = \begin{cases} 0 & (m+2)dN \leq n < mdN \\ w_0(n - mdN) & \text{otherwise} \end{cases} \tag{5.2}$$

where w_0 is a window function (typically Hann or Triangular), N or T is the number of samples within the period or acoustic period, respectively ($\Delta t = T/N$, $N\Delta t = T$), n or t is the simulation time step or time, resp., m is a window (the basis) index, and d is an integer multiple of overlap size (MOS). The length of the window is therefore $2dN$ or $2dT$, resp. We obtain complex exponential sliding-window basis vectors

$$b(t, m, h, d) = w(t, m, d)e^{-jh\omega t} \tag{5.3}$$

or

$$b(n, m, h, d) = w(n, m, d)e^{-jh\Omega n} \tag{5.4}$$

where

$$\omega = \frac{2\pi}{T} \text{ and } \Omega = \frac{2\pi}{N} \tag{5.5}$$

with the number of the harmonic frequency (wavenumber) h and the known fundamental angular frequency ω (Ω).

Let M be the total number of periods of the fundamental frequency of the signal (let us assume that MN is the total number of samples taken, also MT is the total duration of the signal). The whole reconstructed signal s can then be expressed as

$$s(n) = \sum_{h=1}^{H} \frac{2}{dN} \sum_{m=0}^{M-1} b(n, m, h, d)\widehat{k}(m, h) \tag{5.6}$$

where H is the number of harmonics (1 to H), h is a harmonic index, and k are the resulting complex coefficients. The normalization factor $2/dN$ is based on the sum of the window function samples $dN/2$, i.e., the area $dT/2$ in continuous time.

An illustration of three half-overlapped windows with $d = 2$ (length $= 4N$) and the real-part sum of window functions is shown in Figure 5.1.

The coefficients k for the harmonic frequency h used to model the output simulation signal x are approximately computed for every frame m (usually with a minimum length of two periods $2N$, which experimentally proved to be the most suitable) as a dot product of the simulation signal sample $x(n)$ and the windowed exponential basis vector b (the

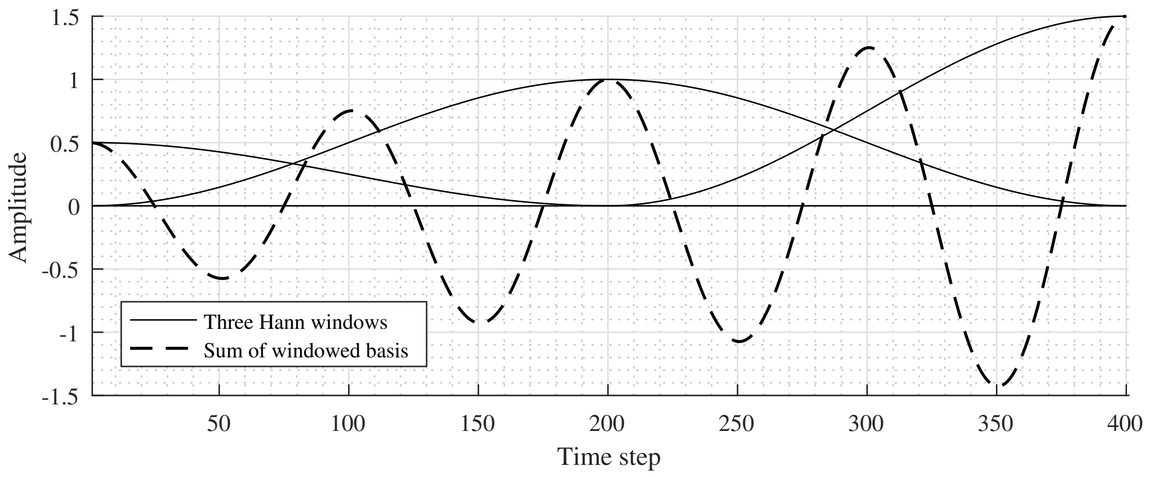


Figure 5.1: The illustration of three Hann window bases.

vinculum denotes complex conjugate of b)

$$\widehat{k}(m,h) = \sum_{n=0}^{MN-1} \overline{b(n,m,h,d)}x(n). \tag{5.7}$$

The bases of the individual harmonic components are independent/perpendicular to each other because

$$\sum_{m=0}^{M-1} b(n,m,g,d)b(n,m,h,d) = 0 \text{ whenever } g \neq h. \tag{5.8}$$

The coefficients for other harmonic frequencies can be computed independently and are summed in the reconstruction phase. Every point in 3-D space can be processed separately and in parallel within both the encoding and decoding phases. It is not necessary to have the entire signal x available to calculate one coefficient, because the sliding-window basis vectors b are zero for

$$(m+2)dN \leq n < mdN. \tag{5.9}$$

The number of memory cells c (single-precision floating-point numbers) required for computing intermediate results in one time step depends on the number of harmonics H and it can be evaluated as

$$c = 4H \tag{5.10}$$

as two complex numbers are needed per every harmonic frequency.

Within the decoding phase, the memory requirements remain the same as in the encoding phase. Two complex coefficients are needed for every harmonic frequency; however, the decoded samples are computed independently, which can be useful, for example, for fast data visualization.

The compression ratio of the method can be expressed as

$$\mathrm{length}(x)/(\mathrm{length}(\widehat{k})2H). \tag{5.11}$$

The ratio is proportional to the width of the overlapping window bases. This signal approximation method yields a very small distortion for the stable parts of the signal, where the distortion depends only on the number of considered harmonic frequencies. The distortion is greater in the transient parts of the signal.

5.3.2 Implementation

Currently, two baseline implementations of the proposed methods are available—an implementation in MATLAB and FFmpeg to process 1-D signals and a second implementation in C++ to process large 4-D datasets. Both versions process the simulation data sequentially and offline by reading datasets in the HDF5 format. Another parallel on-the-fly implementation with CUDA and with the OpenMP version is also available.

The MATLAB and FFmpeg implementations were developed specifically to process 1-D data (time series at a single grid point). The FFmpeg implementation mediates simple ways for the compression method's debugging, comparison with other coding methods (e.g., audio codecs), and the visualizations of processing steps and results. The C++ implementation differs from the MATLAB implementation in the ability to process a large amount of 4-D data. Individual HDF5 datasets are loaded by 3-D blocks depending on the main memory size. Both implementations were tested on a desktop computer with 24 GB memory and Intel Core i5-6500 processor. For experiments with extensive HDF5 files, the Salomon cluster with 128 GB memory, two Intel Xeon E5-2680v3 processors, and Lustre shared storage space with a maximal theoretical throughput of 6 GB s^{-1} for one computing node was used [31].

5.3.3 Experiments and Results

Two sets of experiments were performed. The first set was focused on testing the compression method on 1-D signals, and the second set was conducted for large 4-D datasets. A triangular window was used as a window function for the proposed compression method for all experiments. The main objectives of the experiments were to determine the peak signal-to-noise ratio (PSNR) with respect to the bit rate and to compare the proposed method with other compression methods, particularly with audio codecs.

For the 1-D signal compression tests, three different types of signals were selected. We always chose 50 random 1-D signals (points within the sensor mask) from the given datasets. The schematic overview of the test signals is shown in Tables 5.5 and 5.6.

Table 5.5: The types of testing signals - part 1. All signals represent acoustic pressure.

Name	Max number of harmonics (H)	Simulation size
Linear	1	$256 \times 256 \times 350$
Nonlinear 2	2	$512 \times 384 \times 384$
Nonlinear 6	6	$1536 \times 1152 \times 1152$

Table 5.6: The types of testing signals - part 2. All signals represent acoustic pressure.

Name	Sensor mask size	Samples	Period (N)
Linear	$55 \times 55 \times 82$	3301	15
Nonlinear 2	$101 \times 101 \times 101$	10,105	35
Nonlinear 6	$101 \times 101 \times 101$	30,604	106

The signals differ mainly in the simulation size, sensor mask size, number of sampled simulation steps, input period, and type of the simulation. The dense sensor mask was always defined within the highly focused HIFU area in which the highest amplitude and most of the harmonics are observed. All signals contain acoustic pressure stored in 32-bit floating-point format. The dataset referred to as Linear is the output of a linear simulation; thus, there are no harmonics. The other datasets comprise the outputs from nonlinear simulations, and the number of harmonics depended, among other factors, on the simulation resolution. For a better connection to reality, a realistic simulation that was representative of the clinical situation with heterogeneous tissue used a simulation grid size of $1536 \times 1152 \times 1152$ and contained about six significantly strong harmonics (referred to as nonlinear 6). The same simulation (nonlinear 2) but with a smaller simulation grid size $512 \times 384 \times 384$ contained only two harmonics.

The compression experiments compared the results of the proposed HIFU compression method (HCM) with several audio codecs implemented in the FFmpeg framework (either natively or through an external library), specifically with ADPCM, FLAC, ALAC, AAC, AC-3, Opus, and PCM. The main properties and settings of these codecs, including the proposed method, are shown in Table 5.7. The above-listed methods were selected for comparison purposes only. Except for PCM, none of them meets the memory requirements needed for implementation in a distributed parallel environment. Because we target a memory-limited environment, we set the frame size to the minimum size supported by the specific sampling rate, sample format, and codec. We note that as a result of the frame-based system, a delay is introduced after the encoding and decoding processes. This delay is shown in the last column of the referenced table.

Table 5.7: Overview of tested formats and codec settings.

Codec	Long Name	Class	Frame Size	Delay
PCM	Pulse-code modulation	Lossless	1	0
FLAC	Free lossless audio codec	Lossless	16	0
ALAC	Apple lossless audio codec	Lossless	4096	0
ADPCM	Microsoft ADPCM	Lossy	2036	0
Opus	Opus	Lossy	120	120
AAC	Advanced audio coding	Lossy	1024	1024
AC-3	Dolby digital (ATSC A/52)	Lossy	1536	256
HCM	HIFU compression method	Lossy	1	0

Because several audio codecs require the signal to be normalized on the $\langle 0, 1\rangle$ interval, we used the maximal absolute signal value for normalization. In some cases, the input signal had to be stored in 16-bit PCM format before the codec could be applied (ADPCM and Opus). This conversion was performed with the use of the maximum 16-bit integer value.

By default, the HCM expects an unknown maximum absolute signal value and therefore uses 32-bit floating-point numbers to store compression coefficients. If the dynamic range of values is known in advance, the method can convert 32-bit floating-point numbers into 16-bit or even 8-bit integers with negligible loss of information (4 or 2 bytes for one complex number). Thus, for normalized signals, two additional modifications of the HCM were implemented. As shown in all figures and tables except Figures 5.2 to 5.4, 5.10 and 5.11, the HCM used 8-bit integers for coefficients and thus 2 bytes for one complex number. For a better understanding, a comparison of three types of HCM coding is shown in Figures 5.2 to 5.4.

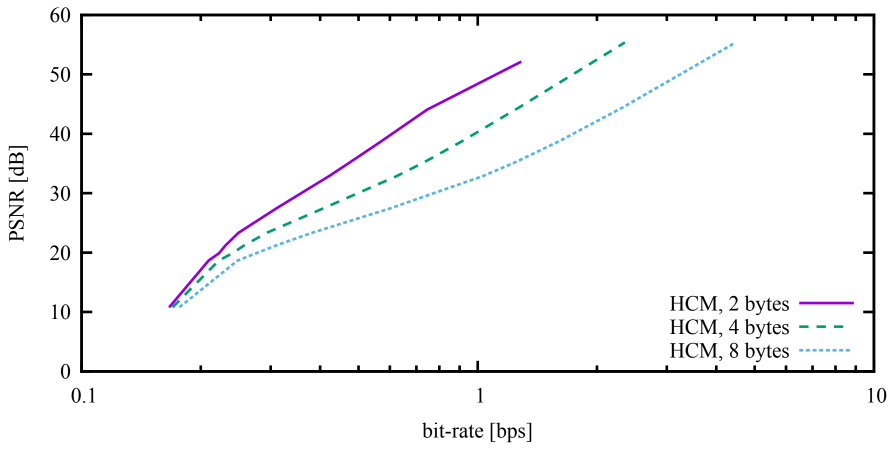


Figure 5.2: Different variations of the high-intensity focused ultrasound (HIFU) compression methods (HCMs). The average bit rate and the corresponding peak signal-to-noise ratio (PSNR) for Linear dataset.

The dependence of the distortion on the bit rate is shown in Figures 5.5 to 5.7 for the datasets Linear, nonlinear 2, and nonlinear 6. The lossless methods are represented by a single point, as they do not have, by their nature, the ability to choose any custom bit rate. Except for the ADPCM, the lossy methods are represented as a curve for which the independent variable is the custom bit rate. Although the ADPCM is a lossy method, it has no ability to choose a bit rate. Thus, in this case, the method is also represented by a single point. Looking more closely at the referenced figure, we can see that the HCM exhibited at least comparable compression performance. More precisely, the PSNR for the Linear case, as shown in Figure 5.5, reached comparable values for all lossy methods except the ADPCM and Opus, which had significantly worse results. The proposed HCM reached the lowest bit rate. At its highest bit rate, the HCM also exhibited the best PSNR, which was about 52 dB. It is possible to obtain better bit rates with the proposed method by setting the multiple of overlap size (MOS) to higher values.

The results for the nonlinear simulation signals were slightly different from those for the Linear dataset. In Figure 5.6, the PSNR and bit rate for the Nonlinear 2 dataset are shown. We obtained a slightly worse PSNR, approximately 48 dB, for the proposed HCM,

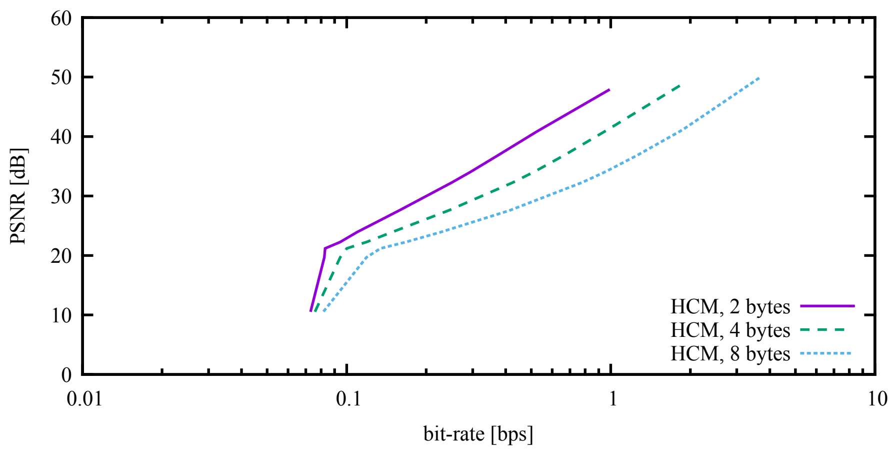


Figure 5.3: Different variations of the high-intensity focused ultrasound (HIFU) compression methods (HCMs). The average bit rate and the corresponding peak signal-to-noise ratio (PSNR) for Nonlinear 2 dataset.

and better values for the other methods. We believe this was due to the presence of more harmonics.

In the case of the Nonlinear 6 dataset (larger simulation grid size and about six harmonics), the proposed method gave similar results as for the previous cases. ADPCM exhibited comparable PSNR, and it was only slightly worse in terms of bit rate than the proposed HCM. Moreover, the AAC codec achieved a slightly better bit rate, followed by the competitive AC-3. Further details can be seen in Figure 5.7.

Different variations of the HCMs used for the Linear, Nonlinear 2, and Nonlinear 6 compression datasets can be observed in Figures 5.2 to 5.4. Importantly, the PSNR reached comparable-quality values for all cases. It can be noticed that the bit rates converged to a single point in the plot—this was caused by the use of a WAVE file header for storing the compressed data. The WAVE format was used only for testing purposes with the FFmpeg tool.

The particular results, including compression ratios, with the PSNRs set as close as possible to 50 dB are listed in Tables 5.8 to 5.10. The values correspond to Figure 5.8. We note the comparable bit rates of the HCM, AC-3, and AAC.

A short segment of a signal in the 1-D Nonlinear 6 dataset is shown in Figure 5.9. The individual sub-figures illustrate the original, HCM-reconstructed, and corresponding difference signals, respectively. All of these correspond to a MOS value set to 1. We note the maximal error in a part with very transient signal values.

Table 5.8: Results for Linear dataset. Bit rates taken as close as possible to 50 dB.

	HCM	AC-3	AAC	Opus	ADPCM	ALAC	FLAC	PCM
Bit rate (bit s^{-1})	1.28	1.33	5.33	6.38	4.27	7.54	43.42	16.19
Compression ratio	25:1	24:1	6:1	5:1	7.5:1	4.2:1	0.7:1	2:1
PSNR (dB)	52.05	44.66	48.67	22.43	33.83	101.11	101.11	101.11

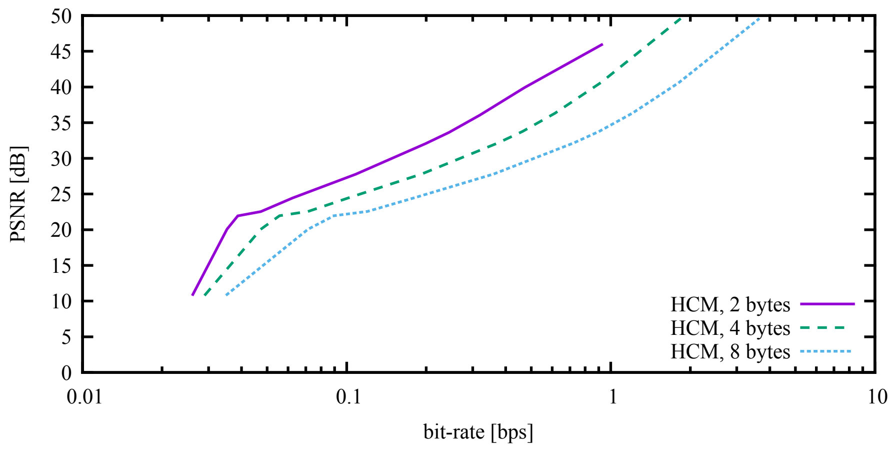


Figure 5.4: Different variations of the high-intensity focused ultrasound (HIFU) compression methods (HCMs). The average bit rate and the corresponding peak signal-to-noise ratio (PSNR) for Nonlinear 6 dataset.

Table 5.9: Results for Nonlinear 2 dataset. Bit rates taken as close as possible to 50 dB.

	HCM	AC-3	AAC	Opus	ADPCM	ALAC	FLAC	PCM
Bit rate (bit s^{-1})	0.98	1.33	2.03	5.65	4.12	6.69	27.04	16.06
Compression ratio	32.6:1	24:1	15.8:1	5.7:1	7.8:1	4.8:1	1.2:1	2:1
PSNR (dB)	47.81	47.87	49.85	25.35	42.03	101.23	101.23	101.23

We also conducted compression experiments with two 4-D datasets. The tested signals were chosen from Nonlinear 2 and Nonlinear 6, and all the points stored in the sensor mask were compressed ($101 \times 101 \times 101 \times 10,105$ points for Nonlinear 2 4-D signal and ($101 \times 101 \times 101 \times 30,604$ points for Nonlinear 6 4-D signal). The proposed compression method was tested with the setting of the MOS to values of 1, 2, 3, and 4.

We obtained interesting results for the Nonlinear 2 case (Figure 5.10). The PSNR values for the 4-D dataset were approximately 20 dB better than for the 1-D signals.

In the case of the Nonlinear 6 signal, the results were slightly different (Figure 5.11). The PSNR values for the 4-D signal were about 13 dB better than for the 1-D signal. This phenomenon was likely caused by a 3-fold larger simulation grid size in the Nonlinear 6 case but the same size of the sensor mask. We could conduct experiments with a corresponding higher sensor mask size (e.g., $301 \times 301 \times 301$) and expect results similar to those of the Nonlinear 2 case, but more than 3 TB of data would be need for the testing dataset, and the time for offline compression would be enormous.

Table 5.10: Results for Nonlinear 6 dataset. Bit rates taken as close as possible to 50 dB.

	HCM	AC-3	AAC	Opus	ADPCM	ALAC	FLAC	PCM
Bit rate (bit s^{-1})	0.92	1.00	0.69	5.55	4.05	4.53	19.41	16.02
Compression ratio	34.6:1	32:1	46.7:1	5.8:1	7.9:1	7.1:1	1.6:1	2:1
PSNR (dB)	45.92	49.80	50.82	28.94	51.28	102.34	102.34	102.34

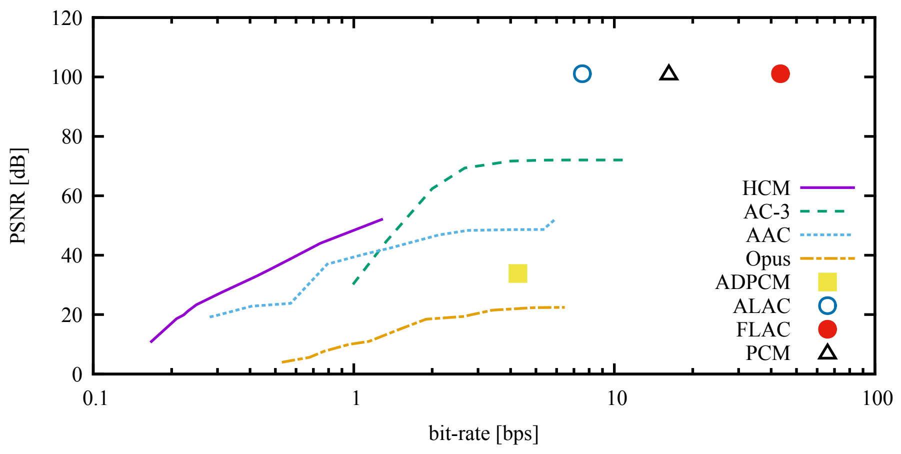


Figure 5.5: A relationship between the average bit rate and the corresponding peak signal-to-noise ratio (PSNR) for Linear dataset with one harmonic.

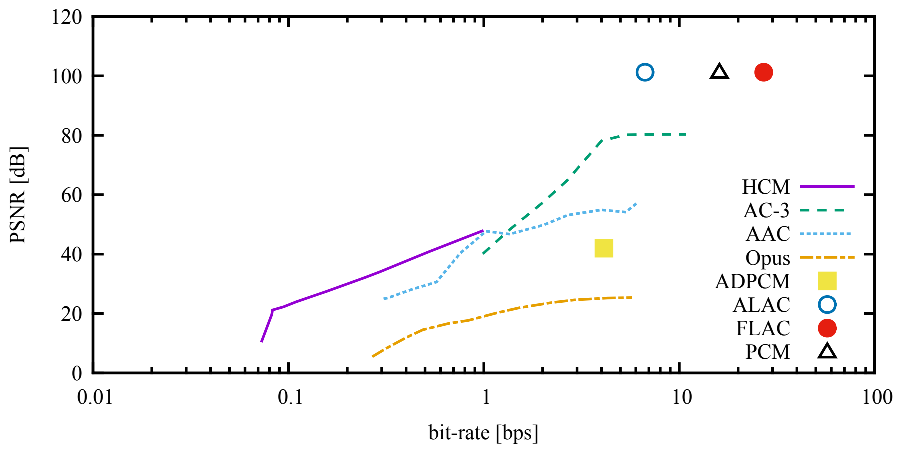


Figure 5.6: A relationship between the average bit rate and the corresponding peak signal-to-noise ratio (PSNR) for Nonlinear 2 dataset with two harmonics.

The overall results indicate useful properties of the proposed method. The PSNRs of all the signals had values comparable with those of the state-of-the-art audio codecs. The maximum errors occurred only in short transient parts of the signals. The errors in the stable segments were negligible. We expect better PSNRs with larger sensor masks and smaller MOS values.

In further work from the point of view of publication [41, 42] it would be worth trying to further optimize the basis and window functions so that the errors in transient signal segments will possibly be reduced or to apply follow-up compression algorithms to further compress the resulting coefficients so that higher compression ratios are achieved.

An efficient compression algorithm for HIFU simulation data was proposed, and offline experiments were performed to evaluate it. We have shown that our method produces very useful results. The important stable parts of the simulation signals are compressed with very small distortion (0.1 %) at compression ratios over 80 %. The very short transient parts of the signals are compressed with acceptable errors. The proposed compression algorithm was also implemented in the existing implementation of the k-Wave simulation toolbox [78].

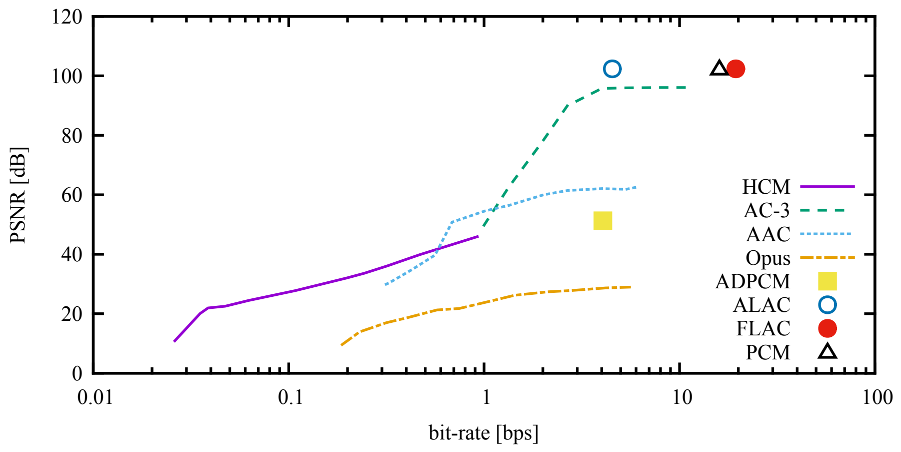


Figure 5.7: A relationship between the average bit rate and the corresponding peak signal-to-noise ratio (PSNR) for Nonlinear 6 dataset with six harmonics.

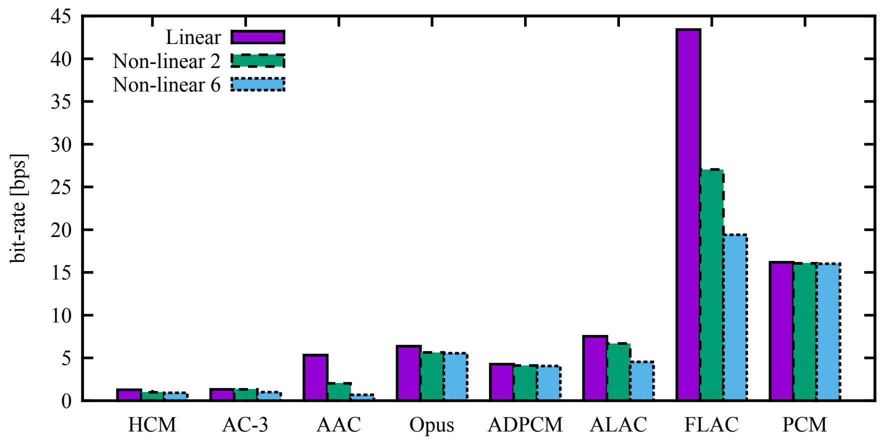


Figure 5.8: Average bit rates for the peak signal-to-noise ratio (PSNR) as close as possible to 50 dB for three testing datasets.

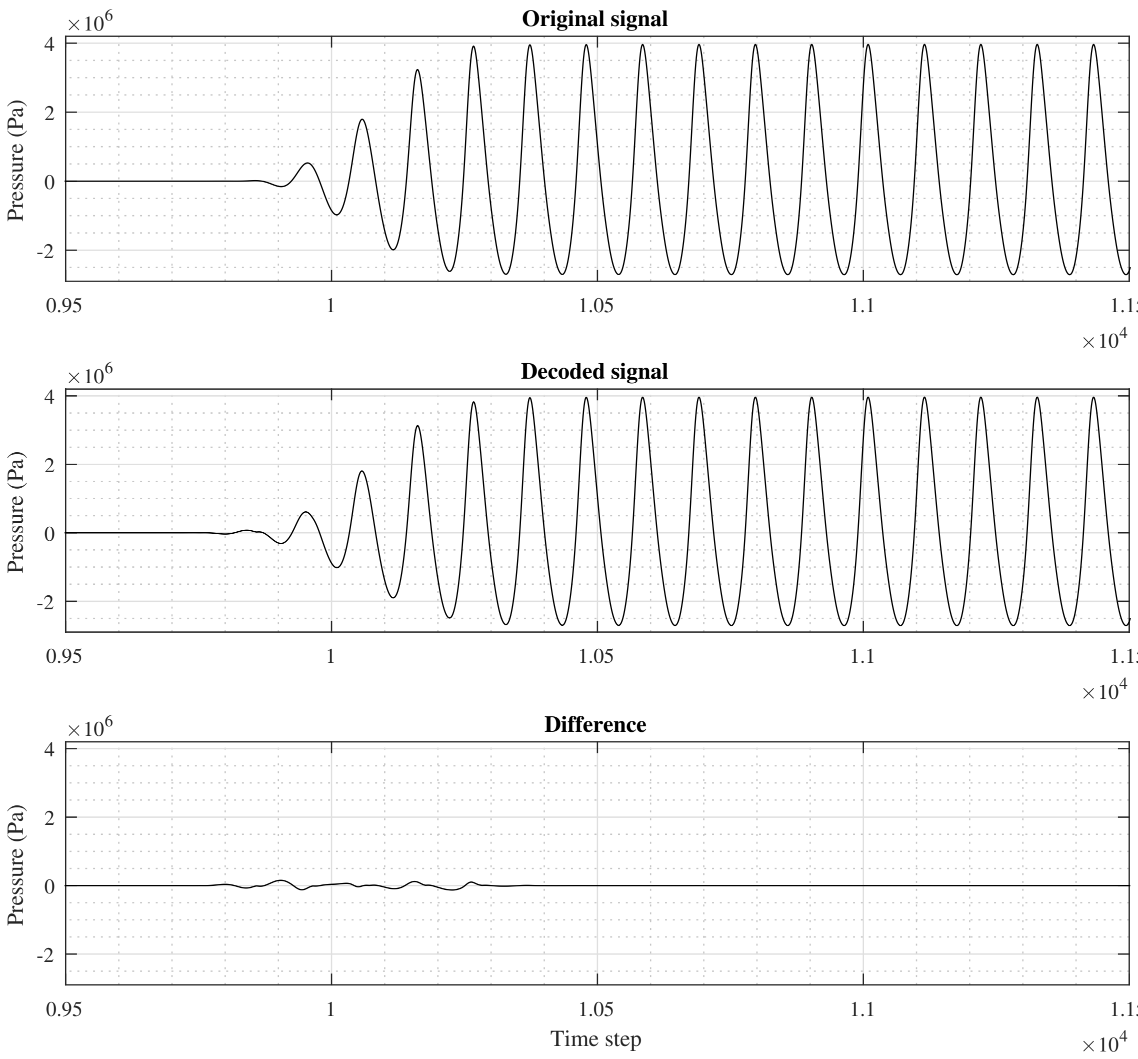


Figure 5.9: Illustration of the Nonlinear 6 original, reconstructed, and difference signals with multiple of overlap size (MOS) equal to 1.

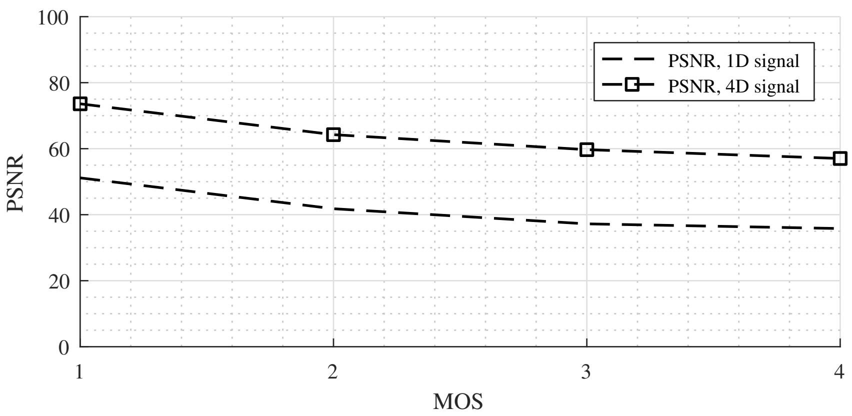


Figure 5.10: Peak signal-to-noise ratios (PSNRs) for different multiples of overlap size (MOSs) with Nonlinear 2 1-D and 4-D signals.

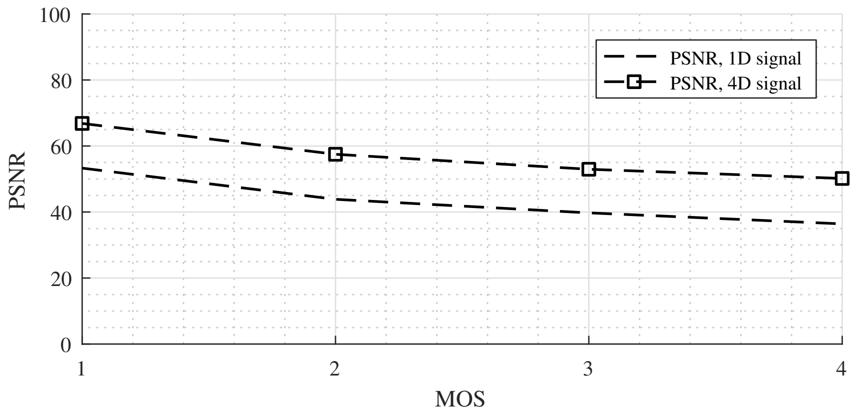


Figure 5.11: Peak signal-to-noise ratio (PSNRs) for different multiples of overlap size (MOSs) with Nonlinear 6 1-D and 4-D signals.

5.4 On-the-fly calculation of time-averaged acoustic intensity

The subject of this section is the application of the compression method presented in the previous chapter, i.e. on-the-fly calculation of the average acoustic intensity in time-domain ultrasound simulations. The main article [43], On-the-Fly Calculation of Time-Averaged Acoustic Intensity in Time-Domain Ultrasound Simulations Using a k-Space Pseudospectral Method, that I published with my colleagues Pavel Zemčík, Bradley E. Treeby, and Jiří Jaroš in the IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control journal, is the core publication of this work. This article presents a method for calculating the average acoustic intensity during ultrasound simulation using a new approach that takes advantage of compression of intermediate results.

The thermal simulation is preceded by the calculation of the average intensity within the acoustic simulation. Due to the time staggering between the particle velocity and the acoustic pressure used in such simulations, the average intensity calculation (Section 2.3) is typically executed offline after the acoustic simulation, which consumes both disk space and time (the data can spread over terabytes). Our new approach calculates the average intensity during the acoustic simulation using the output coefficients of a new compression method, which enables resolving the time staggering on-the-fly with huge disk space savings. To reduce RAM requirements, the article also presents a new 40-bit method for encoding compression complex coefficients.

Experimental numerical simulations with the proposed method have shown that disk space requirements are up to 99 % lower. The simulation speed was not significantly affected by the approach and the compression error did not affect the prediction accuracy of the thermal dose. From the standpoint of supercomputers, the new approach is significantly more economical. Saving computing resources increases the chances of real use of acoustic simulations in practice.

This section contains a description of a new time-averaged acoustic intensity calculation approach and an experimental evaluation of the application of this calculation, i.e. the section is based on the revised texts of the journal paper [43].

The following subsection follows the simulation workflow and offline average intensity calculation that are described in the state-of-the-art part of this work - Section 2.3, as well as the compression method described in the Section 5.3.1.

5.4.1 Proposed approach

This subsection is divided into 3 parts. The first is devoted to the calculation of the intensity itself, the second to the memory requirements of the method and the third to the improved coding of complex coefficients.

On-the-fly calculation of intensity

Here, we describe how to calculate the time-averaged intensity vector during the simulation using on-the-fly data compression (Section 5.3.1) [41, 42]. This directly uses compression coefficients to calculate the average intensity, which are not stored in files during the simulation.

In case of the average intensity calculation, we are specifically interested in the coefficients of the acoustic pressure and the particle velocity. Let k_p and $\mathbf{k}_{\mathrm{u\_staggered}}$ be the computed compression coefficients for the pressure and the staggered velocity from the previous Section 5.3.1. For simplicity, we consider the coefficients only for the one window base. The shift of the particle velocity in time by half the sampling period ($\Delta t/2$, 1/2 sample, thus phase shift by $\Omega/2$) is being calculated by exploiting a shift of the phase, therefore

$$\mathbf{k}_u(h) = \mathbf{k}_{\text{u\_staggered}}(h) e^{jh\Omega/2} \tag{5.12}$$

where $\mathbf{k}_u$ is the particle velocity coefficient that is no longer shifted in time by ($\Delta t/2$) relative to the pressure.

Equations (5.13) and (5.17) show only the derivation and only the last Equation (5.18) or Equation (5.19) are important for the calculations. To use the integral for derivation, the continuous notation is used. Using complex magnitude and phase angle of the coefficients, the harmonic functions for the pressure p and the velocity $\mathbf{u}$ with the angular frequency w (first harmonics) and time t for the one frame can be expressed as

$$p(t) = |k_p| \sin(\omega t + \arg(k_p)) \tag{5.13}$$

$$\mathbf{u}(t) = |\mathbf{k}_u| \sin(\omega t + \arg(\mathbf{k}_u)). \tag{5.14}$$

The average intensity can be calculated as the integral of product of pressure and particle velocity over time from 0 to T, dividing by T to take the average

$$\begin{aligned}\mathbf{I}_{\text{avg}} = \frac{1}{T}\int_0^T & |k_p| \sin(\omega t + \arg(k_p)) \\ & \times |\mathbf{k}_u| \sin(\omega t + \arg(\mathbf{k}_u)) \mathrm{d}t\end{aligned} \tag{5.15}$$

$$\mathbf{I}_{\text{avg}} = |k_p||\mathbf{k}_u| \cos(\arg(k_p) - \arg(\mathbf{k}_u))/2 \tag{5.16}$$

by modifying the expression using trigonometric functions, we achieve

$$\begin{aligned}\mathbf{I}_{\text{avg}} = |k_p||\mathbf{k}_u| \operatorname{Re}(&\cos(\arg(k_p) - \arg(\mathbf{k}_u)) \\ &+ j\sin(\arg(k_p) - \arg(\mathbf{k}_u)))/2\end{aligned} \tag{5.17}$$

$$\mathbf{I}_{\text{avg}} = \operatorname{Re}(k_p \overline{\mathbf{k}_u})/2 \tag{5.18}$$

The average intensity over multiple frames (M) including all harmonic frequencies H can be calculated using a simple principle of numerical integration with exploitation of non-staggered velocity as

$$\mathbf{I}_{\text{avg\_all}} = \frac{1}{M} \sum_{m=0}^{M-1} \sum_{h=1}^{H} \operatorname{Re}(k_p(m,h) \overline{\mathbf{k}_u(m,h)})/2. \tag{5.19}$$

To obtain suitable results using the compression method, the half-width of the complex exponential window basis should be an integer multiple of $N = 2\pi/(\omega \Delta t)$ (i.e. the input period T), where ω is the known driving fundamental angular frequency and Δt is known time step. The minimum value of the half-width is equal to one period, and therefore, we need at least $2N$ of signal samples for the complete calculation of one complex coefficient. However, if the signal already contains steady-state amplitudes, we can calculate an equally accurate coefficient from the N samples of the signal by mirroring the envelope (window function). Thus, the window function has a constant value in the processed signal frame. This is illustrated in Figure 5.12 (for the first harmonic frequency). The period is 106 time steps. The first coefficient is “mirrored” and calculated as the sum of even (2nd, 4th,...,) and odd (1st, 3rd,...,) coefficient for a signal length of one period. The second and third coefficient are computed from two periods. To reconstruct one point in time of the modeled

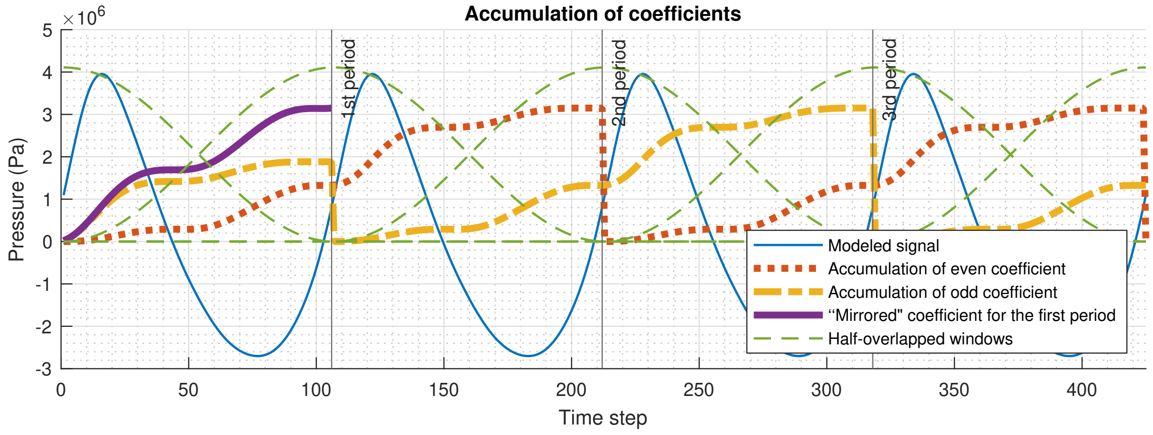


Figure 5.12: Accumulation of compression coefficients for the first and more periods.

signal, we need two coefficients whose weights are given by the overlapping envelopes. For special cases, thus for the first and last period, the first and last coefficients are duplicated.

The minimum number of memory cells c (single-precision floating-point numbers, 32 bits) required for computing intermediate results in one-time step for the stable parts of the signal is

$$c = 2H \tag{5.20}$$

as one complex number is needed per every harmonic frequency. Section 5.4.1 further describes the method of encoding a complex coefficient to 40 bits instead of 2×32 bits.

Compared to the original average intensity calculation procedure, the new approach does not need to save the pressure and velocity data to a file during the simulation, but needs more RAM. The calculation of the volume rate of heat deposition term Q is performed in the same way as in the case of non-use of the compression method (offline).

Resource consumption

The compression method described above is advantageous especially in terms of saving disk space, but also increases memory consumption. Consider the following several simulation scenarios representing clinical HIFU simulations.

Table 5.11 shows the basic simulation parameters and the comparison of the minimum file sizes required to calculate the Q term. The columns named N_x, N_y, and N_z are simulation domain sizes. The column named “Period” represents the number of simulation steps per period ($N = 1/(f\Delta t) = T/\Delta t$, where f is the known transducer driving frequency, and

Δt is the known time step). The parameters (period, harmonics and number of simulation steps) are calculated as part of creating the input simulation file, using the domain sizes, transducer driving frequency, sound speed, real size in z-axis, and the Courant–Friedrichs–Lewy (CFL) number [78].

The transducer driving frequency f is 1 MHz. Due to the nature of the input simulation data (heterogeneous absorbing material properties) and in order to obtain reasonable accuracy and stability of the simulations, the CFL number is set to 0.1. The real size in the z-axis z_{size} is 22 cm. The aperture diameter of the transducer bowl is 12 cm, and the radius of curvature is 14 cm. The reference sound speed c_{ref} is 1,524 $\mathrm{m\,s^{-1}}$. The number of points per z-axis $\Delta z = z_{\text{size}}/N_z$ and the number of points per wavelength $\text{PPW} = c_{\text{ref}}/(f\Delta z)$. The time step $\Delta t = 1/(f\lfloor \text{PPW}/\text{CFL} \rfloor)$ and therefore the period $N = 1/(f\Delta t)$. End time is calculated as $t_{\text{end}} = 2z_{\text{size}}/c_{\text{ref}}$ and the number of simulation steps, i.e. the total number of simulation steps from the beginning to the end of the simulation $N_t = \lfloor (t_{\text{end}}/\Delta t) \rceil$. The number of harmonics supported by the spatial grid is given by $H = \lceil 1 \times 10^{-6}(c_{\text{ref}}/(2\Delta z)) \rceil$.

The larger the grid size, the more accurate and usable results (more harmonics). As already mentioned in Section 2.1, for planning HIFU therapy, we need a reasonably high number of harmonic frequencies and the reasonably high spatial resolution. A typical scenario using a single supercomputer node is the case 4. Case 9 is approaching the limits of available supercomputers, using multiple nodes, if we do not want to wait a few days for the result of the simulation.

In case of the proposed method that uses compression, it is enough to store only one the Q term (a single 3-D matrix). Without the compression, the time series of pressure and velocity data time series is necessary to store (leading into storage of four 4-D matrices). Please note the fundamental difference in the amount of disk space required.

Table 5.11: Minimum file sizes for Q term calculation.

							File sizes generated during simulation		
	Domain size						without compression		with compression
Case	N_x	N_y	N_z	N	H	N_t	1 period	3 periods	1 and more periods
1	256	256	350	24	2	6929	8.49 GB	25.3 GB	88 MB
2	384	384	512	35	2	10105	40.6 GB	121 GB	288 MB
3	576	576	768	53	3	15302	207 GB	619 GB	972 MB
4	768	768	1024	70	4	20210	647 GB	1.94 TB	2.30 GB
5	960	960	1280	88	5	25407	1.59 TB	4.76 TB	4.50 GB
6	1152	1152	1536	106	6	30604	3.30 TB	9.90 TB	7.78 GB
7	1344	1344	1792	124	7	35801	6.14 TB	18.4 TB	12.3 GB
8	1536	1536	2048	141	8	40709	10.4 TB	31.2 TB	18.4 GB
9	1728	1728	2304	159	8	45906	16.7 TB	50.1 TB	26.2 GB

Table 5.12: RAM used for the on-the-fly average intensity calculation.

	with compression (2×32 bits)		with 40-bit compression (Section 5.4.1)	
Case	1 period	2 and more periods	1 period	2 and more periods
1	1.66 GB	3.06 GB	1.14 GB	2.01 GB
2	5.47 GB	10.1 GB	3.74 GB	6.62 GB
3	26.2 GB	49.6 GB	17.5 GB	32.1 GB
4	80.6 GB	154 GB	53.0 GB	99.1 GB
5	194 GB	374 GB	126 GB	239 GB
6	397 GB	770 GB	257 GB	490 GB
7	729 GB	1.42 TB	469 GB	901 GB
8	1.23 TB	2.41 TB	793 GB	1.53 TB
9	1.76 TB	3.44 TB	1.13 TB	2.18 TB

The RAM required for the on-the-fly average intensity calculation is given in Table 5.12. Here, we see that the amount of memory required depends on the number of harmonic frequencies. 40-bit compression refers to the reduction of memory (reduce format) used for complex coefficients from 64 to 40 bits, which is described in Section 5.4.1. The memory calculation is performed according to

$$\text{memory[MB]} = \frac{4N_yN_z(4\lceil N_xm\rceil nH + 3N_x)}{1024^2} \tag{5.21}$$

where N_x, N_y, and N_z are simulation domain sizes, H is the number of harmonic frequencies, n is 1 for one period or 2 for any number of periods larger than 1, and complex size multiplier m is equal to 2 for compression and 1.25 for 40-bit compression. The first number 4 represents the number of bytes per float while the second number 4 represents the number of compressed 3-D matrices, i.e. pressure and velocity for the x-, y- and z-axes. The number 3 represents uncompressed 3-D matrices for the time-averaged intensity in each Cartesian direction. The operating memory for the original pressure and velocity data is not included in Equation (5.21) as it is part of the simulation itself (described in the following paragraph).

Table 5.13 shows the common memory requirements for the remaining partial operations of the entire acoustic simulation process. They are the same with and without the compression. The first column is an estimate of the memory requirements of the simulation itself, without other operations, such as compression or postprocessing, calculated on the basis of simulation experiments. These values refer to the C++ OpenMP version of k-Wave. By the way, data obtained from the k-Wave researcher from the MPI versions of the simulations showed that the RAM requirements for the simulation itself are more than twice as large. The second column contains the memory requirements for the calculation of the Q term, which is performed as part of postprocessing, and the size corresponds to the three auxiliary matrices that are needed to copy the average intensity matrices due to the sensor mask (the sensor mask is a defined set of locations that will be sampled. In our examples, all points in the domain are sampled, but in general the sensor mask can be an arbitrary and sparse set of locations [65, 78]).

Table 5.13: Other common RAM requirements.

Case	RAM estimation used for simulation itself	RAM used for offline Q term calculation
1	3.50 GB	263 MB
2	11.0 GB	864 MB
3	36.5 GB	2.92 GB
4	87.0 GB	6.91 GB
5	170 GB	13.5 GB
6	294 GB	23.3 GB
7	467 GB	37.0 GB
8	697 GB	55.3 GB
9	992 GB	78.7 GB

A comparison of the total memory usage with partial operations of the whole acoustic simulation process using the new approach is shown in Figure 5.13. The graph shows the case of using 40-bit compression over one period.

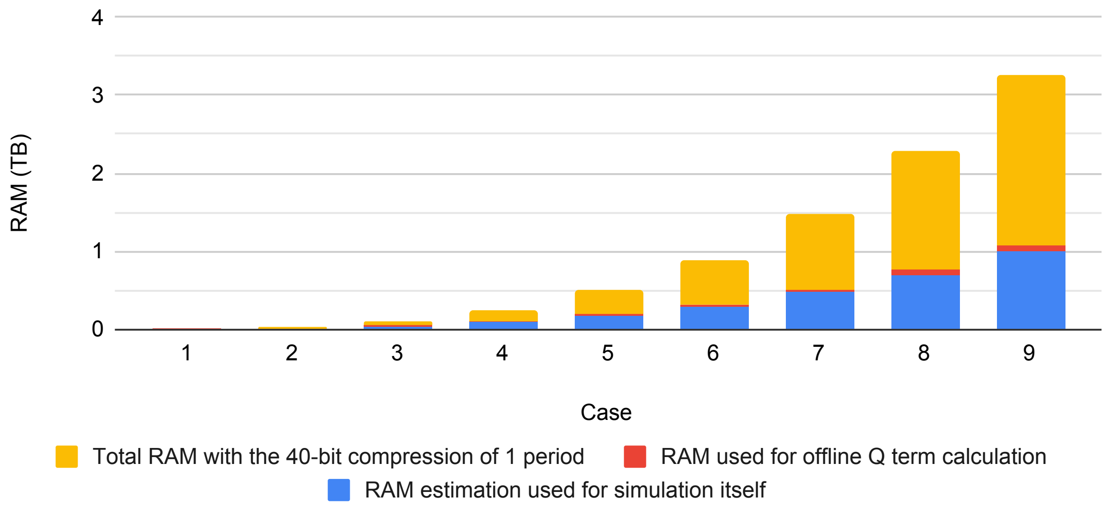


Figure 5.13: RAM memory requirements of the proposed method in TB.

It is difficult to evaluate what memory requirements the offline process of calculating the average intensity without the compression has. In the presented case, the amount depends on how much free memory is available on a given computing resource, and more RAM means faster reading and computing. The ideal amount of RAM corresponds to the size of the entire time series of pressure and velocity in the output files. In addition, the size of the auxiliary matrix for the FFT is needed. It is important that due to the time shifts, it is necessary to read at least the whole time series in time, while we can load and process blocks of different sizes. The total offline intensity calculation time does not depend only on the disk speed. In terms of memory layout of the stored 4-D data, it is more advantageous to load as much data as possible at once.

From the point of view of today's clusters, one node contains up to hundreds of GB of RAM. An example is the Barbora supercomputer in Ostrava (IT4Innovations), where each standard computational node is equipped with 192 GB of RAM [31]. Therefore, this memory limits us to using the compression if it wants to use only one node, e.g., with OpenMP technology. In the case of simulation on multiple nodes using the message passing interface (MPI), the operating memory is not such a problem. Conversely, the amount of free disk space and disk write speed can be a bigger complication, such as unavailability of disk space, its high price or disk space quota for the user (e.g., user space quota 10 TB on Barbora scratch filesystem).

If we consider, in the case of no compression, the possibility of storing the entire time series in RAM instead of in the file, in terms of the size of these data, it will not be overall advantageous. This would be possible for smaller simulations, but, for example, already in case 3 we would need at least 246 GB RAM (207 + 36.5 + 2.92, according to Tables 5.11 to 5.13), which is not realistic on one node of a common supercomputer. In addition, if we needed to calculate the average intensity from two or more periods. RAM requirements would multiply with each period. In the case of using the compression, the RAM requirements for calculations from two or more periods will be essentially the same.

Efficient coefficient encoding

To reduce the RAM memory required for temporary complex coefficients kept during accumulation (scalar product or computing intermediate results in a one-time step), we have proposed a method that uses 40 bits instead of 64 bits (2×32 bits) for the float complex number. There are many methods for lossy and lossless compression of float data, the best known of which are FPZIP and ZFP [53, 54]. These algorithms do not solve our problem because they are designed for single- or double-precision floating-point arrays. Furthermore, procedures for compressing blocks of complex numbers have been published. For example, an exponent is shared across the block of samples and the encoding box is used for the shared exponent to reduce quantization error [16]. Another approach is based on the principle that the number of bits per mantissa is determined by the maximum magnitude sample in the group and the exponent differences are encoded [85].

Our algorithm encodes one complex number independently of neighboring values and uses an approximate range of pressure and particle velocity values. The assumption is that we have at the input a complex number whose exponents of the imaginary and real components do not differ significantly. Thanks to this and the assumed maximum range of the values, only 4 bits are used to encode the larger exponent. The second exponent is stored as the difference in the shifted mantissa. The format of the 40-bit encoded complex number is shown in Table 5.14. Mantissa is composed from: 0–16 zero bits, 1 flag bit, and 0–16 data (mantissa or fraction) bits, in total it consists of exactly 17 bits. Number of zero bits means exponent shift from the stored exponent. For comparison, the Table 5.15 shows the standard format (IEEE-754) for encoding 2×32-bit complex number [30].

The encoding procedure is illustrated by Algorithm 1. The analogous decoding procedure is illustrated by Algorithm 2.

The number of bits for the mantissa can potentially be further reduced, however, $16 + 1$ bits, thus a total of 40 bits, is practical in terms of memory alignment to bytes and acceptable errors. Within this article, the relative normalized ℓ_∞ error of the Q term calculation caused by compression up to about 1 % is considered acceptable [6].

Table 5.14: 40-bit complex floating-point format.

Type of data	real sign	imag. sign	real mantissa	imag. mantissa	shifted expo-nent
Number of bits	1	1	17	17	4

Table 5.15: Standard 2×32-bit complex floating-point format (IEEE-754).

Type of data	real sign	real ex-ponent	real mantissa	imag. sign	imag. expo-nent	imag. mantissa
Number of bits	1	8	23	1	8	23

5.4.2 Experimental evaluation

The goal of the experimental numerical simulations was to investigate how the compression method affects the simulation execution time, the consumption of computing resources, and the numerical accuracy for realistic HIFU simulations.

Within the time measurement of the experimental simulations, the most important time is the time of the simulation phase itself (iteration of simulation steps), which can range from a few minutes to days, depending on the size of the simulation domain. The purpose of this measurement is to show that applying compression does not slow down the simulation process. Furthermore, we are interested in the time of the postprocessing phase, where the calculation of the Q term and offline calculation of the average intensity takes place.

Considering the consumption of computing resources, we are mainly interested in the consumption of RAM and disk space. The aim is to confirm the assumption that despite the higher demands of the compression method on the operating memory, the total memory requirements, including disk space, are significantly smaller.

Finally, the evaluation of compression errors is performed, both for the Q term, the average intensity, and for the outputs of the thermal simulation. The purpose is to show that the number of different points of ablated tissue is ideally the same with and without the use of compression. The proposed method was implemented within the k-Wave toolbox [78]. Simulations using the k-Wave (k-space pseudospectral methods) were experimentally verified with phantoms and biological tissues [18, 58, 59, 84]. The compression method was implemented in both C++ OpenMP and CUDA versions, but due to the extent of the measured data, this work contains detailed measured results of only the OpenMP version. The original version of the intensity and Q term calculation was implemented only in the MATLAB version. To compare the performance of both approaches, the calculation was ported to C++ to the postprocessing stage. Spatial gradients are computed using Fourier transform. The compression algorithm was implemented in a parallel environment and is performed during the simulation.

Acoustic simulations were performed on one node of the Barbora supercomputer cluster, where 36 processor nodes (2× Intel Cascade Lake 6240, 2.6 GHz) and at least 192 GB of RAM are available. For reading and writing files, Barbora provides the Luster shared

Algorithm 1 The 40-bit coefficient encoding procedure

```
1: Get the real and imaginary part of the input float complex number and their sign bits.
2: Get 8-bit exponents and subtract $e$ constant from them which allow the exponent to
   be stored for only 4 bits.
   In case of the acoustic pressure:
      $e = 138$, max exponent is $2^{26}$ $(15 + 138 = 153,\ 153 - 127 = 26)$
      maximal encoding value is $2^{26-16} \times$ 0x1FFFF $= 134216704$
      minimal encoding value is $2^{26-16-15} \times$ 0x1 $= 0.03125$
   In case of the particle velocity:
      $e = 114$, max exponent is $2^{2}$ $(15 + 114 = 129,\ 129 - 127 = 2)$
      maximal encoding value is $2^{2-16} \times$ 0x1FFFF $= 7.99993896484375$
      minimal encoding value is $2^{2-16-15} \times$ 0x1 $= 0.0000000018626451492309570312 5$
3: Get 23-bit mantissas and set their default shift to the right by 6 bits - these least
   significant bits will be discarded.
4: Find the higher exponent to be saved. Add the difference between the larger and smaller
   exponents to the shift to the right for the mantissa of a number with a smaller exponent.
5: Crop exponents less than zero and the right shifts greater than 23. Apply right shifts
   to the mantissas.
6: Round the least significant bits in the mantissas.
7: Set 1 flag bit for the shifted mantissa with a smaller exponent.
8: Check exponent overflow, and set maximum values if necessary.
9: Store the output data at 40 bits (using bitwise operators) as shown in Table 5.14.
```

filesystems. On the positive side, it provides a theoretical maximum throughput of $5\,\mathrm{GB\,s^{-1}}$ ($38\,\mathrm{GB\,s^{-1}}$ with burst mode) [31]. Unfortunately, the fact that the filesystem is shared does not guarantee this throughput. Experimental simulations have shown that the times of such calculation phases, in which large files were written or read, sometimes differed significantly (e.g., by a factor of 10).

Due to the available computing resources, four sizes of the simulation domain between $256 \times 256 \times 350$ and $768 \times 768 \times 1024$ were tested, corresponding to cases 1–4 presented in Section 5.4.1. The average intensity was calculated only in the last simulation period. The input simulations material properties such as sound speed, attenuation, density and B/A (nonlinearity parameter) were generated from the AustinWoman electromagnetic voxels model [60]. The heterogeneous parameters were specified for every grid point independently wherever enabled by the simulation tool. Individual book values for the material properties in the human body were used [28].

Tables 5.16 and 5.17 show the measured performance data for each simulation case without the use of compression (N), with the use of compression (C) and with the use of compression using 40-bit coding (C 40-bit). The "file size" represents the size of the file that must be used during the simulation. The RAM memory is divided into two columns. The first is the memory needed for the simulation and sampling itself. The total RAM corresponds to the amount of memory used in the whole simulation case, including compression and postprocessing. In the case no compression is used, the effort is to use the maximum amount of free RAM so that the data for offline calculation of the average intensity within the postprocessing phase is read from the file as quickly as possible, to make the compar-

Algorithm 2 The 40-bit coefficient decoding procedure

```
1: Get the mantissas, signs, and exponent from the input 40-bits value (using bitwise
   operators).
2: Shift the mantissas 6 bits to the left (we now have 23-bit mantissas).
3: Add the $e$ constant to the exponents ($e = 138$ for the acoustic pressure, $e = 114$ for the
   particle velocity).
4: For the both mantissas (mR, mI) and exponents (eR, eI):
5: if the mantissa is zero then
6:    set the exponent zero (zero mantissa means zero float number),
7: else
8:    find the index of the most left one bit in mantissa using the specialized function
      (_BitScanReverse or __builtin_clz)
9:    and shift the mantissa according to the index value to the left (mR <<= 23 - index)
10:   and recompute the final exponent by the index (eR -= 22 - index).
11: end if
12: Put together the output complex float numbers using bitwise operators at 2×32 bits
    from signs, mantissas, and exponents.
```

ison as fair as possible. In the postprocessing phase, differences can be seen between the times when only the Q term calculation is performed and when the average intensity is also calculated. Given the overall simulation time, these values are negligible. To determine the variability of the total times, the simulation time was measured for every 5 % of the total simulation steps. The Coefficient of Variation of simulation times $c_\mathrm{v} = \sigma/\mu$, where σ is the standard deviation and μ is the mean, was about 9 %. Based on the measurement results, we can say that the total simulation times with and without compression for the given domain sizes do not differ significantly (variability is about 3 %). Due to the fact that only the last period was sampled for the calculation of the average intensity, which is approximately 0.35 % of all simulation steps, the total times are not significantly affected by this sampling. However, we can also see the average iteration times in which the sampling takes place in the table, and we can see that compression is faster than writing to the files. The average non-sampling iteration time of a given simulation case is calculated as the ratio of the sum of individual iteration times to the number of iterations, within the simulation, when sampling was not performed. The average sampling iteration time is calculated as the ratio of the sum of individual iteration times to the number of iterations, within the simulation, when sampling was performed. In particular, the iteration times are 2 to 10 times faster with the compression than without the compression. In terms of the memory used - the sum of the file size and RAM, the new approach is considerably more economical. A disadvantage of the new approach may be the need for a minimum amount of free RAM depending on the number of coded harmonics.

The numerical error caused by the compression is expressed as relative normalized ℓ_∞ error, i.e., maximum absolute difference between non-compression (calculated without the use of the compression) and compression data (calculated using compression) divided by the absolute maximum value of non-compression data. The maximum values were calculated across the entire domain. So for Q term and the average intensity over the whole simulation 3-D space and for the pressure and the velocity in addition also over the sampling simulation time (4-D). Table 5.18 shows the error values in the percent of the volume rate of heat deposition (Q term), the average intensity for the individual axes (I_{x_avg}, I_{y_avg}, and I_{z_avg}),

Table 5.16: Comparison of memory usage in individual cases of simulations on one node of the Barbora supercomputer cluster, with 36 processor cores (2× Intel Cascade Lake 6240, 2.6 GHz) and at least 192 GB of RAM.

Case	Method	File size	Simulation + sampling RAM	Total RAM including postprocessing
1	N	8.54 GB	**3.44 GB**	14.1 GB
1	C	**88 MB**	5.08 GB	5.34 GB
1	C 40-bit	**88 MB**	4.55 GB	**4.82 GB**
2	N	40.6 GB	**11.0 GB**	61.7 GB
2	C	**288 MB**	16.5 GB	17.3 GB
2	C 40-bit	**288 MB**	14.7 GB	**14.7 GB**
3	N	207 GB	**36.9 GB**	168 GB
3	C	**972 MB**	63.2 GB	66.1 GB
3	C 40-bit	**972 MB**	54.4 GB	**57.3 GB**
4	N	648 GB	**87.1 GB**	168 GB
4	C	**2.30 GB**	168 GB	175 GB
4	C 40-bit	**2.30 GB**	140 GB	**147 GB**

the acoustic pressure (p), and the non-staggered particle velocity for the individual axes (u_x, u_y, u_z). If we take into account the accuracy of the float data type (∼7.2 decimal digits), then the intensity errors are very small. Higher error values for the Q term are most likely due to the type of gradient calculation, where many products are performed between FFT and inverse fast Fourier transform (IFFT). In the case of compression, especially with 40-bit coding, the error generally increases with the number of samples in the period.

The CUDA version is fundamentally limited especially by the amount of memory available on the GPU. The amount of memory listed in Table III, in the first column (approximately) should also be available on the GPU and this is quite a major and fundamental limitation. The compression with calculating the average intensity it is not performed on a GPU and uses a CPU and a RAM connected to it. The compression on the GPU does not make sense yet, as in one iteration its computational time is negligible compared to the simulation and in addition it would need the amount of RAM similar to the sizes available to CPUs on the GPUs, which is not yet true.

To be able to meaningfully evaluate the magnitudes of errors caused by the compression, the Q term is applied to the calculation of the thermal simulation. This will show how large the differences will be caused by compression in the heat applied to the tissue, and specifically how the ablated tissue will differ. Thermal simulations were performed in MATLAB using the `kWaveDiffusion` function for the time-domain solution of the Pennes' bioheat equation.

The input parameters of the thermal simulation are shown in Table 5.19. The heating with the Q term calculated in the acoustic simulation was set to 10 s, the cooling time with the $Q = 0$ was set to 20 s.

The results of the thermal simulation shown in Table 5.20 are the temperature after heating, the temperature after cooling, cumulative equivalent minutes relative to $T = 43\,°\mathrm{C}$ (CEM43) in %, maximum absolute value of CEM43 for cases without compression, and

Table 5.17: Comparison of computational times in individual cases of simulations on one node of the Barbora supercomputer cluster, with 36 processor cores (2× Intel Cascade Lake 6240, 2.6 GHz) and at least 192 GB of RAM.

Case	Method	Postprocessing time [seconds]	Simulation time [seconds]	Average non-sampling iteration time [seconds]	Average sampling iteration time [seconds]
1	N	11.4	906	0.11	0.82
1	C	1.05	814	0.11	**0.27**
1	C 40-bit	**0.27**	**767**	0.11	0.34
2	N	57.4	3,834	0.36	3.63
2	C	**1.18**	**3,665**	0.36	**0.81**
2	C 40-bit	2.73	3,816	0.36	1.11
3	N	529	20,456	1.33	45.5
3	C	6.64	21,383	1.33	**3.13**
3	C 40-bit	**6.32**	**19,532**	1.33	4.54
4	N	1,880	**74,531**	3.51	53.5
4	C	6.56	78,232	3.51	**9.42**
4	C 40-bit	**7.22**	74,849	3.51	12.4

the number of different points (also expressed as ablated volume in mm^3) of binary matrix representing ablated tissue, where CEM43 $\geq$ 240 min.

A very important result of the thermal numerical simulations is the number of ablated tissue points. This value is essentially the same without and with the use of compression. The maximum thermal dose ℓ_∞ errors are around 0.5 %, which is negligible. The temperature differences are also minimal.

Figures 5.14 to 5.18 show sections of the output 3-D data in the center of the x-axis for the case 4. Some of the figures also include a zoomed-in figure cutout from the focused region. Average intensity in z-axis and volume rate of heat deposition is shown in Figure 5.14, errors caused by the compression in Figure 5.15 and Figure 5.16. The thermal dose in CEM43 units is shown in Figure 5.17 on the top and the ablated tissue (CEM43 $\geq$ 240 min) is shown in red on the bottom, where shades of gray show the mass density derived from the AustinWoman voxel model. The thermal dose errors caused by compression can be seen in Figure 5.18, the compression error is on the top, the 40-bit compression error on the bottom. The absolute thermal dose errors caused by compression shown in Figure 5.18 are, in fact, only very small relative errors (0.24 % top and 0.036 % bottom) due to the very large maximum value of thermal dose 2.57×10^{13} (see Table 5.20).

The overall results of the experimental evaluation showed that the application of the new compression method for calculating the average intensity brings significant savings in disk space, while other demands on computing resources are comparable. The quality of the outputs is not fundamentally affected by the compression and is comparable to the outputs without compression.

Table 5.18: Relative errors caused by compression. The particle velocities (u_x, u_y, u_z) are non-staggered.

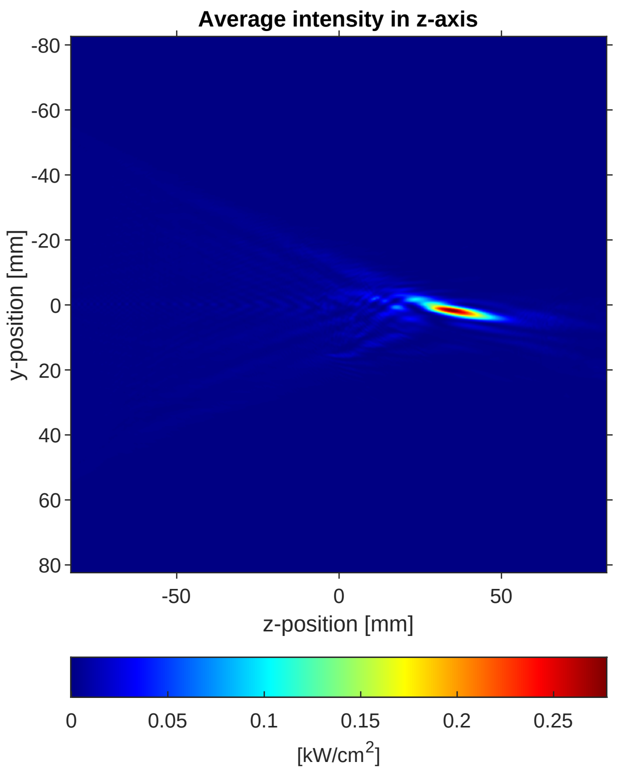
Case	Q	$I_{x_{\text{avg}}}$	$I_{y_{\text{avg}}}$	$I_{z_{\text{avg}}}$	p	u_x	u_y	u_z
Compression ℓ_∞ error in % (method C in Table 5.16)								
1	0.00030	0.000045	0.000040	0.000031	0.0068	0.042	0.040	0.012
2	0.0092	0.000066	0.000058	0.000041	0.018	0.083	0.084	0.010
3	0.0098	0.000066	0.000094	0.000071	0.023	0.065	0.061	0.013
4	0.016	0.000090	0.000089	0.000085	0.014	0.048	0.044	0.010
40-bit compression ℓ_∞ error in % (method C 40-bit in Table 5.16)								
1	0.034	0.0045	0.0031	0.0038	0.0068	0.043	0.040	0.013
2	0.46	0.0046	0.0042	0.0043	0.020	0.083	0.085	0.013
3	0.85	0.0060	0.0049	0.0063	0.024	0.067	0.063	0.015
4	1.21	0.0093	0.0067	0.0068	0.017	0.051	0.046	0.013

Table 5.19: Thermal simulation parameters.

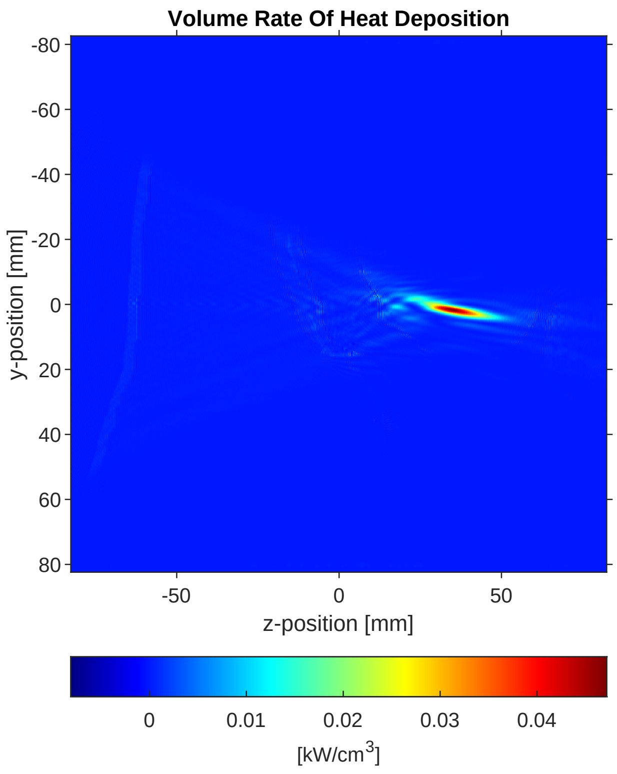
The initial temperature	37 °C
Density	1,020 kg m^{-3}
Thermal conductivity	0.5 W m^{-1} K
Specific heat capacity	3,600 J kg^{-1} K
Number of heating time steps	100
Number of cooling time steps	200
Size of the time step (dt)	0.1

Table 5.20: Thermal simulation errors.

Case	Maximum temperature after heating error [°C]	Maximum temperature after cooling error [°C]	Maximum thermal dose (CEM43) ℓ_∞ error [1 %]	Absolute maximum thermal dose without the compression [CEM43]	Number of different points of ablated tissue (ablated volume in mm^3 in brackets)
Compression error					
1	0.0001	0.000019	0.0013	2.82×10^7	0 (0)
2	0.00023	0.000019	0.0082	8.06×10^9	0 (0)
3	0.00029	0.000027	0.0089	4.37×10^{12}	0 (0)
4	0.00028	0.000038	0.010	2.57×10^{13}	0 (0)
40-bit compression error					
1	0.0094	0.000088	0.049	2.82×10^7	0 (0)
2	0.016	0.00011	0.62	8.06×10^9	0 (0)
3	0.020	0.00019	0.48	4.37×10^{12}	1 (0.0235)
4	0.020	0.00015	0.42	2.57×10^{13}	0 (0)

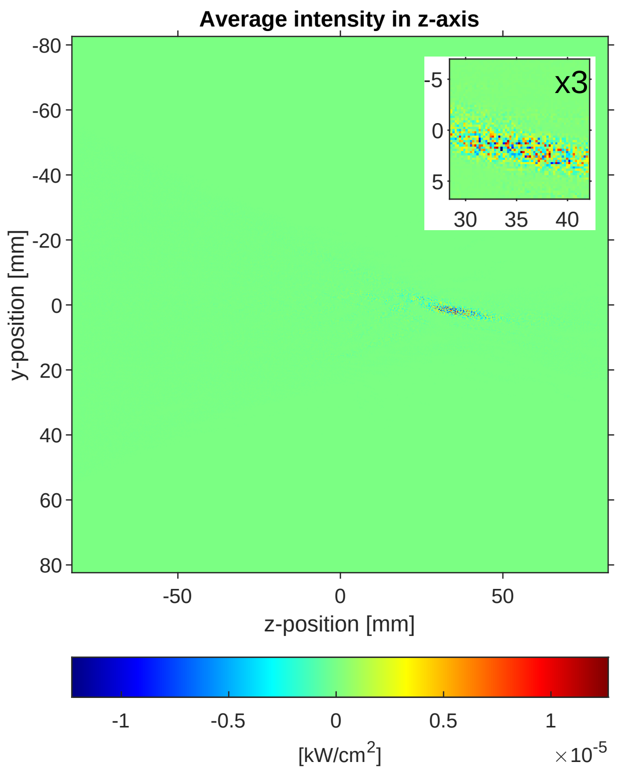


(a)

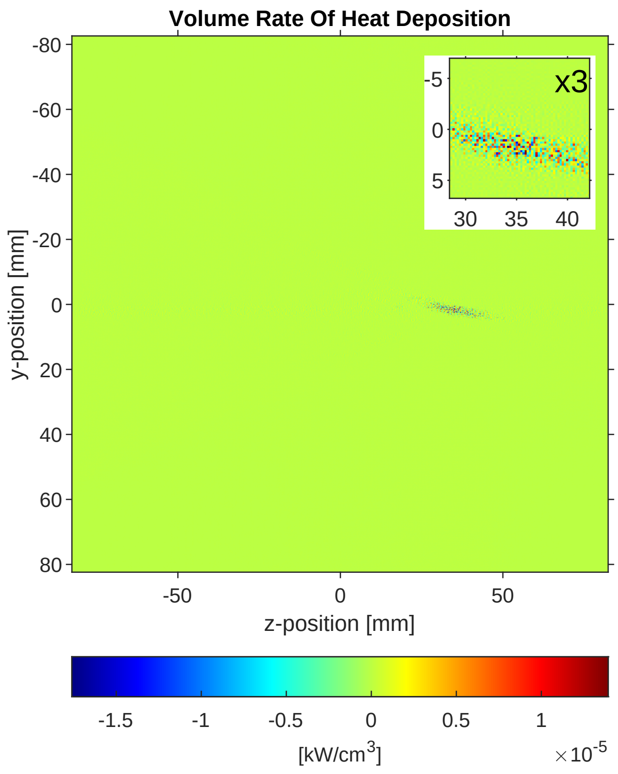


(b)

Figure 5.14: Visualization of (a) average intensity in z-axis and (b) volume rate of heat deposition without the compression.

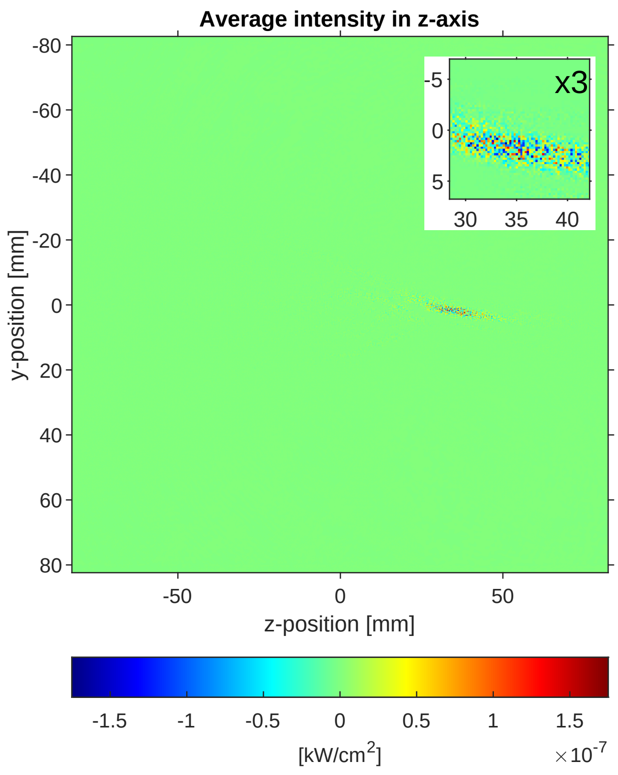


(a)

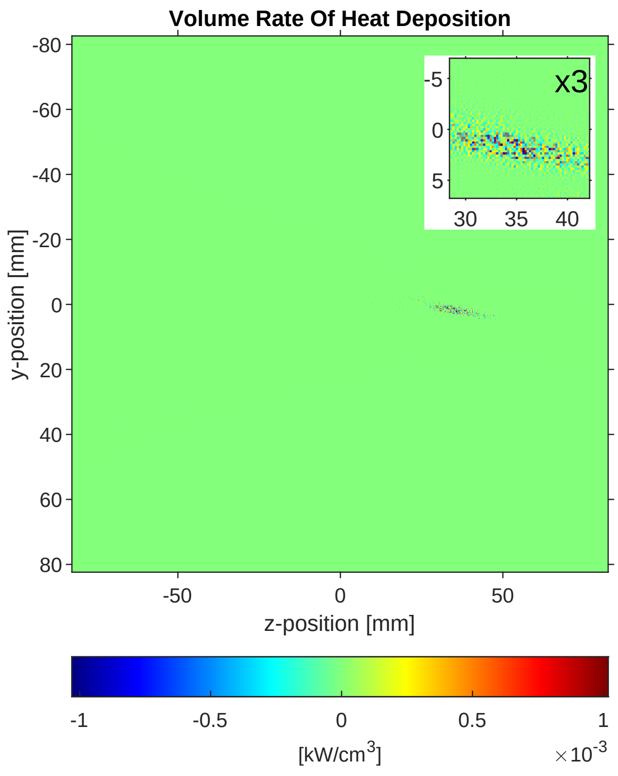


(b)

Figure 5.15: Visualization of (a) average intensity error in z-axis and (b) volume rate of heat deposition with the compression.

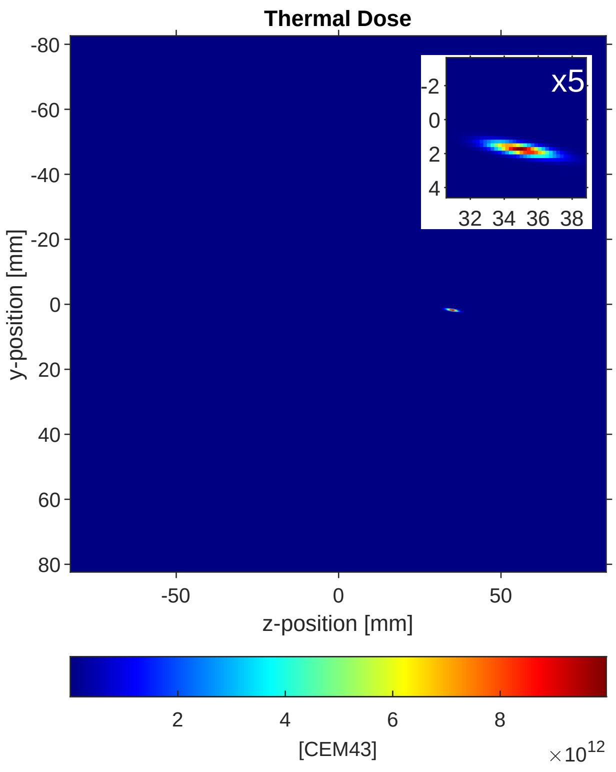


(a)

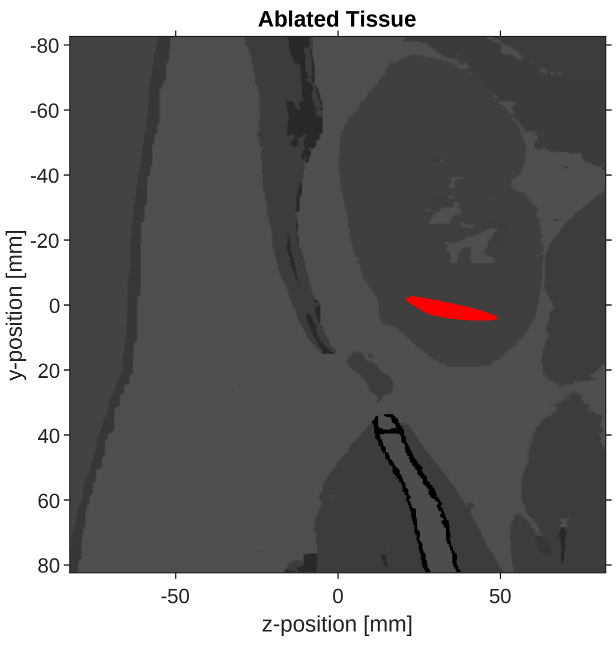


(b)

Figure 5.16: Visualization of (a) average intensity error in z-axis and (b) volume rate of heat deposition with the 40-bit compression.

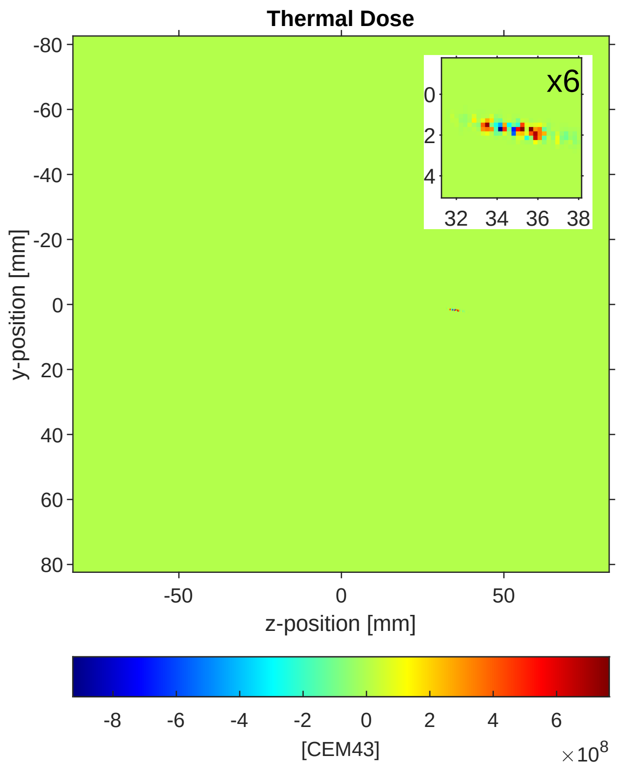


(a)

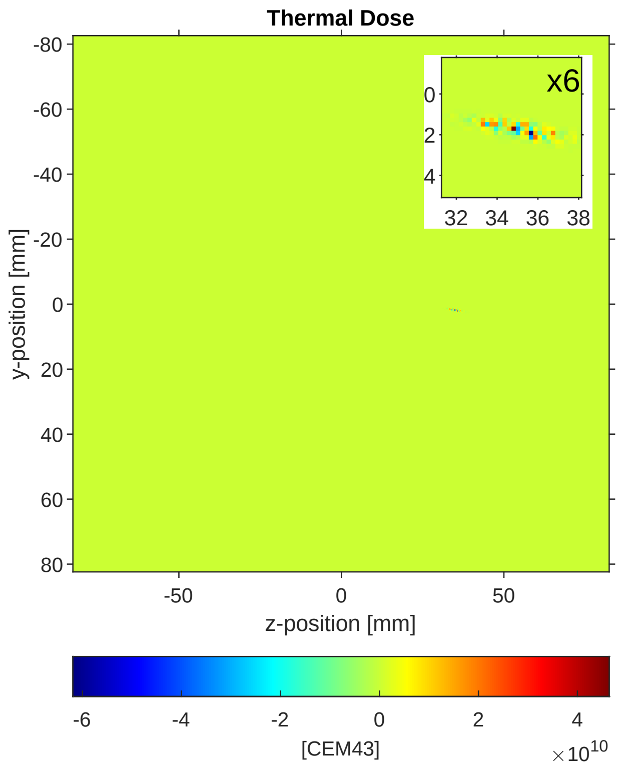


(b)

Figure 5.17: Visualization of (a) thermal dose and (b) ablated tissue without use of the compression.



(a)



(b)

Figure 5.18: Visualization of the thermal dose errors. (a) Compression error and (b) 40-bit compression error.

5.5 Validation and scientific contribution

In Section 5.3 a novel on-the-fly compression method for HIFU simulations outputs is introduced and validates the first part of the hypothesis in the following way: The method is focused on 1-D time-varying data series. Furthermore, the results of a comparison with other state-of-the-art compression methods are presented. The compression ratio at the maximum quality that this method enables is comparable to other state-of-the-art compression methods, e.g. for compression of audio signals and significantly exceeds some. The quality of outputs compressed by the method is comparable to the quality of compressed by other state-of-the-art methods within general measures such as MSE or PSNR. From the point of view of users, according to the Section 4.3, the ℓ_∞-errors caused by the new compression can be evaluated as acceptable, as they are at most in the order of percent.

The remaining part of the hypothesis is validated in Section 5.4 in the following way: The compressed intermediate results are used for the on-the-fly calculation of the time-averaged acoustic intensity. This quantity is further used to calculate the thermal simulation from which the ablated tissue is then determined. The worst error in the performed experimental evaluation caused by compression in ablated volume in is $0.0235\,\mathrm{mm}^3$ a maximum thermal dose (CEM43) ℓ_∞-error is $0.62\,\%$. It is verified here that the quality of the outputs compressed by the new method is also acceptable from the point of view of the application of these outputs. The new procedure in the calculation of acoustic intensity significantly reduces the consumption of disk space by up to $99\,\%$. At the same time, this procedure has almost the same demands on total RAM consumption. The computing time is comparable and in the case of longer simulations, the method can even reduce this time. Thus, the last condition of the hypothesis in the field of computing resources is also fulfilled. Finally, the entire hypothesis was proven by the above experiments, both parts of hypothesis on the same types of datasets. The hypothesis worked on all the data it was tested on and it can be assumed to be valid generally; however, further experiments would be needed for this.

In terms of contribution to science, the published methods allow users (scientists, doctors) who use demanding simulations not only in the field of HIFU, to make more economical calculations, especially in simulations, which use a staggered-grid pseudospectral time-domain method.

5.6 Possible applications

Ultrasound simulations are widely used, for example, in the healthcare industry. Due to the effect of nonlinearity, absorption, refraction, and scattering, the position of the focus and the shape of the focal volume can be significantly affected, unlike the propagation of an ultrasound wave, for example, in water or in a homogeneous environment. Using simulations, it is possible to model these effects, and thus predict wave propagation more accurately. And specifically to these simulations, it is possible to apply the presented compression method, which significantly saves the disk space required for storing large-scale output simulation data. The large-scale data represent a fundamental problem for better deployment of simulations in practice.

The main application of the proposed method is presented in experiments with thermal simulations. HIFU is used for fast precise localized non-invasive tissue destruction, for example, in the treatment of cancer. The HIFU has been used in clinical trials, e.g., for the treatment of tumors in the prostate, kidney, liver, bone, breast, and brain. The destruction of tissue in the focus of the ultrasound beam is caused by thermal and cavitation effects.

For the most accurate predictability of thermal effects, thermal simulations are performed. However, only if the absorption of ultrasound waves is accurately captured in models and simulations, a dose threshold in cumulative equivalent minutes leading to tissue devitalisation can be determined. Therefore, these thermal simulations are directly dependent on the acoustic simulations, which are able to calculate the absorption. For accurate modeling and simulation of the absorption, it is advisable to use general, i.e. more accurate methods of calculating thermal simulation inputs, namely the heat deposition term, instead of using other approximations. The improved calculation of the thermal simulation inputs, i.e. first the average intensity and then the volume rate of heat deposition with the new proposed compression method, brings significant savings in the use of computing resources, and thus significantly positively pushes the boundaries of usability in practice.

Another possible application of the compression is, for example, in the area of fast visualizations of simulation data. By the way, this has already been partially implemented in compression experiments, but it has not yet found direct use for scientists. In the case of a more significant deployment of HIFU simulations in practice, it could be useful. Reading the compression coefficients instead of the complete uncompressed data from the files will significantly reduce the required data flow, and thus the time required for fast visualization. For visualizations, much larger distortions caused by compression, which are not detectable by human vision, could theoretically be acceptable. This could again achieve higher data flow and faster decompression. If the goal of the visualizations is to quickly display time series of selected points in space or parts of the domain, the proposed compression is also a suitable candidate for this, as it currently compresses individual points independently of others and in parallel.

The application of effectively calculated average intensity, which could be explored and tested, for example, in the field of 2-D/3-D reconstruction. Some of the publications in which I participated deal with 3-D reconstruction of fractured long bones from plain 2-D radiographs [44, 45, 46, 47, 48, 49]. Based on the 3-D statistical shape and intensity models and two 2-D X-ray images, a 3-D model is calculated. If the subject was not the bone registration, but e.g., registration of ablated tissue of the HIFU procedure, perhaps it would be possible, analogously to two 2-D X-ray images, to perform two cheap and fast 2-D simulations in different directions and with these the registration would then be performed. Precisely 3-D statistical intensity models could be created using many spatial simulations, which would not be so expensive and unavailable due to the use of the average acoustic intensity calculation approach proposed in this work.

The method may also be useful in other industries. Wherever it is useful to perform acoustic simulations, there are harmonic functions at the input of these simulations and where the user is interested, for example, in the average acoustic intensity itself. The possible use of simulations is offered, for example, in the production of acoustic instruments, audio technology or for the propagation of sound in water pipes.

Chapter 6

Conclusion

This work introduces a new compression method within ultrasound simulations and the application of this method for calculating the time-averaged acoustic intensity vector during ultrasound simulations performed using a staggered-grid PSTD method.

An efficient compression algorithm for HIFU simulation data is proposed, and offline experiments were performed to evaluate it. Is it shown that our method produces very useful results. The important stable parts of the simulation signals are compressed with very small distortion (0.1 %) at compression ratios over 80 %. The very short transient parts of the signals are compressed with acceptable errors.

The results of the application of the proposed method are based on the improvement of the important intermediate step in the acoustic simulations - calculation of the average intensity. The presented approach calculates it using the compression coefficients obtained on-the-fly during the simulation, avoiding saving of the intermediate results of acoustic pressure and particle velocity to the disk during the simulation, as used in state-of-the-art approaches. The method has significant advantages over the state-of-the-art simulation with uncompressed output. The main advantage is largely (up to 99 %) reduced consumption of precious disk space during the simulation, which may significantly reduce the price of the computational platform and, in some existing configurations of such platforms, it can even present an enabling factor for execution of the simulations. At the same time, the presented method generally has approximately the same demand for RAM and in longer simulations it can even reduce the computational time. Moreover, the compression errors in the proposed method are negligible.

While acoustic simulation (without the intensity calculation) with the compression has higher RAM requirements than without the compression, it brings significant disk space savings. From the standpoint of supercomputers, the extensive consumption of fast I/O disk storage space is a much bigger problem than the need for RAM. In terms of disk space requirements, the new method is significantly more economical.

Through experimental numerical simulations, it has been shown that the average iteration time during sampling is 2–10 times shorter, which can reduce the simulation time in some cases. The compression does not adversely affect the overall simulation time.

The accuracy of the new method was evaluated using thermal simulations. Using the new method, essentially the same results were achieved in the determination of the ablated tissue as with other approaches. The maximum errors are around 0.5 % for thermal dose and 0.02 °C for temperature after heating, which are minimal or even negligible. The accuracy is equivalent to the state-of-the-art.

The future work may show that the new method could be applicable for signals of a similar nature, e.g., for electromagnetic radio waves, where the problem of immediate calculation of intensity is the mutual time shift of the signals in time. Future work could focus, e.g, on MPI implementation of the compression. This implementation has not been done yet. Here, it could be proven that the higher demand for operational memory is not such a big problem, since there should be enough of it on each node. The problem without using compression will be expensive disk space and disk access policy, i.e. shared user access and unguaranteed throughput. However, further optimizations of the compression algorithm in the area of RAM utilization would also be useful. Furthermore, it is possible to focus, for example, on the optimization of overlapped window functions and bases or the reduction of the operational memory for higher harmonic frequencies, where the values are usually very small but still very important. Extending compression to the 2-D or 3-D space and using spatial correlation of the data is another area in which this work could be followed.

Bibliography

[1] de A. Freitas, M.; Jimenez, M. R.; Benincaza, H.; et al.: A new lossy compression algorithm for ultrasound signals. In *2008 IEEE Ultrasonics Symposium*. IEEE. 2008. ISBN 978-1-4244-2428-3. pp. 1885–1888. doi: 10.1109/ULTSYM.2008.0464.
URL http://ieeexplore.ieee.org/document/4803348/

[2] Abdulbaqi, A. S.; Najim, S. A.-d. M.; Mahdi, R. H.: Robust multichannel EEG signals compression model based on hybridization technique. *International Journal of Engineering & Technology*. vol. 7, no. 4. 2018: pp. 3402–3405.

[3] Ainsworth, M.; Tugluk, O.; Whitney, B.; et al.: Multilevel Techniques for Compression and Reduction of Scientific Data—The Multivariate Case. *SIAM Journal on Scientific Computing*. vol. 41, no. 2. 2019: pp. A1278–A1303. doi: 10.1137/18M1166651.
URL https://doi.org/10.1137/18M1166651

[4] Angla, C.; Larrat, B.; Gennisson, J.-L.; et al.: Transcranial ultrasound simulations: A review. *Medical Physics*. 09 2022. doi: 10.1002/mp.15955.

[5] Arya, S.; Mulla, Z. D.; Kupesic Plavsic, S.: Role of pelvic ultrasound simulation. *The Clinical Teacher*. vol. 15, no. 6. 2018: pp. 457–461. doi: https://doi.org/10.1111/tct.12714.
URL https://onlinelibrary.wiley.com/doi/abs/10.1111/tct.12714

[6] Aubry, J.-F.; Bates, O.; Boehm, C.; et al.: Benchmark problems for transcranial ultrasound simulation: Intercomparison of compressional wave models. *The Journal of the Acoustical Society of America*. vol. 152, no. 2. aug 2022: pp. 1003–1019. doi: 10.1121/10.0013426.
URL https://doi.org/10.1121%2F10.0013426

[7] Bakaric, M.; Martin, E.; S. Georgiou, P.; et al.: Experimental study of beam distortion due to fiducial markers during salvage HIFU in the prostate. *Journal of Therapeutic Ultrasound*. vol. 6, no. 1. 2018. ISSN 2050-5736. doi: 10.1186/s40349-018-0109-3.

[8] Barina, D.; Najman, P.; Kleparnik, P.; et al.: The parallel algorithm for the 2D discrete wavelet transform. In *Ninth International Conference on Graphic and Image Processing (ICGIP 2017)*, vol. 10615, edited by H. Yu; J. Dong. International Society for Optics and Photonics. SPIE. 2018. page 106151P. doi: 10.1117/12.2302881.
URL https://doi.org/10.1117/12.2302881

[9] Blelloch, G. E.: Introduction to Data Compression. *Computer Science Department, Carnegie Mellon University*. 2013.

[10] Bosi, M.; Goldberg, R. E.: *Introduction to Digital Audio Coding and Standards*. Norwell, MA, USA: Springer Science+Business Media, Kluwer Academic Publishers. 2002. ISBN 978-1-4613-5022-4.

[11] Cardoso de Cardoso, G.: *Compression, Estimation, and Analysis of Ultrasonic Signals*. PhD. Thesis. Illinois Institute of Technology. 2005.

[12] Cetin, E.; Karaboce, B.; Durmus, H. O.; et al.: Temperature Effect of HIFU with Thermal Dose Estimation. *2020 IEEE International Instrumentation and Measurement Technology Conference (I2MTC)*. 2020: pp. 1–6. doi: 10.1109/I2MTC43012.2020.9128841.
URL https://ieeexplore.ieee.org/document/9128841/

[13] Chatillon, S.; Loyet, R.; Brunel, L.; et al.: Applications of intensive HIFU simulation based on surrogate models using the CIVA HealthCare platform. *Journal of Physics: Conference Series*. vol. 1761, no. 1. jan 2021: page 012007. doi: 10.1088/1742-6596/1761/1/012007.
URL https://doi.org/10.1088/1742-6596/1761/1/012007

[14] Chen, J.; Wan, L.; Liang, X.; et al.: Accelerating Multigrid-based Hierarchical Scientific Data Refactoring on GPUs. *CoRR*. vol. abs/2007.04457. 2020. 2007.04457.
URL https://arxiv.org/abs/2007.04457

[15] Cheng, P.-W.; Shen, C.-C.; Li, P.-C.: Ultrasound RF channel data compression for implementation of a software-based array imaging system. In *2011 IEEE International Ultrasonics Symposium*. IEEE. Oct 2011. ISBN 978-1-4577-1252-4. ISSN 1051-0117. pp. 1423–1426. doi: 10.1109/ULTSYM.2011.0352.
URL http://ieeexplore.ieee.org/document/6293721/

[16] Choo, Y. F.; Evans, B. L.; Gatherer, A.: Complex block floating-point format with box encoding for wordlength reduction in communication systems. In *2017 51st Asilomar Conference on Signals, Systems, and Computers*. IEEE. 2017. ISBN 978-1-5386-1823-3. pp. 1023–1028. doi: 10.1109/ACSSC.2017.8335504.
URL https://ieeexplore.ieee.org/document/8335504/

[17] Clifford, G. D.; Shoeb, A.; Mcsharry, P. E.: Model-based filtering, compression and classification of the ECG. *Int. J. Bioelectromagn.* vol. 7, no. 1. 2005: pp. 158–162.

[18] Cox, B. T.; Laufer, J. G.; Kostli, K. P.; et al.: Experimental validation of photoacoustic k-Space propagation models. In *Photons Plus Ultrasound: Imaging and Sensing*, vol. 5320, edited by A. A. Oraevsky; L. V. Wang. International Society for Optics and Photonics. SPIE. 2004. pp. 238–248. doi: 10.1117/12.531178.
URL https://doi.org/10.1117/12.531178

[19] Di, S.; Cappello, F.: Fast Error-Bounded Lossy HPC Data Compression with SZ. In *2016 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*. May 2016. ISSN 1530-2075. pp. 730–739. doi: 10.1109/IPDPS.2016.11.
URL http://ieeexplore.ieee.org/document/7516069/

[20] Dogra, V. S.; Zhang, M.; Bhatt, S.: High-Intensity Focused Ultrasound (HIFU) Therapy Applications. *Ultrasound Clinics*. vol. 4, no. 3. 2009: pp. 307–321. ISSN

1556-858X. doi: 10.1016/j.cult.2009.10.005.
URL https://linkinghub.elsevier.com/retrieve/pii/S1556858X09000668

[21] Elad, M.: *Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing*. Springer Publishing Company, Incorporated. first edition. 2010. ISBN 144197010X.

[22] Fletcher, S.-M. P.; Choi, M.; Ogrodnik, N.; et al.: A Porcine Model of Transvertebral Ultrasound and Microbubble-Mediated Blood-Spinal Cord Barrier Opening. *Theranostics*. vol. 10, no. 17. 2020: pp. 7758–7774. ISSN 1838-7640. doi: 10.7150/thno.46821.
URL https://www.thno.org/v10p7758.htm

[23] Fornberg, B.: High-Order Finite Differences and the Pseudospectral Method on Staggered Grids. *SIAM Journal on Numerical Analysis*. vol. 27, no. 4. 1990: pp. 904–918. ISSN 00361429.
URL http://www.jstor.org/stable/2157688

[24] Georgiou, P. S.; Jaros, J.; Payne, H.; et al.: Beam distortion due to gold fiducial markers during salvage high-intensity focused ultrasound in the prostate. *Medical Physics*. vol. 44, no. 2. 2017: pp. 679–693. ISSN 0094-2405. doi: 10.1002/mp.12044.
URL https://onlinelibrary.wiley.com/doi/10.1002/mp.12044

[25] Granai, L.; Vlachos, T.; Hamouz, M.; et al.: Model-Based Coding of 3D Head Sequences. In *2007 3DTV Conference*. IEEE. 2007. ISBN 978-1-4244-0721-7. pp. 1–4. doi: 10.1109/3DTV.2007.4379442.
URL http://ieeexplore.ieee.org/document/4379442/

[26] Grisey, A.; Yon, S.; Letort, V.; et al.: Simulation of high-intensity focused ultrasound lesions in presence of boiling. *Journal of Therapeutic Ultrasound*. vol. 4, no. 1. 2016. ISSN 2050-5736. doi: 10.1186/s40349-016-0056-9.
URL
http://jtultrasound.biomedcentral.com/articles/10.1186/s40349-016-0056-9

[27] Haigh, A. A.; McCreath, E. C.: Acceleration of GPU-Based Ultrasound Simulation via Data Compression. In *2014 IEEE International Parallel & Distributed Processing Symposium Workshops*. Washington, DC, USA: IEEE. 2014. ISBN 978-1-4799-4116-2. pp. 1248–1255. doi: 10.1109/IPDPSW.2014.140.
URL http://ieeexplore.ieee.org/document/6969522/

[28] Hasgall, P.; Di Gennaro, F.; Baumgartner, C.; et al.: IT'IS database for thermal and electromagnetic parameters of biological tissues. Version 3.0, September 1st, 2015. 2015. doi: 10.13099/VIP21000-03-0.
URL https://doi.org/10.13099/VIP21000-03-0

[29] Hung, N. Q. V.; Jeung, H.; Aberer, K.: An Evaluation of Model-Based Approaches to Sensor Data Compression. *IEEE Transactions on Knowledge and Data Engineering*. vol. 25, no. 11. 2013: pp. 2434–2447. ISSN 1041-4347. doi: 10.1109/TKDE.2012.237.
URL http://ieeexplore.ieee.org/document/6378372/

[30] IEEE Computer Society: IEEE Standard for Floating-Point Arithmetic. *IEEE Std 754-2019 (Revision of IEEE 754-2008)*. 2019: pp. 1–84. doi: 10.1109/IEEESTD.2019.8766229.

[31] IT4Innovations Documentation. 2022.
URL https://docs.it4i.cz/

[32] Jaros, J.; Nikl, V.; Treeby, B. E.: Large-Scale Ultrasound Simulations Using the Hybrid OpenMP/MPI Decomposition. In *Proceedings of the 3rd International Conference on Exascale Applications and Software*. EASC '15. GBR: University of Edinburgh. 2015. ISBN 978-0-9926615-1-9. page 115–119.

[33] Jaros, J.; Rendell, A. P.; Treeby, B. E.: Full-wave nonlinear ultrasound simulation on distributed clusters with applications in high-intensity focused ultrasound. *The International Journal of High Performance Computing Applications*. vol. 30, no. 2. 2016: pp. 137–155. ISSN 1094-3420. doi: 10.1177/1094342015581024.
URL http://journals.sagepub.com/doi/10.1177/1094342015581024

[34] Jaros, J.; Treeby, E. B.; Robertson, L. J.: *Convergence Testing of a k-space Pseudospectral Scheme for Transcranial Time-reversal Focusing*. chapter 6. VSB-Technical University of Ostrava - IT4I. 2017. ISBN 978-80-248-4037-6. pp. 195–197.

[35] Jaros, J.; Vaverka, F.; Treeby, B.: Spectral Domain Decomposition Using Local Fourier Basis: Application to Ultrasound Simulation on a Cluster of GPUs. *Supercomputing Frontiers and Innovations*. vol. 3. 11 2016: pp. 40–55. doi: 10.14529/jsfi160305.

[36] Javaherian, A.; Cox, B.: Ray-based inversion accounting for scattering for biomedical ultrasound tomography. *Inverse Problems*. vol. 37, no. 11. oct 2021: page 115003. doi: 10.1088/1361-6420/ac28ed.
URL https://doi.org/10.1088/1361-6420/ac28ed

[37] Juthakanjana, T.; Techavipoo, U.; Lamsrichan, P.; et al.: A Study of Data Compression for Ultrasound Rf Signals before Beamforming. In *The 3rd Biomedical Engineering International Conference 2010*. 2010. pp. 112–115.

[38] Kaufman, J. J.; Luo, G.; Siffert, R. S.: Ultrasound simulation in bone. *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*. vol. 55, no. 6. 2008: pp. 1205–1218. doi: 10.1109/TUFFC.2008.784.

[39] Kim, J. H.; Yeo, S.; Kim, J. W.; et al.: Real-Time Lossless Compression Algorithm for Ultrasound Data Using BL Universal Code. *Sensors*. vol. 18, no. 10. 2018. ISSN 1424-8220. doi: 10.3390/s18103314.
URL https://www.mdpi.com/1424-8220/18/10/3314

[40] Klepárník, P.: *Vizualizace šíření ultrazvuku v lidském těle*. Master's Thesis. Vysoké učení technické v Brně, Fakulta informačních technologií. 2013.

[41] Kleparnik, P.; Barina, D.; Zemcik, P.; et al.: Efficient Low-Resource Compression of HIFU Data. *Information*. vol. 9, no. 7. 2018: pp. 1–14. ISSN 2078-2489. doi: 10.3390/info9070155.
URL http://www.mdpi.com/2078-2489/9/7/155

[42] Kleparnik, P.; Zemcik, P.; Jaros, J.: Efficient lossy compression of ultrasound data. In *2017 IEEE International Symposium on Signal Processing and Information*

Technology (ISSPIT). IEEE. 2017. ISBN 978-1-5386-4662-5. pp. 232–237. doi: 10.1109/ISSPIT.2017.8388647.
URL https://ieeexplore.ieee.org/document/8388647/

[43] Kleparnik, P.; Zemcik, P.; Treeby, B. E.; et al.: On-the-Fly Calculation of Time-Averaged Acoustic Intensity in Time-Domain Ultrasound Simulations Using a k-Space Pseudospectral Method. *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*. vol. 69, no. 10. 2022: pp. 2917–2929. doi: 10.1109/TUFFC.2022.3199173.
URL https://ieeexplore.ieee.org/document/9857931

[44] Klima, O.; Barina, D.; Kleparnik, P.; et al.: Lossy Compression of 3D Statistical Shape and Intensity Models of Femoral Bones Using JPEG 2000. *IFAC-PapersOnLine*. vol. 49, no. 25. 2016: pp. 115–120. ISSN 24058963. doi: 10.1016/j.ifacol.2016.12.020.
URL https://linkinghub.elsevier.com/retrieve/pii/S2405896316326568

[45] Klima, O.; Chromy, A.; Kleparnik, P.; et al.: Model-based Radiostereometric Analysis Using Intensity-based 2D/3D Registration Pipeline: Feasibility Study. *Proceeding of Spring Conference on Computer Graphics*. vol. 2017, no. 5. 2017: pp. 31–33. ISSN 1335-5694.

[46] Klima, O.; Chromy, A.; Zemcik, P.; et al.: A Study on Performace of Levenberg-Marquardt and CMA-ES Optimization Methods for Atlas-based 2D/3D Reconstruction. *IFAC-PapersOnLine*. vol. 49, no. 25. 2016: pp. 121–126. ISSN 2405-8963. doi: https://doi.org/10.1016/j.ifacol.2016.12.021. 14th IFAC Conference on Programmable Devices and Embedded Systems PDES 2016.
URL https://www.sciencedirect.com/science/article/pii/S240589631632657X

[47] Klima, O.; Kleparnik, P.; Spanel, M.; et al.: GP-GPU Accelerated Intensity-based 2D/3D Registration Pipeline. In *Shape 2015 Symposium on Statistical Shape Models & Applications Proceedings*. Swiss Institute for Computer Aided Surgery: Swiss Institute for Computer Aided Surgery. september 2015. pp. 19–19.

[48] Klima, O.; Kleparnik, P.; Spanel, M.; et al.: Intensity-based femoral atlas 2D/3D registration using Levenberg-Marquardt optimisation. In *Medical Imaging 2016: Biomedical Applications in Molecular, Structural, and Functional Imaging*, vol. 9788, edited by B. Gimi; A. Krol. International Society for Optics and Photonics. SPIE. 2016. page 97880F. doi: 10.1117/12.2216529.
URL https://doi.org/10.1117/12.2216529

[49] Klima, O.; Kleparnik, P.; Spanel, M.; et al.: Towards an accurate 3D reconstruction of fractured long bones from plain 2D radiographs. In *International Journal of Computer Assisted Radiology and Surgery*. 11. Springer Verlag: Springer Verlag. june 2016. ISSN 1861-6410. pp. S180–S181.

[50] Kriemann, R.; Ltaief, H.; Luong, M. B.; et al.: High-Performance Spatial Data Compression for Scientific Applications. In *Euro-Par 2022: Parallel Processing*, edited by J. Cano; P. Trinder. Cham: Springer International Publishing. 2022. ISBN 978-3-031-12597-3. pp. 403–418.

[51] Kuklis, F.; Jaros, M.; Jaros, J.: Accelerated Design of HIFU Treatment Plans Using Island-Based Evolutionary Strategy. In *Applications of Evolutionary Computation*. Cham: Springer International Publishing. 2020. ISBN 978-3-030-43721-3. pp. 463–478. doi: 10.1007/978-3-030-43722-0_30.
URL http://link.springer.com/10.1007/978-3-030-43722-0_30

[52] Li, B.; Zhao, A.; Zhang, J.; et al.: Simulation study of compressed sensing photoacoustic tomography based on k-space pseudospectral method. In *Signal Processing, Sensor/Information Fusion, and Target Recognition XXX*, vol. 11756, edited by I. Kadar; E. P. Blasch; L. L. Grewe. International Society for Optics and Photonics. SPIE. 2021. page 117561H. doi: 10.1117/12.2589660.
URL https://doi.org/10.1117/12.2589660

[53] Lindstrom, P.: Fixed-Rate Compressed Floating-Point Arrays. *IEEE Transactions on Visualization and Computer Graphics*. vol. 20, no. 12. 2014-12-31: pp. 2674–2683. ISSN 1077-2626. doi: 10.1109/TVCG.2014.2346458.
URL https://ieeexplore.ieee.org/document/6876024/

[54] Lindstrom, P.; Isenburg, M.: Fast and Efficient Compression of Floating-Point Data. *IEEE Transactions on Visualization and Computer Graphics*. vol. 12, no. 5. 2006: pp. 1245–1250. ISSN 1077-2626. doi: 10.1109/TVCG.2006.143.
URL http://ieeexplore.ieee.org/document/4015488/

[55] Ling, Y. T.; Martin, E.; Treeby, B. E.: A discrete source model for simulating bowl-shaped focused ultrasound transducers on regular grids: Design and experimental validation. In *2015 IEEE International Ultrasonics Symposium (IUS)*. 2015. pp. 1–4. doi: 10.1109/ULTSYM.2015.0281.

[56] Liu, B.; Mohandes, M.; Nuha, H.; et al.: A Multitone Model-Based Seismic Data Compression. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*. vol. 52, no. 2. 2022: pp. 1030–1040. doi: 10.1109/TSMC.2021.3077490.

[57] Lucka, F.; Pérez-Liva, M.; Treeby, B. E.; et al.: High resolution 3D ultrasonic breast imaging by time-domain full waveform inversion. *Inverse Problems*. vol. 38, no. 2. dec 2021: page 025008. doi: 10.1088/1361-6420/ac3b64.
URL https://doi.org/10.1088/1361-6420/ac3b64

[58] Martin, E.; Jaros, J.; Treeby, B. E.: Experimental Validation of k-Wave. *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*. vol. 67, no. 1. 2020: pp. 81–91. ISSN 0885-3010. doi: 10.1109/TUFFC.2019.2941795.
URL https://ieeexplore.ieee.org/document/8839830/

[59] Martin, E.; Treeby, B. E.: Experimental validation of computational models for large-scale nonlinear ultrasound simulations in heterogeneous, absorbing fluid media. *AIP Conference Proceedings*. vol. 1685, no. 1. 2015: page 070007. doi: 10.1063/1.4934444.
URL https://aip.scitation.org/doi/abs/10.1063/1.4934444

[60] Massey, J. W.; Yilmaz, A. E.: AustinMan and AustinWoman. In *2016 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. IEEE. 2016. ISBN 978-1-4577-0220-4. pp. 3346–3349. doi:

10.1109/EMBC.2016.7591444.
URL http://ieeexplore.ieee.org/document/7591444/

[61] Masui, K.; Amiri, M.; Connor, L.; et al.: A compression scheme for radio data in high performance computing. *Astronomy and Computing*. vol. 12. 2015: pp. 181–190. ISSN 2213-1337. doi: https://doi.org/10.1016/j.ascom.2015.07.002.
URL https://www.sciencedirect.com/science/article/pii/S2213133715000694

[62] Mensmann, J.; Ropinski, T.; Hinrichs, K.: A GPU-Supported Lossless Compression Scheme for Rendering Time-Varying Volume Data. In *IEEE/ EG Symposium on Volume Graphics*, edited by R. Westermann; G. Kindlmann. The Eurographics Association. 12 2010. ISBN 978-3-905674-23-1. ISSN 1727-8376. pp. 109–116. doi: 10.2312/VG/VG10/109-116.

[63] Pennes, H. H.: Analysis of Tissue and Arterial Blood Temperatures in the Resting Human Forearm. *Journal of Applied Physiology*. vol. 1, no. 2. 1948: pp. 93–122. ISSN 8750-7587. doi: 10.1152/jappl.1948.1.2.93.
URL http://www.physiology.org/doi/10.1152/jappl.1948.1.2.93

[64] Pilikos, G.; Horchens, L.; Batenburg, K. J.; et al.: Deep data compression for approximate ultrasonic image formation. In *2020 IEEE International Ultrasonics Symposium (IUS)*. 2020. pp. 1–4. doi: 10.1109/IUS46767.2020.9251753.

[65] Robertson, J. L. B.; Cox, B. T.; Jaros, J.; et al.: Accurate simulation of transcranial ultrasound propagation for ultrasonic neuromodulation and stimulation. *The Journal of the Acoustical Society of America*. vol. 141, no. 3. 2017: pp. 1726–1738. ISSN 0001-4966. doi: 10.1121/1.4976339.
URL http://aip.scitation.org/doi/10.1121/1.4976339

[66] Rong, L.: *Data Compression in Ultrasound Computed Tomography*. PhD. Thesis. Institut für Biomedizinische Technik (IBT), Fakultät für Elektrotechnik und Informationstechnik (ETIT). 2011. doi: 10.5445/IR/1000023057.

[67] Sayood, K.: *Introduction to data compression*. Waltham, MA: Morgan Kaufmann. fourth edition. c2012. ISBN 978-0-12-415796-5.

[68] Son, S.; Chen, Z.; Hendrix, W.; et al.: Data Compression for the Exascale Computing Era - Survey. *Supercomput. Front. Innov.: Int. J.*. vol. 1, no. 2. July 2014: pp. 76–88. ISSN 2409-6008. doi: 10.14529/jsfi140205.
URL https://superfri.org/index.php/superfri/article/view/13

[69] Song, X.; Liu, X.; Zhou, X.; et al.: Bessel-beam photoacoustic microscopy based on k-space pseudospectral method: simulation study. In *SPIE Future Sensing Technologies*, vol. 11525, edited by M. Kimata; J. A. Shaw; C. R. Valenta. International Society for Optics and Photonics. SPIE. 2020. page 115252M. doi: 10.1117/12.2584874.
URL https://doi.org/10.1117/12.2584874

[70] Song, X.; Wei, J.; Song, L.: Simulation platform of large volumetric photoacoustic microscopy based on k-space pseudospectral method. In *Medical Imaging 2021: Biomedical Applications in Molecular, Structural, and Functional Imaging*, vol. 11600, edited by B. S. Gimi; A. Krol. International Society for Optics and Photonics.

SPIE. 2021. page 1160019. doi: 10.1117/12.2580205.
URL https://doi.org/10.1117/12.2580205

[71] Sung, J.: Transform Coding Based on Source Filter Model in the MDCT Domain. *ETRI Journal*. vol. 35, no. 3. 2013-06-01: pp. 542–545. ISSN 1225-6463. doi: 10.4218/etrij.13.0212.0368.
URL http://doi.wiley.com/10.4218/etrij.13.0212.0368

[72] Suomi, V.; Jaros, J.; Treeby, B.; et al.: Nonlinear 3-D simulation of high-intensity focused ultrasound therapy in the Kidney. In *2016 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. IEEE. 2016. ISBN 978-1-4577-0220-4. pp. 5648–5651. doi: 10.1109/EMBC.2016.7592008.
URL http://ieeexplore.ieee.org/document/7592008/

[73] Suomi, V.; Jaros, J.; Treeby, B.; et al.: Full Modeling of High-Intensity Focused Ultrasound and Thermal Heating in the Kidney Using Realistic Patient Models. *IEEE Transactions on Biomedical Engineering*. vol. 65, no. 11. 2018: pp. 2660–2670. ISSN 0018-9294. doi: 10.1109/TBME.2018.2870064.
URL https://ieeexplore.ieee.org/document/8464260/

[74] Suomi, V.; Treeby, B.; Jaros, J.; et al.: Transurethral ultrasound therapy of the prostate in the presence of calcifications: A simulation study. *Medical Physics*. vol. 45, no. 11. 2018: pp. 4793–4805. doi: https://doi.org/10.1002/mp.13183.
URL https://aapm.onlinelibrary.wiley.com/doi/abs/10.1002/mp.13183

[75] Suprijanto; Kurniadi, D.; Sinta: Modelling and Simulation 3D Ultrasound Wave Propagation using k-space Pseudospectral Method in the Railway Track Geometry. In *2019 6th International Conference on Instrumentation, Control, and Automation (ICA)*. 2019. pp. 153–157. doi: 10.1109/ICA.2019.8916739.

[76] Taylor, J.; Paull, C.; Hayes-Gill, B.; et al.: Data compression of fetal Doppler ultrasound audio signals using zero-crossings analysis. *Medical Engineering & Physics*. vol. 19, no. 6. 1997: pp. 572–580. ISSN 13504533. doi: 10.1016/S1350-4533(97)00005-2.
URL https://linkinghub.elsevier.com/retrieve/pii/S1350453397000052

[77] Treeby, B.; Vaverka, F.; Jaros, J.: Performance and accuracy analysis of nonlinear k-Wave simulations using local domain decomposition with an 8-GPU server. *Proceedings of Meetings on Acoustics*. vol. 34, no. 1. 2018: page 022002. doi: 10.1121/2.0000883.
URL https://asa.scitation.org/doi/abs/10.1121/2.0000883

[78] Treeby, B. E.; Cox, B. T.: k-Wave: MATLAB toolbox for the simulation and reconstruction of photoacoustic wave fields. *Journal of Biomedical Optics*. vol. 15, no. 2. 2010: pp. 021314–021314–12. ISSN 1083-3668. doi: 10.1117/1.3360308.
URL http://biomedicaloptics.spiedigitallibrary.org/article.aspx?doi=10.1117/1.3360308

[79] Treeby, B. E.; Jaros, J.; Rendell, A. P.; et al.: Modeling nonlinear ultrasound propagation in heterogeneous media with power law absorption using a k-space pseudospectral method. *The Journal of the Acoustical Society of America*. vol. 131,

no. 6. 2012: pp. 4324–4336. ISSN 0001-4966. doi: 10.1121/1.4712021.
URL http://asa.scitation.org/doi/10.1121/1.4712021

[80] Treeby, B. E.; Saratoon, T.: The contribution of shear wave absorption to ultrasound heating in bones. In *2015 IEEE International Ultrasonics Symposium (IUS)*. IEEE. 2015. ISBN 978-1-4799-8182-3. pp. 1–4. doi: 10.1109/ULTSYM.2015.0296.
URL http://ieeexplore.ieee.org/document/7329586/

[81] Treeby, B. E.; Wise, E. S.; Kuklis, F.; et al.: Nonlinear ultrasound simulation in an axisymmetric coordinate system using a k-space pseudospectral method. *The Journal of the Acoustical Society of America*. vol. 148, no. 4. 2020: pp. 2288–2300. doi: 10.1121/10.0002177.
URL https://doi.org/10.1121/10.0002177

[82] Varray, F.: *Simulation in nonlinear ultrasound : application to nonlinear parameter imaging in echo mode configuration*. PhD. Thesis. Université Claude Bernard - Lyon I ; Università degli studi (Florence, Italie). 10 2011. thèse de doctorat dirigée par Cachard, ChristianBasset, Olivier et Tortoli, Piero Acoustique Lyon 1 2011.

[83] Vaverka, F.; Treeby, B. E.; Jaros, J.: Performance Evaluation of Pseudospectral Ultrasound Simulations on a Cluster of Xeon Phi Accelerators. In *High Performance Computing in Science and Engineering*, edited by T. Kozubek; P. Arbenz; J. Jaroš; L. Říha; J. Šístek; P. Tichý. Cham: Springer International Publishing. 2021. ISBN 978-3-030-67077-1. pp. 99–115.

[84] Wang, K.; Teoh, E.; Jaros, J.; et al.: Modelling nonlinear ultrasound propagation in absorbing media using the k-Wave toolbox. In *2012 IEEE International Ultrasonics Symposium*. IEEE. 2012. ISBN 978-1-4673-4562-0. pp. 523–526. doi: 10.1109/ULTSYM.2012.0130.
URL http://ieeexplore.ieee.org/document/6562209/

[85] Wegener, A.: Block floating point compression with exponent difference and mantissa coding. 10 2014.
URL https://www.freepatentsonline.com/8874794.html

[86] Williams, E. G.: Chapter 2 - Plane Waves. In *Fourier Acoustics: Sound Radiation and Nearfield Acoustical Holography*. London: Academic Press. 1999. ISBN 978-0-12-753960-7. pp. 15–87. doi: 10.1016/B978-012753960-7/50002-4.

[87] Zagar, M.: *4D Medical Data Compression Architecture: The Novel Approach*. Koln, DEU: LAP Lambert Academic Publishing. 2011. ISBN 3846543616.

[88] Zhao, K.; Sakamoto, N.; Koyamada, K.: Compression for Large-Scale Time-Varying Volume Data Using Spatio-temporal Features. In *AsiaSim 2013*. Berlin, Heidelberg: Springer Berlin Heidelberg. 2013. ISBN 978-3-642-45036-5. pp. 136–148. doi: 10.1007/978-3-642-45037-2_13.
URL http://link.springer.com/10.1007/978-3-642-45037-2_13

[89] Zhou, Y.-F.: High intensity focused ultrasound in clinical tumor ablation. *World Journal of Clinical Oncology*. vol. 2, no. 1. 2011: pp. 8–27. ISSN 2218-4333. doi: 10.5306/wjco.v2.i1.8.
URL http://www.wjgnet.com/2218-4333/full/v2/i1/8.htm