BRNO UNIVERSITY OF TECHNOLOGY
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF INFORMATION TECHNOLOGY
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

DEPARTMENT OF COMPUTER SYSTEMS
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

LARGE-SCALE ULTRASOUND SIMULATIONS USING ACCELERATED CLUSTERS

ROZSÁHLÉ SIMULACE ULTRAZVUKU ZA POUŽITÍ AKCELEROVANÝCH CLUSTERŮ

PHD THESIS
DISERTAČNÍ PRÁCE

AUTHOR **Ing. FILIP VAVERKA**
AUTOR PRÁCE

SUPERVISOR **Doc. Ing. JIŘÍ JAROŠ, Ph.D.**
ŠKOLITEL

BRNO 2023

Abstract

Efficient utilization of accelerated HPC clusters is particularly sensitive to communication efficiency of deployed algorithms. In this thesis, we reexamine pseudo-spectral solvers for wave-like problems in medical ultrasonics to allow their deployment on these machines. The domain decomposition is shown to be a preferable approach to improving data locality of these solvers as a range of suitable alternative discretizations exhibited considerably worse numerical properties. The local Fourier basis domain decomposition is then used to construct a novel solver based on the state of the art model for ultrasound in medicine – k-Wave. We show that this approach is up to 7.5× faster and achieves almost perfect weak-scaling up to 512 GPU accelerated nodes, while being able to take full advantage of advanced GPU interconnects such as NVLink in NVIDIA DGX-2 multi-GPU nodes. The method offers flexible accuracy–efficiency trade off, which allows to nearly match accuracy of the global k-Space method or maximize performance at sufficient accuracy by subdomain overlap scaling.

Abstrakt

Efektivní využití akcelerovaných HPC clusterů je obzvlášť závislé na efektivitě komunikace použitých algoritmů. Tato práce se tedy věnuje přezkoumání pseudo-spektrálních algorimů používaných pro řešení vlnových problémů převážně v oblasti medicínského ultrazvuku s cílem umožnit jejich běh na akcelerovaných strojích. Je ukázáno, že doménová dekompozice je preferovaný způsob dosažení daného cíle, jelikož řada alternativních přístupů vykazuje výrazně horší numerické vlastnosti. Na základě tohoto přístupu a k-Wave modelu ultrazvuku, široce používaného v medicíně, je navržen nový simulační algoritmus. Následnými experimenty je ukázáno, že tento přístup dosahuje až 7.5× zrychlení a dosahuje téměř perfektního slabého škálování až do 512 GPU akcelerovaných uzlů. Zároveň toto řešení umožňuje plné využití výpočetních uzlů s několika GPU akcelerátory a pokročilým propojením jako je NVIDIA DGX-2 s NVLink. Tato metoda také nabízí možnost flexibilní volby mezi přesností a efektivitou. Volbou hloubky překryvu subdomén lze dosáhnout jak přesnosti srovnatelné s původní k-Space metodou, tak i maximalizovat výkon při zachování dostatečné přesnosti.

Keywords

pseudo-spectral methods, supercomputing, domain decomposition methods, local Fourier basis, medical ultrasound simulation, non-linear ultrasound, HIFU, MPI+X, CUDA, GPU, accelerated computing

Klíčová slova

pseudo-spektrální metody, superpočítání, metody doménové dekompozice, lokální Fourierova báze, simulace ultrazvuku v lékařství, nelineární ultrazvuk, HIFU, MPI+X, CUDA, GPU, akcelerované počítání

Reference

VAVERKA, Filip. *Large-scale Ultrasound Simulations using Accelerated Clusters.* Brno, 2023. PhD thesis. Brno University of Technology, Faculty of Information Technology. Supervisor Jaroš Jiří.

Large-scale Ultrasound Simulations using Accelerated Clusters

Declaration

Hereby I declare that this PhD thesis was prepared as an original author's work under the supervision of Doc. Ing. Jiří Jaroš, Ph.D. All the relevant information sources, which were used during preparation of this thesis, are properly cited and included in the list of references.

.....
Filip Vaverka
January 26, 2023

Acknowledgements

This project has received funding from the European Union's Horizon 2020 research and innovation programme H2020 ICT 2016-2017 under grant agreement No 732411 and is an initiative of the Photonics Public Private Partnership. This work was supported by The Ministry of Education, Youth and Sports from the National Programme of Sustainability (NPU II) project "IT4Innovations excellence in science - LQ1602" and by the IT4Innovations infrastructure which is supported from the Large Infrastructures for Research, Experimental Development and Innovations project "IT4Innovations National Supercomputing Center - LM2015070". This work was supported by Czech Science Foundation project 19-10137S and GA16-17538S. This work was supported by the Ministry of Education, Youth and Sports of the Czech Republic through the e-INFRA CZ (ID:90140). This work was supported by Brno University of Technology under project number FIT-S-20-6309 and FIT-S-17-3994.

Contents

Chapter 1

Introduction

During last 15 years, the supercomputing landscape has gone through seismic shift with the introduction of accelerators, which now comprise more than 50 % of total available computational capability[1]. These accelerated clusters achieve significant efficiency and performance gains at the cost of increased system architecture complexity. Typical accelerated node may contain a few compute accelerators (such as GPUs) each with dedicated memory and couple of smart network interfaces. Careful data movement planning and scheduling is crucial for an efficient utilization of such a node. Inter-node communication in these clusters is relatively more expensive than before due to increases in performance of each node as interconnect bandwidth increases often lag behind.

Such a significant shift in the supercomputer architecture necessarily leads to redesign of core algorithms or even whole applications to take full advantage of these machines. The degree to which the application may need to be redesigned may range from a small developer effort all the way to taking a completely different approach to the solution of the original problem. In this thesis, we present an example of such a complete bottom up analysis and the redesign of the ultrasound wave propagation simulation package.

1.1 Ultrasound in Medicine

The first medical applications of ultrasonics were suggested in 1920s during the development of ultrasound transmission based nondestructive testing methods used to detect hidden flaws in metals. However, the first medical applications were focused on high intensity ultrasound and its heating effects, which were discovered earlier. The first diagnostics efforts in 1930s focused on brain imaging using transmission intensity measurements and pulse-echo methods were used in soft-tissue imaging with first two-dimensional tomograms published in 1952 [50]. Since then, it has proliferated to many fields of medicine, where it is used primarily for diagnostics of various soft tissues. The sonographic methods developed to the point where blood flow can be monitored using Doppler effect and 3D images can be reconstructed in real-time using 2D phased transducer arrays.

However, other classes of diagnostic and treatment methods using ultrasound are also, perhaps even more so, actively developed. One of relatively recent developments in ultrasound imaging is Photoacoustic Imaging (PAI) [15]. This technique takes an advantage of ultrasound waves generated by absorption of nanosecond laser pulse in the tissue. The initial pressure distribution (and therefore laser absorption map) can be reconstructed by collect-

[1] According to Top500 (https://www.top500.org) statistics.

ing generated ultrasound waves and solving an inverse ultrasound propagation problem. While the solution of this inverse problem with incomplete data is very computationally demanding, this technique offers advantages such as blood oxygenation measurement in the tissue [70].

Significantly higher ultrasound intensities are utilized in the area of treatment (ultrasound therapy). Noninvasive procedures such as opening brain blood barrier [8] for drug delivery or brain stimulation [98, 129] utilize lower intensities in this range. While the highest intensities are used in High Intensity Focused Ultrasound (HIFU) [65, 126] to take advantage of heating and cavitation effects for cancer tissue ablation [75].

The downside of these advanced methods is the requirement for a significant amount of processing. Photoacoustic Imaging generally requires to solve an inverse ultrasound propagation problem with incomplete data, hundreds of ultrasound sources and large domains [113]. Similarly, HIFU procedure requires careful planning, so that only targeted tissue is hit, which once again consists of an optimization problem. Each evaluation of a plan candidate involves ultrasound propagation simulation over a large domain (see table 1.1) which has to model nonlinear behavior of high intensity ultrasound wave.

Table 1.1: Typical modeling scenarios required for diagnostics or treatment using ultrasound. Shortcut "DU" stands for "Diagnostic Ultrasound", "MC" for "Minimal Cavitation", "IC" for "Intense Cavitation" and "ST" for "Source Type". "TB" and "CW" stands for "Tone Burst" or "Continuous-Wave". Numbers in parentheses in the $f_{\max}$ column denote maximal number of harmonic frequencies generated from the source frequency.

	Modeling Scenario	f_{src} [MHz]	S.T.	$f_{\max}$ [MHz]	Domain Size [mm]	Domain Size [λ]
DU:	Curvilin. Transducer	3	TB	18 (5)	$150 \times 80 \times 25$	$1800 \times 960 \times 300$
DU:	Lin. Transducer	10	TB	60 (5)	$50 \times 80 \times 30$	$2000 \times 3200 \times 1200$
MC:	Prostate HIFU	4	CW	64 (15)	$80 \times 60 \times 20$	$3413 \times 2560 \times 853$
MC:	MR-Guided HIFU	1.5	CW	15 (10)	$250 \times 250 \times 150$	$2500 \times 2500 \times 1500$
IC:	Histotripsy	1	CW	50 (50)	$250 \times 250 \times 150$	$8333 \times 8333 \times 5000$

1.2 Ultrasound Modeling for Medical Applications

The most general approach to modeling ultrasound propagation through tissue is a direct application of continuum mechanics. However, accuracy required in most medical applications can be achieved using simplified models, which capture only a subset of real physical phenomena. Generally, there are three important factors that have to be considered in accurate models of ultrasound wave propagation in soft tissue. The acoustic wave amplitude is typically large enough so that nonlinear wave propagation has to be considered. The nonlinear models are critical for HIFU-type applications, however, some imaging applications such as tissue harmonic imaging [13] also take advantage of it. The material properties (sound speed and density) of soft tissue are weakly heterogeneous (on the order of 5 % [77]) between different tissue types. The soft tissue exhibits frequency dependent absorption, which follows frequency power law. The absorption process is an important part of nonlinear model as it dampens frequencies generated by the nonlinear propagation.

The elastic wave propagation models are generally considered unnecessary for modeling wave propagation in soft tissue as shear waves can be neglected without a significant loss of accuracy [103]. The elastic models are therefore more prevalent in other fields such as

seismology or geophysics [39, 95, 123], while fluid models are sufficient for most medical applications. One of notable exceptions are applications involving bone tissue, especially in cases with non-normal incidence angle of the ultrasound beam [121].

According to recent review [51] the Westervelt and Khokhlov-Zabolotskaya-Kuznetsov (KZK) equations are the two most widely used models for medical ultrasound. Both of these models can be derived from Kuznetsov equation, which itself, can be viewed as an approximation of isentropic Navier-Stokes (for viscous media) and Euler (for the inviscid media) systems [31]. The Kuznetsov equation is not widely used in medical ultrasound modeling as it doesn't offer significant advantages over Westervelt in those applications and is considerably more difficult to directly numerically solve. The Westervelt equation simplifies Kuznetsov equation by dropping the term describing Lagrangian density of acoustical energy, which is zero in case of progressive plane waves. However, even in the case of non-plane waves, this approximation introduces error only in local (non-cumulative) nonlinear effects [26]. The KZK equation, which can be viewed as a parabolic approximation of Westervelt equation, is in theory only accurate for waves traveling within 16° of the nominal axis.

The models based on Kuznetsov equation can also be viewed as extensions to the second-order wave equation. Models derived by Treeby et al. [115] and Tabei et al. [110] rather use a system of first-order wave equations. This approach is advantageous when mass and acoustic particle velocity source terms are to be introduced and also allows to easily incorporate perfectly matched layer (PML) [16, 17, 127] to model absorbing boundary. Further restricted linear models (such as wave equation) can be used in development of phased array transducers or in ultrasound imaging applications.

The linear wave equation based models are easiest to solve. The Green's function method [114] can be used to rapidly calculate the wave field from a phased array transducer driven by a single frequency continuous wave in homogeneous media with power law absorption. The fast near-field method (FNM) [27, 79, 80] and angular spectrum approach (ASA) [29, 107, 118, 128] can be also used in similar settings up to first-order reflections in heterogeneous media [117]. The three most commonly used methods to solve the heterogeneous media wave equation in the medical ultrasound community are finite-difference time-domain method (FDTD) [52], pseudo-spectral time-domain method (PSTD) [74] and k-space time-domain metod (KSTD) [110]. It appears that a typical FDTD solver used in medical ultrasound applications is fourth order in space and second order in time and requires 8 to 10 grid points per minimal wavelength to achieve the accuracy targets. Even higher order solvers are often used in areas such as geophysics [12]. In contrast, both spectral approaches can reduce necessary grid resolution down to 3 grid points per wavelength for smooth fields with PML boundary [98]. The KSTD method augments PSTD method by using a semi-analytical time-stepping scheme [82] and is thus considered even more efficient.

Given simulation requirements of typical medical applications listed in table 1.1, the spectral methods are, due to significant reduction in spatial resolution, very popular and often even the only choice. Table 1.2 highlights the fact that even at the theoretical limit of the grid resolution achievable by spectral methods, many simulations require machines with distributed memory and would be impractical with FDTD methods.

Table 1.2: Possible domain sizes necessary to simulate sound wave propagation at frequencies encountered in high-intensity focused ultrasound (HIFU) applications. Assuming a uniform Cartesian grid at the Nyquist limit of two points per minimum wavelength (PPMW) and sound speed of 1500 m/s.

Domain Size [mm]	Maximum Freq [MHz]	Domain Size [λ]	Grid Size (2 PPMW)	Memory per Matrix [GB]
$50 \times 50 \times 50$	5	333^3	667^3	1.1
	10	667^3	1333^3	8.8
	20	1333^3	2667^3	71
	50	3333^3	6667^3	1100
$100 \times 100 \times 100$	5	667^3	1333^3	8.8
	10	1333^3	2667^3	71
	20	2667^3	5333^3	570
	50	6667^3	13333^3	8800
$200 \times 200 \times 200$	5	1333^3	2667^3	71
	10	2667^3	5333^3	570
	20	5333^3	10667^3	4500
	50	13333^3	26667^3	71000

1.3 Scope of this thesis

This thesis focuses on efficient numerical modeling and computer simulation of wave-like physical phenomena in light of models used in medical ultrasonics and current and near-future developments in high performance and client computing platforms.

First, we will overview typical architectures of modern accelerated and non-accelerated supercomputers and identify architectural features critical for designing efficient numerical methods for these platforms. With that, we will turn our attention to numerical methods typically used for discretization of PDE based models describing wave-like phenomena. In particular, we are interested in systems of conservation laws which are often preferred over more complex equations due to simplicity and flexibility. In following chapters, we will analyze possible approaches to adapting existing (and increasingly popular) Fourier collocation numerical schemes for modern accelerated cluster environments.

To illustrate an application of our approach we will adapt fully developed and widely used Fourier collocation scheme modeling propagation of non-linear ultrasound wave in heterogeneous and absorbing media designed specifically for biological tissues in medical applications called k-Wave [113]. Finally, we will provide detailed analysis of achieved results from the perspective of both computational efficiency and numerical properties.

Chapter 2

Modern High Performance Computing Platforms

This chapter offers a short overview of high performance computing (HPC) platforms, which are expected to be available for personalized medicine applications. This overview will serve as grounds for evaluation of computational properties of numerical methods considered in following chapters.

Tables 1.1 and 1.2 illustrate the primary characteristic of the problem at hand, quick growth of the computational requirements dependent on combination of physical domain size and supported frequencies. This is compounded by heterogeneity of the medium and non-linearity of the problem, both of which severely limit options to ease this dependency (such as boundary methods). These memory requirements are the main determining factor for the simulation to be performed on workstation, hospital's on premises data center or has to be offloaded to large scale HPC or cloud platform.

In any case, it is reasonable to expect the basic building block (a node) of an available platform to be based on architectures typical (or closely related) to commodity hardware. A typical node (or workstation) then might feature a multi-core CPU in single or dual socket configuration with up to eight DDR4 memory channels per socket and up to 128 PCI Express 5.0 lanes for peripherals. Each CPU can be expected to consist of up to 64 super-scalar cores with simultaneous multi-threading (SMT), wide (eg., 512-bit) SIMD vector units and up to 256 MB of last level cache (LLC). For example, an Intel Xeon Platinum 8380 with 40 cores can sustain upwards 3 TFLOP/s (R_{peak}) in single precision AVX-512 workload with 170 GB/s memory bandwidth (17.6 FLOP/B) at about 250 W TDP (or 12 GFLOP/s and 680 MB/s per watt).

However, a GPU accelerator such as AMD MI200 can achieve 48 TFLOP/s in double precision with 3.2 TB/s memory bandwidth (15 FLOP/B) at nearly 400 W (nearly 120 GFLOP/s and 8 GB/s per watt). The GPU accelerator offers more than 10-fold improvement in energy efficiency over the Intel Xeon CPU in this case. For this reason, majority of newly deployed machines are equipped with GPU accelerators used to offload floating point math heavy or memory bandwidth demanding tasks. Alternatively, custom CPUs with emphasis on vector processing and memory bandwidth, such as A64FX in Japanese Fugaku cluster[1], are deployed with great results.

[1] Riken Center for Computational Science, Japan

2.1 Accelerator Architectures

The primary focus of general purpose accelerators is to extend compute capabilities of traditional CPU architectures in terms of throughput and efficiency. These goals are typically achieved by embracing parallelism to a much higher degree and exposing it to higher levels of abstraction. For example, while a super-scalar CPU architecture attempts to exploit parallelism in a sequential instruction stream to fill its execution units, the accelerator may rather expose the same units in terms of Very Long Instruction Word (VLIW) instruction set – significantly simplifying necessary circuitry. Similarly, memory consistency models in accelerators are often weaker than those of traditional CPUs – reducing potentially unnecessary on-chip communication. The aim of elimination of all this complexity is to free up power and area budget of the chip, which can be utilized to integrate more execution units. In most cases, the massive increase in execution capabilities has to be balanced by increasing off-chip memory bandwidth, which often necessitates a tighter memory integration (eg., extremely wide memory bus of High Bandwidth Memory). The MI200 GPU accelerator, introduced above is an example of a mismatch between compute performance and memory bandwidth. While the accelerator manages to achieve almost tenfold increase of both compute and memory bandwidth, the crucial ratio of these two metrics is slightly worse compared to the top of the line CPU.

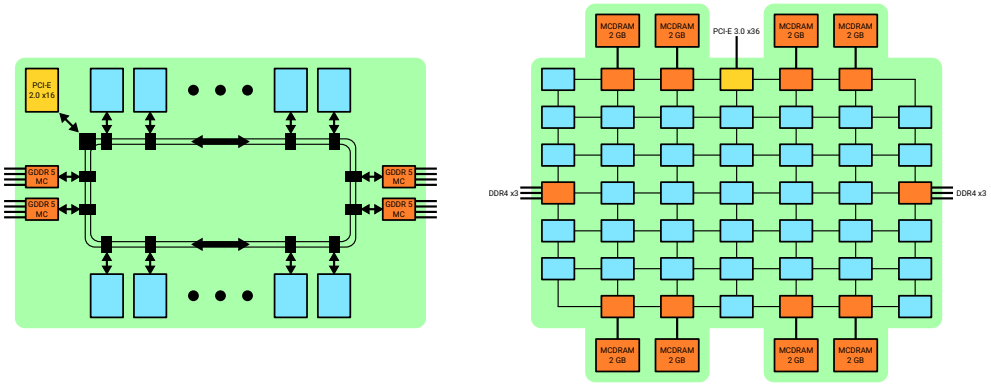
The novel (marketing focused) compute taxonomy introduced by Intel recognizes four architecture categories: scalar, vector, matrix and spatial. The scalar architecture typically refers to traditional CPUs with focus on sequential tasks, while GPUs and Vector Processing Units (VPUs) fall into vector architectures optimized for parallel vectorizable workloads. The matrix architectures focus on matrix multiplication heavy workloads such as deep learning (eg., systolic array based Google TPU architecture). Finally, the spatial architecture describes data flow oriented algorithms often implemented by means of FPGAs. The majority of accelerators would fall into one of the last three categories (vector, matrix and spatial).

2.1.1 Many-Core Accelerators

The idea behind many-core accelerator architectures is to expose more of the hardware parallelism in a manner similar to traditional multi-core processors. This is achieved primarily by greatly simplifying architecture of each core (eg., out-of-order execution, branch prediction, etc. are removed), which allows to integrate many more of such small cores on the chip. However, compute capabilities of these small cores are typically enhanced by addition of wide vector units, which allow to further increase the amount of parallelism exposed to the user. Once again, the off-chip memory bandwidth has to be increased accordingly to maintain reasonable compute-to-bandwidth ratio.

The advantage of such an accelerator design is that it allows to extend the programming model well known from multi-core CPUs. However, doing so necessitates to also maintain stronger memory consistency models used in multi-core architectures, which can induce unnecessary complexity in the design.

The prime example of many-core accelerators is line of Intel Xeon Phi accelerators built on the Intel Many Integrated Cores (Intel MIC) architectural family. The two commercially available members of this family Knights Corner (KNC) and Knights Landing (KNL) follow the design philosophy of integrating many simpler x86-64 cores with 512-bit wide vector extensions (later adopted as Intel AVX-512 extensions into mainstream CPUs).



(a) Knight Corner (KNC)

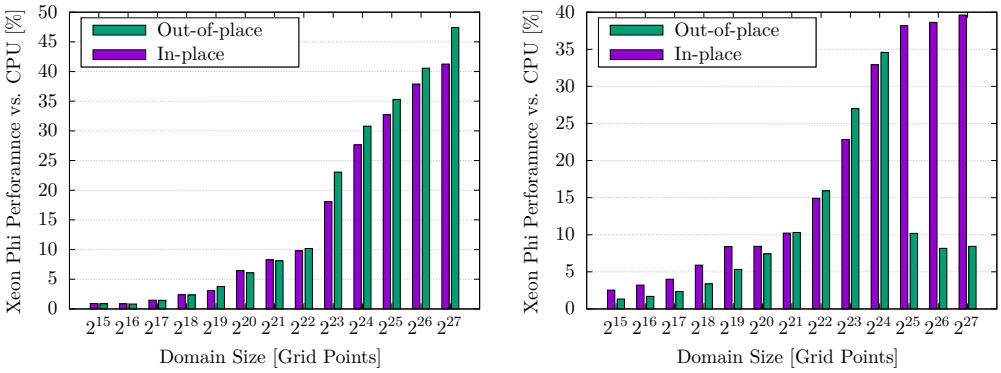
(b) Knight Landing (KNL)

Figure 2.1: Block architecture of the first (left) and the second (right) generation of Intel Many-Core (MIC) accelerators. The two-way ring-bus interconnect was replaced with a 2D mesh and each processing tile (light blue) was expanded to include two cores. The orange blocks denote memory controllers and on-package memory, while system interfaces are in yellow.

The KNC architecture (eg., Intel Xeon Phi 7120P) shown in fig. 2.1a consists of 61 P45C in-order cores extended by 4-wide simultaneous multithreading (SMT) and a 512-bit wide vector processing unit (VPU). The KNC memory subsystem consists of 30.5 MB of L2 cache distributed among cores which are interconnected via a ring bus (2×64-byte wide). The main memory is 16 GB of GDDR5 divided between four memory controllers with four channels each. The R_{peak} of the accelerator is 2 TFLOP/s with 352 GB/s of memory bandwidth.

The combination of 60 cores on a bidirectional ring interconnect with peak theoretical bi-sectional bandwidth of 170 GB/s and MOESI [67] cache coherency protocol is the primary weakness of the KNC architecture. The cache coherency traffic can easily saturate the interconnect and high remote cache access latency (only moderately lower than main memory access latency [35]) poses significant performance problems. We have observed these issues in the case of multi-dimensional FFTs, which serve as a basis of many spectral and pseudo-spectral methods. The algorithm exhibited unexpected performance drop at certain transform sizes (see fig. 2.2).

The KNL architecture significantly improves upon KNC in multiple aspects ranging from individual cores to memory hierarchy. The KNL cores are two-wide out-of-order derivative of Silvermont cores (designed for Intel Atom product line) enhanced with two AVX-512-capable VPUs each. The pair of cores with shared 1 MB of L2 cache makes up a tile, which is connected to the 2D-mesh interconnect (see fig. 2.1b). The main memory is split in two pools, fast Multi-Channel DRAM (MCDRAM) and slower DDR4 connected to two three-channel controllers. Intel Xeon Phi 7210-E based on KNL achieves R_{peak} of 5 TFLOP/s in single precision with 460 GB/s and 80 GB/s of memory bandwidth to fast and slow pool respectively. The main memory of the accelerator is 16 GB of the fast MCDRAM memory and 96 GB of slower DDR4 memory. The bi-sectional bandwidth of the 2D-mesh interconnect is 700 GB/s and the cache coherency protocol is changed to MESIF [60].



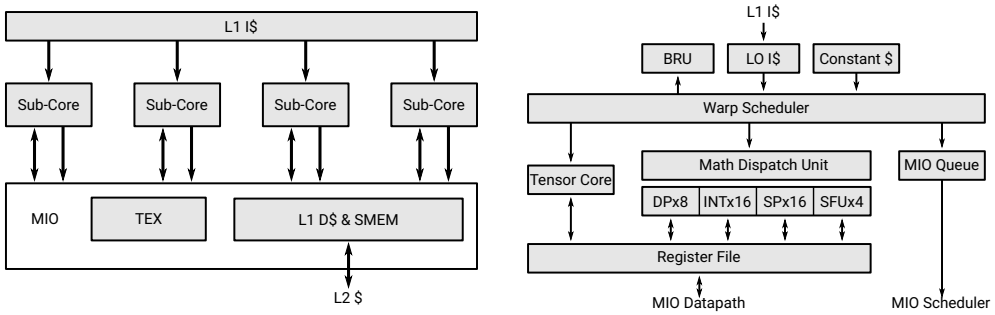
(a) 3D real-to-complex transform (b) 3D complex-to-real transform

Figure 2.2: Performance of a single KNC accelerator vs. a single Intel Xeon E5-2680v3 CPU while running in-place and out-of-place forward and inverse 3D FFTs. The analysis of performance counters on KNC strongly suggests high L2 cache miss-rate caused by false sharing.

The many-core architectures exemplified by Intel Many Integrated Cores (Intel MIC) is largely superseded by growing core counts of traditional CPUs and general purpose compute capabilities of GPUs. The successful 2D-mesh interconnect of KNL is being utilized by high-core count Intel Xeon CPUs.

2.1.2 GPU Accelerators

The GPU accelerators, originally designed to speedup 2D and later 3D graphics rendering tasks using largely fixed function hardware, gradually evolved into general purpose compute accelerators (general purpose GPUs – GPGPUs). Generally, the GPGPU architecture can be logically divided into an array of general purpose compute units (CUs), plethora of domain specific (graphics) fixed function blocks (video decode/encode, texture decompression, ray intersection, etc.) and memory hierarchy tying it all together. The fixed function hardware of the GPU can be usually left out of the discussion in the context of compute applications.



(a) Nvidia V100 SM (b) Nvidia V100 Sub-Core

Figure 2.3: Streaming Multi-processor (SM) and SIMD (Sub-Core) unit in Nvidia V100 (Volta architecture).

The design of the basic building block of the compute unit has largely settled on the SIMD architecture since general purpose computing on GPUs took off (before that, the typical architecture was VLIW). The modern compute unit (see figs. 2.3 and 2.4) consists of multiple SIMD units (here 512-bit wide), each of which has a dedicated register file. The group of these SIMDs belonging to a single compute unit share a relatively small "shared memory", which allows for communication between the SIMDs. The whole parallel machine is then exposed in the Single Instruction Multiple Threads (SIMT) paradigm, which logically views each SIMD lane as a thread of execution. The divergence of the threads executed in lock-step by a single SIMD is handled by predicate lane masking. The natural property of the SIMT paradigm is therefore hierarchical parallelism and memory coherency. This means that the threads assigned to a single SIMD share register file, while communication across SIMDs is possible only through shared memory and there is generally no implicit coherency between CUs.

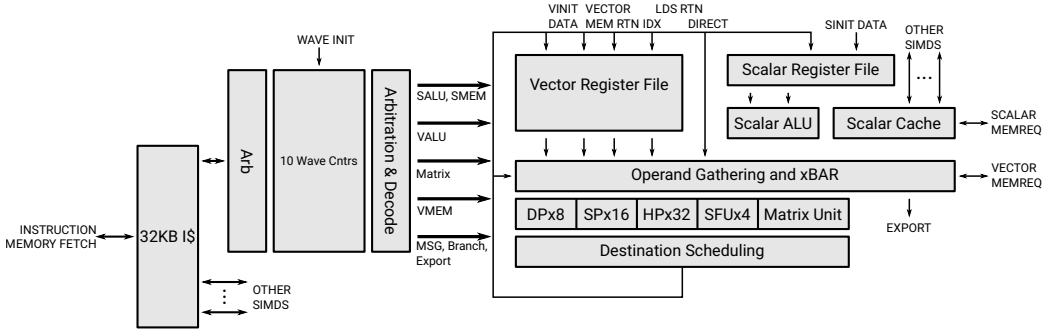
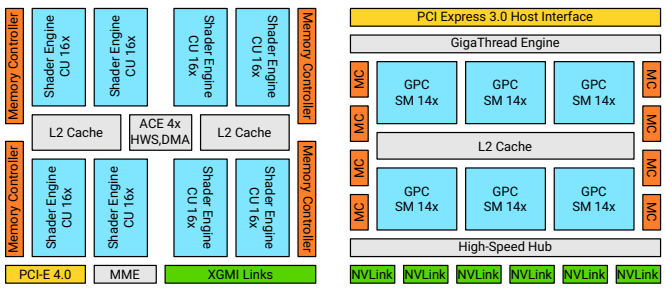


Figure 2.4: AMD CDNA architecture compute unit consists of four SIMD units with shared scalar ALU and register file. The scalar ALU is typically used to evaluate SIMD lane masking predicates or other expressions constant across the work group.

The threads running in lock-step on a single SIMD are called "warp" (NVIDIA) or "wavefront" (AMD). The size of the warp is hardware defined as an exact multiple of the SIMD width, and is executed over subsequent cycles by the SIMD (eg., 32 thread warp is executed in two cycles by V100 sub-core). Similarly, the warps assigned to a single CU (SM) are aggregated into "thread block" or "work group".



(a) AMD MI100 (b) Nvidia V100

Figure 2.5: High-level block diagram of AMD MI100 (left) and Nvidia V100 GPU (right).

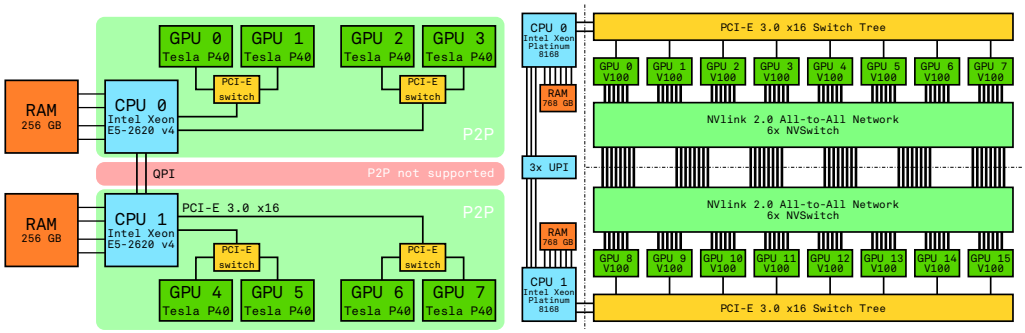
The CU typically also has dedicated L1 instruction and data caches. However, due to weak (hierarchical) memory consistency model, there is no need for cache coherency protocols and this responsibility is delegated to the user. These distributed L1 caches are typically backed by a single shared L2 (last level) cache of the GPU (see fig. 2.5).

In practice, a GPU such as NVIDIA Tesla V100 (see fig. 2.5b) may contain 5120 FP32 cores across 80 SMs achieving 15.7 TFLOP/s at 1.53 GHz. The GPU has 20 MB register file (256 kB per SM) and 7 MB of shared memory (96 kB per SM), but only 6 MB of last level cache (L2). This shows that the GPU architectures tend to rely on data reuse in lower level caches and the LLC is primarily used to coalesce memory access requests from SMs. The memory subsystem consists of eight memory controllers connected to four High Bandwidth Memory (HBM2) stacks (16 GB) over 4096-bit bus (900 GB/s).

Compared to the massive memory bandwidth of the accelerator, the interface to the rest of the system (PCI-E 3.0 or 4.0 16x) is significantly limiting – especially in communication with other accelerators in the system. Both AMD and NVIDIA introduced new high-bandwidth interfaces (XGMI and NVLink, respectively) to provision for fast direct GPU-to-GPU communication. Tesla V100 offers six NVLink 2.0 links with aggregate bandwidth 300 GB/s and cache coherency capabilities. The benefit of these advanced communication capabilities is very much application dependent. However, these capabilities are critical for applications with large working datasets (larger than local memory of a GPU) with no easy way of decoupling subsets of the data from each other.

2.2 Accelerated Nodes

An accelerated node is a basic building block in most of newly commissioned HPC clusters, with majority of these nodes being GPU accelerated. The classical approach to building an accelerated node is to equip a CPU-centric compute node with GPUs connected via PCI-E bus.



(a) Classic GPU Node (PNY) (b) NVIDIA DGX-2

Figure 2.6: Classic dual-socket GPU accelerated compute node with eight GPUs (left), and a modern dense GPU compute node with NVlink 2.0 interconnect between all 16 GPUs.

The dual-socket Intel Xeon based server such as PNY (see fig. 2.6a) is an example of such a machine. The drawbacks of standalone compute server of this type are primarily two fold: the connectivity between GPUs and CPUs is limited (pair of GPUs share 16 PCI-E 3.0 links) and direct connectivity between GPUs is limited to the interface used for GPU-to-CPU communication. There can also be issues with direct Peer-to-Peer (P2P) communication

between GPUs connected to different CPU sockets (eg. Intel Quick Path Interconnect of Intel Xeon E5 does not allow P2P PCI-E communication). The secondary issue, when building cluster of these nodes, is that the external interconnect (such as Infiniband) of such nodes is often insufficient given the massive increase in compute capabilities of the node achieved by addition of the accelerators.

Therefore, more modern compute nodes used in clusters lean more toward architecture of dense GPU accelerated compute servers designed by NVIDIA (see fig. 2.6b). The NVIDIA DGX-2 is ground up designed for GPU accelerated workloads, which is primarily enabled by all-to-all NVLink 2.0 interconnect network between all 16 Tesla V100 GPUs. Figures 2.7 and 2.8 show a comparison of PCI-E and NVLink 2 interconnect behavior on PNY and DGX-2 machines. Similarly, DGX-2 offers eight 100 Gbit/s Infiniband interfaces to connect to a cluster.

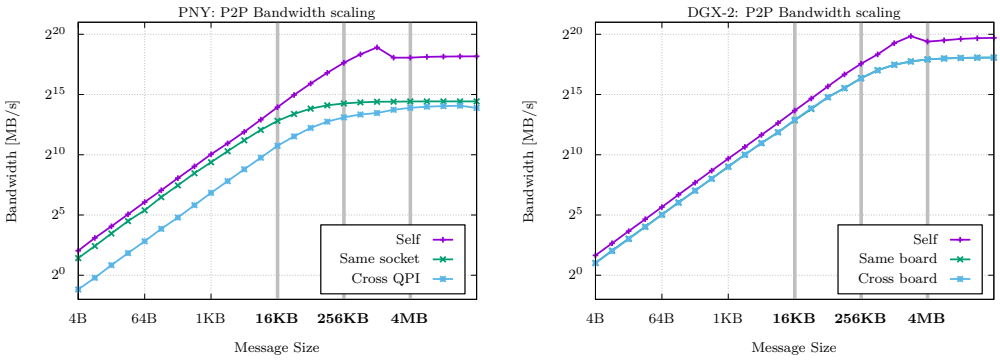


Figure 2.7: Comparison of inter-GPU communication bandwidth scaling using CUDA P2P on PNY and DGX-2. In contrast to cross QPI communication, there is no penalty in cross GPU board communication on DGX-2.

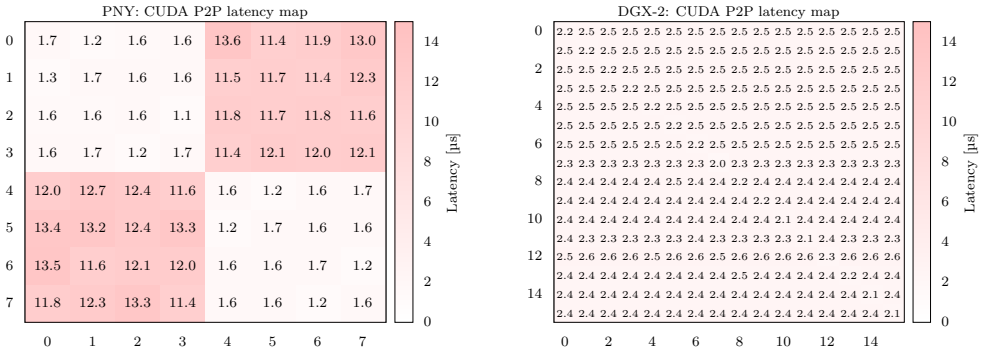
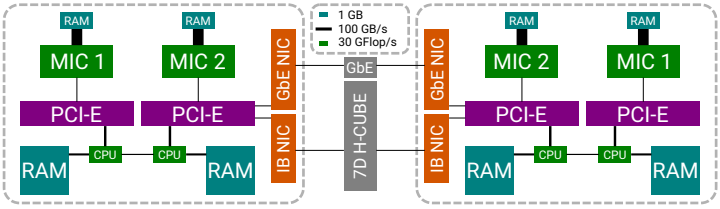


Figure 2.8: Comparison of inter-GPU communication latency using CUDA P2P on PNY and DGX-2. The latency of cross QPI communication is an order of magnitude higher than through PCI-E switches.

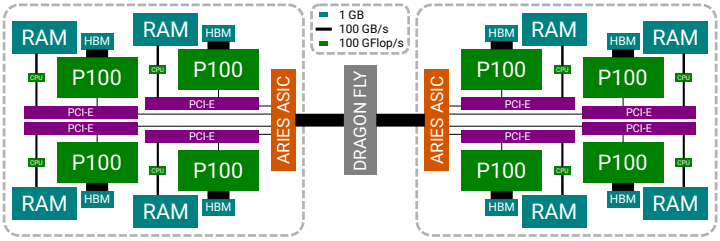
A typical GPU accelerated node falls somewhere between the PNY and DGX-2 server. For example, an accelerated node of the Karolina cluster[2] is a dual-socket AMD EPYC machine equipped with eight NVIDIA A100 GPUs with NVLink 2.0 interconnect and four 200 Gbit/s Inifiniband interfaces (note that non-accelerated nodes of the same cluster have only a single 100 Gbit/s interface).

2.3 Cluster Architectures

The interconnect network is one of points where HPC clusters diverge from cloud computing ones. The interconnect is of vital importance for efficient execution of distributed HPC workloads, while cloud computing typically focuses on workloads with lesser interdependencies.



(a) Salomon Cluster



(b) Piz Daint Cluster

Figure 2.9: Intel Xeon Phi (MIC) accelerated Salomon cluster with Infiniband based interconnect in 7D hyper-cube topology and Nvidia Tesla P100 (GPU) accelerated cluster Piz Daint with ARIES interconnect in Dragon Fly topology.

The Salomon cluster[2] (see fig. 2.9a) is a representative of the classical approach to building HPC interconnect networks. The cluster consists of 1008 nodes (432 equipped with a couple of Intel Xeon Phi 7120P) connected with InfiniBand FDR56 (56 Gbit/s) network in a 7D Enhanced hypercube topology [68]. The theoretical peak performance of an accelerated node of the cluster is about 6 TFLOP/s with network bandwidth of 3.5 GB/s, making compute-to-bandwidth ratio of the node 1715 FLOP/B.

An accelerated portion of the Karolina cluster[2] (successor of Salomon cluster) consists of 72 GPU accelerated nodes in a non-blocking Fat-tree topology built upon HDR InfiniBand network. Each accelerated node consists of two AMD EPYC 7763 CPUs and eight NVIDIA A100 GPUs with NVLink 2.0 interconnect and four 200 Gbit/s InfiniBand NICs. The R_{peak} of the node is 170 TFLOP/s (general compute), while its aggregate interconnect bandwidth is 50 GB/s doubling the compute-to-bandwidth ratio to 3400 FLOP/B.

[2]IT4Innovations National Supercomputing Center, Czech Republic

The Piz Daint cluster[3] (see Fig. 2.9b) is a more specialized GPU accelerated cluster. Each of Piz Daints' 5704 nodes consists of a single Intel Xeon E5-2690v3 (comparable to the CPUs of Salomon cluster) and a single NVIDIA P100 GPU (11 TFLOP/s per node). However, each four of these nodes are grouped and connected to a single Aries interconnect ASIC by PCI-E 3.0 x16 (32 GB/s aggregate), these groups are then interconnected in Dragonfly topology [10] using combination of backplane, metallic and fiber connections. The bandwidth available to a single node through an Aries router is about 8 GB/s (in a single direction) resulting in compute-to-bandwidth ratio of 1375 FLOP/B.

The compute-to-bandwidth ratio of compute nodes in these clusters illustrate importance of minimization of communication of HPC workloads even on platforms with heavy focus on the interconnect. Similar conclusion could be drawn by comparing internal aggregate memory bandwidth of a node to its connection to the rest of a cluster. The global communication will therefore be of the primary concern in the design of suitable numerical method.

2.4 Trends and Future Development

The development of accelerated computing architectures seems to be heading in three major directions. The first direction is introduction of increasingly more domain specific instruction set extensions to CPUs, which are enabled by increasing transistor budget. The optional subsets of AVX-512 or Intel Advanced Matrix Extensions (Intel AMX) instruction set can be viewed as domain specific accelerators tightly integrated in CPU µ-architecture. The modular nature and extensibility of RISC-V instruction set (as one of the latest developments in this field) points in similar direction. This seamless tight integration of the accelerator is both primary advantage and limitation to this approach. It allows to achieve minimal latency, but also poses severe limitations on accelerator complexity.

The second direction is Accelerated Processing Unit (APU), which integrates accelerator (typically GPU) on the CPU die or chip. The accelerator in this configuration usually shares memory or even last level cache (LLC) with the CPU. While this configuration is most widely used in mobile consumer products, there are upcoming datacenter products from all three major hardware vendors in accelerated computing (AMD, Intel and Nvidia). These Chiplet or Multi-Chip-Module designs allow more flexibility to implement complex accelerators with dedicated memory interfaces, while maintaining close integration with the CPU through optimized interfaces.

Finally, dedicated accelerators connected to the rest of the system through standardized interfaces such as PCI-E allow the most flexibility at the cost of looser integration with the CPU (lower bandwidth, possibly missing cache-coherency, etc.).

Overall, we expect the FLOP/B ratio of both accelerators and CPUs to worsen as any off-chip communication is expensive in the latency and energy terms. This is being partially counteracted by significant increases in the LLC size, which is rather new development in GPU accelerator architectures. This trend can be expected to cascade further to node and cluster level, where non-local communication becomes more and more critical part of distributed algorithms. It is also reasonable to expect more tasks to be offloaded from CPUs to accelerators and Smart NICs to reduce communication latency [86, 130].

[3] Swiss National Supercomputing Center

Chapter 3

Numerical Methods Overview

A few well known and widely used numerical schemes are considered and their numerical and computational properties compared in this chapter. The primary concern is not in precise evaluation of these schemes, but rather their asymptotic behavior. This is due to variations in possible implementation of the schemes and enormous number of problem specific adaptations which are proposed through the relevant literature. From the computational perspective, not only an asymptotic properties (such as time and space complexity) are of interest, but also possible algorithmic solutions and their mapping to real cluster architectures discussed previously.

First, the family of Finite Differences (FD) methods will be considered since it has numerous advantages in its simplicity, flexibility, ease of implementation and well developed mathematical theory. Many of those appealing properties are lost as the order of FD schemes is increased in the pursuit of higher order accuracy. Therefore, it is worth to investigate also less flexible, but much higher order spectral methods.

Although spectral methods have excellent convergence rate, they are efficiently applicable only to linear problems. This limitation can be alleviated by employing collocation (often called pseudo-spectral or transform) methods which evaluate non-linear terms in the normal space (compared to convolution required when these terms are computed in the transform space). Another limitation common to both spectral and pseudo-spectral methods is the requirement on regularity of the solutions and limited flexibility in terms of boundary conditions.

Finally, discretizations which combine spectral and element-based approaches are considered. These methods advantageously combine properties of both element and spectral methods in such a way as to maintain their respective advantages while suppressing the drawbacks. It is also important to note that these discretizations stem from weak formulation of the PDE which allows weakening of regularity constrains on the solutions. The Discontinuous Galerkin Method (DGM) will be used as a representative of these methods.

3.1 Prototype Differential Equations

One of typical approaches to evaluate the properties and suitability of numerical schemes is to select a few simple (yet still representative) PDEs and discretize them using these schemes. This allows to somewhat separate characteristic properties of an original complex problem and compare numerical schemes at more restricted tasks. Such a way of a problem

decomposition is certainly predicated upon the assumption that the results achieved on these simplified problems also apply to the original problem.

The description of ultrasound wave propagation is usually based on some form of Acoustic Wave equation [40]. The model is then augmented to capture phenomena specific to a given application. In the case of ultrasound wave propagation modeling for medical applications, these specifics can be divided into two categories: specifics of medium properties and problem configuration.

Most of typically encountered materials (ie., biological tissues) in propagation medium can be described as thermoviscous fluid (with the exception of bone tissue) [85, 102, 120] with power law absorption due to large number of relaxation processes involved [119]. The diffraction and refraction effects are usually (again with exception of hard tissue) rather weak as medium is only weakly heterogeneous and the sound propagation speed is usually close to that of water. Finally, the soft biological tissue does exhibit non-linear wave propagation which cannot be omitted from the model, particularly for high intensity ultrasound models [77].

The specifics in problem configuration are mostly related to high ratio between propagation distances, which need to be modeled, and wavelengths of interest. The large propagation distances together with the lack of diffusion leads to high sensitivity to numerical dispersion in the model. This usually leads to spatial oversampling and necessarily large and memory intensive models (if tractable at all).

3.1.1 Linear Wave Equation

The linear wave equation can be simply written (in strong form) over infinite domain for any number of spatial dimensions as

$$\frac{\partial^2 u}{\partial t^2} = c^2 \nabla^2 u \qquad \text{in } \mathbb{R}^n, \qquad \forall t \in \mathbb{R}, \tag{3.1}$$

where u is the solution and c is a wave propagation speed. Solution u is a function of both space and time, while c is usually only function of space (although it is possible to have time varying wave propagation speed). In this case, only a constant wave propagation speed is considered and c is therefore constant in both space and time.

Of course, to get a complete initial value problem, Eq. (3.1) has to be coupled with proper initial conditions (as there are no boundaries):

$$u(x,0) = \phi(x), \qquad u_t(x,0) = \psi(x), \qquad \text{in } \mathbb{R}^n. \tag{3.2}$$

For numerical experiments, this can be further simplified by restricting the problem to a single spatial dimension and only a finite domain which simplifies Eq. (3.1) to

$$\frac{\partial^2 u}{\partial t^2} = c^2 \frac{\partial^2 u}{\partial x^2} \qquad \text{in } \Omega, \qquad \forall t \in \mathbb{R}, \tag{3.3}$$

where Ω is some finite, but periodic, interval (a, b) in $\mathbb{R}$. Such a domain can be described as $(0, d) + d\mathbb{Z}$ where $d = b - a$ and effectively results in a boundary condition in the form of $u(a,t) = u(b,t) \quad \forall t \in \mathbb{R}$.

The periodic domain allows to avoid difficulties associated with boundary condition enforcement while maintaining ability to observe wave propagation over arbitrary distances. In the case of FD methods, these difficulties would typically translate into the use of special stencils [11] along the boundary. Similarly spectral collocation and Galerkin methods need

appropriate basis functions [25] to handle boundaries. When restricted to Fourier collocation methods, arbitrary boundary conditions enforcement gets even more difficult and often leads to hybrid schemes [92]. Note that this simplification doesn't negatively impact similarity between this simplified model and the complete model as free boundary is typically required. The free boundary is then usually modeled as a non-reflective absorbing layer at the boundary of the domain (known as Perfectly Matched Layer).

There are multiple ways leading to the wave equation in the form Eq. (3.3). A typical example of the scalar wave equation in a single spatial dimension is to consider a mechanical system of a simple string. The string can be imagined as an array of N point masses m connected with mass-less springs of length h. The equation is then directly derived by equating forces exerted on individual point masses, which are given by the second Newton's law and the Hooke's law, and taking the limit $N \to \infty$, $h \to 0$. This leaves the equation of the form Eq. (3.3) with $c^2 = \frac{KL^2}{M}$, where K is total spring constant, L is length of the string and M is total mass.

Linear Acoustic Wave Equation

In the realm of acoustics, the same equation can be derived as an acoustic wave equation from continuity equation and force equation which are connected through an equation of state. This approach is very general and can be used to derive acoustic wave equations accounting for variety of phenomena such as absorption, non-linearity of wave propagation or heterogeneous medium. However, to derive a linear wave equation, the simplest case of small amplitude waves in homogeneous non-absorbing medium at rest is considered here. This allows to write density, pressure and velocity variables in terms of small perturbations around respective mean values independent of time and position:

$$p' = p_0 + p, \qquad \rho' = \rho_0 + \rho, \qquad \mathbf{u}' = \mathbf{0} + \mathbf{u} \tag{3.4}$$

where p_0, ρ_0 are mean pressure and density while $\mathbf{u}' = \mathbf{u}$ as we assume medium at rest.

The continuity equation and force equation describing conservation of mass and momentum respectively can be in general written as

$$\begin{aligned} \frac{D\rho'}{Dt} + \rho' \nabla \cdot \mathbf{u}' &= 0 \\ \rho' \frac{D\mathbf{u}'}{Dt} + \nabla p' &= 0 \end{aligned} \tag{3.5}$$

where ρ' is density, $\mathbf{u}'$ is velocity and D/Dt notation represents material derivative defined as $Dy/Dt \equiv \partial y/\partial t + \mathbf{u}' \cdot \nabla y$. Plugging p', ρ' and $\mathbf{u}'$ from Eq. (3.4) to Eq. (3.5) in a single spatial dimension and neglecting higher order terms yields

$$\begin{aligned} \frac{\partial \rho}{\partial t} + \rho_0 \frac{\partial u}{\partial x} &= 0 \\ \rho_0 \frac{\partial u}{\partial t} + \frac{\partial p}{\partial x} &= 0 \end{aligned} \tag{3.6}$$

Linearized mass and momentum conservation equations are then combined by taking time derivative of the first and spatial derivative of the second and subtracting the two:

$$\frac{\partial^2 p}{\partial x^2} - \frac{\partial^2 \rho}{\partial t^2} = 0 \tag{3.7}$$

The last missing piece is the equation of state which relates density and pressure fluctuations. Here, an adiabatic process is assumed and the relation can be therefore linearized as

$$p = C\rho \tag{3.8}$$

where $C = \partial p'/\partial \rho'$ is a constant. Substitution of the linearized equation of state into Eq. (3.7) results in a familiar wave equation:

$$\frac{\partial^2 p}{\partial x^2} - \frac{1}{c^2}\frac{\partial^2 p}{\partial t^2} = 0 \tag{3.9}$$

where $c = \sqrt{C}$ is wave propagation or sound speed.

System (3.6) will be used for numerical experiments in subsequent sections as it is the form of the wave equation which is used for the derivation of the final solver. The system is however missing the equation of state, for this reason equation of state Eq. (3.8) will be substituted for ρ and ρ_0 set to 1.

$$\begin{aligned} \frac{\partial p}{\partial t} + c^2\frac{\partial u}{\partial x} &= 0 \\ \frac{\partial u}{\partial t} + \frac{\partial p}{\partial x} &= 0 \end{aligned} \tag{3.10}$$

Solution to linear Wave equation

The advantage of using one-dimensional wave equation is that it has a relatively simple general solution in the form of d'Alembert's formula:

$$u(x,t) = \frac{f(x-ct) + f(x+ct)}{2} + \frac{1}{2c}\int_{x-ct}^{x+ct} g(s)\mathrm{d}s \tag{3.11}$$

where $u(x,0) = f(x)$ and $u_t(x,0) = g(x)$ are initial conditions of the initial value problem (IVP) and c is the wave propagation speed. Note that the integral in Eq. (3.11) vanishes when $g(x) = 0$ which simplifies the solution to the point, where it becomes an average of the same wave profile ($f(x)$) moving in opposite directions.

3.1.2 Burgers' Equation

To investigate the performance of the discretizations in respect to solutions which exhibit non-linear behavior, Inviscid Burgers' equation will be considered. Burgers' equation is a hyperbolic quasilinear PDE which can be derived as a simplification of the Navier-Stokes equations for Newtonian incompressible fluid [56] by dropping its pressure term. The equation in strong form then reads as:

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u}\cdot\nabla)\,\mathbf{u} = \nu\nabla^2\mathbf{u} \qquad \text{in } \Omega, \qquad \forall t > 0, \tag{3.12}$$

where $\mathbf{u}$ is the velocity vector field and $\nu = \mu/\rho$ is the kinematic viscosity (with μ denoting viscosity and ρ density). The equation can be further simplified by assuming $\nu = 0$ which results in

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u}\cdot\nabla)\,\mathbf{u} = 0 \qquad \text{in } \Omega, \qquad \forall t > 0, \tag{3.13}$$

which is a prototype conservation equation that can develop discontinuities (shocks). In a single spatial dimension Eq. (3.12) and Eq. (3.13) turn into

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = \nu\frac{\partial^2 u}{\partial x^2} \qquad \text{in } \Omega, \qquad \forall t > 0 \tag{3.14}$$

and

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = 0 \qquad \text{in } \Omega, \qquad \forall t > 0 \tag{3.15}$$

respectively, where Ω is now chosen to be a periodic interval $(0, d) + d\mathbb{Z}$. Each equation is completed by an initial condition

$$u(x, 0) = f(x) \tag{3.16}$$

forming an IVP on a periodic domain.

The advantage of using an IVP based on the Burgers' equation for evaluation of numerical experiments is that the initial condition $f(x)$ can be selected so that the analytical solution can be found in an implicit or even closed form.

Solution to Burgers' Equation

Lets consider Eq. (3.15) with an initial condition Eq. (3.16), the solution to this problem can be constructed by the method of characteristics. The characteristic equations are

$$\frac{\mathrm{d}x}{\mathrm{d}t} = u, \qquad \frac{\mathrm{d}u}{\mathrm{d}t} = 0, \tag{3.17}$$

which (when integrated in t) show that the characteristics are lines and solution u is constant along these lines

$$x = ut + \xi, \qquad u = c, \tag{3.18}$$

where ξ is a parameter denoting the point at which the characteristic line intersects the x-axis. The constant c for a given characteristic line is therefore known from $c = u(\xi, 0) = f(\xi)$ and yields trajectory $x = f(\xi)t + \xi$. The solution then can be written in an implicit form as

$$u(x, t) = f(\xi) = f(x - ut). \tag{3.19}$$

This implicit relation describes the solution of the PDE only as long as characteristics don't intersect. The point in time at which the characteristics intersect is called breaking time (t_b) and it's the point at which shock is formed and the PDE does no longer have a classical solution. The breaking time can be determined as

$$t_b = \frac{-1}{\min f'(x)}. \tag{3.20}$$

Although the explicit solution to inviscid Burgers' equation is known only for a linear initial condition ($f(x) = ax+b$) we will still use this equation for our numerical experiments. Specifically, we will use $f(x) = \sin(x)$ as it has properties which come handy in the analysis of spectral discretizations and analytic solution $u(x, t)$ (for $t < t_b$) can still be computed by solving the implicit form Eq. (3.19) (see Figure 3.1).

For the sake of completeness, it should be noted analytical solutions to general Burgers' equation Eq. (3.14) can be found by the Cole-Hopf transformation [57]. This approach is often demonstrated on derivation of periodic or non-periodic N-wave solution [101] to the

equation. The advantage of this approach is that the N-wave solution can be found in an explicit form.

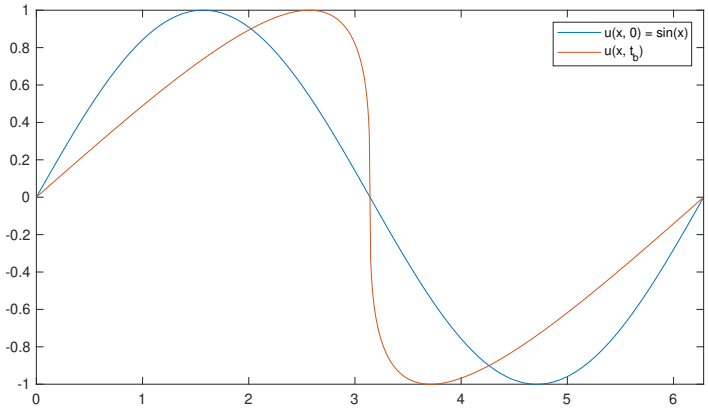


Figure 3.1: Analytic solution to inviscid Burgers' equation initialized with $\sin(x)$ at the breaking time $t_b = 1$.

3.2 Strong and Weak Formulations of PDEs

Before we dive into various approaches to discretization of our PDEs, it is useful to briefly introduce concepts of weak and strong formulation of differential problems. The *strong form* means that the PDE is required to be satisfied at each point of its space-time domain (i.e. at each point in its domain and for each time). All of the problems investigated in section 3.1 (eqs. (3.3), (3.10), (3.14) and (3.15) with necessary constraints) are in *strong form*. A solution to a problem in this formulation could be called *strong solution*. A *weak formulation*, loosens the requirements on the solution so that it has to satisfy an equation only with respect to certain "test functions" – *weak solution*. This is equivalent to reformulating the problem so that it requires a solution in the sense of a *distribution* [108] (as opposed to *strong form* which requires solution in the form of a function).

In practical terms the weak form of the PDE is obtained by requiring that the integral of the PDE against all functions from a suitable space X of test functions is satisfied. For example, the weak form of eq. (3.14) would be derived by multiplying both sides by each test function and integrating in space:

$$\int_a^b \frac{\partial u}{\partial t} v \mathrm{d}x + \int_a^b u \frac{\partial u}{\partial x} v \mathrm{d}x = \int_a^b \nu \frac{\partial^2 u}{\partial x^2} v \mathrm{d}x \qquad \forall v \in X, \qquad \forall t > 0, \tag{3.21}$$

assuming $\Omega = (a, b)$. The advantage of this *integral form* might not be immediately obvious, but it can serve as stepping stone for another weak formulation which is formed by integration-by-parts of eq. (3.21) which yields

$$\begin{aligned} &\int_a^b \frac{\partial u}{\partial t} v \mathrm{d}x - \frac{1}{2} \int_a^b u^2 \frac{\partial v}{\partial x} \mathrm{d}x + \nu \int_a^b \frac{\partial u}{\partial x} \frac{\partial v}{\partial x} \mathrm{d}x \\ &+ \left[\frac{\partial u}{\partial x} v \right]_a^b - \nu \left[\frac{\partial u}{\partial x} v \right]_a^b = 0 \qquad \forall v \in V, \qquad \forall t > 0. \end{aligned} \tag{3.22}$$

While eq. (3.21) requires solution u to be at least twice differentiable and places no restrictions on test functions, eq. (3.22) lessens regularity constrains on the solution, at the expense of increased regularity requirements placed on test functions. This means the space of test functions is restricted to a subspace V of the original space X.

Applying the same approach to first order equation such as the first equation of system (3.10) results in:

$$\int_a^b \frac{\partial \rho}{\partial t} v \mathrm{d}x + [uv]_a^b - \int_a^b u \frac{\partial v}{\partial x} \mathrm{d}x = 0 \qquad \forall v \in V, \qquad \forall t > 0, \tag{3.23}$$

where $\Omega = (a, b)$ is assumed and $V \subset X$ is space of differentiable test functions. However, solution u (and ρ for that matter) doesn't need to be differentiable in x at all.

Weak formulations also prove useful in respect to boundary conditions as they provide multiple ways to incorporate boundary into the equation. In a weak form such as eq. (3.22), the boundary condition can be enforced either by a suitable selection of test function space or through boundary terms.

While some properties of these formulations vary and each of them is useful in different context (usually leading to different discrete formulation), they are roughly equivalent at the continuous level. This equivalence is possible essentially because the space of test functions is infinite and allows to recover the strong form from the weak form. As number of independent test functions is restricted in the discretization process this is no longer possible and the equivalence is lost.

3.3 Numerical Solutions

Having our benchmark problems together with their analytic solutions and basic forms laid out, its time to turn our attention to approaches usable to formulate numerical solutions to these problems. Effectively, our task is to appropriately reduce infinite dimensionality of our problem to a finite one and find the solution there.

As an example, let's consider $u \in X$, where X is some infinite space of functions, which contains solution u to some hypothetical problem. Now, to be able to attempt to search for this solution $u(\mathbf{x}, t)$ using finite structures of our computational resources, we need to be able to describe this space X by a finite number of coefficients. Typically, the space X would be restricted to some subspace $Y \subset X$ so that $u \in Y$ and Y would then be discretized.

A typical angle of attack is to split the task into spatial (in $\mathbf{x}$) and temporal (in t) discretization problems and solve these two more or less independently. In the case of time-dependent problems, we aim to cast the problem into a finite system of ordinary differential equations (ODEs), each of which can then be solved by one of well known implicit or explicit integration methods such as Euler method, Runge-Kutta etc., see section 3.3.4. It should be noted that there is a significant body of work on space-time discretizations [87, 122], where temporal and spatial dimensions are handled at once. The relationship between spatial and temporal discretizations is also significantly tighter in methods which implement corrections of temporal errors in their spatial discretization (such as k-space pseudo spectral method which will be introduced later).

3.3.1 Finite Difference Methods

The finite difference method (FDM) is probably the simplest, yet powerful and flexible approach to spatial discretization of PDEs. The method is based on the approximation of derivatives in the PDE by a linear combination of function values at discrete grid points. This means a function $u(x)$ on some interval $\Omega = (0, L)$ is discretized as $u_i = u(x_i)$, $i = 0, 1, \ldots, N$ corresponding to a set of discrete grid points $x_i = i\Delta x$, where $\Delta x = L/N$ is a grid resolution.

The first-order derivative at some point $\bar{x}$ (or corresponding x_i in a discrete space) can be (by definition) expressed as

$$\begin{aligned}\frac{\partial u}{\partial x}(\bar{x}) = \lim_{\Delta x \to 0} \frac{u(\bar{x} + \Delta x) - u(\bar{x})}{\Delta x} &= \lim_{\Delta x \to 0} \frac{u(\bar{x}) - u(\bar{x} - \Delta x)}{\Delta x} \\ &= \lim_{\Delta x \to 0} \frac{u(\bar{x} + \Delta x) - u(\bar{x} - \Delta x)}{2\Delta x},\end{aligned} \tag{3.24}$$

which yields following approximations in the discrete space

$$\begin{aligned}\left(\frac{\partial u}{\partial x}\right)_i &\approx \frac{u_{i+1} - u_i}{\Delta x} && \text{forward difference,} \\ \left(\frac{\partial u}{\partial x}\right)_i &\approx \frac{u_i - u_{i-1}}{\Delta x} && \text{backward difference,} \\ \left(\frac{\partial u}{\partial x}\right)_i &\approx \frac{u_{i+1} - u_{i-1}}{2\Delta x} && \text{central difference.}\end{aligned} \tag{3.25}$$

Approximations in the eq. (3.25) are based on the Taylor series expansion of function u around the point x_i

$$u(x) = \sum_{n=0}^{\infty} \frac{(x - x_i)^n}{n!} \left(\frac{\partial^n u}{\partial x^n}\right)_i, \qquad u \in C^\infty([0, X]), \tag{3.26}$$

which allows to compute values of u_{i+1} and u_{i-1} as in

$$\begin{aligned}T_1 : u_{i+1} &= u_i + \Delta x \left(\frac{\partial u}{\partial x}\right)_i + \frac{(\Delta x)^2}{2} \left(\frac{\partial^2 u}{\partial x^2}\right)_i + \frac{(\Delta x)^3}{6} \left(\frac{\partial^3 u}{\partial x^3}\right)_i + \cdots, \\ T_2 : u_{i-1} &= u_i - \Delta x \left(\frac{\partial u}{\partial x}\right)_i + \frac{(\Delta x)^2}{2} \left(\frac{\partial^2 u}{\partial x^2}\right)_i - \frac{(\Delta x)^3}{6} \left(\frac{\partial^3 u}{\partial x^3}\right)_i + \cdots\end{aligned} \tag{3.27}$$

or alternatively allows to express derivative approximations from eq. (3.25) as

$$\begin{aligned}T_1 \Rightarrow \left(\frac{\partial u}{\partial x}\right)_i &= \frac{u_{i+1} - u_i}{\Delta x} - \frac{\Delta x}{2} \left(\frac{\partial^2 u}{\partial x^2}\right)_i - \frac{(\Delta x)^2}{6} \left(\frac{\partial^3 u}{\partial x^3}\right)_i + \cdots, \\ T_2 \Rightarrow \left(\frac{\partial u}{\partial x}\right)_i &= \frac{u_i - u_{i-1}}{\Delta x} + \frac{\Delta x}{2} \left(\frac{\partial^2 u}{\partial x^2}\right)_i - \frac{(\Delta x)^2}{6} \left(\frac{\partial^3 u}{\partial x^3}\right)_i + \cdots, \\ T_1 - T_2 \Rightarrow \left(\frac{\partial u}{\partial x}\right)_i &= \frac{u_{i+1} - u_{i-1}}{2\Delta x} - \frac{(\Delta x)^2}{6} \left(\frac{\partial^3 u}{\partial x^3}\right)_i + \cdots.\end{aligned} \tag{3.28}$$

Expressions T_1 and T_2 include terms containing higher order derivatives, which are unknown. The significance of these higher order terms decreases as the power of Δx goes up (assuming $\Delta x < 1$). In the case of approximations eq. (3.25), error on the order of $\mathcal{O}(\Delta x)$

(forward/backward difference) or $\mathcal{O}(\Delta x^2)$ (central difference) is introduced depending of first neglected expansion term. This way, higher-order schemes can be derived by using more points around x_i to improve convergence rate of finite differences.

In general Taylor series, however, converges only for a small subset of smooth functions and only with its generalization by using finite differences instead of derivatives. The series then converges for any bounded continuous function on $(0, \infty)$ [55]. This is an issue not only because non-smooth and discontinuous solutions are hard to represent, but it also makes imposition of boundary conditions more difficult. Assuming that the positions of discontinuities are known, both problems can be tackled by using special stencils, which capture required properties of the solution (such as immersed interface method [71]).

Linear Acoustic Wave Equation

Let's consider system (3.10) to see how FDM may be applied to spatially discretize simple system of hyperbolic first order PDEs with periodic boundary conditions. The FDM is based on a point-wise approximation of the function and we will use it in this manner to find the solution fulfilling system (3.10) at discrete points. Therefore, *strong form* of the system is considered at set of points x_i in the interval $\Omega = (0, L)$ defined as

$$x_i = i\Delta x, i = 0, 1, \ldots, N, \qquad \text{where} \qquad \Delta x = L/N. \tag{3.29}$$

The restriction of the system to these discrete points with a central differences approximation of derivatives together with an application of boundary conditions yields following system of ODEs

$$\begin{aligned} \frac{\partial p_i}{\partial t} &= -c^2 \frac{u_{i+1} - u_{i-1}}{2\Delta x}, & u_{-1} = u_N,\ u_{N+1} = u_0 \\ \frac{\partial u_i}{\partial t} &= -\frac{p_{i+1} - p_{i-1}}{2\Delta x}, & p_{-1} = p_N,\ p_{N+1} = p_0 \end{aligned} \tag{3.30}$$

where $u_i = u(x_i, t)$, $p_i = p(x_i, t)$, $u_i, p_i : i \notin 0, 1, \ldots, N$ are so called "*ghost cells*" and c is a constant over the whole domain Ω. The ghost cell or extension grid point values are computed so that the required boundary conditions are met in the original domain. The number of ghost cells depends on the order of the finite difference stencil and its type. In eq. (3.30), second order central differences are used, therefore, only a single grid point has to be added on each side of the domain.

Burgers' Equation

The general Burgers' equation eq. (3.14) is a good example of how FDM discretization would apply to a non-linear PDE with the second order term. Similarly to the wave equation, the spatial domain is restricted to a finite set of points x_i and derivatives are replaced with central differences of a required order (in both accuracy and derivative). The periodic boundary is again required and the following system of ODEs is assembled

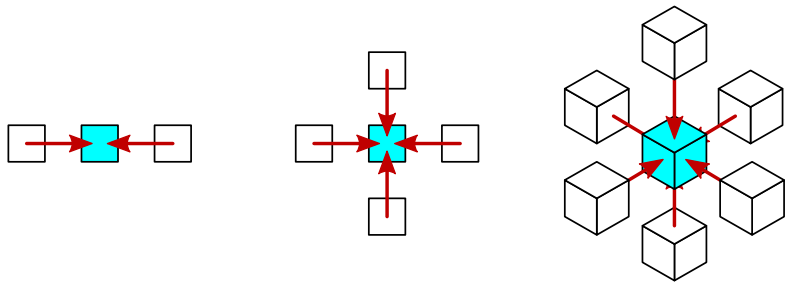
$$\begin{aligned} &\frac{\partial u_i}{\partial t} = \nu \frac{u_{i+1} - 2u_i + u_{i-1}}{\Delta x} - u_i \frac{u_{i+1} - u_{i-1}}{2\Delta x} \\ &u_{-1} = u_N,\ u_{N+1} = u_0 \end{aligned} \tag{3.31}$$

where $u_i = u(x_i, t)$ and ν is constant over the whole domain Ω, while the same ghost cells approach is taken to satisfy periodic boundary conditions in the same way as in eq. (3.30).

Generalization of FDMs

The systems of equations (3.30) and (3.31) hint on a general structure of the finite difference based methods in a single spatial dimension. The FD operators used in these systems can be described as $\mathbf{D}u_i$, where $\mathbf{D}$ is a k-diagonal matrix with each diagonal equal to a corresponding coefficient. The number of diagonals k depends on the order of the approximated derivative operator and the order of the approximation itself (e.g. central differences scheme requires values $u_{i\pm 0,1,\ldots,k/2}$ to compute derivative at i).

It is apparent that the operator $\mathbf{D}$ can be efficiently implemented as a convolution given that the boundary conditions are appropriately handled by ghost cells, special stencils or simply by a circular convolution as $\mathbf{D}$ becomes circulant in the case of a periodic boundary conditions. The similar approach can be applied even for problems in higher dimensions and it is often used to perform more complex operators such as gradient or Laplace operator in a single pass. These compound operators, which operate over data in multiple dimensions, are typically realized as n-dimensional convolutions with a von Neumann style stencils such as 3-point, 5-point or 7-point in one, two or three spatial dimensions (see fig. 3.2) and their respective extensions for higher order approximations.



(a) 3-point 1D Stencil (b) 5-point 2D Stencil (c) 7-point 3D Stencil

Figure 3.2: Narrow Von Neumann style stencils in one, two and three spatial dimensions typical for FDMs.

Computational Properties of FDMs

From the algorithmic perspective, FD operator application is a sweep through an n-dimensional array updating its elements (cells). The new value of each element is computed using its neighboring elements in a fixed pattern (stencil). While complex solvers would use these update sweeps only to implement specific operators, some simple solvers can be entirely represented as their iterative application (so called stencil codes [104]). However, in both cases it is useful to take advantage of formal descriptions which structure of stencil codes offers.

While computational complexity of stencil codes is linear in both space and time with respect to the size of the simulation domain, the matter gets more complicated when accuracy is considered. As we have shown, the approximation accuracy of FD method can be increased by either increasing the resolution or the order of the operator. Asymptotically, the error behaves as $\mathcal{O}(1/n^k)$ while time and space complexity are $\mathcal{O}(kn^d)$ and $\mathcal{O}(n^d)$ respectively, where n is the number of grid points in a single spatial dimension, d is the number of dimensions and k is an order of the approximation. This means that only algebraic convergence is achieved by the grid refinement.

Another, and increasingly more important, view on an algorithm complexity is the amount of communication that is necessary when computation is performed on architectures with distributed memory. The typical approach to distribute FDMs across a cluster is to uniformly split the domain into grid-aligned blocks. Considering such an arrangement together with previously mentioned stencils leads to a neighbor only communication pattern with $3^d - 1$ neighbors. The total amount of data exchanged during a single sweep then grows as $\mathcal{O}(2^d l^{d-1} pk)$ for narrow Von Neumann's and $\mathcal{O}(p((2k + l)^d - l^d))$ for Moore's neighborhood, where $l = np^{-1/d}$ is the edge length of each block, p is the number of blocks, d is dimensionality of the domain and k is given by the stencil size.

3.3.2 Spectral Methods

The basic idea behind spectral methods is to use a finite series expansion to represent the unknown function. The unknown function $u(x)$ is approximated by a sum of $N + 1$ "basis functions" $\phi_n(x)$ so that it can be represented by a set of coefficients a_n:

$$u(x) \approx u_N(x) = \sum_{n=0}^{N} a_n \phi_n(x). \tag{3.32}$$

The substitution of this approximate series into an equation such as

$$Lu = f(x) \tag{3.33}$$

where L is the differential or integral operator, yields a residual function of the form

$$R(x; a_0, a_1, \ldots, a_N) = Lu_N - f \tag{3.34}$$

The objective of spectral methods is the minimization of the residual function $R(x; a_n)$ as it vanishes for the exact solution. The main difference between various types of spectral approaches is in the strategy employed to minimize the residual. As much these methods may differ in the residual minimization they all rely on expansions in basis functions which span the whole domain. In contrast, element methods (such as Finite element method or FEM for short), which also rely on similar expansions, use basis functions with support restricted to a particular element (sub-interval). In both cases, the expansion is constructed for some set of grid points at once using all others in that set. Unlike FDMs, which make use of a neighborhood of each grid point independently.

At the coarsest level, these algorithms can be divided into "interpolating" and "non-interpolating" (see [18]). The term "spectral", which is now used as an umbrella term for all high-order expansion based methods, was originally reserved only for "non-interpolating" methods such as Galerkin method [41] and Tau method [91], which were developed first. These methods rely on integrating product of a function $f(x)$ and a given basis function $\phi_i(x)$ to derive the expansion coefficient a_i. The integration should be ideally analytic (e.g., strict Galerkin method), but for more complicated problems it is often necessary to use numerical quadratures (e.g., Galerkin with Numerical Integration – G-NI). In comparison, the family of "interpolating" or "pseudospectral" methods finds expansion coefficients a_i so that the expansion $f_N(x)$ agrees with $f(x)$ exactly at each grid point. The presumption here is that with the increasing density of the grid points the residual $R(x, a_i)$ is forced to vanish everywhere.

While global spectral methods offer many appealing properties, they have some major drawbacks too. Perhaps, the most troublesome property of these methods is their inflexibility in terms of domain shapes and boundary conditions. Another issue is the representation

of solutions with shock-waves or other discontinuities with known locations. Although spectral shock-capturing schemes are still an area of active research [106, 124] the rest of these issues can be largely alleviated by merging the concepts of local elements from Finite Elements Methods (FEMs) with spectral methods. This approach gives rise to schemes such as Spectral Elements Methods (SEM) which is derived directly from FEM by increasing the order of elements beyond 6-th order, and Discontinuous Galerkin Methods (DGM) which loosens coupling between elements by allowing discontinuities at boundaries. These spectral element based approaches have also significant computational implications as they lead to sparse matrices.

In following sections, we illustrate both advantages and drawbacks of spectral methods relevant to the problem at hand. Specifically, we will show classical spectral method with its limitations related to nonlinear problems which will be solved by a transition to pseudospectral method and finally discontinuous Galerkin approach will be used as a representative of element based methods.

Classical Spectral Method

The straightforward way to introduce classical spectral method is to use a Fourier Galerkin Method for spatial discretization of the wave equation (3.10). For simplicity, the problem is completed by considering periodic boundary conditions and initial conditions $p(x,0) = f(x)$ and $u(x,0) = 0$. The approximate solution can then be represented as

$$u^N(x,t) = \sum_{k=-N/2}^{N/2} a_k(t)\phi_k(x), \qquad p^N(x,t) = \sum_{k=-N/2}^{N/2} b_k(t)\phi_k(x), \tag{3.35}$$

where ϕ_k are the trial (or basis) functions, while $a_k(t)$ and $b_k(t)$ are expansion coefficients of approximations to $u(x,t)$ and $p(x,t)$ respectively (see eq. (3.32)). With approximation defined it's necessary to specify a metric in which the residual is required to vanish. The Galerkin method can be considered as a method of weighted residuals which can be written as

$$\int_0^{2\pi} \left[\mathfrak{D}\begin{bmatrix} u^N \\ p^N \end{bmatrix}\right] \psi_k(x)\mathrm{d}x = 0, \qquad \mathfrak{D} = \begin{bmatrix} \frac{\partial}{\partial t} & \frac{\partial}{\partial x} \\ c^2\frac{\partial}{\partial x} & \frac{\partial}{\partial t} \end{bmatrix}. \tag{3.36}$$

The operator $\mathfrak{D}$ describes the original wave equation and $\psi_k(x)$ are test functions, which are effectively weights in the weighted residuals approximation. The differential equation is now transformed into an integral or weak formulation described in section 3.2.

The equation (3.36) and the solution expansions eq. (3.35) have to be completed by choosing an appropriate trial and test functions. While the Galerkin approach prescribes that the test functions should be the same as trial functions, the selection of these is not entirely straightforward. Typically, the set of trial functions is selected so that it is easy to enforce the boundary conditions later. This often leads to orthogonal polynomials such as Chebyshev polynomials [97]. Another choice, especially suitable for problems with periodic boundary conditions, are trigonometric polynomials, which are used here:

$$\phi_k(x) = e^{ikx}, \qquad \psi_k(x) = \frac{1}{2\pi}e^{-ikx}. \tag{3.37}$$

Here, i is the imaginary unit and k is an integer wave-number. It should be noted that these two functions are essentially the same and that they are pairwise orthogonal on $[0, 2\pi)$,

which is a very useful property. Expanding approximations eq. (3.35) and substituting these basis functions into eq. (3.36) yields

$$\frac{1}{2\pi}\int_0^{2\pi}\left[\sum_{l=-N/2}^{N/2}\left(\mathfrak{D}\begin{bmatrix}a_l(t)\\ b_l(t)\end{bmatrix}\right)e^{ilx}\right]e^{-ikx}\mathrm{d}x = 0. \tag{3.38}$$

At this point, both equations can be analytically differentiated in the spatial dimension. This is reflected in the following equation by a simple change of the operator $\mathfrak{D}$ to $\mathbf{D}$ as only the basis and test functions are dependent on x:

$$\frac{1}{2\pi}\int_0^{2\pi}\left[\sum_{l=-N/2}^{N/2}\left(\mathbf{D}\begin{bmatrix}a_l(t)\\ b_l(t)\end{bmatrix}\right)e^{ilx}\right]e^{-ikx}\mathrm{d}x = 0, \qquad \mathbf{D} = \begin{bmatrix}\frac{\mathrm{d}}{\mathrm{d}t} & il\\ c^2 il & \frac{\mathrm{d}}{\mathrm{d}t}\end{bmatrix}. \tag{3.39}$$

Finally, propagating the integral and test function e^{-ikx} into the sum and taking the advantage of $\frac{1}{2\pi}\int_0^{2\pi} e^{ilx}e^{-ilx} = \sigma_{kl}$, yields the set of dynamical equations

$$\frac{\mathrm{d}b_k}{\mathrm{d}t} + c^2 ika_k = 0, \qquad \frac{\mathrm{d}a_k}{\mathrm{d}t} + ikb_k = 0, \qquad k = -N/2,\ldots,N/2, \tag{3.40}$$

which can be used with a suitable time-stepping scheme to obtain the approximate solution at an arbitrary time t. According to the strict Galerkin method, the initial conditions $a_k(0)$ and $b_k(0)$ for system (3.40) should be computed analytically as

$$a_k(0) = \int_0^{2\pi} u(x,0)\psi_k(x)\mathrm{d}x, \qquad b_k(0) = \int_0^{2\pi} p(x,0)\psi_k(x)\mathrm{d}x. \tag{3.41}$$

Most if not all reasonably complicated problems will require a numerical quadrature to be used for evaluation of these integrals instead. Considering that the chosen approximation (eq. (3.35) together with eq. (3.37)) is a truncated Fourier series, one would use Fourier transform to transition between coefficients and original function values. The solution at time t can be recovered similarly by an inverse Fourier transform.

Perhaps, one of the most severe limitations of the purely spectral approach is revealed when a non-linear or variable (multiplicative) parameter problems are considered. The simplest example could be a spatially variable wave speed $c(x)$ in the wave equation. However, to stick with our theme of showing both linear and non-linear problems, let's consider the inviscid Burgers' equation (eq. (3.15)) with periodic boundary conditions. Taking the conservative form of inviscid Burgers' equation

$$\frac{\partial u}{\partial t} + \frac{1}{2}\frac{\partial u^2}{\partial x} = 0, \tag{3.42}$$

and following the same procedure as above, we arrive at the integral form

$$\frac{1}{2\pi}\int_0^{2\pi}\left[\frac{\partial}{\partial t}\sum_l a_l(t)e^{ilx} + \frac{\partial}{\partial x}\frac{1}{2}\sum_p\sum_q a_p(t)a_q(t)e^{i(p+q)x}\right]e^{-ikx}\mathrm{d}x = 0. \tag{3.43}$$

Although the overall form is very similar to eq. (3.39), the double summation should seem rather suspicious from the computational perspective (as it means at least $O(N^2)$ complexity). Taking the integral as before results once again in a set of dynamical equations

$$\frac{\mathrm{d}a_k}{\mathrm{d}t} + \frac{ik}{2}\sum_{p+q=k} a_p a_q = 0, \qquad k = -N/2,\ldots,N/2, \tag{3.44}$$

which still contain the double summation in the form of convolution thus prohibitively increasing the complexity of each time step in the final algorithm from $O(N)$ to $O(N^2)$.

Pseudospectral Method

Whereas Fourier Galerkin method minimizes the residual in integral terms, the pseudospectral or collocation approach requires that the approximate solution satisfies the original equation exactly at a set of collocation points x_j. The choice of these points is tightly related to the selected expansion basis so that the high accuracy of approximation and quadratures is achieved. In contrast to conventional Galerkin method, the collocation methods are implemented entirely in the terms of nodal values $u_j(t) = u^N(x_j, t)$. The nodal expression of the approximate solution can be written as

$$u^N(x,t) = \sum_{j=0}^{N-1} u_j(t)\psi_j(x), \tag{3.45}$$

where ψ_j are approximation of shifted discrete delta-functions (formally distributions) satisfying $\psi_j(x_i) = \delta_{ij}$ for $0 \leq i, j \leq N-1$. The choice of appropriate approximation polynomials is determined by the selected expansion basis, which in turn strongly depends on the boundary conditions of the equation at hand.

In the case of inviscid Burgers' equation in its conservative form (eq. (3.42)) with periodic boundary condition, the suitable expansion can be derived from Fourier basis as

$$\psi_j(x) = \frac{1}{N} \sum_{k=-N/2}^{N/2-1} e^{ik(x-x_j)}, \tag{3.46}$$

which can be (in combination with eq. (3.45)) easily derived from the interpolation on modal Fourier expansion by rearranging sums. The substitution of this expansion of approximate solution into the strong form of Burgers' equation yields

$$\frac{\partial u^N}{\partial t} + \frac{\partial}{\partial x}\mathcal{F}\left(u^N\right)\Big|_{x=x_j} = 0, \qquad j = 0, \ldots, N-1, \tag{3.47}$$

where u^N is a nodal expansion eq. (3.45) with basis defined by eq. (3.46). While the first term simply collapses to the nodal values $u_j(t)$ the flux term $\mathcal{F}\left(u^N\right) = \frac{1}{2}\left(u^N\right)^2$ requires a bit more attention. The flux has to be replaced with its numerical variant, which is evaluated on the collocation points and only then interpolated:

$$\mathcal{F}^{\mathcal{N}}\left(u^N\right) = I_N\left[\mathcal{F}\left(u^N\right)\right] = \sum_{j=0}^{N-1}\left[\frac{1}{2N}u_j(t)^2 \sum_{k=-N/2}^{N/2-1} e^{ik(x-x_j)}\right]. \tag{3.48}$$

Here I_N is the Fourier interpolation operator, which is used to interpolate the flux. This operator can be analytically differentiated in x and written as a Fourier collocation differentiation matrix

$$(D_N)_{jl} = \frac{\partial}{\partial x}(I_N)_{jl} = \frac{1}{N}\sum_{k=-N/2}^{N/2-1} ike^{ik(x_j-x_l)}. \tag{3.49}$$

In this case, the result is evaluated at points x_j, while the differentiated function is known at points x_l. Both x_j and x_l are expected to be in the interval $[0, 2\pi)$.

The substitution of the numerical flux F^N together with the Fourier collocation differentiation matrix D_N yields a system of dynamical equations

$$\frac{\mathrm{d}u_j}{\mathrm{d}t} + \sum_{l=0}^{N-1} (D_N)_{jl} \frac{1}{2} u_l^2 = 0, \qquad j = 0, \ldots, N-1, \tag{3.50}$$

which can be combined with a suitable time-stepping scheme to obtain the solution.

Once again, each time step requires a matrix-vector multiplication or a pair of space-to-space transforms to evaluate the derivative operator in a transform space. This is a slight improvement over the classical spectral method as expensive $O(N^2)$ operations are now restricted only to evaluation of derivative operators. Moreover, given a suitable basis where a cheaper transform method is available, this cost can be further reduced.

Computational Properties of Spectral Methods

Considering non-linear problems the price for an exponential convergence achieved by these spectral approaches is an introduction of either convolutions or multiple space-to-space transforms in each time step. Although the complexity of these operations in general is $O(N^2)$ it can be significantly reduces by the choice of suitable basis.

The popular options are, therefore, Fourier or Chebyshev basis, both of which offer transform algorithms based on Fast Fourier Transform (FFT [30]) and thus requiring only $O(N \log_2(N))$ time per transform. The choice between the two is mostly driven by boundary conditions of the problem, where Chebyshev basis offers more flexibility. The disadvantage of Chebyshev basis is however that a non-uniform grid-point distribution is required which tends to severely limit time-step size for large problems (see fig. 3.3). Many applications therefore choose Fourier basis even in cases where this choice means that the explicit boundary handling will be necessary.

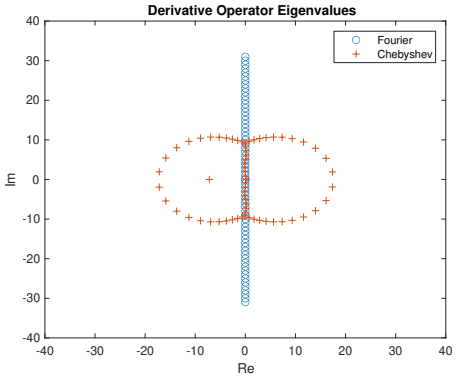


Figure 3.3: Eigenvalues of 1st order derivative operator using Fourier or Chebyshev basis.

3.3.3 Element Methods

The discretization strategies discussed so far assumed that our toy problems are solved on an uniformly spaced grid-points. These approaches can be readily extended to Cartesian grids (tensor product domains) and even further to other regular grids. However, only FDM is relatively easily applicable to problems with more complicated geometries. The

global nature of spectral methods in combination with smoothness requirements they impose on the solution limits their applicability to problems with a complex geometry and discontinuities in the medium properties.

The idea of finite element methods is to split the original domain into a set of coupled subdomains (or elements)

$$\Omega = \bigcup_{e=1}^{N_e} \Omega^{(e)}, \tag{3.51}$$

where Ω is the simulation domain, $\Omega^{(e)}$ is a single element and N_e is the total numbers of elements in the decomposition. The shape of the elements is typically chosen to be easily tractable (k-simplexes or hypercubes) or otherwise suitable for given problem (pyramids, prisms, polyhedrons, curvilinear, etc.).

Practically speaking the solution u is once again found by minimizing the residual R over its finite sum approximation u_N

$$\begin{aligned} u(x) \approx u_N(x) &= \sum_{n=0}^{N} a_n \phi_n(x), \qquad u_N \in V \\ R(x; a_0, a_1, \ldots, a_N) &= L u_N, \end{aligned} \tag{3.52}$$

where L is a differential operator. Typical strategies used to minimize the residual R are Galerkin projection (Galekrin FEM) and least squares (LS FEM) [54, 131]. Both of these strategies can be viewed as specializations of the method of weighted residuals, which demands a residual R to be orthogonal to some space W:

$$(R, v) = \int_\Omega R v \mathrm{d}x = 0, \qquad \forall v \in W, \tag{3.53}$$

where $W = V$ for Galerkin method or $W = \mathrm{span}\{w_0, \ldots, w_N\}$ and $w_i = \partial R / \partial a_i$ in the case of least squares method.

The key differentiating factor of finite element methods is the selection of the test function space. The test functions are selected so that they have only local support in each element $\Omega^{(e)}$ (see fig. 3.4). The adjacent elements are then suitably coupled so that coherent global solution U_n could be formed. This approach ultimately results in more or less sparse system of equations, which is to be solved at each time step.

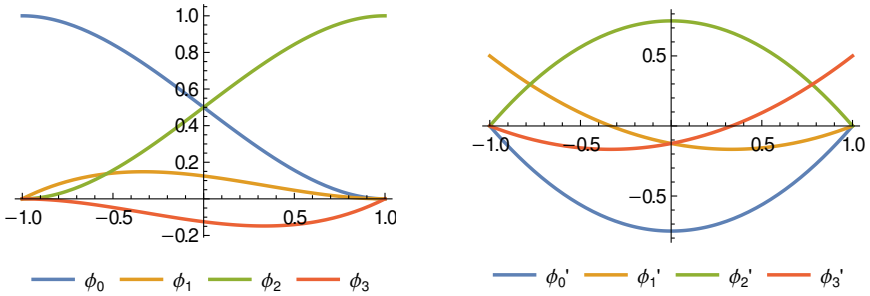


Figure 3.4: Third order hat polynomial FEM basis (left) and its derivative (right).

Finally, these methods use notion of the reference element, which allows to factor out geometry information from per-element operations. This is achieved by mapping between each element in the domain and the reference element on which the test function bi-linear forms are precomputed.

Finite and Spectral Element Method

The finite element methods in their most widely used form use a Galerkin projection to minimize the residual. This also means that the space of the test and trial (basis) functions is the same. The adjacent elements are coupled by sharing coefficients a_n, which belong to nodes at shared boundary between elements. The basic variant of FEM would typically use a low-order piecewise basis such as Lagrange polynomials on equispaced nodes. However, Hermite polynomials may be also used when C^1 continuity between elements is required.

The spectral element method extends FEM by using high degree piecewise polynomial basis functions such as Legendre polynomials over non-uniformly spaced nodes. For example Legendre-Gauss-Lobatto grids [7, 20] are often used as they can be directly used for numerical quadrature. See [34] for comprehensive overview of classical orthogonal polynomials.

To better illustrate this method's derivation and numerical performance, we use a Galerkin FEM with a Lagrange basis to discretize a simple wave equation. We start with the first order system (3.10) over $\Omega = (0, 2\pi)$ with periodic boundary conditions. The interval Ω is then uniformly subdivided to form a mesh $\mathcal{T}_h$ with element size $h = 2\pi/N_e$. At this point we can define trial and test functions space as

$$V_h = \left\{ u_h \in \mathcal{C}(\Omega) \,\middle|\, u_h\big|_K \in \mathbb{P}_n, \quad \forall K \in \mathcal{T}_h \right\}, \tag{3.54}$$

which is space of $\mathcal{C}^0$ continuous functions on Ω, where each element K is represented by an n-th order Lagrange polynomial. The solution u and ρ can now be approximated by coefficients a_j and b_j, respectively, using basis functions $\phi_j \in V_h$ as

$$u^N(x,t) = \sum_j a_j(t)\phi_j(x), \qquad p^N(x,t) = \sum_j b_j(t)\phi_j(x). \tag{3.55}$$

Rewriting the system (3.10) as differential operator $\mathfrak{D}$ and applying Galerkin projection yields

$$\begin{gathered} \mathfrak{D}_t = \begin{bmatrix} \partial_t & 0 \\ 0 & \partial_t \end{bmatrix}, \ \mathfrak{D}_x = \begin{bmatrix} 0 & \partial_x \\ \partial_x & 0 \end{bmatrix}, \ \mathbf{C} = \begin{bmatrix} 0 & 1 \\ c^2 & 0 \end{bmatrix}, \ \mathbf{u^N} = \begin{bmatrix} u^N & p^N \end{bmatrix}^T \\ \int_\Omega \mathfrak{D}_t \mathbf{u^N} \psi \mathrm{d}x + \int_\Omega \mathfrak{D}_x \mathbf{C} \mathbf{u^N} \psi \mathrm{d}x = 0, \qquad \forall \psi \in V_h, \end{gathered} \tag{3.56}$$

which can be rearranged by expanding $\mathbf{u^N}$ to a vector of coefficients $\mathbf{u} = \begin{bmatrix} a_j & b_j \end{bmatrix}^T$ and realizing that $\mathbf{u}$ is now independent of x we get

$$\mathfrak{D}_t \sum_j \mathbf{u} \int_\Omega \phi_j \psi_i \mathrm{d}x + \mathbf{C} \sum_j \mathbf{u} \int_\Omega \phi_j' \psi_i \mathrm{d}x = 0, \qquad \forall i \in \mathcal{I}. \tag{3.57}$$

The bi-linear forms are now independent and can be pre-computed, arriving on system of first order ODEs

$$\begin{gathered} \begin{bmatrix} \partial_t \mathbf{I_N} & \mathbf{0} \\ \mathbf{0} & \partial_t \mathbf{I_N} \end{bmatrix} \begin{bmatrix} \mathbf{0} & \mathbf{M} \\ \mathbf{M} & \mathbf{0} \end{bmatrix} \mathbf{u} = - \begin{bmatrix} \mathbf{0} & \mathbf{K} \\ \mathbf{K} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{0} & \mathbf{I_N} \\ c^2 \mathbf{I_N} & \mathbf{0} \end{bmatrix} \mathbf{u} \\ \mathbf{M}_{j,i} = \int_\Omega \phi_j \psi_i, \quad \mathbf{K}_{j,i} = \int_\Omega \phi_j' \psi_i \end{gathered} \tag{3.58}$$

where matrices $\mathbf{M}$ and $\mathbf{K}$ are generally being called mass and stiffness matrix. The system is to be solved at each evaluation of time derivatives needed by the time stepping scheme of choice.

The trial and test function space V_h, together with the mesh $\mathcal{T}_h$, determine the structure of both mass and stiffness matrices. The matrices are sparse as each function $\phi_j \in V_h$ is zero everywhere except element K and possibly its neighbors. This allows to assemble the matrices element-by-element such as

$$\mathbf{M} = \sum_k \mathbf{M}^k, \qquad \mathbf{K} = \sum_k \mathbf{K}^k, \tag{3.59}$$

where $\mathbf{M}^k_{i,j} = \int_k \phi^k_i \psi^k_j$ and $\mathbf{K}^k = \int_k \phi'^k_i \psi^k_j$ are matrices local to element $k \in \mathcal{T}_h$. For example, the decomposition in question with $\mathbb{P}_n$ polynomial basis results in $(n+1)$-by-$(n+1)$ local matrices which are assembled into global block diagonal matrices. Further, the mass matrix $\mathbf{M}$ is often symmetric, positive-definite and well conditioned. These properties enable the approximation of the mass matrix by mass lumping process, which replaces the matrix by its purely diagonal approximation $\widetilde{\mathbf{M}}$. This in turn eliminates the need to solve the system (3.58) as $\widetilde{\mathbf{M}}$ can now be easily inverted thus making FEM fully explicit method. The mass lumping process can also be used as means to preserve locality, which may be useful in modeling certain physical phenomena (such as wave equation [53]).

Finally, to illustrate elimination of some per element calculations, we show how to construct matrices $\mathbf{M}^k$ and $\mathbf{K}^k$ using reference element mapping. Let's define reference element R as $\Omega^{(R)} = [-1, 1]$ and a linear affine mapping from the reference element to an element $\Omega^{(e)} = [x_L, x_R]$ in the domain as

$$x = \frac{1}{2}(x_L + x_R) + \frac{1}{2}(x_R - x_L)\, r, \qquad x \in \Omega^{(R)},\ r \in \Omega^{(e)}. \tag{3.60}$$

The local mass and stiffness matrices can then be expressed in terms of integrals over the reference element R as

$$\mathbf{M}^k_{i,j} = \int_R \phi^R_i(r)\psi^R_j(r)\frac{\mathrm{d}x}{\mathrm{d}r}\mathrm{d}r, \qquad \mathbf{K}^k_{i,j} = \int_R \phi'^R_i(r)\psi^R_j(r)\mathrm{d}r, \tag{3.61}$$

where $\frac{\mathrm{d}x}{\mathrm{d}r} = (x_R - x_L)/2$ is the stretch factor of the mapping eq. (3.60). Depending on the element type and dimensionality of the problem, it may be useful to introduce coordinate systems such as barycentric coordinates in derivation of these relationships. While this process may get quite complicated, it is still preferable over computing these integrals for each element separately.

Discontinuous Galerkin Method

Discontinuous Galerkin Method (DGM) or Discontinuous Galerkin FEM (DG-FEM) takes FEM approach one step further by incorporating some ideas of Finite Volume Method (FVM). The FVM [69] treats divergence terms in PDEs by splitting the space into small cells (elements) in which volume integrals are used to represent the local volume average. These volume integrals can be converted to surface integrals using the divergence theorem. The problem is then described in terms of flows (fluxes) between adjacent cells and the solution is naturally discontinuous (piecewise constant) as it consists of cell averages.

The DGM expands the space of solutions used in FEM by requiring the solution to be only piecewise continuous. It, just like FVM, allows for discontinuities in the solution at element boundaries. Formally, such a solution space can be described as "broken Sobolev space" [33]

$$V_h = \left\{ u_h \in L^2(\Omega) \,\middle|\, u_h\big|_K \in \mathbb{P}_n(K), \quad \forall K \in \mathcal{T}_h \right\}, \tag{3.62}$$

which consists of n-th degree polynomials over each element K in the mesh $\mathcal{T}_h$. The adjacent elements are coupled by numerical flux, which is typically a low order approximation of the solution to the Riemann problem [111].

The weak formulation eq. (3.56) derived to find a FEM formulation of wave equation problem eq. (3.10) can be used to highlight changes necessary to arrive at the DGM discretization. While the definition of the solution space V_h had to be changed the mesh decomposition $\mathcal{T}_h$ with elements K remain the same. Splitting the domain Ω into constituent elements and applying divergence theorem yields

$$\sum_{K\in\mathcal{T}_h}\left[\int_K \mathcal{D}_t\mathbf{u}^{\mathbf{N}}\psi\mathrm{d}x+\int_{\partial K}\mathbf{C}\mathbf{u}^{\mathbf{N}}\mathbf{n}_K\psi\mathrm{d}S-\int_K\mathbf{C}\mathbf{u}^{\mathbf{N}}\psi'\mathrm{d}x\right]=0,\quad\forall\psi\in V_h, \tag{3.63}$$

where $\mathbf{n}_K$ denotes outward unit normal of ∂K. At this point, we still require the element surface integrals $\int_{\partial K}\mathbf{C}\mathbf{u}^{\mathbf{N}}\mathbf{n}_K\psi\mathrm{d}S$ at interior faces between adjacent elements to cancel out, thus the approximate solution still has to be continuous. This restriction is alleviated by replacing these integrals with a numerical flux function $\hat{f}$, which together with expanding $\mathbf{u}^N$ and restriction of the trial/test function space V_h results in

$$\sum_{K\in\mathcal{T}_h}\left[\mathcal{D}_t\sum_j\mathbf{u}\int_K\phi_j\psi_i\mathrm{d}x+\left[\hat{F}\left(\mathbf{u}\right)\right]_{x_L^K}^{x_R^K}-\mathbf{C}\sum_j\mathbf{u}\int_K\phi_j\psi_i'\mathrm{d}x\right]=0,\quad\forall i\in\mathcal{I}, \tag{3.64}$$

where ϕ_j,ψ_i are trial and test functions from finite subspace of V_h, x_L^K, x_R^K denote left and right endpoint of the element K and $\hat{F}(\mathbf{u})=\hat{f}\left(\mathbf{C}\sum_j\mathbf{u}\phi_j\right)\psi_i$ is the numerical flux.

The primary purpose of numerical flux functions is to describe the communication between elements, allowing recovery of the global solution, while permitting discontinuities at element boundary. However, they also offer a way of injecting problem specific knowledge into the solver. Consider a simple advection problem with velocity c, the upwind flux [89]:

$$\hat{f}\left(u\right)=\begin{cases}cu^- & \text{if } c>0\\ cu^+ & \text{if } c<0\\ 0 & \text{otherwise}\end{cases}, \tag{3.65}$$

can be used to express the advection process in the solver. Here u^- and u^+ denote solution values at respective sides of the element boundary. The numerical flux can also have a significant impact on the properties of the solver (see [73, 99]). While there are numerical fluxes optimized for wave problems [28, 58, 99], we use central flux for its simplicity. The central flux is a non-directional numerical flux which averages solution values across the boundary as

$$\hat{f}\left(u\right)=\frac{u^-+u^+}{2}. \tag{3.66}$$

The last missing piece necessary for the derivation of the semi-discrete matrix form is the choice of approximation polynomials $\mathbb{P}_n$ for test and trial functions. This time, we use Legendre polynomials over Gauss-Legendre points (see fig. 3.5), which offer number of useful mathematical properties. The set of Legendre polynomials $\mathbb{P}_n$ up to the order n is, by definition, pairwise orthogonal over $[-1,1]$ with respect to the weight function $w(x)=1$ and therefore

$$\int_{-1}^{1}\phi_i^R\psi_j^R\mathrm{d}x=0\quad\text{if } i\neq j \tag{3.67}$$

holds for test and trial functions $\phi_i^R, \psi_j^R \in \mathbb{P}_n$ over the reference element R. This naturally leads to a purely diagonal mass matrix $\mathbf{M}$, which can be trivially inverted. Having element internal nodes located at zeroes of Legendre polynomials allows to directly compute integrals using Gauss-Legendre quadrature, which is exact for polynomials of degree up to $2n-1$. One of the disadvantages of these points is that they don't include interval endpoints, which are needed in evaluation of numerical fluxes and extrapolation has to be used.

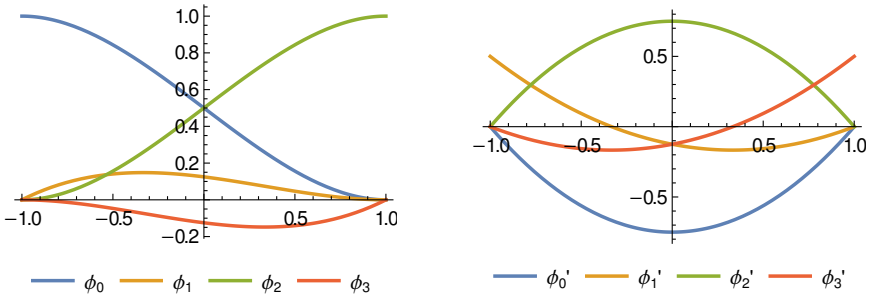


Figure 3.5: Third order Legendre polynomial basis used in Gauss-Legendre DGM elements (left) and its derivative (right).

Finally, having derived a weak form of the problem and chosen a numerical flux function and approximation polynomials, the semi-discrete system can be written as

$$\begin{bmatrix} \partial_t \mathbf{I_N} & \mathbf{0} \\ \mathbf{0} & \partial_t \mathbf{I_N} \end{bmatrix} \mathbf{u} = \begin{bmatrix} \mathbf{0} & \mathbf{M} \\ \mathbf{M} & \mathbf{0} \end{bmatrix}^{-1} \left(\begin{bmatrix} \mathbf{0} & \mathbf{K} \\ \mathbf{K} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{0} & \mathbf{I_N} \\ c^2\mathbf{I_N} & \mathbf{0} \end{bmatrix} \mathbf{u} - \begin{bmatrix} \mathbf{0} & \mathbf{F} \\ \mathbf{F} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{0} & \mathbf{I_N} \\ c^2\mathbf{I_N} & \mathbf{0} \end{bmatrix} \mathbf{u} \right) \tag{3.68}$$

where $\mathbf{M}$, $\mathbf{K}$ and $\mathbf{F}$ are global mass, stiffness and numerical flux matrices, respectively. Both mass and stiffness matrices have a block-diagonal structure as they consist of per-element local blocks $\mathbf{M} = \sum_k \mathbf{M}^k$ and $\mathbf{K} = \sum_k \mathbf{K}^k$, respectively, which are essentially independent of each other. The numerical flux matrix $\mathbf{F}$ is typically block-sparse and while it can be constructed in terms of blocks $\mathbf{F} = \sum_k \mathbf{F}^k$, these blocks are no longer independent as they tie together adjacent elements. These local matrices can be constructed over a reference element R using affine projections as

$$\begin{aligned} \mathbf{M}_{i,j}^k &= \int_R \phi_i^R(r)\psi_j^R(r)\frac{\mathrm{d}x}{\mathrm{d}r}\mathrm{d}r, \quad \mathbf{K}_{i,j}^k = \int_R \phi_i^R(r)\psi_j^{R\prime}(r)\mathrm{d}r, \\ \mathbf{F}_{i,j}^k &= \frac{\mathbf{L}_{i,j}^{k+1} + \mathbf{R}_{i,j}^k}{2}\psi_j^R(x_r^k) - \frac{\mathbf{R}_{i,j}^{k-1} + \mathbf{L}_{i,j}^k}{2}\psi_j^R(x_l^k), \end{aligned} \tag{3.69}$$

where $\mathbf{M}^k$, $\mathbf{K}^k$ are $(n+1)$-by-$(n+1)$ matrices whereas $\mathbf{F}^k$ is $(n+1)$-by-$3(n+1)$ as it's non-zero for elements k and $k \pm 1$. Matrices $\mathbf{L}^k$ and $\mathbf{R}^k$ denote extrapolation of the solution to left and right edge of element k.

Slight complications arise in the case of nonlinear problems such as Burgers equation. The choice of modal approximation by Legendre polynomials leads to the same issues as those encountered in spectral methods. The solution is to either switch to nodal approximation, evaluate products as convolutions or use transform methods. However, in this case the transform methods are not as compelling because the fast transforms (such as FFT) are not available for approximations typically used in element methods. While these approaches with $O(N^2)$ time complexity are not really usable for global methods, they are feasible for element methods as N is determined by the order of the local approximation which is orders of magnitude smaller than the global approximation.

Computational Properties of Element Methods

There are several factors which make the analysis of computational complexity of element methods significantly more complicated. In situations where an element method is applied to discretize the spatial part of the time dependent problem, we can expect to end up with a sparse system of equations $\mathbf{M}\vec{x} = \mathbf{K}\vec{b}$, which has to be solved at each evaluation of spatial derivatives. However, the properties of the sparse matrices $\mathbf{M}$ and $\mathbf{K}$ will depend on factors such as weak formulation of the problem, mesh and domain structure in combination with the element shape, properties of basis and flux functions. The structure of these matrices may be also affected by incorporating domain decomposition schemes such as FETI-DP [36], which allow to improve parallel efficiency over general linear solvers.

Given the wide variety of possible approaches to element methods, let's consider the best case, where the mass matrix $\mathbf{M}$ is diagonal and can be trivially inverted beforehand. In this case, the cost of the right hand side evaluation can be reduced down to matrix-vector multiplication, where the matrix is block sparse. The bandwidth of the matrix is primarily determined by the number of nodes (FEM) or faces (DGM) and the order of approximating polynomials. This would make sequential time complexity of each time step $O(NW)$, where N is the number of degrees of freedom and W is the bandwidth of the matrix.

Considering parallel solvers running on distributed memory architectures, good scaling close to $O(NW/P)$, where P is number of processors, can be expected. The communication should be only local and scale with element basis order and shape as it determines the number of shared degrees of freedom between adjacent elements, which may belong to separate shared memory pools.

3.3.4 Temporal Discretization Methods

The solutions to semi-discrete problems derived in previous sections can be approximated by one of well known numerical integration methods for ordinary differential equations (ODEs). The typical numerical method will treat the temporal dimension by splitting it into finite number of (small) sub-intervals (time steps) at which ODE system becomes an algebraic system. This system then has to be solved (explicitly or implicitly) at each of the time steps. The major part of the time step is (often multiple) evaluation of right-hand side (RHS) of the original semi-discrete problem. The number of time steps and the number of these evaluations at each time step, therefore, largely determines overall computational time.

An appropriate choice of the numerical integration method then has to balance the number of RHS evaluations with accuracy requirements and stability of the solution. For example, a stiff ODE system may necessitate the use of an implicit method, which allows to guarantee stability with larger time step.

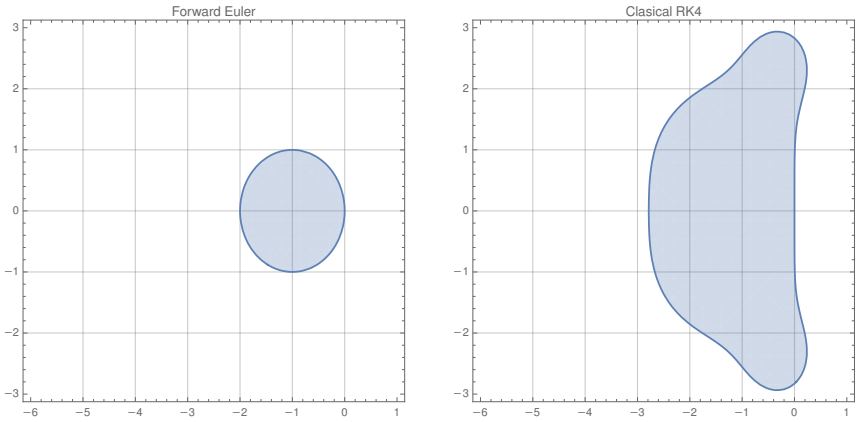
The canonical family of these numerical integration methods are Runge-Kutta methods, which include Euler method and allow for construction of higher order methods. These higher order methods, however, require multiple RHS evaluations and can become too expensive computationally, especially when the RHS evaluation is expensive. The multi-step methods attempt to circumvent this shortcoming by reusing RHS evaluations from previous time steps. Finally, certain ODEs allow to derive non-standard schemes, which are exact regardless of time step size.

Runge-Kutta Methods

The Runge-Kutta methods are a class of widely used implicit and explicit iterative methods for numerical integration of ODEs. These methods include well known Euler method and higher order schemes, which improve upon it in both accuracy and stability. While these methods achieve high accuracy (especially in combination with high order spatial discretizations – such as spectral methods), they also require multiple evaluations of RHS (and relatively expensive spectral gradients). The following experiments use the classical explicit 4th order Runge-Kutta (RK4) method, which can be written as

$$\begin{aligned} k_1 &= \mathrm{f}\left(t_n, y_n\right), \\ k_2 &= \mathrm{f}\left(t_n + h/2, y_n + hk_1/2\right), \\ k_3 &= \mathrm{f}\left(t_n + h/2, y_n + hk_2/2\right), \\ k_4 &= \mathrm{f}\left(t_n + h, y_n + hk_3\right), \\ y_{n+1} &= y_n + \frac{1}{6}h\left(k_1 + 2k_2 + 2k_3 + k_4\right). \end{aligned} \tag{3.70}$$

Here y_{n+1} is the next time step value computed as a combination of four sub-steps $k_1, \ldots, k_4$. Note that both computational and storage requirements are increased as the right-hand side evaluation is needed in each sub-step (which are also dependent on each other). However, the method offers relatively large stability region (see fig. 3.6) and good accuracy [22].



(a) Forward Euler (b) 4th Order Runge-Kutta

Figure 3.6: Stability regions of Forward Euler (left) and 4th order Runge-Kutta (right) temporal integration schemes.

There are further variants of Runge-Kutta methods, which focus on improving stability (Strong Stability Preserving Runge-Kutta methods [49]) or reducing storage requirements (Low-storage Runge-Kutta schemes [88]).

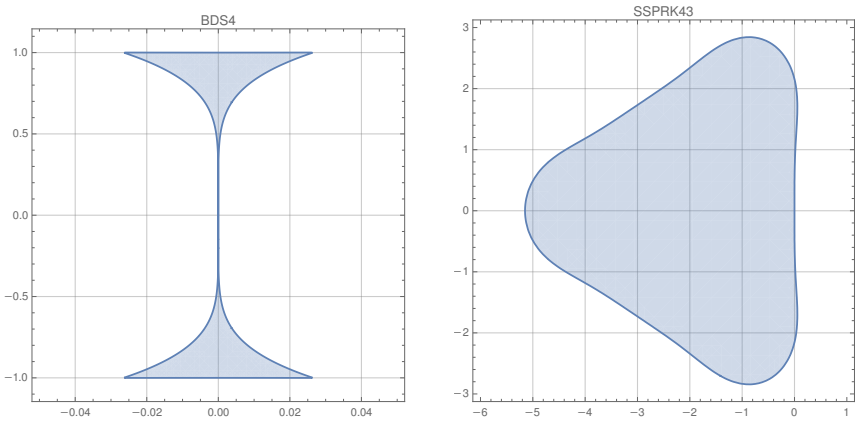
Multi-Step Schemes

The multi-step methods belong to the family of higher-order schemes, however, they take the advantage of the results computed in previous time steps (as opposed to multiple right-hand side evaluations). These methods, therefore, lean even more toward using storage, but avoid additional computation in return.

The fourth-order staggered backward difference (BDS4) method [48] is used as a representative of these methods in further experiments. This method, which requires results of four previous time steps and only a single right-hand side evaluation per step, can be written as

$$y_{n+1} = \frac{12}{11}\mathrm{f}(t_n, y_n) - \frac{17}{22}y_n - \frac{9}{22}y_{n-1} + \frac{5}{22}y_{n-2} - \frac{1}{22}y_{n-3}. \tag{3.71}$$

While stability (see fig. 3.7) and dispersion properties of this methods are not as good as those of high-order methods (such as 4th order RK method), it's especially useful in cases where the evaluation of right-hand side is rather expensive.



(a) 4th Order Backward Differences (b) Strong Stability Preserving RK43

Figure 3.7: Stability regions of 4th order Backward Differences (left) and Strong Stability Preserving Runge-Kutta 4-3 (right) schemes.

Nonstandard Finite-Difference Schemes

Nonstandard finite difference (FD) schemes [83] allow to derive exact finite difference schemes by taking the traditional definition of derivative which can be generalized as

$$\frac{\mathrm{d}y}{\mathrm{d}t} = \lim_{h\to 0} \frac{y(t + \psi_1(h)) - y(t)}{\psi_2(h)}, \tag{3.72}$$

where $\psi_i(h) = h + O(h^2), \quad h \to 0;\ i = 1, 2$. Such a choice of $\psi(h)$ ensures that taking the limit $h \to 0$ recovers the standard FD scheme. However, finite h allows to construct a large class of FD schemes including the exact ones for particular differential equations.

Such an exact scheme can be derived by taking advantage of the fact that a given set of N linearly independent functions, it is always possible to construct an N-th order linear difference equation that has these functions as its solutions [81].

Considering a spatially discretized second-order wave equation eq. (3.3), which can be thought of as a system of harmonic oscillators, each described by second-order ODE

$$\frac{\mathrm{d}^2 y}{\mathrm{d}t^2} + \omega^2 y = 0. \tag{3.73}$$

Given two linearly independent solutions to $y^{(1)}(t) = e^{i\omega t}$ and $y^{(2)}(t) = e^{-i\omega t}$ the exact linear difference equation can be derived by solving

$$\begin{vmatrix} y_k & e^{i\omega hk} & e^{-i\omega hk} \\ y_{k+1} & e^{i\omega h(k+1)} & e^{-i\omega h(k+1)} \\ y_{k+2} & e^{i\omega h(k+2)} & e^{-i\omega h(k+2)} \end{vmatrix} = 0, \tag{3.74}$$

which is

$$y_{k+2} - 2\cos(\omega h) y_{k+1} + y_k = 0, \tag{3.75}$$

where k denotes the time-step index and h is time-step size. The standard central difference form can be recovered from eq. (3.75) by shifting the step index k by one and rewriting $2\cos(\omega h)$ as $2 - 4\sin^2(\omega h/2)$:

$$\frac{y_{k+1} - 2y_k + y_{k-1}}{\left(\frac{4}{\omega^2}\right)\sin^2\left(\frac{h\omega}{2}\right)} + \omega^2 y_k = 0. \tag{3.76}$$

This approach can be used to derive k-Space pseudospectral method suitable for efficient modeling of acoustic wave propagation in weakly heterogeneous medium [110].

3.4 Comparative Study of Numerical Methods

This section complements the estimated computational complexity of each proposed solver with its numerical performance derived by the analysis of its behavior in two dominant effects contributing to ultrasound wave behavior in tissue. The primary effect is a simple wave propagation in homogeneous medium over long distances, which will be represented by a constant coefficient wave equation eq. (3.10). The inviscid Burgers' equation eq. (3.15) will be used to represent a secondary effect, which is non-linearity observed as a high intensity ultrasound wave propagates through the tissue. The wave equation experiments are used to evaluate numerical dispersion and stability of the numerical scheme, while the Burgers' equation stretches its ability to model non-linear problems.

The following experiments cover most of discussed spatial discretization approaches: Finite Differences Method (FDM), Least Squares Finite Element Method (FEM), Gauss-Legendre Discontinuous Galerkin Methos (DGM), Pseudo Spectral Method (PSM) and k-Space Pseudo Spectral Method (KSP). While time-stepping schemes are limited to: Forward Euler (FE), 4th order Runge-Kutta (RK4), 3rd order Strong Stability Preserving Runge-Kutta (SSPRK43) and 4th order Backward Differences (BD4). The benchmark problems are then disretized by solvers built as an appropriate combination of a spatial and a temporal scheme. This approach allows to minimize the impact of one component of the solver (eg. time-stepping) while an other is being investigated (eg. spatial discretization).

The solvers are characterized using experimental evaluation and step-matrix analysis. The experimental approach allows to compare the performance of the solvers against the analytic solution in specific settings and extract quantitative results, while the step-matrix analysis allows to draw qualitative conclusions about properties such as the stability of the solver. The combination of these two tools should provide sufficient information to draw necessary conclusions.

The experimental evaluation attempts to compare solvers under as equal conditions as possible. To achieve this, the grid resolution of the simulation is reduced in the case

of element methods (GL-DGM and LS-FEM) according to an element order so that the overall number of degrees of freedom remains constant (eg., 4th order GL-DGM would be run with only a quarter of elements compared to the original number of grid-points). This requirement necessitates interpolations to map from equispaced grid-points to element nodes with a non-uniform distribution. These interpolations are performed using Fourier series or trigonometric interpolation, which exhibited relative error on the order of 10^{-12} with the data used in our experiments.

Finally, the step-matrix factorization is used to analyze stability of each solver. The fully discrete linear solver can be rewritten as a single matrix $\mathbf{A}$, which allows to advance the solution $\mathbf{u}_t$ at time t by $n\Delta t$ as

$$\mathbf{u}_{t+n\Delta t} = \mathbf{A}^n \mathbf{u}_t. \tag{3.77}$$

The step-matrix of the nonlinear inviscid Burgers' equation solver depends on the solution at the current time $\mathbf{u}_t$ yielding

$$\mathbf{u}_{t+\Delta t} = \mathbf{A}\left(\mathbf{u}_t\right)\mathbf{u}_t. \tag{3.78}$$

Both $\mathbf{A}$ and $\mathbf{A}\left(\mathbf{u}_t\right)$ are diagonalizable and can be factorized by eigendecomposition as $\mathbf{A} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^{-1}$, where columns of $\mathbf{Q}$ consist of $\mathbf{A}$'s eigenvectors and $\Lambda_{ii} = \lambda_i$ are corresponding eigenvalues. In the linear case, the factorized matrix can be substituted into eq. (3.77) as

$$\mathbf{u}_{t+n\Delta t} = \mathbf{Q}\mathbf{\Lambda}^n\mathbf{Q}^{-1}\mathbf{u}_t \tag{3.79}$$

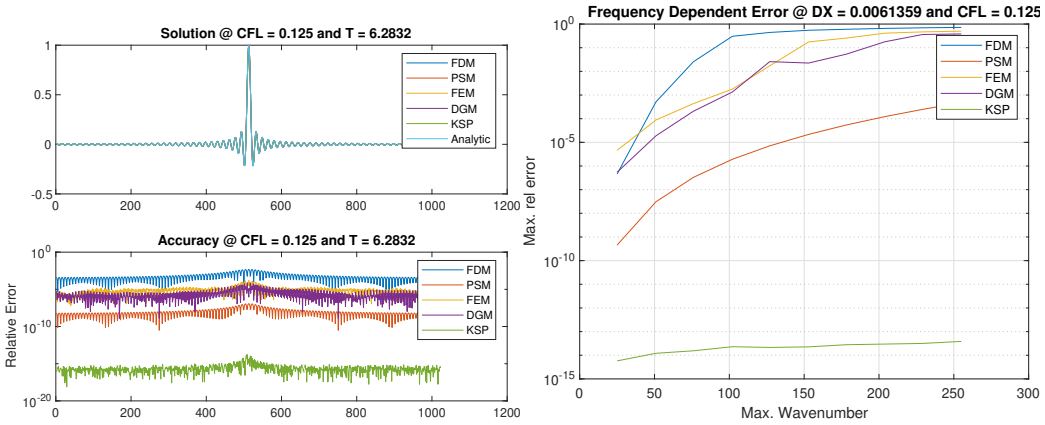
revealing that the stability is effectively determined by the behavior of $\mathbf{\Lambda}^n$ and therefore eigenvalues λ_i of $\mathbf{A}$. This analysis in following sections is performed on small grids (16 to 64 points) to keep eigenvalue plots readable.

3.4.1 Linear Wave Equation

The wave propagation behavior of the solvers is evaluated using lossless wave equation (eq. (3.10)) with constant coefficients in the initial value problem with periodic boundary conditions. The IVP is initialized with a Dirichlet function (or periodic cardinal sinus function)

$$D_N\left(x\right) = \begin{cases} \frac{\sin(Nx/2)}{N\sin(x/2)} & \text{if } x \neq 2\pi k,\ k \in \mathbb{Z} \\ (-1)^{k(N-1)} & \text{if } x = 2\pi k,\ k \in \mathbb{Z} \end{cases}, N \in \mathbb{Z}_{\neq 0}, \tag{3.80}$$

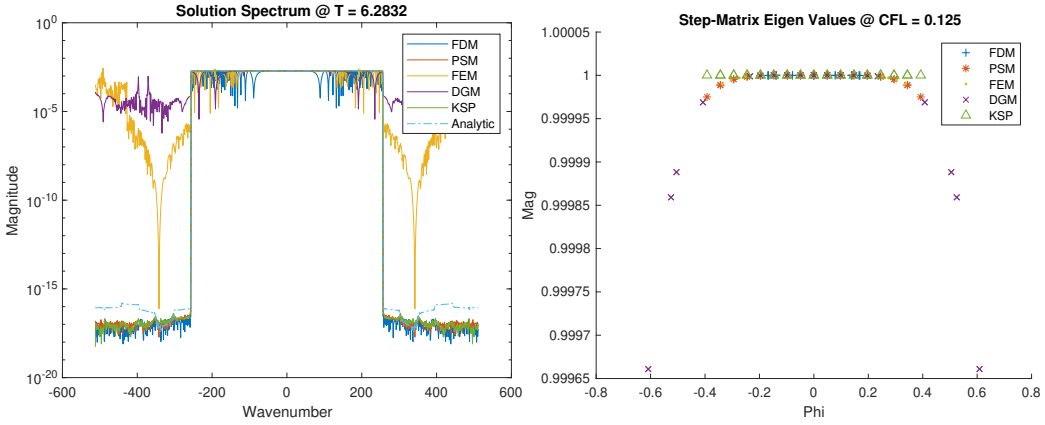
which has a period of 2π for odd N and 4π for even. The convolution of a Dirichlet kernel with any function f of period 2π is the nth-degree Fourier series approximation of f. In other words, the Fourier series of $D_n(x)$ is a window function, which is useful in evaluation of spectral properties of investigated solvers (see fig. 3.9).



(a) Spatial error distribution (b) Frequency error dependency

Figure 3.8: Spatial error distribution (left) and error frequency dependency (right) of periodic wave equation IVP initialized with odd order periodic sinc function (Dirichlet function) after one period. All spatial discretization methods are coupled with 4th order Runge-Kutta time stepping method. The k-space (KSP) solver uses its exact time-stepping scheme.

Figure 3.8 shows a comparison of all five solvers in combination with the 4th order Runge-Kutta time stepping scheme. The temporal resolution was chosen so that CFL of 0.125 is achieved with a spatial grid resolution being 1024 grid points. The number of elements for elements method was reduced to 256 to account for 4th order elements as described above. The initial condition $u(x,t) = D_{128+1}(x)$ (see fig. 3.8a top) represents a periodic pulse with reasonable, but non-trivial frequency content.



(a) Solution frequency spectrum (b) Step-matrix eigen values

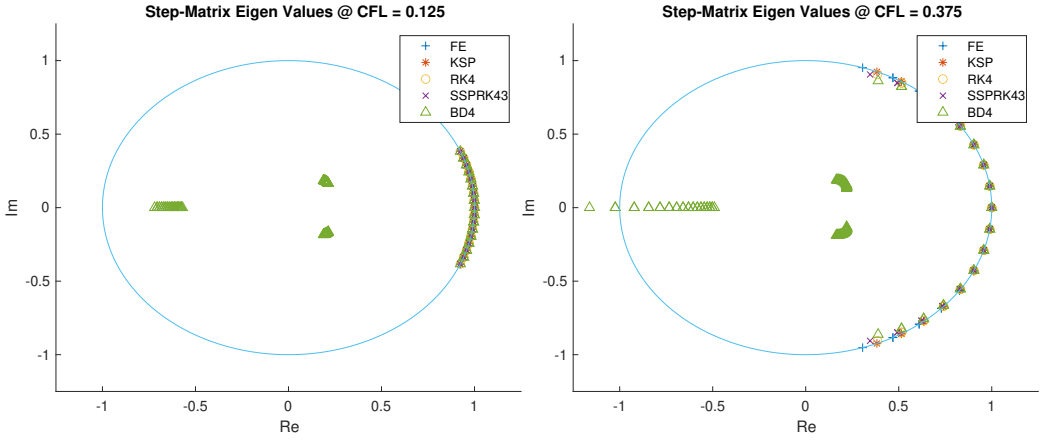
Figure 3.9: Spectral content of the solution to a periodic wave equation IVP initialized with 512+1 order periodic sinc function after one period (left). Eigen values of sub-sampled (16 grid-points) step-matrix of each spatial discretiation. The k-space (KSP) solver uses its exact time-stepping scheme.

The experiment (see fig. 3.8a bottom) confirms that both pseudo-spectral solvers (PSM and KSP) considerably outperform the rest as expected due to their spectral convergence. The advantage of KSP over PSM illustrates advantage of k-space time-stepping scheme,

which is exact in this simple case (wave propagation speed is constant). The 4th order element methods (FEM and DGM) perform similarly to each other while offering only moderate advantage over FDM of the same order. The advantage of the element schemes (especially DGM) would be more significant in the case of heterogeneous medium with sharp interfaces. Figure 3.8b confirms that these observations mostly hold true as higher modes are introduced (by increasing the order of the Dirichlet kernel).

Figure 3.9a offers some more insight into the sources of the error in the previous experiment by analyzing the spectral content of the solution computed by each solver. The solution is expected to maintain the exact spectral content of the IVP, in this case the window function (due to properties of $D_N(x)$) as illustrated by the analytic solution. The majority (except KSP) of the solvers introduce error by attenuation of higher modes. This behavior is especially obvious with FDM solver, but it is a source of the error in PSM solver too. The element methods not only attenuate modes, but also introduce new ones outside of expected window.

These results can be generalized by looking at eigenstructure of the step-matrix of each solver (see fig. 3.9b). The equispaced eigen values with unitary magnitude of KSP method (suggesting no attenuation or instability and uniform wave speed across all modes) can be contrasted with other solvers which exhibit attenuation (PSM) or even non-uniform wave speed.



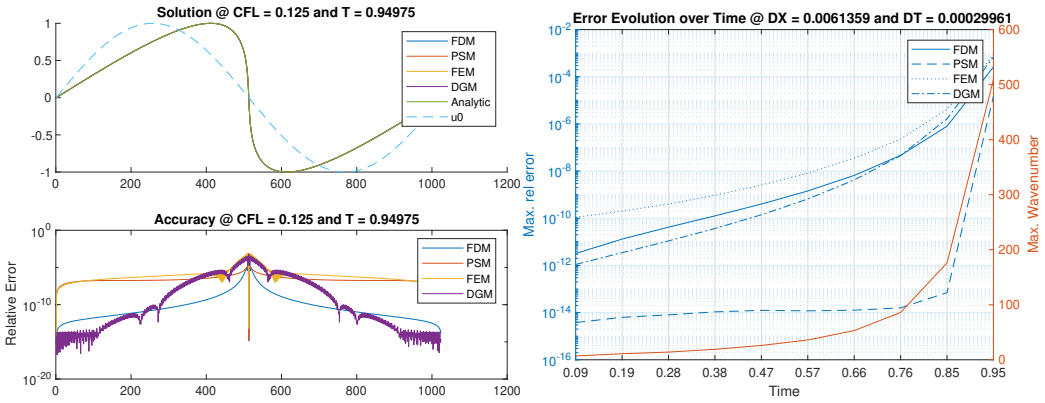
(a) Step-matrix eigen values (BD4 stable) (b) Step-matrix eigen values (BD4 unstable)

Figure 3.10: Eigen values of sub-sampled (16 grid-points) step-matrix of each time-stepping scheme in combination with the pseudo-spectral spatial discretization (PSM) and k-space (KSP) with its exact time-stepping scheme at CFL = 0.125 (left). The 4th order backward differences scheme becomes unstable already at CFL = 0.375 (right).

While the 4th order Runge-Kutta time-stepping scheme used to evaluate spatial discretization methods performs very well, it is rather computationally expensive as it requires four evaluations of equation's right-hand side. The 4th order backward difference scheme (BD4) is able to achieve similar results while utilizing results of previous time-steps – trading storage for computation. The BD4 scheme has a significant disadvantage in that it becomes unstable for relatively low CFLs (see fig. 3.10). The rest of analyzed time-stepping schemes exhibit significant attenuation or non-uniform propagation speed of some modes.

3.4.2 Inviscid Burgers' Equation

While non-linearity has to be included in models capable of describing high-intensity ultrasound propagation, it's typically counteracted by frequency-dependent absorption of the tissue. The inviscid Burgers' equation (eq. (3.15)) therefore represents somewhat extreme or degenerate case, which doesn't take absorption into account. However, it is advantageous for our purpose in two ways: first, its analytical solution is readily available and second, it allows to evaluate behavior of the solvers all the way up to the breaking time of the equation (here $T = 1$).



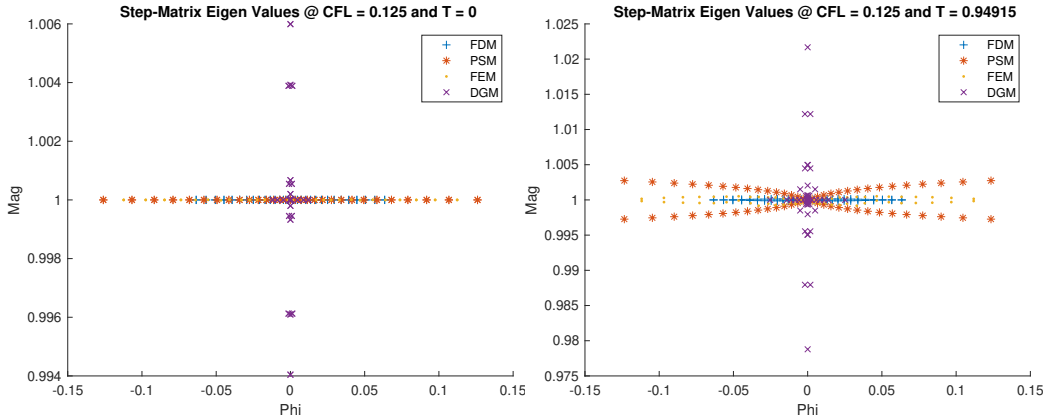
(a) Spatial error distribution

(b) Frequency/Time error dependency

Figure 3.11: Spatial error distribution (left) and time error dependency (right) of inviscid Burgers' equation IVP initialized with a simple sin-wave near $T = 0.95$. All spatial discretization methods are coupled with the 4th order Runge-Kutta time stepping method.

Similarly to the wave equation experiments, the IVP is setup with periodic boundary conditions. However, it is initialized with a simple $u(x, 0) = \sin(x)$ as higher modes develop naturally as time progresses. The simulations are run up to $T = 0.95$ at which point significant aliasing manifests (at chosen resolution of 1024 grid-points). Figure 3.11a shows the initial condition and the solution near (up to time discretization) $T = 0.95$ (top) together with the spatial distribution of the relative error of solutions computed by each solver (bottom). The solvers struggle to capture the forming shock near the center of the domain as its steepness approaches the sampling limit. The FDM and DGM solvers have an advantage over PSM solver in that the large error around the shock is rather well localized, while the global modes of PSM spread it over the whole domain. This can be confirmed by plotting the relationship between the maximum relative error and time (and therefore highest wavenumber in the solution) for each solver (see fig. 3.11b). The PSM solver performs the best up to about $T = 0.85$ at which point aliasing begins to occur as chosen discretization resolution is no longer sufficient.

The thorough stability analysis for inviscid Burgers' equation solvers is considerably more complicated as the equation typically develops a discontinuity in a finite time (breaking time). This property in combination with nonlinearity of the equation limits usefulness of the step-matrix decomposition. However, it is still possible to compare behavior of our solvers at specific time points. Figure 3.12 shows such a comparison at time $T = 0$ (initial condition) and $T = 0.95$ (near breaking time).



(a) Step-matrix eigen values at $T = 0$ (b) Step-matrix eigen values at $T = 0.95$

Figure 3.12: Eigenvalues of sub-sampled (64 grid-points) step matrices around analytical solution at $T = 0$ and $T = 0.95$. Each step-matrix is constructed by discretization of inviscid Burgers' equation using each spatial discretization method combined with 4th order Runge-Kutta time stepping.

3.4.3 Summary

The pseudo-spectral method in combination with 4th order Runge-Kutta time-integration significantly outperformed explored alternatives in both linear and non-linear benchmark problems. The 4th order backward differences time-integration offers attractive alternative to the Runge-Kutta as it allows to reuse right-hand side evaluations. However, it significantly restricts maximum time-step size as it becomes unstable otherwise. The k-Space variant of pseudo-spectral method proved to be the best choice for the wave equation as it's computationally comparable to simple Forward Euler method and achieves analytic accuracy in constant coefficient case.

Both of the most promising alternatives FDM and DGM, which can naturally reduce expensive global communication in the solver, exhibit considerably slower convergence. The lower convergence rate would have to be counteracted by oversampling, which would significantly increase memory requirements and limit maximum problem size.

Overall these results highlight significant advantages of the pseudo-spectral (especially k-Space) approach, which achieved the best results given fixed spatial and temporal sampling. Therefore we will investigate domain decomposition methods which, offer an alternative way to alleviate computational drawbacks of the pseudo-spectral approach.

Chapter 4

Non-linear Ultrasound Wave Propagation

Having concluded in the previous chapter that the combination of k-Space discretization with domain decomposition is the best approach to construction of highly scalable ultrasound wave propagation solver we take a slight detour to overview widely used models and derive solver in a single domain. This solver will be used in the following chapter as a basis to derive suitable domain decomposition approach.

The three models of nonlinear ultrasound propagation in soft tissue that appear to be widespread through medical community are Kuznetsov, Westervelt and KZK – eqs. (4.5) to (4.7) mentioned in the introductory chapter. The fundamental assumption made by these models is that soft tissue can be considered a continuous fluid medium. The continuum approximation relies on the ultrasound wavelength (above 0.1 mm) being orders of magnitude larger than the cell size. While tissue (even soft tissue) would be better described as an elastic solid, the fluid approximation is equivalent to ignoring shear waves, which travel slowly and are strongly absorbed [102].

4.1 Governing Equations

To simplify following equations, we add medium homogeneity to the rest of our assumptions and note that the resulting equations can be generalized to heterogeneous media. This allows to model the medium as a homogeneous thermo-elastic one and acoustic wave motion can be described by the Navier-Stokes system:

$$\begin{aligned}
\partial_t \hat{\rho} + \nabla \cdot (\hat{\rho}\hat{\mathbf{v}}) &= 0, \\
\hat{\rho}\left[\partial_t \hat{\mathbf{v}} + (\hat{\mathbf{v}} \cdot \nabla)\,\hat{\mathbf{v}}\right] &= -\nabla \hat{p} + \eta \Delta \hat{\mathbf{v}} + \left(\zeta + \frac{\eta}{3}\right) \nabla^2 \cdot \hat{\mathbf{v}}, \\
\hat{\rho}\hat{T}\left[\partial_t \hat{S} + (\hat{\mathbf{v}} \cdot \nabla)\,\hat{S}\right] &= \kappa \Delta \hat{T} + \zeta\,(\nabla \cdot \hat{\mathbf{v}})^2 + \frac{\eta}{2}\left(\partial_i \hat{v_j} + \partial_j \hat{v_i} - \frac{2}{3}\nabla \cdot \hat{\mathbf{v}}\delta_{ij}\right)^2, \\
\hat{p} &= p\left(\hat{\rho}, \hat{S}\right),
\end{aligned} \tag{4.1}$$

where density $\hat{\rho}$, velocity $\hat{\mathbf{v}}$, temperature $\hat{T}$ and entropy $\hat{S}$ are unknown functions. The equation of state is denoted by $p\left(\hat{\rho}, \hat{S}\right)$ and η, ζ are constant shear and volume viscosity coefficients respectively. The thermal conductivity is denoted by κ.

Further, expressing unknowns in terms of small perturbations $(\rho, \mathbf{v}, S, T)$ around a stationary solution $(\rho_0, \mathbf{v_0}, S_0, T_0)$ of the system (4.1) and using Galilean transformation to set $\mathbf{v}_0 = 0$. Using an approximate state equation together with assumption of smallness of viscosity, thermal coefficients and perturbations of unknowns in terms of dimensionless parameter $\epsilon > 0$ allows to derive isentropic Navier-Stokes system

$$\begin{aligned} \partial_t \rho + \nabla \cdot (\rho \mathbf{v}) &= 0, \\ \rho \left[\partial_t \mathbf{v} + (\mathbf{v} \cdot \nabla)\, \mathbf{v}\right] &= -\nabla p\,(\rho) + \epsilon \nu \Delta \mathbf{v}, \end{aligned} \tag{4.2}$$

with state equation $p(\hat{\rho}, \hat{S}) = p(\rho) + O(\epsilon^3)$ approximated by a Taylor expansion:

$$p(\rho) = p_0 + c^2\,(\rho - \rho_0) + \frac{(\gamma - 1)\, c^2}{2\rho_0}\,(\rho - \rho_0)^2 \tag{4.3}$$

and a small viscosity coefficient:

$$\epsilon \nu = \beta + \kappa \left(\frac{1}{C_V} - \frac{1}{C_p}\right). \tag{4.4}$$

Here $\beta = (\zeta + (4/3)\eta)$ is a combination of shear and volume viscosity, and $\gamma = C_p/C_V$ is the ratio of heat capacity at constant pressure and at constant volume.

4.1.1 Full-wave Equations

The full-wave models describe full range of wave phenomena including backscattering and multiple reflections in quiescent fluid.

To derive the Kuznetsov equation as an approximation to the isentropic Navier-Stokes system (4.2)–(4.4), we assume that the flow is irrotational ($\nabla \times \mathbf{v} = 0$). This allows to write the equation in terms of the velocity potential $\mathbf{v} = -\nabla u$

$$\partial_t^2 u - c^2 \Delta u = \epsilon \partial_t \left((\nabla u)^2 + \frac{\gamma - 1}{2c^2}\,(\partial_t u)^2 + \frac{\nu}{\rho_0} \Delta u\right) + O(\epsilon^3) \tag{4.5}$$

Here, the left hand side is a familiar second-order wave equation, while the second and third term on the right hand side account for quadratic nonlinearity and thermo-viscous attenuation. Details on this derivation can be found in [31].

The Kuznetsov equation is hard to solve numerically, and therefore, is not widely used in practice, instead it's further approximated by neglecting the local noncumulative nonlinear effects. Such an approximation yields Westervelt equation

$$\partial_t^2 \Pi - c^2 \Delta \Pi = \epsilon \partial_t \left(\frac{\nu}{\rho_0} \Delta \Pi + \frac{\gamma + 1}{2c^2}\,(\partial_t \Pi)^2\right) + O(\epsilon^2), \tag{4.6}$$

where $\Pi = u + (1/(2c^2))\epsilon \partial_t (u^2)$. The importance of neglected local nonlinear effects decreases with propagation distance making this model suitable especially for problems with propagation distance much larger than the wavelength. The classical second-order wave equation can be recovered by neglecting both nonlinear and absorption terms.

4.1.2 One-way Equations

The one-way models are simplest models that can be used to describe transmission through tissue layers with refraction, diffraction, absorption and nonlinear effects. The limitation of these models is that they can no longer capture backscattering and multiple reflections. The most widely used equation of this type is the KZK equation

$$2c\partial_{\tau z}^2\Phi - \frac{\gamma+1}{2c^2}\partial_\tau^2\Phi^2 - \frac{\nu}{\rho_0 c^2}\partial_\tau^3\Phi - c^2\Delta_y\Phi = 0, \tag{4.7}$$

which can be derived from eq. (4.5) by paraxial change of variable, which amounts to taking velocity potential $u(x,t) = \Phi(t - x_1/c, \epsilon x_1, \sqrt{\epsilon}x') = \Phi(\tau, z, y)$. The accuracy of the KZK equation decreases as angle between the main axis and the direction of the traveling wave increases, therefore, it is considered relatively accurate only for angles less than 25° off the nominal axis. Further approximation by discarding the diffraction term yields the Burgers' equation, which can be used to model nonlinear effects and absorption.

4.2 First-order Full-wave Models

In many cases it's more convenient to derive the model of nonlinear acoustic wave propagation in the form of a first order system. Such a system is typically easier to solve than classical equations and allows to easily introduce source terms and integrate PML. Treeby et al. [115] used this approach to derive the model used by widespread ultrasound modeling toolbox – k-Wave. Their approach, which builds upon the equations of fluid mechanics under the assumption of a quiescent, isotropic, and inviscid medium also allows to introduce more accurate model of acoustic absorption in the tissue.

Following Treeby et al. [115], the equations describing nonlinear propagation of acoustic waves through heterogeneous fluid can be derived from mass and momentum conservation equations of fluid mechanics

$$\frac{\partial\hat{\rho}}{\partial t} + \nabla\cdot(\hat{\rho}\hat{\mathbf{u}}) = 0, \tag{4.8a}$$

$$\hat{\rho}\frac{D\hat{\mathbf{u}}}{Dt} + \nabla\hat{p} = 0 \tag{4.8b}$$

where $\hat{\rho} = \rho + \rho_0$, $\hat{\mathbf{u}} = \mathbf{u} + \mathbf{u}_0$ and $\hat{p} = p + p_0$ are total density, particle velocity and pressure, which split into acoustic and ambient components. Plugging these expansions into eq. (4.8) while keeping only the terms up to the second order and assuming zero ambient velocity ($\mathbf{u}_0 = 0$), constant ambient density in time ($\partial\rho_0/\partial t = 0$) and uniform ambient pressure in the domain ($\nabla p_0 = 0$) yields

$$\frac{\partial\rho}{\partial t} + \nabla\cdot(\rho_0\mathbf{u}) = -\nabla\cdot(\rho\mathbf{u}), \tag{4.9a}$$

$$\rho_0\frac{\partial\mathbf{u}}{\partial t} + \nabla p = -\rho\frac{\partial\mathbf{u}}{\partial t} - \frac{1}{2}\rho_0\nabla\left(u^2\right). \tag{4.9b}$$

Here p, ρ and $\mathbf{u}$ are the acoustic pressure, density and particle velocity, $u^2 = \mathbf{u}\cdot\mathbf{u}$ and ρ_0 is ambient density. The system has to be completed with a state equation $\hat{p} = \hat{p}(\hat{\rho}, \hat{s})$, which relates total pressure, density and entropy ($\hat{s}$) variables.

Nonlinear Pressure-density Relation

A typical approach is to use a Taylor series expansion around ambient density and entropy to describe properties of an arbitrary fluid. Treeby assumes that nonlinear effects and entropy changes (absorption) are both second order. Therefore, only up to the second order density terms and the first order entropy terms are kept. The truncated Taylor series describing the change in total pressure over small time step $\delta t = t_1 - t_0$ can be written as

$$\begin{aligned}\hat{p}(t_1) - \hat{p}(t_0) = \left(\frac{\partial \hat{p}}{\partial \hat{\rho}}\right)_{\hat{s}} (\hat{\rho}(t_1) - \hat{\rho}(t_0)) + \frac{1}{2}\left(\frac{\partial^2 \hat{p}}{\partial \hat{\rho}^2}\right)_{\hat{s}} (\hat{\rho}(t_1) - \hat{\rho}(t_0))^2 \\ + \left(\frac{\partial \hat{p}}{\partial \hat{s}}\right)_{\hat{\rho}} (\hat{s}(t_1) - \hat{s}(t_0)).\end{aligned} \tag{4.10}$$

Considering homogeneous medium, the change in total density $\hat{\rho}(t_1) - \hat{\rho}(t_0)$ is caused solely by acoustic perturbations. However, in the heterogeneous case, the change may be also the result of the fluid element displacement to a new position with a different ambient density. Total density $\hat{\rho}$ becomes a function of both position x and time t, its change $\hat{\rho}(t_1) - \hat{\rho}(t_0)$ can be also described by Taylor series expansion as

$$\hat{\rho}(t_1) - \hat{\rho}(t_0) = \left(\frac{\partial \hat{\rho}}{\partial t}\right)_x (t_1 - t_0) + \left(\frac{\partial \hat{\rho}}{\partial x}\right)_t (\xi_1 - \xi_0), \tag{4.11}$$

where ξ_1 and ξ_0 are positions of the fluid element at time points t_1 and t_0. The two terms of the expansion describe acoustic density ρ at a fixed point (assuming no flow in the medium) and the influence of element displacement, respectively. This expansion can be rewritten in a vector form as

$$\hat{\rho}(t_1) - \hat{\rho}(t_0) = \rho + \mathbf{d} \cdot \nabla \rho_0, \tag{4.12}$$

where $\mathbf{d} = \xi_1 - \xi_0$ is a displacement vector and $\nabla \rho_0$ comes from rewriting the spatial derivative of $\hat{\rho}$ at constant time in a vector notation $(\partial \hat{\rho}/\partial x)_t \equiv \nabla \hat{\rho} = \nabla \rho_0$. Assuming the medium is initially in thermodynamic equilibrium, which means that spatial gradients of $\hat{s}$ and $\hat{p}$ are zero, and therefore, $\hat{s}(t_1) - \hat{s}(t_0) = s$ and $\hat{p}(t_1) - \hat{p}(t_0) = p$. Equation (4.10) can then be rewritten as

$$p = c_0^2 (\rho + \mathbf{d} \cdot \nabla \rho_0) + \left(\frac{\partial \hat{p}}{\partial \hat{s}}\right)_{\hat{\rho}} s + \frac{B}{2A} \frac{c_0^2}{\rho_0} \left(\rho^2 + (\mathbf{d} \cdot \nabla \rho_0)^2 + 2\rho \mathbf{d} \cdot \nabla \rho_0\right), \tag{4.13}$$

where $A \equiv \rho_0 (\partial \hat{p}/\partial \hat{\rho})_{\hat{s}} = \rho_0 c_0^2$ (which defines isentropic sound speed c_0), $B \equiv \rho_0^2 (\partial^2 \hat{p}/\partial \hat{\rho}^2)_{\hat{s}}$. Note that the first term contains a linear pressure-density relation for heterogeneous medium, while the last term, weighted by a parameter of nonlinearity (B/A), describes nonlinear effects to the sound speed. The entropy term is energy loss due to acoustic absorption.

Acoustic Absorption

Following the typical approach of modeling soft tissue as thermoviscous medium, the acoustic absorption can be modeled by thermal conductivity and a specific heat capacity using the energy conservation equation. This leads to attenuation exhibiting a squared frequency dependency, $\gamma = 2$ [109], while empirical data show soft tissue to exhibit attenuation in

$\gamma = (0.6, 2)$ range [77]. Treeby therefore implements acoustic absorption using a more general approach in the form of the phenomenological loss term

$$\left(\frac{\partial \hat{\rho}}{\partial \hat{s}}\right)_{\hat{\rho}} s = -\left(\frac{\partial \hat{p}}{\partial \hat{\rho}}\right)_{\hat{s}} \left(\frac{\partial \hat{\rho}}{\partial \hat{s}}\right)_{\hat{\rho}} s \equiv -c_0^2 L\rho, \tag{4.14}$$

where L is a general loss operator. In this case, the absorption follows power law and is modeled by the fractional Laplacian operator of the form

$$L = \tau \frac{\partial}{\partial t}\left(-\nabla^2\right)^{y/2-1} + \eta\left(-\nabla^2\right)^{(y+1)/2-1}. \tag{4.15}$$

For $0 < y < 3$ and $y \neq 1$ the two terms account separately for power law absorption and dispersion with τ and η being respective proportionality coefficients. Substitution of the phenomenological loss term eq. (4.14) into the state equation eq. (4.13) and ignoring higher-order terms yields the following pressure-density relation

$$p = c_0^2\left(\rho + \mathbf{d}\cdot\nabla\rho_0 + \frac{B}{2A}\frac{\rho^2}{\rho_0} - L\rho\right). \tag{4.16}$$

Reduced nonlinear equations

Treeby further simplifies the system of equations (4.9) by considering only cumulative nonlinear effects and assuming that the effect of acoustic heterogeneity is second order, while all higher order terms are discarded. This can be achieved by rewriting second-order terms on the right hand side of both conservation equations in terms of acoustic Lagrangian density, which can be then set to be zero. To this end, the linearized homogeneous acoustic equations are repeatedly substituted into equations (4.9). In the case of mass equation, the series of substitutions is $\rho = p/c_0^2$ and $\nabla\cdot\mathbf{u} = -(1/\rho_0)\partial\rho/\partial t$ followed by $\nabla p = -\rho_0\partial\mathbf{u}/\partial t$ and once more $\rho = p/c_0^2$. Similarly, $\rho = p/c_0^2$ and $\partial\mathbf{u}/\partial t = -\nabla p/\rho_0$ are substituted into the momentum equation. The result is the second-order accurate system of the following form

$$\frac{\partial\rho}{\partial t} + \nabla\cdot(\rho_0\mathbf{u}) = \frac{1}{c_0^2}\frac{\partial\mathcal{L}}{\partial t} + \frac{1}{\rho_0 c_0^4}\frac{\partial p^2}{\partial t}, \tag{4.17a}$$

$$\rho_0\frac{\partial\mathbf{u}}{\partial t} + \nabla p = -\nabla\mathcal{L}, \tag{4.17b}$$

where $\mathcal{L}$ is the second-order Lagrangian density given by

$$\mathcal{L} = \frac{1}{2}\rho_0 u^2 - \frac{p^2}{2\rho_0 c_0^2}. \tag{4.18}$$

Combination of coupled equations (4.17) with the pressure-density relation eq. (4.16) under assumption of zero Lagrangian density ($\mathcal{L} = 0$) and neglecting higher-order absorption terms results in a modified Westervelt equation valid for heterogeneous media with power law absorption

$$\nabla^2 p - \frac{1}{c_0^2}\frac{\partial^2 p}{\partial t^2} - \frac{1}{\rho_0}\nabla\rho_0\cdot\nabla p + \frac{\beta}{\rho_0 c_0^4}\frac{\partial^2 p^2}{\partial t^2} - L\nabla^2 p = 0, \tag{4.19}$$

where $\beta = 1 + B/2A$ is the coefficient of nonlinearity.

4.2.1 Numerical Implementation

Both reduced system (4.17) with pressure-density relation eq. (4.16) and combined equation (4.19) can be directly solved. However, considering spectral methods it's convenient to further modify the governing equations. Following substitutions used to derive equations (4.17), the final term of the mass equation can be rewritten as $-2\rho\nabla\cdot\mathbf{u}$. The system of governing equations then can be written as

$$\frac{\partial\rho}{\partial t} = -(2\rho+\rho_0)\nabla\cdot\mathbf{u} - \mathbf{u}\cdot\nabla\rho_0, \tag{4.20a}$$

$$\frac{\partial\mathbf{u}}{\partial t} = -\frac{1}{\rho_0}\nabla p, \tag{4.20b}$$

$$p = c_0^2\left(\rho + \mathbf{d}\cdot\nabla\rho_0 + \frac{B}{2A}\frac{\rho^2}{\rho_0} - L\rho\right). \tag{4.20c}$$

When these equations are combined or solved as a coupled system terms $\mathbf{u}\cdot\nabla\rho_0$ in mass equation and $\mathbf{d}\cdot\nabla\rho_0$ in pressure-density equation cancel each other out. This allows for additional simplification of discrete equations.

Discrete equations

Discrete equations are derived by solving for particle velocity in the momentum equation using an explicit first-order forward difference method and an implicit first-order forward difference method for the density variable in the mass equation. The acoustic density is also artificially divided into Cartesian components to allow use of an anisotropic PML. Complementing these first-order in time methods with pseudo-spectral method in space and k-space correction results in discrete system

$$\frac{\partial}{\partial\xi}p^n = \mathbb{F}^{-1}\{ik_\xi\kappa\mathbb{F}\{p^n\}\}, \tag{4.21a}$$

$$u_\xi^{n+1} = u_\xi^n - \frac{\Delta t}{\rho_0}\frac{\partial}{\partial\xi}p^n, \tag{4.21b}$$

$$\frac{\partial}{\partial\xi}u_\xi^{n+1} = \mathbb{F}^{-1}\{ik_\xi\kappa\mathbb{F}\{u_\xi^{n+1}\}\}, \tag{4.21c}$$

$$\rho_\xi^{n+1} = \frac{\rho_\xi^n - \Delta t\rho_0\frac{\partial}{\partial\xi}u_\xi^{n+1}}{1+2\Delta t\frac{\partial}{\partial\xi}u_\xi^{n+1}}, \tag{4.21d}$$

$$p^{n+1} = c_0^2\left(\rho^{n+1} + \frac{B}{2A}\frac{1}{\rho_0}\left(\rho^{n+1}\right)^2 - L_d\right). \tag{4.21e}$$

The equations are repeated for each Cartesian dimension $\xi\in\{x,y,z\}$ (in $\mathbb{R}^3$ case), the superscript n and $n+1$ denote current and next time step respectively, and $\rho=\sum_\xi\rho_\xi$ is total acoustic density. Imaginary unit is denoted by i, k_ξ is the wavenumber, Δt is the time step length and κ is the k-space operator (see section 3.3.4 and [110]) defined by

$$\kappa = \operatorname{sinc}\left(c_{\text{ref}}k\Delta t/2\right), \tag{4.22}$$

where $k^2=\sum_\xi k_\xi^2$, and c_{ref} is a reference sound speed used for k-space correction. Finally, the discrete form of loss operator eq. (4.15) (term L_d in eq. (4.21)) can be written as

$$L_d = -\tau\mathbb{F}^{-1}\left\{k^{y-2}\mathbb{F}\left\{\rho_0\sum_\xi\frac{\partial}{\partial\xi}u_\xi^{n+1}\right\}\right\} + \eta\mathbb{F}^{-1}\left\{k^{y-1}\mathbb{F}\left\{\rho^{n+1}\right\}\right\}, \tag{4.23}$$

where Fourier transform of negative fractional Laplacian $\mathbb{F}\{(-\nabla^2)^a\rho\} = k^{2a}\mathbb{F}\{\rho\}$ is used and the acoustic density in the absorption term is replaced by linearized mass equation $\partial\rho/\partial t = -\rho_0\nabla\cdot\mathbf{u}$.

In the k-Wave toolbox, the system (4.21) is further extended by including mass and acoustic velocity sources. The k-space corrected forward differences time integration scheme is further improved by using the leap-frog scheme instead. The acoustic velocity gradients are therefore evaluated at staggered grid points [42] using spectral shifts (see eqs. (6.1) and (6.3)).

$$\frac{\partial}{\partial\xi}p^n = \mathbb{F}^{-1}\{ik_\xi\kappa e^{ik_\xi\Delta\xi/2}\mathbb{F}\{p^n\}\}, \tag{4.24a}$$

$$u_\xi^{n+1/2} = u_\xi^{n-1/2} - \frac{\Delta t}{\rho_0}\frac{\partial}{\partial\xi}p^n + \Delta S_{F_\xi}^n, \tag{4.24b}$$

$$\frac{\partial}{\partial\xi}u_\xi^{n+1/2} = \mathbb{F}^{-1}\{ik_\xi\kappa e^{-ik_\xi\Delta\xi/2}\mathbb{F}\{u_\xi^{n+1/2}\}\}, \tag{4.24c}$$

$$\rho_\xi^{n+1} = \frac{\rho_\xi^n - \Delta t\rho_0\frac{\partial}{\partial\xi}u_\xi^{n+1/2}}{1+2\Delta t\frac{\partial}{\partial\xi}u_\xi^{n+1/2}} + \frac{\Delta t S_{M_\xi}^{n+1/2}}{1+2\Delta t\frac{\partial}{\partial\xi}u_\xi^{n+1/2}}, \tag{4.24d}$$

$$p^{n+1} = c_0^2\left(\rho^{n+1} + \frac{B}{2A}\frac{1}{\rho_0}\left(\rho^{n+1}\right)^2 - L_d\right). \tag{4.24e}$$

$$L_d = -\tau\mathbb{F}^{-1}\left\{k^{y-2}\mathbb{F}\left\{\rho_0\sum_\xi\frac{\partial}{\partial\xi}u_\xi^{n+1/2}\right\}\right\} + \eta\mathbb{F}^{-1}\left\{k^{y-1}\mathbb{F}\left\{\rho^{n+1}\right\}\right\}, \tag{4.25}$$

Similar system of equations can be derived using one of other discretizations investigated in chapter 3.

4.2.2 Computational and Numerical Properties

The system (4.21) restricted to a linear model with homogeneous lossless medium yields a simple wave equation. In this context, the k-space scheme is exact and unconditionally stable (under assumption of periodic domain and smooth initial condition). Therefore, the scheme can be considered optimal in a sense of minimizing both spatial and temporal sampling requirements.

Heterogeneity in the medium properties introduces primary sources of error in the model. The k-space operator is no longer exact in regions, where c_0 and c_{ref} are mismatched. The phase error introduced by k-space operator is significant only for $c_{\text{ref}} \gg c_0$, otherwise the error is always smaller than that of leap-frog scheme with no correction [115]. Such a medium also allows to generate non-smooth fields, which can no longer be accurately represented by a given number of Fourier coefficients [18]. Even here, Treeby shows that only three grid points per wavelength are necessary to achieve error less than 1 %, while first-order and fourth-order finite difference schemes requires 14 and 6 points respectively.

The harmonics generated by nonlinear wave propagation have to be generally accounted for by increasing resolution of the simulation as expected. While it is possible to avoid aliasing by filtering, doing so leads to a significant error as energy in these modes is lost.

Computational Complexity

The direct implementation of system (4.21) in three spatial dimensions results in an algorithm, where each time step consists of per-element (both scalar and complex) operations on 3D arrays interleaved with eight forward and inverse 3D DFTs. Both time complexity and arithmetic intensity of each time step is determined by the 3D FFT algorithm. The overall time complexity is therefore $O(N^3 \log N)$ with arithmetic intensity less than $O(\log N)$, assuming a cube shaped simulation domain with an edge length of N grid points.

Due to the logarithmic factor in the time complexity, the proportion of time spent on 3D DFTs during each time step of the simulation is expected to grow with the domain size. These higher rank DFTs are often implemented by reduction into lower rank problems, which are solved recursively (see FFTW [43]). The straightforward approach is to use traditional FFTs for rank one problems with transpositions so that FFTs are performed on data in an optimal layout, thus introducing data dependencies between dimensions. In practice, a typical shared memory system (a single CPU node or a GPU) can spend about half to two thirds of the time in 3D DFTs.

Distributed Multi-dimensional FFT Algorithms

However, the full impact of the global nature of FFT algorithm shows once a distributed memory machine is used. Our performance evaluation shows that up to 80 % of the time may be spent in the communication between nodes necessary to perform 3D DFTs (see section 6.2.1). The typical approach to computing global higher rank DFTs is to use transposed distributions, where lower rank FFTs are performed only on local data. While this approach avoids complicated data exchanges necessary in distributed 1D FFT, each of the transposition steps between local FFTs require up to $O(P^2)$ messages to be exchanged.

The situation gets progressively worse as the gap between local compute performance and interconnect throughput grows [32]. The high amount of communication is especially prominent in GPU accelerated clusters, where local FFTs can be performed extremely quickly due to high local memory bandwidth of these accelerators. Modern highly optimized libraries for computing 3D FFTs on GPU accelerated clusters such as AccFFT [47] or heFFTe [14] show up to 98 % of time can be spent in the communication routines.

While the development of these high performance FFT libraries is an active research area as new HPC platforms emerge, the fundamental limitation is the inherent amount of data that has to be exchanged between nodes. Significant improvements are made in asynchronous communication, optimization of communication on both node and cluster level and in the area of data decomposition. For example in 3D domain, split between P processors, reshaping a single slab to pencil requires $O(P)$ messages, while brick to pencil only $O(P^{1/3})$ and pencil to pencil $O(P^{1/2})$. The choice of communication scheme helps to take advantage of hardware features of the cluster.

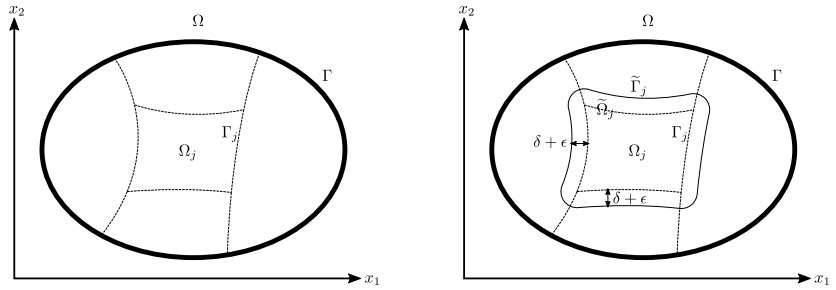
Chapter 5

Domain Decomposition for Pseudo-Spectral Methods

The domain decomposition methods (DDM) attempt to address host of issues that arise in the design process of large scale, parallel and distributed PDE solvers. The applications of DDMs range from preconditioning of large systems of linear equations, which are at the heart of many PDE solvers, through reducing communication in distributed systems to coupling between different discretizations.

Pseudo-spectral discretizations (such as eq. (4.21)) don't necessarily require solution of large linear systems and transforms such as FFT allow for reasonable parallelism, but they do suffer from high amount of communication in distributed systems. The impact of communication is especially pronounced in accelerated clusters as local memory bandwidth and compute capabilities of each accelerator typically evolve at a faster pace than cluster interconnect networks, thus widening the gap. The reduction in the amount of non-local communication is an attractive property of DDMs in our case.

The key concept of DDMs is to recast the original problem over a domain Ω with the boundary Γ as a set of coupled problems defined over subdomains Ω_j, so that $\Omega = \bigcup_{j=1}^{M} \Omega_j$ (see Fig. 5.1a). Such a decomposition creates new boundaries Γ_j at interfaces between subdomains and subdivides boundary Γ into Γ_j^0. Depending on whether a pair of subdomains Ω_i and Ω_j overlap or not ($\Omega_i \cap \Omega_j = \emptyset, i \neq j$), the DDMs can be broadly classified as overlapping or non-overlapping methods.



(a) Non-overlapping Decomposition (b) Overlapping Decomposition

Figure 5.1: General non-overlapping (left) and overlapping (right) domain decomposition in two spatial dimensions.

Subdomain Coupling

Given the decomposition Ω_j, the solution to the original problem over Ω is recovered by a ensuring proper "synchronization" or coupling between subdomains at interfaces Γ_j. The complexity of the coupling process is largely determined by the physics of the problem, mathematical model and its discretization.

Consider an equilibrium process, such as Poisson's equation described by an elliptic PDE, which exhibits instant global interactions. Domain decomposition of such problems often leads to global communication or iterative schemes [45] so that the information can be propagated through all subdomains. Similarly, parabolic equations exhibit non-local behavior, which is however dependent on the temporal discretization [61].

The problem at hand is dominated by the wave equation behavior, which is hyperbolic. The wave equation is well localized as local disturbances travel at a finite speed determined by the medium properties. The situation described by model eq. (4.20) is slightly more complicated by nonlinear effects and non-locality of the fractional Laplacian operator [59] used to accurately describe acoustic absorption in the tissue. However, for our purposes it's sufficient to consider our model well localized.

To illustrate DDMs, we will first look at a simple non-overlapping patching method and continue with an overlapping Schwarz waveform relaxation, and finally, a restricted additive Schwarz method (RAS) to justify our domain decomposition approach.

5.1 Non-overlapping Methods

Non-overlapping methods are directly derived from the idea of domain decomposition into a set of coupled non-overlapping subdomains. The coupling happens exclusively through the boundary conditions of each subdomain – continuity or transmission conditions. Let's consider a simple Poisson equation in a single spatial dimension over the domain Ω with a boundary Γ

$$\begin{aligned} \frac{\partial u}{\partial x} &= f \qquad \text{in } \Omega, \\ u &= 0 \qquad \text{on } \Gamma, \end{aligned} \tag{5.1}$$

and its decomposition into M problems over non-overlapping subdomains Ω_j with boundaries $\Gamma_j \cup \Gamma_j^0$

$$\begin{aligned} \frac{\partial u_j}{\partial x} &= f && \text{in } \Omega_j, \\ u_j &= 0 && \text{on } \Gamma_j^0. \\ u_j &= u_{j+1} && \text{on } \Gamma_j, \\ \frac{\partial u_j}{\partial x} &= \frac{\partial u_{j+1}}{\partial x} && \text{on } \Gamma_j, \end{aligned} \tag{5.2}$$

where last two equations required on boundaries Γ_j between subdomains ensure continuity of solutions u_j defined locally on each subdomain. These continuity conditions also ensure equivalence between eq. (5.1) and eq. (5.2).

5.1.1 Patching

The patching method [90] originally proposed for elliptic problems relies exactly on continuity conditions in eq. (5.2). In higher dimensions, condition on derivatives would be replaced with normal derivatives $\partial u/\partial \vec{n}$.

The matching conditions for hyperbolic problems are more complex as they have to conform to the behavior of the solution to ensure stability of the method. For example, one dimensional advection equation requires an upwinding scheme with one-way information propagation

$$\begin{aligned} \frac{\partial u_j}{\partial t} + \frac{\partial u_j}{\partial x} &= 0 && \text{in } \Omega_j, \\ u_0(0,t) &= u_L(t) && \text{on } (\Gamma_0^0 \times T), \\ u_j(x,t) &= u^0(x) && \text{in } \Omega_j, \\ u_j &= u_{j+1} && \text{on } (\Gamma_j \times T), \\ \frac{\partial u}{\partial x} + \alpha \frac{\partial u_j}{\partial x} + (1-\alpha)\frac{\partial u_{j+1}}{\partial x} &= 0 && \text{on } (\Gamma_j \times T). \end{aligned} \tag{5.3}$$

Here, the last equation describes both averaging ($\alpha = 1/2$), which might be unstable under certain conditions [24] and upwinding ($\alpha = 1$) schemes. The averaging scheme may sometimes be necessary for more complex hyperbolic full-wave systems as upwinding would have to be applied separately to waves traveling in each direction, thus requiring diagonalization of the problem [66].

Discretization and Efficiency

A simple patching method is not sufficient to allow subdomain solutions to be computed independently in parallel. The matching conditions can be included in implicit systems used to solve elliptic and parabolic problems or coupled with explicit time-discretization in the case of hyperbolic problems. The benefit of patching is that the global algebraic systems have a block structure, where only adjacent blocks are coupled.

The key to efficient patching method is a weak subdomain coupling [18], which allows to decouple intra-subdomain computations from subdomain coupling. This is achieved by rewriting the subdomain solution u_j as a sum of particular solution p_j and homogeneous solutions

$$u_j(x) = p_j(x) + U_{j-1} h_{L,j}(x) + U_j h_{R,j}(x), \tag{5.4}$$

where U_j is an unknown value of the solution u at subdomain boundaries. The particular integral p_j is chosen so that it vanishes at both boundaries of the subdomain, while the homogeneous solutions $h_{L,j}$, $h_{R,j}$ are set to one at the left or right boundary, respectively, and zero on the opposite. Writing continuity conditions using the solution in the form eq. (5.4) allows to derive a tridiagonal system which can be easily solved for unknowns U_j. With values of U_j known, subdomain computations can be performed independently in parallel.

5.1.2 Element Methods

Element methods such as FEM are often combined with Finite Element Tearing and Interconnect (FETI) method [37, 38] for domain decomposition or preconditioning as both

approaches originate in variational principles. The method is a parallel interative technique originally devised as a minimization approach for elliptic problems. Each iteration of the method requires two solves on each subdomain followed by flux (normal derivative values on subdomain boundaries) estimation update. The first step is to solve an independent Neumann problem on each subdomain using initial guess of flux, which allows to obtain Dirichlet values on subdomain boundaries. The difference of these values across boundaries is then used as boundary condition of the Dirichlet problem on each subdomain (second solve). Finally, the second solve yields normal derivative values, which are used for correction of the initial guess.

5.1.3 Pseudo-spectral Methods

It is apparent that the key component of non-overlapping methods is the ability to enforce boundary conditions at subdomain boundaries (as this is the only way subdomains are coupled). This poses a significant difficulty for Fourier pseudo-spectral methods as there is no obvious way to impose generic boundary conditions. These methods would have to employ another subdomain basis (such as Chebyshev), which would inevitably lead to a non-uniform node spacing [93], and restrictions on subdomain size and/or time-step length.

5.2 Overlapping Methods

Overlapping domain decompositions originate from Schwarz methods, developed as an analytical tool to prove the Dirichlet's principle [45] developed by Riemann. At the continuous level, there are two main classical Schwarz methods: the original alternating Schwarz method [105] and the parallel Schwarz method [72] developed later for parallel computing. Both methods can be mapped on their discrete counterparts, which will be useful later to show the equivalence between our local Fourier basis approach (derived at discrete level) and continuous Schwarz method.

The concept of non-overlapping decomposition can be easily extended to describe overlapping decomposition by introducing $\widetilde{\Omega}_j$, which denotes enlarged counterpart to subdomain Ω_j so that its new boundary $\widetilde{\Gamma}_j$ is at least $\delta + \epsilon$ far from Γ_j (see fig. 5.1b).

To illustrate these methods, we will focus solely on the parallel Schwarz method and its discrete equivalent, the restricted additive Schwarz method. These methods are typically discussed in relation to elliptic and parabolic problems, and we therefore use a simple Laplace's equation for illustration purposes. The Schwarz waveform relaxation method is then applied to the linear wave equation to establish baseline properties expected in the context of hyperbolic problems.

5.2.1 Restricted Additive Schwarz Method

The restricted additive Schwarz method (RAS) is attributed to accidental modification of additive Schwarz method (AS), which itself can be shown to be equivalent to the parallel Schwarz method under the assumption AS being algebraically non-overlapping. We therefore start at continuous level and work our way to discrete AS and RAS methods.

The decomposition of a simple Laplace's equation over domain Ω with boundary Γ

$$\Delta u = 0 \quad \text{in}\,\Omega, \qquad u = g \quad \text{on}\,\Gamma, \tag{5.5}$$

into two subdomains $\widetilde{\Omega}_1$ and $\widetilde{\Omega}_2$ with boundaries $\widetilde{\Gamma}_1$ using the parallel Schwarz method can be written as

$$\begin{aligned} \Delta u_1^{n+1} &= 0 && \text{in } \widetilde{\Omega}_1, & \Delta u_2^{n+1} &= 0 && \text{in } \widetilde{\Omega}_2, \\ u_1^{n+1} &= u_2^n && \text{on } \widetilde{\Gamma}_1, & u_2^{n+1} &= u_1^n && \text{on } \widetilde{\Gamma}_2, \\ u_1^{n+1} &= g && \text{on } \widetilde{\Gamma}_1 \cap \Gamma, & u_2^{n+1} &= g && \text{on } \widetilde{\Gamma}_2 \cap \Gamma. \end{aligned} \tag{5.6}$$

This iterative scheme is almost identical to the alternating Schwarz method, with the only difference being the boundary values of all subdomains are used from the previous iteration (instead latest results). The elimination of intra-step subdomain dependencies removes the need for coloring schemes necessary for parallel implementations of alternating method. However, the decomposition still has to ensure that there is precisely one neighboring subdomain from which the boundary values can be taken (ie. if $\Omega_i \cap \Omega_j \neq \emptyset$ and $\Omega_i \cap \Omega_k \neq \emptyset$, then $\Omega_j \cap \Omega_k = \emptyset$).

To describe the discrete AS and RAS method the equation (5.5) has to be discretized to obtain a linear system of the form

$$A\mathbf{x} = \mathbf{f}. \tag{5.7}$$

Just like the continuous domain was partitioned into subdomains, the unknowns in the vector $\mathbf{u}$ have to be partitioned into subsets. This is typically represented by restriction operators, which can be written in the matrix form as

$$R_1 = \begin{bmatrix} 1 & & \\ & \ddots & \\ & & 1 \end{bmatrix}, \qquad R_2 = \begin{bmatrix} & 1 & \\ & & \ddots \\ & & & 1 \end{bmatrix}, \tag{5.8}$$

with zeros everywhere else. These matrices allow to recover a particular set of unknowns as $R_i\mathbf{u}$ and similarly to define the restriction of the matrix A as

$$A_i = R_i A R_i^T, \qquad i = 1, 2. \tag{5.9}$$

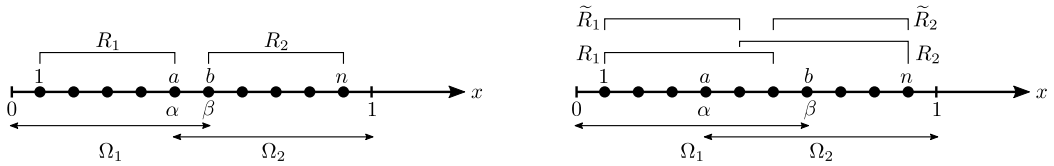
Using these definitions, the AS method can be applied to eq. (5.7) as a preconditioner leading to

$$\left(R_1^T A_1^{-1} R_1 + R_2^T A_2^{-1} R_2\right) A\mathbf{u} = \left(R_1^T A_1^{-1} R_1 + R_2^T A_2^{-1} R_2\right) \mathbf{f}. \tag{5.10}$$

The AS preconditioner can also be used in a stationary iterative method such as

$$\mathbf{u}^{n+1} = \mathbf{u}^n + \left(R_1^T A_1^{-1} R_1 + R_2^T A_2^{-1} R_2\right) \left(\mathbf{f} - A\mathbf{u}^n\right), \tag{5.11}$$

which shows how both sub-problems can be solved in parallel. It can be shown that the AS method is identical to the discretization of parallel Schwarz method as long as R_i are non-overlapping. In fact, the algebraically non-overlapping AS method (see fig. 5.2a) corresponds to the discretization of the parallel Schwarz method with minimal overlap and is also equivalent to the block Jacobi method [100].



(a) Non-overlapping AS

(b) Restricted Overlapping AS

Figure 5.2: Illustration of non-overlapping (left) and restricted overlapping (right) Additive Schwarz decomposition with two domain in a single spatial dimension. The domain decomposition at continuous and discrete level is denoted by Ω_i and R_i respectively.

The behavior of the method significantly changes when R_i actually do overlap as it becomes divergent at least in the overlaps. The overlapping region of both subdomains is now effectively solved twice and the solutions are added together. This issue can be resolved by using a relaxation parameter in the cases of direct AS iteration (such as eq. (5.11)), which allows the method to converge albeit at slower rate [44]. The method is also useful as a preconditioner for a Krylov method [112].

Finally, somewhat accidentally [23] it was discovered that the AS method can be improved by replacing R_i^T in eq. (5.11) with a non-overlapping $\widetilde{R}_i^T$ such that $\widetilde{R}_1^T\widetilde{R}_1+\widetilde{R}_2^T\widetilde{R}_2 = I$ (see fig. 5.2b) yields

$$\mathbf{u}^{n+1} = \mathbf{u}^n + \left(\widetilde{R}_1^T A_1^{-1} R_1 + \widetilde{R}_2^T A_2^{-1} R_2\right)(\mathbf{f} - A\mathbf{u}^n)\,, \tag{5.12}$$

where $\widetilde{R}_i$ eliminates the non-converging modes in the overlap. Such a restricted AS method is now equivalent to a discretization of the parallel Schwarz method even with overlaps between subdomains. Just as other discussed methods, the RAS method can be easily generalized to N subdomains as

$$\mathbf{u}^{n+1} = \mathbf{u}^n + \sum_{i=1}^{N} \widetilde{R}_i^T A_i^{-1} R_i\,(\mathbf{f} - A\mathbf{u}^n)\,. \tag{5.13}$$

To emphasize, the restriction at the continuous level, we define a partition of unity $\chi_i \in C^\infty$ associated with a non-overlapping subdomain Ω_i, so that $\chi_i = 1$ in Ω_i and $\chi_i = 0$ everywhere else, except narrow $(2\epsilon > 0)$ transition layer centered at boundary Γ_i, where χ_i smoothly decays to zero. The global solution in each step u^n then can be recovered as a sum of local solutions on overlapping subdomains $\widetilde{\Omega}_i$ restricted to non-overlapping regions by χ_i. The parallel Schwarz method for Laplace's equation with N subdomains then can be written as

$$\begin{aligned} \Delta u_i^n &= 0 && \text{in } \widetilde{\Omega}_i, \\ u_i^n &= u^{n-1} && \text{on } \widetilde{\Gamma}_i, \qquad u^n = \sum_{i=1}^{N} \chi_i u_i^n, \\ u_i^n &= g && \text{on } \widetilde{\Gamma}_i \cap \Gamma. \end{aligned} \tag{5.14}$$

5.2.2 Schwarz Waveform Relaxation

The classical approach to domain decomposition of time-dependent problems is to discretize the problem in time first, and then apply one of DDMs to stationary problem at each time

step. Such a decomposition requires a uniform time step size over the whole domain, which may be limiting in some scenarios.

The waveform relaxation can be used to describe a space-time domain decomposition, where the domain is likewise subdivided only in space, but interfaces span both space and time. This potentially allows to solve subdomain problems independently over the whole simulation time interval and only then exchange the information on these space-time boundaries. However, typically the simulation time is subdivided into shorter time windows to achieve a faster convergence. In the case of wave equation eq. (3.10) we get:

$$\begin{aligned}
\frac{\partial p_i^n}{\partial t} + c^2 \frac{\partial u_i^n}{\partial x} &= 0, & \frac{\partial u_i^n}{\partial t} + \frac{\partial p_i^n}{\partial x} &= 0 && \text{in } \Omega_i \times (0,T), \\
p_i^n &= p^{n-1}, & u_i^n &= u^{n-1} && \text{on } \Gamma_i \times (0,T), \\
p^n &= \sum_{i=1}^{N} \chi_i p_i^n, & u^n &= \sum_{i=1}^{N} \chi_i u_i^n, && \\
p_i^n(\cdot,0) &= p_0 & u_i^n(\cdot,0) &= u_0 && \text{in } \Omega_i
\end{aligned} \tag{5.15}$$

5.2.3 Convergence of Schwarz Methods

Convergence and stability of overlapping DDMs such as RAS is primarily dependent on the domain of dependency of the problem in question. We have shown that RAS method applied to simple hyperbolic problems such as advection or wave equation can converge in a single iteration of eq. (5.13). This result follows from the finite speed of propagation and in the context of Schwarz methods was proved in [46] for wave equation. The condition to achieve convergence in a single iteration is

$$t < \frac{\delta}{\bar{c}}, \tag{5.16}$$

where t is time, δ is overlap depth and $\bar{c}$ is maximum wave propagation speed.

In the case of k-Wave eq. (4.21), the maximum time step length is severely limited by other parts of the model so that convergence in a single iteration can be achieved even with the minimal overlap.

5.3 Local Fourier Basis Methods

The Local Fourier Basis (LFB) domain decomposition method was introduced in [63, 116] as a combination of patching method with LFB approaches used in signal analysis applications. The resulting method allows to maintain most of the Fourier basis advantages in the domain decomposition setting. Taking the advantage of the local nature of parabolic and hyperbolic problems allows to reduce the global matching procedure eq. (5.4) to only a local matching (see Parabolic Domain Decomposition [62]) thus allowing for good efficiency on distributed systems.

Regardless of matching procedure, the use of the spectral Fourier method in LFB brings an additional challenge as it requires each subdomain solution $f(x)$ to have a smooth periodic extension. If this requirement is not fulfilled, then the Fourier series of the solution will converge very slowly, typically as $f_j \sim \mathcal{O}(1/j)$ as $j \to \infty$ [18]. This problem can be mitigated by extending $f(x)$ to a periodic function $\tilde{f}$ on somewhat a larger interval. The problem of constructing the periodic function $\tilde{f}$ given a non-periodic f is widely known as the "Fourier extension problem" and many ways to solve the problem were proposed [19].

5.3.1 Fourier Extension Problem

The Fourier Extension Problem can be defined (following [19]) as: Given a generally non-periodic function $f(x)$ on a physical interval $[-\chi, \chi]$, define a function $\tilde{f}(x)$ on an extended interval $[-\Theta, \Theta]$ with $\Theta \geq \chi$ so that

$$\tilde{f} = f \quad \forall x \in [-\chi, \chi], \tag{5.17}$$

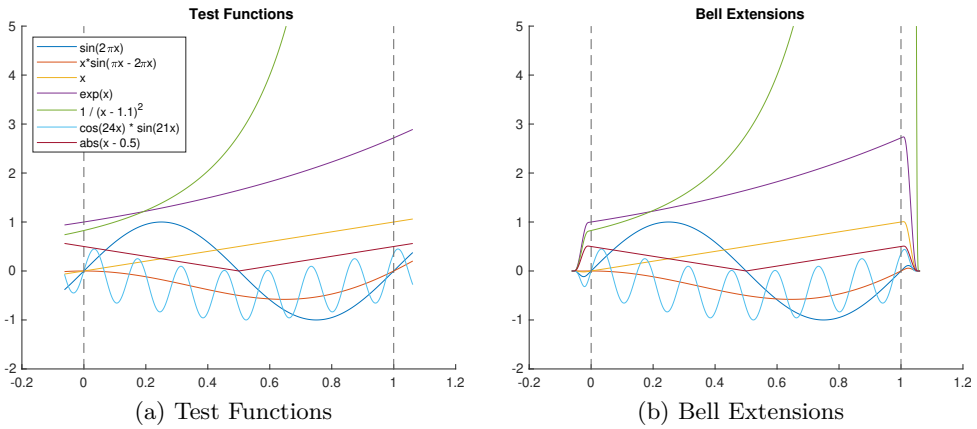
$\tilde{f}$ is periodic with period 2Θ, and $\tilde{f}$ has a rapidly convergent Fourier series. The problem can be further classified into three variants depending on whether the function f is known outside of the physical interval: f is well defined over the whole extended interval (first kind), f has singularities in the extended interval (second kind) and f is known only in the physical interval (third kind). In the context of LFB, we are primarily interested in the first kind as whole function is known (albeit distributed over multiple sub-domains). However, strategies developed to solve the third kind extensions may allow to reduce the amount of communication between sub-domains as we will show later.

Given these requirements, Boyd shows that a good extension $\tilde{f}$ has to match the first k derivatives of f at the endpoints of the physical interval as discontinuity in $(k+1)$st derivative would limit decay of Fourier coefficients of $\tilde{f}$ to $\mathcal{O}(1/j^{k+2})$ at best. The extension also cannot be an analytic continuation of f as it couldn't deviate from f at all and avoid jump at $f(\Theta) \neq f(-\Theta)$. Therefore, the extension can be at most $\mathcal{C}^\infty$ and its coefficients converge only at sub-geometric rate $\mathcal{O}(e^{-pj^r})$ for some $r < 1$.

Analytic extensions with geometric convergence can be achieved by loosening the requirement eq. (5.17) to allow extension $\tilde{f}$ to only approximate f in the physical interval. Such an approximation may be sufficient as the extension is expected to be approximated itself by a truncated Fourier series later.

Fourier Extensions of the First Kind

Extensions of the first and second kind, for which eq. (5.17) holds, are typically constructed by multiplying function f (which is known over whole extended interval) with a suitable bell shaped (window [94]) function, which smoothly transitions from one to zero beyond both ends of the physical interval (see fig. 5.3).



(a) Test Functions (b) Bell Extensions

Figure 5.3: The first kind extensions require the extended function to be known outside of the physical region (left). The periodic extension then can be achieved easily by multiplication of the function with a suitable bell function on the extended region (right).

One example, among many suitable alternatives [94], is a Gauss error function based bell function. Such a function can be constructed by squeezing $\mathrm{erf}(x)$ function into a finite interval

$$\mathcal{E}(x;L) = \begin{cases} -1, & x < -1 \\ \mathrm{erf}\left(L\frac{x}{\sqrt{1-x^2}}\right), & x \in [-1,1] \\ 1, & x > 1 \end{cases}, \tag{5.18}$$

so that its derivatives vanish at the endpoints of the interval. The parameter L describes steepness of the squeezed error function, which has to be scaled and shifted

$$\mathcal{H}(x;L) = \frac{1}{2}\left(1 + \mathcal{E}(x;L)\right), \tag{5.19}$$

into an appropriate range forming a smooth ramp function. Finally, the bell over an extended interval is constructed by taking the ramp function $\mathcal{H}$ and using it to smoothly transition between one and zero on the intervals $x \in [-\Theta, -\chi]$ and $x \in [\chi, \Theta]$:

$$\mathcal{B}(x;L,\chi,\Theta) = \begin{cases} \mathcal{H}\left([x+\chi+\Xi]/\Xi;L\right), & x \in [-\Theta,-\chi] \\ 1, & x \in [-\chi,\chi] \\ \mathcal{H}\left(-[x-\chi-\Xi]/\Xi;L\right), & x \in [\chi,\Theta] \end{cases}, \tag{5.20}$$

where $\Xi = (\Theta - \chi)/2$. While the resulting bell function $\mathcal{B}$ is not analytic at the breakpoints, it is infinitely differentiable for all real x (due to $\mathcal{E}$ being infinitely flat) [19].

Fourier Extensions of Third Kind

There are two main options when it comes to finding a Fourier extension of the function f, when it's only known in the physical interval. The first approach is to use a polynomial extrapolation of f to generate missing information in extension intervals followed by a bell function treatment just like for any other first kind extension problem. Alternatively, a Fourier series F that minimizes a suitable norm $\|F - f\|_X$ over the physical interval and describes approximate $\tilde{f}$ can be found directly.

The Fourier Continuation (FC) - the process of directly finding F can be expressed as a discrete least squares problem

$$F_{N_\Theta}(f) = \underset{g \in \mathcal{G}_{N_\Theta}}{\arg\min} \sum_{x \in P_\chi} \left(f(x) - g(x)\right)^2, \tag{5.21}$$

where F_{N_Θ} is the Fourier series of band-limited function g on the extended interval, which is the best approximation of f over equispaced points P_χ in the physical interval. The least squares problem eq. (5.21) can be written as a dense, rectangular, rank-deficient system

$$\mathbf{Aa} = \mathbf{b}, \qquad \mathbf{A} \in \mathbb{C}^{N_\chi \times N_\Theta}, \qquad \mathbf{b} \in \mathbb{C}^{N_\chi} \tag{5.22}$$

where

$$A_{kj} = \frac{1}{\sqrt{N_\Theta}}, \qquad b_k = f(x_k). \tag{5.23}$$

While the system (5.22) is ill conditioned, there are multiple algorithms (such as FPIC-SU [19] and others [78]) which can take advantage of particular structure of matrix $\mathbf{A}$ and distinct profile of its singular values (see fig. 5.4) to find an approximate solution.

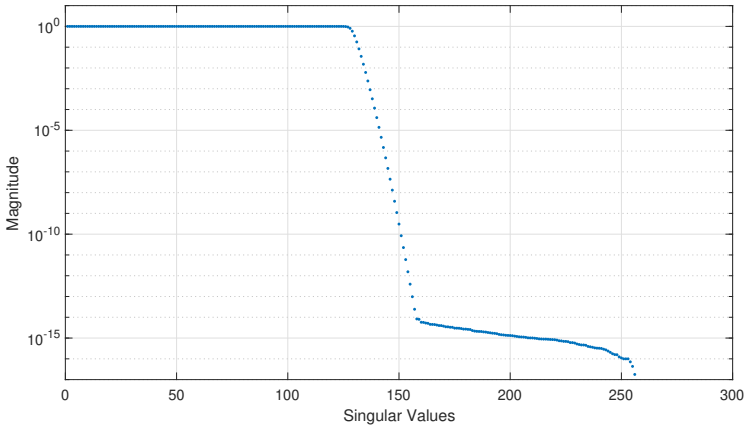
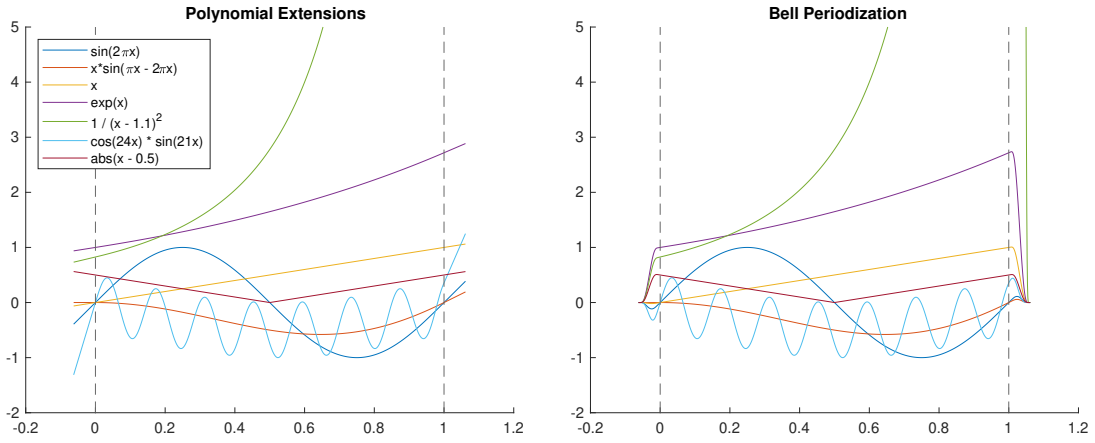


Figure 5.4: Normalized singular values of $\mathbf{A}$ in the Fourier Continuation problem (eq. (5.22)) for $N_\Theta = 2N_\chi$.

The polynomial extension method consists of three steps: smooth extension, windowing and periodization. The smooth extension of function $f \in \mathcal{C}^p$ over interval $[0,1]$ can be formed by using the Taylor polynomial approximation around the interval endpoints

$$\bar{f}(x) = \begin{cases} f(x), & x \in [0,1] \\ \sum_{j=0}^{d} f^{(j)}(0) \frac{x^j}{j!}, & x \in (-\infty, 0) \\ \sum_{j=0}^{d} f^{(j)}(1) \frac{(x-1)^j}{j!}, & x \in (1, \infty) \end{cases}, \tag{5.24}$$

where f is extended by $(d < p)$-th degree Taylor approximation. The compactly supported $\tilde{f}$ can now be formed by windowing $\bar{f}$ just like in the case of the first kind extension problem. However, the windowing function now has to smoothly decay quickly enough to avoid accuracy loss in subsequent FFT caused by potentially rapid growth of $\bar{f}$ in the extrapolation regions (see fig. 5.5).



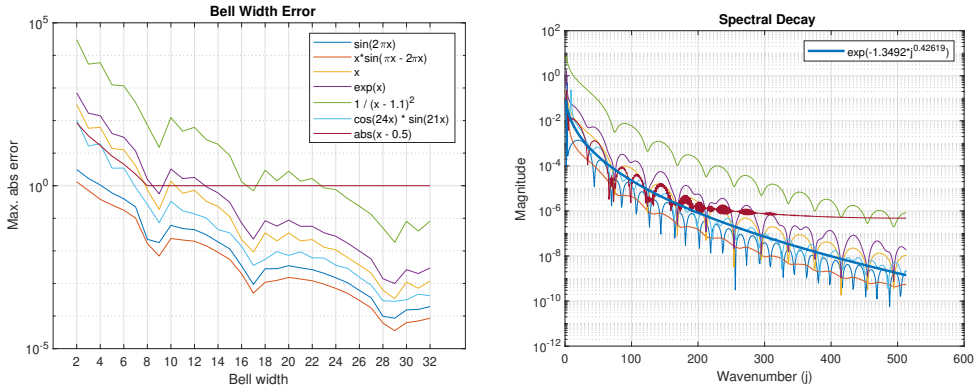
(a) Polynomial Extensions (b) Bell Periodization

Figure 5.5: The third kind extensions using a polynomial extrapolations (left) coupled with periodization based on bell functions (right). The low order polynomial extrapolation (two collocation points) is used for clarity.

This approach is implemented in algorithms such as FC-Gram [21] or its alternative [9], which formally have polynomial convergence. This is to be expected due to a combination of exact approximation of f in the physical interval and extended function $\bar{f}$ being only $\mathcal{C}^d$ smooth (see eq. (5.24)). In practice, the polynomial convergence can be observed near boundaries, while spectral-like behavior is often preserved within the interior. These algorithms are also very fast as extension can be performed in $\mathcal{O}(1)$ time, but with significant single-time data-independent preprocessing (algorithm described in [9] aims to improve upon FC-Gram especially in this area).

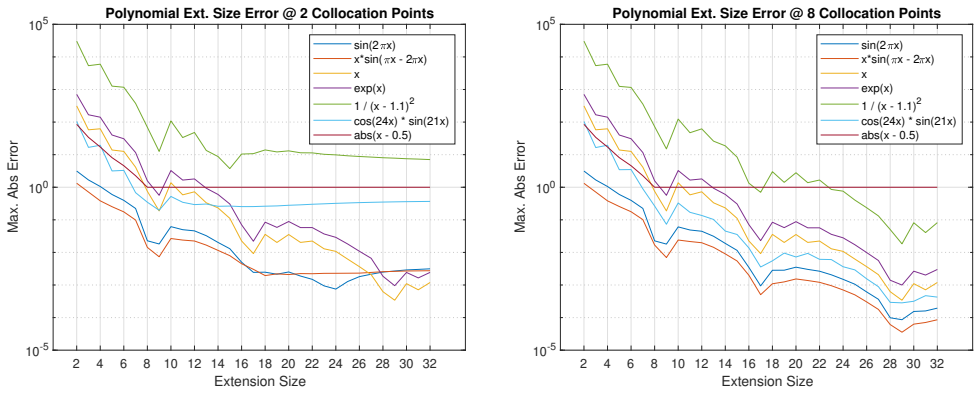
Numerical Properties

Both, the error function bell based first kind extension algorithm and the polynomial extension algorithms for third kind extensions behave similarly following a sub-geometric Fourier coefficient decay rate. Figure 5.6 shows how this decay rate translates into convergence with respect to extension (smoothing) region width.



(a) Bell Width Error (b) Fourier Coefficient Decay

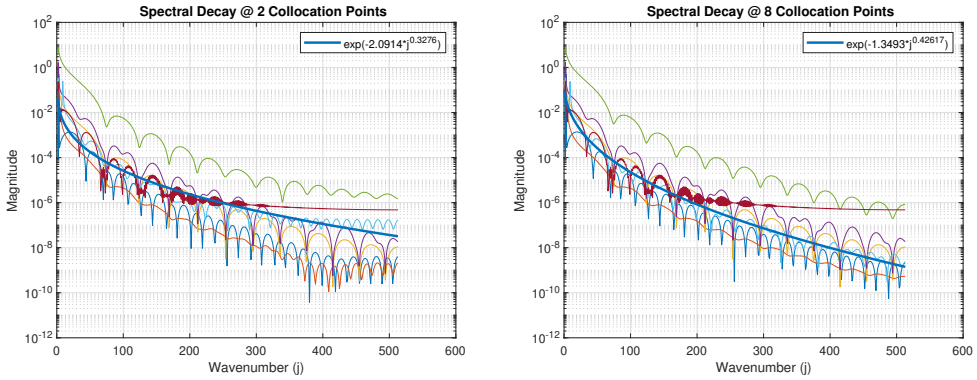
Figure 5.6: Convergence of bell function based first kind extensions in terms of maximum absolute error (left) and Fourier coefficient decay at 32-point overlap width (right).



(a) Two Collocation Points (b) Eight Collocation Points

Figure 5.7: Convergence of polynomial extensions with bell function smoothing using two (left) and eight (right) collocation points for extrapolation. The extension accuracy is significantly dependent on the extrapolation quality, especially for oscillatory functions.

The convergence of polynomial extension algorithms is almost identical (see fig. 5.7) provided a sufficient number of collocation points is chosen. Figure 5.8 compares Fourier coefficient decay achieved with two and eight collocation points. The number of collocation points impacts especially more oscillatory functions such as $f(x) = \cos(24x) * \sin(21x)$.



(a) Two Collocation Points (b) Eight Collocation Points

Figure 5.8: Fourier coefficient decay for third kind extensions based on polynomial extrapolation with two (left) and eight (right) collocation points. In both cases expected sub-geometric coefficient decay rate is observed.

While the polynomial extension algorithm opens the possibility to reduce communication between sub-domains to absolute minimum required by a decomposition method rather than LFB, it is slightly more computationally expensive and may lead to extrapolation issues. We will therefore proceed with first kind extension approach.

5.4 Prototype Equations using LFB Approach

We will use our linear wave equation (eq. (3.10)) and inviscid Burgers' (eq. (3.15)) to illustrate how the first kind Fourier extension might be used to construct a multi-domain PDE solvers in a single spatial dimension. These simple solvers are also useful for evaluating behavior of LFB and its interaction with various time stepping schemes.

Inviscid Burgers' Equation

First, let's consider inviscid Burgers' equation with a pseudo-spectral discretization in space and forward Euler method for time stepping. Such an equation has a hyperbolic behavior, which allows overlapping decomposition with only minimal overlaps as the time step size would be severely restricted by the Euler method. The overlap size is therefore dictated by LFB method, which we use for domain decomposition as:

$$\frac{\partial}{\partial x}{}_j u^n = \mathbb{F}^{-1}\{ik\mathbb{F}\{u^n \mathcal{B}_j\}\}, \tag{5.25a}$$

$${}_j u^{n+1} = {}_j u^n - \Delta t \, {}_j u^n \frac{\partial}{\partial x}{}_j u^n, \tag{5.25b}$$

$$u^n = \sum_j {}_j u^n \Pi_j. \tag{5.25c}$$

Here ${}_j u^n$ denotes overlapping solutions over each subdomain at time step n. Equation (5.25c) describes reconstruction of the global solution u^n by summing non-overlapping parts (masked by rectangular function Π_j) of all ${}_j u^n$. The LFB is completed by restricting global solution to each subdomain in eq. (5.25a) using a suitable bell function $\mathcal{B}_j$. These two steps are usually realized as an overlap exchange between every pair of neighboring subdomains so that the global solution does not need to be explicitly reconstructed.

Linear Wave Equation

The decomposition of the linear wave equation system (3.10) can be handled in a similar manner, with an exception of the time stepping scheme. The forward Euler method has to be replaced with a better time stepping scheme as its unconditionally unstable in this case. We therefore opted for a k-space corrected leap-frog scheme, which allows for an arbitrary time step length. Such a discretization of eq. (3.10) can be written as

$$\frac{\partial}{\partial x} {}_j p^n = \mathbb{F}^{-1}\{ik\kappa e^{ik\Delta x/2}\mathbb{F}\{p^n \mathcal{B}_j\}\}, \tag{5.26a}$$

$${}_j u^{n+1/2} = u^{n-1/2}\mathcal{B}_j - \Delta t \frac{\partial}{\partial x} {}_j p^n, \tag{5.26b}$$

$$\frac{\partial}{\partial x} {}_j u^{n+1/2} = \mathbb{F}^{-1}\{ik\kappa e^{-ik\Delta x/2}\mathbb{F}\{{}_j u^{n+1/2}\}\}, \tag{5.26c}$$

$${}_j p^{n+1} = {}_j p^n - \Delta t c^2 \frac{\partial}{\partial x} {}_j u^{n+1/2}, \tag{5.26d}$$

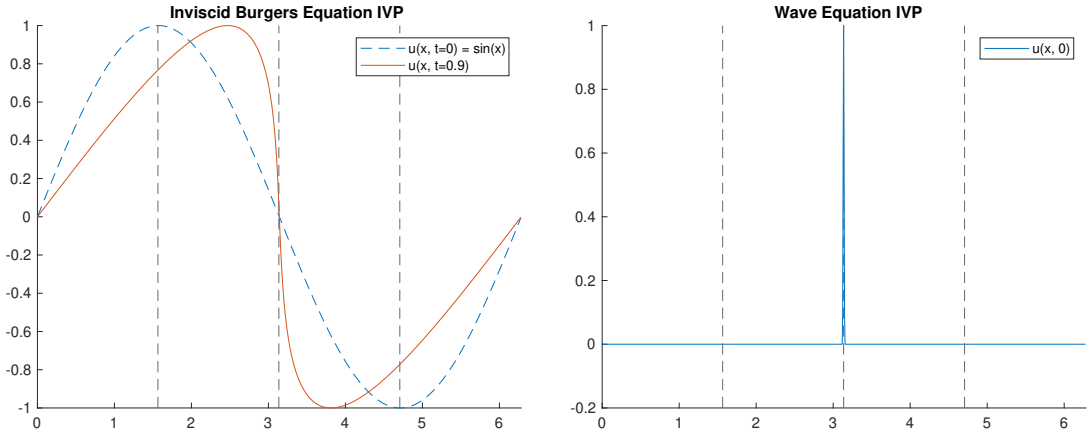
with global solution reconstructed as

$$p^n = \sum_j {}_j p^n \Pi_j, \qquad u^{n-1/2} = \sum_j {}_j u^{n-1/2} \Pi_j. \tag{5.27a}$$

Overlaps in both acoustic pressure p^n and particle velocity u^n are exchanged once per time step as we take strictly the same approach as in the case of the Burgers' equation. However, this means that the second derivative of ${}_j p^n$ (after computing ${}_j u^{n+1/2}$) is computed without a direct synchronization between subdomains. Our numerical experiments (see fig. 5.11b) showed that this leads to a slight loss of accuracy as compared to exchanging ${}_j u^{n+1}$ just before its derivative is taken in eq. (5.26c). The LFB with a direct synchronization of variables just before pseudo-spectral derivatives are taken can be viewed as an approximation to the derivative operator itself.

Numerical Properties

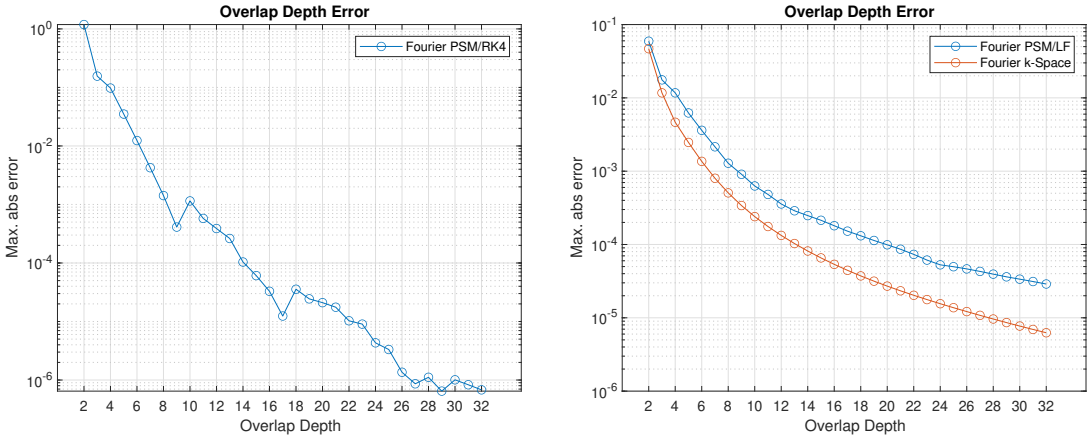
Similarly to our Fourier extension experiments, we discretize both LFB solvers over 1024 grid points in space with periodic boundary conditions. While the Burgers' equation solver is discretized with the combination of a plain PSM with the 4th order Runge-Kutta time integration scheme, the wave equation experiments are also repeated with a k-Space PSM discretization and the Leap-Frog integration scheme.



(a) Inviscid Burgers' Equation IVP (b) Linear Wave Equation IVP

Figure 5.9: Initial Value Problems for evaluation of domain decomposition error. Both domains are decomposed into four subdomains (vertical dashed lines) with overlaps of 2 to 32 grid-points.

The Burgers' IVP is setup with a sin-wave and the final time set so that truncation error does not interfere with our results (see fig. 5.9a). The wave equation solver is initialized with a normalized unit pulse filtered with a Blackmann window and the final time set for exactly one period (see fig. 5.9b). This setup should avoid aliasing while all frequencies are still represented. In both cases the solution is compared to the respective solver with no decomposition. Figure 5.10 show convergence for both LFB solvers as the overlap between sub-domains is increased. These results are largely consistent with our standalone convergence tests of Fourier extensions.

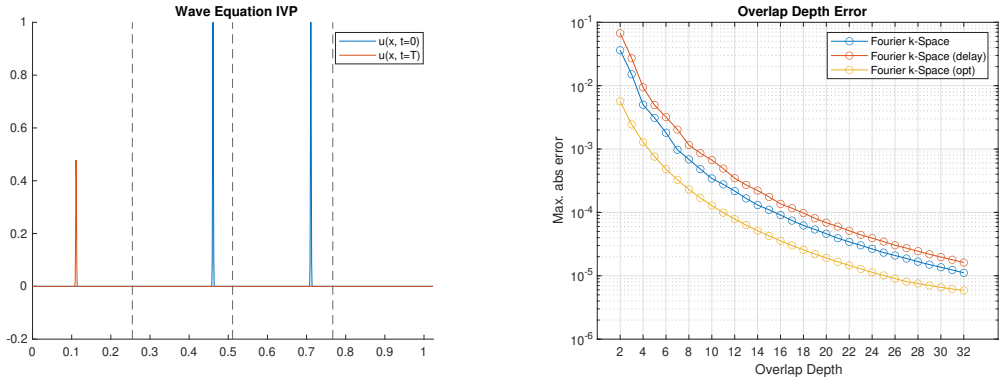


(a) Inviscid Burgers' Equation (b) Linear Wave Equation

Figure 5.10: Local Fourier basis domain decomposition convergence with respect to subdomain overlap depth in problems shown in Figure 5.9. Both solvers use the Error Function based bell to smooth subdomain data.

However, most simulations typically require free boundary rather than periodic one. We therefore setup similar wave equation experiment with addition of the PML layer to

attenuate waves leaving the domain. Figure 5.11a shows IVP and solution at time T at which error is evaluated (only last wave remains in the domain). The PML layer has minimal impact on the solver (see fig. 5.11b) and its qualitative behavior remain virtually unchanged (compared to fig. 5.10b). The figure also shows slight accuracy loss (for all overlap depths) for delayed overlap exchanges approach, which we used previously to demonstrate similarity between LFB and RAS domain decomposition. The final solver therefore uses immediate overlap exchanges to avoid this issue.

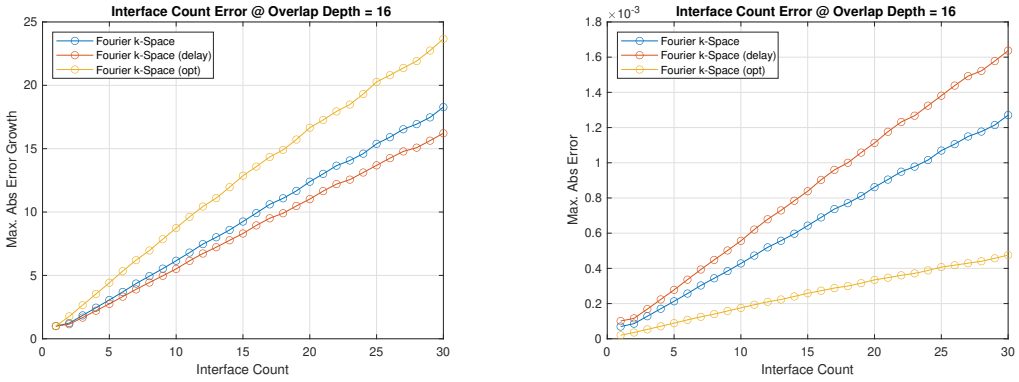


(a) IVP and Domain Decomposition

(b) Overlap Depth Error

Figure 5.11: Overlap depth convergence of LFB k-Space wave equation solver with PML used to implement free boundary. The convergence is shown for the Error Function bell (immediate and delayed overlap exchanges) and for a numerically optimized bell shapes.

The scalability of the LFB domain decomposition is also limited by accumulation of the error as wave passes through interfaces between subdomains. The experiment used to evaluate this property is similar to the previous one. Once again we setup simulation domain with PML and filtered unit pulse as the initial condition. However, the domain is subdivided into 32 subdomains (256 grid points each) and initial pulse is placed in the middle of leftmost subdomain.

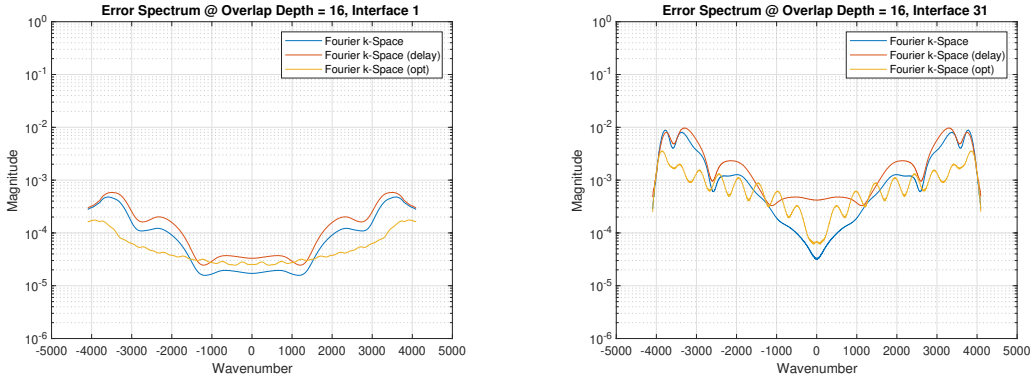


(a) Relative L_∞ Error Growth

(b) Cumulative L_∞ Error

Figure 5.12: Error accumulation along the path of wave propagating through subdomain interfaces in LFB k-Space wave equation solver. The accumulated error is shown for up to 30 interfaces with 16 points deep interfaces with Error Function and optimized bells shapes.

This allows to track half of the pulse as it propagates to the right through all 31 interfaces. The accumulated error (see fig. 5.12b) is averaged over 25 time-steps as the pulse propagates through each subdomain. Figure 5.12a shows that the error accumulates faster when optimized bell shapes is used, however it is still advantageous due to much lower initial error (see fig. 5.12b). The frequency spectrum of the error (see fig. 5.13) shows that the error tends to grow with frequency and that the optimized bell somewhat alleviates this.



(a) Error Spectrum, Interface 1

(b) Error Spectrum, Interface 31

Figure 5.13: Smoothed spectral content of error accumulated along the path of wave propagating through interfaces in LFB k-Space wave equation solver. The overlap depth is set to 16 grid points with Error Function and optimized bell shapes.

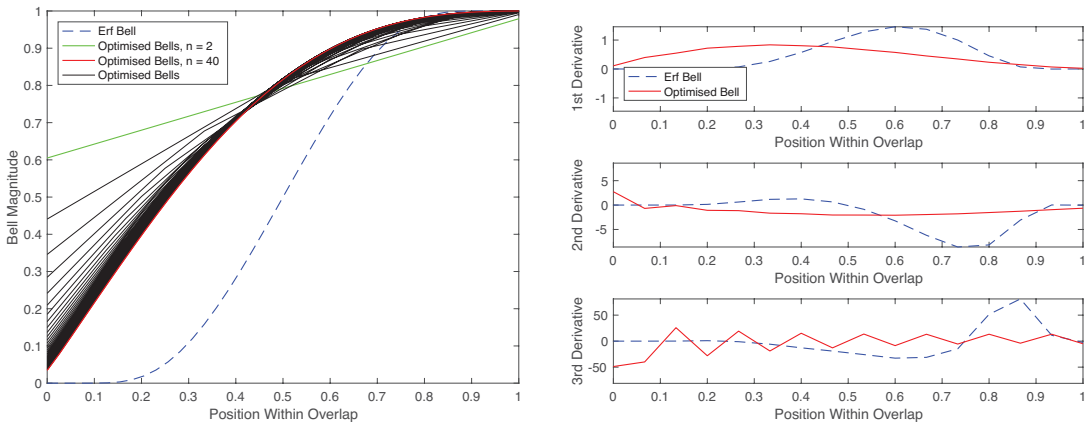
5.4.1 Bell Functions Optimization

As we have seen, the discretely sampled continuous bell performs reasonably well. However, the performance guarantees derived in a continuous setting don't necessarily apply in a discrete setting. This is due to the discrete sampling and an implicit band-limited interpolation of the bell function with the Fourier interpolant.

We therefore explore an alternative approach, where the bell shapes are chosen using numerical optimization. The optimization problem is posed as a minimization of the L_2 error norm between the final solution $u(x, T)$ computed with LFB and global spectral solver. Once again, the solvers were initialized with a delta function filtered with a Blackmann window to cover full frequency range. The LFB solver was using only two subdomains and the final time T was chosen so that the wave crosses the interface only once, so that the single interface crossing error could be estimated. The bell shapes at the interfaces were initialized with a simple ramp function and optimized with a constrained active set method to determine value between 0 and 1 at each grid point so that an optimal bell shape is formed.

This approach closely tailored to our acoustic wave problem resulted in a set of bell functions of 2 to 40 grid points, which can outperform the sampled error function bell (see fig. 5.11b). Figure 5.14 shows that the optimized bell shapes are no longer symmetric and their derivatives vanish only at the end inner to a subdomain. Using such a bell shape also doesn't enforce perfect periodicity of a subdomain. However, these shapes are significantly smoother towards the local subdomain interior, which may allow for better

performance, while sub-optimal shape towards the outer edge does not induce significant error.



(a) Bell Functions

(b) Bell Function Derivatives

Figure 5.14: Optimized bell functions for overlap depths n between 2 and 40 (left). First three derivatives of the erf bell and optimized bells for $n = 16$, weighted by the magnitude of the bell. The optimized bells are significantly smoother towards interior of the subdomain.

Chapter 6

k-Wave Model using Local Fourier Basis

The LFB domain decomposition approach derived in previous chapters can be straightforwardly applied to the full k-Wave model eq. (4.21). In doing so, we arrive at the following model

$$\frac{\partial}{\partial \xi} {}_j p^n = \mathbb{F}^{-1}\{ik_\xi \kappa e^{ik_\xi \Delta\xi/2}\mathbb{F}\{p^n \mathcal{B}_j\}\}, \tag{6.1a}$$

$${}_j u_\xi^{n+1/2} = u_\xi^{n-1/2}\mathcal{B}_j - \frac{\Delta t}{\rho_0}\frac{\partial}{\partial \xi} {}_j p^n + \Delta S_{F_\xi}^n \mathcal{B}_j, \tag{6.1b}$$

$$\frac{\partial}{\partial \xi} {}_j u_\xi^{n+1/2} = \mathbb{F}^{-1}\{ik_\xi \kappa e^{-ik_\xi \Delta\xi/2}\mathbb{F}\{{}_j u_\xi^{n+1/2}\}\}, \tag{6.1c}$$

$${}_j \rho_\xi^{n+1} = \frac{{}_j\rho_\xi^n - \Delta t \rho_0 \frac{\partial}{\partial \xi} {}_j u_\xi^{n+1/2}}{1 + 2\Delta t \frac{\partial}{\partial \xi} {}_j u_\xi^{n+1/2}} + \frac{\Delta t S_{M_\xi}^{n+1/2}\mathcal{B}_j}{1 + 2\Delta t \frac{\partial}{\partial \xi} {}_j u_\xi^{n+1/2}}, \tag{6.1d}$$

$${}_j p^{n+1} = c_0^2 \left({}_j\rho^{n+1} + \frac{B}{2A}\frac{1}{\rho_0}\left({}_j\rho^{n+1}\right)^2 - L_d \right). \tag{6.1e}$$

$${}_j \rho^{n+1} = \sum_\xi {}_j \rho_\xi^{n+1} \tag{6.2}$$

Here, the time-stepping scheme of the model eq. (4.21) is swapped for Leap-frog method, making space-time acoustic velocity u_ξ and pressure p grids staggered by half a grid point relative to each other. Spatial shifts of these fields are implemented in transform space by a spectral operator $e^{\pm ik_\xi \Delta\xi/2}$ at no additional cost. The model also introduces pressure and mass sources as S_{F_ξ} and S_{M_ξ} respectively. Both of those source distributions have to be defined globally so that a subdomain local source can be formed by a bell function $\mathcal{B}_j$ smoothing (and restriction).

$$L_d = -\tau \mathbb{F}^{-1}\left\{ k^{y-2}\mathbb{F}\left\{ \rho_0 \sum_\xi \frac{\partial}{\partial \xi} {}_j u_\xi^{n+1/2} \right\} \right\} + \eta \mathbb{F}^{-1}\left\{ k^{y-1}\mathbb{F}\left\{ {}_j\rho^{n+1} \right\} \right\}, \tag{6.3}$$

$$p^n = \sum_j {}_j p^n \Pi_j, \qquad u_\xi^{n-1/2} = \sum_j {}_j u_\xi^{n-1/2}\Pi_j \tag{6.4}$$

The model is dominated by linear wave behavior captured in equations (6.1a)–(6.1d) and the first term of pressure-density relation eq. (6.1e), which also describes non-linearity and

absorption. The weak diffusivity of the model stems from the acoustic absorption described by eq. (6.3). The acoustic absorption operator remains largely the same as eq. (4.23) with the only change being that it operates over fields local to each subdomain.

Finally, the overlap exchanges of all fields can be performed once before each time step or directly before each forward Fourier transform. Our simplified models did not require the overlap exchange of acoustic density field ρ as there were no spectral operators applied directly to it. However, this is no longer the case with the introduction of the acoustic absorption operator eq. (6.3), which does require a Fourier transform of ${}_j\rho$ in each subdomain. For this reason, we perform additional exchanges of ${}_j\rho$ overlaps to avoid any potential cumulative negative effects, which may be propagated back into the subdomain interior.

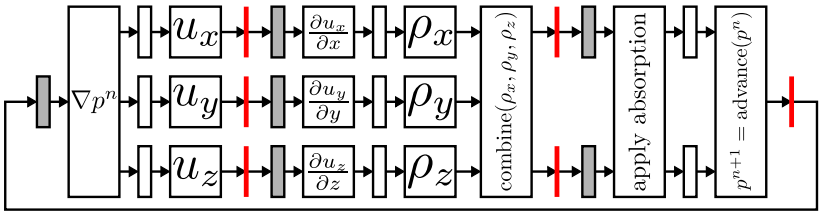


Figure 6.1: Simplified main simulation loop of the k-Wave algorithm. The empty small rectangles denote forward (gray) and backward (white) 3D FFTs. The red bars represent overlap exchanges before each forward FFT.

Figure 6.1 shows a simplified version of the simulation loop in which direct overlap exchanges are used as this approach introduces less decomposition error compared to delayed overlap exchanges (see section 5.4 and fig. 5.11b). In total, six overlap exchanges have to be performed during a time-step. Each of these exchanges involves $3^d - 1$ direct neighbor point-to-point communications depending on dimensionality d of the simulation (see fig. 6.2 for $d = 2$ case).

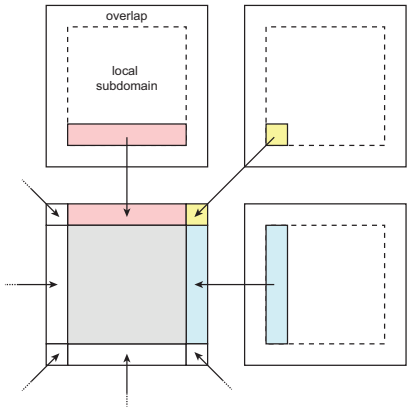


Figure 6.2: Domain decomposition with overlap exchanges in two spatial dimensions.

The size of exchanged data varies according to neighborhood relation between the domains (corner, edge or face in a 3D simulation), and width of the overlap (or smoothing) region. Such a communication pattern in three spatial dimensions requires to exchange $\mathcal{O}(26P)$ messages as each subdomain communicates with other 26 subdomains around it.

This amounts to a significant reduction compared to global approach (analyzed in section 4.2.2) which may require up to $\mathcal{O}(P^2)$ due to global transpositions. The secondary advantage of neighbor-only communication is that it largely eliminates the need for potentially costly communication planning required to achieve efficient scaling of distributed FFT algorithms.

6.1 Implementation

The practical implementation of the simulation package introduced in [1, 3] is based on a widely used MPI plus X (MPI+X) design pattern [96]. The Message Passing Interface (MPI) is used to manage distributed memory (node level) parallelism, while X denotes some complementary technology to manage shared memory parallelism or accelerators.

6.1.1 MPI+X Design Pattern

The typical combination for supercomputer architectures based on multi-core CPU or MIC accelerated nodes is MPI+OpenMP. Similarly in GPU accelerated environments, MPI is combined with suitable (often proprietary) accelerator focused programming interface. In GPU accelerated HPC space, MPI+CUDA is a de facto standard due to proliferation of NVIDIA GPU accelerators. However, MPI+OpenCL can be used to cover most of accelerator vendors. In our case, CUDA or OpenCL is used to manage single accelerator or all accelerators attached to a single shared memory node. The boundary between nodes and accelerators in a single node gets a bit blurry as vendor interfaces such as CUDA often offer faster communication paths.

6.1.2 Simulation Code Overview

Our implementation specifically allows to combine MPI with either OpenMP, CUDA or OpenCL to cover the widest possible gamut of platforms. This allows to address hardware ranging from single multi-GPU nodes such as Nvidia DGX-2 up to clusters with multi-GPU accelerated nodes with CUDA or OpenCL support. While MPI+OpenMP combination allows to fully utilize parallelism of CPU-only and MIC accelerated clusters.

Main Simulation Loop

The simulation loop of k-Wave can be considered a canonical representative of PSTD methods. Considering a global method running on a single shared-memory machine, the majority of computational time is spent in simple per-element kernels and 3D FFTs used to transition between spaces. These 3D FFTs can be usually further decomposed into sets of classic 1D FFTs interleaved with transpositions on 3D arrays. Overall, this means the simulation is an algorithm with a low arithmetic intensity running over relatively large datasets, which leads to poor cache utilization and high memory bandwidth dependence. The most arithmetically intensive part of the simulation loop are therefore 1D FFTs with $\mathcal{O}(\log_2 n)$ intensity [84]. The rest of the simulation loop consists of a streaming I/O used to load source signals at each time step (or at the beginning of the simulation in the case of initial value problems) and store data from virtual sensors in the simulation domain.

In distributed memory environment, we have additional communication component to consider. The communication is either part of the transpositions during 3D FFT stages (global PSTD methods) or explicit overlap exchanges in the case of our LFB approach.

Communication Stack

The communication stack for the overlap exchanges in its complete form (used for CUDA accelerated clusters) consists of three tiers shown in fig. 6.3. Tier 1 employs CUDA Peer-to-Peer (P2P) via PCI-E or NVlink to exchange data within each P2P domain, while tier 2 uses CUDA-Aware MPI to allow fast RDMA communication between GPUs across cluster interconnect network. In this case tier 3 a is fallback where pure MPI is used when CUDA-Aware MPI is not available. The tier at which the exchange of each overlap is performed is decided at runtime as a part of the communication initialization process. This approach allows to minimize communication overhead on the granularity of a single overlap, while covering a wide range of hardware configurations.

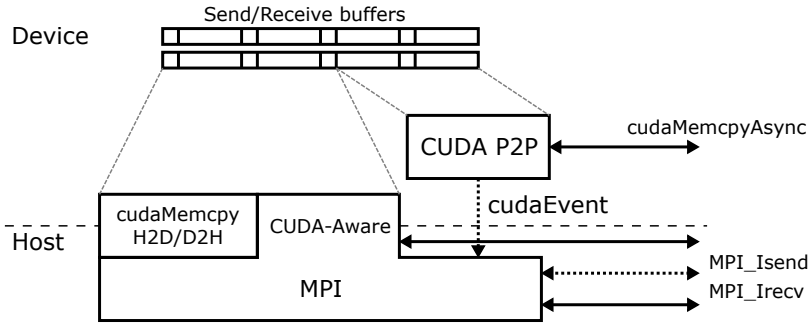


Figure 6.3: Communication stack of k-Wave LFB implementation. The first of five overlaps (two edges and three corners) in send/receive buffers are exchanged through a combination of CUDA memory copy (host ↔ device) and host-to-host MPI communication, or preferably through CUDA-Aware MPI. The last three overlaps can be transferred via CUDA Peer-to-Peer using direct memory copy between devices while the synchronization with the host and the neighbors is achieved through a combination of CUDA Events (locally) and empty MPI messages.

The communication stack is initialized by identifying neighbor MPI ranks for each subdomain using a Cartesian communicator defined by the chosen domain decomposition. With the topology defined, additional storage is allocated for each distributed field to hold both incoming and outgoing overlap data. This approach allows to use a GPU kernel (as opposed to MPI primitive) to extract the overlap data from a 3D array into a linear overlap buffer and reuse the original array during the communication.

Assuming the CUDA P2P support was enabled at the compile time, the second step is to negotiate Peer-to-Peer access between neighbor GPUs. Since every GPU is managed by an MPI process, the direct memory access is enabled by an inter-process communication support provided by CUDA (CUDA IPC memory handles). Effectively, each GPU sends device memory pointers to appropriate parts of its receive buffer to the neighbor GPUs that copy the overlap data in that space. These transfers are managed asynchronously by a secondary CUDA stream so that the other CUDA kernels can be running in parallel to data transfers. If the negotiation fails, the overlap is marked to be handled by the CUDA-Aware MPI or pure MPI transfer with explicit host-device copies. This may happen due to respective GPUs being located at different nodes or lack of the PCI-E or NVlink P2P support.

The last stage is the initialization of persistent asynchronous MPI communications for each overlap. These communication channels are used either as a fallback when P2P access is not available or for synchronization only.

During the simulation loop, the overlaps are synchronously extracted from a given 3D array into the send buffer. Next, they are exchanged using one of the methods described above and the received data injected back into the array. During the asynchronous overlap exchanges, other computations may be performed with the array. This is used to hide some overlap exchanges by computing 3D FFTs during these exchanges. Theoretically, three out of six total overlap exchanges can be hidden, see fig. 6.4.

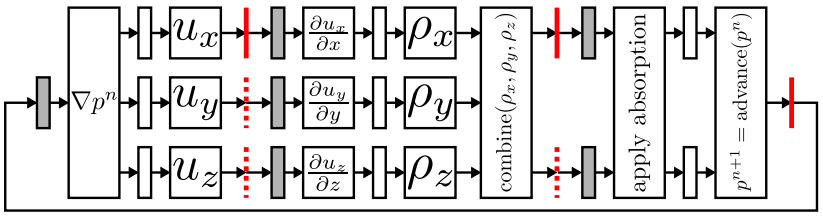


Figure 6.4: Simplified main simulation loop of the k-Wave algorithm. The empty small rectangles denote forward (gray) and backward (white) 3D FFTs. The solid red bars represent overlap exchanges which cannot be overlapped, while dashed ones can be hidden. For example, overlap exchange of u_y can be overlapped with FFT computation on u_x, which was exchanged before.

File I/O Subsystem

The I/O subsystem is built on the top of Parallel HDF5 to maintain compatibility with the rest of k-Wave ecosystem and make efficient use of highly parallel filesystems. The subsystem is responsible for reading simulation parameters such as medium properties, streaming source and virtual sensor data and storage of simulation checkpoints.

The medium reading process is quite straightforward as each subdomain can easily read an appropriate part of the dataset using parallel I/O and possibly exchange overlaps with neighbors where necessary (using overlap exchange routines). The simulation checkpoints can be handled similarly as each working field over the whole simulation domain has to be stored preferably in domain decomposition independent form so that decomposition can be easily changed when resuming the simulation later.

The situation gets more complicated in the case of source and sensor data streaming. The source and sensor points may be arbitrarily distributed through the simulation domain, which is challenging in several ways. Firstly, the distribution of these points is typically clustered to model transducer or sensor surfaces, this leads to unbalanced I/O, where only a few subdomains/nodes participate thus restricting I/O parallelism and achievable bandwidth. Similar issues arise due to sources in subdomain overlaps and off-grid sources [125], which may need to be replicated into both overlapping subdomains. This ties into the second issue, both source and sensor series samples are stored with no regard to domain decomposition, which causes discontinuous access patterns and once again impairs storage bandwidth. Overall these issues are tackled by a combination of caching of the data over multiple time steps and reordering between nodes so that I/O workload is more spread and balanced.

6.2 Experimental Results

In this section, we investigate the performance of the proposed simulation code to validate our theoretical claims about its computational and numerical properties.

6.2.1 Performance Characteristics

The starting point for the assessment of the performance benefit of the proposed code will be the comparison with equivalent traditional global decomposition code based on the MPI FFTW library [64]. While the direct comparison is limited to CPU-based clusters without accelerators, we should already be able to observe a positive impact of reduced communication. From there, we will move to (largely deprecated) MIC accelerated machine to see whether such an architecture is favorable to our approach to pseudo-spectral methods. Finally, GPU accelerated nodes will be added to the comparison as clusters with these high performance nodes further emphasize importance of reducing communication workload in pseudo-spectral solvers.

The majority of our experiments focus on traditional strong and weak scaling as the number of subdomains and spatial resolution are varied relatively to each other. A set of linear wave propagation (which leads to the simplest form of pressure-density relation eq. (6.1e)) initial value or single point source problems is used to collect required data and only the final pressure field is captured after 100 time steps. While 100 step simulation is unrealistically short for real problems, it's sufficient for benchmarking purposes. The spatial resolution of the 3D domain is increased from 256^3 to 4096^3 grid points (or up to the memory limit on smaller clusters) yielding simulation datasets between 2 GB and 7.6 TB. The size of the domain is exponentially increased by doubling the resolution in each Cartesian dimension in a round robin fashion. These experiments are complemented by varying the overlap size between 4 and 16 grid points to get a complete picture of available tradeoffs between accuracy and performance.

CPU Based Clusters

The head to head comparison of the global decomposition (GDD) and our LFB based code (LDD) was carried out using a non-accelerated portion of the Salomon cluster[2]. Each node of the cluster based on dual-socket 12-core Haswell accompanied with 128 GB of main memory. The interconnect is running on 56 Gbit/s FDR Infiniband network in hybrid 7D hypercube topology (see section 2.3 for details).

The strong scaling comparison shown in fig. 6.5 already reveals few interesting properties of the LFB code. Considering a single socket (and a single subdomain in the LFB case), the GDD code is considerably faster than the proposed LFB code. This is due to less optimization work put towards the OpenMP backend of the LFB code as there should be no inherent overhead. The GDD code is also significantly faster on a per-socket basis for small domains (such as 256^3), This is due to a significant computational overhead of local decomposition as each subdomain is extended in each dimension by up to 16 grid points. However, the lead of the GDD code diminishes as the simulation resolution and the local subdomain size with it grows. This amounts to a situation where LFB code has a median speedup of 1.41 over the GDD, but can be up to 4.81 times faster for largest domains (due to limitations of the global decomposition in a single dimension). Figure 6.8 shows how median speedup improves to 2.15 and up to 7.5 for large domains when the overlap size of the LFB simulation is decreased to only 4 grid points.

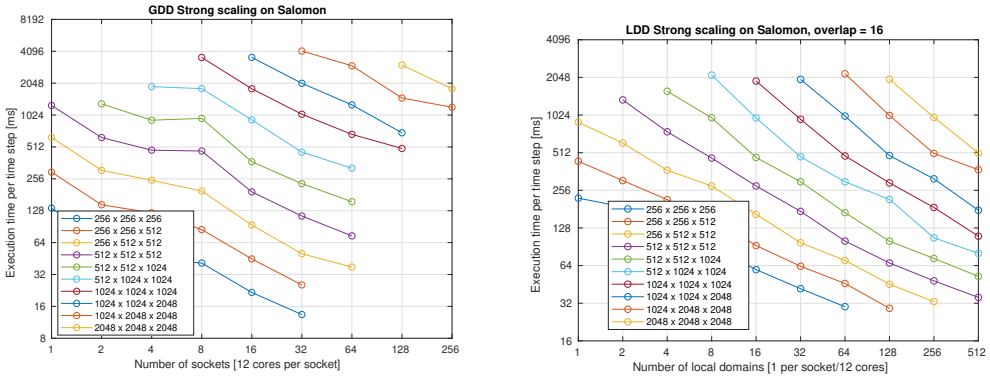


Figure 6.5: Strong scaling comparison of the global domain decomposition (left) and the local Fourier basis (right) approach. The LFB code is configured with an overlap size of 16 and one subdomain per CPU socket.

Moving on to weak scaling plots shown in fig. 6.6, there are two key observations for explanation. First, the weak scaling is very poor for less than 8 subdomains. This is due to an increasing rank of the domain decomposition and increasing number of neighbors which need to exchange data. We start with a single subdomain then split it in last dimension as $1 \times 1 \times 2$, then $1 \times 2 \times 2$ and finally into eight ($2 \times 2 \times 2$) subdomains, at which point each of them has 26 neighbors (due to torus-like wraparound of the domain). Beyond this point, the weak scaling curves of the LFB code are almost completely flat, which is a sign of near perfect weak scaling. This is a result of the constant amount of work and communication per subdomain.

The weak scaling of the GDD approach is much worse as the scaling curves grow on the whole interval. The primary reason for this behavior is the slab decomposition used in the GDD code, where slab spans the whole domain size in the first two dimensions. Considering that, the size of each slab is growing (even with the number of slabs per node decreasing accordingly) the time complexity of per-slab workload grows as $\mathcal{O}(n^2 \log_2 n)$ due to nature of the Fourier transform. The secondary reason is the aforementioned communication complexity, which grows faster in this case.

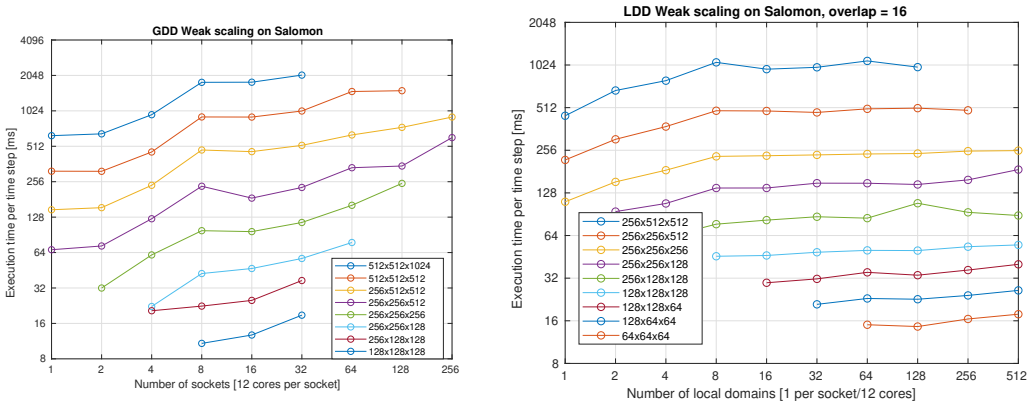


Figure 6.6: Weak scaling comparison of the global domain decomposition (left) and the local Fourier basis (right) approach.

The relative simulation time breakdown shown in fig. 6.7 visualizes the percentage of the overall simulation time spent in computation (Comp) and communication (MPI). In the case of LFB code, additional time is spent on extraction (Extract) and injection (Inject) of the overlaps from/into subdomains. The comparison of GDD and LFB code with 16 point overlaps shows that both codes can spend up to 70 % of the execution time in communication. However, the GDD code hits this point already at 32 sockets, where LFB code wastes only about 40 % of the time. The LFB simulation time breakdown also shows how the amount of work required for overlap handling (extraction and injection) quickly grows between one and eight subdomains due to increasing decomposition rank. This local overhead then stabilizes and is finally overtaken by MPI communication as interconnect usage grows with increasing number of nodes.

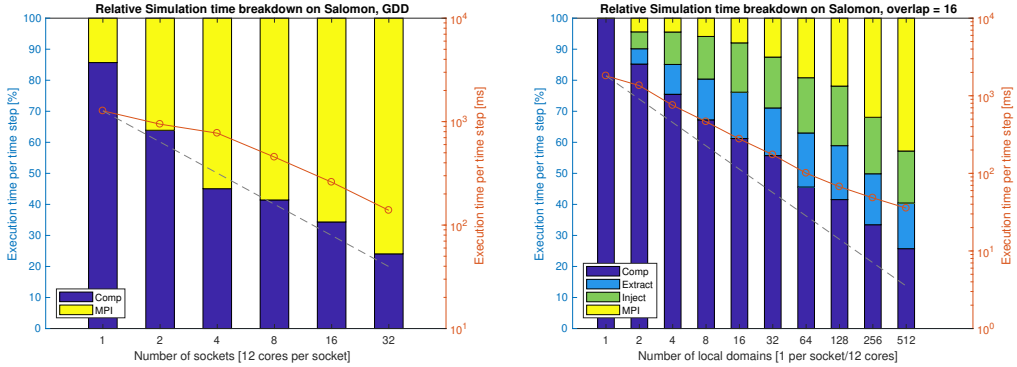


Figure 6.7: Relative simulation time breakdown comparison between GDD (left) and local decomposition (right) with 16 point overlaps on Salomon and a domains size of 1024^3 grid points.

Finally, fig. 6.8 shows the impact of reducing the overlap size to only 4 grid points. With this change, the LFB code is on average 2.63 times faster than GDD code and the average scaling factor (or speedup achieved by doubling computational resources) improves to 1.74.

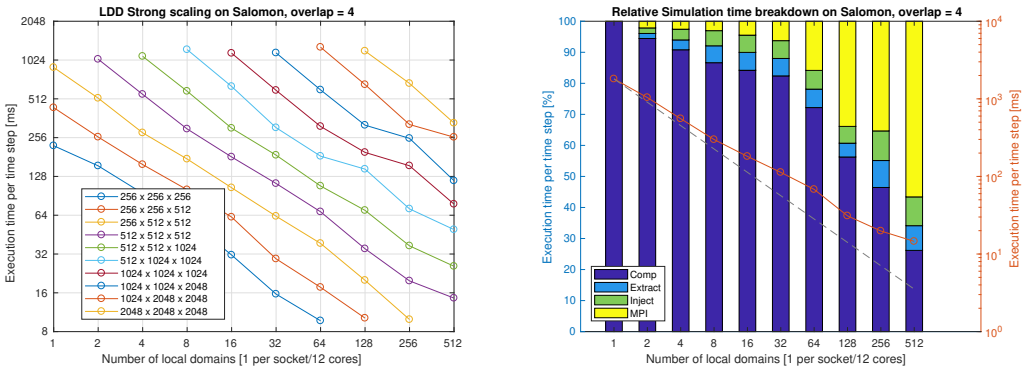
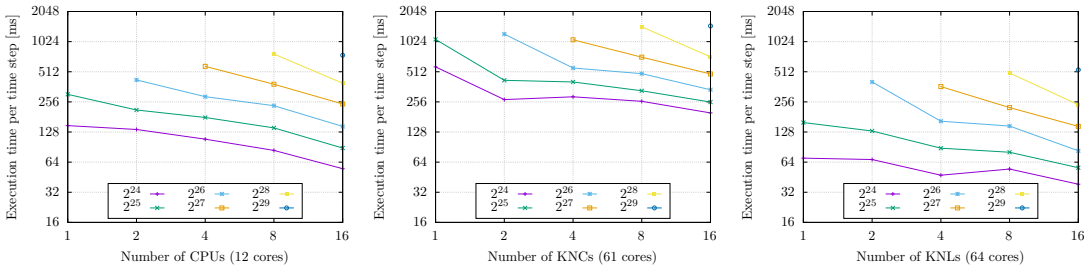


Figure 6.8: Strong scaling (left) and relative simulation time breakdown (right) for LFB code with 4 point overlaps on Salomon and a domains size of 1024^3 grid points.

MIC Accelerated Clusters

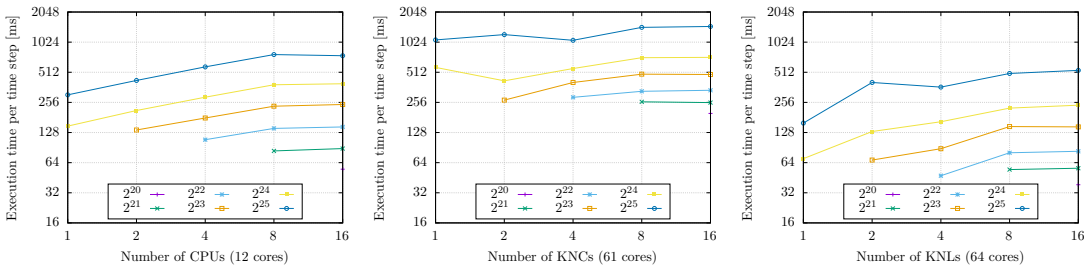
While the LFB approach has considerable advantages when deployed on CPU-based clusters, it's primarily aimed at accelerated clusters. The addition of accelerators significantly increases local computational performance and memory bandwidth of each node, and therefore, skews balance between compute and interconnect bandwidth of the cluster in that direction. In [5, 6] we therefore investigated behavior of the LFB approach on two generations of MIC accelerators.



(a) Haswell CPUs (b) Knight's Corner (c) Knight's Landing

Figure 6.9: Strong scaling with overlap size of 16 grid points.

Similarly to our previous investigation of non-accelerated cluster, we begin with strong and weak scaling by successively increasing the number of subdomains in the decomposition and the resolution of the simulation. Figure 6.9 shows a comparison of strong scaling between Salomons CPUs and Intel Xeon Phi (Knights Corner – KNC, see section 2.1.1 for details) accelerators with domain sizes between $256\times256\times256$ and $1024\times1024\times512$ using 1 to 16 accelerators or CPU sockets. Contrary to the expectations, given parameters of KNC accelerators figs. 6.9a and 6.9b show accelerators between 2.2× to 4.3× slower compared to CPUs. The flat profile in fig. 6.13a shows a couple of reasons for this behavior. First, the overlap exchange among accelerators is on average 2× slower than among CPUs. Such a substantial overhead can be attributed to a combined effect of additional PCI-Express communication and much slower accelerator cores responsible for packing of the overlaps into MPI messages. However, the platform investigation in section 2.1.1 also shows very low core-to-core bandwidth (only up to 2.65 GB/s) on the accelerators. Second, the best performing 3D FFT library optimized for KNC (Intel MKL) still performs up to 100× slower than on CPUs for small subdomains (64^3) and two times slower for large subdomains (such as 256^3).

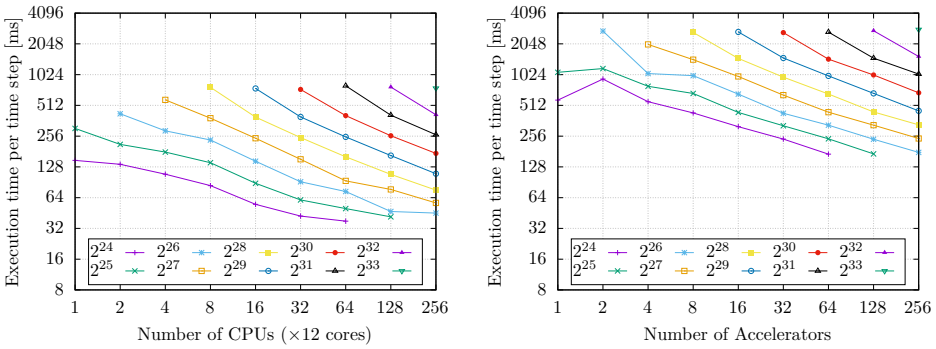


(a) Haswell CPUs (b) Knight's Corner (c) Knight's Landing

Figure 6.10: Weak scaling with overlap size of 16 grid points.

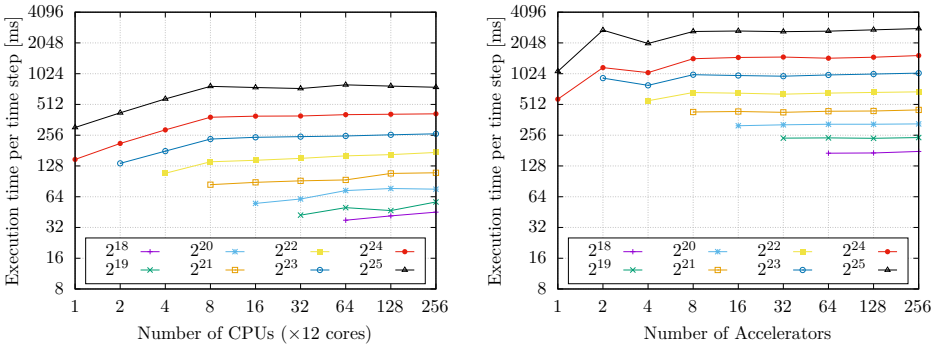
Figures 6.9c and 6.10c show a significant improvement in both strong and weak scaling for KNL accelerators over its predecessor. The single accelerator measurements show up to 8-fold speedup, while KNL is only up to 2.5× faster theoretically. We attribute this significant improvement to a new on-chip interconnect introduced in KNL, which massively increases core-to-core bandwidth. The Omnipath interconnect allows for average scaling factor of 1.62 and average speedup of 4.16 compared to KNC accelerators with the Infiniband on Salomon. The average speedup over Haswell CPUs on Salomon is more modest 1.7 when the same number of accelerators and CPU sockets is considered.

The investigation of larger decompositions had to be limited to 1 Gbit Ethernet interconnect due to stability issues in the Intel MPI Infiniband backend on the Salomon cluster. While this configuration is very unrealistic, it is a useful representative of the worst case from a cluster interconnect perspective. Figures 6.11 and 6.12 show a comparison of strong and weak scaling between CPUs (connected through Infiniband) and KNC accelerators (connected through 1 Gbit Ethernet).



(a) CPUs with FDR Infiniband (b) Accelerators with 1 Gbit Ethernet

Figure 6.11: Strong scaling evaluation on large domains of 2^{24} to 2^{33} grid points with an overlap size of 16 grid points collected on CPUs and accelerators. Since the Infiniband interface is not stable for more than 32 accelerators, 1 Gbit Ethernet is used instead.

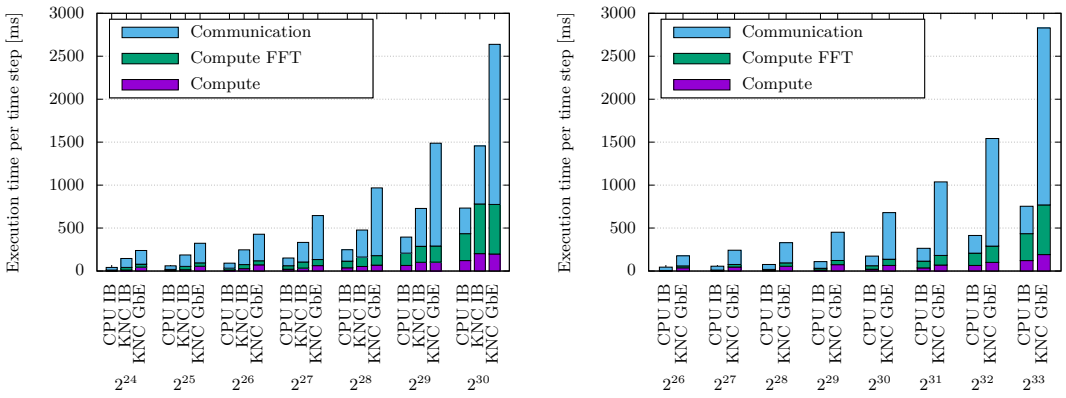


(a) CPUs with FDR Infiniband (b) Accelerators with 1 Gbit Ethernet

Figure 6.12: Weak scaling evaluation on large subdomains with overlap size of 16 grid points collected on CPUs and accelerators. The size of the subdomains ranges from 2^{18} to 2^{25} grid points.

This unfavorable configuration renders KNC accelerators on average 4.3× slower than the same number of CPU sockets, which is another 2-fold drop compared to accelerators with Infiniband interconnect. However, a reasonable strong scaling factor of 1.45 is maintained due to the overall low performance of an individual accelerator which allows to hide much slower communication.

Finally, simulation time breakdowns in fig. 6.13 clearly show communication and 3D FFT computations as areas where KNC accelerators are lacking the most. Both 3D FFT and element-wise computational stages run considerably slower on KNCs, but communication impact is especially pronounced in both Infiniband and 1 Gbit Ethernet case.



(a) 32 CPUs or Accelerators

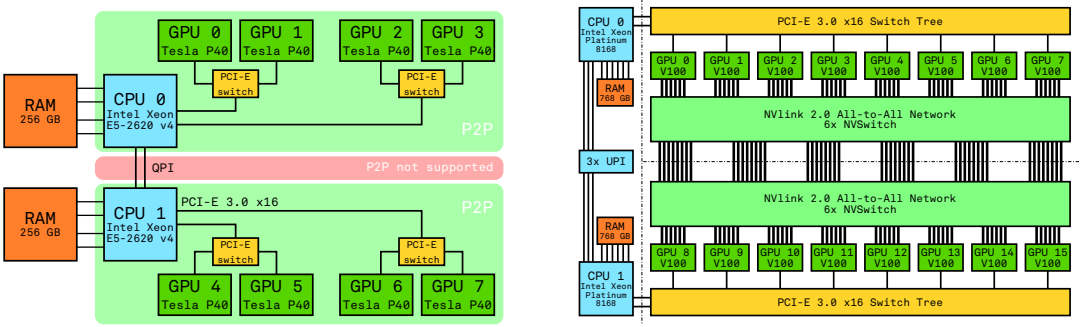
(b) 256 CPUs or Accelerators

Figure 6.13: The execution time breakdown for a time step of the simulation loop collected on 32 and 256 CPUs (CPU) or accelerators (KNC) for a range of domain sizes with an overlap size of 16 grid points. Results for both the Infiniband (IB) and the 1 Gbit Ethernet (GbE) interconnects are shown.

Multi-GPU Accelerated Nodes

Given the excellent results achieved using a single GPU CUDA based implementation of k-Wave, we can reasonably expect a similar success employing LFB code on multi-GPU machines. The limiting factor should be GPU-to-GPU communication and a mild penalty incurred by expanding the size of each subdomain by a specified overlap depth.

We use two multi-GPU machines to investigate the behavior of the LFB code. These machines are chosen to represent the first and second generation of multi-GPU accelerated compute nodes. The first machine (PNY) is a typical server to be found in small research departments. It integrates two Intel Xeon E5-2620 v4 processors and 8 Nvidia Tesla P40 GPUs. The CPUs are coupled with 2× 256 GB of DDR4 2133 memory in a quad-channel configuration. The sockets are connected via two 8 GT/s QPI links. Each CPU provides 40 PCI-E 3.0 lanes, 32 of which are used together with two PCI-E 3.0 x16 switches to communicate with 4 GPUs (two per switch). While these switches are PCI-E Peer-to-Peer (P2P) capable, the QPI socket-to-socket connection is not. This creates two distinct P2P domains, see fig. 6.14a. Each Nvidia Tesla P40 GPU consists of 3840 CUDA cores achieving 11.76 TFLOP/s and provides 24 GB of GDDR5X memory at 480.4 GB/s.



(a) PNY Server (b) Nvidia DGX-2

Figure 6.14: PCI-E 3.0 (a) and NVlink 2.0 (b) based machines used in the experiments.

The second machine (Nvidia DGX-2) represents a super-dense accelerated compute node with a high-speed interconnect between accelerators. DGX-2 has two Intel Xeon Platinum 8168 processors with 2× 768 GB of DDR4 2666 memory in a hexa-channel configuration. The machine contains two 8 GPU boards with all-to-all NVlink 2.0 interconnect at 300 GB/s per GPU. The two boards are connected via 48 NVlink 2.0 lanes achieving a bisection bandwidth of 2.4 TB/s, see fig. 6.14b. Each Nvidia Tesla V100 GPU consists of 5120 CUDA cores achieving 7.8 TFLOP/s and provides 32 GB of HBM 2 memory at 900 GB/s.

The first question which should be answered is whether it's feasible to directly extend single-GPU k-Wave implementation and rely directly on multi-GPU variants of the libraries such as cuFFT. These primitives typically rely on Unified Address Space between host and all GPUs in the system in combination with the runtime to exchange the data between GPUs. As we have shown, the critical part of the algorithm is communication involved in 3D FFT computations. In the absence of global multi-GPU implementation of k-Wave, we use 3D FFT alone to predict the behavior of these two platforms. Figure 6.15 shows that only the second generation multi-GPU machine with NVLink interconnect can be reasonably used for global multi-GPU PSTD codes. The 3D FFT running on a machine using PCI-Express 3.0 x16 interconnect between GPUs still bears a significant penalty and the local approach is therefore necessary.

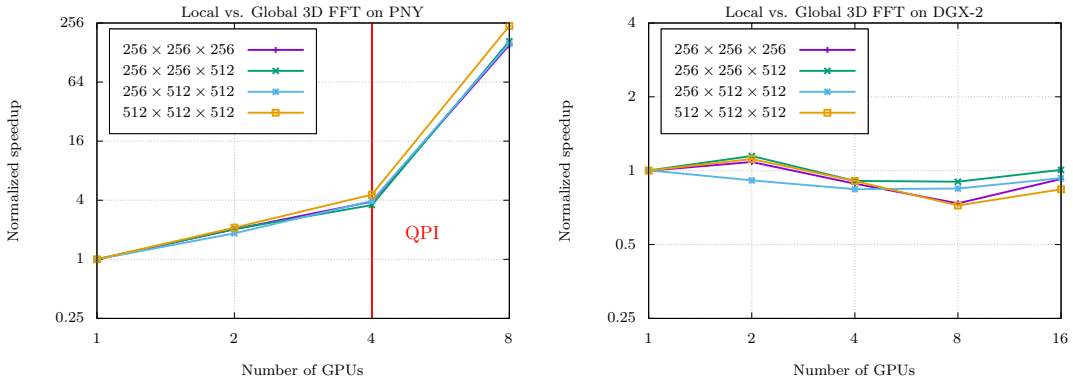


Figure 6.15: Speedup of the local over the global Fourier basis approach on PCI-Express (left) and NVLink (right) machines with 16 grid point subdomain overlaps.

While the local approach is not as much sensitive to bandwidth limitations of PCI-Express 3.0 based platform, it can still be easily spotted in weak scaling behavior and especially in simulation time breakdowns shown in fig. 6.16. The weak scaling on both machines exhibit a typical increasing trend due to the increasing rank of the decomposition. However, PCI-Express platform exhibits a noticeably worse scaling when going from four to eight than between one to four GPUs due to the QPI bridge between two quad GPU clusters. There is no such an impact on DGX-2 machine and weak scaling significantly improves for all sixteen GPUs as the maximum decomposition rank is already achieved.

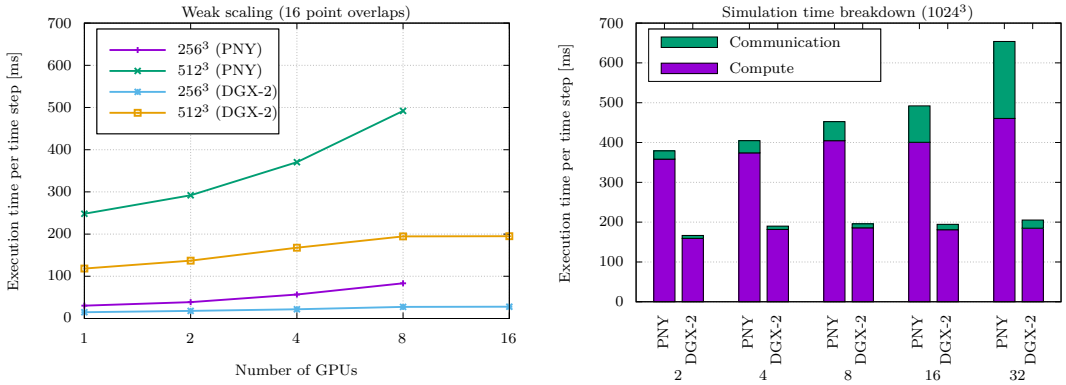


Figure 6.16: Weak scaling with a fixed overlap size of 16 points and subdomain sizes of 256^3 and 512^3 (left), and breakdown of the execution time of a fixed simulation domain size of 1024^3 grid points with a varying overlap size with 8 GPUs (right).

Figure 6.17 confirms this observation by comparing the communication overhead of the same decomposition running on GPUs connected to a single or spread accross both sockets (unless more than four GPUs are used). Although DGX-2 has its GPUs spread across two GPU boards, the interconnect between them is still sufficient and the GPU mapping has no measurable impact.

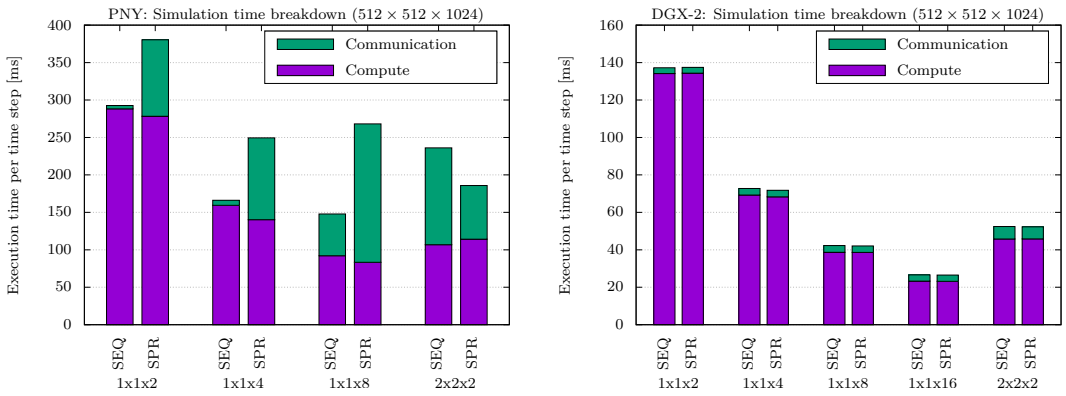


Figure 6.17: Impact of the subdomain mapping onto GPUs in both machines. The sequential mapping (SEQ) maps neighboring subdomains to closest GPUs, while spread (SPR) maps them across QPI domains (PNY) or GPU boards (DGX-2).

GPU Accelerated Clusters

While we showed the LFB approach to bring significant improvements to scalability on CPU based clusters, our primary goal is to enable large scale simulations on GPU accelerated clusters such as Piz Daint[3] and multi-GPU accelerated clusters like Karolina[2]. The LFB approach is necessary in this case as the overhead would be prohibitive otherwise.

First, let's compare the LFB code running on a CPU cluster and a GPU accelerated Piz Daint cluster (see also [4]). Here, the GPU accelerated cluster is on average 2.75× faster with similar scaling factor of 1.44 when comparing CPU sockets and GPUs one-to-one in simulations with 16 point overlaps. The average speedup of the GPU implementation is somewhat skewed by small domains, which exhibit significant overhead (see fig. 6.18). However, on large domains decomposed over many GPUs (e.g. 2048^3 grid points over 512 GPUs) speedups over 5 were observed.

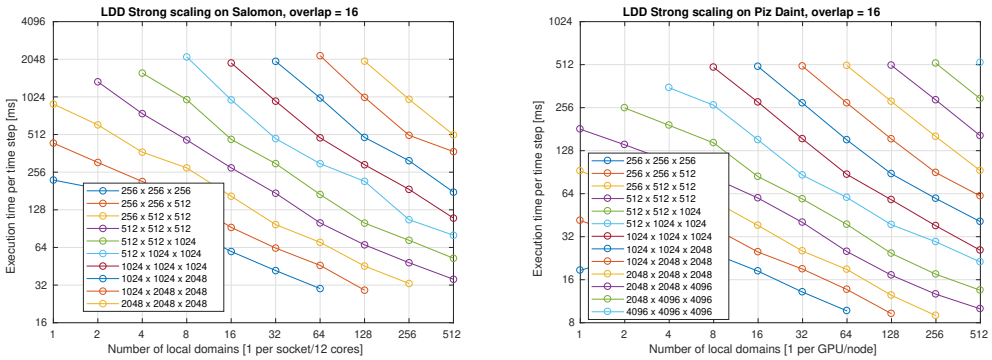


Figure 6.18: Strong scaling of LFB code running on Salomon CPU cluster (left) and Piz Daint GPU cluster (right) with 16-point overlaps between subdomains.

The weak scaling (fig. 6.19) of GPU implementation is inline with the behavior of the CPU code with the exception of a penalty induced by increasing the decomposition rank. While the CPUs experienced about 2× slowdown between one and eight subdomains, the GPUs took twice as large hit (about 4×). This is due to increased communication overhead introduced due to additional communication between CPUs and GPUs as neither CUDA-Aware MPI or RDMA was used on Piz Daint.

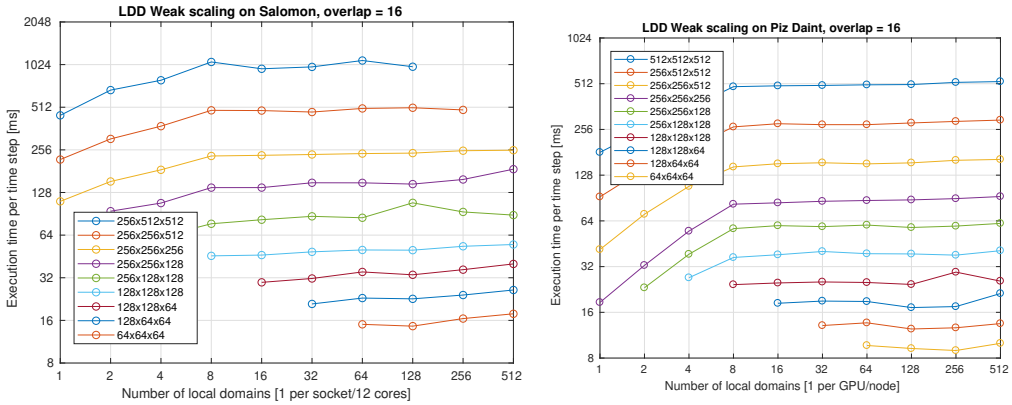


Figure 6.19: Weak scaling of the LFB code running on the Salomon CPU cluster (left) and the Piz Daint GPU cluster (right) with 16-point overlaps between subdomains.

Figures 6.20 and 6.21 show a breakdown of fairly large simulation (1024^3 grid points) using 16 and 4 point overlaps respectively. The overlap depth reduction allows considerable simulation time savings on both architectures, but it has a significantly bigger impact on the CPU cluster as its behavior is more dependent on the communication (up to 70 %). The GPU cluster exhibits a smaller impact as the change in the subdomain size has only a small effect on the compute time and communication seems to be dominated by latency rather than bandwidth.

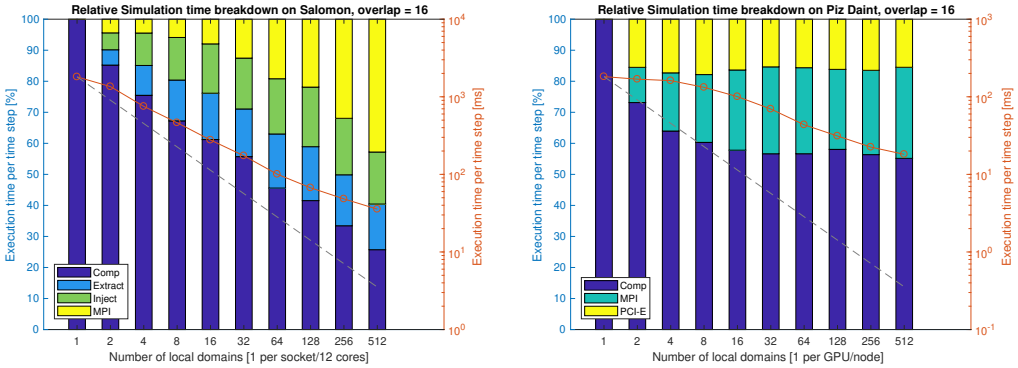


Figure 6.20: The simulation time breakdown for the LFB code with 1024^3 domain uniformly distributed between 1 to 512 subdomains coupled via 16-point overlaps.

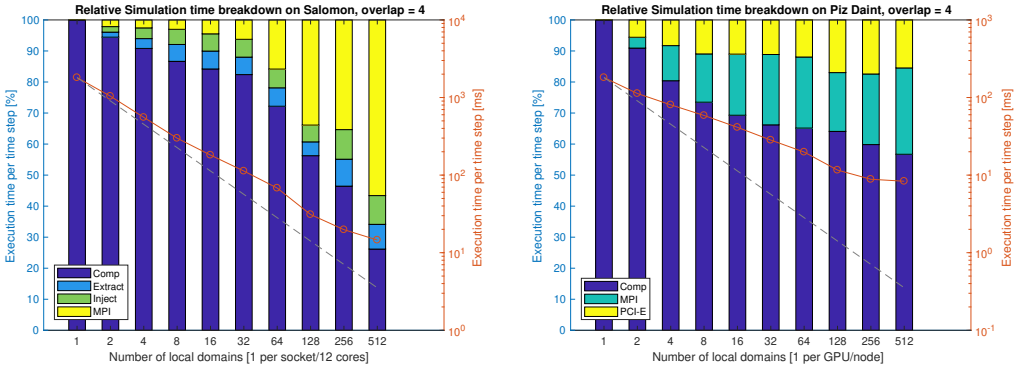


Figure 6.21: The simulation time breakdown for the LFB code with 1024^3 domain uniformly distributed between 1 to 512 subdomains coupled via 4-point overlaps.

6.2.2 Numerical Results

To evaluate both numerical performance and computational efficiency of the proposed LFB approach to ultrasound wave simulation in realistic setting of HIFU treatment planning, we performed a typical kidney sonication simulation [2].

HIFU Kidney Sonication

The simulation of a HIFU sonication of the kidney with a single element bowl-shaped transducer. The transducer radius is 140 mm with circular aperture of 120 mm and frequency of 1 MHz. The transducer is approximated using a simply-connected discrete bowl [76] with a surface intensity of 1 W/cm^2 (175 kPa) and an acoustic power of 119 W.

The simulation domain was discretized in 172 µm resolution, which (given material properties) allows for a maximum supported frequency of 4.4 MHz. Given the simulation volume of 16.5 × 16.5 × 22 cm this resolution yields a simulation domain of 960 × 960 × 1280 grid points with a time step of 11.4 ns (giving a CFL number of 0.1 in the background medium). The simulation is run until steady state is reached after 25407 time steps.

The GPU server (dual Intel Xeon E5-2620 v4 with 8 Nvidia Tesla P40 GPUs) described in section 6.2.1 was used to run the LFB simulations. While the reference simulation was performed using a global domain approach with the MPI implementation of k-Wave running on Anselm cluster[2]. Anselm is Sandy Bridge CPU (2×8 core Intel E5-2665 and 64 GB RAM per node) based cluster with 40 Gbit/s interconnect in Fat-tree topology. The reference simulation was run using 80 nodes (1280 cores) to approximately match 96 TFLOP/s peak performance achieved by the GPU server.

The LFB simulation was run in three decompositions (see fig. 6.22), rank-one (1, 1, 8), rank-two (1, 2, 4) and rank-three (2, 2, 2) with the overlap size ranging from 4 to 16 grid points. Numerically optimized bell functions (discussed in section 5.4.1) were used in all investigated configurations.

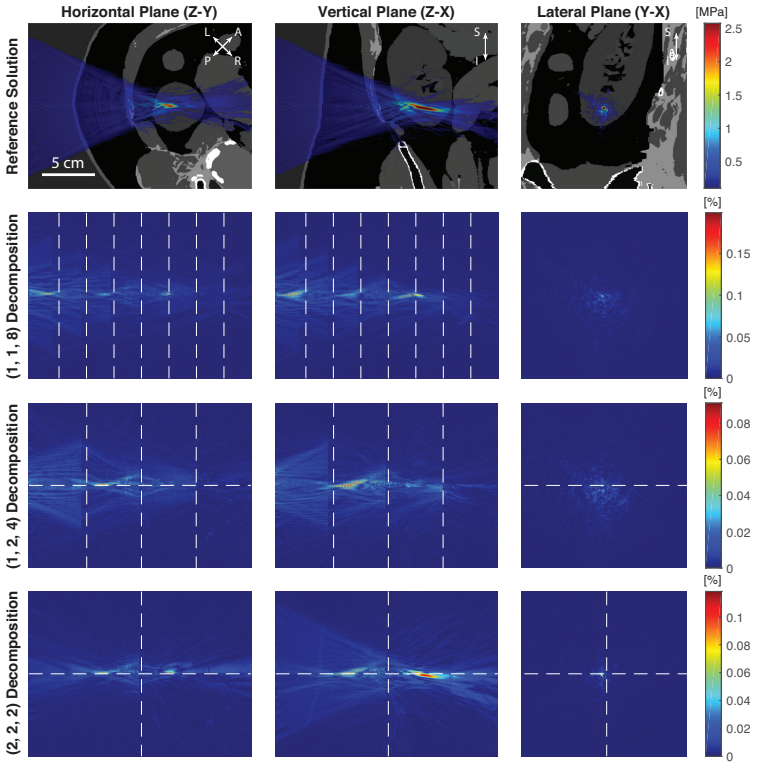


Figure 6.22: (Top Row) Output from the global domain simulation showing the peak positive pressure in steady state overlaid on the map of the tissue sound speed derived from the AustinWoman model. Three slices through the geometrical focus of the transducer are shown. (Lower Rows) Error plots showing the difference between the local domain decomposition and global domain simulations for three different decompositions using an overlap of four grid points. The sub-domain boundaries are shown with dashed lines.

The simulation error scaling (see fig. 6.23 left) roughly follows expected behavior, given our analytical LFB results. The rank-one decomposition exhibits the highest error as it

maximizes the number of interfaces between the transducer and the focus point. The decomposition is also slightly faster than rank-two and three variants (see fig. 6.23 right) as it requires only two overlaps per subdomain and minimizes number of overlaps transferred through QPI interface of the PNY server (discussed in section 6.2.1). The other two decompositions perform similarly to each other in both simulation time and accuracy.

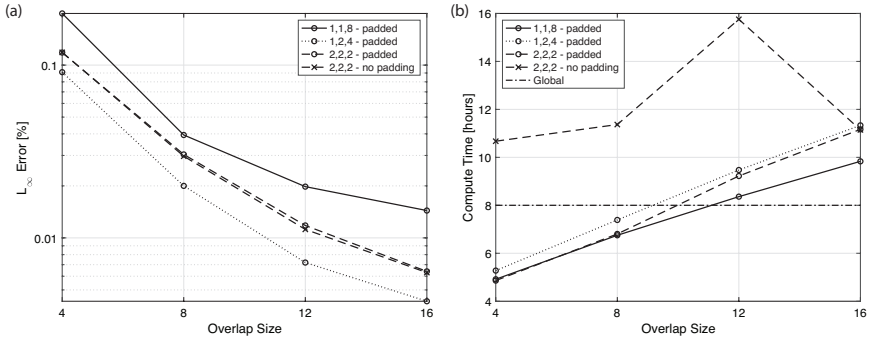


Figure 6.23: Change in the (a) L_∞ error and (b) compute time with the size of the overlap between subdomains for different domain decompositions.

Figure 6.23 also highlights impact of subdomain size on the performance of FFT algorithms. The simulation time of the rank-three decomposition is shown both with and without zero-padding, which is used to minimize prime factors of the subdomain size in each dimension (see table 6.1) thus significantly improving performance of local FFTs.

Table 6.1: Summary of decompositions, compute times and errors for the HIFU kidney sonication simulation using LFB approach with different arrangements of sub-domains and overlap sizes. The sub-domains are zero padded to give small prime factors and the largest prime factor in each dimension is listed. The memory usage is reported per GPU.

Decomp (x, y, z)	Overlap [grid points]	Sub-domain [grid points]	Factors (x, y, z)	Memory [GB]	Compute Time [h:min]	L_∞ error [%]
(1, 1, 8)	4	972 × 972 × 192	(3, 3, 3)	21.8	4:55	0.20
	8	972 × 972 × 192	(3, 3, 3)	22.2	6:45	0.039
	12	972 × 972 × 192	(3, 3, 3)	22.6	8:21	0.020
	16	972 × 972 × 192	(3, 3, 3)	23.0	9:50	0.014
(1, 2, 4)	4	972 × 512 × 384	(3, 2, 3)	23.1	5:17	0.091
	8	972 × 512 × 384	(3, 2, 3)	23.4	7:23	0.020
	12	972 × 512 × 384	(3, 2, 3)	23.8	9:28	0.0072
	16	960 × 512 × 360	(5, 2, 5)	22.8	11:20	0.0043
(2, 2, 2)	4	512 × 512 × 648	(2, 2, 3)	21.0	4:52	0.12
	8	512 × 512 × 672	(2, 2, 7)	22.1	6:48	0.030
	12	512 × 512 × 672	(2, 2, 7)	22.4	9:13	0.012
	16	512 × 512 × 672	(2, 2, 7)	22.8	11:10	0.0064

Chapter 7

Conclusions

With this chapter we conclude our work on enabling high-efficiency pseudo-spectral methods on modern accelerated machines. Here, we summarize the most relevant conclusions of the achieved results presented in this thesis, highlight our contributions and offer directions for future exploitation of our work.

Our work has focused on improving computational efficiency and scalability of numerical models for wave-like problems based on pseudo-spectral approach in the context of (GPU) accelerated machines and clusters. We chose to look into ultrasound models used in medicine as there is a growing number of novel applications for large-scale simulations, while there is also immense pressure on the cost of these simulation at the same time.

Chapter 1 was devoted to introduce problems in this area and models used to solve them. In particular, the k-Wave model stood out among other state of the art models due to its efficiency (especially in shared memory environments) and wide use in practice. In the following chapter (chapter 2), we illustrated the misfit between this model and recent developments in HPC architectures due to its high communication overhead leading to poor scaling.

To improve upon k-Wave approach, we had to weight against each other two conceptual approaches, an alternative discretizations to pseudo-spectral method used by k-Wave and a domain decomposition. The direct comparison of multiple discretizations of the same model is challenging and would require efficient implementations using each discretization. In chapter 3, we therefore used an alternative approach in which we stripped the model down to its simplest elements, wave and Burgers' equations. We discretized these two equations with (k-space) pseudo-spectral, FDM, DGM and FEM methods comparing numerical performance and efficiency of resulting solvers. This approach allowed us to confirm that the k-space pseudo-spectral method is significantly more suitable to our problem than the alternatives and the domain decomposition is a better way to reduce the communication overhead in the k-Wave model.

In chapter 5, we analyzed a variety of domain decomposition techniques and showed that the overlapping decompositions are the most suitable for our purposes. To overcome the periodic nature of the Fourier basis with reasonable computational efficiency, we investigated a range of Fourier extension methods, settling on windowing approach. We showed that the method allows for reasonable degree of flexibility in the terms of accuracy and performance for both of our building block equations. Further, we developed a numerical optimization technique to find windowing functions that outperform Error function in our use case.

Having found optimal techniques (k-space pseudo-spectral method with Local Fourier basis) to achieve aforementioned goals, we applied those techniques to the k-Wave model introduced in chapter 4. Chapter 6 describes derivation of our k-Wave LFB model, its practical implementation and evaluation. The implementation, based on MPI+X design pattern with OpenMP, CUDA or OpenCL, allows to support almost all common accelerator architectures.

We have presented detailed analysis of the implementation across variety of HPC platforms and accelerators. The analysis showed that k-Wave LFB achieves vastly superior strong and weak scalability compared to global k-Wave on non-accelerated clusters. Further we showed that the reduction in communication enables efficient use of GPU accelerated clusters. Finally, realistic scenario of HIFU kidney sonication was used to confirm usability and benefit of our solution in practice.

7.1 Key Contributions

The primary contribution of this thesis is a novel domain decomposition approach based on local Fourier basis (LFB) and its application to the state of the art ultrasound simulation package (k-Wave GDD) widely used in medical applications. We have used the proposed DDM to implement the k-Wave LFB solver, which significantly improves upon efficiency of the original k-Wave solver. We have shown that the k-Wave LFB can be 4.8 to 7.5 times faster (while using the same hardware resources) depending on the chosen accuracy/performance compromise. Further, significant reduction in the communication allowed our implementation to efficiently utilize GPU accelerated machines with near perfect weak scaling.

We have evaluated a range of approaches to LFB based domain decomposition and performed detailed analysis of its variants and their properties in the context of wave-dominated problems. We have contextualized the method with other DDMs and shown its similarity to the Restricted Additive Schwarz method.

Finally, we have proposed an application specific optimization approach to the design of the bell functions utilized by LFB, which can outperform traditionally used windowing functions thus allowing to reduce overlap depth while maintaining accuracy.

Chapter 8

Future Work

The domain decomposition approach for Fourier pseudo-spectral methods devised in this thesis opens a number of research avenues.

8.1 Irregular Simulation Domains

The domain decomposition allows for irregular simulation domains, which are notoriously difficult to model with methods using Fourier basis. Figure 8.1 shows an ultrasound simulation, which utilizes a combination of domain decomposition and PML layer to more closely follow shape of the focused wave and save 25 % of computational resources.

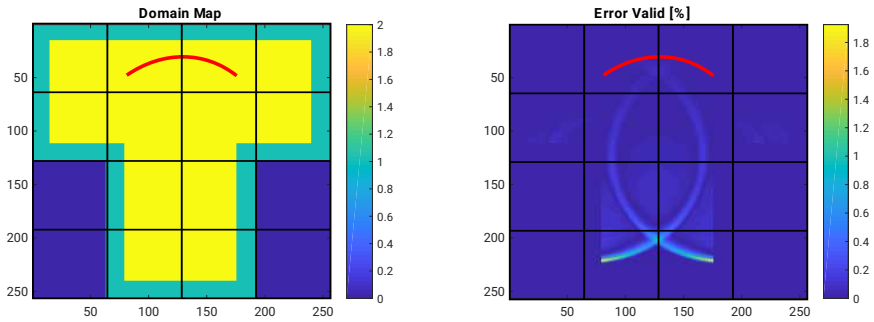
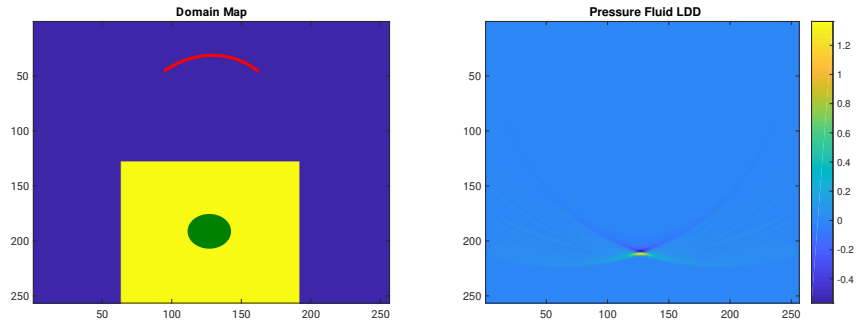


Figure 8.1: Sparse two dimensional simulation of focused ultrasound (initial wave profile in red) in domain decomposed into 4-by-4 sub-domain grid. The sub-domain map (left) shows sub-domains included (yellow) and left out (dark blue) from the simulation with PML layer in between. The simulation exhibited sub-2 % error while saving 25 % of computational resources.

8.2 Hybrid Models Coupling

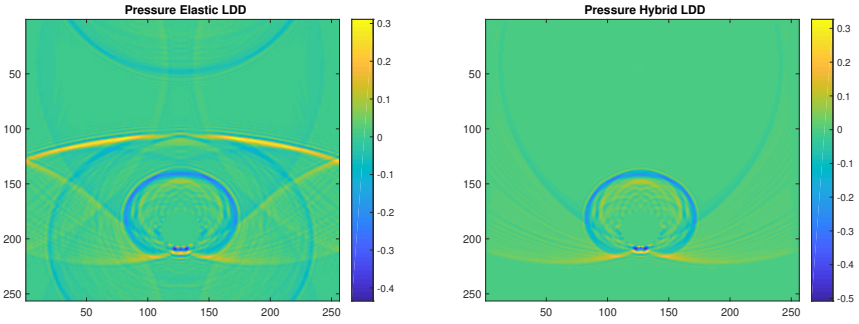
The weak sub-domain coupling of the proposed domain decomposition method allows for model coupling and hybrid simulations. Figure 8.2 shows coupling between ultrasound models for fluid and elastic medium. The simulation domain is subdivided into 4-by-4 sub-domains with four of them (surrounding a green disk with non-zero shear sound speed – see fig. 8.2a) using the elastic model. The equivalency of these models in regions of zero shear sound speed allows for a straightforward coupling through overlaps. The combination of these models allows to accurately model isolated regions of elastic materials submerged

in fluid medium (such as bone tissue surrounded by soft tissue), while maintaining efficiency advantages of fluid model in the majority of the domain.



(a) Simulation domain map

(b) Fluid-only simulation result



(c) Elastic-only simulation result

(d) Hybrid simulation result

Figure 8.2: Hybrid ultrasound simulation in two spatial dimensions in 4-by-4 decomposition using elastic in four (yellow) and fluid model in the remaining sub-domains. In contrast to the fluid simulation (b), the hybrid simulation (d) allows to model non-zero shear sound speed (green disk) using elastic model (c), while maintaining advantages (such as PML and k-Space) of the fluid model in majority of the domain.

The domain decomposition can be similarly used to mitigate drawbacks of the k-Space pseudo-spectral method such as poor ability to capture jump discontinuities in medium properties.

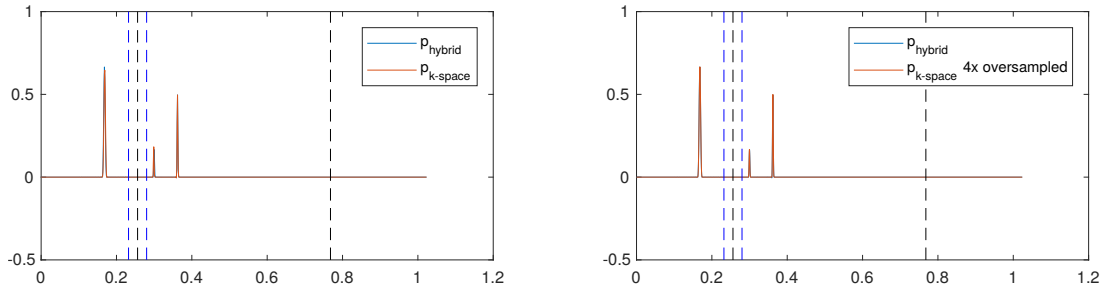


Figure 8.3: Comparison of hybrid k-Space/DGM and pure k-Space simulation of wave propagation through an off-grid jump discontinuity in sound speed (black dashed lines). The DGM patch is applied to the interval between blue dashed lines on the left.

Figure 8.3 shows a hybrid simulation, which uses a small DGM sub-domain over the jump discontinuity in sound speed to capture interface with a sub-grid accuracy. The point-wise difference of the hybrid and k-space solution shows that the hybrid solution converges to the over-sampled k-space solution (see fig. 8.4). However, it also shows spurious oscillations, which suggest that the additional filtering may be required to optimize coupling of the models.

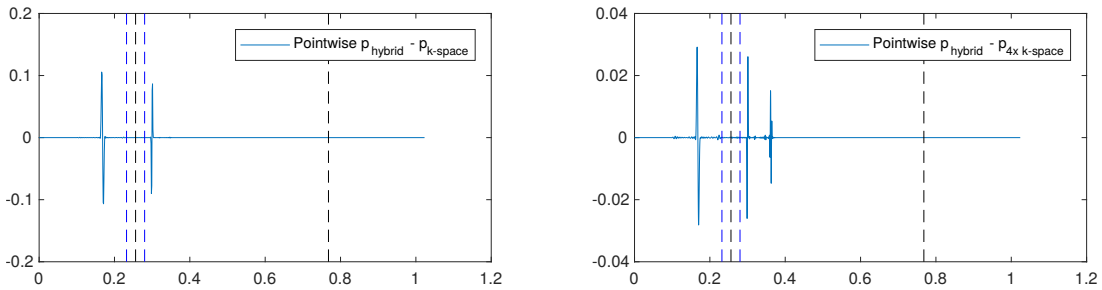


Figure 8.4: Comparison of hybrid k-Space/DGM and pure k-Space simulation of wave propagation through an off-grid jump discontinuity in sound speed (black dashed lines). The DGM patch is applied to the interval between blue dashed lines on the left.

The detail of fig. 8.3 (fig. 8.5) near the interface clearly shows primary source of the error in k-Space method. The method cannot precisely capture the position of the interface which causes noticeable phase shift of both transmitted and reflected part of the wave.

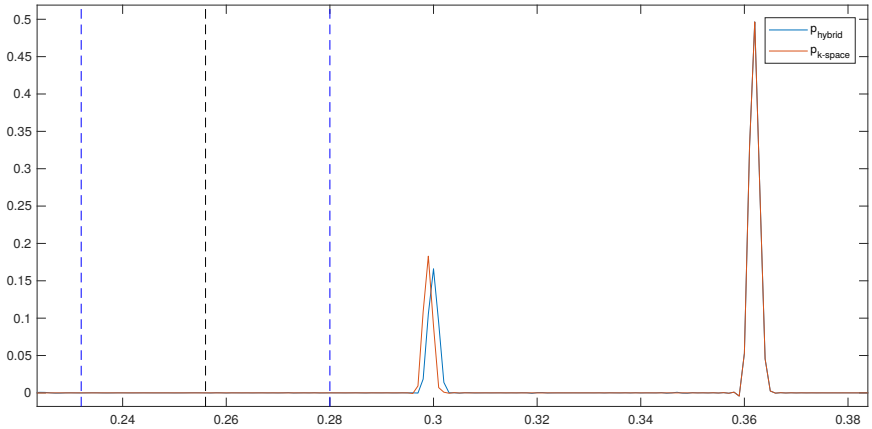


Figure 8.5: Detail of the reflected wave near interface in Figure 8.3 (top-left) clearly shows phase error of the k-Space solution due to its inability to precisely capture interface position.

Bibliography

Direct Contributions

[1] Jiri Jaros, Filip Vaverka, and Bradley E. Treeby. Spectral domain decomposition using local fourier basis: application to ultrasound simulation on a cluster of GPUs. *Supercomputing Frontiers and Innovations*, 3(3), November 2016.

[2] Bradley Treeby, Filip Vaverka, and Jiri Jaros. Performance and accuracy analysis of nonlinear k-wave simulations using local domain decomposition with an 8-gpu server. In *Proceedings of Meetings on Acoustics 21ISNA*, volume 34 of number 1, page 022002. Acoustical Society of America, 2018.

[3] Filip Vaverka. Case study on multi-domain decomposition of k-wave simulation framework. In *Computer architectures and diagnostics 2016*, pages 37–40, Brno, CZ. Faculty of Information Technology BUT, 2016.

[4] Filip Vaverka. Towards large-scale ultrasound simulations in soft tissue for medical applications. In *PAD 2019*, pages 64–67, Doksy, CZ. Academic and Medical Conference Agency, 2019.

[5] Filip Vaverka, Bradley E. Treeby, and Jiri Jaros. Evaluation of the suitability of intel xeon phi clusters for the simulation of ultrasound wave propagation using pseudospectral methods. In *Computational Science – ICCS 2019*. Volume 11538, pages 577–590. Springer International Publishing, Cham, 2019.

[6] Filip Vaverka, Bradley E. Treeby, and Jiri Jaros. Performance evaluation of pseudospectral ultrasound simulations on a cluster of xeon phi accelerators. In Tomáš Kozubek, Peter Arbenz, Jiří Jaroš, Lubomír Říha, Jakub Šístek, and Petr Tichý, editors, *High Performance Computing in Science and Engineering*, pages 99–115, Cham. Springer International Publishing, 2021.

Other

[7] Galal I El–Baghdady, MS El–Azab, and WS El–Beshbeshy. Legendre–gauss–lobatto pseudo–spectral method for one–dimensional advection–diffusion equation. *Eur. J. Oper. Res*, 2(1):29–35, 2015.

[8] Agessandro Abrahao, Ying Meng, Maheleth Llinas, Yuexi Huang, Clement Hamani, Todd Mainprize, Isabelle Aubert, Chinthaka Heyn, Sandra E Black, Kullervo Hynynen, et al. First-in-human trial of blood–brain barrier opening in amyotrophic lateral sclerosis using mr-guided focused ultrasound. *Nature communications*, 10(1):1–9, 2019.

[9] Nathan Albin and Sureka Pathmanathan. Discrete periodic extension using an approximate step function. *SIAM Journal on Scientific Computing*, 36(2):A668–A692, January 2014.

[10] Bob Alverson, Edwin Froese, Larry Kaplan, and Duncan Roweth. Cray xc series network. *Cray Inc., White Paper WP-Aries01-1112*, 2012.

[11] Daniel Appelö and N Anders Petersson. A stable finite difference method for the elastic wave equation on complex geometries with free surfaces. *Communications in computational physics*, 5(1):84–107, 2009.

[12] Jean-Francois Aubry, Oscar Bates, Christian Boehm, Kim Butts Pauly, Douglas Christensen, Carlos Cueto, Pierre Gelat, Lluis Guasch, Jiri Jaros, Yun Jing, et al. Benchmark problems for transcranial ultrasound simulation: intercomparison of compressional wave models. *arXiv preprint arXiv:2202.04552*, 2022.

[13] Michalakis A Averkiou. Tissue harmonic imaging. In *2000 IEEE Ultrasonics Symposium. Proceedings. An International Symposium (Cat. No. 00CH37121)*, volume 2, pages 1563–1572. IEEE, 2000.

[14] Alan Ayala, Stanimire Tomov, Azzam Haidar, and Jack Dongarra. heFFTe: highly efficient FFT for exascale. In Valeria V. Krzhizhanovskaya, Gábor Závodszky, Michael H. Lees, Jack J. Dongarra, Peter M. A. Sloot, Sérgio Brissos, and João Teixeira, editors, *Computational Science – ICCS 2020*. Volume 12137, pages 262–275. Springer International Publishing, Cham, 2020. Series Title: Lecture Notes in Computer Science.

[15] Paul Beard. Biomedical photoacoustic imaging. *Interface focus*, 1(4):602–631, 2011.

[16] Jean-Pierre Berenger. A perfectly matched layer for the absorption of electromagnetic waves. *Journal of Computational Physics*, 114(2):185–200, October 1994.

[17] Jean-Pierre Berenger. Three-dimensional perfectly matched layer for the absorption of electromagnetic waves. *Journal of Computational Physics*, 127(2):363–379, September 1996.

[18] John P Boyd. Chebyshev and fourier spectral methods:690.

[19] John P. Boyd. A comparison of numerical algorithms for fourier extension of the first, second, and third kinds. *Journal of Computational Physics*, 178(1):118–160, May 2002.

[20] Kolja Brix, Claudio Canuto, and Wolfgang Dahmen. Legendre-gauss-lobatto grids and associated nested dyadic grids. *arXiv preprint arXiv:1311.0028*, 2013.

[21] Oscar P. Bruno and Mark Lyon. High-order unconditionally stable FC-AD solvers for general smooth domains i. basic elements. *Journal of Computational Physics*, 229(6):2009–2033, March 2010.

[22] J.C. Butcher. A history of runge-kutta methods. *Applied Numerical Mathematics*, 20(3):247–260, 1996.

[23] Xiao-Chuan Cai and Marcus Sarkis. A restricted additive schwarz preconditioner for general sparse linear systems. *SIAM Journal on Scientific Computing*, 21(2):792–797, January 1999.

[24] C. Canuto, M. Y. Hussaini, A. Quarteroni, and T. A. Zang, editors. *Spectral Methods in Fluid Dynamics*. Springer Series in Computational Physics. Springer, New York, 1988.

[25] Claudio Canuto, M Yousuff Hussaini, Alfio Quarteroni, and Thomas A Zang. *Spectral methods: fundamentals in single domains*. Springer Science & Business Media, 2007.

[26] Milan Červenka and Michal Bednařík. A versatile computational approach for the numerical modelling of parametric acoustic array. *The Journal of the Acoustical Society of America*, 146(4):2163–2169, 2019.

[27] Duo Chen and Robert J McGough. A 2d fast near-field method for calculating near-field pressures generated by apodized rectangular pistons. *The Journal of the Acoustical Society of America*, 124(3):1526–1537, 2008.

[28] Eric Chung and Björn Engquist. Optimal discontinuous galerkin methods for wave propagation. *SIAM Journal on Numerical Analysis*, 44, September 2006.

[29] GT Clement and Kullervo Hynynen. Field characterization of therapeutic ultrasound phased arrays through forward and backward planar projection. *The Journal of the Acoustical Society of America*, 108(1):441–446, 2000.

[30] J. W. Cooley and J. W. Tukey. An algorithm for the machine calculation of complex Fourier series. *Mathematics of Computation*, 19(90):297–301, 1965.

[31] Adrien Dekkers, Vladimir Khodygo, and Anna Rozanova-Pierrat. Models of nonlinear acoustics viewed as an approximation of the navier-stokes and euler compressible isentropicsystems. *arXiv preprint arXiv:1811.10850*, 2018.

[32] James Demmel. Communication-avoiding algorithms for linear algebra and beyond. In *2013 IEEE 27th International Symposium on Parallel and Distributed Processing*, pages 585–585, 2013.

[33] Daniele Antonio Di Pietro and Alexandre Ern. *Mathematical aspects of discontinuous Galerkin methods*, volume 69. Springer Science & Business Media, 2011.

[34] Brian George Spencer Doman. *The Classical Orthogonal Polynomials*. WORLD SCIENTIFIC, 2015.

[35] Jianbin Fang, Henk Sips, LiLun Zhang, Chuanfu Xu, Yonggang Che, and Ana Lucia Varbanescu. Test-driving intel xeon phi. In *Proceedings of the 5th ACM/SPEC international conference on Performance engineering*. ICPE'14: ACM/SPEC International Conference on Performance Engineering, pages 137–148, Dublin Ireland. ACM, March 22, 2014.

[36] Charbel Farhat, Michel Lesoinne, Patrick LeTallec, Kendall Pierson, and Daniel Rixen. Feti-dp: a dual–primal unified feti method—part i: a faster alternative to the two-level feti method. *International Journal for Numerical Methods in Engineering*, 50(7):1523–1544, 2001.

[37] Charbel Farhat, Jan Mandel, and Francois Xavier Roux. Optimal convergence properties of the feti domain decomposition method. *Computer Methods in Applied Mechanics and Engineering*, 115(3):365–385, 1994.

[38] Charbel Farhat and Francois-Xavier Roux. A method of finite element tearing and interconnecting and its parallel solution algorithm. *International Journal for Numerical Methods in Engineering*, 32(6):1205–1227, 1991.

[39] Alena V Favorskaya, Michael S Zhdanov, Nikolay I Khokhlov, and Igor B Petrov. Modelling the wave phenomena in acoustic and elastic media with sharp variations of physical properties using the grid-characteristic method. *Geophysical Prospecting*, 66(8):1485–1502, 2018.

[40] R.P. Feynman, R.B. Leighton, M. Sands, and EM Hafner. *The Feynman Lectures on Physics; Vol. I*, volume 33. AAPT, 1965, page 750.

[41] C. A. J. Fletcher. *Computational galerkin methods*. In *Computational Galerkin Methods*. Springer Berlin Heidelberg, Berlin, Heidelberg, 1984, pages 72–85.

[42] Bengt Fornberg. High-order finite differences and the pseudospectral method on staggered grids. *SIAM Journal on Numerical Analysis*, 27(4):904–918, 1990.

[43] M. Frigo and S.G. Johnson. The design and implementation of fftw3. *Proceedings of the IEEE*, 93(2):216–231, 2005.

[44] Andreas Frommer and Daniel Szyld. Weighted max norms, splittings, and overlapping additive schwarz iterations. *Numerische Mathematik*, 83, February 2000.

[45] Martin J Gander. SCHWARZ METHODS OVER THE COURSE OF TIME:28.

[46] Martin J. Gander and Laurence Halpern. Absorbing boundary conditions for the wave equation and parallel computing. *Mathematics of Computation*, 74(249):153–177, March 18, 2004.

[47] Amir Gholami, Judith Hill, Dhairya Malhotra, and George Biros. AccFFT: a library for distributed-memory FFT on CPU and GPU architectures. *arXiv:1506.07933 [cs]*, June 25, 2015.

[48] Michelle Ghrist, Bengt Fornberg, and Tobin A. Driscoll. Staggered time integrators for wave equations. *SIAM Journal on Numerical Analysis*, 38(3):718–741, January 2000.

[49] Sigal Gottlieb, Chi-Wang Shu, and Eitan Tadmor. Strong stability-preserving high-order time discretization methods. *SIAM Review*, 43(1):89–112, 2001.

[50] Karl F Graff. A history of ultrasonics. In *Physical acoustics*. Volume 15, pages 1–97. Elsevier, 1981.

[51] Juanjuan Gu and Yun Jing. Modeling of wave propagation for medical ultrasound: a review. *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, 62(11):1979–1992, November 2015.

[52] Ibrahim M. Hallaj and Robin O. Cleveland. Fdtd simulation of finite-amplitude pressure and temperature fields for biomedical ultrasound. *The Journal of the Acoustical Society of America*, 105(5):L7–L12, 1999.

[53] Seounghyun Ham and Klaus-Jürgen Bathe. A finite element method enriched for wave propagation problems. *Computers & Structures*, 94-95:1–12, March 2012.

[54] David J. Henwood and Javier Bonet. Finite elements: a gentle introduction. In 1996.

[55] E. Hille. *Functional Analysis and Semi-groups*. American Mathematical Society colloquium publications. American Mathematical Society, 1948.

[56] Charles Hirsch. *Numerical computation of internal and external flows: The fundamentals of computational fluid dynamics*. Elsevier, 2007.

[57] Eberhard Hopf. The partial differential equation $u_t + uu_x = \mu u_{xx}$. Technical report, INDIANA UNIV AT BLOOMINGTON, 1950.

[58] Fang Q. Hu, M.Y. Hussaini, and Patrick Rasetarinera. An analysis of the discontinuous galerkin method for wave propagation problems. *Journal of Computational Physics*, 151(2):921–946, 1999.

[59] Yanghong Huang and Adam Oberman. Finite difference methods for fractional laplacians. *arXiv:1611.00164 [math]*, November 1, 2016.

[60] Herbert H. J. Hum and James R. Goodman. Forward State for use in Cache Coherency in a Multiprocessor System. Patent US 6,922,756 B2, Intel Corporation, July 2005.

[61] M. Israeli, L. Vozovoi, and A. Averbuch. Domain decomposition methods for solving parabolic PDEs on multiprocessors. *Applied Numerical Mathematics*, 12(1):193–212, May 1993.

[62] M. Israeli, L. Vozovoi, and A. Averbuch. Parallelizing implicit algorithms for time-dependent problems by parabolic domain decomposition. *Journal of Scientific Computing*, 8(2):151–166, June 1993.

[63] M. Israeli, L. Vozovoi, and A. Averbuch. Spectral multidomain technique with local fourier basis. *Journal of Scientific Computing*, 8(2):135–149, June 1993.

[64] Jiri Jaros, Alistair P. Rendell, and Bradley E. Treeby. Full-wave nonlinear ultrasound simulation on distributed clusters with applications in high-intensity focused ultrasound. *Int. J. High Perform. Comput. Appl.*, 30(2):137–155, 2016.

[65] Jürgen W Jenne, Tobias Preusser, and Matthias Günther. High-intensity focused ultrasound: principles, therapy guidance, simulations and applications. *Zeitschrift für Medizinische Physik*, 22(4):311–322, 2012.

[66] David A. Kopriva. A spectral multidomain method for the solution of hyperbolic systems. *Applied Numerical Mathematics*, 2(3):221–241, 1986. Special Issue in Honor of Milt Rose's Sixtieth Birthday.

[67] Binod Kumar, Atul Kumar Bhosale, Masahiro Fujita, and Virendra Singh. Validating multi-processor cache coherence mechanisms under diminished observability. In *2019 IEEE 28th Asian Test Symposium (ATS)*, pages 99–995. IEEE, 2019.

[68] Shahram Latifi. *Hypercube-based topologies with incremental link redundancy*. Louisiana State University and Agricultural & Mechanical College, 1989.

[69] Randall J. LeVeque. *Finite Volume Methods for Hyperbolic Problems*. Cambridge Texts in Applied Mathematics. Cambridge University Press, 2002.

[70] Mucong Li, Yuqi Tang, and Junjie Yao. Photoacoustic tomography of blood oxygenation: a mini review. *Photoacoustics*, 10:65–73, 2018.

[71] Zhilin Li and Kazufumi Ito. *The Immersed Interface Method*. Society for Industrial and Applied Mathematics, 2006.

[72] Pierre-Louis Lions et al. On the schwarz alternating method. i. In *First international symposium on domain decomposition methods for partial differential equations*, volume 1, page 42. Paris, France, 1988.

[73] Meng-Sing LIOU. *Probing numerical fluxes: mass flux, positivity, and entropy-satisfying property.* In *Computational Fluid Dynamics Review 1998*, pages 318–330.

[74] Qing Huo Liu. The pseudospectral time-domain (pstd) algorithm for acoustic waves in absorptive media. *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, 45(4):1044–1055, 1998.

[75] Ezekiel Maloney and Joo Ha Hwang. Emerging hifu applications in cancer therapy. *International Journal of Hyperthermia*, 31(3):302–309, 2015.

[76] Eleanor Martin, Yan To Ling, and Bradley E Treeby. Simulating focused ultrasound transducers using discrete sources on regular cartesian grids. *IEEE transactions on ultrasonics, ferroelectrics, and frequency control*, 63(10):1535–1542, 2016.

[77] T Douglas Mast. Empirical relationships between acoustic parameters in human soft tissues. *Acoustics Research Letters Online*, 1(2):37–42, 2000.

[78] Roel Matthysen and Daan Huybrechs. Function approximation on arbitrary domains using fourier extension frames. *arXiv:1706.04848 [math]*, June 15, 2017.

[79] Robert J McGough. Rapid calculations of time-harmonic nearfield pressures produced by rectangular pistons. *The Journal of the Acoustical Society of America*, 115(5):1934–1941, 2004.

[80] Robert J McGough, Thaddeus V Samulski, and James F Kelly. An efficient grid sectoring method for calculations of the near-field pressure generated by a circular piston. *The Journal of the Acoustical Society of America*, 115(5):1942–1954, 2004.

[81] Ronald E Mickens. *Difference equations: theory, applications and advanced topics.* CRC Press, 2015.

[82] Ronald E Mickens. *Nonstandard Finite Difference Models of Differential Equations.* WORLD SCIENTIFIC, December 1993.

[83] Ronald E Mickens. *Nonstandard Finite Difference Models of Differential Equations.* WORLD SCIENTIFIC, December 1993.

[84] Douglas Miles. Compute intensity and the fft. In Bob Borchers and Dona Crawford, editors, *SC*, pages 676–684. ACM, 1993.

[85] Elise F. Morgan, G. Unnikrisnan, and Amira I. Hussein. Bone mechanical properties in healthy and diseased states. *Annual review of biomedical engineering*, 20:119–143, 2018.

[86] Naveen Namashivayam, Krishna Kandalla, Trey White, Nick Radcliffe, Larry Kaplan, and Mark Pagel. Exploring gpu stream-aware message passing using triggered operations. *arXiv preprint arXiv:2208.04817*, 2022.

[87] Martin Neumüller. *Space-Time Methods: Fast Solvers and Applications.* English. PhD thesis, 2013.

[88] Jens Niegemann, Richard Diehl, and Kurt Busch. Efficient low-storage runge–kutta schemes with optimized stability regions. *Journal of Computational Physics*, 231(2):364–372, 2012.

[89] Hiroaki Nishikawa. First, second, and third order finite-volume schemes for advection–diffusion. *Journal of Computational Physics*, 273:287–309, 2014.

[90] Steven A Orszag. Spectral methods for problems in complex geometrics. In *Numerical methods for partial differential equations*, pages 273–305. Elsevier, 1979.

[91] Eduardo L. Ortiz. The tau method. *SIAM Journal on Numerical Analysis*, 6(3):480–492, 1969.

[92] Raúl Pagán Muñoz and Maarten Hornikx. Hybrid fourier pseudospectral/discontinuous galerkin time-domain method for arbitrary boundary conditions. *The Journal of the Acoustical Society of America*, 141(5):3809–3809, 2017.

[93] Rodrigo B. Platte, Lloyd N. Trefethen, and Arno B. J. Kuijlaars. Impossibility of fast stable approximation of analytic functions from equispaced samples. *SIAM Review*, 53(2):308–318, January 2011.

[94] K.M.M. Prabhu. *Window Functions and Their Applications in Signal Processing*. October 2013.

[95] Jose Pujol. *Elastic wave propagation and generation in seismology*, volume 227. Cambridge University Press Cambridge, 2003.

[96] Rolf Rabenseifner, Georg Hager, and Martin Bernreuther. MPI+x - hybrid programming on modern compute clusters with multicore processors and accelerators:231.

[97] T.J. Rivlin. *Chebyshev Polynomials: From Approximation Theory to Algebra and Number Theory*. Pure and Applied Mathematics: A Wiley Series of Texts, Monographs and Tracts. Wiley, 1990.

[98] James LB Robertson, Ben T Cox, J Jaros, and Bradley E Treeby. Accurate simulation of transcranial ultrasound propagation for ultrasonic neuromodulation and stimulation. *The Journal of the Acoustical Society of America*, 141(3):1726–1738, 2017.

[99] Philip L. Roe. A comparison of numerical flux formulas for the euler and navier-stokes equations. In 2003.

[100] Yousef Saad. *Iterative Methods for Sparse Linear Systems*. Society for Industrial and Applied Mathematics, second edition, 2003.

[101] PL Sachdev, KT Joseph, and KR C Nair. Exact n-wave solutions for the non-planar burgers equation. *Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences*, 445(1925):501–517, 1994.

[102] Armen Sarvazyan. *2001 elastic properties of soft tissues*. In July 2001.

[103] Armen P Sarvazyan, Oleg V Rudenko, Scott D Swanson, J Brian Fowlkes, and Stanislav Y Emelianov. Shear wave elasticity imaging: a new ultrasonic technology of medical diagnostics. *Ultrasound in medicine & biology*, 24(9):1419–1435, 1998.

[104] Andreas Schäfer and Dietmar Fey. High performance stencil code algorithms for gpgpus. *Procedia Computer Science*, 4:2027–2036, 2011. Proceedings of the International Conference on Computational Science, ICCS 2011.

[105] Hermann Amandus Schwarz. *Ueber einen Grenzübergang durch alternirendes Verfahren*. Zürcher u. Furrer, 1870.

[106] Khosro Shahbazi, Jan S Hesthaven, and Xueyu Zhu. Multi-dimensional hybrid fourier continuation–weno solvers for conservation laws. *Journal of Computational Physics*, 253:209–225, 2013.

[107] Peter R Stepanishen and Kim C Benjamin. Forward and backward projection of acoustic fields using fft methods. *The Journal of the Acoustical Society of America*, 71(4):803–812, 1982.

[108] Robert S Strichartz. *A Guide To Distribution Theory And Fourier Transforms*. World Scientific, editor. 2003.

[109] Thomas Szabo and Jun-ru Wu. A model for longitudinal and shear wave propagation in viscoelastic media. *The Journal of the Acoustical Society of America*, 107:2437–46, June 2000.

[110] Makoto Tabei, T. Douglas Mast, and Robert C. Waag. A k-space method for coupled first-order acoustic propagation equations. *The Journal of the Acoustical Society of America*, 111(1):53–63, January 2002.

[111] E.F. Toro. Chapter 2 - the riemann problem: solvers and numerical fluxes. In Rémi Abgrall and Chi-Wang Shu, editors, *Handbook of Numerical Methods for Hyperbolic Problems*. Volume 17, Handbook of Numerical Analysis, pages 19–54. Elsevier, 2016.

[112] Andrea Toselli and Olof Widlund. *Domain Decomposition Methods – Algorithms and Theory*, volume 34. January 2005.

[113] Bradley E Treeby and Benjamin T Cox. K-wave: matlab toolbox for the simulation and reconstruction of photoacoustic wave fields. *Journal of biomedical optics*, 15(2):021314, 2010.

[114] Bradley E. Treeby and B. T. Cox. A k-space green's function solution for acoustic initial value problems in homogeneous media with power law absorption. *The Journal of the Acoustical Society of America*, 129(6):3652–3660, June 2011.

[115] Bradley E. Treeby, Jiri Jaros, Alistair P. Rendell, and B. T. Cox. 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*, 131(6):4324–4336, June 2012.

[116] L. Vozovoi, M. Israeli, and A. Averbuch. Spectral multidomain technique with local fourier basis II: decomposition into cells. *Journal of Scientific Computing*, 9(3):311–326, September 1994.

[117] Urvi Vyas and Douglas Christensen. Ultrasound beam simulations in inhomogeneous tissue geometries using the hybrid angular spectrum method. *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, 59(6):1093–1100, 2012.

[118] Urvi Vyas and Douglas A. Christensen. Extension of the angular spectrum method to calculate pressure from a spherically curved acoustic source. *The Journal of the Acoustical Society of America*, 130(5):2687–2693, 2011.

[119] P.N.T. Wells. Absorption and dispersion of ultrasound in biological tissue. *Ultrasound in Medicine and Biology*, 1(4):369–376, 1975.

[120] Peter Wells and Haidong Liang. Medical ultrasound: imaging of soft tissue strain and elasticity. *Journal of the Royal Society, Interface / the Royal Society*, 8:1521–49, June 2011.

[121] P Jason White, Gregory T Clement, and Kullervo Hynynen. Longitudinal and shear mode ultrasound propagation in human skull bone. *Ultrasound in medicine & biology*, 32(7):1085–1096, 2006.

[122] Christian Wieners. A space-time petrov-galerkin method for linear wave equations. In 2016.

[123] Lucas C. Wilcox, Georg Stadler, Carsten Burstedde, and Omar Ghattas. A high-order discontinuous galerkin method for wave propagation through coupled elastic-acoustic media. *J. Comput. Phys.*, 229(24):9373–9396, 2010.

[124] Elliott S Wise, Ben T Cox, and Bradley E Treeby. Mesh density functions based on local bandwidth applied to moving mesh methods. *Communications in Computational Physics*, 22(5):1286–1308, 2017.

[125] Elliott S Wise, James L B Robertson, Ben T Cox, and Bradley E Treeby. Staircase-free acoustic sources for grid-based models of wave propagation:4.

[126] Ruihong Yao, Jihong Hu, Wei Zhao, Yongde Cheng, and Chaofan Feng. A review of high-intensity focused ultrasound as a novel and non-invasive interventional radiology technique. *Journal of Interventional Medicine*, 2022.

[127] Xiaojuen Yuan, David Borup, James W Wiskin, Michael Berggren, Rick Eidens, and Steven A Johnson. Formulation and validation of berenger's pml absorbing boundary for the fdtd simulation of acoustic scattering. *IEEE transactions on ultrasonics, ferroelectrics, and frequency control*, 44(4):816–822, 1997.

[128] Xiaozheng Zeng and Robert J McGough. Evaluation of the angular spectrum approach for simulations of near-field pressures. *The Journal of the Acoustical Society of America*, 123(1):68–76, 2008.

[129] Tingting Zhang, Na Pan, Yuping Wang, Chunyan Liu, and Shimin Hu. Transcranial focused ultrasound neuromodulation-a review of the excitatory and inhibitory effects in human and animals. *Frontiers in Human Neuroscience*:568, 2021.

[130] Hui Zhou, Ken Raffenetti, Yanfei Guo, and Rajeev Thakur. Mpix stream: an explicit solution to hybrid mpi+ x programming. In *EuroMPI/USA'22: 29th European MPI Users' Group Meeting*, pages 1–10, 2022.

[131] Olgierd C. Zienkiewicz, Robert L. Taylor, and Jianzhong Zhu. *The finite element method. Vol. 1: Its basis and fundamentals*. Elsevier Butterworth-Heinemann, Amsterdam, 6. ed., repr edition, 2010. 733 pages. OCLC: 838170287.