Scaling the “Memory Wall” for Multi-Dimensional Seismic Processing with Algebraic Compression on Cerebras CS-2 Systems

Hatem Ltaief
Yuxi Hong
Extreme Computing Research Center
Computer, Electrical and Mathematical Sciences & Engineering Division
King Abdullah University of Science and Technology
Thuwal, Saudi Arabia
firstname.lastname@kaust.edu.sa

Leighton Wilson
Mathias Jacquelin
Cerebras Systems Inc.
Sunnyvale, California, United States
firstname.lastname@cerebras.net

Matteo Ravasi
David Keyes
Extreme Computing Research Center
Computer, Electrical and Mathematical Sciences & Engineering Division
King Abdullah University of Science and Technology
Thuwal, Saudi Arabia
firstname.lastname@kaust.edu.sa

ABSTRACT

We exploit the high memory bandwidth of AI-customized Cerebras CS-2 systems for seismic processing. By leveraging low-rank matrix approximation, we fit memory-hungry seismic applications onto memory-austere SRAM wafer-scale hardware, thus addressing a challenge arising in many wave-equation-based algorithms that rely on Multi-Dimensional Convolution (MDC) operators. Exploiting sparsity inherent in seismic data in the frequency domain, we implement embarrassingly parallel tile low-rank matrix-vector multiplications (TLR-MVM), which account for most of the elapsed time in MDC operations, to successfully solve the Multi-Dimensional Deconvolution (MDD) inverse problem. By reducing memory footprint along with arithmetic complexity, we fit a standard seismic benchmark dataset into the small local memories of Cerebras processing elements. Deploying TLR-MVM execution onto 48 CS-2 systems in support of MDD gives a sustained memory bandwidth of 92.58PB/s on $35,784,000$ processing elements, a significant milestone that highlights the capabilities of AI-customized architectures to enable a new generation of seismic algorithms that will empower multiple technologies of our low-carbon future.

KEYWORDS

Seismic Processing, Low-Carbon Energy Applications, AI-optimized Architecture, Low-Rank Matrix Approximation, High Memory Bandwidth, Extreme Parallelism, Energy Efficiency.

ACM Reference Format:
Hatem Ltaief, Yuxi Hong, Leighton Wilson, Mathias Jacquelin, Matteo Ravasi, and David Keyes. 2023. Scaling the “Memory Wall” for Multi-Dimensional Seismic Processing with Algebraic Compression on Cerebras CS-2 Systems. In *The International Conference for High Performance Computing, Networking, Storage and Analysis (SC ’23), November 12–17, 2023, Denver, CO, USA*. ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3581784.3627042

1 JUSTIFICATION FOR THE GORDON BELL PRIZE

High-performance matrix-vector multiplication using low-rank approximation. Memory layout optimizations and batched executions on massively parallel Cerebras CS-2 systems. Leveraging AI-customized hardware capabilities for seismic applications for a low-carbon future. Application-worthy accuracy (FP32) with a sustained bandwidth of 92.58PB/s (for 48 CS-2s) would constitute the second-highest throughput from June’23 Top500.

2 PERFORMANCE ATTRIBUTES

Performance Attributes	Our submission
Problem Size	Broadband 3D seismic dataset ($\sim 20k$ sources and receivers and frequencies up to 50Hz)
Category of achievement	Sustained bandwidth Scalability
Type of method used	Algebraic compression
Results reported on basis of	Whole application (for GPU cluster) Main kernel (for Cerebras cluster)
Precision reported	Single precision complex
System scale	Up to 48 Cerebras CS-2 systems, i.e., $35,784,000$ processing elements
Measurement mechanism	Timers; Memory accesses; Sustained bandwidth; Performance modeling

3 OVERVIEW OF THE PROBLEM

Reflection seismology is a remote sensing technique that utilizes reflected seismic waves to produce high-resolution images of the subsurface as well as estimates of the associated rock properties. While developed primarily to map anomalies corresponding to mineral or hydrocarbon deposits, it is now also being used for the more demanding requirements of carbon capture and storage [33], geothermal exploration [14], and assessment of near subsurface integrity for offshore wind farms [2].

Traditional algorithms rely on strong assumptions about the propagation of waves in the subsurface; namely, they consider a 1D layered medium and therefore ignore propagation effects due to lateral heterogeneities. The 1990s saw the development of a plethora of wave-equation-based processing methods that better handle the multi-dimensional nature of seismic waves [11, 26, 45]. In the 2010s most of these methods were recast as inverse problems [30, 42, 46, 47], providing unprecedented processing capabilities for enhancing the imaging of seismic data acquired in complex geology. However, such approaches come with a number of computational challenges, mostly associated with their repeated access to the entire seismic dataset when solving the associated inverse problem with iterative schemes[36]. Thus, industrial applications of these novel techniques are still in their infancy.

At the core of typical wave-equation-based processing algorithms lies a Multi-Dimensional Convolution (MDC) operator, which can be defined by the following integral relation:

$$y(t, \mathbf{x}_{VS}, \mathbf{x}_S) = \mathcal{F}^{-1}\left(\int_{\delta\mathbb{D}} K(\omega, \mathbf{x}_R, \mathbf{x}_S)\mathcal{F}\left(x(t, \mathbf{x}_R, \mathbf{x}_{VS})\right) d\mathbf{x}_R\right), \quad (1)$$

where x and y are time-domain seismic wavefields, and K is the kernel of the MDC operator. Here, $\mathcal{F}$ and $\mathcal{F}^{-1}$ represent the forward and inverse Fourier transforms, ω is the angular frequency, $\mathbf{x}_S$, $\mathbf{x}_R$ and $\mathbf{x}_{VS}$ are the spatial locations of the sources, receivers, and virtual sources, respectively. Once discretised, Eqn. (1) can be written in a compact matrix-vector form [36]:

$$\mathbf{y} = \mathbf{F}^H\mathbf{K}\mathbf{F}\mathbf{x}, \quad (2)$$

where $\mathbf{F}$ and $\mathbf{F}^H$ represent the operators performing forward and inverse Fast Fourier Transforms along the time/frequency axes (implemented, of course, as subroutines, not dense matrix multiplications), and $\mathbf{x}$ and $\mathbf{y}$ contain vectorized versions of the input and output functions. Finally, $\mathbf{K}$ is an operator that performs repeated Matrix-Vector Multiplications (MVMs) with the frequency matrices belonging to the seismic bandwidth of interest. Different choices of x, y, and K, as well as the creation of composite modelling operators that contain two or more MDC operators, lead to different applications. In this paper, we are concerned with solving Eqn. (1) directly for $x = r$, the so-called local reflectivity, given $y = p^-$ and $K = P^+$ defined as the time-domain up-going component and the frequency-domain down-going component of the seismic wavefield recorded along the $\delta\mathbb{D}$ boundary, respectively (Fig. 1). The local reflectivity can be interpreted as the wavefield that would be physically generated if sources and receivers were placed along the boundary $\delta\mathbb{D}$ and the medium above it was homogeneous. This problem is commonly referred to as Multi-Dimensional Deconvolution (MDD) [11, 47] and enables a number of different applications depending on where the boundary $\delta\mathbb{D}$ is located within the medium of interest. For example, if the boundary is placed on the seafloor, as in the numerical example in this paper, MDD can be used to eliminate free-surface multiples from ocean-bottom seismic recordings [12, 35, 37]. MDD can be also used to remove overburden-related multiples when the boundary is located at any given depth level in the subsurface [41, 43, 44]. In both cases, the retrieved local reflectivity contains only seismic arrivals originating below the boundary, making it easier to produce high-quality images of the discontinuities in the subsurface without artifacts arising from overburden effects. This is of particular relevance when the times of certain multiple arrivals overlap with that of primaries from the target of interest – e.g., a CO_2 storage site to be monitored over time. To foster the development of wave-equation-based processing algorithms and increase their industrial adoption, we need to alleviate the memory footprint and arithmetic complexity bottlenecks of Eqn. (2). This will enable the processing of seismic datasets that are orders of magnitude larger than currently possible, thus improving the accuracy and resolution of the resulting subsurface images. We here optimize the MVM computational kernel that accounts for the most time-consuming operation in the MDC operator: by leveraging low-rank matrix approximation [19, 20], we exploit the inherent data sparsity of seismic frequency matrices and retain only the critical information needed to apply the MDC operator for the required accuracy of MMD. While the literature is rich in low-rank matrix approximations for the MVM operation, this paper deals with a new research trend: **how to steer emerging specialized hardware exhibiting high memory throughput to deliver high performance for the intrinsically memory-bound MVM kernel?** Our synergistic approach combines algebraic compression with the capabilities of Cerebras CS-2 SRAM architecture to achieve sustained memory bandwidth of an unprecedented level in the field of seismic processing.

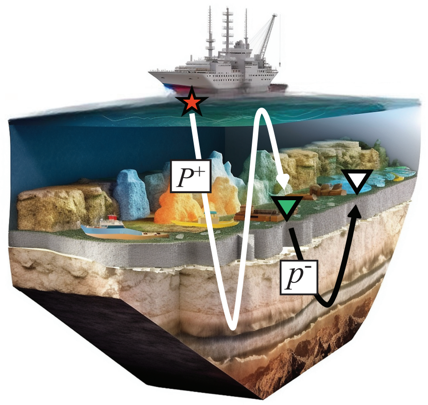


Figure 1: Schematic representation of the Multi-Dimensional Deconvolution problem. A red star indicates the source, a green triangle refers to the receiver, and the virtual source is represented by a white triangle.

"History doesn't repeat itself, but it often rhymes." [Mark Twain, attributed] The Connection Machine series, designed for AI, and the Blue Gene series, designed for protein folding, were each employed to win three Gordon Bell Prizes for applications far removed from their design domains, including for seismic imaging on the CM-2 in 1989. Will the CS-2 "rhyme"?

4 CURRENT STATE OF THE ART

State-of-the-art approaches to MDD mitigate the challenge of handling large amounts of data by solving the inverse problem associated with Eqn. (1) one frequency at a time (e.g., [12]). While this problem can be decoupled in the frequency domain, recent research has shown that this may have detrimental effects on the quality of the retrieved local reflectivity [43]. An alternative approach, originally proposed in [27], takes advantage of the hierarchically low-rank structure of seismic data in the frequency domain. By performing a step of pre-processing, where the different frequency matrices of the downgoing wavefield are compressed using an algebraic method of choice (e.g., rank revealing QR [16, 18], randomized SVD [21], adaptive cross approximation [49], etc.), the overall memory footprint of the kernel of the MDC operator can be significantly reduced. This idea, originally proposed in the context of the surface related multiple elimination (SRME) algorithm for two-dimensional synthetic and real datasets, can also be used to relax the memory and computational burden associated with the MDC operator and therefore enable the practical application of time-domain MDD.

More recently, [22] proposed an extension of the approach of [27], particularly suitable to large 3D seismic datasets. The idea is to split the matrix operator into tiles and compress them independently. Figure 2 shows a standard dense MVM on a 10×6 tiled matrix with tile size nb. Figure 3 represents the compression of the matrix operator and highlights the resulting bases with different ranks. This tile low-rank (TLR) matrix approximation [4, 8] relies on a flat representation of the compressed operator. While this is not the most optimal representation in terms of memory and operation savings compared to hierarchical formats i.e., $\mathcal{H}$-matrices [19, 20, 28], Hierarchically Semi-Separable (HSS) [17, 29, 38], Hierarchically Off-Diagonal Low-Rank (HODLR) [7, 9], and $\mathcal{H}^2$-matrix [13], the simplicity of the TLR data structure has led to broad application to 3D computational science problems at scale, on a myriad of architectures [1, 5, 6, 15, 31].

Additionally, [23] and [24] introduce a distance-aware re-arrangement of the rows and columns of the frequency slices that can leverage the spatial locality and eventually lead to improved compression capabilities. Reordering strategies that aim at reducing the overall distance between sources (and receivers) within the same tile can in fact dramatically decrease the ranks of the corresponding tiles: more specifically, the famous Hilbert sort algorithm, also known as Hilbert space-filling curve [39], is shown to provide the best compression capabilities over alternative strategies (e.g., Morton ordering) [3, 6].

Using TLR to accelerate the matrix-vector multiplication (TLR-MVM) requires a redesign of the algorithm. The resulting U and V bases from the compression phase need to be stacked in memory, as shown in Fig. 4, to ensure contiguous access when fetching in from memory U and V bases with different ranks, e.g., the rank $k_{3,3}$ of the indexed tile (3,3) as illustrated. The TLR-MVM operation can then proceed with three successive computational phases: 1) a batched MVM kernel for the V bases with variable sizes, 2) a memory shuffle operation to project from the V to the U bases, and 3) a batched MVM kernel for the U bases with variable sizes, as highlighted in Figs. 5, 6, and 7, respectively.

The memory layout and the reduced sizes of the bases may create further opportunities to fit in small shared caches of x86, ARM, and vector-based hardware solutions, while mitigating overheads of data motion within the memory subsystem. On GPU hardware accelerators, the challenge resides in ensuring high occupancy by engaging all cores and their associated memory banks during the computation. The current NVIDIA and AMD software ecosystems for GPUs do not provide support for batched execution required to effectively launch TLR-MVM with complex precisions and variable ranks. This serious lack of support for batched matrix operations in vendor-optimized numerical libraries has impeded deployment and performance of low-rank matrix computations on massively parallel architectures. The implementation of new batched kernels for TLR-MVM are needed to exploit the underlying GPU hardware resources and achieve sustained bandwidth close to the theoretical peak bandwidth, as detailed in [24].

5 INNOVATIONS REALIZED

5.1 Riding the Trend of AI-Customized Innovative Hardware

The growth of the AI market has led to significant hardware innovations over the last few years that translate into the on-chip deployment of high memory bandwidth technologies and high compute capabilities. While these innovations are primarily dedicated to supporting deep learning and inference workloads, recent work has highlighted promising ventures from the HPC scientific community to steer these AI-specialized hardware systems and support climate/weather modeling, computational astronomy, wireless communication, and seismic imaging applications on Cerebras CS-2 [25] and Graphcore IPUs [32]. The challenges related to executing HPC workloads on these two AI-customized hardware solutions are similar to those faced by the HPC community in the early days of GPU hardware accelerators and ARM processors, which came from the gaming industry and embedded systems, respectively. Today, these processor architectures provision most of the world's Top10 most powerful computers and have transformed the HPC and AI communities.

5.2 The Cerebras CS-2 Wafer Scale Engine

The Cerebras CS-2 Wafer Scale Engine (WSE) is a 2D array of dies that occupies an entire wafer. A die is itself a grid of tiles linked via a 2D grid fabric interconnect. A tile contains a router, a processing element (PE) and single-cycle access memory (SRAM). Figure 8 highlights a WSE containing a grid of tiles on the right and a zoom-in view of a single tile with a PE and its local SRAM memory on the left. This flat memory machine model has been successful thanks to the high memory bandwidth offered by the SRAM memory technology. However, more memory accesses are required to perform the same workload compared to a cache-based memory subsystem where transient data can be kept in caches before being written back to main memory. Moreover, the resource disaggregation of the WSE makes it akin to a distributed-memory architecture: inter-PE data movement involving the fabric is required to access data residing outside the memory of a given PE. The high throughput fabric allows to transfer data at the same rate as the SRAM memory although at a higher latency, making it relatively more expensive

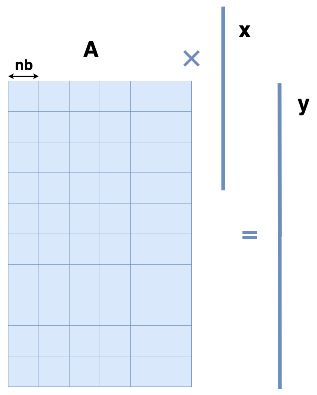


Figure 2: Original dense MVM.

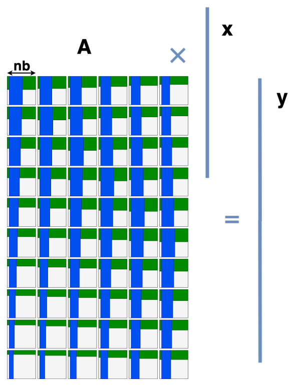


Figure 3: Rank-compressed operator.

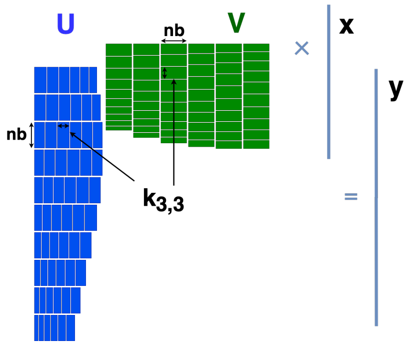


Figure 4: Stacked bases U and V.

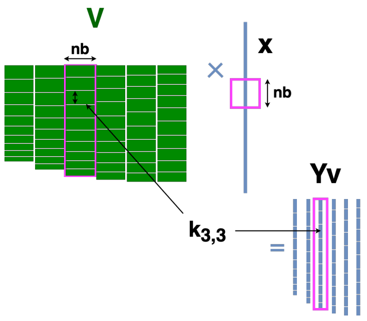


Figure 5: V-batch stage of MVM.

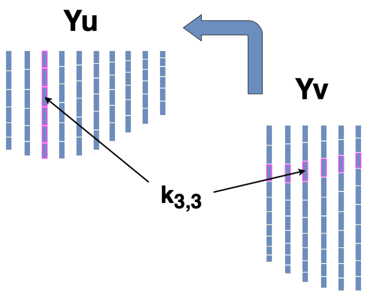


Figure 6: Shuffle from V to U bases.

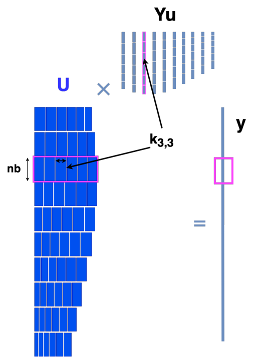


Figure 7: U-batch of MVM.

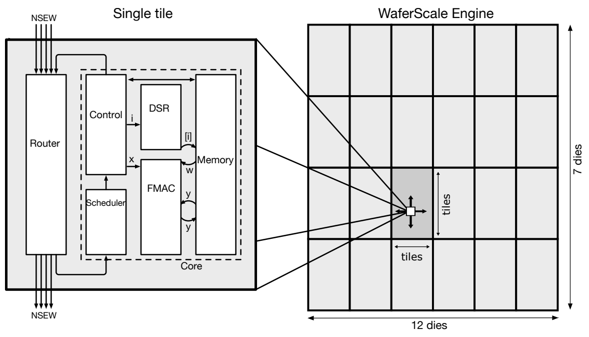


Figure 8: Cerebras CS-2 Wafer Scale Engine.

than accessing data within the local SRAM of a PE.

5.3 Novel Communication-Avoiding Memory Layout

In this paper, we develop a new high-performance TLR-MVM kernel that takes advantage of the high SRAM memory bandwidth of Cerebras CS-2 systems, while mapping the seismic datasets onto the millions of PEs. Our previous implementation on Graphcore IPUs [32] consists of porting the three computational phases of TLR-MVM (Figs. 5, 6, and 7), as originally developed for x86/ARM/vector/GPU systems [22, 24, 31]. While we report higher memory throughput on IPUs than the other conventional hardware systems thanks to the SRAM memory technology, the second phase (i.e., memory shuffling) requires synchronization across the IPUs, which is further exacerbated due to the Bulk Synchronous Parallel (BSP) paradigm that characterizes the Graphcore architecture. With Cerebras CS-2 systems, our new TLR-MVM kernel combines the first and third stages together, while removing the expensive second phase that necessitates cross-fabric communications. Figure 9 pictures our new SRAM-aware implementation of the TLR-MVM kernel on the disaggregated memory resource of Cerebras CS-2. For example, we take all V bases of the tile column $A_{:,3}$, stack them vertically, and execute the V-batch of MVMs similar to Fig. 5. Then, instead of stacking horizontally the U bases of a *tile row* as shown in Fig. 7, we select and reshape the U bases of each *tile column* by storing them side-by-side. The stack width is an important tuning parameter as it enables the exposure of more concurrency when strong scaling operations by splitting the stacked bases into chunks of similar sizes. We can then launch the U-batch MVM, without having to perform the expensive memory shuffle operation described in Fig. 6. This comes at the price of an increase of data movement of multiple y vectors in and out but occurring only within the local SRAM memory of each PE thanks to the flat memory machine model, as explained in Section 5.2. This is where the high throughput advantage of SRAM plays an effective role.

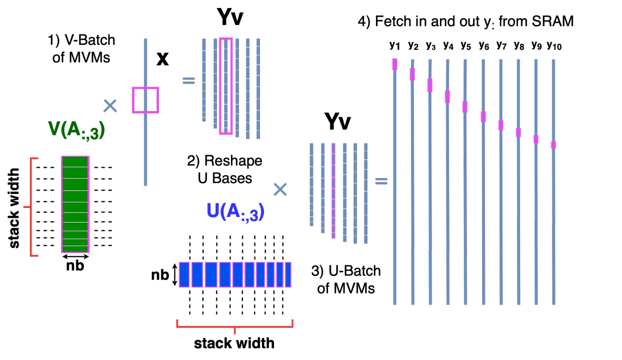


Figure 9: Design of a new communication-avoiding implementation of the TLR-MVM kernel.

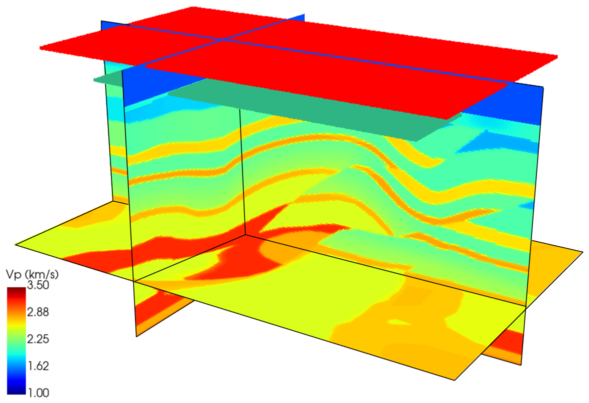


Figure 10: SEG/EAGE Overthrust model. The red grid refers to sources and the green grid to receivers on the seafloor.

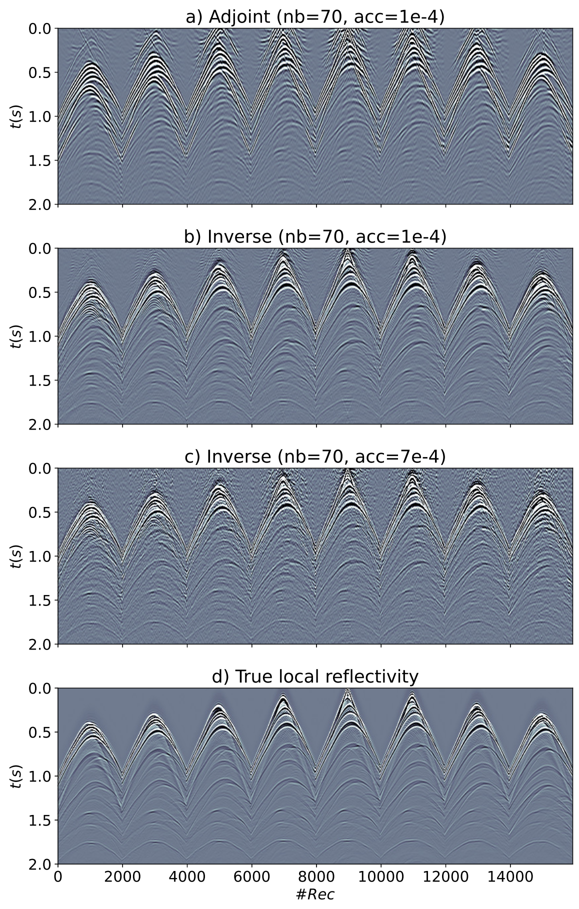


Figure 11: MDD results. a) Cross-correlation (i.e., adjoint) and b) Inversion with $nb = 70$ and $acc = 1\mathbf{e} - 4$, c) Inversion with $nb = 70$ and $acc = 7\mathbf{e}-4$. d) Ground truth from finite-difference modelling.

6 HOW PERFORMANCE WAS MEASURED

6.1 Description of the Seismic Dataset

Our numerical example is based on the SEG/EAGE Overthrust model, an openly available 3D geological model created as part of a joint project between the Society of Exploration Geophysicists (SEG) and the European Association of Geoscientists and Engineers (EAGE) in 1996 [10]. Our model is modified by including a 300m water column in order to be able to mimic an ocean-bottom acquisition scenario, for a total size of $3 \times 5 \times 2.3$ km^3. The acquisition geometry consists of a grid of 217×120 sources at a depth of 10m below the air-water interface and 177×90 receivers at a depth of 300m (i.e., along the seafloor). In both cases, 20m spacing is used for both the inline (x) and crossline (y) directions (Fig. 10).

Pressure and particle velocity data are modeled with a flat wavelet up to 45Hz for a total time of 4.5s (with a 4ms temporal sampling), with each dataset having an effective size of approximately 1.8TB. In pre-processing, wavefield separation is performed to separate the downgoing (p^+) from the upgoing (p^-) components of the pressure wavefield, with the former being transformed to the frequency domain for further computations. Given the maximum frequency available in the data, the length and sampling of the time axis, 230 complex-valued frequency matrices of size 26040×15930 are stored for a total size of 763GB. Finally, each frequency matrix is compressed using the TLR compression algorithm described above with uniform tile size $nb = 70$ (similar results can be achieved using smaller tile sizes $nb = 25$ and $nb = 50$). As previously shown in [23, 24], using frequency matrices with the original arrangement of rows (sources) and columns (receivers) leads to poor compression of the dataset. On the other hand, when Hilbert reordering is applied upfront to each frequency matrix, the main contributions gather towards the matrix main diagonal. This leads to superior compression capabilities with a 7X compression factor of the original dataset when using a tile-wise accuracy tolerance of $acc = 1\text{e} - 4$ (110GB for the compressed version versus 763GB for the original dataset).

6.2 Running the Whole MDD Simulation on Real Datasets

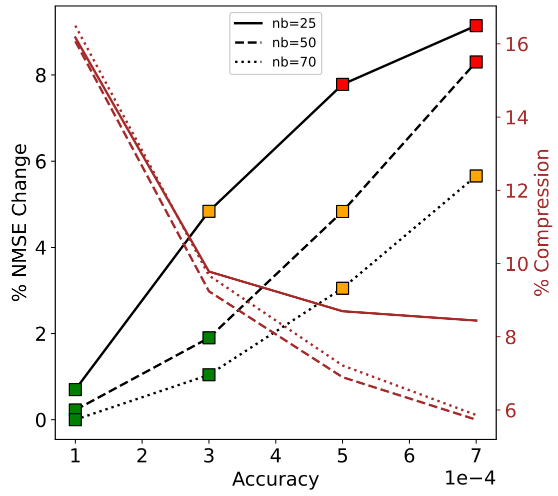
The resulting TLR compressed kernels are now used to solve the MDD problem in Eqn. (1) via 30 iterations of LSQR [34] on a GPU-based system composed of computational nodes equipped by four NVIDIA V100 GPUs, each with 32GB of main memory. Further details on the GPU implementation can be found in [24]. To begin with, we consider a single virtual source along the seafloor at horizontal location ($y = 1620$m, $x = 2460$m), for which the entire compressed seismic datasets max out the memory capacity of four NVIDIA V100 GPUs; the deconvolved wavefield is extracted over eight equidistant receiver lines along the crossline direction and shown in Fig. 11 alongside the cross-correlation (i.e., adjoint) wavefield and the directly modelled reflectivity. We first observe how MDD has removed free-surface effects appearing in the cross-correlation dataset, providing a local reflectivity response that closely resembles the ground truth. However, owing to the ill-posed nature of the inverse problem, we notice some minor differences between the estimated and true solution; most prominently, the frequency content is slightly higher in the true response than in the estimated one, meaning that MDD has struggled to recover some of the higher frequencies in the dataset. Nevertheless, empowered by the compression capabilities of our TLR pre-processing step, we have presented here what is to the best of our knowledge the very first successful implementation of time-domain MDD to a large-scale 3D seismic dataset.

6.3 Impact of the Compression Threshold on MDD Accuracy

To illustrate the effect of the accuracy threshold, we present another MDD result using a much looser accuracy tolerance ($acc = 7e-4$) for the TLR compression of the frequency matrices (Fig. 11c). Comparing with Fig. 11b, it is clear that reducing the accuracy to further increase compression introduces unwanted noise in the solution. A higher level summary of the effect of tile size *nb* and accuracy *acc* is presented in Fig. 12: here, we observe two opposite trends. By loosening the accuracy threshold (from $1e-4$ to $7e-4$), we trade off quality in the final solution for additional compression. We identify three regions: green, orange, and red, referring respectively to accurate, satisfactory (but affected by additional noise), and unacceptably inaccurate solutions. In general, the degree of accuracy required when performing MDD depends on the downstream application as well as the amount of computational (and monetary) resources a user is willing to afford. We consider the green solutions to be accurate for subsequent quantitative analysis (e.g., seismic inversion), while the yellow solutions may still be suitable for qualitative analysis (e.g., seismic interpretation). For the subsequent experiments, we report only on the five green configurations.

6.4 MDD in Action Removes Free-Surface Related Effects

Finally, to further corroborate the ability of the MDD algorithm to remove free-surface related effects from the deconvolved wavefield, a line of 177 virtual sources is considered along a fixed crossline ($y = 1620$m), which can run in an embarrassingly parallel fashion on $177 \times 4 = 708$ NVIDIA V100 GPUs. In real production runs, the number of virtual sources can be of the order of tens of thousands, which can easily fill up the memory capacity of large-scale supercomputers. Zero-offset time sections (i.e., from horizontally co-located source and receiver) are shown in Fig. 13 for the full dataset, upgoing dataset, and MDD dataset. Red arrows indicate downgoing events that are present in the full data and are suppressed in the upgoing component, and green arrows indicate upgoing free-surface multiples that are present in the upgoing dataset and are successfully removed from the MDD data. Given the increased level of background noise in the deconvolved data, a simple stacking procedure is employed to produce the last panel in Fig. 13. Overall, this result highlights the benefit of applying MDD as part of



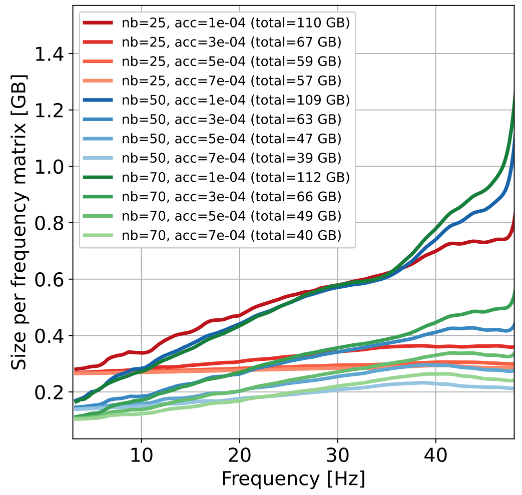


Figure 12: Effect of compression on the quality of the MDD inversion product. Top: Black lines represent percentage change of normalized mean square error of each solution against the benchmark solution with $nb = 70$ and $acc = 1e-4$. Brown lines refer to percentage of compression of each approximation compared to the original, dense solution. Bottom: Aggregated size of the U and V bases as function of frequency for the different combinations of nb and acc.

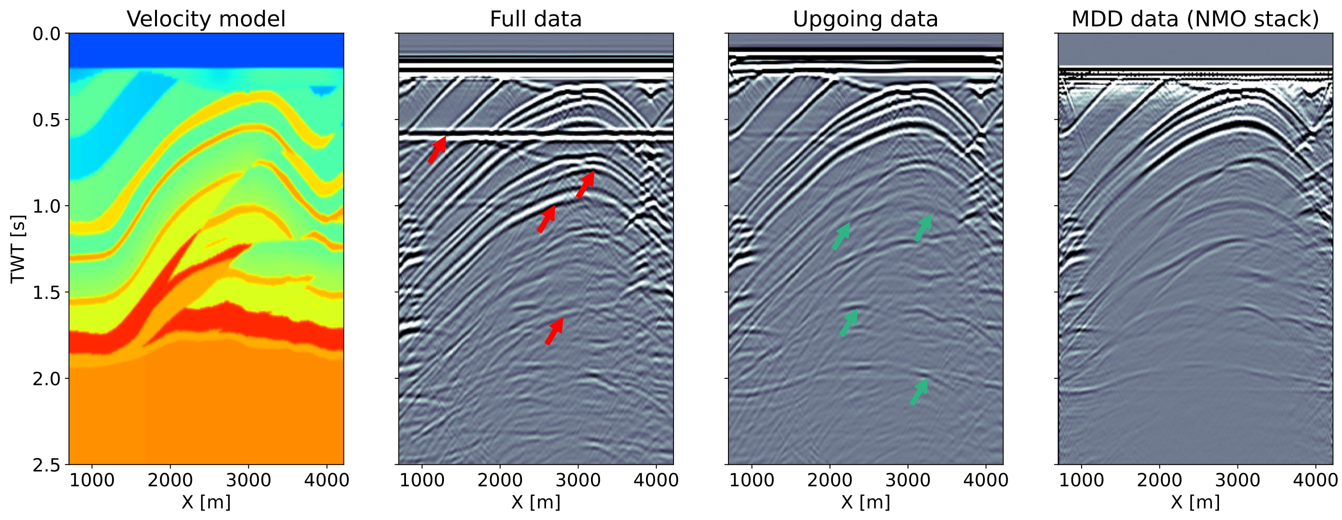


Figure 13: Zero-offset sections along $y = 1620m$ for a) velocity model (converted to time domain), b) full dataset (i.e., $p = p^+ + p^-$), c) upgoing dataset p^-, and d) estimated local reflectivity r. In the latter case, a standard post-processing flow is applied to be able to stack all of those traces corresponding to a single source-to-receiver midpoint; this is required because the zero-offset trace is usually noisy.

a seismic processing project prior to imaging, as mostly primary arrivals are now present in the data and free-surface effects are nicely suppressed.

6.5 CS-2 Environment Settings and Experimental Setup

The experiments are now ported to Cerebras CS-2 hardware by first dividing the matrices of the real datasets introduced in Section 6.1 into six "shards," since accommodating the full compressed matrix in CS-2 SRAM requires a minimum of six CS-2 systems, each comprising a grid of 757×996 processing elements (PEs), with a clock frequency of 850MHz. Each shard is mapped onto a CS-2 system using the Cerebras SDK [40], which allows users to develop and write programs in the Cerebras Software Language (CSL) on a simulator for performance modeling and on the actual hardware. Each shard uses up to 750×994 PEs, with the additional PEs on the fabric used for routing data on and off the wafer. The performance modeling tool captures the number of cycles and memory accesses by leveraging code compilation information and is able to provide reliable estimates of performance on the CS-2. Each PE can perform up to two 64-bit reads and one 64-bit write per cycle. On each PE, the 48kB of SRAM memory is divided up into eight banks of 6kB each. To perform two reads in a cycle, the reads must be from separate banks. Thus, one must properly align memory and pad arrays to guarantee this for every `fmac` instruction that is performed in the MVM.

We use up to 48 systems from G42's Condor Galaxy AI supercomputer, with the assistance of Cerebras. Since seismic processing and imaging MVM workloads are embarrassingly parallel, no communication is required between the CS-2 systems, nor between PEs on a system. We report the sustained bandwidth based on the worst cycle count across all PEs on all systems.

6.6 Complex Batched MVMs and Bandwidth Metrics

Seismic processing requires complex arithmetic. While this is widely supported in vendor-optimized numerical libraries, the batched mode of execution for MVMs lacks support for complex datatypes from all vendors of which we know. Therefore, we implement the complex MVM via four separate MVMs involving the real and imaginary parts. The complex MVM thus translates into four real MVMs, which are in FP32. As shown in Fig. 9, we deal with two batches of MVMs for U and V bases. Therefore, we can expose a total of eight independent MVMs. In terms of bandwidth metrics, we present two measures, which we refer to as "relative" and "absolute." To introduce these, consider a matrix-vector product $y = Ax$, where A is $M \times N$. In a traditional memory architecture, the N elements of x are read once and cached, each element of A is read once for the computation of $A_{ij}x_i$, and each element of y is written once, for a total of $M \times N + M + N$ reads and writes of elements. For single precision, this results in $4 \times (M \times N + M + N)$ bytes of memory accesses. We use this to compute the "relative" memory bandwidth. However, on the CS-2, there is no memory hierarchy and no cache; all memory is SRAM, uniformly distributed between the PEs. An MVM implementation does for each column A_i and each element x_i the following: 1) read y, A_i, and x_i, 2) increment y by the vector-scalar product $A_i x_i$, and 3) write y back to memory. The total is $3M \times N + N$ reads and writes of elements. For single precision, this results in $4 \times (3M \times N + N)$ bytes of memory accesses. We use this to compute the "absolute" memory bandwidth. While the relative bandwidth gives an opportunity to compare CS-2 flat memory throughput against sustained bandwidth on cache-based memory subsystem in a fair manner, the absolute bandwidth demonstrates the actual throughput capability of CS-2 systems. We do not take into account the TLR compression performed on the host in our elapsed time, nor the data transfer. The former requires variants of the work horse of linear algebra, i.e., Singular Value Decomposition (SVD), which is not available in the Cerebras SDK. The latter suffers from overheads due to a slow-bandwidth ethernet interconnect, which may be mitigated with a double buffering mechanism or for instance, the adoption of the Compute Express Link (CXL) standard protocol designed to enable coherency in a heterogeneous computing environment.

6.7 Two Strategies for Strong Scaling Experiments

We elaborate on two strategies for strong scaling experiments for the TLR-MVM kernel. The first one consists of running the eight MVMs for handling the complex datatypes on a single PE. We then choose the stack width to ensure that we max out the SRAM capacity. As we increase the number of PEs, we split the stack width to expose more concurrency, while ensuring an even data distribution for load balancing purposes. This comes at the cost of reducing the arithmetic intensity of the V/U-batch of MVMs, which may degrade parallel efficiency. The second strategy keeps the same stack width parameter to max out the memory capacity of the SRAM memory, but scatters the eight batches of MVMs (i.e., four for each basis) onto eight PEs. While this strategy spares the arithmetic intensity, it engenders an overhead in terms of memory footprint since additional copies of both U and V bases need to be distributed among the PEs. Both strategies require also a reduction step, which is handled by the host to avoid expensive data movement within the fabric interconnect.

7 PERFORMANCE RESULTS

7.1 Impact of Tile Size on Memory Bandwidth

To understand the impact of the tile size on memory bandwidth, we perform a single precision batch of MVMs with constant size N on each PE of a single Cerebras CS-2 system. Figure 14 shows the aggregate memory bandwidth across a fabric of 750×994 PEs, i.e., the maximum the SDK allows to use as some PEs are reserved for routing data on and off the wafer, for a simulated CS-2 and a real CS-2. The simulated memory bandwidth numbers are calculated via the performance model by taking the cycle count for performing a single MVM on a single PE, and scaling up the bandwidth to the entire 750×994 grid of PEs. The bandwidth for the real CS-2 was calculated by performing 10,000 MVMs on each PE, dividing the measured cycle count by 10,000, taking the worst measured cycle count among all 745,500 PEs. We can thus calculate the memory bandwidth by: bytes accessed $\times$ 850MHz / cycle count. The relative bandwidth quickly saturates to 2PB/s as we increase the matrix size, which transitions the batch MVM execution from a memory-bound to a compute-bound operation. This synthetic benchmark using constant ranks is an ideal case and gives a good perspective for our TLR-MVM kernel. The absolute bandwidth based on the actual memory accesses shows the hardware capability by achieving 3X speedup compared to the relative bandwidth.

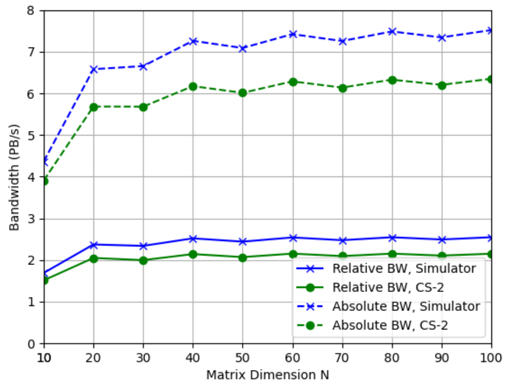


Figure 14: Impact of the tile size on the relative and absolute memory bandwidths for performing a single precision batched MVMs with constant matrix size N.

7.2 Impact of the Stack Width on Hardware Occupancy

We show the performance of the TLR-MVM kernel across six CS-2s on the five validated configurations from Section 6.3. Because these configurations engender larger memory footprints than can fit on a single CS-2, we partition the dataset into six shards by splitting the stack width accordingly to ensure evenly distributed workloads as much as possible. Table 1 identifies the stack width such that each dataset shard nearly fills all PEs.

nb	acc	stack width	PEs used	Occupancy
25	0.0001	64	4417690	99%
50	0.0001	32	4330150	97%
70	0.0001	23	4416383	98%
50	0.0003	18	4445947	99%
70	0.0003	14	4252877	95%

Table 1: Configurations delivering proper MDD accuracy.

7.3 Reporting Bandwidth Metrics

Since the workload is embarrassingly parallel, with no communication or synchronization between shards of the dataset, we can compute total aggregate bandwidth by taking the worst cycle count across all PEs on all systems. Table 2 shows the total number of memory accesses in bytes and worst cycle count across all PEs.

Table 3 presents the aggregate relative and absolute bandwidth across six shards on six CS-2s, as explained in Section 6.5. We achieve the maximum relative sustained bandwidth of 12.26PB/s on six shards (14.44 μs) with $nb = 50$ and $acc = 3\text{e} - 4$, a nearly linear speedup compared to the bandwidth achieved in the ideal synthetic case on a single CS-2 reported in Fig. 14.

nb	acc	Worst cycle cnt	Relative memory accesses	Absolute memory accesses
25	0.0001	21350	2.94E+11	6.85E+11
50	0.0001	19214	2.60E+11	6.71E+11
70	0.0001	19131	2.60E+11	6.89E+11
50	0.0003	12275	1.64E+11	3.89E+11
70	0.0003	12999	1.64E+11	4.06E+11

Table 2: Worst cycle count / # of memory accesses (bytes).

nb	acc	Agg. relative bw (PB/s)	Agg. absolute bw (PB/s)	PFlop/s
25	0.0001	11.24	26.19	3.77
50	0.0001	11.70	30.15	4.60
70	0.0001	11.92	31.62	4.89
50	**0.0003**	**12.26**	**29.05**	**4.16**
70	0.0003	11.60	28.79	4.23

Table 3: Aggregate bandwidth metrics on six shards.

7.4 Performance Scalability

Table 4 shows strong scaling performance on the configuration $nb = 25$ and $acc = 1e-4$, based on the first strong scaling strategy (see Section 6.7) and up to 12, 16, and 20 shards. This corresponds to maximum stack widths of 64, 32, 25, and 19, respectively, to ensure enough concurrency is exposed to keep most PEs busy with work. To compute the aggregate bandwidth, we again use the worst cycle count recorded across all PEs for each shard configuration. We achieve 95% parallel efficiency on 20 shards, due to lower arithmetic intensity of the batched MVMs, as we decrease the stack width.

Table 4 also includes a 48-shard configuration based on the second strong scaling strategy (see Section 6.7). This strategy favors concurrency over memory footprint by replicating the bases such that the real and imaginary components are handled separately via eight batched MVMs, as explained in Section 6.6. The resulting relative bandwidth for the 48-shard configuration is 87.71PB/s (6.698 μs with 97% parallel efficiency).

Shards	Stack width	Agg. relative bw (PB/s)	Agg. absolute bw (PB/s)	PFlop/s
6	64	11.24	26.19	3.77
12	32	22.13	51.17	7.28
16	24	29.28	67.40	9.52
20	19	35.77	81.97	11.51
48	64	87.73	204.51	29.40

Table 4: Aggregate bandwidth for the $nb = 25$ and $acc = 1e-4$ configuration, partitioned into 6, 12, 16, and 20 shards, and the 48-shard case using the first and second strong scaling strategies, respectively.

To further demonstrate the performance of the second strong scaling strategy, we additionally performed runs on Condor Galaxy using this strategy for the $nb = 50$, $acc = 1e-4$ and $nb = 70$, $acc = 1e-4$ configurations. We report metrics of all three configurations, along with number of shards and systems used, in Table 5. We achieve our highest reported relative bandwidth of 92.58PB/s for $nb = 70$, $acc = 1e-4$ across 48 shards.

nb	*acc*	Stack width	Shards	Agg. rel bw (PB/s)	Agg. abs bw (PB/s)	PFlop/s
25	0.0001	64	48	87.73	204.51	29.40
50	0.0001	32	47	91.15	235.04	35.86
70	**0.0001**	**23**	**48**	**92.58**	**245.59**	**37.95**

Table 5: Aggregate bandwidth for the $nb = 25, 50, 70$, $acc = 1e-4$ configurations, using the second strong scaling strategy. While $nb = 25$ and $nb = 70$ required 48 shards and thus 48 systems, $nb = 50$ required only 47 shards.

7.5 Roofline Performance Models

Figure 15 shows the roofline performance models of the minimum configurations of various contemporary vendor offerings required to host our real seismic processing workload in memory, namely six CS-2 systems, compared with one AMD GPU, two NVIDIA GPUs, three NEC vector engines, four Fujitsu ARMs, and one x86 node from AMD and Intel. The arithmetic intensity (Flop/Byte) for the TLR-MVM on Cerebras CS-2 is lower than on the other architectures due to the strategy of splitting real and imaginary parts of complex batched MVMs into four real batched MVMs, as described in Section 6.6. This strategy doubles the number of memory accesses required for TLR-MVM due to the flat memory machine model, as opposed to cache-based architectures. Our communication-avoiding TLR-MVM implementation scores more than three orders of magnitude higher bandwidth than the bandwidth achieved on an AMD MI250X GPU. Figure 16 shows roofline performance models of 48 CS-2 systems, compared with the world's current top 5 supercomputers. In particular, we highlight our relative and absolute sustained bandwidth metrics obtained for our TLR-MVM implementation against the theoretical peak bandwidth of Frontier, Fugaku, LUMI, Leonardo, and Summit. The absolute sustained bandwidth reaches an impressive 245.59PB/s, which demonstrates the hardware capabilities of Cerebras CS-2 systems, as long as the workloads can fit into the local SRAM. Tile low-rank matrix approximation is a versatile algorithmic enabler that has demonstrated the ability to extract high performance from a broad range of hardware solutions. We also provide the relative sustained bandwidth calculated by removing the additional data movements engendered by Cerebras' flat memory architecture, which permits fair comparisons with the top 5 systems. We report 92.58PB/s sustained throughput, more than 3X faster than the aggregated theoretical bandwidth of Leonardo or Summit. We estimate an upper-bound for Fugaku and Frontier based on TLR-MVM with constant ranks (i.e., an ideal scenario without load imbalance issues) using a synthetic dataset to extract the maximum bandwidth on a single A64FX node and a single compute die of MI250X before extrapolating to the respective machine scale. The respective 95.38PB/s and 69.01PB/s are tighter than using the Stream benchmark but still grant benefit relative to our rank-adaptive 92.58PB/s on the Cerebras cluster.

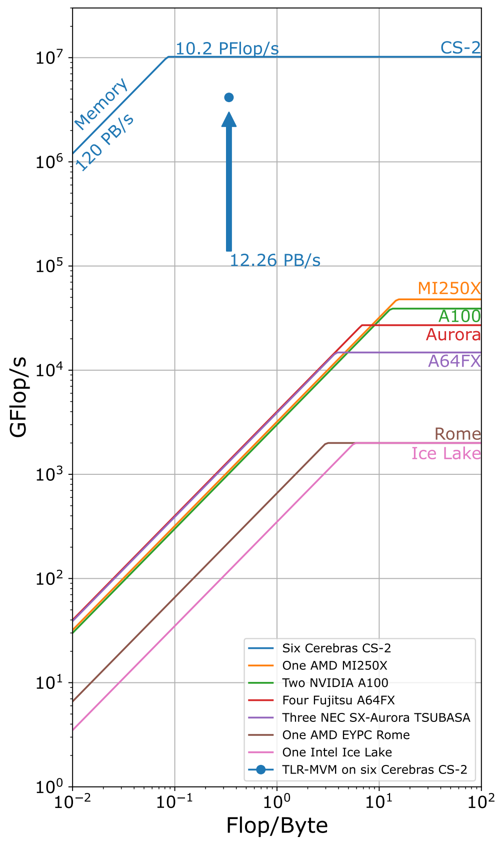


Figure 15: Roofline performance models of 6-shard configuration VS other vendor hardware solutions. TLR-MVM data point is optimal 6-shard configuration of $nb = 50$, $acc = 3\mathbf{e} - 4$.

Our bandwidth score thus outperforms the fastest supercomputer Frontier and is comparable to Fugaku, at a much lower acquisition and operational cost.

7.6 Assessing Power Consumption

We profile the power consumption of a single CS-2 running the worst-case load-balanced shard (out of the six shards) for the real dataset with the configuration of $nb = 25$, $acc = 1e - 4$, and stack width set to 64. The TLR-MVM kernel is run in a loop for 10^8 iterations to collect sustained power statistics, given the five-second coarse granularity of the CS-2 power measurement tool. We report a steady low power consumption of 16kW, which corresponds to 36.50GFlop/s/W in terms of energy efficiency. This contrasts with the higher power profile of matrix-free stencil workloads [25] of close to 23kW. This may be due to the extensive inter-PE communications required via the fabric interconnect for stencil updates, which are not required for our MDD application, thanks to the communication-avoiding memory layout. The 36.50GFlop/s/W of the Cerebras CS-2 for TLR-MVM, which behaves as a compute-bound kernel, compares favorably with the 52GFlop/s/W of Frontier and LUMI systems on the HPL-dominated workload of the Top500/Green500 ranking.

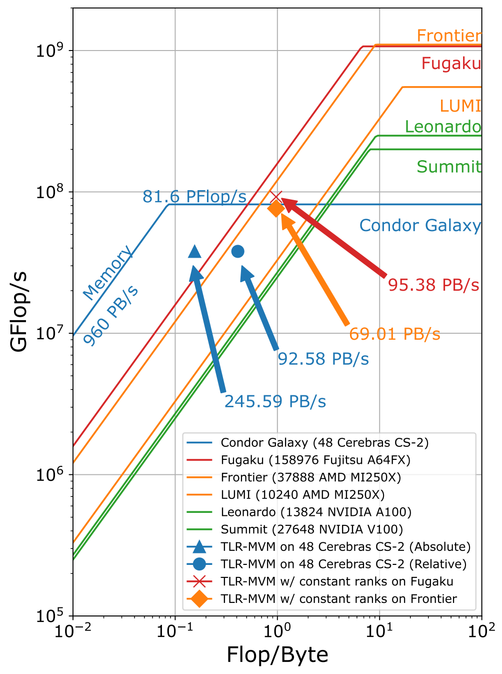


Figure 16: Roofline performance models of 48-shard configuration VS current top 5 supercomputers. TLR-MVM data point is optimal 48-shard configuration of $nb = 70$, $acc = 1\mathbf{e}{-}4$.

8 IMPLICATIONS

The exponential divergence of processing rates and memory bandwidth was recognized before the problem of the "memory wall" was codified in a short note in 1994 [48], bringing to the fore the great trade-off in computer architecture between having essentially unlimited slow memory or having limited fast memory. This exponentially growing gap has traditionally been resolved through a memory hierarchy with an increasing number of levels. We resolve it, instead, by shrinking the problem to reside within a very fast memory with a tunable trade-off in accuracy.

The accuracy of representation of a formally dense tile by its low rank factors (*acc* herein) and the size of the individual tiles (*nb* herein) are parameters under the control of the user. (Herein, we choose both *acc* and *nb* uniformly across all tiles, but this is a simplification that could be relaxed by a user expert in the application of interest.) With knowledge of the other inaccuracies that affect the application (e.g., in acquisition and in downstream usage), *acc* can be adjusted to reduce local rank, instead of defaulting to traditional global full rank with its wasteful implications for data storage, data motion, and computation. With knowledge of the size of the fast memory available to each core and of the rank structure of the operator, *nb* can be adjusted with the stack width so that an arbitrarily large application fits in fast memory – provided only that one has enough PEs.

Now that TLR-MVM has made an impact for the seismic processing of multiple single virtual shots, we want to consider seismic processing of multiple shots simultaneously, by recasting our TLR-MVM kernel into TLR matrix-matrix multiplication (TLR-MMM). This re-exacerbates the memory wall bottleneck and is an open research opportunity that has not yet been addressed at realistic scales.

The future of seismic processing is overall indeed bright, because highly resolving technologies like MDD can increasingly be brought to bear on geological structures too complex to understand well with today’s workhorse alternatives – thanks to investments in architecture that are amortized by their importance to AI workloads. (History rhymes!)

Illustrating for a particular scenario of high contemporary interest to exploration geophysicists on particular hardware of high contemporary interest, *we scale the memory bandwidth wall* for next-generation large-scale seismic processing applications on SRAM-based wafer scale engines by tile low-rank algebraic compression with tuned approximation – and save energy while exploring below the surface for future energy solutions.

ACKNOWLEDGMENTS

For computer time, this research used the resources of the Ibex NVIDIA GPU cluster of the Supercomputing Laboratory (KSL) at King Abdullah University of Science and Technology (KAUST) in Thuwal, Saudi Arabia and the Condor Galaxy-1 CS-2 cluster provided by G42.

REFERENCES

[1] Sameh Abdulah, Hatem Ltaief, Ying Sun, Marc G. Genton, and David E. Keyes. 2018. Parallel Approximation of the Maximum Likelihood Estimation for the Prediction of Large-Scale Geostatistics Simulations. In *2018 IEEE International Conference on Cluster Computing (CLUSTER)*. IEEE, Belfast, UK, 98–108. https://doi.org/10.1109/CLUSTER.2018.00089

[2] Aria Abubakar, Mette Juncker Brædstrup, Haibin Di, Alberto Troya Diaz, Steve Freeman, Simon Hviid, Knud Ho Karkov, Sachin Kriplani, Sunil Manikani, Gwenaëlle Salun, and Tao Zhao. 2021. *Deep learning applications for wind farms site characterization and monitoring.* 3009–3013. https://doi.org/10.1190/segam2021-3583026.1 arXiv:https://library.seg.org/doi/pdf/10.1190/segam2021-3583026.1

[3] Michael Aftosmis, Marsha Berger, and Scott Murman. 2004. Applications of space-filling-curves to Cartesian methods for CFD. In *42nd AIAA Aerospace Sciences Meeting and Exhibit*. AIAA, 1232.

[4] K. Akbudak, H. Ltaief, A. Mikhalev, and D. Keyes. 2017. Tile Low Rank Cholesky Factorization for Climate/Weather Modeling Applications on Manycore Architectures. In *32nd International Conference on High Performance, Frankfurt, Germany*. Springer, 22–40. https://doi.org/10.1007/978-3-319-58667-0_2

[5] N. Al-Harthi, R. Alomairy, K. Akbudak, R. Chen, H. Ltaief, H. Bagci, and D. Keyes. 2020. Solving Acoustic Boundary Integral Equations Using High Performance Tile Low-Rank LU Factorization. In *35th International Conference on High Performance, Frankfurt, Germany*. Springer, 209–229. https://doi.org/10.1007/978-3-030-50743-5_11

[6] Rabab Alomairy, Wael Bader, Hatem Ltaief, Youssef Mesri, and David Keyes. 2022. High-Performance 3D Unstructured Mesh Deformation Using Rank Structured Matrix Computations. *ACM Trans. Parallel Comput.* 9, 1, Article 4 (mar 2022), 23 pages. https://doi.org/10.1145/3512756

[7] Sivaram Ambikasaran, Daniel Foreman-Mackey, Leslie Greengard, David W Hogg, and Michael O’Neil. 2015. Fast direct methods for Gaussian processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 38, 2 (2015), 252–265.

[8] Patrick Amestoy, Cleve Ashcraft, Olivier Boiteau, Alfredo Buttari, Jean-Yves L’Excellent, and Clément Weisbecker. 2015. Improving Multifrontal Methods By Means Of Block Low-Rank Representations. *SIAM Journal on Scientific Computing* 37, 3 (2015), A1451–A1474. https://doi.org/10.1137/120903476

[9] Amirhossein Aminfar, Sivaram Ambikasaran, and Eric Darve. 2016. A fast block low-rank dense solver with applications to finite element matrices. *J. Comput. Phys.* 304 (2016), 170–188.

[10] F. Aminzadeh, J. Brac, and T. Kunz. 1997. 3D Salt and Overthrust models. In *SEG/EAGE Modeling Series, SEG Book Series Tulsa, Oklahoma*, Vol. 1.

[11] L. Amundsen. 2001. Elimination of Free-surface Related Multiples Without Need of a Source Wavelet. *Geophysics* 66 (2001), 327–341. https://doi.org/10.1190/1.1444912

[12] D. Boiero and C. Bagaini. 2020. Up-down deconvolution in complex geological scenarios. *Online EAGE Conference and Exhibition, Extended Abstracts* (2020). doi: 10.3997/2214-4609.2020611021.

[13] Steffen Börm. 2010. *Efficient Numerical Methods for Non-Local Operators: H2-matrix Compression, Algorithms and Analysis*. Vol. 14. European Mathematical Society. https://doi.org/10.4171/091

[14] Kenneth Bredesen, Esben Dalgaard, Anders Mathiesen, Rasmus Rasmussen, and Niels Balling. 2020. Seismic characterization of geothermal sedimentary reservoirs: A field example from the Copenhagen area, Denmark. *Interpretation* 8 (2020). Issue 2.

[15] Qinglei Cao, Sameh Abdulah, Rabab Alomairy, Yu Pei, Pratik Nag, George Bosilca, Jack Dongarra, Marc G. Genton, David E. Keyes, Hatem Ltaief, and Ying Sun. 2022. Reshaping Geostatistical Modeling and Prediction for Extreme-Scale Environmental Applications. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis* (Dallas, Texas). Association for Computing Machinery, 13–24. https://dl.acm.org/doi/10.5555/3571885.3571888 Gordon Bell Prize Finalist.

[16] T. F. Chan. 1987. Rank Revealing QR Factorizations. *Linear Algebra and Its Applications* 88/89 (1987), 67–82.

[17] Eduardo Corona, Per-Gunnar Martinsson, and Denis Zorin. 2015. An $O(N)$ Direct Solver For Integral Equations On The Plane. *Applied and Computational Harmonic Analysis* 38, 2 (2015), 284–317. https://doi.org/10.1016/j.acha.2014.04.002

[18] G. H. Golub and C. F. Van. 2012. *Matrix Computations*. Vol. 3. Third Edition, Johns Hopkins University Press 2012.

[19] SA Goreinov, EE Tyrtyshnikov, and A Yu Yeremin. 1997. Matrix-Free Iterative Solution Strategies For Large Dense Linear Systems. *Numerical Linear Algebra with Applications* 4, 4 (1997), 273–294.

[20] Wolfgang Hackbusch. 1999. A Sparse Matrix Arithmetic Based on $\mathcal{H}$-matrices. Part I: Introduction to $\mathcal{H}$-Matrices. *Computing* 62, 2 (1999), 89–108. https://doi.org/10.1007/s006070050015

[21] N. Halko, P. G. Martinsson, and J. A. Tropp. 2011. Finding Structure with Randomness: Probabilistic Algorithms for Constructing Approximate Matrix Decompositions. *SIAM Rev.* 53, 2 (2011), 217–288. https://doi.org/10.1137/090771806 arXiv:https://doi.org/10.1137/090771806

[22] Y. Hong, H. Ltaief, M. Ravasi, L. Gatineau, and D.E. Keyes. 2021. Accelerating Seismic Redatuming Using Tile Low-Rank Approximations on NEC SX-Aurora TSUBASA. *Supercomputing Frontiers and Innovations* 8 (2021). Issue 2. doi: 10.14529/jsfi210201.

[23] Y. Hong, H. Ltaief, M. Ravasi, D.E. Keyes, and D. Vargas. 2022. Large-scale Marchenko imaging with distance-aware matrix reordering, tile low-rank compression, and mixed-precision computations. *SEG Technical Abstract* (2022).

[24] Yuxi Hong, Hatem Ltaief, Matteo Ravasi, and David E. Keyes. 2022. HPC Seismic Redatuming by Inversion with Algebraic Compression and Multiple Precisions. *Submitted to International Journal of HPC Applications* (2022). http://hdl.handle.net/10754/685271

[25] Mathias Jacquelin, Mauricio Araya-Polo, and Jie Meng. 2022. Scalable Distributed High-Order Stencil Computations. In *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis* (Dallas, Texas) *(SC ’22)*. IEEE Press, Article 30, 13 pages.

[26] H. Jakubowicz. 1998. Wave equation prediction and suppression of interbed multiples. *Society of Exploration Geophysicists* (1998). doi: 10.1190/1.1820204.

[27] Bander Jumah and Felix J. Herrmann. 2012. Dimensionality-reduced Estimation of Primaries by Sparse Inversion. *SEG Technical Program Extended Abstracts* (2012), 3520–3525. https://doi.org/10.1111/1365-2478.12113

[28] Ronald Kriemann. 2013. H-LU Factorization on Many-Core Systems. *Computing and Visualization in Science* 16, 3 (June 2013), 105–117. https://doi.org/10.1007/s00791-014-0226-7

[29] Lin Lin, Jianfeng Lu, and Lexing Ying. 2011. Fast construction of hierarchical matrix representation from matrix–vector multiplication. *J. Comput. Phys.* 230, 10 (2011), 4071–4087. https://doi.org/10.1016/j.jcp.2011.02.033

[30] G. A. Lopez and D.J. Verschuur. 2015. 3D Focal Closed-Loop SRME For Shallow Water. *Geophysical Journal International* 203 (2015), 792–813. doi: 10.1190/segam2015-5921009.1.

[31] Hatem Ltaief, Jesse Cranney, Damien Gratadour, Yuxi Hong, Laurent Gatineau, and David Keyes. 2021. Meeting the Real-Time Challenges of Ground-Based Telescopes Using Low-Rank Matrix Computations. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis* (St. Louis, Missouri). Association for Computing Machinery, New York, NY, USA, Article 29, 16 pages. https://doi.org/10.1145/3458817.3476225

[32] H. Ltaief, Y. Hong, A. Dabah, R. Alomairy, S. Abdulah, C. Goreczny, P. Gepner, M. Ravasi, D. Gratadour, and D. Keyes. 2023. Steering Customized AI Architectures for HPC Scientific Applications. In *International Supercomputing Conference*, Vol. 13948. Springer Lecture Notes in Computer Science (LNCS), 125–143. https://link.springer.com/chapter/10.1007/978-3-031-32041-5_7

[33] D. Lumley. 2019. The Role of Geophysics in Carbon Capture and Storage. In *Geophysics and Geosequestration. Cambridge: Cambridge University Press.*

[34] Christopher Paige and Michael A. Saunders. 1982. LSQR: An Algorithm for Sparse Linear Equations and Sparse Least Squares. *ACM Trans. Math. Software* 8 (1982). Issue 1.

[35] M. Ravasi, T. Selvan, and N. Luiken. 2022. Stochastic multi-dimensional deconvolution. *IEEE Transactions on Geoscience and Remote Sensing* 60 (2022).

[36] Matteo Ravasi and Ivan Vasconcelos. 2021. An Open-source Framework for the Implementation of Large-scale Integral Operators with Flexible, Modern HPC Solutions - Enabling 3D Marchenko Imaging by Least Squares Inversion. *Geophysics* 86 (2021), WC177–WC194. https://doi.org/10.1190/geo2020-0796.1

[37] M. Ravasi, I. Vasconcelos, A. Curtis, and A. Kritski. 2015. Multi-dimensional free-surface multiple elimination and source deblending of Volve OBC data. *77th Conference and Exhibition, EAGE, Extended Abstracts* (2015). doi: 10.3997/2214-4609.201413355.

[38] François-Henry Rouet, Xiaoye S Li, Pieter Ghysels, and Artem Napov. 2016. A Distributed-memory Package for Dense Hierarchically Semi-separable Matrix Computations Using Randomization. *ACM Transactions on Mathematical Software (TOMS)* 42, 4 (2016), 27. https://doi.org/10.1145/2930660

[39] H Sagan. 1994. *Space-Filling Curves*. Springer. https://doi.org/10.1007/978-1-4612-0871-6

[40] Justin Selig. 2022. *The Cerebras Software Development Kit: A Technical Overview*. Technical Report. Cerebras.

[41] J. van der Neut and F. Herrmann. 2013. Interferometric redatuming by sparse inversion. *Geophysical Journal International* 192 (2013), 666–670. doi: 10.1093/gji/ggs052.

[42] G. J. van Groenestijn and D. J. Verschuur. 2009. Estimating Primaries by Sparse Inversion and Application to Near-offset Data Reconstruction. *Geophysics* 74 (2009), 1MJ–Z54. Issue 3. https://doi.org/10.1190/1.3111115

[43] D. Vargas, I. Vasconcelos, M. Ravasi, and N. Luiken. 2021. Time-domain multidimensional deconvolution: A physically reliable and stable preconditioned implementation. *Remote Sensing* 13 (2021). Issue 18.

[44] I. Vasconcelos, M. Ravasi, A. Kritski, J. van der Neut, and T. Cui. 2017. Local, reservoir-only reflection and transmission responses by target-enclosing extended imaging. *SEG Technical Program Expanded Abstracts* (2017), 5289–5293. doi: 10.1190/segam2017-17730961.1.

[45] D. J. Verschuur. 1992. Surface-related Multiple Elimination in Terms of Huygens Sources. *Journal of Seismic Exploration* 1 (1992), 49–59.

[46] K. Wapenaar, J. Thorbecke, J. van der Neut, F. Broggini, E. Slob, and R. Snieder. 2014. Marchenko Imaging. *Geophysics* 79 (2014), WA39–WA57. Issue 3. https://doi.org/10.1190/geo2013-0302.1

[47] K. Wapenaar, J. van der Neut, E. Ruigrok, D. Draganov, J. Hunziker, E. Slob, J. Thorbecke, and R. Snieder. 2011. Seismic interferometry by crosscorrelation and by multidimensional deconvolution: A systematic comparison. *Geophysical Journal International* 185 (2011), 1335–1364. doi: 10.1111/j.1365-246X.2011.05007.x.

[48] William Wulf and Sally McKee. 1995. Hitting the Memory Wall: Implications of the Obvious. *ACM SIGARCH Computer Architecture News* 23 (1995), 20–24. doi: 10.1145/216585.216588.

[49] Kezhong Zhao, M.N. Vouvakis, and Jin-Fa Lee. 2005. The adaptive cross approximation algorithm for accelerated method of moments computations of EMC problems. *IEEE Transactions on Electromagnetic Compatibility* 47, 4 (2005), 763–773. https://doi.org/10.1109/TEMC.2005.857898