Case Study

High-Performance Computing
Intel® Xeon® processors, Intel® Data Center GPUs

Devito Codes Boosts Productivity, Automates HPC Code Generation

Devito Codes generates optimized HPC code from a Python domain-specific language (DSL), enabling scientists to create high-performance finite difference solutions from mathematics.

At a glance

Devito Codes can increase productivity for any discipline that relies on solving partial differential equations using finite differences.

- Automatically generates HPC software optimized for the hardware it is run on.
- Creates highly optimized kernels for heterogeneous hardware, including all major CPUs and GPUs.
- Generates highly optimized code for Intel® Data Center GPUs. SYCL support offers other intriguing options such as executing on FPGAs.
- Supports mixed precision (FP-16/ FP-32) for up to 2x smaller datasets and 2x performance increase.[1]
- Developed for wave-equation based imaging (seismic, medical ultrasound). Open source version available for all disciplines.

The HPC trilemma: choosing between performance, portability, and productivity

The physical sciences describe the world with applied mathematics. Seismology, electromagnetics, and fluid dynamics all rely on techniques like partial differential equations and finite-difference simulations to model physical phenomena.

Python and its scientific ecosystem of libraries like NumPy, SciPy, and MatPlotLib, give scientists and researchers frameworks to develop sophisticated domain-specific solutions using relatively few lines of readily accessible code. It is this versatility and productivity that has propelled Python into one of the leading languages in science and engineering today. In what other programming language can you literally `import antigravity`?

While pure Python may be slower than compiled languages, its primary role in high-performance computing (HPC) is as an accessible interface to highly optimized libraries written in lower-level, compiled languages such as C, C++, Fortran, and even CUDA, SYCL, or HIP for GPU support. Python's scientific libraries, like NumPy and SciPy, leverage these compiled backends to deliver high performance while maintaining Python's ease of use. However, this model still depends on the availability of optimized HPC code for domain-specific tasks, which often requires specialized expertise and resources to develop and maintain.

The Devito solution: let scientists be scientists

The Devito Codes team aimed to bring just-in-time (JIT) compilation of HPC-grade, finite-difference code to the Python ecosystem. With Devito, scientists can work within Python's symbolic and mathematical framework (SymPy), writing complex partial differential equation solvers and goal-driven optimization problems, and seamlessly generate parallelized, hardware-optimized HPC code for all major CPU and GPU architectures.

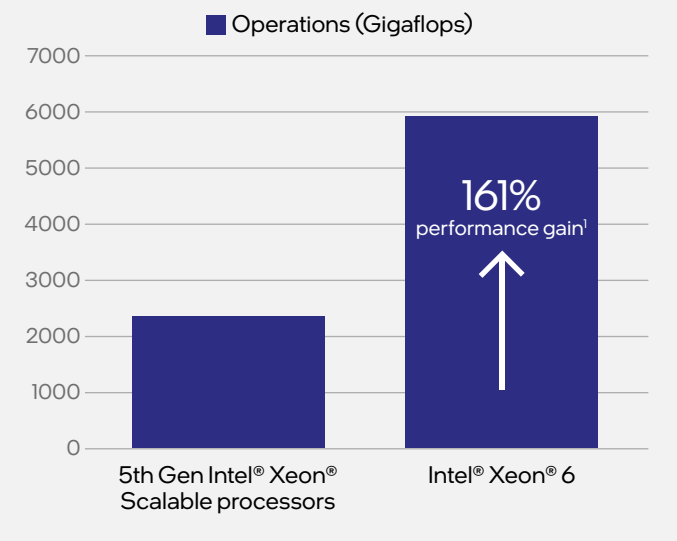
"Most people adopt Devito initially for the productivity boost," says Gerard Gorman, CEO and Co-founder of Devito Codes. "It enables a unique form of rapid prototyping, letting them treat it as a computational laboratory."

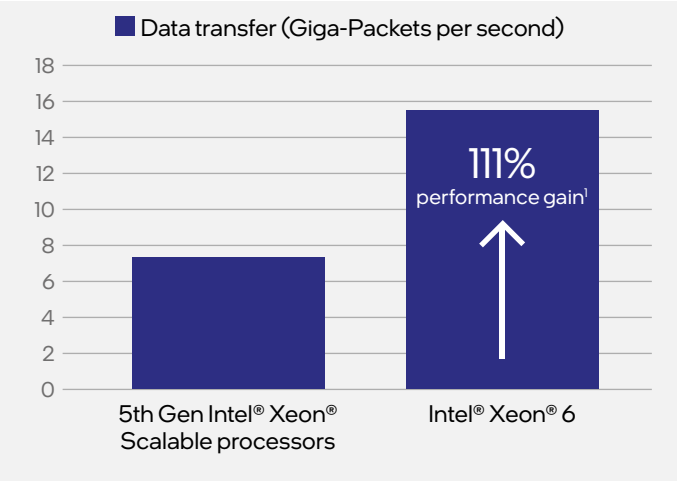
Creating a Python framework that can produce high-performance kernels for simulations, inversion, and optimization tasks across diverse hardware isn't straightforward. Devito integrates multiple HPC technologies, including OpenMP for shared memory systems, OpenACC for accelerators, and MPI for parallelism

and portability. Advanced optimizations require hardware-specific tuning with specialized languages like CUDA, HIP, and SYCL. Devito Codes applies nearly every known optimization technique for structured computation and continuously integrates new advancements, ensuring that performance gains compound over time —often rivaling or even surpassing expert hand-tuned commercial solutions.

By taking on the complexity of HPC code development, Devito Codes enables users to transfer projects across systems effortlessly, leveraging all available computing resources. HPC operators and service providers can maintain current infrastructure, expand with heterogeneous arrays, and upgrade significantly—all while ensuring that workloads remain compatible and performant.

Devito Codes see major gains with Intel® Xeon® 6 (higher is better)





A Devito Codes kernel for an acoustic anisotropic propagator for tilted transverse isotropy (TTI) model (a typical energy industry seismic model) illustrates Intel® Xeon® 6 performance gains over 5th Gen Intel® Xeon® Scalable processors.[1]

DevitoPRO

Devito began as part of an Intel Parallel Computing Centre initiative led by Professor Gerard Gorman at Imperial College London. The initial project created high-performance, open-source software for seismic imaging. As the project matured into a true optimizing compiler for HPC workloads, the team launched DevitoPRO, an enterprise edition with proprietary features, advanced performance optimizations, and commercial support.

DevitoPRO primarily serves exploration geophysics in the energy industry. In addition to compiling highly optimized, portable code for seismic simulations authored in Python, DevitoPRO provides high-performance propagators and gradient operators for Full Waveform Inversion (FWI) and Reverse Time Migration (RTM). DevitoPRO also provides technical support, training, custom software development, and hardware-specific optimization for clients.

Devito continues to provide general-purpose symbolic and compiler software technology as open-source, patent-free software for researchers in academia and industry.

"With DevitoPRO, a geophysicist can take an algorithm from a research paper and implement it in an afternoon —a task that would normally take months of coding and optimization. This rapid turnaround allows teams to experiment and innovate at an unprecedented pace, accelerating the testing and deployment of new algorithms."

— Mathias Louboutin
Senior Solutions Architect, Devito

Expanding code portability with SYCL and Intel

Traditionally, creating portable code that can run across heterogeneous processors required compiling unique kernels for each hardware type—CUDA kernels for NVIDIA GPUs, HIP kernels for AMD GPUs, and C/C++ for x86 and RISC CPUs. In recent years, SYCL has given HPC programmers a new, multi-platform option for compiling optimized HPC code.

SYCL is a cross-platform parallel C++ abstraction layer with APIs that can find and manage data resources and code execution on mixed devices from multiple vendors including CPUs, GPUs, and FPGAs. SYCL is the foundation for oneAPI and Data Parallel C++, which Intel has implemented on Intel® Data Center GPUs.

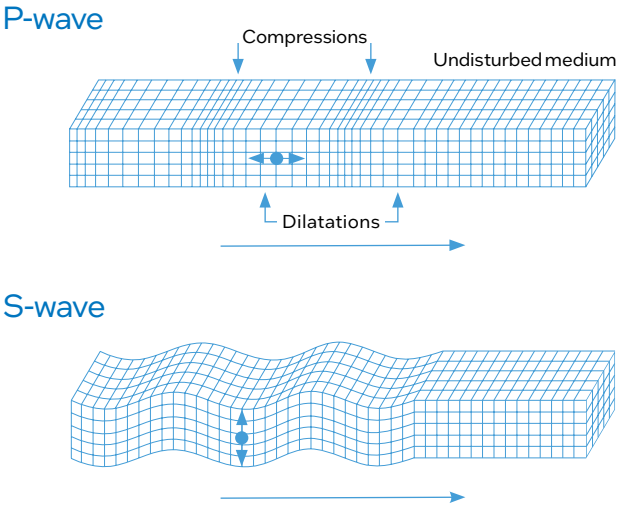
Devito Codes and Intel engineers worked together to bring SYCL code generation to DevitoPRO, including specific optimizations for Intel® Data Center GPU Max 1100 and 1550 series accelerators. To deploy, DevitoPRO users simply target Intel Data Center GPUs for just-in-time compilations that reap the performance benefits of SYCL.

"Devito's fusion of symbolic computation and advanced compiler technology ensures reliable, verifiable, optimized code generation—critical for mission-focused mathematical software. While generative AI code lacks this level of precision and dependability, combining both technologies can unlock even greater productivity in developing and testing new algorithms."

— Gerard Gorman,
CEO and Co-founder of Devito Codes

Expanding performance for elastic wave seismology with mixed precision computing

Seismic waves travel in two forms: longitudinal primary waves (P-waves) and transverse secondary waves (S-waves). Modeling P-waves—a method seismologists call acoustic analysis—is *relatively simple* mathematically and computationally because the wave energy and particle motion travel in the same dimension. Modeling P- and S-waves together—which seismologists call elastic analysis—complicates things geometrically.



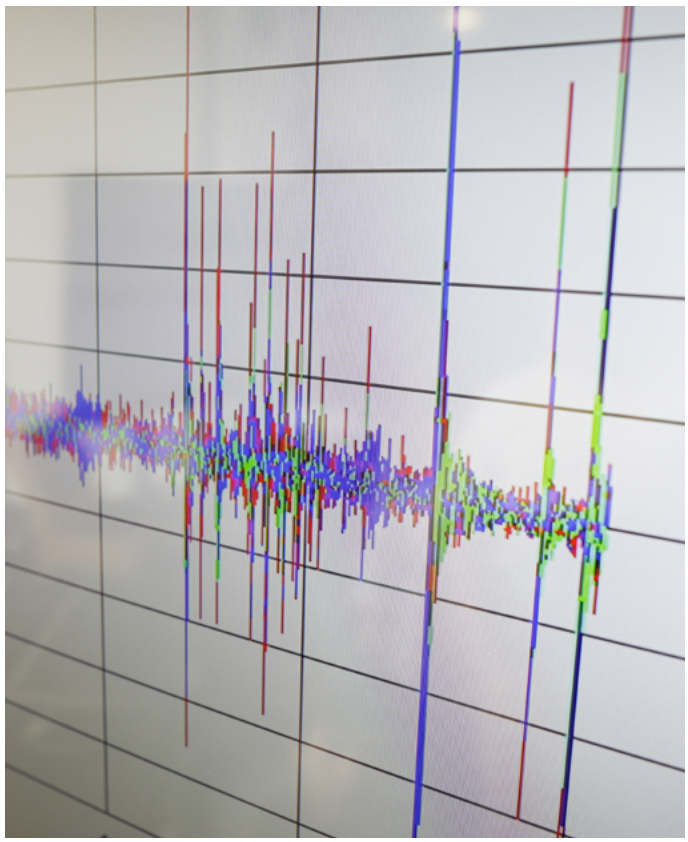
Seismic waves come in two forms: longitudinal Primary (P) waves and transverse Secondary (S) waves

Because the two waveforms move in three dimensions at ninety-degree angles, describing them requires more wave equations with more components. Elastic analysis also requires much higher resolution to produce accurate results, which means gathering vastly more data.

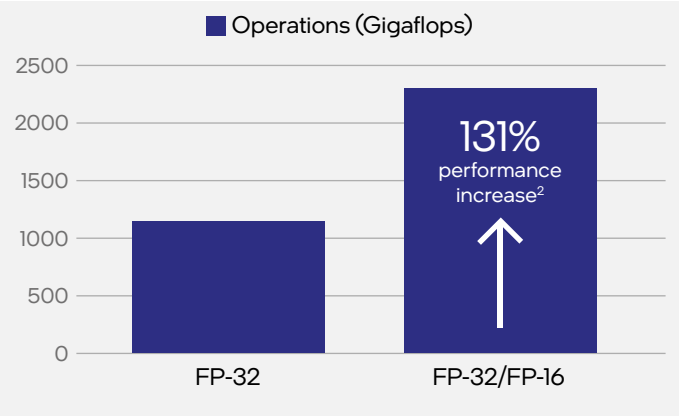
"If you are running elastic, then your memory footprint is going to be, in the best case scenario, between four and five times larger than in the case of acoustic," says Fabio Luporini, CTO and Co-founder of Devito Codes. "It's simply because of the physics. You are modeling more wave fields simultaneously in a coupled partial differential equation, which you have to keep in memory."

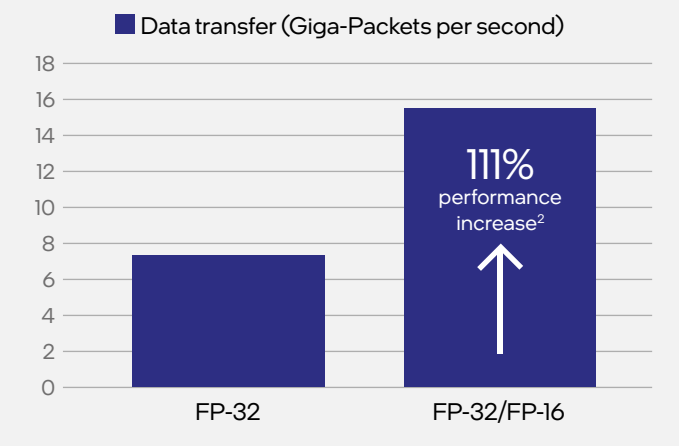
Devito Codes is developing mixed-precision algorithms, an AI computing technique, to make elastic computing workloads possible on current-generation hardware. Workloads that can afford a small loss in precision are converted from FP-32 (32-bit floating point) to a carefully designed mix of FP-32 and FP-16 (16-bit floating point), which represent the same values with half the memory. In the world of elastic analysis, halving a petabyte dataset to 500TB produces cascading performance boosts at every step from memory and I/O management to writing snapshots to disk.

FP-16 workloads also process faster on CPUs and GPUs that support mixed precision, like Intel® Xeon® 6 processors and Intel Data Center GPUs. In initial tests, elastic wave analysis running in mixed precision with Devito Codes produced a up to 2x performance increase,[1] the equivalent of a step change in performance with no hardware upgrades.



Devito Codes doubles performance with mixed precision[2] (higher is better)





Devito Codes tests show shifting to mixed precision (FP-16/FP-32) increases performance 2x and reduces memory footprints 2x, resulting in significantly faster throughput.[2]

Solution ingredients

Devito Codes Python framework for high-performance finite-difference simulations and automate HPC code generation

SYCL C++ cross-platform abstraction layer for programming heterogeneous parallel computing

Intel® Xeon® 6 processors

Intel® Data Center GPUs

Intel® Advanced Matrix Extensions (Intel® AMX)

Conclusion: every performance increment counts

Devito Codes and Intel continue to refine and optimize compiler technologies to extract more performance from heterogeneous HPC systems for finite difference simulations. For Devito's open-source and DevitoPRO clients, the work is indispensable.

"Processing seismic data for subsurface imaging can cost millions in compute expenses per project," says Gorman. "So we need to squeeze out every last percent of performance because time is money."

For the latest updates, visit Devito Codes on GitHub or devitocodes.com.

[1] Results from Devito Codes tests with identical seismic model workloads on 5th Gen Intel® Xeon® Scalable processors and Intel® Xeon® 6 processors. Full node test conducted November 1, 2024 Single NUMA test conducted November 4, 2024. Further details available here: https://www.devitocodes.com/granite

Software configuration - Ubuntu 24.04.1 LTS; Python 3.12.3; Intel® oneAPI DPC++/C++ Compiler 2024.2.1 (2024.2.1.20240711)

Hardware configurations - 5th Gen Intel® Xeon® Scalable processor specifications: Architecture: x86_64; CPU op-mode(s): 32-bit, 64-bit; address sizes: 52 bits physical, 57 bits virtual; byte Order: Little Endian; CPU(s): 256; on-line CPU(s) list: 0-255; vendor ID: GenuineIntel; model name: Intel® Xeon® Platinum 8592+; CPU family: 6; Model: 207

Intel® Xeon® 6 specifications: Architecture: x86_64; CPU op-mode(s): 32-bit, 64-bit; address sizes: 52 bits physical, 57 bits virtual; byte Order: Little Endian; CPU(s): 512; on-line CPU(s) list: 0-511; vendor ID: GenuineIntel; model name: Intel® Xeon® 6980P; CPU family: 6; Model: 173

[2] Results from Devito Codes test running an isotropic elastic finite difference solver (3D elastic solver) with grids at 1024 x 2048 x 1024 .The super-linear >2x speedup results from reduced bandwidth/latency of Halo exchange with half-precision(fp16) at every time step. Further information available here: https://www.devitocodes.com/granite

Additional modeling with single NUMA zone (single MPI rank without domain decomposition) resulted in FP-32: 381.43 GFlops, 1.34 GPts/s; mixed precision FP-32/FP-16 645.82 GFlops. 2.26 GPts/s.

Hardware configuration - Intel® Xeon® 6 specifications: Architecture: x86_64; CPU op-mode(s): 32-bit, 64-bit; address sizes: 52 bits physical, 57 bits virtual; byte Order: Little Endian; CPU(s): 512; on-line CPU(s) list: 0-511; vendor ID: GenuineIntel; model name: Intel® Xeon® 6980P; CPU family: 6; Model: 173

0225/SB/CAT/PDF ♻Please Recycle 363010-001EN