# *Parallel Universe* Issue #31 Abstracts

**Feature:**

## FPGA programming with OpenCL™

### Knowing How to Program an FPGA is a Skill you Need—and Here's How to Start

Field programmable gate arrays (FPGAs) are exciting because they offer high performance, with low latency and power efficiency. These benefits are realized through their massive parallel capabilities coupled with reconfigurability. An FPGA provides a reconfigurable sea of gates on which anyone can:

- Design their own custom hardware accelerator
- Deploy it for a single application
- Quickly reconfigure the device as a new accelerator for a different application

This article explains how you can make your own designs and use them to accelerate your applications. Hardware engineers have long used FPGAs in place of ASICs in a variety of applications. Historically, configuring them (programming) has been done with high-level definition languages like Verilog* or VHDL*. These design methodologies are familiar and useful for hardware engineers but completely foreign to software developers.

New tools centered around OpenCL* bridge this gap to bring the benefits of FPGA hardware platforms to software developers. Using these new tools, we can view FPGAs as highly configurable devices that can transform custom algorithms, written in a C-like syntax, into fast and power-efficient hardware. The flexibility and performance of these solutions are vast thanks to the latest generation of FPGA devices that offer up to 10 TFLOPs of performance.

## Accelerating Eigen Math Library for Automated Driving Workloads

### Meeting the Need for Speed with Intel® Math Kernel Library

Automated driving workloads include several matrix operations at their core. Sensor fusion and localization algorithms—such as different versions of the Kalman filter—are critical components in the automated driving software pipeline. The Intel® Math Kernel Library (Intel® MKL) is a powerhouse of tuned subprograms for numerous math operations, including a fast DGEMM. The automated driving developer community typically uses Eigen* a C++ math library, for matrix operations. In addition to Intel MKL, LIBXSMM*, a highly-tuned library for high-performance matrix-matrix multiplications, shows potential to speed up matrix operations. In this article, we investigate and improve the performance of native Eigen on matrix multiplication benchmarks and the extended Kalman filter (EKF) by using Intel MKL and LIBXSMM with GNU* and Intel® compilers on the Intel® Xeon® processor.

## Speeding Algebra Computations with Intel® Math Kernel Library Vectorized Compact Matrix Functions

### Maximizing the Performance Benefits of the Compact Data Layout

High-performance computing (HPC) and machine learning applications often require performing computations on large groups of very small matrices. The performance of these operations scales well with increased core counts using OpenMP* loops or batched linear algebra functions—but vector utilization in these methods is limited. BLAS (Basic Linear Algebra Subprograms) and LAPACK (Linear Algebra Package) functions traditionally operate on data stored in a standard layout, with matrices in row (or column) major order. Storing data in non-standard layouts that allow for cross-matrix vectorization can provide a significant speedup in BLAS and LAPACK functions for small-sized matrices.

In this article, we describe a new data layout, called Compact, which interleaves matrices in blocks according to the architecture's SIMD (single instruction, multiple data) vector length. We demonstrate its benefits for linear algebra computations such as matrix multiplication and LU factorization—observing up to 60X and 20X speedups, respectively, for double precision over optimized kernels written for standard data layouts. Intel® Math Kernel Library (Intel® MKL) 2018 includes new Compact functions that support and maximize the performance benefits of the compact data layout.

## Boosting Java* Performance in Big Data Applications

### How New Enhancements Enable Faster and Better Numerical Computing

Since its introduction more than two decades ago, Java* has become a popular enterprise language. Embraced initially for its write-once, run-anywhere property, its usage increased due to the relative ease of building new applications by integrating existing Java projects. The growth of several open-source Java application stacks such as Apache Hadoop* has added to this growth. Fundamentally, developers view Java as enhancing their productivity.

New enhancements to Java enable faster and better numerical computing. The JVM has been improved to use the Intel AVX FMA instructions in the implementation of the `Math.fma()`. This results in significant performance improvements of matrix multiplications, the basic workhorse of HPC and AI applications.

## Gaining Performance Insights Using the Intel® Advisor Python API

### Getting Good Data to Make Code Tuning Decisions

Good design decisions are based on good data. Intel® Advisor is a dynamic analysis tool that's part of Intel® Parallel Studio XE, Intel's comprehensive tool suite for building and modernizing code. Intel Advisor answers all essential performance questions and lets you collect insightful program metrics on the vectorization and memory profile of your application. And, besides providing tailored reports using the GUI and command line, Intel Advisor now gives you the added flexibility to mine a collected database and create powerful new reports using Python*.

When you run Intel Advisor, it stores all the data it collects in a proprietary database, which you can access using the new Python API, which provides a flexible way to report on useful program metrics. This article explains how to use this new functionality.

## Welcome to the Intel® AI Academy

### AI Education for All

With the rise of artificial intelligence (AI), it's no surprise that Intel has a clear vision and approach to engaging with those at the cutting edge of innovation—developers and academics. Whether you're just starting out or already an expert, the Intel® AI Academy provides learning materials, tools, and technology to help shape, create, build, and develop the future of AI.