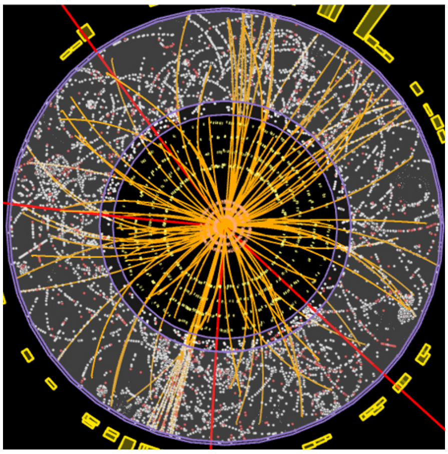
ACTS: CPU Race for Particle Hunting

Taras Kolomatski

Introduction

Experimental particle physics is as a process of reconstruction, as direct detection of the physical phenomena of interest is sometimes impossible. For example, the Higgs boson has a mean lifetime of 10^{-22} seconds, which theoretically limits the expected distance it can travel to 3×10^{-14} metres - far too short to reach a detector's sensitive elements. What is detected are changes in the statistical patterns of the particles that we do measure, originating from the Higgs boson's decay products. However, there are also limitations to the information we can extract from particles that do reach a detector's sensitive elements. For example, we are not able to accurately measure the momentum of electrically neutral particles. Roughly, the game is approximately as follows: we tabulate the measurements that we can, infer missing terms from conservation laws, piece together which particles could give those terms, and finally perform a statistical analysis to infer that certain decays occurred.

Long-lived charged particles are a rich source of information as we can track the curves that they follow under a strong magnetic field. Inner layers of the detector are sensitive media that respond to, but do not significantly impede, the particles that cross them. Thus, the data we obtain are several snapshots of points on this curve, to some error bars, followed by the energy deposition pattern of particles in the calorimeter, an outer layer which stops most of them. By disrupting the particles more significantly, this outer layer is able to make observations that the inner tracking layers do not. The process of recovering the path followed by particles in the inner tracking layers is called track reconstruction. It involves first analysing the inner layers and making a guess at which curves could give rise to that collection of points, and then testing these predictions against the measurements made in the outer layers.



Track predictions fit against observation at the ATLAS detector in the LHC.

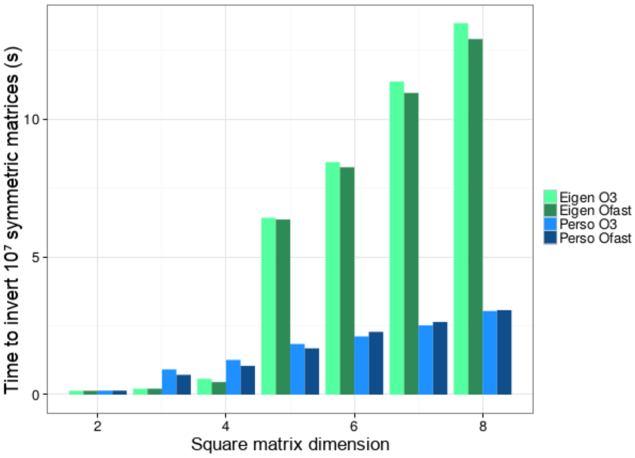
ACTS is an open source library aiming to provide experiment-agnostic tools for track reconstruction, building on the successful statistical methods in the software of previous experiments. Such data analysis software was previously highly experiment-specific. The adoption of efficient universally used libraries is critical as although upcoming accelerator upgrades (e.g. HL-LHC and FCC) are expected to result in orders of magnitude increase in data set sizes, public funding shortages on experimental fundamental physics research result in the prediction that data analysis costs will stand at at 10x budget overrun.

The Proposal

I will work on a part of the ACTS codebase that implements the non-linear Kalman filter algorithm, used for preliminary fitting (simulating and testing of track hypotheses). Track hypotheses live in a five dimensional parameter space, and the implementation makes heavy use of linear algebra on symmetric 5×5 matrices. Many of the steps involve independently applying some operation to a large collection of matrices, and is hence amenable to vectorisation by use of SIMD instructions. Additionally, the symmetric 5×5 case necessarily admits optimisations that are not efficient for square matrices of larger size.

ACTS chose the `Eigen` library for its linear algebra needs, which benchmarking showed to be a clear choice among its contenders. However, even this library's handling of low-dimensional matrices remains imperfect. Matrix operations are merely translated into loops which the compiler is expected automatically vectorise, but may not, leading to large performance fluctuations across compiler configurations. And the library's algorithms have not been particularly optimised for efficient handling of smaller matrices.

Previously, a research intern with the ACTS team made a processor specific low level implementation of square matrix multiplication and inversion algorithms which demonstrated a substantial improvement over the program produced by `Eigen`. They produced the following table in the case of inversion. It represents time taken to invert 10^7 matrices as compared with `Eigen`, on two different compiler optimisation settings (-O3 and -Ofast).



Unfortunately, this substrate-specific low level code could not be integrated. I will aim to provide those improvements in a cross-platform manner. Specifically, I will investigate replacing the use of `Eigen` with the C++ library `xtensor`. `xtensor` aims to provide the functionality and feel of NumPy, a Python package with which HEP physicists are likely to be familiar, and may thus ease algorithmic contributions to the ACTS project. The threefold benefit of adopting it would thus be, more extensible code, potentially better cross-platform vectorisation, and room for my contribution of more effective algorithms for handling of small matrices, with emphasis on 5×5.

The latter objective has been mathematically neglected. 5×5 is the awkward boundary between the ubiquitous square matrices of smaller size carefully scrutinised by people working on computer graphics, and large matrices for which the primary consideration is asymptotics over constants.

Timescales, Purpose, and Deliverables

Please note that the examination period of my current degree clashes for the first three weeks of the coding period. I believe that the remaining two months is sufficient to produce a substantial contribution, during which I will consider this a full time job. I however expect to be able to intimately familiarise myself with `xtensor` and the ACTS codebase prior to June 6^{th}.

The first deliverable would be an implementation of all that is involved in the Kalman filter algorithm, making effective use of the SIMD instruction capabilities provided by `xtensor`. I do not anticipate that producing clean and well documented code for this would take longer than a month, as I can identify a short list of the operations that I need to support, beyond which is a matter of negotiating with the library.

Subsequently, I would spend the rest of the project benchmarking, optimising, and considering the integration of this `xtensor` project into the ACTS code base proper. I expect that I would also make a mathematical contribution to constant cutting on some algorithms.

Finally, in the course of completing the qualifying task, I acquainted myself with `xtensor`. The documentation in several sections could be substantially improved. It would also be nice to let `xtensorf` objects support non-resizing views. Consequently, I may potentially make several contributions to `xtensor` in the course of this project.

My Background

Undergraduate: University of Waterloo, BMath in Pure Mathematics

Master's: University of Cambridge, MASt in Pure Mathematics (in progress)

PhD: Stony Brook University (starting in September)

At Waterloo, my CS coursework consisted of:

- Two semesters of freshman advanced computer science courses taught in Racket, a Lisp dialect.
- A course on object oriented design, C++, and the Unix command line.
- A basic compilers course, in which I wrote a rudimentary compiler for a C-like language in Racket. I have a half-finished reworking of this project currently in limbo, implementing optimisation through conversion to SSA form.
- An algorithms course, where I otherwise have done contest programming since high school.
- A course on real-time operating systems in my final term. My partner and I wrote a bare-metal embedded microkernel and operating system for controlling real life Märklin trains, this was primarily in C. For the kernell, we obsessed over excess optimisation of our scheduler and context switch (this was for an ARM4 processor), so I have spent a lot of time thinking about low level code.

Beyond coursework, I have frequently participated in Hackathons. Achievements include: 'Most Functional' award at HackWaterloo in 2013. Finalist at WaterlooHacks in 2016. First place Wear Hacks KW 2017. My Devpost is: Antares. I have used git for version control in both classes and hackathons.

My primary interest is mathematics. I have a strong background in analysis and topology/geometry. I completed eight graduate courses during my undergrad, presented as several conferences, and for the past three summers worked on research projects. I have one joint publication: [1711.09526].

I believe that my background makes me a strong match for this project because of the combined challenge of understanding and optimising code at a low level and the mathematical content. Peripherally, I have also studied as much physics as I have computer science and would be very happy to produce code that's used by researchers in experimental fundamental physics.

My Basic Information

Email: `REDACTED`

Homepage: `REDACTED`

Nationality: Canadian

Timezones: `REDACTED`

Availability: `REDACTED`

Permanent Mailing Address: `REDACTED`

Emergency contact: `REDACTED`