Faster matrix algebra for ATLAS

(Project Proposal by STFC Rutherford Appleton Laboratory in 2018,http://hepsoftwarefoundation.org/gsoc/2018/proposal_ATLASEigen.html)

Introduction

ATLAS is one of two general-purpose detectors at the Large Hadron Collider(LHC). It investigates a wide range of physics, from the search of Higgs boson particles to extra dimensions and particles that could make up the dark matter. Dark matter can be detected by the change in energy, momentum after the collision of the particles at LHC.

Beams of particles from the LHC collide at the center of the ATLAS detector which creates new particles, which fly out from the collision point in all directions. Six different detecting subsystems arranged in layers around the collision point record the paths, momentum, and energy of these particles, allowing them to be individually identified.

The interactions due to collision of particle in the ATLAS detectors create an enormous flow of data. Complex data-acquisition and computing systems are then used to analyze the collision events recorded.

When the LHC is operating, 40 million packets of protons collide every second at the center of the ATLAS detector. Every time there is a collision, the ATLAS Trigger selects interesting collisions and writes them to disk for further analysis.

Background

To track these trajectory of charged particles and their collisions a large number of alignment parameters are required which can range from **10^4** to **10^5**. The alignment precision depends on the amount of data used in the alignment and it may be necessary to use millions of tracks. The amount of data and the number of parameters, with weakly defined or undefined degrees of freedom, represents a problem even for today's computers. Several alignment algorithms as well as heavy CPU sonsumption is used to optimize these parameters. One such method requires the solution of a system of linear equations. To store these equations the symmetric *n*-by-*n* matrix is used.Symmetric matrices make up a large fraction of the matrices used in track reconstruction. ATLAS Tracking Software makes heavy use of matrix algebra, implemented with the **Eigen** library. The CLHEP maths library was previously used throughout the ATLAS software, but after investigating potential alternatives, the decision was made to move to Eigen. Eigen was chosen since it offered the largest performance improvements for ATLAS.

Eigen is a C++ template library for linear algebra: matrices, vectors, numerical solvers, and related algorithms. It is a fast, robust, open source linear algebra library, used by e.g. the Google Tensorflow, Ceres Large Survey Synoptic Telescope projects as well as ATLAS.

Currently, Eigen does not provide any macros or templates to store Symmetric Matrices. Due to which a huge amount of storage space is wasted to store large data of particles. So the current need is to store the symmetric metrices such that storage space is reduced and all the operations which are supported for a Eigen Matrix can be applied on a Symmetric Matrix.

Literature Survey

1. **The CLHEP project** - A Class Library for High Energy Physics

   It is C++ library that provides utility classes for general numerical programming, vector arithmetic, geometry, pseudorandom number generation, and linear algebra, specifically targeted for high energy physics simulation and analysis software.

   It was proposed by Swedish physicist Leif Lönnblad in 1992 at a Centre for High Energy Physics conference.

2. **ROOT**

   ROOT is an object-oriented program and library developed by CERN. It was originally designed for particle physics data analysis and contains several features specific to this field, but it is also used in other applications such as astronomy and data mining.

   ROOT is optimized for describing small matrices and vectors and **It can be used only in problems when the size of the matrices is known at compile time**, like in the tracking reconstruction of physics experiments.

   The Matrix Class of ROOT(SMatrix)

   The template class **ROOT::Math::SMatrix** represents a matrix of arbitrary type with nrows x ncol dimension. The class has following 4 template parameters, which define at compile time, its properties:

   1. type of the contained elements, T, for example float or double;
   2. number of rows;
   3. number of columns;
   4. representation type. This is a class describing the underlined storage model of the Matrix. Presently exists only two types of this class:
      1. **ROOT::Math::MatRepStd** - for a general *n* rows x *n* cols matrix.
      2. **ROOT::Math::MatRepSym** - for a symmetric matrix of size NxN. This class is a template on the contained type and on the symmetric matrix size N. It has as data member an array of type T of size N*(N+1)/2. It provides basic matrix and vector functions such as matrix-matrix, matrix-vector, vector-vector operations, plus some extra functionality for square matrices, like inversion and determinant calculation.

   Criticisms of ROOT include its difficulty for beginners, as well as various aspects of its design and implementation. Frequent causes of frustration include extreme code bloat,

heavy use of global variables, and a perverse class hierarchy. But as of now, Root is mainly used for ploting and data analysis at CERN.
Issues like , class structure issues, functionality issues,