# Plans for migrating from doubles to floats in specific data objects

Giuseppe Cerati (FNAL)

LArSoft Coordination Meeting

Nov. 6, 2018

# Introduction

- MicroBooNE is about to start its largest a production campaign
  - probably other experiments will be doing the same in the future
- Staging of files from tape to disk is currently the biggest bottleneck in the production and analysis workflows
- Need to minimize the size of the files!
- After dropping all non-vital collections, we can reduce the file size by >20%
- The way to do this is to change the type of selected data product members from double to float
  - this includes derived types of course
- Here focus on recob::Tracks, anab::Calorimetry, recob::SpacePoint, recob::MCSFitResult, recob::TrackFitHitInfo
  - other data products are either already storing floats (e.g. Hits), or they are not easy to migrate and not migrate and not a dominant contributor (e.g. Showers)

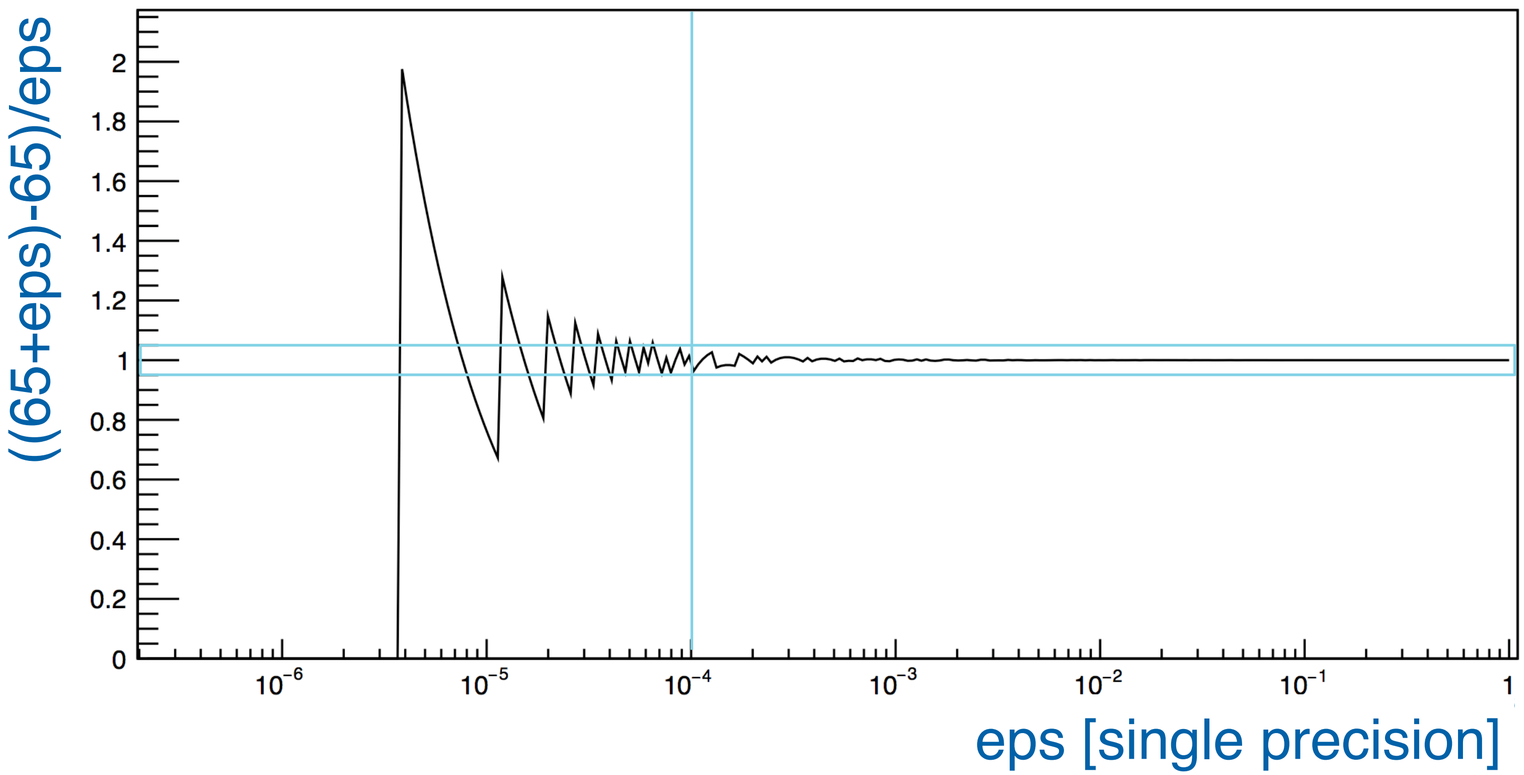
# Choices to make

- Your favorite data format stores doubles. You can…
- Do nothing
  - no reduction in precision nor in file size, no interface break
- Change to float
  - reduction in precision and in size on file (2x for affected branches), interface break
  - to be used when the double precision is never needed
- Change to Double32_t
  - same precision as double (8 bytes) when in memory, no interface break
  - same as float (4 bytes) when written to disk (reduction in precision and in size on file)
  - to be used when the calculations in double precision may be needed, but single precision is enough when storing the object

# Floating point precision at DUNE scales

- Largest coordinate value is 65m at end of a FD module
- With single precision, we get stable results down to ~100$\mu$m, where we have ~5% level fluctuations
- This is probably good enough, even if not ideal
  - a case for Double32_t
- Mitigation may be obtained in the future by moving the origin to the middle, or storing a per-track offset

Credit: Tom Junk






Fermilab

# recob::Track and recob::SpacePoint

- Replace doubles with Double32_t
- For recob::Track this is done in TrackingTypes
  - so that recob::tracking::Point_t is now different from geo::Point_t (same for Vector_t)
  - SVector and SMatrix now also use Double32_t

**lardataobj/RecoBase/TrackingTypes.h**

```
 2  2 #define TRACKINGTYPE_H
 3  3
 4  4 // LArSoft libraries
 5    #include "larcoreobj/SimpleTypesAndConstants/geo_vectors.h"
      #include "larcorealg/Geometry/geo_vectors_utils.h"
 6  6
 7  7 // ROOT libraries
 8  8 #include "Math/GenVector/Rotation3D.h"
...
14 14  namespace tracking {
15 15
16 16    /// Type used for coordinates and values in general.
17    -  using Coord_t = double;
      +  using Coord_t = Double32_t;
18 18
19 19    /// Type for representation of position in physical 3D space.
20    -  using Point_t = geo::Point_t;
      +  using Point_t = ROOT::Math::PositionVector3D<ROOT::Math::Cartesian3D<Coord_t>, ROOT::Math::GlobalCoordinateSystemTag>;
21 21
22 22    /// Type for representation of momenta in 3D space.
23    -  using Vector_t = geo::Vector_t;
      +  using Vector_t = ROOT::Math::DisplacementVector3D <ROOT::Math::Cartesian3D<Coord_t>, ROOT::Math::GlobalCoordinateSystemTag>;
```

**lardataobj/RecoBase/SpacePoint.h**

```
15 15 #include <iosfwd>
16 16
17 17 #include "larcoreobj/SimpleTypesAndConstants/PhysicalConstants.h"
   18 #include <Rtypes.h>
18 19
19 20 namespace recob {
20 21
...
31 32
32 33   private:
33 34     ID_t                        fID;        ///< SpacePoint ID
34       -   double                    fXYZ[3];    ///< position of SpacePoint in xyz
35       -   double                    fErrXYZ[6]; ///< Error matrix (triangular).
36       -   double                    fChisq;     ///< Chisquare.
   35    +   Double32_t                fXYZ[3];    ///< position of SpacePoint in xyz
   36    +   Double32_t                fErrXYZ[6]; ///< Error matrix (triangular).
   37    +   Double32_t                fChisq;     ///< Chisquare.
37 38
38 39   public:
39       -   SpacePoint(double const*xyz,
40       -              double const*err,
41       -              double  chisq,
   40    +   SpacePoint(Double32_t const*xyz,
   41    +              Double32_t const*err,
   42    +              Double32_t  chisq,
42 43                 int     id=InvalidID);
43 44
44 45     ID_t                        ID()      const;
45       -   const double*             XYZ()     const;
46       -   const double*             ErrXYZ()  const;
47       -   double                    Chisq()   const;
   46    +   const Double32_t*             XYZ()     const;
   47    +   const Double32_t*             ErrXYZ()  const;
   48    +   Double32_t                    Chisq()   const;
```



Fermilab

# recob::MCSFitResult and recob::TrackFitHitInfo

- Migrate double to floats
- SVector5 and SMatrixSym55 converted to Double32_t
- Interface broken but these are not widely used

**lardataobj/RecoBase/MCSFitResult.h**

```
67 67  private:
68     double pid_;       ///< particle id hypothesis used in the fit
69     double momFwd_;    ///< momentum value from fit assuming a forward track direction
70     double momFwdUnc_; ///< momentum uncertainty from fit assuming a forward track direction
71     double llhdFwd_;   ///< minimum negative log likelihood value from fit assuming a forward track direction
72     double momBwd_;    ///< momentum value from fit assuming a backward track direction
73     double momBwdUnc_; ///< momentum uncertainty from fit assuming a backward track direction
74     double llhdBwd_;   ///< minimum negative log likelihood value from fit assuming a backward track direction
75     std::vector<double> radlengths_; ///< vector of radiation lengths of the segments used in the fit
76     std::vector<double> angles_;     ///< vector of angles between the segments used in the fit
   68  int   pid_;       ///< particle id hypothesis used in the fit
   69  float momFwd_;    ///< momentum value from fit assuming a forward track direction
   70  float momFwdUnc_; ///< momentum uncertainty from fit assuming a forward track direction
   71  float llhdFwd_;   ///< minimum negative log likelihood value from fit assuming a forward track direction
   72  float momBwd_;    ///< momentum value from fit assuming a backward track direction
   73  float momBwdUnc_; ///< momentum uncertainty from fit assuming a backward track direction
   74  float llhdBwd_;   ///< minimum negative log likelihood value from fit assuming a backward track direction
   75  std::vector<float> radlengths_; ///< vector of radiation lengths of the segments used in the fit
   76  std::vector<float> angles_;     ///< vector of angles between the segments used in the fit
77 77  };
78 78 }
79 79
```

**lardataobj/RecoBase/TrackFitHitInfo.h**

```
49 49     geo::WireID WireId() const { return geo::WireID(fCryostatId,fTpcId,fPlaneId,fWireId); }
50 50
51 51   private:
52        double            fHitMeas;       ///< hit position measurement
53        double            fHitMeasErr2;   ///< squared uncertainty of the hit position measurement
   52     float             fHitMeas;       ///< hit position measurement
   53     float             fHitMeasErr2;   ///< squared uncertainty of the hit position measurement
54 54     SVector5          fTrackStatePar; ///< track parameters
55 55     SMatrixSym55      fTrackStateCov; ///< covariance matrix
56 56     unsigned int      fWireId;        ///< wire id where the hit is located
```



Fermilab

# anab::Calorimetry

- Migrate double to floats and TVector3 to Point_t<Double32_t>
- Interface broken requires fixing quite some code…

**lardataobj/AnalysisBase/Calorimetry.h**

```
16 16 #include <TVector3.h>
17 17
18 18 #include "larcoreobj/SimpleTypesAndConstants/geo_types.h"
   19 #include "lardataobj/RecoBase/TrackingTypes.h"
19 20
20 21 namespace anab {
21 22
   23   using Point_t = recob::tracking::Point_t;
   24
22 25   class Calorimetry{
23 26   public:
24 27
25 28     Calorimetry();
26 29
27       double              fKineticEnergy;   ///< determined kinetic energy
28       std::vector<double> fdEdx;            ///< dE/dx, should be same size as fResidualRange
29       std::vector<double> fdQdx;            ///< dQ/dx
30       std::vector<double> fResidualRange;   ///< range from end of track
31       std::vector<double> fDeadWireResR;    ///< dead wire residual range, collection plane
32       double              fRange;           ///< total range of track
33       std::vector<double> fTrkPitch;        ///< track pitch on collection plane
34       std::vector<TVector3> fXYZ;           ///< coordinates of space points
   30     float               fKineticEnergy;   ///< determined kinetic energy
   31     std::vector<float>  fdEdx;            ///< dE/dx, should be same size as fResidualRange
   32     std::vector<float>  fdQdx;            ///< dQ/dx
   33     std::vector<float>  fResidualRange;   ///< range from end of track
   34     std::vector<float>  fDeadWireResR;    ///< dead wire residual range, collection plane
   35     float               fRange;           ///< total range of track
   36     std::vector<float>  fTrkPitch;        ///< track pitch on collection plane
   37     std::vector<Point_t> fXYZ;            ///< coordinates of space points
35 38
```



Fermilab

# Status

- What is done already:
  - change to data formats, see lardataobj feature/cerati_double2float_v2

- What still needs to be done (but aiming at this week's release):
  - Write I/O rules
    - need to double check if float->Double32_t needs I/O rules or not
  - Fix all downstream code (mostly done, but needs clean up)
    - the idea is to make the code working, optimizing it will be responsibility of code owners
  - Documentation
  - Change recob::Track interface?
    - finally replace return of TVector3 with Point/Vector_t

# Other changes to anab::Calorimetry

- Updates to the calibration workflow require correcting the track positions for the space charge effect
  - also need to retrieve the original positions and hit used in the track
- Proposal:
  - use fXYZ for space-charge corrected positions
  - add vector<size_t> to store the positions of the corresponding TrajectoryPoints in the track
  - hits can be retrieved either through the TrajectoryPointFlags or assuming the hit association is in the same order as the TrajectoryPoints
    - to be double-checked… want to avoid adding another vector<size_t> and make the object bigger

# Discussion…