



ROOT Math Libraries

ROOT Workshop

30 September 2005

Lorenzo Moneta, Andràs Zsenei

CERN/PH



ROOT 2005
Users Workshop
CERN-September 28,29,30

Workshop topics

- Use of ROOT as a general framework
- Feedback from experiments
- Progress with objects persistency
- Merge with SEAL Reflex
- Progress with the new version of CINT
- Progress with the Python interface
- What is new with the Math libraries
- Distributed Data Analysis with PROOF
- Progress with GUIs and Graphics
- Progress with the new GL viewer
- Progress with the Geometry classes

<http://root.cern.ch>

Organizing Committee

- René Brun (CERN)
- Philippe Canal (FNAL)
- Fons Rademakers (CERN)
- Nathalie Knoors (CERN)

ROOT



Outline

- ★ New ROOT Math libraries
 - ★ *Mathcore* contents
 - ★ Physics and geometry vector package
 - ★ Future additions to *MathCore*
 - ★ Fitting and Minimization
 - ★ New C++ MINUIT package



Math Work Package

- ✦ Work package from ROOT-SEAL merge

- ✦ people: Andras Zsenei, Anna Kreshuk, Lorenzo Moneta, Eddy Offermann

- ✦ contribution also from many other people

- ✦ Walter Brown and Mark Fischler

- ✦ Main responsibilities for this work package:

- ✦ Basic Mathematical functions

- ✦ Fitting and Minimization

- ✦ Random Numbers

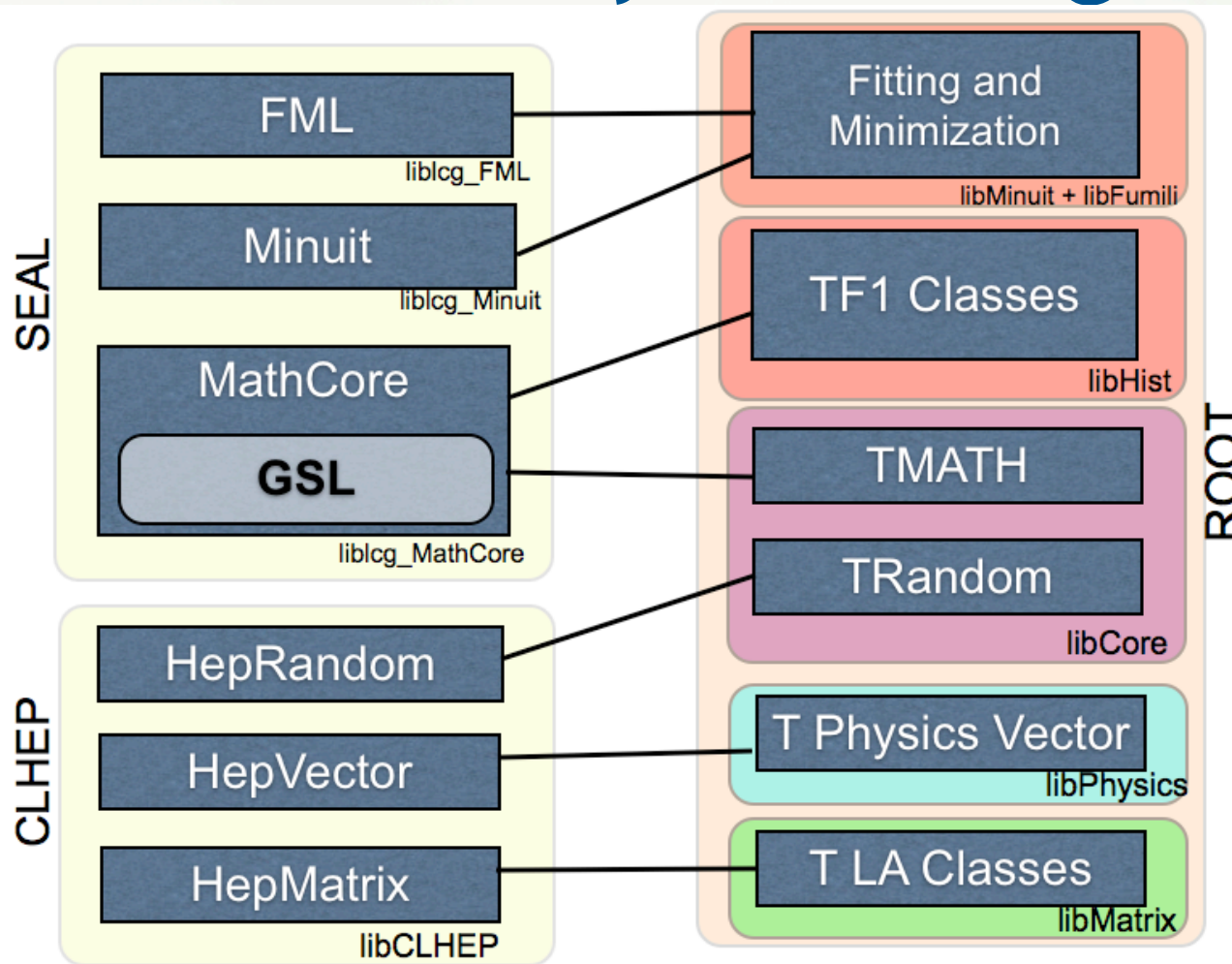
- ✦ Linear Algebra

- ✦ Physics and geometry Vectors (3D and 4D)

- ✦ Histograms

- ✦ Statistics (confidence levels, multivariate analysis, etc..)

Math Libraries before merge





New Math Libraries organization

MathMore

Sophisticated
Numerical algorithms

Extra Math functions

GSL and more

Histograms

Statistics

MathCore

Physics Vectors

Basic Math functions

Function interfaces

Basic algorithms

Random numbers

Fitting and Minimization

TMinuit

GMinuit

(new C++ Minuit)

TFumili

Linear Fitter

Robust Fitter

Linear Algebra



MathCore

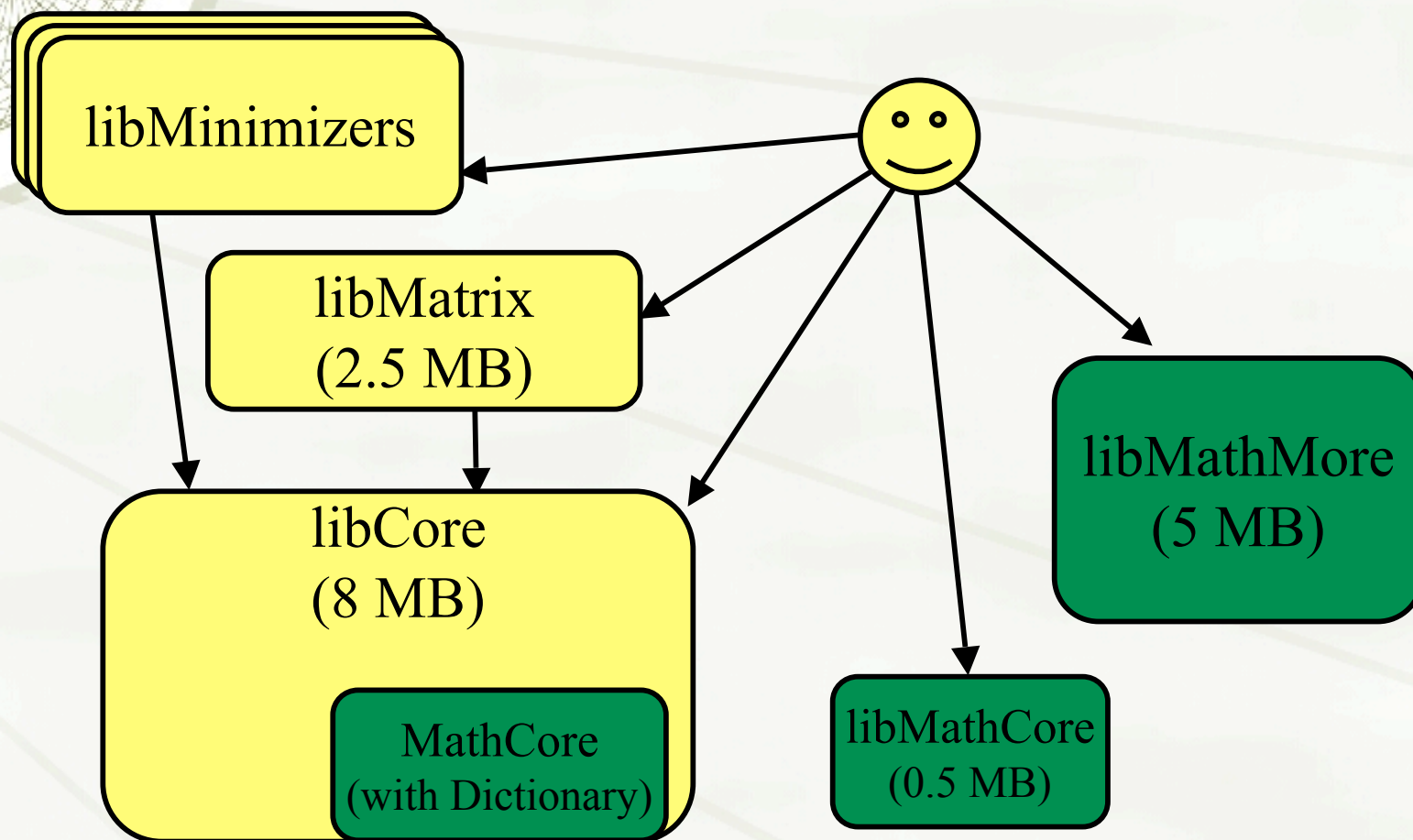
- ★ Library with the basic Math functionality
- ★ build-able as a standalone library
 - ★ no dependency on others ROOT packages
 - ★ no external dependency
- ★ Planned content of *MathCore*:
 - ★ Basic and commonly used mathematical functions
 - ★ Special and statistics (pdf, cdf) functions
 - ★ Implemented using C++ standard extension proposal
 - ★ Interfaces to function and algorithm classes
 - ★ Basic implementation of some numerical algorithms
 - ★ 3D and LorentzVectors
 - ★ Random numbers



MathMore

- ★ Library with more sophisticated mathematical functionalities
- ★ Current content:
 - ★ C++ interface to mathematical functions and numerical algorithms implemented using the Gnu Scientific Library (GSL)
- ★ See next talk (Andras Zsenei)
- ★ repository for needed and useful extra Math functionality
 - ★ could include algorithms extracted from other existing useful Math libraries

Library organization





Physics and Geometry Vectors

- ★ Classes for **3D Vectors** and **LorentzVectors** with their operations and transformations
 - ★ Merge old ROOT Physics classes and CLHEP Vector and Geometry classes
- ★ Requested by LHC experiments to be used in reconstruction, analysis and in the event data model
- ★ Work done in collaboration with C++ experts from Fermilab computing group
 - ★ M. Fischler and W. Brown
- ★ New classes designed with cleaner and minimal interfaces
 - ★ Try to avoid duplications and aim for stability



New Vector Class Properties

- ★ Vector classes templated on **scalar type**
 - ★ have vectors based on double's and float's
- ★ Generalize concept of **coordinate systems**
 - ★ Vector based on various coordinate system types
 - ★ `LorentzVector<PxPyPzE4D<double> >`
 - ★ `LorentzVector<PtEtaPhiE4D<double> >`
 - ★ Allow conversions and operations between mixed vectors
 - ★ User can choose representation optimal for his use case
 - ★ Use of typedefs to hide template complexity
 - ★ `XYZTVector` for cartesian LorentzVectors
 - ★ `PtEtaPhiEVector` for cylindrical LorentzVectors



New Vector Class Properties (cont.)

- ✦ Points and Vector distinction in 3D
 - ✦ Different transformations and operations
 - ✦ **PositionVector3D** class
 - ✦ Rotation and translation
 - ✦ Cannot be added and difference result is a DisplacementVector
 - ✦ **DisplacementVector3D** class
 - ✦ Only rotation (no translation)
 - ✦ Have addition and subtraction
 - ✦ Describe them as different types
 - ✦ not allowed transformations will not compile



Rotations

★ 3D Rotations

★ Various classes to describe rotations :

- ★ 3x3 orthogonal matrix representation
- ★ 3 Euler angles
- ★ Direction Axis (Vector) + angle
- ★ Quaternion (4 numbers)
- ★ Specialized rotations around X,Y,Z axis

★ Implemented conversions between different types

★ LorentzRotation

- ★ Generic LorentzRotation (4x4 matrix)
- ★ Boost classes



LorentzVector Example

Constructors

```
XYZTVector      v0;           // create an empty vector (x=y=z=t=0)
XYZTVector      v1(1,2,3,4);  // create a vector (x=1, y=2, z=3, t=4)
PtEtaPhiEVector v2(1,2,M_PI,5); // create a vector (pt=1, eta=2, phi=PI, E=5)
```

```
PtEtaPhiEVector v3(v1);           // create from a Cartesian4D LV
HepLorentzVector q(1,2,3,4);      // CLHEP Lorentz Vector
XYZTVector      v3(q)             // create from a CLHEP LV
```

Accessors

```
double x      = v1.X() = v1.Px(); // have both x() and px()
double t      = v1.T() = v1.E();  // have both t() and e()
double eta    = v1.Eta();
XYZVector w   = v1.Vec();         // return vector with spatial components
```

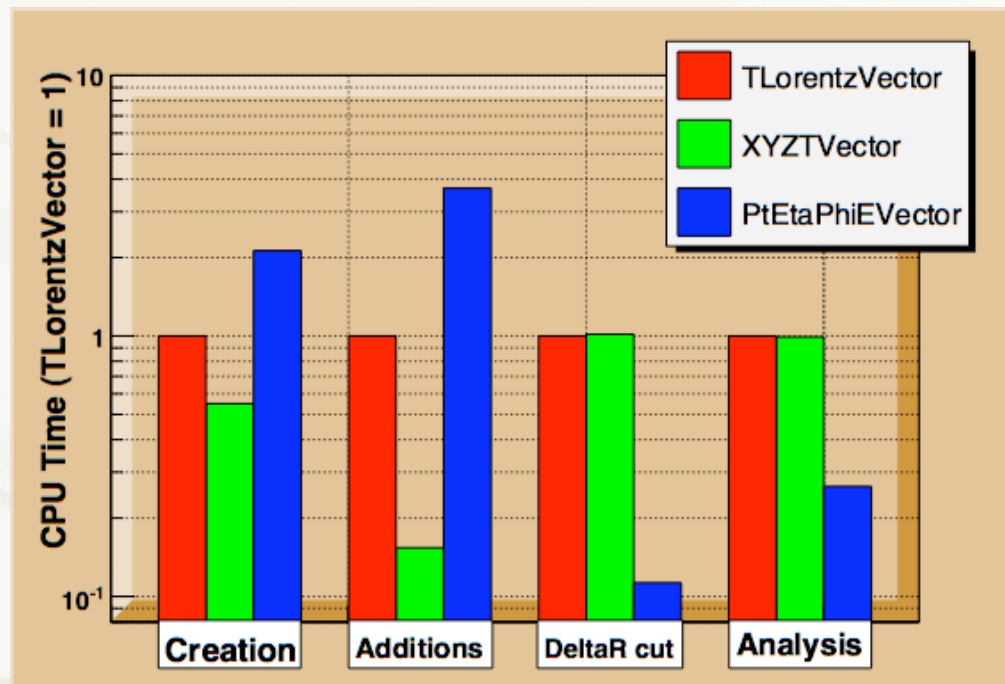
Operations

```
v1 += v2;  v1 -= v2;           // additions and subtractions
v3 = v1 + v2;
v3 = v1 - v2;
double a; v1 *= a; v1 /= a;    // multipl. and divisions with a scalar
double p = v1.Dot(v2);        // prefer dot (less ambiguous)
```

Vector Performance Tests

- ★ comparison tests with TLorentzVector (and also CLHEP)
 - ★ performance improvements if used optimal coordinate system
- ★ I/O tests
 - ★ some performance obtained with TLorentzVector if TObject stream is ignored (otherwise performance improvement ~ 20%)

from test program
stressVector in
root/test





Random Numbers

- ★ Plan to have as well a new Random number package
 - ◆ Merge CLHEP and ROOT Random number classes
 - ◆ CLHEP and ROOT have a different design
 - ★ ROOT has a common base class for all the engines and defining also the distributions
 - ◆ easier to use (no need to create separate classes)
 - ★ CLHEP separates distribution classes from engine classes
 - ◆ easier to extend if user wants to add new distributions
 - ◆ distributions classes can have a state
- ★ New design has been proposed to the C++ standard
 - ◆ A first implementation is being done by FNAL computing group.
 - ◆ We will evaluate it and try to re-implement the TRandom classes using the new library



Linear Algebra

- ◆ Base on current ROOT Linear Algebra
 - ◆ Improvements are being done (E. Offermann) to have template matrices
- ◆ Continue evaluation tests with other existing packages
 - ◆ tvmet, MET, C++ wrappers to BLAS and LAPACK, GLAS (Boost)
- ◆ Requests from the experiments for an independent package to replace CLHEP matrices
- ◆ Evaluate work needed for producing optimized package for small matrices
 - ◆ Typically matrix used in tracking have sizes up to 5
 - ◆ Needed only inversions + multiplication
 - ◆ No need for full linear algebra functionality

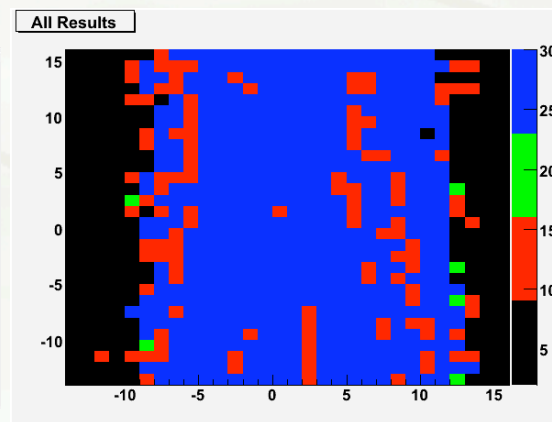
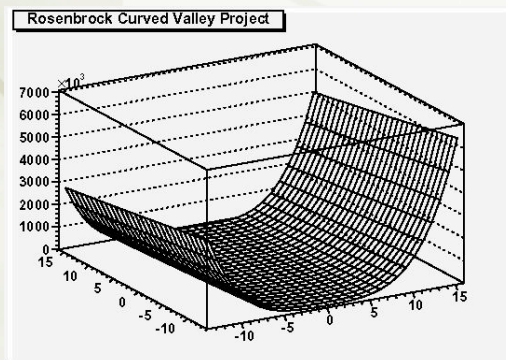


Fitting and Minimization

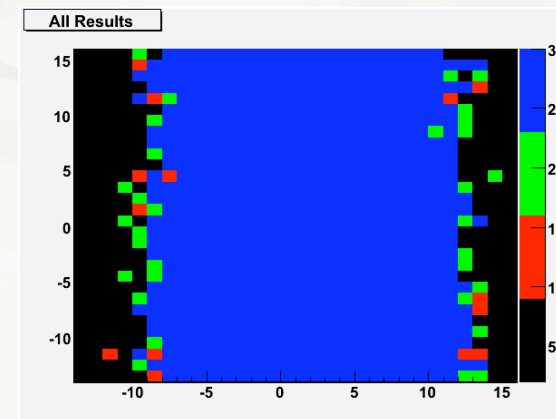
- ★ A new C++ version of MINUIT has been developed within the SEAL project
 - ★ Fortran algorithms have been carefully translated in C++
 - ★ supervised from original author (F. James)
- ★ Same basic functionality as in old version
 - ★ Migrad, Simplex, Minos algorithms
- ★ Extended functionality:
 - ★ Single side parameter limits
 - ★ Added Fumili minimization method for Chi2 and likelihood minimization
- ★ OO package for generic function minimization
 - ★ Easy to extend by inserting new minimization methods

C++ *Minuit* tests

- ★ Going under extensive performance and convergence tests
 - ◆ Same function calls needed to find minimum as in old version
 - ◆ Basically the same performance
- ★ Example: test convergence with RosenBrock function
 - ◆ (see more in A. McLennan report)



before



After bug fixes in C++ Minuit



GMinuit

- ◆ New *TVirtualFitter* implementation based on new C++ MINUIT
 - ◆ *GMinuit* class
- ◆ Performance improvements obtained using an optimized C++ interface
 - ◆ Minimization function (FCN) is passed to MINUIT as an object
 - ◆ cache internally its data points used in fitting
 - ◆ suppress un-used histogram bins (zero entries)
- ◆ Plan to add *GMinuit* in the next ROOT release
- ◆ Implement a backward-compatible interface for new C++ MINUIT



Plans for New Fitter interfaces

- ★ Improve fitting and minimization interface in ROOT
- ★ Redesign taking into account all fitter and minimizer methods present in ROOT
 - ★ Linear and robust fitter, fraction fitter
 - ★ Minuit (Migrad, Simplex), quadratic optimization
 - ★ RooFit
- ★ Idea is to have separate interfaces:
 - ★ Fitting interfaces used to construct an objective function (chi2, likelihood) from data and model function (pdf)
 - ★ Minimization interfaces to perform the task of finding the function minimum
 - ★ Various minimizers implementations as separate plug-ins
- ★ Use common function interfaces defined in *MathCore* for numerical algorithms



Integration in ROOT

- ★ New CVS directory existing in ROOT 5
 - ★ *mathcore* and *mathmore*
 - ★ Build now as separate library *libMathCore* and *libMathMore*
 - ★ plan is to have *MathCore* in *libCore*
- ★ Classes are in namespace *ROOT::Math*
- ★ Public header files are in *include/Math*
 - ★ Possible transparent transition of classes *MathCore* ↔ *MathMore*



MathCore Current Status

- ★ Available now in *MathCore*:

- ★ 3D and 4D Vector package

- ★ basic Math functions

- ★ Can be downloaded as a tar file and built independently from other ROOT libraries

- ★ Documentation available at

- ★ http://seal.web.cern.ch/seal/MathLibs/MathCore-5_0_4/html/index.html

[Main Page](#) | [Modules](#) | [Namespace List](#) | [Class List](#) | [Directories](#) | [File List](#) | [Namespace Members](#) | [Class Members](#) | [File Members](#) | [Related Pages](#)

MathCore library

5_0_4

MathCore provides a collection of functions and C++ classes for HEP numerical computing. This library provides only the basic mathematical functions and algorithms and not all the functionality required by the HEP community. The current set includes classes and functions for:

- Basic special functions used in HEP like Gamma, Error functions,
- Mathematical functions used in statistics such as probability density functions, cumulative distributions functions and their inverse.
- **Generic Vector for 3 and 4 Dimensions**



Conclusions

- ★ Released since June (ROOT) first version of new Math library (*MathCore*)
- ★ *MathMore* is available in 5.04 ([see next talk](#))
- ★ Future work:
 - ★ Add Random numbers
 - ★ Define interfaces for functions and numerical algorithms to be used by client libraries (TF1)
 - ★ Redesign fitting interfaces
 - ★ Import new C++ MINUIT