ROOT

Data Analysis Framework

presented by Marius Dörrie and Amelie Coumans

Contents

- General information about ROOT
  - What is Root?
  - The development of Root
  - What is ROOT capable of?
- Examples of applications in ROOT
- Our project
- References

What is ROOT?

- Open source software for processing of data
- Developed at CERN and based on C++
- Use:
  - By nearly all Nuclear and High Energy Physics experiments
  - Medical and financial industries, other applications
- Data Analysis *Framework*
  - Many different functions in one program
  - Simpler for the user

The development of ROOT

- Former data analysis software: PAW based on FORTRAN
- In the ’90, explosion of experiments’ data rate

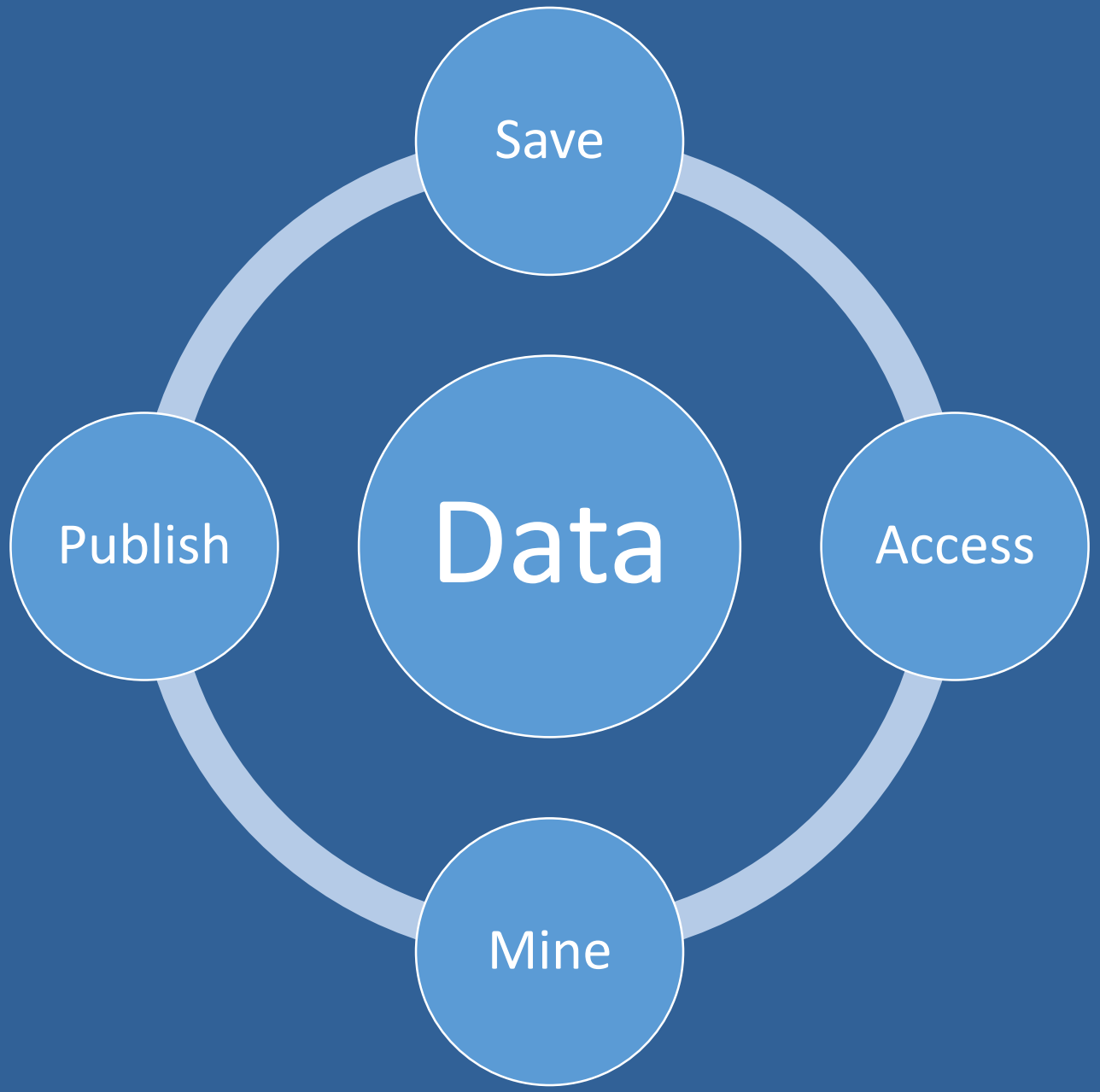
  → more efficient data storage and analysis software was needed
- software for the experiments now based on C++
- First public release at the end of 1995

- Since 25 years a team improves ROOT constantly

  → new versions are released frequently

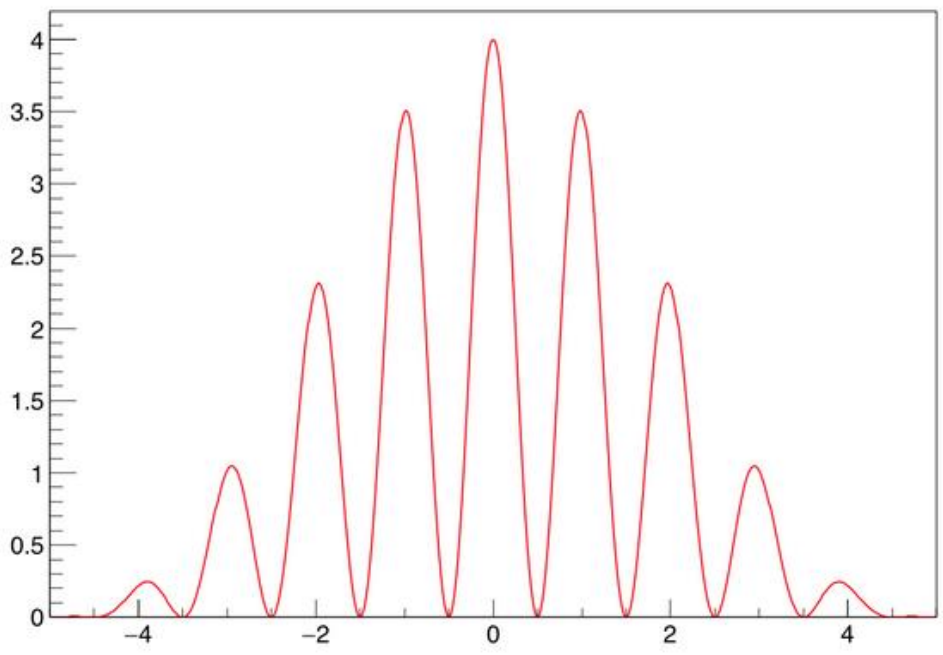
What is ROOT capable of?

- Data analysis
  - Visualise data
  - Compare measurements to theoretical models
  - Interactive analysis


Save
Publish
Data
Access
Mine

Examples of applications in ROOT

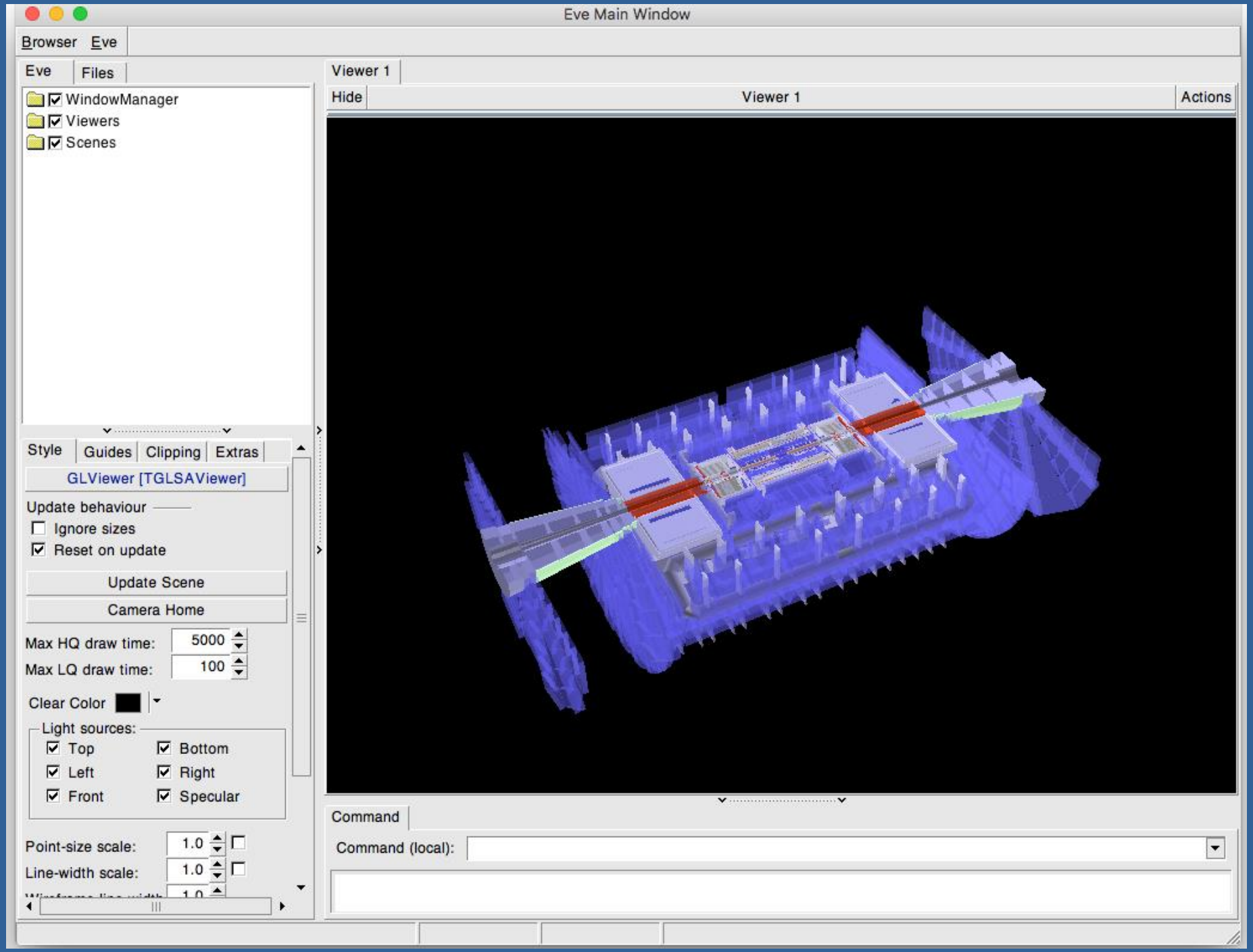
```
template <typename T, unsigned int NDIM>
class THnHelper : public ROOT::Detail::RDF::RActionImpl<THnHelper<T, NDIM>> {
public:
   /// This is a handy, expressive shortcut.
   using THn_t = THnT<T>;
   /// This type is a requirement for every helper.
   using Result_t = THn_t;

private:
   std::vector<std::shared_ptr<THn_t>> fHistos; // one per data processing slot

public:
   /// This constructor takes all the parameters necessary to build the THnTs. In addition,
   /// the columns which will be used.
   THnHelper(std::string_view name, std::string_view title, std::array<int, NDIM> nbins, std
             std::array<double, NDIM> xmax)
   {
      const auto nSlots = ROOT::IsImplicitMTEnabled() ? ROOT::GetImplicitMTPoolSize() : 1;
      for (auto i : ROOT::TSeqU(nSlots)) {
         fHistos.emplace_back(std::make_shared<THn_t>(std::string(name).c_str(), std::string
                                                      NDIM, nbins.data(), xmins.data(), xmax
         (void)i;
      }
   }
   THnHelper(THnHelper &&) = default;
   THnHelper(const THnHelper &) = delete;
   std::shared_ptr<THn_t> GetResultPtr() const { return fHistos[0]; }
   void Initialize() {}
   void InitTask(TTreeReader *, unsigned int) {}
   /// This is a method executed at every entry
   template <typename... ColumnTypes>
   void Exec(unsigned int slot, ColumnTypes... values)
   {
      // Since THnT<T>::Fill expects a double*, we build it passing through a std::array.
      std::array<double, sizeof...(ColumnTypes)> valuesArr{static_cast<double>(values)...};
```



Output of slits.C with parameters 0.2 and 2.

The example first asks for user input, namely the ratio of slit width over slit distance, and the number of slits. After entering this information, you should see the graphical output as is shown in Figure 2.1.

```
4
5  auto pi = TMath::Pi();
6
7  // function code in C
8  double single(double *x, double *par) {
9    return pow(sin(pi*par[0]*x[0])/(pi*par[0]*x[0]),2);
10 }
11
12 double nslit0(double *x,double *par){
13   return pow(sin(pi*par[1]*x[0])/sin(pi*x[0]),2);
14 }
15
16 double nslit(double *x, double *par){
17   return single(x,par) * nslit0(x,par);
18 }
19
20 // This is the main program
21 void slits() {
22   float r,ns;
23
24   // request user input
25   cout << "slit width / g ? ";
26   scanf("%f",&r);
27   cout << "# of slits? ";
28   scanf("%f",&ns);
29   cout <<"interference pattern for "<< ns
30        <<" slits, width/distance: "<<r<<endl;
31
32   // define function and set options
33   TF1 *Fnslit  = new TF1("Fnslit",nslit,-5.001,5.,2);
34   Fnslit->SetNpx(500);
35
36   // set parameters, as read in above
37   Fnslit->SetParameter(0,r);
38   Fnslit->SetParameter(1,ns);
39
40   // draw the interference pattern for a grid with n slits
41   Fnslit->Draw();
42 }
```


Eve Main Window
Browser Eve
Eve Files
WindowManager
Viewers
Scenes
Style Guides Clipping Extras
GLViewer [TGLSAViewer]
Update behaviour
Ignore sizes
Reset on update
Update Scene
Camera Home
Max HQ draw time: 5000
Max LQ draw time: 100
Clear Color
Light sources:
Top Bottom
Left Right
Front Specular
Point-size scale: 1.0
Line-width scale: 1.0
Viewer 1
Hide Viewer 1 Actions
Command
Command (local):

```
void geom_atlas()
{
   TEveManager::Create();

   TFile::SetCacheFileDir(".");
   gGeoManager = gEve->GetGeometry("http://root.cern.ch/files/atlas.root");
   gGeoManager->DefaultColors();

   auto node1 = gGeoManager->GetTopVolume()->FindNode("INNE_1");
   TEveGeoTopNode* inn = new TEveGeoTopNode(gGeoManager, node1);
   gEve->AddGlobalElement(inn);

   auto node2 = gGeoManager->GetTopVolume()->FindNode("CENT_1");
   TEveGeoTopNode* cnt = new TEveGeoTopNode(gGeoManager, node2);
   gEve->AddGlobalElement(cnt);

   auto node3 = gGeoManager->GetTopVolume()->FindNode("OUTE_1");
   TEveGeoTopNode* out = new TEveGeoTopNode(gGeoManager, node3);
   gEve->AddGlobalElement(out);

   gEve->FullRedraw3D(kTRUE);

   // EClipType not exported to CINT (see TGLUtil.h):
   // 0 - no clip, 1 - clip plane, 2 - clip box
   auto v = gEve->GetDefaultGLViewer();
   v->GetClipSet()->SetClipType(TGLClip::EType(1));
   v->RefreshPadEditor(v);

   v->CurrentCamera().RotateRad(-.7, 0.5);
   v->DoDraw();
}
```

Author

Matevz Tadel

Our Project

- Create a video about ROOT to give a short overview of its functions
- On the website the video can quickly introduce people to ROOT

A short intro video for ROOT’s Website

ROOT

Data Analysis Framework

References

- Textual sources:
  - http://www.physik.uni-regensburg.de/studium/edverg/ckurs/Aufgabensammlung_loesungen.pdf
- Other sources:
  - https://root.cern/doc/master/geom__atlas_8C.html
  - https://root.cern/doc/master/geom__alias_8C.html
  - https://root.cern/doc/master/geom__cms_8C.html
  - https://root.cern/doc/master/geom__lhcb_8C.html
  - https://root.cern/doc/master/candleplotstack_8C.html
  - https://root.cern/doc/master/ConfidenceIntervals_8C.html

ROOT

Data Analysis Framework

Thank you for attention and your concern!

For more detailed information visit *root.cern.ch*