https://root.cern/js/

JavaScript ROOT

Bertrand Bellenot (CERN), Sergey Linev (GSI Darmstadt)

Introduction

JavaScript ROOT provides interactive graphics in the web browsers for the major classes like histograms (TH1/TH2/TH3), graphs (TGraph), functions (TF1) and many others. Reading of binary and JSON ROOT files is supported.
The flexible and powerful JSROOT API used in different web-based applications to implement interactive display of user data - CERNBox, iPython notebooks, THttpServer UI and many others.
The JSROOT code, documentation, and many examples can be found on https://root.cern/js/ website, the source code repository is https://github.com/root-project/jsroot/.

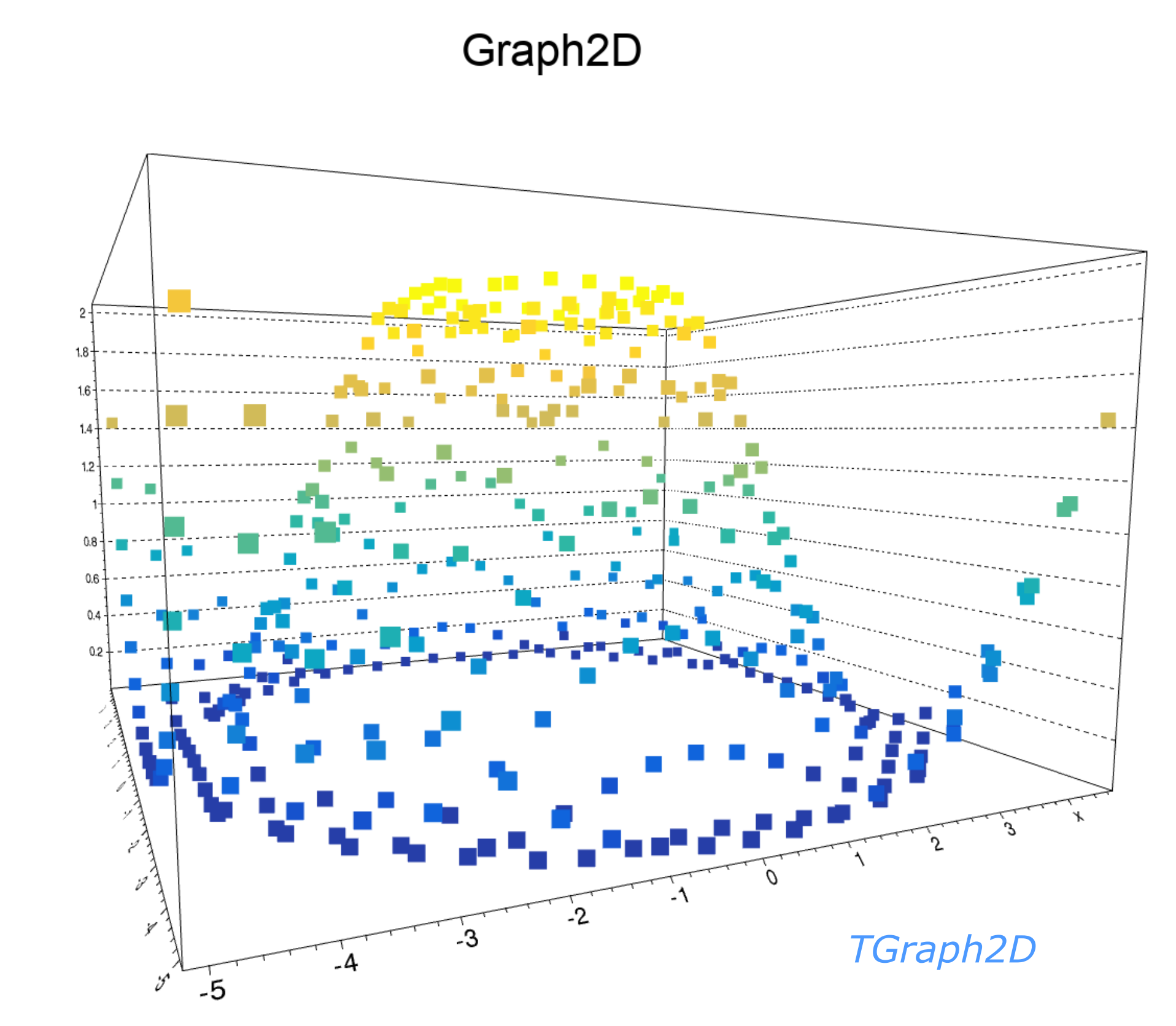
New in JSROOT v5

- Full support of ROOT binary files:
  - all kind of STL containers
  - LZ4 compression
  - custom streamers
  - reading files from local disk
  - very old (before 2009) ROOT files
  - TTree reading
- Many new classes and draw options
- All kinds of fill styles
- Full integration with Node.js
- Built-in TLatex parser (alternative to MathJax.js)
- Codebase for ROOT7 webgui clients

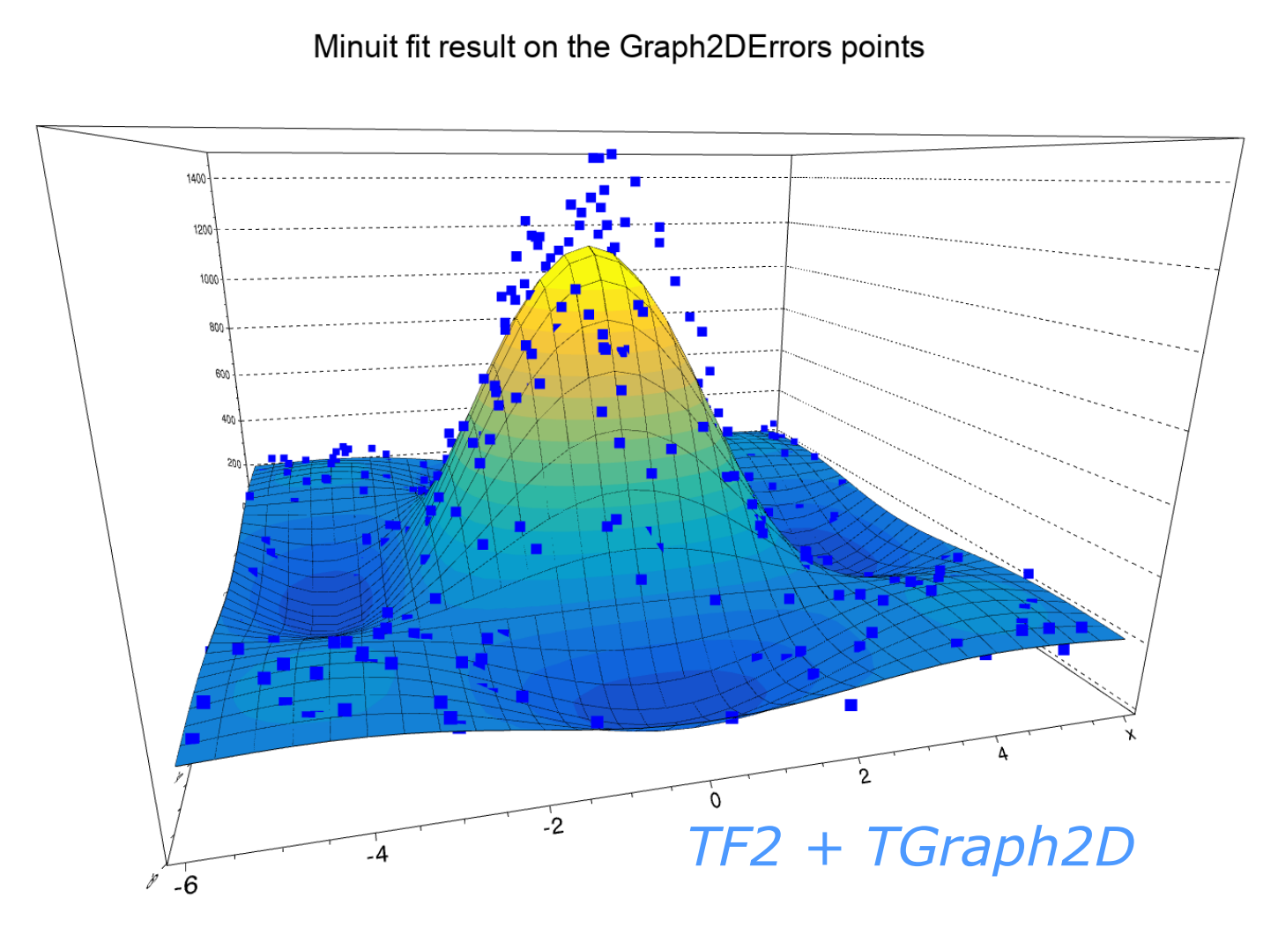
New classes and options

- TGraph - 'z', 'x', '||', '[]', '>', '|>', '5', 'X+', 'Y+‘, ‘RX’, ‘RY’
- TH1 - '*', 'L', 'LF2', 'B', 'B1', 'TEXT', 'E0', 'E3', 'E4', 'EX0', 'X+', 'Y+'
- TH2 - 'E', 'col1', 'box', 'box1', 'surf3', 'surf7', 'base0'
- TH2 - 'same' with 'box', 'col', 'cont', 'lego', 'surf‘
- TH3 - 'scat‘, ‘box2’, ‘box3’, ‘glbox2’, ‘glcol’
- TF1/TF2 - 'nosave' to ignore saved buffer
- TCanvas - logx/y/z, gridx/y, tickx/y
- THStack - 'lego' and other 3D draw options
- TProfile2D, TGraph2D, TGraph2DErrors
- TGraphPolar, TGraphTime, TSpline3, TSpline5
- TPavesText, TArc, TDiamond, TCrown, TMarker
- TPolyLine3D, TPolyMarker, TCurlyLine, TCurlyArc

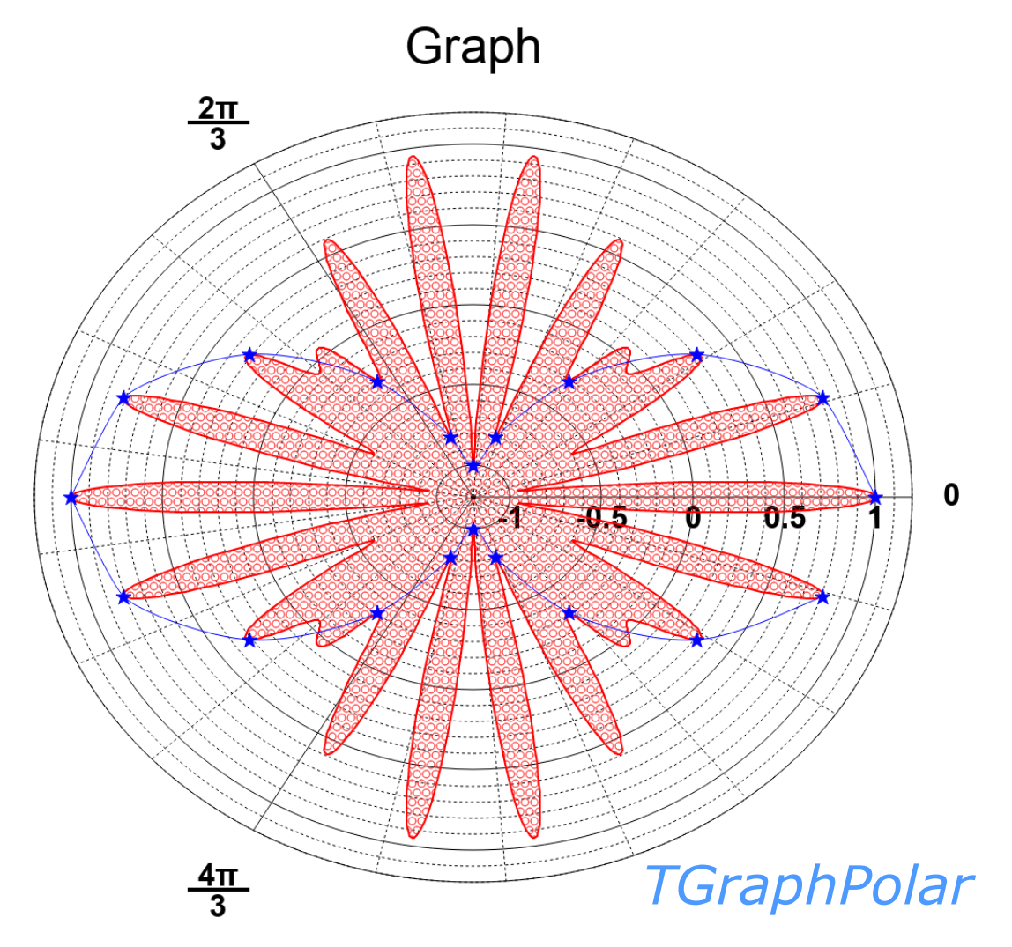
New supported classes



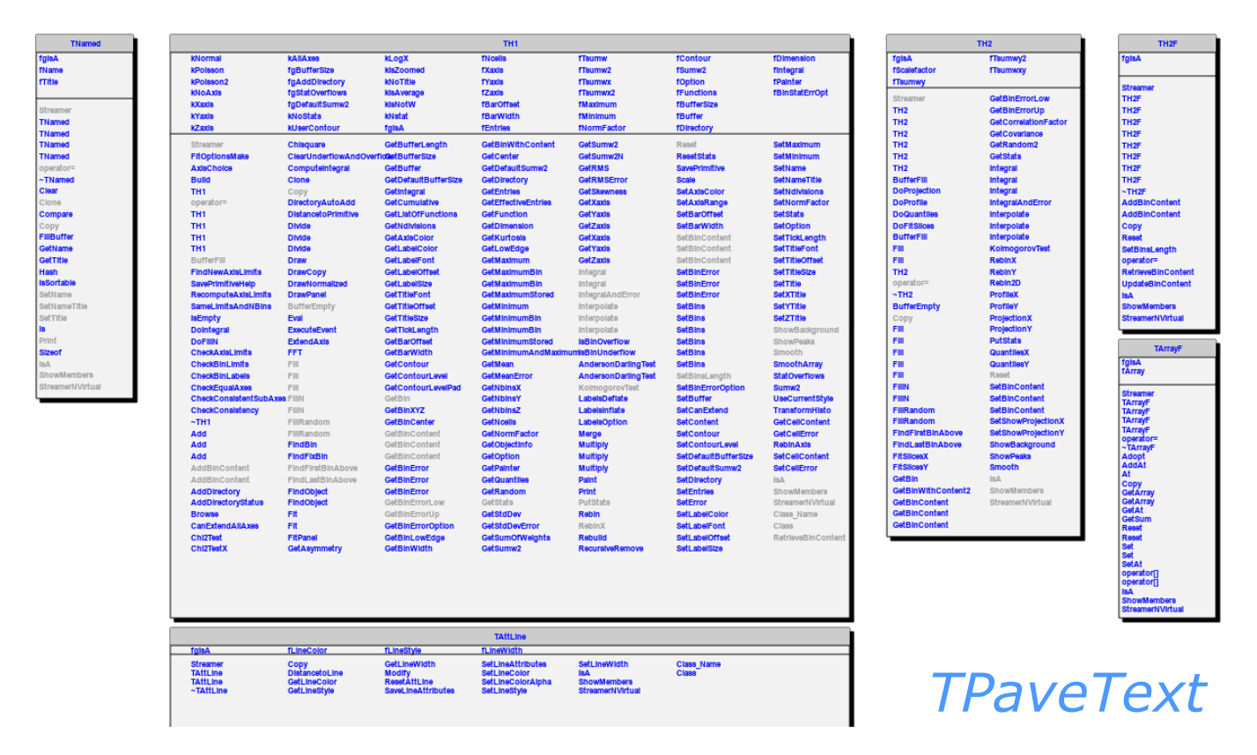
TGraph2D

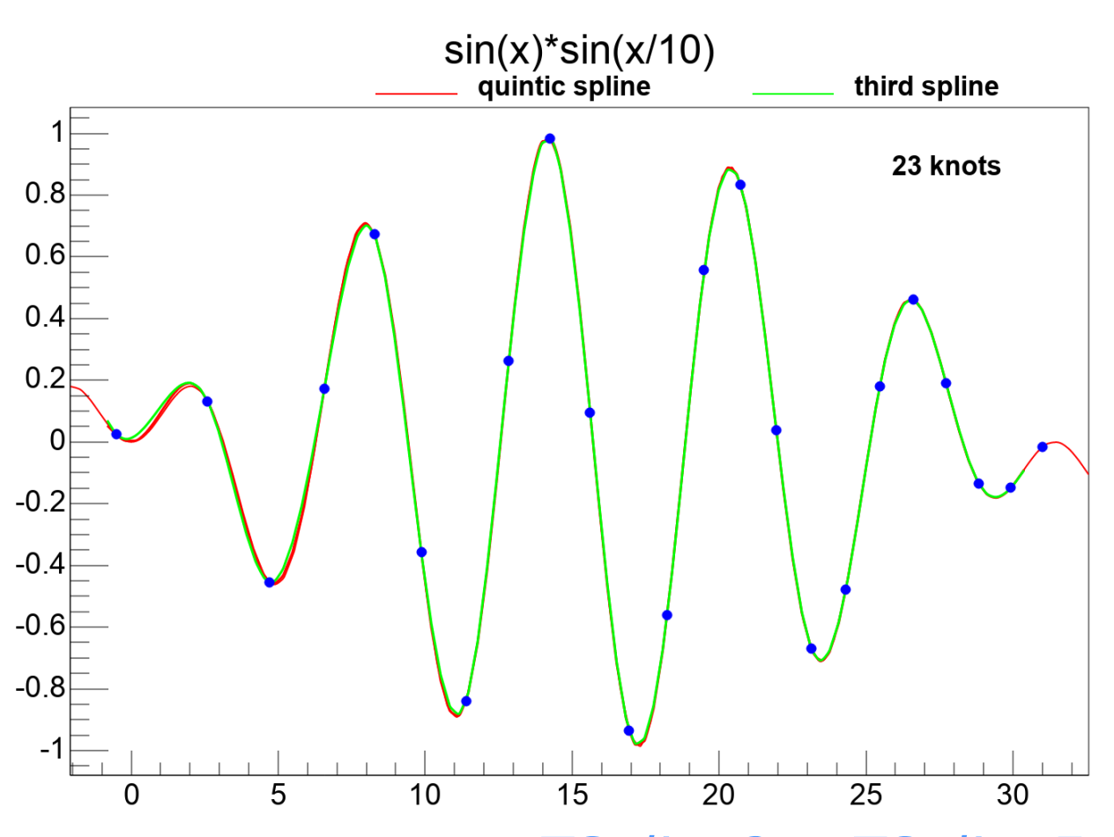


TF2 + TGraph2D

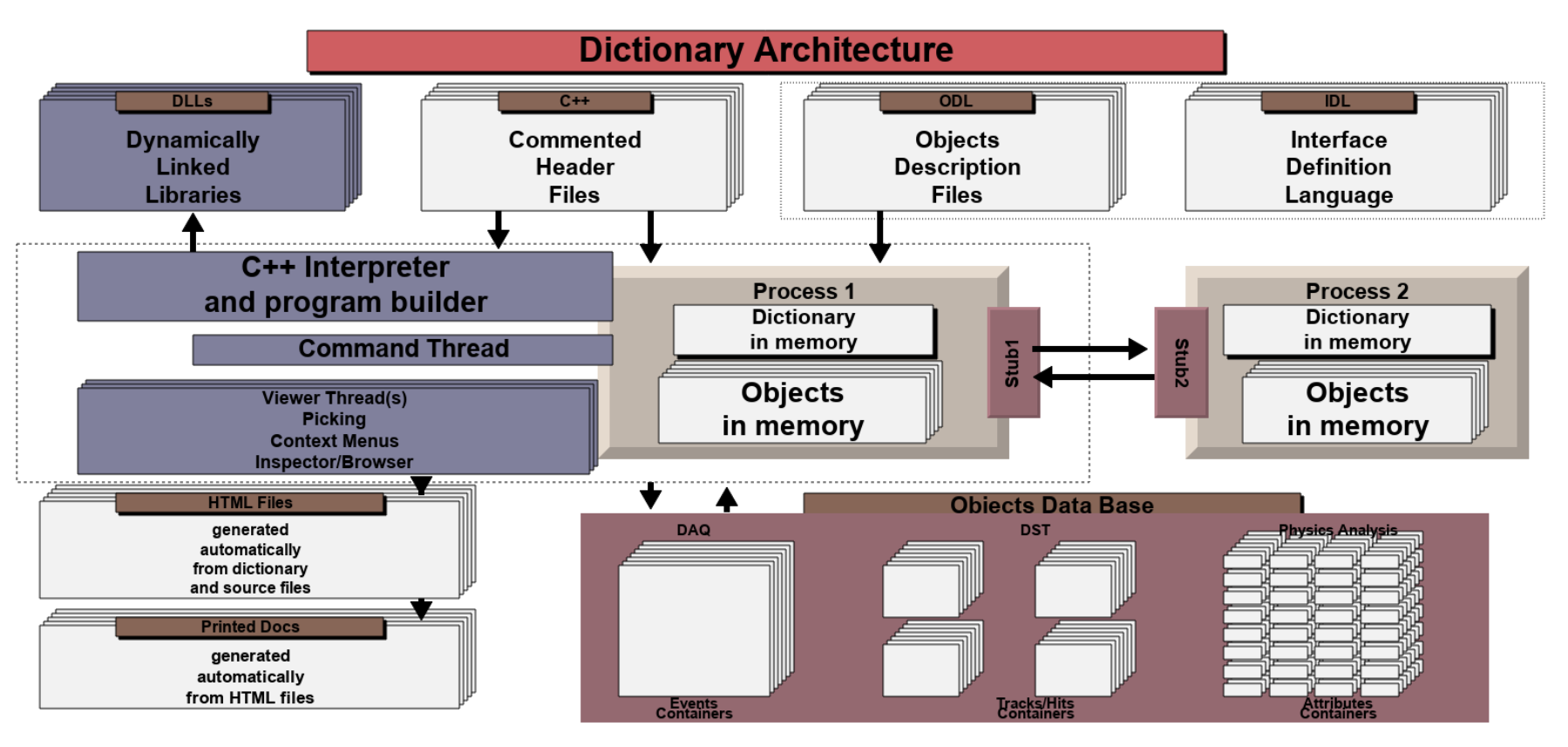


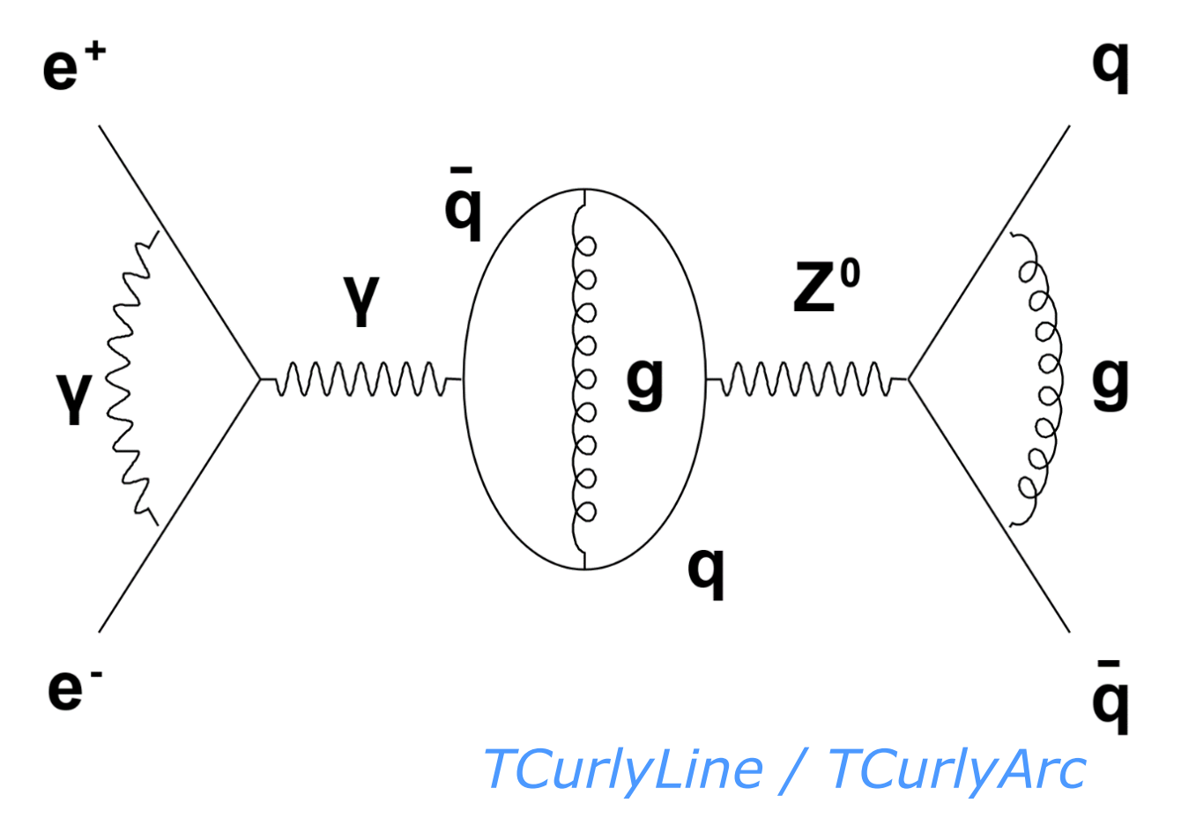
TGraphPolar


TPaveText



TSpline3 + TSpline5


Dictionary Architecture
Dynamically Linked Libraries
Commented Header Files
Objects Description Files
Interface Definition Language
C++ Interpreter and program builder
Command Thread
Process 1 Dictionary in memory
Objects in memory
Process 2 Dictionary in memory
Objects in memory
Viewer Thread(s) Picking Context Menus Inspector/Browser
Objects Data Base
TPavesText



TCurlyLine / TCurlyArc

Data exchange via JSON

- New JSON format
  - better objects referencing
  - array compression (optional)
  - JS native array (optional)
  - full support of previous format
- Server -> Client:

```
auto ptr = new UserClass;
TString json = TBufferJSON::ToJSON(ptr);

var obj = JSROOT.parse(json);
```

- Client -> Server

```
var json = JSROOT.toJSON(obj);

UserClass *ptr = nullptr;
TBufferJSON::FromJSON(ptr, json);
```

Node.js

- Reading binary and JSON files
- Access data from TTree, perform TTree::Draw
- Produce SVG images
- Available as npm package
  - no any dependency from ROOT C++ code
  - just do **npm install jsroot**
- Used for automated JSROOT tests
  - https://github.com/linev/jsroot-test

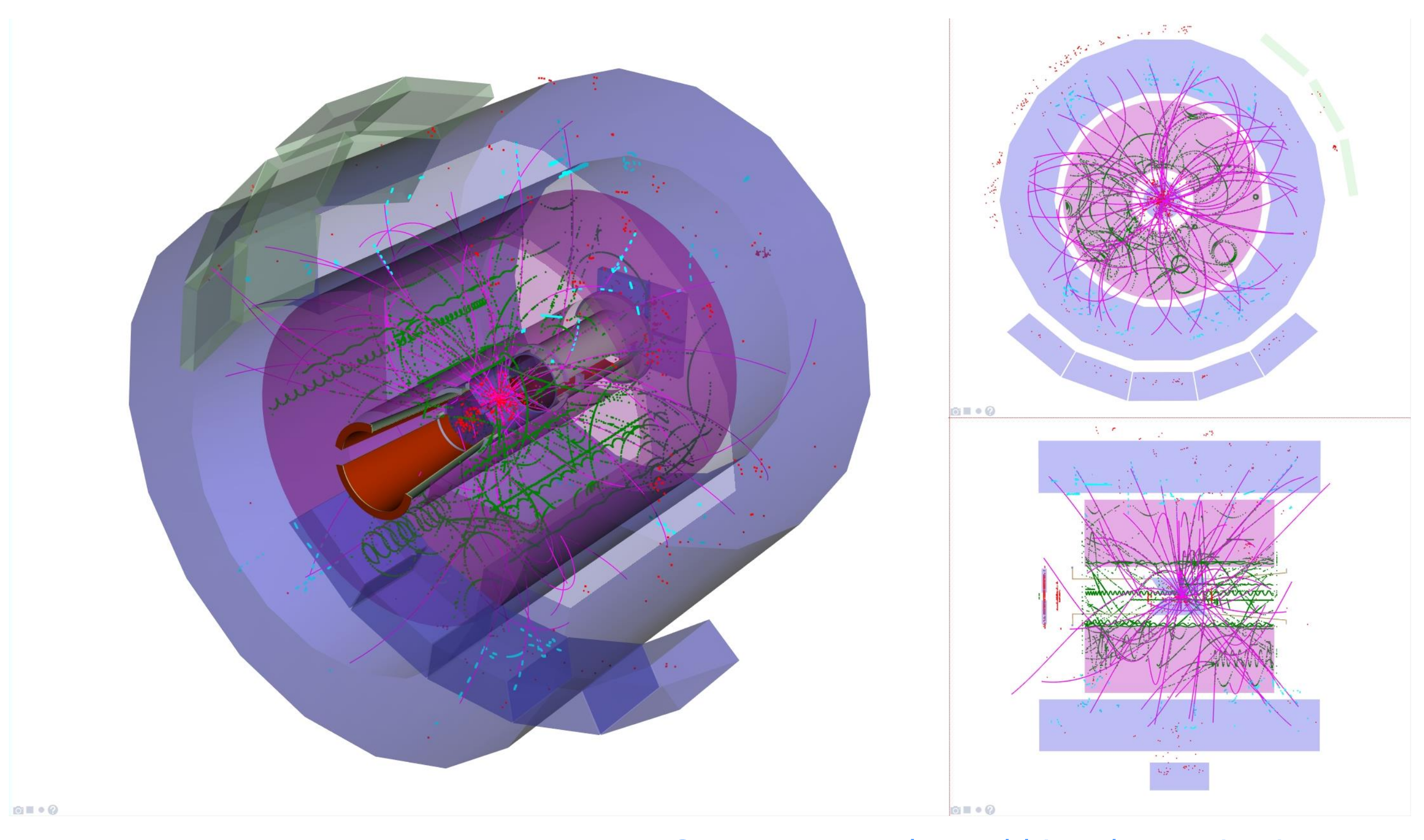
```
// Example of SVG file generation

var jsroot = require("jsroot"), fs = require("fs"),
     fname = "https://root.cern/js/files/hsimple.root";

jsroot.OpenFile(fname, function(file) {
  file.ReadObject("hpx;1", function(obj) {
    jsroot.MakeSVG({ object: obj, option: "lego2" },
      function(svg) {
        fs.writeFileSync("lego2.svg", svg);
      }
    );
  });
});
```

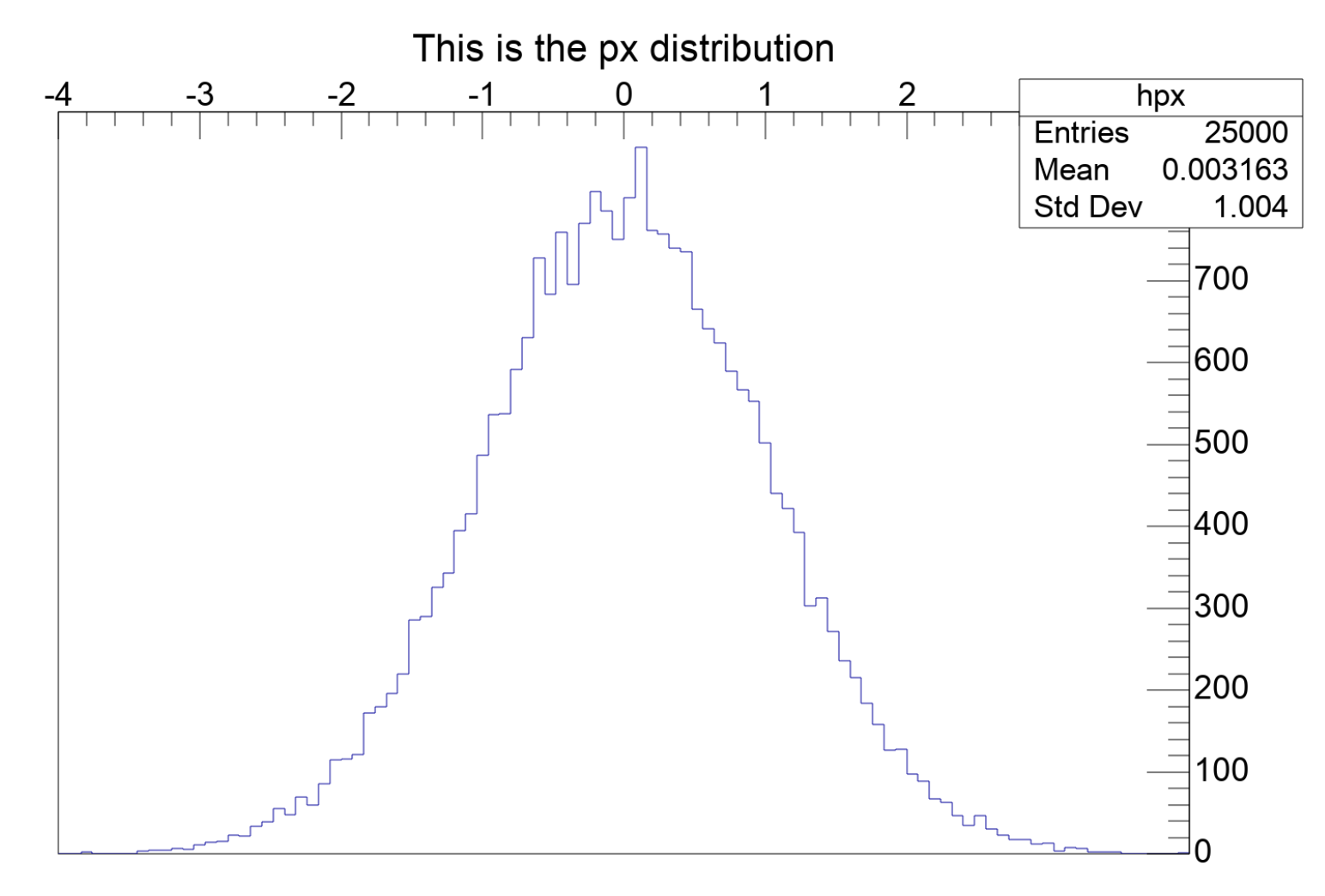
TTree

- Fast reading of any data
  - use multirange HTTP requests
- TSelector-like API
  - method called for every next read event
  - bulk array operations when applicable
- Advanced TTree::Draw
  - can be invoked from default JSROOT browser

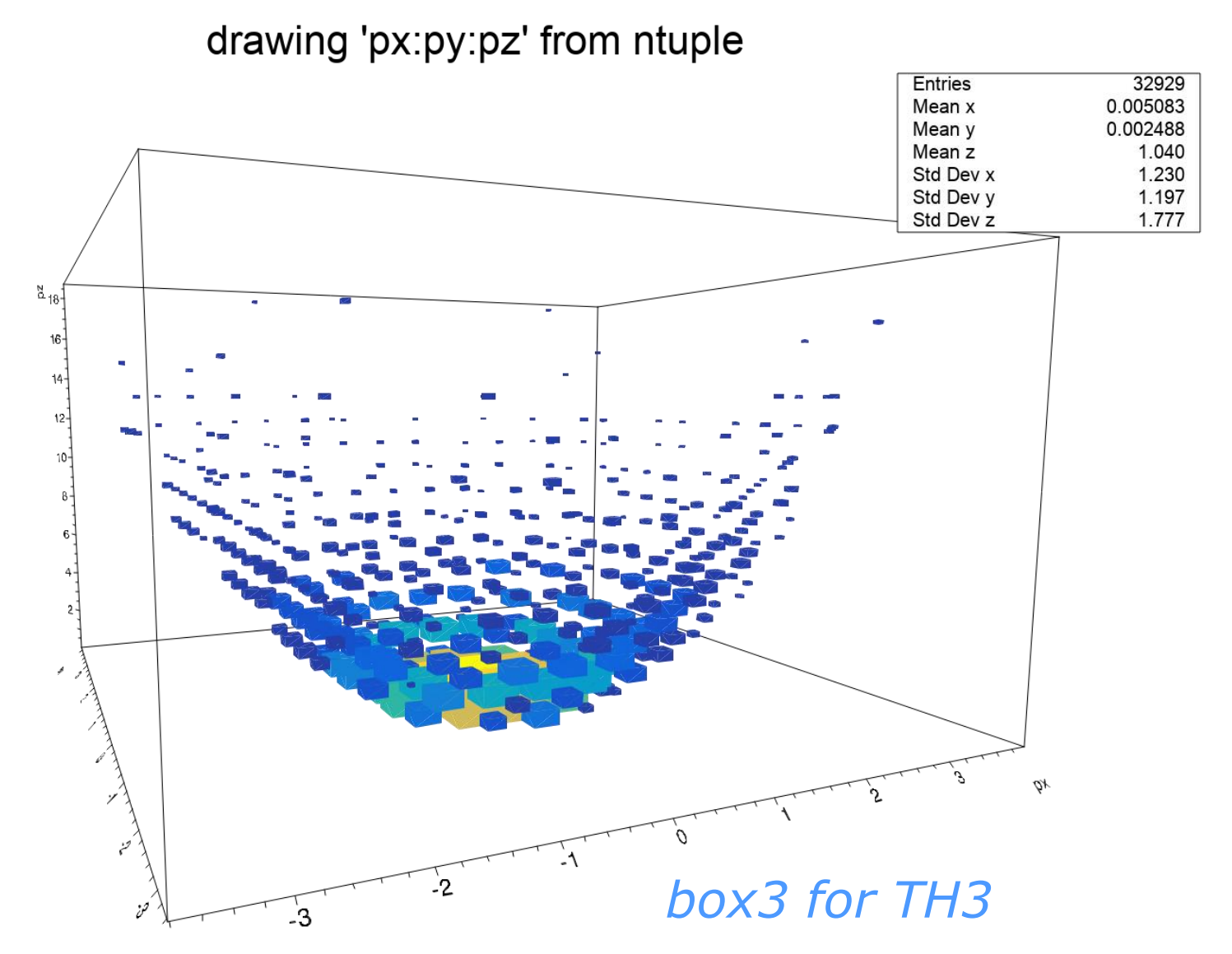
```
var jsroot = require("jsroot"),
     fname = "https://root.cern/js/files/hsimple.root";

jsroot.OpenFile(fname, function(file) {
  file.ReadObject("ntuple;1", function(tree) {
    tree.Draw({ expr:"px:py:pz", dump:true, numentries:100 },
      function(res){
        var sumx = 0, sumy = 0, sumz = 0, k = 1/res.length;
        res.forEach(function(item) {
          sumx += item.x; sumy += item.y; sumz += item.z;
        });
        console.log("Mean x/y/z", k*sumx, k*sumy, k*sumz);
    });
  });
});
```

New supported draw options

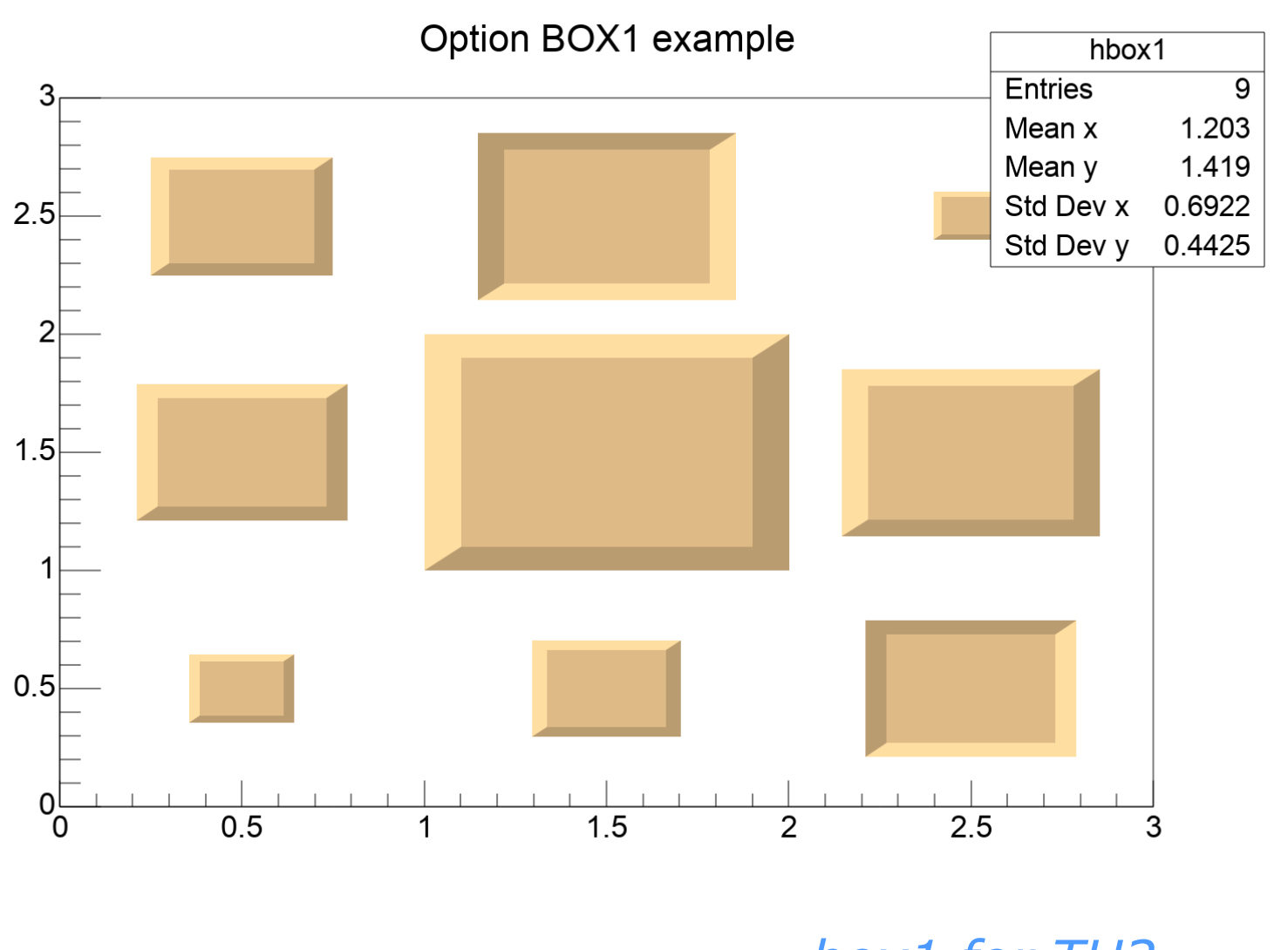

Geometry, tracks and hits plus projections



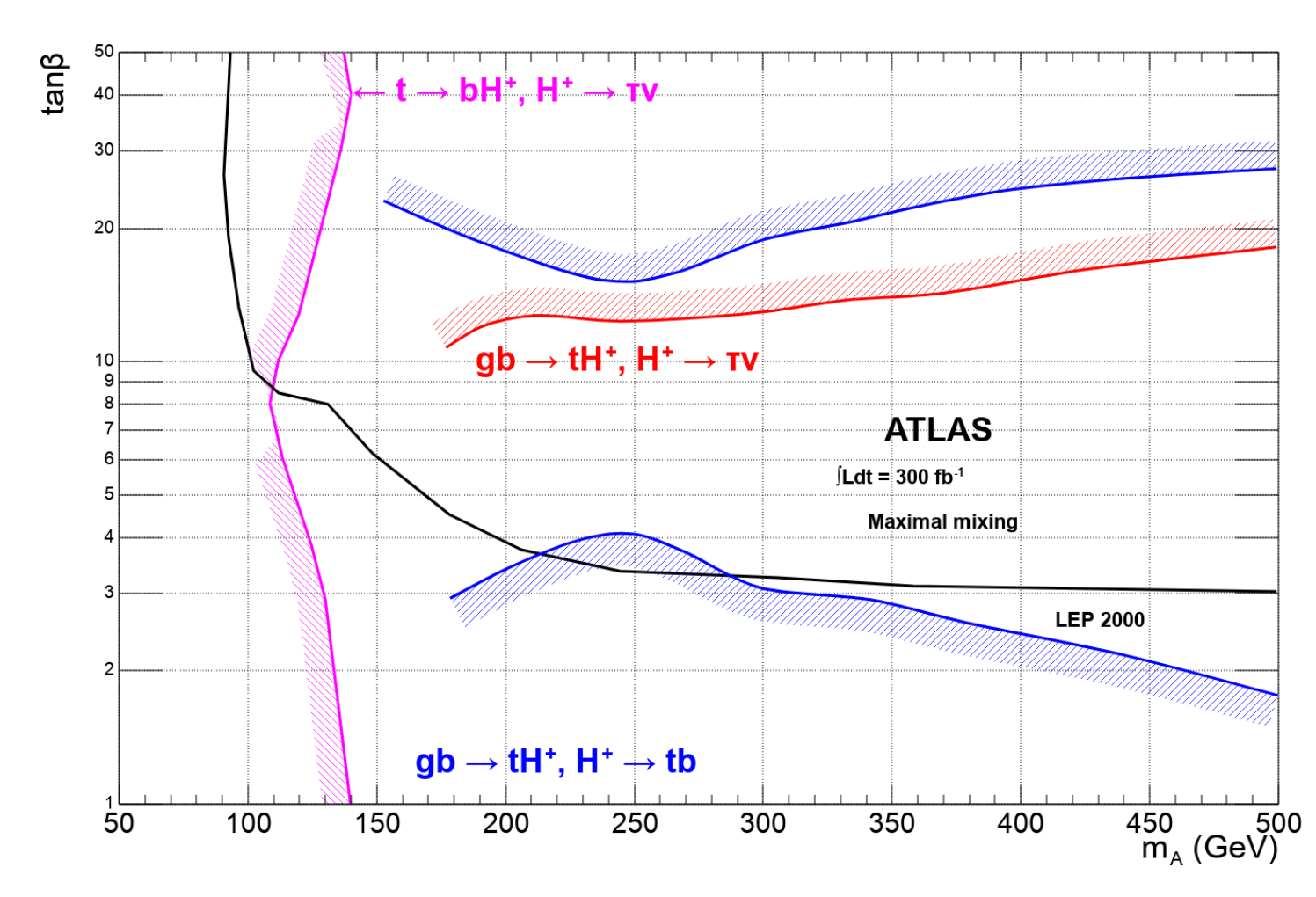
X+Y+ options for axes drawing



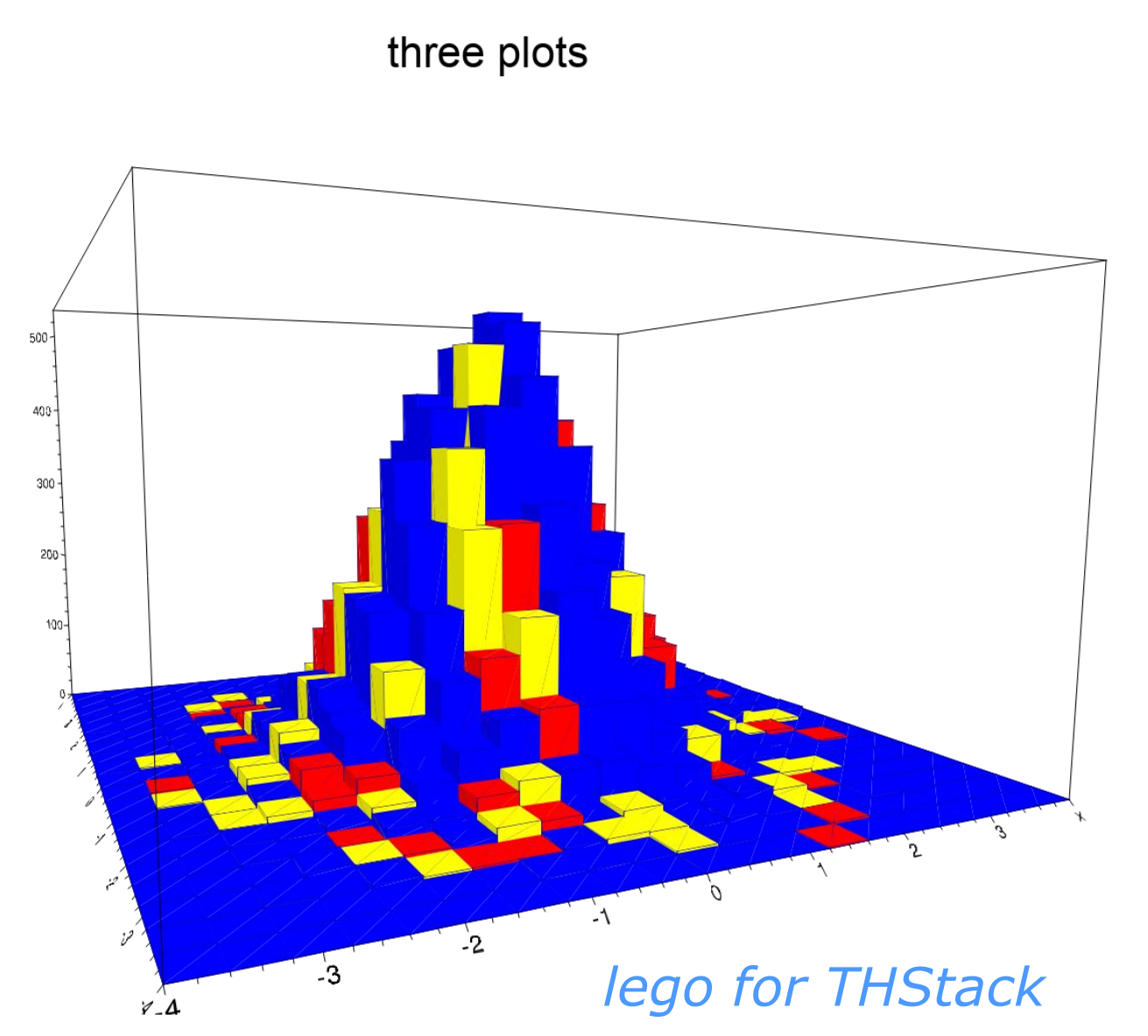
box3 for TH3



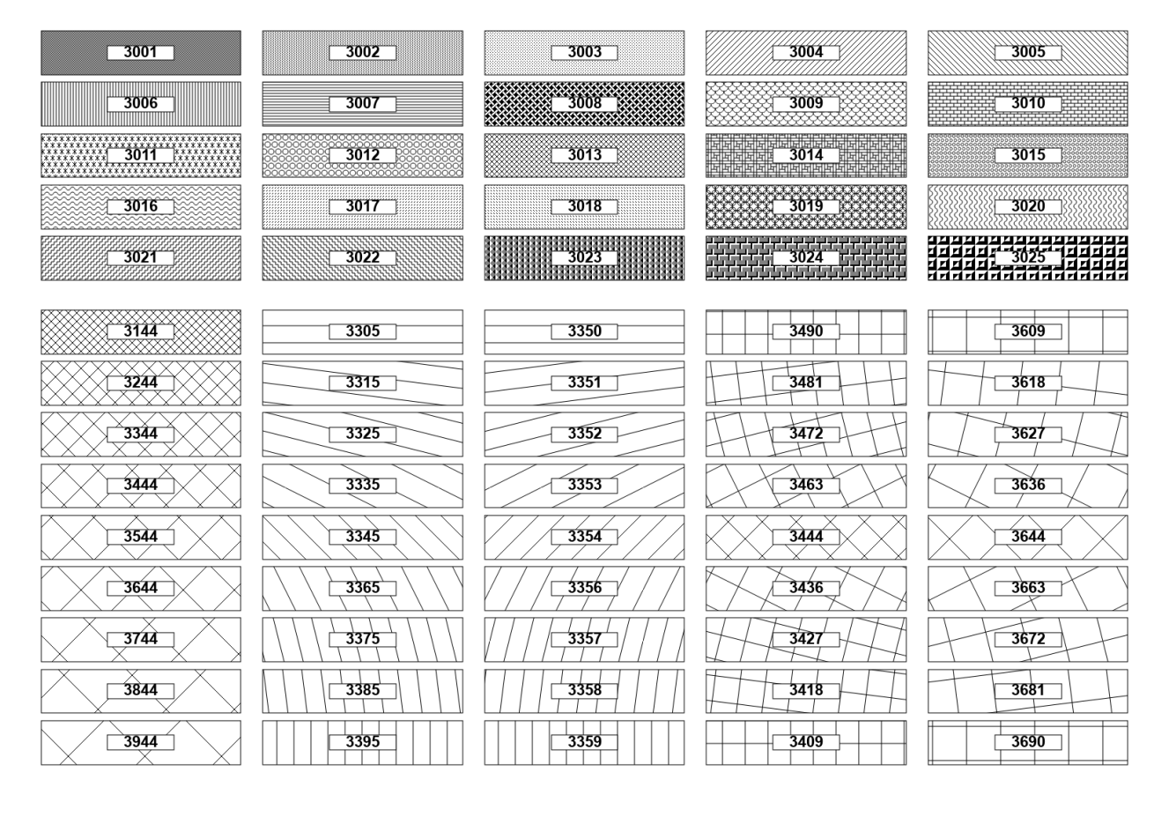
box1 for TH2



new TLatex parser, exclusion for TGraph



lego for THStack


All supported fill styles