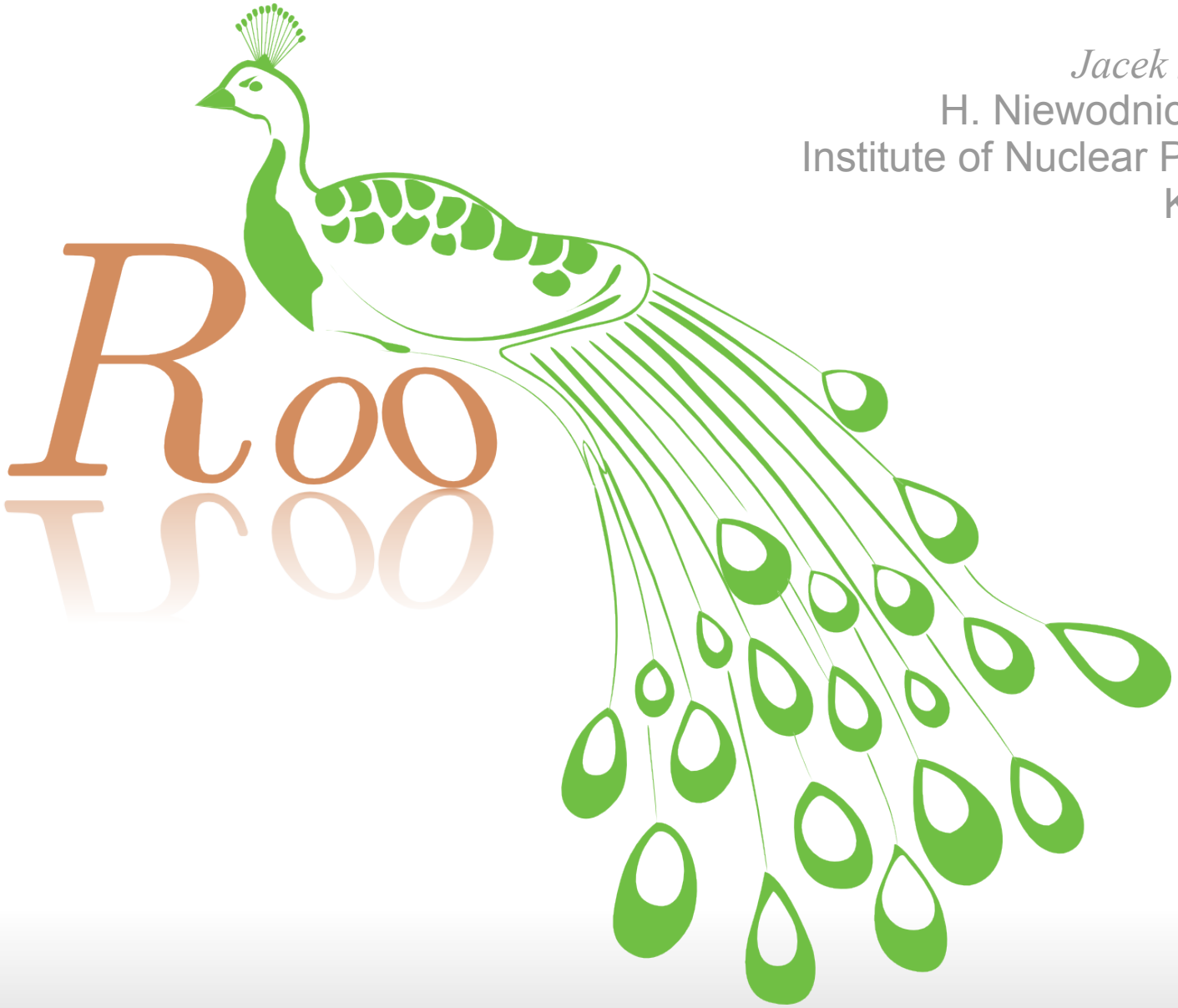
Jacek Stypuła
H. Niewodniczański
Institute of Nuclear Physics
Kraków



ROOT based Physics Analysis Workstation

What for?

We all know that ROOT performs quite well as a framework but horrible as a workstation

The goal is to make physics analysis with ROOT much more user friendly

IMHO Python is better for the end-user than C++

RooPAW? What's that?

It is a set of magic commands that extends IPython shell

It is written in Python and thus very easy to extend

It has started as a set of tools for my present Belle analysis

End-user problems:

1. Chains

ROOT

```
TChain h2_sig("h2");
.! ls sig*.root>sig.list
FILE *f = fopen("sig.list", "r");
char ftoch[256];
UInt_t i=0;
while (!feof(f))
{
   fscanf(f,"%s\n", &ftoch);
   h2_sig.Add((char*)&ftoch);
   ++i;
}
fclose(f);
```

I do not know how to do it better in ROOT...

:-(

RooPAW

```
chain h2_sig sig*.root h2
```

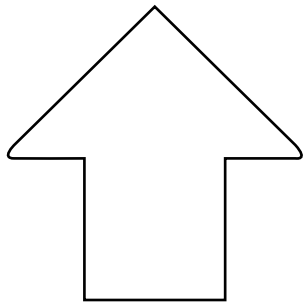
Simple and fast as it should be

:-)

2. Defining simple functions and cuts

ROOT

```
TCut cbest = "cl>0";
TCut call = cbest+"&&pt>.15&&!E<0";
TString p("sqrt(px**2+py**2+pz**2)");
tree->Draw("sqrt(E**2-"+p+"**2)", call,
"same");
```



This is the simplest but not very elegant way. You can try `tree->MakeClass` ending up with hundreads of lines of code that needs to be edited and compiled...

RooPAW

```
cut cbest cl>0
cut call cbest&&pt>.15.and.!E<0
form p sqrt(px**2+py**2+pz**2)
ntpl tree sqrt(E**2-p**2) call s
```

3. Normalization, macros (kumacs)

ROOT

```
TH1F h10("h10","",20,-10, 20);
h10.Scale(1/h10.Integral());
```

```
.x macro.C
```

You have to use C++ syntax in ROOT macros. You cannot put into a macro statements that you can execute interactively.

:-(

RooPAW

```
1d h10 20 -10 10
norm h10
```

```
exe kumac.py
```

RooPAW scripts use Python syntax extended with magic exactly as in the interactive mode.

:-)

4. Histogram TH1F from a function TF1

ROOT

```
TH1F hout("hout","", 30, -1, 1);
Double_t bin = 2./30.;
for (UInt_t i=0; i<32; ++i)
{
   Double_t x=1+(i-1)*bin+bin/2.;
   hout.SetBinContent(i,fun.Eval(x));
}
```

No, you cannot use `GetHistogram` to obtain 30 bins.
Do not even think about it...

:-(

RooPAW

```
1d hout fun 30 -1 1
```

Simple and fast...

:-)

5. Divided canvas

ROOT

```
c1->Divide(2,3);
c1_1->cd();
h1->Draw();
c1_2->Draw();
c1_2->cd();
h2->Draw();
```

You have to switch pads by hand

:-(

RooPAW

```
zone 2 3
hipl h1
h2.Draw
```

Simple and fast...

:-)

Status

As for now it is in an early stage of development (stable but not all desired magic is available an not foolproof) and there is no tarball released yet but it is installed here in the Instiute on the computing cluster (at KEK there is a problem with both ROOT and Python versions).

It is being developed along with my physics analysis...

The project webpage http://belle2.ifj.edu.pl/roopaw/
will be created soon