The Future of ROOT with R

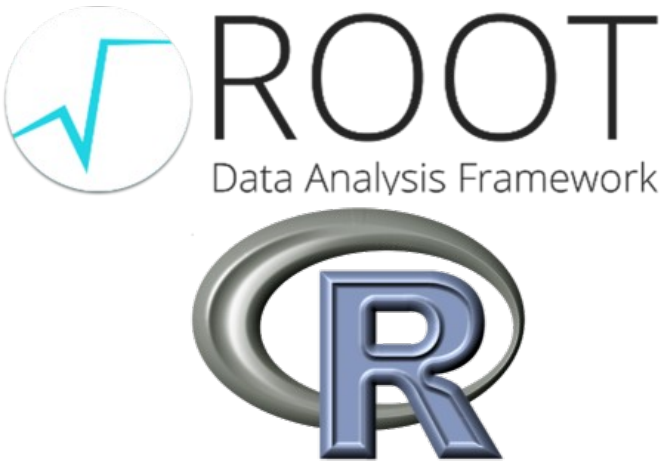
(ROOT Workshop 2015)

Omar Zapata (Metropolitan Institute Of Technology & University of Antioquia)

Lorenzo Moneta (CERN)

Sergei Gleyzer (University of Florida & CERN)

Summary

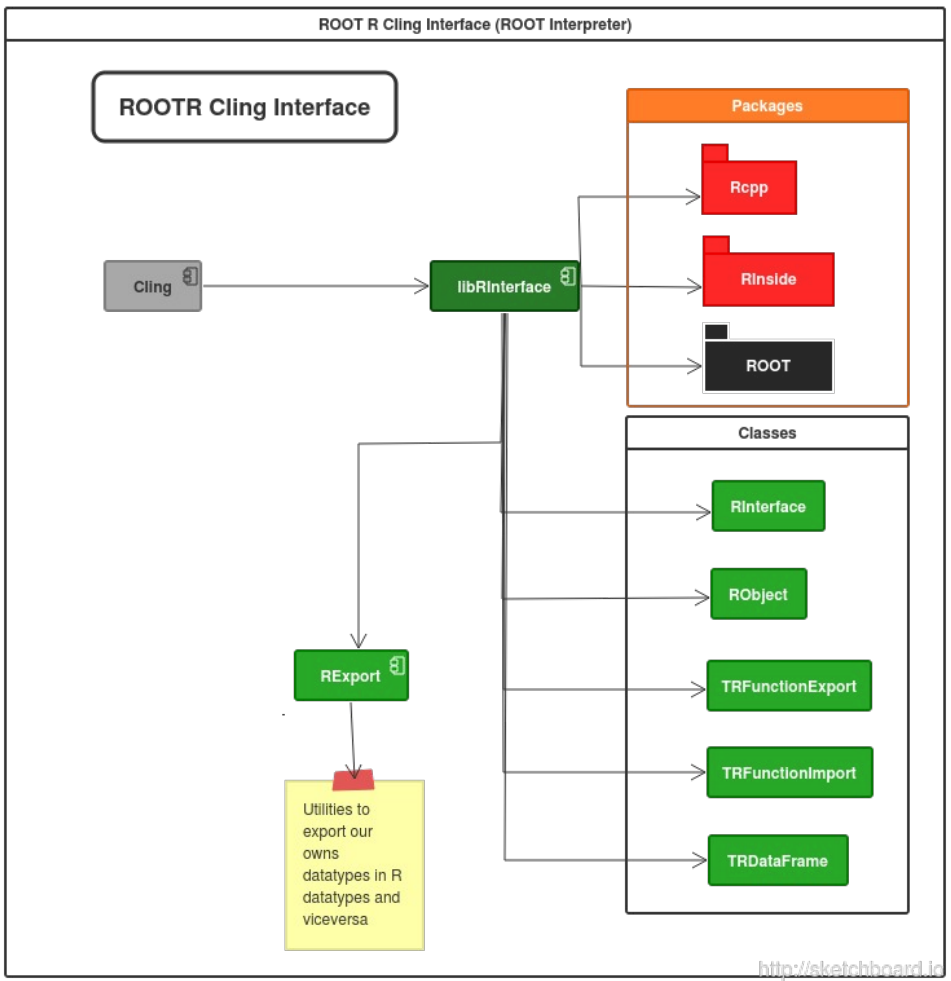


- ROOT with R
- Examples
- RMinimizer
- RMVA (R with TMVA)
- Future directions

ROOT with R

R was integrated to ROOT using:

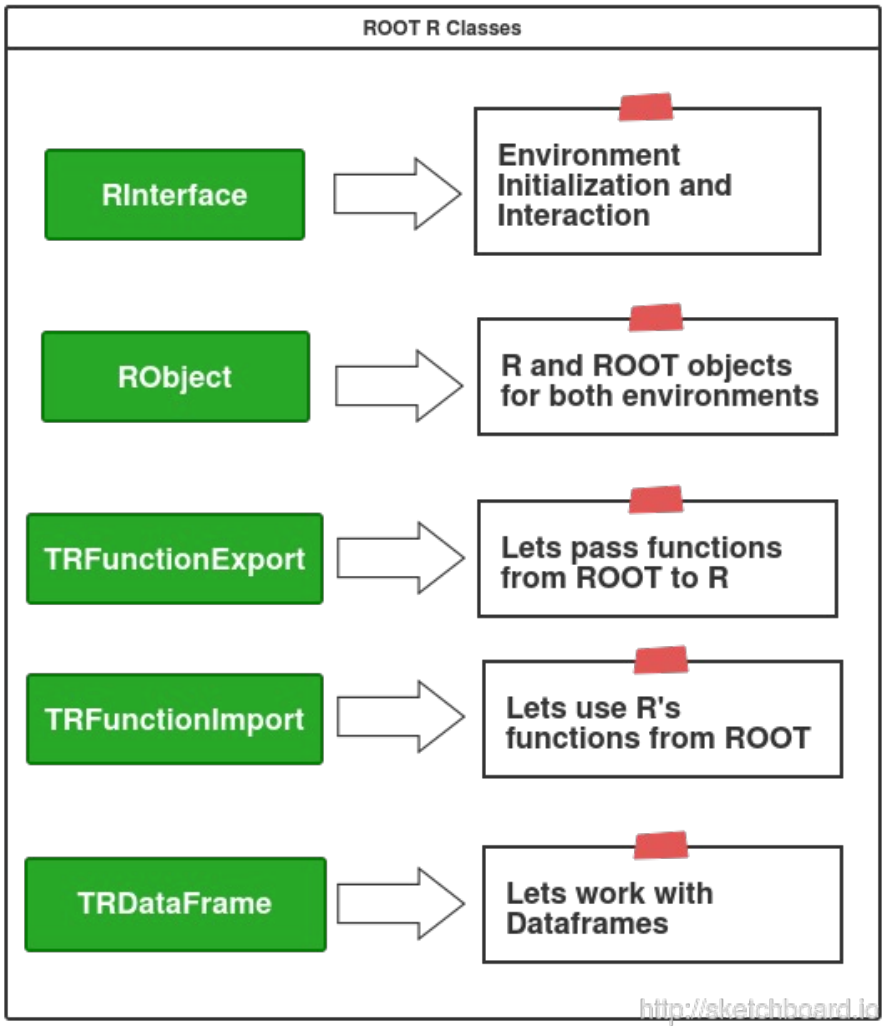
- The R api.
- Rcpp/Rinside packages that was writing in C++
- Macros defines to wrap ROOT datatypes.
- Globals variables to use native R types.



ROOT with R

ROOTR's classes

- The objects of these classes can be serialized and stored in R's files except **(TRFunctionExport)**
- It have overloaded operators to work like using streams, to pass and get objects from/to R.



Examples

TRInterface

```
TRInterface &r = TRInterface::Instance();
TMatrixD          mat(2,2);
vector<int>       vec={1,2,3};

//executing and getting
r["matrix(c(0.1,0.2,0.3,0.4),nrow=2)"]>>mat;
//print matrix result
mat.Print();
//passing variables
r["vec"]<<vec;
//executing command
r<<"print(vec)";
```

Output

```
root [0]
Processing trinterface.C...

2x2 matrix is as follows

    |      0    |      1    |
------------------------------
  0 |        0.1         0.3
  1 |        0.2         0.4

[1] 1 2 3
```

Examples

TRDataFrame

```
TRInterface &r = TRInterface::Instance();

vector<Double_t>  var1={0.1,0.2,0.3};
array<Int_t,3>    var2{ {1,2,3} };
TVectorD          var3(3,var1.data());
list<string>      names={"sig","bgk0","bgk1"};

TRDataFrame data;
data["var1"]  = var1;
data["var2"]  = var2;
data["var3"]  = var3;
data["names"] = names;
data.Print();
```

Output

```
root [0]
Processing dataframe.C...
  var1 var2 var3 names
1  0.1    1  0.1   sig
2  0.2    2  0.2  bgk0
3  0.3    3  0.3  bgk1
```

Examples

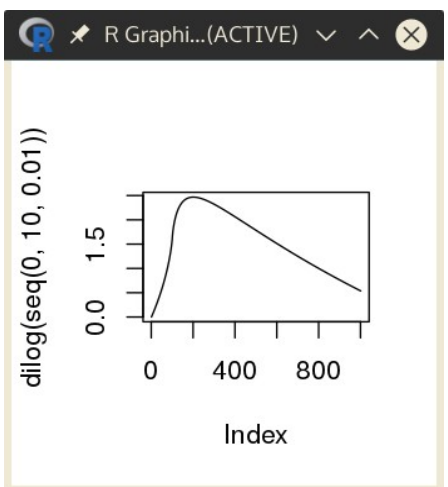
TRFunctionExport

```
template<class T> T fun(T x)
{
  return x+x;
}
TVectorD dilog(TVectorD x)
{
  TVectorD v(x.GetNoElements());
  for(int i=0;i<x.GetNoElements();i++) v[i]=TMath::DiLog(x[i]);
  return v;
}
void fexport()
{

  TRInterface &r = TRInterface::Instance();
  //assigning function with template class
  r["fun"]<<fun<TVectorF>;
  r<<"print(fun(c(0.5,1,1.5,2,2.5)))";
  //assigning function vectorized function
  r["dilog"]<<dilog;
  r<<"plot(dilog(seq(0,10,0.01)),type='l')";
}
```

Output

```
root [0]
Processing fexport.C...
[1] 1 2 3 4 5
```



Examples

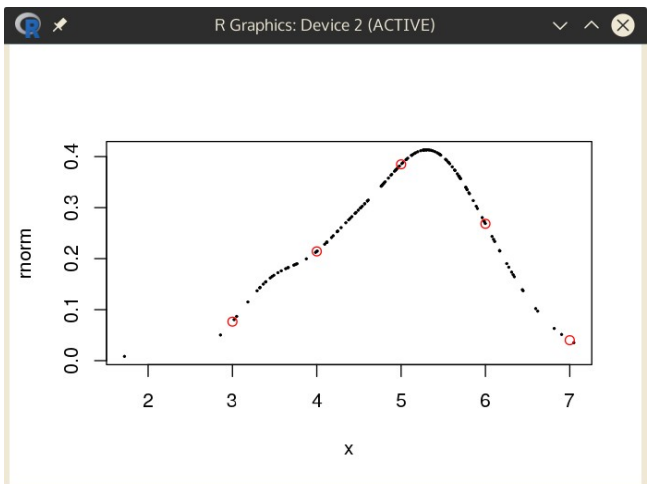
TRFunctionImport

Output

```
TRInterface &r = TRInterface::Instance();

TRFunctionImport c("c");
TRFunctionImport rnrom("rnorm");
TRFunctionImport approxfun("approxfun");
TRFunctionImport density("density");
TRFunctionImport integrate("integrate");
TRFunctionImport plot("plot");
TRFunctionImport points("points");

TRObject x = rnrom(150,5);
TRObject xnew = c(3,4,5,6,7);
TRFunctionImport pdf=approxfun(density(x));
plot(x,pdf(x),Label["type"]="p",Label["xlab"]="x",\
Label["ylab"]="rnorm",Label["pch"]=10, Label["cex"]=.2);
points(xnew,pdf(xnew),Label["col"]=2);
```



Examples

R

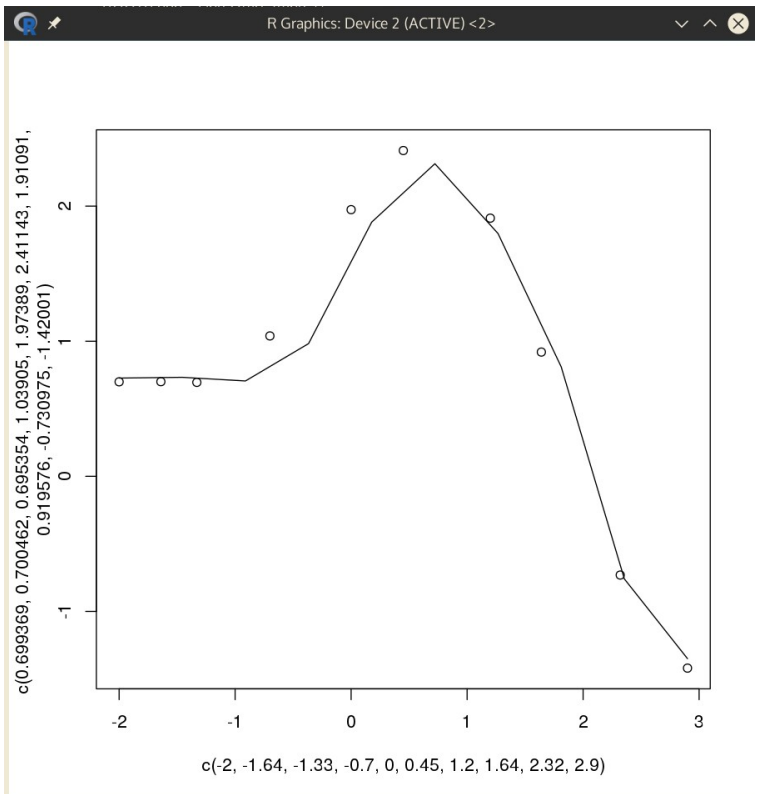
```
//original R code
xdata = c(-2,-1.64,-1.33,-0.7,0,0.45,1.2,1.64,2.32,2.9)
ydata = c(0.6993,0.7004,0.6953,1.039,1.973,2.411,1.910,0.9195,-0.7309,-1.420)
fit = nls(ydata ~ p1*cos(p2*xdata) + p2*sin(p1*xdata), start=list(p1=1,p2=0.2))
summary(fit)
confint(fit)
plot(xdata,ydata)
xgrid=seq(min(xdata),max(xdata),len=10))
lines(xgrid,predict(fit,xgrid))
```

C++

```
//ROOTR C++ code
TRDataFrame data;
data["xdata"] = c(-2,-1.64,-1.33,-0.7,0,0.45,1.2,1.64,2.32,2.9);
data["ydata"] = c(0.6993,0.7004,0.6953,1.039,1.973,2.411,1.910,0.9195,-0.7309,-1.420);
TRObject fit = nls(asformula("ydata ~ p1*cos(p2*xdata) + p2*sin(p1*xdata)"),\
                           Label["data"]=data, \
                           Label["start"]=list(Label["p1"]=1,Label["p2"]=0.2));
summary(fit);
confint(fit);
plot(data["xdata"],data["ydata"]);
TRObject xgrid=seq(min(data["xdata"]),max(data["xdata"]),Label["len"]=10);
lines(xgrid,predict(fit,xgrid));
```

Examples

Plot



Console Output

```
[omazapa] [tuxito] [~/ROOT/Workshop]$ root -l example.C
root [0]
Processing example.C...

Formula: ydata ~ p1 * cos(p2 * xdata) + p2 * sin(p1 * x
data)

Parameters:
   Estimate Std. Error t value Pr(>|t|)
p1 1.881851   0.027430   68.61 2.27e-12 ***
p2 0.700230   0.009153   76.51 9.50e-13 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.
1 ' ' 1

Residual standard error: 0.08202 on 8 degrees of freedo
m

Number of iterations to convergence: 7
Achieved convergence tolerance: 2.189e-06

Waiting for profiling to be done...
root [1]
```

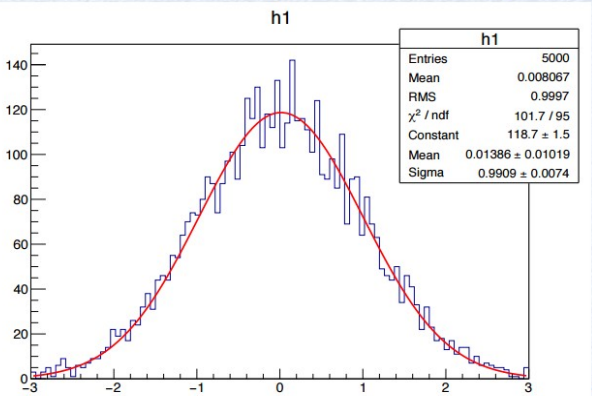
RMinimizer

ROOT plugin for Minimisation implemented using R

- Developed by Kirby Hermann (GSOC student 2014) gives access to R optimisation tools when fitting or multidimensional function minimisation.
- Based on R optim and optimx packages

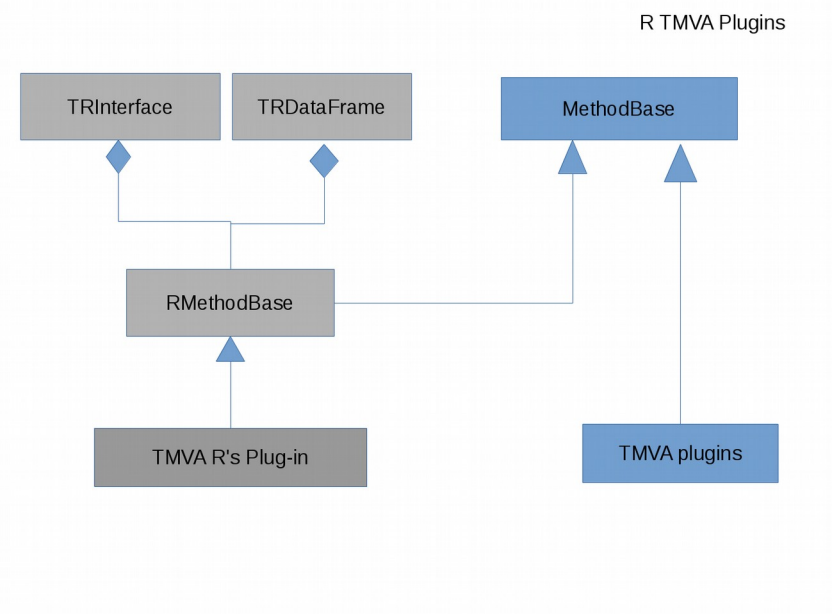
```
ROOT::Math::MinimizerOptions::SetDefaultMinimizer("RMinimizer","L-BFGS-B");
hist->Fit("gaus");
```

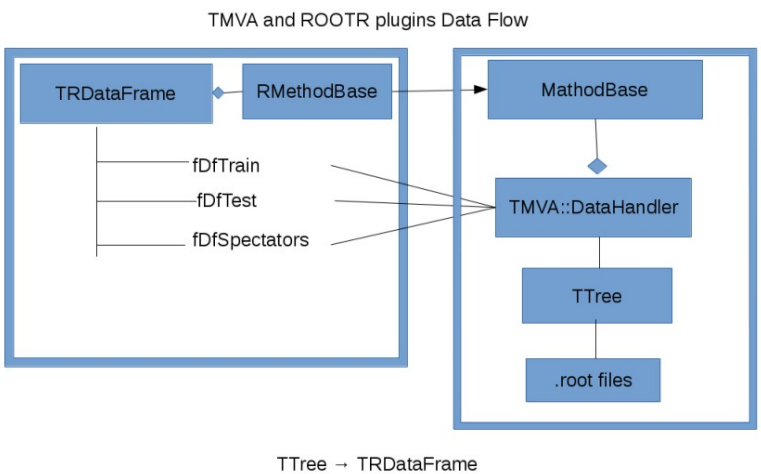
```
root [4] h1.Fit("gaus")
Value at minimum =101.673
****************************************
Minimizer is RMinimizer / L-BFGS-B
Chi2                      =      101.673
NDf                       =           95
NCalls                    =          265
Constant                  =      118.694   +/-   1.47659
Mean                      =    0.0138555   +/-   0.0101907
Sigma                     =     0.990906   +/-   0.00741443
```



RMVA (R with TMVA)

RMVA is a set of plugins for TMVA package based on ROOTR that consist in a set of classes that engage TMVA and allows new methods of classification and regression calling R's packages.



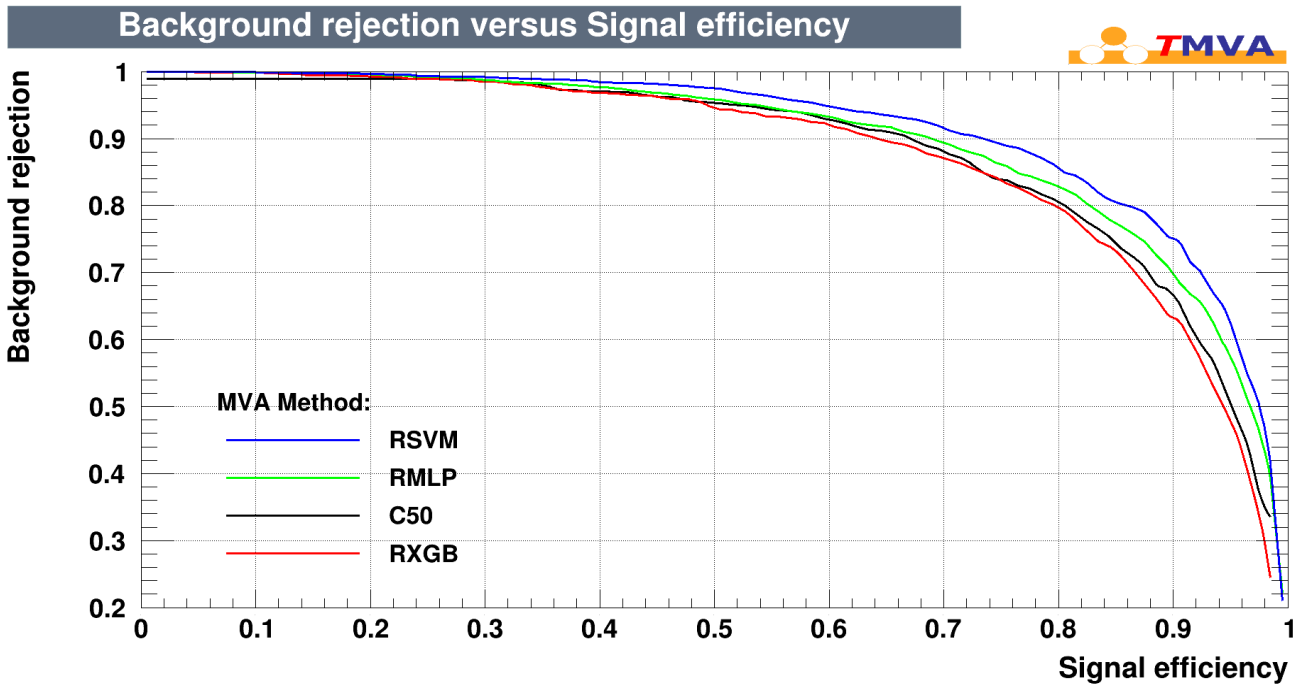


RMVA (R with TMVA)

- **(C50) C5.0** decision trees and rule-based models for pattern recognition.
- **(RSNNS)**Neural Networks in R using the Stuttgart Neural Network Simulator (SNNS)
- **(e1071)** Support Vector Machine can be used to carry out general regression and classification (of nu and epsilon-type), as well as density-estimation. A formula interface is provided.
- **eXtreme Gradient Boost** (R package xgboost) An optimized general purpose gradient boosting library.

It implements machine learning algorithms under the Gradient Boosting framework, including Generalized Linear Model (GLM) and Gradient Boosted Decision Trees (GBDT). XGBoost can also be distributed and scale to Terascale data

RMVA (R with TMVA)



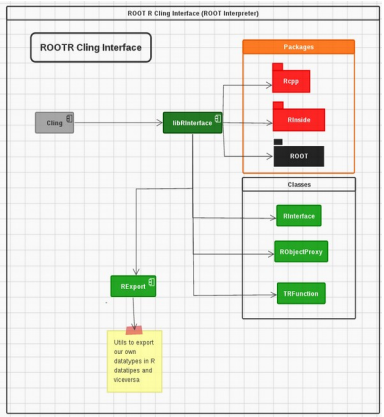
Evaluation results ranked by best signal efficiency and purity (area)

MVA Method:	Signal efficiency at bkg eff.(error): @B=0.01	@B=0.10	@B=0.30	ROC-integ.	Separation:	Significance:
RSVM	0.328(08)	0.735(08)	0.924(04)	0.913	0.526	1.355
RMLP	0.286(08)	0.689(08)	0.899(05)	0.897	0.481	1.310
C50	0.000(00)	0.671(08)	0.878(05)	0.881	0.462	1.253
RXGB	0.233(07)	0.643(08)	0.867(06)	0.875	0.434	1.194

Future Directions

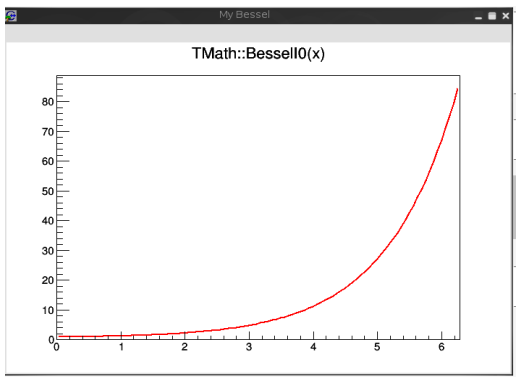
- Give to ROOT R support for c++11 and c++14 (Ex: lambda functions, std::function,smart pointers etc..)
- To create more classes to support other datatypes from R like vector, matrix and list. (TRVector,TRMatrix and TRList)
- To write new modules for RMVA (also use some that are paralleized in the package mlr)
- To write support to save ROOT R classes into ROOT files using TFile.
- To write a design to create a R's package for ROOT that lets to use ROOT from R directly. It can be implemented using reflection.

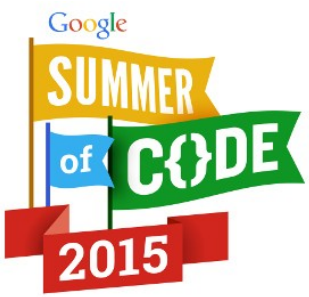
More Information




```
require(ROOT)

c1    <- TCanvas('c1','My Bessel')
bessel<- TF1('bessel','TMath::BesselI0(x)')
bessel$SetRange(0,2*pi)
bessel$Draw('') #plotting with ROOT's graphics system
c1$Update()
```





Websites

https://root.cern.ch/...r-interface

http://oproject.org

Thanks