ROpenTURNS

toward the [R] User Interface

Y. Richet & R. François

[R] in a few numbers

18

years of development. R is an adult.

[R] in a few numbers

4576

packages contributed to CRAN by the community covering many aspects of statistical computing

[R] and C++

- R is written in C
- R packages may contain C/C++ code through antique `.Call` interface
  - All R objects are opaque pointers (`SEXP`)
  - R provides a C API (macros, functions) to deal with them

```
SEXP add( SEXP x, SEXP y){
    SEXP res = PROTECT( allocVector( REALSXP, 1 ) ) ;
    REAL(res)[0] = REAL(x)[0] + REAL(y)[0] ;
    UNPROTECT(1);
    return res ;
}
```

- The API is boring and error prone

Enters Rcpp

- C++ Class hierarchy to map R types to C++ types
- NumericVector, IntegerVector, List, Environment, ...

```
// [[Rcpp::export]]
double add( double x, double y){
    return x + y ;
}

// [[Rcpp::export]]
double sum( NumericVector x){
    double res = 0.0 ;
    for( int i=0; i<x.size(); i++) res += x[i] ;
    return res ;
}
```

- No need to learn a complex and error prone api.

Rcpp Modules

```
// a c++ class
class Example{
    public:
        Example( double x, double y){ ... }
        double doStuff(){ ... }
} ;
```

```
// expose it
class<Example>( "Example" )
    .constructor<double,double>()
    .method( "doStuff", Example::doStuff )
;
```

```
# use it from R
x <- new( Example, 1.0, 2.0 )
x$doStuff( )
```

ROpenTurns 1.0

120

C++ classes from the OpenTurns C++ API

ROpenTurns Implementation

```
class_<AnalyticalResult>( "AnalyticalResult" )
    .constructor()
    SHOW(AnalyticalResult)
    METHOD(AnalyticalResult, getPhysicalSpaceDesignPoint )
    ...
;
class_<FORMResult>( "FORMResult" )
    .derives<AnalyticalResult>( "AnalyticalResult" )
    .constructor()
    .constructor<NumericalPoint,Event,Bool,OT::String>()
    METHOD(FORMResult,getEventProbability)
    ...
;
class_<SORMResult>( "SORMResult")
    .derives<AnalyticalResult>( "AnalyticalResult" )
    .constructor()
    .constructor<NumericalPoint,Event,Bool,OT::String>()
    METHOD(SORMResult,getEventProbabilityBreitung)
    ...
;
```

ROpenTurns Implementation

```
class_<SolverImplementation>( "SolverImplementation" )
        .constructor()
        .constructor<NumericalScalar,NumericalScalar,UnsignedLong>()
        // TODO: operator==
        SHOW(SolverImplementation)
        .method( "solve", ROpenTurns::NumericalScalar_solve_0 )
        .method( "solve", ROpenTurns::NumericalScalar_solve_1 )
        METHOD_GET_SET(SolverImplementation, AbsoluteError)
        METHOD_GET_SET(SolverImplementation, RelativeError)
        METHOD_GET_SET(SolverImplementation, MaximumFunctionEvaluation)
    ;

    class_<Bisection>("Bisection" )
        DERIVES(SolverImplementation)
        .constructor()
        .constructor<NumericalScalar, NumericalScalar, UnsignedLong>()
    ;
```

ROpenTurns Example

```
distributionE <- new( Beta, .93, 3.2, 2.8e7, 4.8e7 )
distributionF <- new( LogNormal, 30000, 9000, 15000, LogNormal.MUSIGMA )
distributionL <- new( Uniform, 250, 260 )
distributionI <- new( Beta, 2.5, 4.0, 3.1e2, 4.5e2 )

RS <- new( CorrelationMatrix, 4 ); RS$set( 2, 3, -.2 )
mat <- NormalCopula.GetCorrelationFromSpearmanCorrelation( RS )
copule <- new( Copula, new( NormalCopula, mat ) )

inputDistribution <- new( ComposedDistribution, list(
    E = distributionE, F = distributionF, L = distributionL, I = distributionI
) , copule )
inputDistribution$getDimension()
# [1] 4
inputRandomVector <- new( RandomVector, inputDistribution )
inputRandomVector$getRealization()
# class=NumericalPoint name=Unnamed dimension=4 values=[3.63475e+07,21373.8,253.306,435.587]
```

ROpenTurns Example

using an R function as a NumericalFunction

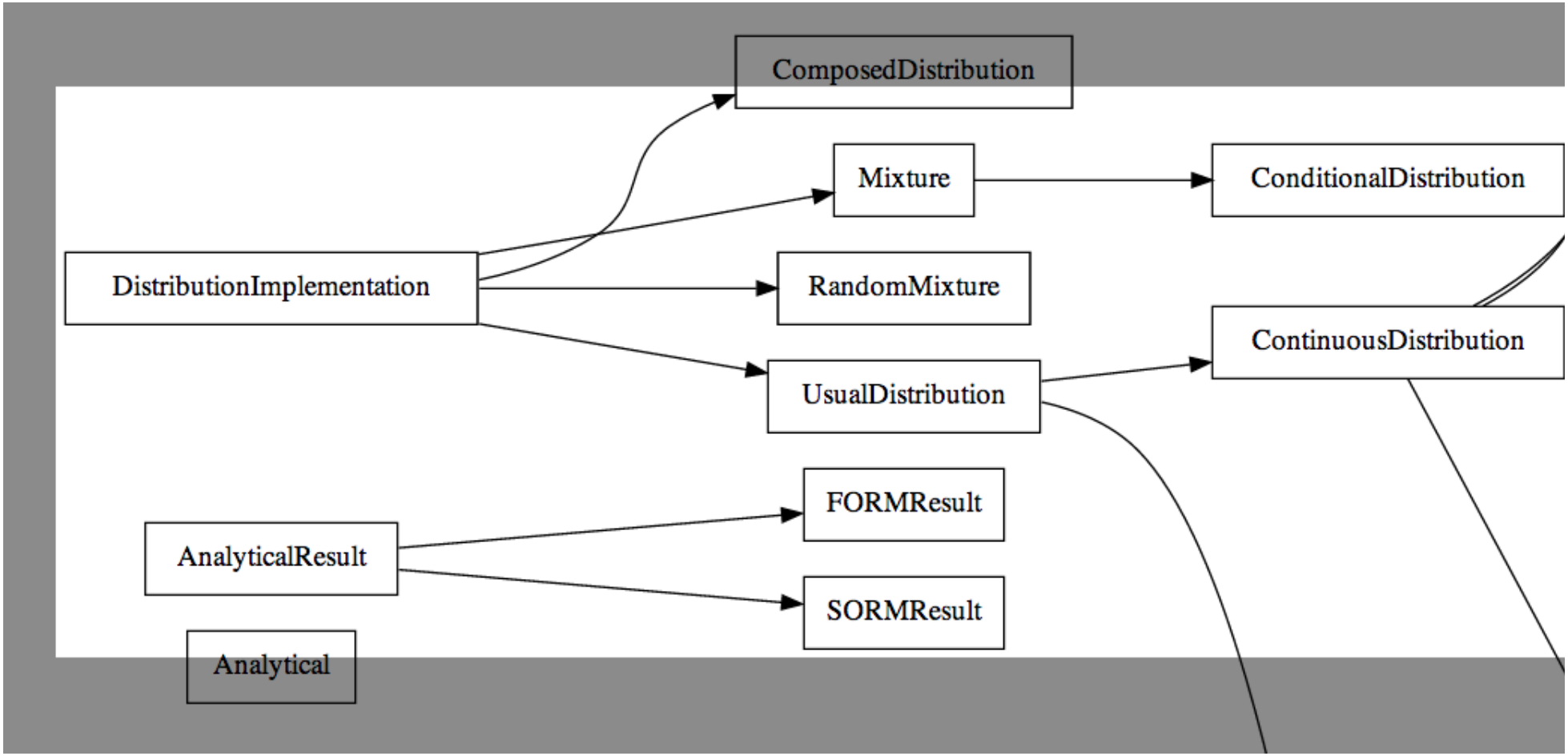
```
deviation_fun <- function(x){
   E = x[1] ; F = x[2] ; L = x[3] ; I = x[4]
   (F*L*L*L) / (3*E*I)
}
deviation <- asNumericalMathFunction( deviation_fun, 4, 1 )
outputVariableOfInterest <- new( RandomVector, deviation, inputRandomVector )
```

Or perhaps more directly:

```
deviation <- asNumericalMathFunction(
    function(x){
        E = x[1] ; F = x[2] ; L = x[3] ; I = x[4]
        (F*L*L*L) / (3*E*I)
    }
, 4, 1 )
```

ROpenTurns classes in version 1.0

Click the image for the full version



What's next

- More classes
- More Examples
- More testing
- More R like syntax