A new development cycle of the Statistical Toolkit

To live in the hearts we leave behind is not to die. (T. Campbell)
In memory of Paolo Viarengo

M. Batič[1,2], A. M. Paganoni[4], A. Pfeiffer[3], M. G. Pia[1], A. Ribon[3]

[1] INFN Sezione di Genova, Genova, Italy
[2] Jožef Stefan Institute, Ljubljana, Slovenia
[3] CERN, Geneva, Switzerland
[4] MOX - Dept. of Mathematics, Politecnico di Milano, Italy

Abstract

The Statistical Toolkit (ST) [1,2] is an open source system specialized in the statistical comparison of distributions. It addresses requirements common to different experimental domains, such as simulation validation (e.g. comparison of experimental and simulated distributions), regression testing in software development and detector performance monitoring.

Various sets of statistical tests have been added to the existing collection to deal with the one sample problem (i.e. the comparison of a data distribution to a function, including tests for normality, categorical analysis and the estimate of randomness). Improved algorithms and software design contribute to the robustness of the results. A simple user layer dealing with primitive data types (e.g. parsing comma-separated-values files) facilitate the use of the toolkit both in standalone analyses and in large scale experiments.

New development cycle

The new development cycle of the ST has been moved to SVN repository. To be as self-consistent as possible, the number of dependencies on external software systems has been minimized. The only proper requirement is the GNU Scientific Library [3]. In order to facilitate running the ST on (wide) variety of operating systems, the Cross Platform Make [4] build system is used. The ctest testing tool, distributed as a part of CMake, is used for unit testing.

The new development cycle of the toolkit has three different user layers. Without any external dependencies the user layer that reads comma separated list of values (csv ascii files) is built by default. Additional user layers, to AIDA [5] and ROOT [6], facilitating usage of ST using objects from the these toolkits (e.g. histograms and ntuples) as data that provides distributions. On properly set-up environments, either or both will be found by cmake when user specifies correct variables.

The ST now also comes with an extensive set of unitTests, which are meant to test the consistency of the statistical test for each new version of the ST.

Statistical tests in the ST

The ST started out as a statistical data analysis tool for the problem of comparing two distributions. The first development cycles of the ST delivered a wide set of a goodness-of-fit (GoF) tests, which are suitablefor regression testing, validation of simulation to the experimental data, comparison of expected vs. reconstructed distributions and comparison of data from different sources. The collection of implemented tests encompasses:

GoF test	Distribution Type	`<ComparisonAlgorithm>` Class

Anderson-Darling	`Binned`
	
	`Unbinned`
		`AndersonDarlingBinned`
		`AndersonDarlingBinnedApproximated`
		`AndersonDarlingUnbinned`
		`AndersonDarlingUnbinnedApproximated`

Chi-squared	`Binned`	`Chi2`
		`Chi2Approximated`
		`Chi2Integrating`

Fisz-Cramer-von-Mises	`Binned`
	`Unbinned`
	`Unbinned`
		`CramerVonMisesBinned`
		`CramerVonMisesUnbinned`
		`WeightedCramerVonMisesBuningUnbinned`

Girone	`Unbinned`	`Girone`
Goodman	`Unbinned`	`KolmogorovSmirnovApproximated`

Kolmogorov-Smirnov	`Unbinned`	`KolmogorovSmirnov`
		`WeightedADKolmogorovSmirnov`
		`WeightedBuningKolmogorovSmirnov`

Kuiper	`Unbinned`	`Kuiper`
Tiku	`Binned` `Unbinned`	`TikuBinned` `TikuUnbinned`
Watson	`Unbinned`	`Watson`

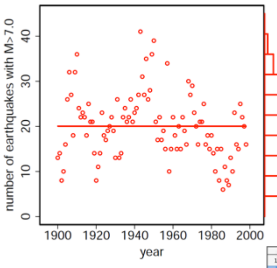
The new development cycle extends the ST with one sample goodness-of-fit tests, i.e. comparing data to reference functions, and tests for randomness. These tests provide complementary information to the existing tests: for instance, tests for randomness can highlight the presence of systematic effects in the distributions subject to comparison, which goodness-of-fit tests cannot detect. The extension of functionality of the ST also includes tests for categorical data. A list of new tests, some of which are already available in the new development version of the ST, is here:

GoF test	Distribution Type	Class
Wald-Wolfowitz runs test	`OneDimData` (signs -1/1)	`WaldWolfowitzTwoSamplesRunsTest` `WaldWolfowitzOneSampleRunsTest`
Wald-Wolfowitz test of randomness	`OneDimData`	`WaldWolfowitzOneSampleRandomnessTest`
Mann-Withney U test	`OneDimData`	`MannWithneyTwoSamplesTest` `MannWithneyOneSampleTest`
Fisher's Exact test	`TwoDimData` (2x2 matrix)	`FishersExact2x2Test`
Chi-squared paired test	Paired Values	`Chi2CurvesComparisonAlgorithm`

Example

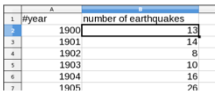
Is the number of earthquakes per year a random number?

We have two data samples. If the differences between the two were due only to statistical fluctuations, one would expect the differences to be a rather large number of short sequences (runs) of consecutive positive and negative values. Using Wald-Wolfowitz runs test [8] one can infer some systematic effects between the two samples. One can use the runs test to test the randomness of the numerical data around some given value (or function).



Distribution of earthquakes with magnitude larger than 7.0 over the last century. Data from [7].

Question: *Is the number of earth-quakes per year random around mean value?*



	A	B
1	#year	number of earthquakes
2	1900	13
3	1901	14
4	1902	8
5	1903	10
6	1904	16
7	1905	26

Importing the data from comma separated list of values (csv) ascii file

```
CsvReader* data=new CsvReader();
data->setDelimiter(";");   //default delimiter is comma
//read only second column (column numbers start from 0)
data->readColumnsFromFile("earthquakes.csv",1);
```

Construct OneDimData object, and make it binary around mean value

```
OneDimData num(data->getXData());      //construct one
                                       //dimensional data
num.Print();                           //print data
num.makeBinary(num.getMean());         //make runs around mean
num.Print();                           //print data
```

Run the Wald-Wolfowitz runs test and get the results

```
WaldWolfowitzOneSampleRunsTest* test =
           new WaldWolfowitzOneSampleRunsTest();
vector<double> ww = test->calculate(num);
cout<<"m = "<<ww.at(0)<<endl; //number of elements of first
cout<<"n = "<<ww.at(1)<<endl; //and second type (1/-1)
cout<<"r = "<<ww.at(2)<<endl; //number of runs
The results:     m = 52
                 n = 47
                 R = 31
```

Calculate the p-value from given test results

```
WaldWolfowitzQualityChecker* qc =
           new WaldWolfowitzQualityChecker();
ComparisonResult result=qc->quality(ww,n+m);
cout<<"P[exact] = "<<result.quality()<<endl;

The result:      P[exact] = 0.000109172
```

The calculated p-value is 0.00011. So: can we answer the question? No. But we can say that the exact p-value is small enough to have strong statistical evidence to reject the null hypothesis that in the 20th century the number of earthquakes with magnitude above 7.0 is not randomly distributed around their mean value.

Bibliography

[1] G.A.P. Cirrone et al., A Goodness-of-Fit Statistical Toolkit, IEEE TNS, vol. 51, no. 5, (2004), pp. 2056-2063.
[2] B. Mascialino et al., New Developments of the Goodness-of-Fit Statistical Toolkit, IEEE TNS, vol. 53, no. 6, (2006), pp. 3834-3841.
[3] GSL - GNU Scientific Library, www.gnu.org/software/gsl/, Accessed 2012.
[4] Cross-platform Make system, www.cmake.org.
[5] G. Barrand et. al, Abstract interfaces for data analysis: Component architecture for data analysis tools," in Proc. CHEP Int. Conf. Computing in High Energy and N uclear Physics, Beijing, China, 2001, pp. 215–218.
[6] R. Brun and F. Rademakers, ROOT - An Object Oriented Data Analysis Framework, Nucl. Inst. & Meth. in Phys. Res. A 389 (1997) pp. 81-86.
[7] Hyndman, R.J, Time Series Data Library, robjhyndman.com/TSDL. May 2012.
[8] A. Wald and J. Wolfowitz, On a test whether two samples are from the same population, Ann. math. Statist., vol. 11, (1940) pp. 147-162.
See also the associated contribution to the conference proceedings.