Test Report on the Module statistics_lib.distribution_classes

Conventions

Each test is defined following the same format. Each test receives a unique test identifier and a reference to the ID(s) of the requirements it covers (if applicable). The goal of the test is described to clarify what is to be tested. The test steps are described in brief but clear instructions. For each test it is defined what the expected results are for the test to pass. Finally, the test result is given, this can be only pass or fail.

The test format is as follows:

Test Identifier: TEST-[I/A/D/T]-XYZ

Requirement ID(s): REQ-uvw-xyz

Verification method: I/A/D/T

Test goal: Description of what is to be tested

Expected result: What test result is expected for the test to pass

Test steps: Step by step instructions on how to perform the test

Test result: PASS/FAIL

The test ID starts with the fixed prefix 'TEST'. The prefix is followed by a single letter, which defines the test type / verification method. The last part of the ID is a 3-digits *hexadecimal* number (0..9|A..F), with the first digit identifing the module, the second digit identifing a class / function, and the last digit - the test ordering number for this object. E.g. 'TEST-T-112'. Each test type has its own counter, thus 'TEST-T-112' and 'TEST-A-112' tests are different entities, but they refer to the same object (class or function) within the same module.

The verification method for a requirement is given by a single letter according to the table below:

Term	Definition
Inspection (I)	Control or visual verification
Analysis (A)	Verification based upon analytical evidences
Test (T)	Verification of quantitative characteristics with quantitative measurement
Demonstration (D)	Verification of operational characteristics without quantitative measurement

Tests definition (Analysis)

Test Identifier: TEST-A-400

Requirement ID(s): REQ-FUN-400

Verification method: A

Test goal: All required functionality is implemented and performs correctly.

Expected result: All required classes implenting continuous and discrete random distributions are present and function as expected, i.e. all TEST-T-4xy tests defined in this document are passed.

Test steps: Analyze the source code of the module distribution_classes as well as of the unit-test module /Tests/UT004_distribution_classes. Execute the mentioned unit-test module.

The implementation of the PDF, CDF and QF of the distributions is checked against the NIST documentation - *Engineering Statistics Handbook* located at NIST/SEMATECH e-Handbook of Statistical Methods, http://www.itl.nist.gov/div898/handbook/, Digital Object Indentifier https://doi.org/10.18434/M32189.

The Student`s t-distribution, Chi-squared distribution and F-distributions are also verified in terms of CDF / QF vs the tabulated values listed therein.

Test result: PASS

Tests definition (Test)

Test Identifier: TEST-T-400

Requirement ID(s): REQ-FUN-401

Verification method: T

Test goal: Instantiation of a class - assignment of parameters.

Expected result: A class being tested can be instantiated, provided that the required number of the required data type and value's range attributes are passed into the initialization method (except for Z-distribution, which is w/o parameters). The passed arguments are properly assigned to the parameters of the distribution.

Test steps: Instantiate the class being tested with random but proper values of the parameters. Check that no exception is raised. Check that the parameters of the distribution are properly assiged, and the statistical properties of the distribution are as expected for the given parametes. Repeat the process several time. This test should be performed with all implemented distribution classes.

Test result: PASS

Test Identifier: TEST-T-401

Requirement ID(s): REQ-FUN-401

Verification method: T

Test goal: Instantiation of the class - all required attributes are present.

Expected result: All required attributes: methods and properties - are present in an instance of the class regardless of the passed parameters of the distribution, as long as they are of the proper values.

Test steps: Instantiate the class being tested with random but proper values of the parameters. Check that all requried attributes are present. Repeat the process several time. This test should be performed with all implemented distribution classes.

Test result: PASS

Test Identifier: TEST-T-402

Requirement ID(s): REQ-FUN-402

Verification method: T

Test goal: Variability of the distribution parameters

Expected result: The parameters of the distribution (except Z-distribution) can be modified at any time via the respective setter properties. The statistical properties of the distribution change accordingly.

Test steps: Instantiate the class being tested with random but proper values of the parameters. Assign new proper value to the all parameters of the distribution. Check that the parameters are changed, and the statistical properties are calculated as expected for the set parameters. Repeat several times with different (random) values of the parameters. This test should be performed with all implemented distribution classes, except for Z_Distribution.

Test result: PASS

Test Identifier: TEST-T-403

Requirement ID(s): REQ-FUN-403

Verification method: T

Test goal: Read-only access to the statistical properties of the distribution

Expected result: The class' properties representing the statistical properties of the distribution, can be accessed for reading, but they cannot be deleted or modified (assigned to) on an instance of that class.

Test steps: Instantiate the class being tested with random but proper values of the parameters. Attemt to modify and to delete each of the properties, unless it is also a parameter of the distribution (e.g. mean and sigma for Gaussian). Check that AttributeError or its sub-class is raised each time. This test should be performed with all implemented distribution classes.

Test result: PASS

Test Identifier: TEST-T-404

Requirement ID(s): REQ-FUN-404

Verification method: T

Test goal: Proper implementation of pdf() method.

Expected result: The method pdf() accepts any real number as its single argument and returns a non-negative real number, which is:

- for continuous distributions:
  - the positive value of the probability density function at this value of the random variable as long as it is within the range of the accepted values (Min to Max) for the distibution
  - zero (0) for the values outside the accepted range
- for discrete distributions:
  - the positive value of the probability mass function at this value of the random varibale as long as the value is an integer number within the range of accepted values (Min to Max)
  - zero (0) for any real number value outside the accepted range, or any non-integer real number within the accepted range

Test steps: Instantiate the class being tested with random but proper values of the parameters. Call the method with different random values of its argument. Check that the returned results is as expected (see DE002 document). The returned values can be checked either against 'manually calculated' ones (for the simple formulas) or against the tabulated values, in which case the specific unittest method must provide reference to the source of data. If applicable (except Z-distribution), change the parameters of the distributio and repeat the test.

This test should be performed with all implemented distribution classes.

Test result: PASS

Test Identifier: TEST-T-405

Requirement ID(s): REQ-FUN-405

Verification method: T

Test goal: Proper implementation of cdf() method.

Expected result: The method cdf() accepts any real number as its single argument and returns a non-negative real number, which is:

- for continuous distributions - the value of the cummulative density function, which is a monotonically growing from 0 to 1 function for the values within the range of the accepted values (Min to Max) for the distibution, and strictly 0 for values <= Min, and strictly 1 for values >= Max
- for discrete distributions - the value of the cummulative density function, which is a monotonically growing from 0 to 1 function for the values within the range of the accepted values (Min to Max) for the distibution, and strictly 0 for values < Min, and strictly 1 for values >= Max

Test steps: Instantiate the class being tested with random but proper values of the parameters. Call the method with different random values of its argument. Check that the returned results is as expected (see DE002 document). The returned values can be checked either against 'manually calculated' ones (for the simple formulas) or against the tabulated values, in which case the specific unittest method must provide reference to the source of data. If applicable (except Z-distribution), change the parameters of the distributio and repeat the test.

This test should be performed with all implemented distribution classes.

Test result: PASS

Test Identifier: TEST-T-406

Requirement ID(s): REQ-FUN-406

Verification method: T

Test goal: Proper implementation of qf() and getQuantile() methods.

Expected result: The method qf() accepts any floating point number p in the range (0, 1) and returns a real number x, for which the following relations are valied:

- continuous distributions: Min < x < Max (returned value is within the range of the random variable) and cdf(x) = p
- discrete distributions:
  - Min <= x < Max if p >= pdf(Min), where $cdf(\lfloor x \rfloor) \leq p \leq cdf(\lfloor x \rfloor + 1)$
  - Min -1 < x < Min if 0 < p < pdf(Min)

The method getQuantile(k, m) accepts two positive integer numbers 0 < k < m and returns the value equal to qf(k/m).

Test steps: Instantiate the class being tested with random but proper values of the parameters. Call the qf() method with different random values of its argument. Check that the returned results is as expected (see DE002 document). The returned values can be checked either against 'manually calculated' ones (for the simple formulas) or against the tabulated values, in which case the specific unittest method must provide reference to the source of data.

Call the method getQuantile(k, m) with different random values of the arguments and compare the returned value with the result of the call qf(k/m).

If applicable (except Z-distribution), change the parameters of the distributio and repeat the test.

This test should be performed with all implemented distribution classes.

Test result: PASS

Test Identifier: TEST-T-407

Requirement ID(s): REQ-AWM-400

Verification method: T

Test goal: Rejection of the arguments of improper data types.

Expected result: TypeError or its sub-class exception is raised upon passing improper data type argument(s) to:

- Initialization method - not a real number for all classes, not an integer for some (see DE002)
- pdf() and cdf() methods - not a real number for all classes
- qf() - not a floating point number for all classes

- getQuantile() - not an integer (any of the two argument) - for all classes
- getHistogram() - not a real number for the first two arguments, not an integer for the third argument - all classes

Concerning the assigment to the setter properties (parameters of the distribution) the same rules are applied as for the instantiation of the class.

Test steps: Try to instantiate the class being tested with different improper data types of each of the parameters in turn. Check that the expected exception is raised each time. Repeat with the different improper data types.

Instantiate the class being tested with random but proper values of the parameters. Try to assign different improper data types values to the setter properties. Check that the expected exception is raised each time. Repeat with the different improper data types. Try to pass different improper data type argument(s) of the listed methods. Check that the expected exception is raised each time. Repeat with the different improper data types. This test should be performed with all implemented distribution classes.

Test result: PASS

Test Identifier: TEST-T-408

Requirement ID(s): REQ-AWM-401

Verification method: T

Test goal: Rejection of the arguments of proper data types but improper values.

Expected result: TypeError or its sub-class exception is raised upon passing proper data type argument(s) of inapproptiate values to:

- Initialization method - depending on the distribution (see DE002)
- qf() - not a floating point number within (0, 1) range - for all classes
- getQuantile() - any of the arguments is not positive, or the first argument is greater than or equal to the second argument - for all classes
- getHistogram() - the third argument is < 2 (integer), or the first argument is greater than or equal to the second argument - for all classes

Concerning the assigment to the setter properties (parameters of the distribution) the same rules are applied as for the instantiation of the class.

Test steps: Try to instantiate the class being tested with different improper values (but proper data type) of each of the parameters in turn. Check that the expected exception is raised each time. Repeat with the different improper values.

Instantiate the class being tested with random but proper values of the parameters. Try to assign different improper values (but proper data type) to the setter properties. Check that the expected exception is raised each time. Repeat with the different improper values. Try to pass different improper values (but proper data type) argument(s) of the listed methods. Check that the expected exception is raised each time. Repeat with the different improper values. This test should be performed with all implemented distribution classes.

Test result: PASS

Tests definition (Demonstration)

Test Identifier: TEST-D-400

Requirement ID(s): REQ-FUN-407, REQ-FUN-408

Verification method: D

Test goal: Check implementation of the methods *random*() and *getHistogram*()

Expected result: A historgam of the distribution itself (method *getHistogram*()) is similar (almost equal) in shape to the step-wise approximation of the shape of PDF of the distribution. A histogram of a large sample of random numbers generated by the corresponding distribution class resembles the both previous shapes.

Test steps: Perform the following procedure with all distributions to be tested.

- Instantiate the respective class with arbitrary chosen parameters
- Generate a sequence of 10000 random number with this instance
- Instantiate 1D statistics class (module *data_classes*) with that sequence
- Generate a histogram of the sample with the default 20 bins
- Normalize the values (frequences) in each bin by the length of the sequence 10000
- Determine the minimum and maximum central values of the bins and bin width
- Generate the histrogram of the entire distribution (tested class) using the determined minimum and maximum central values and the number of bins - 20
- For each bin with the cental value x_k (and the same width *S*) manually calculate the value
  - for a continuous distribution $S * [pdf(x_k - s/2) + pdf(x_k) + pdf(x_k + s/2)]/3$
  - for a discrete distribution $S * pdf(\mathbf{int}(\mathtt{round}(x_k)))$
- Print out the results of the calculation (3 histograms) on the screen as 6-columns table and compare

This test should be performed with all implemented distribution classes.

Test result: PASS

Traceability

For traceability the relation between tests and requirements is summarized in the table below:

Requirement ID	Covered in test(s)	Verified [YES/NO])
REQ-FUN-400	TEST-A-400	YES
REQ-FUN-401	TEST-T-400, TEST-T-401	YES
REQ-FUN-402	TEST-T-402	YES
REQ-FUN-403	TEST-T-403	YES
REQ-FUN-404	TEST-T-404	YES
REQ-FUN-405	TEST-T-405	YES

Requirement ID	Covered in test(s)	Verified [YES/NO])
REQ-FUN-406	TEST-T-406	YES
REQ-FUN-407	TEST-D-400	YES
REQ-FUN-408	TEST-D-400	YES
REQ-AWM-400	TEST-T-407	YES
REQ-AWM-401	TEST-T-408	YES

Software ready for production [YES/NO]	Rationale
YES	All tests are passed