Requirements for the Module statistics_lib.distribution_classes

Conventions

Requirements listed in this document are constructed according to the following structure:

Requirement ID: REQ-UVW-XYZ

Title: Title / name of the requirement

Description: Description / definition of the requirement

Verification Method: I / A / T / D

The requirement ID starts with the fixed prefix 'REQ'. The prefix is followed by 3 letters abbreviation (in here 'UVW'), which defines the requiement type - e.g. 'FUN' for a functional and capability requirement, 'AWM' for an alarm, warnings and operator messages, etc. 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 requirement ordering number for this object. E.g. 'REQ-FUN-112'. Each requirement type has its own counter, thus 'REQ-FUN-112' and 'REQ-AWN-112' requirements 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

Functional and capability requirements

Requirement ID: REQ-FUN-500

Title: Functionality implemented by the module (scope)

Description: The module should implement the following special mathematical functions:

- Calculation of k-permutations (n permute k) and combinations (n choose k) - to cover the older Python interpreters v < 3.8
- Calculation of the inverse error function
- Calculation of beta function
- Calculation of incomplete beta function as well as of the regularized version of the same function
- Calculation of incomplete lower and upper gamma functions as well as of the regularized versions of the same functions

All these functions should perform calculations correctly.

Verification Method: A

Requirement ID: REQ-FUN-510

Title: Permutations function

Description: The function *permutation*(n, k) accepts two non-negative integer numbers 0 <= k <= n and returns an integer number equal to n! / (n-k)!

Verification Method: T

Requirement ID: REQ-FUN-520

Title: Combinations function

Description: The function *combination*(n, k) accepts two non-negative integer numbers 0 <= k <= n and returns an integer number equal to n! / [k! (n-k)!]

Verification Method: T

Requirement ID: REQ-FUN-530

Title: Inverse error function

Description: The function *inv_erf*(y) accepts an integer (0) or floating point number -1 < y < 1 and returs a floating point number x such, that erf(x) = y.

Verification Method: T

Requirement ID: REQ-FUN-540

Title: Beta function

Description: The function *beta*(x, y) accepts two positive floating point or integer numbers x > 0 and y > 0 and returns the value of the *beta* functions, which is a positive floating point number, i.e. gamma(x) * gamma(y) / gamma(x+y), where *gamma*() is the complete gamma function. The function *log_beta*(x,y) returns the natural logarithm of the same value, i.e. ln(beta(x,y)).

Verification Method: T

Requirement ID: REQ-FUN-550

Title: Incomplete beta functions

Description: The function *beta_incomplete*(z, x, y) should return the value of the incomplete beta function B(z; x, y), which is a positive floating point number. The function *beta_incomplete_reg*(z, x, y) should return the value of the regularized incomplete beta function Iz(x,y) = B(z; x, y) / B(x,y), which is a floating point number in the range (0, 1). The function *log_beta_incomplete*(z, x, y) should return ln(beta_incomplete(z, x, y)). All these functions must accept the positive integer or floating point second and third arguments (x and y), whilst the first argument (z) must be a floating point number or an integer in the closed range [0, 1].

Verification Method: T

Requirement ID: REQ-FUN-560

Title: Incomplete gamma functions

Description: The function *lower_gamma*(x > 0, y >= 0) should return the value of the lower incomplete gammma function $\gamma(x, y)$, which is a non-negative floating point number. The function *log_lower_gamma*(x > 0, y > 0) should return the value of ln(lower_gamma(x, y)). The function *lower_gamma_reg*(x > 0, y >= 0) should return the value of regularized lower incomplete gamma function $P(x,y) = \gamma(x,y) / \Gamma(x)$, which is a floating point number in the range [0, 1).

The function *upper_gamma*(x > 0, y >= 0) should return the value of the upper incomplete gammma function $\Gamma(x, y)$, which is a positive floating point number. The function *log_upper_gamma*(x > 0, y > 0) should return the value of ln(upper_gamma(x, y)). The function *upper_gamma_reg*(x > 0, y >= 0) should return the value of the regularized upper incomplete gamma function $Q(x,y) = \Gamma(x,y) / \Gamma(x)$, which is a floating point number in the range (0, 1].

Verification Method: T

Alarms, warnings and operator messages

Requirement ID: REQ-AWM-500

Title: Improper type of an agrument

Description: The **TypeError** or its sub-class should be raised if an improper data type argument is passed into any function. Specifically,

- Any data type of argument(s) except **int** for *permutation*() and *combinations*() functions
- Any data type of argument(s) except **int** or **float** for all other functions

Verification Method: T

Requirement ID: REQ-AWM-501

Title: Improper value of an agrument

Description: The **ValueError** or its sub-class should be raised if an argument of a proper data type, but of unacceptable value is passed into any function. Specifically,

- Any of the arguments is < 0 for for *permutation*() and *combinations*() functions, OR
- k > n for *permutation*() and *combinations*() functions
- The argument of *inv_erf*() function is >= 1 or <= -1
- The second or / and the third arguments (x and y) of the incomplete beta functions is <= 0, OR

- The first argument (z) of the the incomplete beta functions is not in the range [0, 1]
- The first argument (x) of the incomplete gamma functions is <= 0, OR
- The second argument (y) of the incomplete gamma functions is < 0 (not logarithmic), OR
- The second argument (y) of the logarithmic incomplete gamma functions is <= 0

Verification Method: T