NEON ALGORITHM THEORETICAL BASIS DOCUMENT: QA/QC DATA VALIDATION AND PLAUSIBILITY TESTING OF TOS and AOS FIELD and LAB DATA

PREPARED BY	ORGANIZATION	DATE
Natalie Robinson	FSU	05/14/2014
Katie Jones	FSU	05/14/2014
Sarah Elmendorf	DPS	05/14/2014
Kate Thibault	FSU	05/14/2014
Jeff Taylor	FIU	05/14/2014

APPROVALS (Name)	ORGANIZATION	APPROVAL DATE
Dave Tazik	SCI	02/06/2015

RELEASED BY (Name)	ORGANIZATION	RELEASE DATE
Judy Salazar	CM	02/11/2015

See Configuration Management System for approval history.

Change Record

REVISION	DATE	ECO #	DESCRIPTION OF CHANGE
A	02/11/2015	ECO-01825	Initial release

TABLE OF CONTENTS

LIST OF TABLES AND FIGURES

1 DESCRIPTION

1.1 Purpose

This document details validity and plausibility algorithms that are part of the automated Quality Assurance/Quality Control plan for organismal and observational data streaming from the Terrestrial and Aquatic Observation Systems (TOS and AOS respectively). The tests specified in this document will be called by Data Product specific ATBDs, discussed in the algorithm descriptions here as the 'referring ATBD'. Specifically, each referring ATBD will provide site and/or system-specific parameters that are needed in order to perform the automated test routines described herein, and to thus check the validity and plausibility of reported observations for many of the Level 0 (L0) Data Products.

Initial operations (L0) data collection may occur prior to the implementation of automated testing. In this case, the tests described in this document will then be applied to these L0 data after collection and data transcription, in order to verify that reported values are valid and plausible. For this latter category of plausibility tests, realistic low and high data values, and directions and magnitudes of change across successive sampling bouts, are required in order to verify that the reported data fall within a plausible range. Such 'plausibility values' may be defined in two ways: 1) through historical data obtained from public sources, when available, or 2) through calculations based off of large quantities of raw field data. In cases where historical data are unavailable and sufficient raw data have not yet been collected, plausibility tests cannot be performed until a later date (when more L0 data have been collected). As a result, while this document describes the theory of both data validation and plausibility tests, algorithmic implementation steps are only made available for validation tests. Such algorithmic implementation steps will be expanded in the future to include quantitative plausibility tests.

The application of the tests described in this document will vary by data product, as will be specified in referring ATBDs. Likewise, each referring ATBD will specify the variables to which to apply each test, and the temporal resolution at which the test is relevant (e.g. per bout, per year). Many of the tests proposed in this ATBD may become functionally obsolete as data collection transitions away from paper datasheets and to digital collection methods. That is because some rules for data ingest and automatic QC flag generation will likely be pre-programmed into digital devices (e.g. a pre-programmed rule specifies that only integer data are allowed to be entered in a specific field, and either forbids non-integer data from being ingested or creates an automatic quality flag if this occurs). In addition, some plausibility tests may be completed during data ingest, on digital devices, such that the plausibility tests which are referred to above as future additions to this document may also become obsolete. However, even with mobile devices for digital data collection, instances may still arise where datasheets and manual data entry will be necessary, and these tests would then still be useful in order to check data for potential transcription errors and verify that the L0 data are presented in a consistent format and are sufficient to generate higher level data products.

In addition to testing ingested data for validity and plausibility, the algorithms presented in this document provide steps for L1 variable generation that will feed directly into each referring ATBD's L1 data publication products. The variables generated include standard unique identifiers for records, samples, and individuals, spatial and temporal identifiers, and supporting data for identifying species, assigning information based on pre-defined lookup tables, etc.

1.2 Scope

This document describes the theoretical background and algorithmic process for applying validation and plausibility tests to input data. It does not provide computational implementation details, except in cases where these stem directly from algorithmic choices explained here.

These algorithms are intended to be applied to observational L0 data (i.e., raw data) to provide quality control in producing Level 1 data products. These tests will be used to examine data as they are ingested and determine the validity and/or plausibility of each observation. The tests described in this document will be referenced by algorithms in other documents.

The data product algorithms described herein can be broadly categorized as tests to check data entry validity and data value plausibility. Multiple tests may be employed to check entries within single data products. The specific tests, and the order in which they should be applied, will be specified in the referring ATBD. Where possible, efficiency will be maximized through the use of combined tests.

The outputs and auto-generated metadata resulting from this algorithm will be reported according to the referring ATBD's publication document and include (with data product fieldnames in parentheses):

Unique Record ID (uid)
Domain ID (domainID)
Site ID (siteID)
Technician ID (recordedBy, measuredBy, etc.)
Date (date)
Day of Year (dayOfYear)
Day of Water Year (dayOfWaterYear)
Latitude (decimalLatitude)
Longitude (decimalLongitude)
Latitude Uncertainty (latitudeUncertainty)
Longitude Uncertainty (longitudeUncertainty)
Elevation (elevationInMeters)
Elevation Uncertainty (elevationUncertainty)
Various quality flags

2 RELATED DOCUMENTS AND ACRONYMS

2.1 Applicable Documents

AD[01]	NEON.DOC.000001	NEON Observatory Design (NOD) Requirements
AD[02]	NEON.DOC.005003	NEON Scientific Data Products Catalog
AD[03]	NEON.DOC.005004	NEON Level 1-3 Data Products Catalog
AD[04]	NEON.DOC.005005	NEON Level 0 Data Product Catalog
AD[05]	NEON.DOC.011081	NEON Algorithm Theoretical Basis Document QA/QC Plausibility Testing
AD[06]	NEON.DOC.001408	NEON Raw Data Ingest Workbook for TOS Plant phenology observations
AD[07]	NEON.DOC.001113	Quality Flags and Quality Metrics for TIS Data Products – ATBD
AD[08]	NEON.DOC.001420	NEON Data Publication Workbook for TOS Plant phenology observations

2.2 Reference Documents

RD[01]	NEON.DOC.000008	NEON Acronym List
RD[02]	NEON.DOC.000243	NEON Glossary of Terms

2.3 Acronyms

Acronym	Explanation
SD	Standard deviation
LHDD	Location hierarchy definition document

3 DATA PRODUCT DESCRIPTION

The tests described in this document may be applied to data from throughout TOS and AOS; these data are generally direct organismal measurements made by NEON technicians in the field and lab, or samples collected by NEON technicians and analyzed by external laboratories. All tests are intended to be individual steps in an automated ingest framework through which data are delivered to the NEON data portal, passed on for additional calculation in higher level data products, or flagged for follow up and additional review.

There are two types of tests described here:

1.) **Validation Tests** – which are focused on data structure and ensuring that data submitted by technicians are complete and are reported in the expected format. Many of these tests will be automatically applied to all data streaming from TOS and AOS, others are optional and will be called as they apply.
2.) **Plausibility Tests** – which assess the likelihood that a reported value is correct. These tests may be applied to either categorical or quantitative data.

Table 1. Summary of Validation and Plausibility tests described in this ATBD.

Test Type	Test Name	Question addressed
Validation	complete record	Were all required fields populated with data?
	missing record	Was the bout completed and all data entered?
	duplicate record	Are there any records with the exact same info?
	data type	Are all entries of the correct data type for the field?
	validation rules	Are the data entries consistent with the rules provided in the data ingest workbook?
	date	Is date in the correct format and range?
	location	Does the site and/or plotID match with those in a given domain?
	technicianID	Is the reported technician ID a known NEON ID?

	lookup	Are values that should correspond to those in lookup tables valid?
	consistency	Do values not expected to change remain the same across sampling bouts?
	lab audit	Did the lab responsible for this analysis pass internal audit?
Plausibility*	order	Do data values progress in a logical order (e.g., developmental stages increase, time values advance as records are added in a given day)?
	range	Do reported values fall within known range for that variable?
	Step direction	Is there change in a logical direction from one sampling bout to the next?
	Step magnitude	Is the observed change reasonable given historical rates of change?
* Not all tests described here have algorithms described in the implementation sections of this document, those still in development are discussed in the Sec 9: Future Modifications and Plans.		

The plausibility tests and outputs are described in detail below.

3.1 Variables Reported

The data products and associated metadata to be provided to end users are determined by the referring ATBDs that call algorithms described in this document, and thus will vary among NEON Data Products. Generally, however, calls to the algorithms described below will result in quality flags and variables that are generated or assigned from ingested data values.

3.2 QA/QC Test Definitions

3.2.1 Validation Tests

These are tests that check data structure and formatting to determine whether the data are ready for data publication and/or additional data product generation algorithms. The tests described here are essential to identifying data entry transcription errors, inconsistencies in data reporting formats, and incomplete bouts

3.2.1.1 Complete Records

Some data fields are required to be filled out during every sampling or lab analysis event; these are identified by the entry value of ‘no’ in the ‘can be null’ field in the ‘MODTableSummary_in’ tab (where MOD is the three-letter module abbreviation) of the Data Ingest Workbooks (e.g., NEON Raw Data Ingest Workbook for TOS Plant phenology observations (AD[06])). The **Validation Test: Complete Records** flags entries (e.g., rows in a data table) in which no value has been entered in any field where a value is required.

3.2.1.2 Missing Record

The **Validation Test: Missing Records** checks whether sufficient sampling was completed for each bout or date and, if desired, whether the expected documentation is in place if sampling was not completed. Sufficient sampling is defined by a pre-defined number of records per plot, date, bout or year (*n*); if the expected *n* is not reached the algorithm in the referring ATBD may specify that the bout level remarks are also non-null.

3.2.1.3 Duplicate Record

Duplicate reporting of field data is one type of transcription error that would not be caught by another validation test and could negatively impact quality of higher level derived data products. The information needed to define duplicate records (e.g., date + tagID) will be specified by the referring ATBD, and duplicates identified by the **Validation Test: Duplicate Records** will be flagged for follow up.

3.2.1.4 Data Type

Data that are in an unexpected format will likely cause downstream algorithms, for the further derivation of data products, to fail, and will make L0 and L1 data downloaded by end users unusable. The **Validation Test: Data Type** compares the data type in a given cell (integer, string, etc.) to the specifications provided in the referring ATBD ingest workbook for that field.

3.2.1.5 Validation rules

The **Validation Test: Validation Rules** checks that data entered into each cell conform to validation rules, as specified in the referring ATBD ingest workbook.

3.2.1.6 Date

As detecting trends over time is a foundational goal for the NEON observatory, scrubbing incoming data for date-related errors is essential to providing usable data for derived data products and for end users. The **Validation Test: Date** checks the date format for the standardized YYYYMMDD or YYYY-MM-DD ingest format; if the date is in this or another recognizable format, the algorithm will convert it to the ISO standard date format. The properly formatted date will then be assessed for whether it falls within an acceptable date range, that is after 2013-06-01 (the beginning of field sampling at the first NEON sites) and before the date on which the test is run (the date of ingest into the NEON database).

3.2.1.7 Location

The **Validation Test: Location** goes through each row in each ingest sheet and compares the value in the plotID field (if present) or siteID field (if there is no plotID field) to values in the CI data store, to verify that all recorded plot or site ID's are valid in comparison with plot or site IDs for the domains in which the data were collected.

3.2.1.8 Technician ID

In order to assess quality metrics associated with individual technicians and to track the source of, and remedy, any errors that make it to the NEON database, it is necessary to verify that values entered for individual technicians are correct and consistent. The **Validation Test: Technician ID** is a specific example of a look up test that compares reported values for technician IDs with a controlled list of NEON field technicians. This list may be site specific or be a universal list for all field operations staff.

3.2.1.9 Lookup

For categorical data in which possible entries are constrained to a list of expected values (e.g., a species list), the **Validation Test: Lookup** may be employed to compare reported values to a controlled list in a lookup table. Entries that do not exactly match values provided in the lookup table will be flagged unless the ATBD specifies a fuzzy matching algorithm that resolves the issue.

3.2.1.10 Consistency

The **Validation Test: Consistency** checks (1) that values not expected to change over time remain the same at each sampling bout and (2) that all values related to one another are internally consistent. For example, the species, sex or location of many individuals is considered fixed; a mouse captured at time *t*, and recorded as male, will be flagged if it is reported as female during any future sampling bout (*t+n*). Moreover, a mouse recorded as male cannot also be recorded as pregnant. This test primarily (but not exclusively) identifies data transcription errors, and may no longer be necessary, in all cases, if consistency flags are built into the data entry software for a mobile device used in the field.

3.2.1.11 Lab Audit

All external laboratories with which NEON has contracted to conduct analyses on field collected samples are subject to an internal QC audit. The results of those audits will be stored in the CI data store. The **Validation Test: Lab Audit** compares the ingested labID field to the results of the internal audit, and flags records associated with labs that failed the QC audit.

3.2.2 Plausibility Tests

The categorical tests check the plausibility of nominal or qualitative data. The tests described here are essential to identifying illogical progression of developmental stages, species reported outside of known geographical range, unlikely measurements that may be the result of use of incorrect measurement units, etc.

3.2.3 Quality Flag Conventions

The quality flagging conventions used within the AOS and TOS measurement streams generally follow those described for TIS (AD[07]): 1 indicates that the quality test failed; 0 indicates that the quality test passed; and -1 indicates that insufficient data were available to run a specified test (although this last category will only be applied during plausibility tests for the purposes of this document). Data product-specific ATBDs will describe which tests are run on which variables. The naming convention for these subproducts is AbbreviationOfTestNameVariableNameQF (see examples in the NEON Data Publication

Workbook for TOS Plant phenology observations (AD[08])). The tests are abbreviated according to the conventions defined in Table 1.

Table 2. Quality flag naming conventions**.**

	Test Name	**Quality flag name**
Validation Tests	Complete record	completeRecordQF
	Missing record	missingVariableNameQF
	Duplicate record	duplicateVariableNameQF
	Data type	invalidVariableNameQF
	Validation rules	invalidVariableNameQF
	Date	variableNameQF
	Location	locationQF
	Technician ID	technicianIDVariableNameQF
	Lookup	invalidVariableNameQF
	Consistency	consistencyVariableNameQF
	Lab audit	labAuditQF

Flagged data will either be:

1) passed for release to the data portal with a flag and alphaQF report that summarizes the result of all applied quality tests
2) passed for additional QC algorithms
3) returned to the domain lab from which the initial report came for review by the technicians responsible for collecting those data points or
4) reviewed by staff scientist and accepted, corrected or rejected.

The workflow for how flagged data will be handled will be specified in the data product (referring) ATBD. Records of all flagged data, regardless of the workflow, should be detailed in the quality control flag report for later consideration and general statistics of the data flags should be output for regular scrutiny. All iterations of data will be stored and updates will be recorded as new versions of the dataset.

3.2.4 Variable generation

The variable generation algorithms provide the steps for generating and assigning variables to data publication worksheets, from ingested TOS and AOS data. In some cases, these variables will be transferred directly from L0 data ingest sheets (e.g., through the **Assign: L1 Data from L0 Data** algorithm). In other cases, new variables will be assigned/generated, including: identifiers (e.g., unique identifiers for data records), temporal and spatial data beyond those recorded by field or lab technicians (e.g., domain IDs where only plot IDs were recorded), values from lookup tables that are associated with data recorded by field or lab technicians (e.g., species names that correspond to recorded taxon IDs), and simple summary statistics for quality flags produced by the ATBD (through **Generate: Quality Flag Summary**).

3.2.4.1 Identifiers

There are three identifier variables that may be generated/assigned using algorithms designed to generate identifiers: uid, individualID, and technicianID. uid is generated by the **Generate: Unique ID** algorithm, during which a unique identifier is generated for each record ingested to the NEON database. The **Generate: Individual ID** algorithm assigns a unique value to organisms on which repeated measurements will be performed through time (e.g., individual mice or trees) at NEON sites. The **Generate: Technician IDs** algorithm generates an anonymized value for ingested technician IDs, in order to preserve the privacy of field and/or lab technicians.

3.2.4.2 Temporal data

From date values that were first ingested, and then quality checked via the algorithms described above, the variables of dayOfYear (1-365/366 from Jan. 1 to Dec. 31), and dayOfWaterYear (1-365/366, from Oct. 1 to Sept. 31) may be generated by the **Generate: Day of Year** and **Generate: Day of Water Year** algorithms, respectively. Additionally, data that include time values (e.g., startTime) may be formatted to standardized ISO Time format through the **Generate: ISO Time** algorithm.

3.2.4.3 Spatial data

The generation of new spatial data variables relies on ingested plotID and/or siteID values, which are linked to spatial data in the CI data store in order to obtain additional spatial data for reporting in publication worksheets. In this way, siteID (if not ingested) and/or domainID values may be assigned for each sampling event, by the **Assign: Location IDs** algorithm. In addition, geographic coordinates, elevation, and the uncertainty of these measurements can be generated on a sampling event-specific bases, via the **Generate: Spatial Information and Uncertainty** algorithm.

3.2.4.4 Values from lookup tables

At times, data recorded by field or lab technicians will be used to obtain additional information about the organism or sample on which a measurement is performed. The **Assign: Taxon Identification** algorithm is a specific case of this, where values ingested in the taxonID field may be compared against information in a lookup table in order to assign values to the scientificName and taxonRank fields of a data publication worksheet. The **Assign: L1 Variables from Lookup Table** algorithm performs this same function, although generically (e.g., by comparing ingested tagID values to a lookup table and assigning values to the sex field of a data publication worksheet).

3.2.4.5 Quality flag summary statistics

QA/QC processing has the potential to generate large numbers of quality flags, not all of which will be of interest to every end-user. As an alternative to publishing all flags with every dataset, NEON will provide a simple summary statistic to indicate whether errors were found in any variable within a record (e.g., row in a spreadsheet). Users may then download detailed quality reports, if desired. The summary statistic is generated through the **Generate: Quality Flag Summary** algorithm, by which the occurrence of a quality flag for *any* variable in a record results in an **alphaQF** value of 1.

4 SCIENTIFIC CONTEXT

4.1 Theory of algorithm

4.2 Algorithm Framework

When all validation/plausibility test parameters have been defined for tests required by individual, referring ATBDs, the tests are implemented in sequence for each observation at each site. Figure **1** shows an example process for this procedure.

The sequence shown here need not be followed in every case. Different observations and data products will require different sequences of tests that shall be detailed in the referring ATBD. Furthermore, in the interest of computational efficiency, it may be sufficient to not subject data that has already been flagged to more plausibility testing. That is, if a datum has failed a range test, it may not need to be subsequently subjected to a step direction or step magnitude test. All of these details can be found in the specific ATBD of interest.

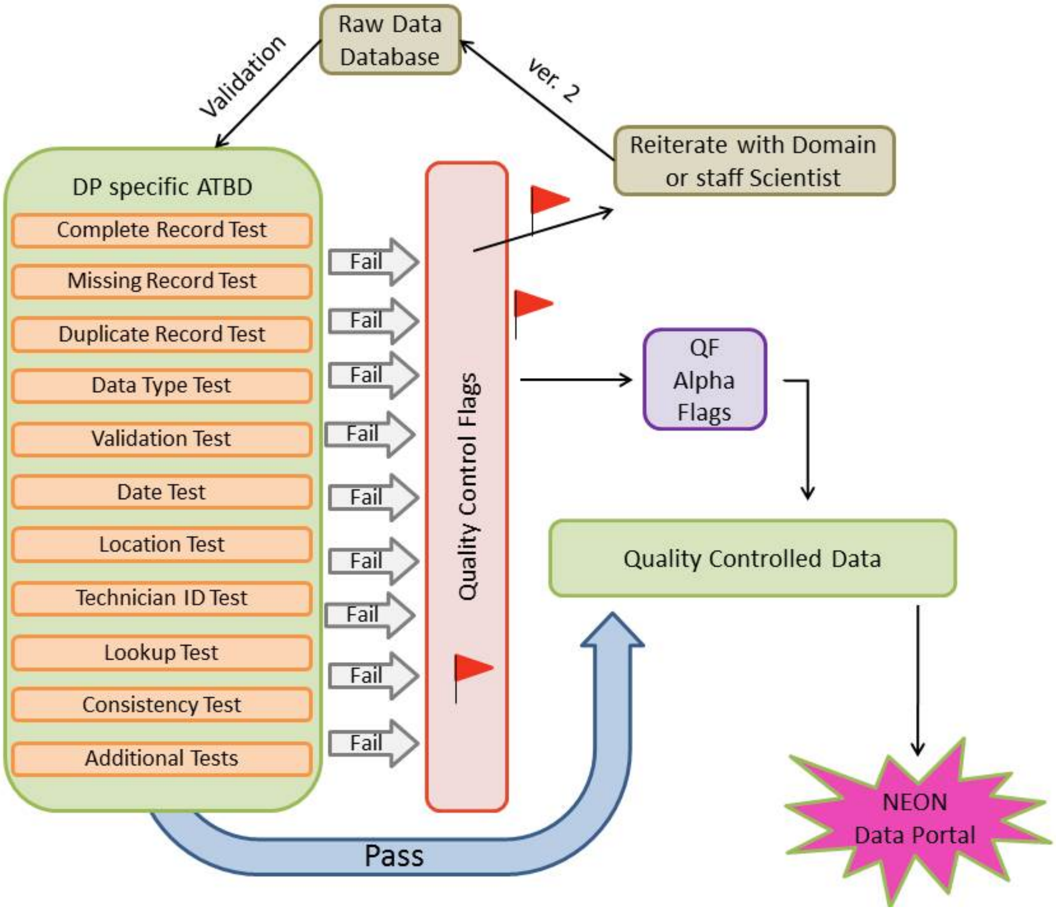

Figure 1: **Data flow diagram for automated validation testing.** The sequence of these tests will vary by data products and may call some but not necessarily all of these tests. Referring ATBDs will define the required tests, some parameters, the necessary order for plausibility tests, whether flagged data should be passed with flag or require follow up reiteration and retesting, and whether quality controlled data are used to refine test parameters.

5 ALGORITHM IMPLEMENTATION

5.1 Algorithm Processing Steps

The data processing steps are separated into two categories: 1) validity and plausibility tests that are used for the validation of L0 data, and 2) L1 variable generation processes. Many of these tests and/or processing steps apply to all NEON Field Observation and Laboratory Report products, and will be called by all other ATBDs. Other tests and/or processes apply only to select NEON Field Observations and/or Laboratory Report products, and these will be utilized by referring NEON ATBDs as needed and on a case-by-case basis.

5.1.1 General Explanation of QA/QC and Plausibility Procedure:

1. **Validation and Plausibility Tests**
   a. The first set of processing steps are QA/QC tests and plausibility assessments that apply to NEON L0 data products that were manually collected in the field or analyzed in an external laboratory. These tests verify the basic quality of the L0 data product, including that the dataset is complete, that standard information (e.g., the data collection point location) matches with known values, and that L0 data are valid, consistent, and within various user-defined expected ranges (e.g., a recorded species is expected in the location in question, a recorded tail length falls within the known range of tail lengths of a given species, etc.). Not all processing steps will be applicable to all L0 NEON data products, and so each step can be implemented in referring ATBDs on an as-needed basis. The validation and plausibility tests in this document include: a complete records test, a data type validation test, a date validation test, a duplicate records test, a location validation test, a missing records test, a validation test for data that must be compared to values in a lookup table, a coded data validation test, a test to validate data against an expected range of possible values, and a test to verify that data are consistent with one another throughout the dataset.

2. **L1 Variable Generation**
   a. The second set of processing steps are L1 data generation processes by which L0 data are used to generate additional information for NEON L1 Data Products, such as auto-generated universally unique identification numbers for every record and geographic location information for all samples/surveys.

5.1.2 Automated Processing Steps for Data Generated by NEON Field Collections

The tests and procedures described herein require a combination of: (1) pre-defined (i.e., requiring no input from referring ATBDs) data ingest workbook field names (as described in the following text), and referring ATBD-defined (2) data ingest field names, (3) lookup tables, (4) lookup table field names, (5) CI data store databases, and (6) values that comprise acceptable data entries and/or numbers of data records. Referring ATBDs shall provide the necessary information to run all applicable tests, as described below.

Processing steps for common validation QA/QC tests:

```
1. Check that all required fields are filled out (Validation Test: Complete Records)
   a. For i in 1:length(referring ATBD provided [list of data ingest sheets]):
         i. Generate completeRecordQF field in $i^{th}$ database table (from referring ATBD
            provided [list of database tables]) and populate with zeros
   b. For i in 1:length(referring ATBD provided list of data ingest sheets):
       For each row in canBeNull field of the data ingest workbook sheet ending in
       '_FieldSummary_in':
           if value == 'no':
         i. Get value in corresponding cell in fieldName field of the data ingest workbook
            sheet ending in '_FieldSummary_in'
        ii. Locate matching fieldname in $i^{th}$ data ingest sheet
         for each row in $i^{th}$ data ingest sheet:
            if value == ' ' or 'NA' or 'N/A' or 'null' or 'nodata' or 'NAN':
                A.  insert -9999 into cell
                B.  insert 1 into cell in completeRecordQF field of $i^{th}$ database table, even if
                    1 already exists here

2. Check that all expected records are present (Validation Test: Missing Records)
   a.  Generate QF field= paste(missing + referring ATBD provided [QF Name] + QF) in
       referring ATBD provided [database table] and populate with zeros
         i. Example: missingRecordsPerBoutQF
   b.  For each row in the referring ATBD provided [data ingest sheet]:
         i. Concatenate values from every fieldname in the referring ATBD provided [list of
            'Fieldnames']
        ii. add string to cell in Temp field of data ingest sheet
   c.  For each row in Temp field of data ingest sheet:
        if value occurs < N times,  where N is a referring ATBD provided [# records for
        complete set]:
            i. insert 1 into cells in generated QF field of database table (e.g.
               missingRecordsPerBoutQF)
   d.  delete Temp field of data ingest sheet

3. Check for duplicate records (Validation Test: Duplicate Records)
   a. Generate QF field= paste(duplicate + referring ATBD provided [QF Name] + QF) in
      referring ATBD provided [database table] and populate with zeros
        i. Example: duplicateTrapCoordinateQF
   b. For each row in the referring ATBD provided [data ingest sheet]:
        i. Concatenate values from every fieldname in the referring ATBD provided [list of
           'Fieldnames']
       ii. add string to cell in Temp field of data ingest sheet
```

```
    c. For each row in Temp field of the data ingest sheet:
          if value is not unique:
             i. Insert 1 into cell in generated QF field of database table
    d. delete Temp field in data ingest sheet

4. Check that all records are of the correct data type (Validation Test: Data Type)
    a. For i in 1:length(referring ATBD provided [list of data ingest sheets]):
            i. For each fieldname in ith data ingest sheet except plotID, siteID,
               samplingProtocol, and any containing 'date' OR 'Date':
                  A. Fname= paste (invalid + fieldname + QF)
                     aa. Example: invalidTailLengthQF
                  B. If the ith database table (from referring ATBD provided [list of database
                     tables]) contains no fieldname that matches Fname:
                     aa. Generate QF field= Fname in ith database table and populate with
                     zeros
                  C. Locate fieldname in fieldName field of the data ingest workbook sheet
                     ending in '_FieldSummary_in'
                  D. locate corresponding value in Data Type field of the data ingest
                     workbook sheet ending in '_FieldSummary_in'
               for each row in the ith data ingest sheet:
                     if data type of value in cell does not match value from a.i.D:
                              aa.  Insert -9999 into cell
                              bb. Insert 1 into corresponding cell in Fname (from step a.i.A)
                              field of ith database table

5. Check that entries in data ingest workbook pass validation rules in referring ATBD provided
   sheet of data ingest workbook (Validation Test: Validation Rules)
       a.  For i in 1:length(referring ATBD provided [list of data ingest sheets]):
            i. For each fieldname in ith data ingest sheet except plotID, siteID,
               samplingProtocol, or those in the referring ATBD provided [list of date
               fieldnames]:
                  A. Fname= paste (invalid + fieldname + QF)
                     aa. Example: invalidTailLengthQF
                  B. If the ith database table (from referring ATBD provided [list of database
                     tables]) contains no fieldname that matches Fname:
                     aa. Generate QF field= Fname in ith database table and populate with
                     zeros
                  C. locate fieldname in fieldName field of the data ingest workbook sheet
                     ending in '_FieldSummary_in'
                  D. locate corresponding value in ingestValidationRules field of the data
                     ingest workbook sheet ending in '_FieldSummary_in'
               If value in a.i.D is not 'lookup table', 'NA', or does not start with 'conditional':
                      for each row in the ith data ingest sheet:
                            aa. If value does not comply with rules specified in a.i.D:
                                    AA. Insert 1 into corresponding cell in Fname (from step
                                        a.i.A) field of ith database table
```

```
6. Check for validity of date (Validation Test: Date)
   a. For i in 1:length(referring ATBD provided [list of data ingest sheets]):
         For j in 1: length(referring ATBD provided [list of date fieldnames]):
            If jth fieldname is in ith data ingest sheet:
                  i. Generate QF Field= paste(jth date fieldname + QF) in ith database table (from referring ATBD provided [list of database tables]) and populate with zeros
                        A. Example: startDateQF
   b. For each row in the ith data ingest sheet:
         For j in 1: length(referring ATBD provided list of date fieldnames):
            if jth date fieldname is in ith data ingest sheet and value is of format YYYYMMDD:
                 i. Convert value to ISO standard format: YYYY-MM-DD and insert into the jth date fieldname of the ith database table. Do not change entry in ingest sheet
            Else if jth date fieldname is in ith data ingest sheet and value is of format YYYY-MM-DD (dashes in the 5th and 8th positions):
                 i. Insert value into the jth date fieldname of the ith database table
            Else if jth date fieldname is in ith data ingest sheet and value is not of format YYYYMMDD:
                 if value is in format: MM/DD/YYYY or MM/D/YYYY or M/D/YYYY or M/DD/YYYY or MM/DD/YY or MM/D/YY or M/D/YY or M/DD/YY or MM-DD-YYYY or MM-D-YYYY or M-DD-YYYY or M-D-YYYY or MM-DD-YY or MM-D-YY or M-DD-YY or M-D-YY:
                      i. Insert 1 into cell in correct generated QF field of ith database table (e.g. startDateQF)
                     ii. Insert into cell in remarks field of ith database table: “Date was of format ___” where ___ is the format in which the value in the jth date fieldname field is recorded
                    iii. Convert value to YYYY-MM-DD and insert into the jth date fieldname of the ith database table. Do not change entry in ingest sheet
                 Else if jth date fieldname is in ith data ingest sheet and value cannot be converted to format YYYY-MM-DD:
                      i. Insert 1 into cell in correct generated QF field of ith database table (e.g. startDateQF)
                     ii. Insert into cell in remarks field of ith database table: “Date is of unknown format”
                    iii. Insert -9999 into the jth date fieldname of the ith database table
   c. For i in 1:length(referring ATBD provided list of database tables):
         For j in 1: length(referring ATBD provided list of date fieldnames):
            If jth date fieldname is in ith database table and value is not of format YYYY-MM-DD:
                 i. skip
            Else if jth date fieldname is in ith data ingest sheet and value < 2013-06-01 or > data ingest sheet processing date:
```

```
        i. Insert 1 into cell in correct generated QF field of $i^{th}$ database table (e.g.
           startDateQF)
        ii. Insert into cell in remarks field of $i^{th}$ database table: “Date is out of range”
```

7. Check that values in the **plotID** or **siteID** fields match location information in referring ATBD provided location hierarchy document (**Validation Test: Location**)

```
a. For i in 1:length(referring ATBD provided [list of data ingest sheets]):
      i. Generate locationQF field in $i^{th}$ database table (from referring ATBD provided [list
         of database tables]) and populate with zeros
b. For i in 1:length(referring ATBD provided list of data ingest sheets):
      For fieldnames in $i^{th}$ data ingest sheet:
         if plotID is in fieldnames:
            for each row in plotID field of $i^{th}$ data ingest sheet:
               if value is not in CI data store of plots for the domain and site:
                  A. insert -9999 into cell in plotID field of $i^{th}$ database table
                  B. insert 1 into cell in locationQF field of $i^{th}$ database table
         Else if siteID is in fieldnames but plotID is not:
            for each row in siteID field of $i^{th}$ data ingest sheet:
               if value is not in CI data store of sites for the domain:
                  A. insert -9999 into cell in siteID field of $i^{th}$ database table
                  B. insert 1 into cell in locationQF field of $i^{th}$ database table
```

8. Check for validity of technician IDs, against CI-stored lookup table in which employee IDs for technicians working for NEON are recorded (**Validate: Technician IDs**)

```
a. For i in 1:length(referring ATBD provided [list of data ingest sheets]):
      For j in 1: length(referring ATBD provided [list of technician fieldnames]):
         If $j^{th}$ technician fieldname is in $i^{th}$ data ingest sheet:
            i. Generate QF Field= paste(technicianID + $j^{th}$ technician fieldname + QF) in
               $i^{th}$ database table (from referring ATBD provided [list of database tables])
               and populate with zeros
                  A. Example: technicianIDRecordedByQF
b. For each row in the $i^{th}$ data ingest sheet:
      For j in 1: length(referring ATBD provided list of technician fieldnames):
         if $j^{th}$ technician fieldname is in $i^{th}$ data ingest sheet and value is not in CI data store
         of technician IDs:
            i. Insert 1 into cell in correct generated QF field of $i^{th}$ database table (e.g.
               technicianIDRecordedByQF)
```

9. Check that entries in referring ATBD provided field match expected values based on referring ATBD provided lookup table (**Validation Test: Lookup**)

```
a. For i in 1:length(referring ATBD provided [list of data ingest fieldnames]):
      i. Fname= paste (invalid  + fieldname + QF)
            aa. Example: invalidSamplingMethodQF
      ii. If the referring ATBD provided [database table] contains no fieldname that
          matches Fname:
         A. Generate QF field= Fname  in database table  and populate with zeros
```

iii. For each row in the i^{th} data ingest fieldname of the referring ATBD provided [data ingest sheet]:

if value is not in the i^{th} lookup fieldname (from a referring ATBD provided [list of lookup fieldnames]) of the i^{th} lookup table (from a referring ATBD provided [list of lookup tables]):

A. Insert 1 into corresponding cell in Fname (from step a.i) field of database table

10. Verify that values in referring ATBD provided sets of fields always correspond with one another (**Validation Test: Consistency**)
    a. Generate QF field= paste(**consistency** + **referring ATBD provided [QF Name] + QF)** in referring ATBD provided [database table] and populate with zeros
        i. Example: **consistencyTagIDSexQF**
    b. For each individual value in the first fieldname, from the referring ATBD provided [list of fieldnames], in the referring ATBD provided [data ingest sheet]:
        i. Generate a list of concatenated values from all fieldnames in the referring ATBD provided list of fieldnames
        ii. If the size of the list is > 1 AND the elements in the list are not identical:
            A. Insert 1 into cell in generated QF field of database table for all records used to generate the list of concatenated values
11. Verify that an external lab facility complies with NEON quality control standards, as specified by 'pass/fail' results of internal audits (**Validation Test: Lab Audit)**
    a. Generate QF field= **labAuditQF** in referring ATBD provided [database table] and populate with zeros
    b. For each row in referring ATBD provided [lab ID fieldname] of referring ATBD provided [data ingest sheet]:
        i. Locate value in CI data store of external testing facilities
        ii. If corresponding lab audit results indicate that the facility failed audit
            A. Insert 1 into corresponding cell in **labAuditQF** field of database table

Processing steps for L1 variable generation:

1. Assign values from data ingest workbook to data publication workbook (**Assign: L1 Data from L0 Data**)
    a. For i in 1:length(referring ATBD provided [list of data ingest sheets]):
       For each **fieldname** in i^{th} data ingest sheet *except* those in the referring ATBD provided [list of non-transferring fieldnames]:
        i. Generate **fieldname** field in i^{th} database table (from referring ATBD provided [list of database tables])
        ii. Populate **fieldname** field in i^{th} database table with data from **fieldname** field of i^{th} data ingest sheet

2. Generate unique identifier (**Generate: Unique ID**)
    a. For i in 1:length(referring ATBD provided [list of database tables]):
       For each row in i^{th} database table:
        i. Auto generate Number

        ii. insert Number into cell in **uid** field of i^{th} database table

3. Generate **individualID,** a NEON unique identifier across space and time (**Generate: Individual ID**)
    - a. CI shall maintain a lookup table (hereafter lookup:NEONMODID, where 'MOD' is the 3-letter module abbreviation for the module in question) in which unique sets of **domainID** and **tagID** values are linked for each domain, through a 6-digit CI-autogenerated **individualID** value of the form NEON.MOD.DXX.123456, where:
        - i. MOD= 3-character module abbreviation
        - ii. DXX = domain number
        - iii. 123456= auto-increment number for each unique **domainID/tagID** combination
    - b. For each row in [database table]:
        - i. Concatenate values in **domainID** and **tagID** fields

          If value from b.1 is in **individualIDs** field of lookup:NEONMODID:
            - A. Insert value from **individualID** field of lookup: NEONMODID into **individualID** field of database table

          Else:
            - A. Auto-increment **individualID** value as specified in step a
            - B. Insert auto-incremented number into **individualID** field of database table

4. Generate technician IDs for reporting in database tables (**Generate: Technician IDs**)
    - a. For i in 1:length(referring ATBD provided [list of database tables]):

      For j in 1: length(referring ATBD provided [list of technician fieldnames]):

      if j^{th} **technician fieldname** is in i^{th} database table and value is in CI data store of Employee IDs for each site:
        - i. Locate corresponding 'NEONID' value in CI data store
        - ii. Insert 'NEONID value in corresponding cell in j^{th} **technician fieldname** (e.g. **recordedBy)** in i^{th} database table

      Else:
        - i. Insert -9999 into corresponding cell in j^{th} **technician fieldname** (e.g. **recordedBy)** in i^{th} database table

5. Generate Day of Year (**Generate: Day of Year**)
    - a. For each row in **date** field of [database table]:

      if value is not null and is in format YYYYMMDD or YYYY-MM-DD:
        - i. Convert YYYYMMDD or YYYY-MM-DD to **dayOfYear** (from 1 to 365, starting on January 1, with leap years every four years starting from 2012. Tables for conversion at: http://disc.gsfc.nasa.gov/julian_calendar.shtml)
        - ii. Insert value into cell in **dayOfYear** field of database table

      Else:
        - i. Insert -9999 into cell in corresponding row in **dayOfYear** field of database table

6. Generate Water Year (**Generate: Day of Water Year**)
    - a. For each row in **date** field of [database table]:

```
    if value is not null and is in format YYYYMMDD or YYYY-MM-DD:
        i. Convert YYYYMMDD to dayOfWaterYear (from 1 to 365, starting on October 1
           and extending through September 30 of the following year, with leap years every
           four years starting from 2012)
        ii. Insert value into cell in dayOfWaterYear field of database table
    Else:
        i. Insert -9999 into cell in corresponding row in dayOfWaterYear field of database
           table

7. Generate ISO standard time (Generate: ISO Time)
   a. For i in 1:length(referring ATBD provided [list of database tables]):
          For j in 1: length(referring ATBD provided [list of time fieldnames]):
              If jth time fieldname is in ith database table:
                  For each row in the ith database table:
                      If value is of format HH:MM or HHMM:
                          i. Get value from plotID field
                          ii. Locate plotID value in CI data store of plots for each site in the
                              domains
                          iii. Get corresponding timeZone value from CI data store of plots for each
                               site in the domains
                          iv. Use time zone to get UTC offset (e.g. at
                              http://en.wikipedia.org/wiki/List_of_UTC_time_offsets)
                          v. Convert value in jth time fieldname (e.g. startTime) in ith referring
                             ATBD provided data ingest sheet, to ISO standard time format:
                             paste(HH,':',MM, UTC offset from step a.iv)
                                 A. Example: 09:30-07:00
                          vi. Get value in date  field of ith  database table
                          vii. Paste value from date field + string from step a.v into cell in jth time
                               fieldname of the ith database table. Do not change any values in ith
                               data ingest sheet
                                 A. Example: 2014-02-14T09:30-07:00
                      Else:
                          i. Insert -9999 into jth time fieldname of the ith database table

8. Assign domain and/or site identifications (Assign: Location IDs)
   a. For i in 1:length(referring ATBD provided [list of database tables]):
          For fieldnames in ith database table:
              if plotID is in fieldnames:
                  for each row in plotID field of ith database table:
                      i. Locate value in CI data store of plots for sites in the domain
                      ii. Insert domain, from CI data store, into cell in domainID field of ith
                          database table
                      iii. Insert site, from CI data store, into cell in siteID field of ith database
                           table
              Else if siteID is in fieldnames but plotID is not:
                  for each row in siteID field of ith database table:
```

```
                iv. Locate value in CI data store of sites for the domain
                 v. Insert domain, from CI data store, into cell in domainID field of ith database table
            Else:
                vi. Insert -9999 into cells in domainID and siteID fields of ith database table

9. Generate spatial location and uncertainty information (Generate: Spatial Information and Uncertainty)
   a. If referring ATBD provided [spatial table] contains a pointID field:
        i. For each row in referring ATBD provided [database table]:
              A. If value in referring ATBD provided [fieldname] field == value in pointID field of spatial table AND value in plotID field == value in plotID field of spatial table AND referring ATBD provided [subtype] == value in subtype field of spatial table:
                    a. Insert values from pointIDDecimalLatitude, pointIDDecimalLongitude, pointIDElevation, pointIDCoordinateUncertainty, pointIDElevationUncertainty  fields of spatial table into decimalLatitude, decimalLongitude, elevation, coordinateUncertainty, elevationUncertainty  fields of database table
              B. Else:
                    a. insert -9999 into decimalLatitude, decimalLongitude, elevation,  coordinateUncertainty, elevationUncertainty fields of database table
   b. Else:
        ii. For each row in database table:
              A. If value in referring ATBD provided [fieldname] field == value in plotID field of spatial table AND referring ATBD provided [subtype] == value in subtype field of spatial table:
                        a. Insert values from decimalLatitude, decimalLongitude, elevation, coordinateUncertainty, elevationUncertainty  fields of spatial table into fields of same name in  database table
              B. Else:
                        a. Insert -9999 into decimalLatitude, decimalLongitude, elevation, coordinateUncertainty, elevationUncertainty  fields of database table

10. Assign L1 scientificName and taxonRank values (Assign: Taxonomic Identifications)
   a. For each row in lookupTaxonIDQF field of referring ATBD provided [database table]:
      If value==0:
          i. Get value in corresponding cell in taxonID field
         For taxonID field of referring ATBD provided [lookup table]:
             If value matches value from step a.i:
                    A. Get value in corresponding scientificName field of lookup table
                    B. Insert value into cell in correct row in scientificName field of
                       database table
                    C. Get value in cell in corresponding taxonRank field of lookup
                       table
                    D. Insert value into cell in correct row in taxonRank field of
                       database table
            Else:
              i. Insert -9999 into cells in scientificName and taxonRank fields of database
                 table

11. Assign L1 variables from lookup table (Assign: L1 Variables from Lookup Table)
    a. For i in 1:length(referring ATBD provided [list of fieldnames]):
         For each row in the ith fieldname of the referring ATBD provided [database table]:
           i. If  value is in ith 'Lookup' fieldname (from referring ATBD provided [list of
              'Lookup' fieldnames]) of ith lookup table (from referring ATBD provided [list
              of lookup tables]):
                A. Locate corresponding value in ith 'Get values' fieldname (from
                   referring ATBD provided [list of 'Get values' fieldnames])
                B. Insert value from ith 'Get values' fieldname into cell in ith 'L1
                   fieldnames' fieldname (from referring ATBD provided [list of 'L1
                   fieldnames']) of the database table
           ii. Else:
                A. Insert -9999 into corresponding cell in ith 'L1 fieldnames' fieldname
                   of the database table

12. Calculate quality flag summary for publication with L1 data products (Generate: Quality Flag
    Summary)
    a.  For i in 1:length(referring ATBD provided [list of database tables]):
           i.  Generate QF field= alphaQF in ith database table and populate with zeros
           ii. For each row in ith database table:
                 A. QFsum= sum(values in all fields where fieldname includes 'QF')
                 B. If QFsum > 0:
                      aa. Insert 1 into corresponding cell in alphaQF field of ith database
                          table
```

6 UNCERTAINTY

6.1 Analysis of Uncertainty

These data validation and plausibility tests are intended as an automated process for flagging unlikely measurement values or incorrectly entered data. The number of flags generated may contribute to uncertainty estimates for reporting of observed data but the tests described in this document do not have an uncertainty estimate in and of themselves.

6.2 Reported Uncertainty

Upon completion of all plausibility testing for a given set of observations at a given location, and if specified in the Data Product specific ATBD, all of the data and associated quality information shall be made available for the next sequence of automated quality control testing or calculation algorithm in the case of derived Level 1 data products.

7 VALIDATION AND VERIFICATION

7.1 Algorithm Validation

7.2 Data Product Validation

7.3 Data Product Verification

8 SCIENTIFIC AND EDUCATIONAL APPLICATIONS

Before any of the data collected by NEON can be served to the community or integrated into higher level and derived data products the raw observations must first be pass basic quality control tests. This document provides the framework for automating the first pass quality check on TOS and AOS organismal data and suggests a system for refining the process by which data are assessed as the observatory matures.

9 FUTURE MODIFICATIONS AND PLANS

Quantitative tests for checking data plausibility will be an essential step in QC of reported field data and are the next anticipated step for this ATBD. Here are some plausibility tests that may be applied in the future, though these tests are not ready to be implemented.

9.1 QA/QC Test Definitions

9.1.1 Quantitative Tests

The quantitative tests evaluate the plausibility of reported numeric values. Tests described here (see Table 4) compare either (1) data values themselves (observation range test), or (2) changes in data values over time (observation order, step, and magnitude tests), to threshold minimum, maximum, and/or expected values. These thresholds may be determined either dynamically, based on NEON data, or using predefined values provided by Science to CI via the CI data store. During the first years of observatory sampling, predefined values may be used exclusively, given the absence of large volumes of data. Following the acquisition of sufficient data (see section 9.2.1 for further detail), threshold values will be defined dynamically based on observed minima, maxima, averages, etc.

Table 3.. Quality flag naming conventions for quantitative tests.

Test Group	Test Name	Quality flag name
Plausibility	Order test	orderVariableNameQF
	Observation Range Test	RangeVariableNameQF
	Step direction test	StepDirectionVariableNameQF

	Step magnitude test	StepMagnitudeVariableNameQF

The quantitative tests will be designed to test the following questions:

- Is a recorded value reasonable? (observation range test)
- Are sequential records progressing in the expected sequence order (observation order test)
- Is there change in a logical direction from one sampling bout to the next? (step direction test)
- Is the observed change reasonable given historical rates of change? (step magnitude test)

9.1.1.1 Observation Order Test

The **Plausibility Test: Observation Order** applies to data that are sequential, such as those describing developmental stages; mammals progress from juveniles to adults, emerging leaf phenophases only occur before leaves are full sized or senescent. When this test is referenced in an ATBD, the expected order will be provided. Values provided at time *t* will be compared to *t-1* or to all *t-n* previous observations of a particular individual, and all *t-n* observations that violate the expected order will be flagged.

9.1.1.2 Observation Range Test

The **Plausibility Test: Observation Range** checks that every numerical observation falls within a reasonable range of values for a given observation resolution (e.g., species, growth form, decay class, soil horizon), location, and/or time of year. Ideally, the min/max values used to determine the range limits are determined from existing data sources. For example, if the range of diameter at breast height (DBH) values for Red Maple (*Acer rubrum*) is generally 20 to 80 cm, and a reported value of 500 cm is encountered, the observation range test would flag this as implausible (i.e., out of range).

9.1.1.3 Step Tests

Step tests are designed to ensure that changes in data collected through time are realistic over a given period. They check the plausibility of data based on temporal variation, and are concerned with maximum fluctuations in data sets. A step test evaluates successive data points and identifies values that do not conform to expected variation. The **Plausibility Test: Step Direction** identifies unexpected increases or decreases through time, and is most often applied to organismal measurements that are expected to increase over the life of an individual (e.g., the length of a mouse's tail increases from juvenile to adult). The **Plausibility Test: Step Magnitude** compares successive data points to determine if their difference exceeds a maximum threshold in any direction. For infrequent or inconsistently sampled variables, the difference between subsequent measurements may be divided by the period of time between measurements in order to calculate step magnitude for a biologically relevant growth period. For example, the inter-sampling interval for captured mammals may range from 1 night (in the case of an individual trapped on consecutive nights) to several months (for an individual that is captured only rarely). A step test on size increment should therefore be applied to Δsize/Δtime, rather than simply on Δsize.

10 BIBLIOGRAPHY

Dunning, J.B.J. (2008) CRC handbook of avian body masses. CRC Press, Boca Raton, FL.

Taylor LR (1961) Aggregation, variance and the mean. *Nature* 189, 732–735.

Thibault KM, White EP, Hurlbert AH, Ernest MSK. 2011. Multimodality in the individual size distributions of bird communities. Global Ecology and Biogeography. 20(1):145-153.

11 CHANGELOG