A Numerical Data Standard Joining Units, Numerical Accuracy and Full Metadata with Numerical Values

Joseph E. Johnson

Department of Physics, University of South Carolina, Columbia SC, 29208, U.S.A.

Abstract. Numerical measurements have little meaning without the associated units of measurement, level of accuracy, and defining metadata tags. Yet these three associated items are scattered in rows, columns, references, the title, and other positions in data tables separated from the values themselves and usually immersed in unstructured text. Thus unambiguous electronic readability is not possible but requires human preformatting to link these three data components to the value for computer processing. This paper proposes a tight linkage among numerical values, units, accuracy, and meaning as a string expression that provides a numerical standard which we call a "metanumber" along with a set of requirements for this structure. We require that this metanumber string object be instantly readable by both humans and computers. Our work then develops the requisite algorithms for automated computer processing of the metanumber expressions resulting in a new metanumber where (a) all units and dimensional analysis are automatically processed, (b) numerical accuracy of results are computed, and (c) unlimited metadata tags and structures provide a trace of all historical operations with component meaning providing both the exact evolutionary path of each computed number, and a unique name (as an internet path) for every single measured v4alue. We also propose a flexible table-like structure for archiving all numerical data as metanumbers that allows the automated sharing of data among users. This structure can serve as a foundation for high –speed data exchange removing the costs, errors, and time delays required with human preprocessing and thus serving as a critical component in Big Data processing and as a foundation for advanced artificial intelligence. Other features are explored including an optional proposed expansion of the base units for the SI (metric) system as well as user defined units that are natural for users with commercial, industrial, medical, and scientific problems. These components offer a transformational increase in computational power in every domain of inquiry.

Keywords. Units, Dimensional Analysis, Metadata, Numerical Uncertainty & Accuracy, Metanumber, Error Management, Data Tags, Numerical Information Standards, Si & Metric Units, Big Data.

1 Introduction

Numeric data by itself, except for pure numbers, is meaningless without three associated components: a. units of measurement, b. accuracy level (numerical uncertainty), and c. the associated defining and modifying information. All these are requisite for the unambiguous meaning of numerical values. Yet in every domain, numerical data tables have these three components scattered in the title, row &

Johnson J.
A Numerical Data Standard Joining Units, Numerical Accuracy and Full Metadata with Numerical Values.
DOI: 10.5220/0006156400140030
In *European Project Space on Computational Intelligence, Knowledge Discovery and Systems Engineering for Health and Sports (EPS Rome 2014)*, pages 14-30
ISBN: 978-989-758-154-0

column headings, and footnotes. Sometimes such auxiliary data is just assumed to be apparent. There is no universal standard for the attachment of such metadata to numeric values or even a unique computer readable symbol structure for these components. The resulting arbitrary positions, fonts, and formatting styles prevent unambiguous automated electronic reading and requires that all data must be manually converted to a useful form by each user prior to their computer processing each and every time it is used thus incurring substantial costs, time delays, and associated errors. The reason we believe that no standard has emerged is that no single convention seems preferable or natural. Also storage space, in both print and electronic representation, has historically always been a primary consideration, so the compressed formats that are currently used were historically optimal for printing, direct human viewing, and to minimize electronic storage. A substantial component of this problem is that the units, accuracy, and defining tags are not only separated from the values, they are also embedded in text thus making it extremely difficult to extract their content by computer algorithms. Those space considerations are no longer an issue with current technology. Having a numerical metadata standard for automatic electronic data processing would not only remove all of these problems and lead to fully automated sharing and processing of all numerical data, but could lay the foundation for far more advanced and intelligent automated analysis and decision systems, that will not require human preprocessing or intervention. The standardization can be implemented automatically from the initial observation or measurement device when the data is originally recorded. Such a solution is essential for the emergence of a new level of artificial intelligence. When any numerical value is retrieved, using a unique name, it arrives with units, uncertainly level, and complete metadata so that logical and numerical processing is executed not only for dimensional and error analysis but also with advanced metadata tag management and tracing of the historical evolution of numbers. So there are two problems that have to be simultaneously addressed: (a) create a standard for such integrated metanumber formats so that they are easily read by both humans and computers, and (b) create the algorithms for mathematical and logistic processing that evaluates all expressions among such objects.

2 Proposed Solution

The author here proposes standards for attaching the units, error, and all other descriptive metadata, to numerical values along with the associated three software algorithms that mathematically processes such extended information [1]; [2], [3]; [4]. We call such objects “metanumbers” to denote their informational completeness. Our integrated algorithm satisfies our criteria that these metanumbers must (A) be easily readable both by humans and by computers, (B) require the minimum of storage, (C) process at the fastest possible speed, (D) be easily extensible to allow for user defined units, and (E) support computation at all levels from simple calculations as web “apps” on internet devices to unlimited “Big Data” applications in large scale computing. It must also (F) automatically manage all computations among metanumbers with dimensional analysis to trap invalid unit combinations as well as (G) compute the associated accuracy (numerical uncertainty) of the result using error

analysis with user selectable means of combining error. It must (H) provide a simple rapid means for the one-time initial formatting of data into the requisite metanumber format, and (I) allow one to easily retrieve and use all past results. Finally, (J) the algorithms must allow unlimited metadata to be linked to numerical values and still manage that attachment with the absolute minimum number of characters and without any reduction in the processing speed. The author has developed a methodology for a simple one-time reformatting of numerical data sets into such a standardized metanumber form that, with the algorithms we have developed, will satisfy each and all of the listed requirements and still provide unlimited traceability of all metadata linkages and origins.

3 Historical Development & Current Status

Early versions of the author's unit and dimensional management software were built and deployed on a web site that has been in use for over a decade and is still used by students in several university courses but it was limited. Recent developments by the author led to (a) a new optimized design methodology that vastly extends numerical operations with units and dimensionality analysis including (b) the reuse of any previous expression, (c) improved means for output of an expression in any desired units by using a resolution of the identity, (d) an improved means for users to define personal units for their specialized domain of work, (e) a new method of error analysis with an awarded U.S. patent to the author, and (f) a new innovative design for optimally managing unlimited metadata attachments to numerical values. This new algorithm can also support a ten-fold increase in processing speed using a parallelization of the code over multiple processors for Big Data problems. Then a future stage will support a dedicated accelerator chip for ultra-fast processing. The author has completed the full design, programming, and testing of both the units and metadata management components along with several important metanumber tables in a fully operational system. This software and methodology is now fully tested and operational on our central server as a cloud application available from any internet device. The central server approach is necessary in order to support multiple users, rapidly deploy code upgrades, and, most important, to share a rapidly growing interdisciplinary standardized library. Such a cloud system allows use by any internet linked device, (tablets/iPads, smart phones, PCs) as well as large central systems. The software has been developed in the Python language which is open source with extensive additional components, and is a widely accepted very modern framework for rapid development. The addition of a Python algorithm for the processing of numerical accuracy has also been fully integrated using the "Python Uncertainty" system developed by Leibigot and associates. The final phase of building user interface tools is currently active. We will discuss each of these three active components separately.

4 The Units (Dimensional Analysis) Algorithm

4.1 Units Introduction

There are a large number of 'unit conversion programs' and some that can process expressions containing multiple units. Our algorithm however is optimal with both maximum speed and minimum space whereby one simply attaches unit names as variables within any valid algebraic expression such as (4.3*ft/s+7*meter/hour). The 'value' of each unit and constant provides the instantaneous requisite conversion with effortless readability for humans. With this algorithm, each unit name or constant is a unique variable defined in terms of the foundational metric (SI) system units. The units algorithm contains most normally encountered units (over 800 units and essential fundamental constants). It also supports the introduction of other units as defined by users in terms of existing units and constants (such as the lightyear). These results have led us to change a number of traditional notations.

4.2 Units Standards

In order to be compatible with modern computer languages and optimal electronic processing without ambiguity, we define all units in lower case with standard alphanumeric characters, without Greek characters, without superscripts or subscripts, and without any other special characters, symbols, or fonts. The past denotations for unit names and constants lead to ambiguities in different computer languages and thus we seek unit and constant names that are expressible in simple lower case alphanumeric characters. Specifically we remove the capitalization of all proper names assigned to a unit and thus we write "newton" or "nt" rather than "Newton" or "Nt" because the value of a variable in modern computer languages is case sensitive for alphabetical symbols. Likewise the potentially ambiguous encoding of units and constants with multiple fonts, superscripting and subscripting, and of foreign alphabets is removed. Thus we use "ohm" rather than Ω; and hb (h-bar) for Planks constant divided by 2π. Pi is likewise written as pi rather than as π. Our motives are format consistency, adherence to modern software convention (variables are lower case), the use of only standard alphanumeric characters for variables, and to remove the ambiguity that can result from font and format imbedded information. Although a few of the most common units are allowed in the plural form, we generally restrict the spelling to only the singular form thus removing some ambiguities in spelling and simultaneously making the underlying code shorter. Our core units algorithm is essentially the same in any language (C/C++, Java, Ruby, Python…) and can be easily enhanced by parallel processing on a dedicated accelerator chip [5]. Our algorithm catches all exceptions for dimensional errors, and returns expressions in standard format for use in subsequent expressions. These in turn, can be referenced with a unique name for insertion in any future expressions and are functionally valid although they are expressed in different units. All computational history is maintained for each user.

4.3 Units Features

Mathematical expressions of values are returned in metric unless other units are specified using the format “(expression) ! (units desired)” . The speed of light in miles per minute would be written as c!(mile/min). Formally the algorithm multiplies the expression by (desired_units / desired_units) which is technically unity and does not change the value. The software divides the value of the expression (in default units) by the desired_units and then writes the final expression as “final_expression * desired_units”. The effect is to both divide by the desired units then multiply by the desired units. So although the value is numerically different when expressed in different units, it is computationally equivalent to the original value since it is now multiplied by different output units and either expression will evaluate equivalently. For example 35*acre*!(ft2) would give the value of 35 acres in square feet and then multiply that value by square feet (ft2).

4.4 Units Base Si System

The units algorithm is defined and based upon the SI (metric) system in terms of meter, kilogram, second, ampere, kelvin, and candela. All the other primary units and core constants are then defined in terms of these allowing also for altered spelling and abbreviations such as m for meter, sec and s for second, etc. Dimensionless prefixes (kilo, mega, centi, million, dozen, etc. can be mixed with units in any mathematically valid manner thus eliminating the need for joined definitions. The 800 or so primary units and physical constants are then each defined in terms of previous units, sequentially back to the fundamental metric units. Values assume the accuracy level of the computational environment while the accuracy is indicated as the median value with an associated accuracy (uncertainty).

4.5 Powers of Base Units

In the original definitions we also define m2= m*m; and m3=m*m2 up to the fourth power in order to assign basic unit variables that have these multiple powers in an abbreviated manner. We likewise define the inverse powers of the base SI units as m_2 = 1/m2, m_3 = 1/m3 also up to the inverse fourth power. If any mathematical expression is then well formed using unit names then the result will be returned in SI units by default. Thus (34.6*acre*(1.2*ft+7.2*inch-3.4*cm)) will give the computed volume (numerically) in cubic meters (m3). In order to obtain that volume in gallons, one would instead enter: (34.6*acre*(1.2*ft+7.2*inch-3.4*cm)) ! gallon.

4.6 Standardized Metanumber Data Tables

The core fundamental constants and primary units are defined within the metanumber program. All other numeric data, user defined units, and special constants are defined in tables of various dimensionalities using the format [server-path_ directory-path_table-name_row-name_column-name…)]. For example the elements table (which has the unique abbreviated name of “e”), contains all elements in rows, with

columns providing over 50 associated properties. This path name (currently using our default server) of [file_row_column] can itself be used as a variable to retrieve that metanumber value from the stored data tables. Thus [E_Gold_Density] / [E_Silver_Density] computes the ratio of the density of gold to that of silver. All stored data is to be standardized and retrievable by its unique internet path name as here described.

4.7 Unique Path Names

These internet path names are unique, and thus exactly define the metanumber of interest as well as provide an unambiguous name for that single unique metanumber. Furthermore the […] structure can be used as a variable in any mathematical expression, function, or algorithm to retrieve that metanumber. For the file name, row name, and column name, the comparison is made with all white space removed and all case changed to lower case. We envision that all numerical data be standardized as metanumbers in tables appropriate dimensionalities: zero dimension (a single value), or one dimension (a list of values such as the masses of objects), or two dimensions such as the elements table, or even other data tables of any higher dimensionality. The metanumbers in each of these tables are to be stored in comma separated values (CSV) formats thus allowing easy viewing and editing in spreadsheets such as Excel and avoiding any extraneous symbols in any type of markup language which could be problematic with the retrieving algorithm.

5 The Metanumber Algorithm

5.1 Metadata Overview

By “metadata” we refer to the collection of all information (other than units and accuracy) that describe, define, and explain the meaning of the associated metanumber. By “metanumber” we refer to a single numerical value with attached units, accuracy level, and its metadata that describes it. We propose that all numerical data be standardized as metanumbers and stored in archived tables as described here. In the last section we concluded that the units are to be attached as variable names in valid mathematical forms to the numerical value as for example with the velocity of 3.759e4*m*s_1. Here we discuss the metanumber archive tables as mentioned above with greater detail on how metadata is stored and associated with a value. When one normally stores structured numerical data, it can be as a single value, a list of values, a rectangular table of values, or an array of higher dimension (such as when economic data has an associated date or location). Our methodology applies to data arranged in any dimensionality. But data which is laid out as a two dimensional table is the most common with a format like “Entity vs Property”. The “ Entity” might be the 110 chemical elements corresponding to the rows, and where the “Property” might be any of 40 to 50 different properties given in columns, such as density, atomic number, atomic mass, thermal or electrical conductivity, heat capacity, melting point, boiling point , etc. Likewise, the rows might specify a person’s ID and the columns might be properties such as medical data for each person such as DOB, weight, pulse, blood

pressure ... We will use the two dimensional table as an example. The format is to be in rows of comma separated values. This is the most common and open format, and it is easily imported and exported from Excel and other editors as well as read by Python. It is also structured so that at any point it can be automatically loaded easily into any relational database such as Oracle, or SQL if desired by a simple program and fully automated.

5.2 Metadata for a Table as a Whole

Some metadata values are associated with (a) an entire table, while other metadata is associated with just the (b) entries in a given column, (c) or entries in a given row, or (d) perhaps only associated with a specific value (such as longitude and latitude or date-time). To capture metadata associated with the table as a whole, we standardize the first two rows of each table with row one containing the names of fields describing the table, while their associated values are contained in the second row in the corresponding column. This 'table metadata' will contain the table name, a unique table abbreviation, the source(s) of the data such as a web address, the table creator's name, and email address, date created & last updated, security level and codes, references, footnotes, and other associated data related to the collective data in the table as a whole. All of this data is metadata that pertains to every metanumber in the table. Thus this data is applicable and linked to all values in that table.

5.3 Metadata Associated with Specific Rows and Columns

The first column (beginning in row three) is to contain the unique row (entity) name/index identifier (such as the element name of gold) while row three will contain a unique column (property) name/index identifier (such as Thermal Conductivity). Other columns (after the first) and other rows (after the third) can contain other metadata associated with that row or column. For example if the third row has an entry of "Thermal Conductivity" for a given column, then several of the rows beneath the heading of Thermal Conductivity could contain references to the source(s) of those data values, web links to discussions of thermal conductivity, associated equations and web links to explanatory videos or notes. Units would preferably be attached to the end of each numerical value or one could have row or column headings labeled as "*units" in which case the units in that column or row would be distributed by multiplication over the values in the associated row or column. The reference [E_Gold_Thermal Conductivity] will find the unique table named "E" referring to the elements, then find the row labeled "Gold" in column 1, and the corresponding column labeled "Thermal Conductivity" in row three. The metanumber string for gold's thermal conductivity will then be retrieved from that table, that row, and that column and the metadata that pertains to that table, associated with that row and that column are linked to the value without having to be transferred into expressions. Although the heading is Thermal Conductivity, the search will compare the name with all case lowered and all white space removed. And thus it would match 'thermalconductivity'. A comparison of lowered case with removed white space is used for all components of the [table_row_column] lookup. Tables of metanumbers cannot contain any set of the symbols ", _ {}()[ ]" as each of these has a reserved use.

When metadata occupies a row or column, this is to be indicated by having the name of that row or column prefixed by a "%". If the metadata values are unique and could be used as an alternative index then the prefix is to be "%%". An example with the elements table is that both the atomic number and the symbol are unique metadata and thus those columns are labeled as %% Atomic Number and %% Symbol.

5.4 Metadata Associated with Individual Values

Finally, there are times when specific metanumbers have an associated metadata just for that one value such as a longitude-latitude or date-time of measurement. This value specific metadata is encoded in the format {var1=value1|var2=value 2|…} which multiplies the associated metanumber and which always evaluates to unity in any mathematical expression and thus disappears. Thus this expression serves as a wrapper to carry any information to be attached to a specific numeric value. By using the reference [E_Gold_Thermal Conductivity] as a variable, one has therewith a unique name for every archived metanumber along with potentially vast associated metadata that is linked but not transferred until requested.

5.5 Data Archived on Other Servers

When data is stored on other servers (as opposed to the central cloud system in the default directory) then the table name is to be preceded with "(the web address of the server) _ (director path to the file)" followed by a double underscore '__' and then "(the table-name_row_name_column-name)". This gives unlimited capacity to the coding for data retrieval as is done with web addresses with a unique name for every single metanumber value.

5.6 A Distinct Pathname for every Metanumber

A powerful feature of the reference string […] is that it gives a unique name to each metanumber via the unique internet path to each separate value of metadata in the table. Thus the metanumber name links to the table metadata, row metadata, and column metadata without having to include that metadata with the value in the expression. This is a critical concept. Furthermore, as all computational history is archived, this structure supports an essentially unlimited unique line of tracing the meaning, method of measurement, and unlimited other metadata via the indirect reference to the numerical value. This can be of great value with the properties of pharmaceuticals, where the non-numeric metadata can give critical associated data such as batch and expiration data, or in accounting for the tracing of the origins of funds and the means by which they were processed to give the current value. A separate application easily reformats existing data tables. Once this conversion to standard metanumber form is done, one can operate with the data at a speed and accuracy never before possible with full automation and no human intervention. This is especially critical with extremely large data sets and even more so when there are a very large number of data tables from multiple sources. Finally, if data is formatted this way as it is created, by that creator, then it can be subsequently used and shared at

maximum speed and confidence by everyone.

5.7 User Defined Units and Archived Results

A users submitted expressions are always archived into a single table under the user's name. This "user's archive table" is created for each user and contains the values: Seq#, DateTime, Submitted Expression, Resultant MetaNumber, and UnitID. The Seq# is the unique submission sequence number for that user: 1, 2, …. The DateTime is in the format YYYYMMDD:HHMMSS. The Submitted Expression is the string that is submitted for evaluation. The Resultant MetaNumber is the metanumber that results from the evaluation of the submitted expression. The UnitID is an integer consisting of 10 single digits that contains a code for the powers of the fundamental metric units in a given order and thus which is unique for each combination of units. This integer is thus representative of the resulting concept and can be used for different types of analysis, classification of results, tracking of work done, and the creation of networks among users, metadata tabs, and concepts studied. One recalls that any string that is bounded by {} will evaluate to unity (1) and thus can multiply any metanumber at any position although the formal structure places it at the end of a metanumber to denote the specific metadata such as Lon Lat for that value. If one prefaces a submitted expression with {= expression name} such as {= Physics 211 Lab problem 4.38}, then later analysis of that users work, when downloaded to a PC into Excel, can be used to filter and sort different computational efforts. Within Metanumber, one can also reference the result as [my_expression name] in order to use it in other expressions. One can also reference past work as [my_ seq#]. More advanced features and operations are also possible. This means that all metanumbers that are used, whether they are tables using [name_row_column] or past input results as [my_name or #] provide unique names for every possible metanumber. In a sense, the ability to create a (one dimensional) table with [my_name or #] means that a users work creates a standardized table of all evaluated expressions by each user. As the form {=name} is maintained as part of the input expression, it follows that it as well as all component parts of the input expression with other metadata can always be searched for content. If one is careful to use unique neumonic names, then such results can constitute a user's own set of 'units' as units are simply values of metanumbers. It is easy to add additional user defined personal units and constants to the system by prefixing any metanumber expression with {=unit or constant name}*(metanumber expression) such as {=bluetruckvolume}*38*m3. A full discussion of these advanced features would take us beyond the scope of this introduction. The expression "[my_string]" is executed by the substitution of the users unique PIN address for "my" and then all the previous discussions on archived table data apply. Thus the term "my" is automatically replaced by the users PIN when used and thus is kept private for that person. Data and special units for a company or government agency can also be kept secure and private and thus shared only among members of a closed user group. The system allows the use of any previous result with the unique sequence number "i", referenced as [my_i] , in a new expression as a variable. Thus all the linkages back to the origin for every numerical value are maintained among contributing users and their work. The software can peel back layer after layer of how a given value was created along with all associated metadata

tags, assumptions, equations, accuracy levels, literally everything, in manufacturing a product, pharmaceutical, account, and scientific measurement, thus supporting levels of artificial intelligence never before possible and thus providing the complete evolutionary history of every number.

5.8 Current Archived Data Tables

Our interdisciplinary team has been creating sample metanumber data tables in several disciplines as examples for training. It is the identification of those tables and that data that is most generally utilized and shared among groups that will enable highly innovative computations crossing multiple disciplines supporting both research programs and student course work. Optimizing our methodology for these tables has been the main research thrust in parallel with refining our algorithms, notations, and user tools.

5.9 Conclusions

To conclude the metadata descriptor section, we have shown that by standardizing all metanumbers in tables with unique file, row and column names, that the internet path to the metanumber [file_row_column …] provides a unique name for entering the metanumber into a mathematical expression and also makes all action sequences fully traceable. The same functionality applies also to each users past work with [my_name or seq#] although this table is private to the user unless more advanced actions are executed for the sharing of data among users.. But an equally important feature is that all table, row, and column metadata is linked by that name enabling intelligent systems to form networks among tags, concepts, dates and locations, units, special constants, and even computational actions. Metadata for pharmaceuticals can describe side effects, dosages, batch numbers, expiration dates, and other metadata which can be automatically linked to recipient patients without the specific transfer of such extensive metadata in the computational process as one only needs the internet path. We will discuss these potential networks in the following.

6 The Numerical Accuracy Standard and Algorithm

6.1 Numerical Accuracy Overview

The most complex domain is the automated processing of numerical uncertainty which is a core research area of the author and is too technical for a full presentation here. The author has jointly published this research with a university colleague and also presented it in the AMUEM international conference on numerical uncertainty in Sardagna, Trento Italy [6] and has been awarded a U.S. patent for this algorithm. Numerical “accuracy” or equivalently “uncertainty” can be computed by several methods: A numerical accuracy algorithm could (a) treat the metanumber value as being exact. But with the exception of the integer counting of objects, values that are represented by real numbers cannot be either measured or represented exactly.

However there are a few “real numbers” that are by definition established as exact such as the speed of light or the number of cm in an inch. We could (b) treat the value with an uncertainty following the last listed valid digit when expressed in scientific notation and follow the standard statistical rules for determining the accuracy of the result for each operation given the accuracy of the operands. This method has the advantage that if one intercepts the raw data prior to processing, then one can count the number of digits present. We could (c) assume a normal (Gaussian) distribution to the value and combine the standard deviations of operands thus basically treating each value as a distribution instead of a real number which is done in the most exacting of cases and which is the most accurate. But the problem with this is that one almost never knows the actual standard deviation. Most measurements are made once or twice and time, effort and cost are not available to reasonably determine the standard deviation of the value except when different scientific teams expend a great effort over many measurements as with the NIST fundamental constants. It is furthermore an assumption that such normal probability distributions accurately represent the true distributions without skewness or kurtosis. Furthermore normal distributions do not close under operations other than multiplication and division meaning that the mean and variance as a pair do not close mathematically except in approximation. This is in contrast to the complex numbers which do close mathematically. We must choose the method of combining values where the accuracy level is simply indicated by an uncertainty of one in the last digit since that information is known and the standard deviation is not known. This method of representing error is always known as one has only a fixed number of valid digits in the value which is always observable. We will choose this method because it is always known and displayed. When a more accurate representation of uncertainty of the probably distribution such as standard deviation is available, the algorithm is able to utilize that framework. The very complex patented technique of the author is so exacting that the supporting information is almost never available and it will not be offered in this release version.

6.2 Use of Accuracy as Measured by Significant Digits

Even this method is in general very complex because one is now representing each numerical value with a pair of values: the known value as represented by a sequence of digits along with an uncertainty value. However it is essential that this error of the last digit be captured in the algorithm at the point when the number is introduced either (a) keyed in by a user or (b) listed in a data table. This is because all mathematical operations between the value of interest and other values in an expression will erase the knowledge of the accuracy of each and provide results of operations that only indicate the limit of accuracy being kept by that computer software. Computer generated results, without an accuracy algorithm, always generate a great overrepresentation of the true accuracy with excessive digits for the value as one can see when dividing say 17 by 3 when the result might only be accurate to one digit. Even this method of accurately computing the number of significant digits of the result is very complex algorithmically as different operations and different functions follow different rules for accuracy combinations and thus the expression must be parsed for the correct operational sequence and the values themselves cannot now be just real numbers but must be a new mathematical class of objects. Here we

are extremely fortunate that several such algorithms have been created for inclusion with the Python language.

6.3 Python Uncertainty Package

We will use the Python Uncertainty package which can be found at http://pythonhosted.org/uncertainties by Dr. Eric O. Lebigot [2]. Our procedure is this: Each data value when loaded from MetaNumber tables, or a keyed entry by the user must be immediately captured (using python code and 'regular expressions' methodology) and converted to a new type of object which is defined as a "ufloat" of a string which contains the known digits. Once this is done then this uncertainty algorithm evaluates the component values and maintains both the full accuracy of the result as though there is no error and simultaneously gives the resulting uncertainty of the numerical result. This algorithm also correctly mediates the operations between the ufloat values and the other values which are known exactly. To utilize the Python Uncertainty algorithm values, they must be identified by the software in order to be converted to the ufloat class of objects or must be identified as exact and thus treated as an exact standard value with no error. Our technique is to identify all uncertain values by numbers which contain a decimal point such as 5.793e5 where the value of 5.793 will be identified via the decimal point and converted to a ufloat and then multiplied by 1e5. An exact value must therefore be entered without a decimal by adjusting the exponent to remove the decimal as 5.793e5, if exact would entered as 5793e2 and thus stored as an exact real number. This methodology requires no new symbol or other indicator and is fully automatic. Thus the coding of numerical certainty or uncertainty does not need an explicit coding but rather uses this implicit coding methodology of our capture algorithm. This algorithm must act on all numeric values in both the data tables as values are brought into an expression and on keyed values that are entered by the user.

6.4 Conventions for Exact & Uncertain Values

We will take all unit conversions to be exact. The current algorithm automatically converts all table values to ufloat objects along with automatic conversion of all user keyed numerical components to ufloat based metanumbers when there is a decimal present as well as decimal present values that are keyed in by the user. For exact values, the user will only need to remove the decimal point by altering the associated exponent in the scientific notation in order to enter a value as exact, either in the tables or in the keyed values. Thus 3.2e4 becomes 32E3 with no decimal in the value to indicate that the value is exact. Keyed or table values that have a decimal present will be assumed to be ufloat objects. Any expression that contains a missing value will automatically return a missing metanumber with the metanumber code. Thus the result of this convention on encoding uncertainty is that no explicit notation is required other than the presence or absence of the decimal point in the value to get the value to be treated as uncertain or exact respectively.

7 Network and Cluster Analysis

7.1 Networks and Cluster Identification

Networks represent one of the most powerful means for representing information interrelationships (topologies) among abstract objects called nodes which are identified by sequential integer 1, 2,... Cluster analysis on these networks can uncover the nature of those structures. We have been able to link objects in this numerical standardization using our past research on the mathematical classification of networks and associated cluster analysis. A network is defined as a square (sparse) matrix (with the diagonal missing) that consists of non-negative real numbers, and which normally is very large, that represent the connection degree between node i and node j as C_{ij}. The mathematical classification and analysis of the topologies represented by such objects is one of the most challenging and unsolved of all mathematical domains.

We have built some far-reaching extensions of the metanumber system that can be constructed on the foundation laid by the standardization described above. Let us first consider the standardization of numerical data as metanumbers in tables with units, accuracy, and metadata descriptors linked as described above. Next consider how each number in such a design has a unique reference name (such as [server path_dir_table_row_column_...]). Likewise there is a unique name for every past computation of each user (such as [my_ref#]), as well as a link for each shared computational path by a project team under a given subject name (or [subject_user_ref#]. We suggest that the resulting system supports vast and powerful automated networks which can be constructed as described in 7.3 and 7.4 below.

7.2 Mathematics Underlying Our Network and Cluster Discoveries

This section will provide a very brief overview of our mathematical results that provide a new foundation for network and cluster analysis. In previous work the author discovered a new method of decomposing the continuous general linear (Lie) group of (n x n) transformations into a Markov type Lie group (with n^2-n parameters) and an Abelian scaling group (with n parameters). Each is generated as is standard, by exponentiation of the associated (Markov or scaling) Lie algebra. The Markov type generating Lie algebra consists of linear combinations of the basis matrices that have a “1” in each of the (n^2-n) off-diagonal positions with a “-1” in the corresponding diagonal for that column. When exponentiated, the resulting matrix M(a) = exp(a C) conserve the sum of elements of a vector upon which it acts, but can take a vector with positive components into one with some negative components (which is not allowed for a true Markov matrix). However, if one restricts the linear combinations to only non-negative values then we proved that one obtains all discrete and continuous Markov transformation of that size. This links all of Markov matrix theory to the theory of continuous Lie groups and provides a foundation for continuous Markov transformations.

Our next discovery was that every possible network (C_{ij}) corresponds to exactly one element of the Markov generating Lie algebra (those with non-negative combinations) and conversely, every such Markov Lie algebra generator corresponds to exactly one network! Thus they are isomorphic and one can now study all networks

by studying the associated Markov transformations and Lie algebra and associated groups. Our subsequent recent discoveries are (a) all nodes in any network can be ordered by the second order Renyi entropy of the associated column in that Markov matrix thus representing the network by a Renyi entropy spectral curve and removing the combinatorial problems so that now one can both compare two networks (by comparing their entropy spectral curve) as well as study the change of a networks topology over time. One can even define the "distance" between two networks (as the exponentiated negative of the sum of squares of difference of the Renyi entropies). We recently showed that the entire network topology of any network can be represented by the sequence of the necessary number of Renyi entropy orders. This is similar to the Fourier expansion of a function such as for a sound wave where each order represents successively less important information. Our next and equally important result was that an agnostic (assumption free) identification of the n network clusters can be found from the eigenvectors for this Markov matrix which not only show the clustering of the nodes but actually rank the clusters using the associated eigenvalues thus giving a name (the eigenvalue) for each cluster.

Our final recent development that is important for the current issues is that one can generate two different networks from a table of values T_{ij} for entities (such as the chemical elements in rows) with properties (such as density, boiling point, …) for each element in columns. To do this we first normalize the table by finding the mean and standard deviation of each column and then rewrite each column value as the number of standard deviations (represented by the table value), divided by the mean for that column, away from the norm. This process also removes any units that are present as the results are dimensionless. We then define a network C_{ij} among the n entities (here the elements listed in rows) as the exp of the negative of the sum of the squares of the differences between T_{ik} and T_{jk} thus

$C_{ij} = \exp-\Sigma_k(T_{ik} - T_{jk})^2$. This gives a maximum connectivity of '1' if the values are the same and a connection of '0' if they are far apart as would be required for the definition of a network. We form a similar network among the properties for that table. One then, as before, adjusts the diagonal to be the negative of the sum of all terms in that column to give a new C, forms the Markov matrix M(a) = exp (aC) and finds the eigenvectors and eigenvalues for M to reveal the associated clustering. The rationale for how this works can be understood when the Markov matrix is viewed as representing the dynamical flow of an invisible conserved substance among the nodes. This methodology includes and generalizes the known methodology with the Lagrange matrix for a network.

7.3 User-data Type Networks

Users (using the PIN#), can each be linked to (a) unit id (UID) hash values of the results of their calculations, as well as to (b) the table, row, and column names of each value. These linkages can be supplemented with linkages of each user to those universal constants that occur in the expressions which they evaluate. The resulting network links users to (a) concepts such as thermal conductivity, (b) substances such as silver alloys, and (c) core constants such as the triple point of methane, the Boltzmann constant, Planks constant, or the neutron mass. The expansion of this network in different powers, giving the different degrees of separation, can then link

users via their computational profiles, (the user i x user j component of C^2) as well as linkages among substances, metadata tags, and constants. The clustering revealed in different levels of such expansions then reveals groups of users with linkages that are connected by common computational concepts. Users working on particular domains of pharmaceuticals and methodologies are thus identified as clusters as well as groups of astrophysicists that are utilizing certain data and models. At that same time, the clustering can identify links among specific substances, models, and methodologies. Our current research is exploring such networks and clusters as the underlying metanumber usage expands.

7.4 Table Generated Networks among Entities and also among Properties

The methodology for converting a table to a network among row items or among column items was briefly discussed above. We are currently exploring the clusters and network analysis that can be generated from the tables of standardized metanumber values. We have done this for the elements table which was very revealing as it displayed many of the known similarities among elements. The cluster of iron, cobolt and nickel was very clear as well as other standard clusters of elements. We are currently analyzing a table on the properties of pesticides and we have just obtained a table of 56 nutrients for 8600 natural and processed foods. The study of clustering in foods based upon their nutrient and chemical properties as well as the clustering among the properties themselves will be reported as available on the www.metanumber.com web site. As usage of the metanumber system expands, we will also report the results of clustering among similar scientific investigations. These results are expected by the end of August 2015.

7.5 Multiuser Collaboration Application

Often with more complex computations and multi-tier problems, it is useful for a number of engineers, scientists, students, or business workers to collaborate on a problem. This requires that comments be shared as explanatory of the work that is in progress as well as questions and responses to others in the collaboration. This is even true for a user who wishes to document his or her own work. The metanumber application supports documenting text which is archived as an entry just like an expression that is to be evaluated. All that is necessary is to enter a "#" as the first character (similar to the method of entering a remark or comment line in python or other languages). When the "#" is seen, then all processing is bypassed and all text up to the end of that entry (cr lf) is archived as a string. Then when a user's past history is exported or read or shared, then these text entries are seen correctly positioned among the computations giving explanations or recording ones technique. The collaboration part is achieved by entering a command {subject = 'some name'|PW} where 'some name is the name of the subject to be shred. A file is created with that name (in lower case with white space removed and with the attached password. The creator of the subject must then share the name and password (if any) with users in a closed group. The subject stays active until the end of the session or until one enters

{subject = }. While the subject is active, each user with such an active subject has each line that is entered to be written to that new (subject name) file in the form: UserPIN, Seq#. This is all that is needed in the shared file because each user can then open that file to read only all of the entries from other in the group. More information will be available from www.metanumber.com help screens.

8 Conclusions

The use of these algorithms with the approximately 800 units and fundamental constants, can support a standardization of all numerical data. This metanumber environment also presents all data in human and machine readable formats and satisfies the listed set of 10 essential requirements. The resulting system provides a vast saving of time, costs, and a removal of associated errors by supporting the instantaneous use of data without preprocessing. This system can support new levels of artificial intelligence and improved human interaction. It can also support new methods for the creation of novel networks [9] in numerical data that in turn can support new methods of cluster analysis [10].

References

1. Johnson, Joseph E., 2009, Dimensional Analysis, Primary Constants, Numerical Uncertainty and MetaData Software", American Physical Society AAPT Meeting, USC Columbia SC
2. Leibigot, Eric O., 2014, A Python Package for Calculations with Uncertainties, http://pythonhosted.org/uncertainties/.
3. Johnson, Joseph E., 2013, The Problem with Numbers – MetaNumbers – A Proposed Standard for Integrating Units, Uncertainty, and MetaData with Numerical Values, USC Physics Colloquium
4. Johnson, Joseph E., 2014, New Software Developed for Interdisciplinary Data Sharing and Computation: Units, Uncertainty, & Metadata, USC School of the Earth, Ocean, and Environment Colloquium
5. Johnson, Joseph E. 1985 US Registered Copyrights TXu 149-239, TXu 160-303, & TXu 180-520
6. Johnson, Joseph E, Ponci, F. 2008 Bittor Approach to the Representation and Propagation of Uncertainty in Measurement, AMUEM 2008 International Workshop on Advanced Methods for Uncertainty Estimation in Measurement, Sardagna, Trento Italy
7. Johnson, Joseph E., 2006 Apparatus and Method for Handling Logical and Numerical Uncertainty Utilizing Novel Underlying Precepts US Patent 6,996,552 B2.
8. Johnson, Joseph E. 1985, Markov-Type Lie Groups in GL(n,R) Journal of Mathematical Physics. 26 (2) 252-257
9. Johnson, Joseph E. 2005 Networks, Markov Lie Monoids, and Generalized Entropy, Computer Networks Security, Third International Workshop on Mathematical Methods, Models, and Architectures for Computer Network Security, St. Petersburg, Russia, Proceedings, 129-135US
10. Johnson, Joseph E. 2012 Methods and Systems for Determining Entropy Metrics for Networks US Patent 8271412
11. Johnson, Joseph E, & Cambell, William 2014, A Mathematical Foundation for Networks with Cluster Identification, KDIR Conference Rome Italy

12. Johnson, Joseph E. 2014, A Numeric Data-Metadata Standard Joining Units, Numerical Uncertainty, and Full Metadata to Numerical Values, EOS KDIR Conference Rome Italy