Applications of Iterated Exponentials

Andrew Robbins

Introduction

What may have begun as a solution to the inverse function of $x^{1/x}$ has become a large and intricate discipline that attempts to find simple answers to the question: *What lies beyond exponentiation?* (Geisler). Iterated exponentials have a long and obscure history as part of pure mathematics. Today they are also classified as part of iteration theory, which itself is a subject that has some overlap with dynamical system theory and functional equation theory. Iteration theory as a whole has many applications to all sciences and throughout mathematics (Woon 4). Applications of iteration theory abound, but applications of iterated exponentials are more scarce.

Throughout the subject's past, mathematicians have spent a lot of time working out the details of how to calculate iterated exponentials, but have focused very little on their applications. The goal of this paper is to show that iterated exponentials in and of themselves have applications to certain sciences and to other subjects in mathematics. The subjects that employ iterated exponentials include but are not limited to: algorithmic complexity theory (Rubstov and Romerio 2), approximation theory (Yukalov and Gluzman), combinatorics and discrete math (Geisler), computer arithmetic (Holmes), dynamical systems (Wolfram), iteration theory (Bennet), series

expansions (Barrow), and population modeling (Robbins). Each one of these subjects has found some common ground in which iterated exponentials play a role.

Nearly 230 years ago, Euler proved the first theorem about iterated exponentials (Knoebel). It seems fitting, then, that near this anniversary of the subject, we gather the purposes which have motivated so many great mathematicians on the search for the answer to the question: *What lies beyond exponentiation?*

I. Terminology and notation

The terms for different aspects of exponentiation should be well-known, but for review, we will cover them here. *Exponentiation* refers to the proccess of raising a *base* b to an *exponent* a and is written as b^a. Exponentiation is a function of two variables. When a function of one variable is considered, one of two other functions naturally arise. A *power* function is a function of one variable x, defined as x^a, and an *exponential* function is a function of one variable x, defined as b^x. When exponentiation appears in the exponent, then the convention $a^{(b^c)} = a^{b^c}$ is used.

The terms for iterated exponentials are based on the terms for exponentiation, but are more specialized. There are three major types of these expressions. An expression of the form $\mathrm{T}_{k=1}^{n}(a_k) = a_1^{a_2^{\cdot^{\cdot^{a_n}}}}$ is called a *tower* of height n. When all exponents but the last are the same, we get an expression of the form $\exp_b^n(x) = b^{\cdot^{\cdot^{b^x}}}$ with n instances of b, called an *iterated exponential*. Lastly, when all of the exponents are the same we get an expression of the form ${}^a b = b^{b^{\cdot^{\cdot^{b}}}}$ with a instances of b, and this is called *tetration*. When considering tetration as a function of two variables, the variable a is called the *height* and the variable b is called the *base*. A function of one variable x defined as ${}^a x$ is called the a-th *tetrate*, and a function of one variable

x defined as ${}^{x}b$ is called the base-b *tetrational*. The inverse functions of tetration are called *super-roots* for the functions that satisfy $\sqrt[a]{{}^{a}x} = x$ and *super-logarithms* for the functions that satisfy $\operatorname{slog}_b({}^{x}b) = x$. Super-roots are inverses of tetrate functions, and super-logarithms are inverses of tetrational functions.

The terms for iteration in general should also be well-known, but are covered here for review. Given a function $f(x)$, the *iteration* of that function is a function of two variables n and x, and is written $f^n(x) = f(f^{n-1}(x))$. There are two functions of one variable based on the iteration of a function. The n-th *iterate* is a function of the variable x, and the *iterational* from x is a function of the variable n.

II. Difficulties of iterated exponentials

The study of iterated exponentials is not a well-developed subject, despite having existed for hundreds of years. One of the reasons for this is that using them is extremely difficult, so many have left the issue for someone else to consider. It is common to assume that *some* of the properties of exponentials carry over to iterated exponentials, but it soon is evident that almost *none* of the properties carry over. This is in sharp contrast with exponentiation, which satisfies a treasure trove of equations. The *only* law of tetration is ${}^{(a+1)}b = b^{({}^{a}b)}$ which can be rather limiting at first, until it becomes familiar. Iterated exponentials, on the other hand, stem from iteration, and thus satisfy two such laws: $\exp_b^{a+1}(x) = \exp_b^a(b^x) = b^{\left(\exp_b^a(x)\right)}$.

Although the usual properties of exponentiation do not apply to tetration, it is common to think that they do. The most common misconception is with the reciprocal-power-root rule. For exponentiation, $x^{1/n} = \sqrt[n]{x}$, but it has been proven to not apply to tetration (Rubstov and Romerio 26). Rubstov *et.al.* showed with a

limiting case that assuming ${}^{1/n}x = \sqrt[n]{x}$ leads to a contradiction because:

$$\lim_{n\to\infty} {}^{1/n}x = {}^{0}x = 1 \qquad \text{and} \qquad \lim_{n\to\infty} \sqrt[n]{x} = x^{1/x} \neq 1$$

and if it leads to a contradiction, then it cannot be true.

Classifying iterated exponentials has not been very obvious, as it is with many mathematical concepts. Since the study of iterated exponentials has a lot of overlap with dynamical systems and functional equations, many mathematicians classify it as one or the other or both (Woon 1). Two of the most versatile tools used with tetration are results from dynamical systems (fixed points) and functional equations (natural Abel function) which form the foundation of iterated exponential theory.

III. Methods of calculating iterated exponentials

Iterated exponentials are easy to calculate for integer iterations since one can just use exponential functions repeatedly, but for non-integer iterations their calculation is much more difficult. It is not impossible, though, because non-integer iteration methods do exist. The two tools mentioned earlier provide such a way to calculate some instances of iterated exponentials which arise most frequently. Also, not only do dynamical systems apply to tetration, but tetration also applies to dynamical systems, as noted in the following:

> For our purposes, super-exponentiation naturally arises from analyzing fixed points of exponential functions ... (Wassell 111)

so tetration is not as much a rare topic as it is of core interest in the dynamics of elementary functions, specifically exponential functions.

There many methods of calculating the continuous iteration of a function. One way is just to use the function, and apply it to itself n times, but this is limiting the domain of iteration to integer n. A second way is to use pattern recognition to reduce the iteration of $f(x)$ to some simple rule. If the simple rule allows extension from integers to real numbers, then there is no need for other methods. For example, the addition of a constant ($f(x) = x + a$) can be iterated to find that $f^2(x) = x + 2a$ and $f^3(x) = x + 3a$. Using pattern recognition we can see that the iteration of f is $f^n(x) = x + na$, and we can now replace n with any number we like.

For functions that we can not use pattern recognition on, more advanced methods of iteration are needed. Each method comes from a different subject, so although each method could be described by the subject name, most authors use specific terminology for each method. The first method stems from the idea of a fixed point in dynamical systems, called *regular iteration* (Szekeres 92). The second method (Walker) stems from the solution of functional equations, called *natural iteration* (Trappmann). The third method stems from the construction of an infinite matrix representation of a function, called *matrix iteration* (Aldrovandi 9). All methods require a series expansion, and use the series to construct a new series that is then used to find arbitrary iterations of that function. Although each of these methods can give slightly different results, together they are a fairly complete set of tools.

IV. Applications due to growth rate

> We may sum up these considerations as the 'principle of crudity': the practical upshot is that in estimating a number $a^{b^{c^{\cdot^{\cdot}}}}$ it is worth taking trouble to whittle down the top index, but we can be as crude as we like about things that bear only on the lowest ones (Littlewood 164).

Large numbers appear naturally in astronomy, quantum physics, computational complexity, enumerative combinatorics, and many more subjects. This has led to an increasing interest in the representation of large numbers, specifically for computer arithmetic (Holmes). The first step in representing large numbers is to begin to use alternatives to scientific notation (which uses logarithms) such as tetrational notation (which uses super-logarithms). Just as logarithms assign small numbers to large numbers (for example $\log_{10}(1000) = 3$), *super-logarithms* assign smaller numbers to larger numbers (for example $\operatorname{slog}_{10}(10000000000) = 2$). The super-logarithm is defined implicitly by the equation $\operatorname{slog}_b(b^x) = \operatorname{slog}_b(x) + 1$, and the initial condition $\operatorname{slog}_b(1) = 0$. Using this definition, one can produce the values $\operatorname{slog}_b(0) = -1$, $\operatorname{slog}_b(b) = 1$, $\operatorname{slog}_b(b^b) = 2$, and so on. As you can see, the super-logarithm helps condense very large numbers into small numbers that we can handle.

The way we perceive numbers can be strange at times. Intuitively we care a lot about the difference between 5 and 20, but not as much about the difference between 5 million and 20 million. There is not as much feeling behind such large numbers, even orders of magnitude 10^5 and 10^{20} have little meaning beyond the numbers used to represent them. A number class system was developed (Munafo) that assigns to every positive number a *Munafo class number* in such a way that numbers in the same Munafo class are approximately the same to our initial perception of that number, in other words, that the numbers are in the same perception class.

Robert Munafo describes this number classification system by saying that the numbers from one to six are of class 0, because when we see anywhere from one to six dots, we can perceive the number immediately and exactly. He then goes on to say that numbers from seven to a million (10^6) are class 1, because although we could

count them, we usually have some approximate idea of how many dots are showing. He continues by saying that numbers from $10^6 + 1$ all the way up to 10^{10^6} are class 2 numbers, which includes googol. Following this further, we can say numbers from $10^{10^6} + 1$ up to $10^{10^{10^6}}$ are class 3 numbers, which includes googol-plex. Class 3 numbers are those which are far beyond our direct perception, and also beyond the current state of computer arithmetic. Since not even computers can handle class 3 numbers with today's methods, one could say that they would never be used in any math or science. As we shall see later, this has not been the case.

Using the super-logarithm we can reduce Munafo's definition to the equation:

$$C(n) = \lceil \mathrm{slog}_{10}(n) - \mathrm{slog}_{10}(6) \rceil$$

which allows us to calculate Munafo classes much easier, if the super-logarithm is known. If the super-logarithm is not known, or not a reasonable option, then the *discrete super-logarithm* may be a better choice. The discrete super-logarithm can either be the ceiling or the floor, but here we use the ceiling for simplicity.

$$\lceil \mathrm{slog}_b(n) \rceil = \begin{cases} -1 & \text{if } n \leq 0, \\ \lceil \mathrm{slog}_b(\log_b(n)) \rceil + 1 & \text{if } n > 0. \end{cases}$$

This definition of the discrete logarithm is very similar to the definition of the iterated logarithm (Wikipedia "iterated logarithm"). The only primary difference between these two functions in that the iterated logarithm is zero when $n \leq 0$, and the discrete logarithm is negative one. Aside from this, they are identical.

Iterated exponentials in and of themselves are most useful for representing large numbers. Although some systems of representing large numbers with tetration do require that tetration be extended to real numbers first, using iterated exponentials,

on the other hand, does not require such an extension. With the usual *scientific notation* for large numbers a number is written in the form: 1.234×10^n in which the exponent n is an integer. In order to make this representation unique the mantissa 1.234 is usually restricted to a number between 1 and 10 (not including 10). Using a similar method, we can write any positive number in the form $\exp_{10}^n(1.234)$ with iterated exponentials where the height n is an integer, which does not require an extension to real numbers. In order to make this representation unique we can again restrict the argument 1.234 to be a number between 1 and 10 (not including 10). This is unique because if an expression is of the form $10^{10^x} = \exp_{10}^2(x)$ and $x = 10$, then $10^{10^{10}} = \exp_{10}^3(1)$, so any $x > 10$ will be expressible in iterated exponential form with an argument between 1 and 10 (not including 10). This notation for large numbers is called *tetrational notation* (Holmes 6).

One of the applications of iterated exponentials is in the study of combinatorics. Combinatorics is the study of how things combine and the number of possible objects one can form under certain restrictions. One such system is called *hierarchies* of height n (Geisler). Eric Temple Bell was the first person to realize the connection between these numbers and the iteration of $f(x) = \boldsymbol{e}^x - 1$ (Bell), where he proves many interesting theorems about the number theoretic properties of the iteration of $f(x)$. The first few derivatives of $\boldsymbol{e}^{\boldsymbol{e}^x-1}$ at zero give: $\{1, 1, 2, 5, 15, 52, ...\}$ which are now known as the Bell numbers. These represent hierarchies of height 2 because $\boldsymbol{e}^{\boldsymbol{e}^x-1} = f(f(x)) + 1$ includes the *second* iteration of $f(x)$. To see that these numbers really represent hierarchies of height two, imagine the following. With one element the heirarchies of height two are $\{\{a\}\}$ giving one possibility. With two elements the hierarchies of height two are $\{\{a\}, \{b\}\}, \{\{a, b\}\}$ which gives two possibilities. With

three elements the hierarchies of height two are:

$$\{\{a\},\{b\},\{c\}\},\{\{a\},\{b,c\}\},\{\{b\},\{a,c\}\},\{\{c\},\{a,b\}\},\{\{a,b,c\}\}$$

giving five possibilities. Putting all of this together, we get the sequence $\{1,2,5,...\}$ which are exactly the numbers that the second iteration of $f(x)$ generates.

Another system that can be enumerated with tetration is called *forests* on n nodes of height less than h. These do not map directly to sets, so there may be an abuse of notation as a result. The number of possible forests can be obtained by imagining the following. The number of forests on one node is just $\{\{a\}\}$ which gives one possibility. The number of forests on two nodes: $\{\{a\},\{b\}\},\{\{a,\{b\}\}\},\{\{b,\{a\}\}\}$ gives three possibilities. The number of forests on three nodes is quite large, and to skip the proof, ends up being 10 for height one. So putting this all together we get the sequence $\{1,3,10,41,196,...\}$ (Sloane A000248) for the number of forests on n nodes of height less than two. The function whose derivatives at zero generate this sequence is ${}^{a}(\boldsymbol{e}^{x})$. The repeated derivatives of $\left\{1,\boldsymbol{e}^{x},\boldsymbol{e}^{x\boldsymbol{e}^{x}},\cdots\right\}$ at zero give:

a, n	0	1	2	3	4	5	6	(Sloane OEIS)
0	1	0	0	0	0	0	0	(A000007)
1	1	1	1	1	1	1	1	(A000012)
2	1	1	3	10	41	196	54	(A000248)
3	1	1	3	16	101	756	954	(A000949)
4	1	1	3	16	125	1176	2934	(A000950)
5	1	1	3	16	125	1296	12087	(A000951)
∞	1	1	3	16	125	1296	16807	(A000272)

which are the numbers of forests on n nodes of height less than a.

The *original* application of tetration was to Cantor's ordinal numbers in number theory, and the expressions used to generate the next one (Goodstein). The term *tetration* was coined in a paper written by Robert L. Goodstein about transfinite ordinals, and uses tetration to show that one can continue representing ordinals far beyond the effective limit of ω^ω. He showed that one could write $\omega^{\omega^\omega} = {}^3\omega$ and, in doing so, saved number theorists a lot of trouble. Others have also noticed the effectiveness of tetration and higher operators at describing Cantor's transfinite ordinals (Nambiar), while some are intent on standardizing notation (Knuth).

V. Applications due to other reasons

An application often forgotten is the pedagogical one. In the curriculum of iteration theory, many examples are used to demonstrate the ideas of iteration theory. Usually these examples involve either a polynomial, or a linear fractional of some kind. These functions are easier to iterate than exponential functions. Since exponential functions are so much harder to iterate than polynomials, they make great examples for more advanced iteration techniques. It may well be that the abstract curiosities of today will be the textbook examples of tomorrow.

Another application that is easy to forget is that of aesthetics and beauty. The sequence of operators that tetration is a part of: addition, multiplication, exponentiation, tetration, pentation, and so on, collect all major mathematical operators into a single work of beauty, and in doing so, serve the purpose of pleasing the psyche. Although this can be dismissed in that it doesn't help build bridges or put food on the table, it can be argued that it is just as necessary. In fact some go even further as to try and ascribe physical meanings to the higher operators beyond exponentiation.

The logic behind these investigations usually starts with the observation that the first three operators are so important, as can be seen in the following:

> A hierarchy of operations on the positive integers can be denoted using the arrow symbol. These operations may be of interest to number theorists, as perfect squares, primes, and so on can be generalized into higher orders of basic operations. The family of continuous arrow functions is analogous to linear functions at the multiplicative level and quadratics, cubics, and so on at the exponential level. The arrow functions may find an application in the sciences. Nature uses the first three basic functions profusely: why should she not use arrow? (Bromer 173)

which in some sense is a completely valid question. On the other hand, this is relying on pattern recognition, which is not applicable in all situations.

There is a way to intuitively understand tetration. Addition is the operation that arises when we juxtapose two lengths in the same dimension. Multiplication is the operation that arises when we juxtapose two lengths in two dimensions. Exponentiation is the operation that arises when we juxtapose a length x in n dimensions. So, tetration would be the operation that arises when we juxtapose a length x in ${}^{(n-1)}x$ dimensions. What does this mean? It means that not only is x the length, but it also effects the number of dimensions the length is rotated through.

Another interpretation of tetration uses differential equations. If we differentiate $f(x) = {}^{x}\boldsymbol{e}$ we get: $f'(x) = ({}^{x}\boldsymbol{e})f'(x-1)$. This means the function satisfies the equation: $f'(x) = f(x)f'(x-1)$, or rearranging we get: $f(x) = f'(x)/f'(x-1)$. In other words, the tetrational function $f(x)$ is a function that is equal to the quotient between its *current* rate of change and its *previous* rate of change.

Conclusion

Since many students and professionals might never encounter these applications one could argue that iterated exponentials should remain classified as part of pure mathematics, but the fact that there are so many applications in a wide variety of subjects brings this into question. The hope that the subject brings to computer arithmetic, or to exact solutions of physical systems cannot be ignored. Perhaps when all of these applications are seen together, it will bring a more unified view of iterated exponentials, and their proper place in the mathematical classification system will be evident. Although the instances in which iterated exponentials are used may seem disparate, there is a joy to be found in searching beyond the known mathematical operations for new operations that give answers to difficult questions. If anything else is true about iterated exponentials it is that they quench the thirst for mathematical unity, and as a side note, are also useful in a handful of sciences.

On the other hand, the branches of mathematics in which iterated exponentials are finding the most use are: algorithmic complexity theory, number theory, and computer arithmetic. These subjects would dramatically different without iterated exponentials. With these ever faster growing functions at hand, complexity theorists are finding new upper bounds, number theorists are finding new isomorphisms, and computer scientists are finding new representations. When multiplying two very large numbers that are too big to handle, current computers usually fail in what is called an *overflow*. If we had not taken the time to learn more about iterated exponentials, we would have to settle for computer overflows for the rest of our life. However, mathematicians and scientists across the board are ensuring that within our lifetime, overflows will be a thing of the past.

Work Cited

Author Unknown. "Iterated logarithm." Wikipedia, Nov 2007 <http://en.wikipedia.org/wiki/Iterated_logarithm>.

Author Unknown. "Tetration." Wikipedia, Nov 2007 <http://en.wikipedia.org/wiki/Tetration>.

Aldrovandi, R. Special Matrices of Mathematical Physics: Stochastic, Circulant and Bell Matrices. New Jersey: World Scientific, 2001.

Aldrovandi, R.; Freitas, L. P. "Continuous Iteration of Dynamical Maps." Journal of Mathematical Physics 39 (1998): 5324-5336.

Barrow, D. F. "Infinite Exponentials." American Mathematical Monthly 43.3 (Mar. 1936): 150-160.

Bell, E. T. "The Iterated Exponential Integers." Annals of Mathematics 2nd ser. 39.3 (Jul. 1938): 539-557.

Bennet, A. A. "Note on an Operation of the Third Grade." Annals of Mathematics 2nd ser. 17.2 (Dec. 1915): 74-75.

Bromer, Nick. "Superexponentiation." Mathematics Magazine 60 (1987): 169-174.

Conway, J. H.; Guy, R. K. The Book of Numbers. Berlin: Springer-Verlag, 1996.

Geisler, Daniel. "Tetration.org." Nov 2007 <http://www.tetration.org/>.

Goodstein, Robert L. "Transfinite Ordinals in Recursive Number Theory." Journal of Symbolic Logic 12.4 (1947): 123-129.

Gralewicz, P.; Kowalski, K. "Continuous Time Evolution from Iterated Maps and Carleman Linearization." Chaos, Solitons & Fractals 14.4 (2002): 563-572.

Holmes, W. Neville. "Composite Arithmetic." Computer 30.3 (Mar. 1997): 65-73.

Knuth, Donald E. "Mathematics and computer science: Coping with finiteness." Science Magazine 194 (1976): 1235-1242.

Knoebel, R. A. "Exponentials Reiterated." American Mathematical Monthly 88.4 (Apr. 1981): 235-252.

Kowalski, K.; Steeb, W. H. Nonlinear Dynamical Systems and Carleman Linearization. New Jersey: World Scientific, 1991.

Littlewood, J. E. "Large Numbers." Math. Gazette 32.300 (Jul. 1948): 163-171.

Munafo, Robert. "Large Numbers." Nov 2007 <http://home.earthlink.net/~mrob/pub/math/largenum.html>.

Nambiar, K. K. "Ackermann Functions and Transfinite Ordinals." Applied Mathematics Letters 8.6 (Nov. 1995): 51-53.

Rubstov, C. A.; Romerio, G. F. "Ackermann's Function and New Arithmetical Operations." Nov 2007 <http://forum.wolframscience.com/showthread.php?threadid=1168>.

Sloane, N. J. A. "Online Encyclopedia of Integer Sequences." Nov 2007 <http://www.research.att.com/~njas/sequences/>.

Szekeres, George. "Abel's Equation and Regular Growth: Variations on a Theme by Abel." Experimental Mathematics 7.2 (1998) 85-100.

Szekeres, George. "Regular iteration of real and complex functions." Acta Mathematica 100 (1958) 103-258.

Trappmann, H. (bo198214). "Andrew Robbins' Tetration Extension." Nov 2007 <http://math.eretrandre.org/tetrationforum/showthread.php?tid=3&pid=688#pid688>.

Walker, Peter. "Infinitely Differentiable Generalized Logarithmic and Exponential Functions." Mathematics of Computation 57.196 (Oct. 1991): 723-733.

Wassell, Stephen R. "Superexponentiation and Fixed Points of Exponential and Logarithmic Functions." Mathematics Magazine 73.2 (Apr. 2000): 111-119.

Weisstein, Eric W. CRC Concise Encyclopedia of Mathematics Boca Raton: CRC Press, Chapman & Hall, 2002.

Wolfram, Stephen. A New Kind of Science Champaign: Wolfram Media, 2002.

Woon, S. C. "Analytic Continuation of Operators – Operators acting complex s-times." Nov 2007 <http://arxiv.org/abs/hep-th/9707206>.

Yukalov, V. I.; Gluzman, S. "Weighted Fixed Points in Self-Similar Analysis of Time Series." Nov 2007 <http://arxiv.org/abs/cond-mat/9907422>.