update 4b

Operators

A short review of operators. The discussions about tetration led me to two impressions.

a) It may be better to see operators using 3 parameters, instead of two, as it is common use

b) Searching for another consistent concept for continuous fractional operations, it seemed to require, that somehow the base-parameter for tetration should be thought as "imprinted" in the operator, which, with this "imprint", will then be applied with a fractional iteration-control. So that essentially we do not work with a fucntion of the base-parameter only, but with a function of the operator in connection with the base-parameter. In other words: neither the base-parameter alone nor the abstract operator itself can be fractioned without respect to the other - at least is it so with tetration.

Here I tried to re-generalize this difficult to understand idea also to the common operators addition and multiplication.

The common exponentiation comes out to not to require a special operator at all, so I discuss in fact only three operators and their iterations, instead of four.

Another very nice property of this operator-concept is finally, that it can one-to-one be translated to matrix-operations, where the fractional iterates are then expressed by the fractional powers of matrices, which are typical for a specific operator and modified by the base parameter. See more about this at the appendix.

Basic definition

we have numbers and variable-names, operator-symbols by definition. Later we may define /include the usual function names.

The operators have 3 parameters: start-operand (in example "a"), base-operand (in example "b"), iterator-operand (in example "h").

For convenience we adapt some very basic notations:

1) the unary +/- sign for numbers and variables,
2) the binary "+/-" - sign for numbers and variables
3) the start-operand with "Add"-operator may be omitted, if zero
4) the start-operand with "Mul"-operator may be omitted, if one

"Add"-operator	"Mul"-operator	"Pow"-operator
${}^{h}_{b}\oplus a \overset{def}{\cong} a+(\underbrace{b+b+...+b}_{h-times})$	${}^{h}_{b}\otimes a \overset{def}{\cong} a*(\underbrace{b*b*...*b}_{h-times})$	${}^{h}_{b}\lhd a \overset{def}{\cong} \underbrace{{}^{b...b}b}_{h-times}a$

Allowing the unary minus we have

${}^{-h}_{b}\oplus a \overset{def}{\cong} a-(\underbrace{b+b+...+b}_{h-times})$	${}^{-h}_{b}\otimes a \overset{def}{\cong} a\frac{1}{(\underbrace{b*b*...*b}_{h-times})}$	${}^{-h}_{b}\lhd a \overset{def}{\cong} ??$

The hierarchy of operators occurs most smoothly, if we recurse an expression into the iterator-operand, see below.

The basic idea for developing an expression is:

* we begin with the start-operand as the initial intermediate expression,
* then the operator applies the base-operand with its specific operation
* as many times to the intermediate expression as the iterator-operand dictates.

The operations can be concatenated when one or all of the three operands are replaced by a new instance of an expression. Most interesting here is for the beginning the concatenation of the same type of operation. I didn't consider operator-precedences in details. In a first glance it seems, that by the construction things are automatically in a definite order, but, for instance, I didn't think about concatenation of operators of different type yet.

Some very basic remarks:

$\left.\begin{matrix}{}_{b}^{0}\oplus a\\ {}_{b}^{0}\oplus a\end{matrix}\right\} = a$	${}_{b}^{0}\otimes a = a$	${}_{b}^{0}\triangleleft a = a$
"Start" and "base" can be interchanged, if iterator=1 ${}_{b}^{1}\oplus a = {}_{a}^{1}\oplus b$	"Start" and "base" can be interchanged, if iterator=1 ${}_{b}^{1}\otimes a = {}_{a}^{1}\otimes b$	-not possible- ${}_{b}^{1}\triangleleft a \neq {}_{a}^{1}\triangleleft b$
"iterator" and "base" can be exchanged ${}_{b}^{h}\oplus a = {}_{h}^{b}\oplus a$ Thus also "top" and "down"-iteration can be exchanged	If b<>h: not possible ${}_{b}^{h}\otimes a \neq {}_{h}^{b}\otimes a$	If b<>h: not possible ${}_{b}^{h}\triangleleft a \neq {}_{h}^{b}\triangleleft a$

Horizontal iteration

Syntactically a subsequent expression uses the current expression as its own start-operand.

The order of evaluation is principally from the most elementary position

$${}_a^c\oplus{}_a^d\oplus b = {}_a^{c+d}\oplus b$$ $$\underbrace{a+a+(\overbrace{a+a}^{d-times})b}_{c+d-times} = b+a(c+d)$$	$${}_a^c\otimes{}_a^d\otimes b = {}_a^{c+d}\otimes b$$ $$\underbrace{aaa(\overbrace{aaaa}^{d-times})b}_{c+d-times} = b*a^{c+d}$$	$${}_a^c\triangleleft{}_a^d\triangleleft b = {}_a^{c+d}\triangleleft b$$ $$\underbrace{a^{a^{\overbrace{(a^{a^b})}^{d-times}}}}_{c+d-times} = a^{a^b}$$

Some primitive forms of the expression recursed are expressible in the higher operation, but this cannot define the full range for the higher operators, so this are ***not*** the definitions for the hierarchy:

$$\underbrace{{}_a^1\oplus{}_a^1\oplus\ldots{}_a^1\oplus 0}_{b\,times} = {}_a^{{}_1^b\oplus}\oplus 0 = {}_a^b\oplus 0$$ $$= {}_a^1\otimes b$$	$$\underbrace{{}_a^1\otimes{}_a^1\otimes\ldots{}_a^1\otimes 1}_{b\,times} = {}_a^b\otimes 1$$ $$= {}_a^1\triangleleft b$$	

Left-down-iteration

Replacing the base-operand by a new expression.

There is currently no notation for an operator of this type of iteration. Note, that in effect we create the cyclotomic polynomial by this operation, when applied to the "add"-operator (or in the exponent, when applied to the "Mul"-operator).

$\underbrace{{}^{a}_{c}\oplus_{a}^{a}\oplus c}_{b\,times} \overset{def}{=} {}^{b}_{a}?c$ $= \begin{cases} {}^{b}_{a}\otimes 1 ... +{}^{3}_{a}\otimes 1+{}^{2}_{a}\otimes 1+{}^{1}_{a}\otimes 1+{}^{0}_{a}\otimes 1 \oplus 0 \\ {}^{b}_{a}\otimes c ... +{}^{3}_{a}\otimes c+{}^{2}_{a}\otimes c+{}^{1}_{a}\otimes c+{}^{0}_{a}\otimes c \\ {}^{b}_{a}\otimes 1 \oplus ... {}^{2}_{a}\otimes 1 \oplus {}^{1}_{a}\otimes 1 \oplus {}^{0}_{a}\otimes 1 \oplus 0 \\ {}^{b}_{a}\otimes 1 ... +{}^{3}_{a}\otimes 1+{}^{2}_{a}\otimes 1+{}^{1}_{a}\otimes 1+{}^{0}_{a}\otimes 1 \end{cases} {}^{c}_{1}\otimes c$ $\cong \begin{cases} ((ca+c)a ... +c)a+c \\ (a^{b} ... +a^{3}+a^{2}+a+1)c \\ \frac{a^{b+1}-1}{a-1}c \end{cases}$	$\underbrace{{}^{a}_{c}\otimes_{a}^{a}\otimes c}_{b\,times} = {}^{b}_{a}?c$ $= \begin{cases} {}^{b}_{a}\otimes 1 ... +{}^{2}_{a}\otimes 1+{}^{1}_{a}\otimes 1+{}^{0}_{a}\otimes 1 \otimes 1 \\ {}^{b}_{a}\otimes 1 \otimes ... {}^{2}_{a}\otimes 1 \otimes {}^{1}_{a}\otimes 1 \otimes {}^{0}_{a}\otimes 1 \otimes 1 \\ {}^{-1}_{a-1}\otimes {}^{-1}_{1}\oplus {}^{b}_{a}\otimes a \otimes 1 \end{cases}$ $\cong \begin{cases} c^{a^{0}+a^{1}+a^{2}+a^{3} ... +a^{b}} \\ c^{\frac{a^{b+1}-1}{a-1}} \end{cases}$	
Recursions with primitive expressions		
$\underbrace{{}^{a}_{c}\oplus_{a}^{a}\oplus 0}_{b\,times} = {}^{b}_{a}\otimes c$ $\cong c*a^{b}$ since base- and iteration-parameter are exchangable, this is also valid for left-up-iteration	$\underbrace{{}^{a}_{c}\otimes_{a}^{a}\otimes 1}_{b\,times} = {}^{b}_{a}?c$ $= {}^{{}^{b}_{a}\otimes 1}_{c}\otimes 1$ $\cong c^{a^{b}}$	
	$\underbrace{{}^{1}_{c}\otimes_{1}^{1}\otimes c}_{b\,times} = {}^{b}_{1}?c$ $= {}^{b}_{c}\otimes c$ $\cong c^{b+1}$	

Left-up-iterating

replacing the iterator-operand: this are the rationales for the "hierarchy-of-operator"-definitions

The definitions in this table are not in use:

$\underbrace{{}^{c}_{a}\oplus c\ldots{}^{}_{a}\oplus c}_{b\,times}\oplus c \overset{def}{=} {}^{b}_{a}?\,c$ $\cong \begin{cases} (a^b\ldots+a^2+a+1)c \\ c*\frac{a^b-1}{a-1} \end{cases}$	$\underbrace{{}^{b}_{a}\otimes b\,{}^{}_{a}\otimes b}_{c-times}\otimes b \overset{def}{=} {}^{c}_{a}?\,b$ $\cong \begin{cases} ba^{ba^{ba^{b}}} \\ b\left(a^{\left(a^{a^{b}}\right)^{b}}\right)^{b} \end{cases}$	
$\underbrace{{}^{\cdots}_{a}\oplus c\,{}^{}_{a}\oplus}_{lim\to oo,\lvert a\rvert<1} c = {}^{-1}_{1-a}\otimes c$ $\cong c*\frac{1}{1-a}$		

Primitive forms with start- (or end?) point in the recursion serve as definitions for the operator-hierarchy in the most consistent way:

$\underbrace{{}^{c}_{a}\oplus 0\ldots{}^{}_{a}\oplus 0}_{b\,times}\oplus 0 \overset{def}{=} {}^{b}_{a}\otimes c$
$\cong c*a^b$

$\underbrace{{}^{c}_{a}\otimes 1\,{}^{}_{a}\otimes 1}_{b-times}\otimes 1 \overset{def}{=} {}^{b}_{a}\lhd c$
$\cong a^{a^{a^{c}}}$

Relation to matrix-operators

The operators are one-to-one expressible as matrix-formulae acting on formal powerseries, and the expressions are all extensible to continuous iteration. The matrices contain the coefficients for the powerseries, which are evaluated with the parameter-vector $V(x)$ according to the matrix-multiplication-rules.

The iterator-operand occurs as exponent of the operator-matrix; and since these matrices have either accessible eigensystems or matrix-logarithms, we can use any complex value for the exponent/iterator-operand.

"Add"	*"Mul"*	*"Pow"*
$V(a)\sim * P\sim = V(a+1)\sim$	$V(a)\sim *^{d} V(b) = V(a*b)\sim$	$V(a)\sim * B_b = V(b^a)\sim$
$V(a)\sim * P\sim^{h} = V(a+h*1)\sim$	$V(a)\sim *^{d} V(b)^h = V(a*b^h)\sim$	$V(a)\sim * B_b^{\,h} = V(\{b,a\}\hat{}\hat{}h)\sim$
$V(a)\sim * (P\sim^{b})^h = V(a+h*b)\sim$		
${}_{1}^{h}\oplus a \overset{def}{\cong} a+h*b$	${}_{b}^{h}\otimes a \overset{def}{\cong} a*b^h$	${}_{b}^{h}\triangleleft a \overset{def}{\cong} \{b,a\}\hat{}\hat{}h = {}^{h}b^{a}$

Here the $V(x)$ terms are thought as column-vectors $colvector(x^0,x^1,x^2,...)$, which implements the parameter of the formal powerseries-expression when expanded from the applied matrix-multiplication. The ~-symbol means "transpose". A tiny d-prefix declares this as diagonal-matrix.

"Add": P is the lower triangular matrix of binomial-coefficients (or "Pascal"-matrix). The eigensystem of P is degenerate; but it has an exceptional simple matrix-logarithm, by which then a general power can be easily computed when just multiplied with the h-parameter.

"Mul" is especially simple, since the operator is simply a diagonal-matrix itself and general powers of a diagonal-matrix are defined by just applying the powers to its scalar diagonal-elements.

"Pow" uses the B_b-matrix, as defined in my postings and articles (I usually denote it as B_s-matrix with the parameter s). For the parameter b there is conventionally the range $e^{-e} < b < e^{1/e}$, and for this range a non-degenerate eigen-decomposition could be shown to be valid. The extension for b to the general complex domain is assumed to be possible, but not yet fully established.

However, the eigensystem-decomposition exhibits the relation to the "fixpoint"-concept. Assume the eigensystem-decomposition

$$B_b = W\,D\,W^{-1} \text{ or}$$
$$W^{-1}\,B_b = D\,W^{-1}$$

Now assume (at least) one eigenvalue $d_k = D[k,k] = 1$ ordered to the topmost position in D, so $k=0$ Then using the first row of W^{-1} only we have

$$W^{-1}[0,]\ *\ B_b = 1\ *\ W^{-1}[0,]$$

and the first row in W^{-1} reflects the "fixpoint"-concept, since the rowvector $W^{-1}[0,]$ is invariant under transformation by B_b. The other rows of W^{-1} may be called "pseudo"-fixpoints, since they are only scalar multiples under this transformation (according to the scalar scaling factor d_k.which is the k'th eigenvalue). For the infinite dimensional case we have thus an infinite set of (pseudo)-fixpoints (rowvectors of coefficients for formal powerseries) for the specific operator under consideration and this seems then to be sufficient to uniquely define the mathematical character of such matrix-expressible operators.

Inverse operations

There are two obvious inverses of the "**pow**"- iteration.

Given a constant z, we may ask either for the top-left value, given also the base b, or we may ask for the bottom-left-value, given h.

a) *How many times (possibly fractional) do I have to apply the operator with base b at the starting value 1 until I reach z?*	***b)*** *Which base-operator, h-times repeatedly applied started at 1, leads to my given value z?*
${}^{x}_{b}\triangleleft 1 = z$	${}^{h}_{x}\triangleleft 1 = z$
Example: given z=16, b=2 $\underbrace{2^{2^{\cdots^{2}}}}_{x-times} = 16 \qquad \Rightarrow x = 3$	*Example: given z=16, h=3* $x^{x^{x}} = 16 \qquad \Rightarrow x = 2$
often called ***"slog"***	often called ***"tetra-root"***

The described matrix-operation is best suited for analysis of ***a)***, since most naturally we deal with a fixed base, and discuss the amount of iteration, which is needed to arrive at a certain output starting with a certain input (horizontal start-parameter).

For ***b)*** we currently have only the possibility to find the base by iteratively appling the regula falsi or related procedures for interpolation.

Horizontal concatenation of terms with the same base b is a special simple algebraic operation (addition) on the iterator-parameter, so general fractional iterates of any real h can be reduced to one step of integer-tetration $[h]$ and one step of fractional-tetration with the fractional height-parameter $0<\{h\}<1$.

${}^{h_1}_{b}\triangleleft {}^{h_2}_{b}\triangleleft a = {}^{h_1+h_2}_{b}\triangleleft a$ ${}^{h}_{b}\triangleleft a = {}^{[h]}_{b}\triangleleft {}^{\{h\}}_{b}\triangleleft a$	

In a debate in the internet-newsgroup news://sci.math[1] the position from the view of the tetra-roots were considered: *b^^n^^(1/c) =/= b^^(n/c)*. This problem may be displayed within this scheme as

	c)
	$^{1/c}_{\ {}^{n}_{b}\lhd}\lhd 1 \neq {}^{n/c}_{\ b}\lhd 1$

and continuous tetration was discarded from this observation.

I've currently no good idea about arithmetics in the exponent with different bases, but may be, proper rules can be stated. In the tetration-forum[2] this problem seems to have been adressed in the threads around "base-change", and were mostly posed by Jay Fox. The problem, as stated in

b^^n^^(1/c) =/= b^^(n/c)

in the current view of this article, implicitely involves a base-change, for which I didn't develop smooth rules so far.

But as observed, arithmetical operations of this type in the iterator-parameter can smoothly be described using the *a)*-version (but which, actually, does not fit the problem as stated since it uses a constant, given base-parameter):

$^{1/c}_{\ b}\lhd({}^{n}_{b}\lhd 1) \neq {}^{n/c}_{\ b}\lhd 1$ *but* $^{1/c}_{\ b}\lhd({}^{n}_{b}\lhd 1) = {}^{n+1/c}_{\ b}\lhd 1$ *and* $^{n/c-n}_{\ b}\lhd({}^{n}_{b}\lhd 1) = {}^{n/c}_{\ b}\lhd 1$	

In terms of, for instance, dynamical systems this looks like the following dichotomy:

View of a slog-defender	View of a tetra-root-defender
If I have a basic description of the characteristics *Bs* of a certain system, how many times (possibly fractional times) do I have to apply it to arrive from a starting condition to the final status?	If I look at the starting condition and the final status, which characteristic *Bs* for my system do I need, to arrive at the final status by *x* (possibly fractional) iterations?
If I have iterated the characteristic (*Bs*) of a system 10 times to reach an intermediate status, and then apply it *-10/2* times, then I have the same status, as if I applied it *5* times to the initial status.	I have iterated the characteristic *Bs* of a system *10* times to reach an intermediate status. Then I determine the characteristic *Bt*, which would allow to proceed from the initial condition to the final status in only *2* steps instead. Thus *Bt* should have the meaning of *Bs^^(10/2)*
	But then the characteristic *Bt* is not the characteristic *Bs*. And iterating *Bs 5* times from the initial state is ***not*** the same as iterating *Bt* one time. This is the inherent weakness of continuous tetration.

At the moment I feel not able to make a concluding remark. It is still not clear to me, how the obvious problems with operations, algebraically relating base- and iterator-parameter, can be described and even less, be solved. The conventional binary notation for the tetration-operator suggests, that ***c)***

---

[1] see news://news.t-online.de:119/1190391952.939055@athprx03 or http://groups.google.as/group/sci.math/msg/2a01e281c0263590

[2] see http://math.eretrandre.org/tetrationforum/showthread.php?tid=14&pid=43#pid43

should be an equality. But it seems, this is thus merely a notation-problem. I think, with my scheme here one has tools to point out the core of this problem more precisely than with the common binary-operator ^^ in formulae like $b\hat{}\hat{}h$ and possibly to proceed to an agreement between the concurring views of the formal functionality of the operator.

Some rules

$$ {}^{c}_{a}\otimes({}^{k}_{a}\otimes b) = {}^{{}^{k}_{1}\oplus{}^{c}_{1}\oplus 0}_{a}\otimes b = {}^{c+k}_{a}\otimes b $$

$$ {}^{c}_{a}\otimes({}^{c}_{k}\otimes b) = {}^{c}_{{}^{1}_{k}\otimes{}^{1}_{a}\otimes 1}\otimes b = {}^{c}_{a*k}\otimes b $$

$$ {}^{1}_{a}\otimes 1 = a^{1}*1 $$

$$ {}^{1}_{a}\otimes b = {}^{1}_{b}\otimes a $$

$$ {}^{2}_{a}\otimes b = \left\{ \begin{matrix} {}^{2}_{a}\otimes{}^{1}_{b}\otimes 1 = {}^{1}_{a}\otimes{}_{a*b}{}^{1}\otimes 1 \\ {}^{1}_{a}\otimes{}^{1}_{a}\otimes b = {}^{1}_{a}\otimes{}^{1}_{b}\otimes a \end{matrix} \right\} \neq {}^{2}_{b}\otimes a $$

$$ {}^{a}_{b}\otimes 1 \neq {}^{b}_{a}\otimes 1 \qquad b^{a} \neq a^{b} $$

$$ {}^{c}_{a}\otimes{}^{-c}_{a}\otimes b = {}^{0}_{a}\otimes b = 1b $$

$$ {}^{-c}_{a}\otimes 1 = \frac{1}{a^{c}} $$

$$ {}^{{}^{-c}_{a}\otimes 1}_{a}\otimes 1 = {}_{a}{}^{\frac{1}{a^{c}}} $$

$$ {}^{c}_{a}\otimes{}_{{}^{-1}_{a}\otimes 1}{}^{c}\otimes b = {}_{{}^{1}_{a}\otimes 1}{}^{c}\otimes{}_{{}^{-1}_{a}\otimes 1}{}^{c}\otimes b = {}_{{}^{1}_{a}\otimes{}^{-1}_{a}\otimes 1}{}^{c}\otimes b = {}_{{}^{1-1}_{a}\otimes 1}{}^{c}\otimes b = {}_{{}^{0}_{a}\otimes 1}{}^{c}\otimes b = {}^{c}_{1}\otimes b $$

Iteration downways

$$ {}^{b}_{a}\otimes 1 = {}_{a}b $$

$$ {}^{b}_{a}\otimes 1 = {}_{{}^{b}_{a}\otimes 1}{}^{1}\otimes 1 $$

$$ {}^{{}^{b}_{a}\otimes 1}_{a}\otimes 1 = {}^{{}^{1}_{b}\otimes{}^{1}_{c}\otimes 1}_{a}\otimes 1 = {}_{({}_{a}b)}c = {}_{a}(b*c) $$

$$ {}^{{}^{b}_{a}\otimes 1}_{{}^{b}_{a}\otimes 1}\otimes 1 = {}^{{}^{3}_{b}\otimes 1}_{a}\otimes 1 = {}_{{}_{a}(b}3) $$

Iteration upways

some rules

$$ {}^{2}_{\sqrt{2}}\otimes 1 = {}^{1}_{\sqrt{2}}\triangleleft 2 = 2 $$

$$ {}^{{}^{2}_{\sqrt{2}}\otimes 1}_{\sqrt{2}}\otimes 1 = {}^{2}_{\sqrt{2}}\triangleleft 2 = 2 $$

$$ {}^{{}^{{}^{2}_{\sqrt{2}}\otimes 1}_{\sqrt{2}}\otimes 1}_{\sqrt{2}}\otimes 1 = {}^{3}_{\sqrt{2}}\triangleleft 2 = 2 $$

$$ {}^{oo}_{\sqrt{2}}\triangleleft 2 = 2 $$

Gottfried Helms