EQAN – Introduction to the algorithm involved in equation analysis

EQAN is designed to solve a single or a set of linear equations having unknowns either on RHS or LHS of the "=" sign. The algorithm can be divided into three parts:

- Selecting and rearranging equations to solve the unknown variable
- Solving an equation which contains one instance of the unknown variable
- Solving an equation which contains multiple instances of the unknown variable

Each part is briefly explained in the following sections.

1. Introduction to Equation Analysis

An equation can be considered to be made of two components: the operands which consist of a combination of variables and constants and the operators which act on operands. When a new equation is added to the database the first thing EQAN does is split the equation to its constituent operands and operators. The constants and operators are ignored and variables are added to the variable database which the user references to add and manipulate input data.

Solving an equation in which the unknown variable appears only once (Type 1 equations) and is on the LHS of the "=" sign is relatively simple and straightforward. Consider the example of "a=b+c" where "a" is the unknown variable whose value to be calculated. To obtain this value we simply add the values of "b" and "c".

A layer of complexity is introduced when "a" and "c" are known variables and "b" needs to be calculated. In such cases variables of the equation need to be rearranged and operators modified such that "b" becomes a function of "a" and "c":

$$a=f(b,c) \rightarrow b=f(a,c) : a=b+c \rightarrow b=a-c$$

The variables are rearranged and operators change from one type to another. One will need to know which operators are to be removed and what operator is to be added in its place. This step becomes increasingly complex as the number of operands and operators increase.

This problem becomes even more complex when the unknown variable appears more than once in the equation. This type of problem is divided into two problem sets. In one set the equations can be simplified such as "a=b+b". In this example "b" is a common variable and the equation is simplified to "a=2*b". Now the equation is resolved to a type 1 equation which is solved by rearrangement of variables and change of operations.

$$a=b+b \rightarrow a=b*(1+1) \rightarrow a=2*b \rightarrow b=a/2$$

In another set the equations cannot be simplified and rearranged. Quadratic equations such as "$a=b^2+b$" and other equations where the unknown is a part of a mathematical function such as "a=b+log(b)" fall into this category. EQAN solves such equations by using the bisection method which is one of several iterative ways of solving an equation.

When the equation which contains the unknown variable to be calculated consists of several unknowns and these unknowns are to be calculated from other equations EQAN collects and solves these equations staring with the equation containing least number of unknowns and working up to the equation which contains the variable whose value is needed. Consider the first example "a=b+c" where value of variable "a" is to be found but "b" is also an unknown variable. However variable "b" is a function of variables "d" & "e" i.e. "b=d+e". Therefore the solution to this problem is found by solving equations in following order: "b=d+e” and then “a=b+c”. This concept is explained in greater detail in the following sections.

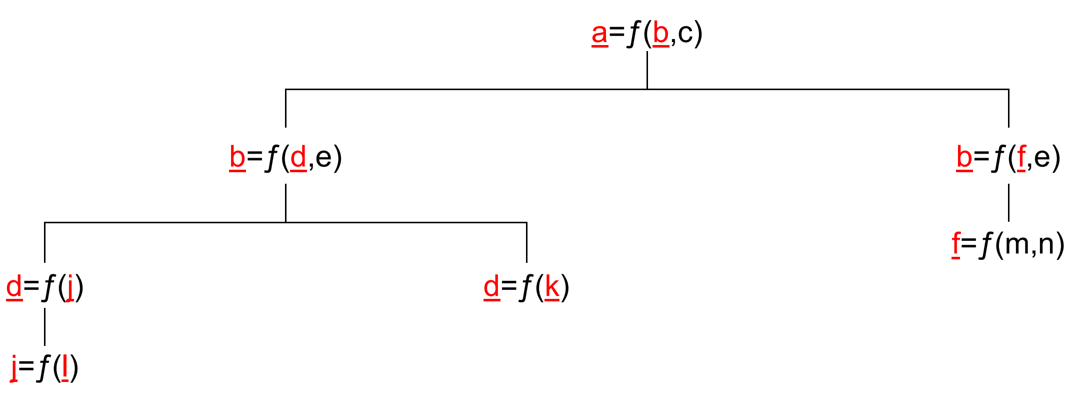
The above ideas laid out covers the basics of equation analysis. EQAN combines these ideas to solve a set or a single equation.

2. Selecting and arranging equations in an equation set

When the unknown variable whose value is to be calculated is a part of an equation which consist of several unique unknown variables whose values can be calculated from other unique equations it becomes necessary to collect and arrange these equations in ascending order of unknowns. EQAN identifies three categories of equations: equations which can be solved, equations which cannot be solved and equations which have the potential to be solved. Equations which can be solved consist of only one unknown irrespective of the number of times the unknown appears in the equation. For example in “a=b+c”, if any two variables are known than the third can be calculated. Equations are considered unsolvable when it contains more than one unique unknown variable and these unknowns cannot be solved from other equations. In the above example if there are more than one unknown (say both “b” & “c” values are not known) and there is no other equation containing these variables then the equation is considered unsolvable. Equations which consist of several unique unknowns and the value of these unknowns can be found by solving other equations are considered as equations which have the potential to be solved. Considering equation “a=b+c” again, if the value of “a” is known but “b” and “c” values are unknown and “b” is linked to another equation “b=d+e” then the equation “a=b+c” has the potential to be solved.

Solving a set of equations to obtain the value of an unknown variable can be better understood from the following illustrations (unknown variables are marked in red and underlined):

Example 1:



The unknown variable whose value is to be calculated is “a” and is a function of unknown variable “b” and known variable “c”. The equation has a potential to be solved since variable “b” value can be found from two other equations (b=ƒ(d,e) & b=ƒ(f,e)). Variable “d” value is also unknown and can be solved by either using unknown variables “j” or “k”. Since the equation which contains the variable “k” cannot be solved it is discarded.

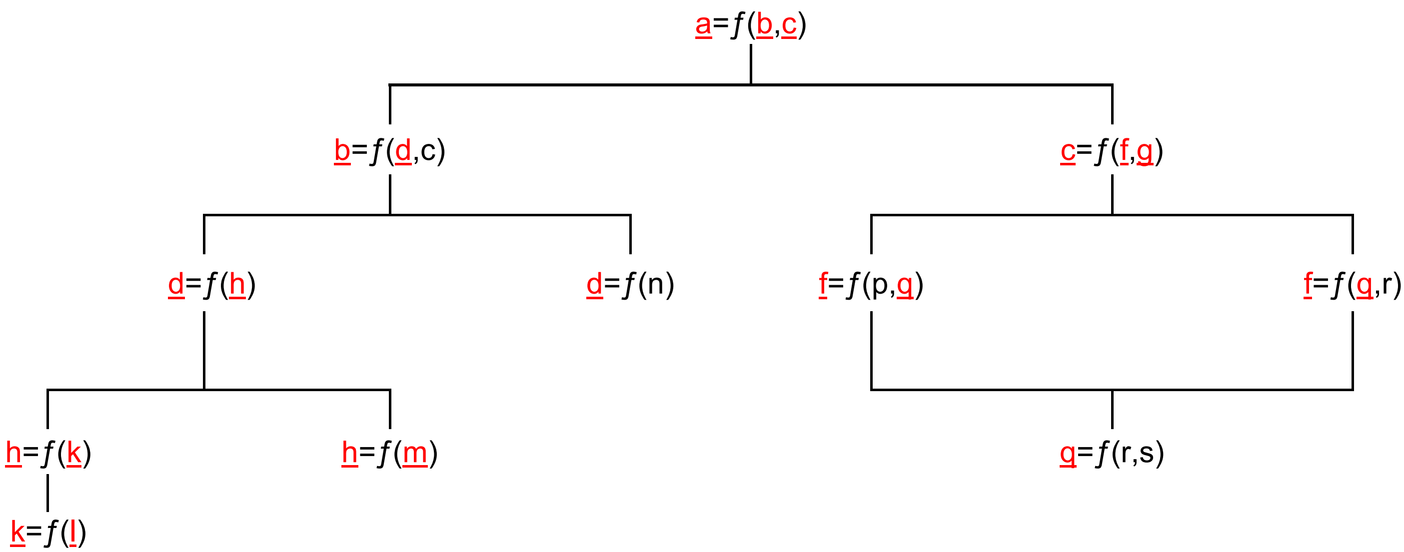
Now two set of solutions are available to EQAN in the following order:

1. j=ƒ(l) → d=ƒ(j) → b=ƒ(d,e) → a=ƒ(b,c)

2. f=ƒ(m,n) → b=ƒ(f,e) → a=ƒ(b,c)

EQAN chooses the equation set which requires the least number of steps to reach the solution. In this example equation set 2 is selected as the number of steps is 3 versus 4 steps in equation set 1.

Example 2:



EQAN uses the following rules to calculate the unknown variable “a”:

1. A one dimensional array (variable list) is created which stores unknown variables other than the variable whose value is to be calculated.
2. Each variable of the variable list is considered individually and equations containing the variable which are solvable or have the potential to be solved are added to a two dimensional array (solution set).
3. If a variable value can be solved from several equations then the equation set containing this variable is duplicated by a number equal to the number of equations and each equation is added to the end of each set.
4. The unknowns of potentially solvable equations are added to the end of variable list and the variable based on which new equations were added to the solution set is removed from the list.
5. If the variable of a potentially solvable equation value cannot be solved than all equation sets containing that variable are removed from the solution set.
6. Once all variables of the variable list have been considered and if the solution set contains several equation sets EQAN will select the equation set with the least number of equations (i.e. the shortest path to the solution)

Using the above outlined rules to solve example 2 the solution will look something like this:

$a=f(b,c)$	
$\downarrow$ b	(b,c)
$b=f(d,c)$	$\downarrow$
$\downarrow$ c	(c,d)
$b=f(d,c),\ c=f(f,g)$	$\downarrow$
$\downarrow$ d	(d,f)
$b=f(d,c),\ c=f(f,g),\ d=f(h)$ $b=f(d,c),\ c=f(f,g),\ d=f(n)$	$\downarrow$
$\downarrow$ f	(f,h)
$b=f(d,c),\ c=f(f,g),\ d=f(h),\ f=f(p,q)$ $b=f(d,c),\ c=f(f,g),\ d=f(h),\ f=f(q,r)$ $b=f(d,c),\ c=f(f,g),\ d=f(n),\ f=f(p,q)$ $b=f(d,c),\ c=f(f,g),\ d=f(n),\ f=f(q,r)$	$\downarrow$
$\downarrow$ h	(h,q)
$b=f(d,c),\ c=f(f,g),\ d=f(h),\ f=f(p,q),\ h=f(k)$ $b=f(d,c),\ c=f(f,g),\ d=f(h),\ f=f(q,r),\ h=f(k)$ $b=f(d,c),\ c=f(f,g),\ d=f(n),\ f=f(p,q)$ $b=f(d,c),\ c=f(f,g),\ d=f(n),\ f=f(q,r)$	$\downarrow$
$\downarrow$ q	(q,k)
$b=f(d,c),\ c=f(f,g),\ d=f(h),\ f=f(p,q),\ h=f(k),\ q=f(r,s)$ $b=f(d,c),\ c=f(f,g),\ d=f(h),\ f=f(q,r),\ h=f(k),\ q=f(r,s)$ $b=f(d,c),\ c=f(f,g),\ d=f(n),\ f=f(p,q),\ q=f(r,s)$ $b=f(d,c),\ c=f(f,g),\ d=f(n),\ f=f(q,r),\ q=f(r,s)$	$\downarrow$
$\downarrow$ k	(k)
$\cancel{b=f(d,c),\ c=f(f,g),\ d=f(h),\ f=f(p,q),\ h=f(k),\ q=f(r,s)}$ $\cancel{b=f(d,c),\ c=f(f,g),\ d=f(h),\ f=f(q,r),\ h=f(k),\ q=f(r,s)}$ $b=f(d,c),\ c=f(f,g),\ d=f(n),\ f=f(p,q),\ q=f(r,s)$ $b=f(d,c),\ c=f(f,g),\ d=f(n),\ f=f(q,r),\ q=f(r,s)$	
$\downarrow$	
$b=f(d,c),\ c=f(f,g),\ d=f(n),\ f=f(p,q),\ q=f(r,s)$ $b=f(d,c),\ c=f(f,g),\ d=f(n),\ f=f(q,r),\ q=f(r,s)$	
$\downarrow$	
$b=f(d,c),\ c=f(f,g),\ d=f(n),\ f=f(p,q),\ q=f(r,s)$	
$\downarrow$	
$a=f(b,c),\ b=f(d,c),\ c=f(f,g),\ d=f(n),\ f=f(p,q),\ q=f(r,s)$	

During execution of equations EQAN will solve in the following order:

$q=f(r,s) \rightarrow f=f(p,q) \rightarrow d=f(n) \rightarrow c=f(f,g) \rightarrow b=f(d,e) \rightarrow a=f(b,c)$

3. Compressed Equations

In order to get an idea of how EQAN solves an equation it's important to understand the concept of what I call "compressed equations". A compressed equation is one in which several types of operators of an equation are reduced to a single operation. An equation consisting of a combination of addition and multiplication is reduced to only multiplication or addition. This can be better understood with an example: consider the equation "a=(b*c)+d". To compress this equation the part of equation in-between brackets is replaced with a temporary variable say, "ž0". Now the equation becomes "a= ž0+d" where ž0="(b*c)". The concepts laid out in this example can be expanded to most equations:

1. a=(b/c)+(d*(e+g)) → a=ž0+ž1 : ž0=(b/c) : ž1=(d*(e+g))
2. a=SIN(b+(c^d)) → a=SINž0 : ž0=(b+(c^d))
3. a=(SIN(b+d))+(COS(e+f)) → a=ž0+ž1 : ž0=(SIN(b+d)) : ž1=(COS(e+f))

Equations containing brackets are compressed. An equation such as "a=b/c" when compressed will result in "a=b/c" but equations such as "a=SIN(b)" when compressed yields "a=SINž0" where ž0=(b). Note that a fully compressed equation is one which contains one kind of operation and does not have any brackets. The importance of compressed equations will become apparent in the following pages.

4. Adding and Removing Brackets

The first thing EQAN does when solving an equation is adding and removing brackets wherever necessary. Cases where the brackets are removed:

1. When the opening bracket is located immediately after the "=" sign and corresponding closing bracket is at the end of the equation. Example: a=(b+c) → a=b+c
2. Cases where variables are located in-between brackets unnecessarily.
   Example: a=(b)+c → a=b+c

The rules involved in adding of brackets are slightly more complex. EQAN keeps adding brackets till equation compression yields a fully compressed equation which contains a single operation and no brackets. EQAN adds brackets in the following logical order:

- Open brackets or variables proceeded by a minus sign and the minus sign is preceded by operators such as "*", "/", "+", "SIN", etc. Examples :
  - a=b*-c → a=b*(-c)
  - a=b*-(SIN(d+e)) → a=b*(-(SIN(d+e)))
- Variables preceded by a mathematical function such as "SIN", "LOG", "LN","SQRT" etc. Examples:
  - a=SINb+c → a=SIN(b)+c
  - a=b*SQRTc → a=b*SQRT(c)
- After steps 1 & 2 brackets are added till equation compression yields a purely compressed equation. They are added in the following order :
  a) Mathematical functions: SIN, COS, TAN, ASIN, ACOS, ATAN, SQRT, LOG, LN and EXP.
  b) Exponentiation operation involving "^" operator.
  c) Division operation with the "/" operator.
  d) Multiplication operation using "*" operator

Consider a few examples:

1) a=b/c^SINd → a=b/c^SIN(d) → a=b/c^(SIN(d)) → a=b/(c^(SIN(d)))
2) a=SIN(b-c/d+e^f) → a=SIN(b-c/d+(e^f)) → a=SIN(b-(c/d)+(e^f))
3) a=SIN(b*c)+COS(d) → a=(SIN(b*c))+COS(d) → a=(SIN(b*c))+(COS(d))
4) a=b^d+LNe → a=b^d+LN(e) → a=b^d+(LN(e)) → a=(b^d)+(LN(e))
5) a=b*c*d/e → a=b*c*(d/e)
6) a=b/c/d → a=(b/c)/d
7) a=b^c^d → a=(b^c)^d

5. Solving equations with single instance of unknown variable

Once the concept of compressed equations become clear it's easier to understand how EQAN rearranges variables and modifies operands to solve for unknown variables on the RHS of the "=" sign. When the unknown variable is a part of addition the operation changes to subtraction of known variables i.e. consider the following equation where "b" is unknown a=b+c → b=a-c. Thus "+" → "-" similarly:

- a=b*c*d → b=a/(c*d)
- a=b/c
  - b is unknown → b=a*c
  - c is unknown → c=b/c
- a=b^c
  - b is unknown → b=a^(1/c)
  - c is unknown → c=(LOG(a))/(LOG(b))
- a=LOG(b) → b=10^a
- a=LN(b) → b=2.71828^a
- a=EXP(b) → b=(LOG(a))/(LOG(2.71828))
- a=SQRT(b) → b=a^2
- Trigonometric functions :
  - a=SIN(b) ↔ b=ASIN(a)
  - a=COS(b) ↔ b=ACOS(a)
  - a=TAN(b) ↔ b=ATAN(a)

When an equation consists of several unique types of operators, the equation is compressed so it reduces to one kind of operation. The variables are then rearranged and operation is modified. The equation is decompressed and the above process continues till the unknown variables are on the LHS of the "=" sign. This concept can be better understood with an example. Consider the equation "a=SIN(b^c)" in which variable "c" is unknown. Following the above procedure the steps EQAN follows to obtain the solution will look like this:

a=SIN(b^c) → a=SINž0 → ž0=(ASIN(a)) → (b^c) =(ASIN(a)) → b^c =(ASIN(a)) → b^c = ž0 → c=(LOG(ž0))/(LOG(b)) → c=(LOG(ASIN(a)))/(LOG(b))

Consider a few more examples:

- a=-(b*(SQRT(d/e))) : d is unknown → a=-ž0 → ž0=(-a) → b*(SQRT(d/e))=(-a) → b*ž0= ž1 → ž0=ž1/b → SQRT(d/e)=((-a)/b) → SQRT ž0=ž1 → ž0=ž1^2 → d/e=(((-a)/b)^2) → d/e=ž0 → d=ž0*e → d=(((-a)/b)^2)*e

- a=(b*c)-(LOG(d/e)) : e is unknown → a=ž0-ž1 → -ž1=a-ž0 → ž1=(-(a-ž0)) → LOG(d/e)=(-(a-(b*c))) → LOGž0= ž1 → ž0=10^ž1 → d/e=(10^(-(a-(b*c)))) → d/e=ž0 → e=d/ž0 → e=d/(10^(-(a-(b*c))))

6. Solving equations with multiple instances of the unknown variable using bisection method

When an equation cannot be simplified so that its unknown variable appears only once then EQAN uses bisection method to solve for the unknown variable. Bisection method is an iterative method which finds the solution to an equation by the occurrence of sign change when substituting values for the unknown variable. Consider an equation a=ƒ(b). The first step to solving this equation is finding the values of b1 and b2 such that a change in sign occurs when substituting these values in the equation i.e. ƒ(b1) results in a negative value and ƒ(b2) results in a positive value. This indicates the solution lies between b1 and b2. EQAN now follows an iterative process in which b3=(b1+b2)/2 is calculated and if the sign of ƒ(b3) is not same as ƒ(b2) then b1=b3 or else if the sign are same then b2=b3. This process is repeated till two values of b3 are equal to a certain degree of accuracy.

Examples of bisection method (even simple ones) are usually long and it's better if the reader refers to some mathematical textbooks for examples on the bisection method. I refereed "Higher Engineering Mathematics" by John Bird while developing EQAN. Wikipedia is another great source to learn more about bisection method.

7. Solving equations with multiple instances of the unknown variable by simplification

Before solving an equation with multiple instance of unknown using the bisection method EQAN simplifies the equation and checks whether the equation can be simplified to the point that the unknown variable appears only once in the equation. In such a case the solution is found by rearrangement of variables and change of operands. The first step of this process is replacing all known variables with their respective values. After this mathematical operation involving known variables are executed. Unless the variable on the LHS of the "=" sign is an unknown variable its location is unaltered. If unknown variable is on the LHS of the "=" sign it is brought to the RHS side of the equation. Let's understand these concepts using some examples:

- a=b+(c*(b+(d/e))) : a=4 : b = "x" : c=3 : d=8 : e=2 → a=b+(3*(b+(8/2))) → a=b+(3*(b+4)) → a=b+((b+4)*3)
- a=(b+(SIN(60+c))/b : a=3 : b="x" : c=30 → a=(b+(SIN(60+30))/b → a=(b+(SIN(90))/b → a=(b+1)/b
- a=5-(2*LOG(b))+(c*(LOG(b))*d) : a=5 : b="x" : c=5 : d=2 → a=5-((LOG(b))*2)+(5*(LOG(b))*2) → a=5-((LOG(b))*2)+(5*(LOG(b))*2) → a=5-(LOG(b)*2)+((LOG(b))*10)
- b=(LOG(c+(d*e)))*b : b="x" : c=80 : d=10 :e=2 → b=(LOG(80+(10*2)))*b→ 0=((LOG(80+(10*2)))*b)-b → 0=((LOG(80+20))*b)-b → 0=((LOG(100))*b)-b → 0=(2*b)-b → 0=(b*2)-b

The reader may have noticed from the above examples that EQAN analyses the equation in parts. Variables and operands in-between brackets are examined. If the variables in-between the brackets are known then the corresponding operation is carried out and they (the variables and operators) are replaced by the result e.g. “(SIN(60+30)) → (SIN(90)) → 1”. However if there are unknown variables between brackets and the operation is of addition/subtraction or multiplication then the known variables are shifted to the right e.g. “(SIN(3*b)) → (SIN(b*3))”, “2*(3+b) → 2*(b+3) → (b+3)*2”, “(-3-b)/7 → (-b-3)/7”.

After simplifying the equation EQAN identifies whether the unknown variable is associated with any operation and this combination is uniform across the equation.
For example:

- (b*3)+(b*3)+6 → (b*3) is the common part
- ((COS(b))*3)+(2/(COS(b))) → (COS(b)) is the common part
- (LOG(b/2))/((b/2)+3) → (b/2) is the common part
- (SIN(b))+(COS(b)) → b is the common part

EQAN identifies the common part by collecting all parts of the equation containing the unknown variable and checks if all these parts are the same. If not then EQAN identifies the shortest part and checks if it is contained within other parts of the equation. If two parts are of the same size then both are reduced and compared. Consider a few examples:

- a=(b*3)+((b*3)/2) : [(b*3), ((b*3)/2)] : (b*3) ≠ ((b*3)/2) | (b*3) is in ((b*3)/2) □ (b*3) if the common part of the equation
- a=(COS(b))+(SIN(b)) : [(COS(b)), (SIN(b))] : (COS(b)) ≠ (SIN(b)) | (COS(b)) → (b) | (SIN(b)) → (b) | (b) ≡ (b) □ (b) is the common part of the equation
- a=(COS(b))+((SIN(b))/2) : [(COS(b)), ((SIN(b))/2)] : (COS(b)) ≠ ((SIN(b))/2) | (COS(b)) → (b) | (b) is contained in ((SIN(b))/2) □ (b) is the common part of the equation

After identifying the common part EQAN creates a temporary equation in which common parts of the equation are replaced by a temporary variable say “α”. For example:

- a=(b*3)+((b*3)/2) : (b*3) is the common part → a= α+(α/2)
- a=(COS(b))+(SIN(b)) : (b) is the common part → a=(COSα)+(SINα)
- COS(b+(b*3)) : b is the common part → COS(α+(α*3))

After creating a temporary equation EQAN tries to determine whether this equation can be further simplified. EQAN does this by ensuring that parts of the equation containing the unknown variable consist of only multiplication and division operations. If there are other operators such as “LOG”, “SIN”, “^”, etc. then the equation cannot be further simplified. Also in the case of division all instances of the unknown variable should be either on the denominator or numerator of the operation. Consider a few examples:

- a=(b*3)+((b*3)/2)+4 : a= α+(α/2)+4 → check=True
- a=b+(4/b) : a=α+(4/α) → check=False : all instances of the unknown variable not a part of denominator of the division operation
- a=(1/(b^2))+(3/(b^2)) : a=(1/α)+(4/α) → check=True
- a=(COS(b))+((COS(b))^2)+3 : a=α+(α^2)+3 → check=False : presence of “^” sign in part of the equation containing the common part
- a=(LOG(b))-b : a=(LOG (α))-α → check=False : one of the unknown variable is a part of LOG operation

If the equation cannot be further simplified then EQAN solves the equation using the bisection method. On the other hand if it can be simplified EQAN proceeds to do so by collecting all the common parts and replacing it with “1”. These parts are placed between brackets and multiplied by the common part. This can be better understood with some examples:

- a=(b*3)-((b*3)/2)+4 : a=α+(α/2)+4 : α=(b*3) → a=((b*3)*(1-(1/2)))+4
- a=SIN((4/b)+(1/(2*b))) : a=SIN((4/α)+(1/(2*α))) : α=b → a=SIN((1/b)*(4+(1/2)))

Now the number of instances of the unknown is reduced to 1. EQAN then solves the equation using the rules and methods for the occurrence of single unknown.

8. Simplifying an equation containing multiple instance of the unknown variable with respect to exponentiation, division and multiplication

When simplifying an equation consisting of several instances of the unknown variable, EQAN takes into consideration that the equation can also be simplified with respect to exponentiation, division and multiplication to obtain an equation consisting of single instance of the unknown variable. Consider an example: a=(b+1)/b → a=1+(1/b). However simplification will occur even if the end equation contains multiple instances of the unknown. Consider another example: a=b*(b+1) → a=(b^2)+b. Let’s consider each operation one by one.

a) Simplification with respect to exponentiation

This operation occurs when part of an equation consist of operations of division, multiplication or exponentiation and is followed by an “^” operator and a number. For example: a=b+((b/3)^2). In this equation the part “((b/3)^2)” can be simplified to “((b^2)/9)” and the equation becomes a=b+((b^2)/9). If the variables in-between the brackets consist of an exponentiation operation but the exponent is a non-numeric variable then EQAN simplifies the operation to multiplication of the exponents. Consider “(2^(COS(b)))^3” is simplified to “(2^(3*(COS(b)))”. Consider a few more examples:

- a=b+((9/b)^2) → a=b+(81/(b^2))
- a=b+((b*2)^4) → a=b+(b^16)
- a=b+((5^b)^3) → a=b+(5^(b*3))

b) Simplification with respect to division

After simplifying the equation w.r.t exponentiation EQAN checks if the equation can be simplified w.r.t division and then multiplication. EQAN first checks that the divisor satisfies the conditions for division. The divisor operands should be composed of the unknown variable only or the operands should be a combination of “*”, “/” and “^” or a compressed equation should yield exponentiation operation. For example EQAN does not consider “(b+2)” as a divisor during simplification but “(b+2)^4” can be considered a divisor as compression will yield ž0^4.

After identifying the divisor EQAN checks for division. The divisor should be preceded by division symbol (“/”) and dividend. The dividend may be the unknown variable itself or an operation involving the unknown variable. EQAN compresses the dividend and if the common operator is "+", "-", "/", "*" or "^" then division operation will take place. EQAN performs the division operation in stages. In cases of addition/subtraction and multiplication EQAN separates the divisible and non-divisible and add "/" and divisor to the end of each. For example: "a=(b+1)/(b^2) → a=(b/(b^2))+(1/(b^2))". In cases where the common operator is "/" EQAN shifts the divisor to the numerator of the operation. For example: "((2^b)/5)/(2^(COS(b))) → ((2^b)/(2^(COS(b)))/5". If the common operator is "^" and numerator and denominator consist of numeric exponent or the numerator is the unknown variable then the power of bases of denominator and numerator are reduced accordingly. For example: b/((b^2)*3) → 1/(b*3), (b^5)/(b^2) → (b^3). However if the common operator is "^" and non-numeric exponent are present EQAN checks the base of the numerator and denominator are same. If so simplification is done by applying the law of indices and subtracting numerator and denominator exponents. For example: a=(2^b)/(2^(b+3)) → a=(2^(b-(b+3))). This process continues till further simplification by division is not possible. Consider a few more examples:

- a=((b^3)+(b^2)-4)/(3*(b/2)) → a=(((b^3)/b)+((b^2)/b)-(4/b))/(3*(1/2)) → a=((b^2)+((b^2)/b)-(4/b))/(3*(1/2)) → a=((b^2)+b-(4/b))/(3*(1/2))
- a=(3+(2^(b*3)))/(2^(COS(b))) → a=((2^(b*3))/(2^(COS(b))))+(3/(2^(COS(b)))) → a=(2^((b*3)-(COS(b))))+(3/(2^(COS(b))))
- a=(3-(b^2))/((b^5)/2) → a=(-((b^2)/(b^5)))+(3/((b^5)/2)) → a=(-(1/(b^3)))+(3/((b^5)/2))
- a=((b^3)*5)/(b/2) → a=(((b^3)/b)*5)/(1/2) → a=((b^2)*5)/(1/2)

c) Simplification with respect to multiplication

The rules for identifying the multiplier are same as the rules involved in identifying the divisor in division operation mentioned earlier. The multiplier should be followed by or preceded by "*" and multiplicand.

The rules EQAN follows while simplifying the equation with respect to multiplication are same as those involved in simplifying the equation with respect to division, however as per the laws of indices when exponentiation operation is involved then the powers of bases are added i.e. (b^2)*(b*3) → (b^3)*3. Consider a few examples:

- a=((b^3)+(b^2)-4)*(b/2) → a=(((b^3)*b)+((b^2)*b)-4)*(1/2) → a=((b^4)+((b^2)*b)-4)*(1/2) → a=((b^4)+(b^3)-4)*(1/2)
- a=(b^2)*((b+2)*3) → a=((b^2)*(b+2))*3 → a=(((b^2)*b)+((b^2)*2))*3 → a=((b^3)+((b^2)*2))*3
- a=(3^b)*(3^(b+2)) → a=3^((b+2)+b) → a=3^((b+2)+b) → a=3^(b+2+b)
- a=(b^2)*((b*3)/2) → a=((b^2)*(b*3))/2 → a=(((b^2)*b)*3)/2 → a=((b^3)*3)/2

9. CONCLUSION

Solutions to real world problems rarely depend on solving equations alone. Generally graphs, diagrams, matrices, vectors, loops, conditional statements, etc. are involved. In such cases the algorithm will be more complex, though I cannot stress enough that such an algorithm which interrelates several mathematical concepts and uses them to obtain the optimum solution to a complex problem is feasible. I think EQAN is the first step to a more user friendly calculator with application in many diverse fields. Any suggestions, remarks, queries, criticism, etc are most welcome.