Language for Array Data Processing

**A Capstone Project Proposal for UHCL CS MS Students – Spring 2010**

**Written by Ravi Ganta, Ansh Labs LLC**

## Aim

To design and develop:

1. A simple **domain specific language** for accessing, transforming, and manipulating arrays of data in a user-friendly manner
2. An **interpreter program** that performs lexical analysis, parsing, and executes statements specified in the language mentioned above
3. A **library** of arithmetic, logical, and statistical functions that are used by the interpreter to execute statements and generate results

## Description

* User is required to manipulate data available in a 2D matrix.
* Data is addressable in multiple ways
	+ By absolute location such as (1, 3) or (2, 8), etc.
	+ By pseudo location such as A1, B4 where A represents rows and the digit that follows represents columns
	+ By group membership – here data in the matrix can also be part of various named groups such as G01, G10, etc.
	+ By other domain concepts – here data can be referenced by certain domain specific concepts such as Standards, Controls, Samples, Blanks, etc.
* Data once accessed can be part of expressions which are either logical (resulting in true or false evaluations) or arithmetic expressions (resulting in real number evaluations) or statistical expressions (resulting in evaluation of average, or median, or SD, or CV%, etc.)

## List of Arithmetic Operations

* Plus, Minus, Multiply, Divide, Modulo
* Parenthesis

## List of Binary Operations

* And
* Or
* Nor
* Xor

## List of Comparison Operations

* >
* <
* ==
* >=
* <=

## List of Functions

* Average()
* Min()
* Max()
* Median()
* Standard Deviation()
* %CV
* Sort()
* Log10()
* LogE()
* Exp()
* X^y(a,b)
* Sin()
* Cos()
* Tan()

## Expressions

* An expression is a statement that contains a string of operands, operators, and / or functions arranged such that it conforms to the grammar of the language, and thus can be interpreted and executed to obtain a result
* An expression can also be a command to the run-time interpreter (not involving arithmetic expressions)
* An expression is any valid statement written in target language
* A statement is a string of characters written using the alphabet of a language and by following the grammar of the language

## Conditional Execution

* A conditional statement such as
	+ if (Boolean expression) then {} else {}

is required to perform operations based on the result of a Boolean expression.

## Examples

>> M1[A1] // return the value of the matrix M1 at location A1

50.0 // this is the returned value

>> M1[S01] // return all the values in matrix M1 whose name tag is S01

12.3, 13.5 // assuming there are two such values, both will be printed out

>> Avg(M1[S01]) // returns the average of the above values

12.9

>> M1 = M1-Avg(M1[S01]) // Subtract all values contained in M1 by the average of the value of

// group S01

-0.6, 0.6,…..

1.2, 5.3,…..

## Example Scripting Environments

* Cmd shell is a similar scripting environment where user issues commands and the system responds
* Mathematica Console is a similar scripting environment where user issues commands and the run-time responds by solving equations or systems or graphing