Documentation

There are three types of documentation available for OpenMx, an Official User's Guide, an Official Reference Manual, and the OpenSem Wiki. The two official documents are maintained by the OpenMx development team. The Wiki is a community document that can be contributed to and edited by anyone registered on the OpenMx website. Open registration will begin when the open beta is released this fall.

Official Documentation (Latest Release, v 0.2.2-951)

- **User Guide** (html, pdf) -- The User Guide provides a tutorial introduction to using OpenMx. It is split into three parts:
  - A Quick-Start tutorial comprised of two introductory chapters that are to be read by everyone
  - A more advanced set of examples for those who think in terms of path model
  - A more advanced set of examples for those who write their models as matrix formulae.

- **Reference Manual** (html, pdf) -- The Reference Manual comprises the help files that describe each of the OpenMx R functions. These help files are also available from the R command line by typing a question mark followed by the function name. For example, `?mxModel` will bring up the help page for the mxModel function.

OpenSEM Wiki
Welcome to OpenMx’s documentation!

Contents:

- Introduction
  - Beginners Guide to OpenMx
    - Pass By Value
    - Path Model Specification
    - Matrix Model Specification
  - Quick Overview
    - Simple OpenMx Script
    - Optimization Script
    - More in-depth Example
  - Two Model Styles - Two Data Styles
    - Univariate Saturated Model
    - Covariance Matrices and Path-style Input
    - Raw Data and Path-style Input
    - Covariance Matrices and Matrix-style Input
    - Raw Data and Matrix-style Input
    - Bivariate Saturated Model
Pass-by-Value

- variables
- values stored in variables
- the only way to update a variable in a function call is to capture the result of the function call
Functions vs Classes

**function**
- mxModel
- mxMatrix
- mxPath
- mxAlgebra
- mxBounds
- mxConstraint
- mxData

**class**
- MxModel
- MxMatrix
- MxPath
- MxAlgebra
- MxBounds
- MxConstraint
- MxData

Functions create objects of specific class
Matrix Algebra

\[ A = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} \quad B = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} \]

\[ q_1 = A + B \]
\[ q_2 = A.A \]
\[ q_3 = t(A) \]
\[ q_4 = A \ast t(A) \]
\[ q_5 = t(A) \ast A \]
#NGroups 1

Title: Matrix Algebra
Calculation
Begin Matrices;
  A Full 3 1
  B Full 3 1
End Matrices;
  Matrix A 1 2 3
  Matrix B 1 2 3
Begin Algebra;
  C = A+B;
  D = A*B;
  E = A';
  F = A*A';
  G = A'*A;
End Algebra;
End
OpenMx Script AI

```
algebraModel <- mxModel(
  mxMatrix(
    type="Full",
    nrow=3,
    ncol=1,
    values=c(1,2,3),
    name='A'
  ),
  mxMatrix(
    type="Full",
    nrow=3,
    ncol=1,
    values=c(1,2,3),
    name='B'
  ),
  mxAlgebra(
    A + B,
    name='q1' # addition
  ),
)`
```
OpenMx Script All

mxAlgebra(
  A * A,
  name='q2'
) # dot multiplication

mxAlgebra(
  t(A),
  name='q3'
) # transpose

mxAlgebra(
  A %*% t(A),
  name='q4'
) # inner product

mxAlgebra(
  t(A) %*% A,
  name='q5'
) # outer product

)
OpenMx script B

```r
algebraModel <- mxModel(
  mxMatrix(type="Full", nrow=3, ncol=1, values=c(1,2,3), name='A'),
  mxMatrix(type="Full", nrow=3, ncol=1, values=c(1,2,3), name='B'),
  mxAlgebra( A + B, name='q1' # addition ),
  mxAlgebra( A * A, name='q2' # dot multiplication ),
  mxAlgebra( t(A), name='q3' # transpose ),
  mxAlgebra( A %*% t(A), name='q4' # inner product ),
  mxAlgebra( t(A) %*% A, name='q5' # outer product )
)

algebraFit <- mxRun(algebraModel)
algebraFit@algebras
result <- mxEval(list(q1,q2,q3,q4,q5),algebraFit)
```

Sunday, February 7, 2010
algebraModel <- mxModel(
  mxMatrix( "Full", 3, 1, c(1,2,3), 'A'),
  mxMatrix( "Full", 3, 1, c(1,2,3),'B'),
  mxAlgebra( A + B, 'q1' # addition ),
  mxAlgebra( A * A, 'q2' # dot multiplication ),
  mxAlgebra( t(A), 'q3' # transpose ),
  mxAlgebra( A %*% t(A), 'q4' # inner product ),
  mxAlgebra( t(A) %*% A, 'q5' # outer product )
)

algebraFit <- mxRun(algebraModel)
algebraFit@algebras
result <- mxEval(list(q1,q2,q3,q4,q5),algebraFit)
Mx vs OpenMx

#NGroups 1
G1 MatrixAlgebra Calculation
Begin Matrices;
  A Full 3 1
  B Full 3 1
End Matrices;
Matrix A 1 2 3
Matrix B 1 2 3
Begin Algebra;
  C= A+B;
  D= A.B;
  E= A';
  F= A'*A;
  G= A'*A;
End Algebra;
End

algebraExercises <- mxModel(
  mxMatrix( "Full", 3, 1, c(1,2,3), 'A'),
  mxMatrix( "Full", 3, 1, c(1,2,3),'B'),
  mxAlgebra( A + B, 'q1' ),
  mxAlgebra( A * A, 'q2' ),
  mxAlgebra( t(A), 'q3' ),
  mxAlgebra( A %*% t(A), 'q4' ),
  mxAlgebra( t(A) %*% A, 'q5' )
)
OpenMx Process

```
mxModel(
  name = "myModelName",
  mxMatrix(),
  mxMatrix(),
  mxAlgebra(),
  mxConstraint(),
  mxModel(mxMatrix(),
    mxData(),
    mxObjective(),
  )
)
```

```
testModel
```

```
testFit
```

```
mxEval(algebra, testFit)
```

```
summary(testFit)
```

```
output
```

```
MxModel R Objects
```

```
R Objects
```

Sunday, February 7, 2010
Key Features

- mxRun
- mxModel
- mxMatrix
- mxPath
- mxAlgebra
- mxAlgebra-Objective

- mxBounds
- mxConstraints
- mxData
- mxOption
- mxEval
- mxRAM/R/ML/FIMLObjective
mxModel

mxModel( model = NA, ....,  manifestVars = NA, latentVars = NA, remove = FALSE, independent = NA, type = NA, name = NA)

model
.....
manifestVars 
latentVars
remove
independent
type
name
optional

mxModel or string
arbitrary number of entities, data sources, Mx.... objects
list of manifest variables
list of latent variables
logical. TRUE= remove elements, else add
logical. TRUE= model is independent
character vector. name of model type
optional character vector. name of object

list of MxMatrix, MxAlgebra, MxModel, MxConstraint, MxBounds, MxData, MxObjective. MxOptions objects
mxRun

mxRun( mxModel)

mxModel any mxModel
mxMatrix

mxMatrix( type = "Full", nrow = NA, ncol = NA, free = FALSE, values = NA, labels = NA, lboun = NA, ubound = NA, byrow =getOption('mxByrow'), dimnames = NA, name = NA)

type character string for matrix type: Full Diag Iden Lower sDiag Stnd Symm Unit Zero
nrow desired number of rows
ncol desired number of columns
free logical. TRUE=free, FALSE=fixed
values vector/matrix of numeric start values
labels vector/matrix of character labels
lboun vector/matrix of numeric upper bounds
ubound vector/matrix of numeric lower bounds
byrow logical. TRUE=by row, FALSE=by column
dimnames list of length 2 for row&column names
name optional character string. name of object
mxAlgebra

mxAlgebra( expression, name = NA, dimnames = NA)

expression R expression of matrix operators & matrix functions
name optional character string. name of object
dimnames list (length=2) for row & column names
# R Matrix operators

<table>
<thead>
<tr>
<th>R (OpenMx)</th>
<th>Mx</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>solve()</code></td>
<td>~</td>
</tr>
<tr>
<td><code>t()</code></td>
<td>' '</td>
</tr>
<tr>
<td><code>+</code></td>
<td><code>+</code></td>
</tr>
<tr>
<td><code>-</code></td>
<td><code>-</code></td>
</tr>
<tr>
<td><code>%*%</code></td>
<td><code>*</code></td>
</tr>
<tr>
<td><code>*</code></td>
<td><code>. </code></td>
</tr>
<tr>
<td><code>/</code></td>
<td><code>%</code></td>
</tr>
<tr>
<td><code>%x%</code></td>
<td><code>@</code></td>
</tr>
<tr>
<td><code>%%&amp;%</code></td>
<td><code>&amp;</code></td>
</tr>
<tr>
<td><code>cbind()</code></td>
<td>`</td>
</tr>
<tr>
<td><code>rbind()</code></td>
<td><code>_^</code></td>
</tr>
</tbody>
</table>
## R Matrix operators

<table>
<thead>
<tr>
<th>R (OpenMx)</th>
<th>Mx</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>tr()</code></td>
<td>trace()</td>
</tr>
<tr>
<td><code>det()</code></td>
<td><code>\det()</code></td>
</tr>
<tr>
<td><code>sum()</code></td>
<td><code>\sum()</code></td>
</tr>
<tr>
<td><code>max()</code>, <code>min()</code></td>
<td><code>\max()</code>, <code>\min()</code></td>
</tr>
<tr>
<td><code>abs()</code></td>
<td><code>abs()</code></td>
</tr>
<tr>
<td><code>exp()</code></td>
<td><code>\exp()</code></td>
</tr>
<tr>
<td><code>log()</code></td>
<td><code>\ln()</code></td>
</tr>
<tr>
<td><code>sqrt()</code></td>
<td><code>\sqrt()</code></td>
</tr>
<tr>
<td><code>diag2vec()</code></td>
<td><code>d2v()</code></td>
</tr>
<tr>
<td><code>c(t())</code></td>
<td><code>m2v()</code></td>
</tr>
<tr>
<td><code>c()</code></td>
<td><code>vec()</code></td>
</tr>
<tr>
<td><code>vech()</code></td>
<td><code>vech()</code></td>
</tr>
<tr>
<td><code>cov2cor()</code></td>
<td><code>stnd()</code></td>
</tr>
<tr>
<td><code>Re(eigen(A..)</code></td>
<td><code>eval()</code>, <code>evec()</code></td>
</tr>
<tr>
<td><code>Im(eigen(A..</code></td>
<td><code>ival()</code>, <code>ivec()</code></td>
</tr>
<tr>
<td><code>colMeans()</code></td>
<td><code>mean()</code></td>
</tr>
<tr>
<td><code>var()</code></td>
<td><code>cov()</code></td>
</tr>
</tbody>
</table>

`sin()`, `sinh()`, `cos()`, `cosh()`, `tan()`, `tanh()` also available

`\prod()`, `\pchi()`, `\pdfnor()`, `\mnor()`, `\moment()`, `\allint()`, `\cumnor()`, `\aorder()`, `\dorder()`, `\sortr()`, `\sortc()`, `\rprod()`, `\cprod()`, `\incrow()`, `\part()`, `\chol()` not implemented yet
**mxEval**

`mxEval(expression, model, compute, show)`

- **expression**: arbitrary R expression
- **model**: model in which to evaluate expression
- **compute**: logical: TRUE= compute value of algebra expression
- **show**: logical: TRUE= print translated expression
mxData

mxData( observed, type = NA, means = NA, numObs = NA)

- **observed**: matrix or data.frame of data
- **type**: character string. type of data: raw cov cor sscp
- **means**: optional vector or means (when type = cov or cor)
- **numObs**: number of observations in data (required unless type = raw)
mxFIMLObjective

mxMLObjective( covariance, means, dimnames, thresholds)
mxFIMLObjective( covariance, means, dimnames, thresholds)

covariance  character string of name of expected covariance matrix/algebra
means       optional character string of name of expected means vector/algebra
dimnames    optional character vector to assign to dimnames of covariance and means
thresholds  optional character string of name of expected thresholds matrix/algebra
mxPath

mxPath( from = NA, to = NA, all = FALSE, arrows = 1, free = TRUE, 
values = NA, labels = NA, lbound = NA, ubound = NA)

from    character vector. sources of paths
to      character vector. sinks of paths
all     logical. TRUE=connect all sources-sinks
arrows  numeric value. 1=single, 2=double
free    logical. TRUE=free, FALSE=fixed
values  vector/matrix of numeric start values
labels  vector/matrix of character labels
lbound  vector/matrix of numeric upper bounds
ubound  vector/matrix of numeric lower bounds
mxRAMObjective

mxRAMObjective(A, S, F, M = NA, thresholds = NA)

A
character string of name of A matrix (of asymmetric paths)

S
character string of name of S matrix (of symmetric paths)

F
character string of name of F matrix (of filter matrix)

M
optional character string of name of M matrix of expected means vector

thresholds
optional character string of name of expected thresholds matrix/algebra
**mxBounds**

```
mxBounds( parameters, min = NA, max = NA)
```

- **parameters**: character vectors of names of parameters on which to apply bound
- **min**: numeric value for lower bound
- **max**: numeric value for upper bound
mxConstraint

mxBounds( alg1, relation, alg2, name = NA)

alg1 character string of name of MxMatrix or MxAlgebra object constrained to object in ‘alg2’
relation character string. relation between ‘alg1’ and ‘alg2’: <, =, >
alg2 character string of name of MxMatrix or MxAlgebra object constrained to object in ‘alg1’
name optional character string
mxOption

mxOption( model, key, value, reset = FALSE)

model          mxModel object
key            name of option
value          value of option
reset          TRUE=reset all options to defaults

getOption('mxOptimizerOptions') for default optimizer options
mxAlgebraObjective

mxAlgebraObjective(algebra)

algebra character string of name of MxMatrix or MxAlgebra object to use for optimization