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 web site. 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

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

\[
\begin{align*}
q_1 &= A + B \\
q_2 &= A \cdot A \\
q_3 &= t(A) \\
q_4 &= A \cdot t(A) \\
q_5 &= t(A) \cdot A
\end{align*}
\]
#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
algebraExercises <- 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
)
algebraExercises <- 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 )
)
answers <- mxRun(algebraExercises)
answers@algebras
result <- mxEval(list(q1,q2,q3,q4,q5),answers)
algebraExercises <- 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 )
)
answers <- mxRun(algebraExercises)
answers@algebras
result <- mxEval(list(q1,q2,q3,q4,q5),answers)
#NGroups 1
G1 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

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' )
)

Wednesday, November 11, 2009
MxModel

- name: name of object
- matrices: list of MxMatrix objects
- algebras: list of MxAlgebra objects
- submodels: list of MxModel objects
- constraints: list of MxConstraint objects
- bounds: list of MxBounds objects
MxModel II

- latentVars: list of latent variables
- manifestVars: list of manifest variables
- data: MxData object
- objective: NULL or MxObjective object
- independent: TRUE or FALSE
- options: list of optimizer options
- output: list of optimization results
Key Features

- mxRun
- mxModel
- mxMatrix
- mxPath
- mxAlgebra
- mxAlgebra-Objective
- mxBounds
- mxConstraints
- mxData
- mxOption
- mxEval
- mxRAM/R/ML/FIMLObjective
mxRun

• Usage:
  • mxRun(model)

• Arguments:
  • MxModel
mxModel usage

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

- **model**: `mxModel` or string
- **manifestVars**: list of manifest variables
- **latentVars**: list of latent variables
- **remove**: logical. TRUE= remove elements, else add
- **independent**: logical. TRUE= model is independent
- **type**: character vector. name of model type
- **name**: optional character vector. name of object
mxMatrix usage

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

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>character string for matrix type. Full Diag Iden Lower sDiag Stnd Symm Unit Zero</td>
</tr>
<tr>
<td>nrow</td>
<td>desired number of rows</td>
</tr>
<tr>
<td>ncol</td>
<td>desired number of columns</td>
</tr>
<tr>
<td>free</td>
<td>logical. TRUE=free, FALSE=fixed</td>
</tr>
<tr>
<td>values</td>
<td>vector/matrix of numeric start values</td>
</tr>
<tr>
<td>labels</td>
<td>vector/matrix of character labels</td>
</tr>
<tr>
<td>lbound</td>
<td>vector/matrix of numeric upper bounds</td>
</tr>
<tr>
<td>ubound</td>
<td>vector/matrix of numeric lower bounds</td>
</tr>
<tr>
<td>byrow</td>
<td>logical. TRUE=by row, FALSE=by column</td>
</tr>
<tr>
<td>dimnames</td>
<td>list of length 2 for row&amp;column names</td>
</tr>
<tr>
<td>name</td>
<td>optional character string. name of object</td>
</tr>
</tbody>
</table>
mxAlgebra usage

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

Wednesday, November 11, 2009
mxAlgebra arguments

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

R

solve() inversion

+ addition

- subtraction

%*% matrix multiplication

* element or dot product

/ element division

%x% Kronecker product

%% quadratic product

cbind() horizontal adhesion

rbind() vertical adhesion

power

Mx

~

‘

+

-

*

.

%

@

&

|

^
## R Matrix operators

<table>
<thead>
<tr>
<th>R</th>
<th>Mx</th>
</tr>
</thead>
<tbody>
<tr>
<td>tr()</td>
<td>\text{trace}</td>
</tr>
<tr>
<td>det()</td>
<td>\text{determinant}</td>
</tr>
<tr>
<td>sum()</td>
<td>\text{sum}</td>
</tr>
<tr>
<td>max()</td>
<td>\text{maximum}</td>
</tr>
<tr>
<td>min()</td>
<td>\text{minimum}</td>
</tr>
<tr>
<td>abs()</td>
<td>\text{absolute value}</td>
</tr>
<tr>
<td>exp()</td>
<td>\text{exponent}</td>
</tr>
<tr>
<td>log()</td>
<td>\text{natural logarithm}</td>
</tr>
<tr>
<td>sqrt()</td>
<td>\text{square root}</td>
</tr>
<tr>
<td>diag()</td>
<td>\text{diagonal 2 vector, v2d}</td>
</tr>
<tr>
<td>c(t())</td>
<td>\text{matrix 2 vector (byRow)}</td>
</tr>
<tr>
<td>c()</td>
<td>\text{matrix 2 vector (byCol)}</td>
</tr>
<tr>
<td>vech()</td>
<td>\text{lower triangular 2 vector}</td>
</tr>
</tbody>
</table>

\text{sin()}, \text{sinh()}, \text{cos()}, \text{cosh()}, \text{tan()}, \text{tanh()} also available
## R Matrix Operators

<table>
<thead>
<tr>
<th>R</th>
<th>Mx</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>cov2cor()</code></td>
<td><code>\stnd()</code></td>
</tr>
<tr>
<td><code>Re(eigen(A..)</code></td>
<td><code>\eval()</code></td>
</tr>
<tr>
<td><code>Re(eigen(A..)</code></td>
<td><code>\evec()</code></td>
</tr>
<tr>
<td><code>Im(eigen(A..)</code></td>
<td><code>\ival()</code></td>
</tr>
<tr>
<td><code>Im(eigen(A..)</code></td>
<td><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>

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

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

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
Optimization

- Estimate Correlation between X & Y

- Test whether Correlation = Zero
require(MASS)
set.seed(200)
rs=.5
xy <- mvrnorm(1000, c(0,0), matrix(c(1,rs,rs,1),2,2))
testData <- xy
selVars <- c('X','Y')
dimnames(testData) <- list(NULL, selVars)
summary(testData)
cov(testData)
Optimization Script

selVars <- c('X','Y')
bivCorModel <- mxModel("bivCor",
    mxMatrix( type="Full", nrow=1, ncol=2,
        free=TRUE, values=c(0,0), name="expMean" ),
    mxMatrix( type="Lower", nrow=2, ncol=2,
        free=TRUE, values=.5, name="Chol" ),
    mxAlgebra( expression=Chol %*% t(Chol),
        name="expCov" ),
    mxData( observed=testData, type="raw" ),
    mxFIMLObjective( covariance="expCov",
        means="expMean", dimnames=selVars) )
bivCorFit <- mxRun(bivCorModel)

bivCorModelSub <- mxModel(bivCorModel,
    mxMatrix( type="Diag", nrow=2, ncol=2,
        free=TRUE, name="Chol")
)  
bivCorFitSub <- mxRun(bivCorModelSub)
selVars <- c('X','Y')

bivCorModel <- mxModel("bivCor",
    mxMatrix("Full", 1, 2, TRUE, 0, name="expMean" ),
    mxMatrix("Lower", 2, 2, TRUE, .5, name="Chol" ),
    mxAlgebra(Chol %*% t(Chol), "expCov" ),
    mxData( observed=testData, type="raw" ),
    mxFIMLObjective("expCov", "expMean", selVars) )
bivCorFit <- mxRun(bivCorModel)

bivCorModelSub <- mxModel(bivCorModel,
    mxMatrix("Diag", 2, 2, TRUE, "Chol" )
    bivCorFitSub <- mxRun(bivCorModelSub)
mxData usage

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

- **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 usage

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

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>covariance</td>
<td>character string of name of expected covariance matrix/algebra</td>
</tr>
<tr>
<td>means</td>
<td>optional character string of name of expected means vector/algebra</td>
</tr>
<tr>
<td>dimnames</td>
<td>optional character vector to assign to dimnames of covariance and means</td>
</tr>
<tr>
<td>thresholds</td>
<td>optional character string of name of expected thresholds matrix/algebra</td>
</tr>
</tbody>
</table>
Two Model Styles
Two Data Styles

- 2 Model Styles
  - Path Specification
  - Matrix Specification

- 2 Data Styles
  - Summary Statistics
  - Raw Data
Univariate Saturated Model
mxPath usage

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

Wednesday, November 11, 2009
mxPath arguments

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 usage

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

arguments

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
Bivariate Saturated Model
mxBounds usage

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

- **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 usage

mxConstraint(
    alg1,
    relation,
    alg2,
    name = NA)
mxConstraint arguments

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 usage

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

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>model</td>
<td>mxModel object</td>
</tr>
<tr>
<td>key</td>
<td>name of option</td>
</tr>
<tr>
<td>value</td>
<td>value of option</td>
</tr>
<tr>
<td>reset</td>
<td>TRUE=reset all options to defaults</td>
</tr>
</tbody>
</table>

`getOption('mxOptimizerOptions')` for default optimizer options
mxAlgebraObjective

usage

mxAlgebraObjective(
    algebra)
mxAlgebraObjective

arguments

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