```
# matrixReview.r                                        Matrix algebra examples
#
# author: Eric Zivot
# created: November 10, 2004
# revision history:
# October 14, 2009
#   Added examples used in matrixReview.tex
# October 7, 2008
#
#
# splus commands used
#
# args          show arguments of function
# as.matrix             coerce object to class matrix
# as.vector             coerce object to class vector
# c                     combine or concatenate elements into vector
# character             create object of class character
# colIds                create or examine column ids for vector or matrix
# crossprod             cross (dot) product of two vectors
# diag          create diagonal matrix
# dimnames              create or examine dimension names of matrix
# dim                   find dimensions of matrix
# list          create list object
# matrix                create matrix object
# names         create or examine name of vector
# rep                   create vector with repeated elements
# seq                   create vector with sequence of numbers
# solve         take inverse of matrix
# t                     take transpose of matrix
# *                     multiplication
# %*%                   matrix multiplication
options(digits=4)
options(chmhelp=TRUE)

#
# create matrix
#
args(matrix)
matA = matrix(data=c(1,2,3,4,5,6),nrow=2,ncol=3)
matA
class(matA)
matA = matrix(data=c(1,2,3,4,5,6),nrow=2,ncol=3,byrow=T)
matA
class(matA)

# find dimensions
dim(matA)

# assign dimension names: rows then columns
dimnames(matA)
dimnames(matA) = list(c("row1","row2"),c("col1","col2","col3"))
matA
colnames(matA) = c("Col1", "Col2", "Col3")
rownames(matA) = c("Row1", "Row2")
matA

# subset matrix
matA[1, 2]
matA["Row1", "Col1"]
matA[1, ]
```

```
matA[, 2]
matA[1, , drop=FALSE]
matA[, 2, drop=FALSE]


# remove dimension names
dimnames(matA) = character(0)
matA

#
# create vector
#

xvec = c(1,2,3)
xvec

xvec = 1:3
xvec

xvec = seq(from=1,to=3,by=1)
xvec

class(xvec)
dim(xvec)
names(xvec) = c("x1", "x2", "x3")
xvec


# coerce vector to class matrix: note column vector is created
xvec = as.matrix(xvec)
xvec
class(xvec)



#
# matrix transpose
#
matA = matrix(data=c(1,2,3,4,5,6),nrow=2,ncol=3,byrow=T)
t(matA)
xvec = c(1,2,3)
t(xvec)

#
# symmetrix matrices
#

matS = matrix(c(1,2,2,1),2,2)
matS
# check for symmetry
matS == t(matS)

#
# Basic matrix operations
#

matA = matrix(c(4,9,2,1),2,2,byrow=T)
matB = matrix(c(2,0,0,7),2,2,byrow=T)
matA
matB
```

```
# matrix addition
matC = matA + matB
matC

# matrix subtraction
matC = matA - matB
matC

# scalar multiplication
matA
matC = 2*matA
matC

# matrix multiplication
matA = matrix(1:4,2,2,byrow=T)
matB = matrix(5:8,2,2,byrow=T)
matA
matB
dim(matA)
dim(matB)

matC = matA%*%matB
matC
# note: A%*%B is generally not equal to B%*%A
matC = matB%*%matA
matC

matA = matrix(c(1,2,3,4), 2, 2, byrow=TRUE)
vecB = c(2,6)
matA%*%vecB

vecX = c(1,2,3)
vecY = c(4,5,6)
t(vecX)%*%vecY


# create identity matrix
matI = diag(2)
matI
matA = matrix(c(1,2,3,4), 2, 2, byrow=TRUE)
matI%*%matA
matA%*%matI

# matrix inversion

matA
matA.inv = solve(matA)
matA.inv
matA%*%matA.inv
matA.inv%*%matA

#
# summation using matrix algebra
#

# create vector of 1's
onevec = rep(1,3)
onevec
xvec = c(1,2,3)

# sum elements in x
```

```
# brute force
t(xvec)%*%onevec
# more efficient
crossprod(xvec,onevec)
# more efficient still
sum(xvec)

# compute sum of squares
xvec
crossprod(xvec)
# sneaky way in splus
sum(xvec^2)

# crossproduct of two vectors
yvec = 4:6
xvec
yvec
crossprod(xvec,yvec)
crossprod(yvec,xvec)

#
# solving systems of linear equations
#

# example:
# x + y = 1
# 2x - y = 1

# plot both lines
curve(1-x, 0, 1, lwd=2, ylab="y")
abline(a=-1, b=2, lwd=2)
title("x+y=1, 2x-y=1")

# represent system in matrix algebra notation
matA = matrix(c(1,1,2,-1), 2, 2, byrow=TRUE)
vecB = c(1,1)
matA.inv = solve(matA)
matA.inv
matA.inv%*%matA
matA%*%matA.inv
z = matA.inv%*%vecB
z
```