St. Xavier's College (Autonomous), Kolkata

Department of Statistics

Handout for Problem sets 5

MDTS 4113/SEM I/CORE3 **DATE:**

1. **Plot Linear Equations in 2D**

install.packages("matlib")

library(matlib)

#enter a matrix A

#enter a vector b

showEqn(A, b) # Shows what matrices A,b look like as the system of linear equations, Ax=b, but written out as a set of equations.

plotEqn(A,b) # Shows what matrices A,b look like as the system of linear equations, Ax=b with two unknowns, x1, x2, by plotting a line for each equation.

Solve(A, b, verbose = TRUE)

[Compare with the function "solve"]

2. **Plot Linear Equations in 3D**

   #enter a matrix A [3x3]
   #enter a vector b [3x1]
   plotEqn3d(A,b)

3. **LU Decomposition**

   Explore the following options:

   i. Install the "Matrix" package in R

```
library(Matrix)
#enter the matrix, name it M
lu(M)
expand(lu(M)) #contains P, L, U
```

   ii. Install the "cmna" package in R

```
lumatrix(M)
#output contains P, L, U
```

iii. Install package “matrixcalc” in R

lu.decomposition(M)

#The Doolittle decomposition without row exchanges is performed generating the lower and upper triangular matrices separately rather than in one matrix.

#output contains L and U

works only for non-singular square matrices

iv. Explore “optR” package for solution of system of lin eqns using Gauss elimination, LU decomposition.

4. QR Factorization of a matrix using R

Explore the following options:

i. **qr(M, tol=1e-07)**
#tol: the tolerance for detecting linear dependencies in the columns of x.

ii. **qr.solve(M, b, tol = 1e-7)**#solves systems of equations via the QR decomposition.

iii. **solve.qr**

solve.qr is the method for solving for qr objects.
qr.solve solves systems of equations via the QR decomposition:
Note: If M is a QR decomposition it is the same as solve.qr, but if M is a rectangular matrix the QR decomposition is computed first. Either will handle over- (m>n) and under-determined (m<n) systems, providing a least-squares fit if appropriate.

is.qr(A)#returns TRUE if A is a list with a component named qr and FALSE otherwise.

The functions **qr.coef, qr.resid,** and **qr.fitted** return the coefficients, residuals and fitted values obtained when fitting y to the matrix with decomposition.
qr. qr.qy and **qr.qty** return Q %*% y and t(Q) %*% y, where Q is the Q matrix.

qr.coef(qr, y)
qr.qy(qr, y)
qr.qty(qr, y)
qr.resid(qr, y)
qr.fitted(qr, y, k = qr$rank)

Value

qr : a matrix with the same dimensions as x. The upper triangle contains the R of the decomposition and the lower triangle contains information on the Q of the decomposition (stored in compact form).

qraux: A vector of length ncol(x) which contains additional information on Q.

rank: the rank of x as computed by the decomposition.

pivot: information on the pivoting strategy used during the decomposition.

qr.X(qrstr, complete = FALSE, ncol =)
returns X, the original matrix from which the qr object was constructed.
If complete is TRUE or the argument ncol is greater than ncol(X), additional columns from an arbitrary orthogonal (unitary) completion of X are returned.

qr.Q(qrstr, complete = FALSE)
qr.Q returns Q, the order-nrow(X) orthogonal (unitary) transformation represented by qrstr.
If complete is TRUE, Q has nrow(X) columns.
If complete is FALSE, Q has ncol(X) columns.

qr.R(qrstr, complete = FALSE)
#qr.R returns R, the upper triangular matrix such that X == Q %*% R. The number of rows of R is nrow(X) or ncol(X), depending on whether complete is TRUE or FALSE.