Matrices in Julia

David Zeng  Keegan Go  Stephen Boyd

EE103
Stanford University

August 24, 2014
Outline

Matrices

Matrix operations
Matrices

- Matrices in Julia are represented by 2D arrays.
- To create the $2 \times 3$ matrix $A$:

\[
A = \begin{bmatrix}
2 & -4 & 8.2 \\
-5.5 & 3.5 & 63
\end{bmatrix}
\]

- Use:

\[
A = [2 \ -4 \ 8.2; \ -5.5 \ 3.5 \ 63]
\]

- Semicolons delimit rows; spaces delimit entries in a row.
- `size(A)` returns the size of $A$ as a pair, i.e.,

\[
A_{rows}, \ A_{cols} = \text{size}(A) \quad \# \text{ or} \\
A_{size} = \text{size}(A)
\]

\[
\# \ A_{rows} \ \text{is} \ A_{size}[1], \ A_{cols} \ \text{is} \ A_{size}[2]
\]

- Row vectors are $1 \times n$ matrices, e.g., $[4 \ 8.7 \ -9]$.
Indexing and slicing

- $A_{13}$ is found with $A[1,3]$
- ranges can also be used: $A[2,1:2:end]$
- `:` selects all elements along that dimension
  - $A[:,3]$ selects the third column
  - $A[2,:]$ selects the second row
  - $A[:,end:-1:1]$ reverses the order of columns

- $A[:]$ returns the columns of $A$ stacked as a vector, i.e., if $A = [2 7; 8 1]$
  then $A[:]$ returns
  $[2, 8, 7, 1]$
Block matrices

- the block matrix

\[ X = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \]

(with A, B, C, and D matrices) is formed with

\[ X = [A \ B; \ C \ D] \]

- all matrices in a row must have the same height

- the total number of columns in each row be consistent
  (c.f. standard math notation, in which A and C must have the same number of columns)
Common matrices

- $0_{m \times n}$ is $\text{zeros}(m,n)$
- $m \times n$ matrix with all entries 1 is $\text{ones}(m,n)$
- $I_{n \times n}$ is $\text{eye}(n)$
- $\text{diag}(x)$ is $\text{diagm}(x)$ (where $x$ is a vector)
- random $m \times n$ matrix with entries from standard normal distribution: $\text{randn}(m,n)$
- random $m \times n$ matrix with entries from uniform distribution on $[0,1]$: $\text{rand}(m,n)$
Outline

Matrices

Matrix operations

Matrix operations
Transpose and matrix addition

- $A^T$ is written A’ (single quote mark)
- +/- are overloaded for matrix addition/subtraction
- for example,

$$\begin{bmatrix}
4.0 & 7 \\
-10.6 & 89.8
\end{bmatrix} + \begin{bmatrix}
19 & -34.7 \\
20 & 1
\end{bmatrix}$$

is written

$[4.0 \ 7; \ -10.6 \ 89.8] + [19 \ -34.7; \ 20 \ 1]$

matrices must have the same size (unless one is a scalar)
Matrix-scalar operations

- all matrix-scalar operations (+, -, \ast) apply elementwise
- for example, matrix-scalar addition:

\[
\begin{bmatrix}
1 & 2 \\
3 & 4
\end{bmatrix}
+ 10
\]
gives
\[
\begin{bmatrix}
1 & 2 \\
3 & 4
\end{bmatrix}
+ 10 \begin{bmatrix}
1 & 1 \\
1 & 1
\end{bmatrix}
= \begin{bmatrix}
11 & 12 \\
13 & 14
\end{bmatrix}
\]

- scalar-multiplication:

\[
\begin{bmatrix}
1 & 2 \\
3 & 4
\end{bmatrix}
\ast 10
\]
gives
\[
10 \begin{bmatrix}
1 & 2 \\
3 & 4
\end{bmatrix}
= \begin{bmatrix}
10 & 20 \\
30 & 40
\end{bmatrix}
\]
Matrix-vector multiplication

- the * operator is used for matrix-vector multiplication
- for example,

\[
\begin{bmatrix}
1 & 2 \\
3 & 4
\end{bmatrix}
\begin{bmatrix}
5 \\
6
\end{bmatrix}
\]

is written
\[
[1 \ 2; \ 3 \ 4] \ * \ [5, \ 6]
\]

- for vectors x and y, x’*y finds their inner product
  - unlike dot(x,y), x’*y returns a 1 × 1 array, not a scalar
Matrix multiplication

* is overloaded for matrix-matrix multiplication:

\[
\begin{bmatrix}
2 & 4 & 3 \\
3 & 1 & 5
\end{bmatrix} \begin{bmatrix}
3 & 10 \\
4 & 2 \\
1 & 7
\end{bmatrix}
\]

is written

\[
[2\ 4\ 3;\ 3\ 1\ 5] \ast [3\ 10;\ 4\ 2;\ 1\ 7]
\]

\(A^k\) is \(A^k\) for square matrix \(A\) and nonnegative integer \(k\).
Other functions

- sum of entries of a matrix: \texttt{sum}(A)
- average of entries of a matrix: \texttt{mean}(A)
- \texttt{max}(A,B) and \texttt{min}(A,B) finds the element-wise \texttt{max} and \texttt{min} respectively
  - the arguments must have the same size unless one is a scalar
- \texttt{norm}(A) is not what you might think
  - to find \( \left( \sum_{i,j} A_{ij}^2 \right)^{1/2} \) use \texttt{norm}(A[:]) or \texttt{vecnorm}(A)
Computing regression model RMS error

the math:

- $X$ is an $n \times N$ matrix whose $N$ columns are feature $n$-vectors
- $y$ is the $N$-vector of associated outcomes
- regression model is $\hat{y} = X^T \beta + v$ ($\beta$ is $n$-vector, $v$ is scalar)
- RMS error is $\text{rms}(\hat{y} - y)$

in Julia:

```julia
y_hat = X'*beta + v
rms_error = norm(y_hat-y)/sqrt(length(y))
```