Matrix inverses in Julia

ENGR108
Stanford University

April 3, 2023

Matrix inverses in Julia

- QR factorization
- inverse
- pseudo-inverse
- backslash operator

QR factorization

- the `qr` command finds the QR factorization of a matrix

```
A = rand(5, 3)
Q, R = qr(A)
```

- when columns of $n \times k$ matrix A are independent, `qr` is same as ours
- when columns are *dependent*, `qr` is *not* same as ours
  - $A = QR$, $Q^T Q = I$, and $R_{ij} = 0$ for $i > j$ always holds
  - R can have zero or negative diagonal entries
  - R is not square when A is wide

Checking linear independence with Julia's QR

- let's check if columns of A are linearly independent
- A must be tall or square
- columns are linearly independent if and only if R has no 0 diagonal entries
- check if columns of (tall or square) A are linearly independent:

```
a1 = rand(5)
a2 = rand(5)
A = [a1 a2 a1+a2] # linearly dependent columns
Q, R = qr(A)
# find the entry of diagonal of R closest to 0
# R can have negative entries
min(abs(diag(R)))
```

Inverse

- `inv(A)` returns the inverse matrix A^{-1}
- Julia will issue an error if
  - A is not square
  - A is not invertible
- you can solve square set of linear equations $Ax = b$, with invertible A, using

  ```
  b = rand(5,1)
  A = rand(5,5)
  x = inv(A)*b
  norm(A*x-b)  # check residual
  ```

  but there is a better way, using backslash

Pseudo-inverse

- for a $m \times n$ matrix A, `pinv(A)` will return the $n \times m$ pseudo-inverse
- if A is square and invertible
  - `pinv(A)` will return the inverse A^{-1}
- if A is tall with linearly independent columns
  - `pinv(A)` will return the left inverse $(A^T A)^{-1} A^T$
- if A is wide with linearly independent rows
  - `pinv(A)` will return the right inverse $A^T (A A^T)^{-1}$
- in other cases, `pinv(A)` returns an $m \times n$ matrix, but
  - it is not a left or right inverse of A
  - what it is is beyond the scope of this class

The backslash operator

- given A and b, the \ operator solves the linear system $Ax = b$ for x
- for a $m \times n$ matrix A and a m-vector b, `A\b` returns a n-vector x
- if A is square and invertible
  - $x = A^{-1}b$
  - the unique solution of $Ax = b$
- if A is tall with linearly independent columns
  - $x = (A^T A)^{-1} A^T b$
  - the least squares approximate solution of $Ax = b$
- if A is wide with linearly independent rows
  - $x = A^T (AA^T)^{-1} b$
  - x is the least norm solution of $Ax = b$
- in other cases, `A\b` will print an error message
- uses a factor and solve method similar to QR

Solving matrix systems with backslash

- solve matrix equation $AX = B$ for X, with A square
- with $X = [x_1 \cdots x_k]$, $B = [b_1 \cdots b_k]$, same as solving k linear systems
$$Ax_1 = b_1, \ldots, Ax_k = b_k$$
- `X = A\B` solves the system, doing the right thing:
  - factor A once (order n^3)
  - back substitution to get $x_i = A^{-1}b_i$, $i = 1, \ldots, k$ (order kn^2)