Linear Least Squares

Scientific Computing Sections 3.1 - 3.4

Dr. Hyrum D. Carroll

October 4, 2016

Background

Systems of Linear Equations:

- Given $m \times n$ matrix A and m-vector b, find unknown n-vector x satisfying $Ax = b$
- System of equations asks "Can b be expressed as linear combination of columns of A?"
- If so, coefficients of linear combination are given by components of solution vector x
- Solution may or may not exist, and may or may not be unique

Vector Norms:

- Magnitude, modulus, or absolute value for scalars generalizes to norm for vectors
- We will use only p-norms, defined by:

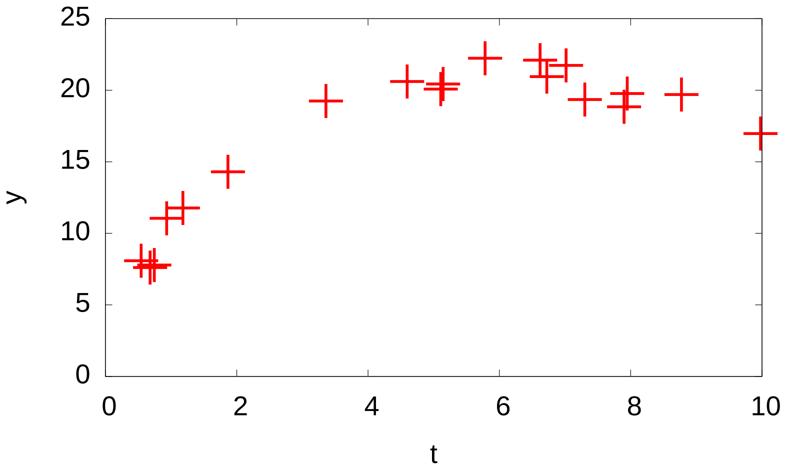
$$||x||_p = \left(\sum_{i=1}^{n} |x_i|^p \right)^{1/p}$$

for integer $p > 0$ and n-vector x

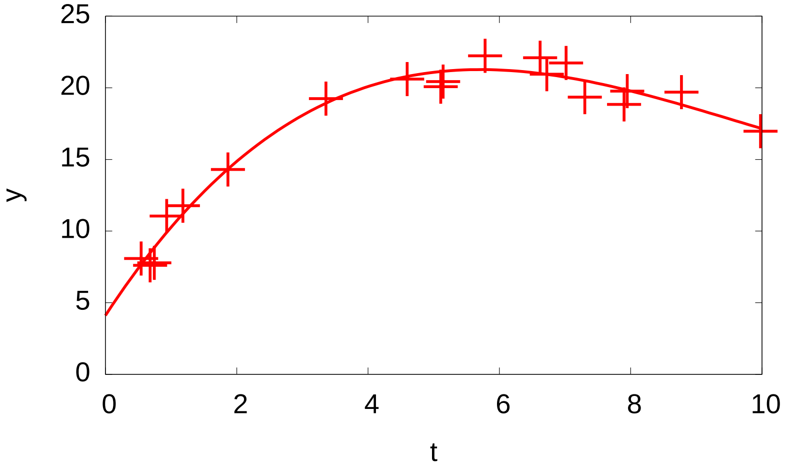
The Data

t	y
0.544395485	8.085714662
0.680401131	7.611420175
0.745381542	7.782489105
0.933087389	11.04933892
1.180802547	11.7725273
1.866634904	14.29874675
3.358481693	19.24515154
4.596300558	20.60620478
5.107420129	20.0765657
5.144204289	20.43660873
5.782206331	22.23474586
6.621454147	22.1024547
6.721689899	20.95018711
7.016279926	21.73047537
7.301068068	19.34939769
7.899499107	18.83958939
7.946676116	19.76649555
8.774087761	19.70059478
9.97773489	16.97371675

The Data



Results



Least Squares

When we fit data to some functional form f, we generally want to find a parameter x which satisfies:

$$\min_x \sum_{i=1}^{m} (y_i - f(t_i, x))^2$$

This is effectively minimizing the $||L||_2$ norm of the residual vector. We could, of course, minimize another norm instead, but the least squares approach is the most commonly used one and perhaps the easiest to implement.

This will give us an *approximation* of the underlying function.

Usually, we just want an approximation.

Polynomial Least Squares

$$Ax \cong b$$

$$\begin{bmatrix} 1 & t_1 & t_1^2 & \cdots & t_1^n \\ 1 & t_2 & t_2^2 & \cdots & t_2^n \\ 1 & t_3 & t_3^2 & \cdots & t_3^n \\ \vdots & \vdots & \vdots & \ldots & \vdots \\ 1 & t_m & t_m^2 & \cdots & t_m^n \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix} \cong \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_m \end{bmatrix}$$

we are looking for a solution

$$y = x_1 + x_2 t + x_3 t^2 + \cdots + x_n t^m$$

where t is a continuous independent variable.

General Least Squares

We could use any set of functions instead of polynomials and make the same type of fit, as long as the coefficients of x_i are combined linearly with these forms.

$$\begin{bmatrix} f_1(t_1) & f_2(t_1) & f_3(t_1) & \cdots & f_n(t_1) \\ f_1(t_2) & f_2(t_2) & f_3(t_2) & \cdots & f_n(t_2) \\ f_1(t_3) & f_2(t_3) & f_3(t_3) & \cdots & f_n(t_3) \\ \vdots & \vdots & \vdots & \ldots & \vdots \\ f_1(t_m) & f_2(t_m) & f_3(t_m) & \cdots & f_n(t_m) \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix} \cong \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_m \end{bmatrix}$$

we are looking for a solution

$$y = x_1 f_1(t) + x_2 f_2(t) + x_3 f_3(t) + \cdots + x_n f_m(t)$$

again, where x_i are constants and where t is a continuous independent variable.

General Least Squares

We could use any set of functions instead of polynomials and make the same type of fit, as long as the coefficients of x_i are combined linearly with these forms.

$$\begin{bmatrix} f_1(t_1) & f_2(t_1) & f_3(t_1) & \cdots & f_n(t_1) \\ f_1(t_2) & f_2(t_2) & f_3(t_2) & \cdots & f_n(t_2) \\ f_1(t_3) & f_2(t_3) & f_3(t_3) & \cdots & f_n(t_3) \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ f_1(t_m) & f_2(t_m) & f_3(t_m) & \cdots & f_n(t_m) \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix} \cong \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_m \end{bmatrix}$$

$$Ax \cong y$$

Residuals

For a rank[1] deficient system,

$$Ax \cong y$$

Define the residual r for the system

$$r = y - Ax$$

The $||L||_2$ norm of this residual is

$$\begin{aligned}
||r||_2 &= (r)^T(r) \\
&= (y - Ax)^T(y - Ax) \\
&= y^Ty - (Ax)^Ty - y^T(Ax) + (Ax)^T(Ax) \\
&= y^Ty - x^TA^Ty - (Ax)^Ty + x^TA^TAx \\
&= y^Ty - 2x^TA^Ty + x^TA^TAx
\end{aligned}$$

[1]rank: number of independent rows or columns

Transpose of a Vector

$$y = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ \dots \\ y_m \end{bmatrix}$$

$$y^T = [y_1, y_2, y_3, y_4, \dots, y_m]$$

What is:

1. yy^T
2. $y^T y$

Transpose of a Vector

$$yy^T = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ \dots \\ y_m \end{bmatrix} [y_1, y_2, y_3, y_4, \dots, y_m] = (a\ matrix)$$

$$\begin{aligned} y^T y &= [y_1, y_2, y_3, y_4, \dots, y_m] \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ \dots \\ y_m \end{bmatrix} \\ &= (y_1 \times y_1 + y_2 \times y_2 + y_3 \times y_3 + \dots + y_m \times y_m) \\ &= (y_1 y_1 + y_2 y_2 + y_3 y_3 + \dots + y_m y_m) \\ &= dot(y, y) \end{aligned}$$

Minimizing the Residual

To find the minimum of this function, we take the derivative wrt x and set it to zero

$$\begin{aligned}\frac{d||r||_2}{dx} &= \frac{d}{dx}(y^T y - 2x^T A^T y + x^T A^T Ax) \\ &= 0 - 2A^T y - A^T Ax + x^T A^T A \\ &= 0 - 2A^T y - A^T Ax + (A^T A)^T x \\ &= 2A^T y - 2A^T Ax = 0\end{aligned}$$

Our problem now becomes

$$\begin{aligned}2A^T y &= 2A^T Ax \\ A^T Ax &= A^T y\end{aligned}$$

Polynomial Least Squares

Our problem now becomes

$$A^T A x = A^T y$$

which can be solved using Gaussian elimination. It is helpful to take a look at what the equation looks like explicitly

$$A^T y = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ t_1 & t_2 & t_3 & \cdots & t_m \\ t_1^2 & t_2^2 & t_3^2 & \cdots & t_m^2 \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ t_1^n & t_2^n & t_3^n & \cdots & t_m^n \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_m \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^m y_i \\ \sum_{i=1}^m t_i y_i \\ \sum_{i=1}^m t_i^2 y_i \\ \vdots \\ \sum_{i=1}^m t_i^n y_i \end{bmatrix}$$

$$A^T A = \begin{bmatrix} 1 & t_1 & t_1^2 & \cdots & t_1^n \\ 1 & t_2 & t_2^2 & \cdots & t_2^n \\ 1 & t_3 & t_3^2 & \cdots & t_3^n \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ 1 & t_m & t_m^2 & \cdots & t_m^n \end{bmatrix}^T \begin{bmatrix} 1 & t_1 & t_1^2 & \cdots & t_1^n \\ 1 & t_2 & t_2^2 & \cdots & t_2^n \\ 1 & t_3 & t_3^2 & \cdots & t_3^n \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ 1 & t_m & t_m^2 & \cdots & t_m^n \end{bmatrix}$$

$$A^T A = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ t_1 & t_2 & t_3 & \cdots & t_m \\ t_1^2 & t_2^2 & t_3^2 & \cdots & t_m^2 \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ t_1^n & t_2^n & t_3^n & \cdots & t_m^n \end{bmatrix} \begin{bmatrix} 1 & t_1 & t_1^2 & \cdots & t_1^n \\ 1 & t_2 & t_2^2 & \cdots & t_2^n \\ 1 & t_3 & t_3^2 & \cdots & t_3^n \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ 1 & t_m & t_m^2 & \cdots & t_m^n \end{bmatrix}$$

$$A^T A = \begin{bmatrix} m & \sum_{i=1}^m t_i & \sum_{i=1}^m t_i^2 & \cdots & \sum_{i=1}^m t_i^n \\ \sum_{i=1}^m t_i & \sum_{i=1}^m t_i^2 & \sum_{i=1}^m t_i^3 & \cdots & \sum_{i=1}^m t_i^{n+1} \\ \sum_{i=1}^m t_i^2 & \sum_{i=1}^m t_i^3 & \sum_{i=1}^m t_i^4 & \cdots & \sum_{i=1}^m t_i^{n+2} \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ \sum_{i=1}^m t_i^n & \sum_{i=1}^m t_i^{n+1} & \sum_{i=1}^m t_i^{n+2} & \cdots & \sum_{i=1}^m t_i^{2n} \end{bmatrix}$$

Expanded Solution

$$A^T A x = A^T y$$

$$\begin{bmatrix} m & \sum_{i=1}^m t_i & \sum_{i=1}^m t_i^2 & \cdots & \sum_{i=1}^m t_i^n \\ \sum_{i=1}^m t_i & \sum_{i=1}^m t_i^2 & \sum_{i=1}^m t_i^3 & \cdots & \sum_{i=1}^m t_i^{n+1} \\ \sum_{i=1}^m t_i^2 & \sum_{i=1}^m t_i^3 & \sum_{i=1}^m t_i^4 & \cdots & \sum_{i=1}^m t_i^{n+2} \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ \sum_{i=1}^m t_i^n & \sum_{i=1}^m t_i^{n+1} & \sum_{i=1}^m t_i^{n+2} & \cdots & \sum_{i=1}^m t_i^{2n} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^m y_i \\ \sum_{i=1}^m t_i y_i \\ \sum_{i=1}^m t_i^2 y_i \\ \vdots \\ \sum_{i=1}^m t_i^n y_i \end{bmatrix}$$

General Numerical Solution

The most robust way to find a solution involves transforming the problem into a triangular form using QR factorization.
Basically, we want to take our rank deficient A matrix and move it into an $m \times m$ orthogonal matrix Q times an upper diagonal matrix R of size $n \times n$ such that

$$A = Q \begin{bmatrix} R \\ 0 \end{bmatrix}$$

In general, we would use orthogonal transforms.
There are three basic ways to do this

- Householder Transformations (reflections)
- Givens Rotations
- Gram-Schmidt orthogonalization

A Simple Case

2 Coefficients

$$y = x_1 + x_2 t$$

$$A^T A = \begin{bmatrix} m & \sum_{i=1}^m t_i \\ \sum_{i=1}^m t_i & \sum_{i=1}^m t_i^2 \end{bmatrix}$$

$$A^T y = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ t_1 & t_2 & t_3 & \cdots & t_m \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_m \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^m y_i \\ \sum_{i=1}^m t_i y_i \end{bmatrix}$$

A Simple Case

2 Coefficients

$$y = x_1 + x_2 t$$

$$\begin{aligned} A^T A x &= A^T y \\ \begin{bmatrix} m & \sum_{i=1}^m t_i \\ \sum_{i=1}^m t_i & \sum_{i=1}^m t_i^2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} &= \begin{bmatrix} \sum_{i=1}^m y_i \\ \sum_{i=1}^m t_i y_i \end{bmatrix} \end{aligned}$$

A Simple Case

2 Coefficients

$$S_1 = \sum_{i=1}^{m} t_i$$
$$S_2 = \sum_{i=1}^{m} t_i^2$$
$$S_3 = \sum_{i=1}^{m} y_i$$
$$S_4 = \sum_{i=1}^{m} t_i y_i$$

$$\begin{bmatrix} m S_1 \\ S_1 S_2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} S_3 \\ S_4 \end{bmatrix}$$

$$mx_1 + S_1 x_2 = S_3$$
$$S_1 x_1 + S_2 x_2 = S_4$$

Multiply the first equation by S_1/m

$$\begin{aligned} (S_1/m)(mx_1 + S_1 x_2) &= S_3(S_1/m) \\ S_1 x_1 + \frac{S_1^2}{n} x_2 &= \frac{S_3 S_1}{m} \end{aligned}$$

Subtract equation 2 from this

$$\begin{aligned}
\left(S_1 x_1 + \frac{S_1^2}{m} x_2 \right. &= \left. \frac{S_3 S_1}{m}\right) \\
-\left(S_1 x_1 + S_2 x_2 \right. &= \left. S_4\right) \\
\left(\frac{S_1^2}{m} - S_2\right) x_2 &= \frac{S_3 S_1}{m} - S_4
\end{aligned}$$

$$x_2 = \frac{\frac{S_3 S_1}{m} - S_4}{\frac{S_1^2}{m} - S_2}$$

$$x_1 = \frac{1}{m}\left(S_3 - S_1 x_2\right)$$

Input Data

$$\begin{bmatrix} t_1 & y_1 \\ t_2 & y_2 \\ t_3 & y_3 \\ \vdots & \vdots \\ t_m & y_m \end{bmatrix}$$

Matrix Algebra

Existence and Uniqueness

- Linear least squares problem $Ax \cong b$ always has solution
- Solution is unique if, and only if, columns of A are linearly independent, *i.e.*, $\text{rank}(A) = n$, where A is $m \times n$
- If $\text{rank}(A) < n$, then A is rank-deficient, and solution of linear least squares problem is not unique
- For now, we assume A has full column rank n

Residuals

For a rank[2] deficient system,

$$Ax \cong y$$

Define the residual r for the system

$$r = y - Ax$$

The $||L||_2$ norm of this residual is

$$\begin{aligned} ||r||_2 &= (y - Ax)^T(y - Ax) \\ &= y^T y - (Ax)^T y - y^T(Ax) + (Ax)^T(Ax) \\ &= y^T y - x^T A^T y - (Ax)^T y + x^T A^T Ax \\ &= y^T y - 2x^T A^T y + x^T A^T Ax \end{aligned}$$

[2] rank: number of independent rows or columns

Minimizing the Residual

To find the minimum of this function, we take the derivative wrt x and set it to zero

$$\begin{aligned}\frac{d||r||_2}{dx} &= \tfrac{d}{dx}(y^Ty - 2x^TA^Ty + x^TA^TAx) \\ &= 0 - 2A^Ty - A^TAx + x^TA^TA \\ &= 0 - 2A^Ty - A^TAx + (A^TA)^Tx \\ &= 2A^Ty - 2A^TAx = 0\end{aligned}$$

Our problem now becomes

$$\begin{aligned}2A^Ty &= 2A^TAx \\ A^TAx &= A^Ty\end{aligned}$$

An $n \times n$ linear system of normal equations

Orthogonality

- Vectors v_1 and v_2 are orthogonal if their inner product is zero, $v_1^T v_2 = 0$
- Space spanned by columns of $m \times n$ matrix A, $\text{span}(A) = Ax : x \in \mathbb{R}^n$, is of dimension at most n
- If $m > n$, b generally does not lie in $\text{span}(A)$, so there is no exact solution to $Ax = b$
- Vector $y = Ax$ in $\text{span}(A)$ closest to b in 2-norm occurs when residual $r = b - Ax$ is orthogonal to $\text{span}(A)$,

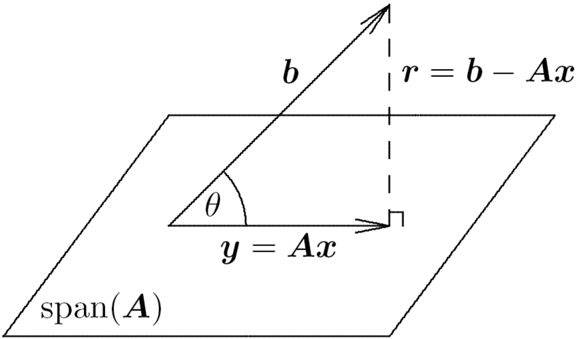
$$0 = A^T r = A^T(b - Ax)$$

again giving system of normal equations

$$A^T Ax = A^T b \tag{1}$$

Orthogonality

- Geometric relationships among b, r, and span(A) are visualized graphically



Sensitivity and Conditioning

- Sensitivity of least squares solution to $Ax \cong b$ depends on b as well as A
- Define angle θ between b and $y = Ax$ by

$$\cos(\theta) = \frac{||y||_2}{||b||_2} = \frac{||Ax||_2}{||b||_2} \tag{2}$$

- Bound on perturbation Δx in solution x due to perturbation Δb in b is given by

$$\frac{||\Delta x||_2}{||x||_2} \leq \text{cond}(A) \frac{1}{\cos(\theta)} \frac{||\Delta b||_2}{||b||_2} \tag{3}$$

Orthogonal Matrix

A matrix Q is orthogonal if

$$Q^T Q = QQ^T = I$$

Orthogonal transformations preserve the Euclidean norm of any vector v

$$||Qv||_2^2 = (Qv)^T(Qv) = v^T Q^T Qv = v^T v = ||v||_2^2$$

QR Factorization

For an $m \times n$ matrix A with $m > n$

$$A = Q \begin{bmatrix} R \\ 0 \end{bmatrix}$$

where Q is an $m \times m$ orthogonal matrix and R is an $n \times n$ upper triangular matrix

$$Ax = b$$

$$Q \begin{bmatrix} R \\ 0 \end{bmatrix} x = b$$

$$Q^T Q \begin{bmatrix} R \\ 0 \end{bmatrix} x = Q^T b$$

$$\begin{bmatrix} R \\ 0 \end{bmatrix} x = Q^T b$$

$$Rx = Q^T b$$

$$||r||_2^2 = ||b - Ax||_2^2$$
$$||r||_2^2 = ||b - Q\begin{bmatrix} R \\ 0 \end{bmatrix}x||_2^2$$
$$||r||_2^2 = ||Q^T b - \begin{bmatrix} R \\ 0 \end{bmatrix}x||_2^2$$
$$||r||_2^2 = ||c_1 - Rx||_2^2 + ||c_2||_2^2$$

where

$$Q^T b = \begin{bmatrix} c_1 \\ c_2 \end{bmatrix}$$

Inputs Gram Schmidt

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \dots & a_{2n} \\ a_{31} & a_{32} & a_{33} & \dots & a_{3n} \\ \vdots & \vdots & \vdots & \dots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & \dots & a_{nn} \end{bmatrix}$$

Inputs Gram Schmidt

Column Vectors

$$A = \left[\begin{array}{ccccc} A_1 & A_2 & A_3 & \dots & A_n \end{array} \right]$$

$$A = \left[\begin{bmatrix} a_{11} \\ a_{21} \\ a_{31} \\ \vdots \\ a_{n1} \end{bmatrix} \begin{bmatrix} a_{12} \\ a_{22} \\ a_{32} \\ \vdots \\ a_{n2} \end{bmatrix} \begin{bmatrix} a_{13} \\ a_{23} \\ a_{33} \\ \vdots \\ a_{n3} \end{bmatrix} \begin{bmatrix} \dots \\ \dots \\ \dots \\ \vdots \\ \dots \end{bmatrix} \begin{bmatrix} a_{1n} \\ a_{2n} \\ a_{3n} \\ \vdots \\ a_{nn} \end{bmatrix} \right]$$

Octave

Generating a random matrix

```
octave:2> A = rand(5)
A =

   0.420930   0.498726   0.335311   0.818492   0.360806
   0.454774   0.969672   0.689980   0.320978   0.025585
   0.827389   0.809028   0.406811   0.577818   0.841232
   0.991793   0.656681   0.533563   0.636661   0.026296
   0.831179   0.265230   0.822857   0.394929   0.675567

octave:3>  A(1,:)
ans =

   0.42093   0.49873   0.33531   0.81849   0.36081
```

Octave

```
octave:2> A = rand(5)
A =

   0.420930   0.498726   0.335311   0.818492   0.360806
   0.454774   0.969672   0.689980   0.320978   0.025585
   0.827389   0.809028   0.406811   0.577818   0.841232
   0.991793   0.656681   0.533563   0.636661   0.026296
   0.831179   0.265230   0.822857   0.394929   0.675567

octave:3>  A(:,2)
ans =

   0.49873
   0.96967
   0.80903
   0.65668
   0.26523
```

Classical Gram Schmidt

1

For k=1 to n

$q_k = a_k$

For j=1 to k-1

$r_{jk} = q_j^T a_k$

$q_k = q_k - r_{jk} q_j$

end

$r_{kk} = ||q_k||_2$

$q_{kk} = q_k / r_{kk}$

end

output: Q = [q1 q2 .... qn]

R= [rij]

```
for k = 1:n

  for j = 1:k-1


  end


end
```

Classical Gram Schmidt

2

For k=1 to n

  $q_k = a_k$

  For j=1 to k-1

    $r_{jk} = q_j^T a_k$

    $q_k = q_k - r_{jk} q_j$

  end

  $r_{kk} = ||q_k||_2$

  $q_k = q_k / r_{kk}$

end

output: Q = [q1 q2 .... qn]

R=A[rij]

```
for k = 1:n
  q(:,k) = A(:,k);
  for j = 1: k-1


  end


end
```

Classical Gram Schmidt

3

For k=1 to n

$q_k = a_k$

For j=1 to k-1

$r_{jk} = q_j^T a_k$

$q_k = q_k - r_{jk} q_j$

end

$r_{kk} = ||q_k||_2$

$q_k = q_k / r_{kk}$

end

output: Q = [q1 q2 .... qn]

R=A[rij]

```
for k = 1:n
  q(:,k) = A(:,k);
  for j = 1: k-1
    r(j,k) = q(:,j)' * A(:,k)

  end


end
```

Classical Gram Schmidt

4

For k=1 to n

$q_k = a_k$

For j=1 to k-1

$r_{jk} = q_j^T a_k$

$q_k = q_k - r_{jk} q_j$

end

$r_{kk} = ||q_k||_2$

$q_k = q_k / r_{kk}$

end

output: Q = [q1 q2 .... qn]

R=A[rij]

```
for k = 1:n
  q(:,k) = A(:,k);
  for j = 1: k-1
    r(j,k) = q(:,j)' * A(:,k)
    q(:,k) = q(:,k) - r(j,k) * q(:,j)
  end


end
```

Classical Gram Schmidt

5

For k=1 to n
  $q_k = a_k$
  For j=1 to k-1
    $r_{jk} = q_j^T a_k$
    $q_k = q_k - r_{jk} q_j$
  end
  $r_{kk} = ||q_k||_2$
  $q_k = q_k / r_{kk}$
end
output: Q = [q1 q2 .... qn]
R=A[rij]

```
for k = 1:n
  q(:,k) = A(:,k);
  for j = 1: k-1
    r(j,k) = q(:,j)' * A(:,k)
    q(:,k) = q(:,k) - r(j,k) * q(:,j)
  end
  r(k,k) = norm(q(:,k), 2)

end
```

Classical Gram Schmidt

6

For k=1 to n

$q_k = a_k$

For j=1 to k-1

$r_{jk} = q_j^T a_k$

$q_k = q_k - r_{jk} q_j$

end

$r_{kk} = ||q_k||_2$

$q_k = q_k / r_{kk}$

end

output: Q = [q1 q2 .... qn]

R=A[rij]

```
for k = 1:n
  q(:,k) = A(:,k);
  for j = 1: k-1
    r(j,k) = q(:,j)' * A(:,k)
    q(:,k) = q(:,k) - r(j,k) * q(:,j)
  end
  r(k,k) = norm(q(:,k), 2)
  q(:,k) = q(:,k) / r(k,k)
end
```

Gram-Schmidt Orthogonalization Interactive Example

http://web.engr.illinois.edu/~heath/iem/least_squares/gram_schmidt/

Modified Gram Schmidt

```
for i= 1 to n
  v_i = a_i
end
for i = 1 to n
  r_ii = ||v_i||
  q_i = v_i / r_ii
  for j = i + 1 to n
    r_ij = q_i^T v_j
    v_j = v_j - r_ij q_i
  end
end
```

Modified Gram-Schmidt Orthogonalization Interactive Example

http://web.engr.illinois.edu/~heath/iem/least_squares/modified_gram_schmidtQR/

Householder Transformations

Householder transformations, in particular, have a specific form. We define F such that

$$F \begin{bmatrix} x_1 \\ x_2 \\ . \\ . \\ . \\ x_n \end{bmatrix} = \begin{bmatrix} ||x||_2 \\ 0 \\ . \\ . \\ . \\ 0 \end{bmatrix} = ||x||_2 \begin{bmatrix} 1 \\ 0 \\ . \\ . \\ 0 \end{bmatrix} = ||x||_2 e_1$$

This is equivalent to a reflection of the vector x across the hyperplane perpendicular to

$$v = ||x||_2 e_1 - x$$

We can choose to reflect any component across any defined hyperplane by varying e_1 to a more general vector.

Householder Transformations

Of course, this can be applied to a matrix, since a matrix is really nothing more than a set of column vectors. Let

$$A = \begin{bmatrix} X & X & X & X & X \\ X & X & X & X & X \\ X & X & X & X & X \\ X & X & X & X & X \\ X & X & X & X & X \end{bmatrix}$$

where each X represents different values in the matrix. A householder transformation creates a new matrix

$$H_1 A = \begin{bmatrix} X' & X' & X' & X' & X' \\ 0 & X' & X' & X' & X' \\ 0 & X' & X' & X' & X' \\ 0 & X' & X' & X' & X' \\ 0 & X' & X' & X' & X' \end{bmatrix}$$

All the values of the A matrix have been modified, but if the right transformation is chosen, we can zero selected portions of a single column of a matrix.

Finding the Householder Reflection

Assume we want to find the F sub-matrix for the Householder reflection for a vector y (which is a column vector of A).

- Find $||y||_2$
- Find v

$$v = y \pm ||y||_2 e_1$$

- Calculate F

$$F = I - 2\frac{vv^T}{v^T v}$$

- Find H_k

$$H_k = \begin{bmatrix} I & 0 \\ 0 & F \end{bmatrix}$$

Note : vv^T is an outer product, not an inner product. Choose $\pm||y||_2$ for the best stability.

Householder Example

```
>>A(1,1) =  3; A(1,2) =   4; A(1,3) = -4;
>>A(2,1) =  5; A(2,2) = -16; A(2,3) =  2;
>>A(3,1) = 13; A(3,2) =  -3; A(3,3) = -7
A =
     3     4    -4
     5   -16     2
    13    -3    -7

>>z = A(:,1)
z =
     3
     5
    13
```

Householder Example, cont'd

```
>>e1 = zeros(3,1) ;e1(1) = 1
e1 =
     1
     0
     0
>>v = z - norm(z) * e1
v =
  -11.2478
    5.0000
   13.0000
```

Householder Example, cont'd

```
>>vvt = v * transpose(v);
>>vtv = transpose(v) * v;
>>h = eye(3,3) - 2 * vvt / vtv
h =
     0.2106    0.3509    0.9124
     0.3509    0.8440   -0.4056
     0.9124   -0.4056   -0.0546

>> h* A

ans =

    14.2478   -7.5099   -6.5273
          0  -10.8835    3.1235
     0.0000   10.3030   -4.0790
```

Repeated Householder Transformations

By repeating this process, you can zero successive columns and transform the matrix into a diagonal form.

$$H_n \cdots H_2 H_1 A = R$$

This means that the H is given by the inverse of $H_n \cdots H_2 H_1$. This is trivial to calculate since each H matrix is unitary.
In reality, you just multiply the right hand side by the H matrix at each stage of the transformation, then back-substitute at the end.

Householder Transformations Interactive Example

http://web.engr.illinois.edu/~heath/iem/least_squares/householder/