Numerical Linear Algebra

SEAS Matlab Tutorial 2

Kevin Wayne
Computer Science Department
Princeton University
Fall 2007

Linear System of Equations

Linear system of equations.

- Given n linear equations in n unknowns.
- Matrix notation: find x such that $Ax = b$.

$$\begin{aligned} 0\,x_1 &+ 1\,x_2 + 1\,x_3 = 4 \\ 2\,x_1 &+ 4\,x_2 - 2\,x_3 = 2 \\ 0\,x_1 &+ 3\,x_2 + 15\,x_3 = 36 \end{aligned}$$

$$A = \begin{bmatrix} 0 & 1 & 1 \\ 2 & 4 & -2 \\ 0 & 3 & 15 \end{bmatrix}, \quad b = \begin{bmatrix} 4 \\ 2 \\ 36 \end{bmatrix}$$

Among most fundamental problems in science and engineering.

- Chemical equilibrium. ← see Lab 2
- Google's PageRank algorithm.
- Linear and nonlinear optimization.
- Kirchoff's current and voltage laws.
- Hooke's law for finite element methods.
- Numerical solutions to differential equations.

Chemical Equilibrium

Ex: combustion of propane.

$$x_1 C_3H_8 + x_2 O_2 \Rightarrow x_3 CO_2 + x_4 H_2O$$

Stoichiometric constraints.

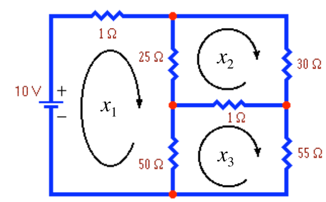
- Carbon: $3x_1 = x_3$
- Hydrogen: $8x_1 = 2x_4$
- Oxygen: $2x_2 = 2x_3 + x_4$
- Normalize: $x_1 = 1$

(Carbon, Hydrogen, Oxygen: conservation of mass)

$$C_3H_8 + 5O_2 \Rightarrow 3CO_2 + 4H_2O$$

Circuit Analysis

Ex: find current flowing in each branch of a circuit.



Kirchoff's current law.

- $10 = 1x_1 + 25(x_1 - x_2) + 50\,(x_1 - x_3)$
- $0 = 25(x_2 - x_1) + 30x_2 + 1(x_2 - x_3)$
- $0 = 50(x_3 - x_1) + 1(x_3 - x_2) + 55x_3$

(conservation of electrical charge)

Solution: $x_1 = 0.2449, x_2 = 0.1114, x_3 = 0.1166.$

Gaussian Elimination

Gaussian elimination.

- Among oldest and most widely used solutions.
- Repeatedly apply row operations until system is upper triangular.
- Solve "trivial" upper triangular system via back substitution.

$$\begin{bmatrix} 0 & 1 & 1 \\ 2 & 4 & -2 \\ 0 & 3 & 15 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 4 \\ 2 \\ 36 \end{bmatrix}$$

```
>> A = [0 1 1; 2 4 -2; 0 3 15];
>> b = [4; 2; 36];
>> x = lsolve(A, b)
x =
   -1
    2
    2
```

we are going to implement this

Gaussian Elimination

```
>> A = [0 1 1; 2 4 -2; 0 3 15]
A =  0     1     1
     2     4    -2
     0     3    15
```

declare a matrix

```
>> A([1 2], :) = A([2 1], :)
A =  2     4    -2
     0     1     1
     0     3    15
```

swap rows 1 and 2

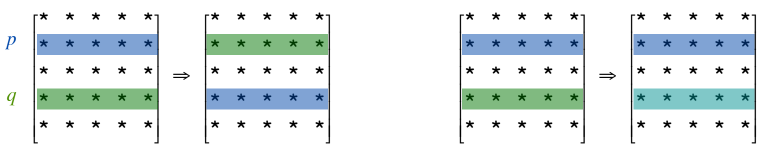
```
>> A(3, :) = A(3, :) - 3 * A(2, :)
A =  2     4    -2
     0     1     1
     0     0    12
```

subtract 3 times row 2 from row 3

Elementary Row Operations

Elementary row operations.

- Exchange row p and row q.
- Add a multiple α of row p to row q.




```
A([p q], :) = A([q p], :);
b([p q], :) = b([q p], :);
```

```
A(q, :) = A(q, :) - alpha * A(p, :);
b(q, :) = b(q, :) - alpha * b(p, :);
```

Key invariant. Row operations preserve solutions.

Gaussian Elimination: Row Operations

$$\begin{bmatrix} 0 & 1 & 1 \\ 2 & 4 & -2 \\ 0 & 3 & 15 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 4 \\ 2 \\ 36 \end{bmatrix}$$

(interchange row 1 and 2)

$$\begin{bmatrix} 2 & 4 & -2 \\ 0 & 1 & 1 \\ 0 & 3 & 15 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 2 \\ 4 \\ 36 \end{bmatrix}$$

(subtract 3x row 2 from row 3)

$$\begin{bmatrix} 2 & 4 & -2 \\ 0 & 1 & 1 \\ 0 & 0 & 12 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 2 \\ 4 \\ 24 \end{bmatrix}$$

Gaussian Elimination: Back Substitution

Back substitution. Upper triangular systems are easy to solve by examining equations in reverse order.

$$\begin{bmatrix} 2 & 4 & -2 \\ 0 & 1 & 1 \\ 0 & 0 & 12 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 2 \\ 4 \\ 24 \end{bmatrix}$$

Eq 3. $x_3 = 24/12 = 2$

Eq 2. $x_2 = 4 - x_3 = 2$

Eq 1. $x_1 = (2 - 4x_2 + 2x_3) / 2 = -1$

$$x_i = \frac{1}{a_{ii}} \left[b_i - \sum_{j=i+1}^{n} a_{ij} x_j \right]$$

```
[m n] = size(A);
x = zeros(n, 1);
for i = n : -1 : 1
   total = 0.0;
   for j = i+1 : n
      total = total + A(i, j) * x(j);
   end
   x(i) = (b(i) - total) / A(i, i);
end
```


Gaussian Elimination: Back Substitution

Back substitution. Upper triangular systems are easy to solve by examining equations in reverse order.

$$\begin{bmatrix} 2 & 4 & -2 \\ 0 & 1 & 1 \\ 0 & 0 & 12 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 2 \\ 4 \\ 24 \end{bmatrix}$$

Eq 3. $x_3 = 24/12 = 2$

Eq 2. $x_2 = 4 - x_3 = 2$

Eq 1. $x_1 = (2 - 4x_2 + 2x_3) / 2 = -1$

$$x_i = \frac{1}{a_{ii}} \left[b_i - \sum_{j=i+1}^{n} a_{ij} x_j \right]$$

vectorized version

```
[m n] = size(A);
x = zeros(size(b));
for i = n : -1 : 1
   j = i+1 : n
   x(i, :) = ((b(i, :) - A(i, j) * x(j, :)) / A(i, i);
end
```

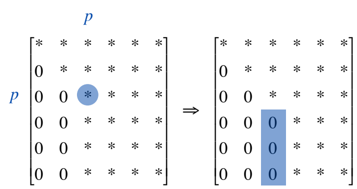
vector inner product

Gaussian Elimination: Forward Elimination

Forward elimination. Apply row operations to make upper triangular.

Pivot. Zero out entries below pivot a_{pp}.

$$\begin{aligned} \alpha &= a_{ip} / a_{pp} \\ a_{ij} &= a_{ij} - \alpha \, a_{pj} \\ b_i &= b_i - \alpha \, b_p \end{aligned}$$

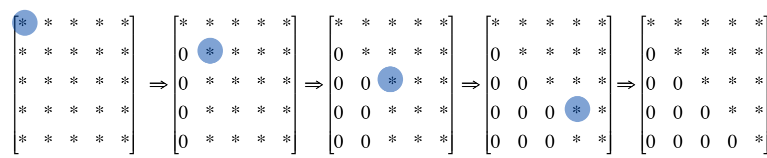


```
for i = p+1 : n
   alpha = A(i, p) / A(p, p);
   b(i, :) = b(i, :) - alpha * b(p, :);
   A(i, :) = A(i, :) - alpha * A(p, :);
end
```


Gaussian Elimination: Forward Elimination

Forward elimination. Apply row operations to make upper triangular.

Pivot. Zero out entries below pivot a_{pp}.



```
for p = 1 : n
   for i = p+1 : n
      alpha = A(i, p) / A(p, p);
      b(i, :) = b(i, :) - alpha * b(p, :);
      A(i, :) = A(i, :) - alpha * A(p, :);
   end
end
```

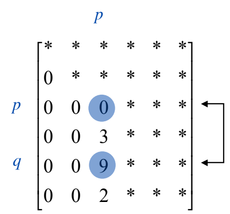
Gaussian Elimination: Partial Pivoting

Remark. Code on previous slide fails spectacularly if pivot $a_{pp} = 0$.

Partial pivoting. Swap row p with the row q that has largest entry in column p among rows below the diagonal.

```
q = p;
for i = p+1 : n
   if (abs(A(i, p)) > abs(A(q, p))
      q = i;
   end
end

A([p q], :) = A([q p], :);
b([p q], :) = b([q p], :);
```



Gaussian Elimination: Partial Pivoting

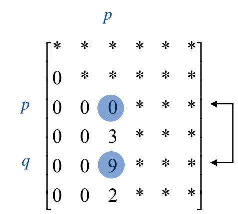
Remark. Code on previous slide fails spectacularly if pivot $a_{pp} = 0$.

Partial pivoting. Swap row p with the row q that has largest entry in column p among rows below the diagonal.

```
[val q] = max(abs(A(p:n, p)));
q = q + p - 1;

A([p q], :) = A([q p], :);
b([p q], :) = b([q p], :);
```

vectorized version



Gaussian Elimination with Partial Pivoting

```
function x = lsolve(A, b)
% LSOLVE  Linear system of equation solver, bare bones version
%   x = lsolve(A, b) returns the solution to the equation Ax = b,
%   where A is an n-by-n nonsingular matrix, and b is a column
%   vector of length n (or a matrix with several such columns).

[m n] = size(A);

% Gaussian elimination with partial pivoting
for p = 1 : n

   % find index q of largest element below diagonal in column p
   [val q] = max(abs(A(p:n, p)));
   q = q + p - 1;

   % swap with row p
   A([p q], :) = A([q p], :);
   b([p q], :) = b([q p], :);

   % zero out entries of A and b using pivot A(p, p)
   for i = p+1 : n
      alpha = A(i, p) / A(p, p);
      b(i, :) = b(i, :) - alpha * b(p, :);
      A(i, :) = A(i, :) - alpha * A(p, :);
   end
end

% back substitution
x = zeros(size(b));
for i = n : -1 : 1
   j = i+1 : n;
   x(i, :) = (b(i, :) - A(i, j) * x(j, :)) / A(i, i);
end
```



```
x = A \ b;
```

Singular Value Decomposition

Singular value decomposition. Given a real, square matrix A, the SVD is $A = U S V^T$, where U and V are orthogonal, and S is diagonal.

($U U^T = I$; singular values in descending order)

- Among most important concepts in matrix computation.
- Applications: statistics, signal processing, acoustics, vibrations, ...

$$\underbrace{\begin{bmatrix} 4 & -1 & 1 \\ -2 & -2 & 2 \\ 0 & 5 & 5 \end{bmatrix}}_{A} = \underbrace{\begin{bmatrix} 0 & \frac{-2}{\sqrt{5}} & \frac{-1}{\sqrt{5}} \\ 0 & \frac{1}{\sqrt{5}} & \frac{-2}{\sqrt{5}} \\ 1 & 0 & 0 \end{bmatrix}}_{U} \underbrace{\begin{bmatrix} \sqrt{50} & 0 & 0 \\ 0 & \sqrt{20} & 0 \\ 0 & 0 & \sqrt{10} \end{bmatrix}}_{S} \underbrace{\begin{bmatrix} 0 & -1 & 0 \\ \frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & 0 & \frac{-1}{\sqrt{2}} \end{bmatrix}}_{V}^{T}$$

Principal Component Analysis

Principal component analysis (PCA). Truncated SVD is $A_r = U_r S_r V_r^T$, where U_r and V_r are the first r columns of U and V, and S_r is the first r rows and columns of S.

Fact. A_r is the "best" rank r approximation to A.

$$\underbrace{\begin{bmatrix} 4 & -1 & 1 \\ -2 & -2 & 2 \\ 0 & 5 & 5 \end{bmatrix}}_{A} = \underbrace{\begin{bmatrix} 0 & \frac{-2}{\sqrt{5}} & \frac{-1}{\sqrt{5}} \\ 0 & \frac{1}{\sqrt{5}} & \frac{-2}{\sqrt{5}} \\ 1 & 0 & 0 \end{bmatrix}}_{U} \underbrace{\begin{bmatrix} \sqrt{50} & 0 & 0 \\ 0 & \sqrt{20} & 0 \\ 0 & 0 & \sqrt{10} \end{bmatrix}}_{S} \underbrace{\begin{bmatrix} 0 & -1 & 0 \\ \frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & 0 & \frac{-1}{\sqrt{2}} \end{bmatrix}}_{V}^{T}$$

r = 2

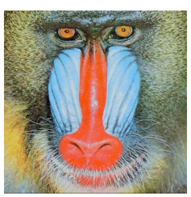
$$\underbrace{\begin{bmatrix} 4 & 0 & 0 \\ -2 & 0 & 0 \\ 0 & 5 & 5 \end{bmatrix}}_{A_r} = \underbrace{\begin{bmatrix} 0 & \frac{-2}{\sqrt{5}} \\ 0 & \frac{1}{\sqrt{5}} \\ 1 & 0 \end{bmatrix}}_{U_r} \underbrace{\begin{bmatrix} \sqrt{50} & 0 \\ 0 & \sqrt{20} \end{bmatrix}}_{S_r} \underbrace{\begin{bmatrix} 0 & -1 \\ \frac{1}{\sqrt{2}} & 0 \\ \frac{1}{\sqrt{2}} & 0 \end{bmatrix}}_{V_r}^{T}$$

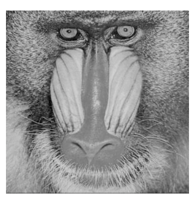
$$\|A_r - A\|_2 = \sqrt{10}$$

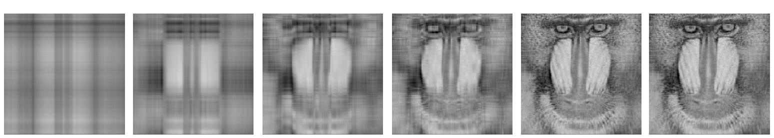
Image Processing: PCA

Image processing.

- Read in color image.
- Convert to grayscale.
- Create n-by-n matrix of grayscale values.
- Compute best rank $\{ 1, 2, 5, 10, 25, 50 \}$ approximation.







baboon.m

```
% MATLAB script that reads in the image baboon.jpg,
% converts it to grayscale, and forms a matrix of its
% grayscale values.
%
% Then it computes and plots the best rank r approximate
% to the matrix using the SVD. It saves each approximation
% as a JPEG image.

A = imread('baboon.jpg');   % read image from a file
A = rgb2gray(A);            % convert from color to grayscale
A = im2double(A);           % convert to double precision matrix
imshow(A);                  % display the image in a window

[U S V] = svd(A);
for r = [1 2 5 10 25 50 100 298]
   Ar = U(:, 1:r) * S(1:r, 1:r) * V(:, 1:r)';
   imshow(Ar);
   pause;
   imwrite(Ar, sprintf('baboon-%d.jpg', r));
end
```

Faces

average

7 Principal Faces

Reference: Diego Nehab, COS 496