Eigenvectors and SVD

Eigenvectors of a square matrix

- Definition

$$\mathbf{A}\mathbf{x} = \lambda\mathbf{x}, \quad \mathbf{x} \neq 0 \ .$$

- Intuition: x is unchanged by A (except for scaling)
- Examples: axis of rotation, stationary distribution of a Markov chain

Diagonalization

- Stack up evec equation to get

$$\mathbf{AX} = \mathbf{X\Lambda}$$

- Where

$$\mathbf{X} \in \mathbb{R}^{n \times n} = \begin{bmatrix} | & | & & | \\ \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_n \\ | & | & & | \end{bmatrix}, \quad \mathbf{\Lambda} = \mathrm{diag}(\lambda_1, \ldots, \lambda_n) \ .$$

- If evecs are linearly indep, X is invertible, so

$$\mathbf{A} = \mathbf{X\Lambda X}^{-1}.$$

Evecs of symmetric matrix

- All evals are real (not complex)
- Evecs are orthonormal

$$\mathbf{u}_i^T \mathbf{u}_j = 0 \text{ if } i \neq j, \qquad \mathbf{u}_i^T \mathbf{u}_i = 1$$

- So U is orthogonal matrix

$$\mathbf{U}^T \mathbf{U} = \mathbf{U}\mathbf{U}^T = \mathbf{I}$$

Diagonalizing a symmetric matrix

- We have

$$\mathbf{A} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T = \sum_{i=1}^{n} \lambda_i \mathbf{u}_i \mathbf{u}_i^T$$

$$\begin{aligned}
\mathbf{A} &= = \begin{pmatrix} | & | & & | \\ \mathbf{u}_1 & \mathbf{u}_2 & \cdots & \mathbf{u}_n \\ | & | & & | \end{pmatrix} \begin{pmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_n \end{pmatrix} \begin{pmatrix} - & \mathbf{u}_1^T & - \\ - & \mathbf{u}_2^T & - \\ & \vdots & \\ - & \mathbf{u}_n^T & - \end{pmatrix} \\
&= \lambda_1 \begin{pmatrix} | \\ \mathbf{u}_1 \\ | \end{pmatrix} \begin{pmatrix} - & \mathbf{u}_1^T & - \end{pmatrix} + \cdots + \lambda_n \begin{pmatrix} | \\ \mathbf{u}_n \\ | \end{pmatrix} \begin{pmatrix} - & \mathbf{u}_n^T & - \end{pmatrix}
\end{aligned}$$

Transformation by an orthogonal matrix

- Consider a vector x transformed by the orthogonal matrix U to give

$$\tilde{\mathbf{x}} \quad = \quad U\mathbf{x}$$

- The length of the vector is preserved since

$$||\tilde{\mathbf{x}}||^2 = \tilde{\mathbf{x}}^T\tilde{\mathbf{x}} = \mathbf{x}^T U^T U^T \mathbf{x} = \mathbf{x}^T\mathbf{x} = ||\mathbf{x}||^2$$

- The angle between vectors is preserved

$$\tilde{\mathbf{x}}^T\tilde{\mathbf{y}} = \mathbf{x}^T U^U \mathbf{y} = \mathbf{x}^T\mathbf{y}$$

- Thus multiplication by U can be interpreted as a rigid rotation of the coordinate system.

Geometry of diagonalization

- Let A be a linear transformation. We can always decompose this into a rotation U, a scaling Λ, and a reverse rotation $U^T = U^{-1}$.
- Hence $A = U \Lambda U^T$.
- The inverse mapping is given by $A^{-1} = U \Lambda^{-1} U^T$

$$
\begin{aligned}
A &= \sum_{i=1}^{m} \lambda_i \mathbf{u}_i \mathbf{u}_i^T \\
A^{-1} &= \sum_{i=1}^{m} \frac{1}{\lambda_i} \mathbf{u}_i \mathbf{u}_i^T
\end{aligned}
$$

Matlab example

- Given

$$\mathbf{A} = \begin{pmatrix} 1.5 & -0.5 & 0 \\ -0.5 & 1.5 & 0 \\ 0 & 0 & 3 \end{pmatrix}$$

- Diagonalize

```
[U,D]=eig(A)
U =
   -0.7071   -0.7071         0
   -0.7071    0.7071         0
         0         0    1.0000
D =
     1     0     0
     0     2     0
     0     0     3
```

Rot(45)

Scale(1,2,3)

- check

```
>> U*D*U'
ans =
    1.5000   -0.5000         0
   -0.5000    1.5000         0
         0         0    3.0000
```

Positive definite matrices

- A matrix A is pd if $x^T A x > 0$ for any non-zero vector x.
- Hence all the evecs of a pd matrix are positive

$$\begin{aligned} A\mathbf{u}_i &= \lambda_i \mathbf{u}_i \\ \mathbf{u}_i^T A \mathbf{u}_i &= \lambda_i \mathbf{u}_i^T \mathbf{u}_i = \lambda_i > 0 \end{aligned}$$

- A matrix is positive semi definite (psd) if $\lambda_i >= 0$.
- A matrix of all positive entries is not necessarily pd; conversely, a pd matrix can have negative entries

```
> [u,v] = eig([1 2; 3 4])
u =
   -0.8246   -0.4160
    0.5658   -0.9094
v =
   -0.3723         0
         0    5.3723
```

```
[u,v]=eig([2 -1; -1 2])
u =
   -0.7071   -0.7071
   -0.7071    0.7071
v =
     1     0
     0     3
```

Multivariate Gaussian

- Multivariate Normal (MVN)

$$\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \Sigma) \stackrel{\text{def}}{=} \frac{1}{(2\pi)^{p/2}|\Sigma|^{1/2}} \exp[-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T \Sigma^{-1} (\mathbf{x}-\boldsymbol{\mu})]$$

- Exponent is the Mahalanobis distance between x and μ

$$\Delta = (\mathbf{x}-\boldsymbol{\mu})^T \Sigma^{-1} (\mathbf{x}-\boldsymbol{\mu})$$

Σ is the covariance matrix (symmetric positive definite)

$$\mathbf{x}^T \Sigma \mathbf{x} > 0 \; \forall \mathbf{x}$$

Bivariate Gaussian

- Covariance matrix is

$$\Sigma = \begin{pmatrix} \sigma_x^2 & \rho\sigma_x\sigma_y \\ \rho\sigma_x\sigma_y & \sigma_y^2 \end{pmatrix}$$

where the correlation coefficient is

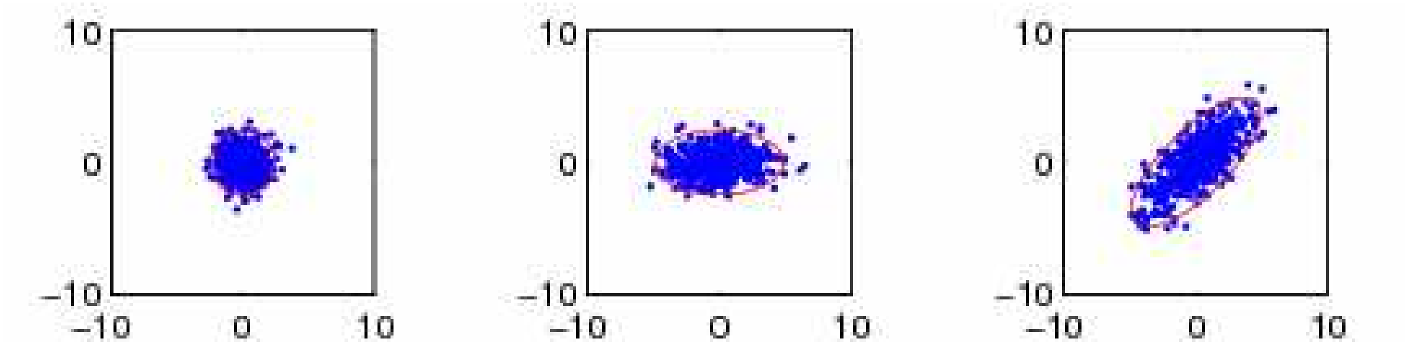
$$\rho = \frac{Cov(X,Y)}{\sqrt{Var(X)Var(Y)}}$$

and satisfies -1 $\leq$ ρ $\leq$ 1

- Density is

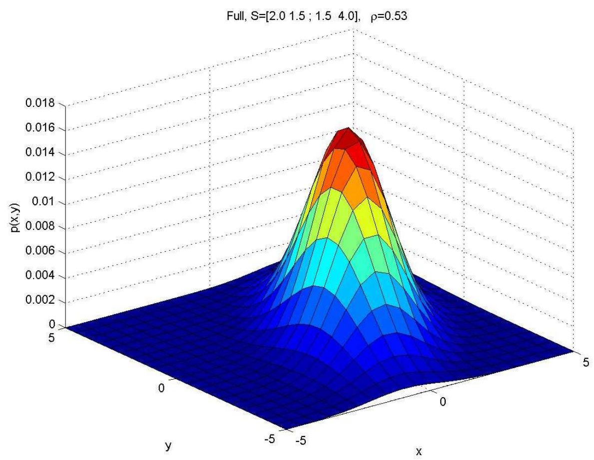
$$p(x,y) = \frac{1}{2\pi\sigma_x\sigma_y\sqrt{1-\rho^2}} \exp\left(-\frac{1}{2(1-\rho^2)}\left(\frac{x^2}{\sigma_x^2} + \frac{y^2}{\sigma_y^2} - \frac{2\rho xy}{(\sigma_x\sigma_y)}\right)\right)$$

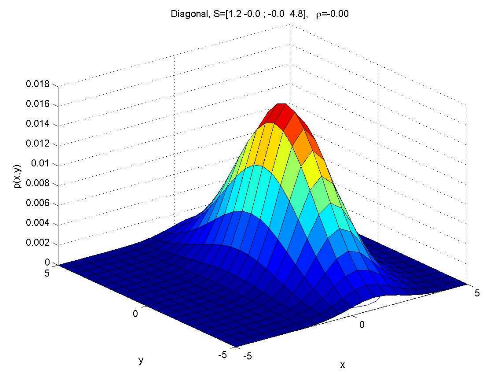
Spherical, diagonal, full covariance

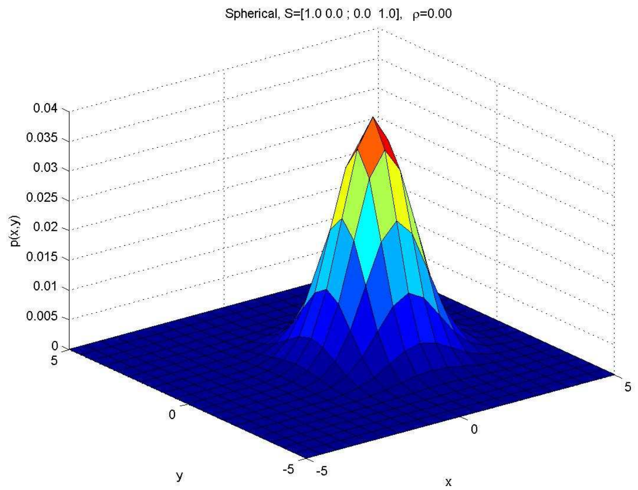


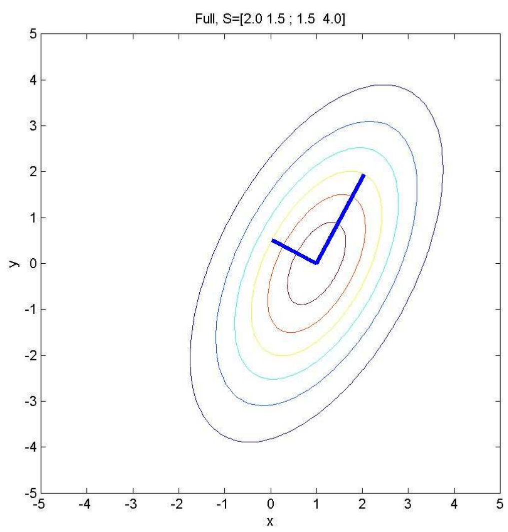
$$\Sigma = \begin{pmatrix} \sigma^2 & 0 \\ 0 & \sigma^2 \end{pmatrix} \qquad \Sigma = \begin{pmatrix} \sigma_x^2 & 0 \\ 0 & \sigma_y^2 \end{pmatrix} \qquad \Sigma = \begin{pmatrix} \sigma_x^2 & \rho\sigma_x\sigma_y \\ \rho\sigma_x\sigma_y & \sigma_y^2 \end{pmatrix}$$

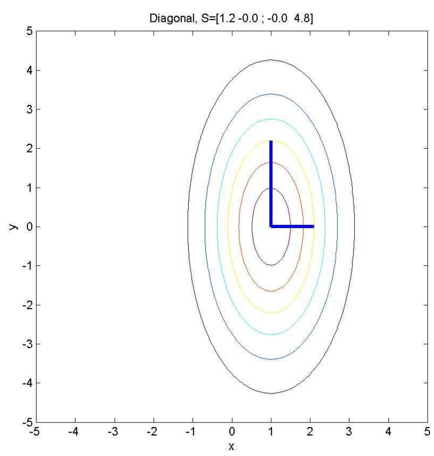
Surface plots

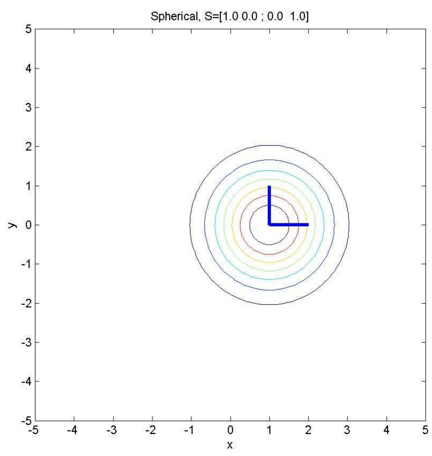












Matlab plotting code

```
stepSize = 0.5;
[x,y] = meshgrid(-5:stepSize:5,-5:stepSize:5);
[r,c]=size(x);
data = [x(:) y(:)];
p = mvnpdf(data, mu', S);
p = reshape(p, r, c);
surfc(x,y,p);                         % 3D plot
contour(x,y,p);                  % Plot contours
```

Visualizing a covariance matrix

- Let Σ = U Λ U^T. Hence

$$\Sigma^{-1} = U^{-T}\Lambda^{-1}U^{-1} = U\Lambda^{-1}U = \sum_{i=1}^{p} \frac{1}{\lambda_i}\mathbf{u}_i\mathbf{u}_i^T$$

- Let y = U(x-μ) be a transformed coordinate system, translated by μ and rotated by U. Then

$$\begin{aligned}(\mathbf{x}-\boldsymbol{\mu})^T\Sigma^{-1}(\mathbf{x}-\boldsymbol{\mu}) &= (\mathbf{x}-\boldsymbol{\mu})^T\left(\sum_{i=1}^{p}\frac{1}{\lambda_i}\mathbf{u}_i\mathbf{u}_i^T\right)(\mathbf{x}-\boldsymbol{\mu}) \\ &= \sum_{i=1}^{p}\frac{1}{\lambda_i}(\mathbf{x}-\boldsymbol{\mu})^T\mathbf{u}_i\mathbf{u}_i^T(\mathbf{x}-\boldsymbol{\mu}) = \sum_{i=1}^{p}\frac{y_i^2}{\lambda_i}\end{aligned}$$

Visualizing a covariance matrix

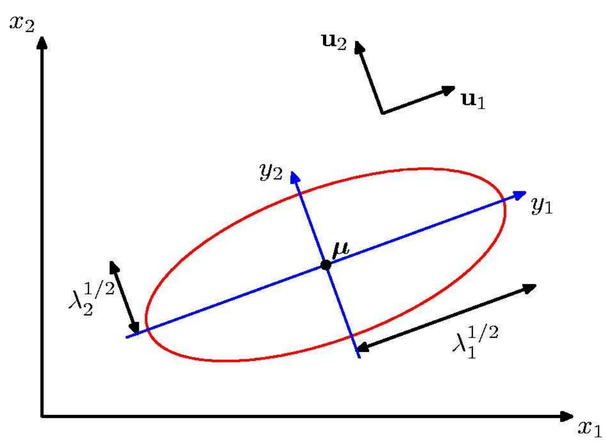
- From the previous slide

$$(\mathbf{x}-\boldsymbol{\mu})^T\Sigma^{-1}(\mathbf{x}-\boldsymbol{\mu}) = \sum_{i=1}^{p}\frac{y_i^2}{\lambda_i}$$

- Recall that the equation for an ellipse in 2D is

$$\frac{y_1^2}{\lambda_1}+\frac{y_2^2}{\lambda_2}=1$$

- Hence the contours of equiprobability are elliptical, with axes given by the evecs and scales given by the evals of Σ



Visualizing a covariance matrix

- Let X ~ N(0,I) be points on a 2d circle.
- If

$$\mathbf{Y} = \mathbf{U}\boldsymbol{\Lambda}^{\frac{1}{2}}\mathbf{X}$$

$$\boldsymbol{\Lambda}^{\frac{1}{2}} = \mathrm{diag}(\sqrt{\Lambda_{ii}})$$

- Then

$$\mathrm{Cov}[\mathbf{y}] = \mathbf{U}\boldsymbol{\Lambda}^{\frac{1}{2}}\mathrm{Cov}[\mathbf{x}]\boldsymbol{\Lambda}^{\frac{1}{2}}\mathbf{U}^T = \mathbf{U}\boldsymbol{\Lambda}^{\frac{1}{2}}\boldsymbol{\Lambda}^{\frac{1}{2}}\mathbf{U}^T = \boldsymbol{\Sigma}$$

- So we can map a set of points on a circle to points on an ellipse

Implementation in Matlab

$$\mathbf{Y} = \mathbf{U}\boldsymbol{\Lambda}^{\frac{1}{2}}\mathbf{X}$$

$$\boldsymbol{\Lambda}^{\frac{1}{2}} = \mathrm{diag}(\sqrt{\Lambda_{ii}})$$

```
function h=gaussPlot2d(mu, Sigma, color)
[U, D] = eig(Sigma);
n = 100;
t = linspace(0, 2*pi, n);
xy = [cos(t); sin(t)];
k = 6; %k = sqrt(chi2inv(0.95, 2));
w = (k * U * sqrt(D)) * xy;
z = repmat(mu, [1 n]) + w;
h = plot(z(1, :), z(2, :), color); axis('equal
```

Standardizing the data

- We can subtract off the mean and divide by the standard deviation of each dimension to get the following (for case i=1:n and dimension j=1:d)

$$y_{ij} = \frac{x_{ij} - \overline{x}_j}{\sigma_j}$$

- Then E[Y]=0 and Var[Y_j]=1.
- However, Cov[Y] might still be elliptical due to correlation amongst the dimensions.

Whitening the data

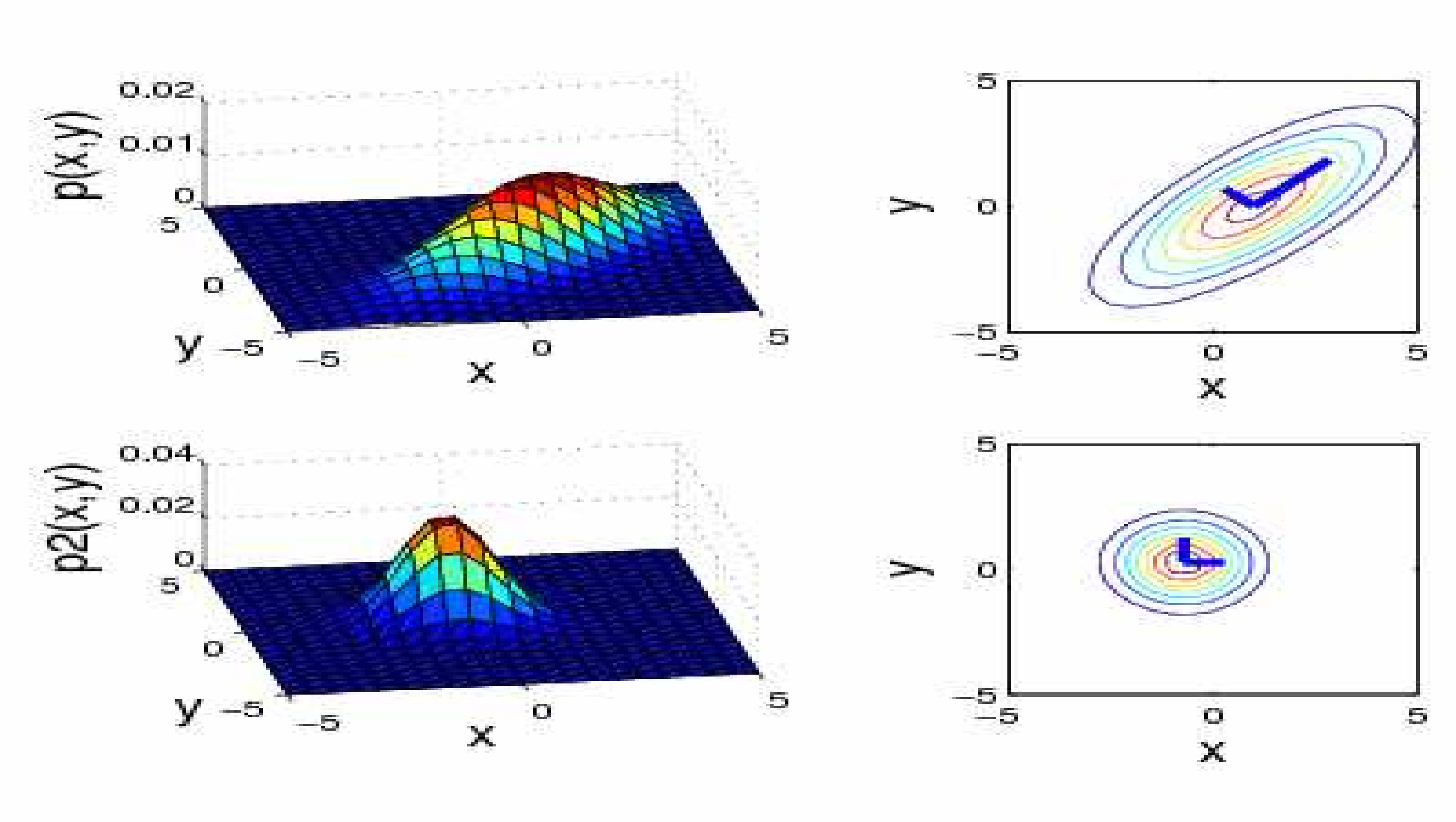
- Let X ~ N(μ,Σ) and Σ = U Λ U^T.
- To remove any correlation, we can apply the following linear transformation

$$\begin{aligned} Y &= \Lambda^{-\frac{1}{2}} U^T X \\ \Lambda^{-\frac{1}{2}} &= \mathrm{diag}(1/\sqrt{\Lambda_{ii}}) \end{aligned}$$

- In Matlab

```
[U,D] = eig(cov(X));
Y = sqrt(inv(D)) * U' * X;
```

Whitening: example



Whitening: proof

- Let

$$\begin{aligned} Y &= \Lambda^{-\frac{1}{2}} U^T X \\ \Lambda^{-\frac{1}{2}} &= \mathrm{diag}(1/\sqrt{\Lambda_{ii}}) \end{aligned}$$

- Using

$$\mathrm{Cov}[AX] = A\mathrm{Cov}[X]A^T$$

we have

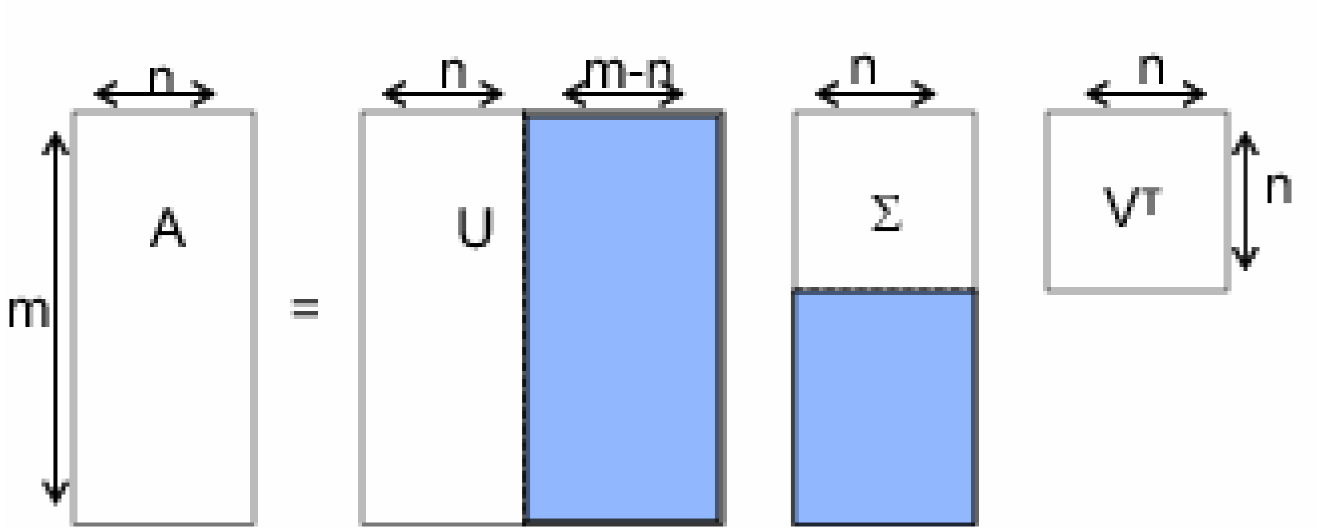
$$\begin{aligned} \mathrm{Cov}[Y] &= \Lambda^{-\frac{1}{2}} U^T \Sigma U \Lambda^{-\frac{1}{2}} \\ &= \Lambda^{-\frac{1}{2}} U^T (U \Lambda U^T) U \Lambda^{-\frac{1}{2}} \\ &= \Lambda^{-\frac{1}{2}} \Lambda \Lambda^{-\frac{1}{2}} = I \end{aligned}$$

and

$$EY = \Lambda^{-\frac{1}{2}} U^T E[X]$$

Singular Value Decomposition

$$\mathbf{A} = \mathbf{U\Sigma V}^T = \lambda_1 \begin{pmatrix} | \\ \mathbf{u}_1 \\ | \end{pmatrix} \begin{pmatrix} - & \mathbf{v}_1^T & - \end{pmatrix} + \cdots + \lambda_r \begin{pmatrix} | \\ \mathbf{u}_r \\ | \end{pmatrix} \begin{pmatrix} - & \mathbf{v}_r^T & - \end{pmatrix}$$



Right svectors are evecs of A^T A

- For any matrix A

$$
\begin{aligned}
\mathbf{A}^T\mathbf{A} &= \mathbf{V}\mathbf{\Sigma}^T\mathbf{U}^T\,\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \\
&= \mathbf{V}(\mathbf{\Sigma}^T\mathbf{\Sigma})\mathbf{V}^T \\
(\mathbf{A}^T\mathbf{A})\mathbf{V} &= \mathbf{V}(\mathbf{\Sigma}^T\mathbf{\Sigma}) = \mathbf{V}\mathbf{D}
\end{aligned}
$$

Left svectors are evecs of A A^T

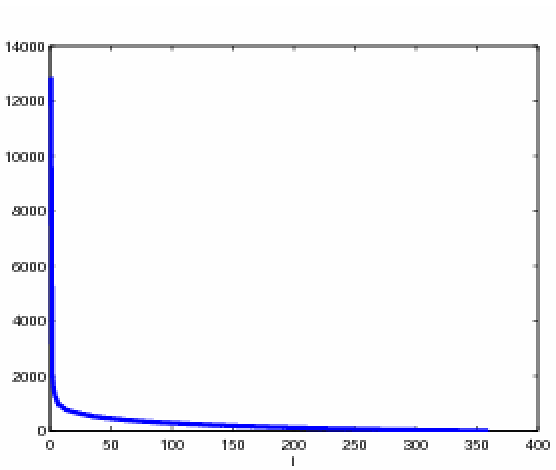
$$
\begin{aligned}
\mathbf{A}\mathbf{A}^T &= \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T\,\mathbf{V}\mathbf{\Sigma}^T\mathbf{U}^T \\
&= \mathbf{U}(\mathbf{\Sigma}\mathbf{\Sigma}^T)\mathbf{U}^T \\
(\mathbf{A}\mathbf{A}^T)\mathbf{U} &= \mathbf{U}(\mathbf{\Sigma}\mathbf{\Sigma}^T) = \mathbf{U}\mathbf{D}
\end{aligned}
$$

Truncated SVD

$$\mathbf{A} \;=\; \mathbf{U}_{:,1:k}\boldsymbol{\Sigma}_{1:k,1:k}\mathbf{V}^T_{1:,1:k} = \lambda_1 \begin{pmatrix} | \\ \mathbf{u}_1 \\ | \end{pmatrix} \begin{pmatrix} - & \mathbf{v}_1^T & - \end{pmatrix} + \cdots + \lambda_k \begin{pmatrix} | \\ \mathbf{u}_k \\ | \end{pmatrix} \begin{pmatrix} - & \mathbf{v}_k^T & - \end{pmatrix}$$

Rank k approximation to matrix

Spectrum of singular values

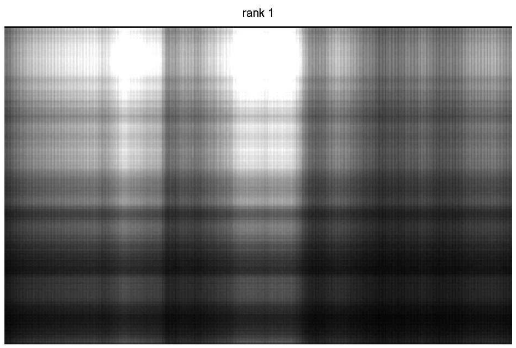


SVD on images

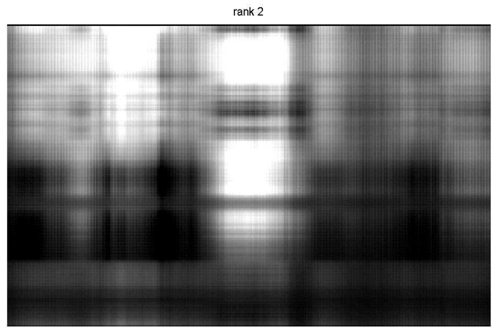
- Run demo

```
load clown
[U,S,V] = svd(X,0);
ranks = [1 2 5 10 20 rank(X)];
for k=ranks(:)'
  Xhat = (U(:,1:k)*S(1:k,1:k)*V(:,1:k)');
  image(Xhat);
end
```

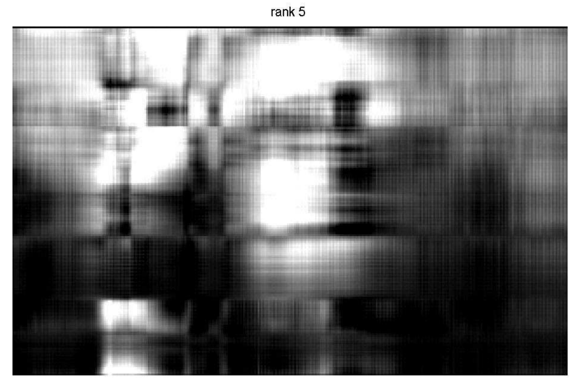
Clown example



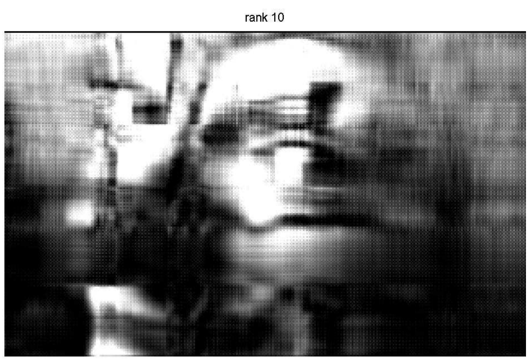
1



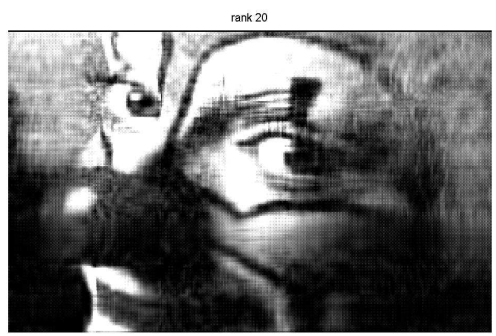
2



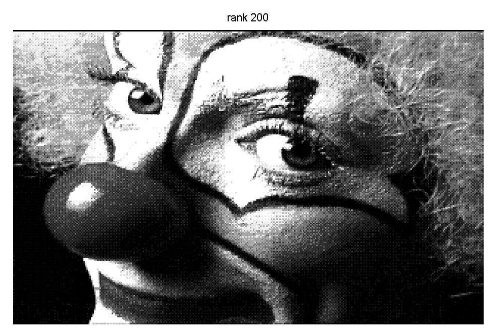
5



10



20



200

Space savings

$$\begin{aligned} \mathbf{A} &\approx \mathbf{U}_{:,1:k}\mathbf{\Sigma}_{1:k,1:k}\mathbf{V}^T_{1:,1:k} \\ m \times n &\approx (m \times k)\ (k)\ (n \times k) = (m+n+1)k \\ 200 \times 320 = 64,000 &\rightarrow (200+320+1)20 = 10,420 \end{aligned}$$