EE16B — Midterm 2 Review

George Higgins Hutchinson, Parth Nobel, *et al.*

April 12, 2019

Disclaimer

This is an unofficial review session and HKN is not affiliated with this course. All of the topics we're reviewing will reflect the material you have covered, our experiences in EE16B, and past exams. We make no promise that what we cover will necessarily reflect the content of this midterm. While some course staff members may be among the presenters, this review session is still not official.

Overview

PCA

Discretization

SVD Theorem

Any matrix $A \in \mathbb{R}^{m \times n}$ can be decomposed into the product of three matrices

$$\begin{aligned} A &= U \Sigma V^T \\ U &: m \times m \\ \Sigma &: m \times n \\ V^T &: n \times n \end{aligned}$$

Such that U, V are unitary matrices and Σ only has nonnegative values along its main diagonal.

SVD: Compact Form

We can also express the SVD as

$$A = \mathcal{U} S \mathcal{V}^T$$
$$\mathcal{U} : m \times r$$
$$S : r \times r$$
$$\mathcal{V}^T : r \times n$$

where r is the rank of A. The compact form matrices maintain properties of the original matrices, but have entries removed whenever they correspond to zero singular values.

SVD: Outer Product Form

Lastly, we can express

$$A = \sum_{i=1}^{r} \sigma_i \vec{u}_i \vec{v}_i^T$$

where $\vec{u}_i, \vec{v}_i$ are the columns of U, V, respectively, and σ_i are corresponding diagonal entry of the matrix Σ

Computing SVD with $A^T A$

$$\begin{aligned} A^T A &= U \Sigma V^T V \Sigma^T U^T \\ &= U \Sigma^2 U^T \end{aligned}$$

This is an eigen decomposition since Σ^2 is diagonal and $U^{-1} = U^T$. Thus solving for the eigenvalues and eigenvectors of $A^T A$ give $\lambda_i = \sigma_i^2$ with eigenvectors which correspond to the right singular vectors. We need to sort by decreasing σ_i.

Side note: $\Sigma^T \Sigma$ is not actually equal to Σ^2, but the former product yields a matrix with singular values squared on the diagonal entries, hence we call it Σ^2

Computing SVD with $A^T A$

Given a right singular vector $\vec{v}_i$ which we found from the previous part, we can apply it

$$
\begin{aligned}
A\vec{v}_i &= \left(\sum_{k=1}^{r} \sigma_k \vec{u}_k \vec{v}_k^T\right) \vec{v}_i \\
&= \sum_{k=1}^{r} \sigma_k \vec{u}_k \vec{v}_k^T \vec{i} \\
&= \sigma_i \vec{u}_i \\
\vec{u}_i &= \frac{1}{\sigma_i} A\vec{v}_i
\end{aligned}
$$

Computing SVD with AA^T

Similar calculations yield $\sigma_i = \sqrt{\lambda_i}$ of AA^T with eigenvectors as left singular vectors, and $\vec{v}_i = \frac{1}{\sigma_i} A^T \vec{u}_i$

Intepretation of SVD

- Unitary matrices act as rotation in a given space. A diagonal matrix stretches in a given coordinate space.
- SVD visualization (open in browser)

Intepretation of SVD

For a product $A\vec{x}$, we can decompose every vector $\vec{x}$ into a linear combination of right singular vectors

$$\vec{x} = \sum_{i=1}^{n} \alpha_i \vec{v}_i$$

Thus, we can see exactly which parts of $\vec{x}$ affect the output.

Compression of Low-Rank Matrices

- Suppose I had a matrix $A \in \mathbb{R}^{m \times n}$ with $m, n >> rank(A)$. How could I more efficiently store A and compute products like $A\vec{x}$?

- With the SVD, we only have to save r set of two vectors and a scalar, which saves us a lot of space if the rank is small with respect to the matrix. Also, less computation is carried out if we represent the matrix as the outer product form.

Compression of Low-Rank Matrices

- Suppose I had a matrix $A \in \mathbb{R}^{m \times n}$ with $m, n >> rank(A)$. How could I more efficiently store A and compute products like $A\vec{x}$?
- With the SVD, we only have to save r set of two vectors and a scalar, which saves us a lot of space if the rank is small with respect to the matrix. Also, less computation is carried out if we represent the matrix as the outer product form.

Overview

PCA

Discretization

PCA

PCA is a linear dimensionality reduction tool. Given data $\vec{x}_i \in \mathbb{R}^d$, we can create a mapping $T : \mathbb{R}^d \to \mathbb{R}^{d'}, d' < d$ such that the variance in the dataset is still captured

PCA — Computation

1. Store data row-major in $A \in \mathbb{R}^{n \times d}$
2. De-mean A
3. Take SVD: $A = U \Sigma V^T$
4. Create $V_{d'} \in \mathbb{R}^{n \times d'}$ from vectors of V corresponding to d' greatest signular values
5. To project data into the representative subspace: $T(x) := V_{d'}^T x$

PCA — Computation

1. Store data row-major in $A \in \mathbb{R}^{n \times d}$
2. De-mean A
3. Take SVD: $A = U \Sigma V^T$
4. Create $V_{d'} \in \mathbb{R}^{n \times d'}$ from vectors of V corresponding to d' greatest signular values
5. To project data into the representative subspace: $T(x) := V_{d'}^T x$

PCA — Computation

1. Store data row-major in $A \in \mathbb{R}^{n \times d}$
2. De-mean A
3. Take SVD: $A = U \Sigma V^T$
4. Create $V_{d'} \in \mathbb{R}^{n \times d'}$ from vectors of V corresponding to d' greatest signular values
5. To project data into the representative subspace: $T(x) := V_{d'}^T x$

PCA — Computation

1. Store data row-major in $A \in \mathbb{R}^{n \times d}$
2. De-mean A
3. Take SVD: $A = U\Sigma V^T$
4. Create $V_{d'} \in \mathbb{R}^{n \times d'}$ from vectors of V corresponding to d' greatest signular values
5. To project data into the representative subspace: $T(x) := V_{d'}^T x$

PCA — Computation

1. Store data row-major in $A \in \mathbb{R}^{n \times d}$
2. De-mean A
3. Take SVD: $A = U\Sigma V^T$
4. Create $V_{d'} \in \mathbb{R}^{n \times d'}$ from vectors of V corresponding to d' greatest signular values
5. To project data into the representative subspace: $T(x) := V_{d'}^T x$

PCA: computation

The mapping T can then be expressed as

$$T(\vec{x}) = V_k^T \vec{x}$$

If we apply this transformation onto the entire dataset (which has row vectors), we can say

$$T(A) = B = AV_k$$

where $B \in \mathbb{R}^{n \times k}$

PCA: computation

If we were to show the projected vectors in the original space, we can multiply back with the projection vectors

$$A' = BV_k^T$$

Overview

Discretization: Q1

Note: this section follows hw8 q1 almost exactly. Suppose we have a scalar system

$$\frac{d}{dt}x(t) = \alpha x + \vec{\beta}^T \vec{u}(t)$$

and we apply a constant input $\vec{u}_n$ for times $t \in [nT, (n+1)T)$ for some $T > 0$. Given $x(nT)$ solve the differential equation

Discretization: Q1 Sol

From $t = nT$ to $t = (n+1)T$, $\vec{\beta}^T \vec{u}$ is a constant scalar. Thus, we can solve this like a normal differential equation. Let $x = x' - \frac{\vec{\beta}^T \vec{u}}{\alpha}$. Then

$$\begin{aligned}
\frac{d}{dt}x(t) &= \alpha(x' - \frac{\vec{\beta}^T \vec{u}}{\alpha}) + \vec{\beta}^T \vec{u}(t) \\
&= \alpha x' \\
x' &= Ae^{\alpha(x-nT)} \\
x + \frac{\vec{\beta}^T \vec{u}}{\alpha} &= Ae^{\alpha(x-nT)} \\
x &= Ae^{\alpha(x-nT)} - \frac{\vec{\beta}^T \vec{u}}{\alpha}
\end{aligned}$$

At which point we can use our initial condition to get

$$x(nT) = A - \frac{\vec{\beta}^T \vec{u}}{\alpha}$$

Discretization: Q2

Using the differential equation derived from question 1, create a discrete-time system to model the continuous time. In other words, if $x[n] = x(nT), \vec{u}[n] = \vec{u}(nT)$, find a relation such that

$$x[n+1] = A_d x[n] + B_d \vec{u}[n]$$

Discretization: Q2 Sol

We can solve the previous solution for $x((n+1)T)$

$$x((n+1)T) = \left(x(nT) + \frac{\vec{\beta}^T \vec{u}(nT)}{\alpha}\right) e^{\alpha((n+1)T - nT)} - \frac{\vec{\beta}^T \vec{u}(nT)}{\alpha}$$

$$x[n+1] = e^{\alpha T} x[n] + \frac{e^{\alpha T} - 1}{\alpha} \vec{\beta}^T \vec{u}[n]$$

We see that $A_d = e^{\alpha T}, B_d = ((e^{\alpha T} - 1)/\alpha)\vec{\beta}^T$

Discretization: Q3

Instead of a scalar, we instead have a diagonal matrix A such that

$$\frac{d}{dt}\vec{x} = A\vec{x} + B\vec{u}$$

Discretize this system in the same was as Q2.

Discretiziation: Q3 Sol

Expanding the original system out line-by-line gives

$$\frac{d}{dt}x_i = a_i x_i + b_i \vec{u}_i$$

where x_i is the ith variable of $\vec{x}$, a_i is the diagonal entry of A, and b_i is the row of B.

Discretization: Generic Matrix

Math not shown, but we can perform a change of basis from our original space to our diagonal space, and then apply the results of the previous part.