Lecture 6: Unit 3 Recap, Unit 4, Unit 5 Slides

6.1 Unit 3 - SVD Notes (on whiteboard)

For an \( m \times n \) matrix \( A \), we have

\[
A = U \Sigma V^T
\]

\[
UU^T = U^T U = I_{m \times m}
\]

\[
VV^T = V^T V = I_{n \times n}
\]

\( \Sigma \in \mathbb{R}^{m \times n} \) diagonal (positive/real entries)

Note that the orthonormal matrices \( U \in \mathbb{R}^{m \times m} \) and \( V \in \mathbb{R}^{n \times n} \) preserve the \( L_2 \) norm. Also note that the diagonal entries in \( \Sigma \) are positive and real, because they are the square roots of the eigenvalues of the symmetric matrix \( AA^T \) and \( A^T A \). Those matrices are symmetric and have real eigenvalues. Technically, one can negate the sign of a singular value in \( \Sigma \) and correspondingly change \( U \), and the equation \( A = U \Sigma V^T \) still holds true. Thus, there is some flexibility in how one can manipulate things with SVD. Changing signs and swapping entries in \( \Sigma \) and \( U, V \) lead to equivalent forms of SVD. However, if one arranges the (positive real) diagonal entries of \( \Sigma \) in descending order, then one can obtain a unique SVD.

6.2 Unit 3 - Matrix Norms

- \( L_2 \), the 2-norm of a matrix, is important: \( \|A\|_2 \) is the largest singular value of \( A \), or the square root of the maximum eigenvalue of \( A^T A \).

- **All norms are not equivalent.** Norms are equivalent only for proving some theory (e.g., functions are bounded from below or above). In reality, the specific choice of norm often matters. For example, solving the least squares problem with different norms leads to different results.

- \( L_\infty \), the infinity norm, is also an important one. For example, when measuring error, it bounds the biggest error one can make.

6.3 Unit 3 - \( A^{-1} \) and Condition Number of \( A \)

Using the SVD \( A = U \Sigma V^T \), one can write \( A(V \Sigma^{-1} U^T) = U \Sigma V^T V \Sigma^{-1} U^T = U \Sigma \Sigma^{-1} U^T = U U^T = I \), so when the inverse of \( A \) exists, it is \( A^{-1} = V \Sigma^{-1} U^T \). That is, in the SVD of \( A^{-1} \), \( V \) and \( U \) simply switches places. The biggest singular value of \( A^{-1} \) is the biggest value in \( \Sigma^{-1} \), and thus \( 1/\sigma_{\min} \). Therefore, the condition number in 2-norm of \( A \) is \( \|A\|_2 \|A^{-1}\|_2 = \sigma_{\max}/\sigma_{\min} \).
6.4 Unit 4 - Symmetric Positive Definite and Semi-Definite Matrices

- Symmetry is important.
- Positive definite is important - equivalent to having all positive eigenvalues. This means that there is no flipping of axes (negative eigenvalue flips a vector).
- If a matrix is symmetric and positive definite (SPD), one can do many things to it.
- Symmetric positive semi-definite matrices have nonnegative eigenvalues, meaning that they have a null space. One can solve $Ax = b$ by projecting $b$ to the range of $A$, and then solve for that. $U\Sigma V^T x = b$ as $\Sigma V^T x = U^T b$, or $\Sigma \hat{x} = b$. For example, if we have

$$
\Sigma \hat{x} = \begin{bmatrix} 5 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \hat{x}_1 \\ \hat{x}_2 \\ \hat{x}_3 \end{bmatrix} = \begin{bmatrix} \hat{b}_1 \\ \hat{b}_2 \\ \hat{b}_3 \end{bmatrix} = \hat{b},
$$

then

$$
\begin{bmatrix} 5\hat{x}_1 \\ 4\hat{x}_2 \\ 0 \end{bmatrix} = \begin{bmatrix} \hat{b}_1 \\ \hat{b}_2 \\ \hat{b}_3 \end{bmatrix}.
$$

If $\hat{b}_3 \neq 0$, then $b = U\hat{b}$ is not in the range of $A$ (since the third column of $U$ is not in the range of $A$).

Many methods that can be applied to SPD matrices can also be applied to symmetric positive semi-definite matrices. But they do not apply to indefinite matrices with mixed positive and negative eigenvalues.

6.5 Unit 4 - Cholesky Factorization 2 × 2 Example Errata

In Cholesky factorization, we should write

$$A = \begin{bmatrix} a_{11} & a_{21} \\ a_{21} & a_{22} \end{bmatrix}$$

since $A$ is symmetric and $a_{12} = a_{21}$.

6.6 Unit 4 - Preconditioner Discussion

Sometimes one simply wants a very fast way to solve a problem, and may just symmetrize a matrix $A$ and solve the approximate problem $((A + A^T)/2)x = b$ instead of $Ax = b$. This is often useful to get a quick approximate answer that can help inform time-sensitive decision making process (such as self-driving cars). Other times one wants to improve a solution, and may want to use a preconditioner to solve the problem $\hat{A}x = (((A + A^T)/2)^{-1}A)x = (((A + A^T)/2)^{-1}b = \hat{b}$ exactly, where $((A + A^T)/2)^{-1}A$ gets closer to the identity (exactly the identity if $A$ is symmetric).
6.7 Unit 5 Slides - Collision Detection using Spheres

Roboticists often use a collection of spheres to model a volumetric object, see Figure 1. For collision checking, if a point is outside of all spheres, it is outside the object, otherwise it is inside. This has the benefit of efficiency, since checking inside/outside for spheres (computing distance to its center and comparing to its radius) is very easy and fast.

![Figure 1](image1.png)

Figure 1: One can approximate the planar surface for collision detection with a sequence of spheres.

6.8 Unit 5 Slides - Collision Detection for Cloth/Shells

The challenge is that for thin surfaces or shells with negligible thickness, one can almost never see something inside. For example, consider a marker colliding with a sheet of paper in Figure 2.

![Figure 2](image2.png)

Figure 2: The marker (shown as rectangular block here) goes through a paper (shown in vertical segment here) without any vertex inside the paper. Even if one thickes the paper a little bit (shown in green), the paper is still too thin for any marker vertex to be inside, and collision detection fails.

The problem is even harder for two thin surface objects colliding with each other. The solution is to do continuous collision detection, as is done in real world: look at the path of the vertices. If the path intersects the surface, then we know that it hits the surface. One can draw a straight line from $x_{\text{old}}$ to $x_{\text{new}}$ and see if it intersects the thin surface; note that if the trajectory is a curved path, then one should discretize it to small pieces of straight line segments for an accurate prediction. See Figure 3.
6.9 Unit 5 Slides - Iterative Cubic Equation Solver

The exact formula for solving cubic equations does not give enough precision for detecting cloth self-collisions. The solution was to implement an iterative cubic equation solver in double precision carefully, and this made the self-collisions work. It is very important to make something efficient and robust so that people can use it, just like Thomas Edison “invented” the light bulb by making it last longer and actually usable.

6.10 Unit 5 - Iterative Methods

- One can combine direct and iterative methods - use the result from the direct solver as a starting point and keep iterating.

- Sparse matrices often have dense inverses and dense LU factorizations - you don’t want to factor them or solve directly!