Geometric Methods in Data Science

George D. Torres

Fall 2021
Prof. Joe Kileel

1 High Dimensional Geometry **2**
1.1 The Sphere and the Ball . . . 2
1.2 Probability Review . . . 4
1.3 Hoeffding's Inequality . . . 7

2 Matrix Factorizations **9**
2.1 Singular Value Decomposition . . . 9
2.2 Principal Component Analysis . . . 11

3 Clustering **15**
3.1 k-means clustering . . . 15
3.2 Spectral Clustering . . . 15

4 Diffusion Maps **19**
4.1 Relationship to Manifold Learning . . . 20

5 Convex Relaxations and Semidefinite Programming **23**
5.1 Max-Cut . . . 23
5.2 Semidefinite Programs . . . 25
5.3 Duality of SDPs . . . 27
5.4 Interior Point Algorithms . . . 29
5.5 Sums of Squares Problems and SDPs . . . 30

6 Statistical Parameter Estimation **34**
6.1 Maximum Likelihood Estimation and Method of Moments . . . 34

7 Tensor Decomposition **36**

Last updated November 9, 2021

1. High Dimensional Geometry

❖

Lecture 8/26

Encountering datasets that live in Euclidean space $\mathbb{R}^d$ where d very large is common in data science, for example image data. The *curse of dimensionality* is a common phrase to describe that many algorithms that involve computation in $\mathbb{R}^d$ have cost that scales exponentially in d.

Example 1.1. Forming a grid for $[0,1]^d \subset \mathbb{R}^d$ with fixed density. Fixing the density (say $1/100$) means that we need to discretize each coordinate with our specified spacing, and hence will need 100^d points to form the grid. This is a very large value, especially considering that d is already large. This shows how grids are almost inaccessible in high dimensions.

1.1 The Sphere and the Ball ❖

Two important objects in all dimensions are the unit ball and cube. When d is large, these two objects have some counterituitive properties.

Definition 1.2. Let $R > 0$. The d-ball of radius R in $\mathbb{R}^d$ is $B^d(R) = \{x \in \mathbb{R}^d \mid \|x\|_2 \leq R\}$. Its boundary S^{d-1} is the $d-1$ sphere in $\mathbb{R}^d$. The unit cube of size R is $C^d(R) = [-R, R]^d$. We denote $B^d := B^d(1)$, $C^d := C^d(1)$ and $S^d := S^d(1)$.

Clearly $B^d \subset C^d$. How much bigger is the cube than the ball for general d? Looking at a picture of $B^2 \subset C^2$, one might guess that they are comparable in volume, even for higher d. However, this is very much not the case, as we can see by calculating their respective volumes. The volume of C^d is 2^d and the volume of $B^d(R)$ is of the form αR^d for some α. This constant $\alpha = \alpha(d)$ determines the proportion of volume between these two objects.

Proposition 1.3. $\alpha(d) = \frac{\pi^{d/2}}{\frac{d}{2}\Gamma(d/2)}$, *where* Γ *is the Gamma-function:*

$$\Gamma(z) = \int_0^\infty x^{z-1}e^{-x}\,dx = 2\int_0^\infty u^{2z-1}e^{-u^2}du, \qquad x \in \mathbb{R}_{\geq 0}$$

This function is a real extension of the factorial function, i.e. $\Gamma(n) = (n-1)!$ *for* $n \in \mathbb{N}$. *The two forumulae above are equivalent by the substitution* $u^2 = x$.

Proof:

The volume of $B^d(R)$ can be calculated by using shells:

$$\begin{aligned}\text{vol}(B^d(R)) &= \int_0^R \text{Surface Area}(S^{d-1}(r))\,dr \\ &= \int_0^R \text{Surface Area}(S^{d-1})r^{d-1}\,dr \\ &= \text{Surface Area}(S^{d-1})\frac{R^d}{d}\end{aligned}$$

The trick to calculate the surface area of the unit sphere is:

$$\begin{aligned}\text{Surface Area}(S^{d-1})\int_0^\infty e^{-r^2}r^{d-1}\,dr &= \int_0^\infty e^{-r^2}\text{Surface Area}(S^{d-1}(r))\,dr \\ &= \int_{-\infty}^\infty \cdots \int_{-\infty}^\infty e^{-(x_1^2+\ldots+x_d^2)}\,dx_1\cdots dx_d \\ &= \left(\int_{-\infty}^\infty\right)^d = \pi^{d/2}\end{aligned}$$

We used the fact that Surface Area$(S^{d-1}(r))$ is radially symmetric. Now we rearrange for Surface Area(S^{-1}), using the formula for the Gamma function. This yields the final desired formula for the volume:

$$\text{vol}(B^d(R)) = \frac{\pi^{d/2}}{\frac{d}{2}\Gamma(d/2)}R^d$$

□

Now we compare the volumes of the cube and ball. We can use Stirling's approximation to understand the asymptotics of Γ:

$$\Gamma(z) \sim \sqrt{\frac{2\pi}{z}}\left(\frac{z}{e}\right)^z$$

This gives the asymptotic:

$$\text{vol}(B^d(R)) \sim \frac{1}{\sqrt{d\pi}}\left(\frac{2\pi e}{d}\right)^{d/2}R^d$$

For $R = 1$ and d large, this term goes to zero very fast. Meanwhile, for the cube of radius 1, its volume is 2^d, which increases to infinity as d gets large. Therefore $\text{vol}(B^d)/\text{vol}(C^d) \to 0$ as $d \to \infty$. In fact, even $\text{vol}(B^d)/\text{vol}(C^d(1/2)) \to 0$ as $d \to \infty$. If we wish to choose R such that $\text{vol}(B^d(R)) \approx \text{vol}(C^d(1/2))$, we would need:

$$R^d\frac{1}{\sqrt{d\pi}}\left(\frac{2\pi e}{d}\right)^{d/2} \approx 1 \Rightarrow R\sqrt{\frac{2\pi e}{d}} \approx 1 \Rightarrow R = \Omega(\sqrt{d})$$

Remark 1.4. The notation $\Omega(\sqrt{d})$ means there exists a constant $C > 0$ such that $R \geq C\sqrt{d}$.

Some other surprising facts in high dimensions are:

- Almost all of the volume of B^d is near the equator (or indeed *any* equator!). To see why, we can assume WLOG that the equator is $\{x \in \mathbb{R}^d \mid \|x\|_2 \leq 1, x \perp e_1\}$. Then define the polar cap $P = \{x \in \mathbb{R}^d \mid \|x\|_2 \leq 1, x_1 \geq p_0\}$. We would like to see why $\text{vol}(P)$ is small compared to $\text{vol}(B^d)$. The volume of a cap can be calculated using standard calculus:

$$\begin{aligned}\text{vol}(P) = \int_{p_0}^1 \text{vol}(B^{d-1}(\sqrt{1-p^2}))\,dp &= \text{vol}(B^{d-1})\int_{p_0}^1 (\sqrt{1-p^2})^{d-1}\,dp \\ &\leq \text{vol}(B^{d-1})\int_{p_0}^1 e^{-(d-1)p^2/2}\,dp \\ &\leq \text{vol}(B^{d-1})\int_{p_0}^\infty e^{-(d-1)p^2/2}\frac{p}{p_0}\,dp \\ &= \frac{\text{vol}(B^{d-1})}{d-1}\frac{e^{-(d-1)p_0^2/2}}{p_0}\end{aligned}$$

using the bound $1-p^2 \leq e^{-p^2}$. The last step used integration by substitution. Using the fact that $\frac{\text{vol}(B^{d-1})}{\text{vol}(B^d)} \leq \frac{d-1}{2}$, then we get:

$$\frac{2\text{vol}(P)}{\text{vol}(B^d)} \leq e^{-(d-1)p_0^2}$$

This shows that the two polar caps have vanishingly small proportion of total volume in the ball when we choose $p_0 = \Omega(1/\sqrt{d})$ (with sufficiently small constant).

- Most of the volume of B^d is near the boundary for sufficiently high d. To see this, compare the volumes of two balls of slightly different radius:

$$\frac{\text{vol}(B^d(1-\epsilon))}{\text{vol}(B^d(1))} = (1-\epsilon)^d$$

Now choose $\epsilon = t/d$ for a constant t. Then $(1-t/d)^d \to e^{-t}$. This can be made arbitrarily small for whatever t we choose.

Lecture 8/31

1.2 Probability Review ❖

A *random variable* is a quantity that depend on some amount of chance (for example the number of heads you get after flipping a coin N times). In this course, it will almost always be enough to think intuitively about random variables. Nevertheless, we state a formal definition:

Definition 1.5. Let $(\Omega, \Sigma, \mathbb{P})$ be a probability space. Let $\mathcal{B}$ be the Borel algebra on $\mathbb{R}$. Then a *random variable* is a measurable function $X : \Omega \to \mathbb{R}$ (meaning for all $B \in \mathcal{B}$ the preimage $X^{-1}(B) \in \Omega$). We say the probability X lies in B is $\mathbb{P}(X^{-1}(B))$ and denote this by $\mathbb{P}(X \in B)$.

Definition 1.6. A *continuous* random variable is one for which there exists a function $p_X : \mathbb{R} \to \mathbb{R}_{\geq 0}$ such that $\mathbb{P}(a \leq X \leq b) = \int_a^b p_X(t)dt$ (i.e. a probability density function). A *discrete* random variable is one for which there exists a set $\{x_1, x_2, ...\} \subset \mathbb{R}$ of values and a set of probabilities $\{p_1, p_2, ...\}$ such that $\mathbb{P}(X = x_i) = p_i$ and $\sum p_i = 1$ (i.e. a probability mass function).

Exercise 1.7. Let $h : \mathbb{R} \to \mathbb{R}$ be a measurable function and X a random variable. State why the composition $h \circ X : \Omega \to \mathbb{R}$ is a random variable. If X is continuous, is $h \circ X$ continuous? What about the discrete case?

The most basic thing to ask about a random variable is its average value or *expectation*:

$$\mathbb{E}[X] := \begin{cases} \int_{-\infty}^{\infty} t p_X(t)\, dt & \text{if } X \text{ is continuous} \\ \sum x_i p_i & \text{if } X \text{ is discrete} \end{cases}$$

The next most basic thing to ask about a random variable is how spread out it is. This is precisely measured by the *variance*:

$$\text{Var}(X) := \mathbb{E}[(X - \mathbb{E}(X))^2] = \mathbb{E}[X^2] - \mathbb{E}[X]^2$$

The standard deviation is defined to be $\sigma(X) := \sqrt{\text{Var}(X)}$. For any $\mathbb{R}$-valued random variable, the *cumulative distribution function* (or CDF) of X is $F_X(t) := \mathbb{P}(X \leq t)$. The tail of X is $t \mapsto \mathbb{P}(|X| \geq t)$.

Proposition 1.8 (Markov's Inequality)**.** *Let X be any non-negative random variable. Then $\mathbb{P}(X \geq t) \leq \mathbb{E}[X]/t$ for all $t > 0$.*

Proof:

Assume X is continuous. Then $\mathbb{E}[X] = \int_0^\infty s p_X(s)\, ds$ because X is non-negative. This same property

also yields:

$$\begin{aligned}\mathbb{E}[X] &\geq \int_t^\infty s p_X(s)\, ds \\ &\geq \int_t^\infty t p_X(s)\, ds = t\int_t^\infty p_X(s) ds = t\mathbb{P}(X \geq t)\end{aligned}$$

□

Corollary 1.9 (Chebyshev's Inequality)**.** *Let X be any random variable with mean μ and standard deviation σ. Then for any $k \in \mathbb{R}_{>0}$ we have $\mathbb{P}(|X-\mu| \geq k\sigma) \leq \frac{1}{k^2}$.*

Proof:

Let $Y := (X-\mu)^2$, which is a non-negative random variable. The expectation of Y is $\mathrm{Var}(X) = \sigma(X)^2$. Now applying Markov's inequality: $\mathbb{P}(Y \leq t) \leq \mathbb{E}[Y]/t = \sigma^2/t$. Putting $t = (k\sigma)^2$, we get $\mathbb{P}((X-\mu)^2 \geq (k\sigma)^2) \leq \sigma^2/k^2\sigma^2 = 1/k^2$. The event $(X-\mu)^2 \geq (k\sigma)^2$ is the same as the event $|X-\mu| \geq k\sigma$.

□

Markov gives a non-vacuous bound for tails provided $\mathbb{E}[X]$ is finite. We can say more if we assume more about its *higher moments*:

Definition 1.10. The nth moment of X is $\mathbb{E}[X^n]$ and the nth centered moment is $\mathbb{E}[(X-\mu)^n]$.

Definition 1.11. The *moment generating function* (mgf) of X is the function $\lambda \mapsto \mathbb{E}[e^{\lambda(X-\mu)}]$, which (by power expansion) is equivalent to:

$$\lambda \mapsto \sum_{n=0}^{\infty} \frac{\lambda^n}{n!} \underbrace{\mathbb{E}[(X-\mu)^n]}_{n\text{th moment}}$$

The moment generating function may not converge for all values of λ; however if all higher moments of X exist, then the mgf converges for all λ.

Proposition 1.12 (Chernoff bound)**.** *Let X be a random variable and assume the mgf exists and is finite for all $\lambda \in I$, where I is an interval containing 0. Then:*

$$\mathbb{P}((X-\mu) \geq t) \leq \inf_{\lambda \in I} \mathbb{E}[e^{\lambda(x-\mu)}]/e^{\lambda t}$$

Proof:

Let $\lambda \in I$ and $Y_\lambda = e^{\lambda(X-\mu)}$. This is a non-negative random variable, and we can apply Markov's inequality. The expected value of Y_λ is the mgf of X at lambda. Thus

$$\mathbb{P}((X-\mu) \geq t) = \mathbb{P}(Y_\lambda \geq e^{\lambda t}) \leq \mathbb{E}[Y_\lambda]/e^{\lambda t}$$

This was true for any $\lambda \in I$, so this bound also holds over taking the infimum in I. This yields the desired bound.

□

Example 1.13. The 1-D *Gaussian* random variable with mean μ and standard deviation σ has pdf

$$\psi(t) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{(-(t-\mu)^2/2\sigma^2}$$

We write $X \sim N(\mu, \sigma^2)$. If $\mu = 0$ and $\sigma = 1$, we say that X is a standard Gaussian, or a standard Normal.

Lemma 1.14. *Let $X \sim N(0,1)$. Then $\mathbb{P}(X \geq t) \leq e^{-t^2/2}$ for all $t > 0$. More generally, for $X \sim N(\mu, \sigma^2)$ we have $\mathbb{P}(X \geq \mu + t) \leq e^{t^2/2\sigma^2}$.*

Proof:

Exercise (*Hint*: calculate the mgf directly, and then apply the Chernoff bound)

□

It is useful to consider random variables which are "dominated" by some Gaussian, because then they must also have rapidly decaying tails too.

Definition 1.15. A random variable X with mean μ is *sub-Gaussian* with parameter σ if $\mathbb{E}[e^{\lambda(X-\mu)}] \leq e^{\sigma^2\lambda^2/2}$ for all $\lambda \in \mathbb{R}$. In other words, the mgf of X is bounded by the mgf of a Gaussian with mean μ and variance σ^2.

Example 1.16. The following are examples of sub-Gaussian random variables:

- Any Gaussian.
- A bounded random variable (i.e. one whose tails vanish past some finite value).
- A Rademacher random variable ϵ. This is a binary random variable defined by $\mathbb{P}(\epsilon = 1) = \frac{1}{2} = \mathbb{P}(\epsilon = -1)$. This has mean zero, and hence:
$$\mathbb{E}[e^{\lambda\epsilon}] = \frac{1}{2}(e^{\lambda} + e^{-\lambda})$$
Now we write the Taylor series expansions:
$$\begin{aligned}\mathbb{E}[e^{\lambda\epsilon}] &= \frac{1}{2}\left(\sum_{n=0}^{\infty}\frac{\lambda^n}{n!} + \sum_{n=0}^{\infty}\frac{(-\lambda)^n}{n!}\right)\\ &= \sum_{m=0}^{\infty}\frac{\lambda^{2m}}{(2m)!}\\ &\leq \sum_{m=0}^{\infty}\left(\frac{\lambda^2}{2}\right)^m\frac{1}{m!} = e^{\lambda^2/2}\end{aligned}$$
This demonstrates that it is sub-Gaussian.

A type of random variable that comes up often is an emperical average, meaning a sum of independent identically distributed (i.i.d.) random variables.

Definition 1.17. A collection of random variables $X_1, X_2, ...$ is called *independent and identically distributed* (iid) if:
$$\mathbb{P}(X_1 \leq t_1, X_2 \leq t_2, ..., X_n \leq X_n) = \mathbb{P}(X_1 \leq t_1) \cdot ... \cdot \mathbb{P}(X_n \leq t_n), \quad \forall t_1, ..., t_n \in \mathbb{R}$$
$$\mathbb{P}(X_i \leq t) = \mathbb{P}(X_j \leq t), \quad \forall t \in \mathbb{R}, \forall i, j$$

There are lots of results about these types of random variables. Several of the main ones are:

Theorem 1.18 (Strong law of large numbers)**.** *Let $X_1, X_2, ...$ be a sequence of i.i.d random variables with mean μ. Let $S_n/n \to \mu$ as $n \to \infty$ almost surely. Almost surely means:*
$$\mathbb{P}(S_n/n \to \mu \text{ as } n \to \infty) = 1$$

Theorem 1.19 (Central limit theorem)**.** *Let $X_1, X_2, ...$ be iid random variables with mean μ and variance σ^2. Let $S_n = X_1 + ... + X_n$ and*
$$Z_n = \frac{S_n - \mathbb{E}[S_n]}{\sqrt{\text{Var}(S_n)}} = \frac{1}{\sigma\sqrt{n}}\sum_{i=1}^{n}(X_i - \mu)$$
Then $Z_n \to \mathbb{N}(0, 1)$ as $n \to \infty$ in distribution.

Lecture 9/2

1.3 Hoeffding's Inequality ❖

The law of large numbers and central limit theorem are both asymptotic statements. In actual applications, we have only finitely many experiments or samples. How close is S_n/n to μ when n is finite? This is the topic of concentration inequalities. The goal is to bound the tail probability $\mathbb{P}(|S_n/n - \mu| > t)$ by something decaying fast with n. The first thing we can try is Chebyshev's inequality applied to the random variable S_n/n. Set $t := k\sqrt{\mathrm{Var}(S_n/n)}$, and then we have:

$$\begin{aligned}\mathbb{P}(|S_n/n - \mu| > t) &\leq \frac{1}{k^2} \\ &= \frac{\mathrm{Var}(S_n/n)}{t^2} \\ &= \frac{\frac{1}{n^2}\mathrm{Var}(S_n)}{t^2} \\ &= \frac{1}{n^2t^2}\left(\mathrm{Var}(X_1) + ... + \mathrm{Var}(X_n)\right) \\ &= \frac{\mathrm{Var}(X_1)}{nt^2}\end{aligned}$$

This gives us a very slowly vanishing bound on the tail probability. However, we can improve on this bound by assuming a bit more. For example, if the random variables are independent sub-Gaussians.

Proposition 1.20. *If X_i are independent sub-Gaussian random variables with means μ_i and parameters σ_i, then given any constants λ_i, the sum $\lambda_1 X_1 + ... + \lambda_n X_n$ is sub-Gaussian. Moreover, its mean is $\lambda_1\mu_1 + ... + \lambda_n\mu_n$ and paramter is $(\lambda_1^2\sigma_1^2 + ... + \lambda_n^2\sigma_n^2)^{1/2}$.*

Proof:

> Exercise in Homework 1.

□

Theorem 1.21. *Let $X_1, ..., X_n$ be independent sub-Gaussian random variables with mean 0 and parameters $\sigma_1, ..., \sigma_n$. Then*

$$\mathbb{P}\left(\frac{1}{n}\left|\sum_i X_i\right| > t\right) \leq e^{\frac{-n^2t^2}{2\sum_i \sigma_i^2}}$$

Proof:

> By the proposition above, we know that $\frac{1}{n}(X_1+...+X_n)$ is sub-Gaussian with parameter $\frac{1}{n}\sqrt{\sigma_1^2 + ... + \sigma_n^2}$. Now we plug this into the tail bound for sub-Gaussians.

□

Note that this gives exponential decay with n. An important and useful special case is Hoeffding's inequality:

Corollary 1.22 (Hoeffding inequality)**.** *Let $X_1, ..., X_n$ be independent variables such that $|X_i| \leq a_i$ for some a_i. Also assume that the means of X_i are all zero. Then*

$$\mathbb{P}\left(\left|\sum_i X_i\right| > t\right) \leq 2e^{\frac{-t^2}{2\sum_i a_i^2}}$$

This follows immediately because X_i are sub-Gaussian with parameters a_i. As an application of Hoeffding, we get the following informal result:

Theorem 1.23. *Almost all of the volume of $C^d(1)$ is located near the corners.*

Proof:

Let $X \in \mathbb{R}^d$ be a random vector uniformly distributed on $C^d(1)$. The coordinates x_i of X are iid uniform on $[-1, 1]$. The vectors $X \in C^d(1) \setminus B^d(1)$ can be considered as "near the corner." We will then show that $\mathbb{P}(X \in B^d(1)) = \mathbb{P}\left(\sum_i x_i^2 \leq 1\right)$ is small for large d. Note that x_i^2 is a bounded random variable, and that:

$$\mathbb{E}[x_i^2] = \int_{-1}^{1} t^2 \frac{1}{2}\, dt = \frac{1}{3}$$

We can equivalently shift everything by $\frac{1}{3}$ to get a mean of zero:

$$\mathbb{P}(X \in B^d(1)) = \mathbb{P}\left(\sum_i \left(x_i^2 - \frac{1}{3}\right) \leq 1 - \frac{d}{3}\right)$$

This is the half tail of bounded sub-Gaussian random variables with mean zero. We can then apply Hoeffding:

$$\mathbb{P}\left(\sum_i \left(x_i^2 - \frac{1}{3}\right) \leq 1 - \frac{d}{3}\right) \leq e^{\frac{-(1-d/3)^2}{2(d(2/3)^2)}} = e^{-\frac{1}{8}d}$$

This is very small when d is large.

□

Another important consequence of concentration is that the inner product between two random vectors in $\mathbb{R}^d$ is almost 0.

Theorem 1.24 ([BSS] Thm 2.19)**.** *Let X, Y be iid Rademacher vectors. Then*

$$\mathbb{P}\left(|\cos \angle(X, Y)| \geq \sqrt{\frac{2 \log d}{d}}\right) \leq \frac{2}{d}$$

There is a refinement of Hoeffding that uses more information than just the bounds a_i. It is called the Bernstein inequality, which we will not state here.

2. Matrix Factorizations

❖

Lecture 9/7

2.1 Singular Value Decomposition ❖

Given a matrix $A \in \mathbb{R}^{m \times n}$, a Singular Value Decomposition (SVD) is a decomposition $A = U\Sigma V^T$, where U, V are orthogonal square matrices and Σ is (pseudo)diagonal (and not necessarily square) with nonnegative entries. Pseudodiagonal means $\Sigma_{ij} = 0$ if $i \neq j$. Expanding this out, this is equivalent to:

$$A = \sum_{i=1}^{\min(m,n)} \sigma_i u_i v_i^T$$

where $\sigma_i = \Sigma_{ii}$ and u_i is the ith column of U and v_i is the ith column of V. We can reorder indices so that $\sigma_1 \geq \sigma_2 \geq \sigma_3 \geq ....$ The truncated SVD is defined as:

$$A^{(r)} = \sum_{i=1}^{r} \sigma_i u_i v_i^T$$

In the homework, we see that $A^{(r)}$ is the best rank $\leq r$ approximation to A with respect to various matrix distances. To store $A^{(r)}$, note that we only need to keep track of $\mathcal{O}(r(m+n))$ numbers.

Proposition 2.1. *The SVD exists for any matrix. If σ_i are distinct, the SVD is unique up to simultaneous sign flips on U and V. If they aren't distinct, it is unique up to multiplication by smaller orthogonal matrices.*

The cost to compute an SVD is $\mathcal{O}(\min(mn^2, m^2n))$, which is comparable to computing a matrix inverse. The cost of computing the truncated SVD is $\mathcal{O}(mnr)$. We will not go into the details of how SVDs are computed, however. In practice, if you have A and you want to compress it to $A^{(r)}$, part of the issue is choosing r appropriately. One way to do this is to look at the plot of the sequence $\{\sigma_i\}$ on a pair of axes[1] and look for an "elbow" (see Figure 2.2).

2.1.1 SVD for Image Compression

An application of the truncated SVD is to image compression. We can interpret a black and white image as a $n \times m$ matrix taking integer values in $[0, 255]$, and a color image as three such $n \times m$ matrices (one for each primary color channel). Calculating the truncated SVD has the effect of compressing the image, as seen in Figure 2.1. The scree plots for each of the color channels (on a log scale) is shown in Figure 2.2.

$$\text{Image} = (M_{\text{red}}, M_{\text{green}}, M_{\text{blue}})$$

$$\text{Compression} = (M_{\text{red}}^{(r)}, M_{\text{green}}^{(r)}, M_{\text{blue}}^{(r)})$$

[1] also called a scree plot

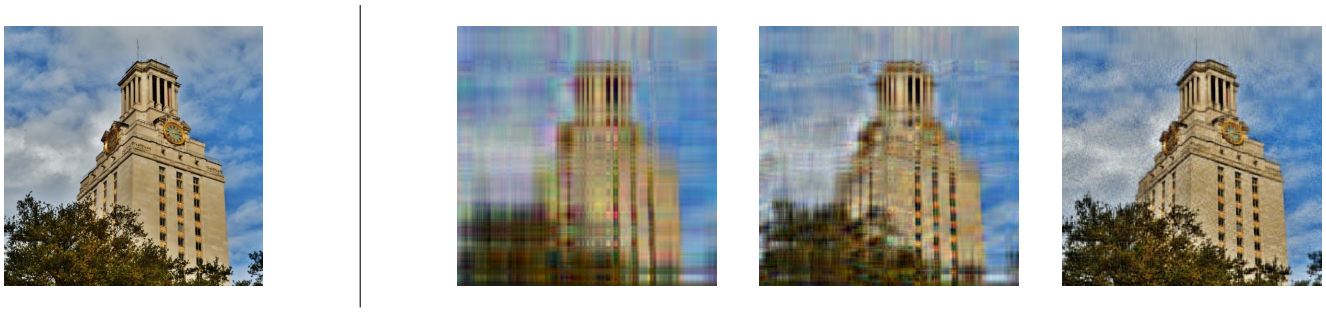

Figure 2.1: The truncated SVD for a color image.

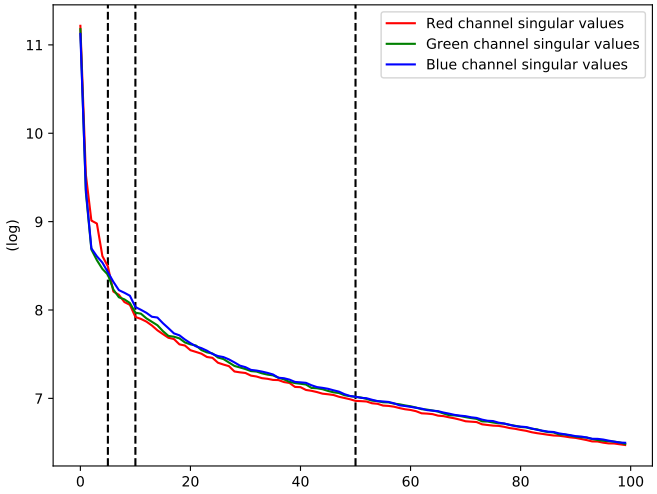

Figure 2.2: The scree plots for the SVD of the image in Figure 2.1 on a log scale. The dotted lines indicate where the three truncations in Figure 2.1 were.

2.2 Principal Component Analysis ❖

Principal Component Analysis (PCA) is a decomposition of a matrix that is equivalent to the eigendecomposition. Suppose we are given points $\{x_1, ..., x_n\} \subset \mathbb{R}^p$ (where we are thinking of p as very large). We would like to map these points to a lower dimension while still capturing information about these points. This is known as dimensionality reduction. There are generally two reasons for wanting to do this: 1) visualization 2) preprocessing to save on computationally expensive algorithms. PCA is a special case of dimensionality reduction where the mapping $\Phi : \mathbb{R}^p \to \mathbb{R}^d$ is linear. Other types of dimensionality reduction techniques are diffusion maps and autoencoders.

PCA solves three optimization problems simultaneously:

1. Find the best d dimensional affine subspace to fit $\{x_1, ..., x_n\}$. That is, choose a subspace such that the sum of distances from x_i to the subspace is minimized.
2. Find the best d dimensional projection that preserves as much of the variance of $\{x_1, ..., x_n\}$ as possible.
3. Find the maximum likelihood estimator (MLE) under the assumption of Gaussian noise. For details, see [TB99].

We will show how PCA solves the first optimization problem. To do this, we will optimize over orthonormal bases for the subspace $v_1, ..., v_d \in \mathbb{R}^p$ (stored in a matrix $V = [v_1|\cdots|v_d] \in \mathbb{R}^{p\times d}$) as well as translation vectors $\mu \in \mathbb{R}^p$ that move the subspace origin and $\beta_k \in \mathbb{R}^d$ coefficient vectors. We are approximating x_k as:

$$x_k \approx \mu + \sum_{i=1}^{d} (\beta_k)_i v_i = \underbrace{\mu + V\beta_k}_{=:\Phi}$$

The RHS defines $\Phi(x_k)$. The sum of squares of residuals is:

$$\sum_{k=1}^{n} \|x_k - (\mu + V\beta_k)\|_2^2$$

Without loss of generality, we can assume $\sum_{k=1}^{n} \beta_k = 0$ by changing μ appropriately. To solve for μ we take the gradient with respect to μ of the residual sum:

$$2\sum_{k=1}^{n} [x_k - (\mu - V\beta_k)] = 0$$

$$\Rightarrow \mu = \frac{1}{n}\sum_{k=1}^{n}(x_k - V\beta_k) = \frac{1}{n}\sum_{k=1}^{n} x_k$$

where the $\sum V\beta_k$ term vanished because of our WLOG assumption. Minimizing the sum over β is again a linear least squares problem as was with μ. It is also separable in k, so it is equivalent to solve the following for each k:

$$\min_{\beta_k} \|x_k - (\mu + V\beta_k)\|_2^2$$

Differentiating with respect to β_k and setting to zero, we get:

$$\beta_k = V^T(x_k - \mu)$$

Substituting, the original optimization problem is now reduced to:

$$\min_{V^TV=I} \sum_{k=1}^{n} \|((x_k - \mu) - VV^T(x_k - \mu)\|_2^2$$

Expanding the summand:

$$\|((x_k-\mu)-VV^T(x_k-\mu)\|_2^2 = \|x_k-\mu\|_2^2 - 2\langle x_k-\mu, VV^T(x_k-\mu)\rangle + \|VV^T(x_k-\mu)\|_2^2$$

The first term doesn't depend on V, so we can drop it. The third term is $(x_k-\mu)^t VV^T(x_k-\mu)$ after writing out the definition of $\|\cdot\|_2^2$ in terms of inner products. All together, these terms combine to:

$$\|((x_k-\mu)-VV^T(x_k-\mu)\|_2^2 = -(x_k-\mu)^T VV^T(x_k-\mu)$$

Therefore:

$$\min_{V^TV=I}\sum_{k=1}^n \|((x_k-\mu)-VV^T(x_k-\mu)\|_2^2 \iff \max_{V^TV}\sum_{k=1}^n (x_k-\mu)^T VV^T(x_k-v)$$

Lecture 9/9

Rewriting the RHS:

$$\sum_{k=1}^n (x_k-\mu)^T VV^T(x_k-v) = \sum_{k=1}^n Tr[(x_k-\mu)^T VV^T(x_k-v)]$$

Using $Tr(AB)=Tr(BA)$, this is the same as:

$$\sum_{k=1}^n Tr[V^T(x_k-\mu)V(x_k-v)^TV] = Tr\left[V^T\left(\sum_{k=1}^n (x_k-\mu)(x_k-\mu)^T\right)V\right] = (n-1)Tr[V^T\Sigma_n V]$$

where Σ_n is the sample variance. Therefore we wish to solve:

$$\max_{V^TV=I} Tr[V^T\Sigma_n V]$$

This is achieved by the top d eigenvectors of Σ_n (see [BSS] chapter 3). Explicitly, compute the diagonalization (eigendecomposition) $\Sigma_n = Q\Lambda Q^T$ where $\Lambda = \mathrm{diag}(\lambda_1,...,\lambda_p)$ is arranged so that $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_p$. Then V is the first d columns of Q.

Therefore, PCA is done using the following steps:

1. Compute the sample mean $\mu = \frac{1}{n}\sum_{k=1}^n x_k$.
2. Compute the sample covariance $\Sigma_n = \frac{1}{n-1}\sum_{k=1}^n (x_k-\mu)(x_k-\mu)^T$ and let V be the top d eigenvectors of Σ_n.
3. Set $\beta_k = V^T(x_k-\mu)$. These are the expansion coefficients for the dimensionally reduced data. Namely, these are the low-dimensional versions of x_k.
4. PCA is the map that projects on to $\mathrm{span}(V)+\mu$. In particular, it sends $x_k \mapsto \beta_k$.

Remark 2.2. The step where we chose the top d eigenvectors of Σ_n elucidates how PCA is also the projection that preserves as much of the variance of x_i as possible. More explicitly, recall we maximized:

$$\sum_{k=1}^n (x_k-\mu)^T VV^T(x_k-v)$$

But we can rewrite that as the following sum of norms:

$$\sum_{k=1}^n \|V^T(x_k-\mu)\|_2^2 = \sum_{k=1}^n \|\beta_k\|_2^2$$

We are thus also maximizing the variance in the projected points β_k.

2.2.1 Computing PCA

The expensive part of PCA is finding the top d eigenvalues of Σ_n. The straightforward way is to calculate Σ_n, which costs $\mathcal{O}(p^2 n)$, and then compute the full eigendecomposition of Σ_n, which costs $\mathcal{O}(p^3)$. A more clever way is to rewrite Σ_n as:

$$\Sigma_n = \frac{1}{n-1}(X - \mu \mathbf{1}^T)(X - \mu \mathbf{1})^T \tag{2.2.1}$$

where $\mathbf{1} = (1, ..., 1) \in \mathbb{R}^n$, $X = [x_1 | \cdots | x_n] \in \mathbb{R}^{p \times n}$. In general, eigendecompositions of AA^T are related to SVDs of A because $A = U\Sigma V^T \Rightarrow AA^T = U(\Sigma\Sigma^T)U^T$. The matrix $\Sigma\Sigma^T$ is now diagonal, and so this is an eigendecomposition of AA^T. To get the top d eigenvectors of Σ_n, we just take the top left singular vectors of $X - \mu\mathbf{1}$. So we have reduced PCA to computing the truncated SVD, which has cost $\mathcal{O}(dnp)$. In fact, one can even use randomized SVD methods to push this to $\mathcal{O}(pn \log d + (p+n)d^2)$.

Lecture 9/14

2.2.2 PCA in high dimensions

Let $x_1, ..., x_n \in \mathbb{R}^p$ be iid draws from $\mathcal{N}(0, \Sigma)$ (a multivariate Gaussian distribution). Just like with the univariate Gaussian, a mean zero multivariate Gaussian has a pdf of the form:

$$f(t_1, ..., t_p) = \frac{1}{\sqrt{(2\pi)^p \det(\Sigma)}} e^{-\frac{1}{2} t^T \Sigma^{-1} t}$$

The matrix Σ is the covariance matrix, which must be positive definite and $p \times p$ size.

Problem (covariance estimation): Estimate Σ from the data $x_1, ..., x_p$. The sample mean and covariance are:

$$\mu_n = \frac{1}{n} \sum_{k=1}^{n} x_k$$

$$\Sigma_n = \frac{1}{n-1} \sum_{k=1}^{n} (x_k - \mu_n)(x_k - \mu_n)^T$$

If p is fixed and $n \to \infty$, the law of large numbers says that $\Sigma_n \to \mathbb{E}[\Sigma_n]$. On homework 1, we showed that $\mathbb{E}[\Sigma_n] = \Sigma$. Thus, in the regime where p is fixed but $n \to \infty$, it is an easy problem to estimate Σ.

In practice, we might have very large p and large (but finite) n. Is this best modeled by the above limit where p is fixed and n is growing? The answer is no; a better limit to consider the limit where p and n both go to infinity such that $p/n \to \gamma \in [0, 1]$. Consider

$$S_n = \frac{1}{n} X X^T$$

where $X = [x_1 | \cdots | x_n] \in \mathbb{R}^{p \times n}$. This is closely related to Σ_n (see Equation 2.2.1). This is a random (covariance) matrix, called a *Wishart matrix*. To understand S_n, it suffices to understand its spectrum. The eigenvalues of S_n are random; but how are they distributed? How they are distributed as $p, n \to \infty$ with p/n fixed is the subject of the following result.

Definition 2.3. The *high dimensional limit* of an $p \times n$ matrix X is the limit where $p, n \to \infty$ and $p/n \to \gamma$ for some $\gamma \in [0, 1]$.

Theorem 2.4 (Marčenko-Pastur Law, 1967)**.** *Let X be a matrix whose columns are random iid draws from $\mathcal{N}(0, \Sigma)$. Then in the high dimensional limit, the distribution of eigenvalues of $S_n = \frac{1}{n} X X^T$ converges almost surely to a continuous distribution with the following pdf*

$$dF_\gamma(\lambda) = \frac{1}{2\pi} \frac{\sqrt{(\gamma_+ - \lambda)(\lambda - \gamma_-)}}{\lambda\gamma} \operatorname{ind}_{[\gamma_-, \gamma_+]} d\lambda$$

where $\gamma_+ := (1+\sqrt{\gamma})^2, \gamma_- := (1-\sqrt{\gamma})^2$*, and* ind_I *is the indicator function on* I*. In other words, given two real numbers* $a \leq b$*, the number of eigenvalues of* S_n *in* $[a,b]$ *converges to* $\int_a^b dF_\gamma(\lambda)$*.*

For more details on this law, see [BVZ19]

2.2.3 Spike Models and the BBP Phase Transition

Let $X \in \mathbb{R}^p$ be of the form $X = \sqrt{\beta} g_0 u + g$, where $g \sim \mathcal{N}(0, I)$, $g_0 \sim \mathcal{N}(0,1)$, β is a positive scalar, and $u \in \mathbb{R}^p$ is fixed and deterministic with $\|u\|_2 = 1$. This is a random vector along the line u with full dimensional noise, which is known as a *spike model*. In this case, can we recover u from iid draws of X? Another way of writing X is $X \sim \mathcal{N}(0, I + \beta u u^T)$, so this is like asking if the leading eigenvector of Σ_n is going to approximate u well[2].

The classical limit of p fixed and $n \to \infty$ is an easy one, because $S_n \to I + \beta u u^T$. The high-dimensional limit $p/n \to \gamma \in [0,1]$ is more interesting however. There is a phase transition that depends on β. This is summarized as:

Theorem 2.5 ([BBP])**.** *Let* $\beta_c = \sqrt{\gamma}$*. Then in the high dimensional limit, the largest eigenvalue of* S_n *converges to* $(\beta+1)(1+\gamma/\beta)$ *if* $\beta \geq \beta_c$ *and otherwise* γ_+*. Moreover, let* v_n *be the leading eigenvector of* S_n*. Then* $|\langle v_n, u\rangle|$ *converges to* $\frac{1-\gamma/\beta^2}{1+\gamma/\beta^2}$ *if* $\beta \geq \beta_c$ *and* 0 *otherwise.*

In particular, this shows that eigenvalues above the γ_+ upper bound of the Marčenko-Pastur law can be considered "significant" for detecting a statistical bias such as this one. There is a refinement about the distribution of v_n, which is that it is uniformly distributed about a cone around u. For details, see [BW20]

[2]Equivalently: does the first component of PCA will approximate u well?

3. Clustering

❖

Lecture 9/16

Given a set of points $\mathcal{X} = \{x_1, ..., x_n\} \subset \mathbb{R}^p$, the problem of clustering $\mathcal{X}$ is to find a partition of $\mathcal{X}$ into clusters $S_1 \sqcup S_2 \sqcup ... \sqcup S_k = \mathcal{X}$ such that points in the same cluster are close to eachother and points in different clusters are far from each other. The notions of "close" and "far" can vary depending on context. This is called point clustering. Another type of clustering is graph clustering, where we have an edge weighted graph and we want to cluster the vertices in some way that minimizes a proximity measure on the nodes. This is caled graph clustering.

3.1 k-means clustering ❖

We formulate the point clustering problem as solving the following minimization problem:

$$\min_{S_1,...,S_k} \min_{\mu_1,...,\mu_k \in \mathbb{R}^p} \sum_{\ell=1}^{k} \sum_{i:x_i \in S_\ell} \|x_i - \mu_\ell\|_2^2$$

That is, find a partition $S_1, ..., S_k$ and points $\mu_1, ..., \mu_k$ such that the sum above is minimized (here k is fixed ahead of time). This is known as the *k-means* problem. The points $\mu_1, ..., \mu_k$ are "centroids" of the clusters $S_1, ..., S_k$ (though a-priori they can be any points). This is a combinatorial optimization problem, and is NP-hard. However, there are algorithms that can give a good approximation to the true minimum. One such algorithm is Lloyd's algorithm, which alternates minimizing with respect to the partition and the central points. It goes as follows:

Lloyd's Algorithm

1. Choose k.
2. Initialize by randomly choosing central points $\mu_1, ..., \mu_k$.
3. Hold the central points fixed and choose the best partition. This is easly done by setting:
$$S_k = \{x_i \ : \ \|x_i - \mu_k\|_2 = \min_\ell \|x_i - \mu_\ell\|_2\}$$
4. Hold the partition fixed and choose the best central points. The best central points of S_ℓ satisfies:
$$\min_{\mu_\ell \in \mathbb{R}^p} \sum_{i:i \in S_\ell} \|x_i - \mu_\ell\|_2^2$$
Solve this via linear least squares to get $\mu_\ell = \frac{1}{|S_\ell|} \sum_{i:i \in S_\ell} x_i$. This is the barycenter of S_ℓ.
5. Return to step 3, stopping whenever satified.

The best partition for the k-means cost function always consists of convex clusters, meaning $\text{Conv}(S_i) \cap \text{Conv}(S_j) = \emptyset$ for all $i \neq j$. This can be useful, but is also a deficiency of the k-means cost function.

3.2 Spectral Clustering ❖

Given $\mathcal{X} = \{x_1, ..., x_n\}$ as above, we can construct a graph G whose vertices are $\mathcal{X}$ and $E(G) = \binom{\mathcal{X}}{2}$. There is also a weighting on each edge $w_{ij} = w(x_i, x_j) = k_\epsilon(\|x_i - x_j\|_2)$, where k_ϵ is called a similarity kernel. A common choice is $k_e(u) = \exp(-u^2/2\epsilon)$. Now we formulate a cost function to perform a graph clustering. Let $V = S \sqcup S^c$

be a partition of the vertices of G (called a cut) and we score each cut by all the weights between the two sets S and S^c:

$$\text{cut}(S) = \sum_{i \in S} \sum_{j \in S^c} w_{ij}$$

We will now show how $\text{rcut}(S)$ can be represented by a a quadratic form in n variables. We will encode the partition as $y \in \{\pm 1\}^n$, where $y_i = 1$ if $i \in S$ and -1 otherwise.

Definition 3.1. Let $D \in \mathbb{R}^{n \times n}$ be a diagonal matrix such that $D_{ii} = \deg(i) = \sum_{j=1}^{n} w_{ij}$. Then the *graph laplacian* is $L_G : D - W$.

Lemma 3.2. *The graph laplacian has the following properties:*

1. *L_G is positive semi-definite, meaning $z^T L_G z \geq 0$ for all $z \in \mathbb{R}^n$. This is also equivalent to all eigenvalues of L_G being non-negative.*
2. *Let $0 \leq \lambda_1 \leq ... \leq \lambda_n$ be the eigenvalues of L_G with corresponding (normalized) eigenvectors $v_1, ..., v_n$. Then $\lambda_1 = 0$ and $v_1 = \mathbf{1}/\sqrt{n}$. This is called the trivial eigenvector.*
3. *$L_G = \sum_{i \leq j} w_{ij}(e_i - e_j)(e_i - e_j)^T$.*
4. *$z^T L_G z = \sum_{i<j} w_{ij}(z_i - z_j)^2$. In other words, the graph laplacian represents a quadratic form.*

The relationship between L_G and $\text{cut}(S)$ is:

Proposition 3.3. *Let $y \in \{\pm 1\}^n$ be the corresponding vector to S. Then $\text{cut}(S) = \frac{1}{4} y^T L_G y$.*

Proof:

By the last item in the above lemma:

$$\frac{1}{4} y^T L_G y = \frac{1}{4} \sum_{i<j} w_{ij}(y_i - y_j)^2$$

When i and j are in the same cluster, $(y_i - y_j)^2 = 0$ and 4 otherwise. Then simplifying that out gives $\text{cut}(S)$.

□

Returning to the graph clustering problem, we would like to solve:

$$\min_{S \subset V} \text{cut}(S) \iff \min_{y \in \{\pm 1\}^n} y^T L_G y$$

The problem with this, however, is that the cut function is a bad cost function by itself because $S = \emptyset$ minimizes it (e.g. $y = \mathbf{1}$ trivially solves). To modify it and make it a well-posed problem, we have a few choices:

1. Use Cheeger's cut:
$$h(S) = \frac{\text{cut}(S)}{\min(\text{vol}(S), \text{vol}(S^c))}$$
where $\text{vol}(S) = \sum_{i \in S} \deg(i)$.
2. Use the normalized cut:
$$\text{Ncut}(S) = \frac{\text{cut}(S)}{\text{vol}(S)} + \frac{\text{cut}(S)}{\text{vol}(S^c)}$$
3. Try to force balanced clusters; i.e. require $\sum_i y_i = 0$. Then the clustering problem becomes:
$$\min_{y \in \{\pm 1\}^n, \mathbf{1}^T y = 0} y^T L_G y$$

Lecture 9/21

3.2.1 Solving the Normalized Cut Problem

We will show in this section how the balanced clusters problem (3. above) can be reformulated to the normalized cut problem (2. above). Then we will show how to solve that by an eigenvector computation.

We can relax the balanced cut formulation 3. above by picking two real numbers $a > 0, b < 0$ and redefining balanced as $a\text{vol}(S) + b\text{vol}(S^c) = 0$. This is equivalent to $\mathbf{1}Dy = 0$, where $y_i = a$ when $i \in S$ and b otherwise. To fix the scale of a and b, we also enforce $a^2\text{vol}(S) + b^2\text{vol}(S^c) = 1$. Alternatively, $y^T Dy = 1$. Note that for any cut S, the two conditions on a and b uniquely determine a and b:

Lemma 3.4. *Fix a cut S of G. Then the two conditions:*

$$avol(S) + bvol(S^c) = 0$$

$$a^2vol(S) + b^2vol(S^c) = 1$$

determine a and b uniquely (up to sign). Moreover, the solutions are:

$$a = \sqrt{\frac{vol(S^c)}{vol(S)vol(G)}}$$

$$b = \sqrt{\frac{vol(S)}{vol(S^C)vol(G)}}$$

where $vol(G)$ is the volume of the whole graph.

The following proposition shows that the normalized cut is represented by the quadratic form L_G evaluated on the reformulated vectors y.

Proposition 3.5. *Given a cut S of G, let a and b be given by the formulae above and let $y \in \mathbb{R}^n$ be given by $y_i = a$ if $i \in S$ and b if $i \in S^c$. Then $y^T L_G y = \text{Ncut}(S)$.*

Proof idea:

> By the lemma, we have a and b fixed. So we can just plug them in to the LHS and simplify.

□

Therefore minimizing $\text{Ncut}(S)$ is the same as solving

$$\min_{\substack{y\in\mathbb{R}^n, a\in\mathbb{R}, b\in\mathbb{R} \\ s.t. y\in\{a,b\}^n \\ y^T D\mathbf{1}=0 \\ y^T Dy=1}} y^T L_G y$$

It is also NP-hard, due to the constraint $y \in \{a, b\}^n$. If we drop this constraint (i.e. eliminate a and b as part of the optimization), we have what is known as the relaxed balanced cut problem:

$$\min_{\substack{y^T D\mathbf{1}=0 \\ y^T Dy=1}} y^T L_G y$$

We can try to solve this first and then try to enforce the combinatorial constraint at the end. Define $z = D^{1/2}y \in \mathbb{R}^n$. Then $y^T Dy = z^T z$, and the linear constraint becomes $z^T D^{1/2}\mathbf{1} = 0$. The objective function becomes $z^T \mathcal{L}_G z$, where $\mathcal{L}_G = D^{-1/2} L_G D^{-1/2}$ (called the normalized graph Laplacian). Conveniently, $D^{1/2}\mathbf{1}$ is the lowest eigenvector of $\mathcal{L}_G$ with eigenvalue 0.[3]. Thus the relaxed balanced cut problem is:

$$\min_{\substack{\|z\|_2=1 \\ z\perp D^{1/2}\mathbf{1}}} z\mathcal{L}_G z$$

[3]This can be shown from the definitions and Lemma 3.2

The Courant-Fischer theorem (see aside below) says that the solution to this is the second lowest eigenvector $\tilde{v}_2$ of $\mathcal{L}_G$. The optimal cost is the second lowest eigenvalue of $\mathcal{L}_G$. Therefore the solution to the relaxed balanced cut problem is $y = D^{-1/2}\tilde{v}_2$.

Courant-Fisher aside: Let $A \in \mathbb{R}^{n\times m}$ be a symmetric matrix, and consider the problem

$$\max_{\substack{U\subset\mathbb{R}^n \text{ subspaces}\\ \dim(U)=n-k+1}} \min_{\substack{x\in U\\ \|x\|_2=1}} x^T A x$$

The Courant-Fisher theorem says that this is attained when U is the orthognal complement of the space spanned by the $k-1$ lowest eigenvectors of A and x is the kth eigenvector.

Now how do we get a partition out of this solution? One idea is to partition the graph according to the sign of y_i. More generally, pick a threshold τ and use the partition determined by $i \in S \iff y_i \leq \tau$. Then one can optimize over different values of τ. It turns out that this works fairly well:

Lecture 9/23

Theorem 3.6 (Lemma 4.8 in [BSS] 4.3)**.** *Let λ_2 be the second lowest eigenvalue of $\mathcal{L}_G$ and y be the solution to the relaxed balanced cut problem. Then there exists $\tau \in \mathbb{R}$ such that the cut $S_\tau = \{i : y_i < \tau\}, S_\tau^c = \{i : y_i \geq \tau\}$ has Cheeger value $h(S) \leq \sqrt{2\lambda_2}$. This upper bound is known as the Cheeger constant.*

Proof idea:

> Consider randomly choosing τ accordig to some well-chosen probability distribution on $\mathbb{R}$. Then prove that $\mathbb{E}[h(S_\tau)] \leq \sqrt{2\lambda_2}$. This would mean there exists τ for which $h(S_\tau) \leq \sqrt{2\lambda_2}$.

□

Note that, since λ_2 was the optimal solution to the relaxed problem, we have $\lambda_2 \leq \mathrm{Ncut}(S)$. Moreover, since $\mathrm{Ncut}(S) \leq 2h(S)$, we then get $\frac{1}{2}\lambda_2 \leq h(S)$. This is a lower bound on $h(S)$, and the theorem above is an upper bound.

3.2.2 Spectral Clustering with k Clusters

How do we leverage the problem we just solved if we want to cluster a graph into more than two clusters? Consider the vectors $D^{-1/2}\tilde{v}_2, ..., D^{-1/2}\tilde{v}_{k+1}$, where $\tilde{v}_j$ are the eigenvectors of $\mathcal{L}_G$, ordered by eigenvalue magnitude. Join these vectors as columns of a matrix $X \in \mathbb{R}^{n\times k}$. Then the rows of X correspond to the nodes of the graph, and we can perform k-means clustering on those rows to those rows to get a clustering. There are k-part versions of the Cheeger constants as well.

4. Diffusion Maps

❖

Diffusion maps are often used as a visualization technique or part of nonlinear dimensionality reduction. Suppose we have a non-negatively weighted graph $G = (V, E, W)$. Consider a random walk with independent steps on V governed by transition probabilities of the form:

$$\mathbb{P}(X(t+1) = j \mid X(t) = i) = \frac{w_{ij}}{\deg(i)}$$

where $X(t) \in V$ is the location of the walk at time step t. Note the denominator $\deg(i)$ is required to maintain normalization to 1. As a matrix, these probabilities are $M = D^{-1}W$. This is an example of a *stochastic matrix*, which means that $M_{ij} \geq 0$ and $M\mathbf{1} = \mathbf{1}$.[4] This particular matrix is called the random walk graph laplacian. By the formalism of Markov chains, for $r \in \mathbb{N}$ we have:

$$\mathbb{P}(X(t) = j \mid X(0) = i) = (M^r)_{ij}$$

That is, the probability of ending up at vertex j given that you start at vertex i is equal to the ij entry of M^r.

With diffusion maps, the goal is to map $V \to \mathbb{R}^d$ for small d in a way that we can better visualize or understand the graph. One initial approach would be to send vertex i to the probability cloud on V after r steps of the above random walk (e.g. the ith row of M^r). We can compute M^r using the fact that $D^{1/2}MD^{-1/2} = D^{-1/2}WD^{-1/2} = I - \mathcal{L}_G$, which is symmetric (and therefore has n real eigenvalues). Let $V\Lambda V^T$ be the orthogonal eigendecomposition of $D^{-1/2}WD^{-1/2}$, so that:

$$M = (D^{-1/2}V)\Lambda(V^T D^{1/2})$$

Note that $\Phi = D^{-1/2}V$ and $\Psi = V^T D^{1/2}$ satisfy $\Phi^T\Psi = I$. We can now compute powers of M as if it had an orthogonal eigendcomposition:

$$M^r = \Phi\Lambda^r\Psi^T$$

Then the ith row of M^r is $\sum_{k=1}^n \lambda_k^r \phi_k(i)\psi_k^T$. In the orthonormal basis $\{\psi_1, ..., \psi_n\}$, this is $[\lambda_1^r\phi_1(i), ..., \lambda_n^r\phi_n(i)]$. These coordinates are called *diffusion coordinates*.

Lemma 4.1. *All of the eigenvalues of M satisfy $|\lambda_k| \leq 1$.*

Proof idea:

> Apply Gershgorin's disk theorem.

□

Definition 4.2. The *truncated diffusion maps* $\text{Diff}_r : V \to \mathbb{R}^d$ are given by $i \mapsto [\lambda_2^r\phi_2(i), ..., \lambda_{d+1}^r\phi_{d+1}(i)]$. The un-truncated diffusion map is $\text{Diff} := \text{Diff}_n$.

Lecture 9/28

Note that $\lambda_1^r\phi_1(i)$ is trivial and independent of G. This is why the truncated diffusion maps start at $\lambda_2\phi_2(i)$.

Proposition 4.3. *For any $i_1, i_2 \in V$ we have:*

$$\|\text{Diff}(i_1) - \text{Diff}(i_2)\|_2^2 = \sum_{j=1}^n \frac{1}{\deg(j)} \left(\mathbb{P}(X(t) = j \mid X(0)) = i_1) - \mathbb{P}(X(t) = j) \mid X(0) = i_2)\right)^2$$

Proof:

[4] See Wikipedia entry for Markov Chains.

The right hand side is:

$$\begin{aligned} RHS &= \sum_{j=1}^{n} \frac{1}{\deg(j)} (M^t_{i_1 j} - M_{i_2 j})^2 \\ &= \sum_{j=1}^{n} \frac{1}{\deg(j)} \left(\sum_{k=1}^{n} \lambda_k^t \phi_k(i_1) \psi_k(j) - \sum_{k=1}^{n} \lambda_k^t \phi_k(i_2) \psi_k(j) \right)^2 \\ &= \left\| \sum_{k=1}^{n} \lambda_k^t (\phi_k(i_1) - \phi_k(i_2)) D^{-1/2} \psi_k \right\|_2^2 \end{aligned}$$

The vectors $\{D^{-1/2}\psi_k\}$ are orthogonal, so this just becomes:

$$RHS = \sum_{k=1}^{n} \left[\lambda_k^t (\phi_k(i_1) - \phi_k(i_2)) \right]^2$$

We can drop $k = 1$ because $\phi_1(i) = \mathbf{1}$ for all i. This is now the squared ℓ_2 norm of $\text{Diff}(i_i) - \text{Diff}(i_2)$.

□

4.0.1 Connection to Spectral Clustering

Diffusion maps operate on $M = D^{-1}W$, whereas spectral clustering operates on $\mathcal{L}_G = I - D^{-1/2}WD^{1/2}$. There were also (left) eigenvectors ϕ_k in the former case and eigenvectors $\tilde{v}_k$ in the latter case. We claim that solving for these eigenvectors is the same problem in both cases (when we take $t = 0$ for the diffusion maps).

Diffusion maps	Spectral clustering
$M = D^{-1}W$	$\mathcal{L}_G = I - D^{-1/2}WD^{1/2}$
Left eigenvectors ϕ_k	Eigenvectors $\tilde{v}_k$

TODO Justify this connection

4.1 Relationship to Manifold Learning ❖

Why do we call these graph Laplacians (M, L_G, $\mathcal{L}_G$) as such? Recall the Laplacian on $\mathbb{R}^d$ is a second-order (compact, self adjoint) linear differential operator Δ. Given $f : \mathbb{R}^d \to \mathbb{R}$, it is defined by:

$$\Delta f := \frac{\partial^2 f}{\partial x_1^2} + \cdots + \frac{\partial^2 f}{\partial x_d^2}$$

The heat equation is expressed through the Laplacian:

$$u_t = \Delta u$$

where $u : \mathbb{R}_t \times \mathbb{R}_x^d \to \mathbb{R}$. When $\Delta u = 0$, it is called equilibrium. It also captures local averages of a function. Given $f : \mathbb{R}^d \to \mathbb{R}$, the average value of f in $B_h(p)$ is $f(p) + \frac{\Delta f(p)}{C} h^2 + o(h^2)$ for a fixed constant C as $h \to 0$. There is a generalization of the Laplacian to any Riemannian manifold (M, g). This is called the Laplace-Beltrami operator $\Delta_M : C^\infty(M) \to C^\infty(M)$ defined by:

$$\Delta_M(f) = \text{tr}(\text{Hess}(f))$$

where $\text{Hess}(f)$ is the Riemannian Hessian of f. This, too, is related to heat flow, equilibrium, and local averages on M. For example, if $M = S^2$ with the round metric, the eigenfunctions of Δ_M are the well-known spherical harmonics.

Theorem 4.4 ([BN08] Theorem 3.1)**.** *Let $M^d \subseteq \mathbb{R}^D$ be a compact embedded Riemannian submanifold. Assume that $x_1, ..., x_n$ are iid samples of M with respect to the uniform measure $d\mu$ on M. Let $f \in C^\infty(M)$ and let $p \in M$. Then, as $n \to \infty$,*

$$f(p)\frac{1}{n}\sum_{j=1}^{n} K_{\epsilon_n}(\|p - x_j\|_2) - \frac{1}{n}\sum_{j=1}^{n} f(x_j)K_{\epsilon_n}(\|p - x_j\|_2) \to \frac{1}{vol(M)}\Delta_M f(p)$$

where $K_{\epsilon_n}(u) = \exp(-u^2/4\epsilon_n)$ and $\epsilon_n = n^{-1/(d+3)}$. This convergence is in probability.

Remark 4.5. Let G be the graph with vertices $V = \{x_1, ..., x_n\}$ and the weights $w_{ij} = \frac{1}{n}K_{\epsilon_n}(\|x_i - x_j\|_2)$. The degree matrix D has components $D_{ii} = \frac{1}{n}\sum_{j=1}^{n} K_{\epsilon_n}(\|x_i - x_j\|_2)$. Let $\tilde{f} : V \to \mathbb{R}$ be the restriction $\tilde{f} = f|_V$. The application of $L_G = D - W$ to $\tilde{f}$ is a vector whose ith component is:

$$\tilde{f}(x_i)\frac{1}{n}\sum_{j=1}^{n} K_{\epsilon_n}(\|x_i - x_j\|_2) - \frac{1}{n}\sum_{j=1}^{n} \tilde{f}(x_j)K_{\epsilon_n}(\|x_i - x_j\|_2)$$

Replacing x_i with any $p \in M$, we get the LHS of the theorem. In this way, the graph Laplacian is extended to all of M and the graph Laplacian converges pointwise to the Laplace-Beltrami operator assuming the points are samples from the manifold. In stronger versions of the above result, it is shown that the eigenvectors of L_G converge in probability to the eigenfunctions of Δ_M.

Lecture 9/30

Example 4.6. Let $S^1 \subseteq \mathbb{R}^2$ be the standard unit circle parameterized by θ. For $f : S^1 \to \mathbb{R}$, we have $\Delta_{S^1} f = \frac{d^2}{d\theta^2} f$. The eigenfunctions of Δ_{S^1} are the solutions to $\frac{d^2 f}{d\theta^2} = \lambda f$, which are $f(\theta) = \cos(m\theta)$ or $\sin(m\theta)$ for $m \in \mathbb{Z}$.

The graph Laplacian was useful for clustering data because of its connection to Ncut. Another powerful use of graph Laplacians / Diffusion maps is useful for understanding data satisfying the *manifold hypothesis*.

Definition 4.7. Let $x_1, ..., x_n \in \mathbb{R}^D$. The *manifold hypothesis* is that there exists a compact embedded Riemannian manifold $M \subseteq \mathbb{R}^D$ with $\dim(M) = d \ll D$ and the points x_i are on or near M.

The two main uses of manifold learning are:

- Visualization / nonlinear dimensionality reduction. Given $\{x_1, ..., x_n\} \subset \mathbb{R}^D$, we can use diffusion maps to reduce them to $\mathbb{R}^h$ for $h \approx d$.
- Function approximation. Given $\tilde{f} : \{x_1, ..., x_n\} \to \mathbb{R}$, suppose we want to approximate it with a compressed verison. Then one can expand $\tilde{f}$ in terms of a few eigenvectors of L_G.

Proof sketch of 4.4:

Let the LHS be denoted $L_n^{\epsilon_n} f(p)$. We are interested in computing the expected value of the LHS:

$$\mathbb{E}\left[L_n^{\epsilon} f(p)\right] = f(p)\int_M \exp(-\|p - x\|_2^2/4\epsilon)d\mu(x) - \int_M f(x)\exp(-\|p - x\|_2^2/4\epsilon)d\mu(x) \qquad (\dagger)$$

The first step is to use Hoeffding's inequality to show that for any $\delta > 0$ and ϵ_n as in the theorem statement, we have:

$$\mathbb{P}\left(\frac{1}{\epsilon_n(4\pi\epsilon_n)^{d/2}}|L_n^{\epsilon_n} f(p) - \mathbb{E}[L_n^{\epsilon_n} f(p)]| > \delta\right) \to 0 \quad \text{as } n \to \infty$$

Therefore it suffices to show that $\mathbb{E}[L_n^{\epsilon} f(p)] \to RHS$ to prove the theorem (here RHS is the right hand side of the theorem statement).

We thus need to evaluate the integrals in $(\dagger)$. Note that the exponential terms allow us to focus only on a neighborhood of p as ϵ shrinks. Namely, let $B \subset M$ be a fixed open neighborhood of p. Then for

sufficiently small ϵ, we have:

$$\left| f(p) \int_M \exp(-\|p-x\|_2^2/4\epsilon) d\mu(x) - f(p) \int_B \exp(-\|p-x\|_2^2/4\epsilon) d\mu(x) \right| = o(\epsilon^{1000d})$$

and likewise with the second term. Thus we can integrate $(\dagger)$ over B. Recall the exponential map $\exp_p$ gives a diffeomorphism $U \cong B$, where $U \subset T_p(M)$, mapping 0 to p. We can do change of coordinates to pull back the integrals to U. Let $F = f \circ \exp_p : U \to \mathbb{R}$ be the pullback of f to U. Then

$$\begin{aligned}
\mathbb{E}[L_n^\epsilon f(p)] &= f(p) \int_M \exp(-\|p-x\|_2^2/4\epsilon) d\mu(x) - \int_M f(x) \exp(-\|p-x\|_2^2/4\epsilon) d\mu(x) \\
&= \int_U \exp(-\|p-\exp_p(u)\|_2^2/4\epsilon)(F(0)-F(u)) \sqrt{|g_{ij}|} du
\end{aligned}$$

The next step is to do a Taylor expansion on everything in sight, using the following two facts:

1. The metric Taylor expands as:

$$|g_{ij}| = 1 - \frac{1}{6} u^T R u + O(\|u\|^3)$$

   where R is the Ricci curvature. Note this doesn't have a linear term.

2. For all $x, y \in M$, we have:

$$\|x-y\|^2 = \mathrm{dist}_M^2(x,y) - O(\mathrm{dist}_M^4(x,y))$$

   where dist_M is the Riemannian distance on M.

Substituting these Taylor expansions will yield a $\mathrm{Tr}(\nabla^2 F)$ term, which is equal to $\Delta_M f$. For details on the remaineder of the proof, consult [BN08].

□

5. Convex Relaxations and Semidefinite Programming

❖

Lecture 10/5

This section follows [BSS] Chapter 7. As we saw in the previous section, the method of relaxation can be used to get approximate solutions to intractible problems. Often these intractible problems would involve impractical amounts of time, space, or data to solve optimally and are typically NP-hard. When faced with such a problem, there are three ways to proceed:

1. Pay the cost: spend the necessary resources to solve it optimally.
2. Find the most efficient method for typical (but not necessarily all) instances.
3. Come up with an efficient approximation.

Convex relaxation is an example of the third way.

Definition 5.1. A set $S \subseteq \mathbb{R}^n$ is *convex* if, for every $p, q \in S$ the line segment joining them is entirely contained in S. In other words, for all $t \in [0,1]$, we have $tp + (1-t)q \in S$.

Definition 5.2. Let $S \subseteq \mathbb{R}^n$ be convex. Then $f : S \to \mathbb{R}$ is *convex* if for all $x, y \in S$ and for all $t \in [0,1]$, we have $tf(x) + (1-t)f(y) \geq f(tx + (1-t)y)$. Alternatively, the epigraph $\text{epi}(f) := \{(z, \lambda) \in \mathbb{R}^n \times \mathbb{R} \ : \ z \in S, \lambda \geq f(z)\}$ is a convex set.

If $S \subseteq \mathbb{R}^n$ is open and convex and and $f : S \to \mathbb{R}$ is twice differentiable, then $\nabla^2 f(x)$ is positive semidefinite for all $x \in S$ implies that f is convex. This can be proved by the mean value theorem. Minimizing convex functions is nice because

Theorem 5.3. *Let $f : S \to \mathbb{R}$ be convex. Then every local minimum of f is a global minimum.*

Proof:

> Proceed by the contrapositive. Let $x \in S$ be a local minimum but not be a global minimum, meaning there exists $y \in S$ such that $f(y) < f(x)$. Then consider the line segment L connecting x and y, which must be contained in the epigraph of f. Then use the property of convexity to show that x cannot be a local minimum.

□

This property is useful because continuous optimization (like gradient descent, trust region methods, Newton's methods) can, under some assumptions, local minima of differentiable functions. If our function is convex, then we can guarantee that such a minimum is global. Convex relaxation deals with taking a non-convex optimization problem and approximating it by a related convex problem.

5.1 Max-Cut ❖

Let $G = (V, E, W)$ be a non-negatively weighted graph, with $V = \{1, ..., n\}$ and $W = \{w_{ij}\}_{i,j \in V}$. The max-cut problem is:

$$\max_{y \in \{-1,1\}^n} \frac{1}{2} \sum_{i<j} w_{ij}(1 - y_i y_j)$$

The summand is $2w_{ij}$ if the edge (i, j) if crosses between the two partitions S, S^c induced by y, and zero otherwise. This is a *polynomial optimization* problem, which means the cost function is a polynomial in the variables and the constraints are polynomial.[5] We will explore polynomial optimization more later.

[5]Notice $y_i \in \{\pm 1\} \iff y_i^2 = 1$.

There are two ways to arrive at convex relaxation for the max-cut problem. The first is to replace each $y_i \in \mathbb{R}^n$ with a unit norm vector $u_i \in \mathbb{R}^n$. Then we consider:

$$\max_{\substack{u_i \in \mathbb{R}^n \\ \|u_i\|_2 = 1}} \frac{1}{2} \sum_{i<j} w_{ij}(1 - u_i^T u_j) \tag{5.1.1}$$

By taking each u_i to be $\pm e_1$, we recover the previous formulation of max-cut. Since we enlarged our domain, the optimal value of this relaxed max-cut is at least the optimal value for max-cut.

A more systematic way to arrive at the same relaxation is to rewrite the cost as:

$$\frac{1}{2} \sum_{i<j} w_{ij}(1 - y_i y_j) = \frac{1}{4} \langle W, 11^T - yy^T \rangle$$

Let $X := yy^T$. Then the problem is:

$$\max_{\substack{X \in \mathrm{Sym}\, \mathbb{R}^n \\ \mathrm{rk}(X) = 1 \\ X \text{ pos. semi def.} \\ X_{ii} = 1}} \frac{1}{4} \langle W, 11^T - X \rangle \tag{5.1.2}$$

where $\mathrm{Sym}(\mathbb{R}^n)$ is the space of symmetric $n \times n$ matrices. The contstraints above are equivalent to the max-cut constraints on y: the first two conditions tell us $X = yy^T$ for some $y \in \mathbb{R}^n$ (by the Cholesky factorization) and the third says $y_i^2 = 1$. Which of these constraints on X is not convex? The positive semidefinite constraint is convex (see section below on semidefinite programs) and so is the diagonal condition $X_{ii} = 1$ (as an exercise, verify). However, the rank condition is not.

Lecture 10/7

Note how the cost function is now linear and there are many more variables involved ($\mathcal{O}(n^2)$ vs $\mathcal{O}(n)$). If we drop the rank condition, we get a convex relaxation:

$$\max_{\substack{X \in \mathrm{Sym}\, \mathbb{R}^n \\ X \text{ pos. semi def.} \\ X_{ii} = 1}} \frac{1}{4} \langle W, 11^T - X \rangle$$

This is an example of a *semidefinite program* (see next subsection). First note that this is equivalent to our first relaxation 5.1.1 by taking $X_{ij} = u_i^T u_j$.

We can thus solve the relaxation 5.1.2 in polynomial time. To get a candidate solution to the non-relaxed problem, we need to come up with with a reasonable notion of rounding. One method, due to Goemans-Williamson, is to factor the SDP solution X as $X = U^T U$ (say, through an eigendecomposition). The ith node of the graph is associated with the ith vector u_i of U. Then pick a random unit vector r and choose the cut based on the sign of their inner product with r:

$$S = \{i \ : \ \langle u_i, r \rangle \geq 0\}$$

$$S^c = \{i \ : \ \langle u_i, r \rangle < 0\}$$

Since r is chosen randomly, we can calculate the expected value of this cut:

$$\begin{aligned}
\mathbb{E}[W] &= \sum_{i<j} w_{ij} \mathbb{P}(\text{vertex } i \text{ and vertex } j \text{ separated by } r) \\
&= \sum_{i<j} w_{ij} \frac{\theta_{ij}}{\pi} \\
&= \sum_{i<j} w_{ij} \frac{\cos^{-1}(\langle u_i, u_j \rangle)}{\pi}
\end{aligned}$$

Goemans and Williamson showed that there exists a constant $\alpha_{GW} > 0.87$ such that

$$\frac{\cos^{-1}(\langle u_i, u_j\rangle)}{\pi} \geq \alpha_{GW}\frac{1}{2}(1 - \langle u_i, u_j\rangle)$$

So we can estimate the best cut based on the expected value of this random cut:

$$\text{Max Cut}(G) \geq \mathbb{E}[W] \geq \alpha_{GW}\frac{1}{2}\sum_{i<j} w_{ij}(1 - \langle u_i, u_j\rangle)$$

Notice that the left hand side is exactly the relaxed max-cut value, so:

$$\text{Max Cut}(G) \geq \mathbb{E}[W] \geq \alpha_{GW}\text{Relaxed Max Cut}(G) \geq \alpha_{GW}\text{Max Cut}(G)$$

Therefore, in expectation, we can achieve 87% approximation ratio and we do it via an efficient method. Goemans and Williamson conjecture that this is the best possible approximation ratio. This conjecture is called the unique games conjecture.

5.2 Semidefinite Programs ❖

Here we expand on more on the notion of semidefinite programs, using Max Cut as an example along the way. Let $\text{PSD}_n = \{Z \in \text{Sym}(\mathbb{R}^n) \mid Z \text{ is positive semidefinite}\}$. We claimed last time that PSD_n is a convex cone.

Lemma 5.4. *The set $PSD_n \subset \text{Sym}(\mathbb{R}^n)$ is a closed, convex, full-dimensional cone. It is also self-dual with respect to the Frobenius inner-product $\langle\cdot,\cdot\rangle_F$. In particular, if $A, B \in PSD_n$, then $\langle A, B\rangle_F \geq 0$.*

Proof:

We will only prove the last statement. Let $A = UU^T$ and $B = VV^T$. Then

$$\langle A, B\rangle_F = \text{Tr}(AB) = \text{Tr}(UU^TVV^T) = \text{Tr}((U^TV)(V^TU)) = \|U^TV\|_F^2 \geq 0$$

□

The self-duality of PSD_n gives us a duality of semidefinite programs (defined below), just like there is a duality in linear programs. Note that $\partial(\text{PSD}_n)$ is the set of psd matrices with at least one zero eigenvalue, or equivalently $\{Z \in \text{PSD}_n \mid \det(Z) = 0\}$.

Definition 5.5. A *semidefinite program* (SDP) is a convex optimization problem of the following form:

$$\max_{\substack{X \in \text{PSD}_n \\ \langle A_i, X\rangle = b_i}} \langle C, X\rangle$$

where $A_i \in \text{Sym}(\mathbb{R}^n), b_i \in \mathbb{R}$ define linear constraints on X.

In other words, a semidefinite program seeks to maximize a linear function over the intersection of PSD_n with an affine subspace. Since PSD_n is convex, an affine slice of it is also convex. SDPs can be solved in polynomial time (in n), for example using the interior point method.

Definition 5.6. A *spectrahedron* is the intersection of PSD_n with an affine space. See Figure 5.1.

Lecture 10/12

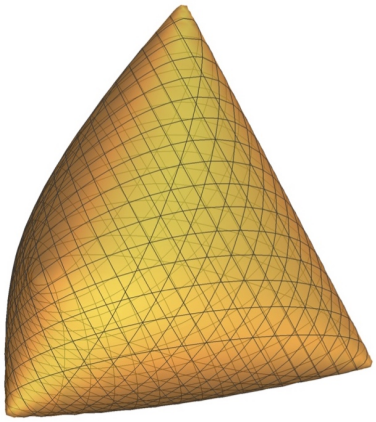

Figure 5.1: An elliptope in $\mathbb{R}^3$, which is an example of a spectrahedron. This is the feasible region for the max-cut problem when $n = 3$.

5.2.1 Dual Relaxed Max Cut

The dual of the relaxed Max Cut problem is:

$$\min_{\substack{D \text{ diagonal} \\ D - \frac{1}{4}L_G \text{pos. semi def.}}} \operatorname{Tr}(D)$$

We won't justify in what way this is dual to the Max Cut problem immediately. Instead, we relate it to the Max Cut problem through the following lemma.

Lemma 5.7. *The optimal value for the dual Relaxed Max Cut problem is at least the optimal value for the Relaxed Max Cut problem.*

Proof:

Let X be feasible for the relaxed max cut problem, and let D be feasible for the dual relaxed cut problem. Because the cone of positive semidefinite matrices is self-dual, we have that $\langle X, D - \frac{1}{4}L_G \rangle_F \geq 0$. This means:

$$0 \leq \operatorname{Tr}(X(D - 1/4L_G)) = \operatorname{Tr}(XD) - \frac{1}{4}\operatorname{Tr}(XL_G)$$

Since $X_{ii} = 1$ and D is diagonal, we have $\operatorname{Tr}(XD) = \operatorname{Tr}(D)$. Thus:

$$\frac{1}{4}\operatorname{Tr}(XL_G) \leq \operatorname{Tr}(D)$$

The RHS is the cost of the dual relaxed cut problem, and the LHS is the cost of the relaxed cut problem. By minimizing over D and maximizing over X, we obtain the desired result.

□

We say that *strong duality* holds for the relaxed max cut problem if both it and the dual relaxed cut problem have the same optimal values. It is a fact that strong duality holds if there exists a strictly feasible primal point or if there exists a strictly feasible dual point.[6] Here "strictly feasible" in either case means that the matrix required to be PSD is actually positive definite (i.e. full rank).

Corollary 5.8. *The relaxed max cut problem is strictly feasible, so strong duality holds and there exists a D such that the dual relaxed max cut optimum value is attained at D.*

[6] This is known as Slater's condition.

We will use this to obtained a sums-of-squares interpretation of what the relaxed max cut problem is doing with respect to Max Cut. Let $\tilde{D}$ be the the optimal solution to the dual relaxed max cut problem. Then $\mathrm{Tr}(\tilde{D})$ is the optimal value of the relaxed max cut problem. This means for all $y \in \{\pm 1\}^n$ feasible for Max Cut, which is also feasible for the relaxed problem, we get:

$$\frac{1}{4}\mathrm{Tr}(L_G y y^T) \leq \mathrm{Tr}(\tilde{D})$$

$$\implies 0 \leq \mathrm{Tr}(\tilde{D}) - \frac{1}{4} y^T L_G y = y^T (D - \frac{1}{4} L_G) y$$

But $\mathrm{Tr}(\tilde{D}) = y^T \tilde{D} y$ because $\tilde{D}$ is diagonal. Since $\tilde{D} - \frac{1}{4}L_G$ is psd, we can factor it as $\tilde{V}\tilde{V}^T$. Thus:

$$y^T(\tilde{D} - \frac{1}{4}L_G)y = y^T \tilde{V}\tilde{V}^T y = \sum_{k=1}^{n} (\tilde{v}_k^T y)^2$$

where $\tilde{v}_k$ is the kth column of $\tilde{V}$. This is a sum of squares, which is manifestly positive.

5.3 Duality of SDPs ❖

Now we define duality in general for semi-definite programs. We will actually derive it for general convex conic problems. Let $K \subset \mathbb{R}^N$ be a convex cone, and let K^* be its dual cone (for example, $K = \mathrm{PSD}_n$). The primal problem we define to be something of the form:

$$p^* = \min_{\substack{x \in K \\ Ax = b}} \langle c, x \rangle \qquad \text{(Primal convex conic program)}$$

We will call this the primal SDP. Here $A \in \mathbb{R}^{N \times M}$ is a constraint matrix. The dual SDP is derived by introducing the Lagrangian:

$$\mathcal{L}(x, \lambda, y) = c^T x + y^T(b - Ax) - \lambda^T x$$

Lemma 5.9. *The optimal solution p^* to the primal problem satisfies:*

$$p^* = \min_{x \in \mathbb{R}^N} \max_{y \in \mathbb{R}^M, \lambda \in K^*} \mathcal{L}(x, \lambda, y)$$

Proof:

Notice that:

$$\max_{y \in \mathbb{R}^M, \lambda \in K^*} \mathcal{L}(x, \lambda, y) = \begin{cases} \infty & b \neq Ax \\ \infty & x \notin K \\ c^T x & \text{otherwise} \end{cases}$$

□

Now let's switch the max and mins of the lemma above. This will give us the dual program. We will use the following fact, which is an exercise to prove:

Lemma 5.10. *Let $F : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$ be any function. Then*

$$\min_{\alpha} \max_{\beta} F(\alpha, \beta) \geq \max_{\beta} \min_{\alpha} F(\alpha, \beta)$$

Applying this to the Lagrangian min-max problem, we get a problem whose solution is a lower bound :

$$d^* = \max_{y \in \mathbb{R}^M, \lambda \in K^*} \min_{x \in \mathbb{R}^n} \mathcal{L}(x, \lambda, y) \leq p^*$$

Looking at the inner minimum term, we get:

$$\min_{x \in \mathbb{R}^N} c^T x + y^T(b - Ax) - \lambda^T x = \begin{cases} y^T b & \text{if } c - A^T y - \lambda = 0 \\ -\infty & \text{otherwise} \end{cases}$$

Thus we arrive at the dual convex conic program:

$$d^* = \max_{\substack{y \in \mathbb{R}^M \\ c - A^T y \in K^*}} \langle y, b \rangle \qquad \text{(Dual convex conic program)}$$

and weak duality $p^* \geq d^*$ holds by Lemma 5.10. When $p^* = d^*$, we say strong duality holds.

Lecture 10/14

Example 5.11 (Linear Programs)**.** The convex conic programming problem can be applied to $K = (\mathbb{R}_{\geq 0})^n$ to get a duality formalism for linear programs. Note that $K^* = K$. In this case, the conic problem is:

$$\min_{\substack{x \geq 0 \\ Ax = b}} \langle c, x \rangle \qquad \text{(Primal LP)}$$

which is a linear program. The dual problem is:

$$\max_{\substack{y \in \mathbb{R}^M \\ A^T y \leq c}} \langle y, b \rangle \qquad \text{(Dual LP)}$$

The feasible domain of the primal linear program is a slice of the non-negative orthant, which is a polyhedron. Similarly, the feasible domain of the dual linear program is the intersection of a collecion of half spaces, which is also a polyhedron.

Example 5.12 (Semidefinite Programs)**.** Recall for SDPs, the primal problem takes the form:

$$\min_{\substack{X \in \mathrm{PSD}_n \\ \langle A_i, X \rangle = b_i}} \langle C, X \rangle_F \qquad \text{(Primal SDP)}$$

where $\langle \cdot, \cdot \rangle_F$ is the Frobenius inner product. The dual takes the form:

$$\max_{C - \sum y_i A_i \in \mathrm{PSD}_n} \langle y, b \rangle_F \qquad \text{(Dual SDP)}$$

Note how this coincides with our claim of the dual Relaxed Max Cut problem in the previous section.

Definition 5.13. A conic program is *strictly feasible* if the feasible region contains a point which is strictly in the interior of the cone.

Theorem 5.14 (KKT Conditions)**.** *Consider the primal and dual convex conic programs for any convex cone K. Assume that both are strictly feasible. By strong duality, $p^* = d^*$ and both values are attained. Then $(x, y) \in \mathbb{R}^N \times \mathbb{R}^M$ is primal-dual optimal if and only if:*

- (*Primal feasibility*) $x \in K, Ax = b$.
- (*Dual feasibility*) $c - A^T y \in K^*$.
- (*Complementary slackness*) $(c - A^T y)^T x = 0$.

Proof idea:

If x is primal-feasible and y is dual-feasible, then $\langle c, x \rangle \geq \langle b, y \rangle$. Since both programs are strictly feasible, there exist x and y where there is equality $\langle c, x \rangle = \langle y, b \rangle$. Then:

$$\begin{aligned} p^* = \langle c, x \rangle &= d^* \\ &= \langle b, y \rangle \\ &= \min_{x'} \mathcal{L}(x', \lambda, y) \\ &\leq \mathcal{L}(x, \lambda, y) \\ &= c^T x - \lambda^T x + y^t (b - Ax) \end{aligned}$$

Since $\lambda = c - A^T y$ is in the dual cone, $\lambda^T x \geq 0$. Moreover, $b - Ax = 0$. So:

$$\mathcal{L}(x, \lambda, y) \leq c^T x$$

Since the left and right hand side are the same, we find that $\lambda^T x = 0$. This is equivalent to the complementary slackness condition.

□

Remark 5.15. This suggests a way to solve convex conic programming: try to solve the complementary slackness equation $(c - A^T y)^T x = 0$ subject to the feasiblity constraints.

Example 5.16. Consider the following SDP:

$$p^* = \max_{\substack{X \in \text{PSD}_n \\ \text{Tr}(X)=1}} \langle C, X \rangle$$

The dual program will only have one variable, since there is only one linear constraint on X. It is:

$$\min_{tI - C \in \text{PSD}_n} t$$

This is now easier to solve. Consider the eigendecomposition $C = Q\Lambda Q^T$, where $\Lambda = \text{diag}(\lambda_1, ..., \lambda_n)$. We assume $\lambda_1 \geq ... \geq \lambda_n$. Then:

$$tI - Q\Lambda Q^T = Q \begin{pmatrix} t - \lambda_1 & & \\ & \ddots & \\ & & t - \lambda_n \end{pmatrix} Q^T$$

The the eigenvalues of $tI - C$ are $t - \lambda_i$. Therefore $tI - C$ is PSD if and only if $t \geq \max\{\lambda_i\}$. Then the solution to the dual SPD is $t = \lambda_1$. Note that strict primal and dual feasiblity hold (exercise), and so strong duality holds. Thus the primal optimum p^* is also λ_1.

5.4 Interior Point Algorithms ❖

There are polynomial time algorithms to solve SDPs with desired accuracy. One of the most standard algorithms to do this is the interior point method. The idea is to replace the constraint $X \in \text{PSD}_n$ using a barrier function for PSD_n. That is, consider a function which is finite on the interior of PSD_n and approaches $+\infty$ as you reach the boundary of PSD_n (see Figure 5.2). One such boundary function is:

$$-\mu \log \det X : \text{PSD}_n \to \mathbb{R}$$

Then we add that to the cost function:

$$\langle C, X \rangle - \mu \log \det X$$

Then our only constraint is $\langle A_i, X \rangle = b_i$, and the new optimization problem is:

$$\min_{\langle A_i, X \rangle = b_i} \langle C, X \rangle - \mu \log \det X \tag{†}$$

This is not equivalent to the original SDP, but for small enough μ it is approximately equivalent. Note our chosen boundary function is convex, which means (†) is a convex optimization problem subject to linear constraints, which is much easier. The idea of the interior point method, then, is to generate a sequence $(\mu_k, X_k)_{k=1}^{\infty}$, where X_k is in the interior of PSD_n and approximately solves the convex optimization problem (†) with $\mu = \mu_k$ and $\mu_k \to 0$ as $k \to \infty$. Then $\lim_{k \to \infty} X_k$ exists, lies on ∂PSD_n and solves the SDP.

Lecture 10/19

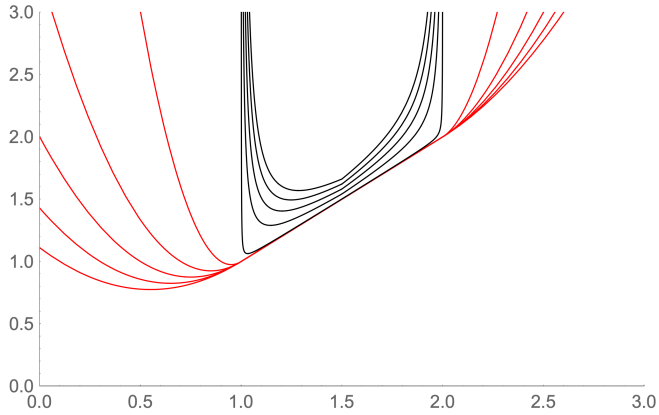

Figure 5.2: A schematic of a boundary function as $\mu \to 0$. The feasible region ("cone") is the line segment supported on $[1, 2]$, and anywhere outside of that becomes infinitely penalized as $\mu \to 0$.

5.5 Sums of Squares Problems and SDPs ❖

Consider a problem of the form:

$$\min_{\substack{x \in \mathbb{R}^n \\ f_i(x) \leq 0 \\ h_i(x) = 0}} f_0(x)$$

where $f_0, f_1, ..., f_m$ and $h_1, ..., h_p$ are all polynomials in $\mathbb{R}^n$. A linear program is an example of such a problem, where everything is linear. Another example of such a problem is a quadratic program. We can turn this optimization problem into an equivalent feasibility/infeasibility problem:

$$\max_{\gamma \in \mathbb{R}} \gamma \quad \text{such that} \begin{Bmatrix} f_i(x) \leq 0 & \forall i = 1, ..., m \\ h_i(x) = 0 & \forall i = 1, ..., p \\ f_0(x) \leq \gamma \end{Bmatrix} \text{ is feasible}$$

This is choosing the largest paramter γ such that a system of constraints depending on that parameter has no solution. This will turn out to let us relax polynomial optimization problems because there exist characterizations ("certificates") for when a polynomial system is infeasible.

We start with the unconstrained case, where we are simply minimizing a polynomial $f_0(x)$ over $\mathbb{R}^n$. When $\deg(f_0) = 2$, this is easy to solve using eigendecompositions of the associated quadratic form. When $\deg(f_0) = 3$, there are more non-trivial (but efficient) algorithms to solve this. However, this is NP-hard when $\deg(f_0) \geq 4$ and $n \geq 2$. The feasibility version of this problem is finding $x \in \mathbb{R}^n$ such that $f_0(x) < 0$. If indeed $\min(f_0) < 0$, then to answer this it suffices to find an x such that $f_0(x) < 0$. However, if $\min(f_0) \geq 0$, then how do we demonstrate that? One way is to write f_0 as a sum of squares:

$$f_0 = \sum_{i=1}^{s} g_i^2$$

These are known as sums of squares certificates. A few immediate questions might be:

Q: If $f_0 \geq 0$, do there always exist g_i which are polynomial?

[A] The answer is no.

Q: How can we search for such polynomials g_i?

[A] Use semidefinite programs.

Example 5.17. Consider the following polynomial:

$$M(x,y) = x^2y^4 + x^4y^2 + 1 - 3x^2y^2$$

This polynomial is always at least zero by applying the arithmetic-geometric mean inequality:

$$\frac{x^2y^4 + x^4y^2 + 1}{3} \geq \left((x^2y^4)(x^4y^2)(1)\right)^{1/3} = x^2y^2$$

However, it cannot be written as a sum of squares, which was proven by Motzkin in the 1960s.

Hilbert showed that there exist non-negative polynomials in n variables of degree $2d$ which is not a sum of squares, if and only if $n \neq 1$ and $d \neq 2$, with the exception of $(n,d) = (2,4)$. However, the sum of squares certificates g_i don't need to be polynomials, so there is hope. In the 1920's, Artin showed that every non-negative polynomial can be written as the sum of squares of rational functions.

Example 5.18. Let $f = 4x^4 + 4x^3y - 7x^2y^2 - 2xy^3 + 10y^4$. We can write f as:

$$f = \begin{bmatrix} x^2 & xy & y^2 \end{bmatrix} \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{12} & a_{22} & a_{23} \\ a_{13} & a_{23} & a_{33} \end{bmatrix} \begin{bmatrix} x^2 \\ xy \\ y^2 \end{bmatrix}$$

Writing this out and equating coefficients, we get a system of equations for the a_{ij}. This does not have a unique solution, since there are 5 equations but 6 variables. The line of solutions is:

$$A(\lambda) = \begin{bmatrix} 4 & 2 & -\lambda \\ 2 & -7+2\lambda & -1 \\ -\lambda & -1 & 10 \end{bmatrix}$$

If we can choose λ such that $A(\lambda) \in \text{PSD}_3$, then we claim we would have a sum of squares certificate for f. Indeed, if $A(\lambda_0)$ is psd, then we can factor $A(\lambda_0)$ as U^TU via the eigendecomposition. Then:

$$\begin{bmatrix} x^2 & xy & y^2 \end{bmatrix} UU^T \begin{bmatrix} x^2 \\ xy \\ y^2 \end{bmatrix} = \sum_{i=1}^{3} \langle u_i, [x^2\ xy\ y^2] \rangle^2$$

where u_i are the columns of U. This is a sum of square representation of f. It turns out that $A(6)$ is psd, and the factorization is:

$$U(6) = \begin{bmatrix} 0 & 2 \\ 2 & 1 \\ 1 & -3 \end{bmatrix}$$

The sum of squares certificate is then:

$$f = (0 \cdot x^2 + 2 \cdot xy + 1 \cdot y^2)^2 + (2 \cdot x^2 + 1 \cdot xy - 3 \cdot y^2)^2$$

Which one can verify by expanding out.

The above example generalizes: given $f \in \mathbb{R}[x_1, ..., x_n]$ of degree $2d$, let z be the vector of all monomials in the variables $x_1, ..., x_n$ of degree $\leq d$. If we can write $f = z^TAz$ for some $A \in \text{PSD}_N$, then we have a sum of squares certificate via the eigendecomposition of A. This is an SDP in the matrix variable A (with no particular objective). The problem with this is that A is very large, on the order of $n^d \times n^d$.

Remark 5.19. A polynomial f as above admits a sum of squares representation if and only if this SDP is feasible.

Example 5.20 (Shor)**.** Consider the problem of minimizing $f(x,y)$ where

$$f(x,y) = 4x^2 - \frac{21}{10}x^4 + \frac{1}{3}x^6 + xy - xy^2 + 4y^4$$

This has many local minima, and hence this is a very non-convex problem. We can convert this to the problem

$$\max_{\gamma \in \mathbb{R}} \gamma \quad \text{such that } f(x,y) - \gamma \text{ is a sum of squares}$$

This gives a lower bound on the true global minimum. We can reformulate this problem into an SDP. That is:

$$\begin{array}{c} \max \\ A, \gamma \\ A \in \text{PSD} \\ z^T A z = f - \gamma \end{array} \quad \gamma$$

This can be solved (say, via the interior point method) and the optimal value comes out to be $\gamma = 1.0316$. This turns out to be the global minimum.

Lecture 10/26

5.5.1 The Positivstellensatz

Now we return to the constrained case of polynomial optimization. In this case, how can we certify a system of polynomial inequalities and equalities has no feasible point? To warm up, suppose we only have a system of equalities, i.e. $h_i(x) = 0$ for $i = 1, ...m$ for polynomials h_i in n variables. Then the answer to this is given by the Nullstellensatz:

Theorem 5.21 (Nullstellensatz)**.** *Let $h_1, ..., h_m \in \mathbb{C}[x_1, ..., x_n]$. Then the system of equations $\{h_i(x) = 0\}$ is infeasible (over $\mathbb{C}$) if and only if there exist polynomials $p_1, ..., p_m \in \mathbb{C}[x_1, ..., x_n]$ such that*

$$p_1(x)h_1(x) + ... + p_m(x)h_m(x) = 1$$

The polynomials $p_1, ..., p_m$ are a certificate of infeasibility for the system of equations. Therefore if we want to try to prove infeasibility, we can search for a Nullstellensatz certificate. One way to do this is to search by degree. Start by assuming $\deg(p_i) = 0$ and try to solve for the p's, which is a linear system. If that doesn't work, assume $\deg(p_i) \leq 1$ and try to solve that resulting linear system as well. Continue this way, increasing the degree by one, until you find a certificate. This shows that the search for certificates is computationally tractible, as it consists of solving a sequence of linear systems. If n is the number of variables, m is the number of equations, and d_i is the degree of h_i, then there exists $B(n, m, d_i) \in \mathbb{Z}$ such that if there exists a certificate, it must be one with degree at most B. Unfortunately, this bound B is very big, so in the worst case this process isn't great. However, often there exists a low degree certificate to make this process reasonably efficient.

If we also have inequalities as well as equalities, there is an analogous result.

Theorem 5.22 (Positivstellensatz)**.** *Let $f_1, ..., f_p, h_1, ..., h_m \in \mathbb{R}[x_1, ..., x_n]$. Then the system $\{f_i(x) \geq 0, h_j(x) = 0\}$ is infeasible in $\mathbb{R}^n$ if and only if there is a certificate of the following form:*

$$\begin{aligned} -1 = & SOS f_1 + ... + SOS f_m \\ & + SOS f_1 f_2 + ... + SOS f_{m-1} f_m \\ & \cdots \\ & + SOS f_1 f_2 \cdots f_p + \\ & + p_1 h_1 + ... + p_m h_m \end{aligned}$$

where "SOS" denotes a polynomial which is a sum of squares and $p_1, ..., p_m$ are polynomials.

There are $2^p + m$ polynomials that comprise a Positivstellensatz certificate, which is very large even for reasonable p. Parillo and Lasserre independently and concurrently had the idea of searching for Positivstellensatz certificates by truncating the degree to get a heirarchy similar to that of the Nullstellensatz:

- Degree 0: The SOS's are all constants ≥ 0 and the polynomials p_j are any constants.

- Degree 1: The SOS's are sums of squares of degree at most one polynomials and p_j are degree at most one.
- Degree 2: etc.

The Positivstellensatz says that the system $\{f_i(x) \geq 0, h_j(x) = 0\}$ is infeasible if and only if there is a certificate of some finite degree in the above heirarchy. The search for such a certificate is computationally tractible (i.e. polynomial time) because the search can be phrased as a sequence of feasibility problems for SDPs.

6. Statistical Parameter Estimation

❖

Lecture 10/28

Suppose we have a paramaterized family of probability distributions D_θ on $\mathbb{R}^d$ parameterized by $\theta \in \Theta \subseteq \mathbb{R}^M$. Let D_θ have probability density function $p(x \mid \theta)$. Suppose we observe iid draws $\{x_1, ..., x_n\}$ of D_θ for some unkown θ. How can we estimate the value of θ given these samples? An *estimator* $\hat{\theta}$ is a function that takes iid samples and return a value in Θ that estimates the true parameter θ.

Definition 6.1. Given an estimator $\hat{\theta}$, if $\mathbb{E}_\theta[\hat{\theta}] = \theta$, then we say that $\hat{\theta}$ is *unbiased*. Moreover, if we denote $\hat{\theta}_n$ to be the estimator that uses the first n draws, then $\hat{\theta}$ is called *consistent* if $\hat{\theta}_n \to \theta$ as $n \to \infty$.

Example 6.2. The multivariate Gaussian family of distributions has parameters $\theta = (\mu, \Sigma)$, where $\mu \in \mathbb{R}^d$ and $\Sigma \in \mathbb{R}^{d \times d}$ is the covariance matrix (which must be positive definite). The pdf of these distributions is:

$$p(x \mid \theta) = \frac{1}{(2\pi)^{d/2}\sqrt{|\Sigma|}} \exp(-(x-\mu)^T \Sigma^{-1}(x-\mu)/2)$$

Given iid samples $\{x_1, ..., x_n\}$, the most obvious estimator for μ and Σ would be the sample mean μ_n and sample covariance Σ_n.

Example 6.3. A mixture of Gaussians consists of r Gaussians on $\mathbb{R}^d$ with parameters $(\mu_1, \Sigma_1), ..., (\mu_r, \Sigma_r)$ and mixing weights $\pi_1, ..., \pi_r$ so that $\pi_1 + ... + \pi_n = 1$ and $\pi_i \geq 0$. The way to sample from a Gaussian mixture is to first sample $j \in [r]$ from the discrete distribution defined by the π_i, (e.g. pick j according to the weights π_i) and then sample from $\mathcal{N}(\mu_j, \Sigma_j)$. There is a universal approximation theorem that says that we can approximate any continuous probability distribution on $\mathbb{R}^d$ as a Gaussian mixture for large enough r.

Example 6.4. Let G be a finite group acting on $\mathbb{R}^d$. Let $s \in \mathbb{R}^d$ be an unkown signal, and suppose we randomly observe $g \cdot s + \epsilon$ for g uniformly distributed in G and $\epsilon \sim \mathcal{N}(0, \sigma^2 I)$. Given many such samples, the problem of finding s (or $g \cdot s$ for some $g \in G$) is called the orbit retrieval problems. This can be modeled as a mixture of Gaussians. For example, when G is cyclic of order n, the problem is called the Multi-Reference Alignment problem (MRA). In this case, the samples look like $x = R_\ell s + \epsilon$, where $R_\ell : \mathbb{R}^d \to \mathbb{R}^d$ rotates the coordinates of a vector by ℓ spots.

6.1 Maximum Likelihood Estimation and Method of Moments ❖

One of the most common examples of an estimator is the Maximum Likelihood Estimator (MLE). Let θ^* be the (unkown) true parameter which governs the sampled data. The likelihood function is defined to be:

$$\mathcal{L}(\theta; x) := p(x \mid \theta)$$

The MLE is then:

$$\hat{\theta}_{\text{MLE}} = \text{argmax}_{\theta \in \Theta} \mathcal{L}(\theta; x)$$

This estimator chooses parameters that render the observation x most likely.

Another common estimator is the Method of Moments (MoM). It proceeds by choosing a degree k and it equates population moments with sample moments and then solves for θ. More preciesly, we set up the following system of equations for θ:

$$\mathbb{E}_\theta[X^{\otimes j}] = \frac{1}{n}\sum_{i=1}^{n} x_i^{\otimes j}, \quad j = 1, ..., k$$

For a formal definition of "$\otimes$" as used above, see the next section. The RHS is numerical (calculated from data) and the LHS is (hopefully) analytical formulae in θ. Often there won't be an exact solution for given k, so we have to get the "closest" solution. Thus the kth MoM estimator is:

$$\hat{\theta}_{\text{MoM},k} := \text{argmin}_{\theta\in\Theta} \sum_{j=1}^{k} \|LHS_j - RHS_j\|^2$$

Both the MLE and MoM estimators are consistent (MoM for sufficiently high k), but they also tend to be biased. The main question is how to compute these estimators. They are both posed as non-convex optimization problems.

Example 6.5. Returning to the MRA problem, recall the model is $x = R_\ell s + \epsilon$. By linearity of expectation:

$$\mathbb{E}[x] = \mathbb{E}[R_\ell s] + \mathbb{E}[\epsilon] = \frac{1}{L}(s + R_1 s + R_2 s + ... + R_{L-1} s)$$

The vector $s + R_1 s + ... + R_{L-1} s$ is $(s_1 + ... + s_L)/L \cdot \mathbf{1}$. Therefore the first moment tells us the average of the signal s. The second moment is:

$$\begin{aligned}\mathbb{E}[x^{\otimes 2}] = \mathbb{E}[xx^T] &= \mathbb{E}[(R_\ell s)(R_\ell s)^T + (R_\ell s)\epsilon^T + \epsilon(R_\ell s)^T + \epsilon\epsilon^T] \\ &= \mathbb{E}[(R_\ell s)(R_\ell s)^T] + \mathbb{E}[\epsilon\epsilon^T]\end{aligned}$$

Where we used the fact that ℓ and ϵ are independent, so $\mathbb{E}[(R_\ell s)\epsilon^T] = \mathbb{E}[\epsilon(R_\ell s)^T] = 0$ because $\mathbb{E}[\epsilon] = 0$. The second moment of ϵ is $\mathbb{E}[\epsilon\epsilon^T] = \sigma^2 I$. We claim that the entries of the symmetrix matrix $\mathbb{E}[(R_\ell s)(R_\ell s)^T]$ look like:

$$\sum_{i}^{L} s_1 s_{i+h} \quad \forall h = 0, ..., L-1$$

Now the question follows: does knowledge of the above sums as well as the average of s uniquely determine the signal s?

Lecture 11/4

The answer is no, and an easier way to see this is to use the discrete Fourier transform (DFT). Taking the DFT of both sides of $x = R_\ell s + \epsilon$, one can show that:

$$\hat{x} = DFT(x) = \Omega^\ell DFT(s) + DFT(\epsilon)$$

where Ω is the diagonal matrix with entries $\{1, \omega, \omega^2, ..., \omega^{L-1}\}$. Since $DFT(\epsilon)$ is another Gaussian, we have traded the rotational shift action for a diagonal matrix action. In Fourier space, the first moment is:

$$\mathbb{E}[\hat{x}] = \frac{1}{2}\sum_{k=0}^{L-1} \Omega^k \hat{s} = \hat{s}[0]$$

where $\hat{s} = DFT(s)$. The second moment is:

$$\begin{aligned}\mathbb{E}[\hat{x}\hat{x}^\dagger] &= \mathbb{E}_\ell[(\Omega^\ell \hat{s} + \hat{\epsilon})(\Omega^\ell \hat{s} + \hat{\epsilon})^\dagger] \\ &= \mathbb{E}_\ell[\Omega^\ell \hat{s}\hat{s}^\dagger(\Omega^\dagger)^\ell] + \mathbb{E}_\ell[\Omega^\ell \hat{s}\hat{\epsilon}^\dagger] + \mathbb{E}_\ell[\hat{\epsilon}\hat{s}^\dagger\Omega^{-\ell}] + \mathbb{E}_\ell[\hat{\epsilon}\hat{\epsilon}^\dagger]\end{aligned}$$

Just as before, the two middle terms are zero because $\hat{\epsilon}$ has expected value 0. Thus this moment is $\mathbb{E}[\Omega^\ell \hat{s}\hat{s}^\dagger(\Omega^\dagger)^\ell]$ plus a (known) multiple of the identity (since σ is assumed to be known). By computing the expected value, one can show that:

$$\mathbb{E}[\hat{x}\hat{x}^\dagger] = \mathbb{E}[\hat{\epsilon}\hat{\epsilon}^\dagger] + \begin{cases} \hat{s}[i]\bar{\hat{s}}[j] & \text{if } i = j \\ 0 & \text{else} \end{cases}$$

Thus the second moment tells us $|\hat{s}[i]|^2$ for each i, which is not enough information to determine s yet. We therefore must compute the third moment, which we will not compute directly here. Up to terms involving the first and second moments, you get:

$$\mathbb{E}[\hat{x}^{\otimes 3}] \rightsquigarrow \begin{cases} \hat{s}[i]\hat{s}[j]\hat{s}[k] & \text{if } i + j + k = 0 \bmod L \\ 0 & \text{else} \end{cases}$$

This is called the bispectrum of s. This is, in fact, enough information to determine s (via $\hat{s}$) up to global rotation.

7. Tensor Decomposition

❖

Lecture 11/9

Definition 7.1. A *tensor* T of order d and lengths $n_1, ..., n_d$ is a real array $T \in \mathbb{R}^{n_1 \times ... \times n_d}$. The elements of this array are denoted $T_{i_1 i_2 \cdots i_d}$. A *symmetric tensor* of order d and length n is a tensor whose lengths are all n such that $T_{i_1 i_2 \cdots i_n} = T_{i_{\pi(1)} i_{\pi(2)} \cdots i_{\pi(d)}}$ for all permutations $\pi \in S_d$.

Vectors and matrices are both examples of tensors. We saw higher order tensors arise in the Method of Moments. For any random variable X on $\mathbb{R}^n$, the d-moment $\mathbb{E}[X^{\otimes d}]$ is symmetric. They also arise from computing higher order derivatives.

Definition 7.2. Given vectors $a^{(j)} \in \mathbb{R}^{n_j}$ for $j = 1, ..., d$, the (rank 1) tensor $a^{(1)} \otimes \cdots \otimes a^{(d)}$ is defined by:

$$(a^{(1)} \otimes \cdots \otimes a^{(d)})_{i_1 i_2 \cdots i_d} := a^{(1)}_{i_1} \cdots a^{(d)}_{i_d}$$

This is also known as the outer product of the vectors $a^{(j)}$. We write $a^{\otimes d} := a \otimes \cdots \otimes a$, which is a symmetric tensor.

Note that, while this outer product contains $n_1 \cdots n_d$ entries, it is merely specified by $n_1 + ... + n_d$ numbers. This is the central feature of tensor decomposition that is useful.

Lemma 7.3. *Any tensor $T \in \mathbb{R}_{n_1 \times \cdots \times n_d}$ can be written as as a sum of rank 1 tensors.*

Proof:

Given T, we can obviously write

$$T = \sum_{i_1=1}^{n_1} \cdots \sum_{i_d=1}^{n_d} T_{i_1 \cdots i_d} e_{i_1} \otimes e_{i_2} \otimes \cdots \otimes e_{i_d}$$

where e_j are standard basis vectors.

□

Lemma 7.4. *Any symmetric tensor T can be written as a linear combination of symmetric rank 1 tensors.*

Proof idea:

Consider $\text{span}(\{a^{\otimes d} \mid a \in \mathbb{R}^n\})$. Show that this is equal to $\text{Sym}^d(\mathbb{R}^n)$ by considering a symmetric tensor which is orthogonal to this span. Derive a contradiction.

□

Definition 7.5 (CP tensor decomposition)**.** Given a tensor $T \in \mathbb{R}^{n_1 \otimes \cdots \otimes n_d}$, if we can write T as

$$T = \sum_{i=1}^{R} \lambda_i (a^{(1,i)} \otimes \cdots \otimes a^{(d,i)})$$

where $a^{(j,i)} \in \mathbb{R}^{n_j}$ for all i and for all j and R is the smallest possible, then we call this the *CP decomposition* of T. Moreover we define the *CP rank* of T to be R.

Definition 7.6 (Symmetric CP tensor decomposition)**.** Given a symmetric tensor T, if we can write:

$$T = \sum_{i=1}^{R} \lambda_i a_i^{\otimes d}$$

where $a_i \in \mathbb{R}^n$ and R is the smallest possible, then we call this the *symmetric CP decomposition* of T. We call R the *symmetric CP rank* of T.

P. Comon conjectured that if T is symmetric, then the CP rank of T is always equal to the symmetric CP rank of T. Clearly, the former is bounded by the latter. However, in 2016 an explicit numerical counterexample was given.

Example 7.7. Let T be a matrix. The CP decomposition of T takes the form

$$T = \sum_{i=1}^{R} \lambda_i a_i b_i^T$$

This corresponds to the matrix factorization $A\Lambda B^T$, where A is the matrix whose columns are a_i, $\Lambda = \mathrm{diag}(\lambda_1, ..., \lambda_R)$ and B is the matrix whose columns are b_i. If T were symmetric, then the symmetric decomposition would correspond to a factorization $A\Lambda A^T$ (which is very similar to an eigendecomposition, except that the columns of A are not required to be orthogonal). We note that this decomposition is *not* unique.

Miraculously, the CP decomposition becomes unique for $d > 2$ and R "low-rank".

7.1 Computing the CP Decomposition ❖

References

❖

[BBP] Baik, Ben Arous, and Peche "Phase transition of the largest eigenvalue for non-null complex sample covariance matrices". arXiv:math/0403022 [math.PR].

[BSS] Bandeira, Singer, and Strohmer "Mathematics of Data Science". Unpublished Book Project 2021.

[BN08] Belkin and Niyogi "Towards a theoretical foundation for Laplacian-based manifold methods". 2008.

[BVZ19] Bryson, Vershyain, Zhao "Marchenko-Pastur law with relaxed independence conditions". arXiv:1912.12724 [math.PR].

[BW20] Bao and Wang, "Eigenvector distribution in the critical regime of BBP transition". arXiv:2009.13143 [math.PR].

[TB99] Tipping and Bishop, "Probabalistic Principal Component Analysis". Microsoft Research. 1999.