8/22/23:

Suppose given $\{\underline{x}_n, t\}_{n=1}^{N}$ where $\underline{x}_n$ is the observation and t_n is the corresponding target. How to match any $\underline{x}$ to some t? We construct a function $y(\underline{x})$ based on given data. This process called regression.

Linear regression: $$y(\underline{x}, \underline{w}) = w_0 + \sum_{j=1}^{N} w_j x_j .$$

Can extend this by considering nonlinear basis functions

$$y(\underline{x}, \underline{w}) = \sum_{j=0}^{M-1} w_j \phi_j(\underline{x}) = \underline{w}^T \underline{\phi}(\underline{x})$$

where by convention, $\phi_0(\underline{x}) = 1$.

Example basis functions:

- Polynomial basis $\phi_j(x) = x^j$
- Gaussian basis $\phi_j(x) = \exp\left(\frac{-(x-\mu_j)^2}{2s^2}\right)$
- Sigmoidal basis $\phi_j(x) = \sigma\left(\frac{x-\mu_j}{s}\right)$ with $\sigma(a) = \frac{1}{1+e^{-a}}$.

How to determine w_j's? Define the loss function

$$L(\underline{w}) = \frac{1}{2}\sum_{n=1}^{N}\left(t_n - \underline{w}^T \underline{\phi}(\underline{x}_n)\right)^2 = \frac{1}{2}\left\| \underline{t} - \underline{\underline{\Phi}}\,\underline{w} \right\|_2^2$$

where $$\underline{\underline{\Phi}} = \begin{bmatrix} - \underline{\phi}(\underline{x}_1) - \\ \vdots \\ - \underline{\phi}(\underline{x}_N) - \end{bmatrix} \in \mathbb{R}^{N\times M}, \quad \underline{\underline{\Phi}}_{nj} = \phi_j(\underline{x}_n).$$

Now, we wish to minimize $L(\underline{w})$

$$\frac{\partial L}{\partial \underline{w}} = \underline{\underline{\Phi}}^T\left(\underline{\underline{\Phi}}\,\underline{w} - \underline{t}\right) = 0 \Rightarrow \underline{w} = \underbrace{\left(\underline{\underline{\Phi}}^T\underline{\underline{\Phi}}\right)^{-1}\underline{\underline{\Phi}}^T}_{\text{Pseudoinverse}}\underline{t}$$

He shows polynomial example where overfitting occurs when M large. One solution: cross-validation. Leave a few points out and compare errors on test & training data.

In polynomial case, notice that the coefficients increase in magnitude telling us that this function is sensitive and oscillatory. We can use regularization by penalizing large w_j's.

L^2 regularization or ridge regression given by:

$$L^2(\underline{w}) = \frac{1}{2}\|\underline{\underline{\phi}}\,\underline{w} - \underline{t}\|_2^2 + \frac{\lambda}{2}\|\underline{w}\|_2^2 \,, \quad \lambda \geq 0$$

where $$\frac{dL^2}{d\underline{w}} = \underline{\underline{\phi}}^T(\underline{\underline{\phi}}\,\underline{w} - \underline{t}) + \lambda \underline{w} = 0$$

$$\Rightarrow \quad \underline{w} = (\underline{\underline{\phi}}^T \underline{\underline{\phi}} + \lambda \underline{\underline{I}})^{-1} \underline{\underline{\phi}}^T \underline{t} .$$

Could we prove that this reduces $\|w\|_2^2$? Can see asymptotically as $\lambda \to \infty$, $\underline{w} \to \underline{0}$.

Can replace 2-norm with another q. $q = 1$ is called Lasso regression:

$$L^1(\underline{w}) = L(\underline{w}) + \gamma \|\underline{w}\|_1^1 \qquad \left(\gamma = \frac{\lambda}{2}\right)$$

But this is no longer differentiable at $w_i = 0$.

8/24/23:

How to minimize $L^1(\underline{w})$? We can rewrite it as

$$L^1(\underline{w}) = \frac{1}{2}\|\underline{t}\|_2^2 + \frac{1}{2}\|\underline{\underline{\phi}}^T \underline{\underline{\phi}}\, \underline{w}\|_2^2 - 2\underline{t}^T \underline{\underline{\phi}}\, \underline{w} + \gamma \|\underline{w}\|_1$$

now, if we orthogonalize $\underline{\underline{\phi}}$, $\underline{\underline{\phi}}^T \underline{\underline{\phi}} = \underline{\underline{I}}$. And we can disregard the first term as we are minimizing w.r.t. $\underline{w}$.

$$\min_{\underline{w}} L^1(\underline{w}) \Leftrightarrow \min_{\underline{w}} \frac{1}{2}\|\underline{w}\|_2^2 - 2(\underline{t}^T \underline{\underline{\phi}})\, \underline{w} + \gamma \|\underline{w}\|_1$$

$$(\underline{\beta}^T = 2\underline{t}^T \underline{\underline{\phi}}) \qquad \Leftrightarrow \min_{\underline{w}} \sum_{i=1}^{n} \left(\frac{1}{2} w_i^2 - \beta_i w_i + \gamma |w_i|\right)$$

$$\Leftrightarrow \min_{w_i} \left(\frac{1}{2} w_i^2 - \beta_i w_i + \gamma |w_i|\right) \ \forall i$$

Now, observe that if $\beta_i \geq 0$, then for minimization, we want $w_i \geq 0$. And visa versa

Case $\beta_i \geq 0$

$$\min_{w_i} \left(\frac{1}{2} w_i^2 - \beta_i w_i + \gamma w_i\right)$$

Case $\beta_i \leq 0$

$$\min_{w_i} \left(\frac{1}{2} w_i^2 - \beta_i w_i - \gamma w_i\right)$$

$\partial_{w_i}: \quad w_i - \beta_i + \gamma = 0$

$\Rightarrow \quad w_i = \beta_i - \gamma \qquad\qquad \rightarrow \quad w_i = \beta_i + \gamma$

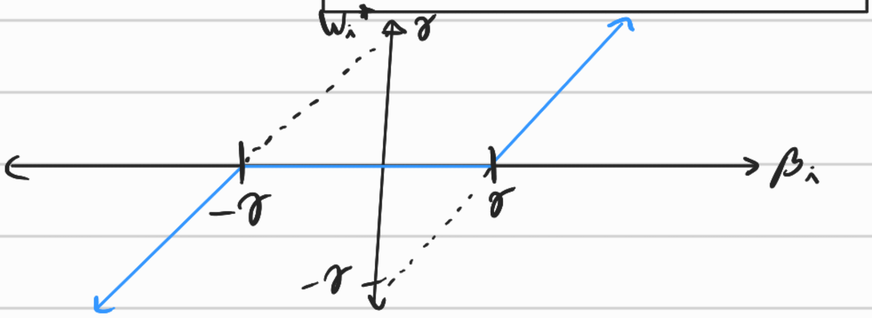
but for $w_i \geq 0$, $w_i = (\beta_i - \gamma)^+$ but for $w_i \leq 0$, $w_i = (\beta_i + \gamma)^- = -(-\beta_i - \gamma)^+$

Putting together, <u>Lasso soln:</u>

$$\boxed{w_i^* = \operatorname{sgn}(\beta_i)\left(|\beta_i| - \gamma\right)^+}$$

with

$$\boxed{\beta_i = 2(\underline{\Phi}^T \underline{t})_i}$$



This is called <u>soft thresholding</u>

Lecture 2: Classification

Suppose we wish to classify inputs $\underline{x}$. One strategy is to linearly separate the data into 2 classes by a <u>discriminant function</u>

$$y(\underline{x}) = \underline{w}^T\underline{x} + w_0 .$$

$y(\underline{x}) \geq 0 \rightarrow$ class 1, $\quad y(\underline{x}) < 0 \rightarrow$ class 2.

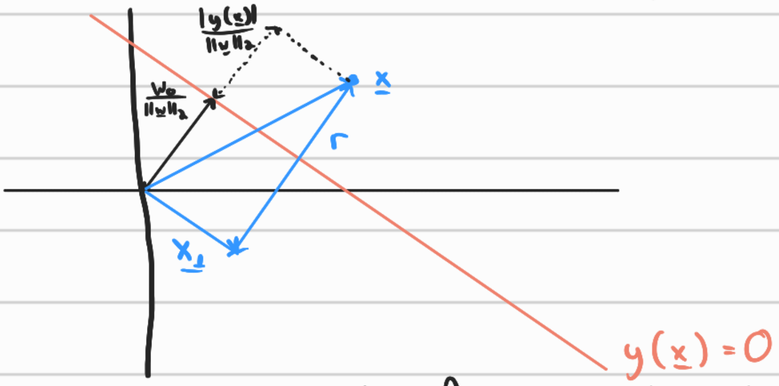
Denote L the hyperplane $y(\underline{x}) = 0$. The normal vector to L is $\nabla y(\underline{x}) = \underline{w}$, so $\underline{w} \perp L$. Suppose we want $a\underline{w} \in L$, so $a\underline{w}^T\underline{w} + w_0 = 0 \Rightarrow a = -w_0 / \|w\|_2^2$. And then $\|a\underline{w}\|_2 = \frac{w_0}{\|\underline{w}\|_2^2} \cdot \|w\|_2 = w_0 / \|\underline{w}\|_2$, the distance from the origin to the hyperplane. And, for any $\underline{x}$, the distance to the plane can be found by

$$\underline{x} = \underline{x}_\perp + r\frac{\underline{w}}{\|\underline{w}\|_2} \Rightarrow \underline{w}^T\underline{x} + w_0 = \underline{w}^T\underline{x}_\perp + r\frac{\underline{w}^T\underline{w}}{\|\underline{w}\|_2} + w_0$$

$$\Rightarrow y(\underline{x}) = 0 + r\|\underline{w}\|_2 + w_0$$

$$\rightarrow r = (y(\underline{x}) - w_0) / \|\underline{w}\|_2$$

$$\text{dist} = \left\| r \frac{\underline{w}}{\|\underline{w}\|_2} - a \underline{w} \right\| = \left| \frac{y(\underline{x}) - w_0}{\|\underline{w}\|_2} + \frac{w_0}{\|\underline{w}\|_2} \right| = \frac{|y(\underline{x})|}{\|\underline{w}\|_2} .$$



$\frac{|y(\underline{x})|}{\|\underline{w}\|_2}$
$\frac{w_0}{\|\underline{w}\|_2}$
$\underline{x}$
r
$\underline{x}_\perp$
$y(\underline{x}) = 0$

Now, if we want to classify several classes. One approach is to have K such functions, and $\underline{x} \in C_k$ if $y_k(\underline{x}) > y_j(\underline{x})$ $\forall\ j \neq k$. Then classes C_k & C_j divided by hyperplanes $(\underline{w_k} - \underline{w_j})^T \underline{x} + (w_{k0} + w_{j0}) = 0$.

8/29/23:

The above process can be written into a matrix operation.

Now, we will split to probabilistic generative modeling. Suppose we wish to approximate a normal distribution's mean and variance from some number of samples.

Probabilistic classification: Model class-conditional densities $\mathbb{P}[x \mid C_k]$, as well as class priors, $\mathbb{P}[C_k]$. We use those to approximate the posteriors, $\mathbb{P}[C_k \mid x]$. Consider only 2 classes

$$\text{Bayes}: \quad \mathbb{P}[C_1 \mid x] = \frac{\mathbb{P}[x \mid C_1] \cdot \mathbb{P}[C_1]}{\mathbb{P}[x \mid C_1]\mathbb{P}[C_1] + \mathbb{P}[x \mid C_2]\mathbb{P}[C_2]}$$

$$\left(\text{def } a = \ln\left(\frac{\mathbb{P}[x|C_1]\mathbb{P}[C_1]}{\mathbb{P}[x|C_2]\mathbb{P}[C_2]}\right)\right) = \frac{1}{1+e^{-a}} = \sigma(a),$$ The logistic sigmoid function.

We can find its inverse, the logit function: $$a = \ln\left(\frac{\sigma}{1-\sigma}\right).$$

For multi-class problems, turns to a sum. Define $a_k = \ln\left(\mathbb{P}[x|C_k]\mathbb{P}[C_k]\right)$,

$$\mathbb{P}[C_k \mid x] = \frac{\mathbb{P}[x \mid C_k]\mathbb{P}[C_k]}{\sum_j \mathbb{P}[x \mid C_j]\mathbb{P}[C_j]} = \frac{e^{a_k}}{\sum_j e^{a_j}} .$$

This last function is called softmax as it is a smooth version of max function. Commonly used at end of Neural Network to assign probabilities.

Now, suppose given samples $\{\underline{x}_n, t_n\}_{n=1}^{N}$ with $t_n = 1$ if $\underline{x}_n \in C_1$, and $t_n = 0$ if $\underline{x}_n \in C_2$. Suppose

$$\mathbb{P}[\underline{x} \mid C_k] \sim \mathcal{N}(\mu_k, \underline{\underline{\Sigma}}) \quad \text{for} \quad k = 1, 2.$$

Let $\mathbb{P}[C_1] = \pi$, so $\mathbb{P}[C_2] = 1 - \pi$.
Can compute liklihood function

$$p(\underline{t} \mid \pi, \underline{\mu_1}, \underline{\mu_2}, \underline{\underline{\Sigma}}) = \prod_{n=1}^{N} \left[\pi \mathcal{N}(\underline{x_n} \mid \underline{\mu_1}, \underline{\underline{\Sigma}})\right]^{t_n} \left[(1-\pi)\mathcal{N}(\underline{x_n} \mid \underline{\mu_2}, \underline{\underline{\Sigma}})\right]^{1-t_n}$$

Wish to optimize this subject to $\pi, \mu_1, \mu_2, \underline{\underline{\Sigma}}$. Easier to maximize log-liklihood:

$$\ln(p) = \sum_{n=1}^{N} \Big(t_n \ln(\pi) + t_n \ln(\mathcal{N}(\underline{x_n} \mid \underline{\mu_1}, \underline{\underline{\Sigma}}) + (1-t_n)\ln(1-\pi) + (1-t_n)\ln(\mathcal{N}(\underline{x_n} \mid \underline{\mu_2}, \underline{\underline{\Sigma}}) \Big).$$

Can find from this, $\pi^* = \frac{1}{N}\sum_{n=1}^{N} t_n$, $\underline{\mu_1} = \frac{1}{N_1}\sum_{n=1}^{N} t_n \underline{x_n}$, $\mu_2 = \frac{1}{N_2}\sum_{n=1}^{N}(1-t_n)\underline{x_n}$.
Trickier to find $\underline{\underline{\Sigma}}$.

PGM

The above called <u>probabilistic generative models</u>, we attempt to compute $\mathbb{P}[C_k \mid \underline{x}]$ given assumptions on $\mathbb{P}[C_k]$, $\mathbb{P}[\underline{x} \mid C_k]$. Solve for parameters, say mean, variance, on $\mathbb{P}[\underline{x} \mid C_k]$.
Alternatively, can do <u>probabilistic discriminative models</u>, PDM, we assume $\mathbb{P}[C_k \mid \underline{x}_n] = y_n(\underline{x}) = \sigma(\underline{w}^T \underline{x}_n)$, must find $\underline{w}$ through log liklihood,

$$p(\underline{t} \mid \underline{w}) = \prod_{n=1}^{N} y_n^{t_n}(1-y_n)^{1-t_n}.$$

Typically use gradient descent to optimize.

<u>9/5/23.</u>

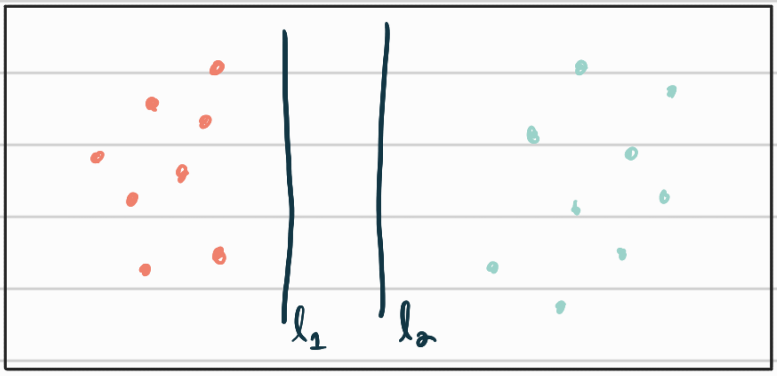
Before, we discussed 3 classification models

1) $y(\underline{x}) = \underline{w}^T \underline{\Phi}(\underline{x}) + w_0$, classify by sign of y
2) <u>Probabilistic model</u>: Bayes Theorem w/ probabilistic assumptions
3) <u>Discriminative model</u>: A sort of hybrid, model probabilities with sigmoid functions

Consider the two-class linear classifier

$$y(\underline{x}) = \underline{w}^T \underline{\phi}(\underline{x}) + b$$

where $\underline{\phi}$ consists of basis functions, $\underline{w}$ are weights, and b is the bias.
Given x_i, classify it by $\text{sign}(y(\underline{x}))$.
Suppose training data is separable $\{\underline{x_n}, t_n\}_{n=1}^{N}$, $t_n \in \{-1, 1\}$.
Consider following classifiers:



Here, l_2 is preferable as it is more robust to noise, perturbations. Define the margin as $\min_n \text{dist}(l, x_n)$. To be most robust, we should try to maximize the margin, mathematically,

$$\text{Choose} \quad \underline{w}, b = \underset{\underline{w}, b}{\text{arg max}} \ \min_n \ \text{dist}(x_n, l(\underline{w}, b)).$$

Recall from before the distance of any point to the divider given by $\text{dist}(\underline{x}, l(\underline{w}, b)) = |y(\underline{x})| / \|\underline{w}\|$. Assuming all data are properly classified, then $t_n y(\underline{x_n}) > 0 \ \forall \ n$, and $t_n \in \{-1, 1\}$. So, we have

$$\text{dist}(\underline{x_n}, l(\underline{w}, b)) = \frac{t_n y(\underline{x_n})}{\|\underline{w}\|} = \frac{t_n(\underline{w}^T \underline{\phi}(\underline{x_n}) + b)}{\|\underline{w}\|}.$$

Plugging this in above, the maximum margin sol'n given by

$$\underset{\underline{w}, b}{\text{arg max}} \left(\frac{1}{\|\underline{w}\|} \min_n \left(t_n \left(\underline{w}^T \underline{\phi}(\underline{x_n}) + b \right) \right) \right).$$

However, this is a tricky optimization problem. How can we simplify it? Note that $\underline{w}^T \underline{x} + b = 0 \Leftrightarrow K \underline{w}^T \underline{x} + K b = 0$ for $K \neq 0$. We can choose K s.t. $t_n(K \underline{w}^T \underline{\phi}(\underline{x_n}) + K b) = 1$ for the minimizing n. And transforming $\underline{w} \to K \underline{w}$, $b \to K b$, the problem simplifies to

$$\underset{\hat{\underline{w}},\,b}{\arg\max}\left(\frac{1}{\|\hat{\underline{w}}\|}\right) = \underset{\underline{w},\,b}{\arg\max}\left(\frac{1}{\|\underline{w}\|}\right)$$

satisfying the constraints $\quad t_n\left(\underline{w}^T \underline{\phi}(\underline{x}_n) + b\right) \geq 1 \quad$ for $\quad n = 1, 2, \ldots, N.$

And maximizing $1/\|\underline{w}\|$ is equivalent to minimizing $\frac{1}{2}\|w\|^2$. To minimize a quadratic relative to linear constraints, insert Lagrange multipliers $\{a_n\}_{n=1}^N$, and the Lagrangian function (KKT conditions)

$$L(\underline{w}, b, \underline{a}) = \frac{1}{2}\|\underline{w}\|^2 - \sum_{n=1}^{N} a_n\left(t_n\left(\underline{w}^T\underline{\phi}(\underline{x}_n) + b\right) - 1\right).$$

Taking the derivatives,

$$\frac{\partial L}{\partial \underline{w}} = \underline{w} - \sum_{n=1}^{N} a_n t_n \underline{\phi}(\underline{x}_n) = 0 \Rightarrow \underline{w} = \sum_{n=1}^{N} a_n t_n \underline{\phi}(\underline{x}_n)$$

$$\frac{\partial L}{\partial b} = -\sum_{n=1}^{N} a_n t_n = 0 \Rightarrow \sum_{n=1}^{N} a_n t_n = 0$$

$$\Rightarrow L = \frac{1}{2}\|\underline{w}\|^2 - \sum_{n=1}^{N} a_n t_n \underline{w}^T \underline{\phi}(\underline{x}_n) - \sum_{n>1}^{N} a_n t_n b + \sum_{n=1}^{N} a_n$$

$$= \frac{1}{2}\|\underline{w}\|^2 - \underline{w}^T\underline{w} - b \cdot 0 + \sum_{n=1}^{N} a_n$$

$$= \frac{-1}{2}\underline{w}^T\underline{w} + \sum_{n=1}^{N} a_n$$

$$= \sum_{n=1}^{N} a_n - \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} a_n a_m t_n t_m k(\underline{x}_n, \underline{x}_m), \quad k(\underline{x}_n, \underline{x}_m) = \underline{\phi}(\underline{x}_n)^T \underline{\phi}(\underline{x}_m)$$

is called the kernel

So the dual problem is given by

$$\underset{\underline{a}}{\arg\min} \sum_{n=1}^{N} a_n - \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} a_n a_m t_n t_m k(\underline{x}_n, \underline{x}_m) \Rightarrow \underline{w} = \sum_{n=1}^{N} a_n t_n \underline{\phi}(\underline{x}_n).$$

subject to $a_n \geq 0$, $\sum_{n=1}^{N} a_n t_n = 0$. This problem has N unknowns (one per training pt), $a_1, \ldots, a_N$, rather than M unknowns $w_1, \ldots, w_M$. When $M < N$, the dual problem is more expensive.

We can reformulate the problem in terms of kernel functions, $k(\underline{x}, \underline{y})$,

$$y(\underline{x}) = \sum_{n=1}^{N} a_n t_n k(\underline{x}, \underline{x}_n) + b.$$

Using KKT, we get that for each n, $a_n = 0$ or $t_n y(\underline{x_n}) = 1$. $\underline{x_n}$ satisfying $t_n y(\underline{x_n}) = 1$ are called <u>support vectors</u>.

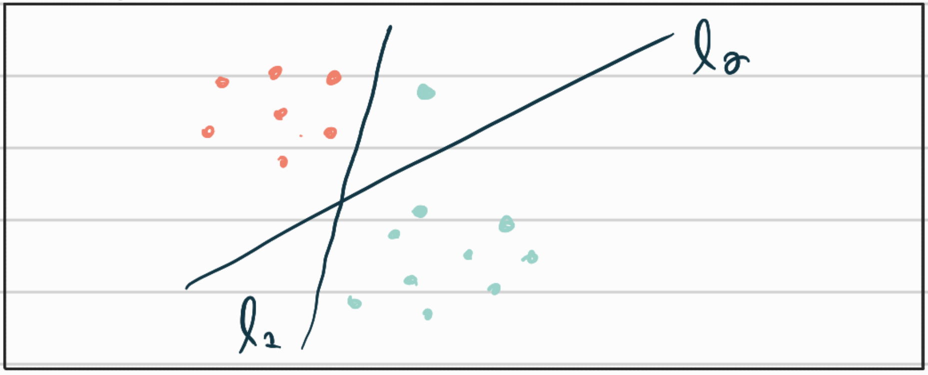
KKT yields $a_n \geq 0$, $t_n y(\underline{x_n}) - 1 \geq 0$, $a_n(t_n y(\underline{x_n}) - 1) = 0$.

> In modern optimization packages, this can be implemented via a hinge-loss function.

SVM Objective Function: $\sum_{n=1}^{\infty} E_\infty(t_n y(\underline{x_n}) - 1) + \lambda \|\underline{w}\|_2^2$,

$$E_\infty(z) = \begin{cases} 0 & z \geq 0 \\ \infty & z < 0 \end{cases}.$$

But consider following:



Here, l_2 is preferable, we need to allow a few misclassifications. Let $\xi_n = 1 - t_n y(\underline{x_n})$. The <u>soft-margin SVM</u> minimizes

$$\min \ \frac{1}{2}\|\underline{w}\|_2^2 + \lambda \sum_{n=1}^{N} \xi_n.$$

<u>9/7/23:</u>

This yields a new Lagrangian w/ corresponding KKT conditions

Now, suppose we wish to perform $\min_{\underline{x}} f(\underline{x})$, but there does not exist a closed form solution. Then we need to perform numerical optimization. <u>Gradient descent</u> iterates as

$$\underline{x}^{(t+1)} = \underline{x}^{(t)} - \eta_t \nabla f(\underline{x}^{(t)})$$

where η_t is a learning rate or step size.

Consider the quadratic function $f(\underline{x}) = \frac{1}{2}(\underline{x} - \underline{x}^*)^T \underline{\underline{Q}} (\underline{x} - \underline{x}^*)$ with $\underline{\underline{Q}}$ positive-definite. Well, $\nabla f(\underline{x}) = \underline{\underline{Q}}(\underline{x} - \underline{x}^*)$. And,

$$\underline{x}^{(t+1)} - \underline{x}^* = \underline{x}^{(t)} - \underline{x}^* - \eta_t \underline{\underline{Q}}(\underline{x}^t - \underline{x}^*) = (\underline{\underline{I}} - \eta_t \underline{\underline{Q}})(\underline{x}^{(t)} - \underline{x}^*)$$

$$\Rightarrow \|\underline{x}^{(t-1)} - \underline{x}^*\|_2 \leq \|\underline{\underline{I}} - \eta_t \underline{\underline{Q}}\|_2 \cdot \|\underline{x}^{(t)} - \underline{x}^*\|_2 .$$

(by symmetry/positive definite)

And, can show that $\|\underline{\underline{I}} - \eta_t \underline{\underline{Q}}\|_2 = \max\left(|1 - \eta_t \lambda_1(\underline{\underline{Q}})|, |1 - \eta_t \lambda_n(\underline{\underline{Q}})| \right)$

So the optimal η_t is then given by, splitting to 3 cases, choose minimum,

$$\eta_t = \frac{2}{\lambda_1(\underline{\underline{Q}}) + \lambda_n(\underline{\underline{Q}})} \Rightarrow \|\underline{\underline{I}} - \eta_t \underline{\underline{Q}}\| = \frac{\lambda_1(\underline{\underline{Q}}) - \lambda_n(\underline{\underline{Q}})}{\lambda_1(\underline{\underline{Q}}) + \lambda_n(\underline{\underline{Q}})} .$$

So, using this optimal η_t, the rate improves as $\lambda_1 \approx \lambda_n$, and decays as $\lambda_1 \gg \lambda_n$.

9/12/23:

Recall the condition number $K = \lambda_1 / \lambda_n$. Then we have the convergence rate

$$\|\underline{x}^{(t)} - \underline{x}^*\|_2 \leq \left(\frac{K - 1}{K + 1} \right)^t \|\underline{x}^{(0)} - \underline{x}^*\|_2 , \quad K = \frac{\lambda_1(\underline{\underline{Q}})}{\lambda_n(\underline{\underline{Q}})} > 1 .$$

Note, slow convergence for $K \gg 1$ as $\frac{K-1}{K+1} \xrightarrow{K \to \infty} 1$. And fast convergence for $K \to 1$ as then $\frac{K-1}{K+1} \xrightarrow{K \to 1^+} 0$.

The <u>exact line</u> search rule instead chooses (greedily)

$$\text{exact line:} \quad \eta_t = \underset{\eta \geq 0}{\arg\min}\, f(\underline{x}^{(t)} - \eta \nabla f(\underline{x}^{(t)})) .$$

In practice, computing eigenvalues are expensive, this is why exact line search may be better.

Theorem: If η_t chosen by exact line search for quadratic problem,

$$f(\underline{x}^{(t)}) - f(\underline{x}^{(*)}) \leq \left(\frac{K - 1}{K + 1} \right)^{2t} \|\underline{x}^{(0)} - \underline{x}^*\|_2 .$$

Note, this is in terms of objective values, f in 1-norm, rather than the 2-norm of $\|\underline{x}^{(t)} - \underline{x}^*\|_2$. Is this faster than constant step size?

Proof of theorem:

$$\eta_t = \underset{\eta > 0}{\arg\min}\, f(\underline{x}^{(t)} - \eta \underline{\underline{Q}}(\underline{x}^{(t)} - \underline{x}^*))$$

$$= \frac{1}{2} \underset{\eta > 0}{\arg\min} \left(\underline{x}^{(t)} - \eta \underline{\underline{Q}}(\underline{x}^{(t)} - \underline{x}^*) - \underline{x}^* \right)^T \underline{\underline{Q}} \left(\underline{x}^{(t)} - \eta \underline{\underline{Q}}(\underline{x}^{(t)} - \underline{x}^*) - \underline{x}^* \right)$$

$$= \frac{1}{2} \underset{\eta > 0}{\arg\min} \Big(\eta (\underline{x}^{(t)} - \underline{x}^*)^T \underline{\underline{Q}}^T \underline{\underline{Q}} \, \eta \, \underline{\underline{Q}} (\underline{x}^{(t)} - \underline{x}^*)$$

$$(\underline{\underline{Q}} = \underline{\underline{Q}}^T) \quad - 2\eta (\underline{x}^{(t)} - \underline{x}^*)^T \underline{\underline{Q}}^T \underline{\underline{Q}} (\underline{x}^{(t)} - \underline{x}^*) \Big)$$

Take ∂_η, set equal to zero,

$$\eta_t (\underline{x}^{(t)} - \underline{x}^*)^T \underline{\underline{Q}}^T \underline{\underline{Q}} \, \underline{\underline{Q}} (\underline{x}^{(t)} - \underline{x}^*)$$

$$- \quad (\underline{x}^{(t)} - \underline{x}^*)^T \underline{\underline{Q}}^T \underline{\underline{Q}} (\underline{x}^{(t)} - \underline{x}^*) \Big) = 0$$

$$\Rightarrow \quad \eta_t = \frac{(\underline{x}^{(t)} - \underline{x}^*) \underline{\underline{Q}}^T \underline{\underline{Q}}^T (\underline{x}^{(t)} - \underline{x}^*)}{(\underline{x}^{(t)} - \underline{x}^*) \underline{\underline{Q}}^T \underline{\underline{Q}} \, \underline{\underline{Q}} (\underline{x}^{(t)} - \underline{x}^*)} .$$

And by plugging this in & using Kantorovich's inequality, can arrive at final result.

9/14/23:

Now, we wish to extend the above to the general problem

$$\min_{\underline{x}} \quad f(\underline{x})$$

Suppose f is M-strongly convex and L-smooth, i.e.

$$0 \preceq M \underline{\underline{I}} \preceq \nabla^2 f(\underline{x}) \preceq L \underline{\underline{I}}, \quad \forall \ \underline{x},$$

or that each eigenvalue satisfies $M \leq \lambda_i(\nabla^2 f(\underline{x})) < L$ where we have

$$\nabla^2 f(\underline{x}) = \nabla \begin{bmatrix} \partial_{x_1} f(\underline{x}) \\ \vdots \\ \partial_{x_n} f(\underline{x}) \end{bmatrix} = \begin{bmatrix} \partial_{x_1}\partial_{x_1} f(\underline{x}) & \partial_{x_1}\partial_{x_2} f(\underline{x}) & \cdots & \partial_{x_1}\partial_{x_n} f(\underline{x}) \\ \partial_{x_2}\partial_{x_1} f(\underline{x}) & \partial_{x_2}\partial_{x_2} f(\underline{x}) & \cdots & \partial_{x_2}\partial_{x_n} f(\underline{x}) \\ \vdots & \vdots & \ddots & \vdots \\ \partial_{x_n}\partial_{x_1} f(\underline{x}) & \partial_{x_n}\partial_{x_2} f(\underline{x}) & \cdots & \partial_{x_n}\partial_{x_n} f(\underline{x}) \end{bmatrix}.$$

I.e., bounded above and below by quadratic functions.

Theorem: Let f be M-strongly convex and L-smooth. Then for gradient descent with $\eta_t = 2/(M+L)$ and condition number $K = \frac{2}{M+L}$,

$$\| \underline{x}^{(t)} - \underline{x}^* \|_2 \leq \left(\frac{K-1}{K+1} \right)^t \| \underline{x}^{(0)} - \underline{x}^* \|_2 \quad \text{and} \quad f(\underline{x}^{(t)}) - f(\underline{x}^*) \leq \frac{L}{2} \left(\frac{K-1}{K+1} \right)^{2t} \left(f(\underline{x}^{(0)}) - f(\underline{x}^*) \right)$$

A similar proof holds for f locally M-strongly convex & L-smooth

A weaker condition is regularity:

$$\langle f(\underline{x}), \underline{x} - \underline{x}^* \rangle \geq \frac{M}{2}\|\underline{x} - \underline{x}^*\|_2^2 + \frac{1}{2L}\|\nabla f(\underline{x})\|_2^2, \ \forall \underline{x}$$

which is a weaker form of convexity. Then, we can show with $\eta_t = \frac{1}{L}$, that

$$\|\underline{x}^{(t)} - \underline{x}^*\|_2^2 \leq \left(1 - \frac{M}{L}\right)^t \|\underline{x}^{(0)} - \underline{x}^*\|_2^2.$$

Alternatively, there is the Polyak-Lojasiewicz condition (PL),

$$\|\nabla^2 f(\underline{x})\|_2^2 \geq 2M\left(f(\underline{x}) - f(\underline{x}^*)\right) \ \forall \underline{x},$$

i.e., gradients grow quickly away from $\underline{x}^*$ and that stationary point is a global minimum. Then can prove, with $\eta_t = \frac{1}{L}$,

$$f(\underline{x}^{(t)}) - f(\underline{x}^*) \leq \left(1 - \frac{M}{L}\right)^t \left(f(\underline{x}^{(0)}) - f(\underline{x}^*)\right).$$

9/21/23:

Lemma: Let f be L-smooth. Then GD w/ $\eta_t = \frac{1}{L}$ obeys

$$f(\underline{x}^{(t+1)}) - f(\underline{x}^{(t)}) \leq \frac{-1}{2L}\|\nabla f(\underline{x}^{(t)})\|_2^2.$$

Use above Lemma and PL to show

Theorem: Suppose f satisfies PL condition and is L-smooth, then with GD, $\eta_t = 1/L$,

$$f(\underline{x}^{(t)}) - f(\underline{x}^*) \leq \left(1 - \frac{M}{L}\right)^t \left(f(\underline{x}^{(0)}) - f(\underline{x}^*)\right)$$

Theorem: Let f be convex & L-smooth, $\eta_t = \frac{1}{L}$, GD obeys

$$f(\underline{x}^{(t)}) - f(\underline{x}^*) \leq \frac{2L\|\underline{x}^{(0)} - \underline{x}^*\|_2^2}{t}$$

i.e., GD obtains ε accuracy in $O(\frac{1}{\varepsilon})$ iterations in terms of solution distance from minimizer.

If f not convex, but L-smooth, with $\eta_t = \frac{1}{L}$, then

$$\min_{0 \le k < t} \| \nabla f(\underline{x}^{(k)}) \|_2 \le \sqrt{\frac{2L(f(\underline{x}^{(0)}) - f(\underline{x}^*))}{t}}$$

note: Converges to stationary point, not necessary minima.

ex: $f(x,y) = x^2 - y^2$ w/ $(x_0, y_0) = (1, 0)$ will converge to $(0,0)$, a saddle point.

And if f convex, we get

$$\min_{t/2 \le k < t} \| \nabla f(\underline{x}^{(k)}) \|_2 \le \frac{4L \| \underline{x}^{(0)} - \underline{x}^* \|_2}{t}.$$

Overview of GD

- Exponential convergence for strongly convex
- Linear $(\frac{1}{t})$ convergence for L-convex
- Slower $(\frac{1}{\sqrt{t}})$ for non-conex, and to stationary point

9/26/23:

So far, we've formulated models as

$$\min f(\underline{x}) = \frac{1}{N} \sum_{i=1}^{N} L_i(\underline{x}) + R(\underline{x})$$

where R is a regulizer term. What to do if R not differentiable? Ex: Lasso

One example problem is matrix completion, such as given an incomplete matrix of movie recommendations, find missing movie recomendations.

Recall GD as

$$\underline{x}^{(t+1)} = \underline{x}^{(t)} - \eta_t \nabla f(\underline{x}^{(t)})$$

This is in fact equivalent to

$$\underline{x}^{(t+1)} = \arg\min_{\underline{x}} \left(f(\underline{x}^{(t)}) + \nabla f(\underline{x}^{(t)})^T (\underline{x} - \underline{x}^{(t)}) + \underbrace{\frac{1}{2\eta_t} \| \underline{x} - \underline{x}^{(t)} \|_2^2}_{\text{Proximal term}} \right)$$

$$= \arg\min_{\underline{x}} \left(\frac{1}{2} \| \underline{x} - (\underline{x}^{(t)} - \eta_t \nabla f(\underline{x}^{(t)})) \|_2^2 \right)$$

Why?
$$\frac{d}{d\underline{x}}\left(f(\underline{x}^{(t)}) + \nabla f(\underline{x}^{(t)})^T(\underline{x} - x^{(t)}) + \frac{1}{2\eta_t}\|\underline{x} - \underline{x}^{(t)}\|_2^2\right)$$

$$= \nabla f(\underline{x}^{(t)}) + \frac{1}{\eta_t}\left(\underline{x} - \underline{x}^{(t)}\right) = \underline{0}$$

$$\Rightarrow \underline{x} = \underline{x}^{(t)} - \eta_t \nabla f(\underline{x}^{(t)}).$$

And:
$$\frac{d}{dx}\left[\frac{1}{2}\|\underline{x} - (\underline{x}^{(t)} - \eta_t \nabla f(\underline{x}^{(t)}))\|_2^2\right]$$

$$= \underline{x} - \underline{x}^{(t)} + \eta_t \nabla f(\underline{x}^{(t)}) = 0$$

$$\Rightarrow \underline{x} = \underline{x}^{(t)} - \eta_t \nabla f(\underline{x}^{(t)})$$

Now, define the proximal operator

$$\text{prox}_h(\underline{w}) = \underset{z}{\arg\min}\left(\frac{1}{2}\|z - w\|_2^2 + h(\underline{z})\right)$$

for any convex function h. This allows us to express the GD update as

$$\underline{x}^{(t+1)} = \text{prox}_0\left(\underline{x}^{(t)} - \eta_t \nabla f(\underline{x}^{(t)})\right).$$

Consider the composite model

$$\min_{\underline{x}} F(\underline{x}) := \min_{\underline{x}} f(\underline{x}) + \lambda h(\underline{x}).$$

Then we can express proximal gradient descent as

$$\underline{x}^{(t+1)} = \text{prox}_{\lambda h}\left(\underline{x}^{(t)} - \eta_t \nabla f(\underline{x}^{(t)})\right)$$

$$= \underset{\underline{x}}{\arg\min}\left(\|\underline{x} - (\underline{x}^{(t)} - \eta_t \nabla f(\underline{x}^{(t)}))\|_2^2 + \lambda h(\underline{x})\right).$$

Ex: If $h(\underline{x}) = \|\underline{x}\|_1 = \sum |x_i|$, then

$$\text{prox}_{\lambda h}(\underline{x}) = \underset{\underline{z}}{\arg\min}\left(\|\underline{z} - \underline{x}\|_2^2 + \lambda \|z\|_2^2\right)$$

$$= \underset{\underline{z}}{\arg\min} \left(\frac{1}{2}\|\underline{z}\|_2^2 - \langle \underline{z}, \underline{x} \rangle + \overset{\text{not important}}{\cancel{\frac{1}{2}\|\underline{x}\|_2^2}} + \lambda \|\underline{z}\|_1 \right)$$

$$= \underset{\underline{z}}{\arg\min} \left(\frac{1}{2}\sum_{i=1}^{n} z_i^2 - \sum_{i=1}^{n} z_i x_i + \lambda \sum_{i=1}^{n} |z_i| \right)$$

$$\left(\begin{array}{c}\text{minimize} \\ \text{componentwise}\end{array}\right) = \underset{z_i}{\arg\min}\ \frac{1}{2} z_i^2 - z_i x_i + \lambda |z_i|$$

Want z_i to have same sign as x_i

$$= \underset{z_i}{\arg\min} \begin{cases} \frac{1}{2} z_i^2 - z_i x_i + \lambda z_i, & x_i \geq 0 \\ \frac{1}{2} z_i^2 - z_i x_i - \lambda z_i, & x_i < 0 \end{cases}$$

Take $\frac{d}{dz_i}$:

$$0 = \begin{cases} z_i - x_i + \lambda, & x_i \geq 0 \\ z_i - x_i - \lambda, & x_i < 0 \end{cases}$$

$$\Rightarrow z_i = \begin{cases} \max(0, x_i - \lambda), & x_i \geq 0 \\ \min(0, x_i + \lambda), & x_i < 0 \end{cases}$$

$$= \begin{cases} x_i - \lambda, & x_i \geq \lambda \\ x_i + \lambda, & x_i < \lambda \\ 0, & \text{otherwise} \end{cases}$$

= "soft thresholding"

So given $h(\underline{x}) = \|\underline{x}\|_1$,

$$\text{prox}_{\lambda h}(\underline{x})_i = \Psi_{st}(x_i ; \lambda).$$

This allows us to implement Lasso Proximal GD easily.

Some properties:

1) If $f(\underline{x}) = a g(\underline{x}) + b$, then $\text{prox}_f(\underline{x}) = \text{prox}_{ag}(\underline{x})$
2) If $f(\underline{x}) = g(\underline{x}) + \underline{a}^T \underline{x} + b$, $\text{prox}_f(\underline{x}) = \text{prox}_g(\underline{x} - \underline{a})$
3) If $f(\underline{x}) = g(\underline{x}) + \frac{\rho}{2}\|\underline{x} - \underline{a}\|_2^2$, $\text{prox}_f(\underline{x}) = \text{prox}_{\frac{1}{1+\rho}g}\left(\frac{1}{1+\rho}\underline{x} + \frac{\rho}{1+\rho}\underline{a}\right)$
4) If $f(\underline{x}) = g(a\underline{x} + \underline{b})$, $a \neq 0$, $\text{prox}_f(\underline{x}) = \frac{1}{a}\left(\text{prox}_{a^2 g}(a\underline{x} + b) - \underline{b}\right)$
5) If $f(\underline{x}) = g(\underline{\underline{Q}}\,\underline{x})$, $\underline{\underline{Q}}$ orthogonal, $\text{prox}_f(\underline{x}) = \underline{\underline{Q}}^T \text{prox}_g(\underline{\underline{Q}}^T \underline{x})$.
6) If $f(\underline{x}) = g(\underline{\underline{Q}}\,\underline{x} + \underline{b})$ with $\underline{\underline{Q}}^T \underline{\underline{Q}} = \alpha^{-1} \underline{\underline{I}}$, then

$$\text{prox}_f(\underline{x}) = (\underline{\underline{I}} - \alpha \underline{\underline{Q}}^T \underline{\underline{Q}})\underline{x} + \alpha \underline{\underline{Q}}^T\left(\text{prox}_{\alpha^{-1}g}(\underline{\underline{Q}}\,\underline{x} + \underline{b}) - \underline{b}\right)$$

7) If $f(\underline{x}) = g(\|\underline{x}\|_2)$, then $\text{prox}_f(\underline{x}) = \text{prox}_g(\|\underline{x}\|_2) \cdot \underline{x} / \|\underline{x}\|_2 \quad (\underline{x} \neq \underline{0})$

9/28/23:

Can prove some statements about proximal GD, $F(\underline{x}) = f(\underline{x}) + h(\underline{x})$

Assume f is convex & L-smooth, take $\eta_t = 1/L$

1) $F(\underline{x}^{(t+1)}) \le F(\underline{x}^{(t)})$, monotonicity towards goal

2) $\| \underline{x}^{(t+1)} - \underline{x}^* \|_2 \le \| \underline{x}^{(t)} - \underline{x}^* \|_2$.

3) $F(\underline{x}^{(t)}) - F^{opt} \le \frac{L \| \underline{x}^{(0)} - \underline{x}^* \|_2^2}{2t}$

Now, let f be μ-strongly convex:

4) $\| \underline{x}^{(t)} - \underline{x}^* \|_2^2 \le \left(1 - \frac{\mu}{L}\right)^t \| \underline{x}^{(0)} - \underline{x}^* \|_2^2$

Application: Backward Euler

Given $\frac{d\underline{h}}{dt} = \underline{f}(\underline{h}(t))$, backward Euler reads as

$$\underline{h}_{k+1} = \underline{h}_k + \eta \underline{f}(\underline{h}_{k+1}).$$

Let F be defined s.t. $F'(\underline{z}) = \underline{f}(\underline{z})$. Then defining

$$g(\underline{z}; \underline{h}_k) = \underline{z}^T \underline{z} - \underline{z}^T \underline{h}_k - \eta F(\underline{z}),$$

$$g'(\underline{z}; \underline{h}_k) = \underline{z} - \underline{h}_k - \eta \underline{f}(\underline{z}) = \underline{0}$$

$$\Rightarrow \underline{z} = \underline{h}_k + \eta \underline{f}(\underline{z}),$$

$$g''(\underline{z}, \underline{h}_k) = \underline{\underline{I}} - \nabla \underline{f}(\underline{z}) \succeq \underline{\underline{0}}$$ assuming $1 - \lambda_f > 0 \ \forall \lambda_f$.

So, we'll say that BE can be equivalent to

$$\underline{h}_{k+1} = \arg\min_{\underline{z}} g(\underline{z}; \underline{h}_k)$$

$$= \arg\min_{\underline{z}} \left(\frac{1}{2} \| \underline{z} - \underline{h}_k \|_2^2 - \eta F(\underline{z}) \right)$$

$$= \text{prox}_{-\eta F}(\underline{h}_k)$$

10/3/23:

Ex.: $f(x,y) = \frac{x^2}{a^2} + \frac{y^2}{b^2}$, $a = 1$, $b = 10^4$.

We have that $L \sim 10^4$, $\mu \sim 1$, $K = \frac{L}{\mu} \sim 10^4$. And we previously showed

$$f(\underline{x}^{(t)}) - f^* \sim O\left(\left(\frac{K-1}{K+1}\right)^t\right)$$

So $f(\underline{x}^{(t)}) - f^* \leq \varepsilon \Rightarrow \left(\frac{K-1}{K+1}\right)^t \leq \varepsilon \Rightarrow \left(\frac{K+1}{K-1}\right)^t \geq \frac{1}{\varepsilon}$

$\Rightarrow t \geq \log\left(\frac{1}{\varepsilon}\right) / \log\left(\frac{K+1}{K-1}\right) \approx 5000 \cdot \log\left(\frac{1}{\varepsilon}\right)$. So this is "slow" for $b \gg a$, but fast for $a \approx b$.

Can show that proximal GD acts as follows

1) Smooth & Strongly Convex: $O\left(K \log\left(\frac{1}{\varepsilon}\right)\right)$

2) Just smooth: $O\left(\frac{1}{\varepsilon}\right)$.

How can we accelerate this convergence?

Ideas: Implement information from previous iterations to utilize more information, and add buffers, ex. momentum, to yield a smoother trajectory.

The heavy-ball method includes a momentum term to mitigate zig-zags

$$\underline{x}^{(t+1)} = \underline{x}^{(t)} - \eta_t \nabla f(\underline{x}^{(t)}) + \underbrace{\theta_t\left(\underline{x}^{(t)} - \underline{x}^{(t-1)}\right)}_{\text{momentum}}$$

Consider again the quadratic problem

$$\min_{\underline{x}} \; (\underline{x} - \underline{x}^*)^T \underline{\underline{Q}} (\underline{x} - \underline{x}^*), \quad \underline{\underline{Q}} \succ 0, \text{ cond. num } K.$$

We can understand heavy-ball through dynamical systems:

$$\begin{bmatrix} \underline{x}^{(t+1)} \\ \underline{x}^{(t)} \end{bmatrix} = \begin{bmatrix} (1+\theta_t)\underline{\underline{I}} & -\theta_t \underline{\underline{I}} \\ \underline{\underline{I}} & \underline{\underline{0}} \end{bmatrix} \begin{bmatrix} \underline{x}^{(t)} \\ \underline{x}^{(t-1)} \end{bmatrix} - \begin{bmatrix} \eta_t \nabla f(\underline{x}^{(t)}) \\ \underline{0} \end{bmatrix}$$

or in terms of the target.

$$\begin{bmatrix} \underline{x}^{(t+1)} - \underline{x}^* \\ \underline{x}^{(t)} - \underline{x}^* \end{bmatrix} = \begin{bmatrix} (1+\theta_t)\underline{\underline{I}} & -\theta_t \underline{\underline{I}} \\ \underline{\underline{I}} & \underline{\underline{0}} \end{bmatrix} \begin{bmatrix} \underline{x}^{(t)} - \underline{x}^* \\ \underline{x}^{(t-1)} - \underline{x}^* \end{bmatrix} - \begin{bmatrix} \eta_t \nabla f(\underline{x}^{(t)}) \\ \underline{0} \end{bmatrix}$$

For the quadratic problem, this simplifies to

$$\begin{bmatrix} \underline{x}^{(t+1)} - \underline{x}^* \\ \underline{x}^{(t)} - \underline{x}^* \end{bmatrix} = \underbrace{\begin{bmatrix} (1+\theta_t)\underline{\underline{I}} - \eta_t \underline{\underline{Q}} & -\theta_t \underline{\underline{I}} \\ \underline{\underline{I}} & \underline{\underline{0}} \end{bmatrix}}_{\underline{\underline{H}}_t} \begin{bmatrix} \underline{x}^{(t)} - \underline{x}^* \\ \underline{x}^{(t-1)} - \underline{x}^* \end{bmatrix}.$$

To understand convergence, wish to understand the spectrum of $\underline{\underline{H}}_t$. Can change η_t, θ_t to change the spectrum. With $\rho(\underline{\underline{H}}_t)$, largest eigenvalue,

$$\left\| \begin{bmatrix} \underline{x}^{(t+1)} - \underline{x}^* \\ \underline{x}^{(t)} - \underline{x}^* \end{bmatrix} \right\|_2 \leq \rho(\underline{\underline{H}}_t) \left\| \begin{bmatrix} \underline{x}^{(t)} - \underline{x}^* \\ \underline{x}^{(t-1)} - \underline{x}^* \end{bmatrix} \right\|_2$$

Theorem: Convergence of Heavy Ball for Quadratic Function: Define f as above, suppose L-smooth, M-strongly convex. Choosing

$$\eta_t = \frac{4}{(\sqrt{L}+\sqrt{M})^2}, \quad \theta_t = \max\left(|1-\sqrt{\eta_t L}|, |1-\sqrt{\eta_t M}| \right)^2,$$

then with $K = L/M$,

$$\left\| \begin{bmatrix} \underline{x}^{(t+1)} - \underline{x}^* \\ \underline{x}^{(t)} - \underline{x}^* \end{bmatrix} \right\|_2 \lesssim \left(\frac{\sqrt{K}-1}{\sqrt{K}+1} \right)^t \left\| \begin{bmatrix} \underline{x}^{(1)} - \underline{x}^* \\ \underline{x}^{(0)} - \underline{x}^* \end{bmatrix} \right\|_2$$

so the complexity is now $O(\sqrt{K} \log \frac{1}{\varepsilon})$ instead of $O(K \log \frac{1}{\varepsilon})$.

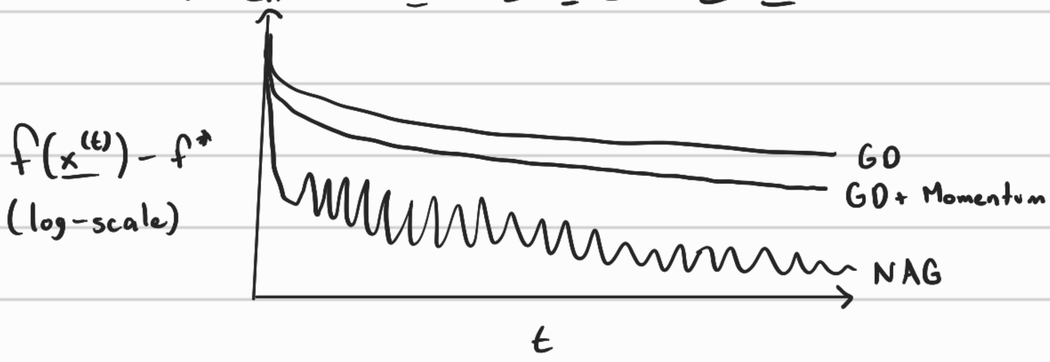
Proof by analyzing eigenvalues.

Another scheme is Nesterov's Accelerated Gradient Methods

$$x^{(t+1)} = y^{(t)} - \eta_t \nabla f(y^{(t)}) \; ; \; \underline{y}^{(t+1)} = \underline{x}^{(t+1)} + \frac{t}{t+3}\left(\underline{x}^{(t+1)} - \underline{x}^{(t)} \right)$$

- Each iteration about same cost as GD
- Uses gradient updates on $\underline{x}^{(t+1)}$, then extrapolation for $\underline{y}^{(t+1)}$
- Not a descent method, i.e. not strict descent.

Shows Numerical ex: $\min_{\underline{x}} \; \underline{x}^T \underline{\underline{L}}\, \underline{x} + \underline{x}^T \underline{e}_1$



Theorem: Convergence of Nesterov's Accelerated Gradient Method: Suppose f is convex & L-smooth. If $\eta_t = 1/L$, then

$$f(\underline{x}^{(t)}) - f^* \leq \frac{2L \, \| \underline{x}^{(0)} - \underline{x}^* \|_2^2}{(t+1)^2}$$

i.e., complexity is $O(\varepsilon^{-1/2})$ instead of $O(\varepsilon^{-1})$, much faster.

Aside: L-smooth if

$$\| \nabla f(\underline{x}) - \nabla f(\underline{y}) \|_2 \leq L \| \underline{x} - \underline{y} \|_2$$

10/5/23:

Recall following:

1) GD: $\underline{x}^{(t+1)} = \underline{x}^{(t)} - \eta_t \nabla f(\underline{x}^{(t)})$

2) HB: $\underline{x}^{(t+1)} = \underline{x}^{(t)} - \eta_t \nabla f(\underline{x}^{(t)}) + \theta_t (\underline{x}^{(t)} - \underline{x}^{(t-1)})$

3) NAG: $\underline{x}^{(t+1)} = \underline{y}^{(t)} - \eta_t \nabla f(\underline{y}^{(t)})$, $\underline{y}^{(t+1)} = \underline{x}^{(t+1)} + \frac{t}{t+3}(\underline{x}^{(t+1)} - \underline{x}^{(t)})$

And convergence rates

Method	Convex	Strongly Convex
GD	$O(1/t)$	$O\left(\left(\frac{\kappa-1}{\kappa+1}\right)^t\right)$
HB	*	$O\left(\left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^t\right)$
NAG	$O(1/t^2)$	*

Now, for GD, consider the limit $\eta_t \to 0$. Then, we get the ODE

$$\frac{d\underline{x}}{dt} = -\nabla f(\underline{x}).$$

This can give us intuitive ideas of what convergence should be.

Consider the same for Nesterov's method. Well, it's equivalent to

$$\underline{x}^{(t+1)} = \underline{x}^{(t)} + \frac{t-1}{t+2}\left(\underline{x}^{(t)} - \underline{x}^{(t-1)}\right) - \eta \nabla f\left(\underline{y}^{(t)}\right)$$

$$\Rightarrow \frac{\underline{x}^{(t+1)} - \underline{x}^{(t)}}{\sqrt{\eta}} = \frac{t-1}{t+2} \cdot \frac{\underline{x}^{(t)} - \underline{x}^{(t-1)}}{\sqrt{\eta}} - \sqrt{\eta}\, \nabla f\left(\underline{y}^{(t)}\right).$$

Now, let $\tau = t\sqrt{\eta}$, let $\underline{X}(\tau) = \underline{x}^{(\tau/\sqrt{\eta})} = \underline{x}^{(t)}$, and then $\underline{X}(\tau + \sqrt{\eta}) = \underline{x}^{(\tau/\sqrt{\eta}+1)} = \underline{x}^{(t+1)}$. Then, Taylor expanding:

$$\frac{\underline{x}^{(t+1)} - \underline{x}^{(t)}}{\sqrt{\eta}} \approx \underline{X}'(\tau) + \frac{1}{2}\underline{X}''(\tau)\sqrt{\eta}$$

and $$\frac{\underline{x}^{(t)} - \underline{x}^{(t-1)}}{\sqrt{\eta}} \approx \underline{X}'(\tau) - \frac{1}{2}\underline{X}''(\tau)\sqrt{\eta}.$$

Hence, putting this back in, letting $\underline{y}^{(t)} \to \underline{x}^{(t)}$ as $t \to \infty$

$$\underline{X}'(\tau) + \frac{1}{2}\underline{X}''(\tau)\sqrt{\eta} = \frac{\tau/\sqrt{\eta} - 1}{\tau/\sqrt{\eta} + 2}\left(\underline{X}'(\tau) - \frac{1}{2}\underline{X}''(\tau)\sqrt{\eta}\right) + \sqrt{\eta}\,\nabla f(\underline{X}(\tau))$$

Now, $\frac{\tau/\sqrt{\eta} - 1}{\tau/\sqrt{\eta} + 2} = \frac{\tau - \sqrt{\eta}}{\tau + 2\sqrt{\eta}} = 1 - \frac{3\sqrt{\eta}}{\tau + 2\sqrt{\eta}} \sim 1 - \frac{3\sqrt{\eta}}{\tau}$ as $\eta \to 0$. So,

$$\underline{X}'(\tau) + \frac{1}{2}\underline{X}''(\tau)\sqrt{\eta} \approx \left(1 - \frac{3\sqrt{\eta}}{\tau}\right)\left(\underline{X}'(\tau) - \frac{1}{2}\underline{X}''(\tau)\sqrt{\eta}\right) + \sqrt{\eta}\,\nabla f(\underline{X})$$

The order η^0 solution is $\underline{X}' = \underline{X}'$ ✓. The order $\sqrt{\eta}$ soln is

$$\underline{X}''(\tau) + \frac{3}{\tau}\underline{X}'(\tau) + \nabla f(\underline{X}(\tau)) = \underline{0}.$$

We assume $O(\eta)$ solution, $\underline{X}'' = 0$, is negligible.

How about the heavy ball method?

Let $\underline{m}^{(t)} := (\underline{x}^{(t)} - \underline{x}^{(t-1)})/\sqrt{\eta}$, $\Theta =: 1 - \gamma\sqrt{\eta}$ where $\gamma \geq 0$ is another hyperparameter.

$$\underline{x}^{(t+1)} = \underline{x}^{(t)} - \eta \nabla f(\underline{x}^{(t)}) + \Theta\left(\underline{x}^{(t)} - \underline{x}^{(t-1)}\right)$$

$$\Rightarrow \underline{m}^{(t+1)}\sqrt{\eta} = -\eta \nabla f(\underline{x}^{(t)}) + \Theta\, \underline{m}^{(t-1)}\sqrt{\eta}$$

$$\Rightarrow \quad \underline{m}^{(t+1)} = (1-\gamma\sqrt{\eta})\underline{m}^{(t)} - \sqrt{\eta}\,\nabla f(\underline{x}^{(t)}).$$

And, $$\underline{x}^{(t+1)} = \underline{x}^{(t)} + \sqrt{\eta}\,\underline{m}^{(t+1)}.$$

Letting $\eta \to 0$, changing to ODEs,

$$\underline{X}'(t) = \underline{M}(t), \quad \underline{M}'(t) = -\gamma \underline{M}(t) - \nabla f(\underline{X}(t))$$

$$\Rightarrow \quad \underline{X}''(t) + \gamma \underline{X}'(t) + \nabla f(\underline{X}(t)) = \underline{0}.$$

Hence, for Heavy Ball, we have

$$\underline{X}''(t) + \gamma \underline{X}'(t) + \nabla f(\underline{X}(t)) = \underline{0}, \quad \gamma \geq 0$$

and for Nesterov's Acceleration, $\tau = t\sqrt{\eta}$,

$$\underline{X}''(\tau) + \frac{3}{\tau}\underline{X}'(\tau) + \nabla f(\underline{X}(\tau)) = \underline{0}$$

Theorem: With this, for Nesterov, $f(\underline{X}(\tau)) - f^* \leq O(\frac{1}{\tau^2})$, matching the $O(1/t^2)$ convergence.

Proof Through defining Lyapunov function $E(\tau) := \tau^2(f(\underline{X}) - f^*) + 2\|\underline{X} + \frac{\tau}{2}\underline{X}' - \underline{X}^*\|_2^2$, and showing $E'(\tau) \leq 0$ if f convex, hence $f(\underline{X}(\tau)) - f^* \leq \frac{E(\tau)}{\tau^2}$ by def., $\frac{E(\tau)}{\tau^2} \leq \frac{E(0)}{\tau^2} = O(\frac{1}{\tau^2})$.

The FISTA (Fast-iterative shrinkage-thresholding algorithm) extends Nesterov to composite models, $\min_{\underline{x}} (f(\underline{x}) + h(\underline{x}))$, where f convex & smooth, h convex.

$$\begin{aligned}
\underline{x}^{(t+1)} &= \operatorname{prox}_{\eta_t h}(\underline{y}^{(t)} - \eta_t \nabla f(\underline{y}^{(t)})), \\
\underline{y}^{(t+1)} &= \underline{x}^{(t+1)} + \frac{\theta_t - 1}{\theta_{t+1}}(\underline{x}^{(t+1)} - \underline{x}^{(t)}), \\
\underline{y}^{(0)} &= \underline{x}^{(0)}, \quad \theta_0 = 1, \quad \theta_{t+1} = \frac{1+\sqrt{4+\theta_t^2}}{2}.
\end{aligned}$$

Can show: $\frac{\theta_t - 1}{\theta_{t+1}} = 1 - \frac{3}{t} + o(\frac{1}{t})$, and that $\theta_t \geq \frac{t+2}{2}$.

10/17/23:

We have formulated machine learning models as

$$\min_{\underline{x}} f(\underline{x}) = \min_{\underline{x}} \frac{1}{n}\sum_{i=1}^{n} \mathcal{L}_i(\underline{x}) + R(\underline{x})$$

where $\underline{x}$ is the parameter, $\mathcal{L}_i(\underline{x})$ are the loss terms, and R is a regulizer term. If n is large, computing ∇f is expensive. For now, ignore the regulizer, can use proximal GD to account for this.

Consider

$$\min_{\underline{x}} \frac{1}{n}\sum_{i=1}^{n} f(\underline{x} ; \underline{a}_i, y_i) =: \min_{\underline{x}} F(\underline{x})$$

where f can be quadratic say, $(\underline{a}_i^T \underline{x} - y_i)^2$. In stochastic gradient descent, SGD, let $\underline{g}(\underline{x}, \underline{\xi})$ be an unbiased estimator for $\nabla F(\underline{x})$ where $\underline{\xi}$ is a random variable. Then,

$$\underline{x}^{(t+1)} = \underline{x}^{(t)} - \eta_t \underline{g}(\underline{x}^{(t)}, \underline{\xi}^{(t)}).$$

How to choose $\underline{\xi}^{(t)}$? One option is sampling with replacement. So for each t, sample i_t from $\mathrm{Unif}\{1, \dots, n\}$, and choose $\underline{g}(\underline{x}^{(t)}, \underline{\xi}^{(t)}) = \nabla f(\underline{x}^{(t)}; \underline{a}_{i_t}, y_{i_t})$. Or, sample a batch $\{i_{t,1}, \dots, i_{t,m}\}$, and let $\underline{g}(\underline{x}^{(t)}, \underline{\xi}^{(t)}) = \frac{1}{m}\sum_{j=1}^{m} \nabla f(\underline{x}^{(t)}; \underline{a}_{i_{t,j}}, y_{i_{t,j}})$, which is called a minibatch SGD method. These methods are particularly efficient and avoid redundancy in the data.

Note: SGD is fast at the start, but slows as loss function decreases.

Convergence Analysis:

Assume F is μ-strongly convex, L-smooth, that $\underline{g}(\underline{x}, \underline{\xi})$ is an unbiased estimator of $\nabla F(\underline{x})$, and, that

$$E[\|\underline{g}(\underline{x}, \underline{\xi})\|_2^2] \leq \sigma_g^2 + c_g \|\nabla F(\underline{x})\|_2^2$$

which suggests, given $c_g = 1$, $\text{var}(\underline{g}) \le \sigma_g^2$.

Theorem: Convergence of SGD for strongly-convex: Under the assumptions above, if $\eta_t \equiv \eta \le 1/(Lc_g)$, then SGD (batch size 1) achieves

$$E[F(\underline{x}^{(t)}) - F^*] \le \frac{\eta L \sigma_g^2}{2\mu} + (1-\eta\mu)^t \left(F(\underline{x}^{(0)}) - F^*\right)$$

Proof in Bottou, Curtis, Nocedal, SIAM 2018.

Note: This expected difference is never zero, it suggests a strategy where we decrease η over time, called stagewise stepsize, so that the constant term, $\frac{\eta L \sigma_g^2}{2\mu} \to 0$. Additionally, if $\sigma_g = 0$, i.e. noiseless sampling / gradient eval, convergence is linear with rate $(1-\eta\mu)$.

There is not an equivalent theorem for just convex or non-convex problems.

Theorem: Convergence of SGD for strongly-convex, diminishing stepsize: Suppose F is μ-strongly-convex, and $c_g = 0$, i.e. bd. 2nd moment. If $\eta_t = \frac{\theta}{t+1}$ for some $\theta > \frac{1}{2\mu}$, then SGD (1) achieves

$$E[\|\underline{x}^{(t)} - \underline{x}^*\|_2^2] \le \frac{c_\theta}{t+1}$$

with

$$c_\theta = \max\left\{\frac{2\theta^2\sigma_g^2}{2\mu\theta - 1}, \|\underline{x}^{(0)} - \underline{x}^*\|_2^2\right\}.$$

I.e., convergence rate $O(\frac{1}{t})$ with $\eta_t \sim \frac{1}{t}$.

Additionally, we know informally that when minimizing strongly-convex problems in t steps, cannot do better than $\frac{1}{t}$, so SGD with $\eta_t \sim \frac{1}{t}$ is optimal.

As a summary, suppose want accuracy ε,

	iteration complexity	per-iteration cost	total cost
batch GD	$\log(1/\varepsilon)$	n	$n\log(1/\varepsilon)$
SGD (1)	$1/\varepsilon$	1	$1/\varepsilon$

So for n large, moderate ε, SGD has lower total cost. This is the big-data regime.

What if we lose strong convexity? Suppose $c_g = 0$ again, and that $\underline{g}$ an unbiased estimator of ∇F.
Do SGD, return $\tilde{\underline{x}}^{(t)} = \sum_{k=0}^{t} \frac{\eta_k}{\sum_{j=0}^{t} \eta_j} \underline{x}^{(k)}$. Then, can show w/ $\eta_t = \frac{1}{\sqrt{t}}$,

$$E[F(\tilde{\underline{x}}^{(t)}) - F^*] \lesssim \frac{\log(t)}{\sqrt{t}}$$

or generally,

$$E[F(\tilde{\underline{x}}^{(t)}) - F^*] \leq \frac{\frac{1}{2} E[\|\underline{x}^{(0)} - \underline{x}^*\|_2^2] + \frac{1}{2} \sigma_g^2 \sum_{k=0}^{t} \eta_k^2}{\sum_{k=0}^{t} \eta_k}$$

10/19/23:

SGD with long stepsizes poorly suppresses noise, tend to observe oscillations about minimum.
Idea: Return averaged solution

$$\overline{\underline{x}}^{(t)} = \frac{1}{t+1} \sum_{i=0}^{t} \underline{x}^{(i)}.$$

A popular SGD variant is SGD with Nesterov Momentum

$$\underline{v}^{(t+1)} = \gamma \underline{v}^{(t)} + \eta \underline{g}(\underline{x}^{(t)} - \gamma \underline{v}^{(t)}; \underline{\xi}^{(t)})$$

$$\underline{x}^{(t+1)} = \underline{x}^{(t)} - \underline{v}^{(t+1)}.$$

In practice, this performs about as well as SGD w/ momentum.
Another is Adagrad

$$x_i^{(t+1)} = x_i^{(t)} - \frac{\eta}{\sqrt{G_{ii}^{t} + \varepsilon}} g_i(\underline{x}^{(t)}; \underline{\xi}^{(t)})$$

where $G_{ii}^{t} = \sum_{j=0}^{t} |g_i(\underline{x}^{(j)}; \underline{\xi}^{(j)})|$ to weight earlier gradients more. (?)

One of most popular: Adam. Compute decaying averages of past and past gradients as

$$\underline{m}^{(t)} = \beta_1 \underline{m}^{(t-1)} + (1-\beta_1)\underline{g}^{(t)},$$

$$\underline{v}^{(t)} = \beta_2 \underline{v}^{(t-1)} + (1-\beta_2)(\underline{g}^{(t)})^2 \leftarrow \text{elementwise}$$

Then, correct for bias

$$\hat{\underline{m}}^{(t)} = \frac{1}{1-\beta_1^t}\underline{m}^{(t)} \; ; \; \hat{\underline{v}}^{(t)} = \frac{1}{1-\beta_2^t}\underline{v}^{(t)}.$$

And finally, update the parameter as

$$\underline{x}^{(t+1)} = \underline{x}^{(t)} - \underbrace{\frac{\eta}{\sqrt{\hat{\underline{v}}^{(t)}}+\varepsilon}\hat{\underline{m}}^{(t)}}_{\text{elementwise}}$$

typically with $\varepsilon = 10^{-8}$ so no divide by 0, $\beta_1 = 0.9$, and $\beta_2 = 0.999$.

High Dimensions:

Notation:

> $B^d(R)$ is the d-dimensional hyperball of radius R
$= \{\underline{x} \in \mathbb{R}^d : \|\underline{x}\|_2^2 \leq R^2\}$

> $S^d(R)$ is the d-sphere of radius R
$= \{\underline{x} \in \mathbb{R}^d : \|\underline{x}\|_2^2 = R\}$

> $C^d(R)$ is the d-dimensional hypercube, sidelength R
$= \{\underline{x} \in \mathbb{R}^d : \max(\underline{x}) \leq R\}$

> Let $B^d := B^d(1)$, $S^d := S^d(1)$, $C^d := C^d(\frac{1}{2})$.

Theorem: The volume of $B^d(R)$ is given by

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

where $\Gamma(n) = \int_0^\infty r^{n-1} e^{-r}\, dr.$

So, note that $\lim_{d\to\infty} \text{Vol}(B^d) = 0$!

Unit spheres in $\mathbb{R}^d$ have small volume. For unit volume, must have radius $R \sim \sqrt{\frac{d}{2\pi e}}$ as $d \to \infty$. It reaches a maximum at $d = 5$ and then goes rapidly to zero. Can prove thru integration. As a comparison, $\text{Vol}(C^d) = 1 \ \forall d$.

<u>Theorem:</u> Almost all volume of $B^d(R)$ lies near its equator.

10/24/23:

<u>Dimension reduction</u> involves mapping data in a high-dimensional space into a new space with smaller dimensionality. <u>Linear dimension reduction</u> does this with a linear mapping $\underline{x} \in \mathbb{R}^d \to \underline{\underline{W}}\,\underline{x} \in \mathbb{R}^n$, $n < d$.

A few methods we will cover:

1) <u>Random projection</u>: Probably approximately preserves pairwise distance
2) <u>Principal Component Analysis (PCA)</u>: Can nearly recover high-dim data
3) <u>Compressed sensing</u>: Acquire reduced data, reconstruct high-dim data

In random projection, a natural choice is to choose $\underline{\underline{W}}$ s.t.

$$P\left[\left|\frac{\|\underline{\underline{W}}\,\underline{x}_1 - \underline{\underline{W}}\,\underline{x}_2\|_2^2}{\|\underline{x}_1 - \underline{x}_2\|_2^2} - 1\right| > \varepsilon\right] \leq o(\varepsilon),$$

i.e., $\|\underline{\underline{W}}\,\underline{x}_1 - \underline{\underline{W}}\,\underline{x}_2\|_2^2 \approx \|\underline{x}_1 - \underline{x}_2\|_2^2$ in probability, nearly preserving pairwise distance. Defining $\underline{x} = \underline{x}_1 - \underline{x}_2$, simply need to look at $\frac{\|\underline{\underline{W}}\,\underline{x}\|_2^2}{\|\underline{x}\|_2^2}$, or $\|\underline{\underline{W}}\|_2^2$.

The transformation $\underline{x} \to \underline{\underline{W}}\,\underline{x}$ when W_{ij} independent random normal, we call it a <u>random projection</u>.

<u>Lemma:</u> Fix $\underline{x} \in \mathbb{R}^d$, let $\underline{\underline{W}} \in \mathbb{R}^{n \times d}$ be a random matrix such that each W_{ij} is an independent Gaussian $N(0,1)$ random variable. Then, for every $\varepsilon \in (0, 3)$, we have that

$$P\left[\left|\frac{\|(1/\sqrt{n})\,\underline{\underline{W}}\,\underline{x}\|_2^2}{\|\underline{x}\|_2^2} - 1\right| > \varepsilon\right] \leq 2e^{-\varepsilon^2 n/6}$$

<u>Lemma:</u> Let $Y \sim \chi_k^2$, then $\forall\ \varepsilon > 0$,

$$P[Y \leq (1-\varepsilon)k] \leq e^{-\varepsilon^2 k/6}$$

and, $\forall\ \varepsilon \in (0,3)$,

$$P\{ Y \geq (1+\varepsilon)k \big] \leq e^{-\varepsilon^2 k/6},$$

hence, $\forall\ \varepsilon \in (0,3)$,

$$P\big[(1-\varepsilon)k \leq Y \leq (1+\varepsilon)k\big] \geq 1 - 2e^{-\varepsilon^2 k/6}$$

Proof of original lemma: WLOG, assume $\|\underline{x}\|_2 = 1$. Then

$$P\left[\left|\frac{\|(1/\sqrt{n})\underline{\underline{W}}\,x\|_2^2}{\|\underline{x}\|_2^2} - 1\right| > \varepsilon\right]$$

$$= P\big[(1-\varepsilon)n \leq \|\underline{\underline{W}}\,\underline{x}\|_2^2 \leq (1+\varepsilon)n\big].$$

Now, letting $\underline{w_i}$ be the i'th row of $\underline{\underline{W}}$, then $(\underline{\underline{W}}\,\underline{x})_i = \underline{x}^T\underline{w_i}$. $\underline{w_i} \sim N(\underline{0}, \underline{\underline{I}})$, so $\underline{x}^T\underline{w_i} \sim N(0, \sum x_i^2) = N(0,1)$. Hence, $\|\underline{\underline{W}}\,\underline{x}\|_2^2 = \sum_i \underline{x}^T\underline{w_i}^2 \sim \chi^2_n$. Hence, use lemma to give final result.

Johnson-Lindenstrauss Lemma: Let Q be a finite set of vectors in $\mathbb{R}^d$. Let $\delta \in (0,1)$, and $n \in \mathbb{Z}^{>0}$ s.t.

$$\varepsilon = \sqrt{\frac{6\log(2|Q|/\delta)}{n}} \leq 3.$$

Then, with probability $1-\delta$ over a choice of random matrix $\underline{\underline{W}} \in \mathbb{R}^{n\times d}$ s.t. each element of $\underline{\underline{W}}$ is iid $N(0, \frac{1}{n})$, then

$$\sup_{\underline{x}\in Q}\left|\frac{\|\underline{\underline{W}}\,\underline{x}\|_2^2}{\|\underline{x}\|_2^2} - 1\right| < \varepsilon.$$

Proof: From lemma 2, for every $\varepsilon \in (0,3)$, using a union bound,

$$P\left[\sup_{\underline{x}\in Q}\left|\frac{\|\underline{\underline{W}}\,\underline{x}\|_2^2}{\|\underline{x}\|_2^2} - 1\right| > \varepsilon\right] \leq 2|Q|e^{-\varepsilon^2 n/6}.$$

Let $\delta = 2|Q|e^{-\varepsilon^2 n/6}$, then,

$$\varepsilon = \sqrt{\frac{6\log(2|Q|/\delta)}{n}}$$

Note: This works for any d, even infinite dimensional.

Takeaway: Can generate $\underline{\underline{W}}$ randomly, reduce data, and mostly preserve distances. But, not invertible.

For PCA, wish to generate $\underline{\underline{W}} \in \mathbb{R}^{d \times n}$, $\underline{\underline{U}} \in \mathbb{R}^{n \times d}$ s.t. $\underline{\underline{U}}\,\underline{\underline{W}}\,\underline{x} = \tilde{\underline{x}} \approx \underline{x}$. Wish to solve (across m data points)

$$① \quad \underset{\underline{\underline{W}} \in \mathbb{R}^{n \times d},\, \underline{\underline{U}} \in \mathbb{R}^{d \times n}}{\arg\min} \sum_{i=1}^{m} \| \underline{\underline{U}}\,\underline{\underline{W}}\,\underline{x_i} - \underline{x_i} \|_2^2 .$$

Lemma: Let $\underline{\underline{W}}, \underline{\underline{U}}$ be a solution to (1), then the columns of $\underline{\underline{U}}$ are orthonormal, i.e. $\underline{\underline{U}}^T \underline{\underline{U}} = \underline{\underline{I}}$, and $\underline{\underline{W}} = \underline{\underline{U}}^T$.

Proof: The mapping $\underline{x} \to \underline{\underline{U}}\,\underline{\underline{W}}\,\underline{x}$ has an n-dimensional linear subspace of $\mathbb{R}^d$ since $\underline{\underline{W}}\,\underline{x} \in \mathbb{R}^n$. Let $\underline{\underline{V}} \in \mathbb{R}^{d \times n}$ s.t. $\underline{\underline{V}}^T \underline{\underline{V}} = \underline{\underline{I}}$ and s.t. $\text{range}(\underline{\underline{V}}) = \text{range}(\underline{\underline{U}}\,\underline{\underline{W}})$, say through Gram-Schmidt. Then, $\forall\, \underline{x} \in \mathbb{R}^d$, $\underline{y} \in \mathbb{R}^n$, (any $\underline{z} \in \mathbb{R}^d$ can be represented like this).

$$\| \underline{x} - \underline{\underline{V}}\,\underline{y} \|_2^2 = \| \underline{x} \|_2^2 + \underline{y}^T \underline{\underline{V}}^T \underline{\underline{V}}\,\underline{y} - 2\underline{y}^T \underline{\underline{V}}^T \underline{x}$$

$$= \| \underline{x} \|_2^2 + \| \underline{y} \|_2^2 - 2\underline{y}^T (\underline{\underline{V}}^T \underline{x}).$$

To minimize the above,

$$0 = \nabla_y \| \underline{x} - \underline{\underline{V}}\,\underline{y} \|_2^2$$

$$= 0 + 2\underline{y} - 2\underline{\underline{V}}^T \underline{x}$$

$$\Rightarrow \underline{y}^* = \underline{\underline{V}}^T \underline{x}.$$

Therefore, for each $\underline{x}$,

$$\underset{\tilde{\underline{x}} \in \mathbb{R}}{\arg\min} \| \underline{x} - \tilde{\underline{x}} \|_2^2 = \underline{\underline{V}}\,\underline{y}^* = \underline{\underline{V}}\,\underline{\underline{V}}^T \underline{x}.$$

So, our new problem is

$$\underset{\underline{\underline{U}} \in \mathbb{R}^{d \times n},\, \underline{\underline{U}}^T \underline{\underline{U}} = \underline{\underline{I}}}{\arg\min} \sum_{i=1}^{m} \| \underline{x_i} - \underline{\underline{U}}\,\underline{\underline{U}}^T \underline{x_i} \|_2^2 .$$

Can rewrite

$$\|\underline{x_i} - \underline{\underline{U}}\,\underline{\underline{U}}^T \underline{x_i}\|_2^2 = \|\underline{x_i}\|_2^2 - 2\underline{x}^T \underline{\underline{U}}\,\underline{\underline{U}}^T \underline{x_i} + \underline{x_i}^T \underline{\underline{U}}\overbrace{\underline{\underline{U}}^T \underline{\underline{U}}}^{\underline{\underline{I}}}\underline{\underline{U}}^T \underline{x_i}$$

$$= \|\underline{x_i}\|_2^2 - \underline{x_i}^T \underline{\underline{U}}\,\underline{\underline{U}}^T \underline{x_i}.$$

So, equivalent to

$$\underset{\underline{\underline{U}} \in \mathbb{R}^{d \times n},\ \underline{\underline{U}}^T\underline{\underline{U}} = \underline{\underline{I}}}{\arg\max} \sum_{i=1}^{m} \underline{x_i}^T \underline{\underline{U}}\,\underline{\underline{U}}^T \underline{x_i}.$$

<u>10/26/23:</u>

We can then rewrite the above in terms of trace:

$$\sum_{i=1}^{m} \underline{x_i} \underline{\underline{U}}\,\underline{\underline{U}}^T \underline{x_i} = \sum_{i=1}^{m} \text{trace}(\underline{\underline{U}}^T \underline{x_i}\,\underline{x_i}^T \underline{\underline{U}}) = \text{trace}\left(\underline{\underline{U}}^T \sum_{i=1}^{m} \underline{x_i}\,\underline{x_i}^T \underline{\underline{U}}\right).$$

Let $\underline{\underline{A}} = \sum_{i=1}^{m} \underline{x_i}\,\underline{x_i}^T$. We can see that $\underline{\underline{A}}$ is symmetric, hence has real diagonalization $\underline{\underline{A}} = \underline{\underline{V}}\,\underline{\underline{D}}\,\underline{\underline{V}}^T$, and $\underline{\underline{A}} \succeq 0$. Now, can show that

$$\underset{\|\underline{u_1}\|_2 = 1}{\arg\max}\ \underline{u_1}^T \underline{\underline{A}}\,\underline{u_1} = \lambda_1(\underline{\underline{A}})$$

and so on for $\lambda_2, \dots,$ with $\underline{u_2} \perp \underline{u_1}$, $\underline{u_3} \perp \text{span}(\underline{u_1}, \underline{u_2})$, ... So, choose the i^{th} row of $\underline{\underline{U}}$ to be the i^{th} eigenvector of $\underline{\underline{A}}$. So, to satisfy the upper argmin (1), choose

$$\underline{\underline{U}} = \begin{bmatrix} | & | & & | \\ \underline{u_1} & \underline{u_2} & \cdots & \underline{u_n} \\ | & | & & | \end{bmatrix}, \quad \underline{\underline{W}} = \underline{\underline{U}}^T$$

for

$$\textcircled{1} \quad \underset{\underline{\underline{W}} \in \mathbb{R}^{n \times d},\ \underline{\underline{U}} \in \mathbb{R}^{d \times n}}{\arg\min} \sum_{i=1}^{m} \|\underline{\underline{U}}\,\underline{\underline{W}}\,\underline{x_i} - \underline{x_i}\|_2^2 .$$

And the objective value is the sum of the eigenvalues.

The complexity for the previous algorithm is $O(d^3)$ to compute the eigenvalues of $\underline{\underline{A}}$, plus $O(md^2)$ for constructing $\underline{\underline{A}}$.

$$\text{Naive PCA}: O(d^3) + O(md^2).$$

What if $d \gg m$?

Trick: Recall $\underline{A} = \underline{X}^T \underline{X} \in \mathbb{R}^{d \times d}$, let $\underline{B} = \underline{X}\,\underline{X}^T \in \mathbb{R}^{m \times m}$. Suppose $\underline{B}\,\underline{x} = \lambda \underline{x}$. Then $\underline{A}(\underline{X}^T \underline{x}) = \underline{X}^T \underline{B}\,\underline{x} = \lambda(\underline{X}^T \underline{x})$. Hence, $\underline{X}^T \underline{x}$ is an eigenvector of $\underline{A}$ with eigenvector λ.

So, can compute diagonalization of $\underline{B}$ to get those of $\underline{A}$

$$d \gg m \text{ trick PCA}: \; O(m^3) + O(md^2).$$

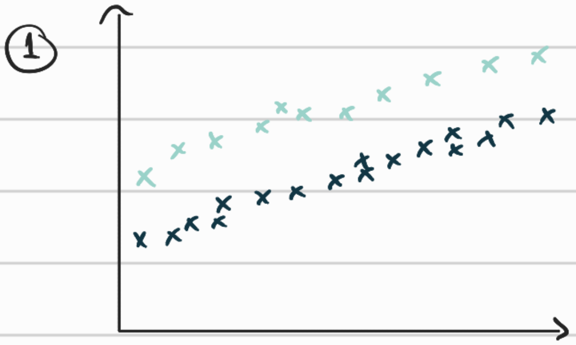
Compressed sensing is a dimensionality reduction technique which utilizes a prior assumption of sparsity.

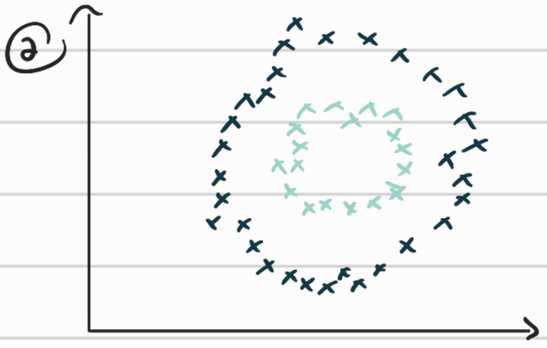
10/31/23:

Clustering is the grouping of a set of objects such that similar objects end up in the same group and dissimilar objects are separated into separate groups. Clustering is an unsupervised learning problem, no labels to predict.

So, we're given a set of elements, $\mathcal{X}$. Also given a function $d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}^+$ that is symmetric, satisfies $d(x,x) = 0 \;\; \forall \; x \in \mathcal{X}$, and often satisfies the triangle inequality (then a metric). Or, could be a similarity function $s : \mathcal{X} \times \mathcal{X} \to [0,1]$, symmetric, satisfies $s(x,x) = 1 \;\; \forall \; x \in \mathcal{X}$. Finally, it may take as input $k \in \mathbb{N}$, the number of required clusters. As output, it partitions $\mathcal{X}$ into disjoint subsets, $C_1, \ldots, C_k$, with $\bigcup_{i=1}^{k} C_i = \mathcal{X}$, $C_i \cap C_j = \emptyset$ if $i \neq j$. Or, can return a function $\underline{p}(x)$ where $p_i(x) = \mathbb{P}[x \in C_i]$.

Aside: Euclidean distance may not be a good metric:



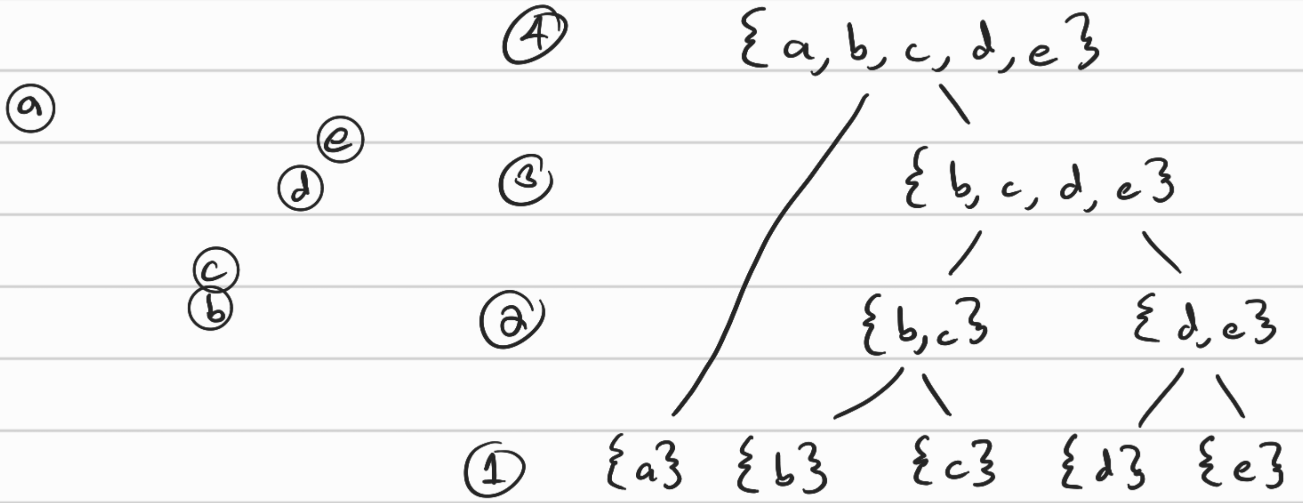


One alternative is a geodesic: The shortest path between two points on a surface.

Clustering requrres learning about the manifolds upon which the data live.

<u>Linkage-based clustering</u> works in a series of rounds. Begin with every point as a cluster, and then merge the most similar clusters. So the parameters are a similarity/distance function, and a stopping criteria.

Ex:



Types of distances

1) <u>Single-linkage clustering</u>: $D(A, B) := \min\{d(a,b) : a\in A, b\in B\}$
2) <u>Average-linkage clustering</u>: $D(A, B) := \frac{1}{|A||B|}\sum_{a\in A, b\in B} d(a,b)$
3) <u>Max-Linkage clustering</u>: $D(A, B) := \max\{d(a,b) : a\in A, b\in B\}$

Creates a decision tree of clustering

Types of stop criterion:

1) Number of clusters, k, stop when # of clusters $\leq k$.
2) Distance bound, $r\in\mathbb{R}^+$, stop when all distances $\geq r$. Can do a scaled distance bound, $r = \alpha \max\{d(x,y) : x, y\in\mathcal{X}\}$, $\alpha\in(0,1)$.

In <u>k-means</u>, we wish to pose clustering as an optimization problem. Suppose given set $\mathcal{X}$, metric d, and clusters $C = (C_1, \dots, C_k)$. Then, goal is to come up with an <u>objective function</u> G, and try to minimize $G((\mathcal{X}, d), C)$. In k-means, represent each C_i by its centroid (center of mass), μ_i. Suppose $\mathcal{X}$ is embedded in a larger space $\mathcal{X}'$. Then, define the centroid as

$$\mu_i(C_i) := \underset{\mu \in \mathcal{X}'}{\arg\min} \sum_{x \in C_i} d(x, \mu)^2 .$$

The <u>k-means objective</u> is

$$G_{k\text{-means}}((\mathcal{X}, d), C) = \min_{\mu_1, \dots, \mu_k \in \mathcal{X}'} \sum_{i=1}^{k} \sum_{x \in C_i} d(x, \mu_i)^2 .$$

The <u>k-medioids objective</u> requires $\mu_i \in \mathcal{X}$

$$G_{k\text{-medioids}}((\mathcal{X}, d), C) = \min_{\mu_1, \dots, \mu_k \in \mathcal{X}} \sum_{i=1}^{k} \sum_{x \in C_i} d(x, \mu_i)^2 .$$

The <u>k-median objective</u> uses ℓ^1 instead of ℓ^2 to be more robust to outliers:

$$G_{k\text{-median}}((\mathcal{X}, d), C) = \min_{\mu_1, \dots, \mu_k \in \mathcal{X}} \sum_{i=1}^{k} \sum_{x \in C_i} d(x, \mu_i) .$$

These are all examples of <u>center-based</u> objectives. Generally,

$$G_f((\mathcal{X}, d), C) = \min_{\mu_1, \dots, \mu_k \in \mathcal{X}'} \sum_{i=1}^{k} \sum_{x \in C_i} f(d(x, \mu_i))$$

where $\mathcal{X}'$ is either $\mathcal{X}$ or a superset of it.

k-means is popular, however, the minimization problem is NP hard, even hard to approximate. Typically an iterative algorithm is employed:

<u>k-means algorithm:</u>

```
1) Input: $\mathcal{X} \subseteq \mathbb{R}^n$, $k \in \mathbb{N}$, $\|\cdot\|$
2) Randomly choose initial centroids $\mu_1, \dots, \mu_k$
3) Until convergence criteria
    a) $\forall i$, set $C_i = \{ \underline{x} \in \mathcal{X} : i = \arg\min_j \|\underline{x} - \mu_j\| \}$
    b) $\forall i$, update $\mu_i = \frac{1}{|C_i|} \sum_{\underline{x} \in C_i} \underline{x}$.
```

<u>Lemma:</u> The k-means algorithm does not increase the k-means objective function.

<u>Proof:</u>

$$G(C^{(t)}) = \min_{\underline{\mu}_1, \dots, \underline{\mu}_k \in \mathbb{R}^n} \sum_{i=1}^{k} \sum_{\underline{x} \in C_i} \|\underline{x} - \underline{\mu}_i\|_2^2$$

$$(G \text{ is optimal}) \quad \leq \sum_{i=1}^{k} \sum_{\underline{x} \in C_i^{(t)}} \|\underline{x} - \underline{\mu}(C_i^{(t-1)})\|_2^2$$

$$(\text{choice of update}) \leq \sum_{i=1}^{k} \sum_{\underline{x} \in C_i^{(t-1)}} \| \underline{x} - \underline{M}(C_i^{(t-1)}) \|_2^2$$

$$= G(C^{(t-1)}).$$

<u>Remarks:</u> No guarantees on finding a global minima or # of iterations required. Also, all resulting clusters are convex, so will struggle with circle example (pic ②) above.

<u>Pagerank</u> is an algorithm that quantitively ranks the importance of each page on the web. It generates nonnegative scores for pages based on how many other pages link to it.

Suppose n pages, each indexed by an integer. We create a directed graph with an edge from page i to j if page i links to page j. Let x_k denote the importance of page k.

One option is to set $x_k = |\{x_i : x_i \to x_k \text{ in the graph}\}|$, called the number of <u>backlinks</u>. Instead, they weigh backlinks based on the popularity of the citing page, and, by how many (forward) links they have, n_k.

So require $X_k = \sum_{j \in L_k} \frac{x_j}{n_j}$. Then, we can express the problem as a matrix equation

$$\underline{A}\,\underline{x} = \underline{x} \quad \Rightarrow \quad \underline{x} \text{ an eigenvector of } \underline{\underline{A}} \text{ w/ eigenvalue } 1.$$

$A_{ij} = \frac{x_j}{n_j}$ if $x_j \to x_i$, otherwise 0. Note, sum of each column of $\underline{\underline{A}}$ is 1. We say $\underline{\underline{A}}$ is <u>column-stochastic</u> as each entry is nonnegative and the sum of the columns is 1.

<u>Theorem:</u> A column-stochastic matrix $\underline{\underline{A}}$ has an eigenvalue equal to 1, and 1 is its largest* eigenvalue. Then can use iterative procedure to find the eigenvector(s).

<u>11/2/23:</u>

<u>Proof:</u> Let $\underline{\underline{A}} \in \mathbb{R}^{n \times n}$ be column stochastic. Note that $\underline{\underline{A}}^T$ has the same eigenvalues (not eigenvectors necessarily) as $\underline{\underline{A}}$. Additionally, since cols of
(same characteristic polynomial)

$\underline{A}$ sum to 1, $\underline{A}^T \underline{1} = \underline{1} \Rightarrow 1$ an eigenvalue of $\underline{A}$ and $\underline{A}^T$. Additionally, Gershgorin's Theorem states that since $A_{ii} = 0$, $|\lambda_i(\underline{A})| \leq \sum_{j=1}^{n} |A_{ji}| = 1$ for each i.

One way to overcome the issues of k-means for nonconvex clusters, (pic (2)), is to transform the data into a graph first. We construct a weighted graph $G = (V, E, \underline{W})$ using a similarity kernel $K_\epsilon(u) = e^{-u^2/(2\epsilon)}$, by assigning each point a vertex and set the edge weights $w_{ij} = K_\epsilon(\|x_i - x_j\|)$. We then wish to cluster the graph. Note: w_{ij} small $\Rightarrow$ x_i, x_j far apart.

Given a weighted graph $G = (V, E, \underline{W})$, we wish to cluster it. First, we'll consider 2 clusters. Given a partition (S, S^c), define

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

Note, cut(S) large means points were dissimilar, so wish to find

$$\underset{S}{\arg\min}\ \text{cut}(S).$$

The degree matrix $\underline{D}$ is a diagonal matrix s.t. $D_{ii} := \deg(i) := \sum_{j=1}^{n} w_{ij}$.
The graph Laplacian of G is given by $\underline{L_G} := \underline{D} - \underline{W}$, or equivalently,
$\underline{L_G} = \sum_{i<j} w_{ij} (\underline{e_i} - \underline{e_j})(\underline{e_i} - \underline{e_j})^T$.

Now, given a partition $\underline{S}$, define $y_i \in \{\pm 1\}$ s.t. $y_i = 1$ if $i \in S$, $y_i = 1$ otherwise.
Then, can rewrite

$$\text{cut}(S) = \frac{1}{4} \sum_{i<j} w_{ij} (y_i - y_j)^2$$

with each term nonzero if $i \in S, j \in S^c$ or $i \in S^c, j \in S$. Next,

$$\begin{aligned}
\sum_{i<j} w_{ij} (y_i - y_j)^2 &= \sum_{i<j} w_{ij} \left[\underline{y}^T (\underline{e_i} - \underline{e_j})\right]^2 \\
&= \sum_{i<j} w_{ij} \left[\underline{y}^T (\underline{e_i} - \underline{e_j})\right]\left[(\underline{e_i} - \underline{e_j})^T \underline{y}\right] \\
&= \underline{y}^T \left[\sum_{i<j} w_{ij} (\underline{e_i} - \underline{e_j})(\underline{e_i} - \underline{e_j})^T\right] \underline{y}
\end{aligned}$$

$$= \underline{y}^T \underline{\underline{L}}_G \underline{y}.$$

So, we have that

$$\text{cut}(S) = \frac{1}{4} \underline{y}^T \underline{\underline{L}}_G \underline{y}, \quad \underline{y} \in \{\pm 1\}^n.$$

One issue with $\underset{S}{\arg\min}\ \text{cut}(S)$ is that $S = \emptyset$ results in a minimum of 0. One way to address this is to require $|S| = |S^c|$ if $|V|$ even. This can be written in above notation as $\underline{1}^T \underline{y} = \sum_{i=1}^{n} y_i = 0$. So, balanced cut problem is

$$\frac{1}{4} \underset{\underline{y} \in \{\pm 1\}^n,\ \underline{1}^T \underline{y} = 1}{\arg\min} \underline{y}^T \underline{\underline{L}}_G \underline{y}.$$

Hovever, this problem is discrete & combinatorily difficult. We can relax the requirements by not forcing $|S| = \frac{n}{2}$ (n even).

A <u>Cheeger cut</u> of S is given by

$$h(S) = \frac{\text{cut}(S)}{\min\{\text{vol}(S), \text{vol}(S^c)\}}, \quad \text{vol}(S) = \sum_{i \in S} \deg(i) = \sum_{i \in S, j \in V} w_{ij}.$$

Note that if S (or S^c) is $\emptyset$, then $h(S)$ is undefined as $\text{vol}(S)$ (or S^c) is 0. Then, the <u>Cheeger's constant</u> of the graph G is

$$h_G := \min_{S \subseteq V} h(S)$$

where V is the set of vertices. A similar object to the Cheeger cut is the <u>Normalized cut</u>, Ncut, defined as

$$\text{Ncut}(S) := \frac{\text{cut}(S)}{\text{vol}(S)} + \frac{\text{cut}(S^c)}{\text{vol}(S^c)}.$$

Can show the relationship that $h(S) \leq \text{Ncut}(S) \leq 2h(S)$, so they are closely related.

Now, we wish to relax $\underline{y} \in \{a, b\}^n$ instead of $\{\pm 1\}^n$. Then, instead of $\underline{1}^T \underline{y} = 0$, choose the less restrictive notion of balance where

$$\underline{1}^T \underline{D}\, \underline{y} = \sum_{i=1}^{n} 1 \cdot \deg(i) \cdot y_i = a \sum_{i \in S} \deg(i) + b \sum_{i \in S^c} \deg(i)$$

$$= a \cdot \text{vol}(S) + b \cdot \text{vol}(S^c)$$

And, since we added 2 degrees of freedom, impose

$$\underline{y}^T \underline{\underline{D}}\, \underline{y} = \sum_i y_i^2 \deg(i) = a^2 \cdot \mathrm{vol}(S) + b^2 \cdot \mathrm{vol}(S^c) = 1.$$

Below proposition says choosing $a = \sqrt{\frac{\mathrm{vol}(S^c)}{\mathrm{vol}(S)\mathrm{vol}(G)}}$, $b = -\sqrt{\frac{\mathrm{vol}(S)}{\mathrm{vol}(S^c)\mathrm{vol}(G)}}$ results exactly in Ncut.

Proposition:

$$\mathrm{Ncut}(S) = \underline{y}^T \underline{\underline{L}}_G\, \underline{y}$$

where

$$y_i = \begin{cases} \sqrt{\frac{\mathrm{vol}(S^c)}{\mathrm{vol}(S)\mathrm{vol}(G)}} & \text{if } i \in S \\ -\sqrt{\frac{\mathrm{vol}(S)}{\mathrm{vol}(S^c)\mathrm{vol}(G)}} & \text{if } i \in S^c \end{cases}$$

Proof:

$$\underline{y}^T \underline{\underline{L}}_G\, \underline{y} = \sum_{ij} w_{ij}(y_i - y_j)^2$$

$$= \sum_{ij} w_{ij}(y_i^2 + y_j^2 - 2y_i y_j)$$

$$= \sum_{i \in S, j \in S^c} \frac{w_{ij}}{\mathrm{vol}(G)} \left(\frac{\mathrm{vol}(S^c)}{\mathrm{vol}(S)} + \frac{\mathrm{vol}(S)}{\mathrm{vol}(S^c)} + 2 \right)$$

$$= \frac{\mathrm{cut}(S)}{\mathrm{vol}(S) + \mathrm{vol}(S^c)} \left(\frac{\mathrm{vol}(S^c) + \mathrm{vol}(S)}{\mathrm{vol}(S)} + \frac{\mathrm{vol}(S) + \mathrm{vol}(S^c)}{\mathrm{vol}(S^c)} \right)$$

$$= \mathrm{cut}(S)\left(\frac{1}{\mathrm{vol}(S)} + \frac{1}{\mathrm{vol}(S_c)} \right) = \mathrm{Ncut}(S)$$

Now, allow $\underline{y} \in \mathbb{R}^n$ so the problem is not combinatoric, get

$$\min_{\underline{y} \in \mathbb{R}^n,\, \underline{y}^T \underline{\underline{D}}\, \underline{y} = 1,\, \underline{y}^T \underline{\underline{D}}\, \underline{1} = 0} \underline{y}^T \underline{\underline{L}}_G\, \underline{y}$$

and then threshold $S = \{ i \in V : y_i \le \tau \}$.

We define the normalized Laplacian

$$\underline{\underline{\mathcal{L}}}_G := \underline{\underline{D}}^{-1/2} \underline{\underline{L}}_G \underline{\underline{D}}^{-1/2} = \underline{\underline{I}} - \underline{\underline{D}}^{-1/2} \underline{\underline{W}}\, \underline{\underline{D}}^{-1/2}, \quad \text{and} \quad \underline{z} := \underline{\underline{D}}^{1/2} \underline{y}.$$

Then, above problem equivalent to

$$\min_{\underline{z} \in \mathbb{R}^n,\, \|\underline{z}\|_2^2 = 1,\, (\underline{\underline{D}}^{1/2} \underline{1})^T \underline{z} = 0} \underline{z}^T \underline{\underline{\mathcal{L}}}_G\, \underline{z}.$$

Can show through the minmax theorem that since $(\underline{\underline{D}}^{1/2} \underline{1})^T \underline{z} = 0$

reduces the solution space by 1, the minimum is an eigenvalue problem with solution $\lambda_2(\mathcal{L}_G)$, and the minimizer, $\underline{v_2}$, is the second smallest eigenvector of $\mathcal{L}_G$. Finally, from $\underline{z} = \underline{\underline{D}}^{1/2}\underline{y}$, the optimal $\underline{y}$ from before is $\underline{\phi_2} = \underline{\underline{D}}^{-1/2}\underline{v_2}$.

Spectral Clustering Algorithm:

1) Generate $\underline{\underline{D}}$, $D_{ii} = \deg(i) = \sum_{j=1}^{n} w_{ij}$, norm. Laplacian $\underline{\underline{\mathcal{L}}}_G = \underline{\underline{I}} - \underline{\underline{D}}^{-1/2}\underline{\underline{W}}\,\underline{\underline{D}}^{-1/2}$
2) Find $\underline{v_2}$, 2^{nd} smallest eigenvector of $\underline{\underline{\mathcal{L}}}_G$.
3) Let $\underline{\phi_2} = \underline{\underline{D}}^{-1/2}\underline{v_2} \in \mathbb{R}^n$
4) Given τ, set $S = \{ i \in V : (\phi_2)_i \leq \tau \}$

Can prove that $\exists\, \tau$ s.t.

$$h(S) \leq 2\sqrt{h_G}$$

so this algorithm is suboptimal to only a square root factor.

Theorem: Cheeger's Inequality:

$$\frac{1}{2}\lambda_2(\underline{\underline{\mathcal{L}}}_G) \leq h_G \leq \sqrt{2\lambda_2(\underline{\underline{\mathcal{L}}}_G)}$$

Generalized Spectral Clustering Algorithm:

1) Calculate 2^{nd} thru $(k-1)$'st eigenvectors of $\underline{\underline{\mathcal{L}}}_G$, $\underline{v_2}, \dots, \underline{v_{k-1}}$
2) Let $\underline{\phi_m} = \underline{\underline{D}}^{-1/2}\underline{v_m} \in \mathbb{R}^n$
3) Embed each $i \in V$ into $\mathbb{R}^{k-1}$ by
$$i \longrightarrow \begin{bmatrix} (\phi_2)_i \\ \vdots \\ (\phi_{k-1})_i \end{bmatrix}$$
4) Use k-means on the embedded points.

11/7/23:

Consider a classification problem. Create a neural network with

k outputs, where k is the number of classes. Then, use softmax to turn floats into scalars:

$$y_j \longmapsto \frac{e^{y_j}}{\sum_{i=1}^{k} e^{y_i}}.$$

And, for a loss function, use a log liklihood

$$\mathcal{L}_i = \mathbb{P}\left[y_i \mid p_i = \frac{e^{y_i}}{\sum_{i=1}^{k} e^{y_i}} \right]$$

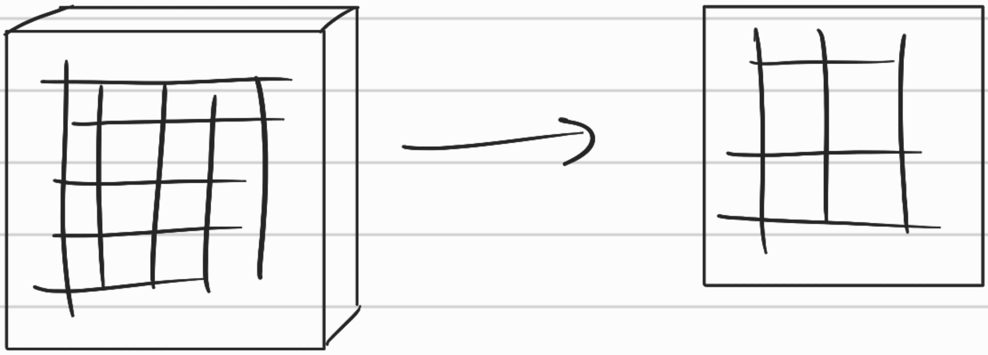
$$p_i^{y_i}(1-p_i)^{1-y_i}$$

$$\Rightarrow \ell_i = \log(\mathcal{L}_i) = y_i \log(p_i) + (1-y_i)\log(1-p_i)$$

$$\Rightarrow \ell = \frac{1}{N}\sum_{i=1}^{N} \ell_i = \frac{1}{N}\sum_{i=1}^{N} y_i \ln(p_i) + (1-y_i)\ln(1-p_i).$$

and we wish to minimize ℓ.

In <u>convolutional neural networks</u>, we apply filters to reduce dimensionality but preserve components of original data, typically an image.
<u>Ex:</u> Begin with a 32×32×3 image. Create a 5×5×3 filter, and "slide" it across the image. Resulting data is 27×27×1.



<u>Ex:</u>

$$\begin{bmatrix} 3 & 5 & 2 \\ 1 & 6 & 4 \\ 7 & 9 & 8 \end{bmatrix} * \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} = \begin{bmatrix} 40 & 43 \\ 70 & 73 \end{bmatrix}$$

Why? $3 \cdot 1 + 5 \cdot 2 + 1 \cdot 3 + 6 \cdot 4 = 40$, and so on

11/9/23:

A CNN is a sequence of convolution layers, interspersed with activation functions.

Ex:

$$32\times32\times3 \text{ data} \xrightarrow[\substack{6\ 5\times5\times3\\ \text{layers}}]{\text{CONV, RELU}} 28\times28\times6 \xrightarrow[\substack{10\ 5\times5\times6\\ \text{filters}}]{\text{CONV, RELU}} 24\times24\times10 \longrightarrow \cdots \longrightarrow \text{Linear classifier}$$

How about theoretical guarantees for neural networks?

The universal approximation theorem states that any continuous function $f: [0,1]^n \to [0,1]$ can be approximated arbitrarily well with at least one hidden layer.

A "visual proof" makes use that neurons can generate a step function, and, we can approximate any continuous function with several steps, a piecewise constant function.

11/14/23:

Recall the feed-forward neural network

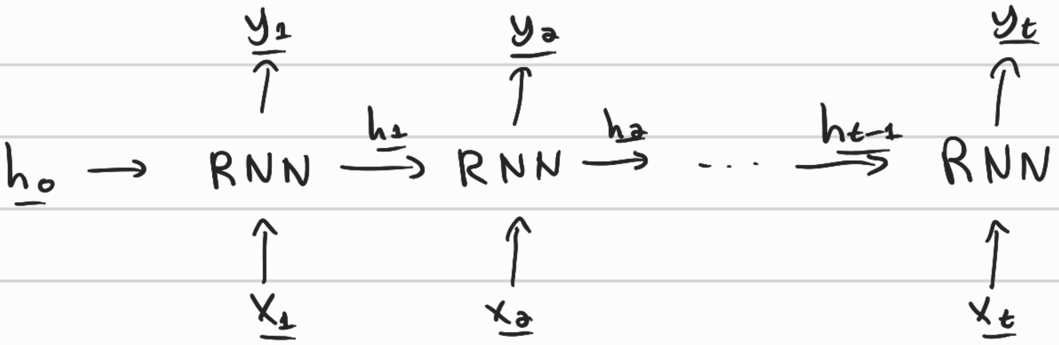
$$f(\underline{x}, W) = \underline{\underline{W}}_n \sigma(\underline{\underline{W}}_{n-1} \sigma(\cdots(\underline{\underline{W}}_1 \underline{x})))$$

A "frame" $\underline{x}$, with fixed length, is passed in.

What if we wish to pass in sentences with various lengths, or a video with many frames for classification?

First, think of a recurrent neural network, RNN as a black-box that has an internal state $\underline{h}$. So $\underline{x_1} \to \text{RNN} \to \underline{y_1}$ updates $\underline{h_0}$ to $\underline{h_1}$, then $\underline{x_2} \to \text{RNN} \to \underline{y_2}$ updates $\underline{h_1}$ to $\underline{h_2}$ and so on. So the state $\underline{h_t}$ is a function of the data $\underline{x_1}, \dots, \underline{x_t}$. Suppose we have the recurrence formula $\underline{h_t} = \underline{h}(\underline{h_{t-1}}, \underline{x_t} : W)$ where W contains parameters. Also, have output formula $\underline{y_t} = \underline{f}(\underline{h_t} ; W_o)$ where W_o contains parameters.

Pic:



A simple RNN lets

$$\underline{h_t} = \tanh\left(\underline{\underline{W_{hh}}}\,\underline{h_{t-1}} + \underline{\underline{W_{hx}}}\,\underline{x_t}\right) \; ; \; \underline{y_t} = \underline{\underline{W_{hy}}}\,\underline{h_t} \text{ or } \operatorname{softmax}\left(\underline{\underline{W_{hy}}}\,\underline{h_t}\right)$$

However, computing the loss function can be very tricky. An algorithm is called backward propagation through time, BPTT.
In practice, this is slow, and it is difficult to access information from many steps back. For large T, the gradients tend to vanish as they involve ~T products of gradients that are typically smaller than one in magnitude.

(Hochreiter & Schmidhuber)

Long-short Term Memory, LSTM, was a proposed fix to this in 1997, difficult to understand. It has no theoretical guarantees but does generally implement long-term dependencies well in practice.

11/16/23:

A neural ODE is of the form $\frac{d\underline{h}}{dt} = \underline{f}(\underline{h}(t), t, \Theta)$, where $\underline{f}$ is a neural network. Encodes time information where vanilla RNNs do not.
We introduce the loss function $L(\underline{h}(t))$ where $\underline{h}(t)$ depends on the NN.
Define the adjoint state, $\underline{a}(t) = dL/d\underline{h}$.

Theorem:

$$\frac{dL}{d\Theta} = \int_0^T \underline{a}(t)^T \frac{\partial \underline{f}(\underline{h}(t), t, \Theta)}{\partial \Theta}\, dt$$

and $\underline{a}(t)$ satisfies the adjoint ODE

$$\frac{d\underline{a}}{dt} = -\underline{a}(t)^T \frac{\partial \underline{f}(\underline{h}(t), t, \Theta)}{\partial \underline{h}}.$$

<u>Proof:</u> From chain rule,

$$\frac{dL}{d\theta} = \frac{dL}{d\underline{h}}^{T} \frac{d\underline{h}}{d\theta} = \underline{a}(t)^{T} \frac{d}{d\theta} \int_0^t \underline{f}(\underline{h}(s), s, \theta)\, ds$$

$$= \int_0^t \underline{a}(t)^{+} \frac{\partial \underline{f}(\underline{h}(s), s, \Theta)}{\partial \theta}\, ds.$$

And, $$\frac{d\underline{a}}{dt} = \frac{d}{dt}\frac{dL}{d\underline{h}} = \frac{d}{d\underline{h}}\left(\frac{dL}{d\underline{h}}^{+} \frac{\partial \underline{h}}{\partial t}\right) = \frac{d}{d\underline{h}}\left(\underline{a}(t)^{T} f(\underline{h}(t), t, \Theta)\right)$$

$$= \underline{a}(t)^{T} \frac{\partial \underline{f}(\underline{h}(t), t, \Theta)}{\partial \underline{h}}.$$

So, first we use an ODE solver to obtain $\underline{h}(t)$, $0 \leq t \leq T$, and get the loss $L(\underline{h}(t))$. Then, knowing the anatomy of the NN, we can obtain $\frac{\partial f}{\partial \underline{h}}$, and solve for $\underline{a}(t)$ using another ODE solver. Lastly, we can solve for $\frac{dL}{d\theta}$ by solving for $\frac{\partial f}{\partial \theta}$ and integrating. Then, can use descent method to update Θ and iterate.