Princeton Machine Learning Summer School

Clark Miyamoto (cm6627@nyu.edu)

August 12th - 21st

Abstract

Notes from the Princeton Theoretical Machine Learning Summer School

Part I
Jianfeng Lu: Neural Networks for PDEs

In these set of lectures, we'll go over how to design machine learning algorithms to solve problems arising from PDEs.

1 August 12th (1st session)

Consider a PDE defined on the domain $\Omega \subset \mathbb{R}^d$. For example let's consider the driven heat equation.

$$-\Delta u = f \qquad \text{Inside } \Omega \tag{1.1}$$

$$\partial_n u = 0 \qquad \text{On } \partial\Omega \tag{1.2}$$

where $\Delta = \frac{d}{2} = -\partial^2$ is the laplacian.

We can massage this into a minimization problem, where you attempt to minimize the residual. This is called the **strong form**

$$\inf_u \int_\Omega |\Delta u + f|^2 \tag{1.3}$$

This technique has origins in mathematics long ago, however today, we ansatz $u = u_\theta$ using a neural network, and modify θ s.t. you minimize that quantity. These are known as **PINNs (Physics Informed Neural Networks)**.

Another method is to use the **variational form**

$$\inf_u \frac{1}{2} \int (|\nabla u|^2 - uf)\, dx \tag{1.4}$$

You can convince yourself that this is correct by taking the variational derivative w.r.t. u, and see the minimizers is of the form of the PDE.

Another method is to use the **weak form**, where we expect this

$$\forall v \ \int v(-\Delta u \cdot f) = 0 \iff \int_\Omega \nabla v \cdot \nabla u - vf = 0 \tag{1.5}$$

Problem 1 *To demonstrate your understanding, show the formulation of the strong, variational, and weak form in linear algebra. I.e. you want to solve $Ax = b$, reformulate this into the various forms.*

When realizing these problem on a computers, u is a high-dimensional (or infinite dimensional) quantity, obviously this becomes a problem...

1.1 Finite Dimensional Approximation (In Function Space)

Consider decomposing u into a basis $\{\varphi_i\}_i$

$$u(x) = \sum_{i=1}^{N} c_i \varphi_i(x) \tag{1.6}$$

Now it is your job to find c_i's s.t. you minimize the variational form. So let's plug it in

$$\min_{\{c_i\}} \left[\frac{1}{2} \sum_{ij} \left(c_i c_j \int \nabla \varphi_i \cdot \nabla \varphi_j \right) - \sum_i \left(c_i \int f \varphi_i dx \right) \right] \tag{1.7}$$

This is now just a quadratic minimization problem. Nice.

However, this problem suffers from the curse of dimensionality. The number of coefficients grows exponential in the dimension. To see this, notice when you descrteize the domain Ω, the number of grid points grows as $(\#\text{ points}) = (L/\epsilon)^d$.

So solving PDEs in high-dimension (in domain space) is an interesting problems. And it seems that neural networks can defeat the curse of dimensionality, so perhaps we need to combine the two!

1.2 What can Neural Networks do for PDEs?

So now that we've motived that neural networks can help us in PDEs, let's go over a few applications

1. PINN: Using neural networks ansatz for solving PDEs. We've outlined this problem above.
2. Operator learning: Your use the neural networks to map a function to another function (thus it learns an operator). I.e. this can be a time-evolution kernel. The difficulty lies in encoding an infinite dimensional space into a finite dimensional space.
3. System identification:

1.3 Bounding of Error Due to Discretization

Let $\mathcal{X}$ be some space over functions. Your true function $u_* \in \mathcal{X}$. Now consider a finite-dimensional approximation of the function space $u_N \in \mathcal{X}_N$, where we assume $\mathcal{X}_N \subset \mathcal{X}$ (this may not hold depending on the boundary condition you set). Note that your true solution u_*, may not necessarily be in $\mathcal{X}_N$. This yields a trivial bound on the residual

$$||u_* - u_N||_x \geq \underbrace{\inf_{u \in \mathcal{X}_n} ||u_* - u||_x}_{\text{Approximation error}} \tag{1.8}$$

Going back to the forced heat equation in the variational form, we want to minimze

$$\Sigma(u) = \frac{1}{2}\int |\nabla u|^2 dx - \int uf dx \tag{1.9}$$

$$\inf_{u\in\mathcal{X}} \Sigma(u) = \Sigma(u_*) \tag{1.10}$$

From the bound, we know that $\Sigma(u_N) - \Sigma(u_*) \geq 0$. Plugging this in

$$\Sigma(u_N) - \Sigma(u_*) = \frac{1}{2}\int (|\nabla u_N|^2 - |\nabla u_*|^2) dx + \int (u_N - u_*)\Delta u_* dx \tag{1.11}$$

$$= \frac{1}{2}\int |\nabla u_N - \nabla u_*|^2 dx \tag{1.12}$$

If Ω is bounded, then

$$\underbrace{\int |u_N - u_*|^2 dx}_{||u_* - u_N||_x^2} < C \underbrace{\int |\nabla u_N - \nabla u_*|^2 dx}_{\sim(\Sigma(u_N)-\Sigma(u_*))} \tag{1.13}$$

And if you make an additional assumption that Ω is bounded AND convex, then you know that C doesn't depend on $\dim(\Omega)$. This gives us a pretty bound

$$\Sigma(u_N) - \Sigma(u_*) \geq C||u_* - u_N||_x^2 \tag{1.14}$$

You can find another bound

$$\Sigma(u_N) - \Sigma(u_*) = \inf_{u\in\mathcal{X}_N} \Sigma(u) - \Sigma(u_*) \tag{1.15}$$

$$= \inf_{u\in\mathcal{X}_n} \frac{1}{2}\int |\nabla u - \nabla u_*|^2 \tag{1.16}$$

$$\leq \inf_{u\in\mathcal{X}_N} \frac{1}{2}||u - u_*||_x^2 \tag{1.17}$$

Putting this together, you get a very nice principled bound on the variational formulation of the problem.

$$\boxed{C||u_* - u_N||_x^2 \leq \Sigma(u_N) - \Sigma(u_*) \leq \inf_{u\in\mathcal{X}_N} \frac{1}{2}||u_* - u||_x^2} \tag{1.18}$$

2 August 12th (2nd session)

Recall that given a PDE, we have some energy functional $\Sigma(u)$ which has a minimizer $u_* \in \mathcal{X}$. We then ask the question, given a discretization of the function space $\text{argmin}_{u\in\mathcal{X}_N}\Sigma(u) = u_N \in \mathcal{X}_N \subset \mathcal{X}$, how does your approximated u_N compare to the true answer u_*

$$\inf_{u\in\mathcal{X}_N} ||u - u_*||_X \tag{2.1}$$

Consider the set of functions $H^k(\Omega) = \{u : \int |\partial_{i_1}...\partial_{i_d} u|^2 dx < \infty, \sum_j i_j = i, i \leq k\}$

Theorem 1 (Bramble-Hilbert Lemma) *Let u be a function defined on the domain Ω, and $v \in P_k(\Omega)$ (k'th order polynomials defined on domain Ω). Under "mild" assumptions...*

$$\inf_{v\in P_k(\Omega)} \int_\Omega |u^{(i)} - v^{(j)}|^2 dx \leq C(\textit{diameter}(\Omega))^{m-k} \int_\Omega |u^{(m)}|^2 dx \tag{2.2}$$

where $u^{(i)}$ is the i'th order derivative, (m) is the highest finite order derivative.

As a result (which is not apparent to me a priori), the number of degrees of freedom scales $\sim h^{-d/s}$, where h is the mesh size, d is the dimension of the domain space Ω, and s are regularity conditions. So reading into it— higher dimensional PDEs end up having "simpler" curves.

Theorem 2 (Barron 1993) *Consider the Barron function space*

$$B(\mathbb{R}^d) = \left\{ u : \mathbb{R}^d \to \mathbb{R} \middle| \int_{\mathbb{R}^d} |\omega||\hat{u}(\omega)| d\omega < \infty \right\} \tag{2.3}$$

where $\hat{u}$ is the Fourier transform of u.

$$H^1(\mathbb{R}^d) = \{ u : \mathbb{R}^d \to \mathbb{R} | \int |u|^2 + |\nabla u|^2 dx < \infty \} \tag{2.4}$$

$$= ... \int |\hat{u}(\omega)|^2 + |\omega|^2 \hat{u}(\omega)|^2 < \infty \tag{2.5}$$

The 2nd equality holds by Parseval's theorem, which states the L_2 norm is invariant under FT Transform.

Then $\forall N$, there exists

$$u_N(x) = \sum_{i=1}^{N} a_i \sigma(b_i x + c_i) \tag{2.6}$$

s.t. the approximation error is bounded

$$\int |u - u_N|^2 \mu(dx) \leq C \frac{||u||^2_{B(\mathbb{R}^d)}}{N} \tag{2.7}$$

> *To me this is kinda surprising because N is the number of basis function which you use to approximate u. You'd imagine that as you increase the number of basis functions that such functions "over fit", and start to have high residuals... But I might be thinking of this wrong.*

Lemma 1 *Let $f \in$ convex hull of a set C in X (Hilber space), with $||g|| \leq b$. You find that $\forall g \in G$, then for any $n \geq 1$ every $c > b^2 - ||f||^2$ there exists lies in the convex hull of G, s.t.*

$$||f - f_n||^2 \leq \frac{c}{n} \tag{2.8}$$

Proof: Let $||f - f^*|| \leq \delta/n$ where f^* in the convex hull of G. Now do a basis decomposition

$$f^* = \sum_{i=1}^{M} \nu_i g^*_{il}, \qquad s.t.\ \gamma_i \in [0,1], \sum_i \gamma_i = 1 \tag{2.9}$$

Randomly sample $g_1, ..., g_n \sim \{g_j\} j$ with probability. This means

$$f_n = \frac{1}{n} \sum_i g_i \tag{2.10}$$

Ok let's consider a screened poisson equation

$$(-\Delta + \gamma^2) u = f \qquad \text{Defined on } \mathbb{R}^d \tag{2.11}$$

Just by inspection, if f is defined to the k derivative, then we'd imagine that u should be well defined for the $k+2$ derivatives. The LHS has two more derivatives. This implies $||u||_{H^{k+2}(\mathbb{R}^d)} \leq ||f||_{H^k(\mathbb{R}^d)}$.

2.1 Cousins of Barron Spaces

2.1.1 Compact Spaces

Consider a domain $\Omega = [0,1]^d$, you can define a barron space

$$B^k([0,1]^d) = \{u = \sum_{\omega} \cos(2\pi\omega x)\hat{u}_\omega, \sum_k (1+|\omega|^k)|\hat{u}_k| < \infty\} \tag{2.12}$$

2.1.2 Large N limit

Consider a two layer neural network

$$u_N = \frac{1}{N}\sum_{i=1}^{N} a_i \sigma(b_i x + c_i) \tag{2.13}$$

In the $N \gg 1$ limit, you can think of $\frac{1}{N}\sum_{i=1}^{N} \approx \int$, so this becomes

$$\int a\sigma(bx+c)\mu(da,db,dc) \tag{2.14}$$

3 August 13th

Recall, we were solving the variational problem, which involves minimizing the energy functional. For example, the forced-heat equation $-\Delta u = f$ has the variational form

$$\begin{aligned}
\Sigma(u) &= \int_\Omega (\frac{1}{2}|\nabla u|^2 - uf)dx && (3.1)\\
&= |\Omega|\ \mathbb{E}_{x\sim \text{Unif}(\Omega)}[|\nabla u(x)|^2 - uf|] && (3.2)\\
&\approx \frac{|\Omega|}{m}\sum_{i=1}^{m} |\nabla u(x_i)|^2 - u(x_i)f(x_i) \equiv \Sigma_m(u) \qquad \text{Monte Carlo sampling} && (3.3)
\end{aligned}$$

Then we get $u_{N,m} = \inf_{u\in\mathcal{X}_N} \Sigma_m(u)$. We now ask how far this monte carlo finite dimensional approximation is from the true answer $u_* \in \mathcal{X}$

$$\begin{aligned}
\mathbb{E}||u_{N,m} - u_*||_X^2 &\le \mathbb{E}\Sigma(u_{N,m}) - \Sigma(u_*) && (3.4)\\
&= \mathbb{E}\Big[\Sigma(u_{N,m} - \Sigma_m(u_{N,m}) + \Sigma_m(u_{N,m}) - \Sigma_m(u_N) \\
&\quad + \Sigma_m(u_N) - \Sigma_m(u_N) + \Sigma(u_N) - \Sigma(u_*)\Big] \qquad (\text{Insert } \pm 0) && (3.5)\\
&\le \text{Approximation Error} && (3.6)
\end{aligned}$$

Lu then spends the rest of the lecture reformulating various problems into this variational form problem. These discussions were non-informative, so I didn't take detailed notes

1. Green's Functions:
2. Eigenvalue problem: He basically just shows you can find the wave function in quantum mechanics using a monte-carlo scheme, see any numerical physics textbook on this.
3. Iterative schemes:
4. Non-linear PDEs (i.e. Optimal Control / Hamilton-Jacobi Bellman equation): He discusses

4 August 14th

In the spirit of Jane Street being a sponsor of the event, Lu will go over Hamilton Jacobi Bellman equations! Recall the HJB equation

$$-\gamma V(x) + \min_{u \in \mathbb{R}^m} \{\mathcal{L}_u + \ell(x,u)\} = 0 \tag{4.1}$$

$$\text{s.t. } dX_t = b(X_t)dt + \sqrt{2}\sigma(X_t)dW_t \tag{4.2}$$

where $\mathcal{L}_u$ is the infinitesimal generator of the stochastic process, $X_t \in \mathbb{R}^d, b(X_t) \in \mathbb{R}^d$, and $\sigma(X_t) \in \mathbb{R}^{d\times d}$.

> Lu goes over a cute review of stochastic differential equations.
>
> Consider an ODE $dX_t = b(X_t)dt$. Say, you want to ask what is the gradient of $Y_t = V(X_t)$, obviously it's $\nabla_x V \cdot b(X_t)$ (via chain rule). But you can also derive this by asking what's the total derviative of the quantity and expanding, but we assume the usual calculus thing $dt \to 0$, so higher order terms $\mathcal{O}(dt^2)$ are negligible.
>
> If you do this for a SDE, all you need to know is $\mathcal{O}(dW_t^2) = \mathcal{O}(dt)$, then take the total derviative again, and kill terms $\mathcal{O}(dt^{3/2})$ or higher, you'll recover Ito's formula
>
> $$dV(X_t) = \nabla_x V(X_t)(b(X_t)dt + \sqrt{2}\sigma(X_t)dW_t) + \text{Tr}(\nabla_x^2 V \sigma(X_t)\sigma(X_t)^T)dt \tag{4.3}$$
>
> Very cute.

So let's see what happens when we compute $dV(X_t)$

$$dV(X_t) = (\Delta + b\cdot\nabla)V(X_t)dt + \sqrt{2}\nabla V(X_t)dW_t \tag{4.4}$$

Now ask, what is $\mathbb{E}[dV(X_t)] = d\mathbb{E}[V(X_t)]$

$$d\mathbb{E}[V(X_t)] = (\Delta + b\cdot\nabla)\mathbb{E}[V(X_t)]dt \tag{4.5}$$

Let's define $\mathbb{E}[V(X_t)] \equiv u(x,t)$, then this starts to look like a heat equation!

$$\partial_t u = (\Delta + b\cdot\nabla)u \tag{4.6}$$

Now let's just assert a boundary conditoin $u(t=T,x) = h(x)$, this translates to $u(0,x) = \mathbb{E}h(X_T)$.

Ok now let's consider a situation where the state equation has access to a control line

$$dX_t = b(X_t,u_t)dt + \sqrt{2}\sigma(X_t,u_t)dW_t \tag{4.7}$$

And we want to minimize a cost J, that is we want to find the best control line u

$$J(x,u) = \mathbb{E}\left[\int_0^\infty e^{-\gamma t}\ell(x_t,u_t)dt | X_0 = x\right] \tag{4.8}$$

$$V(x) = \inf_u J(x,u) \tag{4.9}$$

For simplicity, let's assume u must be Markovian, that is $u_t = u(t,X_t)$. By

Part II
Yury Polyanskiy: Quantization

5 Motivation

Information theory was motivated by attempting to study how much "information" is lost when we move from the continuous time signals to bits. In LLM, you end up just computing matrix multiplication. However keeping the entries of the matrix at full precision (i.e. float32), this is expensive (both in storage, and in transfer time between GPU and CPU), so what happens when we move from float32 $\to$ float4? This is **quantization**.

In a modern context, you have a transformer which converts text to tokens (integers), pictures to soft-tokens (Polyanskiy says this is a vector in $\mathbb{R}^{d_{model}}$).

At the end of these lectures we'll prove a theorem on the optimal compression rate for quantizing the multiplication of two Gaussian iid matrices.

$$\mathbb{E}||\hat{Y} - Y||_F^2$$

where $Y = AB$, and $\hat{Y}$ is the same product where A, B have been quantized.

6 Scalar Quantization (Information Theory)

6.1 Uniform Quantization

Consider a quantization $x \in \mathbb{R}$, and I want to construct a map $x \mapsto \epsilon \in [x/\epsilon]$, where $[n] = [1, ..., n]$. This is called **uniform quantization (UQ)** (in machine learning this is called **round to nearest**). We can then ask, what is the error of UQ?

Consider $x \sim p_x$ (which is $U[-1/2, 1/2]$). This means as you take $\epsilon \to 0$ you find

$$x|[x/\epsilon] = m] \approx \text{Uniform}[-\epsilon/2, \epsilon/2] \tag{6.1}$$

Then you can compute

$$\mathbb{E}[(x - q(x))^2] = \frac{\epsilon^2}{12}(1 + \mathcal{O}(1)) = \frac{2^{-2R}}{12} \tag{6.2}$$

6.2 Dithering

Is there a rigorous way to write $\mathbb{E}[(x - q_\epsilon(x))^2] = \epsilon^2/12$?

To do this, we introduce this cool math trick. Let $U \sim \text{Uniform}[-\epsilon/2, \epsilon/2] \perp X$. Now consider $Y = X + U$. Now quantize Y, which is $q(Y) = q(X + U) = \epsilon[(x + u)/2]$. Now you subtract off the U, so your quantized estimate of X is

$$\hat{X} = q(X + U) - U \tag{6.3}$$

Proposition 1 (Crypto Lemma) $(\hat{X}, X) =^{(d)} (X + \tilde{U}, X)$, *where* $X \perp \tilde{U} \sim$ *Uniform*$[-\epsilon/2, \epsilon, 2]$

Proof: Notice that

$$\hat{X} - X = q(X + U) - (X + U) = \textit{Uniform}[-\epsilon/2, \epsilon/2] \perp X \tag{6.4}$$

Note that this holds for any discretization / lattice scheme.

What's really nice is that this makes any resulting calculation of $f(X)$ very simple. I.e. consider you're wanting to see $Y = X^2$

$$\mathbb{E}[(Y - \hat{Y})^2] = \mathbb{E}[(X^2 - \hat{X}^2)^2] = \mathbb{E}[(X^2 - (X + U)^2)^2] = \frac{\epsilon^2}{3}\mathbb{E}[X^2] + \text{constant} \tag{6.5}$$

6.3 What about non-uniform q?

OpenAI's newest's open source model GPT-OSS uses MX-FP4 quantization. Your quantization domain becomes $\{0, \pm 1/2, \pm 1, \pm 3/2, \pm 2, \pm 4, \pm 6\}$. They got these numbers because it's useful for hardware.

Definition 1 *For a given p_x. The best quantization D_{scalar} is defined as*

$$D_{scalar}(R, P_x) = \inf_{|image\ q| \leq 2^R} \mathbb{E}[|X - q(X)|^2] \tag{6.6}$$

$$= \inf_{p_{x,y}} \{\mathbb{E}[|X - Y|^2] : |\sup p_Y| \leq 2^R\} \tag{6.7}$$

So this is secretly the Wasserstein-2 projection. And it's very non-convex :(

Fix $\text{im}(q) = \{c_1 < ... < c_N\} = G$. The optimal quantization of q is the nearest n-bin $C = \{B_j\}_j$.

Now that you're given the bins $\{B_j\}_{j=1}^N$, the optimal quantization $q(x_= \mathbb{E}[X|X \in B_j\}$.

6.4 What is the best non-uniform quantization?

Theorem 3 (Panter-Diter '51) *Under mild assumptions*

$$D_{scalar}(R; p_x) = \frac{2^{-2R}}{12}\left(\int p_x(x)^{1/3} dx\right)^3 \tag{6.8}$$

Proof: Let $R \to \infty$ and $N \to \infty$. So this means the number of points in interval $I := N \int_I \lambda(x)dx$ So now you need to ask

$$\mathbb{E}[(x - q(x))^2] = \sum \mathbb{E}[(x - c_j)^2]\mathbb{1}_{x \in I_j} \tag{6.9}$$

$$\approx \sum \frac{\epsilon_j^2}{12}\mathbb{P}[x \in I_j] \tag{6.10}$$

$$\approx \frac{1}{12N^2}\int \lambda^{-2}(x) f(x) dx \tag{6.11}$$

$$\geq C\left(\int f^{-1/3}\right)^3 \tag{6.12}$$

QED

There is some heuristic for doing this on a gaussian... But I missed it lol.

7 Vector Quantization

From last time, we did $X \mapsto \hat{X} \in \{\text{Discrete subset of } \mathbb{Z}^R\}$, and we found that $\lambda^* \sim \text{pdf}^{1/3}$.

Now let's do vector quantization. Let $X = [x_1, ..., x_n]$, and perform the quantization map $\hat{X} = [\lfloor x_1 \rfloor, ..., \lfloor x_n \rfloor]$. So obviously, we're interested in determining the optimal quantization, but it is always optimal w.r.t. a particular operation. In this case, Polyanskiy says you're interested in dot products

$$\min_{\hat{X}} \mathbb{E}[(A^T X - A^T \hat{X})^2] = \min_{\hat{X}} \mathbb{E}[(A^T (X - \hat{X}))^2] = (X - \hat{X}) H (X - \hat{X}) \tag{7.1}$$

where $H = \mathbb{E}[AA^T]$.

An observation:

It is impossible to solve this (post-quantum) [He doesn't know what post quantum means, but heuristically this problems depends on nearest-neighbor lattice search, and quantum computers have a hard time with sparse graphs]. Here's a proof that this is a nearest-neighbor lattice search.

Notice that H is a linear transformation, so by 3Blue1Brown, you're skewing the coordinate space that the vectors $X, \hat{X}$ live on. If you quantize the coordinate space, you'll notice you're on a lattice, and figuring out where to optimally place $\hat{X}$ on said lattice it's a lattice search problem. soft-QED.

Boris asks a question: why not just optimize on $\min_{\hat{X} \in Q\mathbb{Z}^R}$ (that is instead of quantizing on the skewed lattice, you quantize on the square lattice, and forget about Q)? This doesn't work because you eventually will need to communicate Q, which costs bits, hence you're work will be lost forever :(

Goal: You have $X \in \mathbb{R}^n$, and we want a map $X \mapsto \hat{X} = q(x)$ where $|\text{image}(q)| = 2^{nR}$.

$$q = \sum_j c_j \mathbb{1}_{\mathcal{D}_j} \tag{7.2}$$

A particular solution is the **Lloyd Max algorithm**

1. For fixed $\mathcal{C} = \{c_1, ..., c_N\}$, the optimal q is

$$q(x) = \text{argmin}_{c \in \mathcal{C}} ||x - c|| \tag{7.3}$$

2. For fixed $\mathcal{D}_1, ..., \mathcal{D}_n$, the optimal

$$c_j = \mathbb{E}[X | X \in \mathcal{D}_j] \tag{7.4}$$

7.1 VQ-VAE

An application of vector quantization in machine learning is VQ-VAE (Vector quantization - variational auto encoder). You have the following autoencoder structure

$$f_\theta(X) = Z = (z_1, ..., z_m) \tag{7.5}$$

$$q_c(Z) = \hat{Z} = (\hat{z}_1, ..., \hat{z}_m) \tag{7.6}$$

$$g_\varphi(\hat{Z}) = \hat{X} \tag{7.7}$$

so your end to end network is $\hat{X} = g_\varphi(q_c(f_\theta(X)))$. Your objective is reconstruction, so your loss function is

$$\mathbb{E}_x||X - \hat{X}||_2^2 \tag{7.8}$$

however the problem is if you differentiate this loss, the gradients won't flow to θ, as q_c is a step function, so certain points won't generate any gradient. The trick to get around this is to turn-off the q_c block on backprop.

$$||X - g_\varphi(\hat{Z})||_2^2 + \lambda \sum_i ||z_i - \hat{z}_i||_2^2 \tag{7.9}$$

People also use this trick in mixture of experts models to allow gradients to flow through.

Polyanskiy also has an aside on analytically computing the gradients on the first loss, and proposes an alterative solution (instead of using the 2nd loss).

7.2 Information Theory

Let $\underline{x} = (x_1, ..., x_n) \sim p_x$. Find the quantizer $q(\underline{x})$ with $|\text{image}(q)| = 2^{nR}$ s.t.

$$\min \mathbb{E}[||\underline{x} - q(\underline{x})||^2] \tag{7.10}$$

Property: For all q, there exists a coupling $p_{x,y} \in \mathcal{P}(\mathbb{R} \times \mathbb{R})$ s.t.

1. $I(x; y) \leq R$

   Proof:

   $$R \geq \frac{1}{n} I(\underline{x}; \underline{\hat{x}}) \tag{7.11}$$
   $$= \frac{1}{n}(S(x_1^n) - S(x_1^n|\hat{x}_1^n) \tag{7.12}$$
   $$= \sum_i S(x_i) - S(x_i|\hat{x}_1^n, x_1^{i-1}) \leq ... - S(x_i|\hat{x}_i^n) \tag{7.13}$$
   $$\geq \frac{1}{n} \sum_i S(x_i) - S(x_i, \hat{x}_i) = \frac{1}{n} \sum_i I(x_i; \hat{x}_1) \tag{7.14}$$

   where $S(\cdot)$ is entropy, and $S(\cdot|*)$ is the conditional entropy. If $x_1^n, \hat{x}_1^n \perp \tau \sim \text{Uniform}([n])$, then we notice that

   $$= I(x; y|\tau) \tag{7.15}$$

   and now $\tau \perp x$, you find that $I(x; y|\tau) = I(x; y, \tau) \geq I(x; y)$. *QED.*

2. $\mathbb{E}[(x-y)^2] \leq \frac{1}{n}\mathbb{E}_x[||x - q(x)||^2]$

   Proof: $\mathbb{E}[(x-y)^2] = \frac{1}{n}\sum_i \mathbb{E}[(x_i - \hat{x}_i)^2] = \frac{1}{n}||x - q(x)||^2$ *QED.*

Corollary 1 *$\forall q$ we must have $\mathbb{E}[||\underline{x} - q(\underline{x})||^2] \geq nD_{vec}(R)$ where*

$$D_{vec}(R) = \inf_{p_{y|x}} \{\mathbb{E}[(x-y)^2] : I(x; y) \leq R\} \tag{7.16}$$

If you're an optimal person (I am not), you'll notice that this is an EOT projection, hence it's a convex problem.

Proof: *Take the lagrange dual of formulation of $D_{vec}(R)$. This yields*

$$D_{vec}(R) = \min \mathbb{E}(x-y)^2 + \lambda I(x; y) \tag{7.17}$$

Example:

Consider $p_x = \mathcal{N}(0,1)$. If $x \sim \mathcal{N}$, then $\forall Y$ (which is a random variable) if we make Y_G be the Gaussian version of Y (that $Y_G \sim \mathcal{N}(\mu_y, \sigma_y^2)$, is set s.t. $\mu_y = \mathbb{E}[Y], \sigma_y^2 = \text{Var}(Y)$). One can show

$$I(\underline{X};\underline{Y}) \geq I(\underline{X};\underline{Y}_G) \tag{7.18}$$

But isn't this trivial? This is a non-linear transformation of the distribution, hence it's strictly decreasing...

Anyways... Let's consider $X = \alpha Y + \beta Z$, where $X, Y \sim \mathcal{N}(0, \Sigma)$ and $Z \sim \mathcal{N}(0,1)$. This implies

$$I(X;Y) = \frac{1}{2}\log(1 + \mathbb{E}(\alpha Y)^2/\mathbb{E}(\beta Z)^2) = \frac{1}{2}\log(1 + \alpha^2\sigma^2/\beta^2 \tag{7.19}$$

$$\mathbb{E}(X-Y)^2 = (1-\alpha)^2\sigma^2 + \beta^2 \tag{7.20}$$

These implies $\alpha = 1$.

Now you redo it

$$D = \beta^2 \tag{7.21}$$

$$I = \frac{1}{2}\log(1/\beta^2) \tag{7.22}$$

This implies $D_vec = 2^{-2R}$

Theorem 4 *For all $p_{y|x}$ with $I(X;Y) \leq R$, there exists q such that $|q(x)| \leq 2^{nR}$. That is $\mathbb{E}||x - q(x)||^2 \leq n\mathbb{E}(x-y)^2 + \mathcal{O}(n)$*

Theorem 5 (Generalization of last theorem) *Consider $(\underline{x}, \underline{y})$ on $\mathbb{R}^n \times \mathbb{R}^n$. For all M, γ, there exists $q : \underline{x} \to \{c_1, ..., c_{n+1}\}$ such that*

$$\mathbb{E}[||\underline{x} - q(\underline{x})||^2] \leq \mathbb{E}[||\underline{x} - \underline{y}||^2] + e^{-M/\gamma}\mathbb{E}[||\underline{x}||^2] + \mathbb{E}||\underline{x}||^2 \mathbb{1}\{i(\underline{x};\underline{y}) > \log\gamma\} \tag{7.23}$$

where $i(\underline{a};\underline{b}) = \log p_{x,y}(a,b)/p_x(a)p_y(b)$ is the information density.

To see what the RHS of this bound means, let's see an application.

Let $x = (x_1, ..., x_n) \sim p_x$ and $y := (y_1, ..., y_n) \sim p_{y|x}$. Where the coordinates are independent of each other.

$$i(a^n; b^n) \equiv \log\frac{p_{x,y}}{p_x p_y} = \sum_{t=1}^{n} \log\frac{p_{x,y}}{p_x p_y}\Big|_{a_t, b_t} \tag{7.24}$$

This means in the limit as $n \gg 1$

$$\frac{1}{n} i(x;y) \to \mathbb{E}[i] = I(x;Y) \tag{7.25}$$

7.3 Dot Products

Ok let's go back to dot products. Our objective is to

$$\min \mathbb{E}[(x^T A - \hat{x}^T A)^2] \iff \min (x - \hat{x})^T H (x - \hat{x}) \tag{7.26}$$

Let's assume $\underline{x} \sim \mathcal{N}(0, \mathbb{I})$, so your minimization becomes

$$\min\{I(\underline{x}; \underline{y}) : \mathbb{E}(\underline{x} - \underline{y})^T H(\underline{x} - \underline{y}) \leq D\} \tag{7.27}$$

Now we diagonalize our $H = U^T \Lambda U$, and this just is rotation your Gaussian's covariance matrix. From the previous example, we learned that you can just do $\underline{y} \sim \mathcal{N}$,

$$\min\{\sum_i I(x_i, y_i) : \sum (x_i - y_i)^2 \beta_i^2 \leq D\} \tag{7.28}$$

$$\min_{\beta_i^2 \in [0,1]} \{\frac{1}{2} \sum_i \log \frac{1}{\beta_i^2} : \sum_i \beta_i^2 c_i^2 \leq D\} \tag{7.29}$$

You can now solve this using Lagrange multipliers. Now you find that

$$\beta_i^2 \sigma_i^2 = \begin{cases} T, & \sigma_i^2 > T \\ \sigma_i^2, & \sigma_i^2 < T \end{cases} \tag{7.30}$$

This implies

$$R_i = \begin{cases} \frac{1}{2} \log \frac{T}{\sigma_i^2}, & \sigma_i^2 > T \\ 0, & \sigma_i^2 < T \end{cases} \tag{7.31}$$

7.4 FSQ

7.5 Information Theoretic Quantization

8 VQI (Vector Quantization Information)

9 Quantization for LLMs

10 Open Problems

Part III
Yue M. Lu: How to Work with Non-Linear Random Matrices in ML

Consider a data matrices

$$X = \left(\vec{x}_1, ..., \vec{x}_n\right) \in \mathbb{R}^{d\times n} \tag{10.1}$$

where $\mathbb{R}^d \ni \vec{x}_1, ..., \vec{x}_n \sim p(x)$ (iid). Then you can ask what's the sample covariance matrix $H := XX^T$ (or the Gram matrix $E = X^TX$).

Obviously, if we assume this is true, RMT can provide some interesting remarks. However in Machine Learning, such matricies get put through non-linear functions (entry-wise), so we should study this regime.

A first example (Taylor Expansions): Consider the matrix

$$H = (h_{ij})_{i,j\leq n} = X^TX \tag{10.2}$$

where $h_{ij} \sim \mathcal{O}_p(1/\sqrt{n})$ (for $i \neq j$). Let $A = \sigma(H)$ be applied entrywise, now Taylor expand per entry

$$A_{ij} = \sigma(h_{ij}) + \sigma'(0)h_{ij} + \frac{\sigma''(0)}{2}h_{ij}^2 + \mathcal{O}(h_{ij}^3) \tag{10.3}$$

The last order terms are negligible in the $n \to \infty$ limit. However this definition relies on σ to be differentiable, this is not always true (i.e. $\sigma = \text{ReLU}$)...

Beyond Taylor Expansions: Lu poses that we assume non-linear matrices look like

$$\text{nonlinear model} = \text{linear model} + \text{noise} \tag{10.4}$$

asymptotically. In particular,

$$A = \sigma(XX^T) \tag{10.5}$$
$$B = \mu_0\mathbb{I} + \mu_1 XX^t + \mu_2^* Z \tag{10.6}$$

and it's believed $A \simeq B$ (when something goes to infinity). We will develop the machinery to set μ_0, μ_1, μ_2. This is called the **(gaussian) equivalence principle**.

10.0.1 Overview

1. Random matrices. Big takeaway: approximate rotational invariance of multilinear chaos.

2. Beyond spectral equivalence: empirical risk minimization. Central limit theorems for wiener chaos.

11 Random Matricies

Suppose you want to learn a function $f(x)$ with domain on S^{d-1} (hypersphere). You construct a training dataset $\mathcal{D} = \{(x_i, y_i = f(x_i)\}_{i=1}^n$.

Consider a matrix

$$A_{ij} = \begin{cases} \sigma(\vec{x}_i^T \vec{x}_j), & i \neq j \\ 0, & i = j \end{cases} \tag{11.1}$$

We can decompose $\sigma(x) = \sum_i \mu_i h_i(x)$ where h_i are the Hermite polynomials. Thus your matrix has an expansion

$$A = \sum_i \mu_i H_i \tag{11.2}$$

where

$$H_{ij} \Big\{ ... \tag{11.3}$$

If you let $n = \alpha d^\ell$ for some $\alpha > 0$ and $\ell \in \mathbb{N}$, you can show the decomposition has a very interesting eigenvalue spectrum.

$$A = \underbrace{\mu_0 H_0 + ... + \mu_{\ell-1} H_{\ell-1}}_{\text{lower-rank components}} + \underbrace{\mu_\ell H_\ell}_{\text{Marchenko-Pastur law}} + \underbrace{\mu_{\ell+1} H_{\ell+1} + \mu_{\ell+2} H_{\ell+2} + ...}_{\text{independent GOE matrices}} \tag{11.4}$$

interestingly, H_ℓ has a Marchenko-Pastur law in the eigenvalue distribution. (It reminded me of Florentin's paper, but that was a Gumbel distribution).

12 Equivalanece for Non-linear matricies

Let you data vectors be sampled $\mathbb{R}^d \ni x_i \sim p(x)$ for $i \in [n]$, where $\mathbb{E}[x_i] = 0$ and $\mathbb{E}[x_i x_i^T] = \mathbb{I}$. Now let's construct a kernel matrix

$$A_{ij} = \begin{cases} \frac{1}{\sqrt{n}} \sigma(x_i^T x_j / \sqrt{d}), & i \neq j \\ 0, & i = j \end{cases} \tag{12.1}$$

where $\sigma : \mathbb{R} \to \mathbb{R}$. Our goal will be to understand the spectrum of A when $n, d \gg 1$.

12.0.1 MP Distribution

Let $A = \frac{1}{\sqrt{np}} GG^T - \text{diag}(\frac{1}{\sqrt{nd}} GG^T)$, where $G_{ij} \sim \mathcal{N}(0, 1)$. This is a covariance matrix where the diagonal has been removed. The distribution (which is the Marchenko-Pastur distribution) of the spectrum $p(\lambda)$ has support on $[\sqrt{n/p} + \sqrt{p/n} - 2, \sqrt{n/p} + \sqrt{p/n} + 2]$. There's a couple cases due to the interesting bounds on the support

1. When $n \propto p$, the probability mass lies on a $\mathcal{O}(1)$ interval.
2. When $n \ll p$, you get a semi-circle law (traditional).
3. When $n \gg p$, you'll get a low rank matrix, so it'll be a smaller semi-circle law.

12.0.2 Worked exampled of the non-linear case

Consider a model where $x_i \sim \text{Unif}(\{\pm 1\}^d)$, the activation function is $\sigma(z) = (z^2-1)/\sqrt{2}$, and the matrix is

$$A_{ij} = \left(\frac{x_i^T x_j}{\sqrt{d}}\right)^2 - 1 \tag{12.2}$$

$$= \frac{(\sum_{a=1}^d x_{i,a}x_{j,b})^2}{d} - 1 \tag{12.3}$$

$$= \underbrace{\frac{\sum_{a=1}^d x_{i,a}^2 x_{j,b}^2}{d}}_{=1} + 2\frac{\sum_{a<b} x_{i,a}x_{i,b}x_{j,a}x_{j,b}}{d} - 1 \qquad x_{i,a}^2 = 1 \tag{12.4}$$

$$= \frac{2}{d}\sum_{a<b}(x_{i,a}x_{i,b})(x_{j,a}x_{j,b}) \tag{12.5}$$

If you think of this as an inner-product

$$= \frac{1}{\sqrt{nd^2/2}}\langle f(x_i), f(x_j)\rangle \tag{12.6}$$

where $f(x) = (x_1x_2, x_1x_3, ..., x_1x_d, x_2x_3, ..., x_{d-1}x_d)$ (particularly $f(x) : \mathbb{R}^d \to \mathbb{R}^{d(d-1)/2}$) (In probability and statistics, this function is the **chaos**, basically when you have to use embedding functions which blow up in dimension, that is chaos (to a probability person)). So to recap

$$A = \frac{1}{\sqrt{nd^2/2}}G^TG - \text{diag}(\star) \tag{12.7}$$

$$\text{s.t. } G = \big(f_1, .., f_n\big) \in \mathbb{R}^{n\times d(d-1)/2} \tag{12.8}$$

where $\text{diag}(\star)$ is the the diagonal the matrix prior to it. Going forward let's call the dimension of the output of f, $d(d-1)/2 \equiv p$.

You should notice a couple of nice properties $\mathbb{E}[f(x)] = 0$ (due to entries being iid), and $\mathbb{E}[ff^T] = \mathbb{I}_p$. So, in approximation $f_i \approx \mathcal{N}(0, \mathbb{I})$. Now you can use the results from the previous section, you can characterize the distribution over the eigenvalues depending on the relationships between n and d.

12.0.3 Higher order version of the previous example

Imagine taylor expanding the activation function $\sigma(x) = \sum_k c_k x^k$, we did the quadratic case, so what happens in the 3'rd order case?

$$\left(\frac{x_i^T x_j^3}{\sqrt{d}}\right)^3 = \frac{(\sum_a x_{i,a}x_{j,a})^3}{d^{3/2}} = \text{Combinatorial factors} \tag{12.9}$$

> You can instead expand in **Gegenbauer Polynomials** (which end up looking similar to Hermite polynomials) $\sigma(z) = \sum_k \mu_k q_k(z)$ (see L. & Yau '22). These have nice properties
>
> $$\mathbb{E}_{x\sim\text{Unif}(S^d)}[q_k(x)q_\ell(x)] = \mathbb{1}_{k=\ell} \tag{12.10}$$

where x is sampled uniformly from the surface of a d dimensional sphere, meaning

$$q_k\left(\frac{x_i^T x_i}{\sqrt{d}}\right) = \frac{1}{\sqrt{N_k}}\langle Y_k(x_i), Y_k(x_j')\rangle \tag{12.11}$$

where Y_k are the sphereical harmonics, $N_k \sim \mathcal{O}(d^k)$ (there's a precise answer but here's the scaling). The expectation value of this as

$$\mathbb{E}[Y_k(x)Y_\ell^T(x)] = \begin{cases}\mathbb{I}_{N_k}, & k=\ell\\ 0, & k\neq\ell\end{cases} \tag{12.12}$$

Consider sampling $x_i \sim \text{Unif}(\sqrt{d}S^{d-1})$, and the activation function is $\sigma(z) = \sum_k \mu_k h_k(x)$ where h_k are **Hermite Polynomials**. Then

$$h_k\left(\frac{x_i^T x_j}{\sqrt{d}}\right) = \frac{1}{\sqrt{N_k}}\langle f_k(x_i), f_k(x_j)\rangle + \mathcal{O}_p(1/\sqrt{d}) \tag{12.13}$$

where

$$f_0 = 1 \tag{12.14}$$
$$f_1 = x \tag{12.15}$$
$$f_2 = (x_a x_b)_{a<b} \tag{12.16}$$
$$f_3 = (x_a x_b x_c)_{a<b<c} \tag{12.17}$$

12.1 How to prove the Gaussian Equivalence

So in the previous section, we basically justified a MP distribution would pop out because the first two moment looked like a unit Gaussian. So now we ask the question, can we prove the equivalence? The key lies in **approximate rotational invariance**.

Consider $f_k : \mathbb{R}^n \to \mathbb{R}^{N_k}$ and a vector $b \in \mathbb{R}^{N_k}$. A dumb quantity to inspect is the overlap $|b^T f_k(x)|$. You can then bound this via Cauchy Schwartz

$$|b^T f_k| \leq ||b||\ ||f_k|| \tag{12.18}$$

However, if $f_k \sim \mathcal{N}(0, \mathbb{I})$, then $b^T f_k \sim \mathcal{N}(0, ||b||^2)$. Therefore we have a condition for when f_k is sampled from a Gaussian. This is called **Walsh Chaos**.

$$\boxed{|b^T f| \sim ||b||} \implies b^T f_k(x)/||b|| = \mathcal{O}(1) \tag{12.19}$$

this is the approximate rotational invariance method.

Before continuing, let's define this more rigorously. Consider $z = O_<(1)$, then $||z||_p = (\mathbb{E}[|z|^p])^{1/p} \leq C_p \leq \infty$. This means for all $\epsilon > 0$, there exists a $D > 0$

$$\mathbb{P}(|z| > d^\epsilon) \leq \frac{1}{d^D} \tag{12.20}$$

13 In Context Learning

Definition 2 *In context learning is the abilit9y of large language models to adapts to new tasks by simply being provided with examples / patterns within their input context, without updating their weights.*

ddd

Part IV
Theodore Misiakiewicz: 5 Flavors of Lower Bounds

14 Overview

- Reductions
- Kernel methods
- Noisy Gradient Descent
- SQ Algorithms
- Low degree polynomials (if we have time)

15 Supervised Learning

Here is the mathematical setup for supervised learning.

Consider a dataset $\mathcal{D} = \{(y_i, x_i)\}_{i \le n} \sim_{iid} \mathcal{P}(\mathcal{Y} \times \mathcal{X})$, where $x_i \in \mathcal{X}$ (called the covariate space) and $y_i \in \mathcal{Y}$ (called the label space).

We have a loss function $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_{\ge 0}$. The goal is to fit a model $\hat{f} : \mathcal{X} \to \mathcal{Y}$, which minimizes the **population risk**

$$\mathcal{R}(\hat{f}, \mathcal{D}) \equiv \mathbb{E}_{(y,x)\sim\mathcal{D}}[\ell(y, \hat{f}(x))] \tag{15.1}$$

Example: Let $\mathcal{X} = \mathbb{R}^d, \mathcal{Y} = \mathbb{R}$, and the loss function is the residuals $\ell = (y - \hat{y})^2$. Your population risk can be the mean squared error $\mathcal{R} = \mathbb{E}[(y - \hat{f})^2]$. We define the **excess test error**

$$\mathcal{R}(\hat{f}, \mathcal{D}) = \mathcal{R}(\hat{f}, \mathcal{D}0 - \inf_f \mathcal{R}(\ell, \mathcal{D}) \tag{15.2}$$

$$= \mathbb{E}[(h(x) - \hat{f})^2], \quad \text{s.t. } h(x) = \mathbb{E}[y|x] \tag{15.3}$$

To start constructing bounds, we must apply a little more setup. In this discipline, the workflow goes as follows:

1. Choose a model class
2. Setup the empirical risk minmization problem
3. See what is achievable via some gradient descent type algorithm

To start constructing bounds, let's assume our model is within some function class $\hat{f} \in \mathcal{F} = \{f(x;\theta) : \theta \in \mathbb{R}^p\}$. And going forward, we'll notate the population risk as

$$\mathcal{R}(\theta) \equiv \mathcal{R}(f(\cdot;\theta), \mathcal{D}) \tag{15.4}$$

We also don't have oracle access to the true distribution $\mathcal{P}(\mathcal{Y} \times \mathcal{X})$, so we will have to use an **empirical risk**

$$\hat{\mathcal{R}}_n(\theta) = \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, f(x_i; \theta)) \tag{15.5}$$

$$\hat{\theta} = \text{argmin}_\theta \{\hat{R}_n(\theta) + \pi(\theta)\} \tag{15.6}$$

where π is a regularizer (for example, you can do Ridge which is $\pi(\theta) = ||\theta||_2^2$).

There are three problems which arrises in the supervised learning setup.

1. **Expressivity:** The model class $\mathcal{F}$ is not expressive enough. That is you can't capture the relationship between y and x well enough. However, most neural networks are expressive enough, so we can ignore this issue.
2. **Sample size:** In practice, your empirical risk is performing a MCMC approximation of the population risk. Meaning $\hat{R}_n(\theta) \not\approx \mathcal{R}(\theta)$. This means $\hat{f}$ will fail to generalize.
3. **Runtime:** It's computational hard to minimize $\hat{R}_n(\theta)$ as it is highly non-convex.

We will focus on the third, as they play nice with lower bounds.

16 Reductions

Reductions are finding mapping between problems. If one problem is hard, then the correct map can show related problems are also hard. We'll show how to determine the 3 Node Neural Network is hard via a mapping.

16.1 Hardness of Training NNs

16.2 3 Node Neural Network

[Judd '87] [Blum & Rivest '88]. Assuming $\mathsf{P} \neq \mathsf{NP}$. There is no polynomial time algorithm that (unconditonally) minimizes the empirical risk.

Assume you have a function $f(x_i; \theta) = \text{sgn}(\langle w_i, x \rangle + b_i)$, where the ERM

$$\min_\theta \frac{1}{n} \sum_i (y_i - f(x_i; \theta))^2 \tag{16.1}$$

where the dataset is $y_i \in \{\pm\}$, $x_i \in \{0, 1\}^d$.

Question: Given a set of $\mathcal{O}(n)$ data points, does there exists a 3 node NN that interpolates this data? *Answer [Blum, Rivest]:* Assuming $\mathsf{P} \neq \mathsf{NP}$, training a 3 node NN is NP-complete.

> *Set Splitting Problem:* Given a finite set S and a collection of subset $C = \{C_j : C_j \subseteq S\}$. Does there exists a partition of S in $S_1 \& S_2$ s.t. $C_j \not\subseteq S_1$ and $C_j \not\subseteq S_2$ for all $C_j \in C$? That answer is no.
>
> To prove this, just have to map $(S, C) \to (y_i, x_i)_{i \leq n}$, and you can use the previous result to finish the proof.

Does this mean that machine learning is doomed? In other words, is this a compelling result? Well:

1. This is a worst case result
2. And we've assumed a very particular architecture.

As a rebuttal, in machine learning

1. Average case hardness $\neq$ worst case hardness
2. This previous example was proper, in machine learning we are often doing improper learning.

Definition 3 ((Im)proper Learning) *Consider the function class $\mathcal{H} \subseteq \{h : \mathcal{X} \to \{\pm 1\}\}$.*

- ***Proper Learning** is when $(y_i, x_i) \sim_{iid} \mathcal{D}$. The algorithm needs to output $\hat{h} \in \mathcal{H}$, s.t. $\mathcal{R}(\hat{f}, \mathcal{D}) \leq \epsilon$.*
- ***Improper Learning** is when $\hat{h} \notin \mathcal{H}$*

Basically in proper learning, you constrain the class of models to be a subset of what is defined by the data. In improper learning, you can (for example) over parameterized, and then find solutions this way.

Once we realize there's a difference between proper / improper learning, we are saved! We can show that hardness of proper learning does not imply hardness of improper learning.

Example [Learning Degenerative Normal Forms]: Let $x = (x_1, x_2, ...) \in \{0, 1\}^d$. Let a normal form be

$$h(x) = M_1 \wedge M_2 \wedge M_3 \tag{16.2}$$
$$\text{s.t. } M_i = L_{i1} \vee L_{i2} \vee ... \vee L_{in_i} \tag{16.3}$$

where $\wedge$ is a logical AND, and $\vee$ is logical OR.

A result of this is there is no polynomial time algorithm that can properly learn 3-DNF. However, if you allow for improper learning, you can learn it in polynomial time.

16.3 Hardness of Improper Learning

We've shown hardness of proper learning, can we show where improper learning breaks down?

Definition 4 (Intersection of k hard-spaces) *Let $x \in \{\pm 1\}^d$*

$$h(x) = \prod_{i=1}^{k} \mathbb{1}[\langle w_j, x \rangle > 0] \tag{16.4}$$

Under assumption of hardness. For all $\epsilon > 0$, the internsection of d^ϵ hard-spaces is hard to learning even improperly.

Remark: $h(x)$ can be rewritten a 2-hidden layer ReLU, with 2k+1 neurons. So it seems that teacher-student models will fail to converge (in polynomial time), even when the teacher is a small model. Wow!

17 Kernel Methods

Let the tuple $(\Theta, \langle\cdot,\cdot\rangle)$ (which is a Hilbert space) be called a **feature space**. We have **feature maps** $\Phi : \mathcal{X} \to \Theta$, and we now consider the class of functions

$$\mathcal{F} = \{f(x;\theta) = \langle\theta, \Phi(x)\rangle_\Theta\} \tag{17.1}$$

A couple remarks

- $\langle f(\cdot;\theta), f(\cdot,\theta')\rangle_\mathcal{F} = \langle\theta,\theta'\rangle_\Theta$
- $||f(\cdot;\theta)||^2_\mathcal{F} = ||\theta||^2_\Theta$

Example [Linear Regression]: Let $\mathcal{X} = \mathbb{R}^d$ and $\Theta = \mathbb{R}^d$. Your feature map is $\Phi(x) = x$, and this mean your norm is $||f(\cdot;\theta)||_\mathcal{F} = ||\theta||_2$

Example [Polynomial Regression]: Let $\mathcal{X} = \mathbb{R}$, $\Theta = \mathbb{R}^{k+\ell}$, $\Phi(x) = (1, x, ..., x^k)$. Your model becomes $f(x;\theta) = \sum_i x_i\theta^k$

Example [Infinite-Width 2 Layer Neural Network]: This one is a bit more involved. First you must define a probability space (Ω, μ).

Your feature space is $\Theta = L^2(\Omega,\mu) = \{a : \Omega \to \mathbb{R} : \int a(\omega)^2\mu(d\omega) < \infty\}$. Your inner-product on this space is just the continuous L_2, $\langle a, b\rangle_\Theta = \int a(\omega)b(\omega)\mu(d\omega)$. Your feature map becomes $\Phi : \mathcal{X} \to L^2(\Omega,\mu)$, $x \mapsto \{\omega \mapsto \sigma(\langle x,\omega\rangle\}$. Your function class is $\mathcal{F} = \{f(x;a) = \int a(\omega)\sigma(\langle x,\omega\rangle)\mu(d\omega) : a \in L^2(\Omega,\mu)\}$.

Remark: $(\mathcal{F}, \langle\cdot,\cdot\rangle_\mathcal{F})$ is a reproducing kernel Hilbert space (RKHS).

Definition 5 (Reproducing Kernel Hilbert Space) *Consider a function $K : \mathcal{X}\times\mathcal{X} \to \mathbb{R}$, which is symmetric in its argument, and positive semidefinite (where you imagine it's a infinite dimensional matrix, where indices are just the arguments).*

$$\forall x, f,\ K(x,\cdot) \in \mathcal{F}\ s.t.\ \langle f, K(x,\cdot)\rangle_\mathcal{F} = f(x) \tag{17.2}$$

Theorem 6 (Moore-Arorgogin) *For every symmetric, positive-definite kernel, there exists a unique RKHS.*

Example: Consider the Feature Space (Θ, Φ), we can construct a RKHS by taking the the kernel to be $K(x,x') = \langle\Phi(x), \Phi(x')\rangle_\Theta$.

In linear regression

$$K(x,x') = \langle x, x'\rangle \tag{17.3}$$
$$\tag{17.4}$$

In polynomial regression

$$K(x,x') = \sum_{s=0}^{k}(xx')^s \tag{17.5}$$

In Infinite Width 2-layer NN

$$K(x,x') = \int \sigma(\langle x,\omega\rangle)\sigma(\langle x',\omega\rangle)\mu(d\omega) \tag{17.6}$$

17.1 Kernel Method

Let's define a problem to solve. Consider a neural network $f(\cdot,\theta) = \langle\hat{\theta}, \Phi(\cdot)\rangle_\Theta$

$$\hat{\theta} = \text{argmin}_{\theta\in\Theta}\left\{\frac{1}{n}\sum_{i=1}^{n}\ell(y_i,\langle\theta,\Phi(x_i)\rangle_\Theta) + \lambda||\theta||_\Theta^2\right\} \tag{17.7}$$

$$\hat{f} = \text{argmin}_{f\in\mathcal{F}}\left\{\frac{1}{n}\sum_{i=1}^{n}\ell(y_i,f(x_i)) + \lambda||f||_{\mathcal{F}}^2\right\} \tag{17.8}$$

Theorem 7 (Representor Theorem) *Any solution $\hat{\theta} \in span(\Phi(x_1),...,\Phi(x_n)\} \subseteq \Theta$*

Proof: *Let $\Theta = V_{||} \oplus V_\perp$ where $V_\perp = \{\theta \in \Theta : \langle\theta,\Phi(x_i) = 0 \quad \forall i \in [n]\}$ (it's perpendicular w.r.t. the inner product defined on Θ). You can decompose*

$$||\theta||_\Theta^2 = ||\theta_{||}||_\Theta^2 + ||\theta_\perp||_\Theta^2 \tag{17.9}$$

Now compute

$$\min_{\theta=\theta_{||}+\theta_\perp}\{\frac{1}{n}\sum_{i=1}^{n}\ell(y_i,\langle\theta_{||},\Phi(x_i)\rangle + \underbrace{\langle\theta_\perp,\Phi(x_i)\rangle}_{\to 0,\ by\ def\ of\ \ theta_\perp}) + \lambda||\theta_{||}||^2 + \lambda\underbrace{||\theta_\perp||^2}_{\to 0} \tag{17.10}$$

QED?

17.2 Kernel Trick

Consider $\theta = \sum_i a_i\Phi(x_i)$, which yields a model $f(x;\theta) = \sum_{i=1}^n a_K(x,x_i)$. where $k_m(x) = (K(x,x_1),...,K(x,x_n))$. The minimization

$$\hat{a} = \text{argmin}_{a\in\mathbb{R}^n}\{\frac{1}{n}\ell(y_i,a^TK_n(x_i) + \lambda a^TK_n a\} \tag{17.11}$$

This is a sum of convex functions in a, so this is simple.

$$f(x;\hat{a}) = \sum a_iK(x,x_i) \tag{17.12}$$

This process is known as the **kernel trick**.

17.3 Domions Lower Bound

Theorem 8 *Let $(\mathcal{R},\langle\cdot,\cdot\rangle_\mathcal{R})$ be a Hilbert Space. Let $T \subset \mathcal{R}$ and $dim(T) = m$. Let $\mathcal{H} = \{h_1,...,h_M\} \subseteq \mathcal{R}$.*

The average approximation error

$$\epsilon \equiv \frac{1}{M}\sum_{i=1}^{M}\inf_{f\in T}||f-h_i||_\mathcal{R}^2 \tag{17.13}$$

Then you can show the bound

$$m \geq \frac{M}{||G||_{op}}(1-\epsilon) \tag{17.14}$$

where $g = (\langle h_i, h_j \rangle)_{i,j\in[n]}$

Now let's apply this lower bound.

Example: Consider you get data $x \sim p_{\mathcal{X}}$, and $h(x) = \mathbb{E}[y|x]$. We construct a dataset $(y_i, x_i)_{i\leq n}$.

You have a set of basis function $h \in \mathcal{H} \in \{h_1, ..., h_M\}$. You now consider a subspace $T = \text{span}\{K(\cdot, x_i) : i \in [m]\}$

$$\hat{f}_j \to ||\hat{f}_j - h_j||^2_{L^2} \geq \inf_{f\in T} ||f - h_j||^2_{L^2} \tag{17.15}$$

You have

$$AAE \geq \frac{1}{M}\sum_j \inf_{f\in T} ||f - h_j||^2_{C^2} = \epsilon \tag{17.16}$$

To prove $AAE \leq \epsilon$, you must have number of samples $n \geq \frac{M}{||G|_{op}}(1-\epsilon)$,

18 Noisy Gradient Descent

Consider a model $f(\cdot;\theta) : \mathcal{X} \to \mathbb{R}$ s.t. $\theta \in \mathbb{R}^p$. Assume this model is differentiable, that is $\nabla_\theta f(x;\theta)$ exists for all x, θ. Our loss is a mean-squared error $\ell(y, \hat{Y}) = (y - \hat{y})^2$, meaning the population risk will be $\mathcal{R}(\theta) = \mathbb{E}_{\mathcal{D}}[(y - f(x;\theta))^2]$. We have diagnostics regression fit $h(x) = \mathbb{E}_{\mathcal{D}}[y|x]$ and an excess risk $\mathcal{R}_h(\theta) = ||f(\cdot;\theta) - h||^2_{L^2}$.

Today, we'll consider minimizing the population risk using **online SGD**, this is the following algorithm

- Initialize the weights from some distribution $\theta^0 \sim p_0$
- At each step k
  - Sample data $(y_k, x_k) \sim \mathcal{D}$
  - Do a weight update according to the gradient

$$\begin{aligned}
\theta^{k+1} &= \theta^k - \eta_k \nabla_\theta \ell(y_k, f(x_k, \theta^k)) && (18.1)\\
&= \theta^k + \eta_k \underbrace{(y_k - f(x_k;\theta^k)) \prod_{B(O,R)} \nabla_\theta f(x_k;\theta^k)}_{\equiv -\Delta(\theta^k; y^k, x^k)} && (18.2)\\
&= \theta^k - \eta_k \underbrace{g(\theta^k)}_{\text{drift}} - \eta_k \underbrace{(\Delta(\theta^k; y^k, x^k) - g(\theta^k))}_{\text{noise / Martingale}} && (18.3)
\end{aligned}$$

where $g(\theta^k) = \mathbb{E}_{(y_k,x_k)\sim\mathcal{D}}[\Delta(\theta^k; y_k, x_k)]$

Remarks

- $z = \frac{1}{\sqrt{\eta}}$
- Notation: $g_h(\theta^k) \equiv g(\theta^k)$

You also have **noisy gradient descent**, this is a slightly different algorithm

1. Sample an initial weight configuration $\theta^0 \sim p_0$
2. Perform a gradient update $\theta_h^{k+1} = \theta_h^k - \eta_k g_h(\theta^k) + \eta_k \xi_k$, where $\xi_k \sim \mathcal{N}(0, \sigma^2 \mathbb{I}_p)$

Our objective will be to find a lower-bound on $R_h(\theta_h^k)$.

Hypothesis class $\mathcal{H} \subseteq \{h : \mathcal{X} \to \mathbb{R}\}$

Definition 6 (Correlation Alignment Complexity (CAC)) *Let $\mathcal{H} \subseteq \{h : \mathcal{X} \to \mathbb{R}\}$ by a hypothesis class that you'll test functions in, and μ_H be a prior over said hypothesis class. Also $\nu \in L^2(\mathbb{P}_\mathcal{X})$.*

$$Align_C(\mu_H, \nu) = \left\{ \sup_{||\phi||_{L_2} \leq 1} \mathbb{E}_{h \sim \mu_H}[\langle \phi, h - \nu \rangle_{L^2}^2] \right\} \tag{18.4}$$

Theo created this measurement, how.

Remark: Large CAC fits in $\mathcal{H}$ centered by reference g are mainly orthogonal to any fixed direction.

Theorem 9 (Avbbe, Boirn-Adsen, 2022) *Consider θ_h^k, which is the parameters after k steps of noisy GD.*

$$\mathcal{R}_h(\theta_h^k) \geq ||h_k - v||_{L^2}^2 - \epsilon \tag{18.5}$$

with probability at least

$$1 - \left(\frac{R}{2z} \sqrt{\frac{k}{A}} + \frac{1}{\epsilon A} \right) \tag{18.6}$$

Remark

- If k is too small, we don't improve on the trivial prediction, i.e. you'll always output ν. This means you want

$$k \geqslant \left(\frac{z}{R} \right)^2 \text{Align} \tag{18.7}$$

- z/R is the gradient precision. If Align is large, we'll either need a large amount of steps or small enough precision. Meaning you want

$$fk \geqslant \frac{\text{Align}}{R^2} \tag{18.8}$$

18.0.1 Proof

The idea is to use "junk flow". This means you compare a noisy GD trajectory to another noisy GD trajectory, where the label is replaced with a "junk" label. Basically, if you can show that the two trajectories stay close together, then you can show the original trajectory can't minimize the test error.

Consider noisy GD on true target h, $\{\theta_h^t\}_{t=0}^k$, and noisy GD on junk target ν $\{\theta_\nu^t\}_{t=0}^k$.

We can further break down this proof by constructing two lemmas (which we'll not prove)

Lemma 2

$$\mathbb{E}_{h\sim\mu_H}[TV(\theta_n u^k, \theta_h^k)] \leq \frac{R}{2z}\sqrt{\frac{k}{A}} \tag{18.9}$$

There exists coupling θ_ν^k, θ_h^k *s.t.* $\mathbb{P}(\theta_h^k \neq \theta_\nu^k) \leq \frac{R}{2z}\sqrt{\frac{k}{A}}$

Lemma 3

$$\mathbb{P}(R_h(\theta_\nu^k) \leq ||h-\nu||_{L^2}^2 - \epsilon) \leq \frac{1}{\epsilon A} \tag{18.10}$$

Proof of theorem

$$\mathbb{P}(R_h(\theta_h^k) \leq t) \leq \mathbb{P}(\theta_h^k \neq \theta_\nu^k) + \mathbb{P}(R_k(\theta_\nu^k) \leq t) \tag{18.11}$$

$$= \frac{R}{2z}\sqrt{\frac{k}{A}} + \frac{1}{\epsilon A} \tag{18.12}$$

where we plug and chug the previous two lemmas

19 Statistical Query Algorithms

Consider a map $\phi : \mathcal{Y}\times\mathcal{X} \to [-1,1]$, where you have an objective $\mathbb{E}_{\mathcal{D}}[\phi(y,x)]$. This framework is nice because it makes it easy to find tight lower bounds in the form, can the algorithm succseed with tolerance $\mathbb{E}_{\mathcal{D}}[\phi(y,x)] + \epsilon$ (for $\epsilon \in [-\delta, \delta]$).

You then ask

Part V
Florent Krzakala: From Spin Glasses to Statistics & ML

Let's introduce a problem which we'll work on throughout the week. We'll introduce the **Spiked-Wigner** problem. This is a vector $x \in \mathbb{R}^d$, where $x \sim p_x(x)$, you want to infer it from observation

$$Y = \sqrt{\frac{\lambda}{d}} x^T x + W \tag{19.1}$$

where $W \sim \text{GOE}(d)$ ($W_{ij} \sim \mathcal{N}(0,1)$). We are interested in the $d \to \infty$ limit.

If I tell you $p_x(x)$, then you get a prior for x, and we can be bayesian.

19.1 Naive Algorithm

Start with $x^0 \in \mathbb{R}^d$. Compute

$$h^t = Ax^t \tag{19.2}$$
$$x^t = f_t(h^t) \tag{19.3}$$

where A is a matrix, $A_{ij} = A_{ji} \sim U(0, 1/d)$. It's very tempting to think $h^t \propto z$ (Gaussian vector) for all time. However after one time-iteration, x becomes correlated with A, so you loose this property... However, I'd love for $h^t \propto z$, it's so nice.

SO let's see if we can massage it. Let's set $f_t(h) = h$

$$h^1 = Ax^0 \tag{19.4}$$
$$h^2 = A^T A x \simeq (\mathbb{I} + \tilde{A})x^0 = \tilde{A}x_0 + x_0 \tag{19.5}$$

19.2 Approximate Message Passing

Let's instead do the following algorithm

$$h^t = Ax^t - \underbrace{b_t x^{t-1}}_{\text{Onsager term}} \tag{19.6}$$
$$x^t = f_t(h_t) \tag{19.7}$$

where $b_t = \frac{1}{d}\sum_i \partial_i f(x)$ (where $\partial_i = \frac{\partial}{\partial x_i}$, the component-wise gradient). What's really nice, is your iterations now go

$$h^1 = Ax^0 \tag{19.8}$$
$$h^2 = A^T A x^0 = \tilde{A}x_0 + x_0 - x_0 = \tilde{A}x_0 \tag{19.9}$$

So if you take the limit where $d \to \infty$. Then you have the following

$$h^t =^{(d)} \sqrt{q^t} z \tag{19.10}$$

where $z \sim \mathcal{N}(0,1)$. Meaning if you take the expectation of some sufficient statistic g acting on h

$$\mathbb{E}[g(h)] = \mathbb{E}[g(\sqrt{q}z)] \tag{19.11}$$

Since we've shown h is recursively related by it's previous step, this defines the variance. Meaning $q_t = \mathbb{E}[f_{t+1}^2(z\sqrt{q_t})]$. Physicists have been thinking of this as a fixed point, not an iteration.

But anyways, perhaps you should replace A with Y, and then choose f s.t. $\mathbb{E}[x^t] \to x$. Meaning you've created an algorithm which picks out the hidden signal!

19.3 Approximate Message Passing for Spiked Wigner Model

Consider the algorithm

$$h^{t+1} = \tilde{Y}x^t - b_t x^{t-1} = A\hat{x}^t - b_t\hat{x}^{t-1} + \sqrt{\lambda}\frac{x^{*T}x^*}{d}\hat{x}^t \tag{19.12}$$

$$x^t = f_t(h^t) \tag{19.13}$$

we'll call $m_t = \hat{x}^t \cdot x^*/d$, this is the **overlap** between the iterative scheme and true signal. Now we can write

$$h^{t+1} = A\hat{x}^t - b_t\hat{x}^{t-1} + \sqrt{\lambda}m^t x^* \tag{19.14}$$

Define quantities

$$\tilde{h} := Ax^t - \tilde{b}_t\hat{x}^{t-1} \tag{19.15}$$

$$x^t = f_t(\tilde{h}_t + \sqrt{\lambda}mx^t) = \tilde{f}_t(\tilde{h}_t) \tag{19.16}$$

Notice, we've recovered the self-consistency condition! This yields the stat equation (self-consistency condition) for AMP

$$h^t = \sqrt{q^t}z + \sqrt{\lambda}m^t x^* \tag{19.17}$$

$$q^{t+1} = \mathbb{E}[f_t^2(\sqrt{q^t}z + \lambda m x^*)] \tag{19.18}$$

$$m^{t+1} = \mathbb{E}_{x^*,z}[x^* f_t(\sqrt{q^t}z + \sqrt{\lambda}m^t x^*)] \tag{19.19}$$

So now how do we find x^*? Well, we have a h^t which is a linear combination between z and x^*, so you can use Tweedy's formula to find the best estimate (which minimizes mean squared error)

$$p(x|h) = \mathcal{N}(h^t/\sqrt{\lambda}m^t, q^t/\lambda m^{t2}) \tag{19.20}$$

Which means

$$f_t(h) = \frac{\mathbb{E}[xp_x(x)\mathcal{N}(h/\sqrt{\lambda}m^t, q^t/\lambda m^{2t})]}{\mathbb{E}[p_x(x)\mathcal{N}(\cdot,\cdot)]} \tag{19.21}$$

19.4 Nishimori Relation

If you're doing any Bayes estimation

$$\boxed{\mathbb{E}_{y,x,x^*}[g(x,x^*) = \mathbb{E}_y\mathbb{E}_{x^{(1)}|y}\mathbb{E}_{x^{(2)}|y}[g(x^{(1)},x^{(2)}]} \tag{19.22}$$

This is trivial via Bayes theorem. Lol.

Via this relationship, you can show for the previous model $m^t = q^t$.

19.5 Example: Ising Spins

Let $x_i^* = \pm 1$ (these are Ising spins), then you recover (exaclty) the mean field self-consistency equation

$$f_t(h) = \tanh(\sqrt{\lambda} h) \tag{19.23}$$

If you know the algorithm converges, then you can say

$$m^{t+1} = \mathbb{E}[\tanh^2(\sqrt{\lambda} z + \lambda m x^\sigma)] \tag{19.24}$$

You can see that there's a phase transition, where m is non-zero when $\lambda > 1$. Wow! So there's a bold claim that this is the bayesian optimal result.

> It's an open-problem to show this algorithm converges for all initalizations.

19.6 Example: BR Model

This is when x^* is

$$\mathbb{P}[x^* = 0] = 1 - \rho \tag{19.25}$$
$$\mathbb{P}[x^* = 1] = \rho/2 \tag{19.26}$$
$$\mathbb{P}[x^* = -1] = \rho/2 \tag{19.27}$$

What's fascinating, is that you still get the phase transition at $\lambda = 1$...

19.7 Example: PCA

Set your Bayes optimal denoiser function to be

$$\hat{x} = f_t(h) = \frac{h\sqrt{d}}{||h||_2} \tag{19.28}$$

Going back to the AMP algorithm

$$h^{t+1} = \tilde{Y} x^t - b_t \hat{x}^{t+1} \tag{19.29}$$
$$= \tilde{Y} \frac{h^t \sqrt{d}}{||h||_2} - \frac{\sqrt{d}}{||h^t||_2} \frac{h^{t-1}\sqrt{d}}{||h^t||_2} \tag{19.30}$$

Let's assume this converges.

$$h = \tilde{Y} \frac{h\sqrt{d}}{||h||_2} - \frac{\sqrt{d}}{||h||_2} \frac{h\sqrt{d}}{||h^t||_2} \tag{19.31}$$

This implies

$$\left(\frac{\sqrt{d}}{||h||_2} + \frac{||h||_2}{\sqrt{d}} \right) h = \tilde{Y} h \tag{19.32}$$

This means that, h is a eigenvector, with eigenvalue $\frac{\sqrt{d}}{||h||_2} + \frac{||h||_2}{\sqrt{d}}$. So if you can get the highest eigenvalue by doing power iteration.

Making use a Tweedy's formula

$$h = z + \sqrt{\lambda} m x^* \tag{19.33}$$

You find $||h||_2 = \sqrt{1+\lambda m^2}$, you now find your eigenvector equation becomes

$$\left(\sqrt{1+\lambda m^2} + \frac{1}{\sqrt{1+m^2}}\right) h = \tilde{Y} h \tag{19.34}$$

You now find a BBAP transition, where your spectrum of $\tilde{Y}$, which is given by $\left(\sqrt{1+\lambda m^2} + \frac{1}{\sqrt{1+m^2}}\right)$.

20 Cavity Method on SK Model

Consider hte SK model

$$p(s) = \frac{1}{Z} \exp\left(\beta \sum_{i \leq j} s_i s_j J_{ij}\right) \tag{20.1}$$

where $s = \{s_1, ..., s_N\}$. You assume there's disordered in the couplings $J_{ij} \sim \mathcal{N}(\frac{\sqrt{\lambda}}{d}, \frac{1}{d})$. To do the cavity method, consider a system where the i'th spin is removed

$$p(s_i) \propto \exp(\beta \sum_j s_i J_{ij} m_j^{(c)}) \tag{20.2}$$

You recover a self-consistency equation

$$m_i = \tanh(\beta \sum_j J_{ij} m_j^{(c)}) \tag{20.3}$$

If you taylor expand m_j

$$m_j = m_j^{(c)} + \frac{\partial m_j}{\partial m_j^{(c)}} + ... \tag{20.4}$$

$$= m_j^{(c)} + (1 - m_j^2) J_{ij} \beta m_i + ... \tag{20.5}$$

plugging this back into the self-consistency equation

$$\vec{m} = \tanh(\beta J \vec{m} - \vec{m} \beta \mathbb{E}[1 - \vec{m}^2]) \tag{20.6}$$

This yields the **TAP equation**. If you attempt to iteratively solve this equation, you recover our previous result from yesterday... Wow!

20.1 Free Energies & How to Prove Them

Yesterday, we claimed that our solution was a Bayes optimal result. We can prove this by making a free-energy argument...

Recall yesterday's problem

$$Y = x^t x \sqrt{\frac{\lambda}{d}} + \xi \tag{20.7}$$

You get the probability distribution of $Y|x$, and you can ask the Bayes' question, what's the probability of observing $x|Y$

$$P(x|Y) =\propto p(x) \exp\left(-\frac{1}{2}\sum_{i\leq j}(y_{ij} - x_i^* x_j^* \sqrt{\frac{\lambda}{d}})^2\right) \tag{20.8}$$

$$= \frac{1}{Z_d(y)} \exp\left(-\frac{\lambda}{2d}\sum_{i<j} x_i^2 x_j^2 + \sqrt{\frac{\lambda}{d}\sum_{i\leq j} y_{ij} x_i x_j}\right) \tag{20.9}$$

So we are interested in the **free-energy density**

$$f = -\frac{1}{d}\log Z_d(y) \tag{20.10}$$

What's nice is this quantity is related to the mutual information between x and Y. What's really nice tho, is this system is **self-averaging**, meaning in the limit $d \gg 1$, it will converge to its mean; so let's just take the mean now.

$$f = \mathbb{E}\left[\frac{1}{d}\log Z_d(y)\right] \tag{20.11}$$

Obviously, this is difficult to do, so we can perform the **replica trick**.

$$\lim_{n\to 0}\frac{Z^n - 1}{n} = \log Z \implies \frac{\mathbb{E}[Z^n] - 1}{n} = \mathbb{E}[\log Z] \tag{20.12}$$

where we take n to be discrete, evaluate $\mathbb{E}$, and then analytically continue it down to zero. This is a non-rigorous, so perhaps there are other methods to do this?

For this problem, assume that you used replica trick to get an answer. Now you'll use other methods to rigorously prove it. So we'll use **Guenne Interpolation (Smart Path)**.

What you'll do is write a time-dependent Hamiltonian, where at $t = 0$ it'll be something super easy to compute H_{easy}, and at $t = 1$ it'll be the problem we're computing H_{real}.

$$f_{t=0} + \mathbb{E}\int_0^1 \frac{\partial(-\log Z)}{\partial t} = -\frac{1}{d}\mathbb{E}[\log Z_d] \tag{20.13}$$

Usually this only gives you a bound, but in simple cases it can solve the problem exactly.

For this problem we choose the following two as our Hamiltonians

1. Vector denoising (easy problem):

$$\tilde{x}_i = \sqrt{\lambda'} x_i^* + \xi_i, \quad i = 1, ..., d \tag{20.14}$$

2. The true problem

$$y_{ij} = \sqrt{\frac{\lambda}{d}} x_i^* x_j^* + \xi_{ij} \tag{20.15}$$

This yields the probability distribution of the interpolated Hamiltonian

$$p(x|y,\tilde{x}) \propto e^{H^{(0)}+H^{(1)}} \tag{20.16}$$

The partition function of the interpolated distribution is given by

$$-\mathbb{E}[\log Z_d] = f_{DM}(m) + \int_0^1 dt \frac{\lambda}{4}\mathbb{E}[\langle(\frac{x^T x}{d} - m)^2\rangle - \frac{\lambda}{2}[\langle(\frac{x^T x^*}{d} - m)^2\rangle] - \frac{\lambda m}{d} + \frac{\lambda...}{2} \tag{20.17}$$

$$f_{DM}(m) = \mathbb{E}_{\xi,x^*} \log \mathbb{E}_x \exp\left(-\frac{\lambda m x^2}{2} + \lambda m x x^* + \sqrt{\lambda m} x\xi\right) \tag{20.18}$$

where $\mathbb{E}$ is average over disorder, and $\langle\cdot\rangle$ is the posterior average.

If you apply the Nishmori identity, you should recall that the $x^T x$ and $x^T x^*$ terms under expectation are the same. Once you apply this, you find your answer is a perturbation away from the solution given by replica method

$$f = f_{replica}(m) + \text{positive constant} \tag{20.19}$$

$$\implies f \leq f_{DM}(m) + \frac{\lambda m^2}{4} \tag{20.20}$$

Now you can solve for f by doing saddle point approximation. Krzakala does a lot of justification why this works, but basically (as a physicists knows), usually you can make a concentration argument as the number of spins goes to infinity; meaning the saddle point solution is the true solution.

21 A Single Index Story

Consider weights $\mathbb{R}^d \ni \theta \sim p(\theta)$.

We'll work with the **generalized linear model (GLM)**, these are now called **single index models**. What is this?

Say you have Gaussian data $X \in \mathbb{R}^{n\times d}$, s.t. $X_{ij} \sim \mathcal{N}(0, \frac{1}{d})$. You create a vector h

$$\mathbb{R}^n \ni h = X\theta \tag{21.1}$$

This said to be a single index model because it only depends on a single vector θ.

Example: $y_i = h_i + \xi_i\sqrt{\Delta} = X_i\theta + \sqrt{\Delta}\xi$

Example: $y_i = h_i^2$, which is a phase / sign retrivial problem

Example: $y_i = \text{sign}(h_i + \xi)$.

We want to determine the behavior of this model in high dimension $d \to \infty$ and a lot of training data $n \to \infty$, s.t. the ratio between them is constant $n/d = \alpha$.

The research question for this model:

1. Can we recompute the free-energy / mutual information (this is the $\log Z$ quantity), and prove it?
2. Can you also do it via Bayes AMP?
3. Can we generalize this calculation to account of logistic regression? Or other penalties such as Lasso or Ridge?

21.0.1 Free Energy

So, after much calculation, you can get the quantity

$$f = \mathbb{E}[\frac{1}{d}\log Z] \to \sup_m \inf_n f_{RS} \tag{21.2}$$

where $\rho = \mathbb{E}[\theta]$

$$f_{RS} = \psi(r) + \alpha\varphi(q,p) - \frac{rq}{2} \tag{21.3}$$

$$\psi(r) = \mathbb{E}_{z,x}\log\mathbb{E}_x \exp\left(-rxx + \sqrt{2}xz_0 - rx^2/2\right) \tag{21.4}$$

$$\varphi(q) = \mathbb{E}\log\mathbb{E}P(y_0|\sqrt{n}v + \sqrt{\rho - n}w) \tag{21.5}$$

21.0.2 Bayes AMP

Consider the algorithm

$$\mu^{t+1} = Ag_t(v^t) - d_t e_t(\mu^t) \tag{21.6}$$

$$v^t = Ae_t(\mu^t) - b_t g_{t-1}(v^{t-1}) \tag{21.7}$$

Some names for these things: e_t is the Bayes denoiser for p_θ + noise, and $g_t = \mathbb{E}[h|Y]$ (up to constant noise). And $d_t = \frac{\alpha}{n}\sum_i g_t(\theta_i^t)$, and $b_t = \frac{1}{d}\sum_{i=1}^n e_t(\mu_i^t)$ is the Onsager term. And the matrix entries are $A_{ij} \sim \mathcal{N}(0, \frac{1}{d})$.

What's cool is that this algorithm is exact to yesterday's algorithm via a transformation

$$h = A_s\beta^t - b_t\beta^{t-1} \tag{21.8}$$

$$\beta^t = f_t(h) \tag{21.9}$$

where $A_s = \sqrt{1/(1+\alpha)}\begin{pmatrix} B & A \\ A^T & C \end{pmatrix}$, $f_{2t}(\cdot) = \sqrt{1+\alpha}(O/e_t(\cdot))$, and $f_{2t+1}\sqrt{1+\alpha}\,(g_t(\cdot)/O)$.