FOR THE LOVE OF MESSAGE PASSING—CONVERTING AMP TO DMFT

MAX LOVIG

1. A Very Brief Introduction To Approximate Message Passing

Given "data" $W \in \mathbb{R}^{n\times n}$, we want to consider an algorithm (z_t, u_t) of the following form given intialization u_1,

$$\begin{aligned} z_t &= Wu_t \\ u_{t+1} &= f(z_t). \end{aligned} \tag{1.1}$$

Why could this algorithm be useful? Lets consider a simple application on deriving the semi-circular distribution [Yang 20-Tensor Programs 3]. Let $W \sim$ GOE(n) (eesentially a symmetric Gaussian matrix with element-wise variance $1/n$).

Hutchinson's Trick: Given a matrix $A \in \mathbb{R}^{n\times n}$ and $g \sim \mathcal{N}(0, I_n)$, we have that

$$\mathbb{E}[g^\top Ag] = \mathrm{Tr}(A).$$

So, we can look at calculating the moments of the spectrum of W in an algorithmic way by running (1.1) for t iterations with $f = \mathrm{Id}$ and $u_0 = g$ and invoking a.s. limits to get that

$$\mu_t = g^\top u_t \overset{a.s.}{\to} \mathbb{E}[\lambda(W)^t].$$

Repeating this for all finite t and then invoking that compact support implies uniqueness of a distribution in terms of moments, we can hopefully recover the semi-circle distribution so long as we canb asymptotically track the iterates of (1.1).

Unfortunately (at least for now), we can't do this immediately for the following reason: Say we have already calculated z_1, u_2 (since we are using $f = \mathrm{Id}$ these objects are identical). A trivial exercise is to prove that element-wise $(z_1)_i \sim \mathcal{N}(0, \|u_1\|_2^2)$ up to some asymptotically vanishing error.

Ok, simple enough, now lets calculate $z_2 = W^2u_1$. There is an initial issue, the matrix W^2 is not of simple Gaussian form meaning we need to utilize some cumbersome central limit theorem type arguments (yucky).

After this realization, a natural observation is to instead write $z_2 = Wz_1$ and condition on z_1, u_1 (meaning we condition on the event that $z_1 = Wu_1$). This still causes problems since these dependencies introduce correlations as the second iterate (conditionally) is of value (where $\tilde{W}$ is an i.i.d. copy),

$$z_2 = Wz_1 = WP_{u_1}z_1 + P^\perp_{z_1}WP^\perp_{u_1}z_1 = \underbrace{Wu_1u_1^\top z_1/\|u\|^2}_{(A)} + \underbrace{(I - z_1z_1^\top/\|z_1\|^2)\tilde{W}(I - u_1u_1^\top/\|u\|^2)z_1}_{(b)},$$

we then have that, up to a small vanishing error, that

$$(A) = \frac{u_1^\top z_1}{\|u\|^2}z_1$$

Date: September 26, 2025.

$$(B) \approx \mathcal{N}(0, \|z_1\|^2).$$

The equation for (B) comes from the fact that the contribution of $\tilde{W}$ into the row-space of z_1 and columns space of u_1 is $o_p(1)$. Then as a final step we have that $(A) = \frac{1}{n}\frac{u_1^\top z_1}{\frac{1}{n}\|u_1\|^2} z_1 = b_1 z_1$. This all leads to

$$z_2 \approx \mathcal{N}(0, \|z_1\|^2) + b_1 z_1.$$

The correlations of the type $b_1 z_1$ will build over iterations and become (for now) very difficult to track in a convenient way.

So, how can we fix this? Let remove the bz terms iteratively in out algorithm, note for a general choice of z this takes the form,

$$\begin{aligned} z_t &= W u_t - b_t u_{t-1} \\ u_{t+1} &= f(z_t). \end{aligned} \tag{1.2}$$

Iteration (1.2) is the famed Approximate Message Passing Algorithm in its simplest form. The high dimensional limit as $n \to \infty$ then allows us to construct variables $\bar{z}_t \sim \mathcal{N}(0, \frac{1}{n}\mathbb{E}[f(\bar{z}_{t-1})^\top f(\bar{z}_{t-1})])$ and have a flavor of concentration bounds, the strongest of which takes form in either [Reeves 25-Dimension-Free Bounds] and [Lovig et al 25-On Universality] which is of the form.

Theorem 1.1. *Under a joint coupling of z_t and $\bar{z}_t$ denoted by $\mathbb{P}$*

$$\mathbb{P}(\|z_t - \bar{z}_t\|_2 \geq C_D \log^D(n)) \leq \frac{1}{n^D}.$$

Remark 1.2. As $\|z_t\|, \|\bar{z}_t\| = \Theta(\sqrt{n})$, this is a rather strong result in terms of the concentration in the difference of our AMP algorithm and the state evolution variables.

Example 1.3. *Consider the random matrix $A = \frac{\lambda}{n} x x^\top + W$ where W is a GOE matrix, then we consider the algorithm*

$$z_t = A u_t - b_t u_{t-1}, \qquad u_{t+1} = f(z_t),$$

we can rewrite this algorithm as,

$$z_t = W u_t - b_t u_{t-1}, \qquad u_{t+1} = f(z_t + \frac{\lambda}{n} x x^\top u_1).$$

The, using the above state evolution results we have that

$$z_t \sim \mathcal{N}\left(0, \frac{1}{n} f(z_{t-1} + \lambda x \frac{x^\top u_{t-1}}{n})^\top f(z_{t-1} + \lambda x \frac{x^\top u_{t-1}}{n})\right).$$

Further assuming that $x_i \sim \mathbb{P}$ independently, f is a seperable function ($f(z) = (\tilde{f}(z_1), \ldots, \tilde{f}(z_n))$ and $x^\top u_t / n \to \mu_t \overset{a.s.}{\to} \mathbb{E}[\Theta f(z_{t-1} + \lambda \mu_{t-1} \Theta)]$, then we get the well known state evolution statement

$$z_t \sim \mathcal{N}\left(0, \mathbb{E}[f(z_{t-1} + \lambda \Theta \mu_{t-1})^2]\right).$$

The above result is sufficient to understand the algorithmic limits of recovery for the $\mathbb{Z}_2$-Syncronization problem.

Before introducing the GFOm method, we leave a more advanced form of AMP and its state evolution here for reference,

Theorem 1.4. *Consider the algorithm,*

$$\begin{aligned} z_t &= W u_t - \sum_{s<t} b_{ts} u_{t-1} \\ u_t &= f_t(u_{1:t}). \end{aligned} \tag{1.3}$$

Then when W is GOE and f_t satisfies some mild regularity conditions then,

$$z_{1:t} \approx \mathcal{N}(0, \Sigma_t),$$

where $(\Sigma_t)_{r+1,s+1} = \frac{1}{n}\mathbb{E}[f_r(Z_{1:r})^\top f_s(Z_{1:s})]$.

2. General First Order Methods

Returning to our algorithm (1.1) for the Wigner matrix, we still don't have a good description of algorithms without the Onsager correction term.

Thankfully, a simple (yet powerful) trick is here to save us, we can always rewrite algorithm (1.1) (in fact a version of this algorithm with history terms ala (1.3)) in the form,

$$\begin{aligned} z_t &= W u_t - \sum_{s<t} b_{ts} u_{t-1} \\ u_{t+1} &= f_t(z_t + \sum_{s<t} b_{ts} u_{t-1}). \end{aligned}$$

This is an AMP algorithm with the functions $\hat{f}_t(\cdot) = f_t(\cdot + \sum_{s<t} b_{ts} u_{t-s})$, and because the coefficients $b_{t,s}$ only rely on the past history before time t and have deterministic almost sure limits then we can apply AMP theory to a general algorithm of the form

$$\begin{aligned} z_t &= W u_t \\ u_{t+1} &= f_t(z_{1:t}), \end{aligned}$$

by iteratively adding and removing higher time-order Onsager correction terms. This is the main basis of the result in [Reeves 25] which given the following (slightly simplified) state evolution statement for a **General First Order Method**.

Theorem 2.1. *Consider the following algorithm*

$$z_t = W f_t(z_{1:(t-1)}) + W \breve{f}_t(z_{1:(t-1)}). \tag{2.1}$$

For each $t \in \mathbb{N}$ we can construct random variable $\bar{z}_t$ under a coupling $\mathbb{P}$ where, as $n \to \infty$,

$$\mathbb{P}(\|z_t - \bar{z}_t\| \geq c\sqrt{r}) \leq C' e^{-r}.$$

Where $\bar{z}_t = m_t + r_t$ with,

$$\begin{aligned} m_t &= \breve{f}_t(\bar{z}_{1:(t-1)}) + \sum_{s<t} b_{st} f_s(\bar{z}_{1:(s-1)}) \\ b_{st} &= \frac{1}{n}\mathbb{E}[\mathrm{div}^s f_t(\bar{z}_{1:(t-1)}] \\ r_{1:t} &\sim \mathcal{N}(0, \Sigma) \\ \Sigma_{st} &= \frac{1}{n}\mathbb{E}[f_s(\bar{z}_{1:(s-1)})^\top f_t(\bar{z}_{1:t})] \end{aligned}$$

where div^s *is the divergence on the argument* $\bar{z}_s$.

Remark 2.2. To close our example of calculating the semi-circle law, if we choose $f_t = \mathrm{Id}$ and $\breve{f}_t = 0$, then we arrive at each m_t being the t-th Catalan number, defining the moments of the semi-circle law.

2.1. **Extensions By Embedding.** Often times, an algorithm may never fit exactly in the form of (2.1). Fortunately, a majority of extensions for desired algorithms can be embedded into (2.1). The most common example is the following type of analysis.

Rectangular AMP iterates: Consider a GFOM algorithm of the following form initialized at $v_0 \in \mathbb{R}^{d\times m}$,

$$\begin{aligned} z_t &= Xg_t(v_{0:t}) + \breve{g}_t(z_{0:(t-1)}) \\ v_{t+1} &= X^\top f_t(z_{0:t}) + \breve{f}_t(v_{0:t}) \end{aligned}$$

where $X \in \mathbb{R}^{n\times d}$, $z^t \in \mathbb{R}^{n\times m}$, $v_t \in \mathbb{R}^{d\times m}$, $f_t : \mathbb{R}^{n\times m\times(t+1)} \to \mathbb{R}^{d\times m}$, $g_t : \mathbb{R}^{d\times m\times(t+1)} \to \mathbb{R}^{n\times m}$, $\breve{g}_t : \mathbb{R}^{n\times m\times(t+1)} \to \mathbb{R}^{n\times m}$, $\breve{f}_t : \mathbb{R}^{d\times m\times(t+1)} \to \mathbb{R}^{d\times m}$ are Lipschitz in all arguments.

First, we consider the single column version of (2.1), i.e.,

$$\begin{aligned} z_t &= Xg_t(v_{0:t}) - \breve{g}_t(z_{0:(t-1)}) \\ v_{t+1} &= X^\top f_t(z_{0:t}) - \breve{f}_t(v_{0:t}) \end{aligned} \tag{2.2}$$

where $z_t \in \mathbb{R}^n$, $v_{t+1} \in \mathbb{R}^d$, where again, each function above is Lipschitz in all arguments. We embed this algorithm into a symmetric version of GFOMs analyzed by **?** and then derive the DMFT equations for the above single column asymmetric GFOM.

Consider the symmetric GFOM model analyzed in **?**, specifically

$$x_t = Ah_t(x_{0:t-1}) + \breve{h}_t(x_{0:(t-1)}), \tag{2.3}$$

where $A \in \mathbb{R}^{(n+d)\times(n+d)}$. Recall from above that we want to write (2.2) as the above symmetric GFOM. To do so, we construct matrix A to have block structure

$$A = \sqrt{\frac{n}{n+d}}\begin{bmatrix} C & X \\ X^\top & D \end{bmatrix} \in \mathbb{R}^{(n+d)\times(n+d)},$$

where C and D are symmetric independent Gaussian matrices with off-digonal elements of variance $1/n$ and on-diagonal elements with variance $2/n$. Further, we consider the initialization

$$x_0 = \begin{bmatrix} 0 \\ v_0 \end{bmatrix}.$$

Lemma 2.3. *We aim to prove that there exists choices of $(h_t)_{t\in\mathbb{N}}$ such that $[z_0, \dots, z_t] = (x_{2j-1}[1 : n])_{j\in[t+1]}$ and $[v_0, \dots, v_{t+1}] = (x_{2j}[(n+1) : (n+d)])_{j\in[t+1]\cup\{0\}}$. Therefore, endowing the DFMT equations of algorithm (2.3) to algorithm (2.2).*

Proof. Consider the base case of the statement, i.e. proving that, $x_1[1 : n] = z_0$ and $x_2[(n+1) : (n+d)] = v_1$. Consider the functions,

$$\begin{aligned} h_1(x_0) &= \sqrt{\frac{n+d}{n}}\begin{bmatrix} 0 \\ g_0(x_0[(n+1) : (n+d)]) \end{bmatrix} \\ \text{and, } \breve{h}_1(x_0) &= \begin{bmatrix} \breve{g}_0(x_0[(n+1) : (n+d)]) \\ 0 \end{bmatrix}, \end{aligned}$$

then,

$$x_1 = \begin{bmatrix} Xg_0(x_0[(n+1) : (n+d)]) + \breve{g}_0(x_0[(n+1) : (n+d)]) \\ Dg_0(x_0[(n+1) : (n+d)]) \end{bmatrix}.$$

As $x_0[(n+1) : (n+d)] = v_0$, then we have

$$x_1[1 : n] = Xg_0(v_0) + \breve{g}_0(v_0) = z_0,$$

as desired. Next, given $h_2(x_0, x_1) = \sqrt{\frac{n+d}{n}} \begin{bmatrix} f_0(x_1[1:n]) \\ 0 \end{bmatrix}$ and $\breve{h}_2 = \begin{bmatrix} 0 \\ \breve{f}_0(x_0[(n+1):(n+d)]) \end{bmatrix}$, we can use that $x_1[1:n] = z_0$ and $x_0[(n+1):(n+d)] = v_0$ to show

$$x_2 = \begin{bmatrix} Cf_0(x_1[1:n]) \\ X^\top f_0(x_1[1:n]) + \breve{f}_0(x_0[(n+1):(n+d)]) \end{bmatrix} = \begin{bmatrix} Cf_0(z_0) \\ X^\top f_0(z_0) + \breve{f}_0(v_0) \end{bmatrix}.$$

Therefore, $x_2[(n+1):(n+d)] = v_1$ as desired.

Now, using the inductive hypothesis, assume that $(x_{2j}[(n+1):(n+d)])_{j\in[t]\cup\{0\}} = v_{0:t}$ and $(x_{2j-1}[1:n])_{j\in[t]} = z_{0:(t-1)}$. We now close the inductive loop by demonstrating that $x_{2t+1}[1:n] = z_t$ and $x_{2(t+1)}[(n+1):(n+d)] = v_{t+1}$.

First, given

$$h_{2t+1}(x_{0:2t}) = \sqrt{\frac{n+d}{n}} \begin{bmatrix} 0 \\ g_t((x_{2j}[(n+1):(n+d)])_{j\in[t]\cup\{0\}}) \end{bmatrix}$$

and

$$\breve{h}_{2t+1}(x_{1:2t}) = \begin{bmatrix} \breve{g}_t((x_{2j-1}[1:n])_{j\in[t]}) \\ 0 \end{bmatrix},$$

we immediately see that

$$x_{2t+1} = \begin{bmatrix} Xg_t((x_{2j}[(n+1):(n+d)])_{j\in[t]\cup\{0\}}) + \breve{g}_t((x_{2j-1}[1:n])_{j\in[t]}) \\ Dg_t((x_{2j}[(n+1):(n+d)])_{j\in[t]\cup\{0\}}) \end{bmatrix} = \begin{bmatrix} Xg_t(v_{0:t}) + \breve{g}_t(z_{0:(t-1)}) \\ Dg_t(v_{0:t}) \end{bmatrix}.$$

Therefore, $x_{2t+1}[1:n] = z_t$. Next, given

$$h_{2(t+1)}(x_{0:(2t+1)}) = \sqrt{\frac{n+d}{n}} \begin{bmatrix} f_t((x_{2j+1}[1:n])_{j\in[t]\cup\{0\}}) \\ 0 \end{bmatrix}$$

and

$$\breve{h}_{2(t+1)}(x_{0:(2t+1)}) = \begin{bmatrix} 0 \\ \breve{f}_{2(t+1)}(([(n+1):(n+d)])_{j\in[t]\cup\{0\}}) \end{bmatrix},$$

we then have that

$$x_{2(t+1)} = \begin{bmatrix} Cf_t((x_{2j+1}[1:n])_{j\in[t]\cup\{0\}}) \\ X^\top f_t((x_{2j+1}[1:n])_{j\in[t]\cup\{0\}}) + \breve{f}_{2(t+1)}(([(n+1):(n+d)])_{j\in[t]\cup\{0\}}) \end{bmatrix}.$$

Similarly plugging in the inductive claim and using that $x_{2t+1}[1:n] = z_t$ gives, $x_{2(t+1)}[(n+1):(n+d)] = v_{t+1}$, concluding the proof. ■

For simplicity in what follows, we define the functions,

$$\begin{aligned}
h'_{2t+1}(x_{0:t}) &= \sqrt{\frac{n+d}{n}} \begin{bmatrix} 0 \\ g_t((x_{2j}[(n+1):(n+d)])_{j\in[t]\cup\{0\}}) \end{bmatrix} \\
\breve{h}'_{2t+1}(x_{1:t}) &= \begin{bmatrix} \breve{g}_t((x_{2j-1}[1:n])_{j\in[t]}) \\ 0 \end{bmatrix} \\
h'_{2(t+1)}(x_{0:t+1}) &= \sqrt{\frac{n+d}{n}} \begin{bmatrix} f_t((x_{2j+1}[1:n])_{j\in[t]\cup\{0\}}) \\ 0 \end{bmatrix} \\
\breve{h}'_{2(t+1)}(x_{1:t+1}) &= \begin{bmatrix} 0 \\ \breve{f}_{2(t+1)}((x_{2j}[(n+1):(n+d)])_{j\in[t]\cup\{0\}}) \end{bmatrix}.
\end{aligned}$$

Matrix Valued Iterates: Notice that the GFOM given in (2.1) only has iterates $z_t \in \mathbb{R}^n$ and $v_{t+1} \in \mathbb{R}^d$. This is unlike our desired application where we may want $z^t \in \mathbb{R}^{n\times m}$ and $v_{t+1} \in \mathbb{R}^{n\times m}$.

This follows by considering a block-size time of $t' = t \mod (m)$ (with the understanding that $0 \mapsto m$) and we consider an indexing of time t by $(\lfloor t/m \rfloor, t')$ which then gives

$$z_{\lfloor t/m \rfloor, t'} = X g_{\lfloor t/m \rfloor, t'}(v_{1:\lfloor t/m \rfloor, 1:m}) + \breve{g}_{\lfloor t/m \rfloor, t'}(z_{1:(\lfloor t/m \rfloor - 1), 1:m})$$
$$v_{\lfloor t/m \rfloor + 1, t'} = X g_{\lfloor t/m \rfloor, t'}(z_{1:\lfloor t/m \rfloor, 1:m}) + \breve{g}_{\lfloor t/m \rfloor, t'}(v_{1:\lfloor t/m \rfloor, 1:m}).$$

Both At Once: Using Lemma 2.3, we can now output the rigorous high dimensional DMFT equations for algorithm (2.2),

$$\bar{z}_t = m^z_t + r^z_t$$
$$\bar{v}_{t+1} = m^v_{t+1} + r^v_{t+1}$$

where $m^z_t, r_{0:t} \sim \mathcal{N}(0, \Sigma_t \otimes \mathrm{Id}_n)$ and $m^v_{t+1}, r_{1:(t+1)} \sim \mathcal{N}(0, \Omega_{t+1} \otimes \mathrm{Id}_d)$ are given by the recursive construction.

$$r^z_{0:t} \sim \mathcal{N}(0, \Sigma_t \otimes I_n)$$
$$\Sigma_t[(t,i),(s,j)] = \frac{1}{n}\mathbb{E}[\langle g_t(\bar{v}_{0:t})_i, g_s(\bar{v}_{0:s})_j \rangle]$$
$$m^z_t = \breve{g}_t(\bar{z}_{0:(t-1)}) + \sum_{s=0}^{t-1} f_s(\bar{z}_{0:s}) B^\top_{t,s}$$
$$B_{t,s}[a,b] = \frac{1}{n}\mathbb{E}[\mathrm{div}^s_b g_{t-1}(\bar{v}_{0:(t-1)})_a]$$
$$r^v_{1:(t+1)} \sim \mathcal{N}(0, \Omega_{t+1} \otimes I_d)$$
$$\Omega_{t+1}[(t+1),i),(s+1),j)] = \frac{1}{n}\mathbb{E}[\langle f_t(\bar{z}_{0:t})_i, f_s(\bar{z}_{0:t})_j \rangle]$$
$$m^v_{t+1} = \breve{f}_t(\bar{v}_{0:t}) + \sum_{s=0}^{t} g_s(\bar{v}_{0:s}) A^\top_{t,s}$$
$$A_{t,s}[a,b] = \frac{1}{n}\mathbb{E}[\mathrm{div}^s_b f_t(\bar{z}_{0:t})_a].$$

3. Dynamical Mean Field Theory [Montenari, Urbani 25]

Dynamical Mean Field Theory Is Just A GFOM With A Gradient Flow Limit:

Consider the model

$$\hat{f}(x) = \frac{1}{m}\sigma(Wx)a,$$

where $[x_1^\top, \dots, x_n^\top] \in \mathbb{R}^{n \times d}$, $W \in \mathbb{R}^{m \times d}$ and $a \in \mathbb{R}^m$. We aim to understand the role of training there parameters when $n/d = \alpha$ with n, d large and m constant. This training is done according to η sized gradient descent steps on the empirical risk,

$$\mathcal{L} = \frac{1}{2n}\sum_{i=1}^{n}(f(x_i) - y_i)^2.$$

Remark 3.1. Note, this following process can be easily extended to study other more general forms of losses, included momentum and weight decay.

Step One: Writing a GFOM.

We need to write the pre-activation immediately after the linear layer and the subseqeuent training steps in terms of the following algorithm:

$$z_t = Xg_t(w_{1:t}) + \breve{g}_t(w_{1:t})$$
$$w_{t+1} = X^\top f_t(z_{1:t}) + \breve{f}_t(z_{1:t}).$$

The choice of f_t and g_t will be revealed in a moment. First, let's consider the forward pass to the preactivation. To find this value we calculate $z_t = Xw_t$ where $w_t \in \mathbb{R}^{d\times m}$ is the GFOM iterate corresponding to the t-th training step parameters.

Next, using a calculus trick, we can write $\frac{\partial \mathcal{L}}{\partial W} = X^\top \frac{\partial \mathcal{L}}{\partial z}$, where $z = Xw$. We can then calculate this derivative as

$$\frac{\partial \mathcal{L}}{\partial z_i^\alpha} = \frac{1}{n}(\hat{f}(x^\alpha) - y^\alpha)\frac{1}{m}\sigma'(x^\alpha w_i)a_i.$$

Therefore, with vectorization of α written using $\circ$ in the superscript, we have that

$$\frac{\partial \mathcal{L}}{\partial z_i^\alpha} = \frac{1}{nm}(\hat{f}^\circ - y^\circ)a_i\sigma'(z_i^\circ).$$

Thus, letting f_t be defined the row-wise seperable function, considering the parameteization $\eta = n\eta$, we have that

$$f_t(z^\alpha)_i = -\frac{\eta}{m}(\hat{f}^\alpha - y^\alpha)a_i\sigma'(z_i^\alpha),$$

and $\breve{f}_t(z_{1:t}) = w_t(z_{1:t}) = w_t$, we can then write our network training as the GFOM,

$$z_t = Xw_t$$
$$w_{t+1} = X^\top f_t(z_t) + w_t.$$

Step Two: Invoking GFOM theory.

In our case, with out choice of f_t from the previous section, we arrive at the following high-dimensional DMFT description of our dyanmics, taking form:

$$z_t \overset{\text{Law}}{=} \sum_{s=0}^{t-1} f_s(z_s)B_{t,s}^\top + r_t^z$$
$$w_{t+1} \overset{\text{Law}}{=} w_t + \sum_{s=0}^{t} w_s A_{t,s}^\top + r_t^w$$

where

$$[r_1^z, \dots, r_t^z] \sim \mathcal{N}(0, \Sigma_t \otimes I_n)$$
$$\Sigma_t[(t,i),(s,j)] = \frac{1}{n}\mathbb{E}[\langle (w_{t-1})_i, (w_s)_j\rangle]$$
$$[r_1^w, \dots, r_{t+1}^w] \sim \mathcal{N}(0, \Omega_{t+1} \otimes I_d)$$
$$\Omega_{t+1}[(t+1,i),(s+1,j)] = \frac{1}{n}\mathbb{E}[\langle f_t(z_t)_i, f_s(z_t)_j\rangle]$$
$$B_{t,s}[a,b] = \frac{1}{n}\mathbb{E}[\mathrm{div}_b^s(w_{t-1})_a]$$
$$A_{t,s}[a,b] = \frac{1}{n}\mathbb{E}[\mathrm{div}_b^s f_t(z_t)_a].$$

We can then push the dynamics of the GFOM into the final layer weights by writing

$$(a_{t+1})_i = a_{t,i} - \frac{\eta}{nm}(\hat{f}_t^\circ - y^\circ)^\top \sigma'(z_{t,i}^\circ).$$

We now want to collapse this down to a low-dimensional DMFT system, In doing so we let the following be replaced by the low-dimensional recursion:

$$\begin{aligned}
\bar{z}_t &\overset{\text{Law}}{=} \sum_{s=0}^{t-1} f_s(\bar{z}_s)\bar{B}_{t,s}^\top + \bar{r}_t^z \\
\bar{B}_{t,t-1}[a,b] &= \mathbb{E}[\mathrm{div}_b^{t-1}(\bar{w}_{t-1})_a] \\
\bar{r}_{1:t}^z &\sim \mathcal{N}(0, \bar{\Sigma}_t) \\
\bar{\Sigma}_t[(t,i),(s,j)] &= \frac{1}{n}\mathbb{E}[(\bar{w}_{t-1})_i(\bar{w}_s)_j] \\
\bar{w}_{t+1} &\overset{\text{Law}}{=} \bar{w}_t + \sum_{s=0}^{t} \bar{w}_s \bar{A}_{t,s}^\top + \bar{r}_t^w \\
\bar{A}_{t,s}[a,b] &= \mathbb{E}[\mathrm{div}_b^s f_t(\bar{z}_t)_a] \\
\bar{r}_{1:t}^w &\sim \mathcal{N}(0, \bar{\Omega}_{t+1}) \\
\bar{\Omega}_{t+1}[(t+1,i),(s+1,j)] &= \frac{1}{n}\mathbb{E}[f_t(\bar{z}_t)_i f_s(\bar{z}_s)_j] \\
(a_{t+1})_i &= a_i - \frac{\eta}{m}\mathbb{E}[(\hat{f}_t(\bar{z}_t) - y)\sigma'(\bar{z}_{t,i})].
\end{aligned}$$

And a further simplification can be made with

$$\begin{aligned}
f_t(\bar{z}_t)_i &= -\frac{\eta}{m}(\hat{f}_t(\bar{z}_t) - y)a_i\sigma'(\bar{z}_{t,i}), \\
\hat{f}(\bar{z}_t) &= \frac{1}{m}\sum_{i=1}^{m} a_i\sigma(\bar{z}_{t,i}).
\end{aligned}$$

Thus,

$$\begin{aligned}
\bar{z}_{t,i} &\overset{\text{Law}}{=} \sum_{s=0}^{t-1} -\frac{\eta}{m} R_s \sum_{\ell\in[m]} a_\ell^t \sigma'(\bar{z}_{t,\ell})\bar{B}_{t,s}[i,\ell] + (\bar{r}_t^z)_i \\
R_t &= \frac{1}{M}\sum_{i=1}^{M} a_i\sigma(\bar{z}_{t,i}) - y \\
\bar{B}_{t,s}[a,b] &= \mathbb{E}[\mathrm{div}_b^s(\bar{w}_{t-1})_a] \\
\bar{r}_{1:t}^z &\sim \mathcal{N}(0, \bar{\Sigma}_t) \\
\bar{\Sigma}_t[(t,i),(s,j)] &= \mathbb{E}[(\bar{w}_{t-1})_i(\bar{w}_s)_j] \\
\bar{w}_{t+1} &\overset{\text{Law}}{=} \bar{w}_t + \sum_{s=0}^{t} \bar{w}_s \bar{A}_{t,s}^\top + \bar{r}_t^w \\
\bar{A}_{t,s}[a,b] &= -\frac{\eta}{m}\mathbb{E}[\mathrm{div}_b^s R_t a_a \sigma'(\bar{z}_a)] \\
\bar{r}_{1:t}^w &\sim \mathcal{N}(0, \bar{\Omega}_{t+1}) \\
\bar{\Omega}_{t+1}[(t+1,i),(s+1,j)] &= \eta\frac{a_i^t a_j^s}{m^2}\mathbb{E}[R_t R_s \sigma'(\bar{z}_{t,i})\sigma'(\bar{z}_{s,j})] \\
(a_{t+1})_i &= (a_t)_i - \frac{\eta}{m}\mathbb{E}[R_t\sigma'(\bar{z}_{t,i})].
\end{aligned}$$

Now taking the limit $\eta \to 0$ we arrive at a sequence of integro-differential equation tracking the continious empirical risk gradient flow,

$$\bar{z}_{t,i} \overset{\text{Law}}{=} -\frac{1}{m}\sum_{\ell\in[m]}\int_0^t R_s a_\ell^s \sigma'(\bar{z}_{t,s})\mathbb{E}[\mathrm{div}_\ell^s(\bar{w}_{t-1})_i]\,ds + (\bar{r}_t^z)_i$$

$$(\bar{w}_{t+1})_i \overset{\text{Law}}{=} (\bar{w}_0)_i - \frac{1}{m}\sum_{\ell\in[m]}\int_0^t (\bar{w}_s)_\ell \mathbb{E}[\mathrm{div}_\ell^s R_t a_i^t \sigma'(\bar{z}_{t,i})] + \bar{r}_s^w\,ds$$

$$R_t = \frac{1}{M}\sum_{i=1}^M a_i\sigma(\bar{z}_{t,i}) - y$$

$$(a_{t+1})_i = (a_0)_i - \frac{1}{m}\int_0^t \mathbb{E}[R_s\sigma'(\bar{z}_{s,i})].$$

$$\bar{r}_{1:t}^z \sim \mathcal{N}(0,\bar{\Sigma}_t), \qquad \bar{\Sigma}_t[(t,i),(s,j)] = \mathbb{E}[(\bar{w}_{t-1})_i(\bar{w}_s)_j]$$

$$\bar{r}_{1:t}^w \sim \mathcal{N}(0,\bar{\Omega}_{t+1}), \qquad \bar{\Omega}_{t+1}[(t+1,i),(s+1,j)] = \frac{a_i^t a_j^s}{m^2}\mathbb{E}[R_t R_s \sigma'(\bar{z}_{t,i})\sigma'(\bar{z}_{s,j})].$$

The values of $\mathbb{E}[\mathrm{div}_\ell^s R_t a_i^t\sigma'(\bar{z}_{t,i})]$ and $\mathbb{E}[\mathrm{div}_\ell^s(\bar{w}_{t-1})_i]$ can be defined self consistently using causality and simple (yet tedious) derivatives or its can be simulated using Monte Carlo. An additional large n limit can also be taken to arrive at a large network limit, although this large network limit is mesoscopic comapred to n and d.

4. Where To Go From Here?

What do these algorithms have in common?

Consider the following problems:

(1) Let $W \sim \mathrm{GOE}(n)$ what is the limiting law of the spectrum of W.
(2) Given observation $A = (A_{i,j})_{i,j\in[n]}$ of the following form,
$$A_{i,j} \sim \begin{cases}\mathrm{Bern}(p_{i,j}) & \text{if } \sigma(i)=\sigma(j)\\ \mathrm{Bern}(q_{i,j}) & \text{otherwise}\end{cases}.$$
The goal is to recover $\sigma : [n] \mapsto \{-1,1\}$.
(3) Given observation $Y = |X\beta + \epsilon|$ for a random design matrix
$$X \in \mathbb{R}^{n\times d} \text{ where } X_{i,j} \sim \mathcal{N}(0,1/n).$$
The goal is to recover β.
(4) Consider Glauber Dynamics on the max cut problem,
$$H(\sigma) = \sigma^\top W\sigma,$$
where $W_{i,j} = W_{j,i} \sim \mathrm{Rad}(1/2)$. What is the limiting energy achieved by the ground state, i.e., $\max_\sigma H(\sigma)$.
(5) Consider the following neural network model
$$\hat{f}(x) = \frac{1}{n}\sum_{i\in[n]} v_i\sigma(W\ \mathrm{MHSA}(K,Q,V;x)),$$
where $W \in \mathbb{R}^{n\times n}$ is initialized at $W_0 \sim \mathcal{N}(0,1/n)$ and $(x,y) \in \mathbb{R}^{d+1} \sim \mathbb{P}_{X,Y}$ and the model is trained with ADAM on quadratic loss.

They all admit an analysis using message passing algorithms (AMP, GFOMs, DMFT, Tensor Programs).