Mathematics of Deep Learning

Infinite width limit of NNs

Lessons: Kevin Scaman

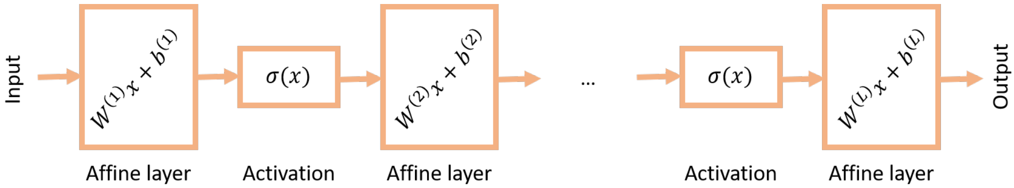
Class overview

1. Introduction and general overview 16/01
2. Non-convex optimization 23/01
3. Structure of ReLU networks and group invariances 06/02
4. Approximation guarantees 13/02
5. Stability and robustness 20/02
6. **Infinite width limit of NNs** 27/02
7. Generative models 12/03
8. Exam 19/03

Behavior at initialization in the infinite-width limit

From neural networks to Gaussian processes

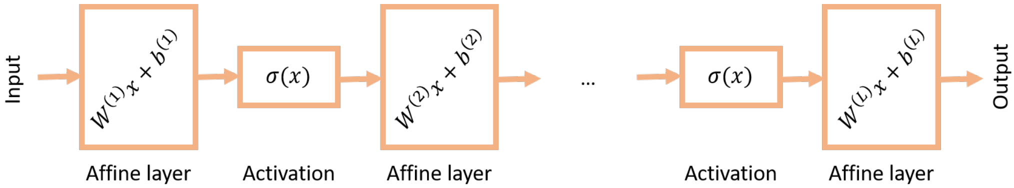
Back to weight initialization



Variance of the output and Jacobian matrix for ReLU networks

- With the notations: $X^{(l)} = g_\theta^{(2l-1)}(x)$, $Y^{(l)} = J_{g_\theta^{(2l-1)}}(x)$ and $\mathrm{var}(W_{ij}^{(l)}) = V^{(l)}$.
- We have $\mathrm{var}\left(X_i^{(l)}\right) = d^{(l-1)}\, V^{(l)}\ \mathrm{var}\left(X_j^{(l-1)}\right)/2$.
- We have $\mathrm{var}\left(Y_{ij}^{(l)}\right) = d^{(l-1)}\, V^{(l)}\ \mathrm{var}\left(Y_{kj}^{(l-1)}\right)/2$.

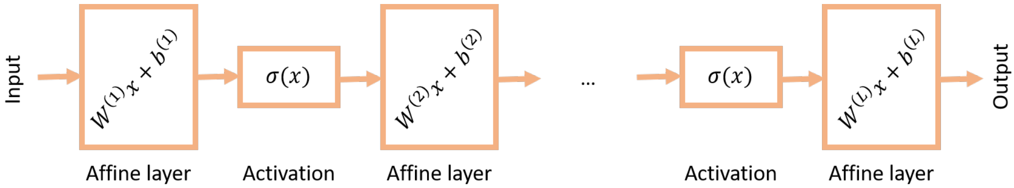
Back to weight initialization



Variance of the output and Jacobian matrix for ReLU networks

- With the notations: $X^{(l)} = g_\theta^{(2l-1)}(x)$, $Y^{(l)} = J_{g_\theta^{(2l-1)}}(x)$ and $\mathrm{var}(W_{ij}^{(l)}) = V^{(l)}$.
- We have $\mathrm{var}\left(X_i^{(l)}\right) = d^{(l-1)} V^{(l)} \ \mathrm{var}\left(X_j^{(l-1)}\right)/2$.
- We have $\mathrm{var}\left(Y_{ij}^{(l)}\right) = d^{(l-1)} V^{(l)} \ \mathrm{var}\left(Y_{kj}^{(l-1)}\right)/2$.
- What happens when the widths $d^{(l)}$ tend to $+\infty$?

Back to weight initialization



Variance of the output and Jacobian matrix for ReLU networks

- With the notations: $X^{(l)} = g_\theta^{(2l-1)}(x)$, $Y^{(l)} = J_{g_\theta^{(2l-1)}}(x)$ and $\mathrm{var}(W_{ij}^{(l)}) = V^{(l)}$.
- We have $\mathrm{var}\left(X_i^{(l)}\right) = d^{(l-1)} V^{(l)} \ \mathrm{var}\left(X_j^{(l-1)}\right)/2$.
- We have $\mathrm{var}\left(Y_{ij}^{(l)}\right) = d^{(l-1)} V^{(l)} \ \mathrm{var}\left(Y_{kj}^{(l-1)}\right)/2$.
- What happens when the widths $d^{(l)}$ tend to $+\infty$?
- Choosing $V^{(l)} = \Theta(1/d^{(l-1)})$ gives $\mathrm{var}\left(X_i^{(l)}\right) = \Theta(1)$ and $\mathrm{var}\left(Y_{ij}^{(l)}\right) = \Theta(1)$.

Infinite-width limit of neural networks

Infinite width limit

- With proper normalization of the weights $V^{(l)} = \Theta(1/d^{(l-1)})$, the variances are controlled.
- When all widths $d^{(l)} \to +\infty$, we can **totally characterize** the behavior of $X^{(l)}$ and $Y^{(l)}$.

Infinite-width limit of neural networks

Infinite width limit

- With proper normalization of the weights $V^{(l)} = \Theta(1/d^{(l-1)})$, the variances are controlled.
- When all widths $d^{(l)} \to +\infty$, we can **totally characterize** the behavior of $X^{(l)}$ and $Y^{(l)}$.

Distribution of the output

- **Assumptions:** $W_{ij}^{(l)}$ are iid, symmetric and of variance $V^{(l)} = 1/d^{(l-1)}$.
- Recall $X_i^{(l)} = \sum_j W_{ij}^{(l)} \sigma(X_j^{(l-1)})$, and $(X_i^{(l)})_{i \in [\![1, d^{(l)}]\!]}$ are ident. distr. and symmetric.

Infinite-width limit of neural networks

Infinite width limit

- With proper normalization of the weights $V^{(l)} = \Theta(1/d^{(l-1)})$, the variances are controlled.
- When all widths $d^{(l)} \to +\infty$, we can **totally characterize** the behavior of $X^{(l)}$ and $Y^{(l)}$.

Distribution of the output

- **Assumptions:** $W_{ij}^{(l)}$ are iid, symmetric and of variance $V^{(l)} = 1/d^{(l-1)}$.
- Recall $X_i^{(l)} = \sum_j W_{ij}^{(l)} \sigma(X_j^{(l-1)})$, and $(X_i^{(l)})_{i \in [\![1, d^{(l)}]\!]}$ are ident. distr. and symmetric.
- As the widths tend to infinity, the $(X_i^{(l)})_{i \in [\![1, d^{(l)}]\!]}$ become **independent**.

Infinite-width limit of neural networks

Infinite width limit

- With proper normalization of the weights $V^{(l)} = \Theta(1/d^{(l-1)})$, the variances are controlled.
- When all widths $d^{(l)} \to +\infty$, we can **totally characterize** the behavior of $X^{(l)}$ and $Y^{(l)}$.

Distribution of the output

- **Assumptions:** $W_{ij}^{(l)}$ are iid, symmetric and of variance $V^{(l)} = 1/d^{(l-1)}$.
- Recall $X_i^{(l)} = \sum_j W_{ij}^{(l)} \sigma(X_j^{(l-1)})$, and $(X_i^{(l)})_{i \in [\![1, d^{(l)}]\!]}$ are ident. distr. and symmetric.
- As the widths tend to infinity, the $(X_i^{(l)})_{i \in [\![1, d^{(l)}]\!]}$ become **independent**.
- By the CLT, $X_i^{(l)}$ converges in law to a **centrered Gaussian of variance** $\mathbb{E}(\sigma\big(X_1^{(l-1)}\big)^2)$.

Infinite-width limit of neural networks

Lemma (independence)

If $(X_i^{(l-1)})_{i\in[\![1,d^{(l-1)}]\!]}$ are independent, then $(X_i^{(l)})_{i\in[\![1,d^{(l)}]\!]}$ converge in law to independent Gaussian random variables.

Infinite-width limit of neural networks

Lemma (independence)

If $(X_i^{(l-1)})_{i\in[\![1,d^{(l-1)}]\!]}$ are independent, then $(X_i^{(l)})_{i\in[\![1,d^{(l)}]\!]}$ converge in law to independent Gaussian random variables.

Proof.

- As $W_{ij}^{(l)}\sigma(X_j^{(l-1)})$ are iid and of bounded variance, the CLT implies that $X_i^{(l)} = \sum_j W_{ij}^{(l)}\sigma(X_j^{(l-1)})$ converge in law to a Gaussian random variable.
- As $\mathbb{E}(X_i^{(l)}X_j^{(l)}) = 0$, the limits are also uncorrelated.
- Two Gaussian r.v. that are uncorrelated are necessarily independent.

□

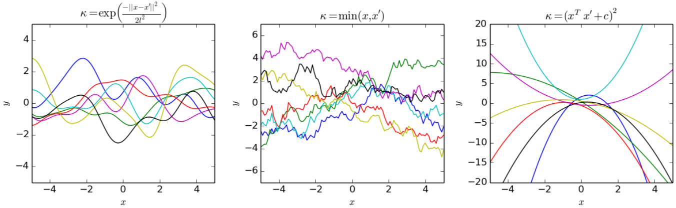
Infinite-width limit of neural networks

Definition (Gaussian process)

Gaussian process is a collection $(\xi_x)_{x \in \mathbb{R}^d}$ of random variables such that every finite collection $(\xi_{x_1}, \ldots, \xi_{x_n})$ has a multivariate Gaussian distribution.

Properties

- A Gaussian process is totally defined by its mean $\mu(x) = \mathbb{E}(\xi_x)$ and *covariance kernel* $\Sigma(x, y) = \mathrm{cov}(\xi_x, \xi_y)$ for $x, y \in \mathbb{R}^d$.
- The kernel controls the regularity of the function.



Infinite-width limit of neural networks

Theorem (Neal, 1994 ; Daniely et.al., 2016)

When the widths $d^{(l)}$ tend to infinity, the intermediate outputs $g_\theta^{(2l-1)}(x)_i$ converge in law to iid centered Gaussian processes of kernel $\Sigma^{(L)}$ where

$$\Sigma^{(1)}(x,y) = \frac{1}{d^{(0)}} x^\top y$$
$$\Sigma^{(l+1)}(x,y) = \mathbb{E}_{\xi \sim \mathcal{N}(0,\Sigma^{(l)})}(\sigma(\xi_x)\sigma(\xi_y))$$

where $\mathcal{N}(0,\Sigma^{(l)})$ is a Gaussian process of covariance $\Sigma^{(l)}$.

- Each coordinate of the output $g_\theta(x)_i$ is thus a **centered Gaussian process of covariance** $\Sigma^{(L)}$.

Convergence of the Jacobian matrix

- A similar result holds for the Jacobians $J_{g,x}(x, \theta)$ and $J_{g,\theta}(x, \theta)$.
- Coordinates of the Jacobian converge to **centered Gaussian random variables**.

Convergence of the Jacobian matrix

- A similar result holds for the Jacobians $J_{g,x}(x, \theta)$ and $J_{g,\theta}(x, \theta)$.
- Coordinates of the Jacobian converge to **centered Gaussian random variables**.
- However, for $J_{g,\theta}(x, \theta) \in \mathbb{R}^{d^{(L)} \times p}$, the number of coordinates also tends to infinity, and this is not well suited to describe the behavior of the whole Jacobian.

Convergence of the Jacobian matrix

- A similar result holds for the Jacobians $J_{g,x}(x, \theta)$ and $J_{g,\theta}(x, \theta)$.
- Coordinates of the Jacobian converge to **centered Gaussian random variables**.
- However, for $J_{g,\theta}(x, \theta) \in \mathbb{R}^{d^{(L)} \times p}$, the number of coordinates also tends to infinity, and this is not well suited to describe the behavior of the whole Jacobian.
- Instead, we consider the convergence of the *Neural Tangent Kernel*, that will capture the **impact of gradient descent on the output value**.

Convergence of the Jacobian matrix

- A similar result holds for the Jacobians $J_{g,x}(x,\theta)$ and $J_{g,\theta}(x,\theta)$.
- Coordinates of the Jacobian converge to **centered Gaussian random variables**.
- However, for $J_{g,\theta}(x,\theta) \in \mathbb{R}^{d^{(L)} \times p}$, the number of coordinates also tends to infinity, and this is not well suited to describe the behavior of the whole Jacobian.
- Instead, we consider the convergence of the *Neural Tangent Kernel*, that will capture the **impact of gradient descent on the output value**.

Definition (NTK)

The *Neural Tangent Kernel* of a model g_θ is the function $\kappa^{\text{NTK}}_{g,\theta} : \mathbb{R}^{d^{(0)}} \times \mathbb{R}^{d^{(0)}} \to \mathbb{R}^{d^{(L)} \times d^{(L)}}$:

$$\kappa^{\text{NTK}}_{g,\theta}(x,y) = J_{g,\theta}(x,\theta) \times J_{g,\theta}(y,\theta)^\top$$

NTK and gradient descent

- If we make a stochastic gradient step for the objective $\frac{1}{N}\sum_i \ell(g_\theta(x_i), y_i)$, then, as a first order approximation, we have

$$g_{\theta_{t+1}}(x) \quad \approx \quad g_{\theta_t}(x) + J_{g,\theta}(x, \theta_t)(\theta_{t+1} - \theta_t)$$

NTK and gradient descent

- If we make a stochastic gradient step for the objective $\frac{1}{N}\sum_i \ell(g_\theta(x_i), y_i)$, then, as a first order approximation, we have

$$\begin{aligned} g_{\theta_{t+1}}(x) &\approx g_{\theta_t}(x) + J_{g,\theta}(x, \theta_t)(\theta_{t+1} - \theta_t) \\ &= g_{\theta_t}(x) - \eta\, J_{g,\theta}(x, \theta_t) \times J_{g,\theta}(x_t, \theta_t)^\top \times \nabla_x \ell(g_{\theta_t}(x_t), y_t) \end{aligned}$$

NTK and gradient descent

- If we make a stochastic gradient step for the objective $\frac{1}{N}\sum_i \ell(g_\theta(x_i), y_i)$, then, as a first order approximation, we have

$$\begin{aligned} g_{\theta_{t+1}}(x) &\approx g_{\theta_t}(x) + J_{g,\theta}(x,\theta_t)(\theta_{t+1}-\theta_t) \\ &= g_{\theta_t}(x) - \eta\, J_{g,\theta}(x,\theta_t) \times J_{g,\theta}(x_t,\theta_t)^\top \times \nabla_x \ell(g_{\theta_t}(x_t), y_t) \\ &= g_{\theta_t}(x) - \eta\, \kappa^{\mathsf{NTK}}_{g,\theta_t}(x, x_t) \times \nabla_x \ell(g_{\theta_t}(x_t), y_t) \end{aligned}$$

NTK and gradient descent

- If we make a stochastic gradient step for the objective $\frac{1}{N}\sum_i \ell(g_\theta(x_i), y_i)$, then, as a first order approximation, we have

$$
\begin{aligned}
g_{\theta_{t+1}}(x) &\approx g_{\theta_t}(x) + J_{g,\theta}(x,\theta_t)(\theta_{t+1}-\theta_t) \\
&= g_{\theta_t}(x) - \eta\, J_{g,\theta}(x,\theta_t) \times J_{g,\theta}(x_t,\theta_t)^\top \times \nabla_x \ell(g_{\theta_t}(x_t), y_t) \\
&= g_{\theta_t}(x) - \eta\, \kappa^{\mathsf{NTK}}_{g,\theta_t}(x, x_t) \times \nabla_x \ell(g_{\theta_t}(x_t), y_t)
\end{aligned}
$$

- This behaves as if we added the function $x \mapsto \kappa^{\mathsf{NTK}}_{g,\theta_t}(x, x_t)$ weighted depending on the gradient of the loss $\nabla_x \ell(g_{\theta_t}(x_t), y_t)$ at the current data point.

Convergence of the NTK

Theorem (Jacot et. al., 2018)

If the activation function σ is Lipschitz, as the widths $d^{(l)}$ tend to $+\infty$, the NTK at initialization $\kappa_{g,\theta_0}^{\mathsf{NTK}}$ converges in probability to a deterministic limiting kernel

$$\kappa_{g,\theta_0}^{\mathsf{NTK}}(x,y) \to \kappa_\infty^{(L)}(x,y) \otimes \mathsf{Id}_{d^{(L)}}$$

where the scalar kernel $\kappa_\infty^{(L)} : \mathbb{R}^{d^{(0)}} \times \mathbb{R}^{d^{(0)}} \to \mathbb{R}$ is defined by

$$\begin{aligned} \kappa_\infty^{(1)}(x,y) &= \Sigma^{(1)}(x,y) \\ \kappa_\infty^{(l+1)}(x,y) &= \kappa_\infty^{(l)}(x,y) \times \dot{\Sigma}^{(l+1)}(x,y) + \Sigma^{(l+1)}(x,y) \end{aligned}$$

where $\dot{\Sigma}^{(l+1)}(x,y) = \mathbb{E}_{\xi\sim\mathcal{N}(0,\Sigma^{(l)})}(\sigma'(\xi_x)\sigma'(\xi_y))$.

Behavior around initialization in the infinite-width limit

Spectrum of the Hessian and linear approximation

Quality of the linear approximation

- What happens when $\theta \neq \theta_0$?

Quality of the linear approximation

- What happens when $\theta \neq \theta_0$?
- If $\|\theta - \theta_0\|_2 \leqslant R$ where $R > 0$ is small, then

$$g_\theta(x) \approx g_{\theta_0}(x) + J_{g,\theta}(x, \theta_0)(\theta - \theta_0)$$

and the behavior is also **Gaussian**.

Quality of the linear approximation

- What happens when $\theta \neq \theta_0$?
- If $\|\theta - \theta_0\|_2 \leqslant R$ where $R > 0$ is small, then

$$g_\theta(x) \approx g_{\theta_0}(x) + J_{g,\theta}(x, \theta_0)(\theta - \theta_0)$$

  and the behavior is also **Gaussian**.
- How far can we go around θ_0?

Quality of the linear approximation

- What happens when $\theta \neq \theta_0$?
- If $\|\theta - \theta_0\|_2 \leqslant R$ where $R > 0$ is small, then

$$g_\theta(x) \approx g_{\theta_0}(x) + J_{g,\theta}(x, \theta_0)(\theta - \theta_0)$$

  and the behavior is also **Gaussian**.
- How far can we go around θ_0?
- Using Taylor-Lagrange inequality, we can control the quality of a first-order approximation by the **spectral norm of the Hessian**:

$$\|g_\theta(x) - g_{\theta_0}(x) - J_{g,\theta}(x, \theta_0)(\theta - \theta_0)\|_2 \leqslant \frac{\max_{\theta' \in \mathcal{B}(\theta_0, R)} \lambda_{\max}\left(H_{g_{\theta'}}(x)\right) R^2}{2}$$

Bound on the spectral norm of the Hessian

Theorem (Daniely et.al., 2016 ; Lee et.al., 2019 ; Liu et. al., 2020)

Let $d^{(1)} = ... = d^{(L-1)} = d$. Given any fixed $R > 0$ and any $\theta \in \mathcal{B}(\theta_0, R)$, with high probability, we have

$$\lambda_{\max}\left(H_{g_\theta}(x)\right) = \tilde{O}\left(\frac{1}{\sqrt{d}}\right)$$

- As a consequence, we have

$$g_\theta(x) \approx g_{\theta_0}(x) + J_{g,\theta}(x, \theta_0)(\theta - \theta_0) + \tilde{O}\left(\frac{1}{\sqrt{d}}\right)$$

- In the infinite-width limit, the **neural network is linear** w.r.t. θ!

Behavior during training in the infinite-width limit

Gaussian process + NTK = trained neural network

Behavior during training

Short recap

- We know the behavior of the output value and Jacobian at initialization.

Behavior during training

Short recap

- We know the behavior of the output value and Jacobian at initialization.
- We know that the model is linear with respect to the parameters.

Behavior during training

Short recap

- We know the behavior of the output value and Jacobian at initialization.
- We know that the model is linear with respect to the parameters.
- We can describe what happens after t iterations of SGD.

Behavior during training

Short recap

- We know the behavior of the output value and Jacobian at initialization.
- We know that the model is linear with respect to the parameters.
- We can describe what happens after t iterations of SGD.

Impact of SGD on the output value

- With $v_t = \nabla_x \ell(g_{\theta_t}(x_t), y_t)$, we have $\theta_{t+1} = \theta_t - \eta J_{g,\theta}(x_t, \theta_t) v_t$ and

$$g_{\theta_T}(x) = g_{\theta_0}(x) - \eta \sum_{t=1}^{T-1} \kappa_{g,\theta_0}^{\text{NTK}}(x, x_t) v_t + \tilde{O}\left(\frac{1}{\sqrt{d}}\right)$$

Random Gaussian process — Deterministic NTK kernel — Negligible second-order

Behavior during training

Discussion

- The same analysis was extended to other neural network architectures such as CNNs, RNNs and GNNs.
- In the infinite-width limit, the NTK gives the impact of a data point on the trained model.
- Moreover, the model is linear, so the objective function is convex... and **optimization is simple**.
- For real architectures though, more work is needed to assess if the widths are sufficiently large, i.e. if the model is sufficiently **over-parameterized**.

Over-parameterized neural networks

When are the widths *nearly infinite*?

Over-parameterized neural networks

Lazy training (Chizat et.al., 2019)

- At each step of SGD, we want a significant drop in the loss:

$$\frac{\mathcal{L}(\theta_{t+1}) - \mathcal{L}(\theta_t)}{\mathcal{L}(\theta_t)} \approx \frac{\eta \|\nabla \mathcal{L}(\theta_t)\|^2}{\mathcal{L}(\theta_t)} \qquad \text{“not negligible”}$$

- At the same time, we want the Jacobian of the model to be almost constant:

$$\frac{\|J_{g,\theta}(x, \theta_{t+1}) - J_{g,\theta}(x, \theta_t)\|}{\|J_{g,\theta}(x, \theta_t)\|} \approx \frac{\eta \|\nabla \mathcal{L}(\theta_t)\| \|H_{g_{\theta_t}}\|}{\|J_{g,\theta}(x, \theta_t)\|} \qquad \text{“negligible”}$$

- For the MSE loss, we thus want the following ratio to be small around initialization:

$$\kappa_{g_\theta} = \frac{\mathcal{L}(\theta)\|H_{g_\theta}\|}{\|\nabla \mathcal{L}(\theta)\| \|J_{g,\theta}(x, \theta)\|} = \frac{\|g_\theta - y^*\| \|H_{g_\theta}\|}{\|J_{g,\theta}(x, \theta)\|} \ll 1$$

Back to the PL condition

Theorem (PL condition for MSE loss)

Let $\mathcal{L}(\theta) = \frac{1}{N}\sum_{i=1}^N \ell(g_\theta(x_i), y_i)$ where $\ell(y, y') = \|y - y'\|_2^2$ and the model g_θ is such that

$$\sigma_{\min}\left(\left(J_{g,\theta}(x_1, \theta)^\top \;\middle|\; \cdots \;\middle|\; J_{g,\theta}(x_N, \theta)^\top\right)\right) \geqslant \varepsilon$$

then f verifies the μ-PL condition with $\mu = 4\varepsilon^2/N$.

Back to the PL condition

Theorem (PL condition for MSE loss)

Let $\mathcal{L}(\theta) = \frac{1}{N}\sum_{i=1}^{N} \ell(g_\theta(x_i), y_i)$ where $\ell(y, y') = \|y - y'\|_2^2$ and the model g_θ is such that

$$\sigma_{\min}\left(\left(J_{g,\theta}(x_1,\theta)^\top \;\middle|\; \cdots \;\middle|\; J_{g,\theta}(x_N,\theta)^\top\right)\right) \geqslant \varepsilon$$

then f verifies the μ-PL condition with $\mu = 4\varepsilon^2/N$.

Theorem (convergence of SGD with PL)

If $\mathcal{L}$ is β-smooth and verifies the PL condition, then, with $\eta \leqslant \frac{1}{\beta}$, SGD achieves the precision

$$\mathbb{E}(\mathcal{L}(\theta_T) - \mathcal{L}(\theta^\star)) \leqslant \Delta e^{-\mu\eta T/2} + \frac{\beta\eta\sigma^2}{\mu}$$

Exponential convergence rate $O(e^{-T})$ without noise, and $O(\ln(T)/T)$ otherwise.

Back to the PL condition

With the NTK

- The bound on the singular values of the Jacobian is equivalent to a bound on the eigenvalues of the NTK:

$$\lambda_{\min}\left(\left(\kappa_{g,\theta}^{\mathsf{NTK}}(x_i, x_j)\right)_{i,j\in[\![1,N]\!]}\right) \geqslant \varepsilon$$

Back to the PL condition

With the NTK

- The bound on the singular values of the Jacobian is equivalent to a bound on the eigenvalues of the NTK:

$$\lambda_{\min}\left(\left(\kappa_{g,\theta}^{\mathsf{NTK}}(x_i, x_j)\right)_{i,j\in[\![1,N]\!]}\right) \geqslant \varepsilon$$

- Moreover, as the Hessian controls the variation of the Jacobian, we have, for $\theta \in \mathcal{B}(\theta_0, R)$,

$$\lambda_{\min}\left(\left(\kappa_{g,\theta}^{\mathsf{NTK}}(x_i, x_j)\right)_{i,j\in[\![1,N]\!]}\right) \geqslant \lambda_{\min}\left(\left(\kappa_{g,\theta_0}^{\mathsf{NTK}}(x_i, x_j)\right)_{i,j\in[\![1,N]\!]}\right) - \widetilde{O}(NR/\sqrt{d})$$

Recap

- For infinite-width neural networks:
  - At initialization, the output is a **centralized Gaussian process**.
  - The spectral norm of the Hessian is negligible, and the model is **linear w.r.t. its parameters**.
  - The Neural Tangent Kernel (NTK) converges to a **deterministic kernel**
  - The output of the model during SGD training is fully characterized by the model's associated Gaussian process and NTK.
- For real neural networks, a ratio between the eigenvalues of the Hessian and Jacobian assess the *linearity* of the model.
- This ratio being small, the objective verifies the PL condition and **training converges to zero loss**.