Self Study

Based on my Understanding of the Subject

Geometric Deep Learning

Compilation of my study materials

Author: Dr. Md Arafat Hossain Khan

Last updated : January 10, 2023

Contents

Chapter 1

Fundamentals of GNN

Artificial intelligence is a tool, not a threat.Artificial intelligence is a tool, not a threat.Artificial intelligence is a tool, not a threat.

Rodney Brooks

1.1 Graph Neural Network Model Mathematics

This note is fundametally inspired by the original paper of graph neural network [1] and based on my undertsanding of the subject.

1.1.1 Labels of a graph

A graph G is a pair (N, E), where $N = \{n_1, n_2, \ldots, n_{|N|-1}, n_{|N|}\}$ is the set of nodes and $E = \{e_1, e_2, \ldots, e_{|E|-1}, e_{|E|}\}$ is the set of edges. $\mathcal{N}(n)$ stands for the neighbors of $n \in N$. If an *undirected* edge e connects node x and y, we can also denote e as (x, y) or (y, x). All nodes are embedded into some euclidean space of dimention $d_N \in \mathbb{N}$ and all edges are embedded into some euclidean space of dimention $d_E \in \mathbb{N}$, e.g.

$$\text{label (embedding) for node } n \text{ is defined as } \ \ell_n : n \mapsto \mathbb{R}^{d_N}, \quad \text{where } n \in N$$
$$\text{label (embedding) for edge } e \text{ is defined as } \ \ell_e : e \mapsto \mathbb{R}^{d_E}, \quad \text{where } e \in E$$

The notion of labels can be extended to multiple nodes and edges. Suppose, we take a node set, $\mathfrak{N} = \{\mathfrak{n}_1, \mathfrak{n}_2, \ldots, \mathfrak{n}_{|\mathfrak{N}|-1}, \mathfrak{n}_{|\mathfrak{N}|}\} \subseteq N$. Then we define,

$$\ell_{\mathfrak{N}} : \mathbb{R}^{d_N \times |\mathfrak{N}|} \mapsto \mathbb{R}^{d_N} \ \text{ as } \ \left(\ell_{\mathfrak{n}_1}, \ell_{\mathfrak{n}_2}, \ldots, \ell_{\mathfrak{n}_{|\mathfrak{N}|-1}}, \ell_{\mathfrak{n}_{|\mathfrak{N}|}}\right) \mapsto \mathbb{R}^{d_N}$$

Here, we want $\ell_{\mathfrak{N}}$ to be permutation invariant. Similarly, we take an edge set, $\mathfrak{E} = \{\mathfrak{e}_1, \mathfrak{e}_2, \ldots, \mathfrak{e}_{|\mathfrak{E}|-1}, \mathfrak{e}_{|\mathfrak{E}|}\} \subseteq E$. Then we define,

$$\ell_{\mathfrak{E}} : \mathbb{R}^{d_E \times |\mathfrak{E}|} \mapsto \mathbb{R}^{d_E} \ \text{ as } \ \left(\ell_{\mathfrak{e}_1}, \ell_{\mathfrak{e}_2}, \ldots, \ell_{\mathfrak{e}_{|\mathfrak{E}|-1}}, \ell_{\mathfrak{e}_{|\mathfrak{E}|}}\right) \mapsto \mathbb{R}^{d_E}$$

Here, we want $\ell_{\mathfrak{E}}$ to be permutation invariant. We extend the notion of the labels even further for the entire graph by the following permutation invariant map,

$$\ell : \mathbb{R}^{d_N \times |N|} \times \mathbb{R}^{d_E \times |E|} \mapsto \mathbb{R}^D \ \text{ as } \ \left(\ell_{n_1}, \ell_{n_2}, \ldots, \ell_{|N|-1}, \ell_{|N|}, \ell_{e_1}, \ell_{e_2}, \ldots, \ell_{|E|-1}, \ell_{|E|}\right) \mapsto \mathbb{R}^D$$

where $D \in \mathbb{N}$.

1.1.2 Local transition and output functions

Let us take $n \in N$ from the graph $G = (N, E)$ and define the following,

- $\ell_n \in \mathbb{R}^{d_N}$: node label of $n \in N$.
- $\ell_{\mathcal{N}(n)} \in \mathbb{R}^{d_N}$: label of the neighborhood $\mathcal{N}(n) = \{\nu_1, \nu_2, \ldots, \nu_{|\mathcal{N}(n)|-1}, \nu_{|\mathcal{N}(n)|}\} \subseteq N$.
- $\mathcal{E}(n)$: edges connected to n i.e $\mathcal{E}(n) = \{(\nu_1, n), (\nu_2, n), \ldots, (\nu_{|\mathcal{N}(n)|-1}, n), (\nu_{|\mathcal{N}(n)|}, n)\}$.
- $\ell_{\mathcal{E}(n)}$: edge label of $\mathcal{E}(n)$.

Now we can define something called the *state* of n, denoted as $x_n \in \mathbb{R}^s$, which captures the the local information of n using its own node label $\ell_n \in \mathbb{R}^{d_N}$, node label of the neighbors $\ell_{\mathcal{N}(n)} \in \mathbb{R}^{d_N}$, connected edges lalel $\ell_{\mathcal{E}(n)}$, and the *state* of the neighbors $x_{\mathcal{N}(n)} \in \mathbb{R}^s$. Note that, $x_{\mathcal{N}(n)}$ is a permutation invariant map, defined by

$$x_{\mathcal{N}(n)} : \mathbb{R}^{s \times |\mathcal{N}(n)|} \mapsto \mathbb{R}^s \ \text{ as } \ \left(x_{\nu_1}, x_{\nu_2}, \ldots, x_{\nu_{|\mathcal{N}(n)|-1}}, x_{\nu_{|\mathcal{N}(n)|}}\right) \mapsto \mathbb{R}^s.$$

Then we define the the *local transition function*, that expresses the dependence of a node on its neighborhood as

$$f_w : \mathbb{R}^{d_N} \times \mathbb{R}^{d_E} \times \mathbb{R}^s \times \mathbb{R}^{d_N} \mapsto \mathbb{R}^s \quad \text{as} \quad \left(\ell_n, \ell_{\mathcal{E}(n)}, x_{\mathcal{N}(n)}, \ell_{\mathcal{N}(n)}\right) \mapsto \mathbb{R}^s.$$

$$\begin{aligned} \text{where} \quad & \left(\ell_{(\nu_1,n)}, \ell_{(\nu_2,n)}, \ldots, \ell_{(\nu_{|\mathcal{N}(n)|-1},n)}, \ell_{(\nu_{|\mathcal{N}(n)|},n)}\right) := \ell_{\mathcal{E}(n)} \\ & \left(x_{\nu_1}, x_{\nu_2}, \ldots, x_{\nu_{|\mathcal{N}(n)|-1}}, x_{\nu_{|\mathcal{N}(n)|}}\right) := x_{\mathcal{N}(n)} \\ & \left(\ell_{\nu_1}, \ell_{\nu_2}, \ldots, \ell_{\nu_{|\mathcal{N}(n)|-1}}, \ell_{\nu_{|\mathcal{N}(n)|}}\right) := \ell_{\mathcal{N}(n)} \end{aligned}$$

And we also express *the local output function* that describes how the output is produced as follows,

$$g_w : \mathbb{R}^s \times \mathbb{R}^{d_N} \mapsto \mathbb{R}^r \quad \text{as} \quad (x_n, \ell_n) \mapsto \mathbb{R}^r$$

Hence, we have the following model,

$$\begin{aligned} \textit{local transition function}, \quad & x_n = f_w\left(\ell_n, \ell_{\mathcal{E}(n)}, x_{\mathcal{N}(n)}, \ell_{\mathcal{N}(n)}\right) \\ \textit{local output function}, \quad & o_n = g_w(x_n, \ell_n) \end{aligned}$$

1.1.3 Model Variation

Minimal model

Definition 1: Minimal Model

A model is said to be minimal if it has the smallest number of variables while retaining the same computational power.

Note that, one may wish to remove the labels $\ell_{\mathcal{N}(n)}$, since they include information that is implicitly contained in $x_{\mathcal{N}(n)}$. Thus one can hunt for a minimal model. It can be shown that the minimal model exists but not unique.

Extension of neighborhood

The neighborhood could contain nodes that are two or more links away from n capturing a bigger local region.

Directed and mixed graph

For directed or mixed graph we may find a variation of model definition by incorporating the following map into the model definition,

$$\mathfrak{d}_{\mathcal{E}(n)} : \mathcal{E}(n) \mapsto \{0, 1\} \quad \text{as}$$

$$\mathfrak{d}_{\mathcal{E}(n)}(e) = \mathfrak{d}_e = \begin{cases} 1 & \text{if } e \in \mathcal{E}(n) \text{ directs towards } n \\ 0 & \text{if } e \in \mathcal{E}(n) \text{ comes out of } n \end{cases}$$

Thus the definition of *local transition function* changes as follows,

$$f_w : \mathbb{R}^{d_N} \times \mathbb{R}^{d_E} \times \mathbb{R}^s \times \mathbb{R}^{d_N} \times \{0,1\}^{|\mathcal{N}(n)|} \mapsto \mathbb{R}^s \quad \text{as} \quad \left(\ell_n, \ell_{\mathcal{E}(n)}, x_{\mathcal{N}(n)}, \ell_{\mathcal{N}(n)}, \mathfrak{d}_{\mathcal{E}(n)}\right) \mapsto \mathbb{R}^s.$$

where

$$\left(\ell_{(\nu_1,n)}, \ell_{(\nu_2,n)}, \ldots, \ell_{(\nu_{|\mathcal{N}(n)|-1},n)}, \ell_{(\nu_{|\mathcal{N}(n)|},n)}\right) := \ell_{\mathcal{E}(n)}$$

$$\left(x_{\nu_1}, x_{\nu_2}, \ldots, x_{\nu_{|\mathcal{N}(n)|-1}}, x_{\nu_{|\mathcal{N}(n)|}}\right) := x_{\mathcal{N}(n)}$$

$$\left(\ell_{\nu_1}, \ell_{\nu_2}, \ldots, \ell_{\nu_{|\mathcal{N}(n)|-1}}, \ell_{\nu_{|\mathcal{N}(n)|}}\right) := \ell_{\mathcal{N}(n)}$$

$$\left(\mathfrak{d}_{(\nu_1,n)}, \mathfrak{d}_{(\nu_2,n)}, \ldots, \mathfrak{d}_{(\nu_{|\mathcal{N}(n)|-1},n)}, \mathfrak{d}_{(\nu_{|\mathcal{N}(n)|},n)}\right) := \mathfrak{d}_{\mathcal{E}(n)}$$

However, unless explicitly stated, all the results proposed in this paper hold also for directed graphs and for graphs with mixed directed and undirected links.

Node specific local transition and output functions

Moreover, each node can have its own *local transition function* and *the local output function* definition,

$$\textit{local transition function}, \quad x_n = f_{w_n}^{(n)}\left(\ell_n, \ell_{\mathcal{E}(n)}, x_{\mathcal{N}(n)}, \ell_{\mathcal{N}(n)}\right)$$

$$\textit{local output function}, \quad o_n = g_{w_n}^{(n)}(x_n, \ell_n)$$

However, for the sake of simplicity, our analysis will consider a particular model where all the nodes share the same implementation.

Positional and nonpositional graphs

Nonpositional graphs are those described so far. Positional graphs differ since a unique integer identifier is assigned to each neighbors of a node to indicate its logical position.

Definition 2: Positional graphs

If for a graph $G = (N, E)$ and for each $n \in N$, there exists,

$$\rho_n : \mathcal{N}(n) \hookrightarrow [|N|]$$

then G is called a positional graph. Note that the symbol $\hookrightarrow$ means an injection and $[|N|]$ means the set $\{1, 2, \ldots, |N|\}$.

For $n \in N$,

$$\rho_n(u) = i \implies u \text{ is the } i\text{-th neighbor of } n.$$

An example of this assignment can be such that ρ_n might enumerate the neighbors of a node following a clockwise ordering convention. Notice that for nodes n_j and n_k we may have $\rho_{n_j} = \rho_{n_k}$ which implies that this is only a relative positional assignment. Note that, for a complete graph, ρ_n is a bijection for all n, on the other hand for a path graph with at least 3 nodes ρ_n is an injection not a surjection for any n.

Now, f_w will take this positional information. Here is how it is done. Suppose, for a graph $G = (N, E)$,

$$M := \max_{n,u} \rho_n(u)$$

Then $\forall n \in N$, we make $x_{\mathcal{N}(n)} = (y_1, y_2, \ldots, y_M)$ where,

$$y_i = \begin{cases} x_u & \text{if } \rho_n(u) = i \implies u \text{ is the } i\text{-th neighbor of } n \\ x_0 & \text{if there is no } i\text{-th neighbor of } n \end{cases}$$

Here x_0 is some predefined null state. In the same way $\ell_{\mathcal{E}(n)}$ and $\ell_{\mathcal{N}(n)}$ are modified as well.

Positional and non-positional form

Note that in general,

$$\textit{local transition function, } \; x_n = f_w\left(\ell_n, \ell_{\mathcal{E}(n)}, x_{\mathcal{N}(n)}, \ell_{\mathcal{N}(n)}\right)$$

For non-positional graph we assume a significant Simplification of the mdoel which turned out to be useful,

$$\textit{local transition function, } \; x_n = \sum_{\nu_i \in \mathcal{N}(n)} h_w\left(\ell_n, \ell_{(\nu_i, n)}, x_{\nu_i}, \ell_{\nu_i}\right)$$

These two representations are called *positional form* and *non-positional form* respectively.

1.1.4 Global transition and output functions

Note that we have *local transition functions* for each node of the graph $G = (N, E)$, where $N = \{n_1, n_2, \ldots, n_{|N|-1}, n_{|N|}\}$ is the set of nodes and $E = \{e_1, e_2, \ldots, e_{|E|-1}, e_{|E|}\}$ is the set of edges.

$$\begin{aligned} x_{n_1} &= f_w\left(\ell_{n_1}, \ell_{\mathcal{E}(n_1)}, x_{\mathcal{N}(n_1)}, \ell_{\mathcal{N}(n_1)}\right) \\ x_{n_2} &= f_w\left(\ell_{n_2}, \ell_{\mathcal{E}(n_2)}, x_{\mathcal{N}(n_2)}, \ell_{\mathcal{N}(n_2)}\right) \\ &\vdots \\ x_{n_{|N|}} &= f_w\left(\ell_{n_{|N|}}, \ell_{\mathcal{E}(n_{|N|})}, x_{\mathcal{N}(n_{|N|})}, \ell_{\mathcal{N}(n_{|N|})}\right) \end{aligned}$$

We define a *global transition function*, F_w that takes the graph G as input and returns the state x_n for each $n \in N$. We represent the system of above equations in the following compact fashion,

$$x = F_w(x, \ell)$$

Now, observe the *local output functions*,

$$\begin{aligned} o_{n_1} &= g_w(x_{n_1}, \ell_{n_1}) \\ o_{n_2} &= g_w(x_{n_2}, \ell_{n_2}) \\ &\vdots \\ o_{n_{|N|}} &= g_w(x_{n_{|N|}}, \ell_{n_{|N|}}) \end{aligned}$$

We define a *global output function*, G_w that takes the graph G as input and returns the output o_n for each $n \in N$. We represent the system of above equations in the following compact fashion,

$$o = F_w(x, \ell_N)$$

At this point we have to make sure that such *global transition function* and *global output function* ensures the existence and uniqueness of the solution. Moreover, we also need to figure out a method of computation for the solution.

Existence and uniqueness of global transition and output functions

Let us observe that under standard topology every euclidean space is a complete metric space. Thus if we can ensure that F_w is a contraction mapping with respect to the states then using *contraction mapping theorem*, Appendix A, we conclude that that there exists unique solution to the states of the nodes. We will enforce the contraction mapping property by an appropriate implementation of the transition function.

1.2 Graph Neural Network Model Implementation

In order to implement the GNN model, the following items must be provided:

1. Computation of the State. That means a method to solve,

$$\begin{aligned} \textit{local transition function}, \quad x_n &= f_w\left(\ell_n, \ell_{\mathcal{E}(n)}, x_{\mathcal{N}(n)}, \ell_{\mathcal{N}(n)}\right) \\ \textit{local output function}, \quad o_n &= g_w(x_n, \ell_n) \end{aligned}$$

2. The Learning Algorithm. That means a learning algorithm to adapt f_w and g_w using examples from the training data set.

3. Transition and Output Function Implementations. That means an implementation of f_w and g_w.

1.2.1 Computation of the State

1.2.1.1 Basic idea of computation

Contraction mapping theorem, Theorem A.1, does not only ensure the existence and the uniqueness of the solution of $x = F_w(x, \ell)$ but it also suggests the following classic iterative scheme for computing the state:

$$x(t+1) = F_w(x(t), \ell).$$

We see the dynamical system converges exponentially fast to the solution of

$$\begin{aligned} x &= F_w(x, \ell) \\ o &= G_w(x, \ell_N) \end{aligned}$$

for any initial value $x(0)$, Theorem A.1. We can, therefore, think of $x(t)$ as the state that is updated by the transition function F_w. In fact, $x(t+1) = F_w(x(t), \ell)$ implements the Jacobi iterative method for solving nonlinear equations [2]. Thus, the outputs and the states can be computed by iterating,

$$\begin{aligned} x_n(t+1) &= f_w\left(\ell_n, \ell_{\mathcal{E}(n)}, x_{\mathcal{N}(n)}(t), \ell_{\mathcal{N}(n)}\right) \\ o_n(t) &= g_w(x_n(t), \ell_n) \end{aligned}$$

1.2.1.2 Example with a graph

We already discussed the variations for directed, and mixed graph. Thus without loss of generality of discussion, we can consider an undirected graph as follows,

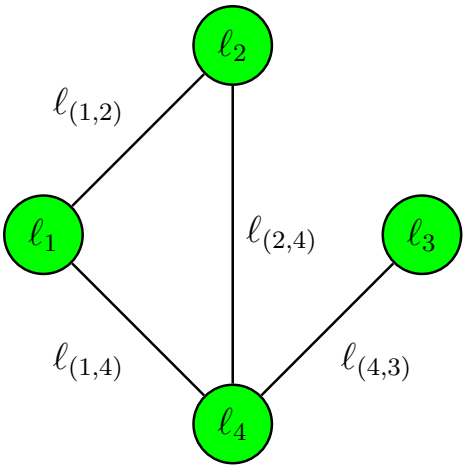
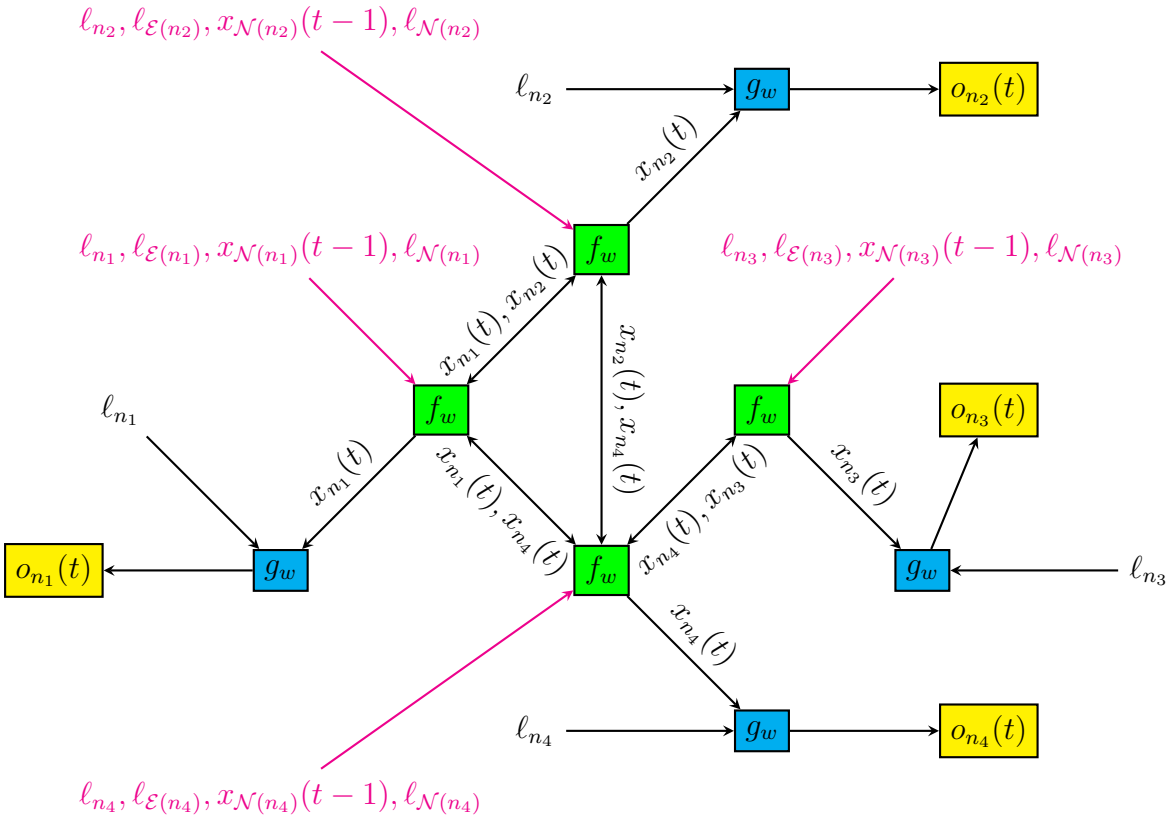
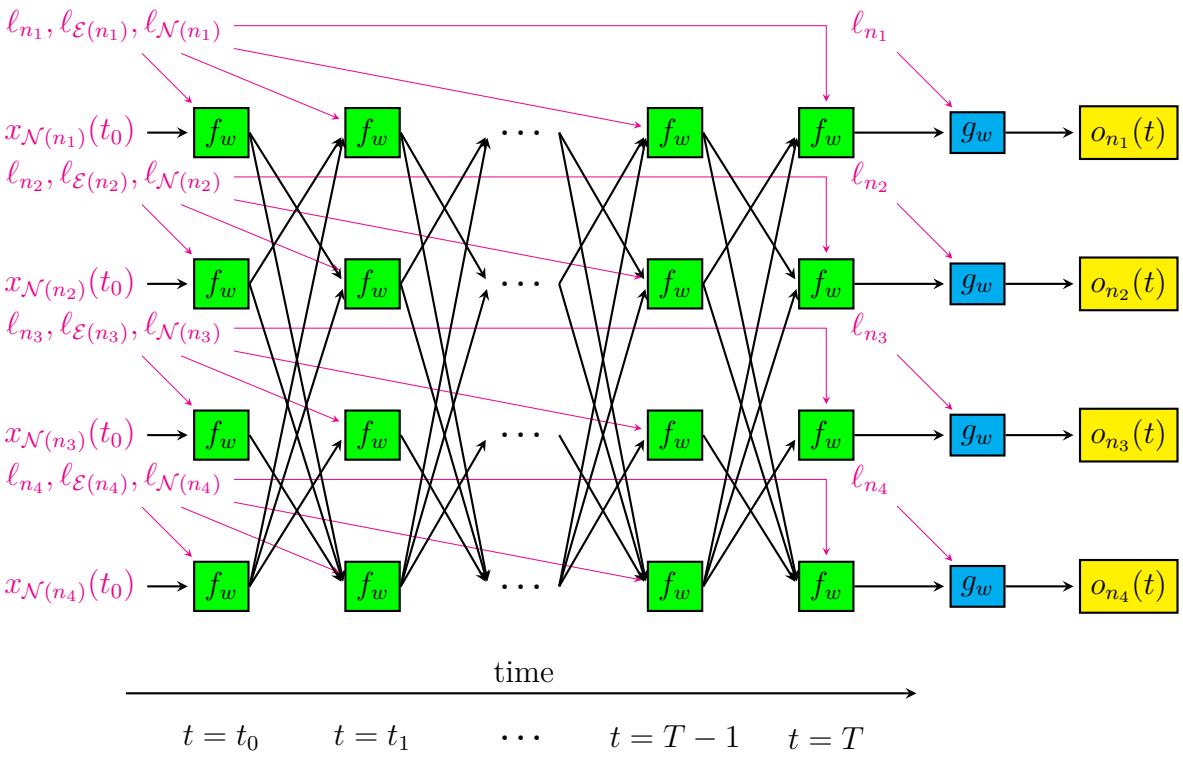


Figure 1.1: Example with a graph

Now from this graph we reach to the following diagram which we call *encoding network*.



The encoding network is a recurrent neural network. And the network obtained by unfolding the encoding network looks like the following,



When f_w and g_w are implemented by feedforward neural networks, the encoding network turns out to be a recurrent neural network where the connections between the neurons can be divided into internal and external connections. The internal connectivity is determined by the neural network architecture used to implement the unit. The external connectivity depends on the edges of the processed graph.

1.2.2 The Learning Algorithm

1.2.2.1 Big picture of the algorithm

the parameters w are estimated using examples contained in the training data set.

Learning set

Let us sonsider an arbitrary set of graphs $\mathcal{G}$.

$$\boldsymbol{\mathcal{G}} = \bigcup_{\alpha \in \Lambda} G_\alpha, \quad \text{where } G_\alpha = (N_\alpha, E_\alpha)$$

Let us also consider,

$$\boldsymbol{\mathcal{N}} = \bigcup_{\alpha \in \Lambda} N_\alpha$$

Definition 3: Domain of a supervised learning framework

The domain is the set of pairs of a graph and a node, i.e. $\boldsymbol{\mathcal{D}} = \boldsymbol{\mathcal{G}} \times \boldsymbol{\mathcal{N}}$.

Definition 4: Learning Set

We take a finite set of graphs from $\boldsymbol{\mathcal{G}}$ as,

$$\{G_1, G_2, \ldots, G_i, \ldots, G_{p-1}, G_p\} \subseteq \boldsymbol{\mathcal{G}} \implies p \leq |\mathcal{G}|.$$

Let us take a subset of the node set of $G_i = (N_i, E_i)$ as,

$$\{n_{i,1}, n_{i,2}, \ldots, n_{i,j}, \ldots, n_{i,q_i-1}, n_{i,q_i}\} \subseteq N_i \in \boldsymbol{\mathcal{N}} \implies q_i \leq |N_i|.$$

We assume a supervised learning framework with the following set,

$$\boldsymbol{\mathcal{L}} = \{(G_i, n_{i,j}, t_{i,j}) | G_i = (N_i, E_i) \in \boldsymbol{\mathcal{G}}, n_{i,j} \in N_i, t_{i,j} \in \mathbb{R}^m, 1 \leq i \leq p, 1 \leq j \leq q_i\}$$

$\boldsymbol{\mathcal{L}}$ is called the learning set.

We consider a desired target association $t_{i,j} \in \mathbb{R}^m$ of the node $n_{i,j}$, which is a vector in some euclidean space of dimension m. Note that $\mathcal{G}$ or $\{G_1, G_2, \ldots, G_i, \ldots, G_{p-1}, G_p\}$ can also be considered as one single graph which may have multiple connected components and modify the definition of learning set accordingly.

Learning in GNNs consists of estimating the parameter w such that the following function,

$$\varphi_w : \boldsymbol{\mathcal{D}} \to \mathbb{R}^m$$

approximates the data in the learning set.

The learning task can be posed as the minimization of a quadratic cost function,

$$e_w = \sum_{i=1}^{p} \sum_{j=1}^{q_i} \left(t_{i,j} - \varphi(G_i, n_{i,j})\right)^2.$$

As common in neural network applications, the cost function may include a penalty term to control other properties of the model. For example, the cost function may contain a smoothing factor to penalize any abrupt changes of the outputs and to improve the generalization performance.

Learning algorithm

The learning algorithm is based on a gradient-descent strategy and is composed of the following steps.

1. The states $x_n(t)$ are iteratively updated by

$$x_n(t) = f_w\left(\ell_n, \ell_{\mathcal{E}(n)}, x_{\mathcal{N}(n)}(t-1), \ell_{\mathcal{N}(n)}\right)$$
$$o_n(t) = g_w(x_n(t), \ell_n)$$

   until at time $t = T$ they approach the fixed point solution of $x(T) \approx x^*$. the hypothesis that F_w is a contraction map ensures the convergence to the fixed point.

2. The gradient $\frac{\partial e_w(T)}{\partial w}$ is computed. This can be carried out in a very efficient way by exploiting the diffusion process that takes place in GNNs. Interestingly, this diffusion process is very much related to the one which takes place in recurrent neural networks, for which the gradient computation is based on backpropagation-through-time algorithm. However, backpropagation through time requires to store the states of every instance of the units. When the graphs and $T - t_0$ are large, the memory required may be considerable. On the other hand, in our case, a more efficient approach is possible, based on the Almeida-Pineda algorithm C.4.

3. The weights w are updated according to the gradient. This is carried out within the traditional framework of gradient descent.

1.2.2.2 Fundamental theorem of GNN

Theorem 1.1: Fundamental theorem of GNN

Consider a graph $G = (N, E)$. Let F_w and G_w be the global transition and the global output functions of a GNN, respectively. Let $F_w(x, \ell)$ and $G_w(x, \ell_N)$ are continuously differentiable w.r.t. x and w. Then we have the following,

1. **Differentiability of φ_w:** φ_w is continuously differentiable w.r.t. w.
2. **Backpropagation:** Let $z(t)$ be defined as,

$$z(t) = z(t+1) \cdot \left(\frac{\partial F_w}{\partial x}\right)(x, \ell) + \frac{\partial e_w}{\partial o} \cdot \left(\frac{\partial G_w}{\partial x}\right)(x, \ell_N).$$

The sequence $z(T), z(T-1), \dots$ converges to a vector

$$z = \lim_{t \to -\infty} z(t).$$

The convergence is exponential and independent of the initial state $z(T)$. Where T corresponds to the time associated with the last layer. Then the gradient of e_w w.r.t. w is,

$$\frac{\partial e_w}{\partial w} = \frac{\partial e_w}{\partial o} \cdot \left(\frac{\partial G_w}{\partial w}\right)(x, \ell_N) + z \cdot \frac{\partial F_w}{\partial x}.$$

which holds where x is the stable state of the GNN.

Proof. **Part 1: Differentiability of φ_w**

Let us define the function Θ,

$$\Theta(x, w) = x - F_w(x, \ell)$$

Clearly, Θ is continuously differentiable w.r.t. x and w. Note that x is the stacking of $|N|$ states coming from the nodes. Also, let us recall that we assumed the dimension of

the states is s (1.1.2). Thus we have the Jacobian matrix of x w.r.t. itself,

$$\frac{\partial x}{\partial x} = I_{s|N|} = \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 0 \\ 0 & 0 & \cdots & 0 & 1 \end{bmatrix}_{s|N| \times s|N|} .$$

Hence, one has,

$$\Theta(x, w) = x - F_w(x, \ell) \implies \left(\frac{\partial \Theta}{\partial x}\right)(x, w) = I_{s|N|} - \left(\frac{\partial F_w}{\partial x}\right)(x, \ell)$$

Now we apply operator norm [Definition 13] on $\left(\frac{\partial F_w}{\partial x}\right)(x, \ell)$ keeping in mind that F_w is a contraction map [Definition 5],

$$\begin{aligned}
&\exists \mu \in [0, 1) \ni \left\| \left(\frac{\partial F_w}{\partial x}\right)(x, \ell) \right\| \leq \mu \\
\implies &\left\| I_{s|N|} - \left(\frac{\partial \Theta}{\partial x}\right)(x, w) \right\| \leq \mu \\
\implies &\left| \left\| I_{s|N|} \right\| - \left\| \left(\frac{\partial \Theta}{\partial x}\right)(x, w) \right\| \right| \leq \left\| I_{s|N|} - \left(\frac{\partial \Theta}{\partial x}\right)(x, w) \right\| \leq \mu \\
\implies &\left\| I_{s|N|} \right\| - \left\| \left(\frac{\partial \Theta}{\partial x}\right)(x, w) \right\| \leq \mu \\
\implies &\left\| \left(\frac{\partial \Theta}{\partial x}\right)(x, w) \right\| \geq \left\| I_{s|N|} \right\| - \mu = 1 - \mu \\
\implies &0 < \left\| \left(\frac{\partial \Theta}{\partial x}\right)(x, w) \right\| \leq 1
\end{aligned}$$

Observe,

$$\left\| \left(\frac{\partial F_w}{\partial x}\right)(x, \ell) \right\| < 1$$

Thus by Lemma C.1, we have

$$\left(I_{s|N|} - \left(\frac{\partial F_w}{\partial x}\right)\right)\left(I_{s|N|} + \left(\frac{\partial F_w}{\partial x}\right) + \left(\frac{\partial F_w}{\partial x}\right)^2 + \cdots\right) = I_{s|N|}$$

Thus $I_{s|N|} - \left(\frac{\partial F_w}{\partial x}\right)$ is invertible so is $\left(\frac{\partial \Theta}{\partial x}\right)(x, w)$. Note that, $\Theta(x, w) = 0$ at the fixed point x^* for any w. And

$$\det\left[\left(\frac{\partial \Theta}{\partial x}\right)(x, w)\right] \neq 0.$$

Thus by Theorem C.1, there exists an open set U containing w such that there exists a unique continuously differentiable function Ψ such that $\Psi(w) = x$, and $\Theta(\Psi(w), w) =$

$\mathbf{0}$ for all $w \in U$. As for any w in the entire domain this argument is true, Ψ is continuously differentiable on the whole domain. Finally, note that,

$$\varphi_w(G, n) = [G_w(\Psi(w), \ell_N)]_n$$

where $[\cdot]_n$ denotes the operator that returns the components corresponding to node n. Thus, φ_w is the composition of differentiable functions and hence is itself differentiable.

Part 2: Backpropagation

$$F_w \text{ is a contraction map } \implies \exists \mu \in [0, 1) \ni \left\| \left(\frac{\partial F_w}{\partial x} \right)(x, \ell) \right\| \leq \mu.$$

By Theorem A.1, there exists a stable fixed point x^* of convergence of F_w. Let z is the corresponding point of x^*. Thus at z,

$$\begin{aligned}
& z \left(I_{s|N|} - \left(\frac{\partial F_w}{\partial x} \right)(x, \ell) \right) = \frac{\partial e_w}{\partial o} \cdot \left(\frac{\partial G_w}{\partial x} \right)(x, \ell_N) \\
\iff & z = \frac{\partial e_w}{\partial o} \cdot \left(\frac{\partial G_w}{\partial x} \right)(x, \ell_N) \cdot \left(I_{s|N|} - \left(\frac{\partial F_w}{\partial x} \right)(x, \ell) \right)^{-1} \quad \left[\because \det \left[\left(\frac{\partial \Theta}{\partial x} \right)(x, w) \right] \neq 0 \right]
\end{aligned}$$

From the proof of Part 1, we have $x = \Psi(w)$ and

$$\Theta(\Psi(w), w) = \mathbf{0} \implies \Psi(w) = F_w(\Psi(w), \ell)$$

Differentiating $\Theta(\Psi(w), w) = \mathbf{0}$ we find

$$\begin{aligned}
& \implies \frac{\partial}{\partial w}(\Theta(x, w)) + \frac{\partial}{\partial x}(\Theta(x, w)) \cdot \frac{\partial x}{\partial w} = \mathbf{0} \\
& \implies \frac{\partial}{\partial w}(x - F_w(x, \ell)) + \frac{\partial}{\partial x}(x - F_w(x, \ell)) \cdot \frac{\partial x}{\partial w} = \mathbf{0} \\
& \implies \frac{\partial}{\partial w}(\Psi - F_w(x, \ell)) + \frac{\partial}{\partial x}(\Psi - F_w(x, \ell)) \cdot \frac{\partial x}{\partial w} = \mathbf{0} \\
& \implies \frac{\partial}{\partial w}(\Psi - F_w(x, \ell)) + \mathbf{0} - \left(\frac{\partial F_w}{\partial x} \right)(x, \ell) \cdot \frac{\partial \Psi}{\partial w} = \mathbf{0} \\
& \implies \frac{\partial}{\partial w}(\Psi - F_w(x, \ell)) = \left(\frac{\partial F_w}{\partial x} \right)(x, \ell) \cdot \frac{\partial \Psi}{\partial w} \\
& \implies \frac{\partial \Psi}{\partial w} - \left(\frac{\partial F_w}{\partial w} \right)(x, \ell) = \left(\frac{\partial F_w}{\partial x} \right)(x, \ell) \cdot \frac{\partial \Psi}{\partial w} \\
& \implies \frac{\partial \Psi}{\partial w} - \left(\frac{\partial F_w}{\partial x} \right)(x, \ell) \cdot \frac{\partial \Psi}{\partial w} = \left(\frac{\partial F_w}{\partial w} \right)(x, \ell) \\
& \implies \left(I_{s|N|} - \left(\frac{\partial F_w}{\partial x} \right)(x, \ell) \right) \cdot \frac{\partial \Psi}{\partial w} = \left(\frac{\partial F_w}{\partial w} \right)(x, \ell) \\
& \implies \frac{\partial \Psi}{\partial w} = \left(I_{s|N|} - \left(\frac{\partial F_w}{\partial x} \right)(x, \ell) \right)^{-1} \cdot \left(\frac{\partial F_w}{\partial w} \right)(x, \ell)
\end{aligned}$$

Now, let us find the gradient of the cost function e_w w.r.t. w. Note that e_w depends on the output of the network $o = G_w(\Psi(w), w)$.

$$\begin{aligned}
\frac{\partial e_w}{\partial w} &= \frac{\partial e_w}{\partial o} \cdot \frac{\partial G_w}{\partial w}(x, \ell_N) + \frac{\partial e_w}{\partial o} \cdot \frac{\partial G_w}{\partial x}(x, \ell_N) \cdot \frac{\partial \Psi}{\partial w} \\
&= \frac{\partial e_w}{\partial o} \cdot \frac{\partial G_w}{\partial w}(x, \ell_N) + \frac{\partial e_w}{\partial o} \cdot \frac{\partial G_w}{\partial x}(x, \ell_N) \cdot \left(I_{s|N|} - \left(\frac{\partial F_w}{\partial x}\right)(x, \ell)\right)^{-1} \cdot \left(\frac{\partial F_w}{\partial w}\right)(x, \ell) \\
&= \frac{\partial e_w}{\partial o} \cdot \frac{\partial G_w}{\partial w}(x, \ell_N) + z \cdot \left(\frac{\partial F_w}{\partial w}\right)(x, \ell)
\end{aligned}$$

■

Note that,

$$z(t) = z(t+1) \cdot \left(\frac{\partial F_w}{\partial x}\right)(x, \ell) + \frac{\partial e_w}{\partial o} \cdot \left(\frac{\partial G_w}{\partial x}\right)(x, \ell_N)$$

Thus,

$$\begin{aligned}
z(t-1) &= z(t) \cdot \left(\frac{\partial F_w}{\partial x}\right)(x, \ell) + \frac{\partial e_w}{\partial o} \cdot \left(\frac{\partial G_w}{\partial x}\right)(x, \ell_N) \\
&= \left[z(t+1) \cdot \left(\frac{\partial F_w}{\partial x}\right)(x, \ell) + \frac{\partial e_w}{\partial o} \cdot \left(\frac{\partial G_w}{\partial x}\right)(x, \ell_N)\right] \cdot \left(\frac{\partial F_w}{\partial x}\right)(x, \ell) \\
&\qquad + \frac{\partial e_w}{\partial o} \cdot \left(\frac{\partial G_w}{\partial x}\right)(x, \ell_N) \\
&= z(t+1) \cdot \left(\left(\frac{\partial F_w}{\partial x}\right)(x, \ell)\right)^2 + \sum_{i=0}^{1} \frac{\partial e_w}{\partial o} \cdot \left(\frac{\partial G_w}{\partial x}\right)(x, \ell_N) \cdot \left(\left(\frac{\partial F_w}{\partial x}\right)(x, \ell)\right)^i
\end{aligned}$$

$$\begin{aligned}
z(t-2) =& z(t-1) \cdot \left(\frac{\partial F_w}{\partial x}\right)(x, \ell) + \frac{\partial e_w}{\partial o} \cdot \left(\frac{\partial G_w}{\partial x}\right)(x, \ell_N) \\
=& \left[z(t+1) \cdot \left(\left(\frac{\partial F_w}{\partial x}\right)(x, \ell)\right)^2 + \sum_{i=0}^{1} \frac{\partial e_w}{\partial o} \cdot \left(\frac{\partial G_w}{\partial x}\right)(x, \ell_N) \cdot \left(\left(\frac{\partial F_w}{\partial x}\right)(x, \ell)\right)^i\right] \cdot \\
& \left(\frac{\partial F_w}{\partial x}\right)(x, \ell) + \frac{\partial e_w}{\partial o} \cdot \left(\frac{\partial G_w}{\partial x}\right)(x, \ell_N) \\
=& z(t+1) \cdot \left(\left(\frac{\partial F_w}{\partial x}\right)(x, \ell)\right)^3 + \sum_{i=0}^{2} \frac{\partial e_w}{\partial o} \cdot \left(\frac{\partial G_w}{\partial x}\right)(x, \ell_N) \cdot \left(\left(\frac{\partial F_w}{\partial x}\right)(x, \ell)\right)^i
\end{aligned}$$

Thus we have,

$$z(t-k) = z(t+1) \cdot \left(\left(\frac{\partial F_w}{\partial x}\right)(x, \ell)\right)^{k+1} + \sum_{i=0}^{k} \frac{\partial e_w}{\partial o} \cdot \left(\frac{\partial G_w}{\partial x}\right)(x, \ell_N) \cdot \left(\left(\frac{\partial F_w}{\partial x}\right)(x, \ell)\right)^i$$

Ptting, $t = T - 1$ and $k = T - t - 1$, we have,

$$z(t) = z(T) \cdot \left(\left(\frac{\partial F_w}{\partial x}\right)(x, \ell)\right)^{T-t} + \sum_{i=0}^{T-t-1} \frac{\partial e_w}{\partial o} \cdot \left(\frac{\partial G_w}{\partial x}\right)(x, \ell_N) \cdot \left(\left(\frac{\partial F_w}{\partial x}\right)(x, \ell)\right)^i$$

Let us assume,

$$z(T) = \frac{\partial e_w(T)}{\partial o(T)} \cdot \left(\frac{\partial G_w}{\partial x(T)} \right) (x(T), \ell_N)$$

and $x(t) = x$, then for $t_0 \leq t \leq T$ we have,

$$\begin{aligned} z(t) &= \sum_{i=0}^{T-t} \left[\frac{\partial e_w(T)}{\partial o(T)} \cdot \left(\frac{\partial G_w}{\partial x(T)} \right) (x(T), \ell_N) \cdot \prod_{j=1}^{i} \left(\frac{\partial F_w}{\partial x(T-j)} \right) (x(T-j), \ell) \right] \\ &= \sum_{i=0}^{T-t} \frac{\partial e_w(T)}{\partial x(T-i)} \\ &= \sum_{i=T}^{t} \frac{\partial e_w(T)}{\partial x(i)} \end{aligned}$$

1.2.2.3 Pseudocode for learning algorithm

At this point we are ready to write the pseudocode for learning algorithm [See Algorithm 1].

1.2.3 Transition and Output Function Implementations

Note that in Algorithm 1, we took F_w and G_w as given. In this section we will discuss the implementation of these two functions. The implementation of the local output function g_w does not need to fulfill any particular constraint. In GNNs, g_w is a multilayered feedforward neural network. On the other hand, the local transition function f_w plays a crucial role in the proposed model, since its implementation determines the number and the existence of the solutions. Two implementations of GNN are provided.

1.2.3.1 Linear (nonpositional) GNN

1.2.3.2 Nonlinear (nonpositional) GNN

1.3 Computational Complexity

Algorithm 1 GNN Learning Algorithm

procedure MAIN:
  $\mathcal{S} \leftarrow$ some stopping criterion depends typically on requirements
  $\lambda \leftarrow$ learning rate
  $\epsilon_f \leftarrow$ convergence tolerance of contractive map F_w
  $\epsilon_z \leftarrow$ convergence tolerance of contractive map z
  $w \leftarrow$ random weight initialization
  $x, T \leftarrow$ FORWARD(w)
  repeat:
    $\frac{\partial e_w}{\partial w} \leftarrow$ BACKWARD(x, w, T)
    $w \leftarrow w - \lambda \frac{\partial e_w}{\partial w}$
    if $\mathcal{S}$ satisfied
      return w
procedure FORWARD(w):
  $t \leftarrow 0$
  $x(0) \leftarrow$ # random initialization of initial state. Theorem A.1 ensures convergence
  repeat:
    $x(t+1) \leftarrow F_w(x(t), \ell)$ # Theorem A.1, construction of Cauchy sequence
    if $\|x(t) - x(t+1)\| \leq \epsilon_f$:
      return $x(t+1), t$
    else:
      $t \leftarrow t + 1$
procedure BACKWARD(x, w, T):
  $o \leftarrow G_w(x, \ell_N)$
  $A \leftarrow \frac{\partial F_w}{\partial x}(x, \ell)$
  $b \leftarrow \frac{\partial e_w}{\partial o} \cdot \frac{\partial G_w}{\partial x}(x, \ell_N)$
  $z(T) \leftarrow$ random initialization, Theorem 1.1 ensures convergence
  $t \leftarrow T - 1$
  repeat:
    $z(t) \leftarrow z(t+1) \cdot A + b$
    if $\|z(t) - z(t+1)\| \leq \epsilon_z$:
      break
    else:
      $t \leftarrow t - 1$
  $C \leftarrow \frac{\partial F_w}{\partial w}(x, \ell)$
  $d \leftarrow \frac{\partial e_w}{\partial o} \cdot \frac{\partial G_w}{\partial w}(x, \ell_N)$
  $\frac{\partial e_w}{\partial w} \leftarrow z(t) \cdot C + d$
  return $\frac{\partial e_w}{\partial w}$

Chapter 2

GNN Application

Appendix A

Contraction mapping theorem

A.1 Contraction mapping

Definition 5: Contraction map

Let (X, d) be a complete metric space. Then a map $T : X \to X$ is called a contraction map on X if there exists $q \in [0, 1)$ such that

$$d\left(T(x), T(x')\right) \leq q\, d(x, x')$$

for all $x, x' \in X$. Contraction map is also known as contractive map. If the above condition is instead satisfied for $q \in [0, 1]$, then the mapping is said to be a non-expansive map.
More generally, if (X, d_X) and (Y, d_Y) are two metric spaces, then $T : X \to Y$ is a contractive mapping if there is a constant $q \in [0, 1)$ such that

$$d_Y(T(x), T(x')) \leq q\, d_X(x, x')$$

for all x and x' in X.

Definition 6: Lipschitz continuous map

Given two metric spaces (X, d_X) and (Y, d_Y), a function $f : X \to Y$ is called Lipschitz continuous if there exists a real constant $K \geq 0$ such that, for all x and x' in X,

$$d_Y(f(x), f(x')) \leq K d_X(x, x').$$

Any such K is referred to as a Lipschitz constant for the function f and f may also be referred to as K-Lipschitz.

- The smallest constant is sometimes called the (best) Lipschitz constant of f or the dilation or dilatation of f.
- If $K = 1$ the function is called a short map.
- If $0 \leq K < 1$, the function is called a contraction [Definition 5].

Lemma A.1

Contraction map $\implies$ Lipschitz continuous map.

Proof. Clearly by definition. ■

Definition 7: Uniformly continuous map

Let $M_1 = (A_1, d_1)$ and $M_2 = (A_2, d_2)$ be metric spaces. Then a mapping $f : A_1 \to A_2$ is uniformly continuous on A_1 if and only if:

$$\forall \epsilon \in \mathbb{R}_{>0} : \exists \delta \in \mathbb{R}_{>0} : \forall x, y \in A_1 : d_1(x, y) < \delta \implies d_2(f(x), f(y)) < \epsilon.$$

Definition 8: Continuous map

Let $M_1 = (A_1, d_1)$ and $M_2 = (A_2, d_2)$ be metric spaces. Then a mapping $f : A_1 \to A_2$ is continuous at a point $c \in A_1$ if and only if:

$$\forall \epsilon \in \mathbb{R}_{>0} : \exists \delta \in \mathbb{R}_{>0} : d_1(c, y) < \delta \implies d_2(f(c), f(y)) < \epsilon.$$

Lemma A.2

Uniformly continuous map $\implies$ Continuous map.

Proof. Given $\epsilon > 0$. Then by uniform continuity, there is δ such that for all $x, y \in A_1$ we have

$$d_1(x, y) < \delta \implies d_2(f(x), f(y)) < \epsilon.$$

In particular if we take $x = c$ we get that for all $y \in A_1$

$$d_1(x, y) < \delta \implies d_2(f(x), f(y)) < \epsilon.$$

This is the definition of continuity [Definition 8]. ■

Lemma A.3

Lipschitz continuous map $\implies$ Uniformly continuous map.

Proof. Given two metric spaces (X, d_X) and (Y, d_Y), a Lipschitz continuous map $f : X \to Y$, we get from Definition 6, there exists a real constant $K \geq 0$ such that, for all x and x' in X,

$$d_Y(f(x), f(x')) \leq K d_X(x, x').$$

- Case 1 ($K = 0$): we get for all x and x' in X,

  $$d_Y(f(x), f(x')) = 0.$$

  Hence, f is a constant map. Clearly f is uniformly continuous [use Definition 6].

- Case 2 ($K > 0$): By Lipschitz continuity,

  $$\forall x, x' \in X : d_Y(f(x), f(x')) \leq K d_X(x, x').$$

  Take $\epsilon > 0$. Then there exists $\delta = \frac{\epsilon}{K}$. Suppose, $d_X(x, x') < \delta$. Hence, we have,

  $$\begin{aligned} \forall \epsilon \in \mathbb{R}_{>0} : \exists \delta = \frac{\epsilon}{K} \in \mathbb{R}_{>0} : \forall x, x' \in X : & d_X(x, x') < \delta \\ & \implies K d_X(x, x') < \epsilon \\ & \implies d_Y(f(x), f(x')) < \epsilon. \end{aligned}$$

■

Lemma A.4

Contraction map $\implies$ Lipschitz continuous map $\implies$ Uniformly continuous map $\implies$ Continuous map.

Proof. Using Lemma A.1, A.3 and A.2. ■

Definition 9: Fixed point of a function

c is a fixed point of a function f if c belongs to both the domain and the codomain of f, and $f(c) = c$.

Definition 10: Cauchy sequence

A sequence $a_1, a_2, \ldots$ is called a Cauchy sequence if the terms of the sequence eventually all become arbitrarily close to one another. That is,

$$\forall \epsilon > 0, \exists N \in \mathbb{N} \ni m, n > N \implies |a_m - a_n| < \epsilon.$$

Definition 11: Complete metric space

A metric space (X, d) is complete if every Cauchy sequence in X converges in X (that is, to some point of X).

A.2 Contraction mapping theorem

Theorem A.1: Contraction mapping theorem

Let (X, d) be a non-empty complete metric space with a contraction mapping $T : X \to X$. Then T admits a unique fixed-point $x^* \in X$ (i.e. $T(x^*) = x^*$). Furthermore, x^* can be found by starting with an arbitrary element $x_0 \in X$ and defining a sequence $(x_n)_{n\in\mathbb{N}}$ by $x_n = T(x_{n-1})$ for $n \geq 1$. Then $\lim_{n\to\infty} x_n = x^*$.

Proof. **Existence of fixed point**

Let $x_0 \in X$ be arbitrary and define a sequence $(x_n)_{n\in\mathbb{N}}$ by setting $x_n = T(x_{n-1})$. We first note that for all $n \in \mathbb{N}$, we have the inequality

$$d(x_{n+1}, x_n) \leq q^n\, d(x_1, x_0).$$

This follows by induction on n, using the fact that T is a contraction mapping. More explicitely, note that as T is a contraction mapping and using the definition $x_n = T(x_{n-1})$, we claim that there exists $q \in [0, 1)$ such that,

$$\begin{aligned}
&d(x_2, x_1) = d(T(x_1), T(x_0)) \leq q\, d(x_1, x_0)\\
&d(x_3, x_2) = d(T(x_2), T(x_1)) \leq q\, d(x_2, x_1) \leq q^2\, d(x_1, x_0)\\
&\vdots\\
&d(x_{n+1}, x_n) \leq q^n\, d(x_1, x_0)
\end{aligned}$$

This follows by induction on n, using the fact that T is a contraction mapping. Then we can show that $(x_n)_{n\in\mathbb{N}}$ is a Cauchy sequence. In particular, let $m, n \in \mathbb{N}$ such that $m > n$:

$$\begin{aligned}
d(x_m, x_n) &\leq d(x_m, x_{m-1}) + d(x_{m-1}, x_{m-2}) + \cdots + d(x_{n+1}, x_n)\\
&\leq q^{m-1} d(x_1, x_0) + q^{m-2} d(x_1, x_0) + \cdots + q^n d(x_1, x_0)\\
&= q^n d(x_1, x_0) \sum_{k=0}^{m-n-1} q^k\\
&< q^n d(x_1, x_0) \sum_{k=0}^{\infty} q^k\\
&= q^n d(x_1, x_0) \left(\frac{1}{1-q}\right).
\end{aligned}$$

Let $\epsilon > 0$ be arbitrary. Since $q \in [0, 1)$, we can find a large $N \in \mathbb{N}$ so that

$$q^N < \frac{\varepsilon(1-q)}{d(x_1, x_0)}.$$

Therefore, by choosing m and n greater than N we may write:

$$d(x_m, x_n) \leq q^n d(x_1, x_0) \left(\frac{1}{1-q}\right) < \left(\frac{\varepsilon(1-q)}{d(x_1, x_0)}\right) d(x_1, x_0) \left(\frac{1}{1-q}\right) = \varepsilon.$$

This proves that the sequence $(x_n)_{n\in\mathbb{N}}$ is Cauchy. By completeness of (X, d), the sequence has a limit $x^* \in X$. Furthermore, x^* must be a fixed point of T:

$$x^* = \lim_{n\to\infty} x_n = \lim_{n\to\infty} T(x_{n-1}) = T\left(\lim_{n\to\infty} x_{n-1}\right) = T(x^*).$$

As a contraction mapping, T is continuous, so bringing the limit inside T was justified.

Uniqueness of fixed point

Let $x, x' \in X$ are both fixed points where $x \neq x'$. Thus $d(x, x') \neq 0$. But notice,

$$\begin{aligned} & d(x, x') = d(T(x), T(x')) \leq q\, d(x, x') \\ \implies & (1-q)d(x, x') \leq 0 \\ \implies & d(x, x') \leq 0 \implies d(x, x') = 0 \iff x = x'. \end{aligned}$$

Hence, it is not possible to have two different fixed points.

Calculating the fixed point

We notice that no matter where we choose our arbitrary starting point $x_0 \in X$, by the construction of the sequence using the contraction mapping (i.e. $x_n = T(x_{n-1})$) we always reach to the unique fixed point as the limit of the sequence,

$$\lim_{n\to\infty} x_n = x^*.$$

Speed of convergence:

For the fixed point x^* the speed of convergence is described by the following,

$$d(x^*, x_n) \leq \frac{q^n}{1-q} d(x_1, x_0)$$

We can see that for any initial value x_0 the system converges to the fixed point x^* exponentially. ■

Appendix B

Classic iterative methods

B.1 Jacobi and Gauss-Seidel iterative method

The current analysis is based on [3]. In this notes we discuss classic iterative methods on solving the linear operator equation,

$$Au = f$$

posed on a finite dimensional Hilbert space $\mathbb{V} \cong \mathbb{R}^N$ equipped with an inner product $(\mathring{u}, \mathring{u})$. Here $A : \mathbb{V} \mapsto \mathbb{V}$ is an *symmetric and positive definite* (SPD) operator, $f \in \mathbb{V}$ is given, and we are looking for $u \in \mathbb{V}$ such that $Au = f$ holds.

Definition 12

the matrix A is diagonally dominant if

$$|a_{ii}| \geq \sum_{j \neq i} |a_{ij}| \quad \text{for all } i.$$

where a_{ij} denotes the entry in the ith row and j-th column. Note that this definition uses a weak inequality, and is therefore sometimes called weak diagonal dominance. If a strict inequality ($>$) is used, this is called strict diagonal dominance.

Theorem B.1

A Hermitian diagonally dominant matrix A with real non-negative diagonal entries is positive semidefinite.

Appendix C

Important definitions and theorems

C.1 Operator norm

Definition 13: Operator norm

The operator norm of a linear operator $T : \mathbb{V} \to \mathbb{W}$ is the largest value by which T stretches an element of $\mathbb{V}$,

$$\|T\| = \sup_{\|v\|=1} \|T(v)\|$$

Following are some equivalent definitions for operator norm,

$$\begin{aligned}
\|T\| &= \inf\{c \geq 0 \ : \ \|Tv\| \leq c\|v\| \quad \text{for all} \quad v \in \mathbb{V}\} \\
&= \sup\{\|Tv\| \ : \ \|v\| \leq 1 \ \text{ and } \ v \in \mathbb{V}\} \\
&= \sup\{\|Tv\| \ : \ \|v\| < 1 \ \text{ and } \ v \in \mathbb{V}\} \\
&= \sup\{\|Tv\| \ : \ \|v\| \in \{0, 1\} \ \text{ and } \ v \in \mathbb{V}\} \\
&= \sup\{\|Tv\| \ : \ \|v\| = 1 \ \text{ and } \ v \in \mathbb{V} \neq \{0\}\} \\
&= \sup\left\{\frac{\|Tv\|}{\|v\|} \ : \ \|v\| \neq 0 \ \text{ and } \ v \in \mathbb{V} \neq \{0\}\right\}
\end{aligned}$$

Properties

The operator norm is indeed a norm on the space of all bounded operators between $\mathbb{V}$ and $\mathbb{W}$. This means

$$\begin{aligned}
&\|A\| \geq 0 \text{ and } \|A\| = 0 \text{ if and only if } A = 0,\\
&\|aA\| = |a|\|A\| \text{ for every scalar } a,\\
&\|A + B\| \leq \|A\| + \|B\|,\\
&\|AB\| \leq \|A\|\|B\|.
\end{aligned}$$

We can also deduce,

$$\|A - B\| \geq \Big|\|A\| - \|B\|\Big|.$$

C.2 Neumann series

Definition 14: Neumann series

A Neumann series is a mathematical series of the form

$$\sum_{k=0}^{\infty} T^k$$

where T is an operator and $T^k := T^{k-1} \circ T$ its k times repeated application.

Lemma C.1: Convergence of Neumann series

$\|T\| < 1 \implies \mathrm{Id} - T$ is invertible.

Proof.

$$\left\|\sum_{k=0}^{\infty} T^k\right\| \leq \sum_{k=0}^{\infty} \|T\|^k$$

$\|T\| < 1 \implies \left\|\sum_{k=0}^{\infty} T^k\right\|$ converges. Consider the partial sums

$$S_n := \sum_{k=0}^{n} T^k.$$

Then we have, $(\mathrm{Id} - T)\, S_n = \mathrm{Id} - T^{n+1}$ Thus,

$$\|T^n\| \leqslant \|T\|^n \to 0 \implies \|T^n\| \to 0 \implies T^n \to 0.$$

Hence,

$$\lim_{n\to\infty} (\mathrm{Id} - T)\, S_n = \lim_{n\to\infty} \left(\mathrm{Id} - T^{n+1}\right) \implies (\mathrm{Id} - T) \sum_{k=0}^{\infty} T^k = \mathrm{Id}$$

Thus, $\sum_{k=0}^{\infty} T^k$ is the inverse of $\mathrm{Id} - T$. ■

Suppose that T is a bounded linear operator on the normed vector space X. If the Neumann series converges in the operator norm, then $\mathrm{Id} - T$ is invertible and its inverse is the series:

$$(\mathrm{Id} - T)^{-1} = \sum_{k=0}^{\infty} T^k$$

C.3 Implicit function theorem

Theorem C.1: Implicit function theorem

Let $f : \mathbb{R}^{n+m} \to \mathbb{R}^m$ be a continuously differentiable function, and let $\mathbb{R}^{n+m}$ have coordinates $(\mathbf{x}, \mathbf{y})$. Fix a point $(\mathbf{a}, \mathbf{b}) = (a_1, \ldots, a_n, b_1, \ldots, b_m)$ with $f(\mathbf{a}, \mathbf{b}) = \mathbf{0}$, where $\mathbf{0} \in \mathbb{R}^m$ is the zero vector. If the Jacobian matrix:

$$J_{f,\mathbf{y}}(\mathbf{a}, \mathbf{b}) = \left[\frac{\partial f_i}{\partial y_j}(\mathbf{a}, \mathbf{b})\right]$$

is invertible, then there exists an open set $U \subset \mathbb{R}^n$ containing $\mathbf{a}$ such that there exists a unique continuously differentiable function $g : U \to \mathbb{R}^m$ such that $g(\mathbf{a}) = \mathbf{b}$, and $f(\mathbf{x}, g(\mathbf{x})) = \mathbf{0}$ for all $\mathbf{x} \in U$.

C.4 Almeida-Pineda algorithm

Index

Bibliography

[1] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. "The Graph Neural Network Model". In: *IEEE Transactions on Neural Networks* 20.1 (2009), pp. 61–80. DOI: 10.1109/TNN.2008.2005605 (page - 2).

[2] Michael JD Powell. "An efficient method for finding the minimum of a function of several variables without calculating derivatives". In: *The computer journal* 7.2 (1964), pp. 155–162 (page - 6).

[3] Roberto Bagnara. "A unified proof for the convergence of Jacobi and Gauss–Seidel methods". In: *SIAM review* 37.1 (1995), pp. 93–97 (page - 22).