CS-E4740 - Federated Learning

FL Networks

Assoc. Prof. Alexander Jung

Spring 2025

YouTube

LinkedIn

GitHub

Table of Contents

Table of Contents

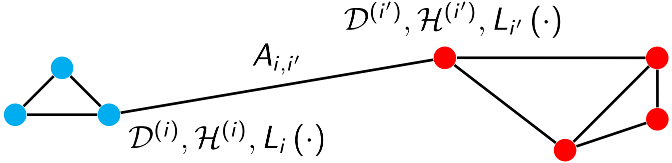
A (“Real-World”) FL System

Abstracting Away Details

To analyze an FL system, we (need to) ignore many details:

- physical properties of communication links
- low-level communication protocols
- hardware configuration of devices
- operating systems of devices
- scientific computing software (Python packages)

An FL Network



- FL network consists of devices, denoted $i = 1, \ldots, n$
- some i, i' connected by edge with the weight $A_{i,i'} > 0$
- device i **generates data** $\mathcal{D}^{(i)}$ and **trains model** $\mathcal{H}^{(i)}$
- data $\mathcal{D}^{(i)}$ used to construct loss func. $L_i(\cdot)$

FL Network is an Approximation

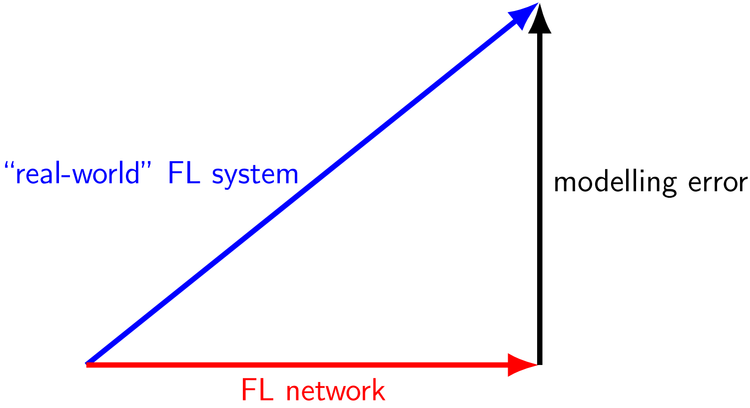


Table of Contents

A Precise Definition

An FL Network is a tuple, consisting of a

- finite number of **nodes** $\mathcal{V} := \{1, \ldots, n\}$
- **local model** $\mathcal{H}^{(i)}$ at each node $i \in \mathcal{V}$
- a **local loss function** $L_i(\cdot)$ at each node $i \in \mathcal{V}$
- set of undirected **edges** $\mathcal{E}$
- positive **edge-weight** $A_{i,i'} \in \mathbb{R}_{++}$ for each $\{i, i'\} \in \mathcal{E}$

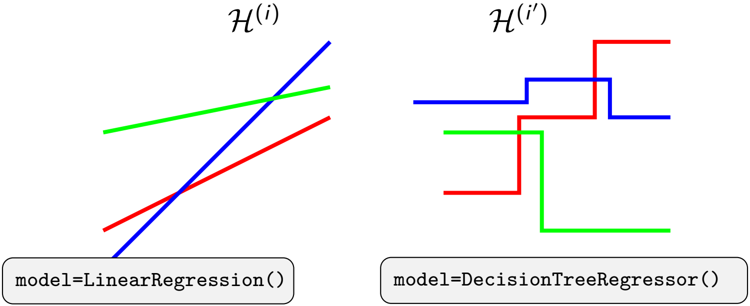
We collect nodes $\mathcal{V}$, edges $\mathcal{E}$ and edge-weights $A_{i,i'}$ of FL network into an **undirected weighted graph** $\mathcal{G}$.

Nodes of an FL Network

- consider an FL system with finite number n of devices
- index devices with natural number $i = 1, \ldots, n$
- indices form the nodes $\mathcal{V}$ of an FL network
- each node $i \in \mathcal{V}$ **represents** a physical device
- we abuse language and use "device i" for "node i" and vice-versa

Local Models of an FL Network

- consider FL system with devices $i = 1, \ldots, n$
- each device trains local (personal) model $\mathcal{H}^{(i)}$
- devices can use different local models (heterogeneity)
- use local model parameters $\mathbf{w}^{(i)}$ for parametric $\mathcal{H}^{(i)}$



Local Loss Functions of an FL Network

- device i trains local model $\mathcal{H}^{(i)}$
- *to train a model* is to learn a useful hypothesis $h^{(i)} \in \mathcal{H}^{(i)}$
- measure usefulness of $h^{(i)}$ by the local loss function

$$L_i(\cdot) : \mathcal{H}^{(i)} \rightarrow \mathbb{R} : h^{(i)} \mapsto L_i\left(h^{(i)}\right)$$

- different nodes can have different loss functions

Local Loss Functions of an FL Network - ctd.

- FL methods use different constructions of loss funcs.
- for param. models $\mathcal{H}^{(i)}$, with parameters $\mathbf{w}^{(i)} \in \mathbb{R}^d$, use

$$L_i\left(\cdot\right) : \mathbb{R}^d \rightarrow \mathbb{R} : \mathbf{w}^{(i)} \mapsto L_i\left(\mathbf{w}^{(i)}\right)$$

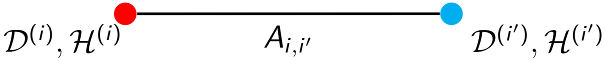
- can use average loss on local dataset

$$L_i\left(\mathbf{w}^{(i)}\right) := \frac{1}{m_i} \sum_{r=1}^{m_i} \left(y^{(i,r)} - \left(\mathbf{w}^{(i)}\right)^T \mathbf{x}^{(i,r)}\right)^2$$

- use reward signals to estimate loss (federated reinf. learning)

Edges (Links) in FL Network

- FL network contains undirected edges $\mathcal{E}$
- edge $\{i, i'\} \in \mathcal{E}$ indicates **similarity** between devices i, i'
- we quantify similarity with edge weight $A_{i,i'} > 0$
- meaning of $\{i, i'\} \in \mathcal{E}$ is two-fold
  - channel between devices i, i' ($A_{i,i'} \approx$ channel capacity)
  - devices should have similar personalized models



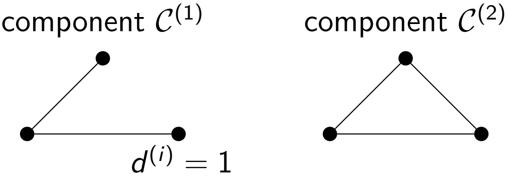
$\mathcal{V}, \mathcal{E}, \left\{A_{i',i'}\right\}_{\{i,i'\}\in\mathcal{E}}$ constitute the graph $\mathcal{G}$ of an FL network

Connectivity of an FL Network

consider FL network with undirected graph $\mathcal{G}$

- $\mathcal{G}$ is connected if there is a path between any two $i, i' \in \mathcal{V}$
- a component $\mathcal{C}$ of $\mathcal{G}$ is a connected sub-graph $\mathcal{C} \subseteq \mathcal{V}$
- neighbourhood of $i \in \mathcal{V}$ is $\mathcal{N}^{(i)} := \{ i' \in \mathcal{V} : \{i, i'\} \in \mathcal{E} \}$
- weighted node degree $d^{(i)} := \sum_{i' \in \mathcal{N}^{(i)}} A_{i,i'}$
- max. node degree $d_{\rm max} := \max_{i \in \mathcal{V}} d^{(i)}$

Connectivity of an FL Network - Example



- FL network with graph $\mathcal{G}$ containing $n=6$ nodes
- uniform edge-weights, $A_{i,i'} = 1$ for all $\{i, i'\} \in \mathcal{E}$
- $\mathcal{G}$ consists of two connected components $\mathcal{C}^{(1)}, \mathcal{C}^{(2)}$
- max. node degree $d_{\max} = 2$

Design Choices

- we use FL networks to design FL algorithms
- each FL network involves design choices for
  - nodes (which devices do we include?)
  - local models and loss functions
  - edges (which devices are connected or similar?)
- trade-offs between computational complexity, accuracy, robustness, explainability, privacy-protection

Design Space and Objectives

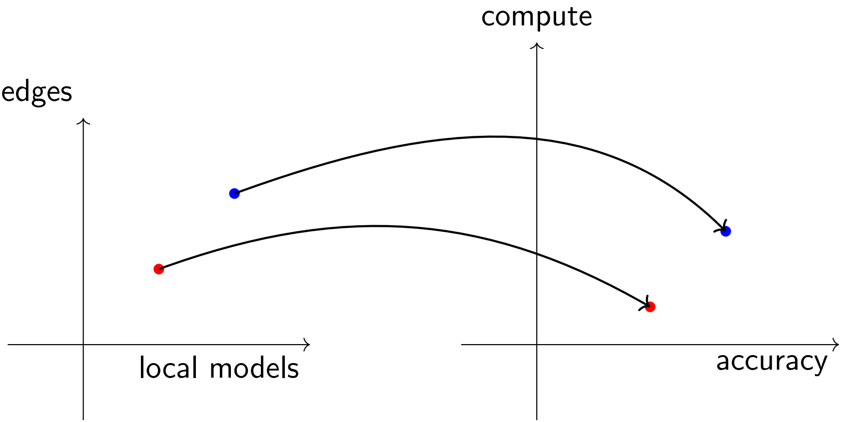


Table of Contents

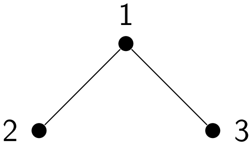
Laplacian Matrix of FL Network

- Consider an FL network with undirected weighted graph $\mathcal{G}$
- Laplacian matrix $\mathbf{L}^{(\mathcal{G})} \in \mathbb{R}^{n \times n}$ of $\mathcal{G}$ defined element-wise

$$L^{(\mathcal{G})}_{i,i'} := \begin{cases} -A_{i,i'} & \text{for } i \neq i', \{i, i'\} \in \mathcal{E} \\ \sum_{i'' \neq i} A_{i,i''} & \text{for } i = i' \\ 0 & \text{else.} \end{cases}$$

Laplacian Matrix - Example

Consider graph $\mathcal{G}$ with uniform edge weights $A_{i,i'} = 1$



$$\mathbf{L}^{(\mathcal{G})} = \begin{pmatrix} 2 & -1 & -1 \\ -1 & 1 & 0 \\ -1 & 0 & 1 \end{pmatrix}$$

Properties of the Laplacian Matrix

The Laplacian matrix $\mathbf{L}^{(\mathcal{G})}$ of any FL network is

- symmetric $\mathbf{L}^{(\mathcal{G})} = \left(\mathbf{L}^{(\mathcal{G})}\right)^T$ (since edges are undirected)
- and positive semi-definite (psd),

$$\mathbf{w}^T \mathbf{L}^{(\mathcal{G})} \mathbf{w} \geq 0 \text{ for every } \mathbf{w} \in \mathbb{R}^n. \tag{1}$$

The psd property (1) follows from the identity

$$\mathbf{w}^T \mathbf{L}^{(\mathcal{G})} \mathbf{w} = \underbrace{\sum_{\{i,i'\} \in \mathcal{E}} A_{i,i'} \left(w^{(i)} - w^{(i')}\right)^2}_{\text{total variation}}$$

which holds for any $\mathbf{w} = \left(w^{(1)}, \ldots, w^{(n)}\right)^T \in \mathbb{R}^n$

The Spectrum of the Laplacian Matrix

- We can decompose any Laplacian matrix $\mathbf{L}^{(\mathcal{G})} \in \mathbb{R}^{n \times n}$ as

$$\mathbf{L}^{(\mathcal{G})} = \sum_{j=1}^{n} \lambda_j \mathbf{u}^{(j)} \left(\mathbf{u}^{(j)}\right)^T,$$

- with orthonormal eigenvecs. $\mathbf{u}^{(1)}, \ldots, \mathbf{u}^{(n)} \in \mathbb{R}^n$, i.e.,

$$\left(\mathbf{u}^{(j)}\right)^T \mathbf{u}^{(j')} = \begin{cases} 1 & \text{for } j = j' \\ 0 & \text{otherwise.} \end{cases}$$

- and non-neg. eigvals

$$0 = \lambda_1 \leq \ldots \leq \lambda_n \leq 2d_{\max}$$

spectrum of $\mathbf{L}^{(\mathcal{G})}$ = set of different eigvals

Spectral Characterization of FL Networks

Consider an FL network with the graph $\mathcal{G}$ consisting of k connected components $\mathcal{C}^{(1)}, \ldots, \mathcal{C}^{(k)}$.

The Laplacian matrix $\mathbf{L}^{(\mathcal{G})} = \sum_{j=1}^{n} \lambda_j \mathbf{u}^{(j)} \big(\mathbf{u}^{(j)}\big)^T$

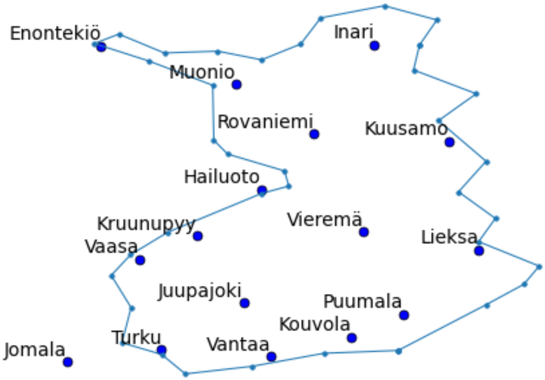
- has eigvals. $\lambda_c = 0$ for $c = 1, \ldots, k$
- corresponding eigvec. $\mathbf{u}^{(c)}$ is the indicator for $\mathcal{C}^{(c)}$,

$$u_i^{(c)} = \begin{cases} \frac{1}{\sqrt{\left|\mathcal{C}^{(c)}\right|}} & \text{for } i \in \mathcal{C}^{(c)} \\ 0 & \text{otherwise.} \end{cases}$$

Note: $\mathcal{G}$ is connected, i.e., consists of single component ($k = 1$), if and only if $\lambda_2 > 0$

Table of Contents

Weather Stations across Finland



Each weather station i collects data (observations) $\mathcal{D}^{(i)}$ that can be used to train a local model $\mathcal{H}^{(i)}$

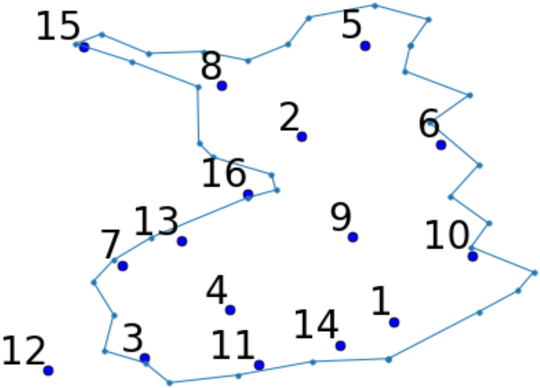
Python script for reproducing the Fig.:

Local Dataset of a FMI Station

Each FMI station i generates a local dataset $\mathcal{D}^{(i)}$ of the form

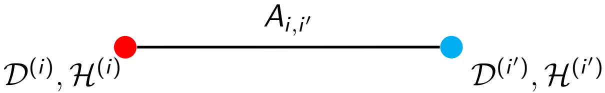
Time	Air Temperature
2025-01-13 16:08:00	-1.5
2025-01-13 16:09:00	-1.5
2025-01-13 16:10:00	-1.4
2025-01-13 16:11:00	-1.5
2025-01-13 16:12:00	-1.5

FL Network for FMI



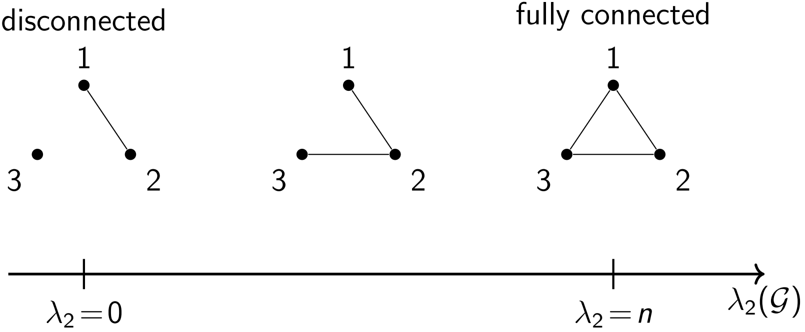
Which nodes (FMI stations) should be connected by edges ?

The Effect of Adding an Edge



- model params. (updates) exchanged across edge $\Rightarrow$ requires a communication link between i, i'!
- model params. $\mathbf{w}^{(i)}, \mathbf{w}^{(i')}$ are coupled with strength $A_{i,i'}$

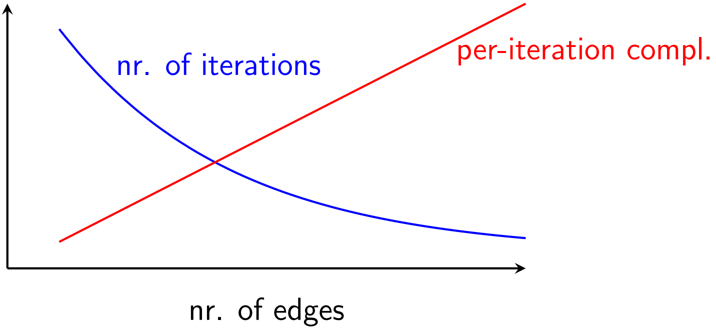
Connectivity measured by λ_2



- FL faster for $\mathcal{G}$ with large $\lambda_2(\mathcal{G})$
- for given total number of edges (per-iteration complexity), place them in order to maximize $\lambda_2(\mathcal{G})$

Computational Aspects

- FL algorithms operate by iterative message passing
- each edge adds compute/comm. per-iteration
- adding edges can speed up convergence (reducing nr. of iterations)



Statistical Aspects

Consider an FL network with nodes $i = 1, \ldots, n$, each generating the data $\mathcal{D}^{(i)}$ and training the model $\mathcal{H}^{(i)}$

- edge $\{i, i'\}$ forces similar trained models at i, i'
- detrimental if $\mathcal{D}^{(i)}, \mathcal{D}^{(i')}$ have different distributions
- place edges only between "statistically similar" nodes i, i'
- how to measure statistical similarity between nodes i, i' ?

Measuring Statistical Similarity

- Consider the local (weather) dataset $\mathcal{D}^{(i)}$

Time	**Air Temperature**
2025-01-13 16:08:00	-1.5
2025-01-13 16:09:00	-1.5
2025-01-13 16:10:00	-1.4
2025-01-13 16:11:00	-1.5
2025-01-13 16:12:00	-1.5

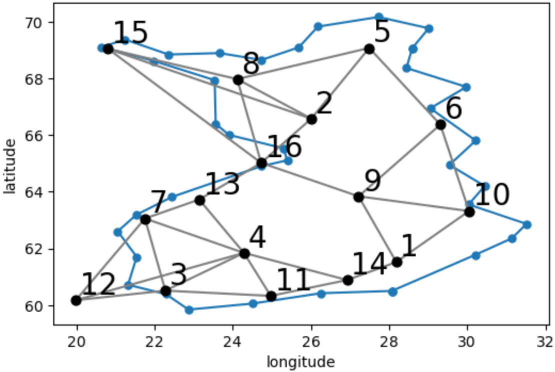
- interpret data as (a realization) of a random process with parametric prob. distr. $p(\mathcal{D}^{(i)}; \boldsymbol{\theta})$
- compute parameter estimate $\widehat{\boldsymbol{\theta}}^{(i)}$ from $\mathcal{D}^{(i)}$
- measure similarity between i, i' via $\left\| \widehat{\boldsymbol{\theta}}^{(i)} - \widehat{\boldsymbol{\theta}}^{(i')} \right\|$

Measuring Statistical Similarity (ctd.)

- parameter estimator $\widehat{\boldsymbol{\theta}}^{(i)}$ is only one example of a vector representation $\mathbf{z}^{(i)} \in \mathbb{R}^k$ for $\mathcal{D}^{(i)}$
- place edges between nearest neighb. using $\left\|\mathbf{z}^{(i)} - \mathbf{z}^{(i')}\right\|$
- many other constructions for vector $\mathbf{z}^{(i)}$ exist, e.g.,
  - for FMI stations, could use $\mathbf{z}^{(i)} := \big(\text{latitude}, \text{longitude}\big)^T$
  - use gradient $\mathbf{z}^{(i)} := \nabla L_i(\mathbf{w})$
  - construct $\mathbf{z}^{(i)}$ by auto-encoder (learnt embedding)

Example: Using Lat/Lon. for Similarity

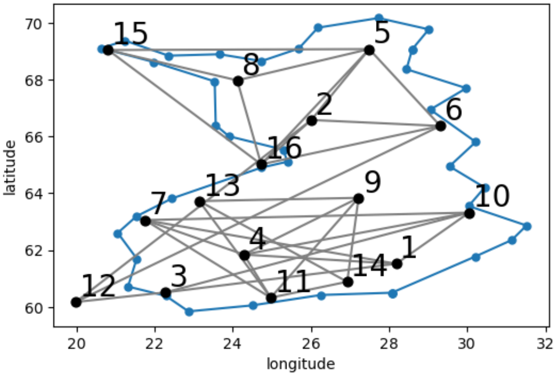
nodes=FMI stations, nearest neighb. graph using lat./lon.



Python script for reproducing the Fig.:

Example: Using Avg. Temp. for Similarity

place edges based on avg. temp during 2024-05-15



Python script for reproducing the Fig.:

What's Next?

The next module formulates FL as an optimization problem defined over an FL network.

Later modules use FL networks for the design and analysis of FL systems.