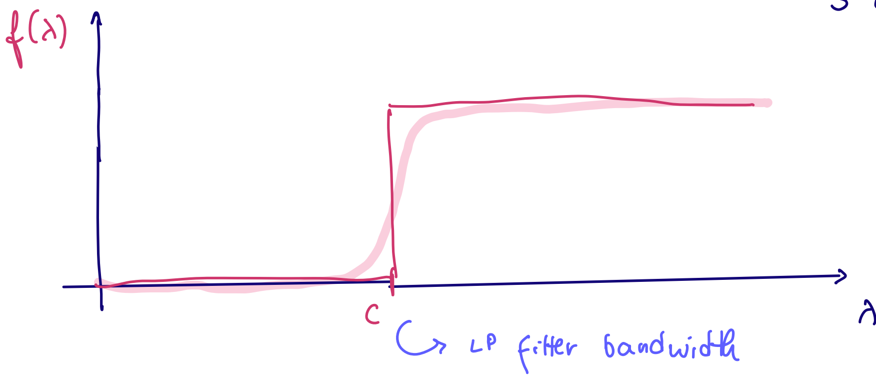
Topics:
- filter design
- spectral filters
- filter learning

- statistical learning & ERM
- types of learning problems on graphs

E.g.: we want to design a lowpass filter

$S = L$



Is this function analytic?

But we can often find good analytic approximations of non-analytic functions,

For Heaviside functions such as the LPF, a good approx. is the logistic function:

$$\tilde{f}(\lambda) = \frac{1}{1+e^{-\alpha(\lambda-c)}}$$

steepness

$$\tilde{f}(0) = \frac{1}{1+e^{\alpha c}} \qquad \tilde{f}'(\lambda) = \frac{+1}{\left(1+e^{-\alpha(\lambda-c)}\right)^2}\, e^{-\alpha(\lambda-c)} \cdot +\alpha$$

$$\tilde{f}'(0) = \frac{\alpha e^{\alpha c}}{(1+e^{\alpha c})^2} \quad \ldots$$

$$\tilde{f}''(\lambda) = \frac{-\alpha^2 e^{-\alpha(\lambda-c)}}{(1+e^{-\alpha(\lambda-c)})^2} + \frac{2\alpha e^{-2\alpha(\lambda-c)}(-\alpha)}{(1+e^{-\alpha(\lambda-c)})^3}$$

$$\tilde{f}''(0) = \frac{\alpha^2 e^{\alpha c}}{(1+e^{\alpha c})^2}\left(\frac{2e^{\alpha c}}{1+e^{\alpha c}} - 1\right)$$

$$h_0 = \tilde{f}(0) \qquad h_1 = \tilde{f}'(0) \qquad h_2 = \frac{\tilde{f}''(0)}{2} \qquad \dots$$

We can write <u>any</u> analytic fcn as a graph conv.

Is there an easier way to design such a filter?

▶ Spectral graph filters

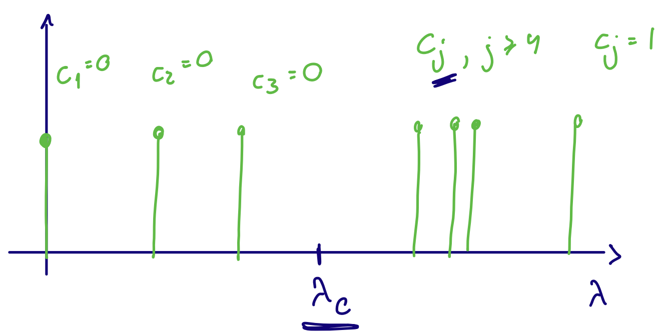
$S = V\Lambda V^H$

$\hat{x} = V^H x$

(DEF) $$y = \sum_{j=1}^{n} \underline{c_j} [\hat{x}]_j v_j$$

↳ we design or learn these coeffs.

E.g.: Let $S = L$ with spectra:



$\hat{j}_c = 4$

Suppose we want a LP filter with bandwidth λ_c

Often, such filters are designed not based on an eigenvalue threshold λ_c but on an index threshold j_c, e.g. $j=3$ above.

In modern applications we have moved away from system engineering to learning systems from data.

▶ The [supervised] statistical learning problem

x & y are assumed to be related by a statistical model $p(x,y)$

↳ We want to predict y from x with the conditional dist'n $y \sim p(y|x)$ (stochastic outputs; think VAEs, diffusion models, ...)

↳ We want to predict y from x with the conditional expectation $y = E(y|x)$ (deterministic outputs; classical reg./supervised learning)

In practice we can only estimate these quantities, using a model $\tilde{y} = f(x)$, $f \in \mathcal{F}$. f comes from a function or hypothesis class $\mathcal{F}$

E.g.: $\mathcal{F} = \{ f(x) = ax + b \mid a, b \in \mathbb{R} \}$

How to pick the best estimator f?

- Define a loss function $\ell(y, \hat{y})$ measuring the "cost" of predicting $z = f(x)$ when the output is y

- Minimize the expected loss over distribution $p(x,y)$:

$$f^* = \underset{f \in \mathcal{F}}{\operatorname{argmin}} \ \mathbb{E}_{p(x,y)} \{ \ell(y, \overset{z}{f(x)}) \}$$

→ this is the statistical risk minimization problem

The optimal estimator is the fcn f with min. expected cost over all $f \in \mathcal{F}$

Empirical Risk Minimization

In practice, we don't have access to the dist'n $p(x,y)$; only to samples $\mathcal{T} = \{(x^j, y^j)\}_{j=1}^M$

$$f^* = \underset{f \in \mathcal{F}}{\operatorname{argmin}} \ \frac{1}{M} \sum_{j=1}^{M} \ell(y^j, f(x^j))$$

↳ this is the empirical risk minimization (ERM) problem

↳ we're minimizing the empirical mean

Typical losses are the quadratic / L^2 loss $\ell(y,z) = \frac{\|y-z\|^2}{2}$ for regression / estimation problems and the

0-1 loss $\ell(y,z) = 1 . \mathbb{I}(y \neq z)$ for classification problems

↳ indicator fcn

(or its differentiable surrogates such as cross entropy, logistic losses)

The ERM problem might have a closed-form solution (like in linear regression), but in modern ML, it is solved using optimization algorithms such as SGD or ADAM.

↳ look up the depts. NL optim. & optim. for DS classes, an cvx optim. in ECE!

Back to filter learning!

Where do our filters fit into, in ERM?

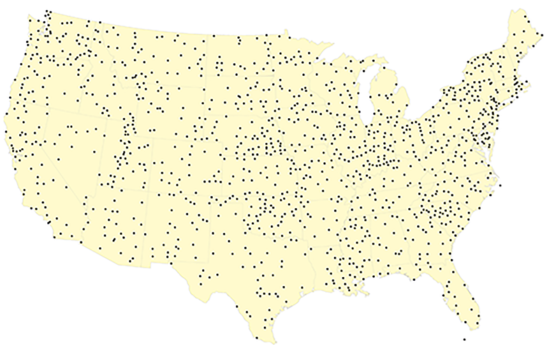
↳ They form the hypothesis class $\mathcal{F}$

We see primarily 3 types of learning problems on graphs:

① graph signal processing problems:

The graph G is the data support, fixed.
Data $x \in \mathbb{R}^n$, $y \in \mathbb{R}^n$. Assuming $x, y \sim p(x,y)$, we regress signals y on predictor signals x

E.g.:



G is US weather station network (edges encode geographic proximity)

$S = A$

y are recorded temperatures today / same day last year / " " 2 yrs ago... $y^1 \in \mathbb{R}^n$ $y^2 \in \mathbb{R}^n$ $y^3 \in \mathbb{R}^n$ $\dots y^M \in \mathbb{R}^n$

x are recorded temperatures 3 months prior to today / 15 " " " " / 27 " " " " ... $x^1 \in \mathbb{R}^n$ $x^2 \in \mathbb{R}^n$ $x^3 \in \mathbb{R}^n$ $\vdots$ $x^M \in \mathbb{R}^n$

Goal: predict February temps. from November temps. YoY

Hypothesis class: graph convs. $\mathcal{F} = \{ z = \sum_{k=0}^{K-1} h_k S^k x,\ h_k \in \mathbb{R} \}$

Problem: $$\min_{\underline{h_k}} \frac{1}{M} \sum_{j=1}^{M} \| y^j - \sum_{k=0}^{K-1} h_k S^k x^j \|_2^2$$

Application: temperature forecasting

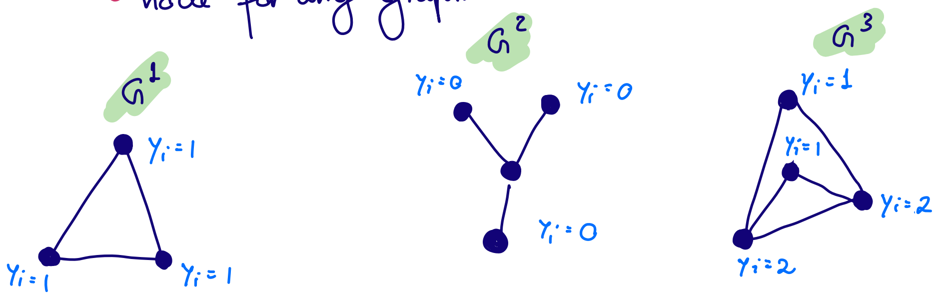
predict Feb. 2026 temps (y') from Nov. 2025 temps (x') as $y' = \sum_{k=0}^{K-1} h_k^* S^k x'$

2) Graph-level problems

In these, each graph G represents a predictor (there are multiple Gs) associated w/ an obsn $y \in \mathcal{Y}$. Assume $G, y \sim p(G, y)$, we regres y on G

E.g.:

Goal: predict the number of Δ's incident to each node for any graph.



Hypothesis class: $\mathcal{J} = \left\{ f(S) = \sum_{k=0}^{K-1} h_k S^k \mathbb{1} \mid h_k \in \mathbb{R} \right\}$ $\quad S = A$

since there are no graph signal obsns

Problem: $\min_{h_k} \frac{1}{M} \sum_{i=1}^{M} \ell\left(\sum_{k=0}^{K-1} h_k S^k \mathbb{1}, y \right) \rightarrow \ell$ is a surrogate of 0-1 loss (e.g. cross ent.)

Application: automate triangle counting

Obs.: Both (1) & (2) are supervised learning problems sometimes called transductive learning problems = none of the test inputs are seen at training time.

(3) Node-level tasks (or inductive learning or semi-supervised learning)

The graph G is once again the data support.

each $[x]_i, [y]_i \sim p(x,y)$. I.e., each node is treated as a sample. We assume we only observe $[y]_i$ for a node set $\mathcal{J} \subset \mathcal{V}$ and estimate $[y]_j$ for $j \in \mathcal{V} \setminus \mathcal{J}$ from $[x]_i$, $i \in \mathcal{V}$

E.g.: consider the contextual SBM:

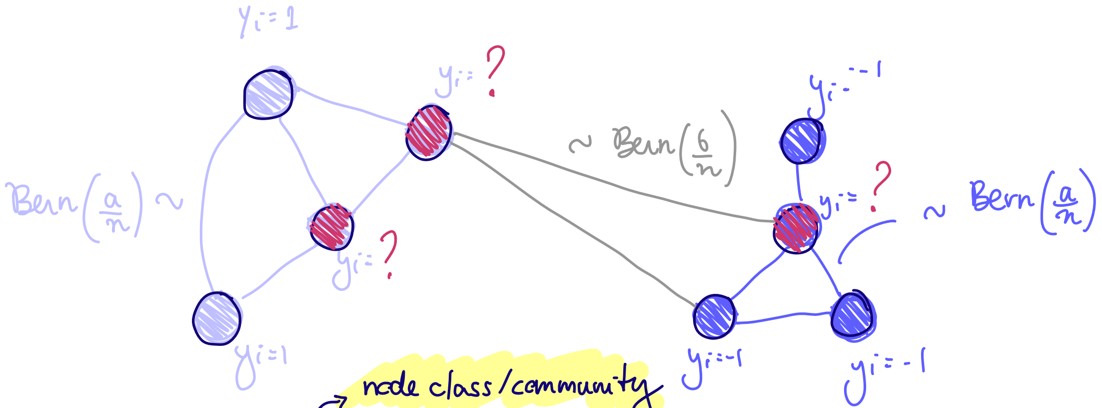
$G = (\mathcal{V}, \mathcal{E})$, undirected ; $\quad y \in \{-1, 1\}^n$

$$P(A_{ij} = 1) = P((i,j) \in \mathcal{E}) = \begin{cases} a/n & \text{if } y_i = y_j \\ b/n & \text{o.w.} \end{cases}$$

with node features / covariates $\begin{cases} x_i = \sqrt{\frac{\mu}{n}}\, y_i \cdot u + z_i \\ u \sim \mathcal{N}(0,1) \\ z \sim \mathcal{N}(0, I_n) \end{cases}$

Conditioned on u & y_i,

$(\mu = 1)$ $\boxed{x_i \sim \mathcal{N}\left(\sqrt{\tfrac{1}{n}}\,u,\ 1\right)}$ $\boxed{x_i \sim \mathcal{N}\left(-\sqrt{\tfrac{1}{n}}\,u,\ 1\right)}$



Goal: predict $\overbrace{y_i,\ i \in \mathcal{V}\setminus\mathcal{J}}^{\text{node class/community}}$ from $x_i,\ i \in \mathcal{V}$

Hypothesis class: graph convs. $\mathcal{J} = \left\{ z = \sum_{k=0}^{K-1} h_k S^k x,\ h_k \in \mathbb{R} \right\}$

Problem: Define a mask $M_{\mathcal{J}} \in \{0,1\}^{|\mathcal{J}| \times n}$, $M_{\mathcal{J}} \mathbb{1}_n = \mathbb{1}_{|\mathcal{J}|}$
$\mathbb{1}^T M_{\mathcal{J}} = \mathbb{1}^T$

$$\min_{h_k} \frac{1}{|\mathcal{J}|} \ell\left(M_{\mathcal{J}} y,\ M_{\mathcal{J}} \sum_{k=0}^{K-1} h_k S^k x\right)$$

some surrogate of 0-1 loss

Application: infer node's class / community / identity locally
i.e., without needing comm. det. / clustering techniques, which

require eigenvectors (global graph information)

Obs.: This problem is an example of inductive learning, because the test data ($j \in \mathcal{V} \setminus \mathcal{J}$) predictors are seen at training time