Dynamics and symmetries in neural network learning

Stefanie Jegelka
MIT

based on joint work with
Nisha Chandramoorthy, Khashayar Gatmiry, Andreas Loukas,
Derek Lim, Joshua Robinson, Lingxiao Zhao, Haggai Maron, Tess Smidt, Suvrit Sra

Neural network learning

1. Neural network generalization without convergence of weights
   (what do the neural network training dynamics tell us about generalization?)
   (Chandramoorthy, Gatmiry, Loukas, Jegelka NeurIPS 2022)

2. Modeling symmetries in learned functions
   (Lim, Robinson, Zhao, Maron, Smidt, Sra, Jegelka ICLR 2023)

Convergence in NN training

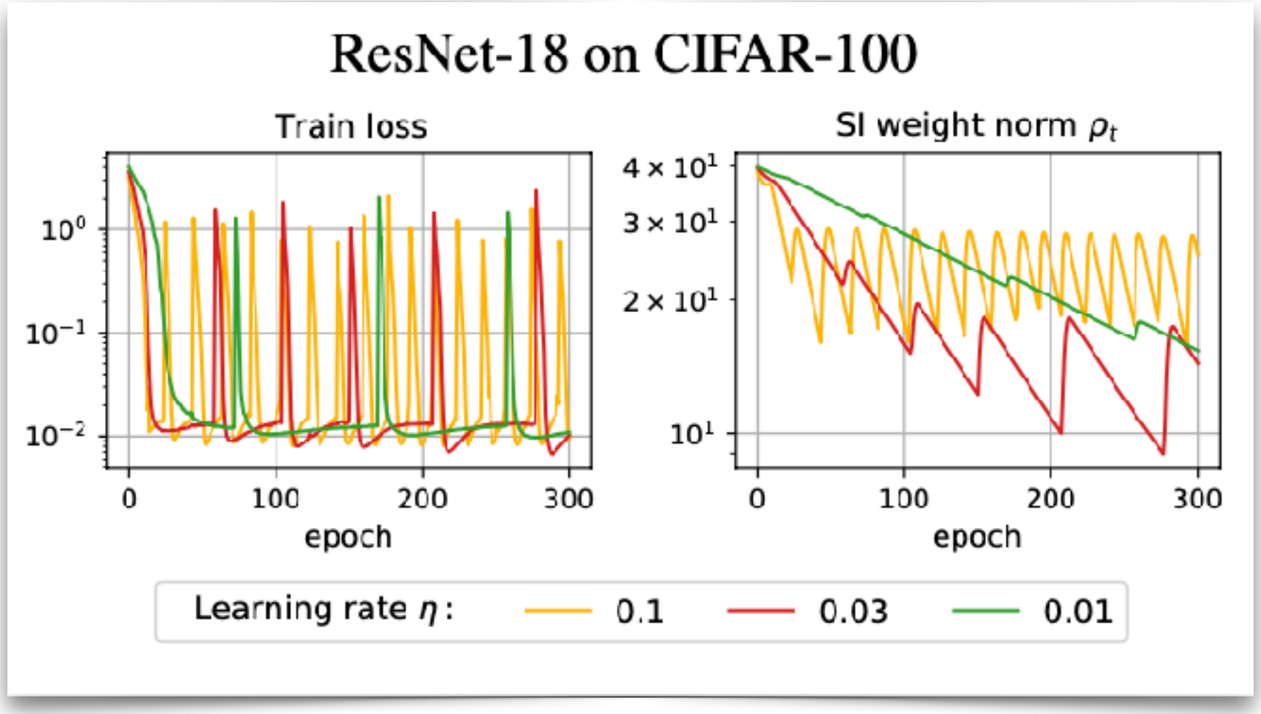
- frequent assumption: **NN parameters converge to a stationary point**

- But, this may not necessarily be the case!
  (Kong & Tao 2020, Li et al 2020, Kunin et al 2021, Cohen et al 2021, Lobacheva et al 2021, Bhojanapalli et al 2021, Zhang et al 2022, Wang et al 2022)

- Training is linearly unstable: small perturbations change learned weights
  (Cohen et al 2021, Ahn et al 2022, Zhang et al 2021, Garipov et al 2018)

Non-convergence in NN training

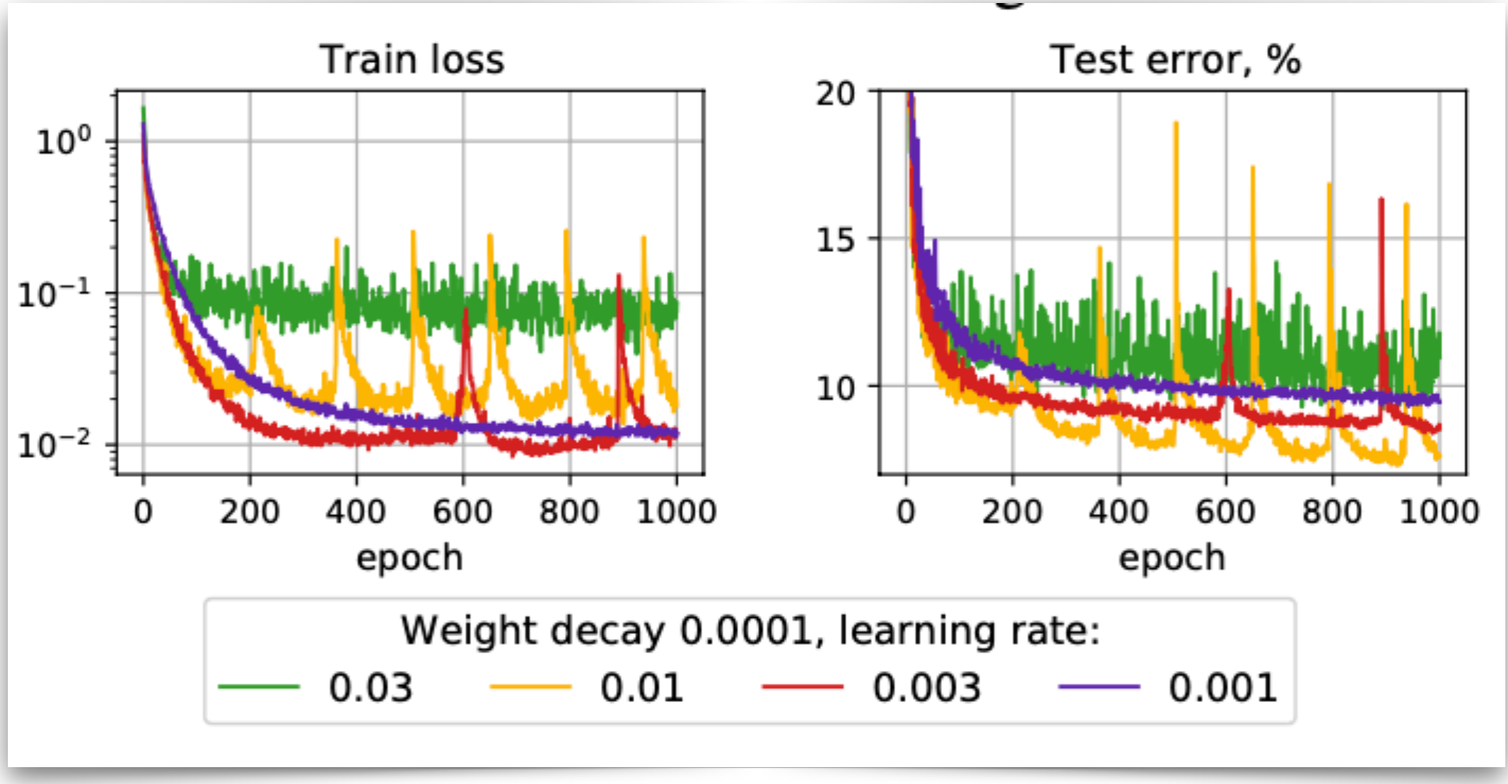
- Batch normalization + weight decay can lead to periodic behavior
  (Lobacheva, Kodryan, Chirkova, Malinin, Vetrov, 2021)



"minima achieved at two neighboring training periods are substantially different, but their similarity is usually higher than that of two independently trained networks."

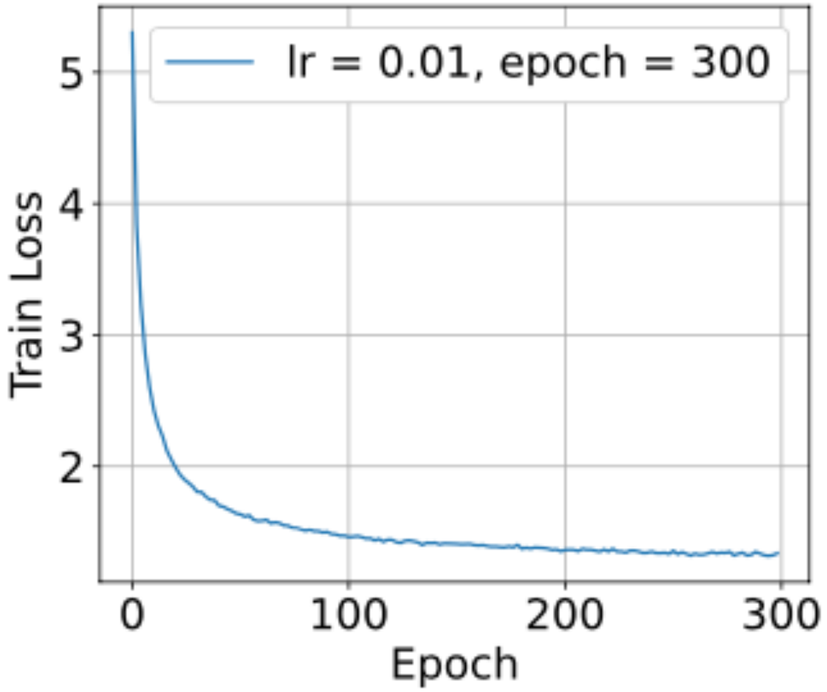
Non-convergence in NN training

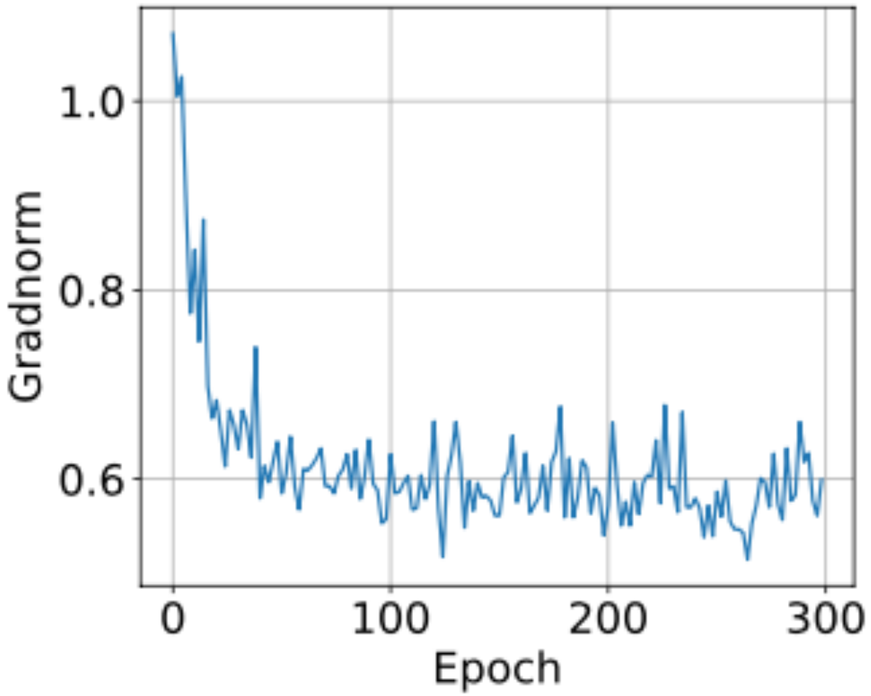
- Batch normalization + weight decay can lead to periodic behavior
  (Lobacheva, Kodryan, Chirkova, Malinin, Vetrov, 2021)



Non-convergence in NN training

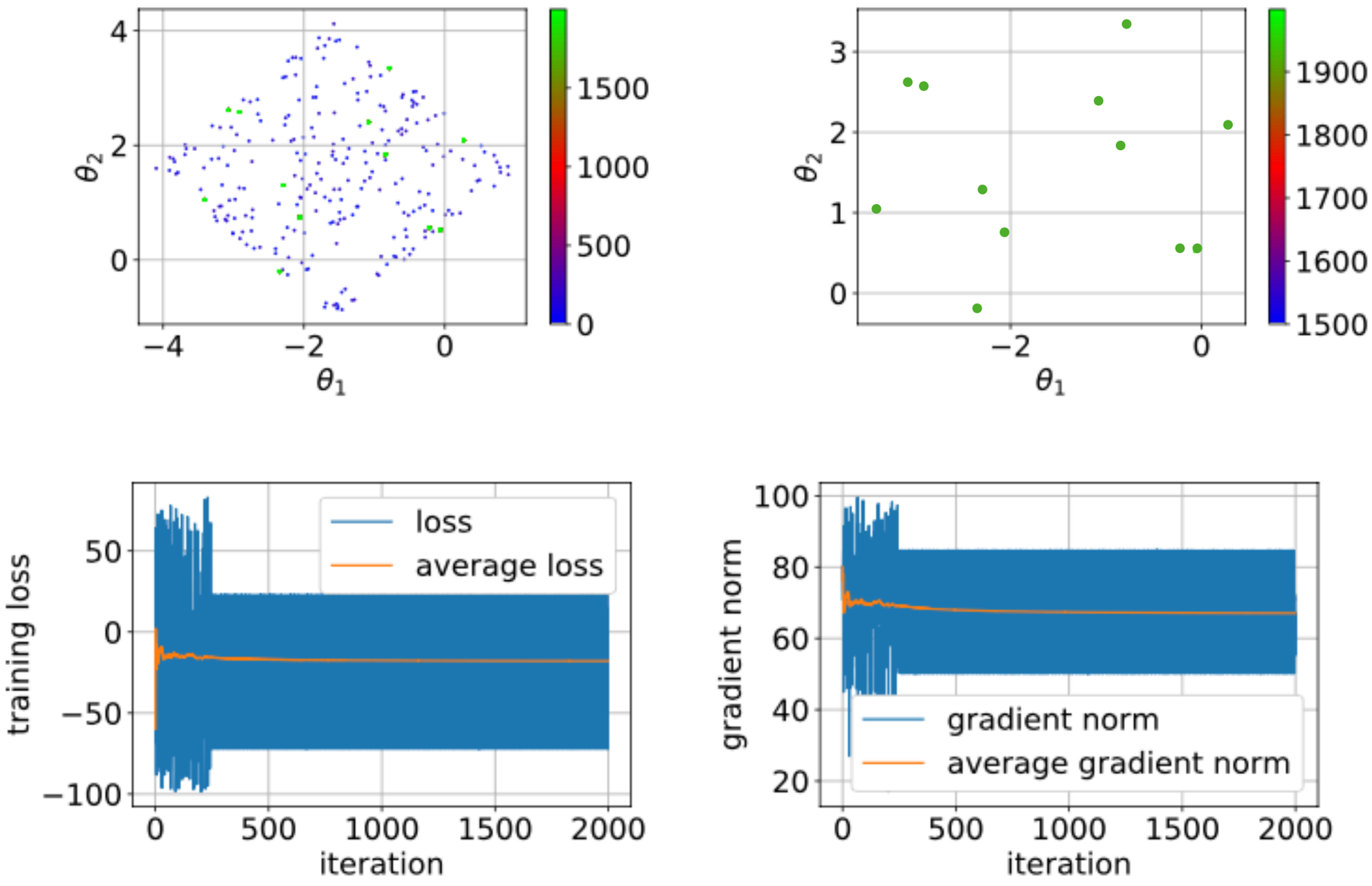
- *"we observe that even though the weights do not converge to stationary points, the progress in minimizing the loss function halts and training loss stabilizes"* *(Zhang, Li, Sra, Jadbabaie 2022)*





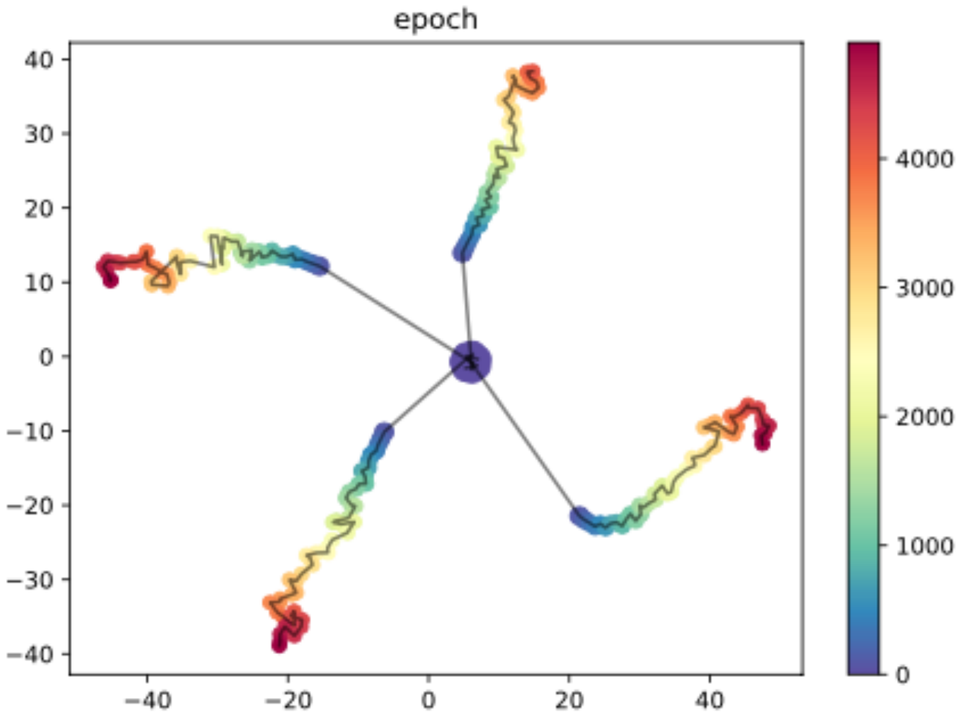
ImageNet experiments (similar for TransformerXL)

Synthetic example

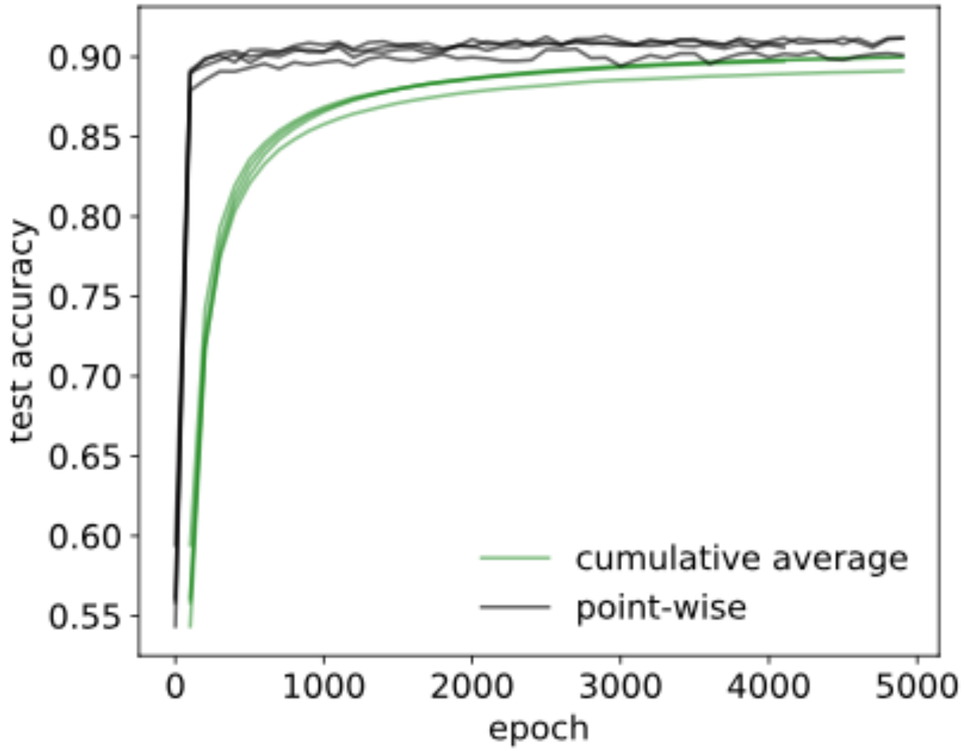


(Figure: Zhang et al 2022)

Non-ergodicity



2nd layer weight trajectories for different SGD runs
(VGG16, CIFAR10)



(S)GD as a nonlinear dynamical system

- Training data $S = \{z_1, \ldots, z_n\};\ z_i = (x_i, y_i)$
- learn hypothesis $h : \mathbb{R}^d \times \mathbb{W} \to \mathcal{Y};\ h(\cdot, w)$

- (S)GD updates:

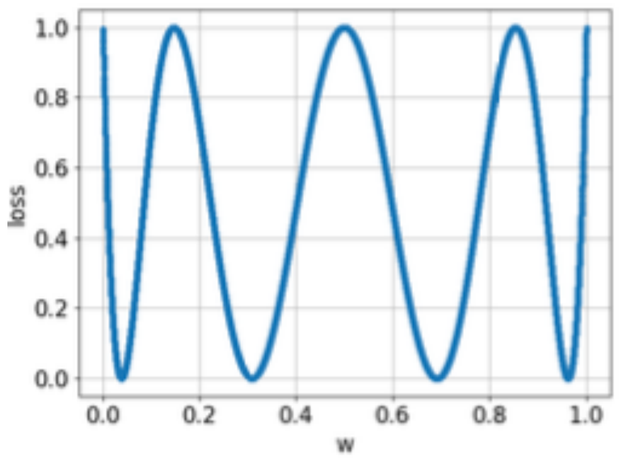
$$w_{t+1} = \phi_S(w_t) := w_t - \eta_t \widehat{\nabla} L_S(w_t)$$

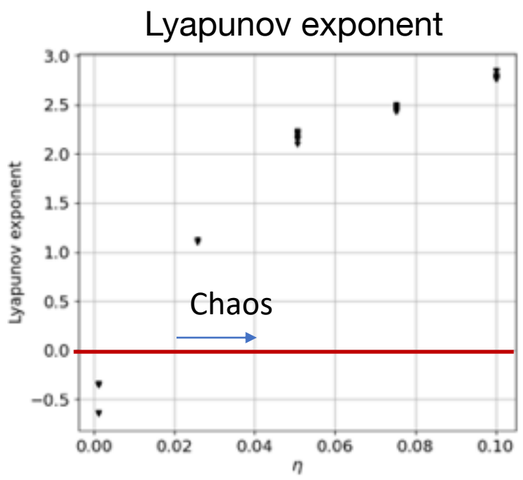
(S)GD as a nonlinear dynamical system

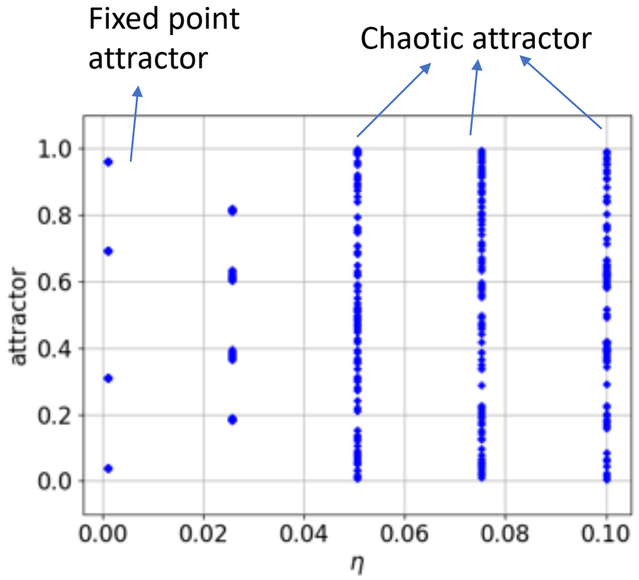
- Training data $S = \{z_1, \ldots, z_n\};\ z_i = (x_i, y_i)$
- learn hypothesis $h : \mathbb{R}^d \times \mathbb{W} \to \mathcal{Y};\ h(\cdot, w)$

- (S)GD updates: $w_{t+1} = \phi_S(w_t) := w_t - \eta_t \widehat{\nabla} L_S(w_t)$

Toy example: dynamics as function of (constant) step size







Generalization?

Generalization: algorithmic stability?

- Samples S, S' differ in exactly 1 datapoint. Result in weights $w_S^*, w_{S'}^*$

- Learning algorithm is algorithmically stable with stability coefficient β if

$$\beta = \sup \left\{ |\ell(z, w_S^*) - \ell(z, w_{S'}^*)| \ : \ z \in \mathbb{R}^d \times \mathbb{R} \right\}$$

- Stability implies generalization *(Bousquet & Elisseef 2002, Rakhlin, 2006, Kuzborskij and Lampert, 2018, Feldman & Vondrák 2018, Bousquet et al., 2020, Zhang et al., 2021, …)*

Generalization: algorithmic stability?

- Samples S, S' differ in exactly 1 datapoint. Result in weights $w_S^*, w_{S'}^*$

- Learning algorithm is algorithmically stable with stability coefficient β if

$$\beta = \sup \left\{ |\ell(z, w_S^*) - \ell(z, w_{S'}^*)| \ : \ z \in \mathbb{R}^d \times \mathbb{R} \right\}$$

- Stability implies generalization *(Bousquet & Elisseef 2002, Rakhlin, 2006, Kuzborskij and Lampert, 2018, Feldman & Vondrák 2018, Bousquet et al., 2020, Zhang et al., 2021, …)*

✘ not applicable here…

Questions

- Generalization analysis without convergence to stationary point (limit cycles, quasi periodic orbits, chaotic orbits)?

- Can dynamical information help determine if local descent algorithms generalize?

Look at average statistics over time

evolution of probability measures

Invariant measure and ergodicity

$$w_{t+1} = \phi_S(w_t) := w_t - \eta_t \widehat{\nabla} L_S(w_t)$$

- Long-term behavior instead of pointwise/local dynamics
- Invariant measure: for any set A,

$$\mu(A) = \mu(\phi^{-1}(A))$$

Invariant measure and ergodicity

$$w_{t+1} = \phi_S(w_t) := w_t - \eta_t \widehat{\nabla} L_S(w_t)$$

- Long-term behavior instead of pointwise/local dynamics
- Invariant measure: for any set A,

$$\mu(A) = \mu(\phi^{-1}(A))$$

- Classical result: for continuous dynamics on compact set, there exists at least one ergodic, invariant measure

Ergodicity: for μ_S-almost every initial state w_0, continuous f, as $T \to \infty$

$$\frac{1}{T}\sum_{t=0}^{T} f(w_t) \to \mathbb{E}_{w\sim\mu}[f(w)]$$

Invariant measure and ergodicity

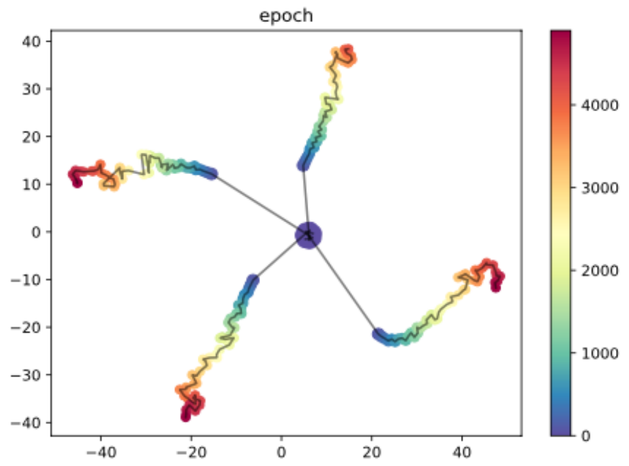
- **Invariant measure**: for any set A,

$$\mu(A) = \mu(\phi^{-1}(A))$$

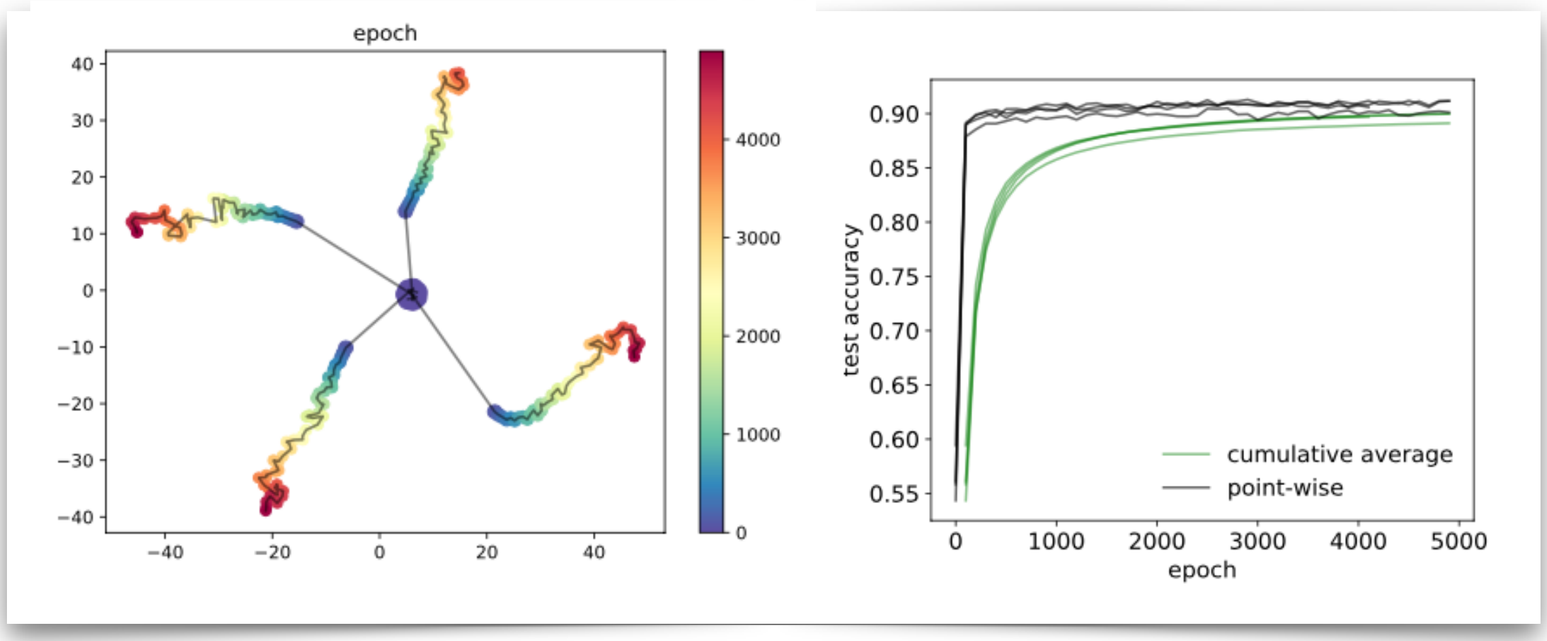
- Classical result: for continuous dynamics on compact set, there exists at least one ergodic, invariant measure
  Ergodicity: for μ_S-almost every initial state w_0, continuous f, as $T \to \infty$

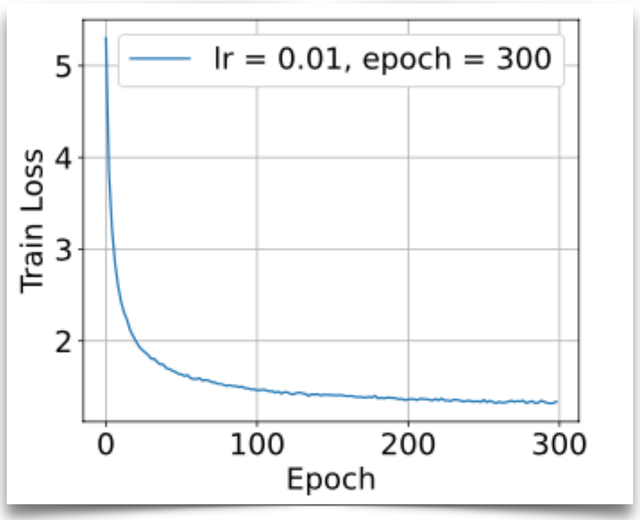
$$\frac{1}{T}\sum_{t=0}^{T} f(w_t) \to \mathbb{E}_{w\sim\mu}[f(w)]$$

- There may be infinitely many invariant measures in weight space!



But…


epoch
40
30
20
10
0
−10
−20
−30
−40
−40
−20
0
20
40
4000
3000
2000
1000
0
test accuracy
0.90
0.85
0.80
0.75
0.70
0.65
0.60
0.55
0
1000
2000
3000
4000
5000
epoch
cumulative average
point-wise


lr = 0.01, epoch = 300
Train Loss
5
4
3
2
0
100
200
300
Epoch

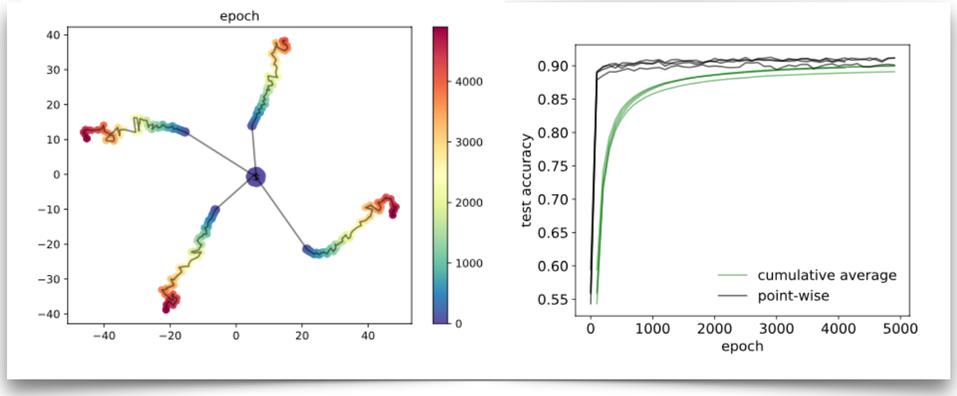
Instead: loss averages over time

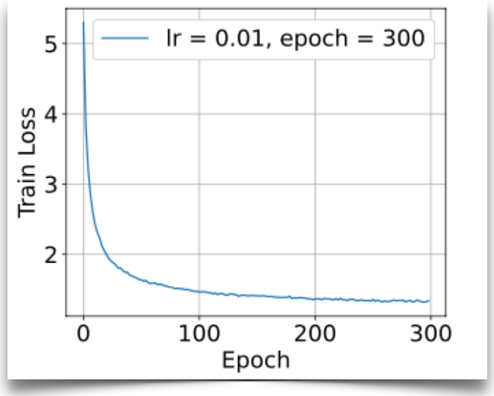
- Instead of weight space, look at loss values

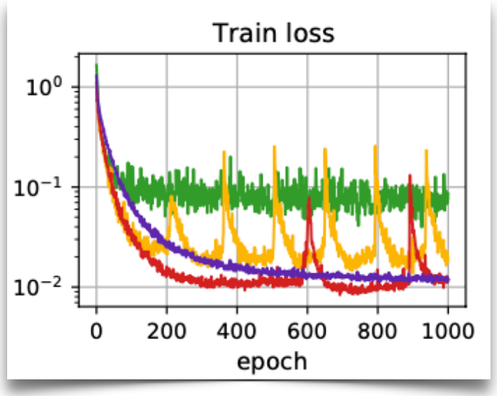
- **Assumption:** For any $S \sim \mathcal{D}^n$ there exists a map $z \to \langle \ell_z \rangle_S$ s.t. for all z

$$\lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} \ell(z, w_t) = \langle \ell_z \rangle_S \in \mathbb{R}$$

- weaker than unique ergodic measure on weight space or of hypotheses







Statistical algorithmic stability

- Samples S, S' differ in exactly 1 datapoint. Result in weights $w_S^*, w_{S'}^*$

- Learning algorithm is statistically algorithmically stable with stability coefficient β if

$$\cancel{\beta = \sup\left\{|\ell(z, w_S^*) - \ell(z, w_{S'}^*)| \,:\, z \in \mathbb{R}^d \times \mathbb{R}\right\}}$$

$$\beta = \sup\left\{|\langle \ell_z \rangle_S - \langle \ell_z \rangle_{S'}| \,:\, z \in \mathbb{R}^d \times \mathbb{R}\right\}$$

Statistical algorithmic stability

- Samples S, S' differ in exactly 1 datapoint. Result in weights $w_S^*, w_{S'}^*$

- Learning algorithm is statistically algorithmically stable with stability coefficient β if

$$\cancel{\beta = \sup\left\{|\ell(z, w_S^*) - \ell(z, w_{S'}^*)| \,:\, z \in \mathbb{R}^d \times \mathbb{R}\right\}}$$

$$\beta = \sup\left\{|\langle \ell_z \rangle_S - \langle \ell_z \rangle_{S'}| \,:\, z \in \mathbb{R}^d \times \mathbb{R}\right\}$$

- Does not need convergence of weights to fixed point
- Reduces to standard algorithmic stability if weights converge

Generalization

- Does statistical algorithmic stability (SAS) imply generalization?

- Ergodic averages for risks: $\widehat{R}_S = \frac{1}{n} \sum_{z \in S} \langle \ell_z \rangle_S$ $\qquad R_S = \mathbb{E}_{z \sim D} \langle \ell_z \rangle_S$

Generalization

- Does statistical algorithmic stability (SAS) imply generalization?

- Ergodic averages for risks: $\widehat{R}_S = \frac{1}{n}\sum_{z\in S}\langle \ell_z \rangle_S$ $\qquad R_S = \mathbb{E}_{z\sim D}\langle \ell_z \rangle_S$

Theorem: Generalization bound

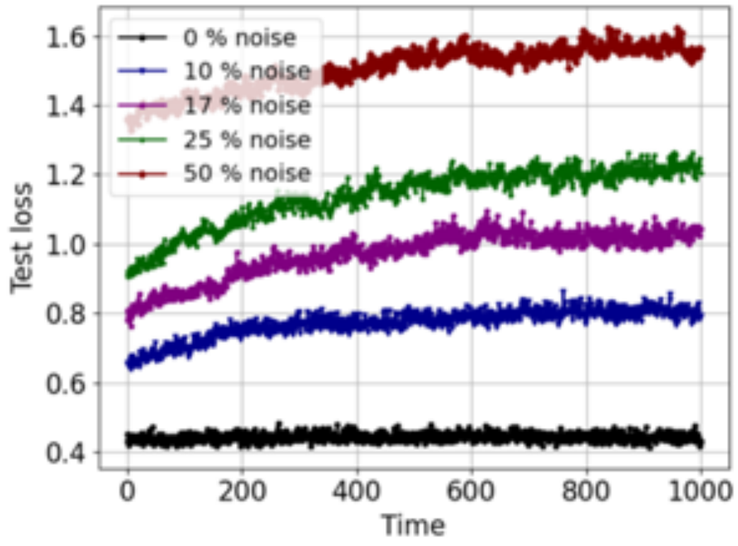
With probability $1-\delta$

$$R_S \leq \widehat{R}_S + \beta + 2(n\beta + L)\sqrt{\frac{\log(2/\delta)}{2n}}$$

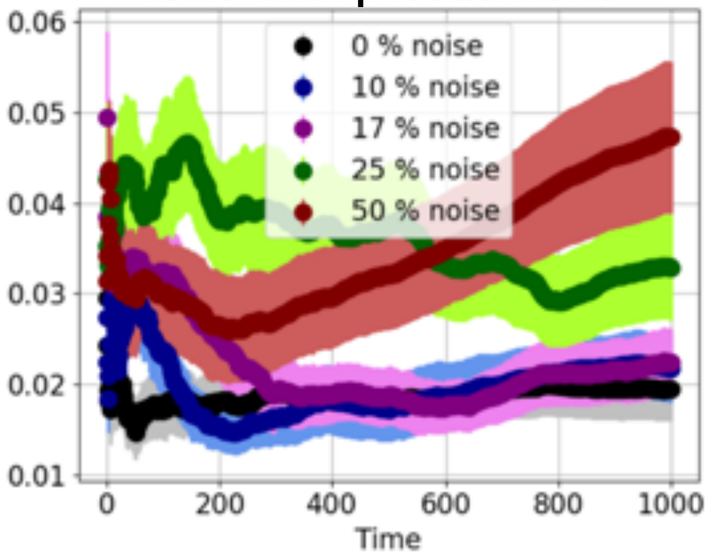
upper bound on loss

Empirical example

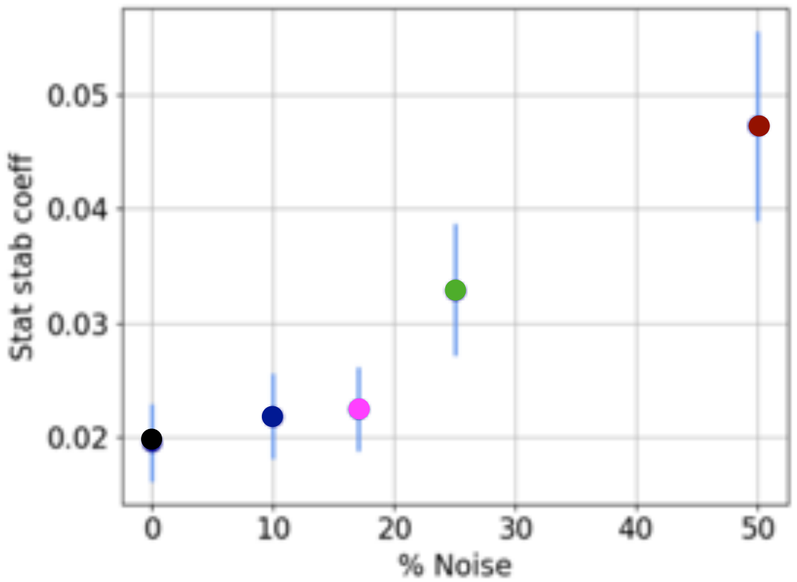
test loss



gap in cumulative loss
after data perturbation



lower bound on SAS coefficient



VGG16, CIFAR10

What makes an algorithm more stable?

- classical notion of stability and (S)GD *(Hardt, Recht, Singer 2016)*:

$$|\ell(z, w_t^S) - \ell(z, w_t^{S'})| \leq C \boxed{\|w_t^S - w_t^{S'}\|}$$

=> bound deviation of weights: e.g., few steps

- vacuous bounds for loss deviation at large times
- not applicable to non-converging weight trajectories
- not informative for time-independent SAS
- what if we look at stability of long-term behavior instead of single trajectories?

Statistical Algorithmic Stability and training behavior

- Look at image of weight distribution under loss, with mixing rate λ

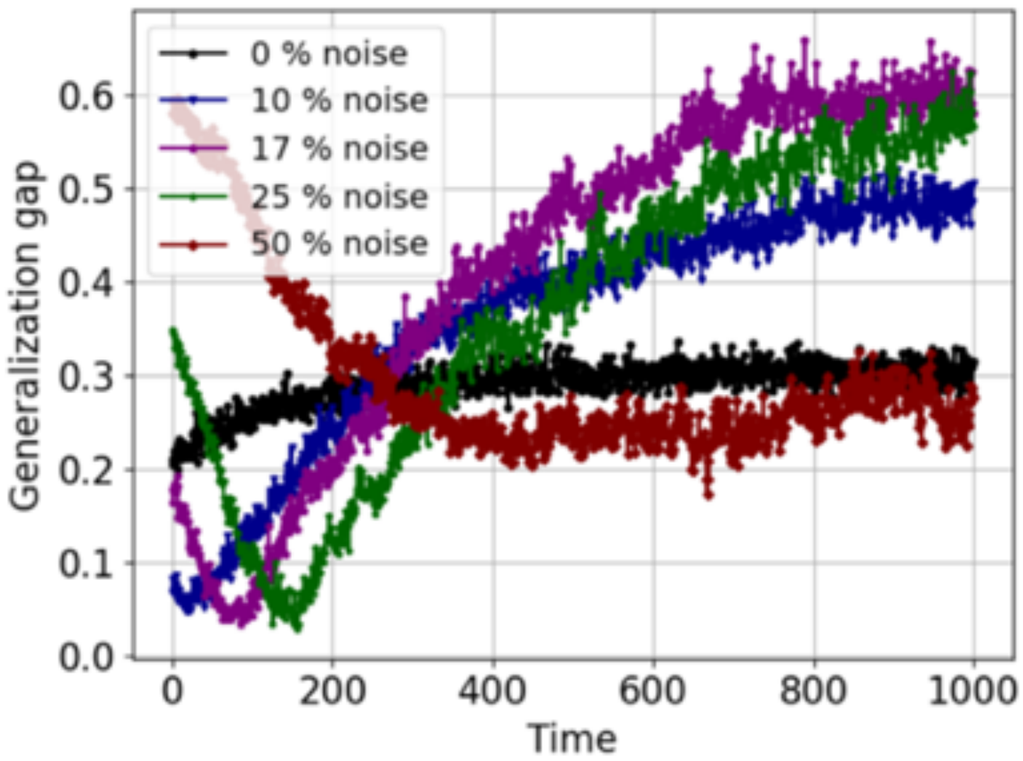
Theorem: faster convergence of loss implies better stability:

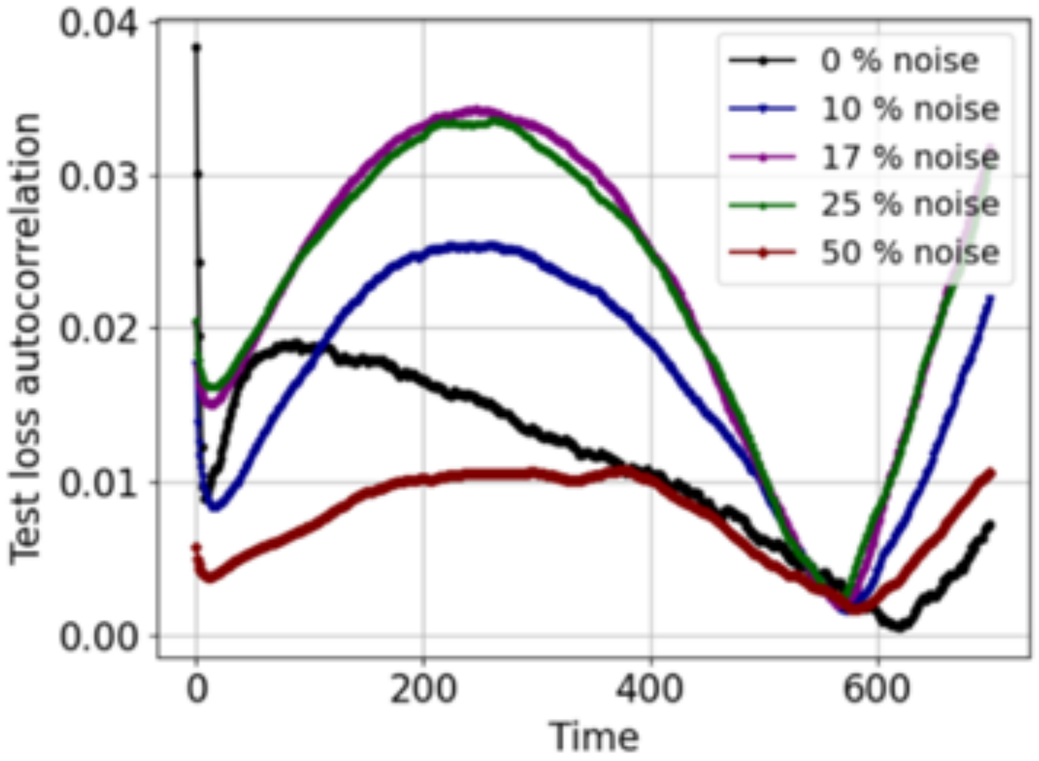
$$\beta = O\left(\frac{1}{n}\frac{L_D}{(1-\lambda)}\right)$$

Empirical example

Theorem: faster convergence of loss distribution implies better stability

proxy: autocorrelation





Neural network learning…

1. Neural network generalization without convergence of weights
   (Chandramoorthy, Gatmiry, Loukas, Jegelka NeurIPS 2022)

2. Modeling symmetries in learned functions
   (Lim, Robinson, Zhao, Maron, Smidt, Sra, Jegelka ICLR 2023)

Learning with invariances

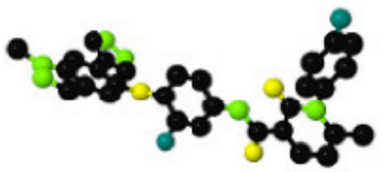
- **Standard learning setup:**
  given data $(x_1, y_1), \ldots, (x_n, y_n)$ estimate $\hat{f} \in \mathscr{F}$
  such that $\mathbb{E}[\ell(\hat{f}(X), Y)]$ is small

- **Learning with invariances:** all of the above, plus:
  select function $\hat{f} \in \mathscr{F}$ such that it is G-invariant for a given group G:

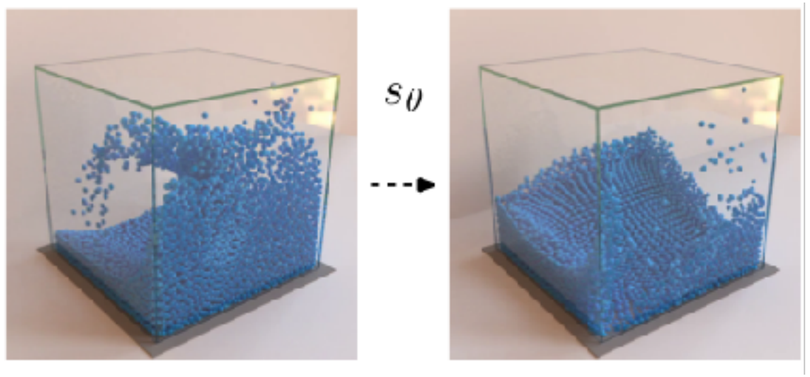
$$\hat{f}(g.x) = \hat{f}(x) \quad \forall g \in G, x \in \mathcal{X}$$

usually: $\mathscr{F}$ is a set of invariant functions

Machine Learning with Graph Data: Applications



molecule property prediction
(Duvenaud et al 2015, Stokes et al 2020)



learning simulations
(Sanchez-Gonzalez et al 2020)

guiding human intuition in mathematics
(Davies et al 2021)

ETA in Google Maps
(Derrow-Pinion et al, 2021)

drug interactions
(Zitnik et al 2018)

recommender systems
(Ying et al 2018)

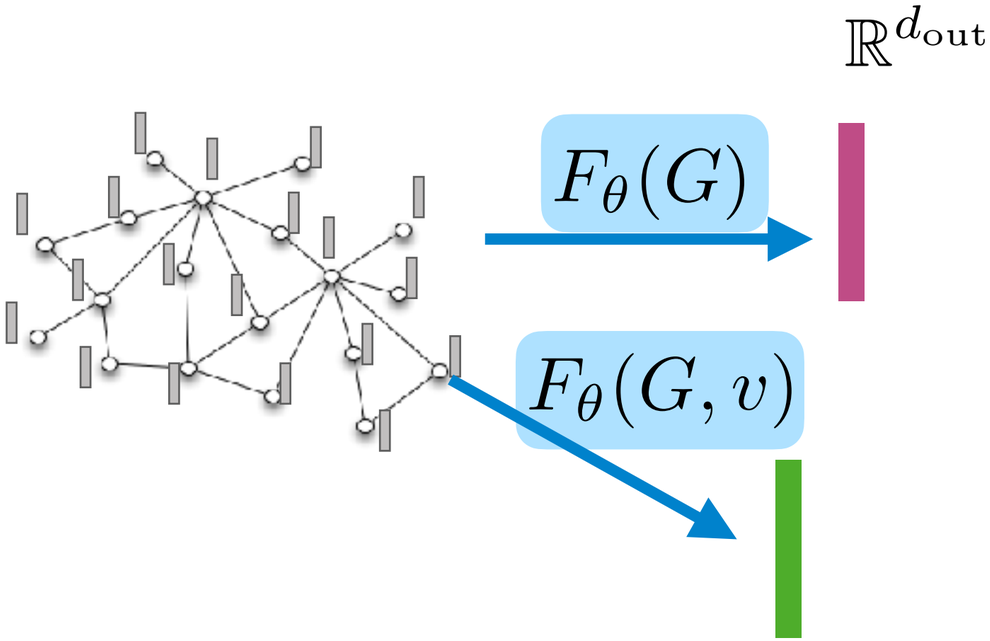
Machine Learning with Graph Data

- Data: **attributed graphs** (of bounded size)

$$G = (V, E, X, W) \quad \in \mathcal{G}$$

$\{x_v\}_{v \in V}$ $\quad$ $\{w(u,v)\}_{(u,v) \in E}$

$x_v \in \mathbb{R}^d$

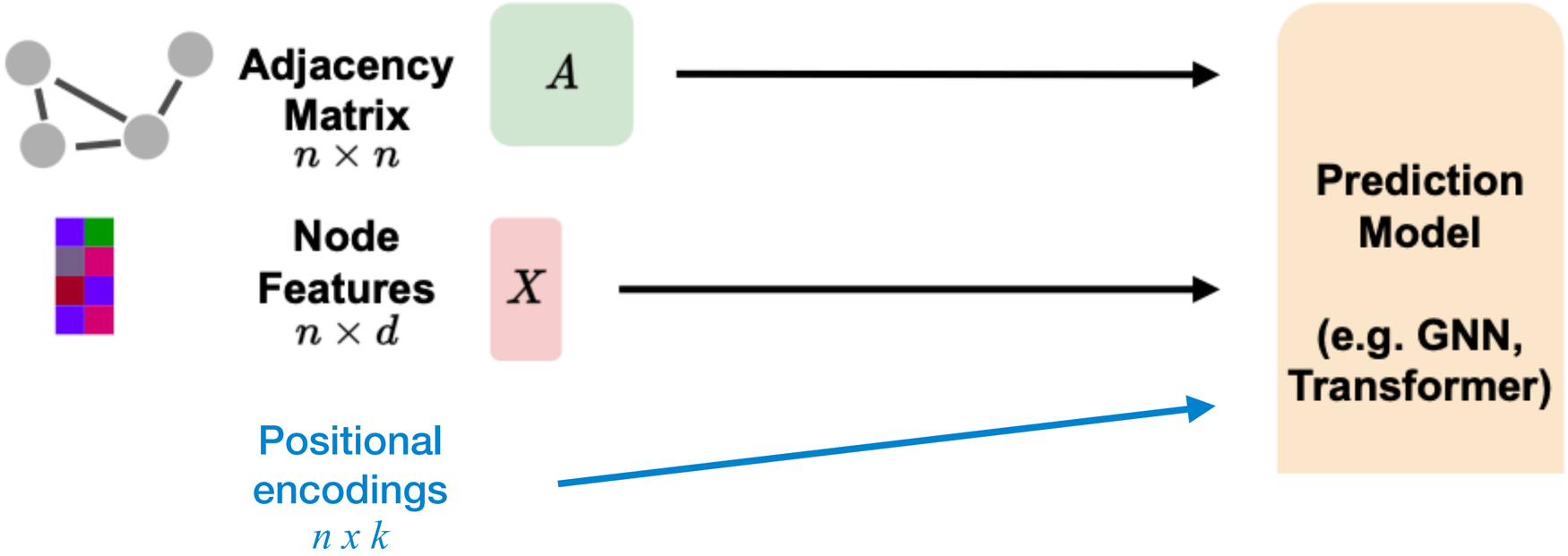


- Want: graph/node invariants

$$F_\theta(PAP^\top, PX) = F_\theta(A, X)$$ **Permutation invariance**

$$F_\theta(PAP^\top, PX, v) = F_\theta(A, X, v)$$ **Permutation equivariance**

Neural Networks for Graphs



- For good performance, (often) need more information: positional encodings

(Feldman et al 2022, Dwivedi et al 2022, Kreuzer et al 2021, Dwivedi & Bresson 2021, Mialon et al 2021)

Laplacian eigenvectors

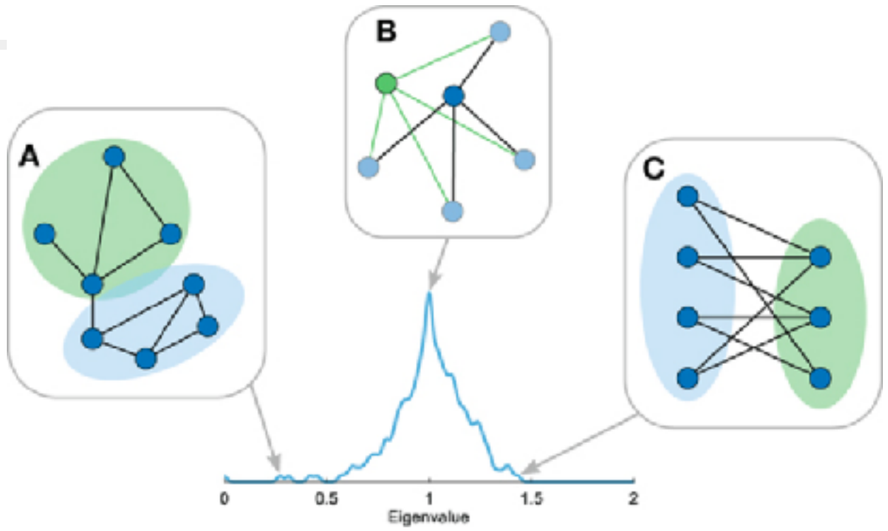
- Graph Laplacian:

$$L = I - D^{-1/2} A D^{-1/2}$$

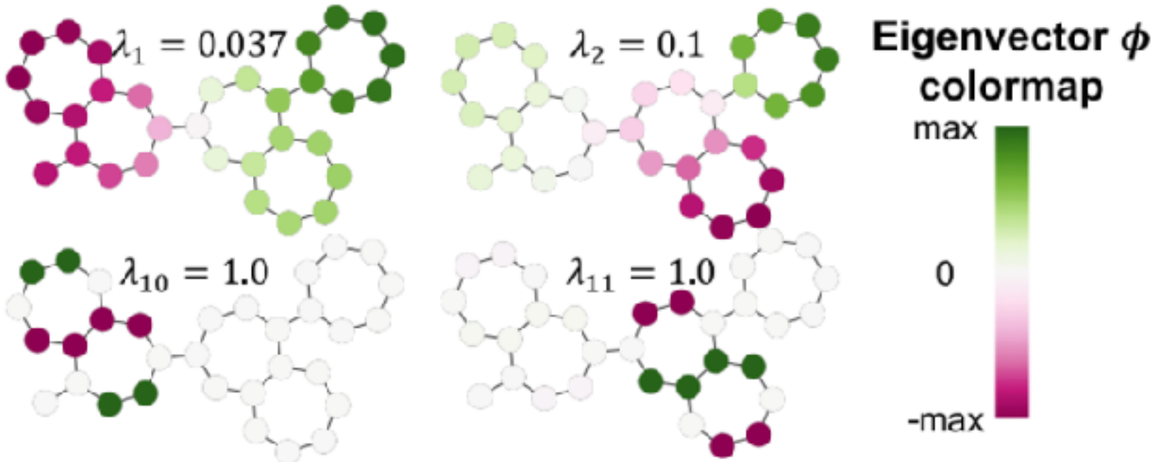
- eigenvalues $\lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$

  eigenvectors $v_1, \ldots, v_n$

- Captures distances, local structures, etc.

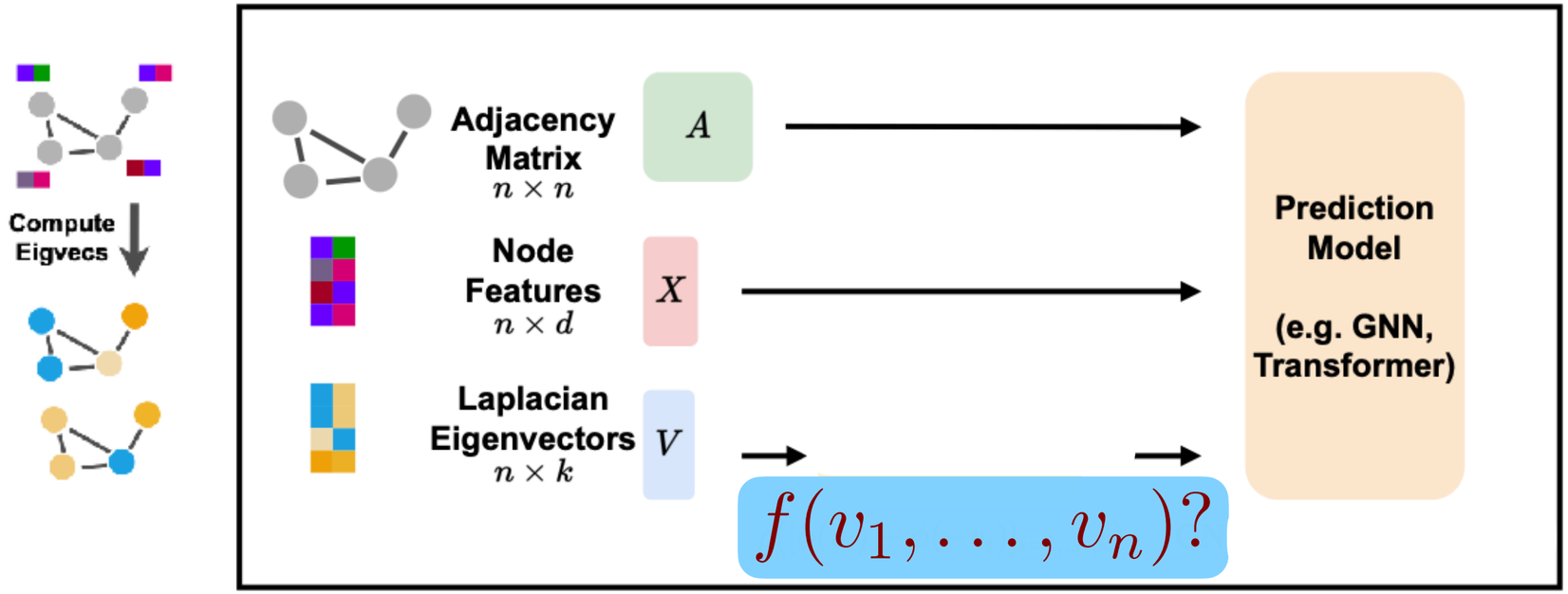


(de Lange, de Reus, van den Heuvel 2014)



(Kreuzer, Beaini, Hamilton, Létourneau, Tossou 2021)

Functions on eigenvectors



Can we learn an arbitrary function on a set of eigenvectors (and eigenvalues)?

What invariances must f have?

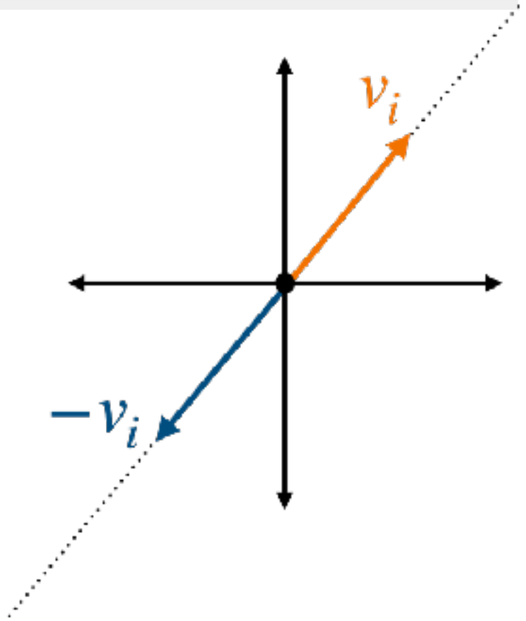
How parametrize architecture to approximate any such f?

Necessary invariances

- **Sign invariance:** with all distinct eigenvalues, $\lambda_i \neq \lambda_j, \forall i, j$

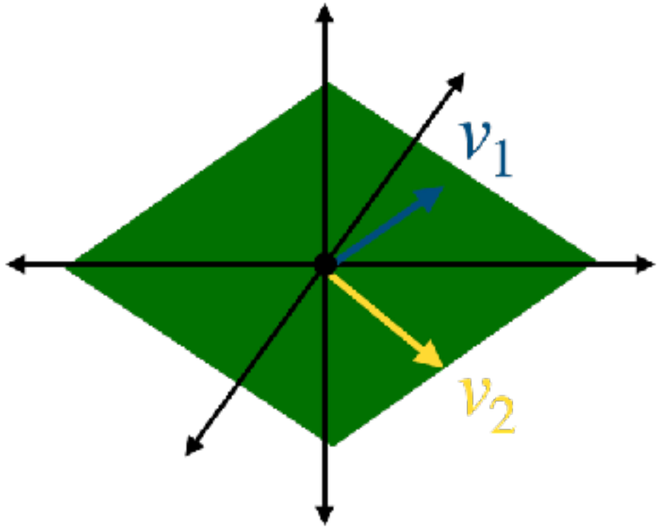
  Solver may return v_i or $-v_i$.

  $$f(v) = f(-v)$$



- **Eigenspaces:** eigenvalue multiplicities $\lambda_{i_1} = \ldots = \lambda_{i_d}$

  Any basis for *d*-dimensional eigenspace is valid.
  Multiplicities are frequent in real data!



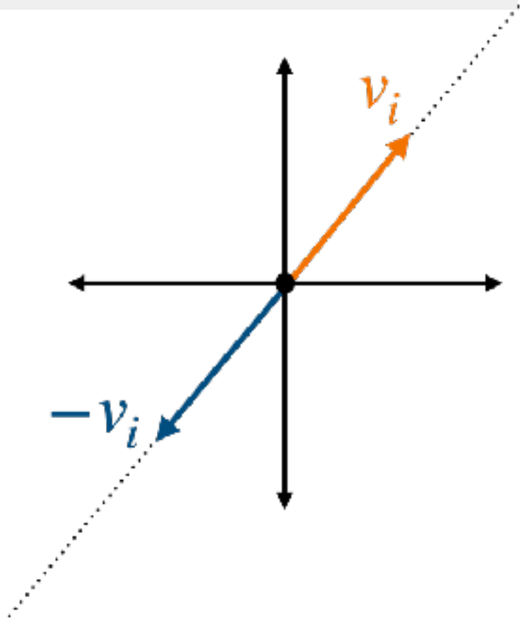
Invariances needed for generalization

Necessary invariances

- **Sign invariance:** with all distinct eigenvalues, $\lambda_i \neq \lambda_j, \forall i, j$

  Solver may return v_i or $-v_i$.

  $$f(v) = f(-v)$$

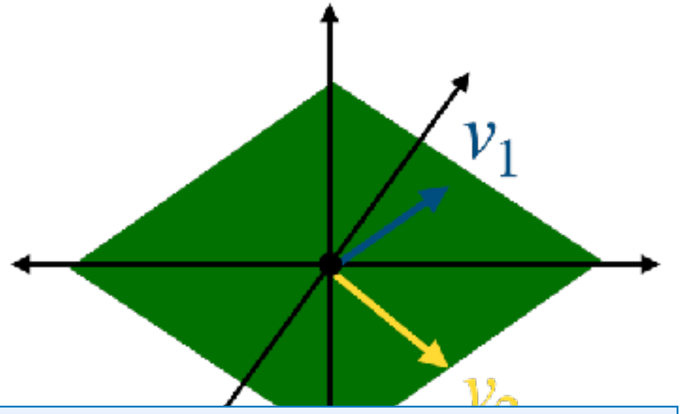


- **Eigenspaces:** eigenvalue multiplicities $\lambda_{i_1} = \ldots = \lambda_{i_d}$

  Solver may return any basis for *d*-dimensional eigenspace.

  $f(V) = f(VQ)$ for all Q in orthogonal group $O(d)$

  $V = [v_{i_1}, \ldots, v_{i_d}]$



+ multiple subspaces
+ permutation equivariance …

One subspace: sign invariance

Proposition

$f : \mathbb{R}^n \to \mathbb{R}$ is continuous and sign invariant if and only if $f(v) = \phi(v) + \phi(-v)$ for some continuous ϕ.

If f is also permutation equivariant, then so is ϕ .

Universal Architecture:

General f : $\phi = \mathrm{MLP}$

Permutation equivariant f: $\phi = \mathrm{DeepSets}$ *(Zaheer et al 2017, Lee et al 2019)*

One subspace: basis invariance

Proposition

If $f : \mathbb{R}^{n \times d} \to \mathbb{R}$ is continuous and $f(VQ) = f(V), \forall Q \in O(d)$, then $f(V) = \phi(VV^\top)$ for some continuous ϕ.

$(VQ)(VQ)^\top = V(QQ^\top)V^\top = VV^\top$

One subspace: basis invariance

Proposition

If $f : \mathbb{R}^{n \times d} \to \mathbb{R}$ is continuous and $f(VQ) = f(V), \forall Q \in O(d)$, then $f(V) = \phi(VV^\top)$ for some continuous ϕ.

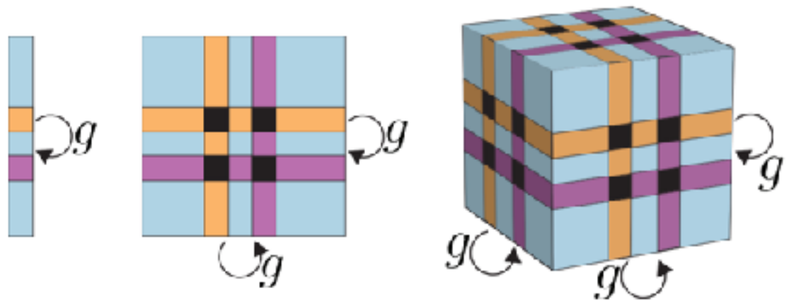
If f is also permutation equivariant, then $\phi : \mathbb{R}^{n \times n} \to \mathbb{R}^n$ is permutation equivariant from matrices to vectors.

Universal approximation of basis-invariant functions

General f: $\phi = \mathrm{MLP}$

Permutation equivariant f: $\phi = \mathrm{IGN}$

Invariant Graph Network (Maron et al 2018)



Multiple subspaces: group invariance

- $V_1, \ldots, V_\ell$ bases of eigenspaces, $\dim V_i = d_i$

- Invariance to change of basis in each eigenspace:

$$f(V_1 Q_1, \ldots V_\ell Q_\ell) = f(V_1, \ldots, V_\ell), \quad Q_i \in O(d_i)$$

invariant to $G = O(d_1) \times \ldots \times O(d_\ell)$

- Sign invariance: $f(\pm v_1, \ldots, \pm v_\ell) = f(v_1, \ldots, v_\ell)$

Multiple subspaces: representation

$$f(V_1 Q_1, \ldots V_\ell Q_\ell) = f(V_1, \ldots, V_\ell), \quad Q_i \in O(d_i)$$

Decomposition Theorem

Under mild assumptions, every continuous f that is invariant to $G_1 \times \ldots \times G_\ell$ can be written as:

$$f(x_1, \ldots, x_\ell) = \rho\left(\phi_1(x_1), \ldots, \phi_\ell(x_\ell)\right)$$

1. ϕ_i is G_i-invariant
2. If $d_i = d_j$ then can take $\phi_i = \phi_j$

- Only need to do G_i invariance for $G_1 \times \ldots \times G_l$ invariance!!
- => Universal Approximation of invariant continuous functions.

Practical instantiations

function on sequence / set / vector

function on set / graph nodes / vector / matrix

$$f(x_1, \ldots, x_\ell) = \rho\left(\phi_1(x_1), \ldots, \phi_\ell(x_\ell)\right)$$

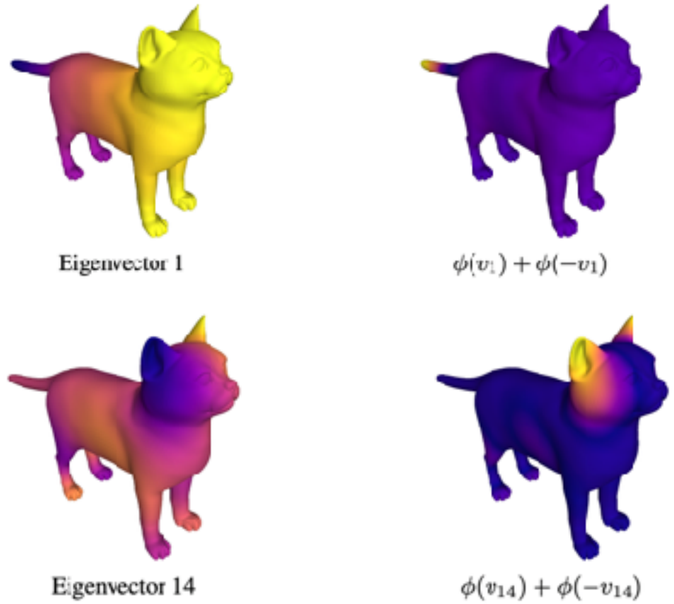
- **SignNet**: $f(v_1, \ldots, v_\ell) = \rho\big(\phi(v_1) + \phi(-v_1), \ldots, \phi(v_\ell) + \phi(-v_\ell)\big)$

  ϕ, ρ : DeepSets, Transformer, or GNN

- **BasisNet**: $f(V_1, \ldots, V_\ell) = \rho\Big([\phi_{d_i}(V_i V_i^\top)]_{i=1}^{\ell} \Big)$

  $\phi_d = \mathrm{IGN}_d$ order 2 (efficiency) or higher-order (universality)

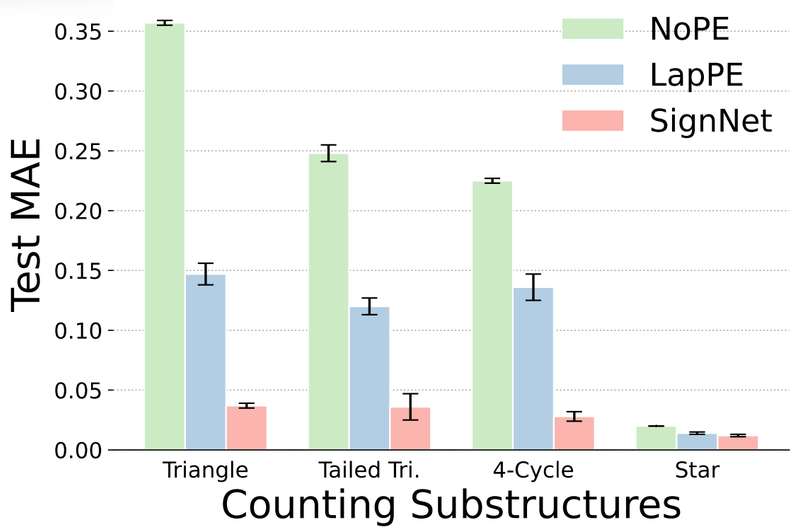
  $\rho =$ MLP, DeepSets, Transformer



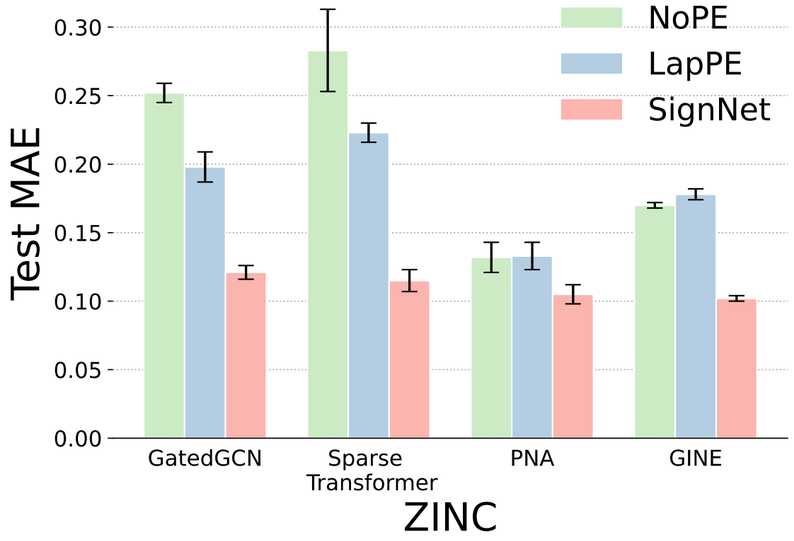
Theoretical and empirical benefits

Lemma: There exist infinitely many pairs of graphs that BasisNet can distinguish, but spectral GNNs cannot.

Lemma: can approximate spectral graph invariants, e.g. graph angles
➔ can express number of 3-, 4-, 5-cycles, connectivity, length-k closed walks.
Message passing GNNs cannot!



ϕ = GIN *(Xu et al 2019)* ρ = Transformer *(Vaswani et al)*



ϕ = GIN *(Xu et al 2019)*, ρ = MLP

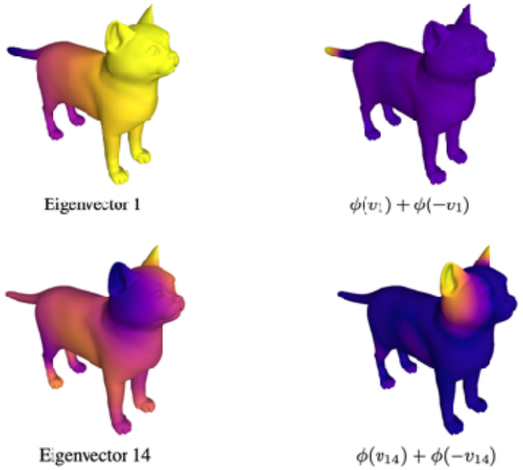
Texture reconstruction

- Neural fields on manifolds: eigenfunctions of Laplace-Beltrami operator as positional encodings

$$f(p) = \mathrm{NN}(v_1(p), \ldots, v_k(p))$$

Table 3: Test results for texture reconstruction experiment on cat and human models, following the experimental setting of (Koestler et al., 2022). We use 1023 eigenvectors of the cotangent Laplacian.

		Cat			Human		
Method	Params	PSNR ↑	DSSIM ↓	LPIPS ↓	PSNR ↑	DSSIM ↓	LPIPS ↓
Intrinsic NF	329k	34.25	.099	.189	32.29	**.119**	.330
Absolute value	329k	34.67	.106	.252	**32.42**	.132	.363
Sign flip	329k	23.15	1.28	2.35	21.52	1.05	2.71
SignNet	324k	**34.91**	**.090**	**.147**	**32.43**	.125	**.316**



Summary

- In many training settings, NN weights do not converge to a fixed point
- Statistical algorithmic stability implies generalization
  robustness of *statistics* on loss space
- Convergence of loss distribution predictive of generalization gap:
  dynamics information tells about generalization
  — in line with other works *(e.g., Loukas et al 2021,..)*, but more generally applicable

- Learning with symmetries: many applications
- SignNet/BasisNet: generic neural network models on eigenvectors
- Improve expressive power of Graph Representation Learning (theory & practice)

Relevant papers

- N. Chandramoorthy, K. Gatmiry, A. Loukas, S. Jegelka. On the generalization of learning algorithms that do not converge. NeurIPS 2022.
- D. Lim, J. Robinson, L. Zhao, H. Maron, T. Smidt, S. Sra, S. Jegelka. Sign and Basis Invariant Networks for Spectral Graph Representation Learning. ICLR 2023.