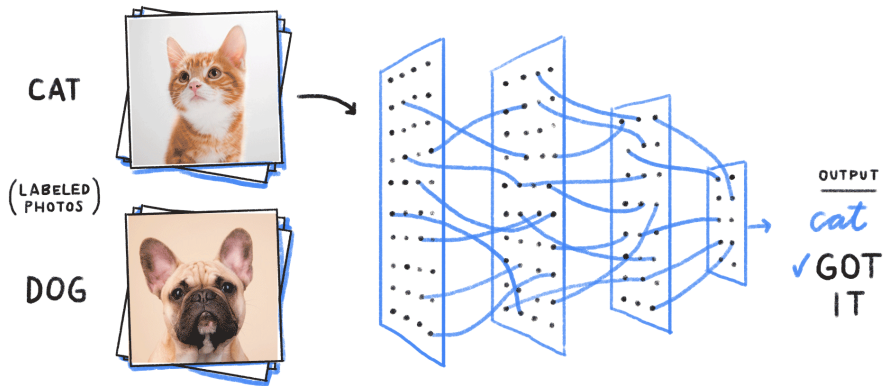


# Learning from ranks, learning to rank

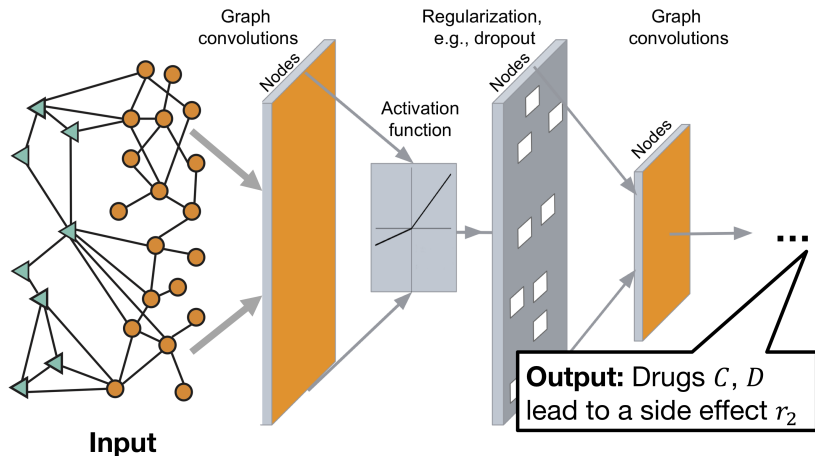
Jean-Philippe Vert



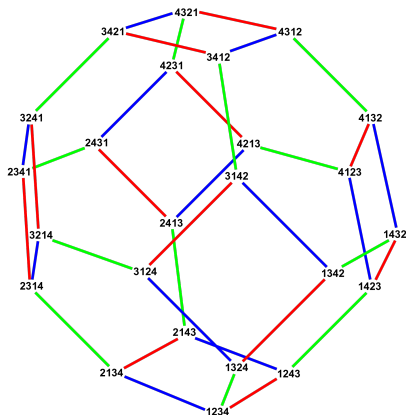
# Differentiable programming



# Beyond images and strings



# What if inputs or outputs are permutations?



- Permutation: a bijection

$$\sigma : [1, N] \rightarrow [1, N]$$

- $\sigma(i) = \text{rank of item } i$
- Composition

$$(\sigma_1 \sigma_2)(i) = \sigma_1(\sigma_2(i))$$

- $\mathbb{S}_N$  the symmetric group
- $|\mathbb{S}_N| = N!$

# Examples

- Rankings (as input or output)



- Discretization / normalization of continuous data



(histogram equalization, quantile normalization...)

## 1 Permutations as **input / intermediate**:

$$\sigma \in \mathbb{S}_N \mapsto f_\theta(\sigma) \in \mathbb{R}^p$$

How to **define / optimize**  $f_\theta : \mathbb{S}_N \rightarrow \mathbb{R}^p$ ?

- SUQUAN (Le Morvan and Vert, 2017), Kendall (Jiao and Vert, 2015, 2017, 2018)

## 2 Permutations as **intermediate / output**:

$$x \in \mathbb{R}^N \mapsto \sigma(x) \in \mathbb{S}_N \mapsto f_\theta(\sigma(x)) \in \mathbb{R}^p$$

How to **differentiate** the ranking operator  $\sigma : \mathbb{R}^N \rightarrow \mathbb{S}_N$ ?

- Sinkhorn ranking (Cuturi et al., 2019)

# Permutations as inputs

- Assume your data are permutations and you want to learn

$$f : \mathbb{S}_N \rightarrow \mathbb{R}$$

- A solutions: **embed**  $\mathbb{S}_N$  to a Euclidean (or Hilbert) space

$$\Phi : \mathbb{S}_N \rightarrow \mathbb{R}^p$$

and learn a linear function:

$$f_\beta(\sigma) = \beta^\top \Phi(\sigma)$$

- The corresponding **kernel** is

$$K(\sigma_1, \sigma_2) = \Phi(\sigma_1)^\top \Phi(\sigma_2)$$

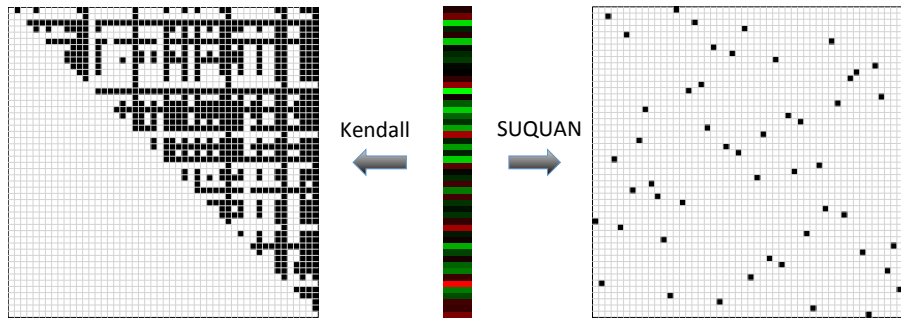
# How to define the embedding $\Phi : \mathbb{S}_N \rightarrow \mathbb{R}^p$ ?

- Should encode **interesting features**
- Should lead to **efficient algorithms**
  
- Should be invariant to renaming of the items, i.e., the kernel should be **right-invariant**

$$\forall \sigma_1, \sigma_2, \pi \in \mathbb{S}_N, \quad K(\sigma_1 \pi, \sigma_2 \pi) = K(\sigma_1, \sigma_2)$$

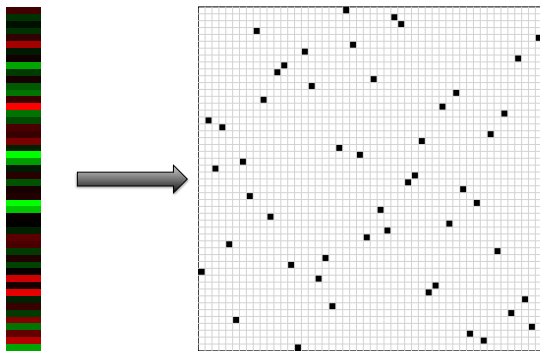


# Some attempts



(Jiao and Vert, 2015, 2017, 2018; Le Morvan and Vert, 2017)

# SUQUAN embedding (Le Morvan and Vert, 2017)



- Let  $\Phi(\sigma) = \Pi_\sigma$  the permutation representation (Serres, 1977):

$$[\Pi_\sigma]_{ij} = \begin{cases} 1 & \text{if } \sigma(j) = i, \\ 0 & \text{otherwise.} \end{cases}$$

- Right invariant:

$$\langle \Phi(\sigma), \Phi(\sigma') \rangle = \text{Tr}(\Pi_\sigma \Pi_{\sigma'}^\top) = \text{Tr}(\Pi_\sigma \Pi_{\sigma'}^{-1}) = \text{Tr}(\Pi_\sigma \Pi_{\sigma'^{-1}}) = \text{Tr}(\Pi_{\sigma\sigma'^{-1}})$$

# Link with quantile normalization (QN)

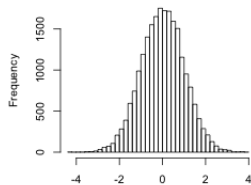


- Take  $\sigma(x) = \text{rank}(x)$  with  $x \in \mathbb{R}^N$
- Fix a **target quantile**  $f \in \mathbb{R}^n$
- "Keep the order of  $x$ , change the values to  $f$ "

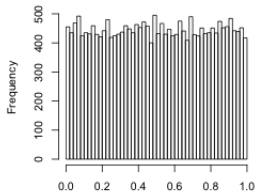
$$[\Psi_f(x)]_i = f_{\sigma(x)(i)} \quad \Leftrightarrow \quad \Psi_f(x) = \Pi_{\sigma(x)} f$$

# How to choose a "good" target distribution?

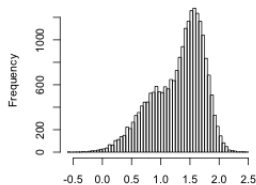
gaussian distribution (mean=0, sd=1)



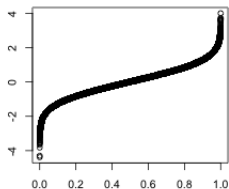
uniform distribution



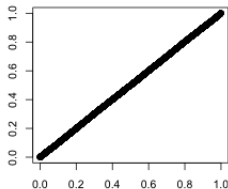
bigaussian distribution



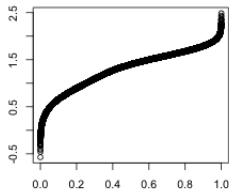
quantile function (->gaussian)



quantile function (-> uniform)



quantile function (->bigaussian)



# Supervised QN (SUQUAN)

Standard QN:

- 1 Fix  $f$  arbitrarily
- 2 QN all samples to get  $\Psi_f(x_1), \dots, \Psi_f(x_N)$
- 3 Learn a model on normalized data, e.g.:

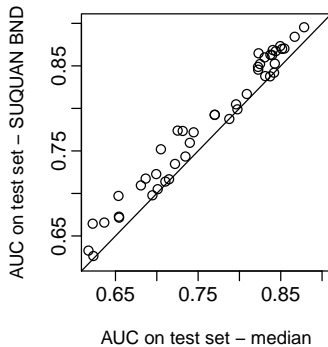
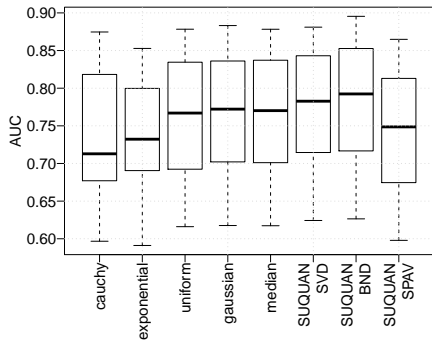
$$\min_{\theta} \left\{ \frac{1}{N} \sum_{i=1}^N \ell_i (f_{\theta}(\Psi_f(x_i))) \right\}$$

SUQUAN: **jointly** learn  $f$  and the model:

$$\min_{\theta, f} \left\{ \frac{1}{N} \sum_{i=1}^N \ell_i (f_{\theta}(\Psi_f(x_i))) \right\} = \min_{\theta, f} \left\{ \frac{1}{N} \sum_{i=1}^N \ell_i (f_{\theta}(\Pi_{\sigma(x_i)} f)) \right\}$$

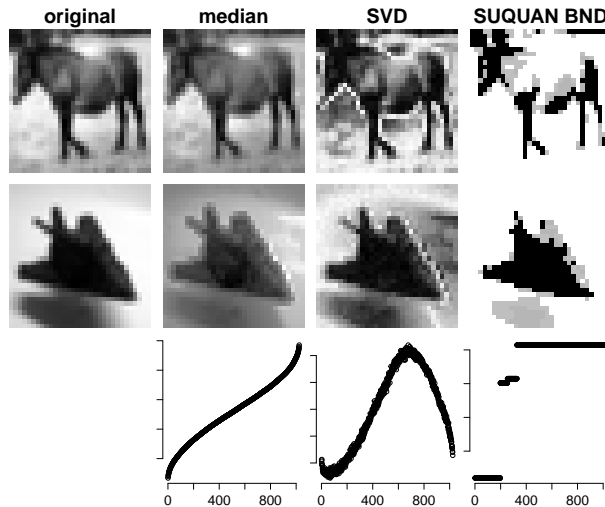
# Experiments: CIFAR-10

- Image classification into 10 classes (45 binary problems)
- $N = 5,000$  per class,  $p = 1,024$  pixels
- Linear logistic regression on raw pixels

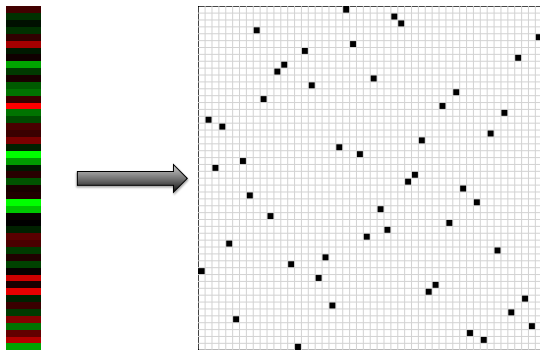


# Experiments: CIFAR-10

- Example: horse vs. plane
- Different methods learn different quantile functions



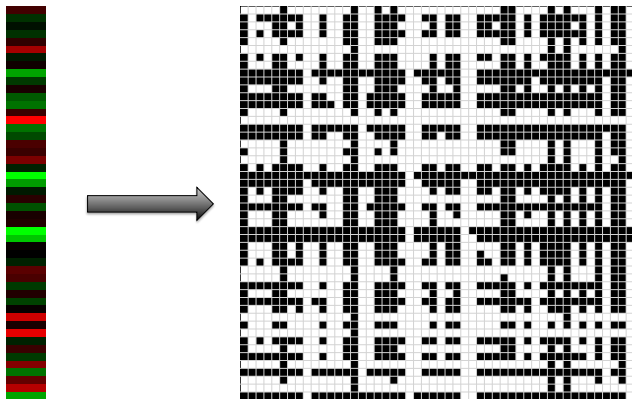
# Limits of the SUQUAN embedding



- Linear model on  $\Phi(\sigma) = \Pi_\sigma \in \mathbb{R}^{N \times N}$
- Captures **first-order** information of the form "*i-th feature ranked at the j-th position*"
- What about **higher-order** information such as "*feature i larger than feature j*"?

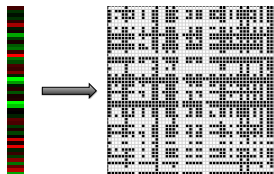


# The Kendall embedding (Jiao and Vert, 2015, 2017)



$$\Phi_{i,j}(\sigma) = \begin{cases} 1 & \text{if } \sigma(i) < \sigma(j), \\ 0 & \text{otherwise.} \end{cases}$$

# Geometry of the embedding



For any two permutations  $\sigma, \sigma' \in \mathbb{S}_N$ :

- Inner product

$$\Phi(\sigma)^\top \Phi(\sigma') = \sum_{1 \leq i \neq j \leq n} \mathbb{1}_{\sigma(i) < \sigma(j)} \mathbb{1}_{\sigma'(i) < \sigma'(j)} = n_c(\sigma, \sigma')$$

$n_c$  = number of concordant pairs

- Distance

$$\|\Phi(\sigma) - \Phi(\sigma')\|^2 = \sum_{1 \leq i, j \leq n} (\mathbb{1}_{\sigma(i) < \sigma(j)} - \mathbb{1}_{\sigma'(i) < \sigma'(j)})^2 = 2n_d(\sigma, \sigma')$$

$n_d$  = number of discordant pairs

# Kendall and Mallows kernels

- The **Kendall kernel** is

$$K_{\tau}(\sigma, \sigma') = n_c(\sigma, \sigma')$$

- The **Mallows kernel** is

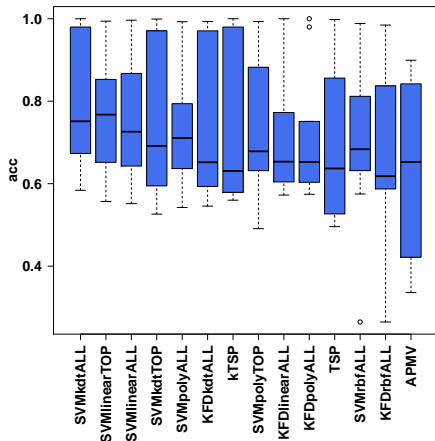
$$\forall \lambda \geq 0 \quad K_M^{\lambda}(\sigma, \sigma') = e^{-\lambda n_d(\sigma, \sigma')}$$

## Theorem (Jiao and Vert, 2015, 2017)

The Kendall and Mallows kernels are **positive definite right-invariant** kernels and can be evaluated in  $O(N \log N)$  time

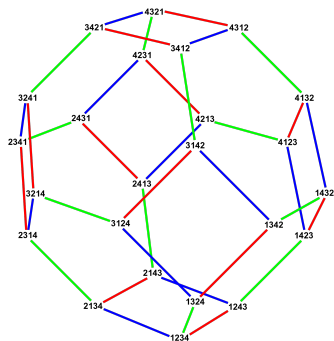
*Kernel trick useful with few samples in large dimensions*

# Applications



Average performance on 10 microarray classification problems (Jiao and Vert, 2017).

# Remark



Cayley graph of  $S_4$

- Kondor and Barbarosa (2010) proposed the **diffusion kernel** on the Cayley graph of the symmetric group generated by adjacent transpositions.
- Computationally intensive ( $O(N^{2N})$ )
- Mallows kernel is written as

$$K_M^\lambda(\sigma, \sigma') = e^{-\lambda n_d(\sigma, \sigma')},$$

where  $n_d(\sigma, \sigma')$  is the **shortest path distance** on the Cayley graph.

- It can be computed in  $O(N \log N)$
- Extension to **weighted** Kendall kernel (Jiao and Vert, 2018)

## Remark

The SUQUAN and Kendall representations are two particular cases of the more general

### Bochner's theorem

An embedding  $\Phi : \mathbb{S}_N \rightarrow \mathbb{R}^p$  defines a right-invariant kernel  $K(\sigma_1, \sigma_2) = \Phi(\sigma_1)^\top \Phi(\sigma_2)$  if and only there exists  $\phi : \mathbb{S}_N \rightarrow \mathbb{R}$  such that

$$\forall \sigma_1, \sigma_2 \in \mathbb{S}_N, \quad K(\sigma_1, \sigma_2) = \phi(\sigma_2^{-1} \sigma_1)$$

and

$$\forall \lambda \in \Lambda, \quad \hat{\phi}(\rho_\lambda) \succeq 0$$

where for any  $f : \mathbb{S}_N \rightarrow \mathbb{R}$ , the **Fourier transform** of  $f$  is

$$\forall \lambda \in \Lambda, \quad \hat{f}(\rho_\lambda) = \sum_{\sigma \in \mathbb{S}_N} f(\sigma) \rho_\lambda(\sigma)$$

with  $\{\rho_\lambda : \lambda \in \Lambda\}$  the irreducible representations of the symmetric group.

# Permutations as intermediate / output?

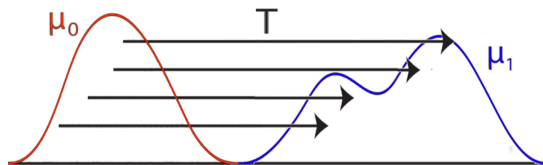
- Ranking operator:

$$\text{rank}(-15, 2.3, 20, -2) = (4, 2, 1, 3)$$

- Main problem:

$$x \in \mathbb{R}^N \mapsto \text{rank}(x) \in \mathbb{S}_N \quad \text{is not differentiable}$$

# Optimal transport (OT)



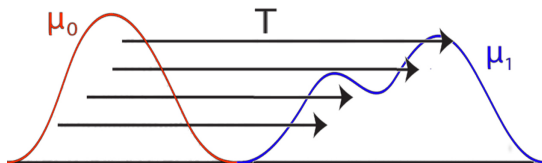
Given a cost matrix  $C \in \mathbb{R}^{n \times n}$  (where  $C_{ij}$  is the cost of moving the  $i$ -th point to the  $j$ -th location), OT solves

$$\min_{P \in B_n} \langle P, C \rangle$$

where  $B_n = \{P \in \mathbb{R}_+^{n \times n} \mid P\mathbf{1}_n = P^\top \mathbf{1}_n = \mathbf{1}_n\}$  is the Birkhoff polytope.



# From OT to rank



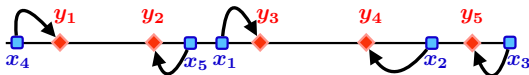
## Lemma

Take

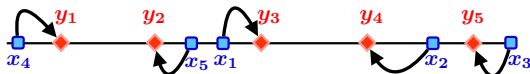
- $y \in \mathbb{R}^n$  with  $y_1 < \dots < y_n$ ,
- $h \in C^2(\mathbb{R}^2)$  and  $\partial^2 h / \partial x \partial y > 0$  (eg,  $h(a, b) = (b - a)^2$ ).

For any  $x \in \mathbb{R}^n$ , let  $C(x) \in \mathbb{R}^{n \times n}$  given by  $C(x)_{ij} = h(y_i, x_j)$ . Then

$$\operatorname{argmin}_{P \in B_n} \langle P, C(x) \rangle = \Pi_{\sigma(x)}.$$



# Differentiable permutation matrix



$$P_\epsilon(x) = \operatorname{argmin}_{P \in B_n} \langle P, C(x) \rangle - \epsilon H(P)$$

---

## Algorithm 1: Sinkhorn

---

**Inputs:**  $\mathbf{a}, \mathbf{b}, \mathbf{x}, \mathbf{y}, \epsilon, \ell$

$K \leftarrow e^{-C_{\mathbf{x}\mathbf{y}}/\epsilon}, \mathbf{u}_0 = \mathbf{1}_n;$

**for**  $t \leftarrow 0$  **to**  $\ell - 1$  **do**

$\mathbf{v}_{t+1} \leftarrow \mathbf{b}/K^T \mathbf{u}_t$   
     $\mathbf{u}_{t+1} \leftarrow \mathbf{a}/K \mathbf{v}_{t+1}$

**end**

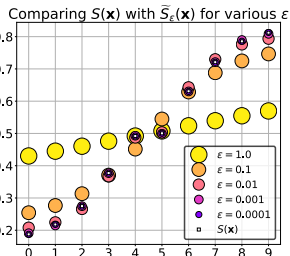
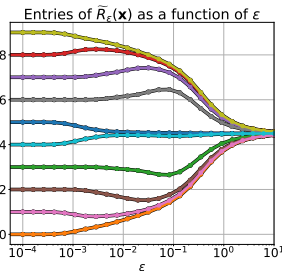
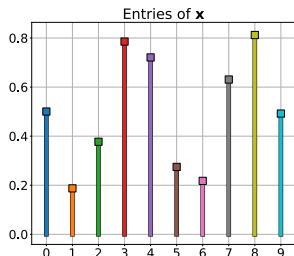
**Result:**  $\mathbf{u}_\ell, K, \mathbf{v}_\ell$

---

- $P = \operatorname{diag}(\mathbf{u}_\ell)K\operatorname{diag}(\mathbf{v}_\ell)$  is the **differentiable** approximate permutation matrix of the input vector  $\mathbf{x}$
- Complexity  $O(nm\ell)$ , GPU-friendly

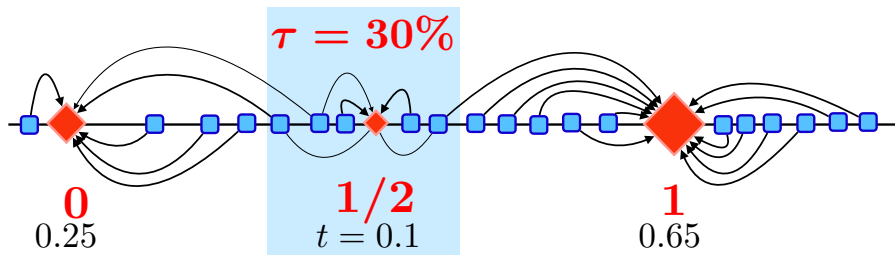
# Differentiable sort and rank

$$S_\epsilon(x) = P_\epsilon(x)x \quad R_\epsilon(x) = P_\epsilon(x)^\top(1, 2, \dots, n)^\top$$



# Soft quantization and soft quantiles

- Take  $m = 3$ ,  $\mathbf{y} = (0, 0.5, 1)$ ,  $\mathbf{b} = (\tau - t/2, t, \tau + t/2)$
- Overall complexity  $O(n\ell)$



[https://github.com/google-research/google-research/tree/master/soft\\_sort](https://github.com/google-research/google-research/tree/master/soft_sort)

# Application: soft top- $k$ loss

$$\text{S-top-}k\text{-loss}(f_{\theta}(\omega_0), l_0) = J_k \left( 1 - \left( \tilde{F}^{\ell} \left( \frac{\mathbf{1}_L}{L}, f_{\theta}(\omega); \frac{\mathbf{1}_m}{m}, \mathbf{y} \right) \right)_{l_0} \right)$$

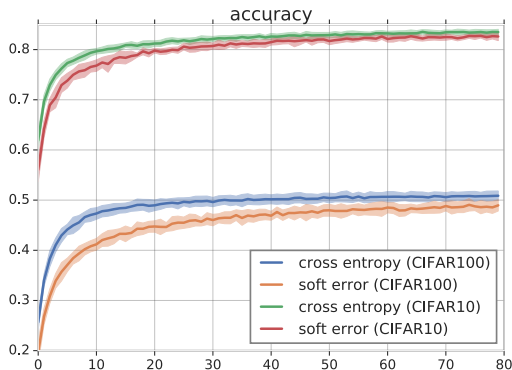
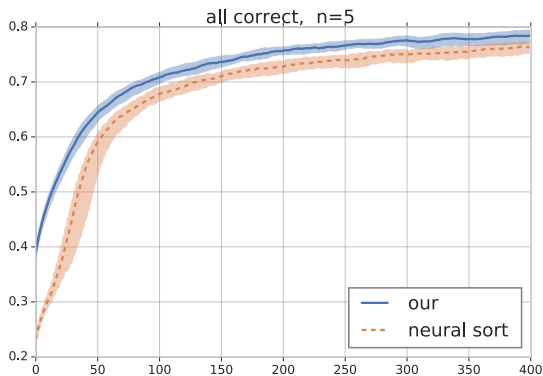


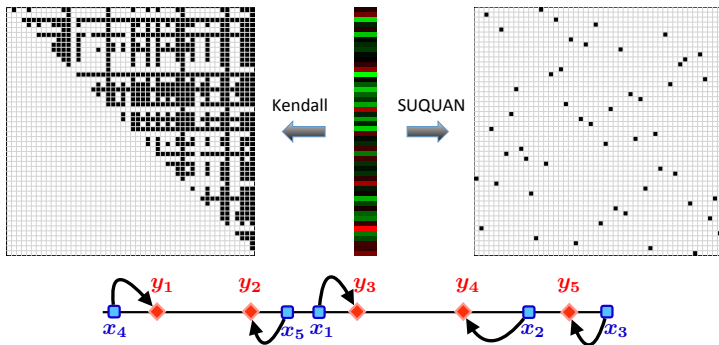
Figure 4: Error bars for test accuracy curves on CIFAR-100 and CIFAR-10 using the same network (averages over 12 runs).

# Application: learning to sort

- Task: Sort 5 numbers between 0000 and 9999 (concatenation of MNIST digits) (Grover et al, 2019)



# Conclusion



- Machine learning beyond vectors, strings and graphs
- Different embeddings of the symmetric group
- Differentiable sorting and ranking
- Can be generalized to other discrete operations

THANK YOU!

# References

- M. Cuturi, O. Teboul, and J.-P. Vert. Differentiable sorting using optimal transport: the Sinkhorn CDF and quantile operator. In *Adv. Neural. Inform. Process Syst.* 31, 2019.
- Y. Jiao and J.-P. Vert. The Kendall and Mallows kernels for permutations. In *Proceedings of The 32nd International Conference on Machine Learning*, volume 37 of *JMLR:W&CP*, pages 1935–1944, 2015. URL <http://jmlr.org/proceedings/papers/v37/jiao15.html>.
- Y. Jiao and J.-P. Vert. The Kendall and Mallows kernels for permutations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017. doi: 10.1109/TPAMI.2017.2719680. URL <http://dx.doi.org/10.1109/TPAMI.2017.2719680>.
- Y. Jiao and J.-P. Vert. The weighted kendall and high-order kernels for permutations. Technical Report 1802.08526, arXiv, 2018.
- M. Le Morvan and J.-P. Vert. Supervised quantile normalisation. Technical Report 1706.00244, arXiv, 2017.
- J.-P. Serres. *Linear Representations of Finite Groups*. Graduate Texts in Mathematics. Springer-Verlag New York, 1977. doi: 10.1007/978-1-4684-9458-7. URL <http://dx.doi.org/10.1007/978-1-4684-9458-7>.



# Harmonic analysis on $\mathbb{S}_N$

- A **representation** of  $\mathbb{S}_N$  is a matrix-valued function  $\rho : \mathbb{S}_N \rightarrow \mathbb{C}^{d_\rho \times d_\rho}$  such that

$$\forall \sigma_1, \sigma_2 \in \mathbb{S}_N, \quad \rho(\sigma_1 \sigma_2) = \rho(\sigma_1) \rho(\sigma_2)$$

- A representation is irreducible (**irrep**) if it is not equivalent to the direct sum of two other representations
- $\mathbb{S}_N$  has a finite number of irreps  $\{\rho_\lambda : \lambda \in \Lambda\}$  where  $\Lambda = \{\lambda \vdash N\}$ <sup>1</sup> is the set of partitions of  $N$
- For any  $f : \mathbb{S}_N \rightarrow \mathbb{R}$ , the **Fourier transform** of  $f$  is

$$\forall \lambda \in \Lambda, \quad \hat{f}(\rho_\lambda) = \sum_{\sigma \in \mathbb{S}_N} f(\sigma) \rho_\lambda(\sigma)$$

---

<sup>1</sup> $\lambda \vdash N$  iff  $\lambda = (\lambda_1, \dots, \lambda_r)$  with  $\lambda_1 \geq \dots \geq \lambda_r$  and  $\sum_{i=1}^r \lambda_i = N$

## Bochner's theorem

An embedding  $\Phi : \mathbb{S}_N \rightarrow \mathbb{R}^p$  defines a right-invariant kernel  $K(\sigma_1, \sigma_2) = \Phi(\sigma_1)^\top \Phi(\sigma_2)$  if and only there exists  $\phi : \mathbb{S}_N \rightarrow \mathbb{R}$  such that

$$\forall \sigma_1, \sigma_2 \in \mathbb{S}_N, \quad K(\sigma_1, \sigma_2) = \phi(\sigma_2^{-1} \sigma_1)$$

and

$$\forall \lambda \in \Lambda, \quad \hat{\phi}(\rho_\lambda) \succeq 0$$