Invariance and Stability to Deformations of Deep Convolutional Representations

Alberto Bietti

Inria Grenoble

Success of deep convolutional networks

Convolutional Neural Networks (CNNs):
- Capture **multi-scale** and **compositional** structure in natural signals
- Provide some **invariance**
- Model **local stationarity**
- **State-of-the-art** in many applications
Understanding deep convolutional representations

- Are they stable to deformations?
- How can we achieve invariance to transformation groups?
- Do they preserve signal information?
- What are good measures of model complexity?
A kernel perspective

Kernels?
- Map data $x$ to high-dimensional space, $\Phi(x) \in \mathcal{H}$ (\mathcal{H}: “RKHS”)
- Non-linear $f \in \mathcal{H}$ takes linear form: $f(x) = \langle f, \Phi(x) \rangle$
- Learning with a positive definite kernel $K(x, x') = \langle \Phi(x), \Phi(x') \rangle$
A kernel perspective

Kernels?

- Map data $x$ to high-dimensional space, $\Phi(x) \in \mathcal{H}$ ($\mathcal{H}$: "RKHS")
- Non-linear $f \in \mathcal{H}$ takes linear form: $f(x) = \langle f, \Phi(x) \rangle$
- Learning with a positive definite kernel $K(x, x') = \langle \Phi(x), \Phi(x') \rangle$
- Here, we construct specific kernels based on convolutional architectures, following Mairal (2016)
A kernel perspective

Kernels?
- Map data $x$ to high-dimensional space, $\Phi(x) \in \mathcal{H}$ ($\mathcal{H}$: “RKHS”)
- Non-linear $f \in \mathcal{H}$ takes linear form: $f(x) = \langle f, \Phi(x) \rangle$
- Learning with a positive definite kernel $K(x, x') = \langle \Phi(x), \Phi(x') \rangle$
- Here, we construct specific kernels based on convolutional architectures, following Mairal (2016)
  - Good empirical performance on image tasks (Mairal et al., 2014; Mairal, 2016)
  - RKHS contains CNNs, leads to good regularizers (Bietti et al., 2019)
  - Also related to neural tangent kernels for CNNs (Bietti and Mairal, 2019b)
A kernel perspective

Why? Separate learning from representation: $f(x) = \langle f, \Phi(x) \rangle$

- $\Phi(x)$: CNN architecture (stability, invariance, signal preservation)
- $f$: CNN model, learning, generalization through RKHS norm $\|f\|$  

$$|f(x) - f(x')| \leq \|f\| \cdot \|\Phi(x) - \Phi(x')\|$$

- $\|f\|$ controls both stability and model complexity!
  - discriminating small perturbations requires large $\|f\|$
  - learning stable functions may be “easier”
A signal processing perspective

- Consider images defined on a **continuous** domain $\Omega = \mathbb{R}^2$.
- $\tau : \Omega \rightarrow \Omega$: $C^1$-diffeomorphism.
- $L_\tau x(u) = x(u - \tau(u))$: action operator.
- Much richer group of transformations than translations.
A signal processing perspective

- Consider images defined on a **continuous** domain $\Omega = \mathbb{R}^2$.
- $\tau : \Omega \to \Omega$: $C^1$-diffeomorphism.
- $L_\tau x(u) = x(u - \tau(u))$: action operator.
- Much richer group of transformations than translations.

**Definition of stability**

- Representation $\Phi(\cdot)$ is **stable** (Mallat, 2012) if:
  \[
  \|\Phi(L_\tau x) - \Phi(x)\| \leq (C_1 \|\nabla \tau\|_\infty + C_2 \|\tau\|_\infty)\|x\|.
  \]

- $\|\nabla \tau\|_\infty = \sup_u \|\nabla \tau(u)\|$ controls deformation.
- $\|\tau\|_\infty = \sup_u |\tau(u)|$ controls translation.
- $C_2 \to 0$: translation invariance.
Outline

1. Construction of the Convolutional Representation
2. Invariance and Stability
3. Learning Aspects: Model Complexity of CNNs
4. Regularizing with the RKHS norm
A generic deep convolutional representation

- $x_0 : \Omega \to \mathcal{H}_0$: initial (continuous) signal
  - $u \in \Omega = \mathbb{R}^d$: location ($d = 2$ for images)
  - $x_0(u) \in \mathcal{H}_0$: value ($\mathcal{H}_0 = \mathbb{R}^3$ for RGB images)
- $x_k : \Omega \to \mathcal{H}_k$: feature map at layer $k$

$$P_k x_{k-1}$$

- $P_k$: patch extraction operator, extract small patch of feature map $x_{k-1}$ around each point $u$
A generic deep convolutional representation

- $x_0 : \Omega \rightarrow \mathcal{H}_0$: initial (continuous) signal
  - $u \in \Omega = \mathbb{R}^d$: location ($d = 2$ for images)
  - $x_0(u) \in \mathcal{H}_0$: value ($\mathcal{H}_0 = \mathbb{R}^3$ for RGB images)
- $x_k : \Omega \rightarrow \mathcal{H}_k$: feature map at layer $k$

\[ M_k P_k x_{k-1} \]

- $P_k$: patch extraction operator, extract small patch of feature map $x_{k-1}$ around each point $u$
- $M_k$: non-linear mapping operator, maps each patch to a new point with a pointwise non-linear function $\varphi_k(\cdot)$
A generic deep convolutional representation

- $x_0 : \Omega \rightarrow \mathcal{H}_0$: initial (continuous) signal
  - $u \in \Omega = \mathbb{R}^d$: location ($d = 2$ for images)
  - $x_0(u) \in \mathcal{H}_0$: value ($\mathcal{H}_0 = \mathbb{R}^3$ for RGB images)
- $x_k : \Omega \rightarrow \mathcal{H}_k$: feature map at layer $k$

$$x_k = A_k M_k P_k x_{k-1}$$

- $P_k$: patch extraction operator, extract small patch of feature map $x_{k-1}$ around each point $u$
- $M_k$: non-linear mapping operator, maps each patch to a new point with a pointwise non-linear function $\varphi_k(\cdot)$
- $A_k$: (linear, Gaussian) pooling operator at scale $\sigma_k$
A generic deep convolutional representation

\[ x_k := A_k M_k P_k x_{k-1} : \Omega \to \mathcal{H}_k \]

\[ x_k(w) = A_k M_k P_k x_{k-1}(w) \in \mathcal{H}_k \]
linear pooling

\[ M_k P_k x_{k-1} : \Omega \to \mathcal{H}_k \]

\[ M_k P_k x_{k-1}(v) = \varphi_k(P_k x_{k-1}(v)) \in \mathcal{H}_k \]
non-linear mapping

\[ P_k x_{k-1}(v) \in \mathcal{P}_k \]
(patch extraction)

\[ x_{k-1}(w) \in \mathcal{H}_{k-1} \]

\[ x_{k-1} : \Omega \to \mathcal{H}_{k-1} \]
Patch extraction operator $P_k$

$$P_kx_{k-1}(u) := (v \in S_k \mapsto x_{k-1}(u + v)) \in \mathcal{P}_k = \mathcal{H}_{k-1}^{S_k}$$

$P_kx_{k-1}(v) \in \mathcal{P}_k$ (patch extraction)

$x_{k-1}(u) \in \mathcal{H}_{k-1}$

$x_{k-1} : \Omega \to \mathcal{H}_{k-1}$
Patch extraction operator $P_k$

$$P_k x_{k-1}(u) := (v \in S_k \mapsto x_{k-1}(u + v)) \in P_k = \mathcal{H}_{k-1}^{S_k}$$

- $S_k$: patch shape, e.g. box
- $P_k$ is linear, and **preserves the $L^2$ norm**: $\|P_k x_{k-1}\| = \|x_{k-1}\|$
Non-linear mapping operator $M_k$

\[ M_k P_k x_{k-1}(u) := \varphi_k(P_k x_{k-1}(u)) \in \mathcal{H}_k \]

$M_k P_k x_{k-1} : \Omega \rightarrow \mathcal{H}_k$

non-linear mapping

$P_k x_{k-1}(v) \in \mathcal{P}_k$

$x_{k-1} : \Omega \rightarrow \mathcal{H}_{k-1}$
Non-linear mapping operator $M_k$

\[ M_k P_k x_{k-1}(u) := \varphi_k(P_k x_{k-1}(u)) \in \mathcal{H}_k \]

- $\varphi_k : \mathcal{P}_k \rightarrow \mathcal{H}_k$ pointwise non-linearity on patches (kernel map)
- We assume **non-expansivity**: for $z, z' \in \mathcal{P}_k$

  \[ \|\varphi_k(z)\| \leq \|z\| \quad \text{and} \quad \|\varphi_k(z) - \varphi_k(z')\| \leq \|z - z'\| \]

- $M_k$ then satisfies, for $x, x' \in L^2(\Omega, \mathcal{P}_k)$

  \[ \|M_k x\| \leq \|x\| \quad \text{and} \quad \|M_k x - M_k x'\| \leq \|x - x'\| \]
Non-linear mapping operator $M_k$

\[ M_k P_k x_{k-1}(u) := \varphi_k(P_k x_{k-1}(u)) \in \mathcal{H}_k \]

- $\varphi_k : \mathcal{P}_k \to \mathcal{H}_k$ pointwise non-linearity on patches
- We assume: for $z, z' \in \mathcal{P}_k$
  \[ \| \varphi_k(z) \| \leq \rho_k \| z \| \quad \text{and} \quad \| \varphi_k(z) - \varphi_k(z') \| \leq \rho_k \| z - z' \| \]
- $M_k$ then satisfies, for $x, x' \in L^2(\Omega, \mathcal{P}_k)$
  \[ \| M_k x \| \leq \rho_k \| x \| \quad \text{and} \quad \| M_k x - M_k x' \| \leq \rho_k \| x - x' \| \]
- (at the cost of paying $\Pi_k \rho_k$ later)
\( \varphi_k \) from kernels

- Kernel mapping of **homogeneous dot-product kernels**:
  \[
  K_k(z', z) = \|z\| \|z'\| \kappa_k \left( \frac{\langle z, z' \rangle}{\|z\| \|z'\|} \right) = \langle \varphi_k(z), \varphi_k(z') \rangle.
  \]

- \( \kappa_k(u) = \sum_{j=0}^{\infty} b_j u^j \) with \( b_j \geq 0 \), \( \kappa_k(1) = 1 \)
- Commonly used for hierarchical kernels
- \( \|\varphi_k(z)\| = K_k(z, z)^{1/2} = \|z\| \)
- \( \|\varphi_k(z) - \varphi_k(z')\| \leq \|z - z'\| \) if \( \kappa'_k(1) \leq 1 \)
- \( \Longrightarrow \) **non-expansive**
$\varphi_k$ from kernels

- Kernel mapping of **homogeneous dot-product kernels**:

$$K_k(z, z') = \|z\|\|z'\| \kappa_k \left( \frac{\langle z, z' \rangle}{\|z\|\|z'\|} \right) = \langle \varphi_k(z), \varphi_k(z') \rangle.$$ 

- $\kappa_k(u) = \sum_{j=0}^{\infty} b_j u^j$ with $b_j \geq 0$, $\kappa_k(1) = 1$

- Commonly used for hierarchical kernels

  $$\|\varphi_k(z)\| = K_k(z, z)^{1/2} = \|z\|$$

  $$\|\varphi_k(z) - \varphi_k(z')\| \leq \|z - z'\| \text{ if } \kappa'_k(1) \leq 1$$

  $$\implies \text{ non-expansive}$$

- Examples:
  
  - $\kappa_{\text{exp}}(\langle z, z' \rangle) = e^{\langle z, z' \rangle} - 1$ (Gaussian kernel on the sphere)
  
  - $\kappa_{\text{inv-poly}}(\langle z, z' \rangle) = \frac{1}{2 - \langle z, z' \rangle}$
  
  - arc-cosine kernel of degree 1 (random features with ReLU activation)
$\varphi_k$ from kernels: CKNs approximation

Convolutional Kernel Networks approximation (Mairal, 2016):
Convolutional Kernel Networks approximation (Mairal, 2016):

- Approximate $\varphi_k(z)$ by projection on $\text{span}(\varphi_k(z_1), \ldots, \varphi_k(z_p))$ (Nystrom)
- Leads to tractable, $p$-dimensional representation $\psi_k(z)$
- Norm is preserved, and projection is non-expansive:

\[
\|\psi_k(z) - \psi_k(z')\| = \|\Pi_k \varphi_k(z) - \Pi_k \varphi_k(z')\| \\
\leq \|\varphi_k(z) - \varphi_k(z')\| \leq \|z - z'\|
\]

- Anchor points $z_1, \ldots, z_p$ (≈ filters) can be learned from data (K-means or backprop)
$\varphi_k$ from kernels: CKNs approximation

$M_1$

$\psi_1(x')$

linear pooling

$M_{0.5}$

$\psi_1(x)$

projection on $\mathcal{F}_1$

$M_0$

$\varphi_1(x)$

$\varphi_1(x')$

Hilbert space $\mathcal{H}_1$

kernel trick
Pool operator $A_k$

\[ x_k(u) = A_k M_k P_k x_{k-1}(u) = \int_{\mathbb{R}^d} h_{\sigma_k}(u - v) M_k P_k x_{k-1}(v) dv \in \mathcal{H}_k \]
Pooling operator $A_k$

$$x_k(u) = A_k M_k P_k x_{k-1}(u) = \int_{\mathbb{R}^d} h_{\sigma_k}(u - v) M_k P_k x_{k-1}(v) dv \in \mathcal{H}_k$$

- $h_{\sigma_k}$: pooling filter at scale $\sigma_k$
- $h_{\sigma_k}(u) := \sigma_k^{-d} h(u/\sigma_k)$ with $h(u)$ Gaussian
- linear, non-expansive operator: $\|A_k\| \leq 1$
Recap: $P_k$, $M_k$, $A_k$

$x_k := A_k M_k P_k x_{k-1} : \Omega \rightarrow \mathcal{H}_k$

$x_k(w) = A_k M_k P_k x_{k-1}(w) \in \mathcal{H}_k$
linear pooling

$M_k P_k x_{k-1} : \Omega \rightarrow \mathcal{H}_k$

$M_k P_k x_{k-1}(v) = \varphi_k(P_k x_{k-1}(v)) \in \mathcal{H}_k$
kernel mapping

$P_k x_{k-1}(v) \in \mathcal{P}_k$ (patch extraction)

$x_{k-1}(u) \in \mathcal{H}_{k-1}$

$x_{k-1} : \Omega \rightarrow \mathcal{H}_{k-1}$
Multilayer construction

Assumption on $x_0$

- $x_0$ is typically a **discrete** signal acquired with physical device.
- Natural assumption: $x_0 = A_0 x$, with $x$ the original continuous signal, $A_0$ local integrator with scale $\sigma_0$ (**anti-aliasing**).
Recap: $P_k, M_k, A_k$

$$x_k := A_k M_k P_k x_{k-1} : \Omega \rightarrow \mathcal{H}_k$$

$$x_k(w) = A_k M_k P_k x_{k-1}(w) \in \mathcal{H}_k$$

linear pooling

$$M_k P_k x_{k-1} : \Omega \rightarrow \mathcal{H}_k$$

$$M_k P_k x_{k-1}(v) = \varphi_k(P_k x_{k-1}(v)) \in \mathcal{H}_k$$

kernel mapping

$$P_k x_{k-1}(v) \in \mathcal{P}_k$$ (patch extraction)

$$x_{k-1}(u) \in \mathcal{H}_{k-1}$$

$$x_{k-1} : \Omega \rightarrow \mathcal{H}_{k-1}$$
Multilayer construction

Assumption on $x_0$

- $x_0$ is typically a **discrete** signal acquired with physical device.
- Natural assumption: $x_0 = A_0 x$, with $x$ the original continuous signal, $A_0$ local integrator with scale $\sigma_0$ (**anti-aliasing**).
Multilayer construction

Assumption on $x_0$

- $x_0$ is typically a **discrete** signal acquired with physical device.
- Natural assumption: $x_0 = A_0 x$, with $x$ the original continuous signal, $A_0$ local integrator with scale $\sigma_0$ (**anti-aliasing**).

Multilayer representation

$$\Phi_n(x) = A_n M_n P_n A_{n-1} M_{n-1} P_{n-1} \cdots A_1 M_1 P_1 x_0 \in L^2(\Omega, \mathcal{H}_n).$$

- $S_k, \sigma_k$ grow exponentially in practice (i.e., fixed with subsampling).
Multilayer construction

**Assumption on** $x_0$

- $x_0$ is typically a **discrete** signal acquired with physical device.
- Natural assumption: $x_0 = A_0 x$, with $x$ the original continuous signal, $A_0$ local integrator with scale $\sigma_0$ (**anti-aliasing**).

**Multilayer representation**

$$\Phi_n(x) = A_n M_n P_n A_{n-1} M_{n-1} P_{n-1} \cdots A_1 M_1 P_1 x_0 \in L^2(\Omega, \mathcal{H}_n).$$

- $S_k, \sigma_k$ grow exponentially in practice (i.e., fixed with subsampling).

**Prediction layer**

- e.g., linear $f(x) = \langle w, \Phi_n(x) \rangle$.
- “linear kernel” $\mathcal{K}(x, x') = \langle \Phi_n(x), \Phi_n(x') \rangle = \int_\Omega \langle x_n(u), x'_n(u) \rangle du$. 
Discretization and signal preservation

$I_1 : \Omega_1 \rightarrow \mathcal{H}_1$

$I_{0.5} : \Omega_0 \rightarrow \mathcal{H}_1$

$I_0(\omega_0) \in \mathcal{H}_0$

$I_1(\omega_2) \in \mathcal{H}_1$

$I_{0.5}(\omega_1) = \varphi_1(P_{\omega_1}) \in \mathcal{H}_1$

Kernel trick

$P_{\omega_1} \in \mathcal{P}_0$ (patch)
Discretization and signal preservation

- $\bar{x}_k$: subsampling factor $s_k$ after pooling with scale $\sigma_k \approx s_k$:

  $$\bar{x}_k[n] = A_k M_k P_k \bar{x}_{k-1}[ns_k]$$

- **Claim**: We can recover $\bar{x}_{k-1}$ from $\bar{x}_k$ if **subsampling** $s_k \leq$ patch size
Discretization and signal preservation

- $\tilde{x}_k$: subsampling factor $s_k$ after pooling with scale $\sigma_k \approx s_k$:

$$\tilde{x}_k[n] = A_k M_k P_k \tilde{x}_{k-1}[ns_k]$$

- **Claim**: We can recover $\tilde{x}_{k-1}$ from $\tilde{x}_k$ if **subsampling** $s_k \leq$ **patch size**

- **How?** Kernels! Recover patches with **linear functions** (contained in RKHS)

$$\langle f_w, M_k P_k x(u) \rangle = f_w(P_k x(u)) = \langle w, P_k x(u) \rangle$$
Signal recovery: example in 1D

\[ x_{k-1} \]

\[ A_k x_{k-1} \]

deconvolution

recovery with linear measurements

\[ x_k \]

downsampling

\[ A_k M_k P_k x_{k-1} \]

linear pooling

\[ M_k P_k x_{k-1} \]

dot-product kernel

\[ x_{k-1} \]

\[ P_k x_{k-1}(u) \in P_k \]
Outline

1. Construction of the Convolutional Representation

2. Invariance and Stability

3. Learning Aspects: Model Complexity of CNNs

4. Regularizing with the RKHS norm
Signal recovery: example in 1D

\[ x_{k-1} \]

\[ A_k x_{k-1} \]

recovery with linear measurements

\[ x_k \]

\[ A_k M_k P_k x_{k-1} \]

downsampling

\[ M_k P_k x_{k-1} \]

dot-product kernel

\[ x_{k-1} \]

\[ P_k x_{k-1}(u) \in P_k \]
Discretization and signal preservation

\[ I_1 : \Omega_1 \rightarrow \mathcal{H}_1 \]
\[ I_{0.5} : \Omega_0 \rightarrow \mathcal{H}_1 \]
\[ I_0 : \Omega_0 \rightarrow \mathcal{H}_0 \]

Linear pooling

\[ I_1(\omega_2) \in \mathcal{H}_1 \]
\[ I_{0.5}(\omega_1) = \varphi_1(P_{\omega_1}) \in \mathcal{H}_1 \]

Kernel trick

\[ P_{\omega_1} \in \mathcal{P}_0 \text{ (patch)} \]

\[ I_0(\omega_0) \in \mathcal{H}_0 \]
Discretization and signal preservation

- $\tilde{x}_k$: subsampling factor $s_k$ after pooling with scale $\sigma_k \approx s_k$:

$$\tilde{x}_k[n] = A_k M_k P_k \tilde{x}_{k-1}[ns_k]$$
Stability to deformations: definitions

- $\tau : \Omega \rightarrow \Omega$: $C^1$-diffeomorphism
- $L_\tau x(u) = x(u - \tau(u))$: action operator
- Much richer group of transformations than translations

- Studied for wavelet-based scattering transform (Mallat, 2012; Bruna and Mallat, 2013)
Stability to deformations: definitions

- Representation $\Phi(\cdot)$ is **stable** (Mallat, 2012) if:

  $$\|\Phi(L_{\tau}x) - \Phi(x)\| \leq (C_1\|\nabla \tau\|_{\infty} + C_2\|\tau\|_{\infty})\|x\|$$

- $\|\nabla \tau\|_{\infty} = \sup_u \|\nabla \tau(u)\|$ controls deformation
- $\|\tau\|_{\infty} = \sup_u |\tau(u)|$ controls translation
- $C_2 \to 0$: translation invariance
Warmup: translation invariance

- Representation:

  \[ \Phi_n(x) := A_n M_n P_n A_{n-1} M_{n-1} P_{n-1} \cdots A_1 M_1 P_1 x. \]

- Translation: \( L_c x(u) = x(u - c) \)
Warmup: translation invariance

- Representation:
  \[ \Phi_n(x) := A_n M_n P_n A_{n-1} M_{n-1} P_{n-1} \cdots A_1 M_1 P_1 x. \]

- Translation: \( L_c x(u) = x(u - c) \)

- Equivariance - all operators commute with \( L_c \): \( \Box L_c = L_c \Box \)

\[
\| \Phi_n(L_c x) - \Phi_n(x) \| = \| L_c \Phi_n(x) - \Phi_n(x) \|
\[
\leq \| L_c A_n - A_n \| \cdot \| x \|
\]
Warmup: translation invariance

- Representation:
  \[ \Phi_n(x) := A_n M_n P_n A_{n-1} M_{n-1} P_{n-1} \cdots A_1 M_1 P_1 x. \]
- Translation: \( L_c x(u) = x(u - c) \)
- Equivariance - all operators commute with \( L_c \): \( \square L_c = L_c \square \)
  \[ \| \Phi_n(L_c x) - \Phi_n(x) \| = \| L_c \Phi_n(x) - \Phi_n(x) \| \leq \| L_c A_n - A_n \| \cdot \| x \| \]
- Mallat (2012): \( \| L_\tau A_n - A_n \| \leq \frac{C_2}{\sigma_n} \| \tau \|_\infty \)
Warmup: translation invariance

- Representation: 
  \[ \Phi_n(x) := A_n M_n P_n A_{n-1} M_{n-1} P_{n-1} \cdots A_1 M_1 P_1 x. \]

- Translation: 
  \[ L_c x(u) = x(u - c) \]

- Equivariance - all operators commute with \( L_c \): \( \square L_c = L_c \square \)

  \[
  \| \Phi_n(L_c x) - \Phi_n(x) \| = \| L_c \Phi_n(x) - \Phi_n(x) \| \\
  \leq \| L_c A_n - A_n \| \cdot \| x \| 
  \]

- Mallat (2012): \( \| L_c A_n - A_n \| \leq \frac{C_2}{\sigma_n} c \)
Stability to deformations

- Representation:
  \[ \Phi_n(x) := A_n M_n P_n A_{n-1} M_{n-1} P_{n-1} \cdots A_1 M_1 P_1 x. \]

- Patch extraction \( P_k \) and pooling \( A_k \) do not commute with \( L_T \)!

- \[ \| A_k L_T - L_T A_k \| \leq C_1 \| \nabla T \|_\infty \] (from Mallat, 2012)
Stability to deformations

- Representation:
  \[ \Phi_n(x) := A_n M_n P_n A_{n-1} M_{n-1} P_{n-1} \cdots A_1 M_1 P_1 x. \]

- Patch extraction \( P_k \) and pooling \( A_k \) do not commute with \( L_\tau \)!

- \( \|[A_k, L_\tau]\| \leq C_1 \| \nabla \tau \|_\infty \) (from Mallat, 2012)
Stability to deformations

• Representation:
  \[ \Phi_n(x) := A_1 M_1 P_1 A_{n-1} M_{n-1} P_{n-1} \cdots A_1 M_1 P_1 x. \]

• Patch extraction \( P_k \) and pooling \( A_k \) do not commute with \( L_\tau \)!

• \( \| [A_k, L_\tau] \| \leq C_1 \| \nabla \tau \|_\infty \) (from Mallat, 2012)

• But: \([P_k, L_\tau]\) is unstable at high frequencies!
Stability to deformations

- Representation:
  \[ \Phi_n(x) := A_n M_n P_n A_{n-1} M_{n-1} P_{n-1} \cdots A_1 M_1 P_1 A_0 x. \]

- Patch extraction \( P_k \) and pooling \( A_k \) do not commute with \( L_\tau \)!

- \( \|[A_k, L_\tau]\| \leq C_1 \|\nabla \tau\|_\infty \) (from Mallat, 2012)

- But: \( [P_k, L_\tau] \) is unstable at high frequencies!

- Adapt to current layer resolution, patch size controlled by \( \sigma_{k-1} \):
  \[ \|[P_k A_{k-1}, L_\tau]\| \leq C_{1,\beta} \|\nabla \tau\|_\infty \quad \sup_{u \in S_k} |u| \leq \beta \sigma_{k-1} \]
Stability to deformations

- Representation:
  \[ \Phi_n(x) := A_n M_n P_n A_{n-1} M_{n-1} P_{n-1} \cdots A_1 M_1 P_1 A_0 x. \]

- Patch extraction \( P_k \) and pooling \( A_k \) do not commute with \( L_\tau \)!

- \( \|[A_k, L_\tau]\| \leq C_1 \|\nabla \tau\|_\infty \) (from Mallat, 2012)

- But: \( [P_k, L_\tau] \) is unstable at high frequencies!

- Adapt to current layer resolution, patch size controlled by \( \sigma_{k-1} \):
  \[ \|[P_k A_{k-1}, L_\tau]\| \leq C_{1,\beta} \|\nabla \tau\|_\infty \sup_{u \in S_k} |u| \leq \beta \sigma_{k-1} \]

- \( C_{1,\beta} \) grows as \( \beta^{d+1} \) \( \Longrightarrow \) more stable with small patches (e.g., 3x3, VGG et al.)
Stability to deformations: final result

**Theorem**

If \( \| \nabla \tau \|_\infty \leq 1/2 \),

\[
\| \Phi_n(L_\tau x) - \Phi_n(x) \| \leq \left( C_{1,\beta} (n + 1) \| \nabla \tau \|_\infty + \frac{C_2}{\sigma_n} \| \tau \|_\infty \right) \| x \|
\]

- translation invariance: large \( \sigma_n \)
- stability: small patch sizes
- signal preservation: subsampling factor \( \approx \) patch size
- \( \Longrightarrow \) needs several layers
Stability to deformations: final result

**Theorem**

If \( \|\nabla \tau\|_\infty \leq 1/2 \),

\[
\|\Phi_n(L_\tau x) - \Phi_n(x)\| \leq \prod_k \rho_k \left( C_{1,\beta} (n + 1) \|\nabla \tau\|_\infty + \frac{C_2}{\sigma_n} \|\tau\|_\infty \right) \|x\|
\]

- translation invariance: large \( \sigma_n \)
- stability: small patch sizes
- signal preservation: subsampling factor \( \approx \) patch size
- \( \implies \text{needs several layers} \)
- (also valid for generic CNNs with ReLUs: multiply by \( \prod_k \rho_k = \prod_k \|W_k\| \), but no direct signal preservation).
Beyond the translation group

Global invariance to other groups?

- Rotations, reflections, roto-translations, ...
- Group action $L_g x(u) = x(g^{-1} u)$
- **Equivariance** in inner layers + (global) pooling in last layer
- Similar construction to Cohen and Welling (2016); Kondor and Trivedi (2018)
G-equivariant layer construction

- Feature maps $x(u)$ defined on $u \in G$ ($G$: locally compact group)
  - Input needs special definition when $G \neq \Omega$
- Patch extraction:
  \[ P_x(u) = (x(\mu v))_{v \in S} \]
- Non-linear mapping: equivariant because pointwise!
- Pooling ($\mu$: left-invariant Haar measure):
  \[ A_x(u) = \int_G x(\mu v) h(v) d\mu(v) = \int_G x(v) h(u^{-1}v) d\mu(v) \]
Group invariance and stability

**Roto-translation group** \( G = \mathbb{R}^2 \rtimes SO(2) \) (translations + rotations)

- **Stability** w.r.t. translation group
- **Global invariance** to rotations (only global pooling at final layer)
  - Inner layers: patches and pooling only on translation group
  - Last layer: global pooling on rotations
  - Cohen and Welling (2016): pooling on rotations in inner layers hurts performance on Rotated MNIST
Stability to deformations: final result

**Theorem**

If $\|\nabla \tau\|_\infty \leq 1/2$,

$$\| \Phi_n(L\tau x) - \Phi_n(x) \| \leq \prod_k \rho_k \left( C_{1,\beta} (n+1) \|\nabla \tau\|_\infty + \frac{C_2}{\sigma_n} \|\tau\|_\infty \right) \|x\|$$

- translation invariance: large $\sigma_n$
- stability: small patch sizes
- signal preservation: subsampling factor $\approx$ patch size
- $\implies$ needs several layers
- (also valid for generic CNNs with ReLUs: multiply by $\prod_k \rho_k = \prod_k \|W_k\|$, but no direct signal preservation).
Signal recovery: example in 1D

$x_{k-1}$

deconvolution

$A_k x_{k-1}$

recovery with linear measurements

$x_k$

downsampling

$A_k M_k P_k x_{k-1}$

linear pooling

$M_k P_k x_{k-1}$

dot-product kernel

$x_{k-1}$

$P_k x_{k-1}(u) \in P_k$
Stability to deformations: definitions

- Representation $\Phi(\cdot)$ is **stable** (Mallat, 2012) if:

\[ \|\Phi(L_\tau x) - \Phi(x)\| \leq (C_1 \|\nabla \tau\|_\infty + C_2 \|\tau\|_\infty)\|x\| \]

- $\|\nabla \tau\|_\infty = \sup_u \|\nabla \tau(u)\|$ controls deformation
- $\|\tau\|_\infty = \sup_u |\tau(u)|$ controls translation
- $C_2 \to 0$: translation invariance
Outline

1. Construction of the Convolutional Representation
2. Invariance and Stability
3. Learning Aspects: Model Complexity of CNNs
4. Regularizing with the RKHS norm
RKHS of patch kernels $K_k$

$$K_k(z, z') = \|z\|\|z'\|\kappa\left(\frac{\langle z, z' \rangle}{\|z\|\|z'\|}\right), \quad \kappa(u) = \sum_{j=0}^{\infty} b_j u^j$$

- RKHS contains **homogeneous functions**:
  $$f : z \mapsto \|z\|\sigma(\langle g, z \rangle / \|z\|)$$

Homogeneous version of (Zhang et al., 2016, 2017)
RKHS of patch kernels $K_k$

$$K_k(z, z') = \|z\| \|z'\| \kappa \left( \frac{\langle z, z' \rangle}{\|z\| \|z'\|} \right), \quad \kappa(u) = \sum_{j=0}^{\infty} b_j u^j$$

- RKHS contains **homogeneous functions**:
  $$f : z \mapsto \|z\| \sigma(\langle g, z \rangle / \|z\|)$$
- **Smooth activations**: $\sigma(u) = \sum_{j=0}^{\infty} a_j u^j$
- **Norm**: $\|f\|_{\mathcal{H}_k}^2 \leq C_\sigma^2(\|g\|^2) = \sum_{j=0}^{\infty} \frac{a_j^2}{b_j} \|g\|^2 < \infty$

Homogeneous version of (Zhang et al., 2016, 2017)
RKHS of patch kernels $K_k$

Examples:
- $\sigma(u) = u$ (linear): $C_\sigma^2(\lambda^2) = O(\lambda^2)$
- $\sigma(u) = u^p$ (polynomial): $C_\sigma^2(\lambda^2) = O(\lambda^{2p})$
- $\sigma \approx \sin$, sigmoid, smooth ReLU: $C_\sigma^2(\lambda^2) = O(e^{c\lambda^2})$

$$f : x \mapsto \sigma(x)$$

$$f : x \mapsto |x| \sigma(wx/|x|)$$
RKHS of patch kernels $K_k$

$$K_k(z, z') = \|z\| \|z'\| \kappa\left(\frac{\langle z, z' \rangle}{\|z\| \|z'\|}\right), \quad \kappa(u) = \sum_{j=0}^{\infty} b_j u^j$$

- RKHS contains **homogeneous functions**:

  $$f : z \mapsto \|z\| \sigma(\langle g, z \rangle / \|z\|)$$

- **Smooth activations**: $\sigma(u) = \sum_{j=0}^{\infty} a_j u^j$

- **Norm**: $\|f\|_{\mathcal{H}_k}^2 \leq C^2_\sigma(\|g\|^2) = \sum_{j=0}^{\infty} \frac{a_j^2}{b_j} \|g\|^2 < \infty$

Homogeneous version of (Zhang et al., 2016, 2017)
RKHS of patch kernels $K_k$

Examples:
- $\sigma(u) = u$ (linear): $C_\sigma^2(\lambda^2) = O(\lambda^2)$
- $\sigma(u) = u^p$ (polynomial): $C_\sigma^2(\lambda^2) = O(\lambda^{2p})$
- $\sigma \approx \sin, \text{sigmoid}, \text{smooth ReLU}: C_\sigma^2(\lambda^2) = O(e^{c\lambda^2})$
Constructing a CNN in the RKHS $\mathcal{H}_K$

- Consider a CNN with filters $W_k^{ij}(u), u \in S_k$
- "Smooth homogeneous" activations $\sigma$
- The CNN can be constructed hierarchically in $\mathcal{H}_K$
- Norm upper bound:

$$\|f_\sigma\|_{\mathcal{H}}^2 \leq \|W_{n+1}\|_2^2 C_\sigma^2(\|W_n\|_2^2 C_\sigma^2(\|W_{n-1}\|_2^2 C_\sigma^2(\ldots))))$$
Constructing a CNN in the RKHS $\mathcal{H}_K$

- Consider a CNN with filters $W^j_k(u), u \in S_k$
- "Smooth homogeneous" activations $\sigma$
- The CNN can be constructed hierarchically in $\mathcal{H}_K$
- Norm upper bound (linear layers):
  \[ \|f_\sigma\|_{\mathcal{H}_L}^2 \leq \|W_{n+1}\|_2^2 \cdot \|W_n\|_2^2 \cdot \|W_{n-1}\|_2^2 \cdots \|W_1\|_2^2 \]
- Linear layers: product of spectral norms
Link with generalization

- Simple bound on Rademacher complexity for linear/kernel methods:

\[ \mathcal{F}_B = \{ f \in \mathcal{H}_K, \| f \|_\mathcal{H} \leq B \} \implies \text{Rad}_N(\mathcal{F}_B) \leq O \left( \frac{BR}{\sqrt{N}} \right) \]
Link with generalization

- Simple bound on Rademacher complexity for linear/kernel methods:

\[ \mathcal{F}_B = \{ f \in \mathcal{H}_K, \| f \|_\mathcal{H} \leq B \} \implies \text{Rad}_N(\mathcal{F}_B) \leq O \left( \frac{BR}{\sqrt{N}} \right) \]

- Leads to margin bound \( O(\| \hat{f}_N \|_\mathcal{H} R / \gamma \sqrt{N}) \) for a learned CNN \( \hat{f}_N \) with margin (confidence) \( \gamma > 0 \)

- Related to generalization bounds for neural networks based on product of spectral norms (e.g., Bartlett et al., 2017; Neyshabur et al., 2018)
Outline

1. Construction of the Convolutional Representation
2. Invariance and Stability
3. Learning Aspects: Model Complexity of CNNs
4. Regularizing with the RKHS norm
Regularizing with the RKHS norm in practice

Deep learning struggles with **small datasets** and **adversarial examples**.
Regularizing with the RKHS norm in practice

Can we obtain better models through regularization?

- Controlling **upper bounds**: spectral norm penalties/constraints
- Controlling **lower bounds** using $\|f\|_H = \sup_{\|u\|_H \leq 1} \langle f, u \rangle$

(Bietti, Mialon, Chen, and Mairal, 2019)
Regularizing with the RKHS norm in practice

Can we obtain better models through regularization?

- Controlling upper bounds: spectral norm penalties/constraints
- Controlling lower bounds using \( \|f\|_H = \sup_{\|u\|_H \leq 1} \langle f, u \rangle \)
- \( \implies \) consider tractable subsets of the unit ball

\[
\|f\|_H \geq \sup_{x, \|\delta\|_1 \leq 1} \langle f, \Phi(x + \delta) - \Phi(x) \rangle_H \quad \text{(adversarial perturbations)}
\]
Regularizing with the RKHS norm in practice

Can we obtain better models through regularization?

- Controlling **upper bounds**: spectral norm penalties/constraints
- Controlling **lower bounds** using $\|f\|_{\mathcal{H}} = \sup_{\|u\|_{\mathcal{H}} \leq 1} \langle f, u \rangle$
- $\implies$ consider tractable subsets of the unit ball

\[ \|f\|_{\mathcal{H}} \geq \sup_{x, \|\delta\| \leq 1} f(x + \delta) - f(x) \quad \text{(adversarial perturbations)} \]

(Rietti, Mialon, Chen, and Mairal, 2018)
Regularizing with the RKHS norm in practice

Can we obtain better models through regularization?

- Controlling **upper bounds**: spectral norm penalties/constraints
- Controlling **lower bounds** using $\|f\|_\mathcal{H} = \sup_{\|u\|_\mathcal{H} \leq 1} \langle f, u \rangle$

$\implies$ consider tractable subsets of the unit ball

$$\|f\|_\mathcal{H} \geq \sup_{x, \|\delta\| \leq 1} f(x + \delta) - f(x) \quad (\text{adversarial perturbations})$$

$$\|f\|_\mathcal{H} \geq \sup_{x, \|\tau\| \leq C} f(L_\tau x) - f(x) \quad (\text{adversarial deformations})$$

(Bietti, Mialon, Chen, and Mairal, 2018)
Regularizing with the RKHS norm in practice

Can we obtain better models through regularization?

- Controlling **upper bounds**: spectral norm penalties/constraints
- Controlling **lower bounds** using $\|f\|_{\mathcal{H}} = \sup_{\|u\|_{\mathcal{H}} \leq 1} \langle f, u \rangle$
- $\implies$ consider tractable subsets of the unit ball

\[
\|f\|_{\mathcal{H}} \geq \sup_{x, \|\delta\| \leq 1} f(x + \delta) - f(x) \quad \text{(adversarial perturbations)}
\]
\[
\|f\|_{\mathcal{H}} \geq \sup_{x, \|\tau\| \leq C} f(L_\tau x) - f(x) \quad \text{(adversarial deformations)}
\]
\[
\|f\|_{\mathcal{H}} \geq \sup_{x} \|\nabla f(x)\|_2 \quad \text{(gradient penalty)}
\]

(Rietti, Mialon, Chen, and Mairal, 2019)
Regularizing with the RKHS norm in practice

Can we obtain better models through regularization?

- Controlling **upper bounds**: spectral norm penalties/constraints
- Controlling **lower bounds** using $\|f\|_\mathcal{H} = \sup_{\|u\|_\mathcal{H} \leq 1} \langle f, u \rangle$
- $\implies$ consider tractable subsets of the unit ball
  
  $\|f\|_\mathcal{H} \geq \sup_{x, \|\delta\| \leq 1} f(x + \delta) - f(x)$ (adversarial perturbations)
  
  $\|f\|_\mathcal{H} \geq \sup_{x, \|\tau\| \leq C} f(L_\tau x) - f(x)$ (adversarial deformations)
  
  $\|f\|_\mathcal{H} \geq \sup_x \|\nabla f(x)\|_2$ (gradient penalty)

- Best performance by combining upper + lower bound approaches

(Bietti, Mialon, Chen, and Mairal, 2019)
Regularizing with the RKHS norm in practice

Can we obtain better models through regularization?

Table 2. Regularization on 300 or 1000 examples from MNIST, using deformations from Infinite MNIST. (*) indicates that random deformations were included as training examples, while $\|f\|_T^2$ and $\|D_\tau f\|_2^2$ use them as part of the regularization penalty.

<table>
<thead>
<tr>
<th>Method</th>
<th>300 VGG</th>
<th>1k VGG</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weight decay</td>
<td>89.32</td>
<td>94.08</td>
</tr>
<tr>
<td>SN projection</td>
<td>90.69</td>
<td>95.01</td>
</tr>
<tr>
<td>grad-$\ell_2$</td>
<td>93.63</td>
<td>96.67</td>
</tr>
<tr>
<td>$|f|_0^2$ penalty</td>
<td>94.17</td>
<td>96.99</td>
</tr>
<tr>
<td>$|\nabla f|_2^2$ penalty</td>
<td>94.08</td>
<td>96.82</td>
</tr>
<tr>
<td>Weight decay (*)</td>
<td>92.41</td>
<td>95.64</td>
</tr>
<tr>
<td>grad-$\ell_2$ (*)</td>
<td>95.05</td>
<td>97.48</td>
</tr>
<tr>
<td>$|D_\tau f|_2^2$ penalty</td>
<td>94.18</td>
<td>96.98</td>
</tr>
<tr>
<td>$|f|_T^2$ penalty</td>
<td>94.42</td>
<td>97.13</td>
</tr>
<tr>
<td>$|f|_T^2 + |\nabla f|_2^2$</td>
<td>94.75</td>
<td>97.40</td>
</tr>
<tr>
<td>$|f|_T^2 + |f|_0^2$</td>
<td>95.23</td>
<td>97.66</td>
</tr>
<tr>
<td>$|f|_T^2 + |f|_0^2$ (*)</td>
<td>95.53</td>
<td>97.56</td>
</tr>
<tr>
<td>$|f|_T^2 + |f|_0^2$ + SN proj</td>
<td>95.20</td>
<td>97.60</td>
</tr>
<tr>
<td>$|f|_T^2 + |f|_0^2$ + SN proj (*)</td>
<td>95.40</td>
<td>97.77</td>
</tr>
</tbody>
</table>

(Bietti, Mialon, Chen, and Mairal, 2018)
Regularizing with the RKHS norm in practice

Can we obtain better models through regularization?

Table 3. Regularization on protein homology detection tasks, with or without data augmentation (DA). Fixed hyperparameters are selected using the first half of the datasets, and we report the average auROC50 score on the second half.

<table>
<thead>
<tr>
<th>Method</th>
<th>No DA</th>
<th>DA</th>
</tr>
</thead>
<tbody>
<tr>
<td>No weight decay</td>
<td>0.446</td>
<td>0.500</td>
</tr>
<tr>
<td>Weight decay</td>
<td>0.501</td>
<td>0.546</td>
</tr>
<tr>
<td>SN proj</td>
<td>0.591</td>
<td>0.632</td>
</tr>
<tr>
<td>PGD-(\ell_2)</td>
<td>0.575</td>
<td>0.595</td>
</tr>
<tr>
<td>grad-(\ell_2)</td>
<td>0.540</td>
<td>0.552</td>
</tr>
<tr>
<td>(|f|^2)</td>
<td>0.600</td>
<td>0.608</td>
</tr>
<tr>
<td>(|\nabla f|^2)</td>
<td>0.585</td>
<td>0.611</td>
</tr>
<tr>
<td>PGD-(\ell_2) + SN proj</td>
<td>0.596</td>
<td>0.627</td>
</tr>
<tr>
<td>grad-(\ell_2) + SN proj</td>
<td>0.592</td>
<td>0.624</td>
</tr>
<tr>
<td>(|f|^2) + SN proj</td>
<td>0.630</td>
<td>0.644</td>
</tr>
<tr>
<td>(|\nabla f|^2) + SN proj</td>
<td>0.603</td>
<td>0.625</td>
</tr>
</tbody>
</table>

(Bietti, Mialon, Chen, and Mairal, 2019)
Regularization for robustness

- Robust optimization yields another lower bound (hinge/logistic loss)

\[ \frac{1}{N} \sum_{i=1}^{N} \sup_{\|\delta\|_2 \leq \epsilon} \ell(y_i, f(x_i + \delta)) \leq \frac{1}{N} \sum_{i=1}^{N} \ell(y_i, f(x_i)) + \epsilon \|f\|_H \]

- Controlling $\|f\|_H$ allows a more global form of robustness
- Leads to margin bounds for adversarial generalization with $\ell_2$ perturbations
  - Using $\|f\|_H \geq \|f\|_{\text{Lip}}$ near the margin
- But, may cause a loss in accuracy in practice

(Bietti, Mialon, Chen, and Mairal, 2019)
Regularization for robustness

Robust vs standard accuracy trade-offs

$l_2, \epsilon_{test} = 0.1$

$l_2, \epsilon_{test} = 1.0$

(Bietti, Mialon, Chen, and Mairal, 2019)
Regularization for robustness

Upper vs lower bounds

(Bietti, Mialon, Chen, and Mairal, 2019)
Deep convolutional representations: conclusions

**Study of generic properties**
- Deformation stability with small patches, adapted to resolution
- Signal preservation when subsampling ≤ patch size
- Group invariance by changing patch extraction and pooling

**Applies to learned models**
- Same quantity \( ||f|| \) controls stability and complexity:
  - “higher capacity” is needed to discriminate small deformations
  - Learning may be “easier” with stable functions
- Better regularization of generic CNNs using RKHS norm

**Links with optimization** (Bietti and Mairal, 2019b)
- Similar kernel (NTK) arises from optimization in a certain regime
- Weaker stability guarantees, but better approximation properties