Theoretical Analysis of Neural Network Models

Kameron Decker Harris

Western Washington University
Computer Science

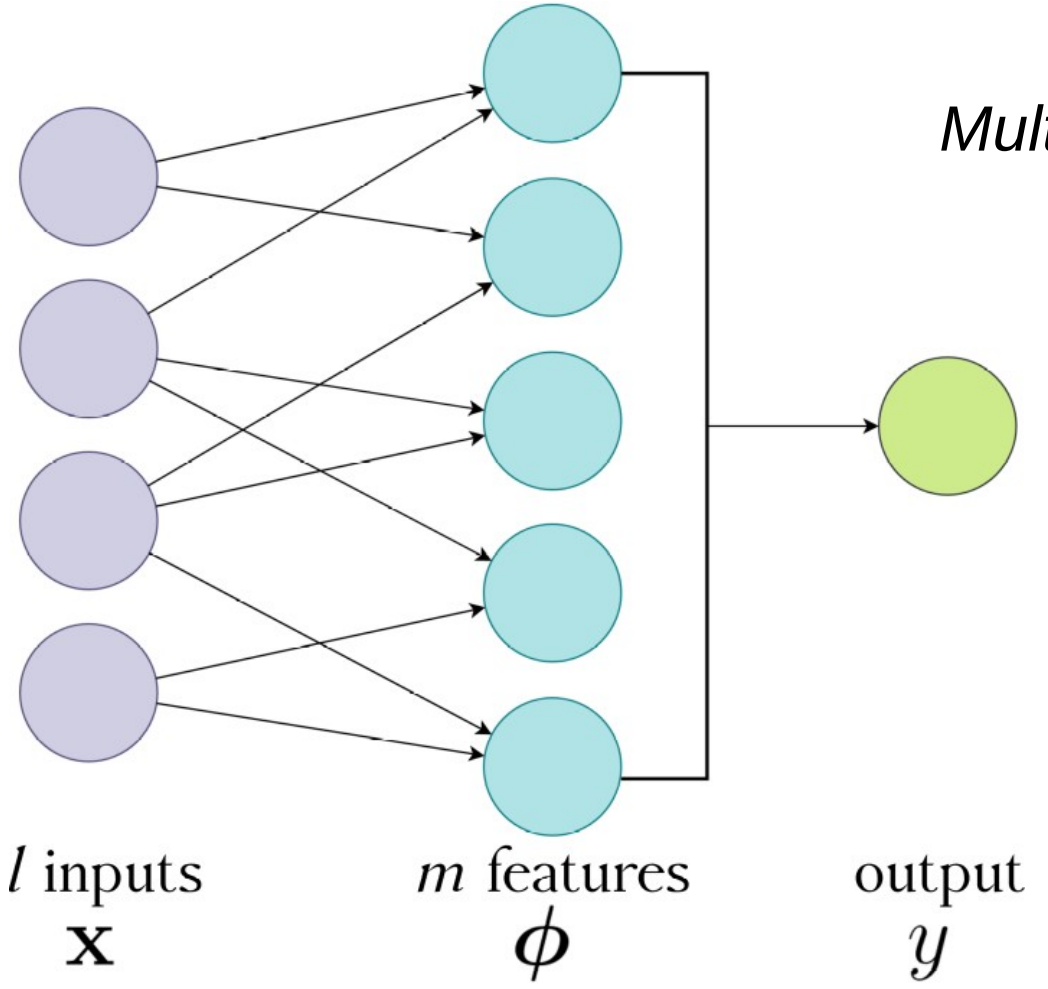
kameron.harris@wwu.edu

pronouns: he/him

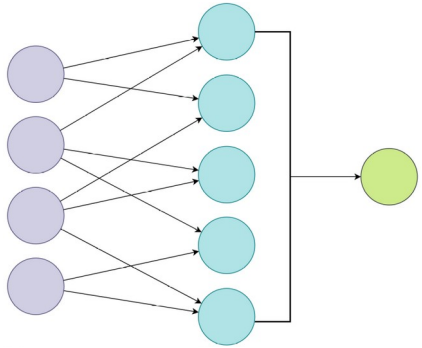
Outline

- Classical theory for ANNs and learning
- Puzzling observations
- Kernel theories of neural networks
- Advantages of “biological” network structure

Multi-layer perceptron

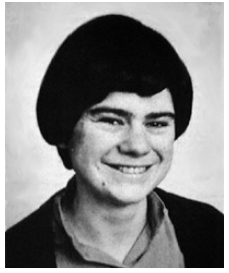


Older theory about neural networks



- Universal function approximation
  - McCulloch & Pitts, 1943: boolean functions
  - Cybenko & others, 1989: 2-layer networks, “nice” functions
  - Not a *statistical result…* does not explain data dependence

- Capacity, learning curves for single neurons & some random models
  - e.g. Gardner & Derrida, Brunel, Sollich, Solla, Sompolinsky, ‘80s-2000s



Elizabeth Gardner

Typical *statistical* learning theory

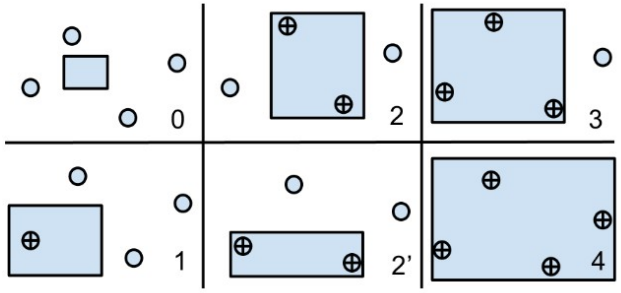
Empirical risk minimization:

$$R_{\text{emp}}(h) = \frac{1}{n} \sum_{i=1}^{n} L(h(x_i), y_i)$$

$$\hat{h} = \arg\min_{h \in \mathcal{H}} R_{\text{emp}}(h)$$

Bounds on error:

$$\text{test - train risk} \leq \frac{\text{complexity}(\mathcal{H})}{\sqrt{n}} + \epsilon$$

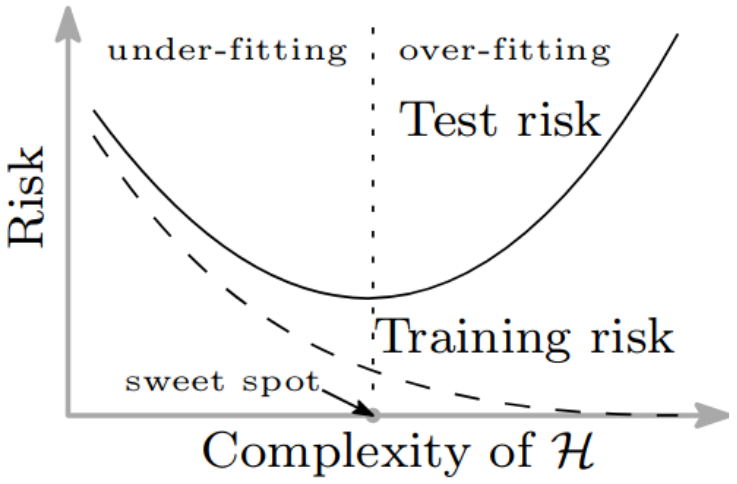


Rademacher, Vapnik-Chervonenkis

See: Shalev-Shwartz & Ben-David (2014)
Shawe-Taylor & Christianini (2004)

https://www.cs.princeton.edu/courses/archive/spring14/cos511/scribe_notes/0220.pdf

Surprising behavior: Double descent



(a) U-shaped "bias-variance" risk curve

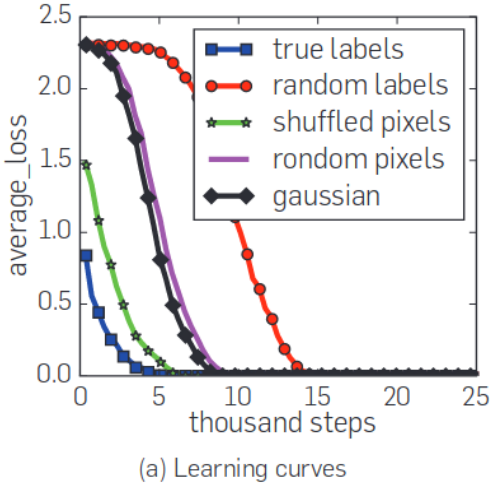
Belkin, Hsu, Ma, Mandal (2018)

Risk = Error

Networks can interpolate *noise*

Understanding Deep Learning (Still) Requires Rethinking Generalization

By Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals (2021)



(a) Learning curves

Fitting random labels =>

Large hypothesis class

Other puzzles & partial solutions

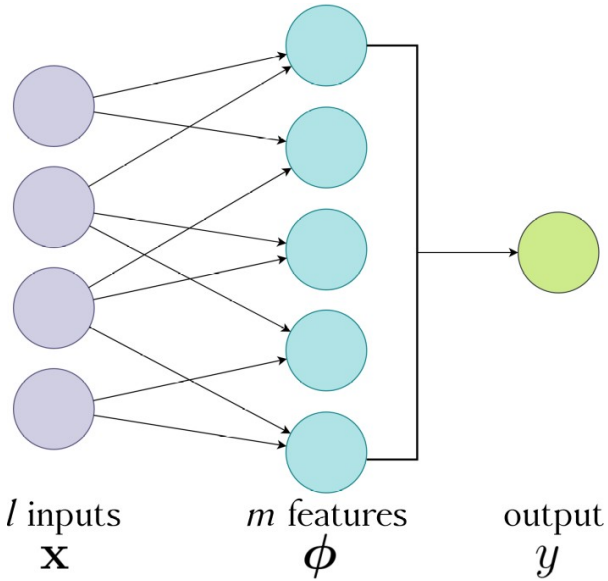
- Optimization in non-convex setting
  - More weights make optimization “easier” (Polyak-Łojasiewicz ineq.)
  - Early-stopping, SGD as implicit control of complexity
- Architecture, structure/function
  - Quantitative theory for CNN, ResNet, etc?
  - **Kernel theory**

Inner products

From random network to kernel

Using the Law of Large Numbers

$$\text{Assume } \mathbf{w}_i \sim \mu \text{ iid random}$$



$$\phi_i(\mathbf{x}) = h(\mathbf{w}_i^\intercal \mathbf{x})$$

Rahimi & Recht, 2008; Neal 1996; Williams 1997

Kernels: inner products in nonlinear space

$$\mathbf{x} \in \mathcal{X} \subset \mathbb{R}^{\ell} \qquad \text{input vector, } \ell\text{-dimensional}$$

$$k(\mathbf{x}, \mathbf{x}') : \mathcal{X} \times \mathcal{X} \to \mathbb{R} \qquad \text{kernel function}$$

$$K = (k(\mathbf{x}_i, \mathbf{x}_j))_{i,j=1}^{n} \qquad \text{kernel matrix}$$

Requirements:

- Symmetric, positive definite function / matrix
- “Nice” function (e.g. continuous)

$$k(\mathbf{x}, \mathbf{x}') = \langle \Psi(\mathbf{x}), \Psi(\mathbf{x}') \rangle$$

“SIMILARITY”

Many ML algorithms built off of this idea

See book by Shawe-Taylor & Christianini

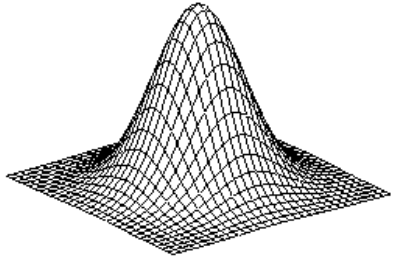
HW2, #2

Examples of kernels

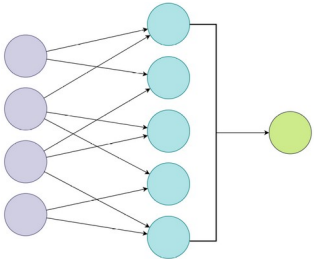
1) Linear $$k(\mathbf{x}, \mathbf{x}') = \langle \mathbf{x}, \mathbf{x}' \rangle = \sum_{i=1}^{\ell} x_i x_i'$$

2) Polynomial $$k(\mathbf{x}, \mathbf{x}') = (c + \langle \mathbf{x}, \mathbf{x}' \rangle)^p$$

3) Radial basis function $$k(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{2\sigma^2}\right)$$



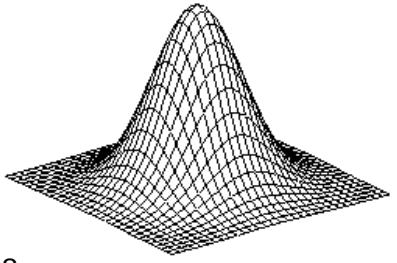
Examples of random feature kernels



Gaussian $\mathbf{w} \sim N(0, \sigma^{-2} I)$ + h = Fourier

$$k(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{2\sigma^2}\right)$$

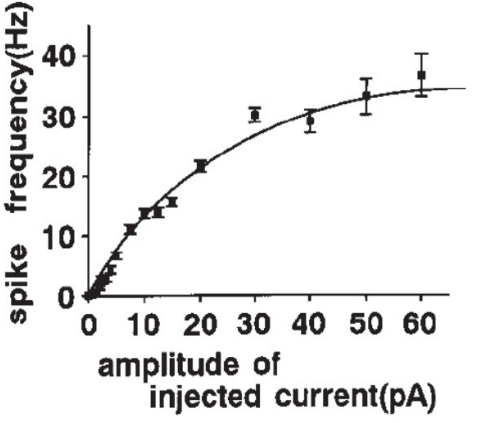
RBF kernel



Rahimi & Recht, 2008

HW2 #5

Also works for “neuronal” transfer functions



Iida & Kashiwayanagi, 1999

Eigendecomposition: Mercer's theorem

Symmetric, positive definite, & continuous *k* on a compact domain:

$$k(\mathbf{x}, \mathbf{x}') = \sum_{i=1}^{\infty} \lambda_i e_i(\mathbf{x}) e_i(\mathbf{x}')$$

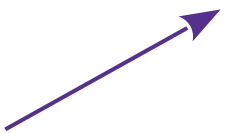
HW2, #3 & #4

linear regression

Abstract geometry of ~~neuronal networks~~

Task

Learn a function
regression / classification

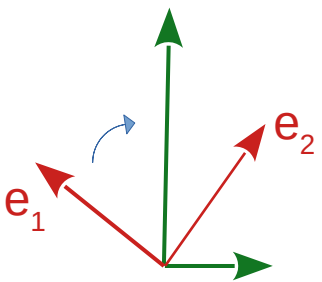


Target function

$f \in L^2$

Network

Structure of weights,
layers, nonlinearities,
learning rules?

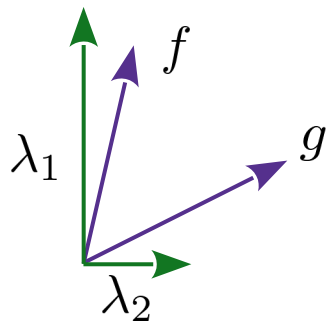


Inner product
between functions
+
Eigendecomposition

$\langle a, b \rangle_{\mathcal{H}}$

Performance

Fitting error,
generalization,
samples needed,
robustness



Targets in kernel
eigen-basis

$\|f\|_{\mathcal{H}} < \|g\|_{\mathcal{H}}$

easy hard

complexity

HW2, #3 & #4

Overview

- **Kernel theory of networks: applications**
  - **Thresholds → filtering in function space**
  - Random receptive fields → basis change & filtering input space

"Task-dependent optimal representations for cerebellar learning"
Submitted
https://www.biorxiv.org/content/10.1101/2022.08.15.504040v1

DeepMath 2021, COSYNE 2021,
CSHL NAISys 2020

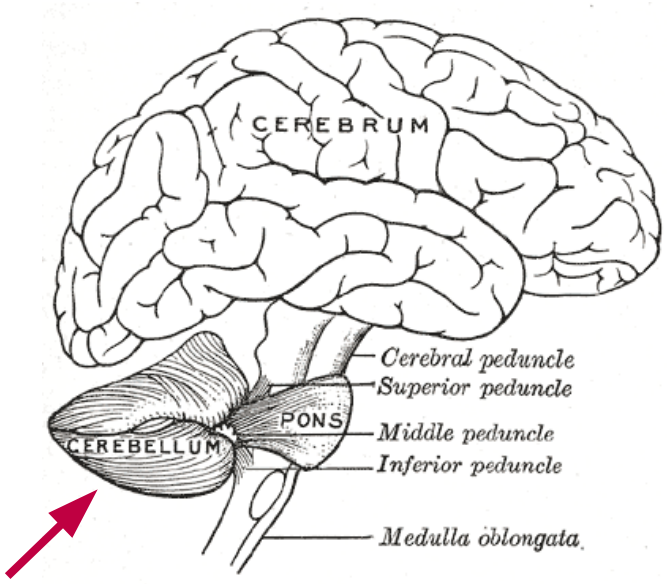
Marjorie Xie

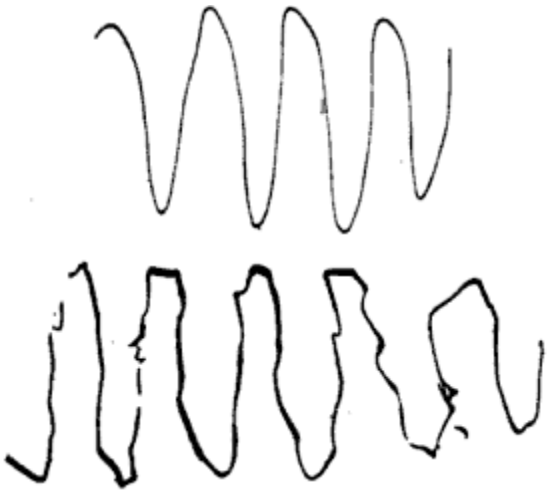
Sam Muscinelli

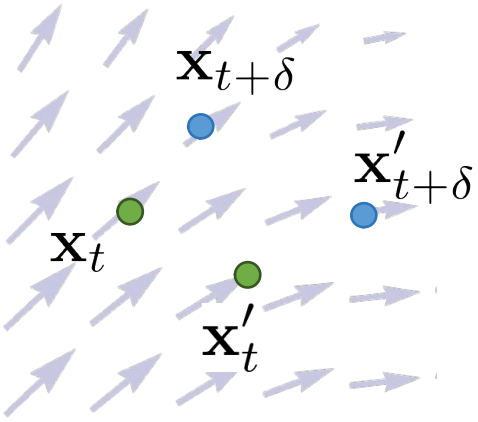
Ashok Litwin-Kumar

Center for Theoretical Neuroscience, Columbia University

The cerebellum







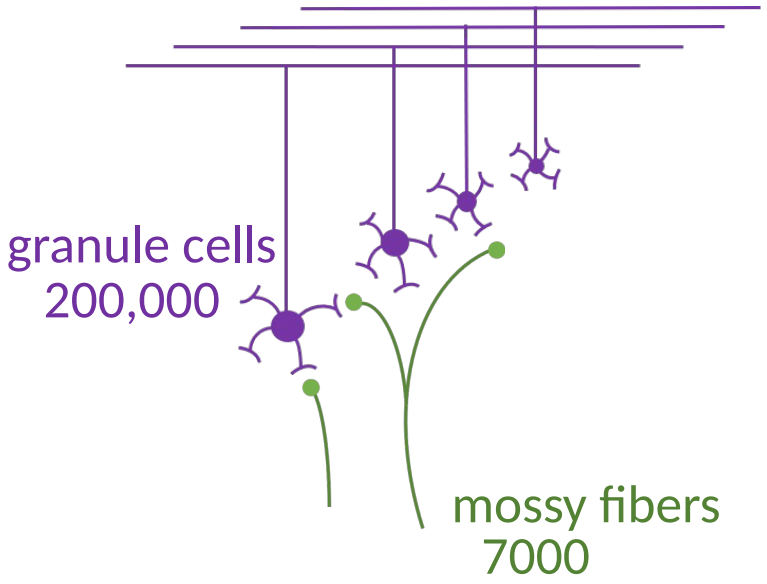
Prediction of *smooth* movement trajectories

(among other functions)

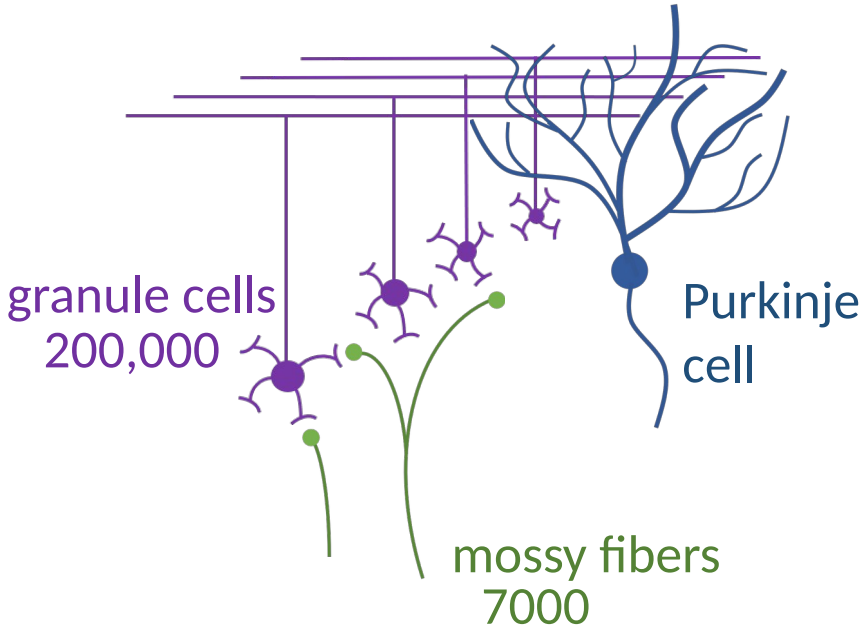
Gray & Carter, 1918; Thomas 1912

Cerebellum architecture

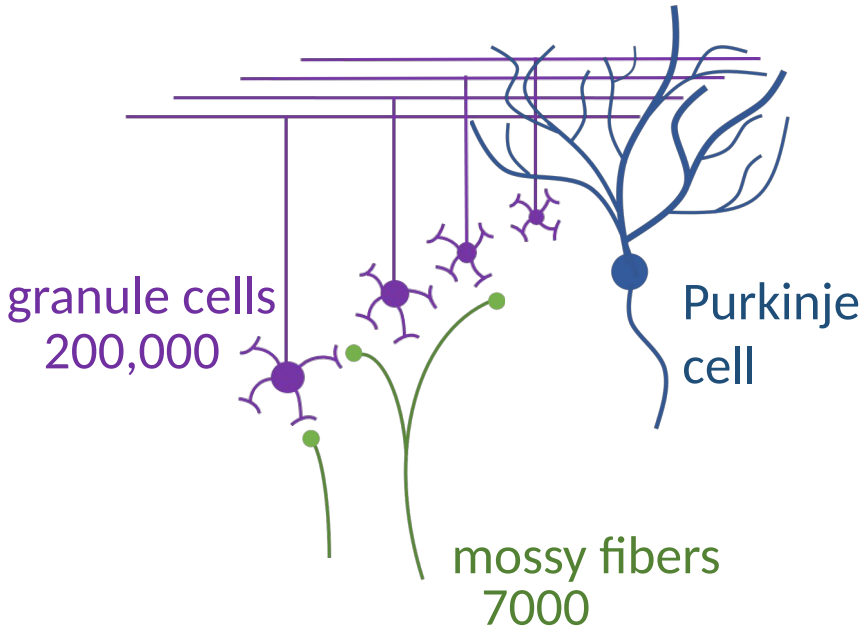
Cerebellum architecture

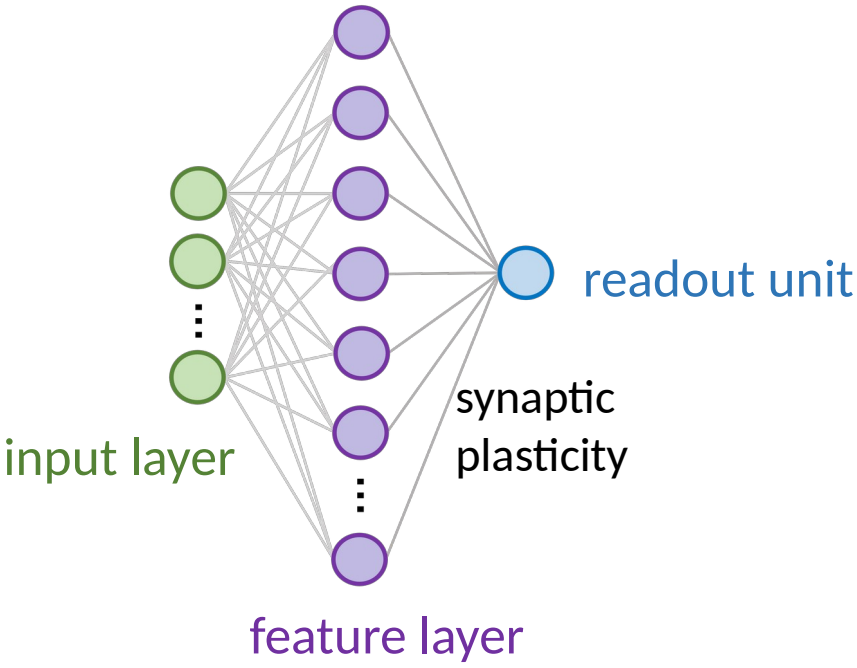


Cerebellum architecture



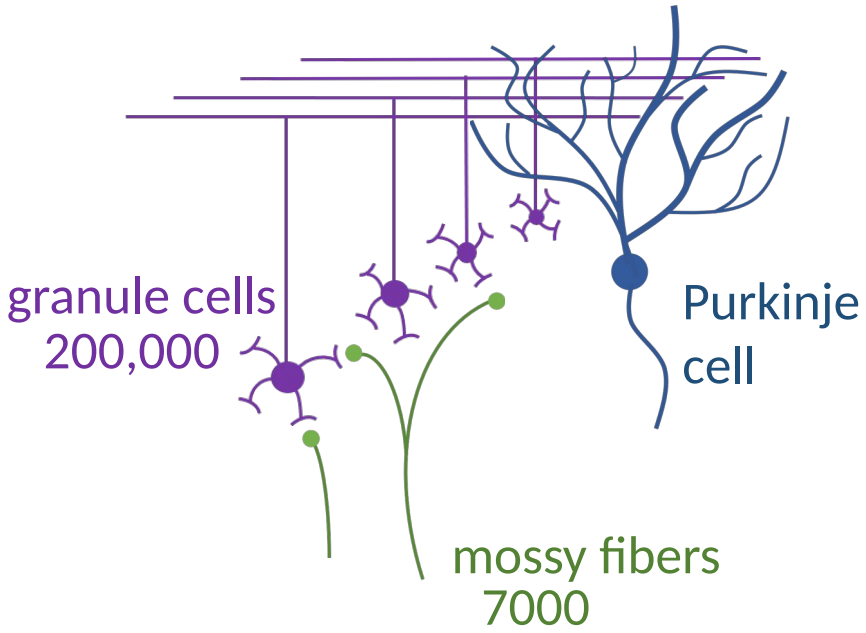
Cerebellum architecture

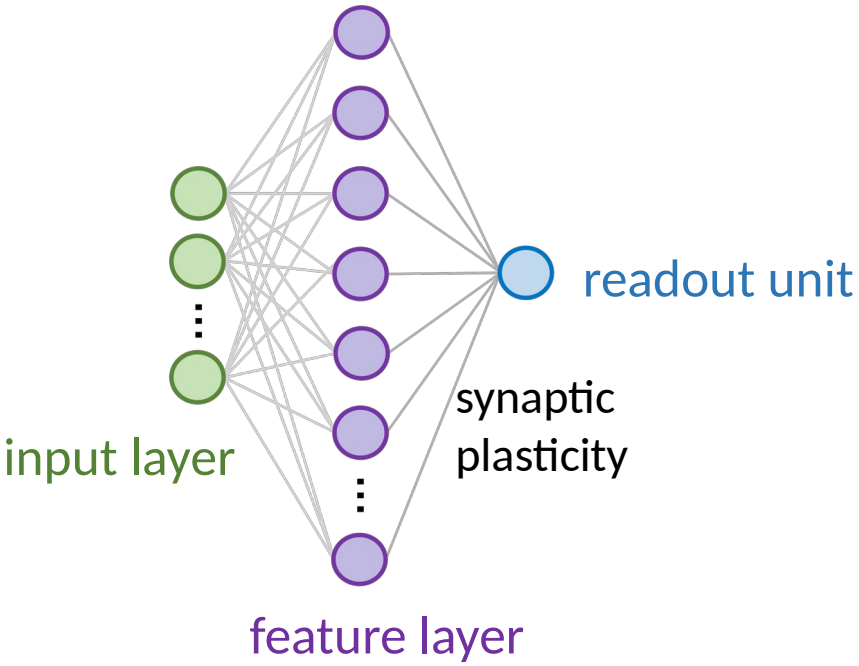




$$\phi_i(\mathbf{x}) = h(\mathbf{w}_i^\intercal \mathbf{x} - \theta)$$

Cerebellum architecture



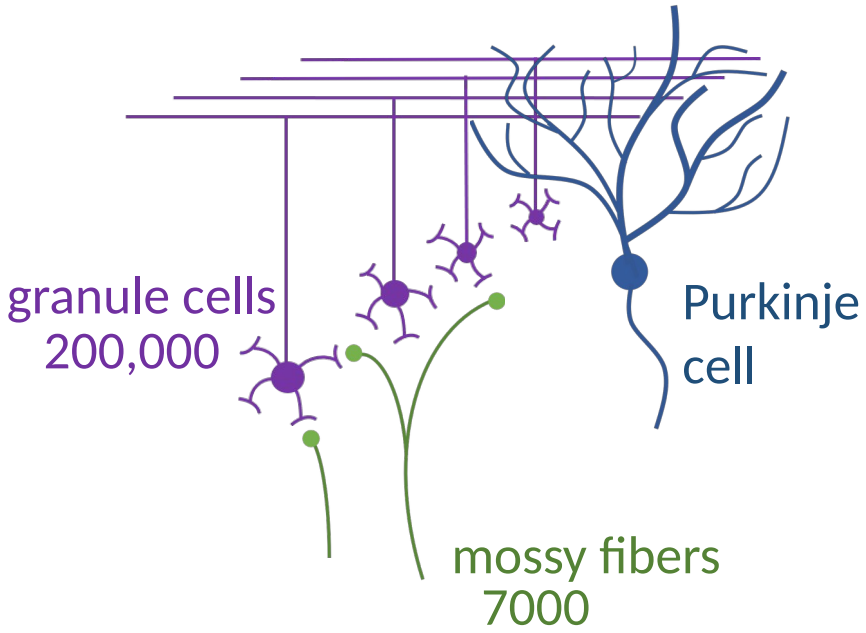


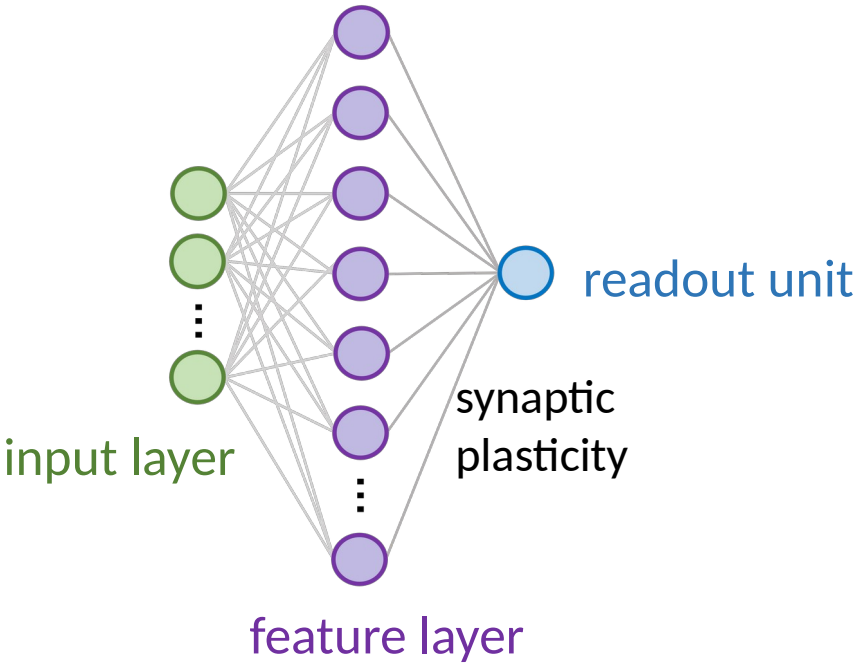
Sparse coding hypothesis:
Small fraction of cells fire at any time
– Low *coding level*

$$\phi_i(\mathbf{x}) = h(\mathbf{w}_i^\intercal \mathbf{x} - \theta)$$

Barak et al., 2013; Babadi & Sompolinsky, 2014

Cerebellum architecture



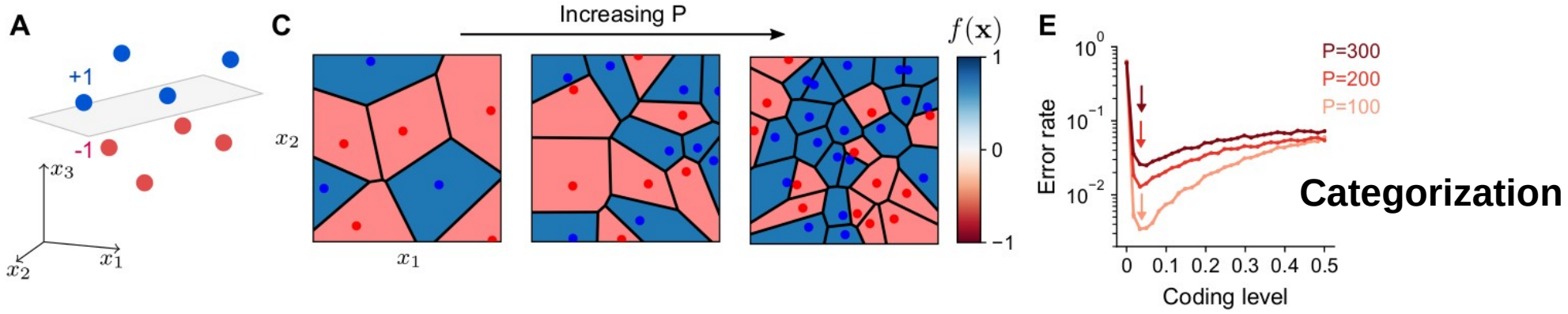


Question:
What role does coding level have on *learning ability of network*?

$$\phi_i(\mathbf{x}) = h(\mathbf{w}_i^\intercal \mathbf{x} - \theta)$$

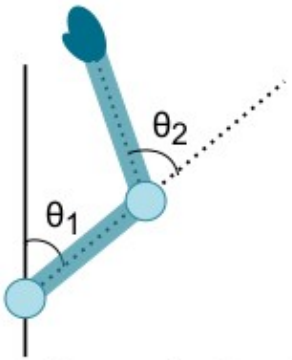
vary threshold

Does optimal coding level depend on task?

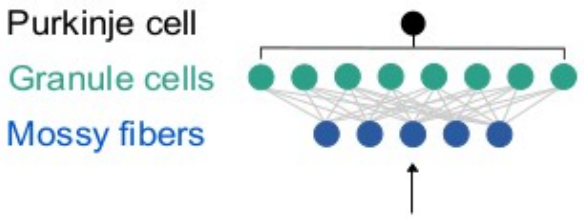


Predicts higher coding level for movement tasks

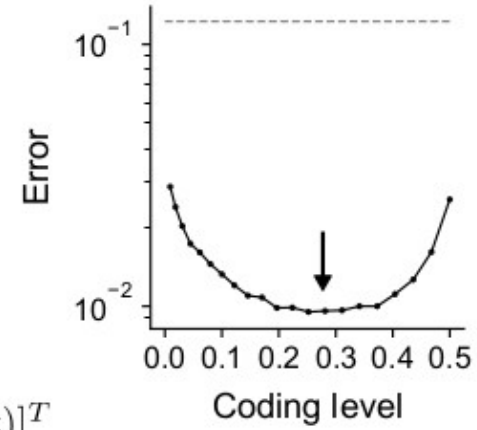
Two-joint arm state prediction



$$f(\mathbf{x}) = \begin{bmatrix} \theta_1(t+\Delta) \\ \theta_2(t+\Delta) \end{bmatrix}$$

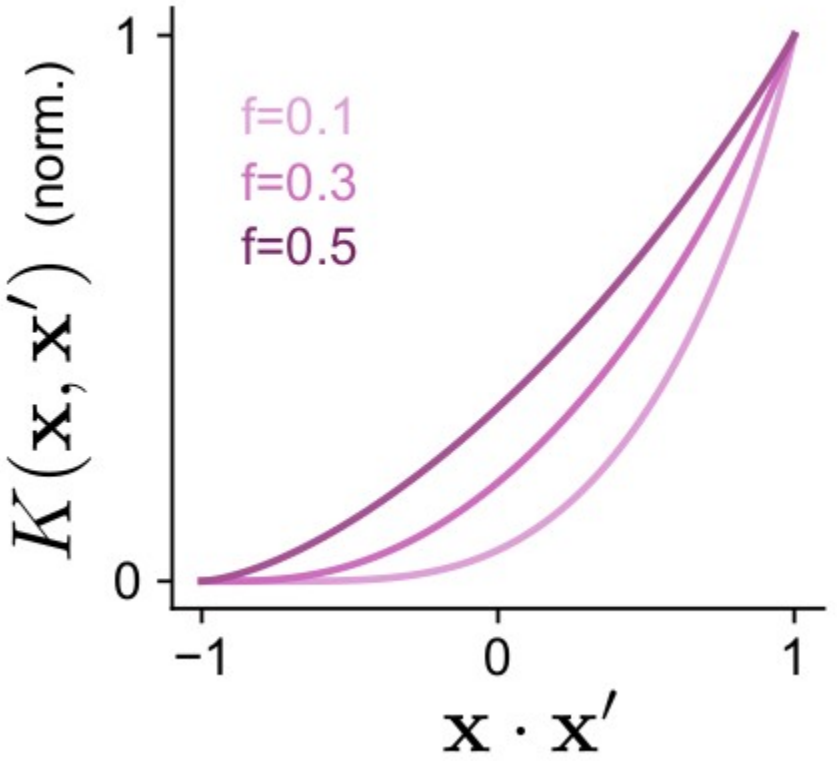


Sensorimotor state $\mathbf{x} = [\theta_1(t), \theta_2(t), \dot{\theta}_1(t), \dot{\theta}_2(t), u_1(t), u_2(t)]^T$



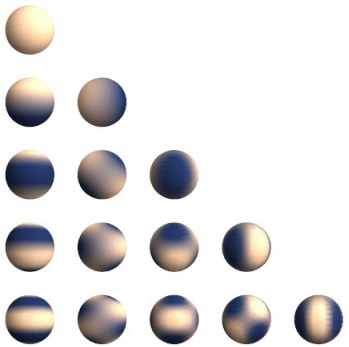
Trajectory prediction

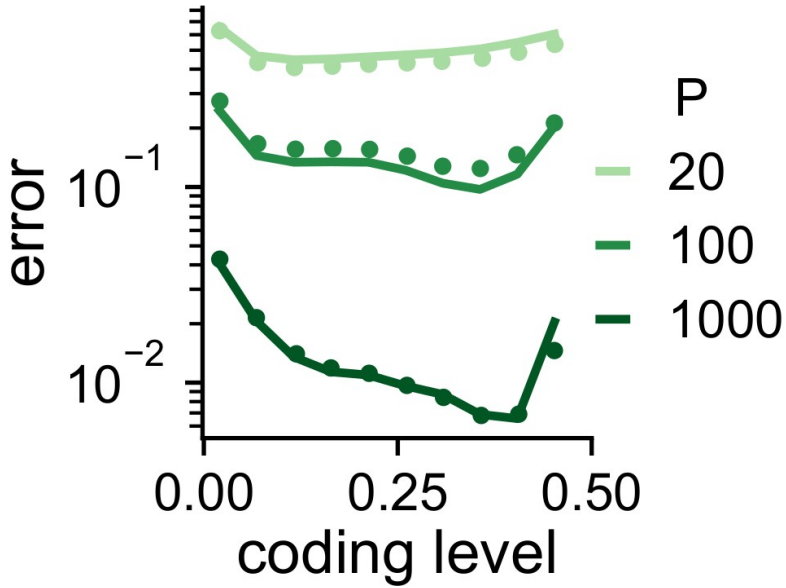
Different threshold, different kernel



Theory of generalization error

$$\text{Error} = C_1 \sum_{\alpha} \left(\frac{c_\alpha}{C_2 + \lambda_\alpha} \right)^2$$





- Kernel regression theory mostly complete*, see work by
  - Pehlevan, Bordelon, Canatar
  - Zdeborová, Mézard, Krzakala, …
  - Montanari, …

* Less so for finite # random features, and some “rigorizing” left to do

* Also exhibits double descent

Overview

- **Kernel theory of networks: applications**
  - Thresholds → filtering in function space
  - **Random receptive fields → basis change & filtering input space**

“Structured random receptive fields
enable informative sensory encodings”
https://www.biorxiv.org/content/10.1101/2021.09.09.459651
To appear, PLoS Comp Bio

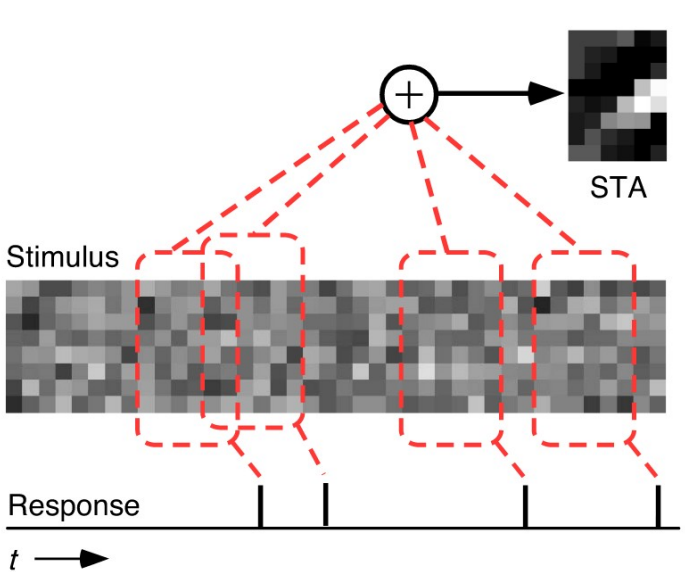
Biraj Pandey
Univ. Washington
Applied Math
NSF Grad Fellow

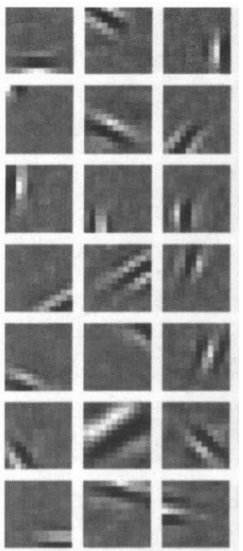
Marius Pachitariu
Janelia

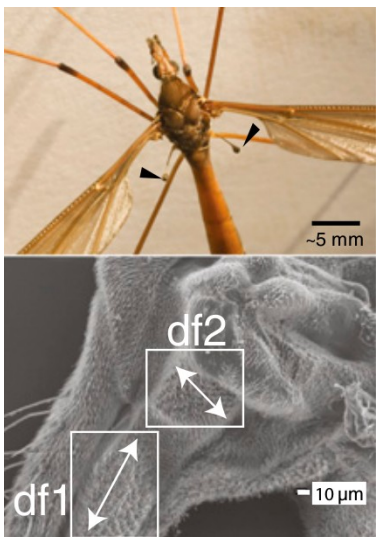
Bing Brunton
Univ. Washington
Biology

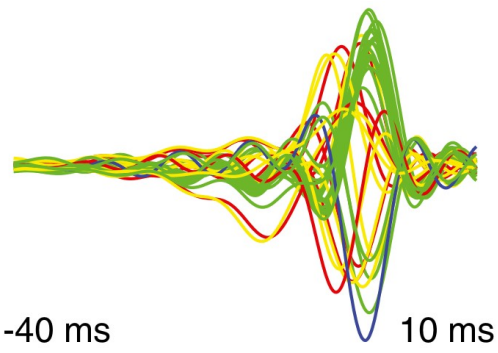
Receptive fields occuring in nature

Average stimulus that causes a neuron to spike, i.e. its **receptive field**



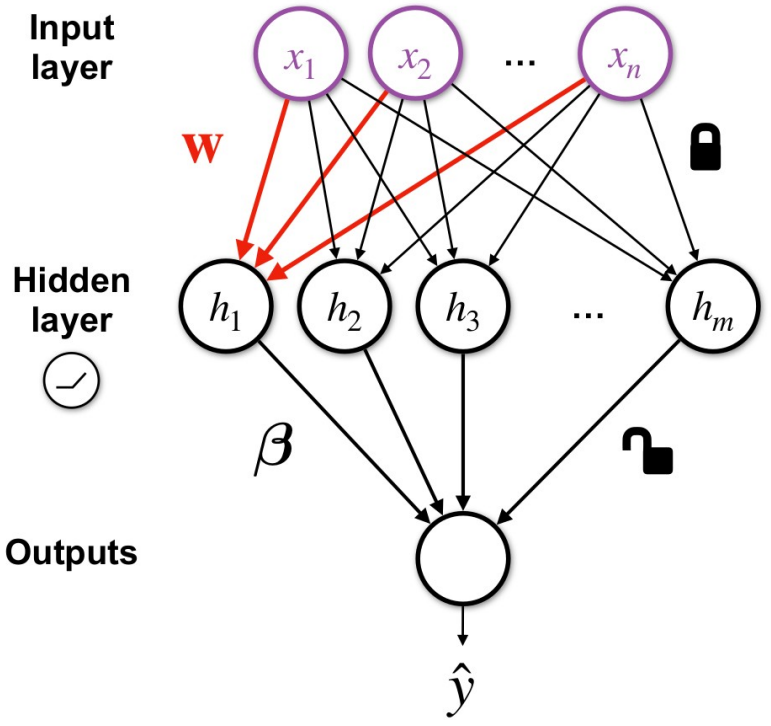


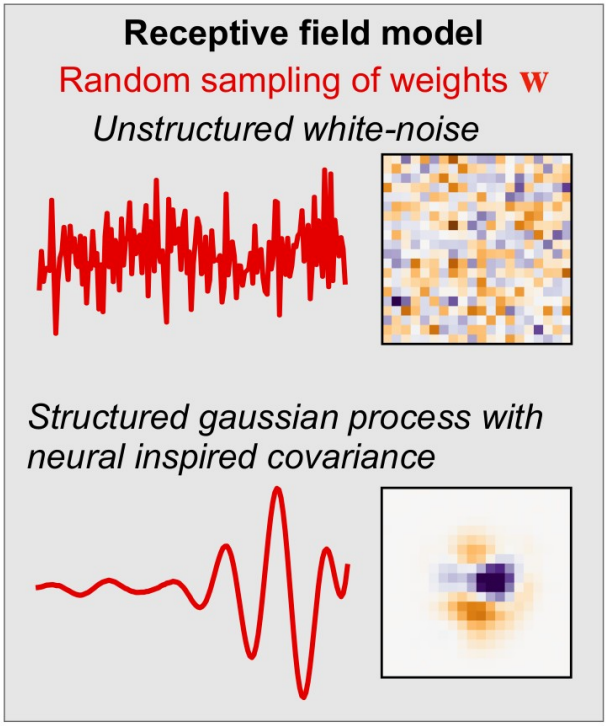




Schwartz et al; Olshausen & Field; Fox, Fairhall, Daniel

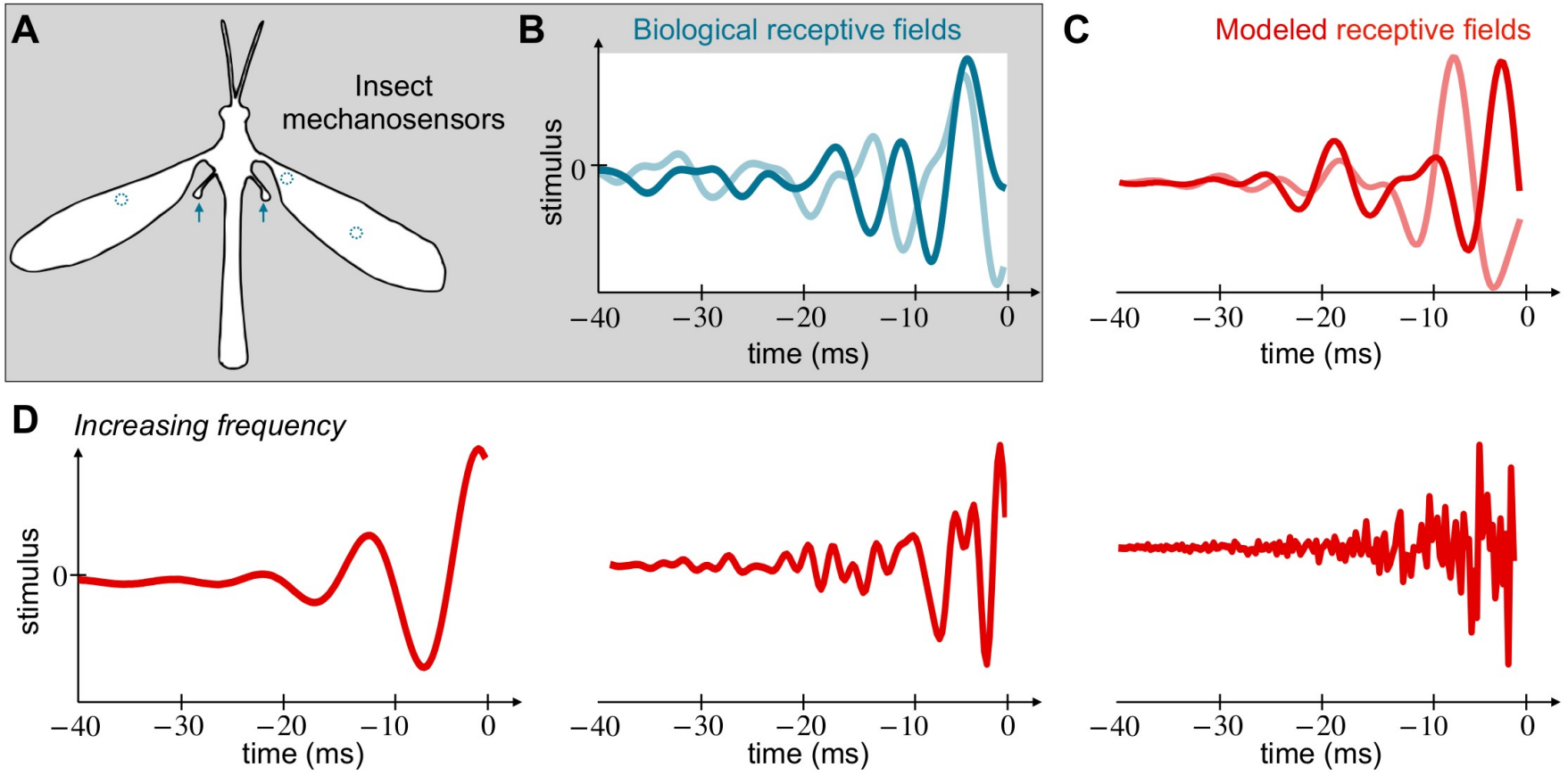
Receptive fields as weights in random network



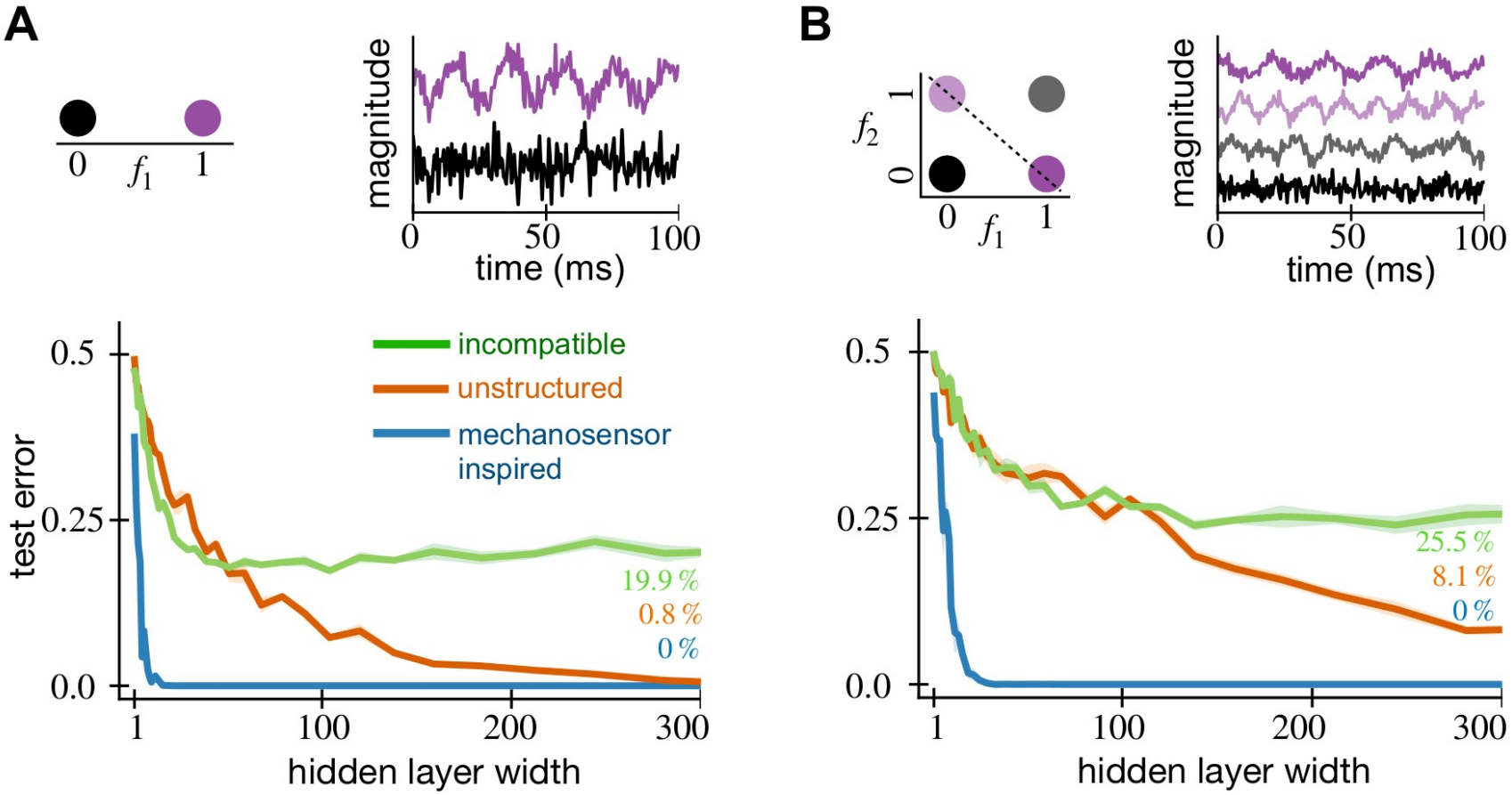


$$\mathbf{w} \sim \mathcal{N}(0, C)$$ ← Parametric function

Haltere-inspired random weights

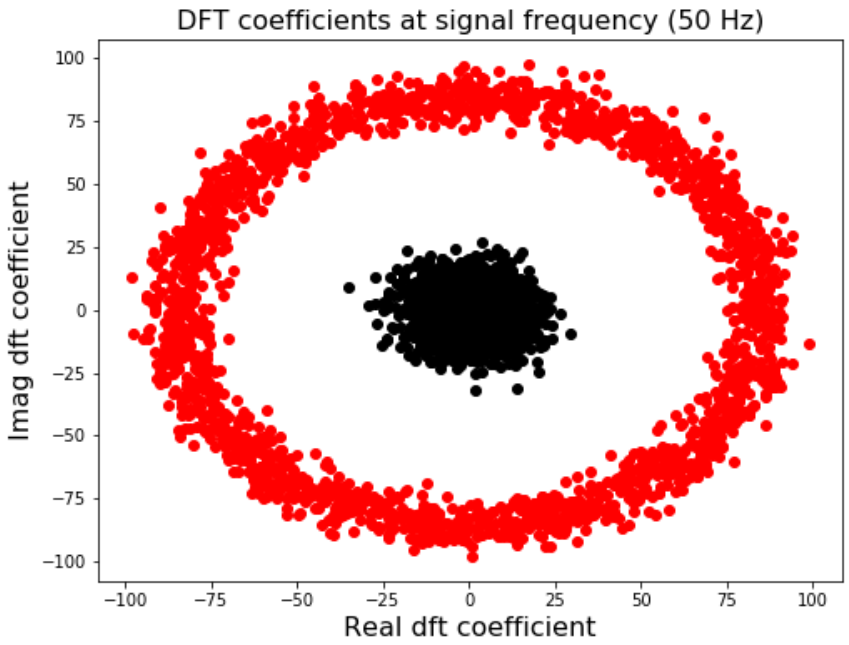


“Haltere” weights do well for timeseries tasks



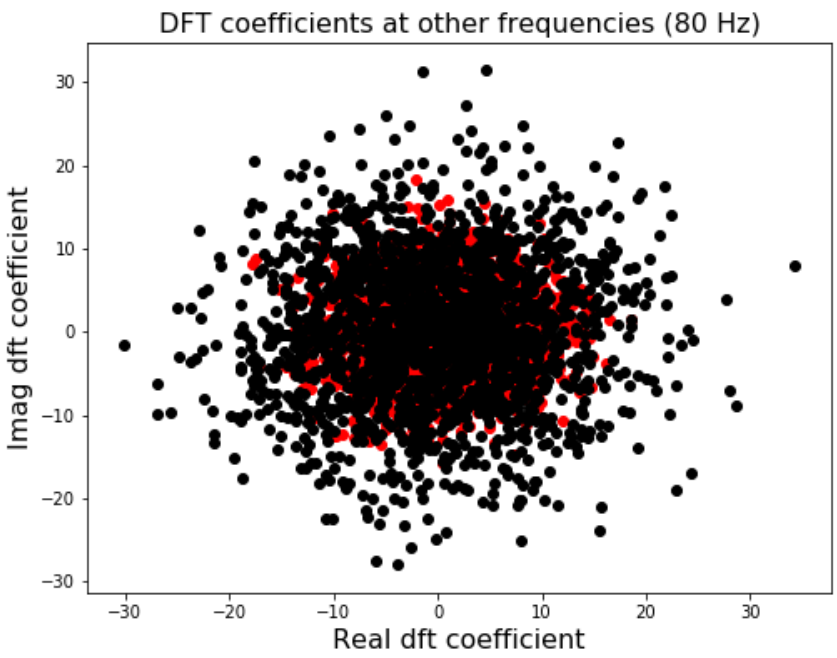
Filters irrelevant components of stimulus

1 component



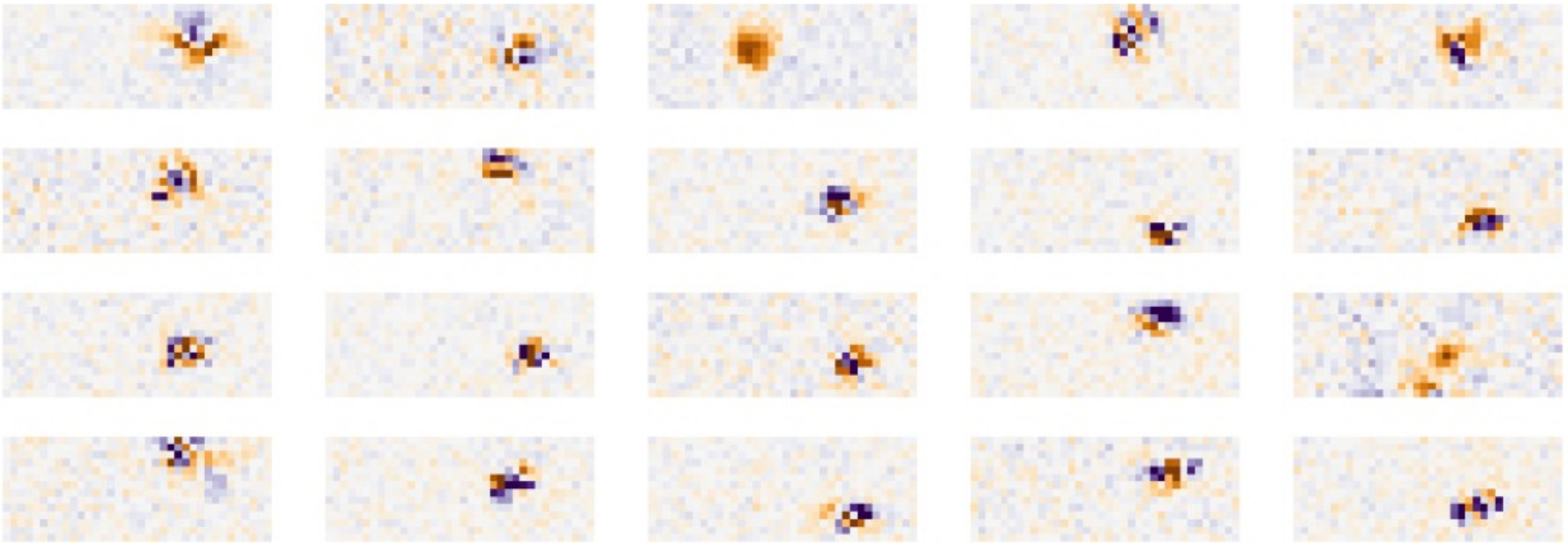
Relevant

T-1 components



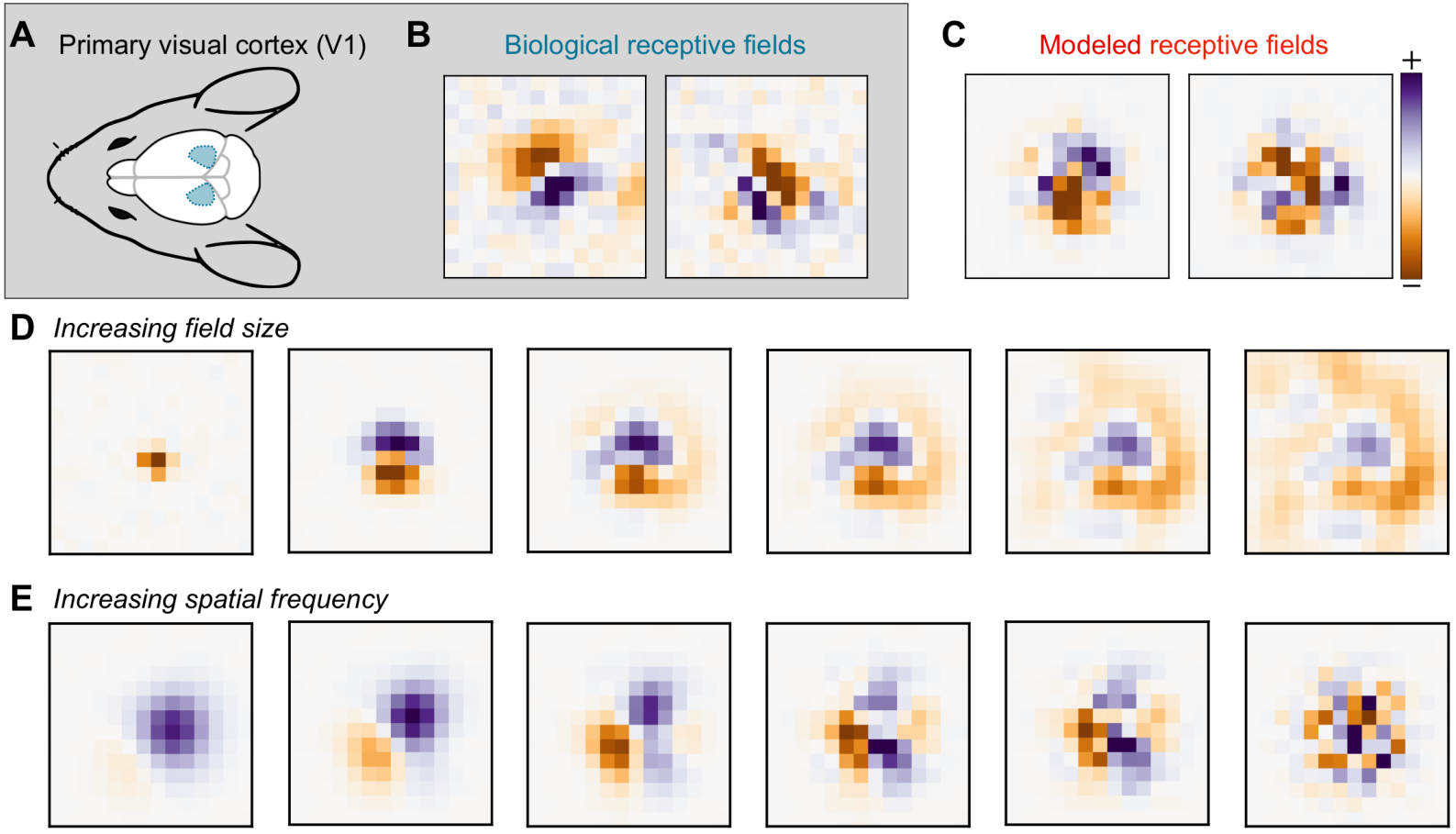
Irrelevant

Mouse V1 receptive fields

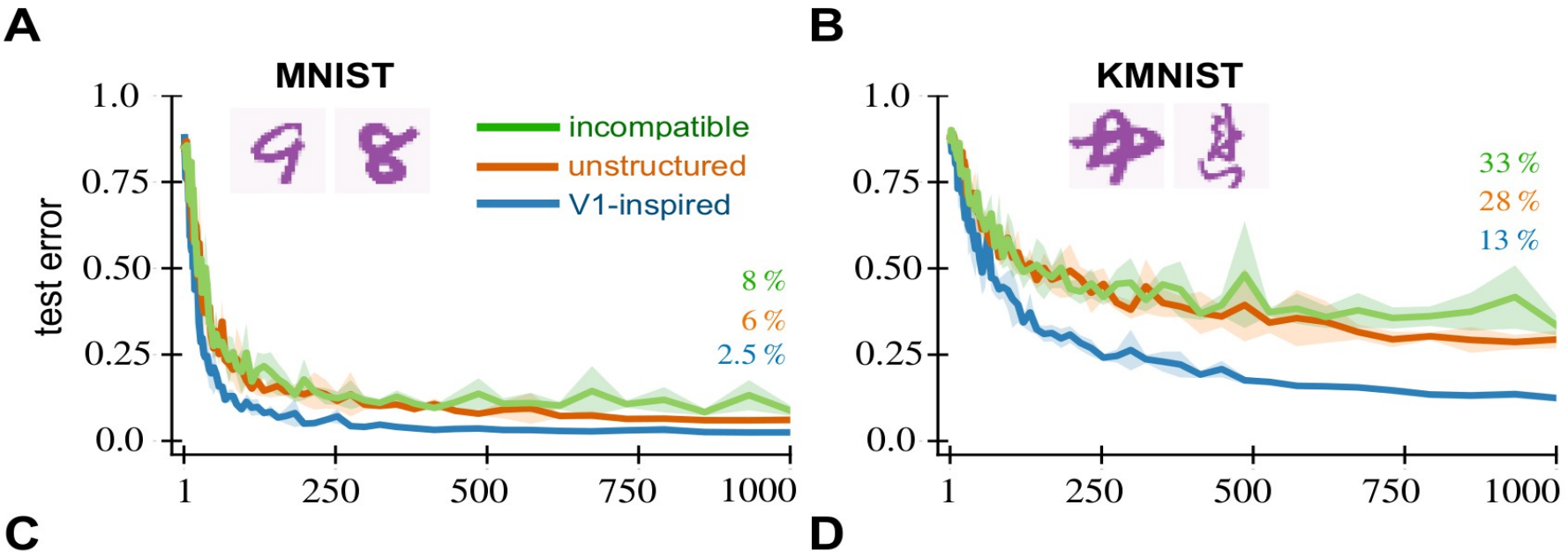


24k white noise stimuli, ~8.3k good neurons
(in paper: data for natural image and other stimuli)

V1-inspired random weights

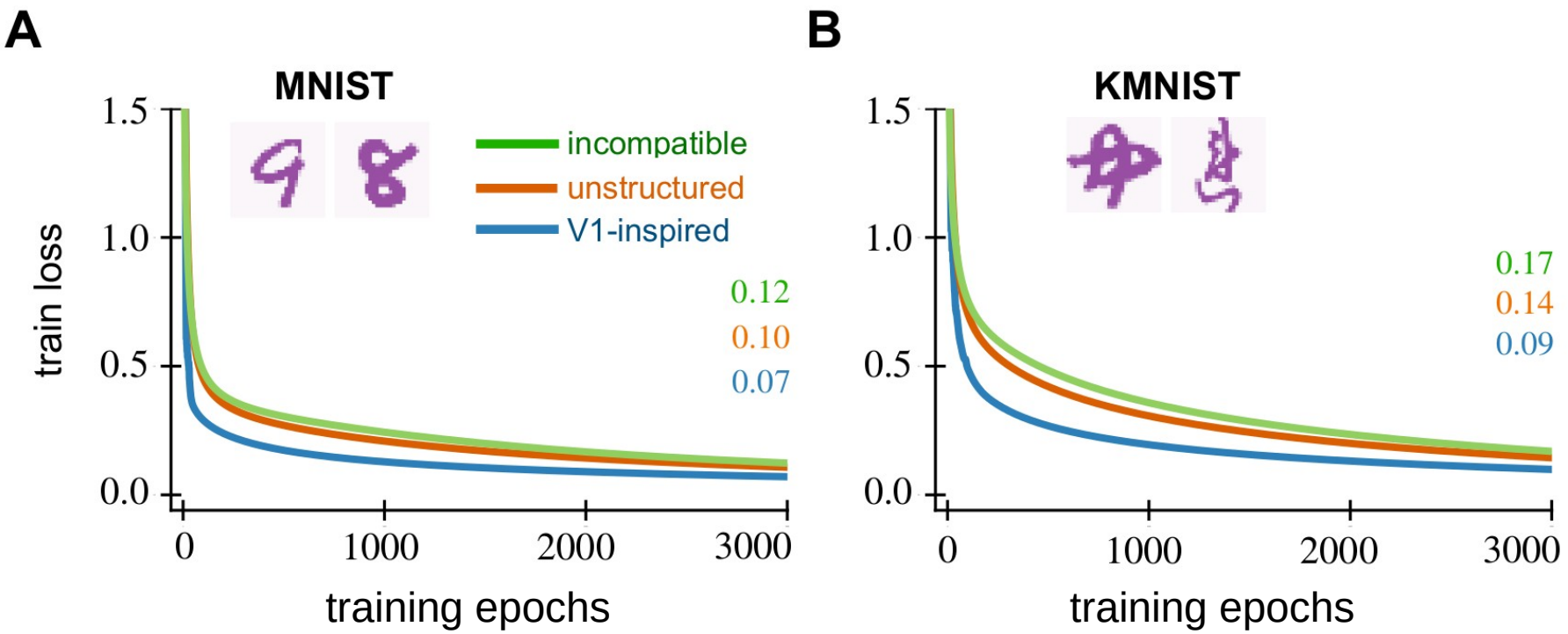


Fewer “V1” features required to learn



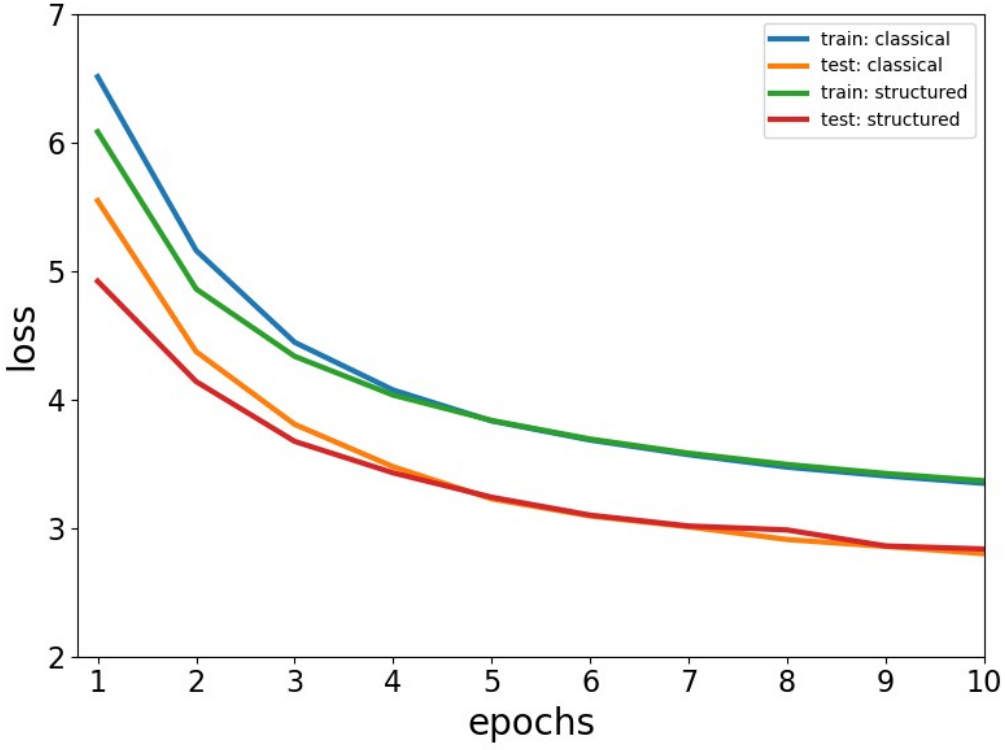
In paper: lower error in small data regime

Structure is a better starting point for training



But structure doesn't help *fully trained* deep CNNs

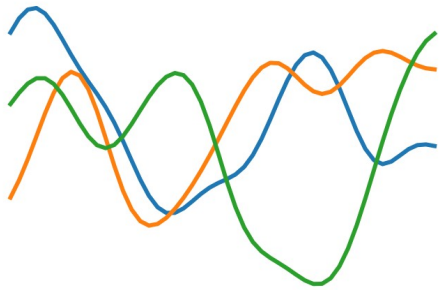
AlexNet: ImageNet classification



Reasons:

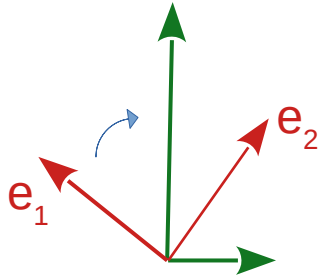
- **Fewer features** (convolution)
- Many layers
- Lots of data

Theory of random receptive fields



1) Receptive fields are random vectors ~ Gaussian process (GP)
2) $\mathbf{w} = UD\mathbf{g} \quad \text{where} \quad \mathbf{g} \sim \mathcal{N}(0, I)$
3) Structured randomness equivalent to a basis change on input
4) Kernel isn't different than unstructured case
5) *Eigenfunctions* of kernel change because of new data distribution

Conclusions



- Kernel representation of random networks
  - **Which functions are easy to learn?**
- Structured randomness = informative random projection
  - Threshold nonlinearity: filter basis *functions*
  - Tuning curves: filtering of *input* signal
- Many future directions
  - Feedback, temporal dynamics, unsupervised settings

Thank you!

To the organizers of CNEURO, Tsinghua, IOB and all the great students

- Funding:
- Collaborators:
  - **Biraj Pandey**, Marius Pachitariu, Bing Brunton (UW)
  - **Marjorie Xie**, Ashok Litwin-Kumar, Samuel Muscinelli (Columbia)

References: Kernel theory of networks

Way of scaling kernel methods for big datasets

- Rahimi & Recht (2008) **random features** … sketching *K*
- older work in Gaussian processes by Neal (1996), Williams (1997)

Exciting work tries to understand success of ANNs trained via gradient descent

- neural tangent kernel (NTK) – Jacot, Gabriel, Hongler, 2018; Arora et al, 2019
- convolutional kernel networks (CKN) – Mallat; Bruna; Harchaoui; Chizat et al
- interpolation & multiple descent – Belkin et al; Mei & Montanari; Pehlevan

Related review: “Randomness in neural networks” by Scardapane & Wang (2017)