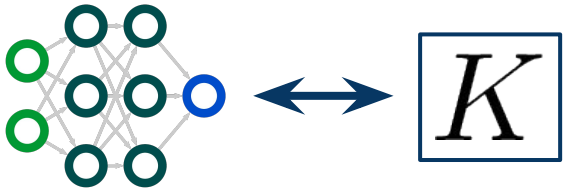
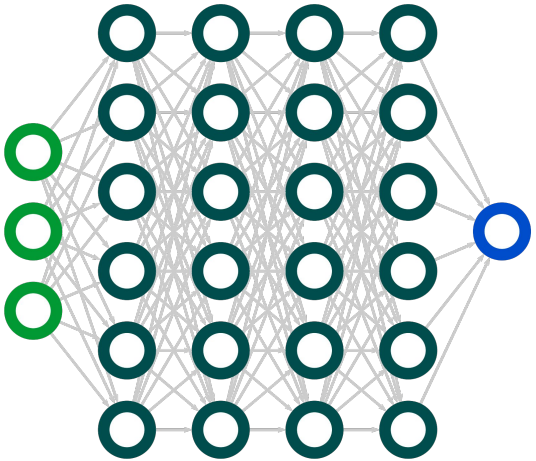
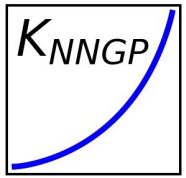
Reverse Engineering the Neural Tangent Kernel



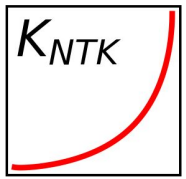
Jamie Simon Sajant Anand Mike DeWeese

Wide neural networks are described by **two kernels**



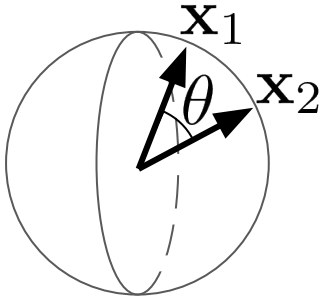


“initialization kernel”
Lee et al. (2018)



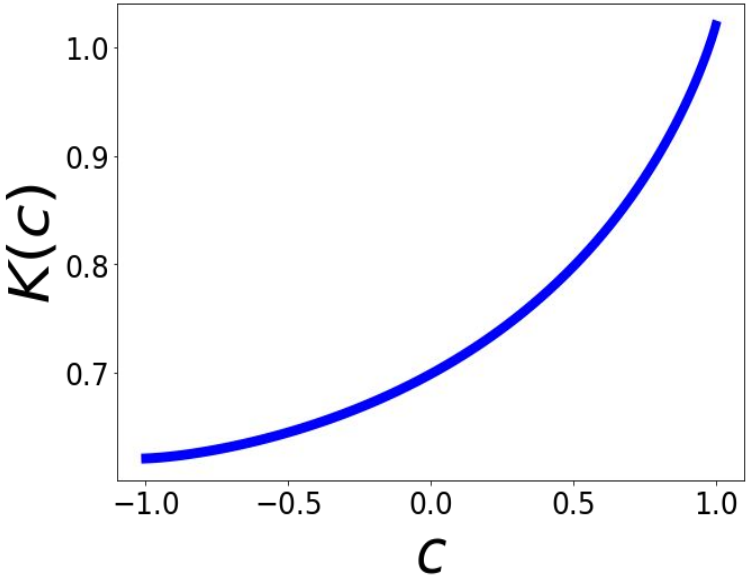
“training kernel”
Jacot et al. (2018)

Example FCN kernels

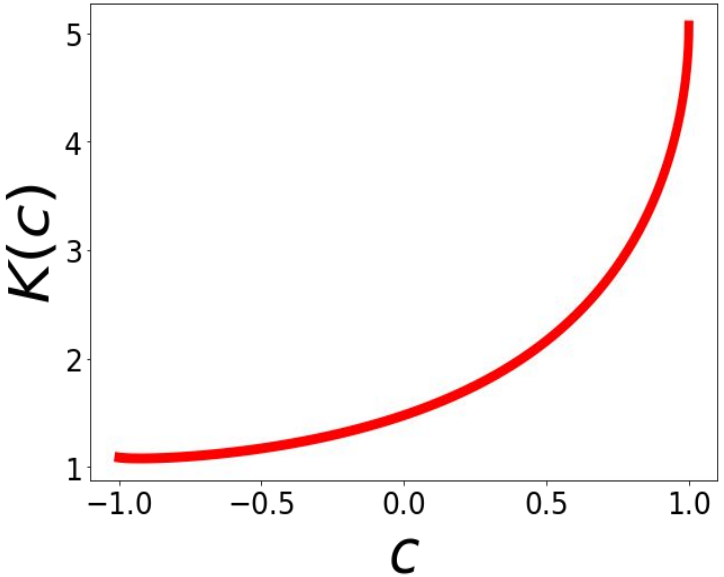


$$c \equiv \cos(\theta)$$

4HL ReLU net - NNGP



4HL ReLU net - NTK

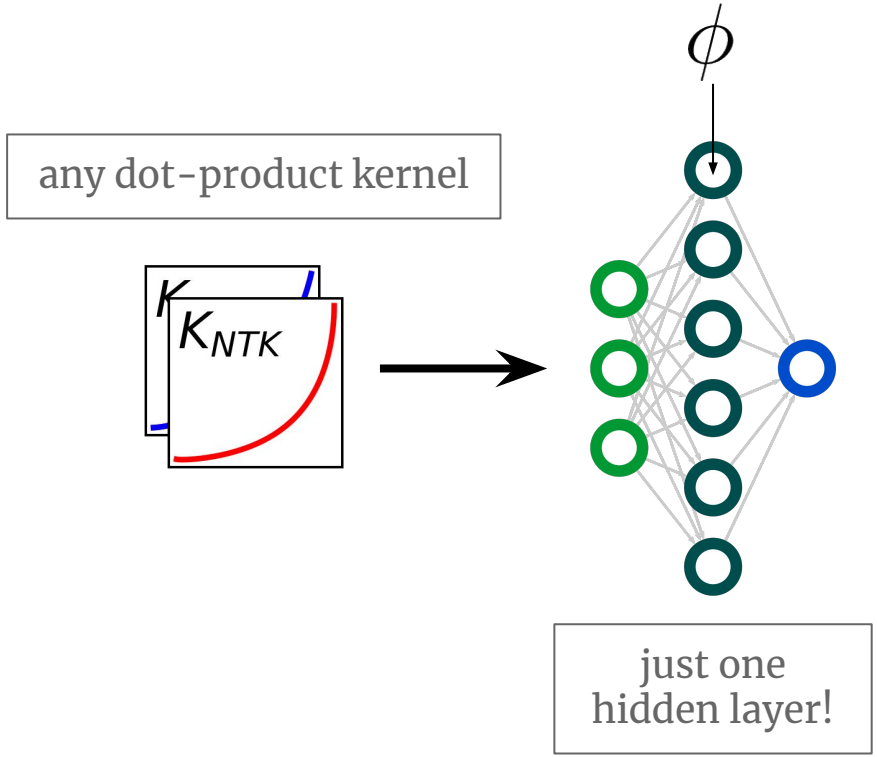


Main result: kernels -> networks

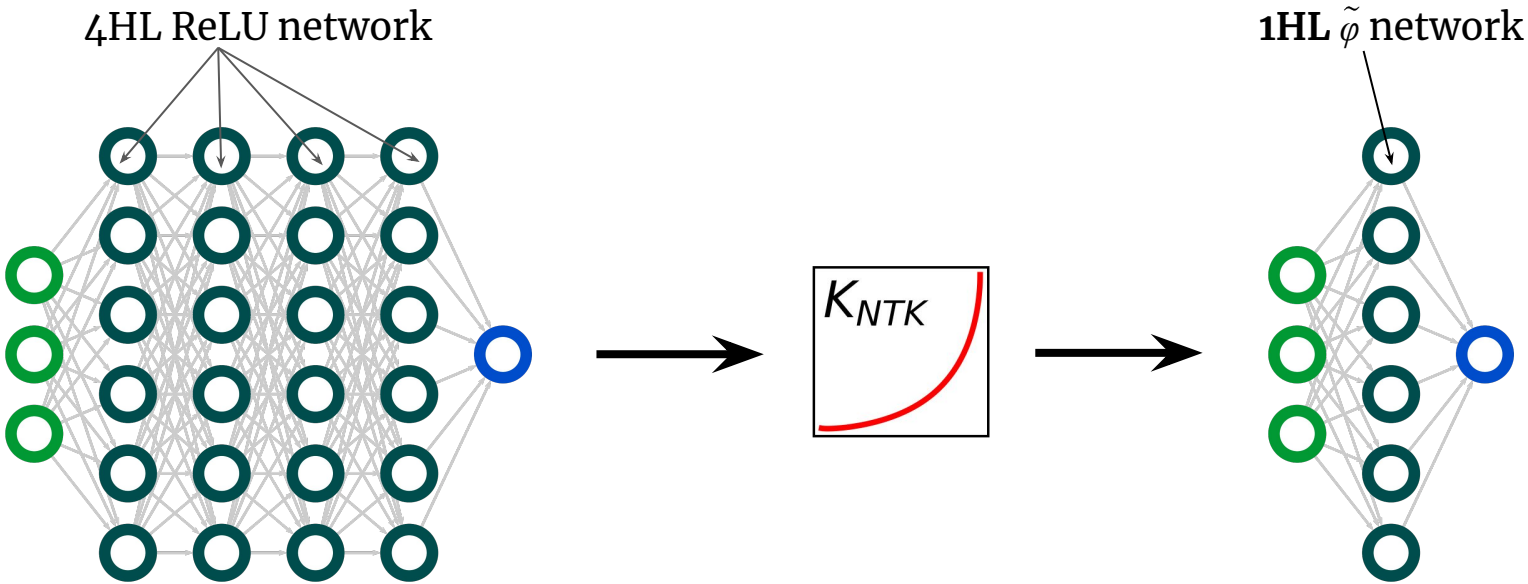
Theorem 3.1 (Kernel reverse engineering). *Any desired dot product kernel* $K(c) = \sum_{k=0}^{\infty} a_k c^k$ *can be achieved as*

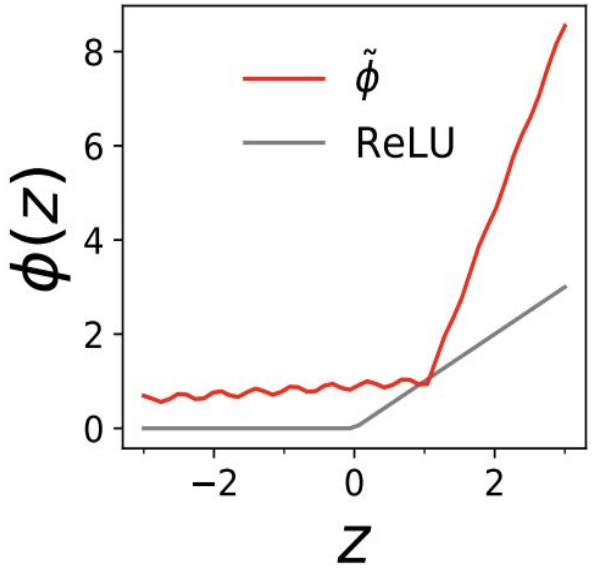
- *the NNGP kernel of a single-hidden-layer FCN with* $\sigma_w = 1, \sigma_b = 0$ *and* $\phi(z) = \sum_{k=0}^{\infty} \pm a_k^{1/2} h_k(z)$,
- *the NTK of a single-hidden-layer FCN with* $\sigma_w = 1, \sigma_b = 0$ *and* $\phi(z) = \sum_{k=0}^{\infty} \pm \left(\frac{a_k}{1+k}\right)^{1/2} h_k(z)$,

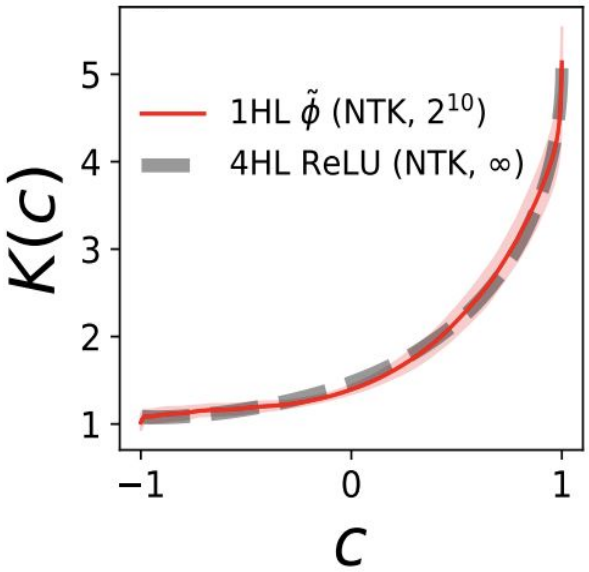
where we use $\pm$ *to indicate that the sign of each term is arbitrary. These are the complete set of activation functions that give this kernel in a single-hidden-layer FCN with* $\sigma_w = 1, \sigma_b = 0$.

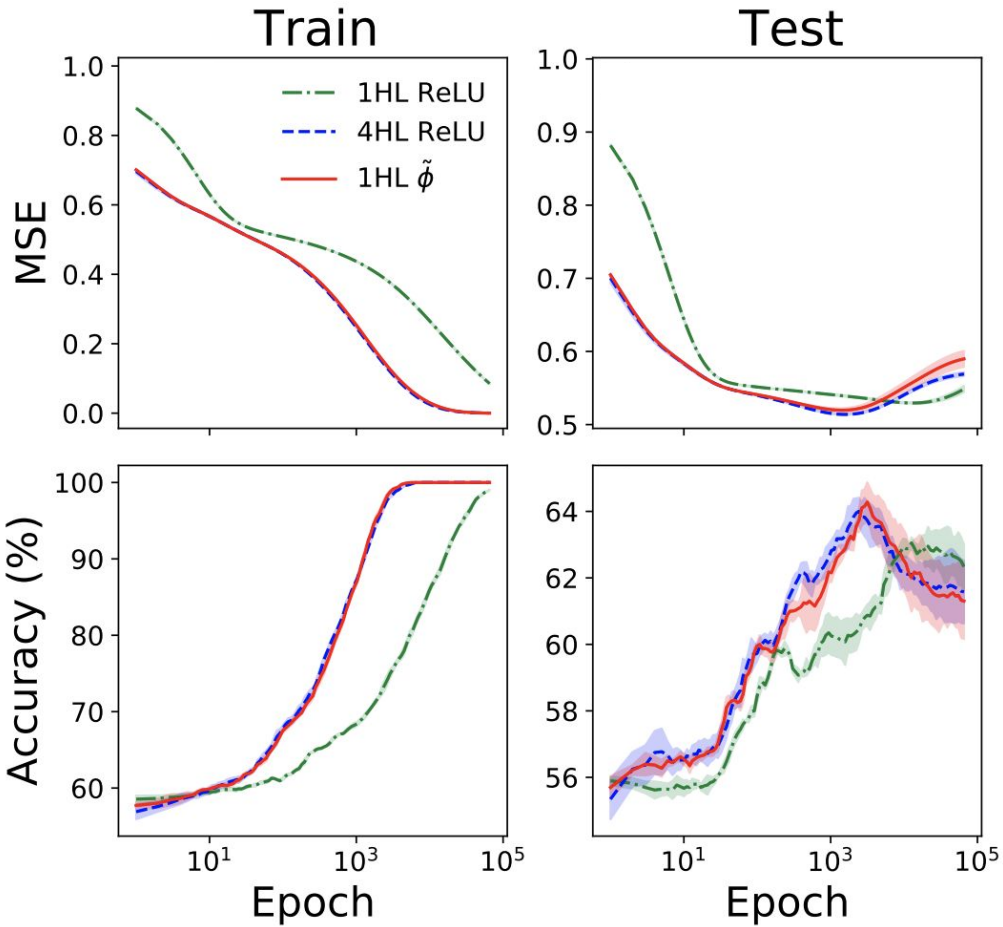


Shallowification of a deep FCN

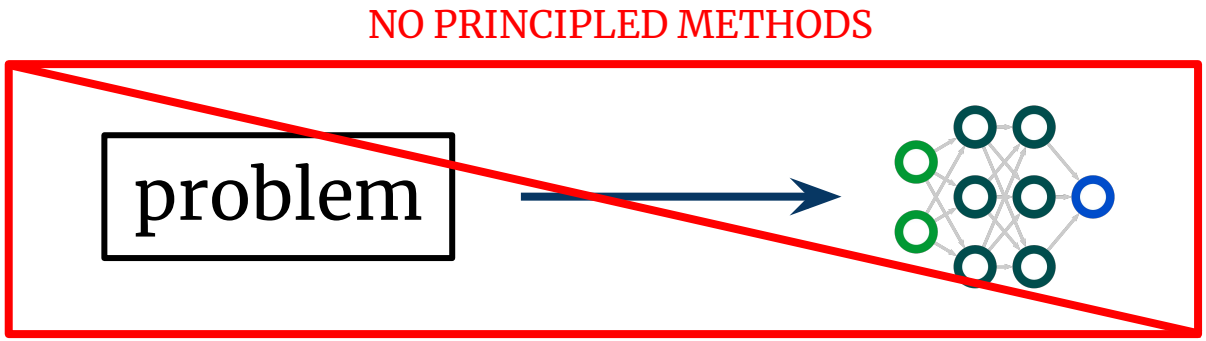



$\tilde{\phi}$
ReLU
$\phi(z)$
z
8
6
4
2
0
−2
0
2


1HL $\tilde{\phi}$ (NTK, 2^{10})
4HL ReLU (NTK, ∞)
$K(c)$
c
5
4
3
2
1
−1
0
1


Train
Test
MSE
1HL ReLU
4HL ReLU
1HL $\tilde{\phi}$
1.0
0.8
0.6
0.4
0.2
0.0
0.9
0.7
0.5
Accuracy (%)
100
90
80
70
60
64
62
58
56
10^1
10^3
10^5
Epoch

Reverse-engineering for neural architecture design



Summary: **any dot-product kernel** can be realized as a **shallow FCN.**

Implications:

- Depth may not benefit FCNs (!!)
- Potential for **principled neural architecture design**