Plan for today

1. Fitting DoG to RFs
2. Color RFs findings:

a) Directly looking at color RFs is unreliable: few examples

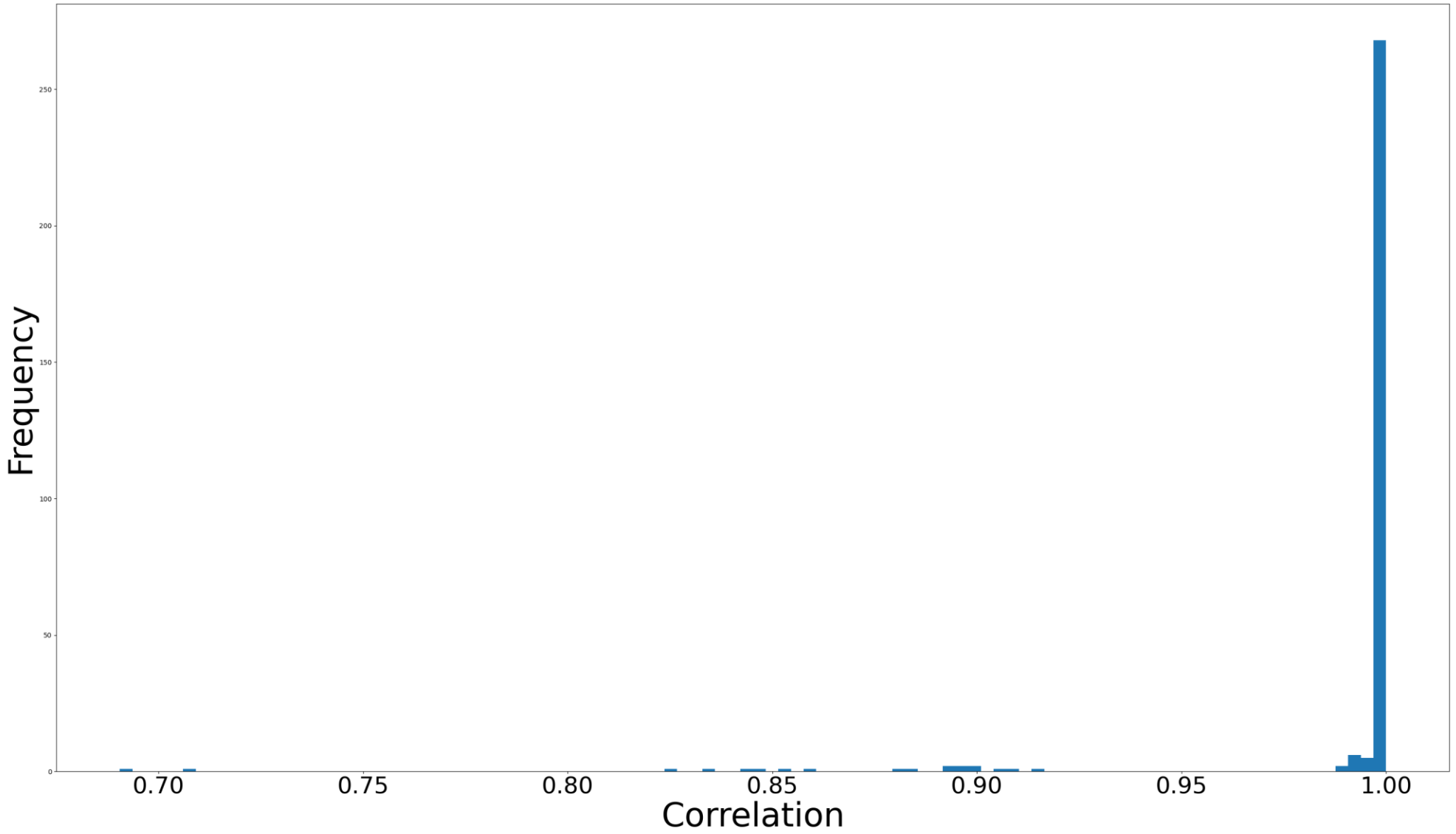
b) Clusters and their mosaics: no –S cluster, but cluster for OFF Parvo + -S inputs instead.

c) S inputs have larger centers, and therefore weaker surrounds

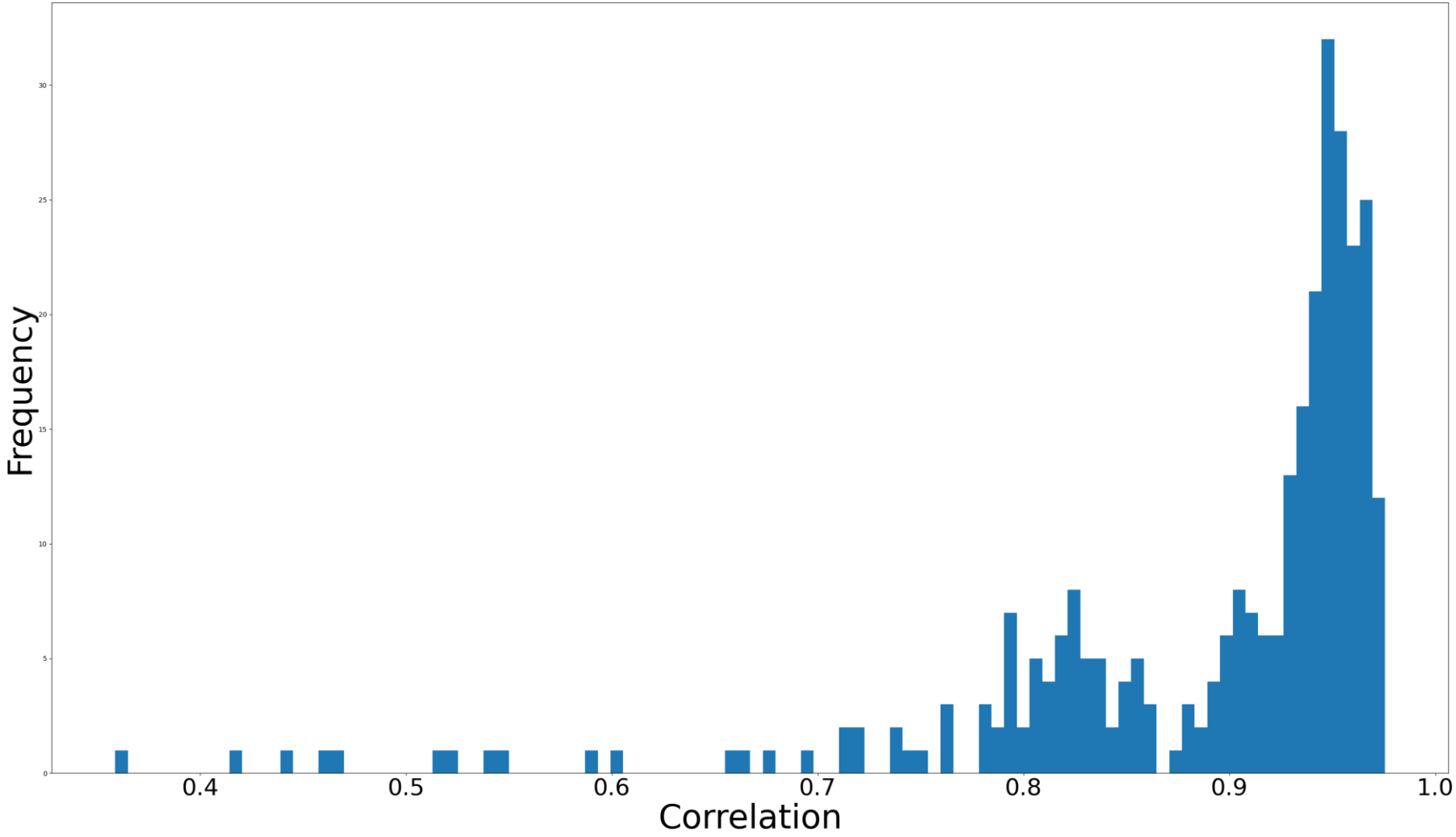
3. Reducing LR at plateau

Fitting DoGs to RFs

DoG fit to parametrized RFs

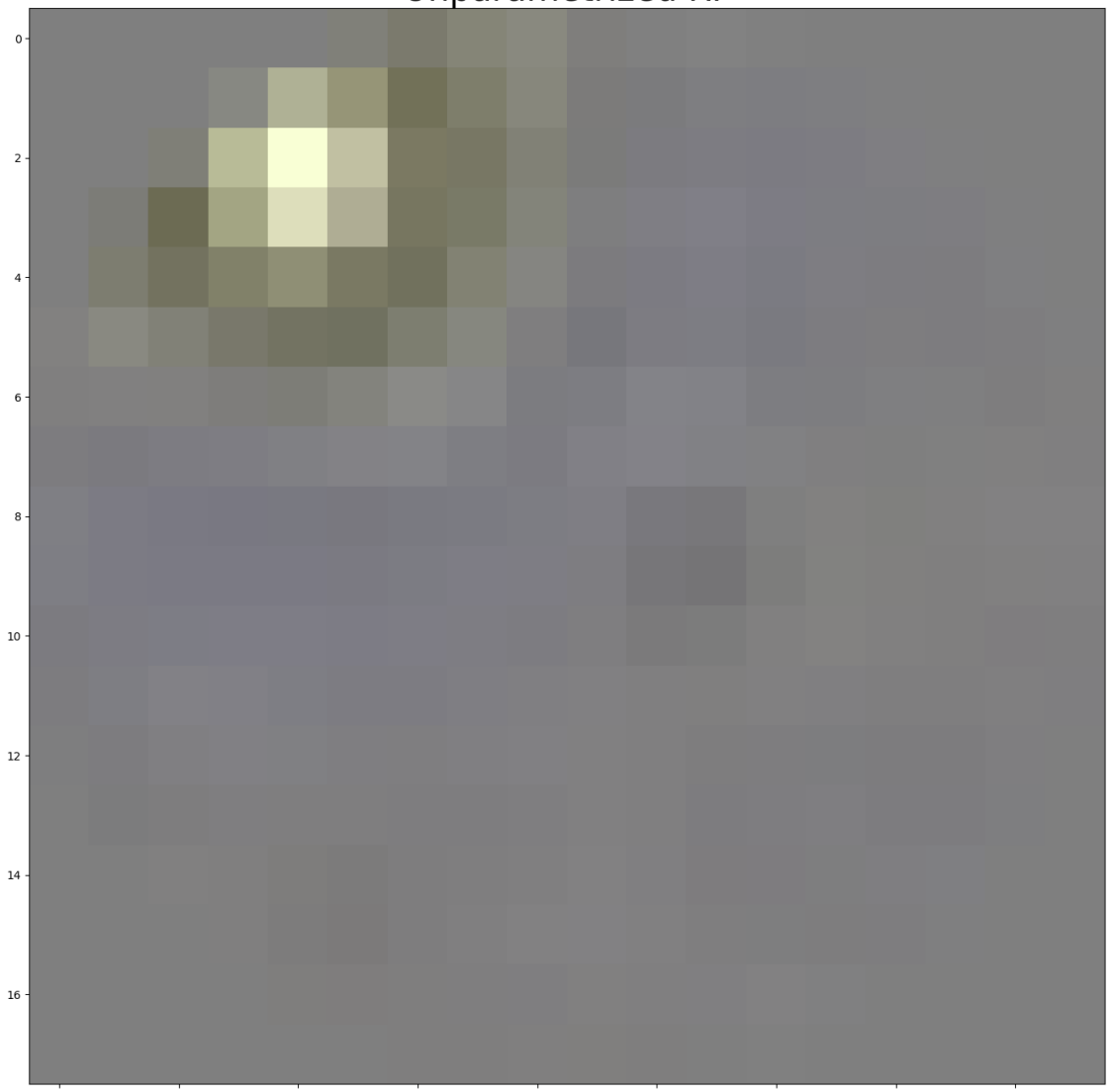


DoG fit to unparametrized RFs

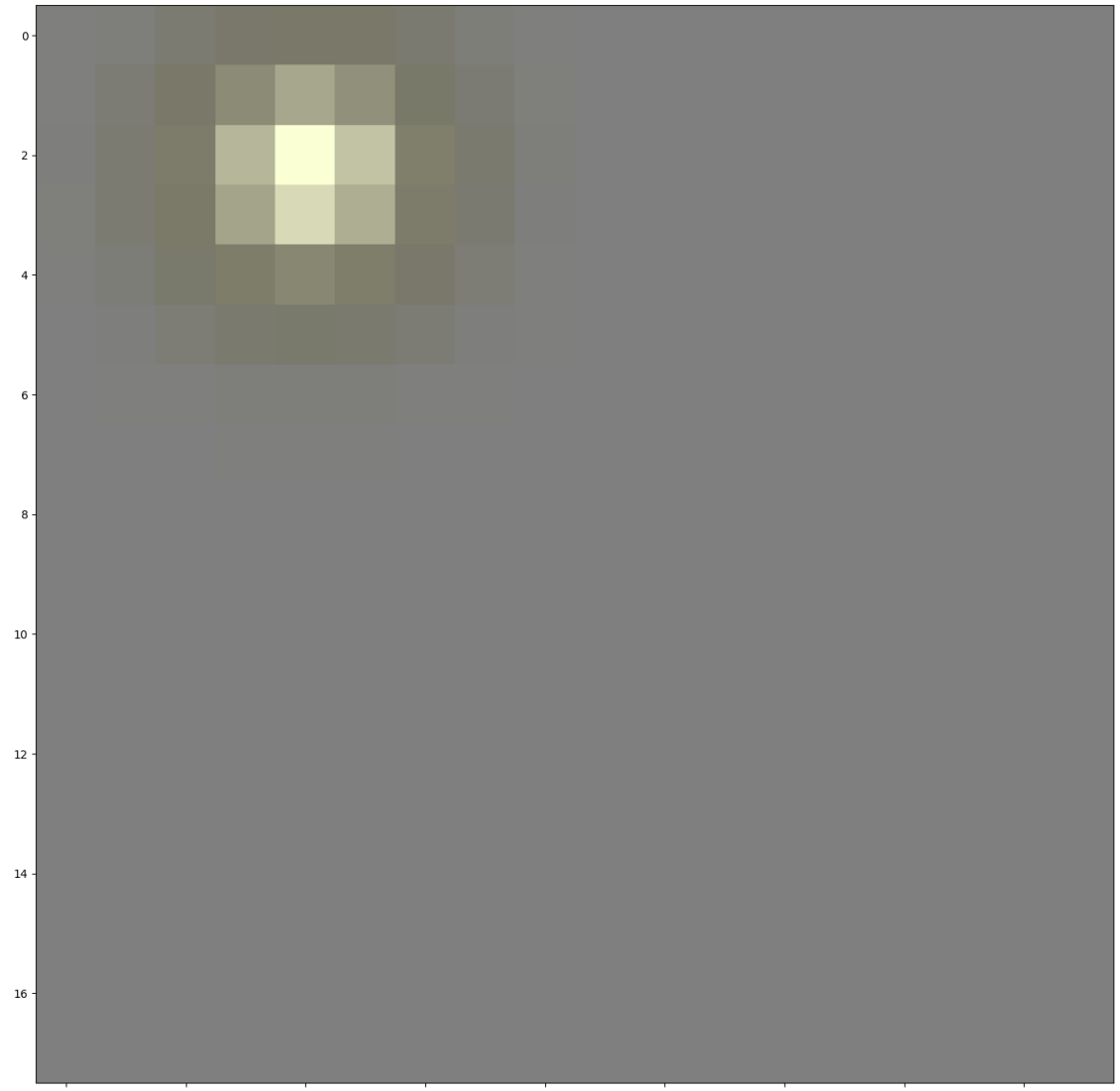


cor = 0.9386, 240129-142655 #0

Unparametrized RF

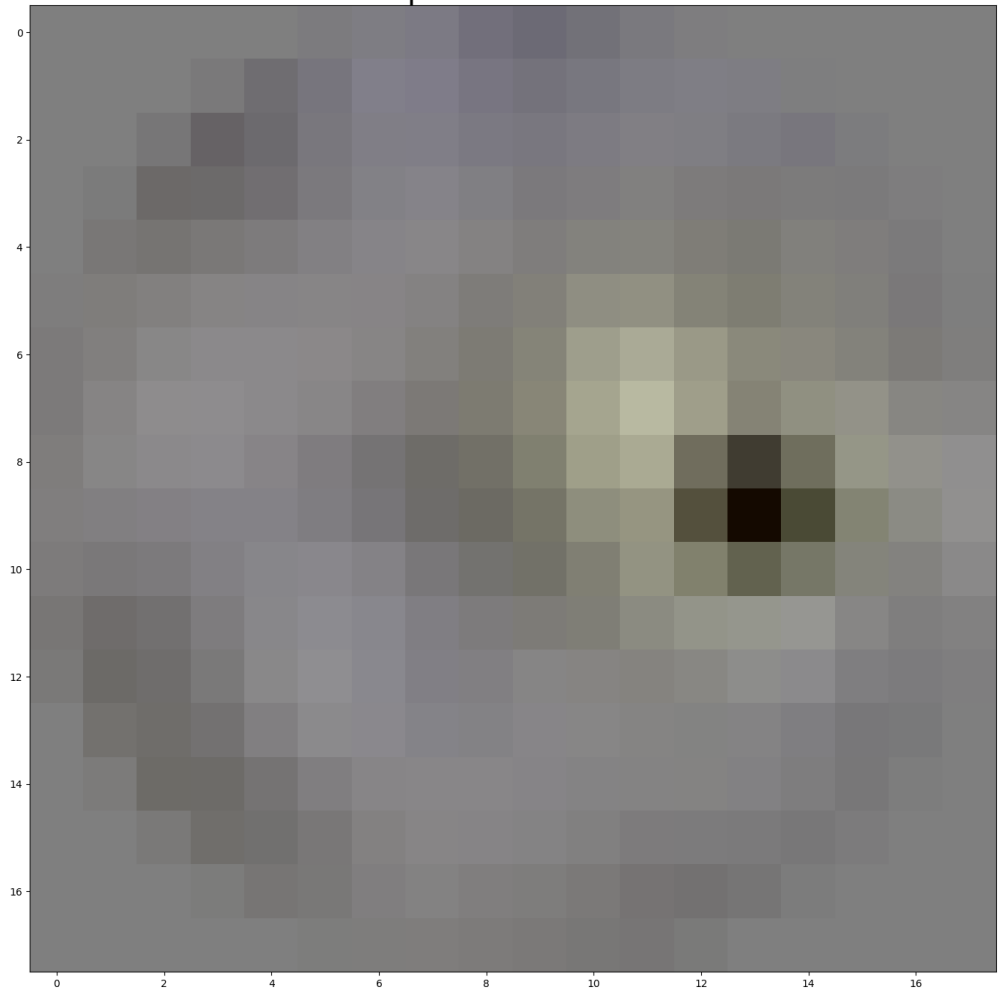


DoG fit

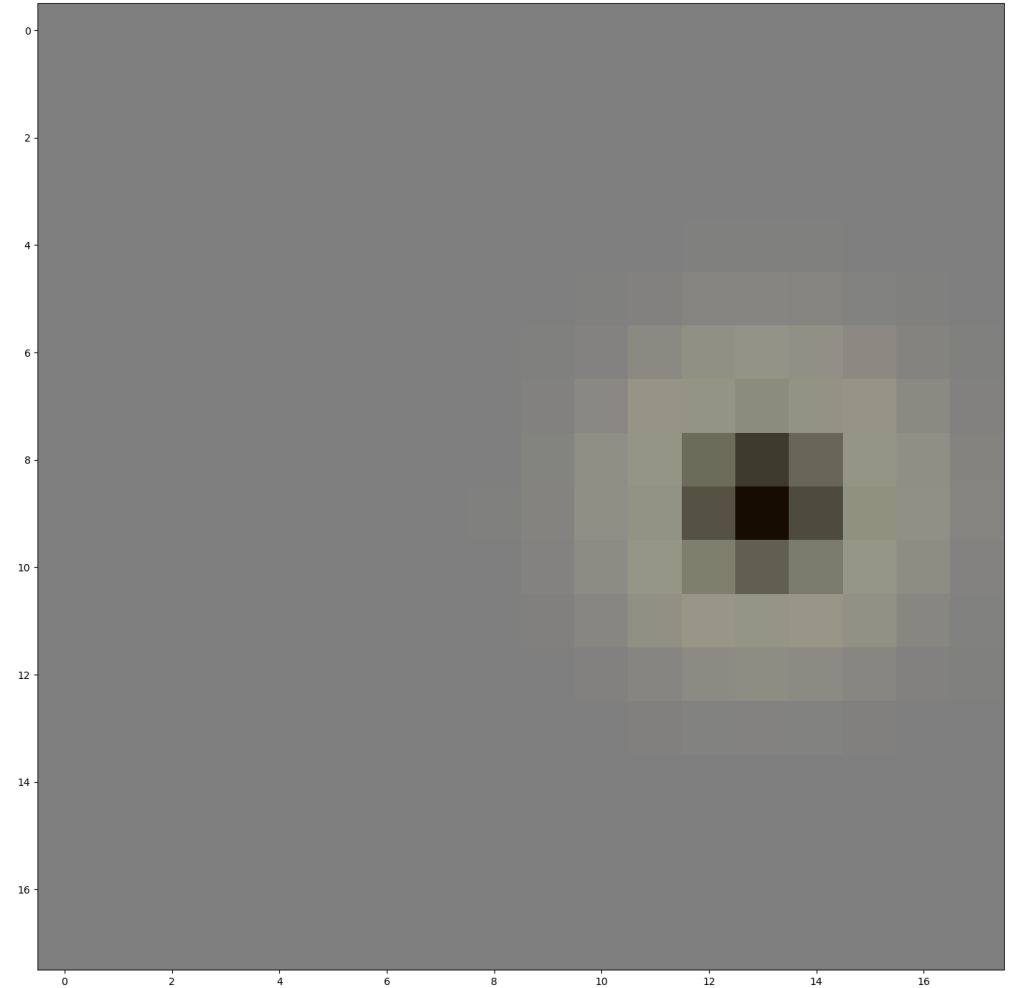


correlation = 0.7841267334808592

Unparametrized RF

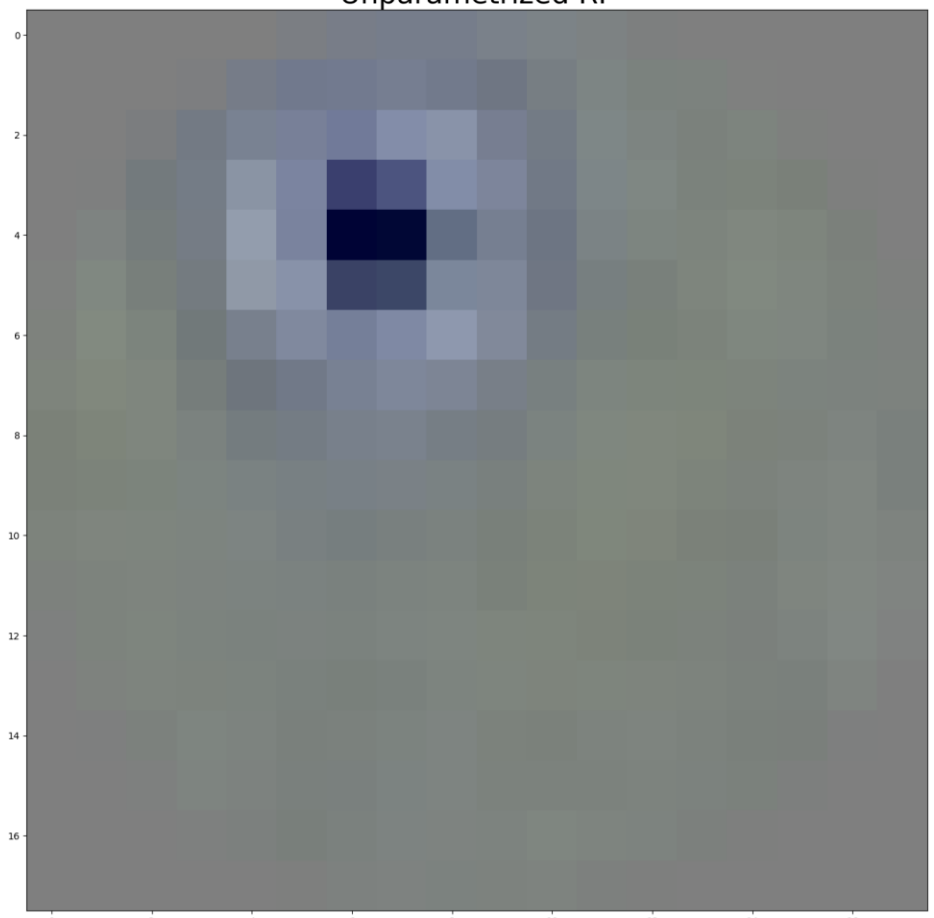


DoG fit

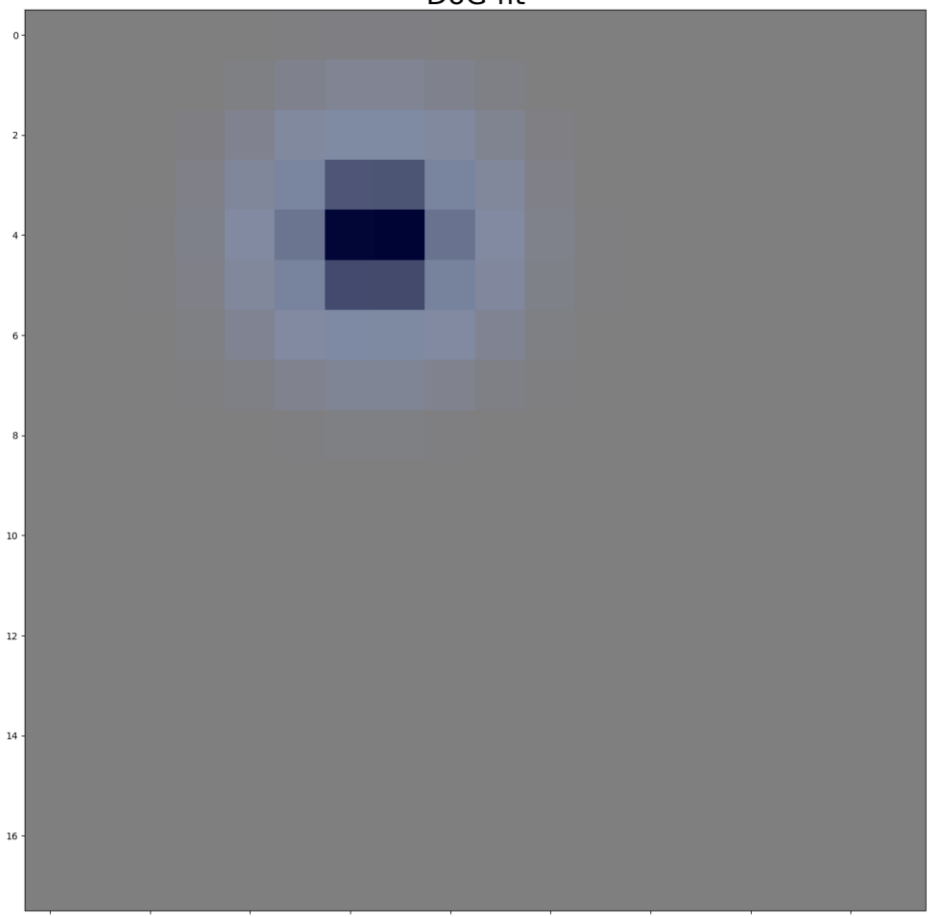


correlation = 0.926745107628051

Unparametrized RF

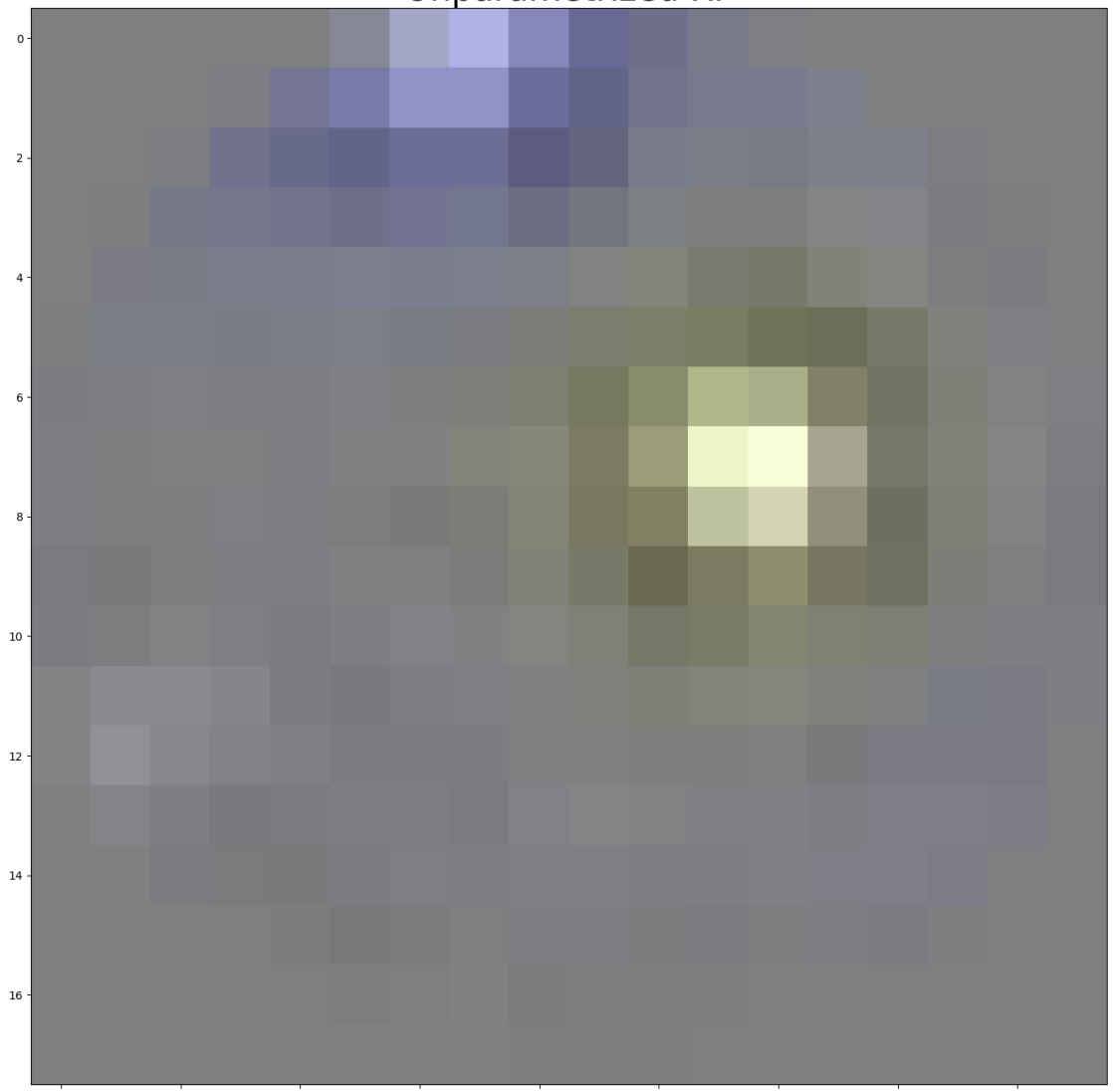


DoG fit

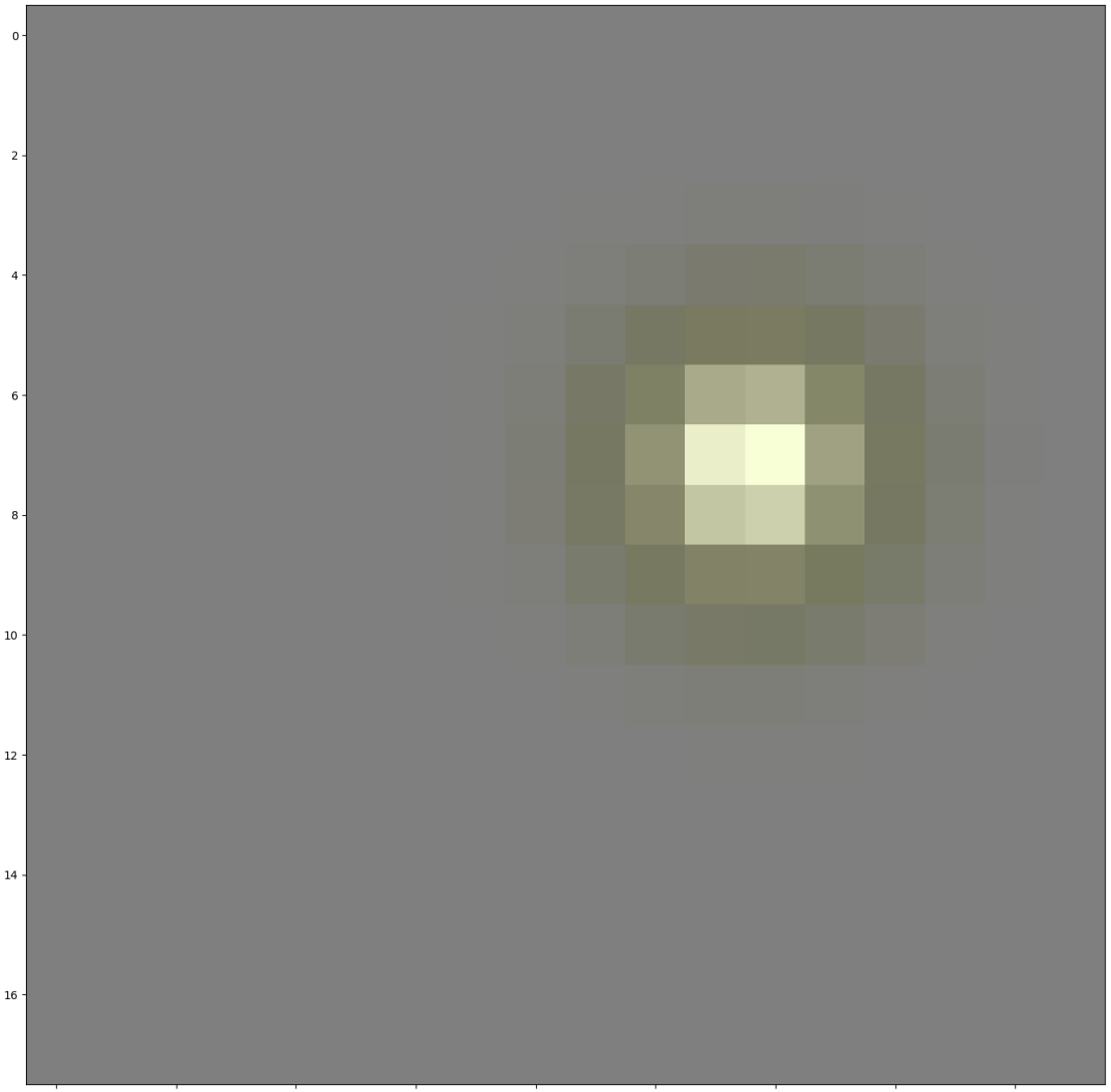


cor = 0.7943, 240129-142655 #34

Unparametrized RF

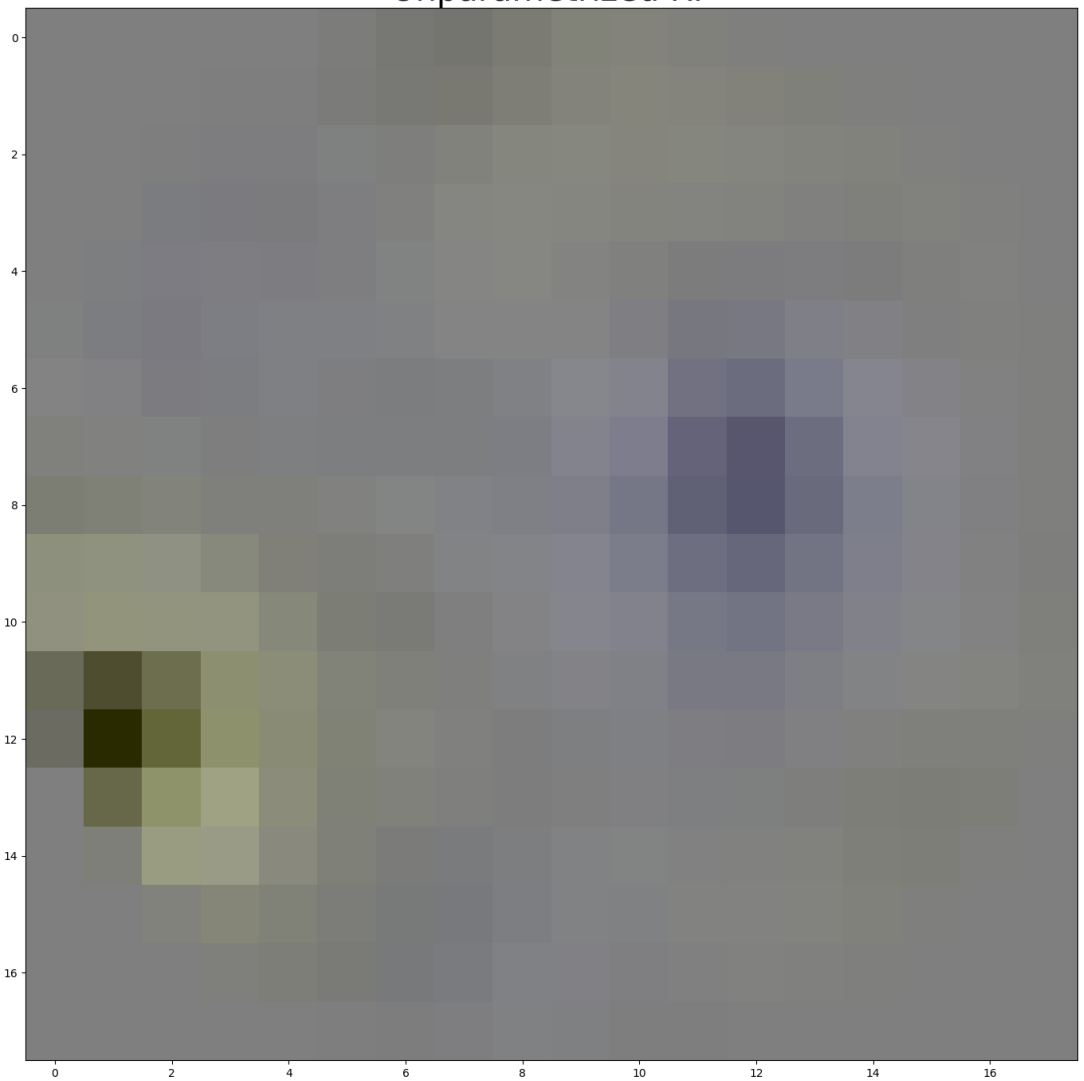


DoG fit

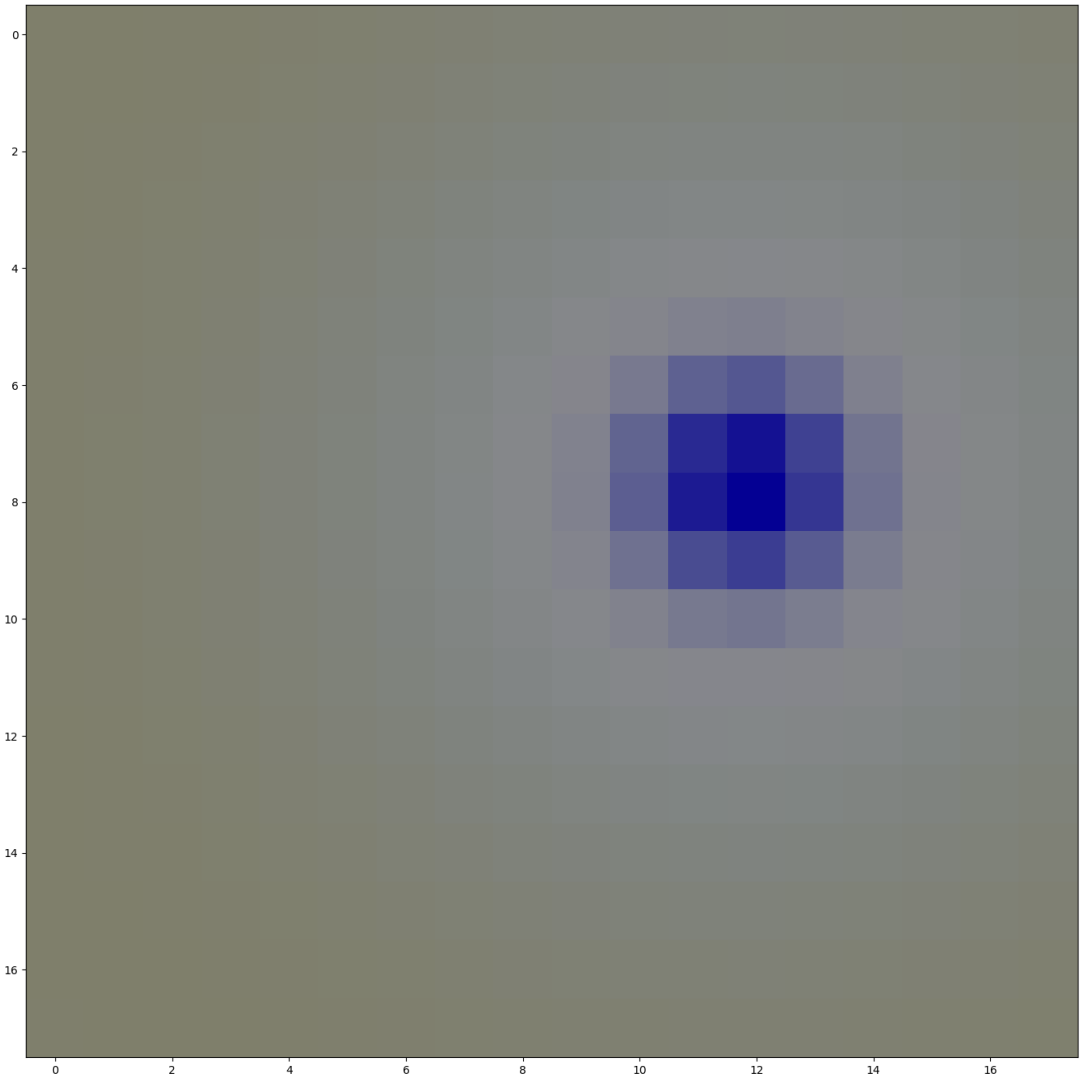


cor = 0.4446, 240129-142655 #82

Unparametrized RF

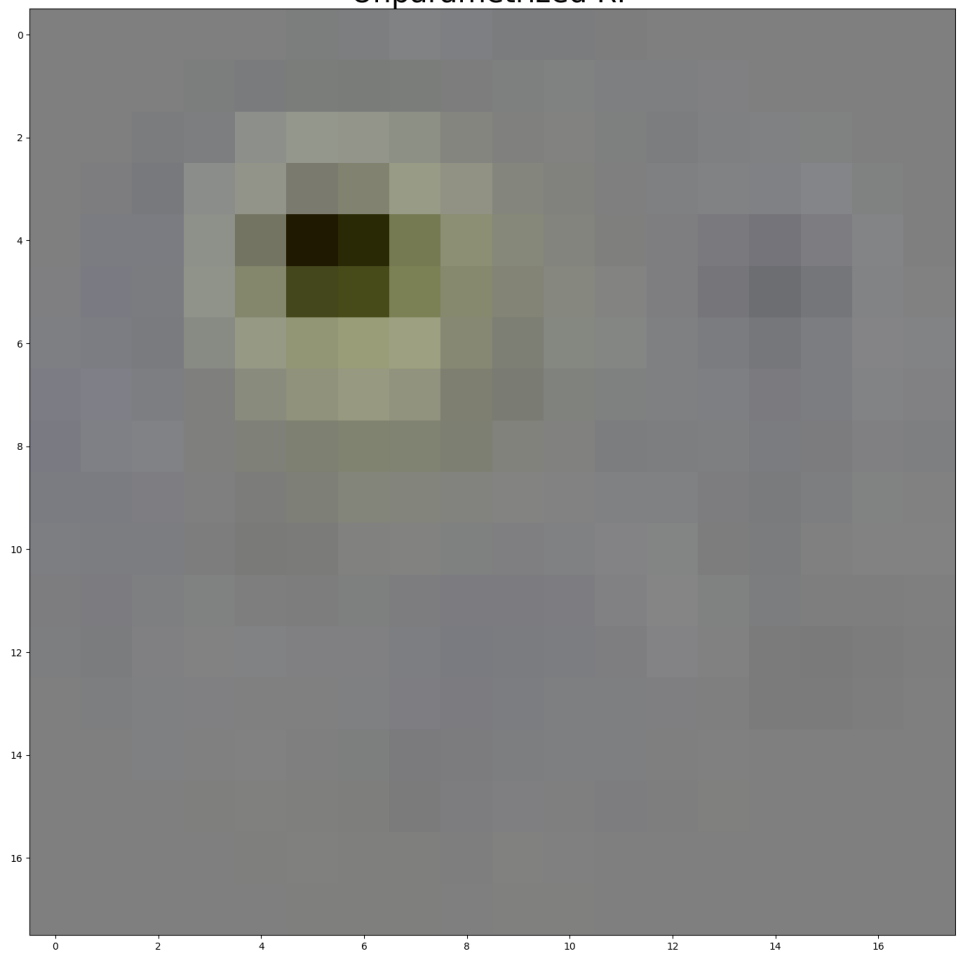


DoG fit

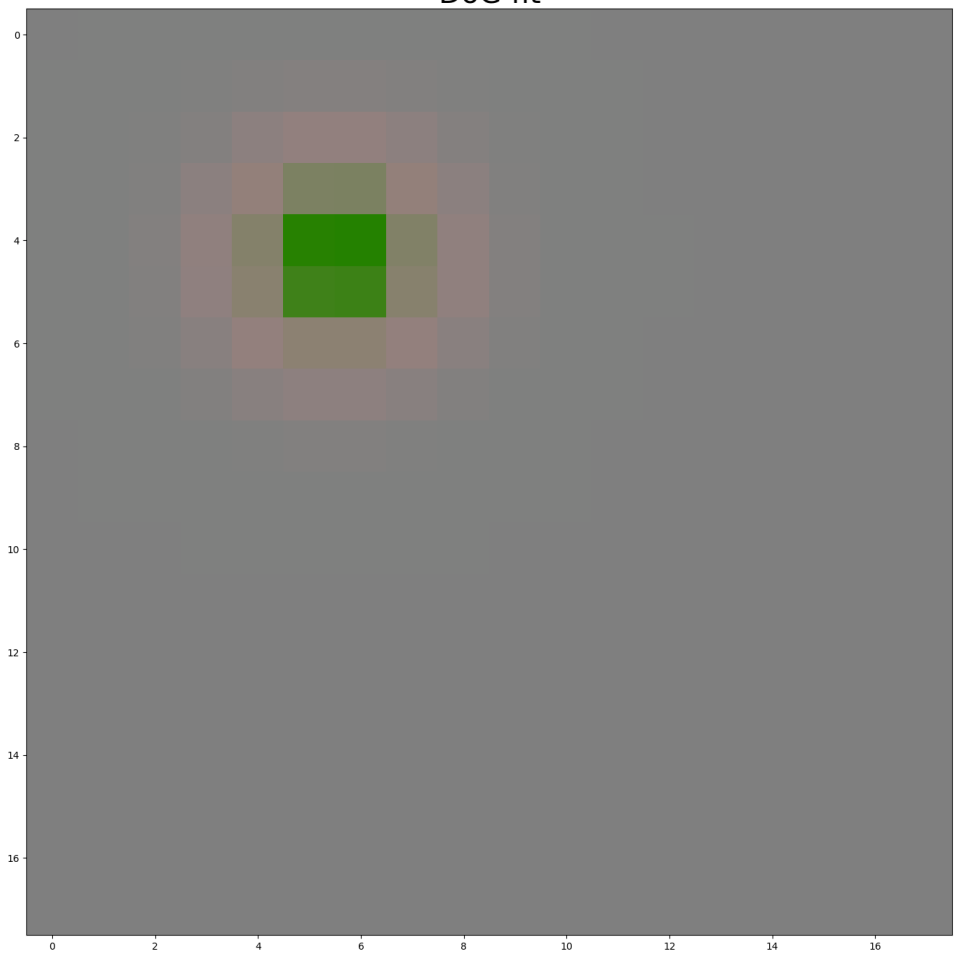


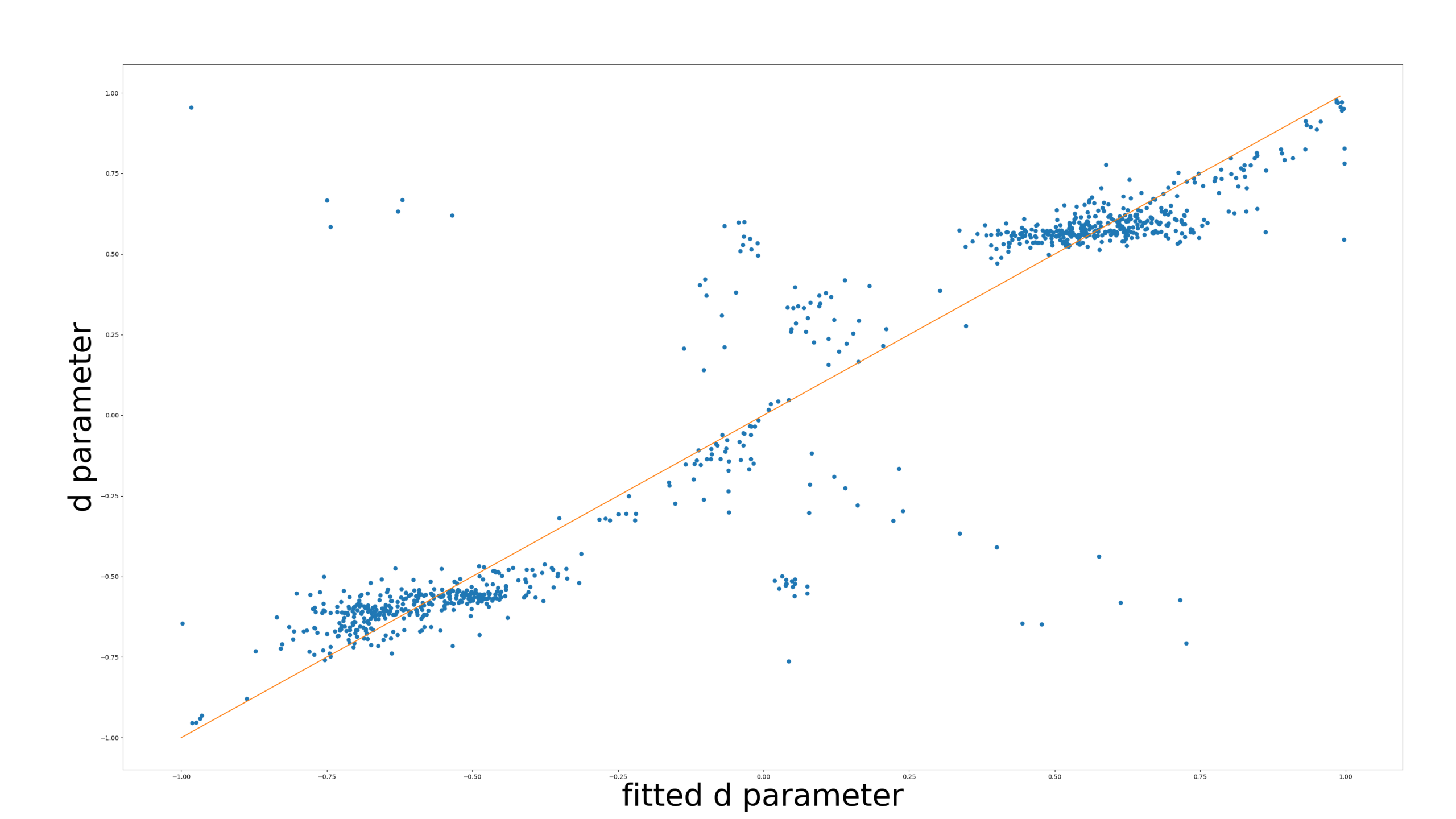
correlation = 0.8018058686737726

Unparametrized RF

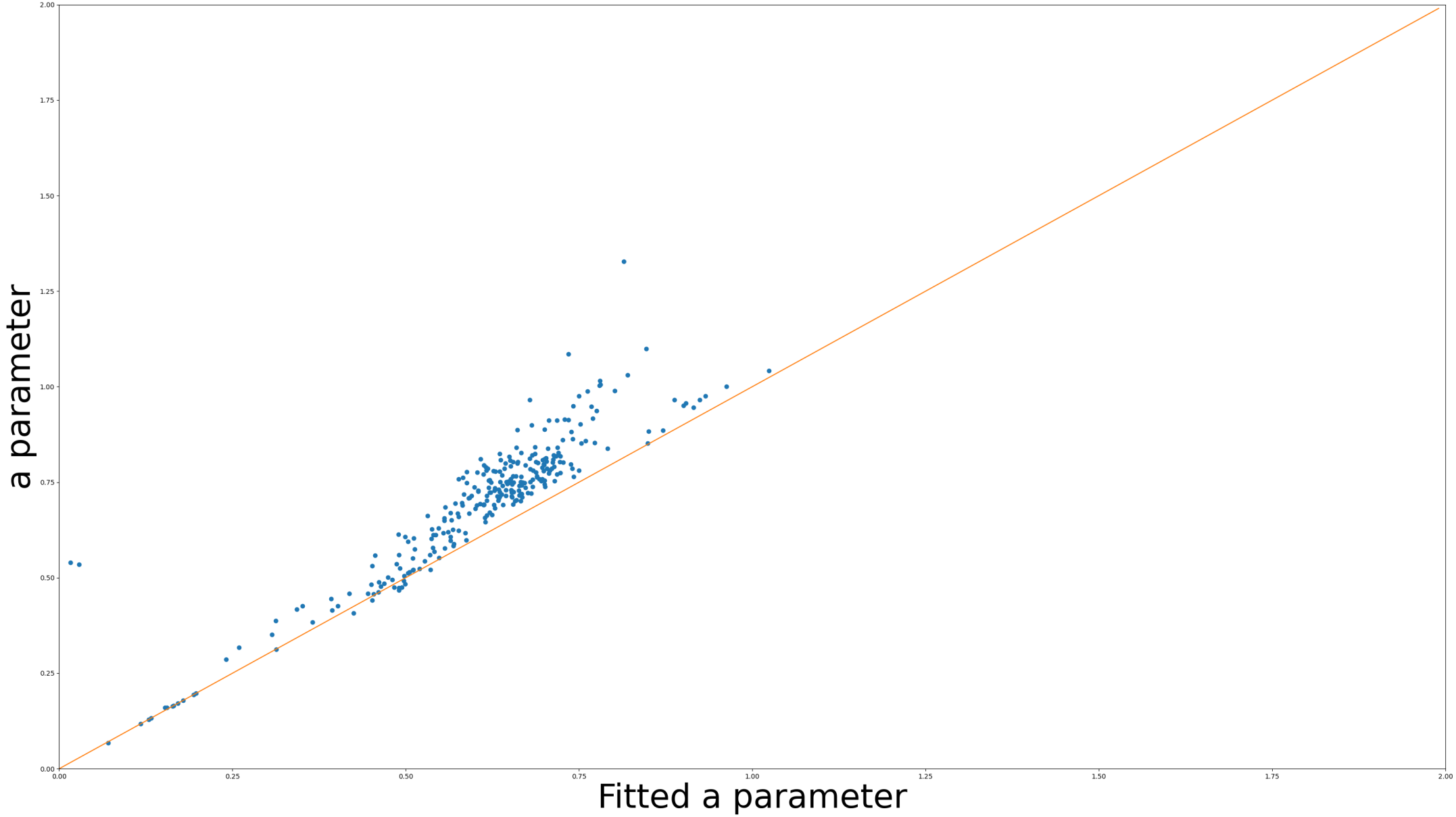


DoG fit

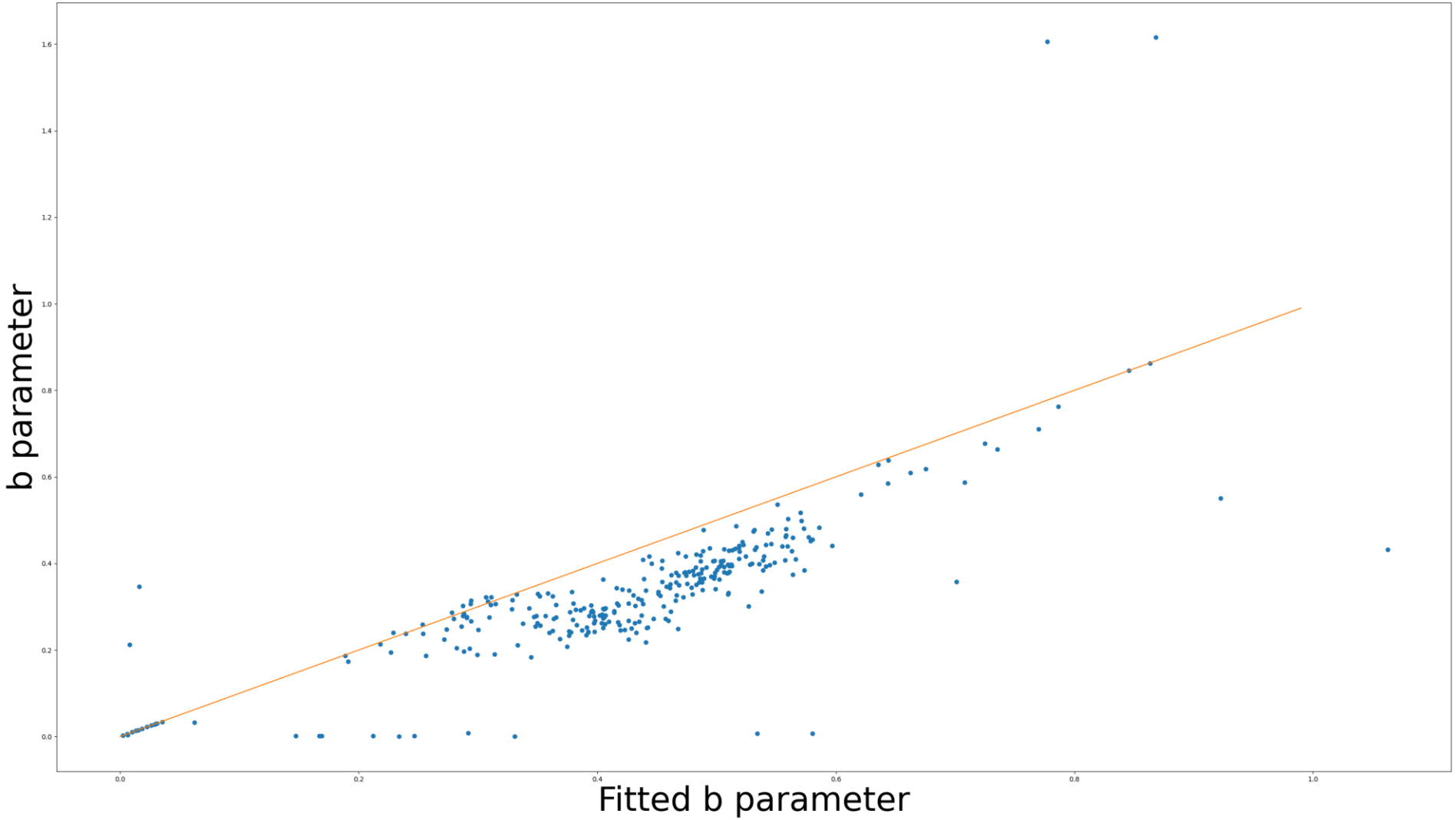



d parameter
fitted d parameter
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00

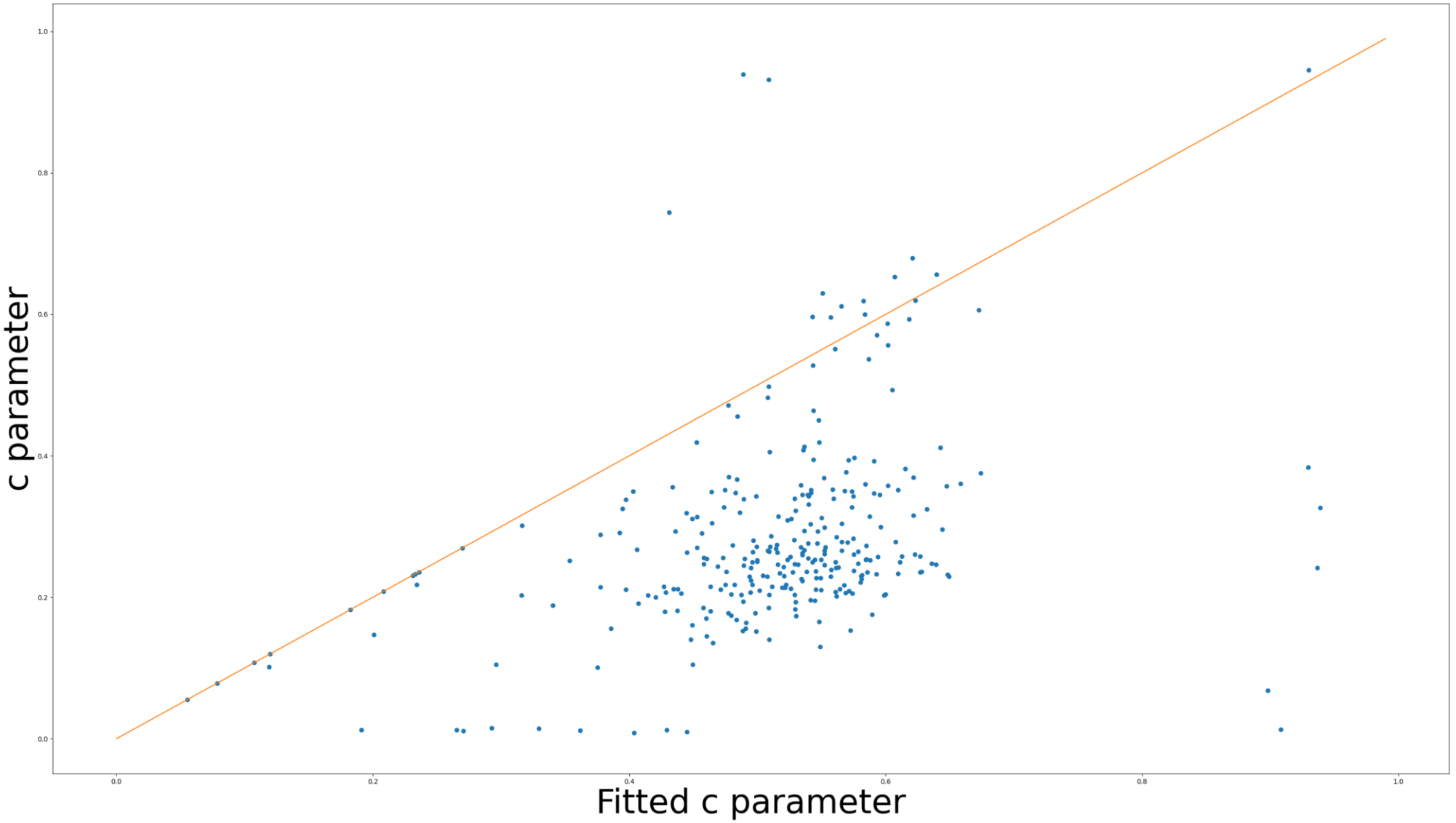
Only the most important channel



Only the most important channel



Only the most important channel



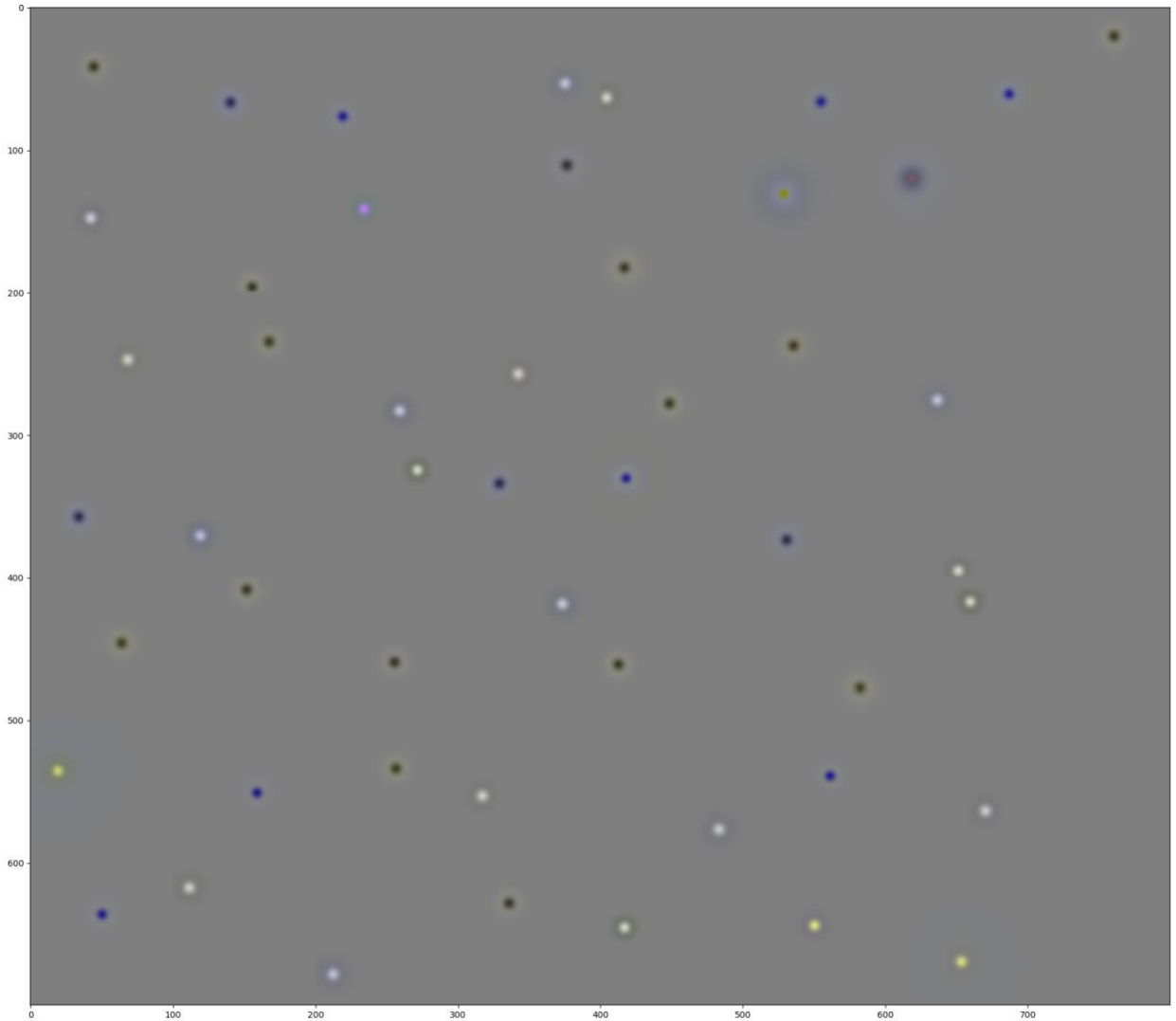
Color RFs

a) Directly looking at color RFs is unreliable

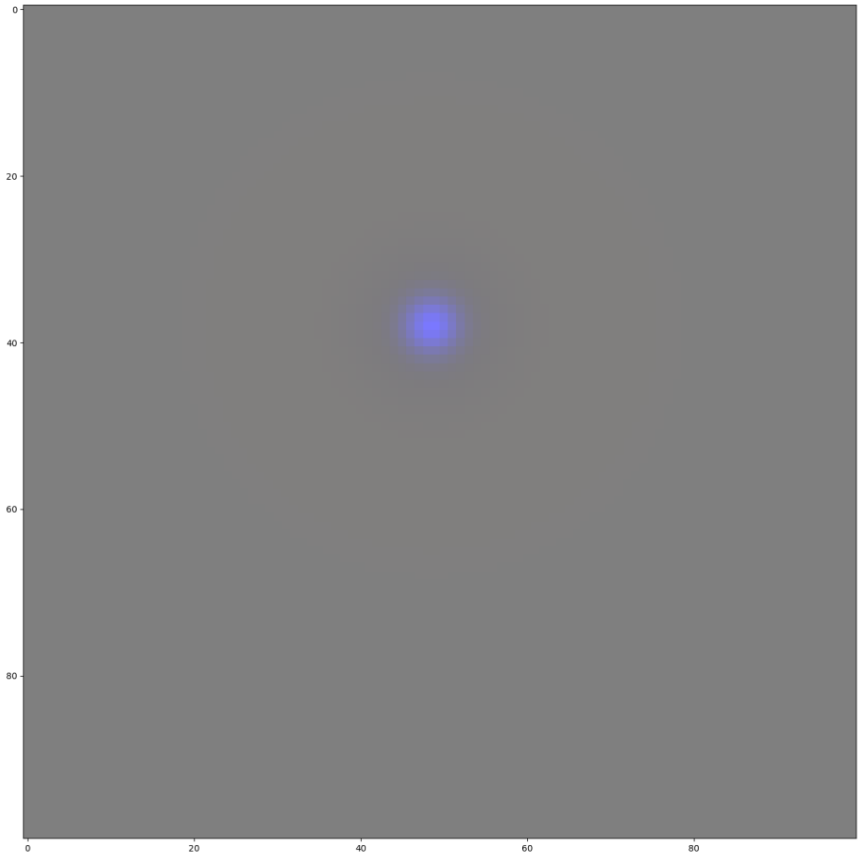
b) Near absence of –S RFs

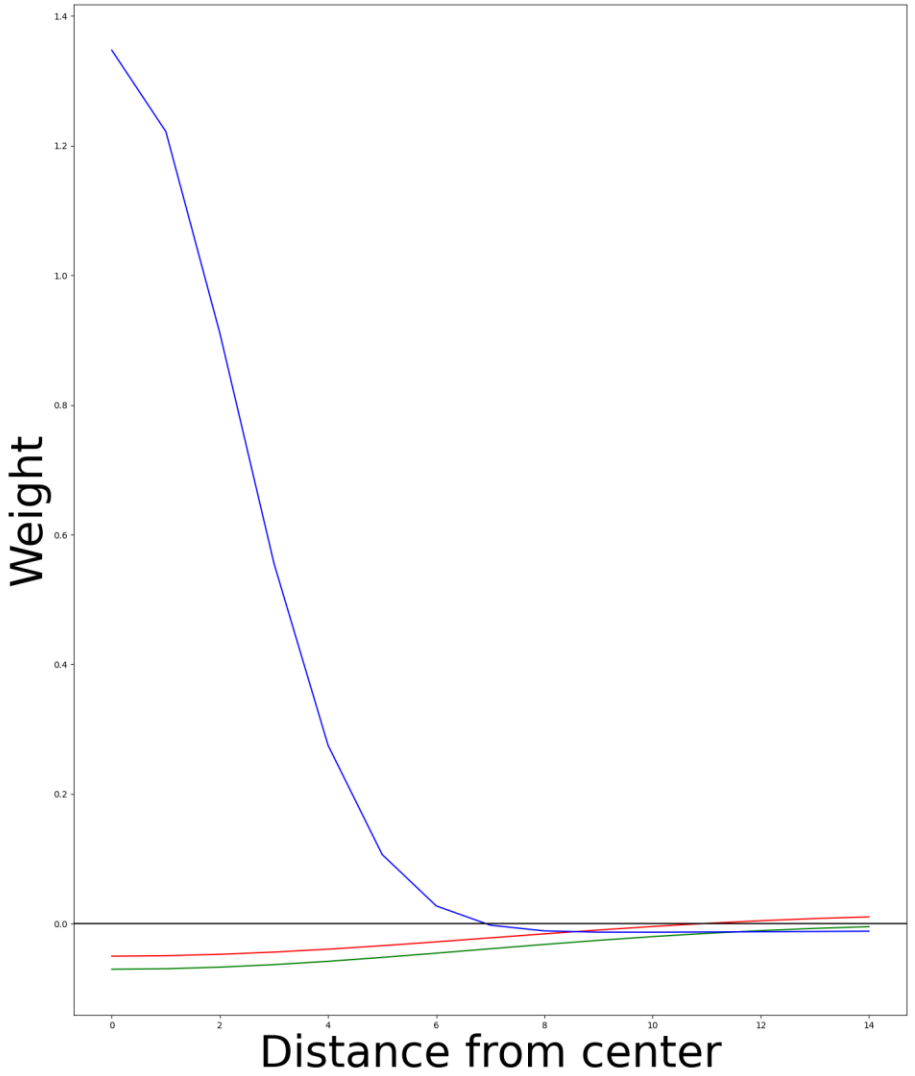
c) S inputs have much weaker surrounds

Looking at colors is unreliable

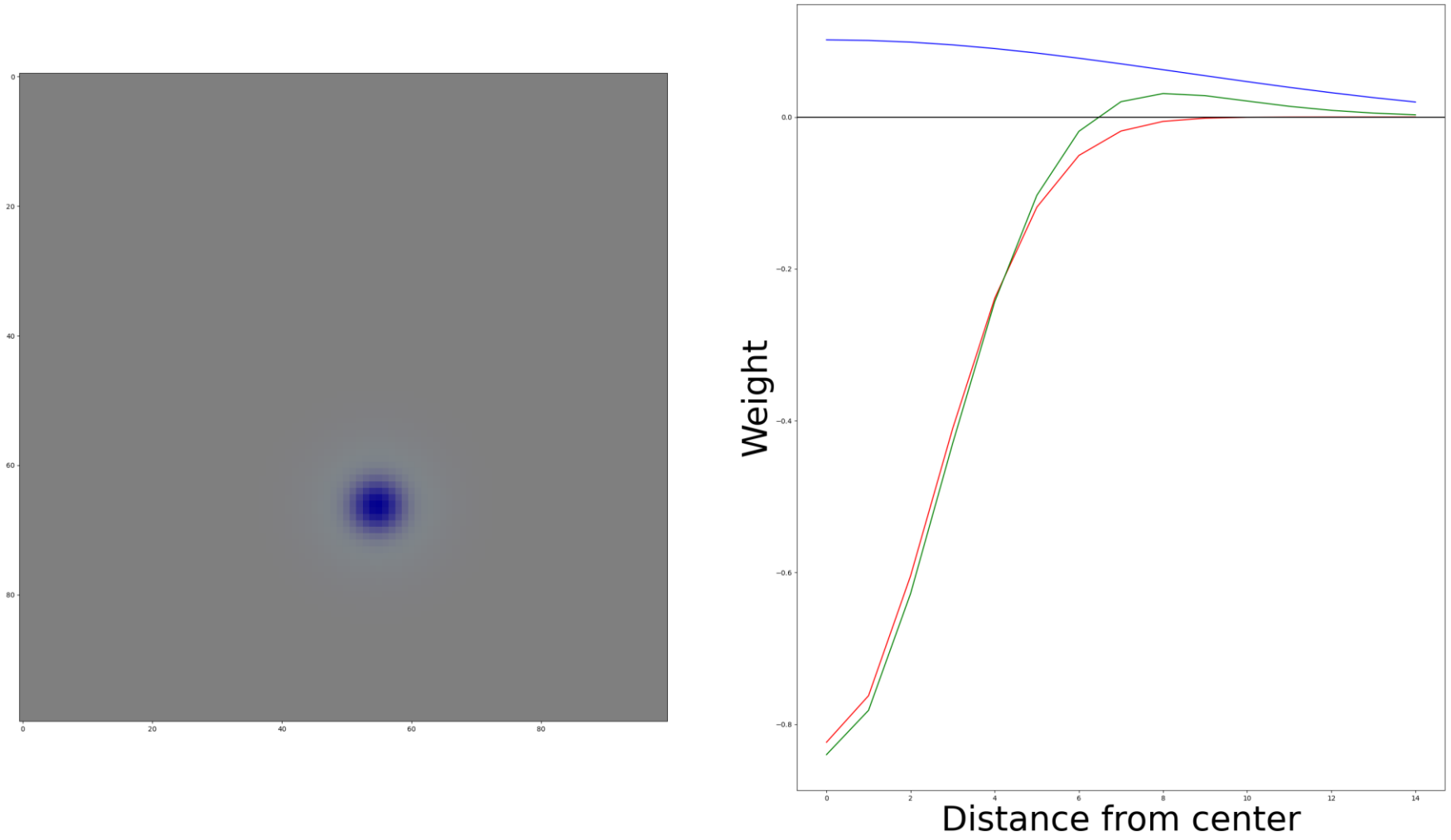


That’s what you would expect a “blue” RF to look like:

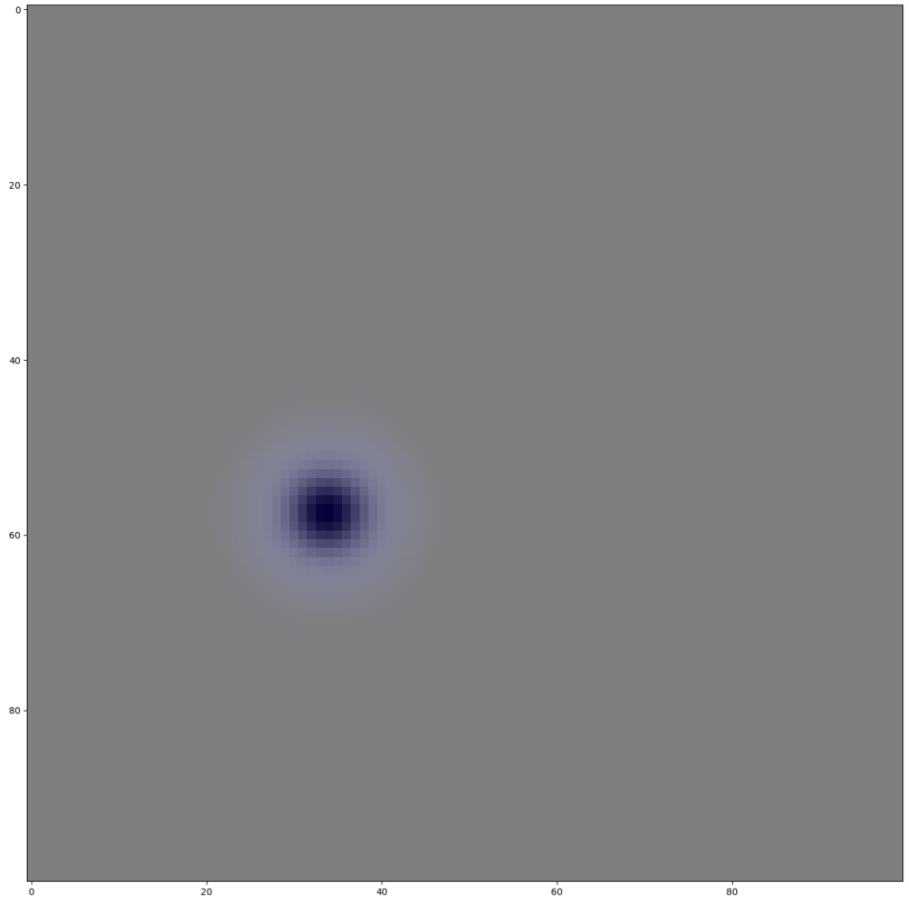


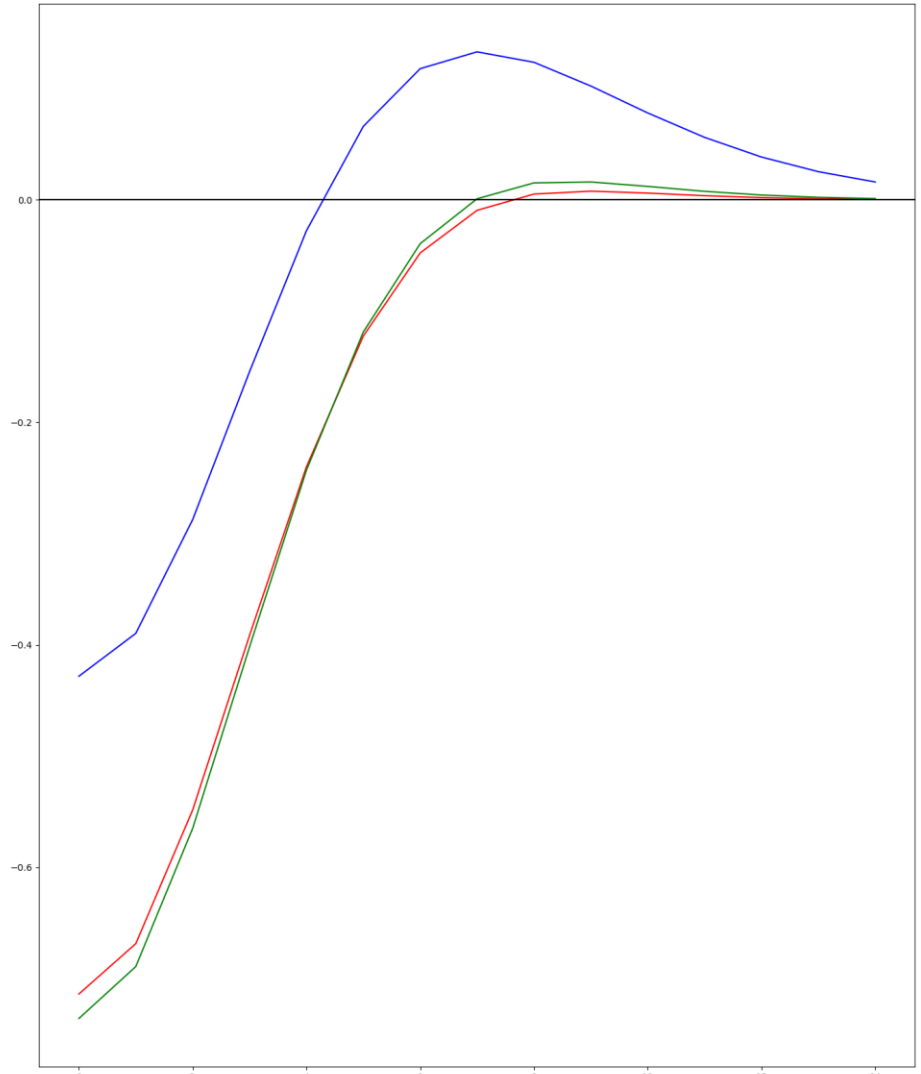


But most “blue” RFs look like this:

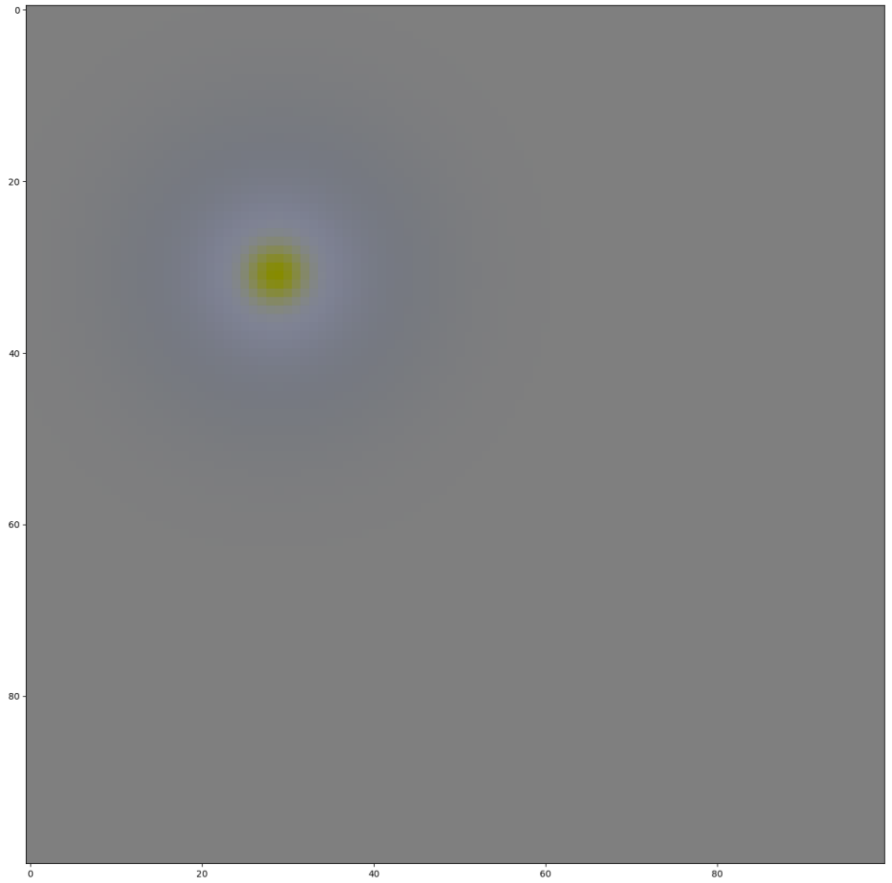


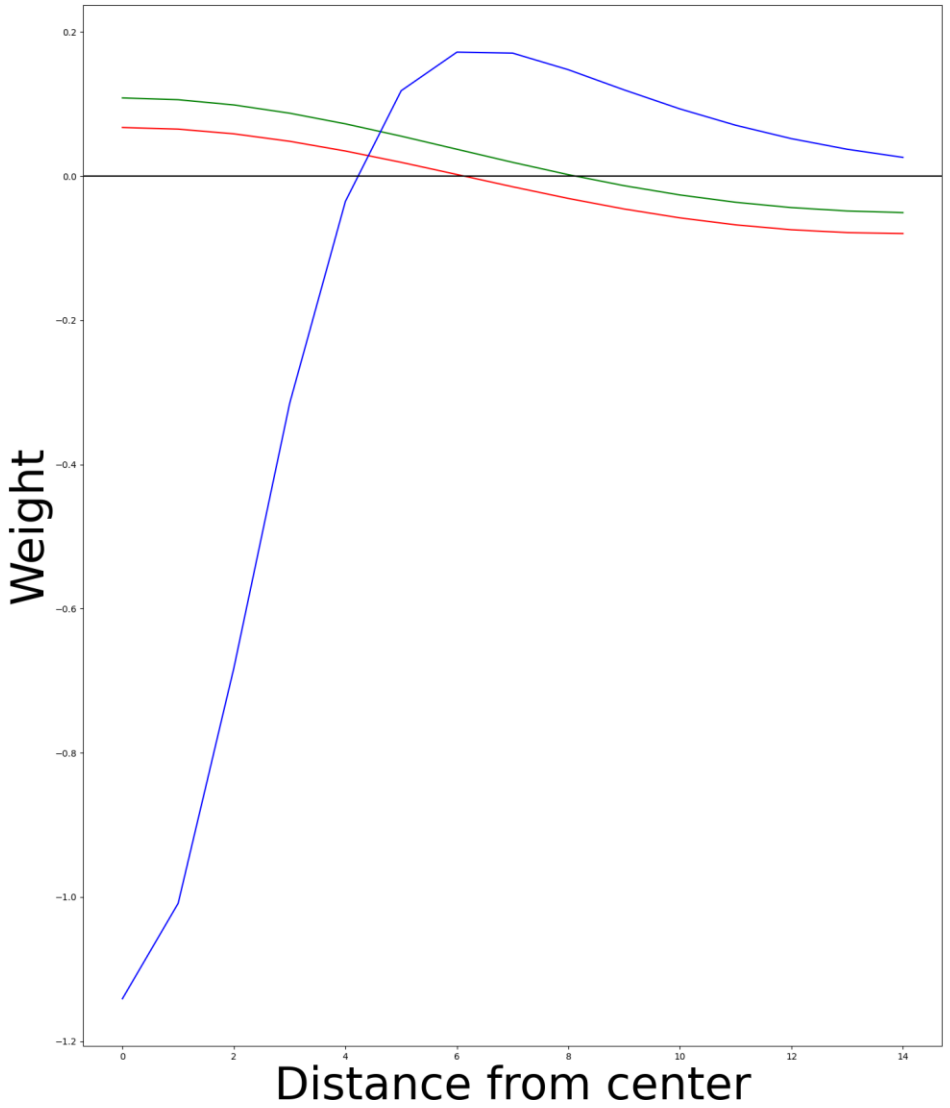
Or even this:





We get only 2 –S RFs:
Hypothesis: These might disappear if we get closer to global minimum



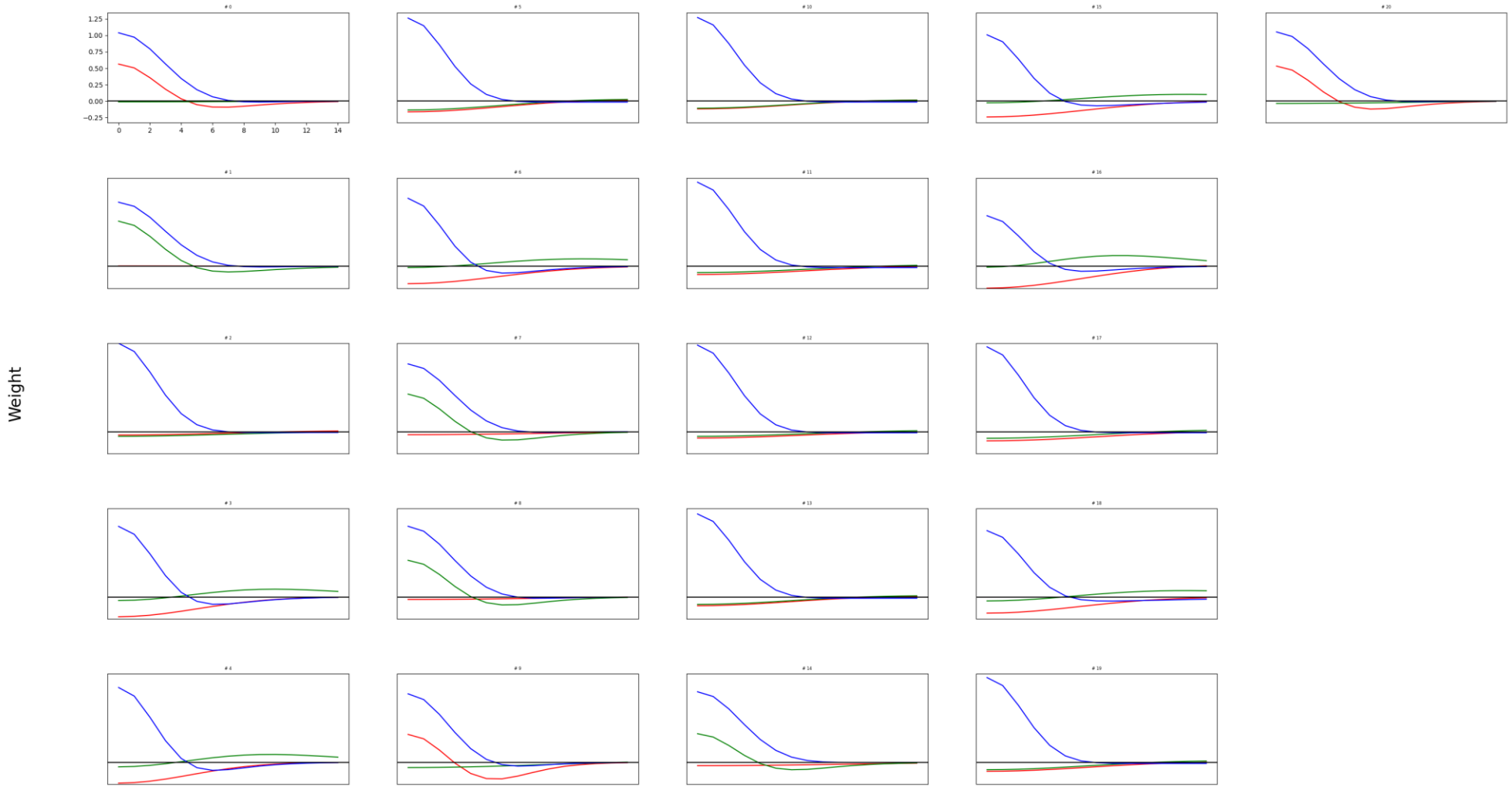


What RFs are in each mosaic?

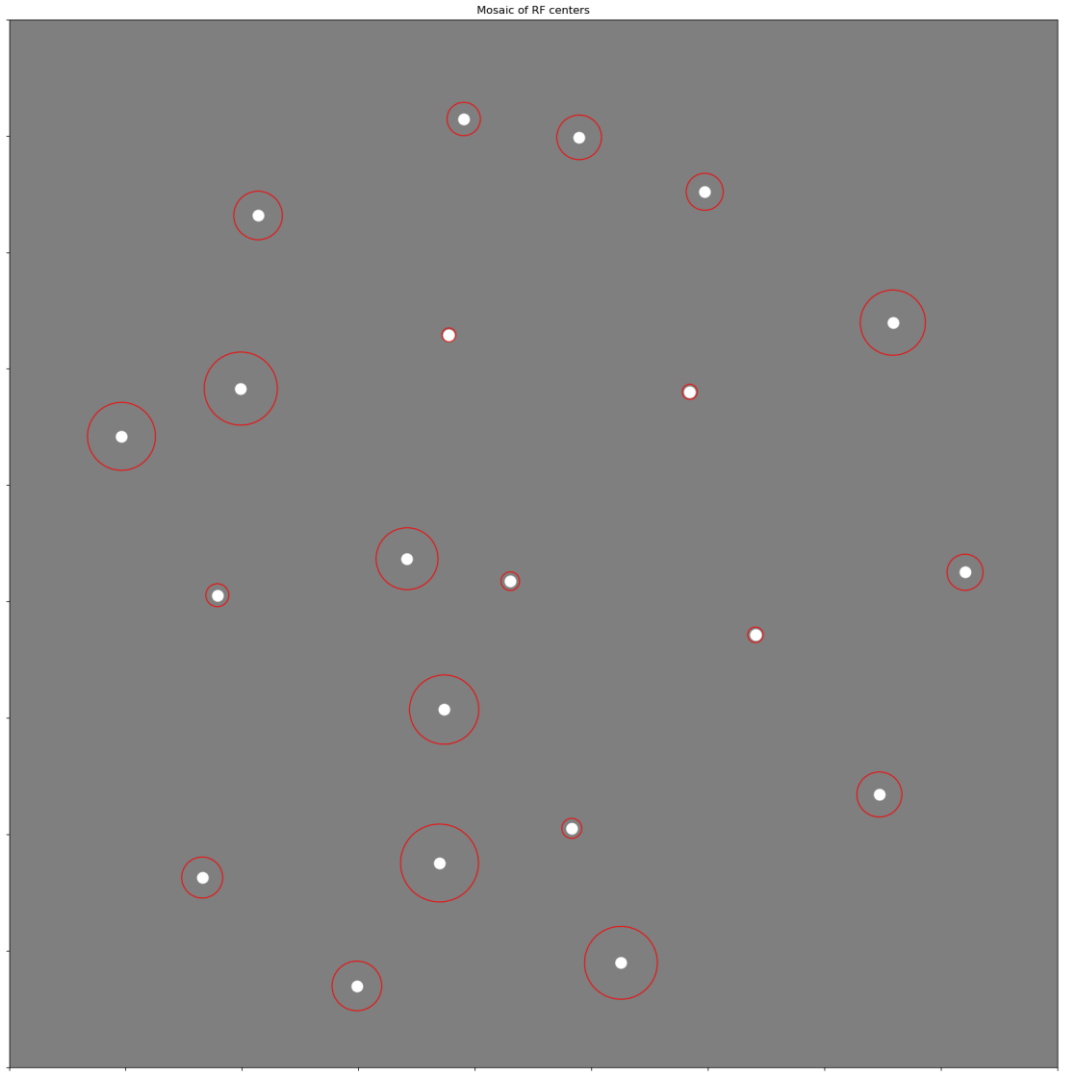
Or how many koniocellular RFs do we get

And are mosaics real

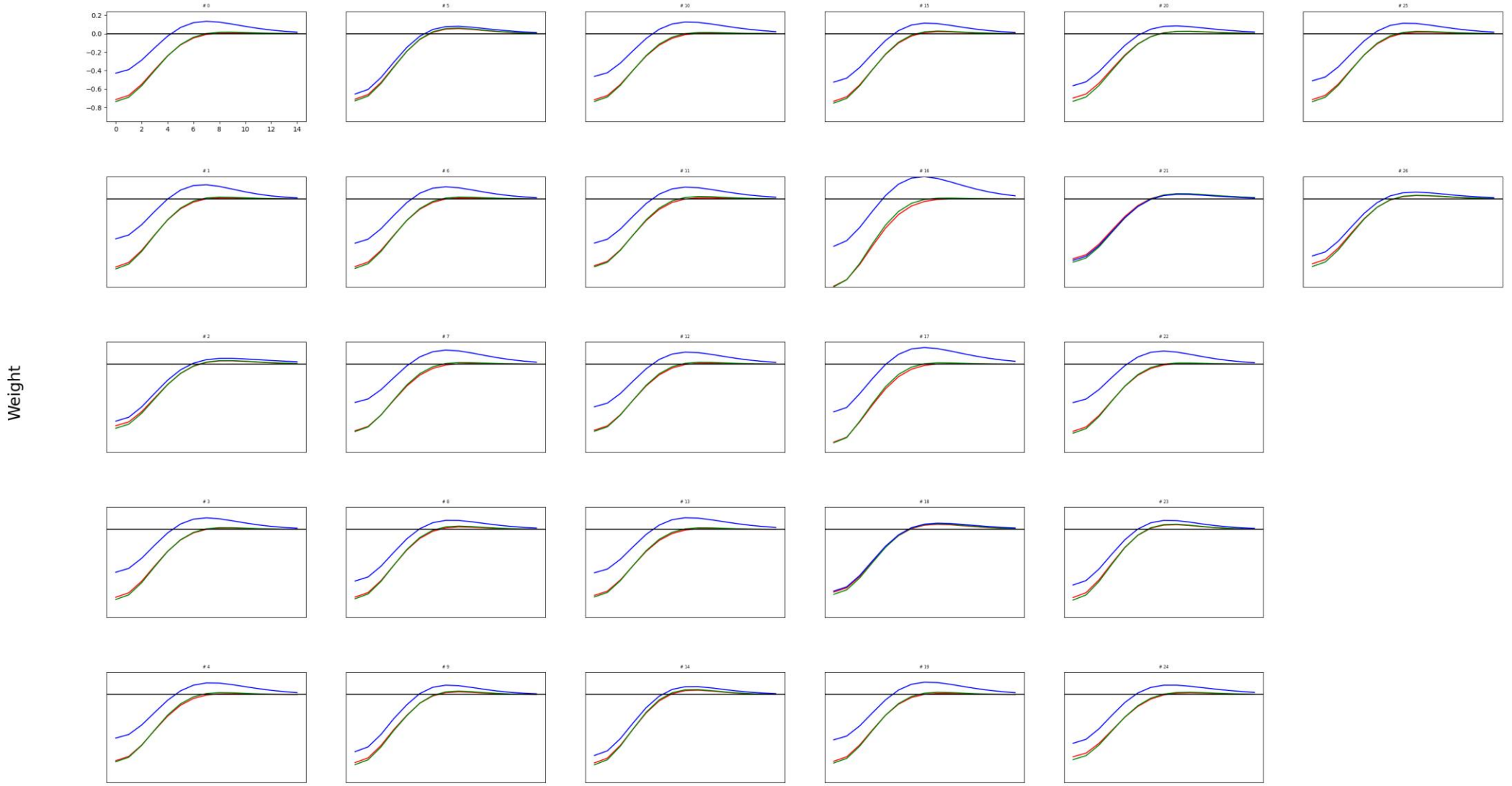
First type (Konio)



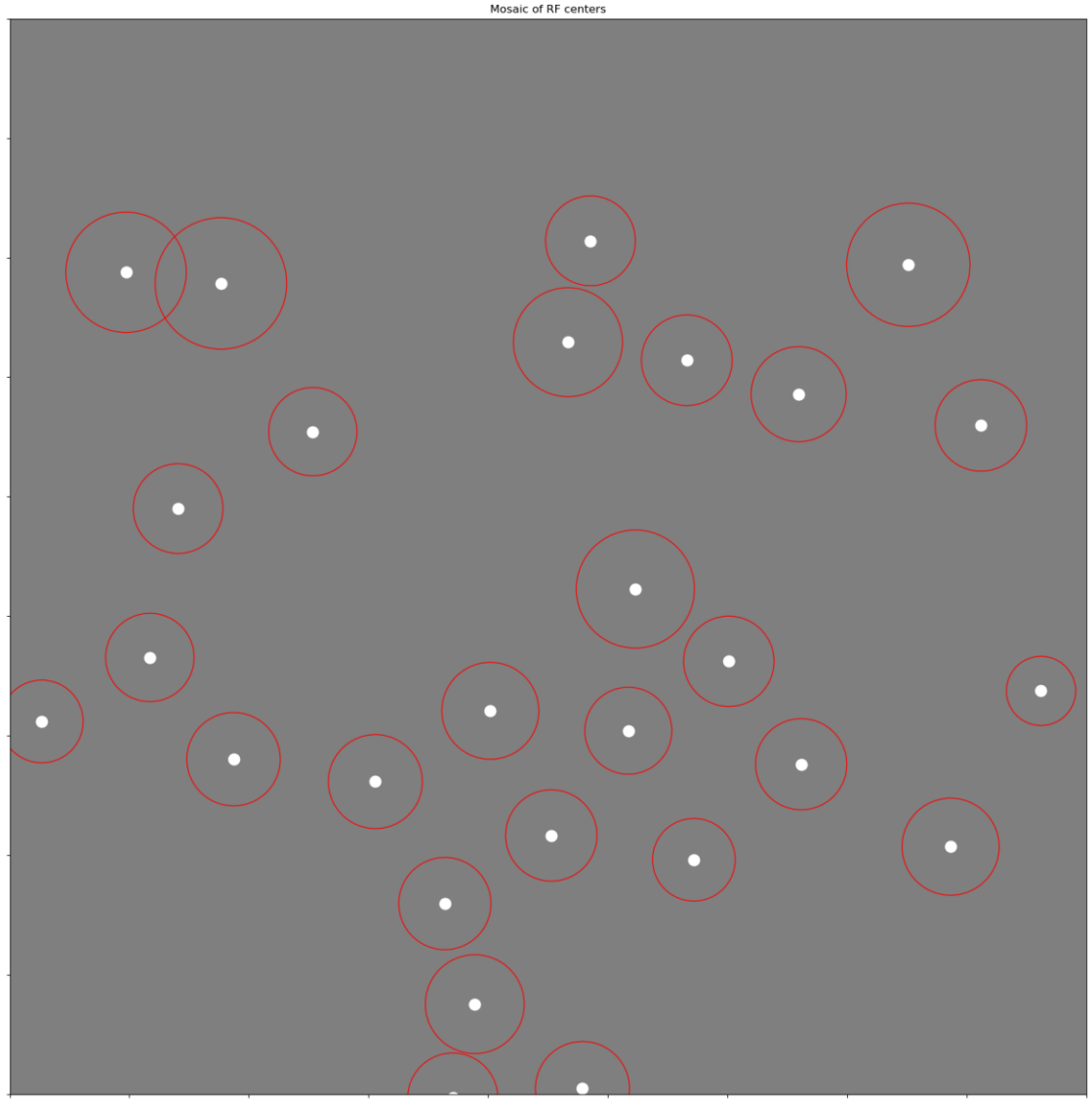
First type (Konio)



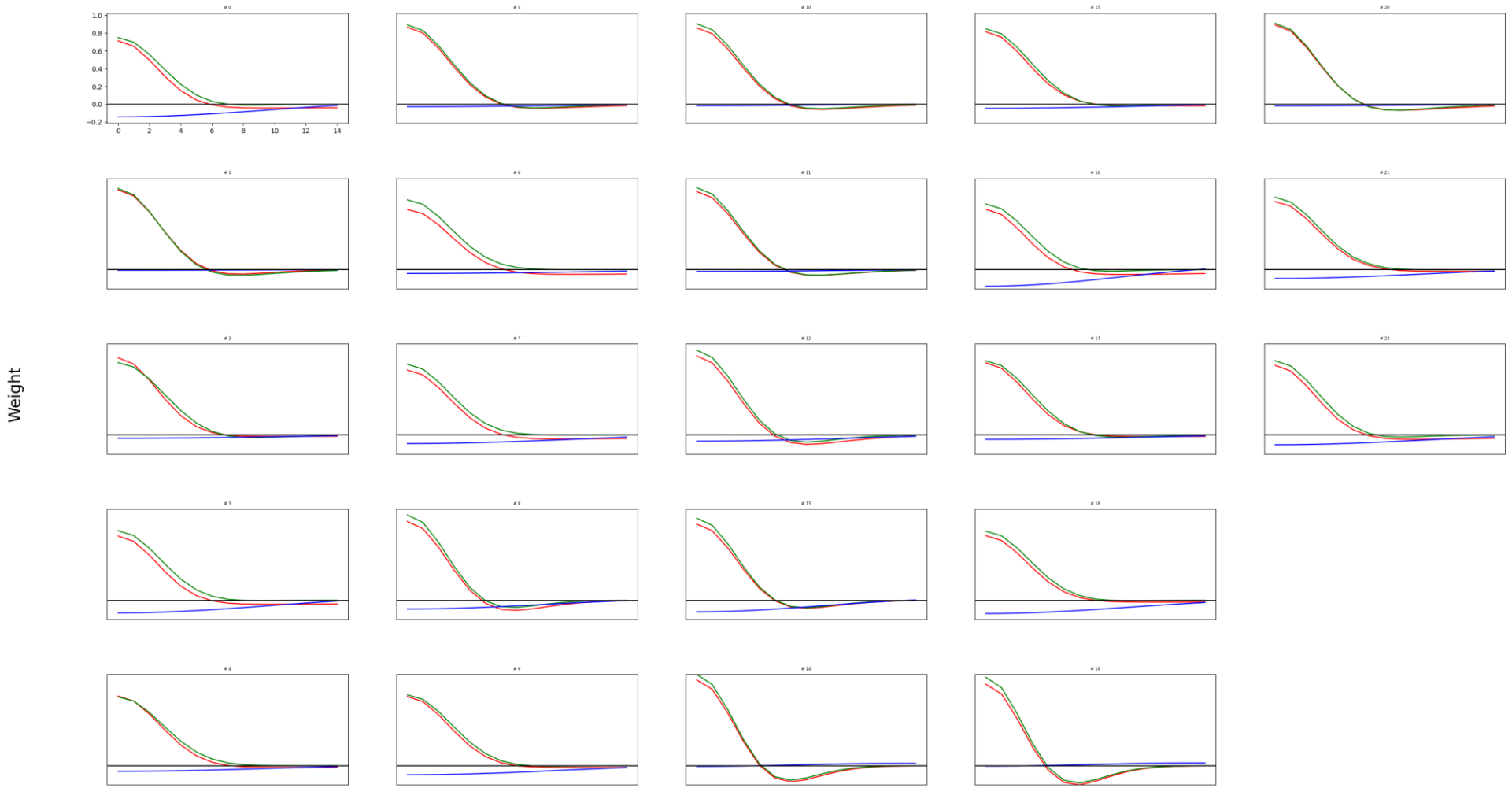
Second type (OFF parvo with –S inputs?)



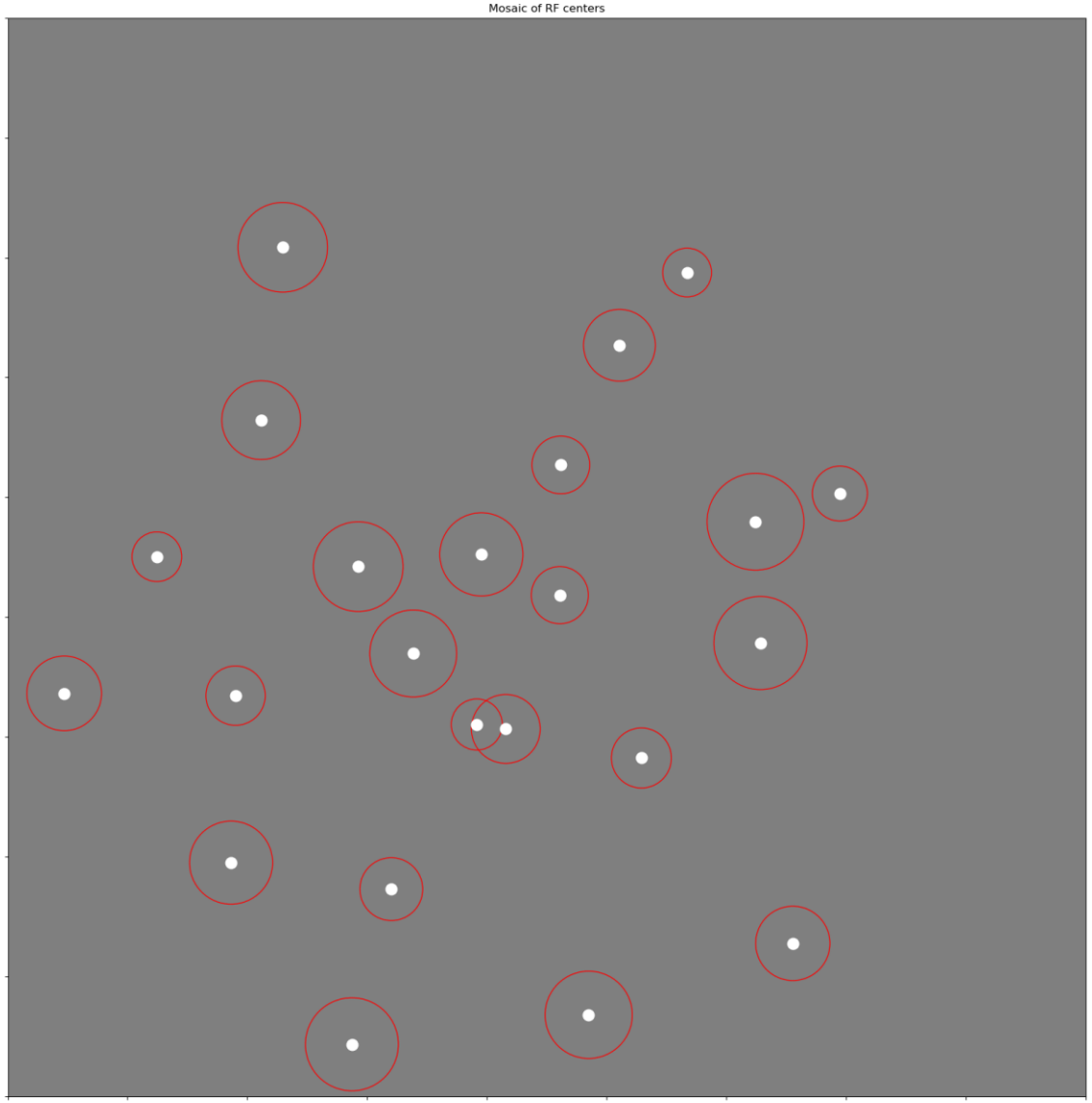
Second type (OFF parvo with –S inputs?)



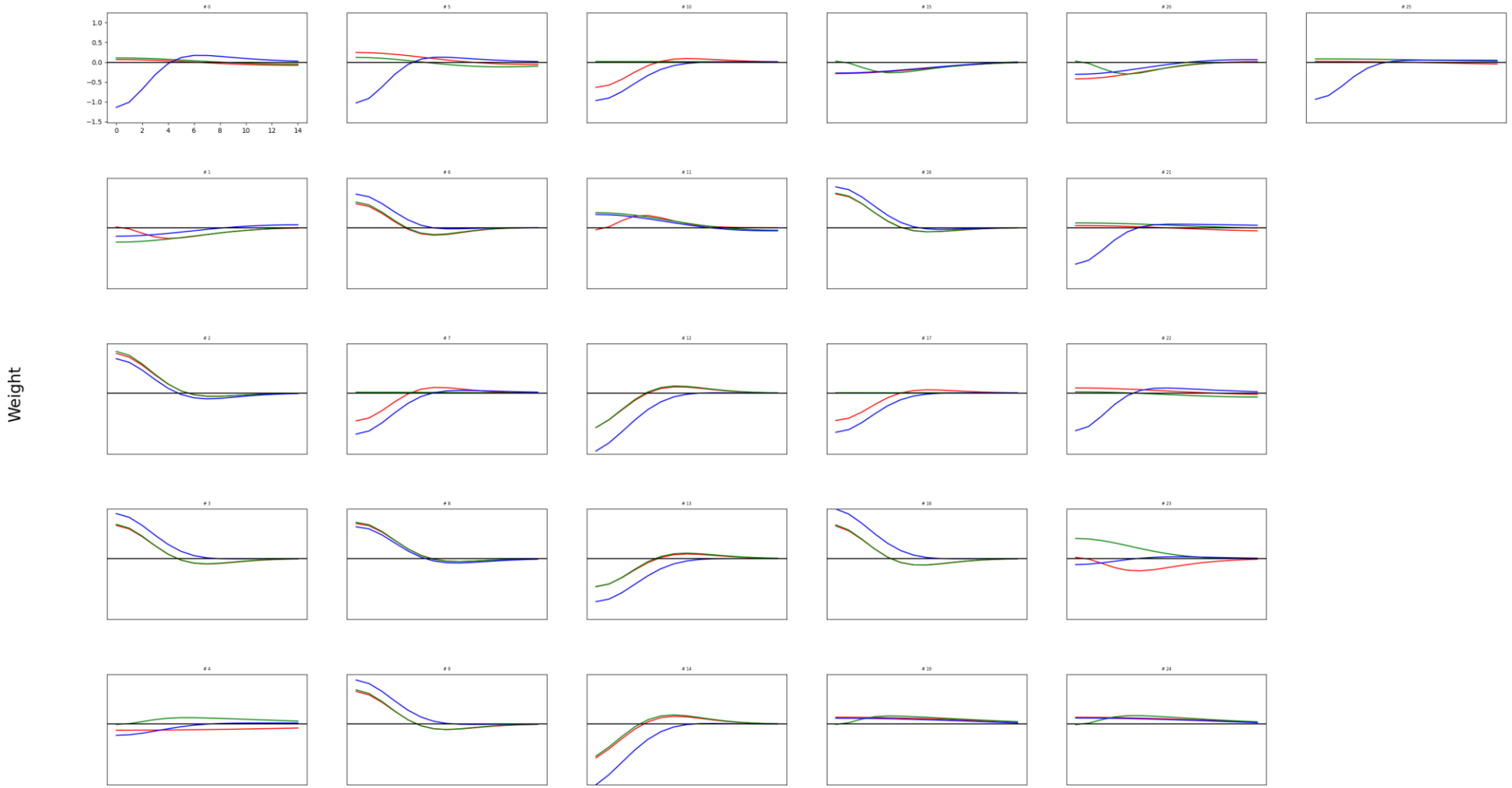
Third type (ON parvo)



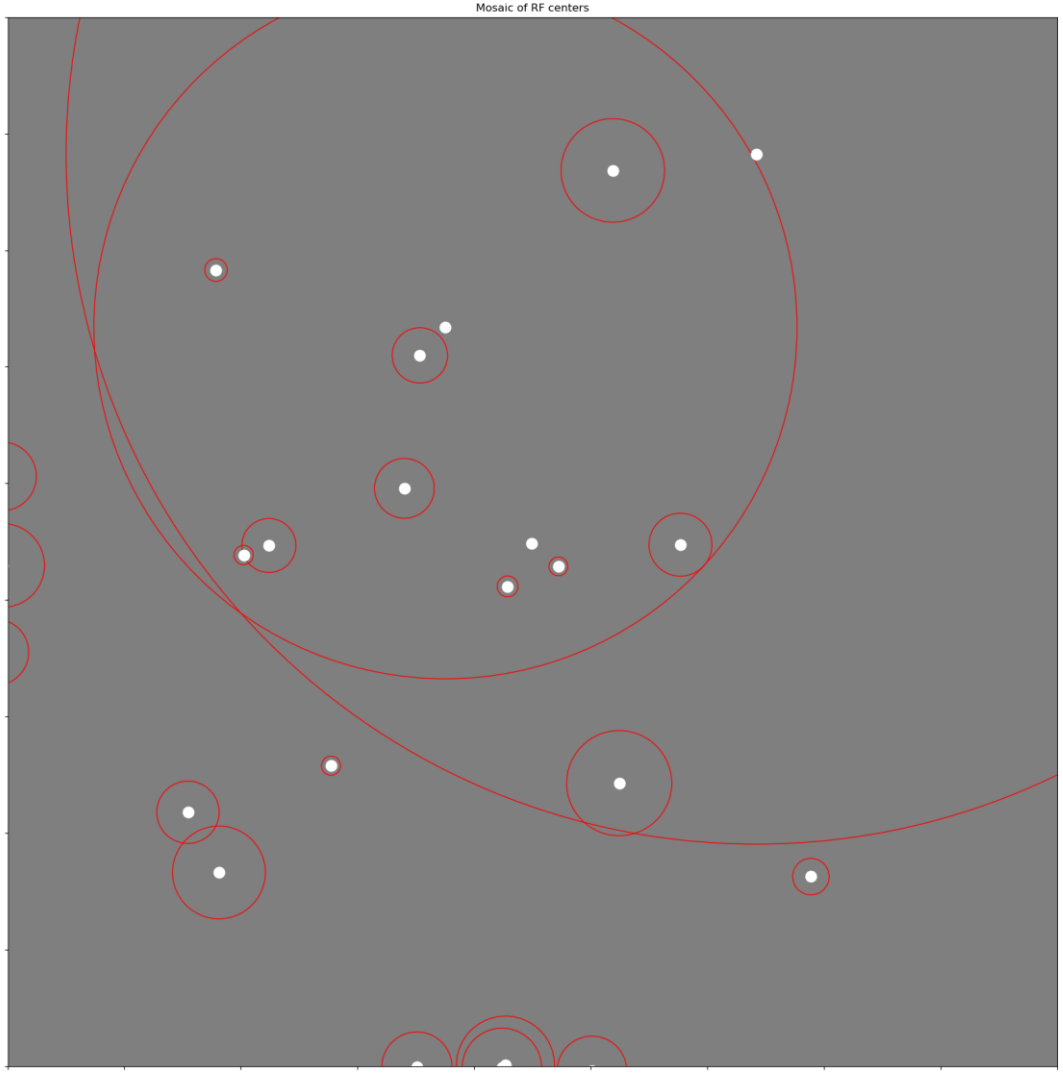
Third type (ON parvo)



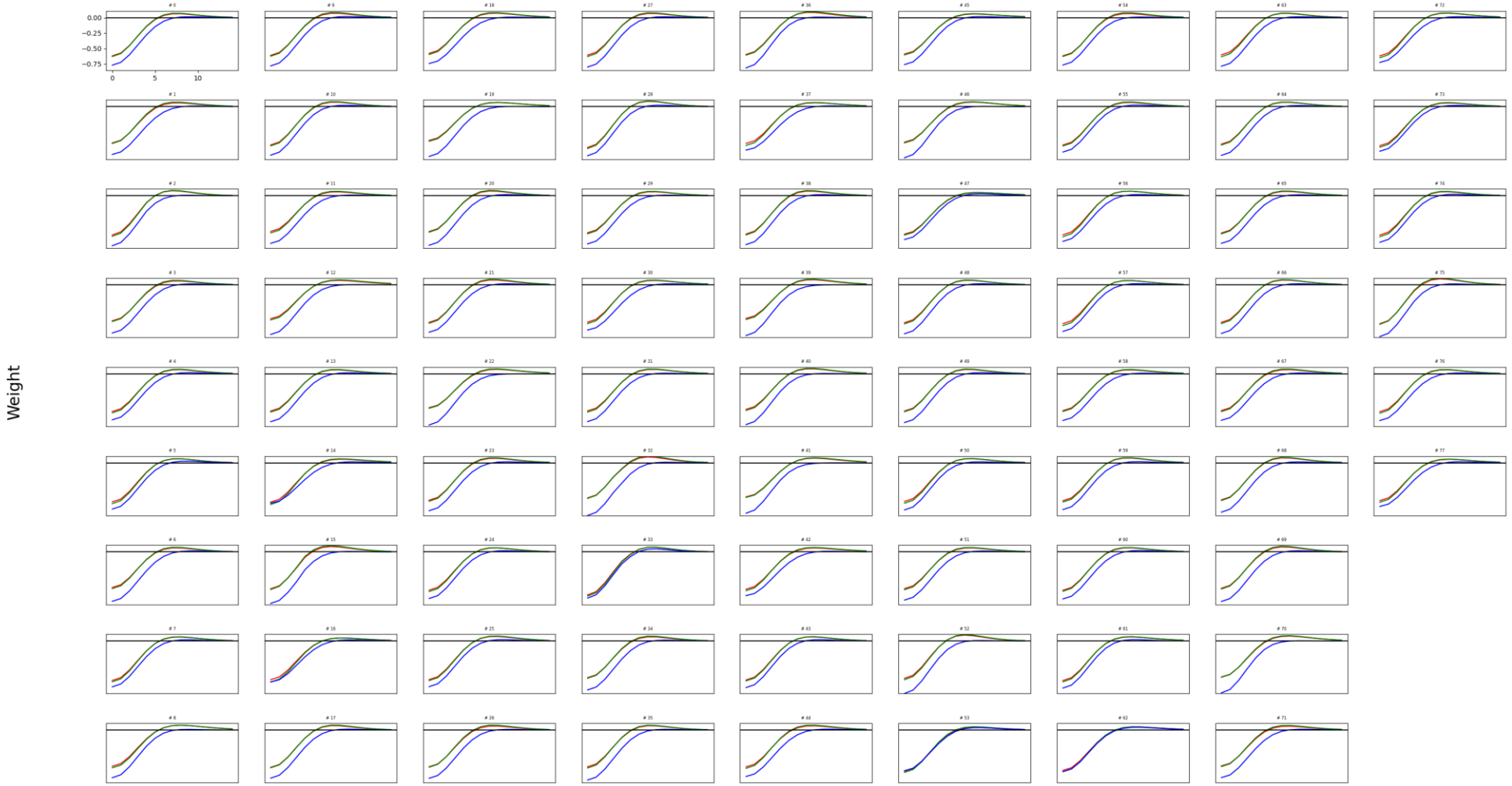
Fourth type (Trash can)



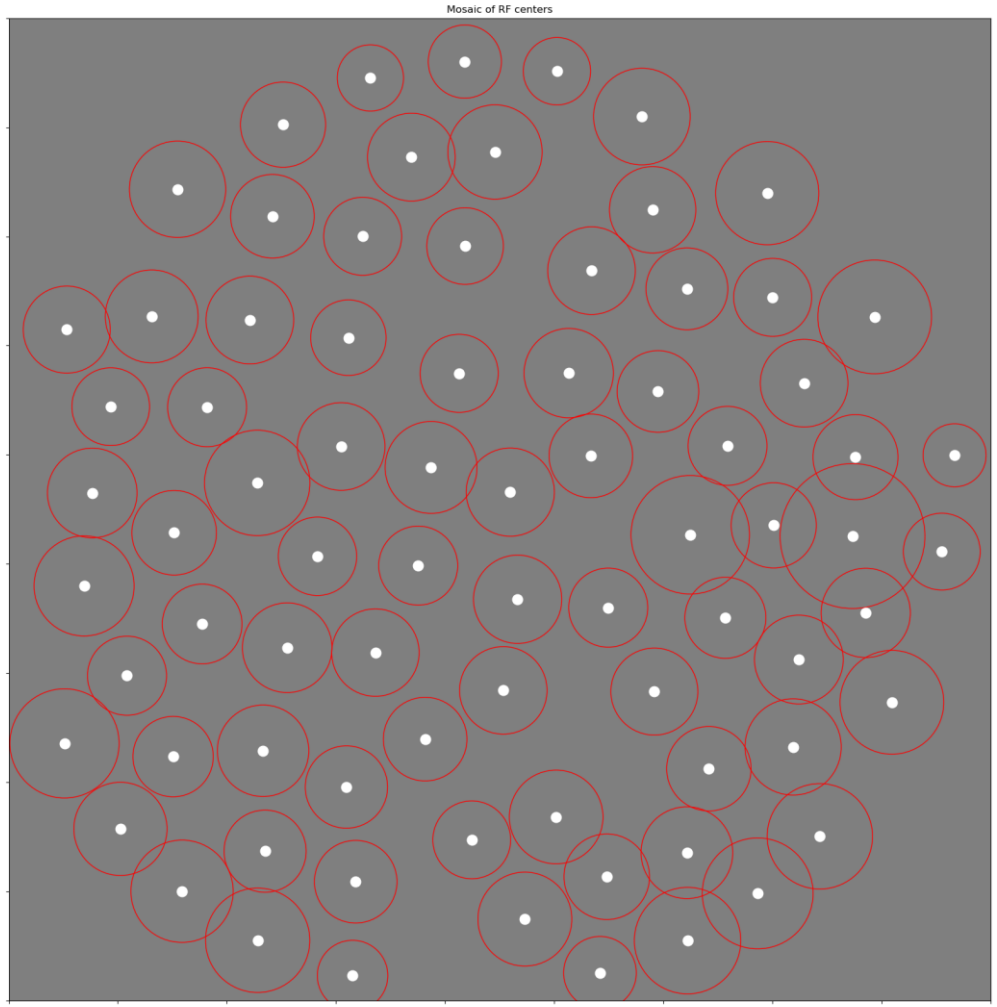
Fourth type (trash can)



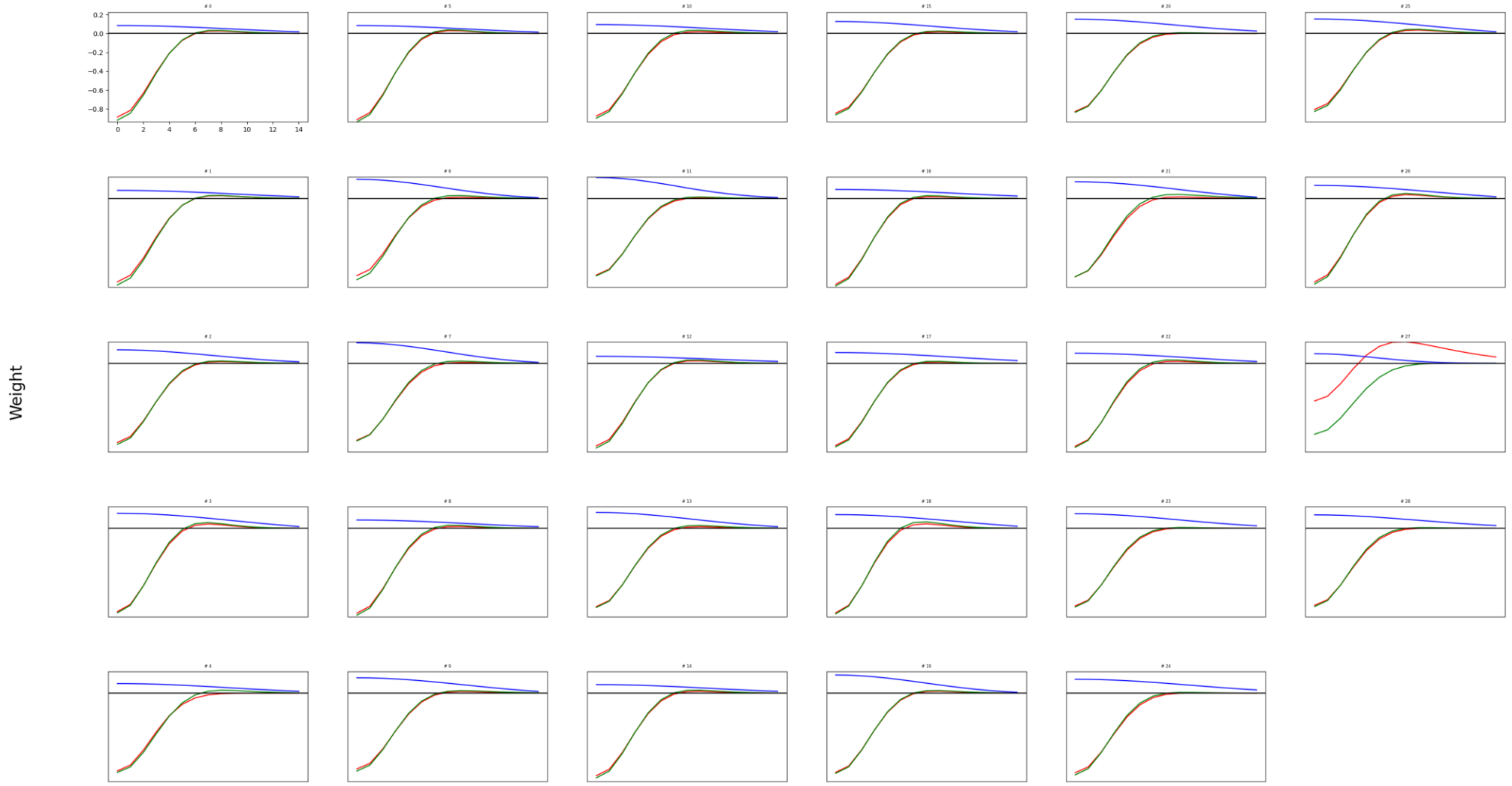
Fifth type (OFF magno)



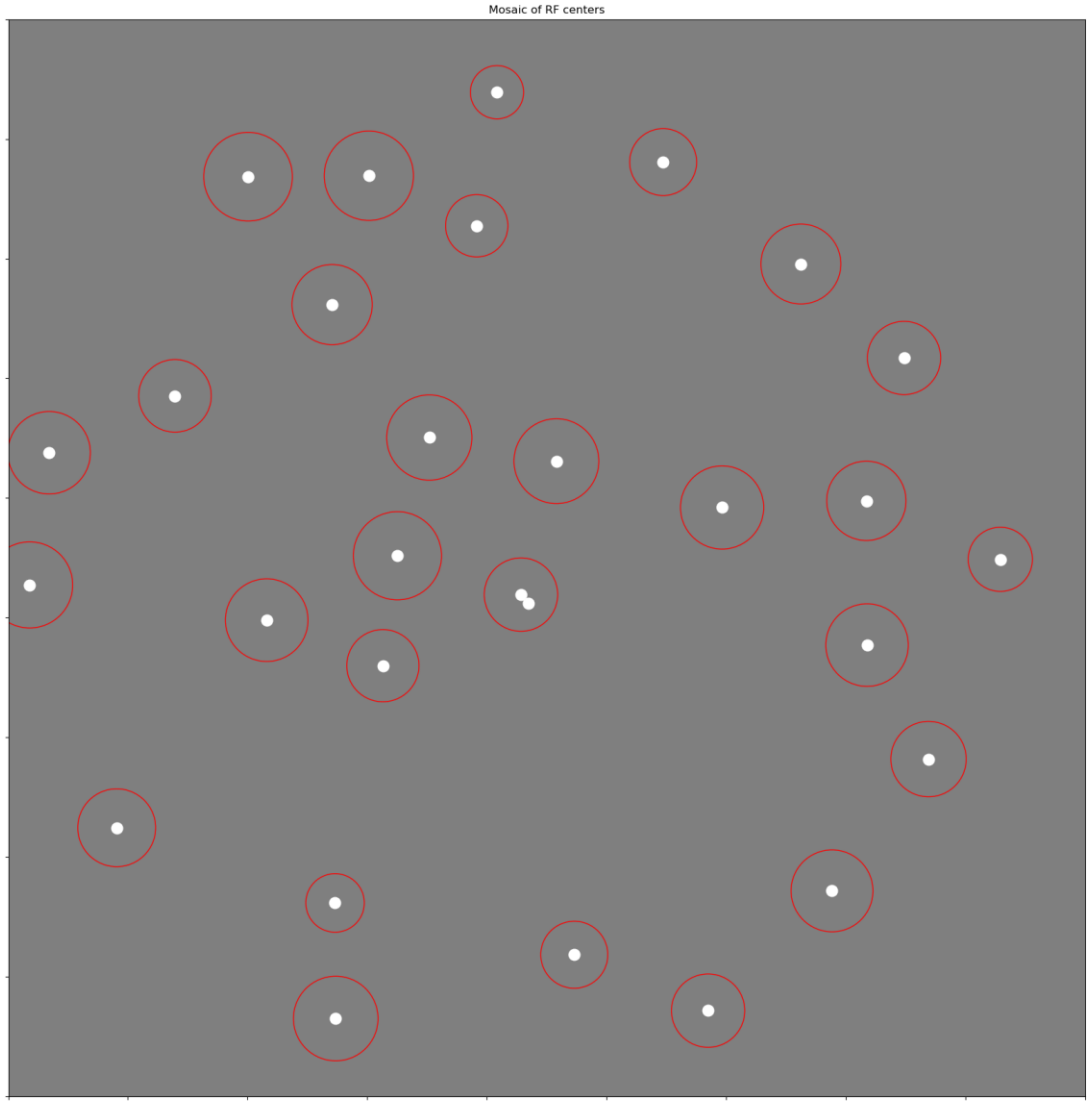
Fifth type (OFF magno)



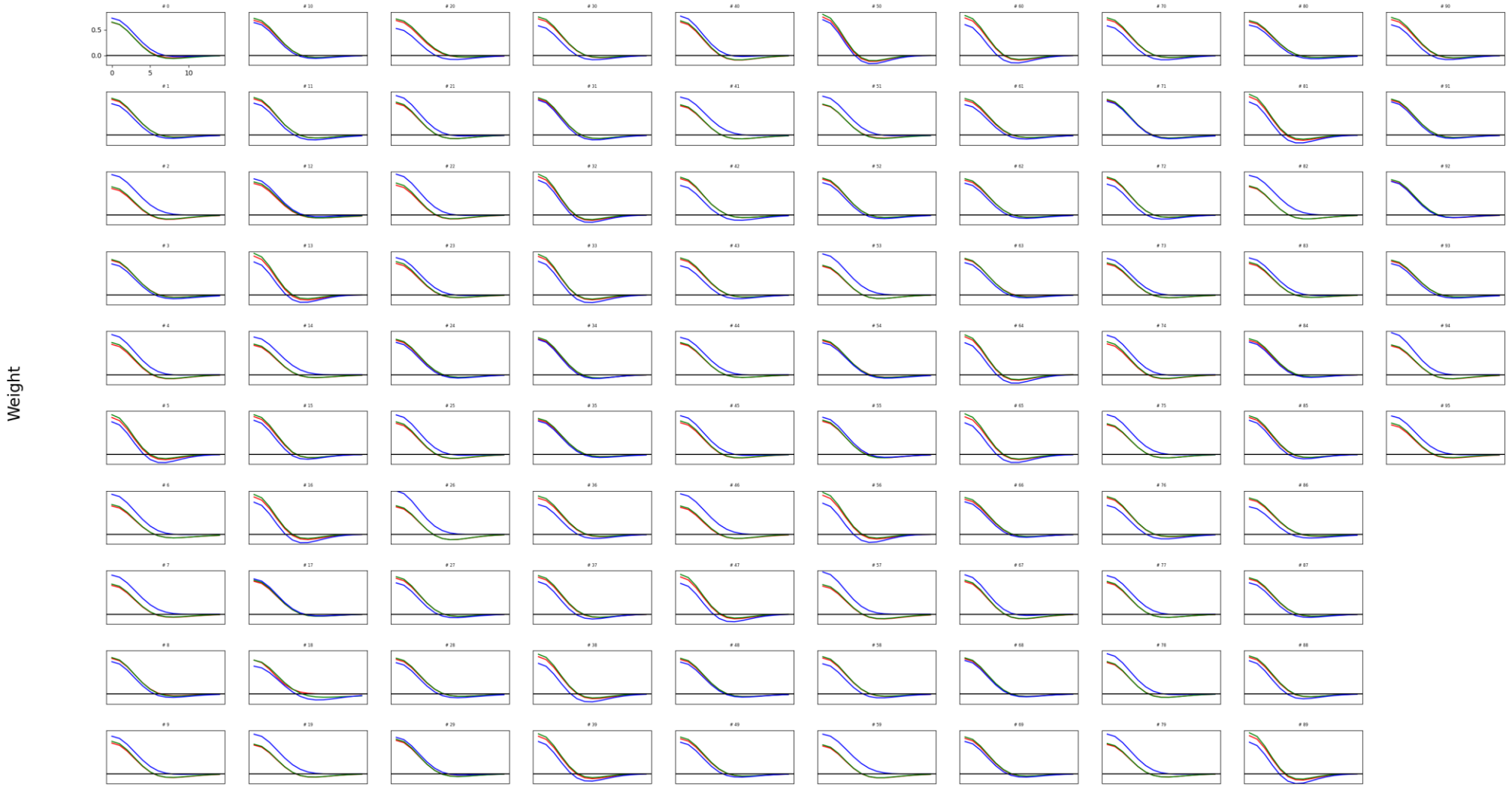
Sixth type (OFF parvo)



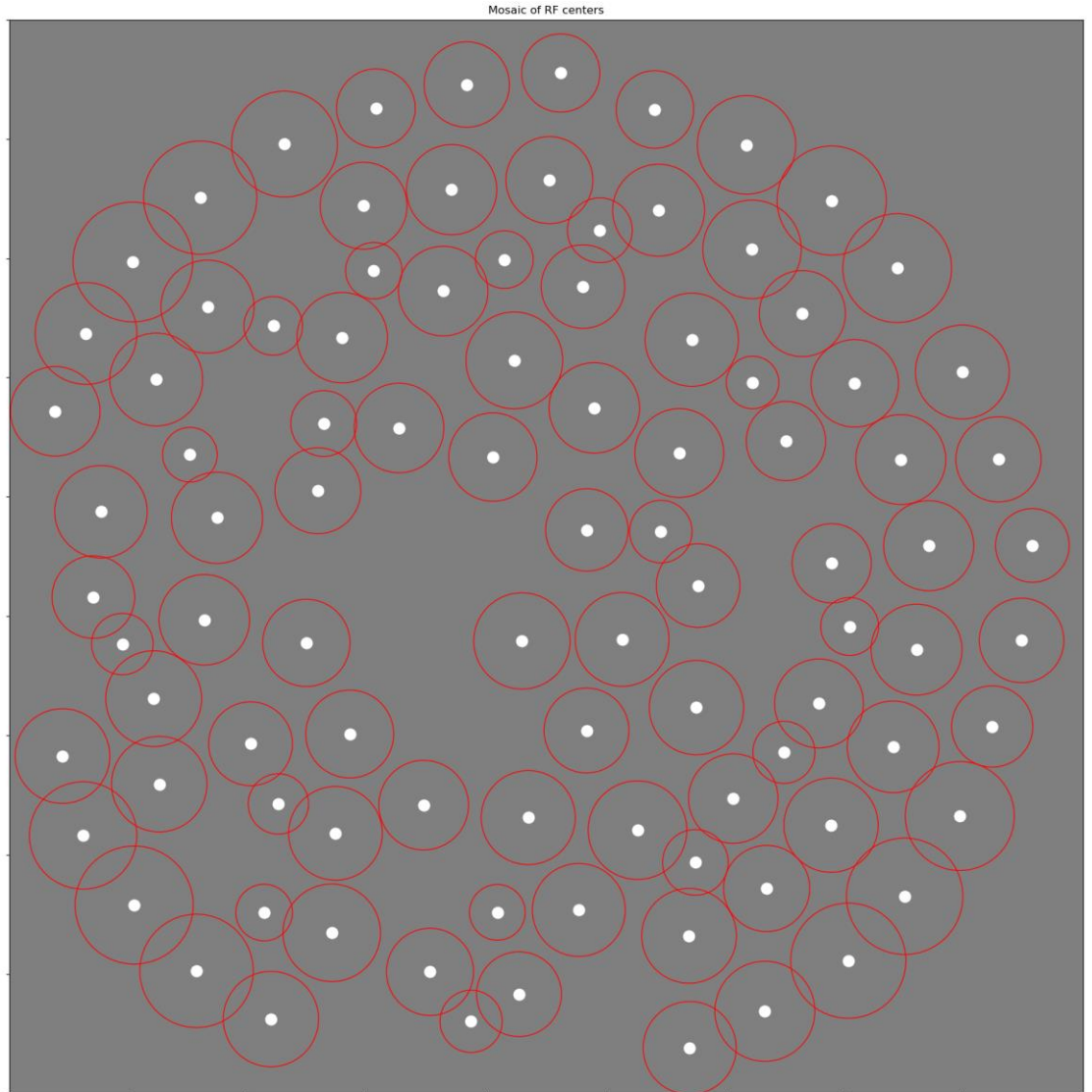
Sixth type (OFF parvo)



Seventh type (ON magno)



Seventh type (ON magno)

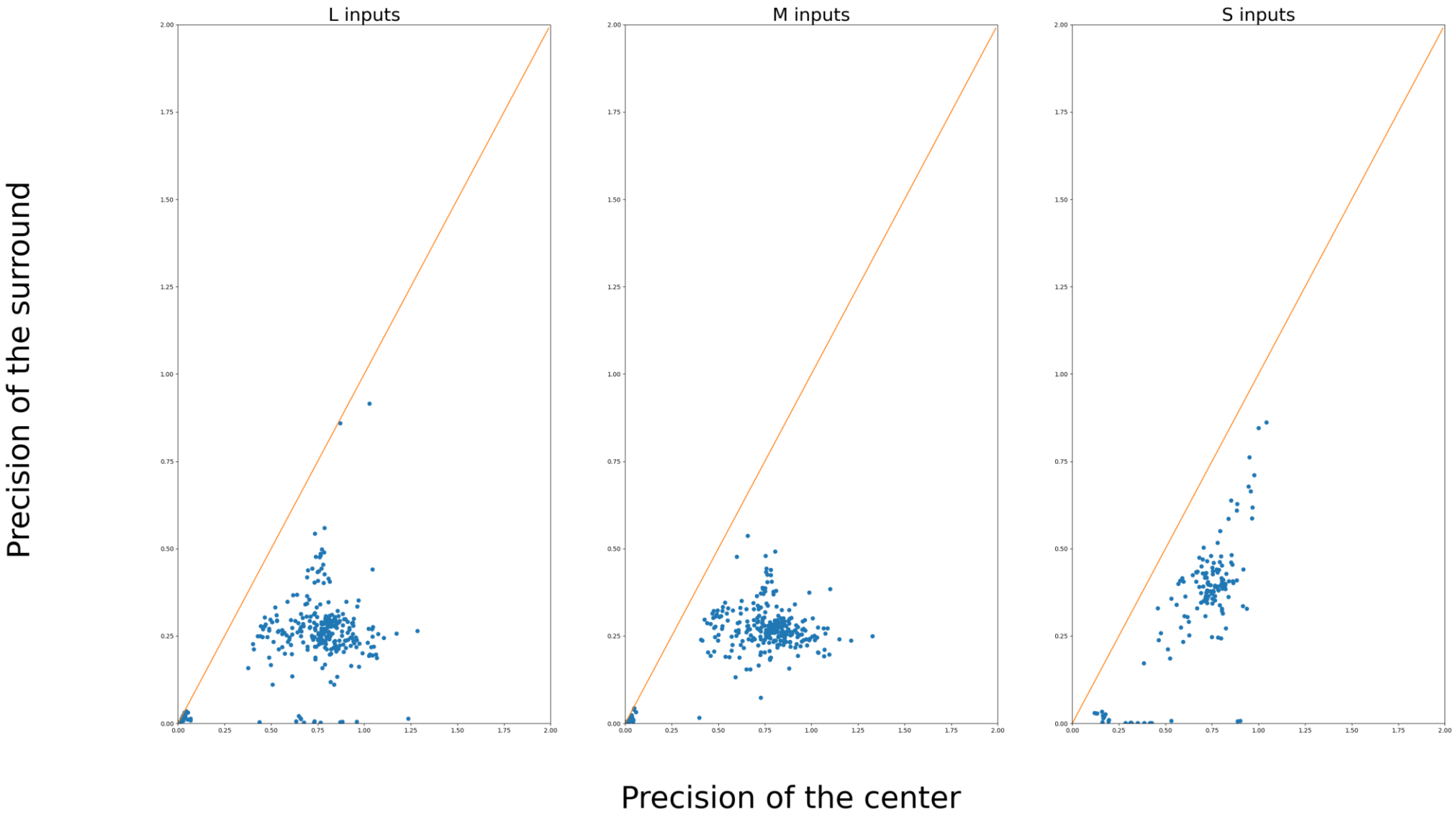


Suggestion

Hypothesis: -S neurons and Trash can neurons contribute less information than the average neuron

MI metric: Remove a single cell from sample and see how much MI decreases.

Surround is smaller in S inputs



Reducing Learning rate over epochs

Comments

1. I've previously tried the ReduceLROnPlateau function from pytorch

2. I've been struggling to make it behave the way I want it to. It tends to reduce LR too often, which can artificially create a plateau.

e.g. Last week, I presented unparametrized RFs that didn't change much from 1m to 3m epochs. That's likely because the LR was reduced way too often.

3. I've implemented my own version of "ReduceLROnPlateau", but I ran into the same problems that the pytorch version did. Likely because there is still a plateau after we lower LR.

3. I now reduce LR manually, which allows for consistent behavior and controlled comparisons.

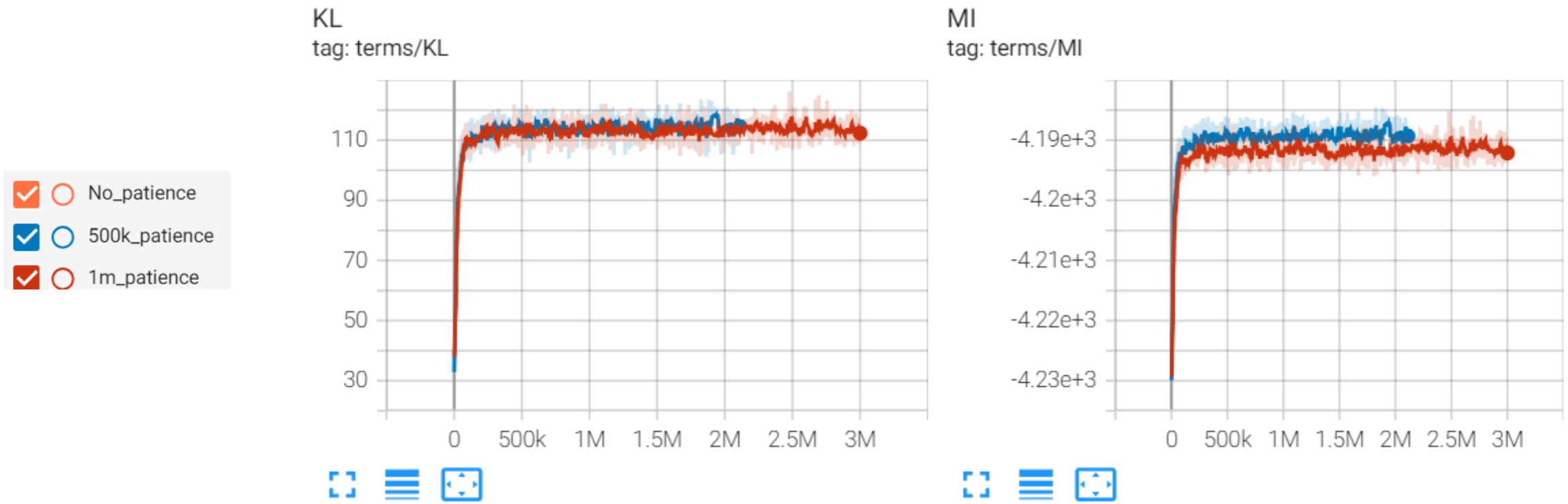
Three experiments

1. LR stays the same (control)
2. LR reduces by 50% every 500k epochs
3. LR reduces by 50% every 1m epochs

Details:

300 neurons, 18x18, DoG parametrization, colors.

Go to tensorboard for updated version



Next week

- Look more closely at unparametrized mosaics, with DoG fits
- Fix Tensorboard bug with the covariance matrices
- Exam and lab meeting presentation