Exploring the paramters

June 11, 2020

1 Problem

There are some extremely large parameters. In this documents, we need to explore them and understand why they are so large

2 Models

$I(x,y) \overset{\text{Pyr(x,y)}}{\rightarrow} C(x,y,\theta,\phi) \overset{\sum_\phi\{\}^2}{\rightarrow} E(x,y,\theta) \overset{\text{normalization}}{\rightarrow} s \overset{g\{\}^n}{\rightarrow}$ BOLD Pred

The following models differ from each other at the normalization step:

- contrast:

  $s = \sum_{x,y,\theta} E(x,y,\theta)$

- normVar:

  $E_{\text{tot}}(\theta) = \sum_{x,y} E(x,y,\theta)$, $s = \sum_\theta \frac{E^2_{\text{tot}}(\theta)}{1+w^2 * Var(E_{\text{tot}}(\theta))}$

- soc:

  $E_{xy}(x,y) = \sum_\theta E(x,y,\theta)$ $\bar{E}_{xy} = \frac{1}{nx \times ny}\sum_{x,y} E_{xy}(x,y)$ $s = \sum_{x,y}(E_{xy}(x,y) - c * \bar{E}_{xy})^2$

- ori surround:

  $s = \sum_{x,y,\theta} \frac{E(x,y,\theta)}{1+w*\sum_{x',y',\theta'} F(x-x',y-y',\theta-\theta')\cdot E(x,y,\theta)}$

3 How to overfit at the target stimuli set

One way that help us understand the problem is to see the both curve as function of level of sparity denoted as v.

The curve of patterns as: $g * P(x)^n$, where $P(x) = s$

The curve of gratings as: $g * G(x)^n$, where $G(x) = s$

What we need now is to do curve fitting using s as input.

since $g > 0$, $n > 0$, the function is monotonically increasing.

if $n > 1$, the 2nd order derivative$ > 0$

if $n < 1$, the 2nd order derivative$ < 0$

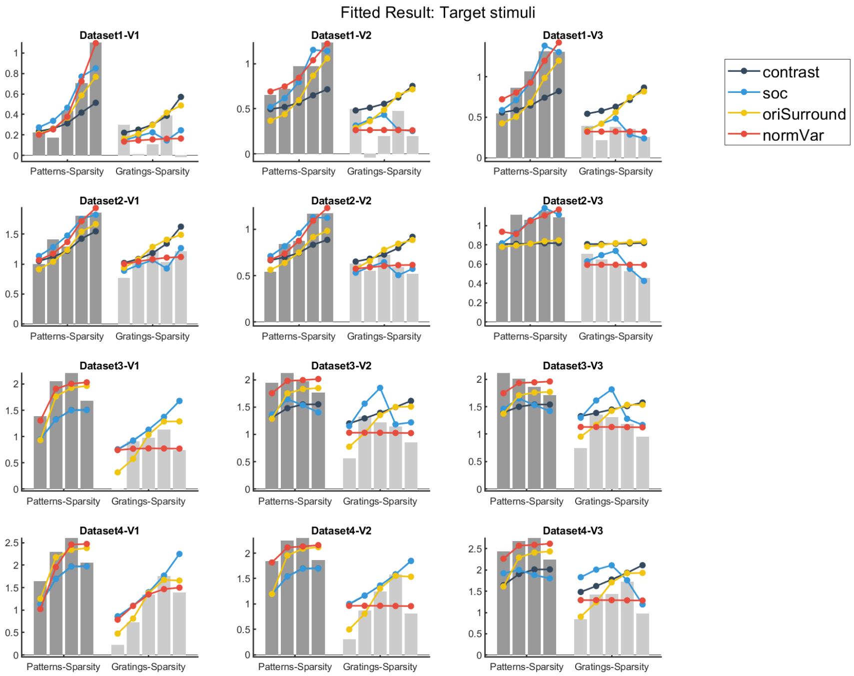
4 Conditions for overfitting the target stimuli

If s statisfy the following conditions.

- all s of patten stimuli is larger than gratings stimuli
- the curve along with the sparsity in partterns is steeper than that of gratings

Meanwhile, we can freely choose params g and n. We can always get very good fit of the target stimuli.

5 Dive into the unreasonable paramters



###

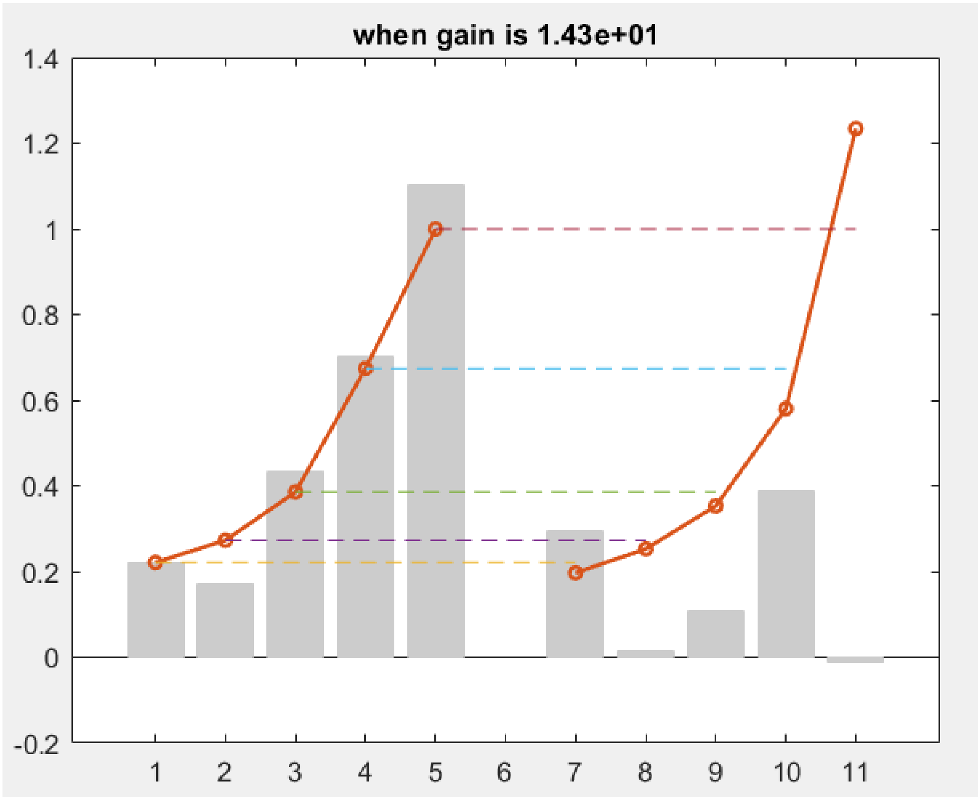
Dataset1, V1, SOC

Parameters

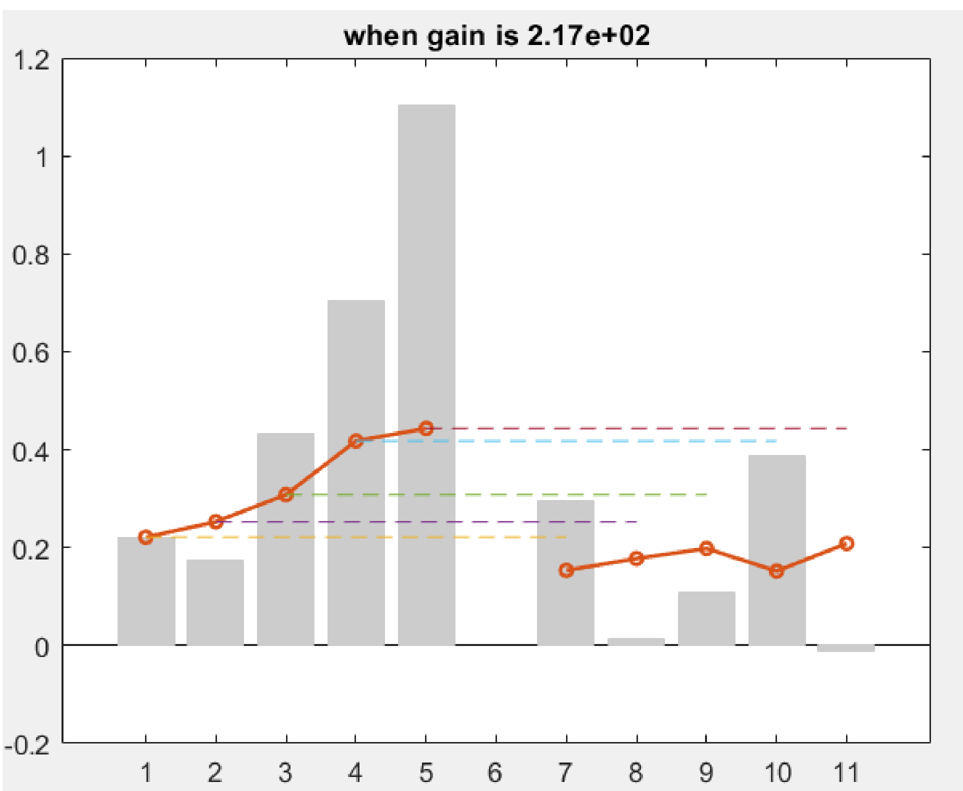
- c: 1.119
- g: 2.963e+06
- n: 2.4126

Because g is a linear parameter which will not less likely to affect the trend of the curve, so we treat it as a flexible paramters.

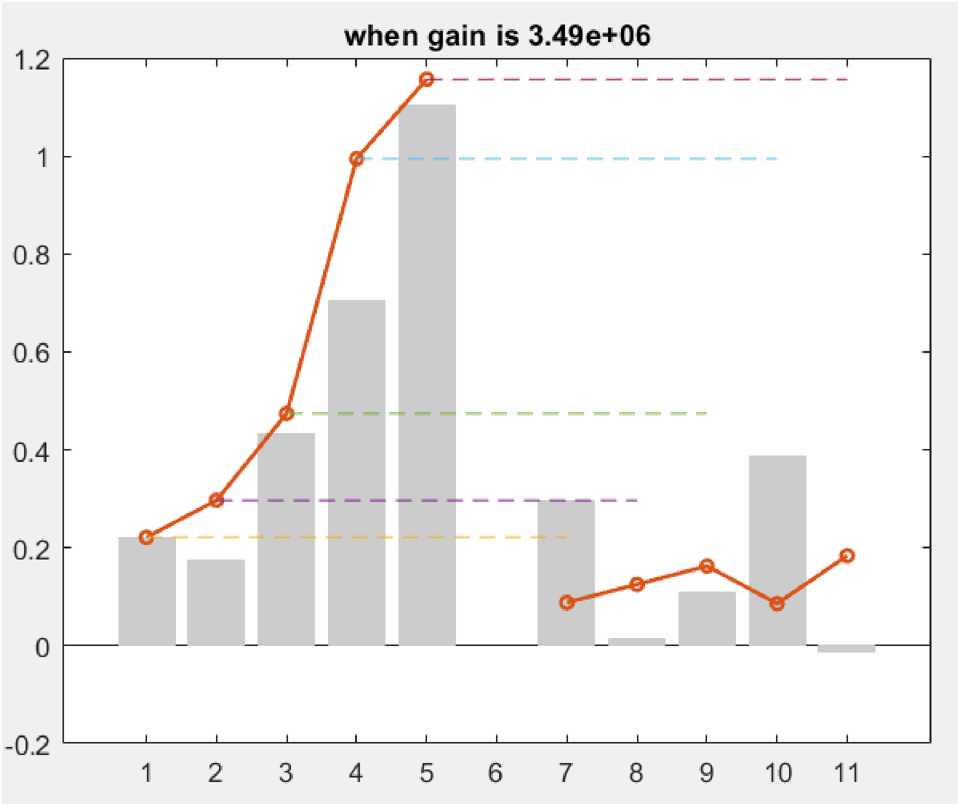
Check: mean(E_xy, 1:4) # dim: 1 x stim



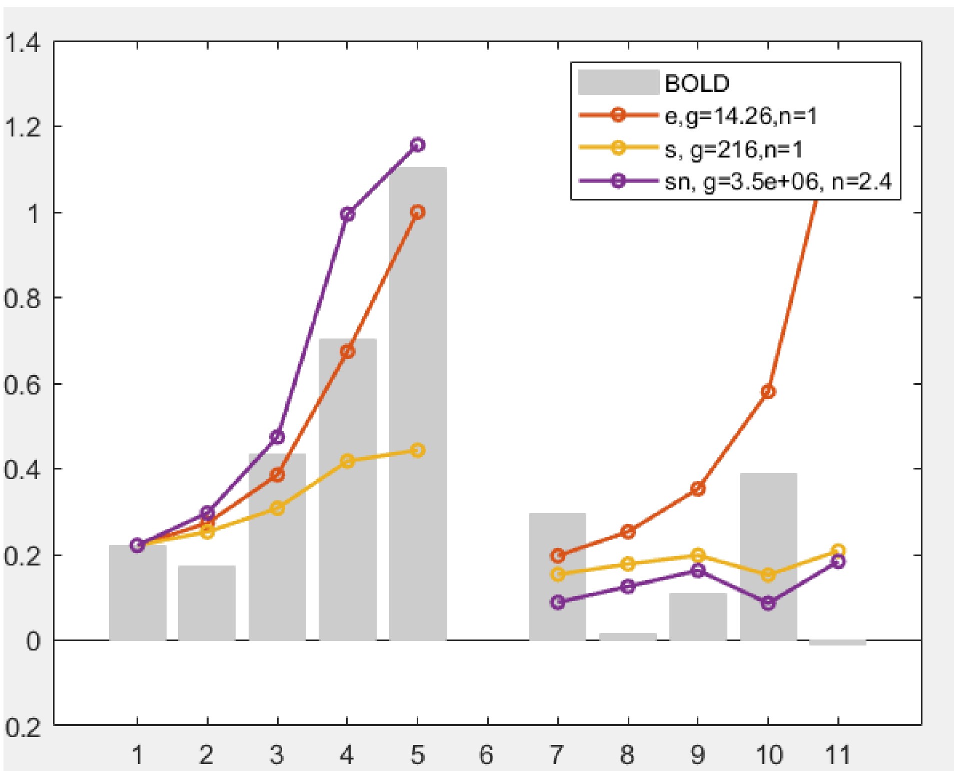
After soc: (E - c * E_mean).^2



After nonlinearity: s .^ n



Group and show:



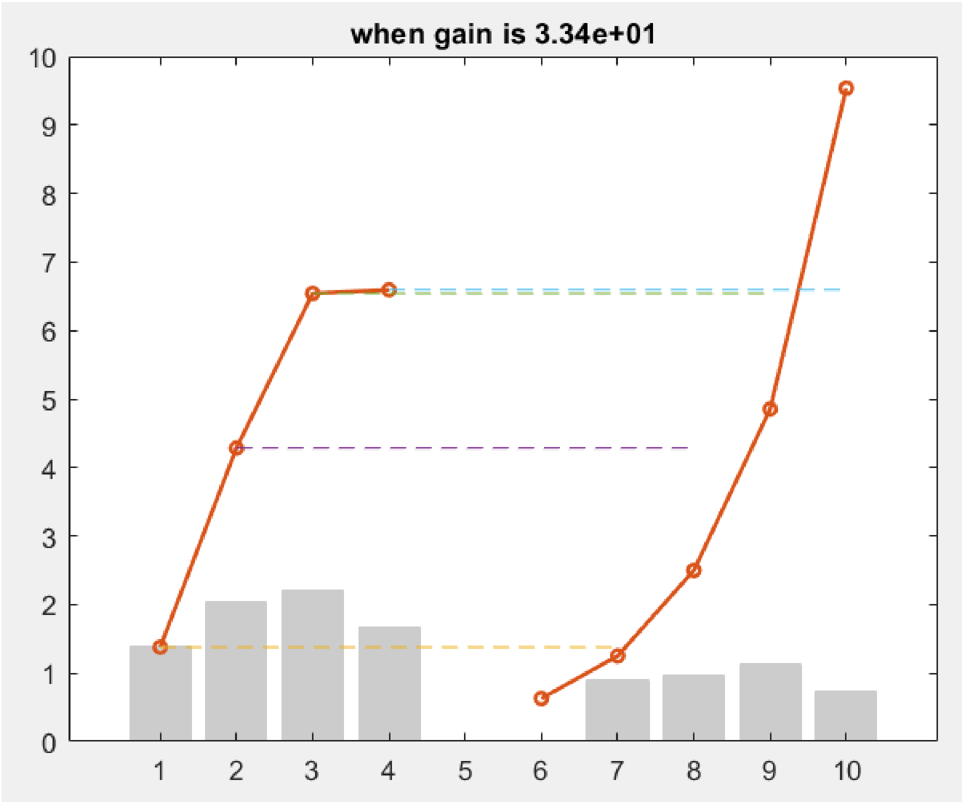
	Dataset 1			Dataset 2			Dataset 3			Dataset 4		
	V1	V2	V3	V1	V2	V3	V1	V2	V3	V1	V2	V3
Contrast												
g	68.4	1.289	1.641	3.083	1.469	0.839	2.46	1.856	1.710	3.56	2.482	2.498
n	1.54	0.242	0.253	0.253	0.186	0.008	0.294	0.105	0.063	0.355	0.227	0.130
SOC												
c	1.119	1.219	1.941	2.033	1.123	1.237	1.484 e+05	1.356	1.409	1.708 e+06	1.77e 06	1.640
g	2.96e +06	6156	1571 0	121.5	163.9	109.6	0.079 5	26.15	11.98	0.024	0.099	4.786
n	2.41	1.360	1.478	0.680	0.790	0.710	0.147	0.561	0.399	0.177	0.113	0.172
Ori Surround												
w	45.6	103.8	101.1	82.67	100.4	82.70	75.61	67.33	77.11	98.32	107.9	67.78
g	64.7	99.98	99.99	25.15	13.88	1.228	99.91	11.38	6.782	100	66.68	19.51
n	0.998	0.890	0.870	0.556	0.524	0.076	0.855	0.405	0.291	0.772	0.700	0.463
NormVar												
w	102.8	861.7	433.4	81.86	112.5	633	114.9	5.e05	2.4e0 6	16.82	867.1 2	1.3e0 6
g	66.68	85.95	51.20	7.270	5.611	6.869	8.929	271.2	88.15	6.882	17.03	274.2
n	0.678	0.446	0.437	0.226	0.248	0.198	0.274	0.184	0.151	0.296	0.222	0.194

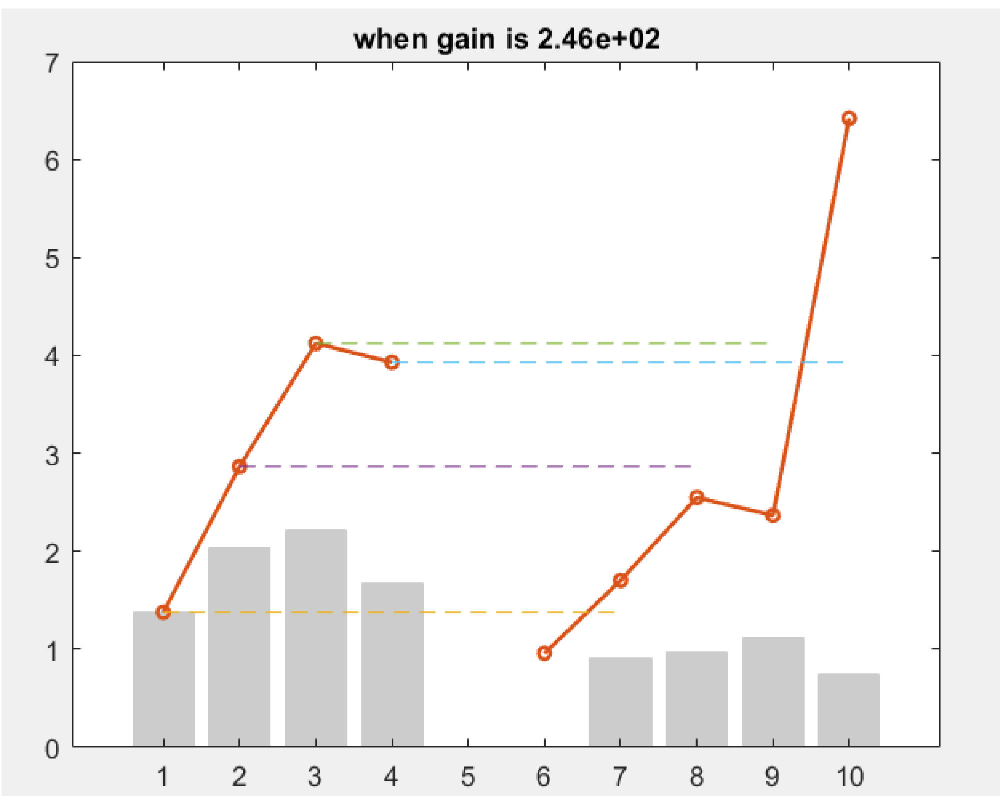
5.0.1 Dataset3, V1, SOC

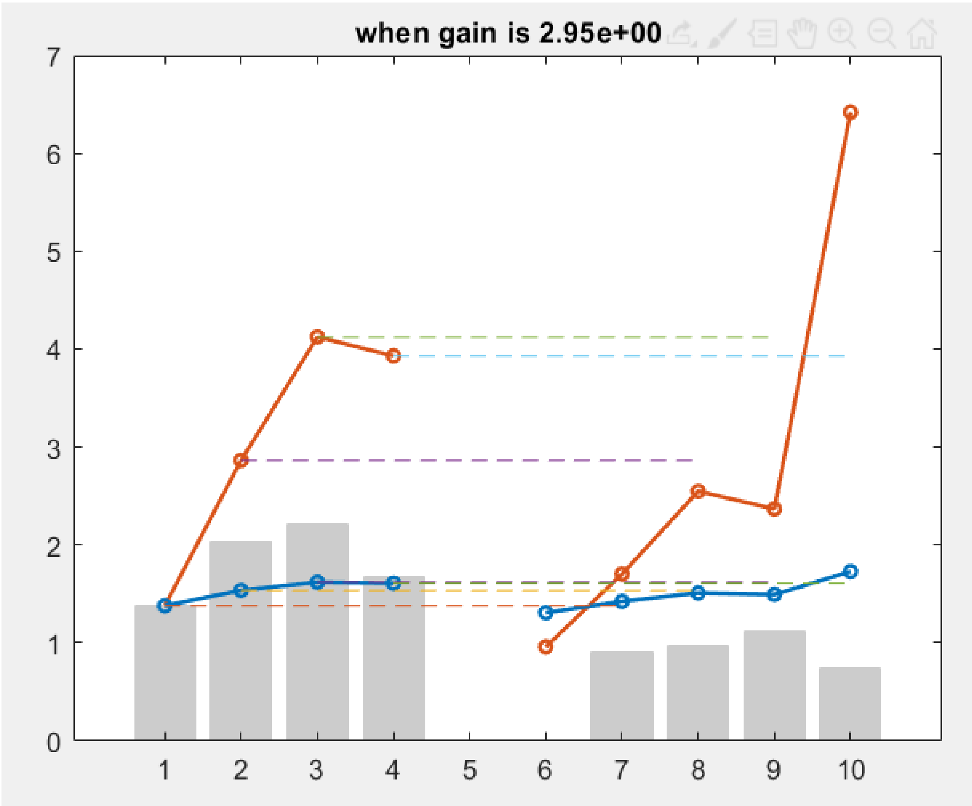
Parameters

Why soc can get good prediction in dataset1 and 2 not dataset3,4.

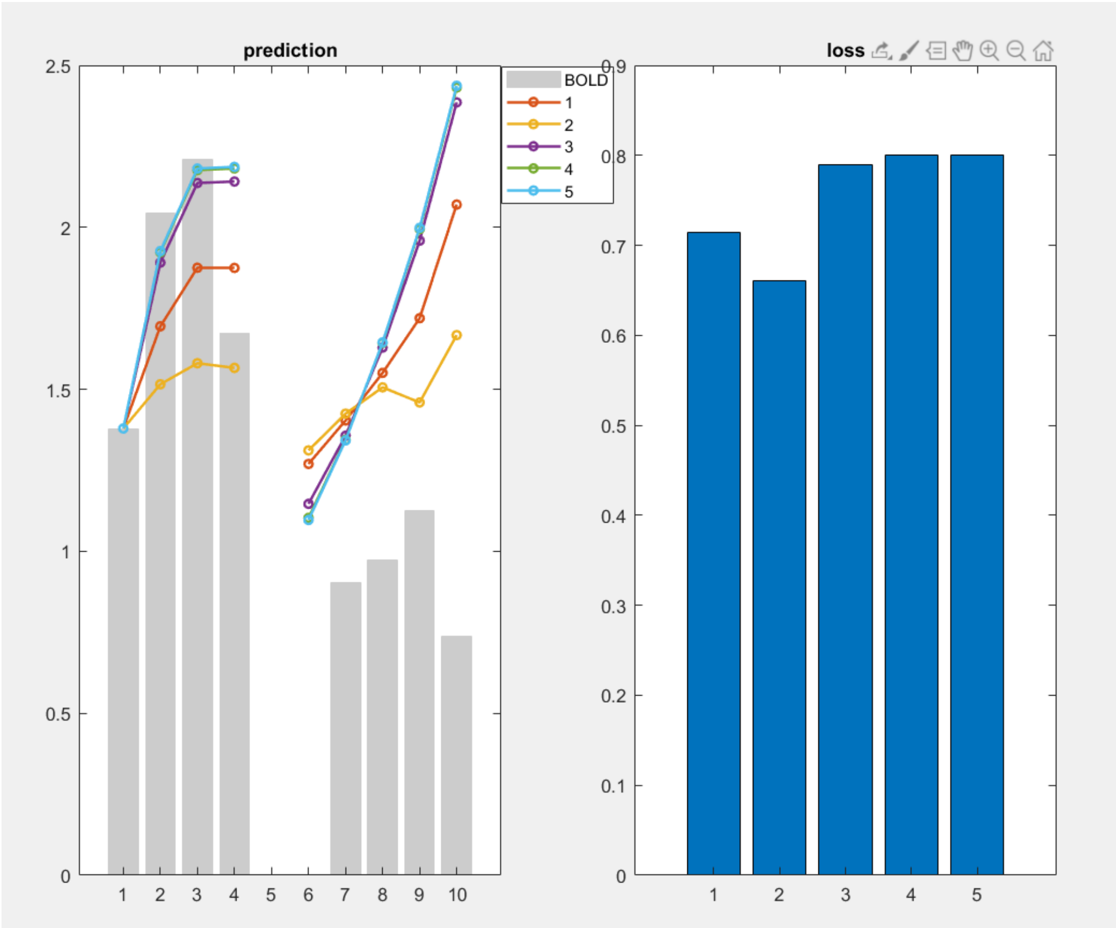
- c: 1.4848e+05
- g: 0.079
- n: 0.1468


when gain is 3.34e+01
10
9
8
7
6
5
4
3
2
1
0
1 2 3 4 5 6 7 8 9 10

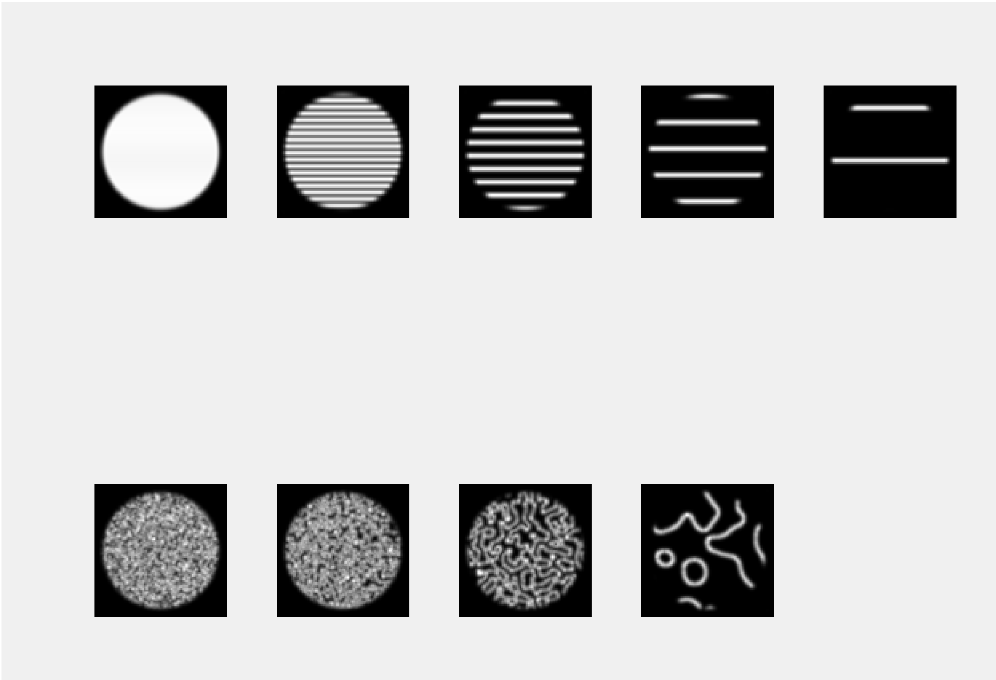

when gain is 2.46e+02
7
6
5
4
3
2
1
0
1 2 3 4 5 6 7 8 9 10



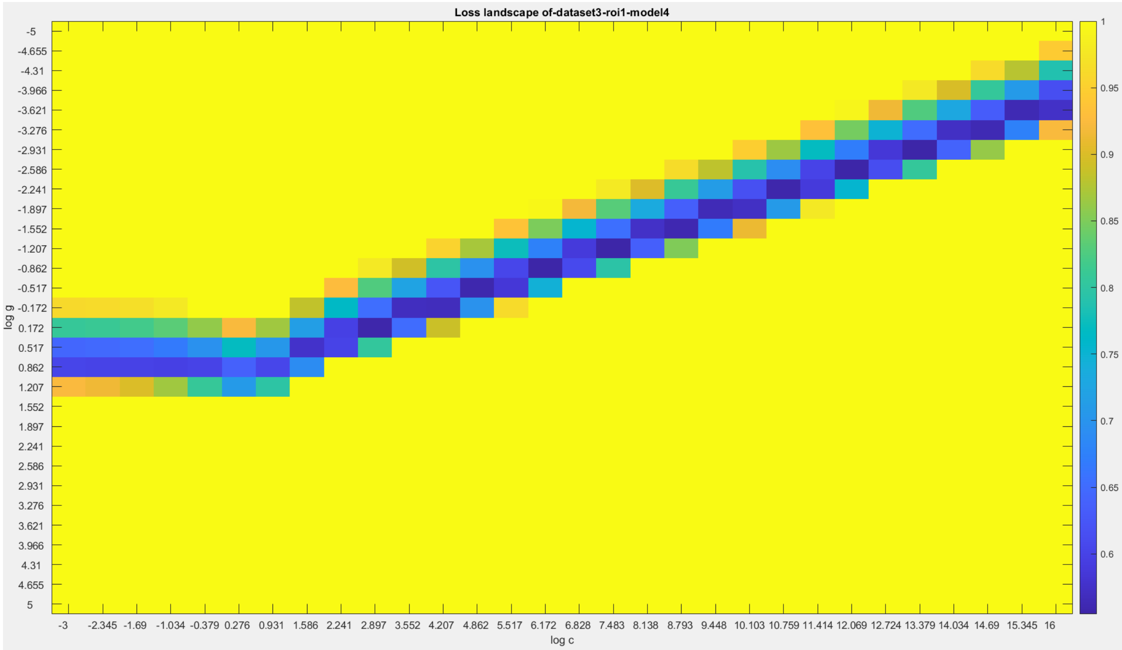
Why c is so large?????



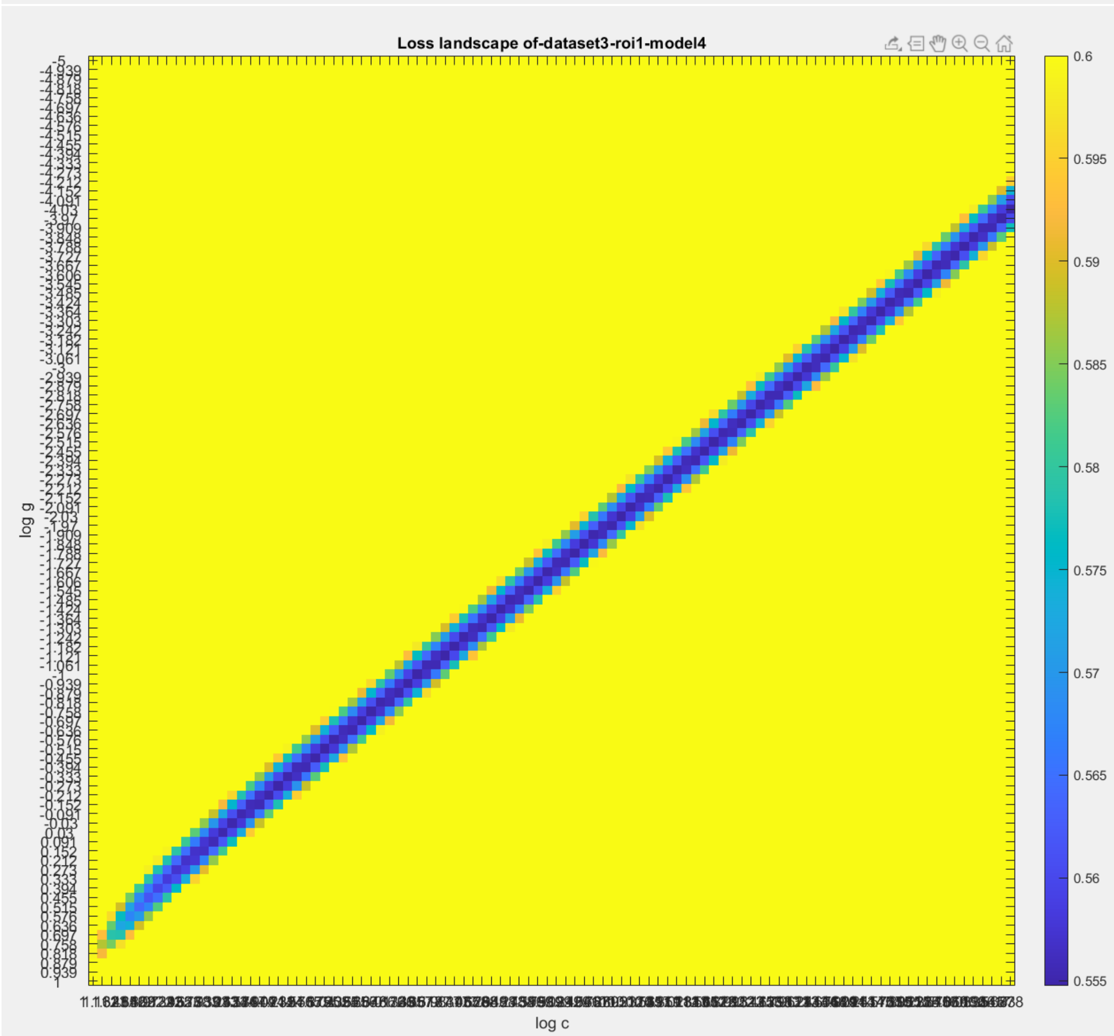
The 4 th curve * c: 20.085 * g: 23.0994 * n: 0.1468



What's the problem with the extreme value c and g somehow form a Plain. Let's have look at the landscape



Loss landscape of-dataset3-roi1-model4
log g
log c



Loss landscape of-dataset3-roi1-model4
log g
log c

We may notice that in Dataset3,4 v1, and Dataset4 v2, the prediction of of soc is overlap with contrast model and, look at the paramters of these three. we may realize why the paramter is so large.

Quick conclusion The condition of SOC to get a good fit:

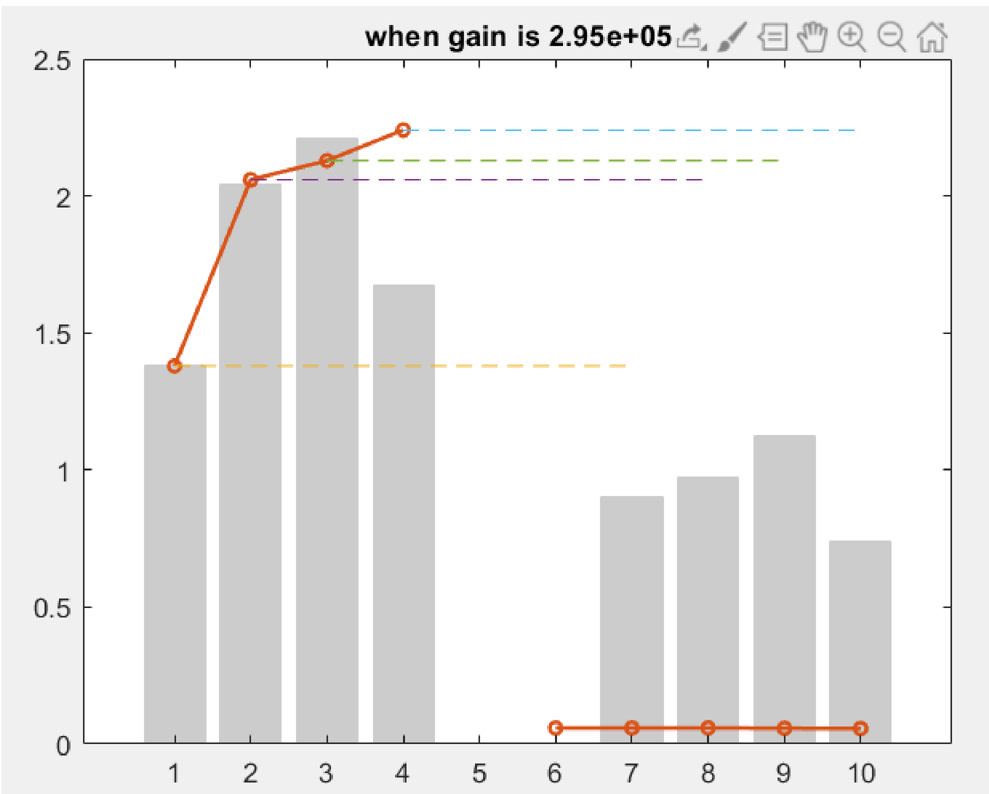
- all s of patten stimuli is larger than gratings stimuli
- the curve of s along with the sparsity in partterns is steeper than that of gratings
- paramter n > 1, making the value ver much and thus a large gain.

However, the the stimuli-contrast group can be extremely sensitive to the nonlinearity and require < 1. This might be the reason why SOC does not perform well in the full dataset.

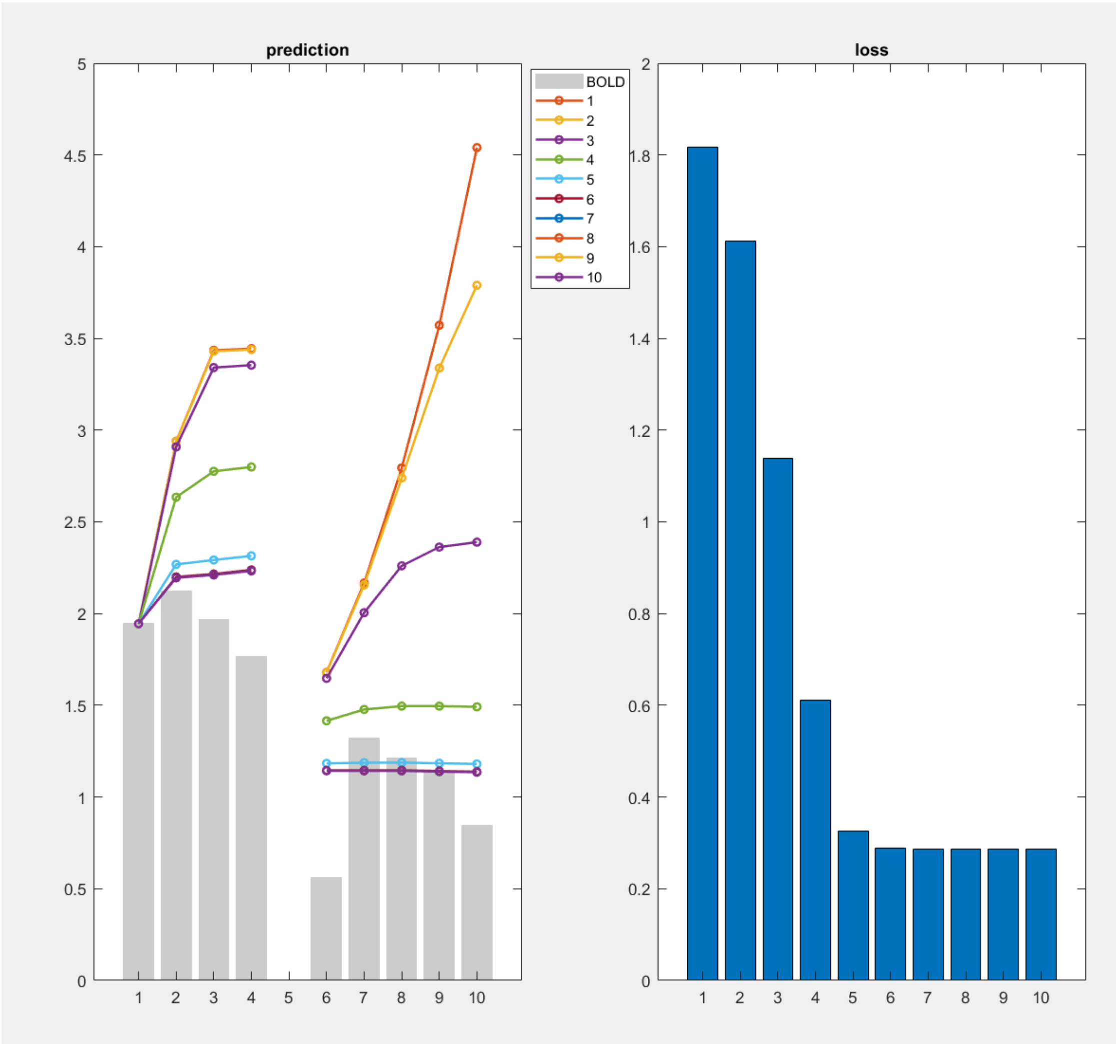
5.0.2 normVAR, dataset3,4 V2, V3

The extreme value of normVar model can be very similar to what happens in the

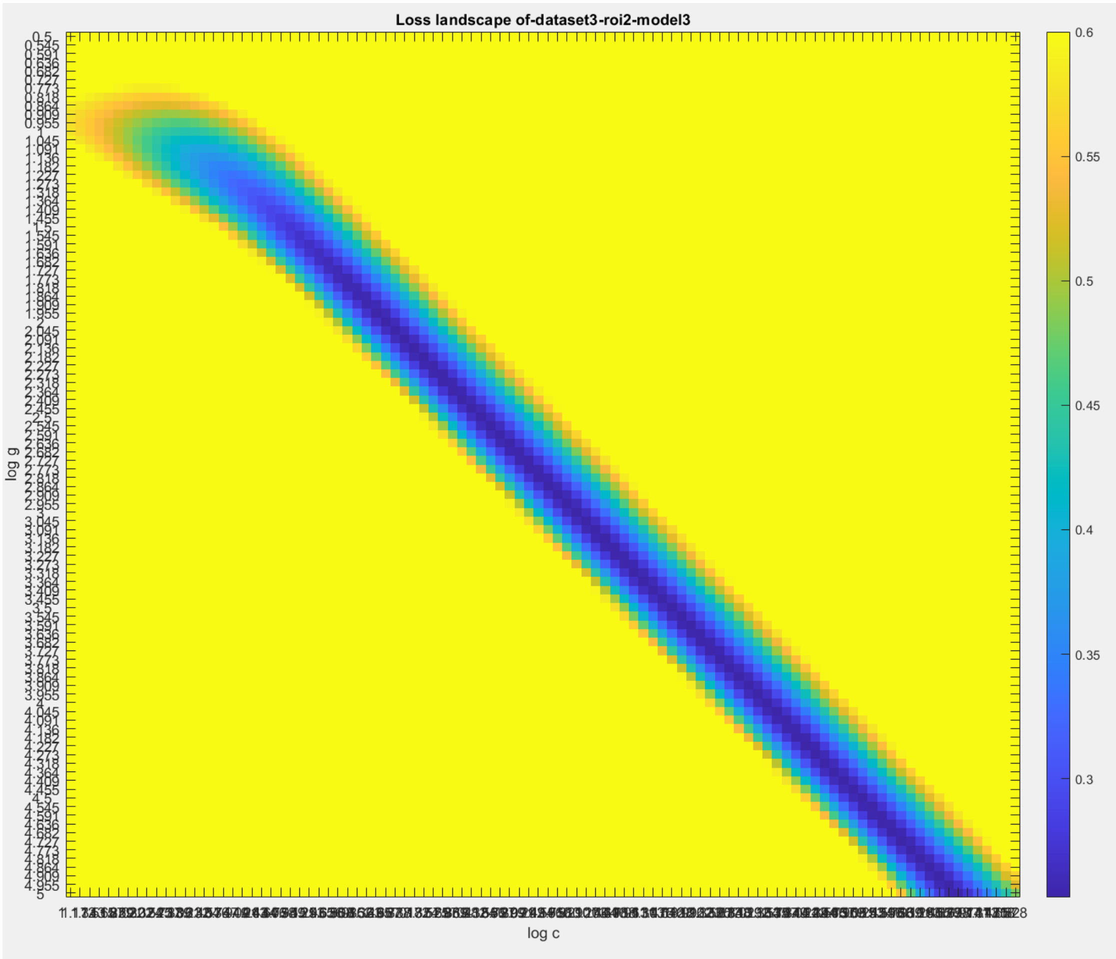
Before that let's look at the s of normVar:



And let's have a look at what role the parameter w play in this normVar model.



Let's look at the landscape

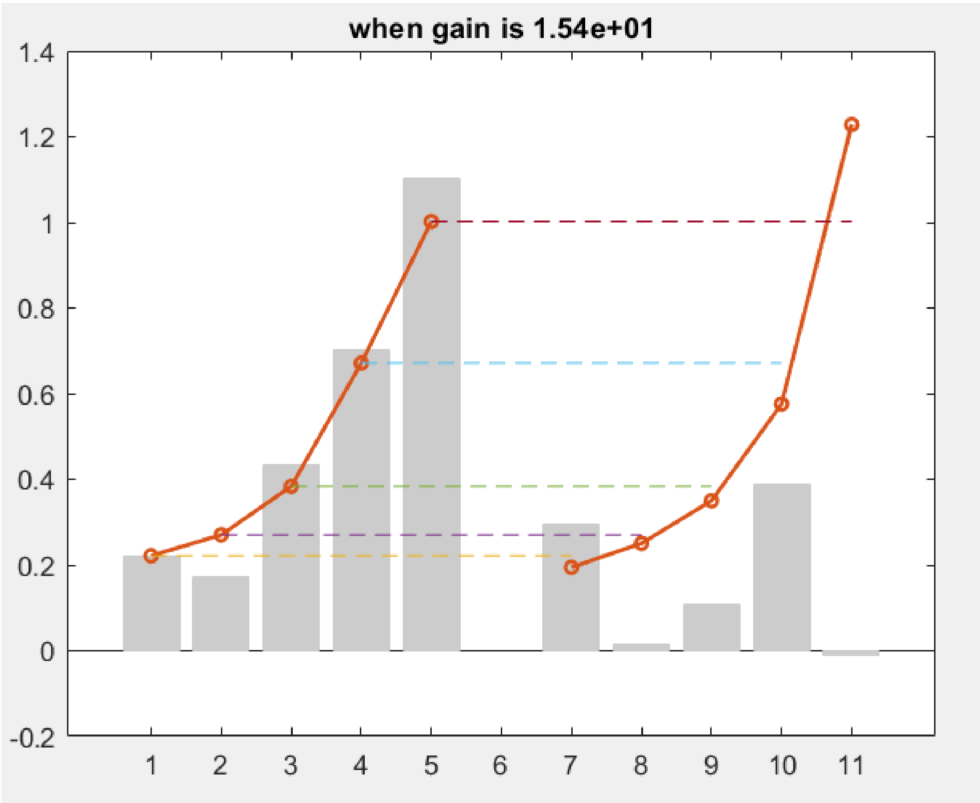


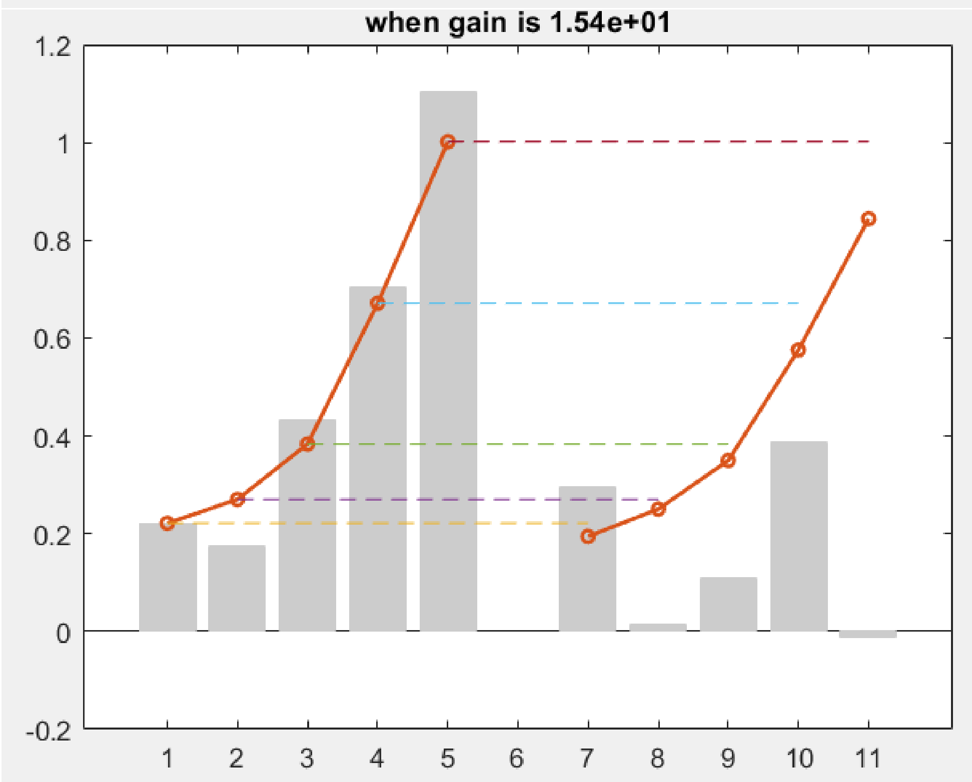
6 Some thoughts

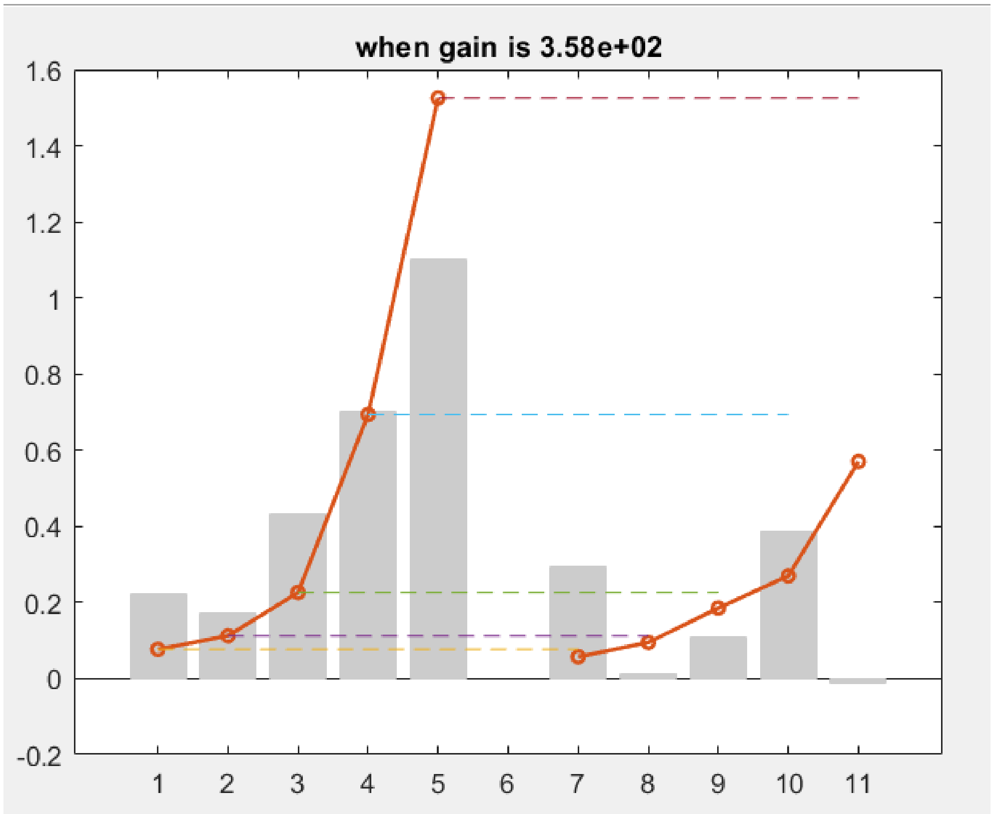
The extreme value happens when one of the components in the function take the dominant position and the parameter lose its ability to balance different components.

One solution: Add contrast stimuli set to the target stimuli to take control of nonlinearity.

7 Appendix


when gain is 1.54e+01
1.4
1.2
1
0.8
0.6
0.4
0.2
0
-0.2
1
2
3
4
5
6
7
8
9
10
11


when gain is 1.54e+01
1.2
1
0.8
0.6
0.4
0.2
0
-0.2
1
2
3
4
5
6
7
8
9
10
11



8 Codes

- contrast

```
% the core model function
function y_hat = forward(model, E_ori, param )

    % input: E_ori: ori x exp x stim

    % set param
    g = exp(param(1));
    n = exp(param(2));

    % d: ori x exp x stim
    d = E_ori;

    % sum over orientation, s: exp x stim
    s = mean(d, 1);

    % add gain and nonlinearity, yi_hat: exp x stim
    yi_hat = g .* s .^ n;

    % Sum over different examples, y_hat: stim
    y_hat = squeeze(mean(yi_hat, 2))';
```

```
end
```

- normVar

```
function y_hat = forward( model, E_ori, param )

    % input: E_ori: ori x exp x stim

    w = exp(param(1));
    g = exp(param(2));
    n = exp(param(3));

    % d: ori x exp x stim
    d = E_ori.^2 ./ (1 + w^2 .* var(E_ori, 1) );

    % sum over orientation, s: exp x stim
    s = mean(d, 1);

    % add gain and nonlinearity, yi_hat: exp x stim
    yi_hat = g .* s .^ n;

    % Sum over different examples, y_hat: stim
    y_hat = squeeze(mean(yi_hat, 2))';

end
```

- soc

```
function y_hat = forward(model, E_xy, param )

    # input: E_xy: x x y x ori x exp x stim

    # choose import receptive field
    if model.receptive_weight ==false
        height = size(E, 1) ;
        model = model.disk_weight(model, height);
    end

    c = exp(param(1));
    g = exp(param(2));
    n = exp(param(3));

    % x x y x ori x exp x stim --> x x y x exp x stim
    E = squeeze( mean( E, 3));

    % d: x x y x exp x stim
    E_mean = mean( mean(E, 1), 2);
    v = (E - c * E_mean).^2;
    d = bsxfun(@times, v, model.receptive_weight);
```

```
    % Sum over spatial position
    s = squeeze(mean(mean( d , 1) , 2)); % ep x stimuli

    % add gain and nonlinearity, yi_hat: exp x stim
    yi_hat = g .* s .^ n;

    % Sum over different examples, y_hat: stim
    y_hat = squeeze(mean(yi_hat, 1));

end
```

- oriSurround

```
function y_hat = forward(model, E_xy, weight_E, param )

    # input: E_xy: x x y x ori x exp x stim

    if model.receptive_weight ==false
        height = size(E, 1) ;
        model = model.disk_weight(model, height);
    end

    w = exp(param(1));
    g = exp(param(2));
    n = exp(param(3));

    % x x y x ori x exp x stim --> x x y x exp x stim
    d_theta = E_xy ./ ( 1 + w * weight_E);

    % x x y x ori x exp x stim --> x x y x exp x stim
    v = squeeze( mean( d_theta, 3));
    d = bsxfun(@times, v, model.receptive_weight);

    % Sum over spatial position ---> exp x stim
    s = squeeze(mean(mean( d , 1) , 2)); %

    % add gain and nonlinearity, yi_hat: exp x stim
    yi_hat = g .* s .^ n;

    % Sum over different examples, y_hat: stim
    y_hat = squeeze(mean(yi_hat, 1));

end
```

[ ]: