Numeric (GD) logic:

$$x_{in} = [x_0, x_1, x_2, \ldots, x_n]$$

(I) for each point x_i we need diff $= [0.99, 1, 1.01]$

So the following vec is created:

$$\text{dimVec} = [1, 2, 3, 4, \ldots, 3 \cdot n]$$

(II) Then we create all combinations:

$$_nC_k = \binom{n}{k} = \frac{n!}{(n-k)!\,k!}$$

in our case we choose n components from the $3 \cdot n$ vec.

(III) the result is a matrix of size

$$\left[\frac{(3 \cdot n)!}{(2 \cdot n)!\,n!}, n\right]$$

This matrix we filter each column j so that all rows in that j should be between $[(3 \cdot j - 2) : 3j]$

$$\text{Mat} = \left|\begin{array}{c|c|c|c|c} j=1 & j=2 & j=3 & \ldots & j=n \\ \hline 1:3 & 4:6 & 7:9 & & (3n-2):3n \end{array}\right|$$

(IV) Then the possible directions of the gradient are: $\text{grad} = x_i \cdot \text{grad}[\text{Mat}]$

The optimization itself

problems:

I) The idea is that the above calculation is expensive and we want to avoid it.

II) The converge to the local minima we need to adjust the grad amplitude.

Solutions: (here we denote x as the solution)

I) After computing the grad, we check in which of the gradients the change in f is minimal:

$$\min\left[f(x_0+\text{grad})-f(x_0)\right]$$

and that dir is chosen as $\hat{d}$.
from here on out we move in $\hat{d}$, $x_i = x_0 + \text{Amp}\cdot\hat{d}$ until $[f(x_i+\hat{d})-f(x_i)]>0$, meaning we're moving away from the minima.
In this case we recompute the grad around x_i. This keeps the computations small.

Kinda abstract but thats the idea
d calculated only at the circles

II) if $[f(x_i+d)-f(x_i)]>0$: $\text{Amp} = \text{Amp}\cdot\frac{1}{2}$ } This allow convergence when in minima