Non convex optimisation: CMA-ES with gradient

A combination of evolution strategy and gradient information

Semester project @CV Lab
Student: Huajian Qiu
Supervisor : Benoit Guillard
Professor: Pascal Fua

Optimisation method

Stochastic search based:

CMA-ES, swarm intelligence, ...

No gradient information,

low search efficiency

Gradient based:

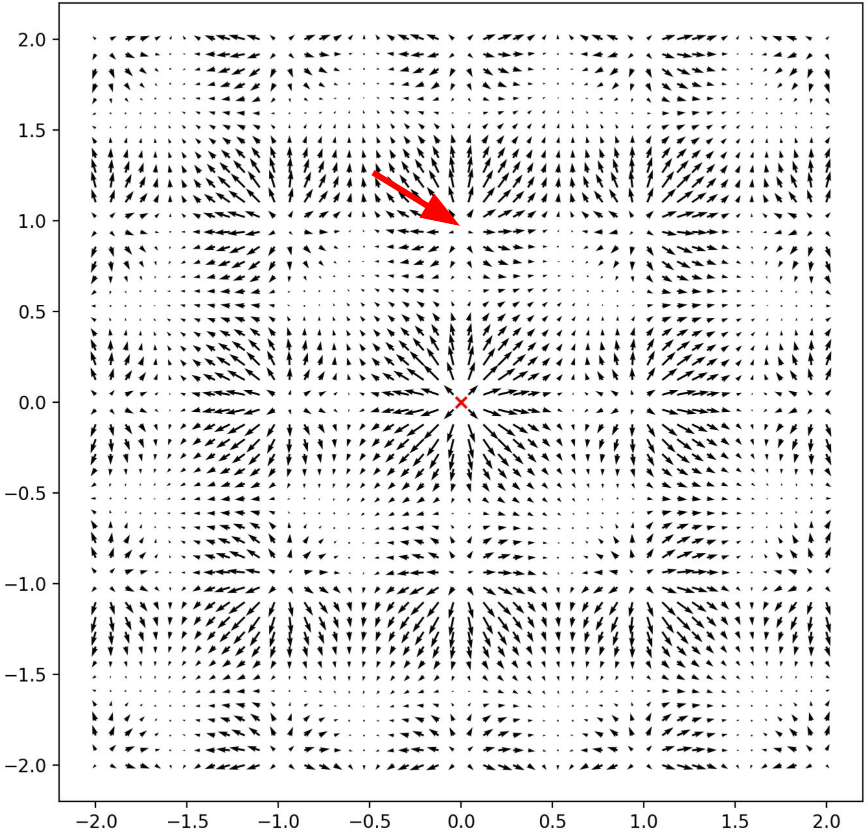
Adam, line search, SGD…

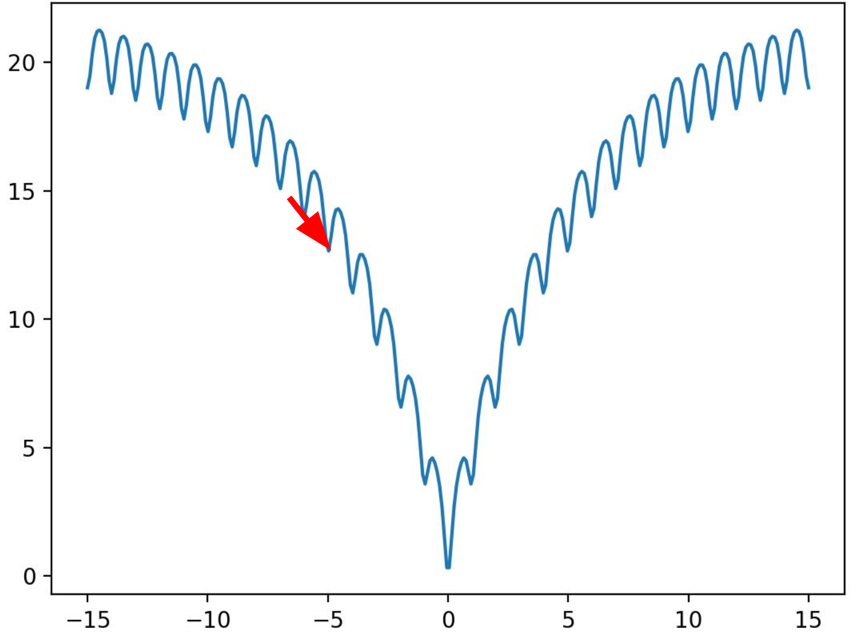
Hard to escape from local minimas

My contribution:

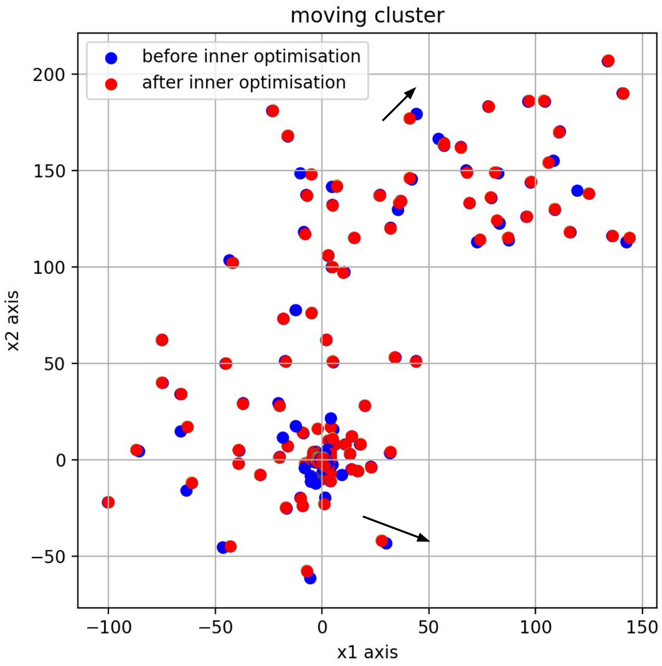
Inject gradient
based optimiser
into cma

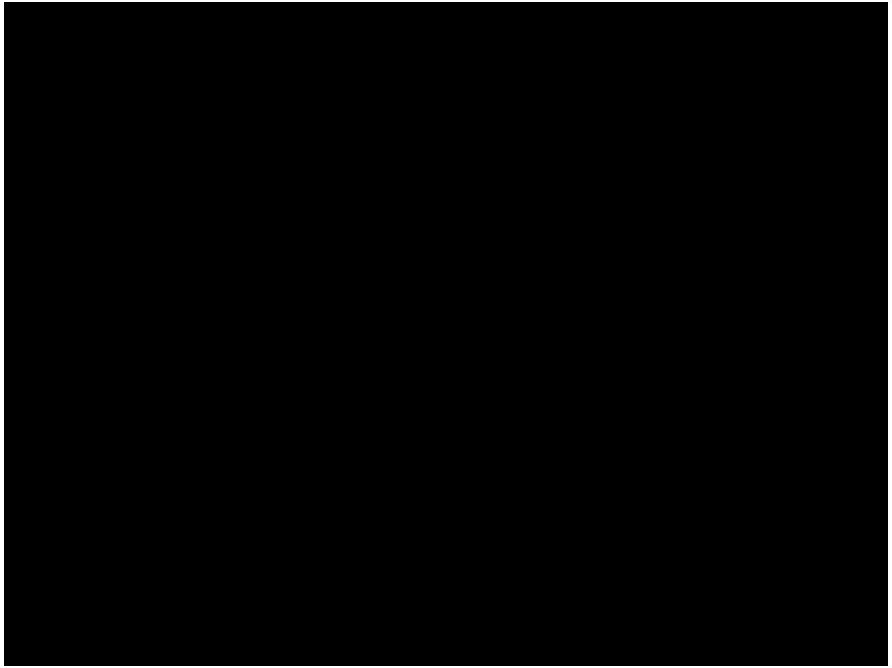
injected inner optimiser into cma





Effect of inner optimiser





State of the art

Covariance Matrix Adaptation-Evolution Strategy(CMA-ES):

- Probably the most successful evolution strategy,
- Easily Parallelizable
- Randomly sample candidates from normal distribution,
- Update mean and cov from better half part of candidates

Connection with neural network:

❖ Cma-es For Hyperparameter Optimization Of Deep Neural Networks

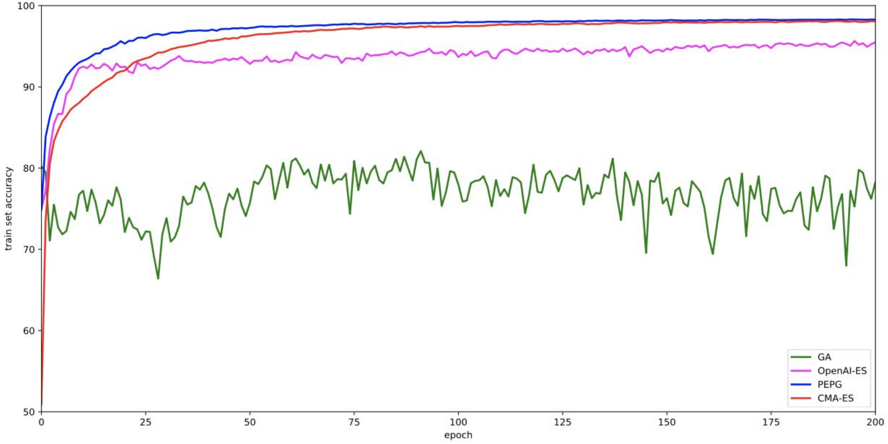
Ilya Loshchilov & Frank Hutter, Univesity of Freiburg, ICLR 2016

❖ Evolution Strategies as a Scalable Alternative to Reinforcement Learning

Tim Salimans, et. al. OpenAI, *arXiv preprint*

Neural network training application

Method	Train Set	Test Set
Adam (BackProp) Baseline	99.8	98.9
Simple GA	82.1	82.4
CMA-ES	98.4	98.1
OpenAI-ES	96.0	96.2
PEPG	98.5	98.0



Dataset: MNIST

Neural network:

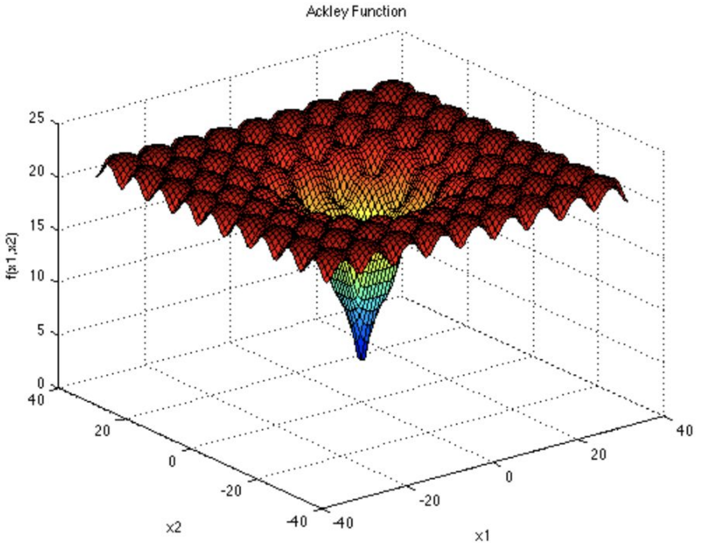
2-layer convnet , ~ 11k parameters

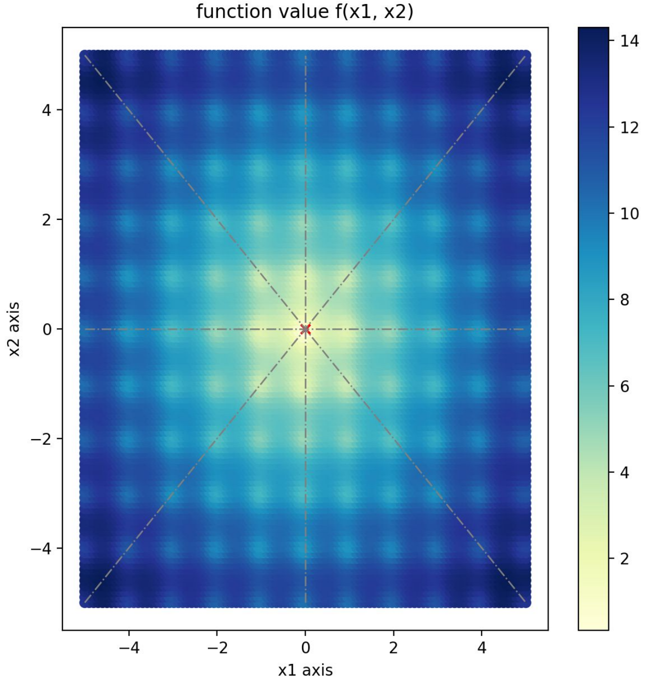
"A Visual Guide to Evolution Strategies"

Ha, David 2017

objective function (as Benchmarks)

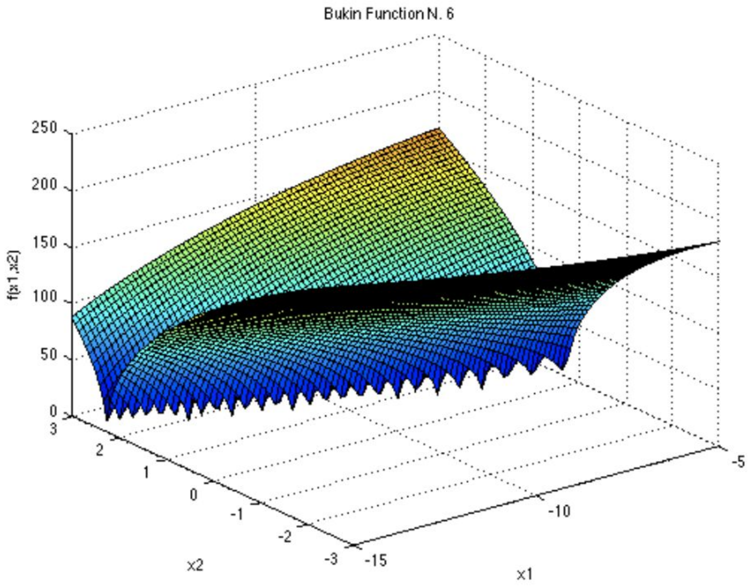
1. Ackley
2. Tuned Ackley
3. Bukin
4. Eggholder



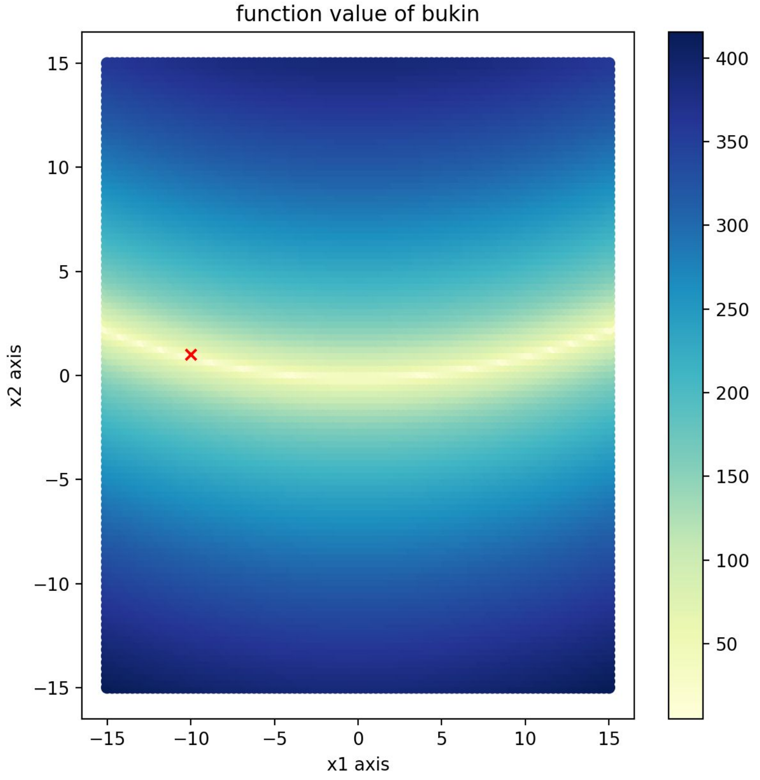


$$f(\mathbf{x}) = -a\exp\left(-b\sqrt{\frac{1}{d}\sum_{i=1}^{d}x_i^2}\right) - \exp\left(\frac{1}{d}\sum_{i=1}^{d}\cos(cx_i)\right) + a + \exp(1)$$

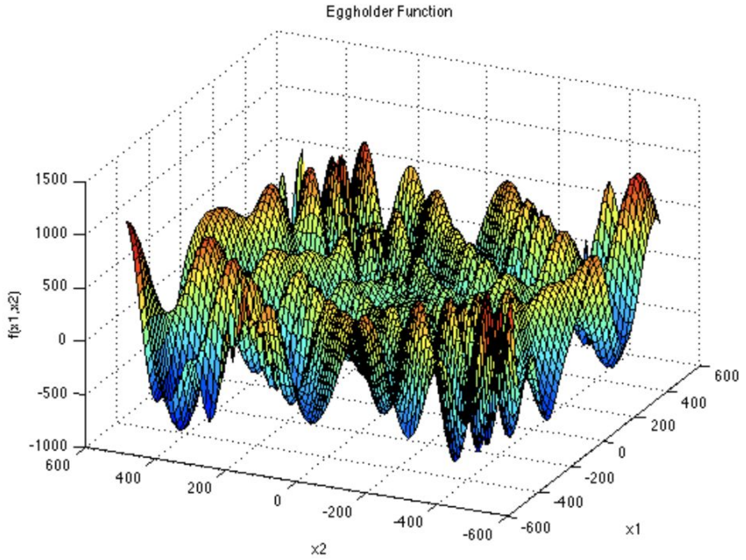
Benchmark: Bukin



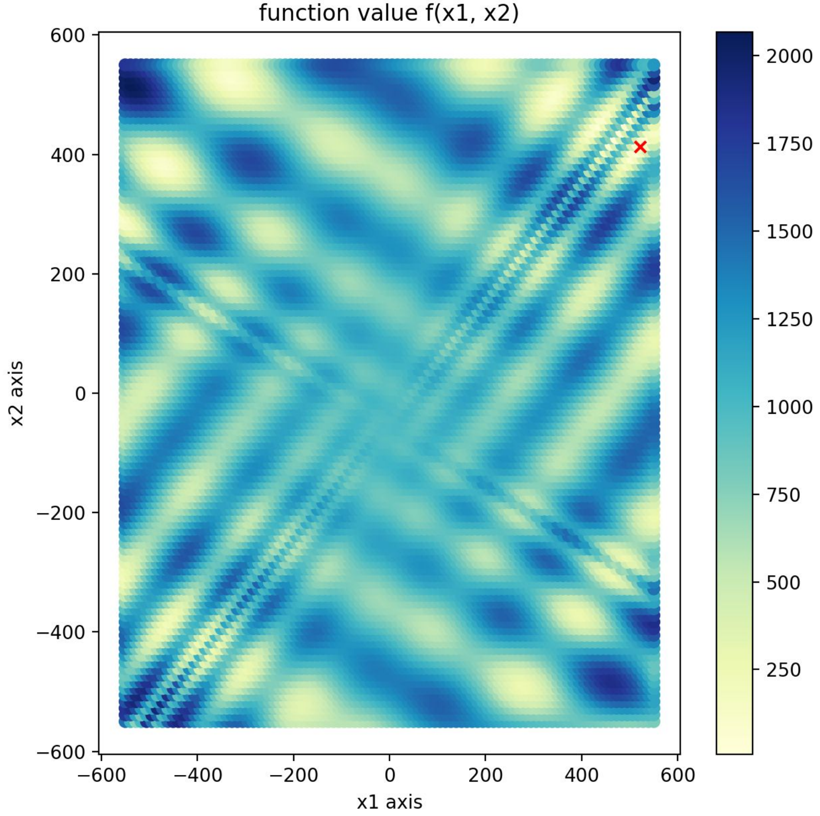
$$f(\mathbf{x}) = 100\sqrt{\left|x_2 - 0.01x_1^2\right|} + 0.01|x_1 + 10|$$



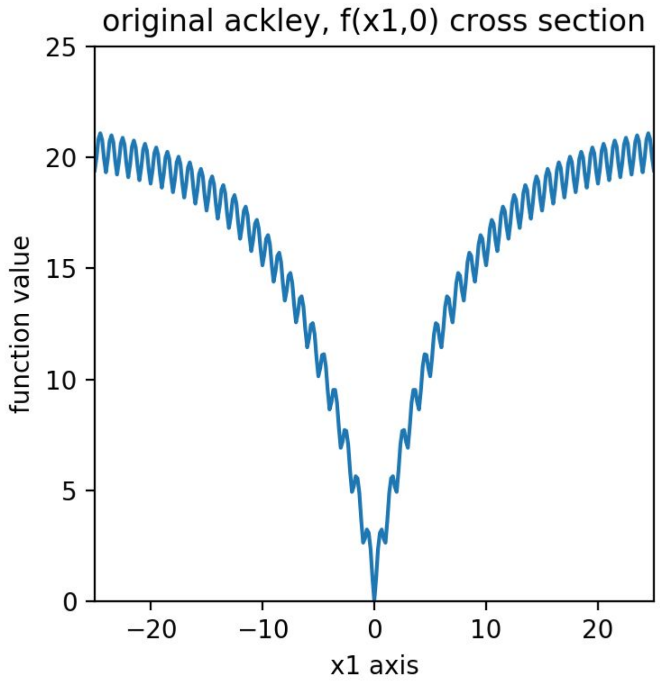
Benchmark: Eggholder

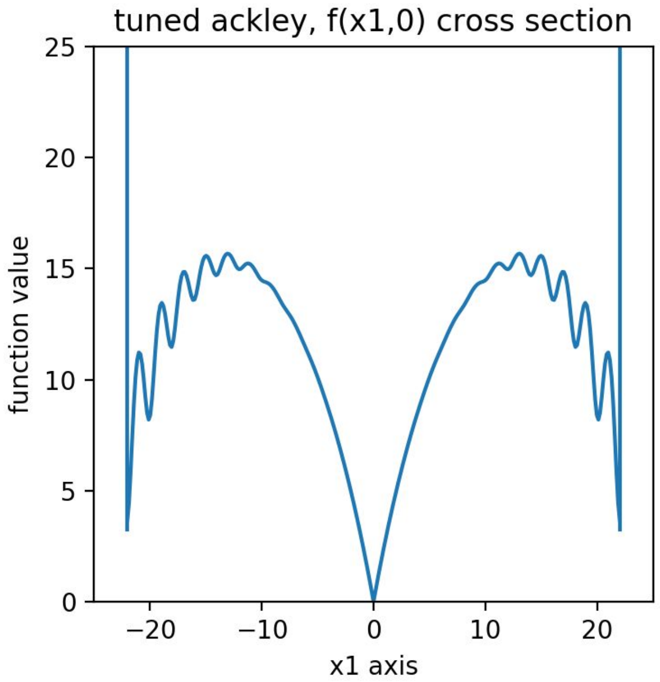


$$f(\mathbf{x}) = -(x_2 + 47)\sin\left(\sqrt{\left|x_2 + \frac{x_1}{2} + 47\right|}\right) - x_1 \sin\left(\sqrt{|x_1 - (x_2 + 47)|}\right)$$



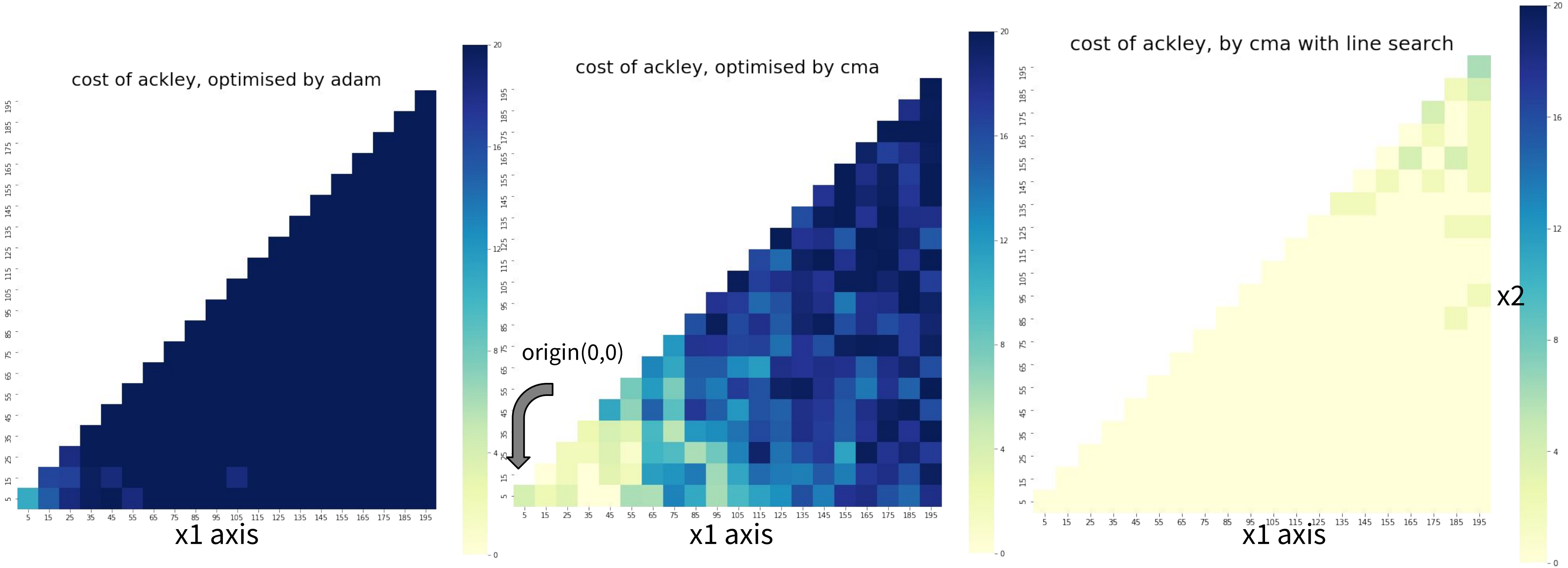
Benchmark: Tuned Ackley





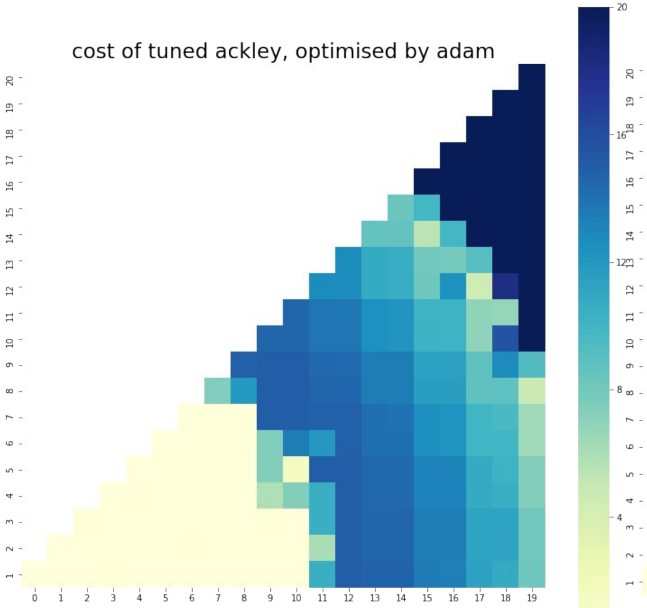
Tuned Ackley has a big trap zone, with local minimas leading to edge rather than global minima

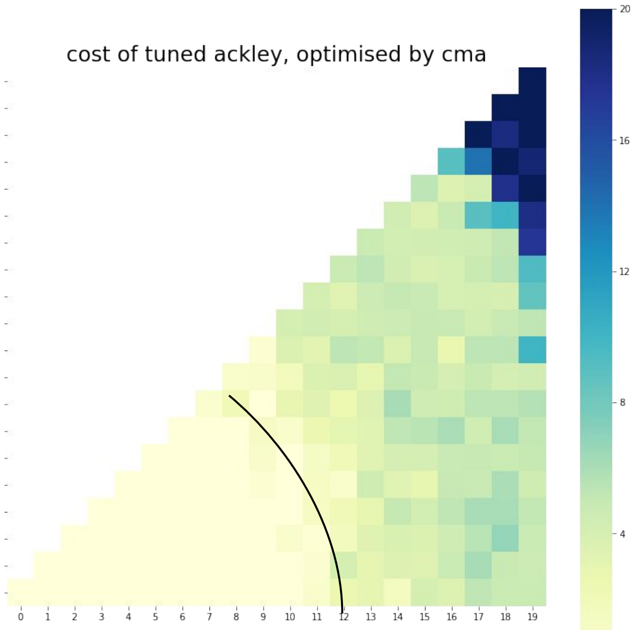
Performance: on Ackley

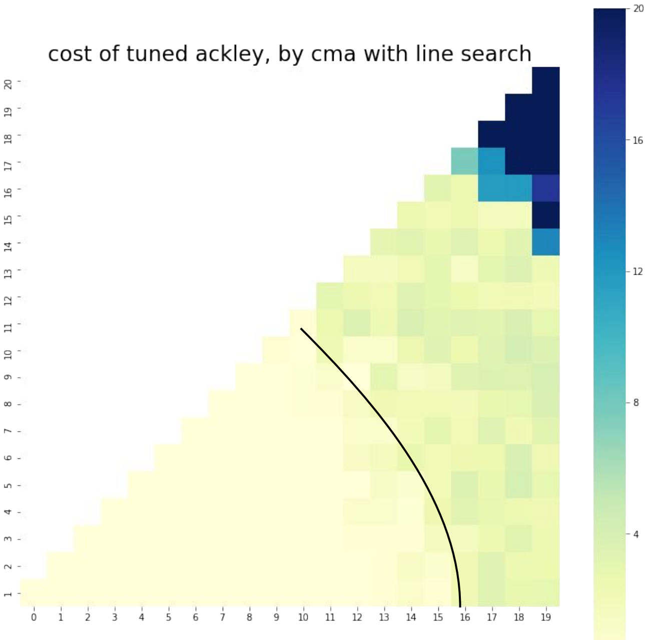


Average cost, function value	20.03	15.06	0.24
P(find global minimum)	0.05%	21.71%	98.81%

Performance: on Tuned Ackley

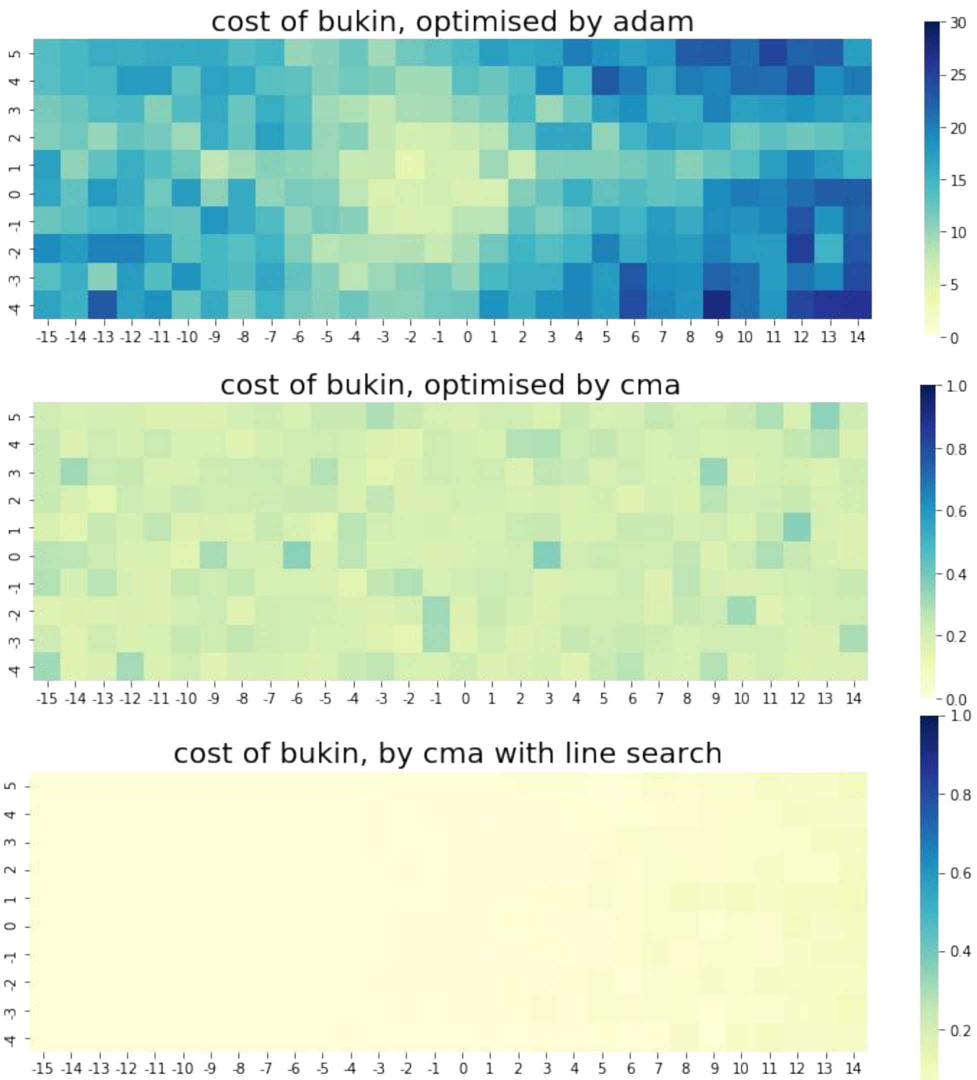






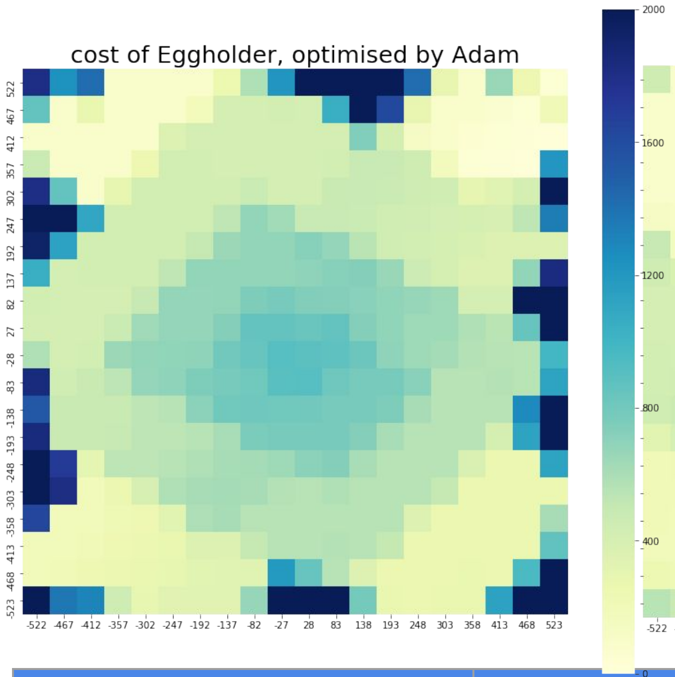
	Adam	CMA-ES	CMA-ES with line search
Average cost	13.83	4.49	3.09
P(find global minimum)	25.1%	43.3%	62.5%

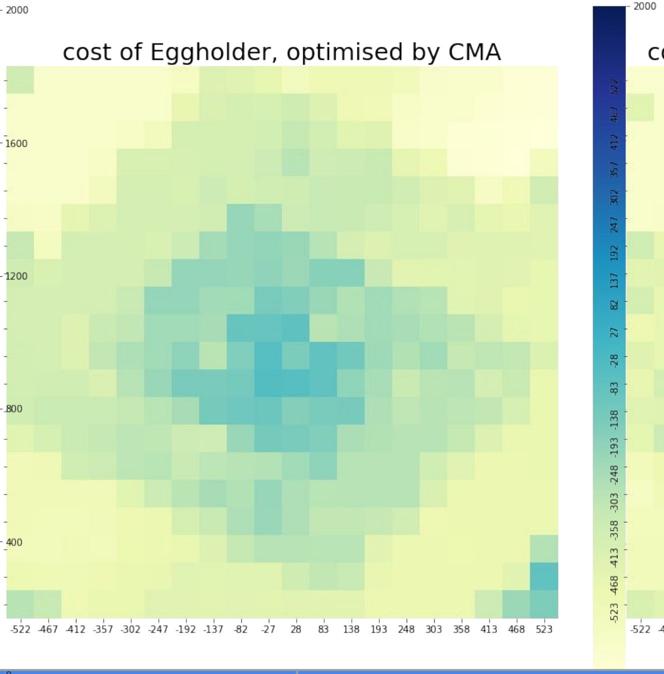
Performance: on Bukin

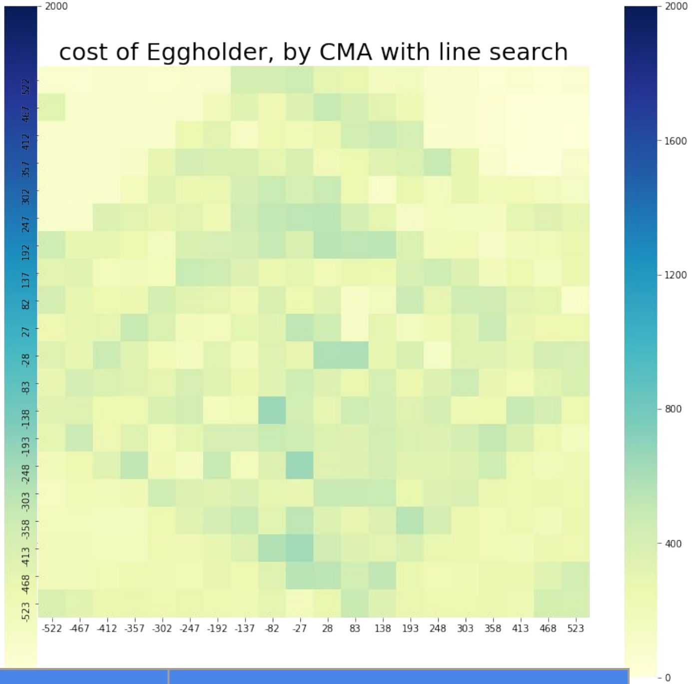


Average cost	P(find global minimum)
14.64	0.43%
0.22	0.7%
0.017	95.6%

Performance: on Eggholder

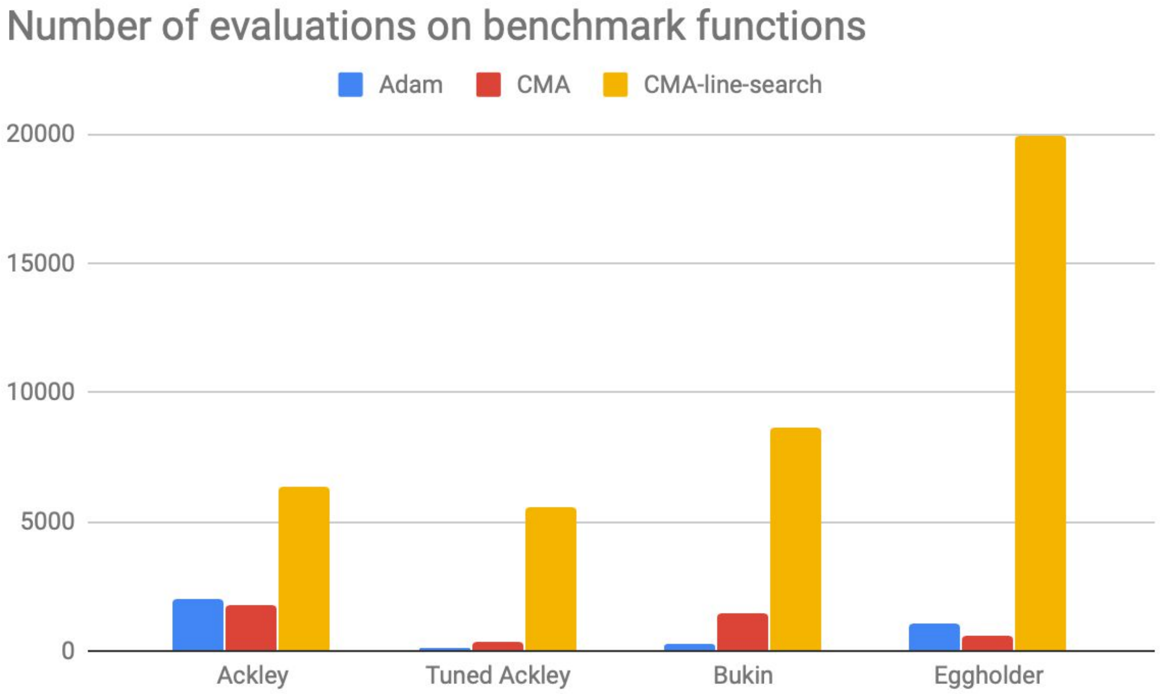






	Adam	CMA-ES	CMA with line search
Average cost	627	394	289
P(find global minimum):	0	1.3%	5.6%

Efficiency



Time per evaluation: ~10^(-6) s

Summary on 2D case

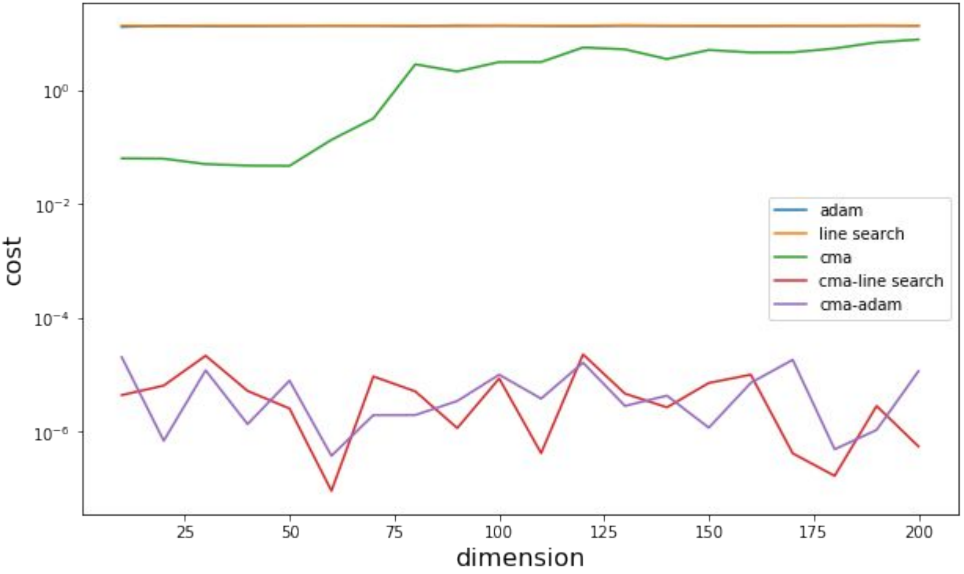
objective	optimizer	prob of convergence	cost	# of evaluations	time
ackley	*do nothing*	*0*	*21.55*	*1*	*0.01s*
	adam	0.	19.98	**0.14k**	0.9s
	line search	0.	19.98	39.9k	**0.12s**
	cma	0.	19.82	1.46k	12.6s
	cma-line-search	**0.41**	**11.73**	70.9k	199s
	cma-adam	0.19	16.27	156.4k	818s
eggholder	*do nothing*	*0*	*940.7*	*1*	*0.019s*
	adam	0.	715	1.6k	15.6s
	line search	0.	464	**0.24k**	**1.3s**
	cma	0.	436	0.27k	4.9s
	cma-line-search	**0.136**	**341**	45k	280s
	cma-adam	0.012	428	0.28k	8.1s
tuned ackley	*do nothing*	*0*	*21.55*	*1*	*0.012s*
	adam	0.2.	15.1	**0.12k**	**0.8s**
	line search	0.27	13.3	0.31k	0.9s
	cma	0.47	4.0	0.46k	4.7s
	cma-line-search	**0.62**	3.6	26.1k	75s
	cma-adam	0.54	**2.9**	14.5k	128s
bukin	*do nothing*	*0*	*260*	*1*	*0.007s*
	adam	0.	11.36	7.2k	21.7s
	line search	0.46	0.12	**2.1k**	**11.8s**
	cma	0.	0.99	2.4k	12.7s
	cma-line-search	**0.97**	**4e-5**	266k	263s
	cma-adam	0.	0.89	22.4k	64s

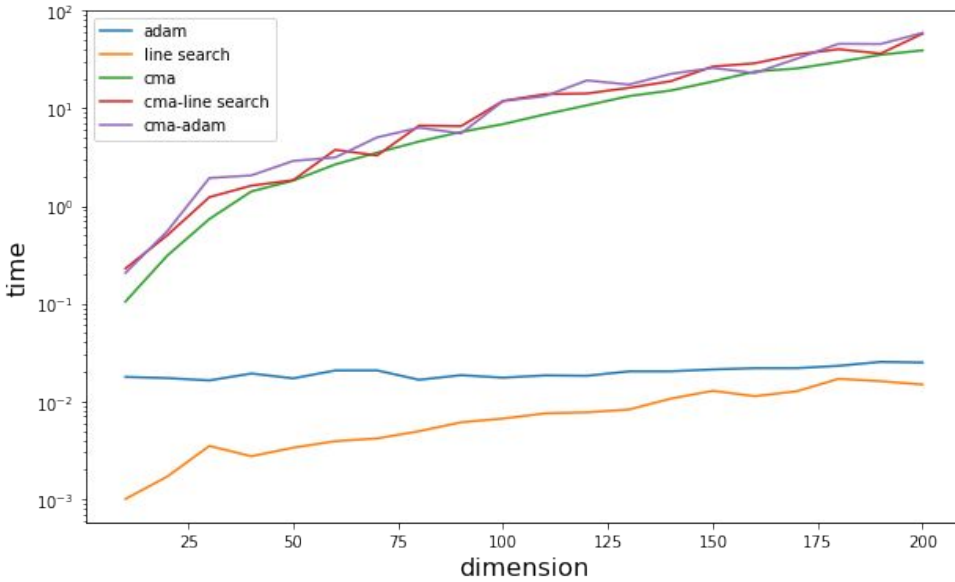
Scalability

Sample starting point

uniformly from (0, 10)^d, d = 10, 20, .., 200

Log scale, unit: (s)





Unsolved limitation

Efficiency:

- How to mitigate Curse of dimensionality?

  Possible solution: early stop of cma-es, rely more on injected gradient based optimiser

Saddle points problem:

- The performance in face of local minimas is excellent, but how about saddle points?

  In high dimensional problem of practical interest, the number of saddle points is exponentially high.

"Identifying and attacking the saddle point problem in high-dimensional non-convex optimization" Dauphin, Yann N., Yoshua Benjio, et al. *NIPS,* 2014.

Conclusion

CMA-ES with line search/adam has big advantages over Adam, original CMA-ES, line search in terms of :

- More convergence on low dimensional benchmark functions
- Keep effective on high dimensional benchmark functions

for objective function satisfying:

- Differentiable and efficient to compute gradient
- low dimensional parameter space(< 500 Dims)
- Multi-modal function with **many local minimas**