CMA-ES requires two components from the user:

- the initial start point x0;
- the initial value for sigma, the so-called step-size or error guess.

See https://www.lri.fr/~hansen/cmaes_inmatlab.html#practical for another set of detailed useful advices by Nikolaus Hansen on how to best use CMA-ES.

Choosing the initial step-size sigma and initial starting point x0

the optimum that is looked after should better not be far away from the interval [x0 - sigma0, x0 + sigma0] in each dimension, where distance is defined by sigma0.

How to best choose among algorithms and restart strategies

Active CMA-ES as 'aCMAES' as in

```
CMAParameters<> cmaparams(...);
cmaparams._algo = aCMAES;
```

is probably the best algorithm in most cases.

When optimization fails or fails finding what appears to be a good minimum, it is likely that increasing the number of offsprings per generation 'lambda' might lead to better results. Note that the initial value of sigma and x0 are also very important.

When increasing lambda seems to improve the minimum, it is a good bet to use active BIPOP-CMA-ES (abipop), that applies a successive scaling of lambda in order to explore both locally and globally.

However, bear in mind that both IPOP and BIPOP can greatly increase the number of calls to the objective function. So when calls to the objective function are expensive, this may not be a good strategy.

Note that the maximum number of restarts for IPOP and BIPOP can be controlled through the CMAParameters.set_restart function.

Dealing with large parameter spaces

CMA-ES complexity is quadratic the parameter space dimension. When this number becomes prohibitive, each iteration will become very costly or will exhaust memory.

When facing such a problem, the alternative is to use separable (active) CMA-ES (sepacmaes), a variant that limits the covariance error update to the diagonal elements. The lack in precision then comes at the benefit of a complexity that is linear the parameter space dimension.

Fixing one or more parameter values

The library supports fixing one or more parameters' values before optimization takes place or, when more control is required, at each iteration as needed.

This is achieved by calling CMAParameters.set_fixed_p function.

The fixed parameters then retain their values until CMAParameters.unset_fixed_p is called.