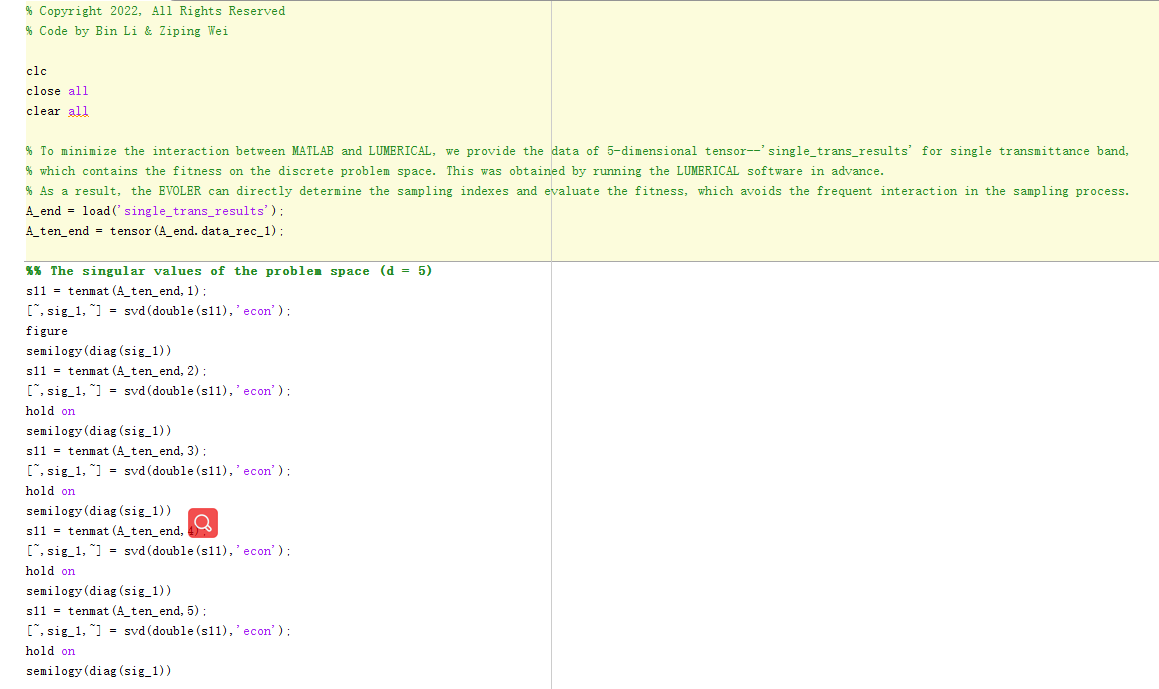
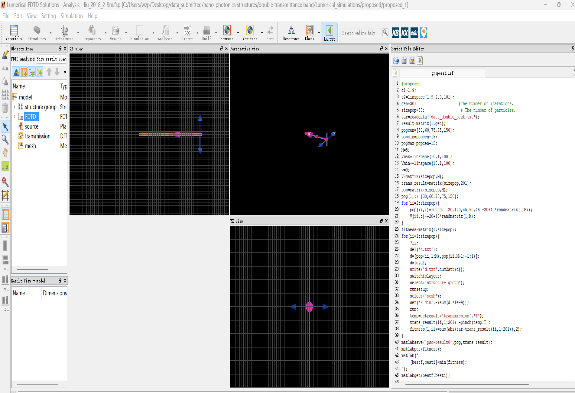
**This file provides the instruction for the related source code of the Fig. 6 in our manuscript.**

The LUMERICAL/MATLAB source code of different PSO variants and our EVOLER algorithm, as well as their running results, are structured into the corresponding folders.

User Instruction for Reproducing Fig. 6

1. **Source Code Running**

Our code can directly run on the proper software version, i.e., the MATLAB R2019a & LUMERICAL 2018a, as seen in the following:

1. **Running time**

For the source code of our proposed EVOLER method (inverse design of nano-photonics structures), the running time of the first stage structured sampling process is less than 3 hours (@ CPU 4GHz, RAM 32GB; can be implemented on multiple computing severs as the solution will not be updated in this stage). Then, for the second stage of finding the [global optimum](javascript:;) via canonic PSO, the estimated running time would be less than 1 hours, for each single trial (50 particles, 5 independent trials, 30 generations for EVOLER).

For various PSO methods, the whole optimization process will consume about 24 hours (50 particles, 5 independent trials, 200 generations; only single computing server, as the solution needs to be cooperatively updated in each generation).

1. **Implementation guidelines**

Since the operations involve the interactions between MATLAB and LUMERICAL, in the following we give the implementation guidelines when reproducing Fig. 6.

* **Reproducing Fig. 6b and Fig. 6c:**

Firstly, open the MATLAB platform, open the folder ‘Nano-Photonics\_Structures\_reproducing\_Fig6’ and open the folder ‘single transmittance band’; and then, run the source code ‘single\_trans\_EVOLER.m’, which outputs the estimated global optimum in a representation space learned by our EVOLER method. In the meantime, the singular values distribution of the problem space (*d* = 5) is also obtained, i.e., the Figure.6-(b). Finally, this estimated global optimum, as the center of identified attention subspace, is saved in the data matrix ‘index\_1\_est’.

Note that, to minimize the interaction between MATLAB and LUMERICAL, we provide the data of *d*-dimensional tensor (*d* = 5), which contains the fitness on the discrete problem space (discrete size *N*=10; for a single slice of this *d*-dimensional tensor, we can also draw the fitness values as in Fig. 6c; *a*2 vs *a*3, *a*1=120 nm, *a*4=80 nm, *a*5=80 nm; two bell-shaped bands). This was obtained by running the LUMERICAL software in advance. As a result, the EVOLER can directly determine the sampling indexes and evaluate the fitness, which avoids the frequent interactions in the sampling process. That is to say, this precomputed tensor only serves as the fitness function, which helps to minimize the interactions between MATLAB and LUMERICAL.

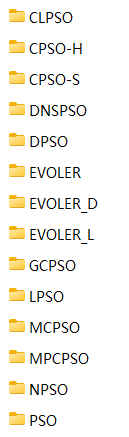
* **Reproducing Fig. 6d and Fig. 6f:**

The implementation flow of reproducing the averaged optimization curves of evolutionary methods is given as follows. We firstly open the folder ‘double transmittance band’, and run the main file ‘double\_trans\_EVOLER.m’ on MATLAB software, which outputs the center of attention subspace (i.e., the estimated global optimum in a discrete representation space). This center of attention subspace is saved to a data matrix ‘index\_2\_est’.

Then, our method initializes the population around the identified attention subspace. In this case, we can directly copy this result to each file ‘**EVOLER.lsf**’ in **\double transmittance band\Lumerical\_simulations\EVOLER.**

For the other case of single transmittance band, the similar operation can be applied.

At last, open the LUMERICAL software and then open the folder ‘**\double transmittance band\Lumerical\_simulations\**; the source code of each evolutionary method is structured into the corresponding folders, i.e.,



Note that, each method was running independently for 5 times. Taking the classical CLPSO method for example, one can directly run the source code “clpso.lsf” (on LUMERICAL, but this may consume tens of hours) and then obtain the result (i.e., saved in the data matrix ‘result.mat’), including the obtained transmittance response as well as the calculated fitness. For convenience, we have also provided the simulation results of 5 independent trials in 5 separated files (i.e., ‘clpso\_1’, ‘clpso\_2’, ‘clpso\_3’, ‘clpso\_4’, ‘clpso\_5’). Based on these 5 independent trials, the average fitness curves can be derived.

Finally, by running the source code ‘plot\_mean\_double.m’ (on MATLAB), the results of Fig. 6d can be obtained.

The same operations can be applied to the case of single transmittance band (Fig. 6f can be obtained).

* **Reproducing Fig. 6e and Fig. 6g:**

Finally, we can obtain the transmittance response of our EVOLER method for the single transmittance band (Fig. 6g), by running the source file ‘**load\_data\_single.m**’ in the following folder: **\** **single transmittance band\Lumerical\_simulations\EVOLER**; and we also obtain the transmittance response of our EVOLER method for the double transmittance band (Fig. 6e), by running the source file ‘**load\_data\_double.m**’ in the following folder: **\** **double transmittance band\Lumerical\_simulations\EVOLER**;