

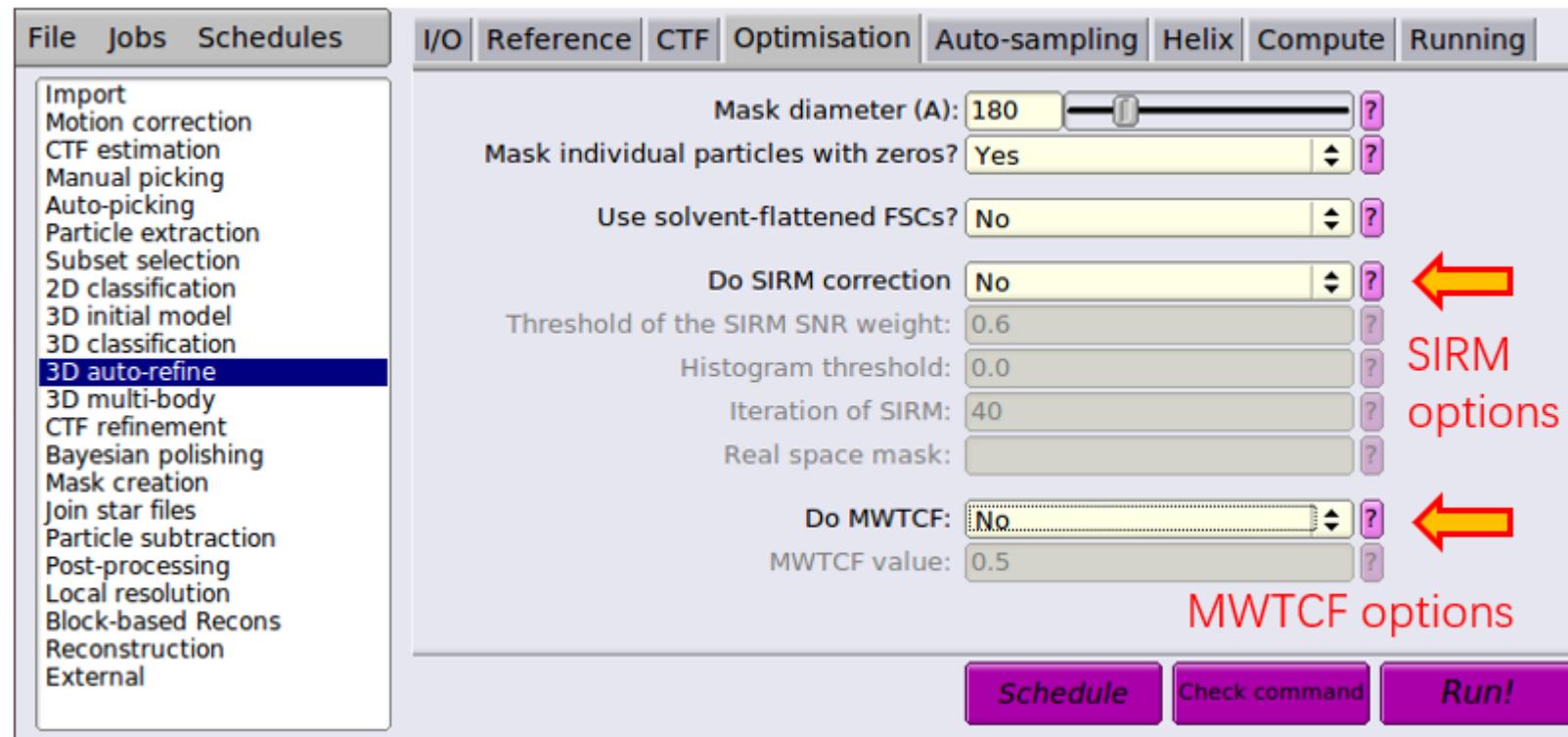
https://github.com/homurachan/SIRM_RELION

加载环境变量

```
conda activate SIRM_ENV  
&&  
cd SIRM_RELION/Hangzhou_Workshop/  
&&  
source source_this.bashrc
```

SIRM-RELION基于RELION 3.1.2，常规使用上与RELION3.1.2无任何不同。

1、在refine过程中使用SIRM矫正



1、在refine过程中使用SIRM矫正

The screenshot shows the EMAN2 software interface with the 'I/O' tab selected. On the left, a sidebar lists various processing steps, with '3D auto-refine' highlighted in blue. The main panel contains several configuration options:

- Mask diameter (A): 180
- Mask individual particles with zeros? Yes
- Use solvent-flattened FSCs? No
- Do SIRM correction? Yes
- Threshold of the SIRM SNR weight: 0.6
- Histogram threshold: 0.0
- Iteration of SIRM: 40
- Real space mask: (empty)
- Do MWTCF? Yes
- MWTCF value: 0.5

A large yellow arrow points from the text "Real space threshold. Using histogram to determine it after each iteration" to the "Histogram threshold" field. Three red arrows point from the right side of the slide to the following fields:

- Threshold of the SIRM SNR weight: Text: Fourier SNR weight threshold. It should be set by trial-and-error.
- Iteration of SIRM: Text: It's recommended to be less than 60 during refinement
- MWTCF value: Text: Additional mask

A yellow double-headed arrow points between the "Schedule" and "Run!" buttons at the bottom.

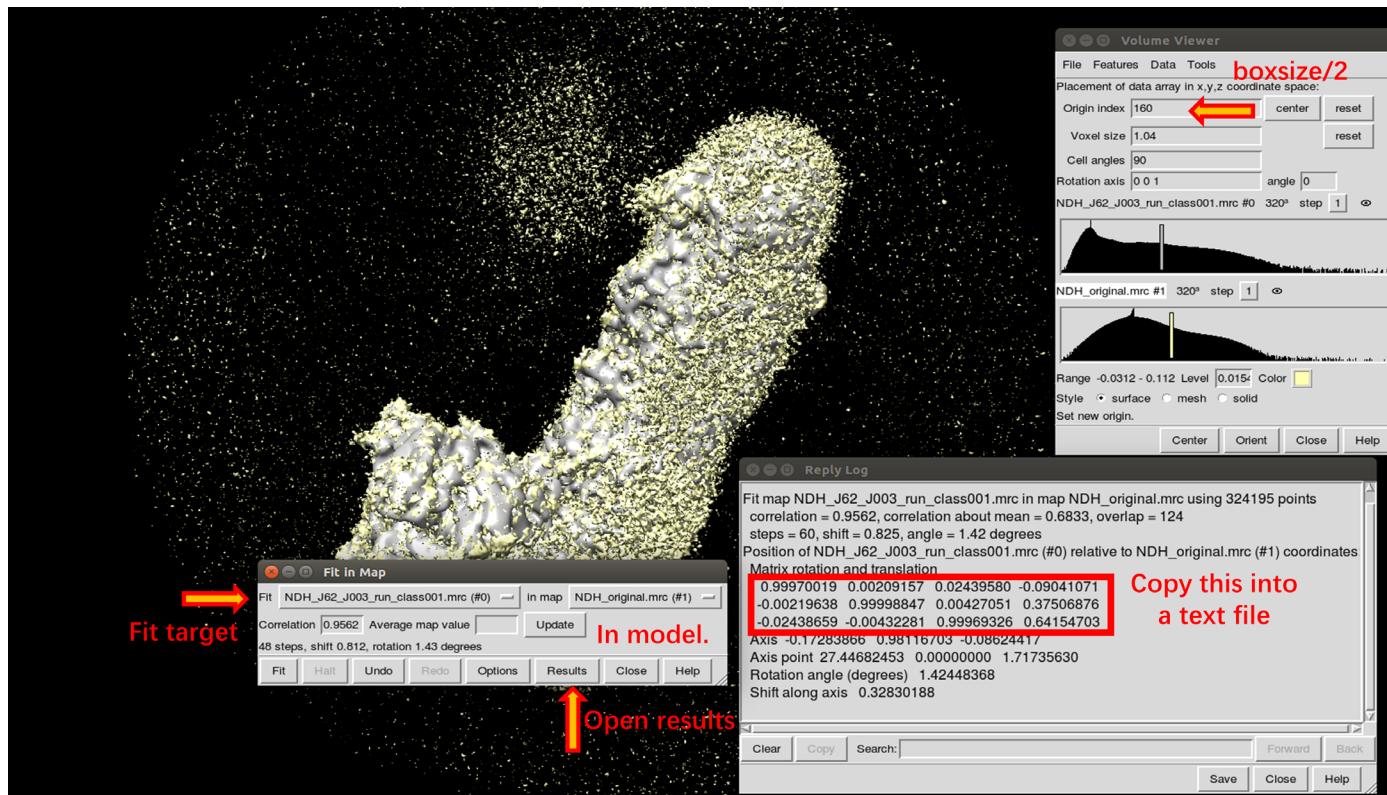
Interpolation number less than this (0.5 here)* average would not account for resolution.

2、Validation, 比较cryoSPARC与RELION结果

1、用pyem的csparc2star.py把cryosparc的.cs文件转换成star文件
推荐使用relion_star_handler --i input.star --o output.star把转换的star文件再调整一下。

2、用SIRM-RELION refine。

3、通常， cryosparc与relion之间会存在一个小的角度偏差。需要矫正



使用scripts文件夹下的Validation_preprocess_read_relion_star_rotate_accordingto_chimera_fitting.py
处理。

```
E:\>type J003.txt
0.99970015 0.00208527 0.02439808 -0.09048470
-0.00218999 0.99998850 0.00426617 0.37514684
-0.02438890 -0.00431832 0.99969322 0.64161460

E:\>python ./read_relion_star_rotate_accordingto_chimera_fitting.py
<input star> <chimera txt> <pixel size> <output star> <output INT translation txt>

E:\>python ./read_relion_star_rotate_accordingto_chimera_fitting.py J003.star J003.txt 1.04 J003_fit.star INT.txt
Is relion3.0? = 0
[[ 0.99970015 0.00208527 0.02439808]
 [-0.00218999 0.99998850 0.00426617]
 [-0.02438890 -0.00431832 0.99969322]]
```

4、Validation本体操作

只需要对 relion 的结果进行偏差矫正, 得到了一个新的 star 文件。我们叫它 relion_rotted.star, cryosparc 的结果叫 cs_refine.star。使用 python Validation_compare_cryosparc_and_relion.py 进行处理。命令为:

Python Validation_compare_cryosparc_and_relion.py relion_rotted.star cs_refine.star “角度阈值” “Cn 对称性的 n” validation_result.star

这里需要注意, 首先要看两个 star 文件名的命名情况。因为 cryosparc 会自动加上一些前缀, 而 relion 不会。因此如果是形如 1111111111_real_filenames 这样的文件名, 使用 cryosparc_filename 这个函数, 如果是 relion 形式的, 即文件名没有 11111111 前缀, 则使用 relion_filename 这个函数。具体来说, 需要修改这里

```
71
72     for i in range(mline,len(instar_line)):
73         if(instar_line[i].split()):
74             imagename=str(instar_line[i].split()[IMG_index])
75             image_serial=int(imagename.split('@')[0])
76             image_filename_tmp=cryosparc_filename(imagename)
77             image_filename=""
78             cryosparc2relion_serial.append([])
79             cryosparc2relion_imagename.append([])
80             cryosparc2relion_line.append([])
81             for mm in range(0,len(image_filename_tmp)):
82                 image_filename+=image_filename_tmp[mm]
83                 image_filename+=" "
84             cryosparc2relion_serial[len(cryosparc2relion_serial)-1]=image_serial
85             cryosparc2relion_imagename[len(cryosparc2relion_imagename)-1]=image_filename
86             cryosparc2relion_line[len(cryosparc2relion_line)-1]=i
87
88             print("Finish reading part1")
89             orirelion_serial=[]
90             orirelion_imagename=[]
91             orirelion_line=[]
92             for i in range(mline2,len(oristar_line)):
93                 if(oristar_line[i].split()):
94                     imagename=str(oristar_line[i].split()[ori_IMG_index])
95                     image_filename_tmp=cryosparc_filename(imagename)
96                     image_serial=int(imagename.split('@')[0])
```

第一个绿框是第一个 star 的形式，第二个同理。如果 relion 的结果是从 cryosparc 转换过来的，用 cryosparc_filename 函数。

简而言之，从cryosparc来的，用cryosparc_filename，从relion来的，用relion_filename

Python Validation_compare_cryosparc_and_relion.py relion_rotted.star cs_refine.star “角度阈值” “Cn 对称性的 n” validation_result.star

角度阈值，指的是同一个颗粒的允许最大角度偏差。一般来说，要看 relion 报的 accuracy angles。对于小蛋白来说，2 到 3 倍是合适的。同样需要测试，因为如果角度太小，会扔掉很多颗粒，角度太大又会留下很多不好的颗粒。

Cn 对称性的 n，以 HA 为例，n=3。非对称蛋白 n=1

运行结果是 3 个文件，validation_result.star 储存每个颗粒的角度变化值，validation_result.star_lessthan_阈值_change_line.star 是有效的结果，其中颗粒的取向是脚本的第一个输入文件，这里是 relion_rotted.star。建议使用 relion 的结果，因为我们的测试当中，这里用 relion 几乎总是比 cryosparc 好。validation_result.star_morethan_阈值_change_line.star 是扔掉的颗粒。

4、Validation本体操作

简单总结一下：

- 1、需要用cryosparc与relion同时refine同一组数据
- 2、要修正重构模型的角度偏差。
- 3、设定阈值，把颗粒角度变化大于该阈值的颗粒去除。

5、Mask-Picking

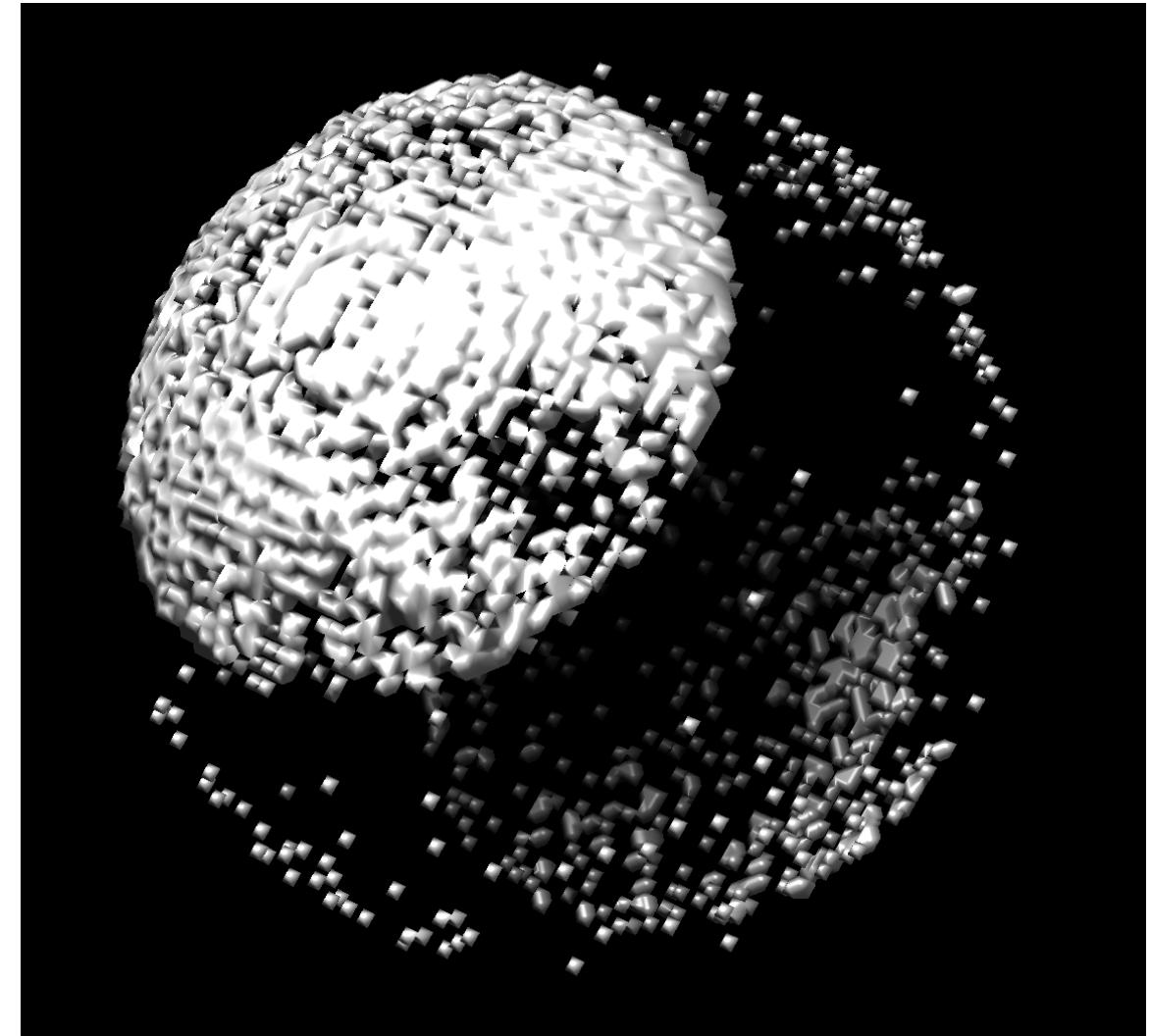
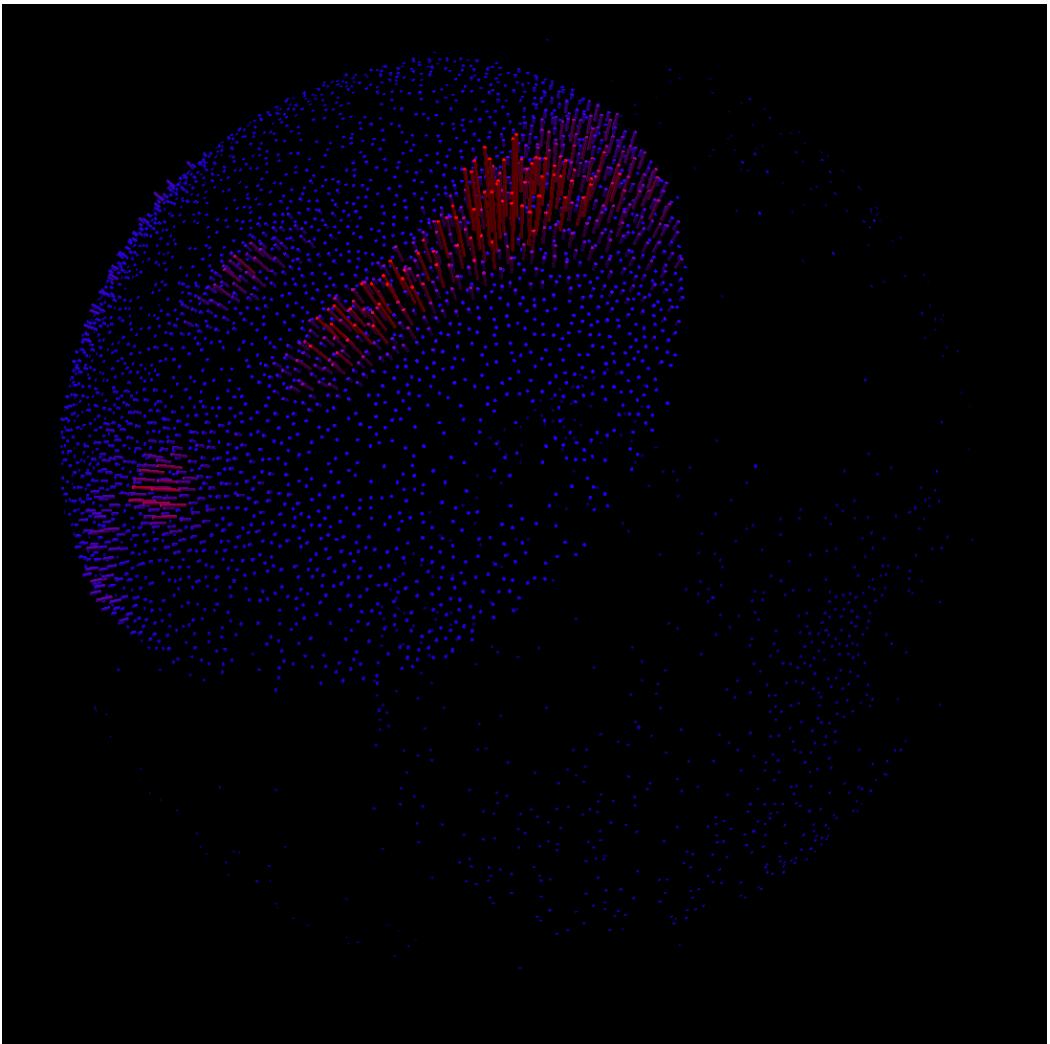
Mask-Picking 一般是接着 validation 做的。这里用 validation_result.star_lessthan_ 阈值_change_line.star 为例。首先，使用 SIRM-RELION 里面的 relion_star_handler（普通版没这个功能），生成 bild 文件。命令是：

Python MaskPicking_step1_read_relion_bild_produce_mrc.py test.bild 0.9375 “输出 mrc 半径，单位 pixel” test.bild.mrc

接着，使用 MaskPicking_step1_read_relion_bild_produce_mrc.py 把 bild 转换成 mrc 文件。
要求 python 安装了 numpy 和 mrcfile 库。

Python MaskPicking_step1_read_relion_bild_produce_mrc.py test.bild 0.9375 “输出 mrc 半径，单位 pixel” test.bild.mrc

5、Mask-Picking



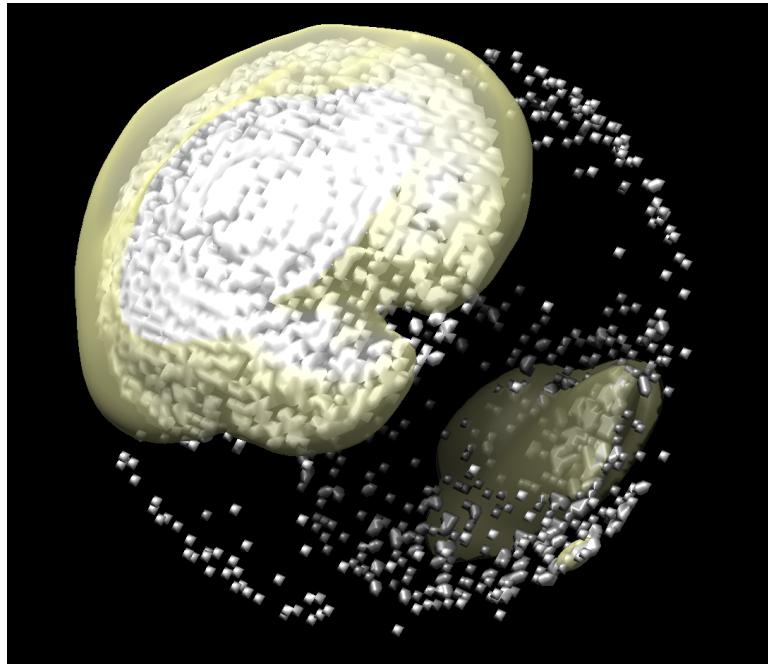
5、Mask-Picking

很容易想到，需要进行连续化，即进行 lowpass。方法有很多，例如用 eman 的 proc3d in.mrc

lp.mrc apix=0.5 lp=5，或者用 relion_image_handler --i in.mrc --o out.mrc --angpix 0.5 --

lowpass 5. 这里的参数表示 lowpass 到了 1/5 倍 Nyquist。

Relion_image_handler有问题。



Low-pass后，选择合适的阈值，擦除不需要的部分，用relion_mask_create生成mask。

接下来用该mask生成一个欧拉角文件。

不属于该欧拉角的颗粒都剔除。

5、Mask-Picking

接下来，利用该 mask 生成一个角度 table，用来挑选颗粒。使用 SIRM-RELION v1.1 的 relion_image_handler 处理，命令为：

```
relion_image_handler --i test_lp_erased_mask.mrc --o nouse.mrc --gen_MP_table --  
MP_table out_anglelist.star --MP_stepsize “角度间隔 a”
```

这里的 nouse.mrc 可直接删除。所需参数如下图中选中区域。

```
===== additional subtract options =====  
--optimise_scale_subtract (false) : Optimise scale between maps before subtraction?  
--optimise_bfactor_subtract (0.) : Search range for relative B-factor for subtraction (in A^2)  
--mask_optimise_subtract () : Use only voxels in this mask to optimise scale for subtraction  
===== SIRM parameter. Not recommended for initial refine, except for local refinement. =====  
--SIRM (false) : Perform SIRM reconstruction  
--thres_SIRM (0.3) : SIRM threshold in Fourier space  
--Hist_cutoff (0.0) : cut-off of real space restrain  
--thres_hist (0.0) : SIRM, histogram threshold in real space  
--iter_SIRM (30) : Number of iteration of SIRM  
--SIRM_snr_weight () : SNR weight file in RFLOAT  
--max_radius_SIRM (100.0) : SIRM, histogram threshold in real space  
--gen_MP_table (false) : Read MRC file and generate Mask-Picking table. Use together with --MP_table and --MP_stepsize  
--MP_table () : Output name of Mask-Picking table.  
--MP_stepsize (0.9375) : Stepsize of Mask-picking table
```

最后，使用 SIRM-RELION v1.1 中的 relion_star_handler 把不属于 mask 以内的颗粒扔掉。

5、Mask-Picking

```
relion_star_handler --i validation_result.star_lessthan_ 值 _change_line.star --o  
maskpick_result.star --MP_write_particles --MP_table out_anglelist.star --MP_stepsize "角度  
间隔 b"
```

注意，这里角度间隔 b 必须大于角度间隔 $a/2$ ，否则会漏掉颗粒，推荐 $b=a$ 。所需参数如下图中选中区域。

```
===== !!!Special option of Block-based reconstruction =====  
--do_invert_BBR_handerness (false) : Suppose your handerness was correct.  
===== Write bild options =====  
    --write_bild (false) : Generate bild from input star file.  
    --bild_angle_step (0.9375) : Angle step of bild  
    --bild_radius (128.0) : Radius of inner sphere of bild  
===== Mask-Picking, read tables write particles. =====  
    --MP_write_particles (false) : Enable Mask-Picking. Use together with --MP_table and --MP_stepsize.  
        --MP_table () : Mask-Picking table name  
        --MP_stepsize (0.9375) : Step size of Mask-picking table
```

5、Mask-Picking

简单总结一下：

- 1、经过Validation后， 把star文件转换成bild文件， 再转换成mrc文件。
- 2、对mrc文件进行lowpass和擦除不需要的取向。
- 3、生成角度table，并筛选颗粒。
对star文件处理用relion_star_handler， 对mrc文件处理用relion_image_handler。

6、最终重构

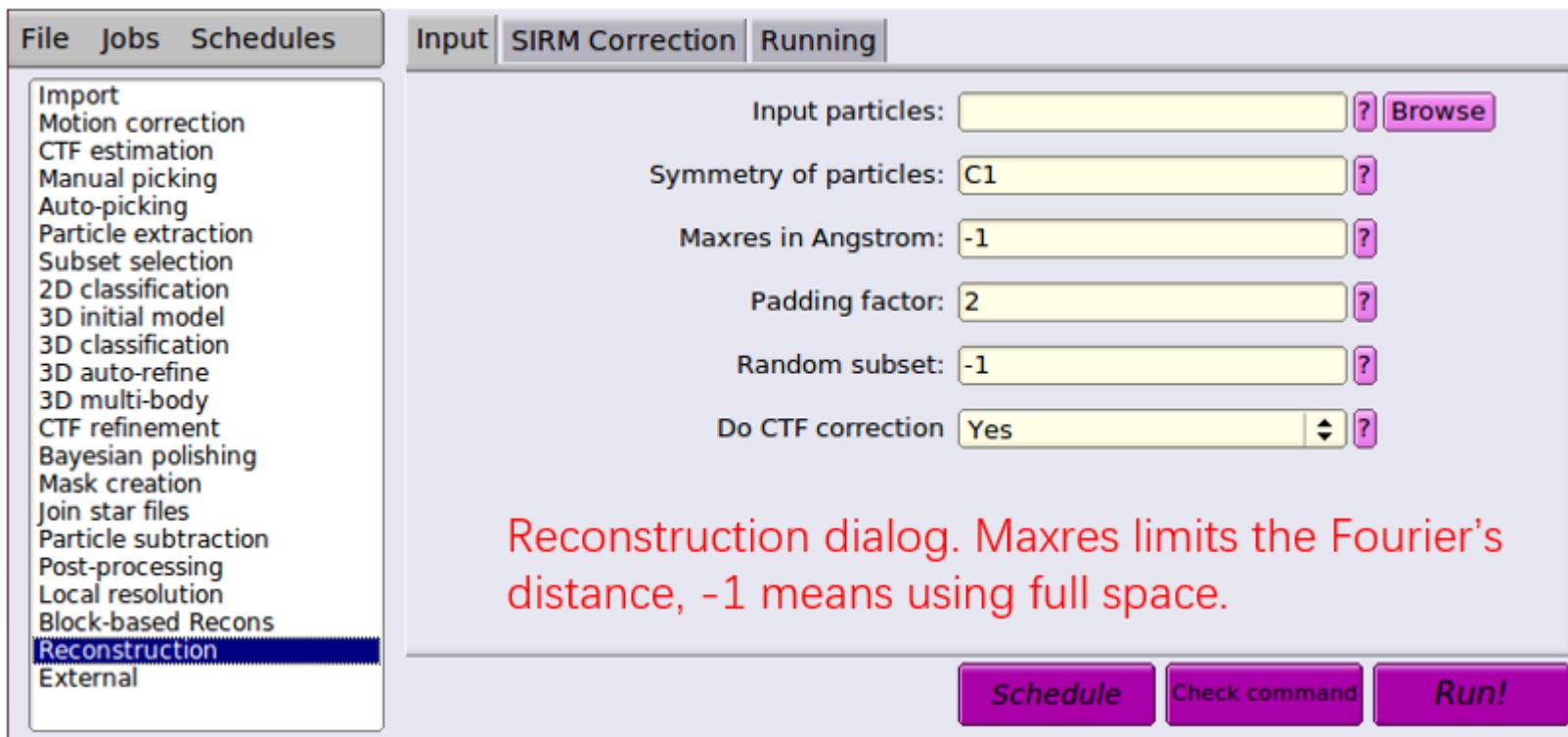
需要注意， refine时SNR weight自动生成，但是，重构时没有FSC，因此要手动生成一个。

先进行postprocess，得到了postprocess.star后，用scripts里的convert_star_2_SIRM_weight.py

```
python convert_star_2_SIRM_weight.py postprocess.star weight.star
```

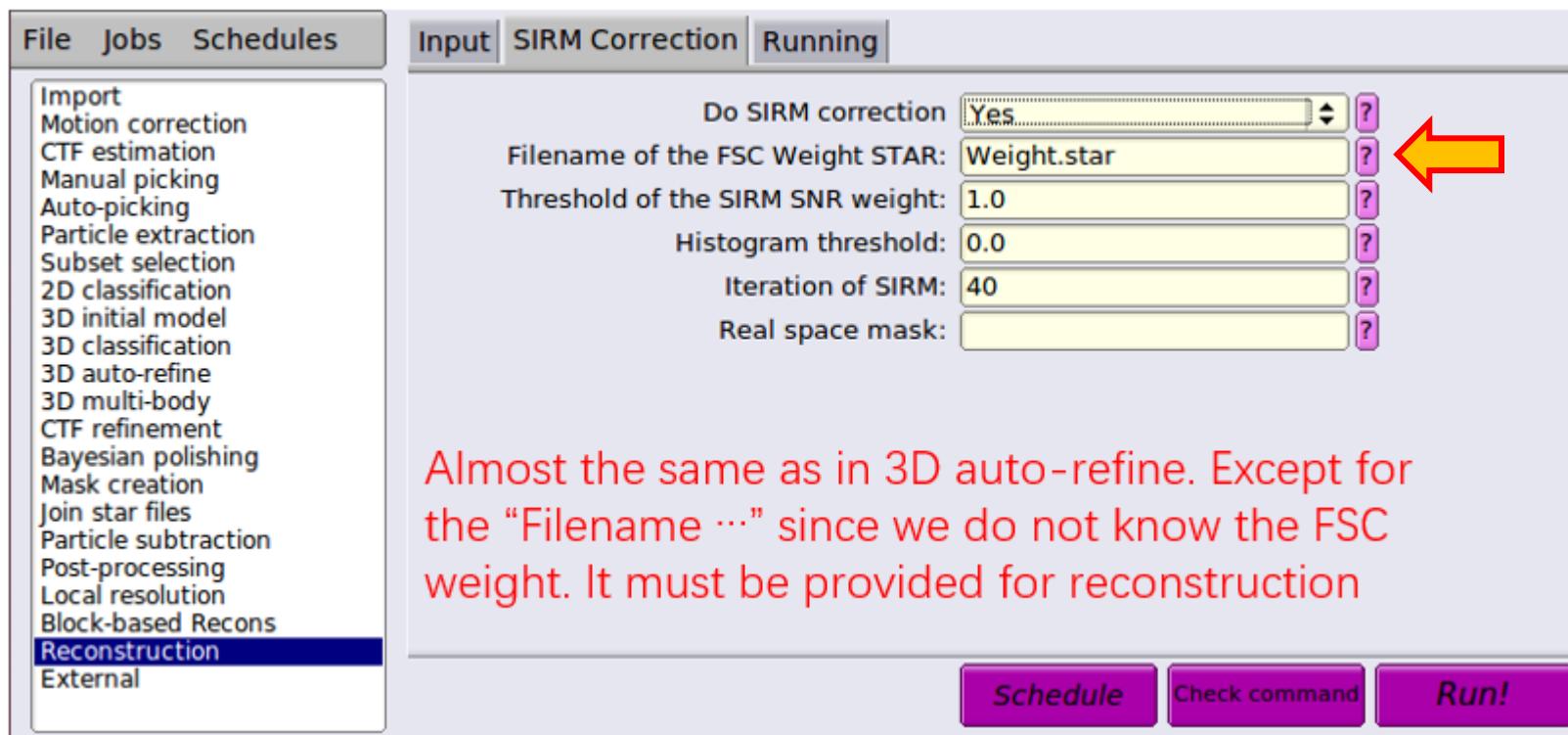
生成weight.star

6、最终重构



由于对even和odd同时进行SIRM重构后，分辨率判断可能会出现问题，因此，我们选择对所有颗粒都进行重构。并人为进行postprocess中加FSC权，b-factor sharpening和lowpass。

6、最终重构



6、最终重构

重构后，使用relion_image_handler进行最后的postprocess。

```
===== Post-process like b-factor and FSC weighting. Should be used together with --angpix =====
    --postprocess (false) : Do postprocess for single map.
        --post_fsc () : FSC to be applied.
            --lp (-1) : Low-passed frequency
                --bf (0.) : B-factor to be applied
```

```
relion_image_handler --i reconstrct_SIRM_full.mrc -o
reconstrct_SIRM_full_lp3p6_bfM60.mrc --angpix 1.42 --postprocess --post_fsc
weight.star --lp 3.6 --bf -60
```