Adaptive Map Binary Grid Search 使用说明

总共有三个文件，包括生成map时使用的adaptive\_map\_generator.py，以及利用map做grid search时的chi2\_calculator.cpp和binary\_grid\_ob16\*\*\*\*.py，另外还需要一个compile\_all来编译chi2\_calculator.cpp；此外还有一些测试用代码；

生成map：

生成map时需要先mkdir map\_set\_\*\*\*，然后在adaptive\_map\_generator.py中修改以下几个地方：

第15行处的box\_size = \*\* ，每张map的大小是(2\*box\_size)\*(2\*box\_size)，map的中心默认为binary的质心（也可以通过246/247行修改），目前同一次生成的一系列map还都是使用同样的box\_size（后续会改进使得每张map的box\_size自动选择），所以需要根据所选的log s、log q形成的caustic大小来确保map的大小能装下所有magnification anomaly的区域；一般可以选用0.4/1.0/3.5等值；map之外用single近似；

第234行处的if current\_layer == \*\* : ，这里输入的数字代表map被加密到第\*\*层之后强行停止加密；判断是否加密的标准是放大倍数的插值误差小于

其中μ为当前位置的放大倍数，所以为了防止在靠近放大中心的位置不断加密，需要加密一定层之后强行停止的机制；加密到第11层就代表最小的小格边长为

如果box\_size = 0.4 ，那么current\_layer == 11所实现的最小小格边长约为，这对于high-magnification event一般也是够用的，对普通的event可以自行选取这一值；

第21行处的map\_set\_name = ‘'，需要修改为预先mkdir的那个准备存map的目录；

第288-328行saveparm函数中修改log s / log q / log rho的范围和步长，也可以限制grid区域为特殊形状；

第346行设置processes数目；

然后直接python3 adaptive\_map\_generator.py即可运行，注意要这个python程序需要import MulensModel包

利用map做grid search：

chi2\_calculator.cpp在正常使用中无需修改，也无需重复编译，只要在首次使用前通过./compile\_all编译一次即可；注：compile\_all中的路径需要修改；

需要在binary\_grid\_kb21\*\*\*\*.py中修改：

第12行处设置所使用的map的目录名称；

第126-129行处设置所使用的alpha初值；

第176-186行以及第191/192行设置所使用的数据；

第188/189/190行设置所使用的t0/u0/tE初值；

第213-218行设置log s / log q / log rho的范围和步长，取其与所使用的map目录的交集；

第319行设置processes数目；

第323行设置grid search结果存储位置及名称；

然后直接python3 binary\_grid\_kb21\*\*\*\*\_new.py即可运行

另外，还有一些测试用的代码：

利用map插值得到放大倍数：

在get\_magnification.py中修改：

第17/18行处设置使用哪张map进行插值；

第12/13行处设置希望得到哪个(x,y)点的放大倍数；

然后直接python3 get\_magnification.py即可运行

将生成好的map可视化：

在**visualizing\_adaptive\_map.py** 中修改：

第45/46行处设置要可视化哪张map ；

第48/49行处设置这张map对应的log s和log q ；

然后直接python3 **visualizing\_adaptive\_map.py**即可运行