一、Matlab与c的混合调试方法：

1. mex的时候在需要调试的问价编译后边添加-g参数，
2. 在matlab中在调用c函数的地方设置断点；
3. 打开VS，在VC的工具->附加到进程的对话框中选择matlab；
4. 在VS中打开需要调试的文件，在需要的地方设置断点，会提示无法调试，不用管这个
5. 回到matlab按F11步进到c函数，此时进程就跳转到了VS中
6. 然后就可以在VS中调试c程序了。

二、如何在libsvm中添加自定义核函数，从训练与预测两个阶段入手：

* 1. 训练阶段：Svmtrian.c中的mexFunction是matlab与c的接口函数，可以从这里入手
     1. 第一调用的相关函数是parse\_command\_line，从函数名可以看出它是对命令行参数的一个解析，但它的功能单一，没有判别，所以不用修改；
     2. 然后是在453行的svm\_check\_parameters:这个函数判定输入参数的合法性，由于新定义了kernel，假定我们将新的kerneltype 为5（1，2，3都是已有的类型，4是precomputed kernel）；这个函数的定义在svm.cpp中，在svm.cpp中找到函数定义，在第3050行有对kernel type的合理性判定，添加kernel\_type != INTERSECTION，kernel\_type的定义是一个枚举变量，在svm.h里面，添加我们新定义的intersection类型。
     3. 接下来回到svmtrain,在478行的svm\_train，它的定义同样在svm.cpp中，可以直接到2219行， 通过调用svm\_train\_one最终进入了solve\_c\_svc函数，接着调用Solver
     4. 类型的Solve函数，调了类型SVC\_Q的构造函数，
     5. Kernel类型的定义中添加kernel\_intersection函数，这里可以参考smaji的代码。
     6. 参照dot函数的定义方式定义intersection函数，将intersection里面的操作换成你想添加的核的操作，这里的intersection就是对应的核，输入的参数就是两个向量（svm\_node）。
     7. 在kernel的构造函数中添加intersection类的情况，并且指向你定义的函数。Kernel构造函数中用了参数初始化，用param.kernel初始化kernel\_type,用param.degree初始化degree。
     8. 在kernel\_type\_table定义里面添加intersection：
     9. 修改kernel的函数k\_function,计算单个的内积。
  2. 预测阶段:matlab与c++的接口是svmpredict 函数里面的mexFunction；
     1. 预测阶段跟踪调试之后没有另外需要修改的文件，可以直接测试效果了。