**并行程序设计期末大作业**

**使用文档及程序报告**

1. **使用的并行技术**

Openmp+mpi：在计算积分时使用了openmp多线程并行，得到矩阵后求特征值和特征向量时，若选用scalapack，则会使用mpi并行计算。

1. **编译说明**

lapack、scalapack和openmpi我都编译安装到了作业文件夹内。云端只需要用apt命令安装openmpi，blas就可以了。

我采用的是makefile，在src文件夹内打开终端，输入make clean删除已经编译的可执行文件，输入make就能编译了（会有warning但没关系）。编译成功后输入mpirun -np x ./final\_openmp即可运行（x是采用的进程数，一般4个即可）。运行时根据提示输入INPUT.txt的路径（输入default使用默认路径"../input/INPUT.txt"）。之后根据提示输入并行的线程数量。

1. **数据结构的设计**

包含的类：读入类，矩阵类，计算类。

读入类：读入基本要求，并存储格点V值、插值节点的值等相关数据。

矩阵类：存储计算得到的矩阵，提供计算特征值和特征向量的函数。

计算类：一些静态成员函数，包含求三次样条插值的函数、计算空间两点间距离的函数等。

存储的数据：源点的坐标，各点的V值，插值节点的值。

1. **优化**
2. 考虑到矩阵的对称性，只计算上三角部分。
3. 存储三次样条插值的中间量m，使得之后每次计算插值只需常数的时间复杂度。
4. 将格点分块处理，计算每块的格点的积分时先存储其关于各个源点的插值，一个块计算完成后立即释放相应插值的存储空间，这样既不需要反复计算同一格点到同一源点的插值，也不会超内存。
5. 每次选定要计算的两个源点后，先检查两个源点的距离是否大于2\*cutoff，若大于则直接令sum=0。
6. **算法设计思路**

Mpi初始化后，进程0读入输入文件，其他进程进入等待状态。进程0对输入的格点做分块，分批计算每块格点对于积分总和的贡献。计算每块格点时采用openmp多线程并行计算。

计算完成后，如果读入文件要求用lapack处理矩阵，则进程0调用lapack的函数接口计算矩阵的特征值和特征向量，其余进程不做计算。若要求用scalapack处理矩阵，则进程0将相应的数据发送给其他进程，所有进程并行计算完成后再将计算结果发送给进程0，由进程0整合得到最终的特征值和特征向量并输出。

1. **输出**

程序会在控制台输出计算用时（不包括读入数据的时间，但包括了输出矩阵和特征值及特征向量的时间）。在output文件夹内可查看矩阵的特征值和特征向量。

1. **运行时间效率及内存占用**

三个正确性测试样例均满足。

对于效率测试样例，当输入为50个源点，512\*512\*512个格点时，使用16个线程效果最佳，测试运行时间在52秒左右，最大内存占用在2.5G左右。

当输入为50个源点，1024\*1024\*1024个格点时，使用16个线程效果最佳，测试运行时间在418秒-425秒。最大内存占用在9.5G左右。

由于矩阵规模不是很大（最多50\*50），采用lapack和scalapack运行时间差别不明显。