

有哪些适合 cfd 初学者练习的题目? 以及用什么工具求解这些题目?



朱辉

主要针对学 CFD 的高年级本科生 / 研究生。假定已学过两到三门流体力学基础课。从这个基础开始的话以下流程需要一年以上。当然你也可以根据自己的实际情况跳过某些环节。个人觉得与其一上来就开始学 CFD 理论和算法, 不如先把整个 CFD 技术上是怎么操作的走通。这样一来可以明白 big picture 是个啥, 二来学算法的时候也能更有的放矢。

具体建议如下, 与实际流程顺序无严格对应关系:

1. 熟练使用一种网格生成软件, 这样你以后拿到任何复杂的问题都不会卡在网格这个基本环节上了。
2. 从使用 Fluent 开始。什么都能算, 鲁棒性非常好, 而且界面相对比较友好。当然如果你有师兄或者老师给你的 in-house code 更好, 这样有人指导的话使用难度也不会太高, 而且之后学习算法会更容易。现在开源的 CFD 代码也很多, 但不建议单枪匹马去学习使用, 因为上手难度比较高。
3. 至少熟练掌握一门编程语言。即使你一直用 Fluent, 总有一天你也会需要写 UDF 的。matlab 不算编程语言。python 用来做 CFD 不是不可以, 但是一般情况下会很慢, 不太具有实用性。推荐 C/C++/Fortran。编程工具的话, 反正我是用 Visual Studio 和 Emacs 的。各位见仁见智了。

(BTW, 江湖上有大神用 python 写程序,

再自己整一个类似编译器的东西“翻译”成 C, 这种高端技术目测不适合初学者)

4. 至少学会使用一种后处理软件。要不然怎么体现 CFD=ColourFul Drawing 呢。

Tecplot 和 ParaView 是目前最受欢迎的两款后处理软件。

5. 推荐的学习算例:

全是二维和三维的, 直接解 NS/RANS 方程。

那些一维算例之类的可以在学算法的时候回过头来补都来得及。

a.NACA0012。网格非常好画, 熟练的话结构非结构的都可以在 20 分钟内搞定, 而且很多网格生成软件都会拿它当 Tutorial, 对新手来说也可以照猫画虎。可以算的 case 非常多, 从不可压到跨音速都有实验数据。体会一下网格对计算结果的影响。

b. 圆柱绕流, $Re=200$ 。非定常计算。群众喜闻乐见的卡门涡街。

c.DLR-F4 翼身组合体。三维算例。体会一下什么叫面向实际飞行器的 CFD。愿意挑战自己的话可以尝试画结构网格。

6. 现在可以回过头开始学理论和算法了。

如果学校里有 CFD 课当然最好。理想情况是有两学期的 CFD 课 (加起来 6 到 7 学分), 这样现有的理论应该都能讲个大概了。完全自学的话会比较难, 毕竟数学内容还是涉及不少。但至少可以先看 Anderson 的《计算流体力学入门》。此书有中译本, 翻译质量还不错。

最后说一句, 如果想真正学懂 CFD, 一定要自己动手写程序。

次之, 可以看别人的程序, 全看懂了也能学个 80%。

如果只是一直在用商业软件算题的话, 实际上



花半楼

更新，最近统计了一下各部分代码行数，在下文中每个项目中给了补充。

提示：只给出功能部分，测试等部分忽略。

分割线 -----

如果只是需要用软件的，不需要看这个答案。
如果想真正变成一个 CFDer，新手就看过来。

不建议做一些比较散的题目，即使做完了很多时候你也 get 不到重点、路子你也没走通。
建议以目标为导向，系统做几个大题目。我出 3 个题目），如果能做下来，CFD 路子你就通了，剩下的就是修修补补，多学习理论了。
基础：熟练一种编程语言，建议 C++ 或者 Fortran。

题目（这几个题目都是学软件时的基本算例）：

1. 二维翼型网格生成，采用椭圆形控制方程生成结构网格。

第一阶段生成 O 型

第二阶段生成 C 型

第三阶段采用泊松方程（椭圆形方程加入源项），可以控制网格正交性和间距。

选做：采用推进方式（双曲型方程）生成 O 型或者 C 型网格

Fortran 行数：500 行

2. 二维翼型非结构网格生成，采用 Delaunay 方法或者阵面推进法。

基础：掌握 C++ 类或者 Fortran 的结构体、指针、叉树结构、排序算法（最小堆 -- 阵面推进要用到）

Fortran 行数: 5000 行

C++: 5500-6000 行

3. 翼型绕流二维求解器编写。

第一阶段 Euler 方程求解器: 建议对流项离散采用 JST 格式。时间离散可以采用显示 Runge-Kutta 法。upwind 格式和隐式推进较难, 以后可以再完善。

第二阶段 NS 方程求解器: 比 Euler 只多了扩散项, 这一项离散比较简单。

第三阶段 采用 upwind 格式和隐式格式: 现在要求解大型代数方程组了, 当然是 LU-SGS 方法。

第三阶段 湍流模型: 建议采用 SA 模型, 这个是一方程的, 简单。湍流方程和 NS 方程耦合求解。

第四阶段 预处理: 之前的都是适合可压流, 预处理之后适合全速域。

第五阶段 动网格: ~~~~~

Fortran 行数: 到达第四阶段 5000 行

第一题和第三题都给出了不同阶段, 完成第一个阶段与没完成天壤之别。后面每完成一个阶段, 懂的东西多一些, 看自己对自己的要求。第二题比较困难, 没有阶段划分, 不成则败, 慎选。



「已注...

CFD 是一门交叉学科, 内容很多。不同的流动对应的偏微分方程的数学性质迥异, 在学习和练习前最好明确未来的发展方向: 针对可压缩流动还是不可压流动? 是否涉及湍流? 是浮力对流还是管流, 抑或是绕流? 是单相流还是多相流? 不同的方向, 侧重点不同。

不可亚流动的控制方程最后一般归类到 Helmholtz/Poisson 方程，而可压缩流动的控制方程具有双曲性质，和波传播方程、Burgers 方程关联密切。如果涉及湍流，那么数值格式的精度和耗散、色散特性都会是考虑的重点，而且还要考虑湍流模拟的具体技术，DNS/LES/RANS，或者他们的杂交 DES 等技术。内流还是外流，一般也会有不一样针对性考虑，特别是边界条件的设置。多相流动相比一般的单相流，需要了解一些额外的技术，比如 VOF，LevelSet 或者 phase field 等技术。

明确方向后，可以从模型方程和标准 benchmark 开始练习。

一：不可压流动的数值模拟

一维对流扩散方程 $u_t + au_x = \nu u_{xx}$,

柏松方程 $u_{xx} + u_{yy} = f$, 然后是

不可压流动控制方程

$$\begin{aligned} u_i + u_j u_{j,i} + p_{,i} - \nu u_{j,j,i} &= 0, \\ u_{j,j} &= 0. \end{aligned}$$

数值求解不可压缩流动的难点在于其控制方程是椭圆型的，计算域中每个点都是全局依赖。但流场连续没有间断，却便于构造高阶格式，多使用多重网格方法加速收敛。

基本的 benchmark 流动有 lid-driven flow，圆柱绕流，前 / 后台阶流动，side-differentially heated cavity 以及 Rayleigh-Benard 对流等。这些 benchmark 流动一般都是二位的。之后，

局可以进入应用阶段，那真是虎入山林，龙临深渊，天高任鸟飞。

二：可压缩流动数值模拟

一维波传播方程 $u_t + u_x = 0$,

一维对流扩散方程 $u_t + au_x = \nu u_{xx}$,

一维 Burgers 方程 $u_t + uu_x = \nu u_{xx}$,

可压缩 NS 方程（不写了，有点繁琐），一般都采用守恒形式的写法，耦合求解流场解矢量 $V = \{\rho, u, v, w, T\}$, 教科书中很方便查阅。

这些方程的共性是它们都属于双曲 / 抛物型方程，因此可以使用推进方法求解。难点在于，流场可能存在间断，低速时，声波、涡波和熵波速度分离，也给数值求解带来了新的难度。由于对流项是非线性的，只能采用显式格式，否则需要求解大型非线性方程组，代价过于高昂。

上述方程中，Burgers 方程除了没有压力项外，具有了 NS 方程的几乎所有性质，因此很多数值方法都是先通过 Burgers 方程验证其可行性。针对流场的间断特性，可压缩流动的求解一般都采用 Godunov 方法，假设单元内连续，界面间断，通过 Riemann 近似解构造数值通量。不过，无论如何，时间方向总是连续的，所以时间推进可以采用 Taylor 展开构造任意高阶格式。至于涉及到的稳定性问题，就另当别论了。

基本的 benchmark 有各种激波管问题，各种翼型绕流问题（一般选择最简单的对称 NACA0012 翼型，可以选择各种迎角），

bump flow, 激波壁面相互作用 flow past a forward facing step, 等等。

三, 数值格式的选择

从练习的角度看, 有限差分或者有限体积是比较好的选择。这些格式原理非常简单, 容易入门, 有助于掌握数值计算中的基本概念, 比如精度、分辨率、耗散、色散、收敛性、相容性、数值通量, 通量重构等等。另外, 了解弱解和熵解的概念, 有助于理解数值方法的深层原理。

上面的表述, 已经假设题主不是计算数学方向, 而是仅仅把数值计算作为工具, 以物理和工程应用为目标。更加具体, 若面向科研, 差分方法是不错的选择, 原理简单, 也容易扩展至高阶格式; 谱方法次之, 谱方法具有所谓的谱精度。他们具有共同的缺点: 对复杂几何边界的适用型很差, 谱方法由其如此。不过, 结合谱方法和有限元优点的谱元法, 如果感兴趣, 可以尝试。

如果主要是做工程方向, 首推有限体积方法, 原理直观, 复杂几何边界的适应性非常优秀; 有限元及其与其杂交格式次之。有限元方法原理对初学者有一定难度, 不那么直观, 但精度高, 适用任何几何边界。

四, 编程语言的选择

按照优先级次序依次推荐: C++, C, Fortran, Python。他们都有很多的科学库可供调用, 但 C++ 直接提供了类和模板的定义, 便于抽象, 有利于开发和后期维护。小心一些, 速度上也不比 C/Fortran 逊色。C 和 Fortran, 几乎不需要做特别的优化, 留给编译器就可以了。但这两种语言都很老, 尤其是 Fortran, 无法或者很难提供

类和模板抽象，代码规模越大，后期维护会越来越发的困难。Python 语言和自然语言最接近，容易上手。它是解释型语言，速度上有欠缺太多。既然要做 CFD，那肯定是追求计算效率第一。所以，Python 更加适合做 CFD 的接口和框架搭建，搭配 C++/C/Fortran 作内核，参考开源代码 PyFr。

五，大规模代码是自己写还是在已有基础上开发？

很多人想要自己开发，主要是前人的框架太垃圾或者干脆找不到合适的开发框架。近些年，已经出现了很多非常优秀的开源框架。这些都是前人优秀工作的汇总。CFD 是一门技术，更是艺术。很多代码“这么做”都是因为经验如此，很难给出漂亮的理论解释。前人踩过的坑，为什么要自己要再去踩一遍呢？

而且，自己开发程序，风险巨大。如果不幸失败，除了时间和精力的虚耗外，对个人的心理摧残更甚于被发好人卡！当然，如果有足够的激情和兴趣，也做好了准备，那就干吧，多说无益。高风险带来的肯定是高收益，如果成功，也可以算是小有成就，对 CFD 的理解更加深入和确定。

有很多的比较好的 CFD 开源框架，比如 OpenFoam，PyFr 等等，一些常用的库有 Petsc，IMSL，LAPACK 等。做起来后，会发现资源很多，比自己原来想像的多的多。

最后建议，有了练习基础后，选择一个合适的 CFD 开发框架，在此基础上开发属于自己的程序代码。



全文完

本文由 简悦 SimpRead 优化，用以提升阅读体验

使用了 全新的简悦词法分析引擎 ^{beta}，[点击查看详细说明](#)

