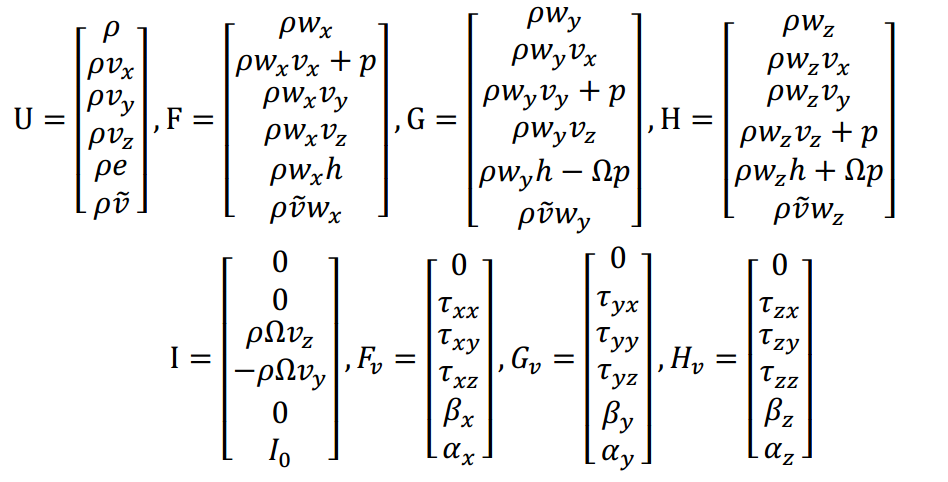
项目程序文档(C版本)

1. 求解模型
2. 对物理模型的理解

模型

上式是在一个控制体内某个物理量的守恒方程，来源于Navier-Stokes控制方程，其中，Q是非定常项，也是求解项，本程序求解答的有压力温度速度，是该物理量在控制体内的变化量，意思是该项随着时间的变化而变化；左端后三项为F、G、H分别是该物理量在x、y、z三个方向上的分量,、、分别是由于流体对流的影响使该物理量在三个方向上产生的变化量，为对流项，对流即流体在流场中密度变化产生浮力驱动流动；右端前三项、、分别是由于流体粘性的影响使该物理量在三个方向上产生的变化量，为扩散项，带v的是粘性，viscosity，流体在扩散过程中收到粘性量的作用，扩散即物质由含量高的地方向含量低的地方传输；I是源项，是流体内部发生变化（如温度变化等）使该物理量产生的变化量。上述等式左端为流体单元的动量变化率，等式右端为各种作用力的结合，包括压力、体积力、粘性力等，在物理上可以理解为牛顿第二定律，由此上式成立。

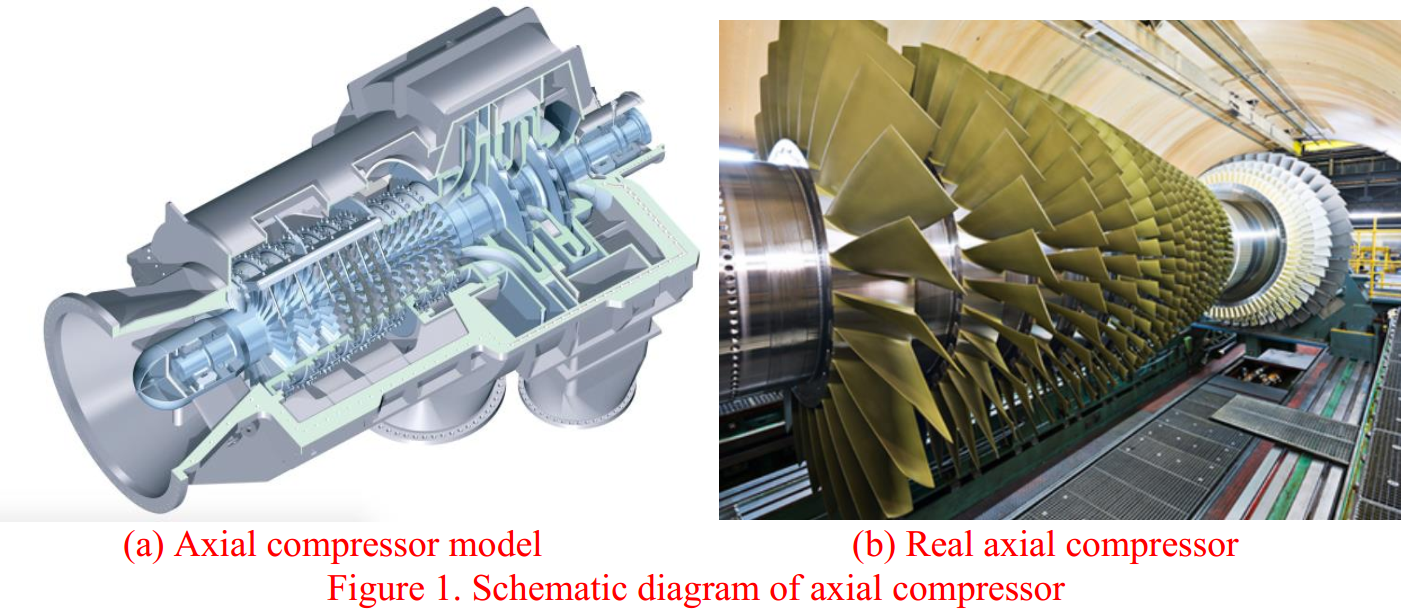
每个向量都有 6 个分量，分别代表着流体控制方程组中的连续性方程、笛卡尔坐标系下三个方向动量分量方程能量方程和 S-A(Spalart-Allmaras)湍流模型方程。

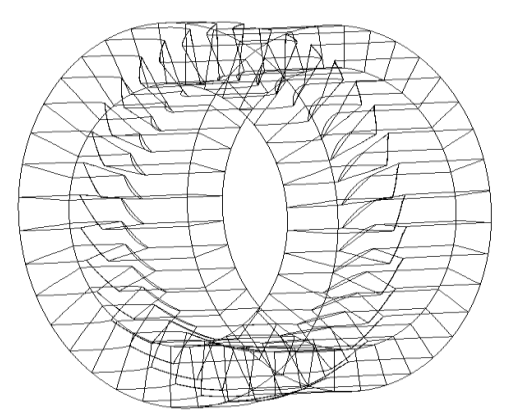
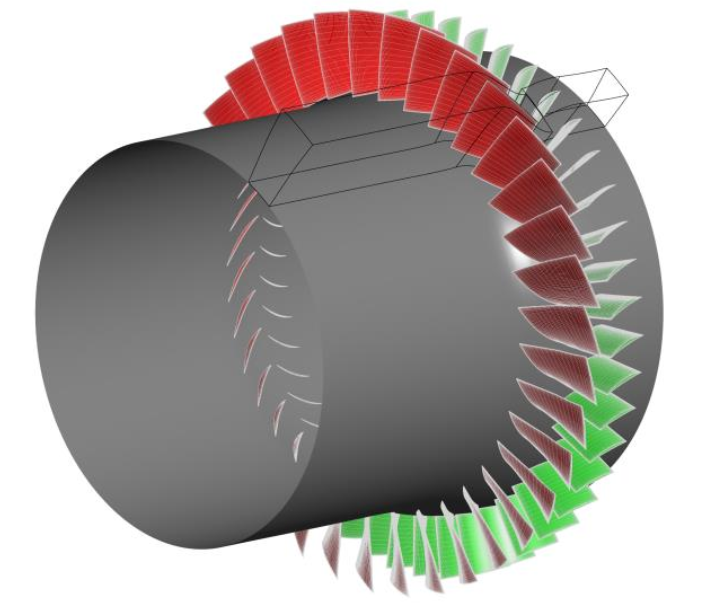


整个迭代是10万分之1秒，将整个空间迭代完成，在每个网格上使用龙格库塔法。

在网格上的求解逻辑是，首先计算速度，由于是非定长，所以需要计算密度，最后计算出压力温度；同时对流通量通过速度计算；粘性通量是速度和密度计算

整个物理模拟的真实场景与模拟示意如图1, 2所示。





1. Stage 35 （b）Rotor 35

图2 模拟示意

1. 相关方法（描述龙格库塔法、多重网格法、多种加速收敛方法的功能与实现）
2. Runge-Kutta法

功能：单步迭代求解微分方程，是一种同时适用于显式或隐式求解非线性常微分方程的迭代法。它将两步之间的误差求解变为 *h* 阶误差求解，使用分段线性来代替一次线性，使得累积的总误差要更小。本文使用的显式龙格-库塔法，4步4阶，4次偏导。

实现：Runge-Kutta算法主要公式如下

此项目中，一步RK迭代如下所示：

同算法中的，为当前迭代值；同算法中的，为时间步长；同算法中的，为RK算法的参数。至于迭代过程，项目中的算法并没有按照RK算法那样的步骤，而是直接利用上述公式，进行3次迭代。其中意味着每次迭代都用新更新的值，而代表这些变量的值只有第一次迭代时才更新。

1. 多重网格法

功能：是一种面向偏微分问题求解的快速迭代方法。传统的迭代法都要求进行方程组迭代的有限体积大小是固定不变的，这意味着如果一个求解域的网格越密，计算量就越大，迭代收敛就越慢。研究表明，低频分量是造成整体缓慢收敛的主要原因，而多重网格法是在同一个几何体的多层网格上交替迭代，

过粗细网格层的光顺与限制，能够有效过滤掉低频分量，提升收敛速度。加速收敛。我们有三层网格，细网格计算太慢，所以引入粗中网格，通过这样可以快速精细的计算出全部网格。

实现：多重网格法求解差分方程

一个最简单的V型多重网格的流程如下所示：

·在细网格上迭代求解得到数值解，残差

·将数值解，残差通过算子限制到粗网格上得到，

·将作为初值迭代求解得到解，并求解误差

·将误差利用算子插值到细网格上得到

·利用对细网格上的解进行修正

·重复上述过程(细网格迭代=>限制=>粗网格迭代=>插值)

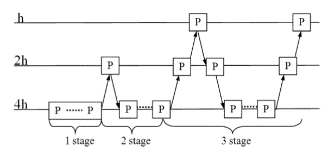
(在上述过程中，限制算子与插值算子均采用体积加权的方式进行设计)

在rotor代码中采用FMG型多重网格方法(如下图所示)，主要步骤为：

·在level3(4h)上迭代计算，直至满足收敛准则

·将数值解插值到level2(2h)上，进行V型循环迭代，直至满足收敛准则

·将计算结果插值到level1(h)上，循环往复，直至满足收敛准则



1. 当地时间步长

功能：通过改变时间步长来加速收敛。

实现：通过公式

来确定时间步长，其中tc与td都是通过网格的几何划分来确定的。

1. 隐式残差光顺

功能：快速抹平残差，扩大时间步长的稳定性范围

实现：将节点处和邻近节点处的残差做加权平均作为新残差值

1. 有限体积法（Finite Volume Method， FVM）

功能：是计算流体动力学中最常用的一种数值算法。

实现：它把计算网格定义为一个个控制体，使用积分形式的守恒方程来描述每个控制体中物质和能量的流入、流出情况。有限体积法着重从物理观点来构造离散方程，每一个离散方程都是在有限大小的网格上描述某种物理量的守恒表示式。其中，离散方程的系数往往都具有一定的物理意义, 使得推导过程清晰易懂, 并且保证离散方程的物理量守恒，是流体机械仿真中最常用的方法。

1. 对求解方法的理解（如何把物理模型、相关方法和计算机联系到一起，描述其中数值分析的思想）

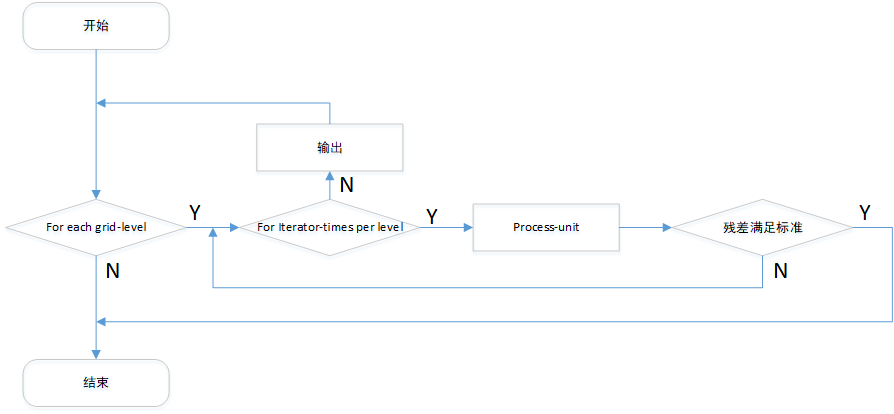
物理模型本质上就是一系列的微分方程，求解物理模型就是求解微分方程的过程，采用有限体积法对物理模型进行离散，采用Runge-Kutta方法这种数值求解方法进行迭代求解，以便计算机求解，再采用当地时间步长、隐式残差光顺以及多重网格等方法来加速迭代并减小误差。

1. 程序流程图

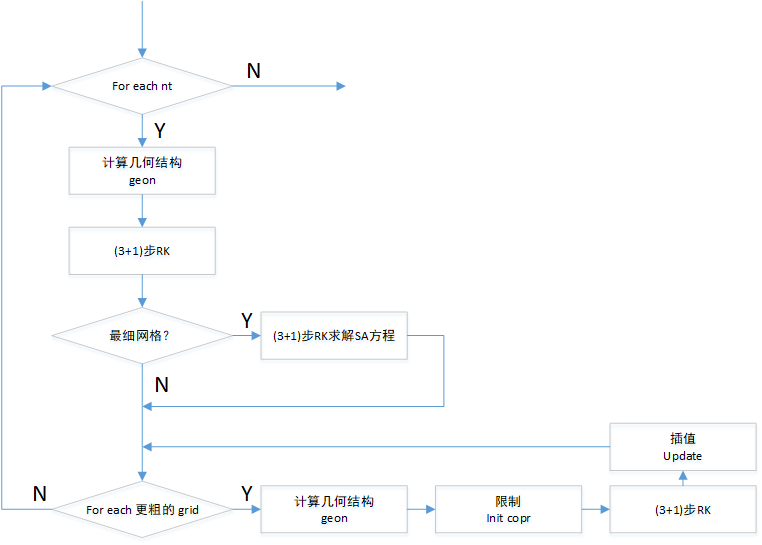
rotor程序流程图如下所示，其中包括各种参数数据的初始化以及多重网格FMG具体求解过程：



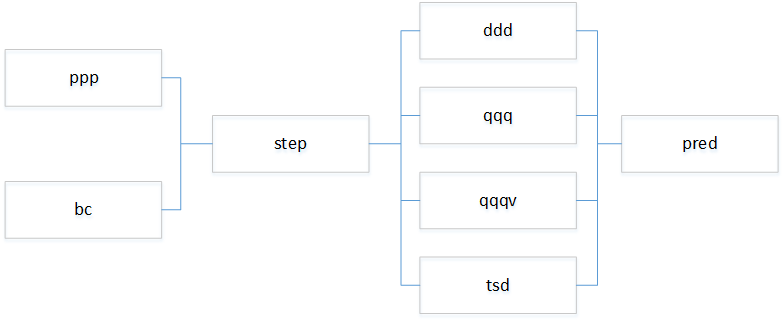
其中FMG求解过程流程如下图所示，表示在每个网格层迭代相应的次数进行运算：



上图中的处理单元process-unit的处理流程如下图所示，其表示计算时先获取该层网格的相关数据，然后在该层网格上进行3+1步RK算法，其中S-A模型计算只在最细网格上进行。处理完该层网格之后，依次处理比该层网格更粗的网格层。

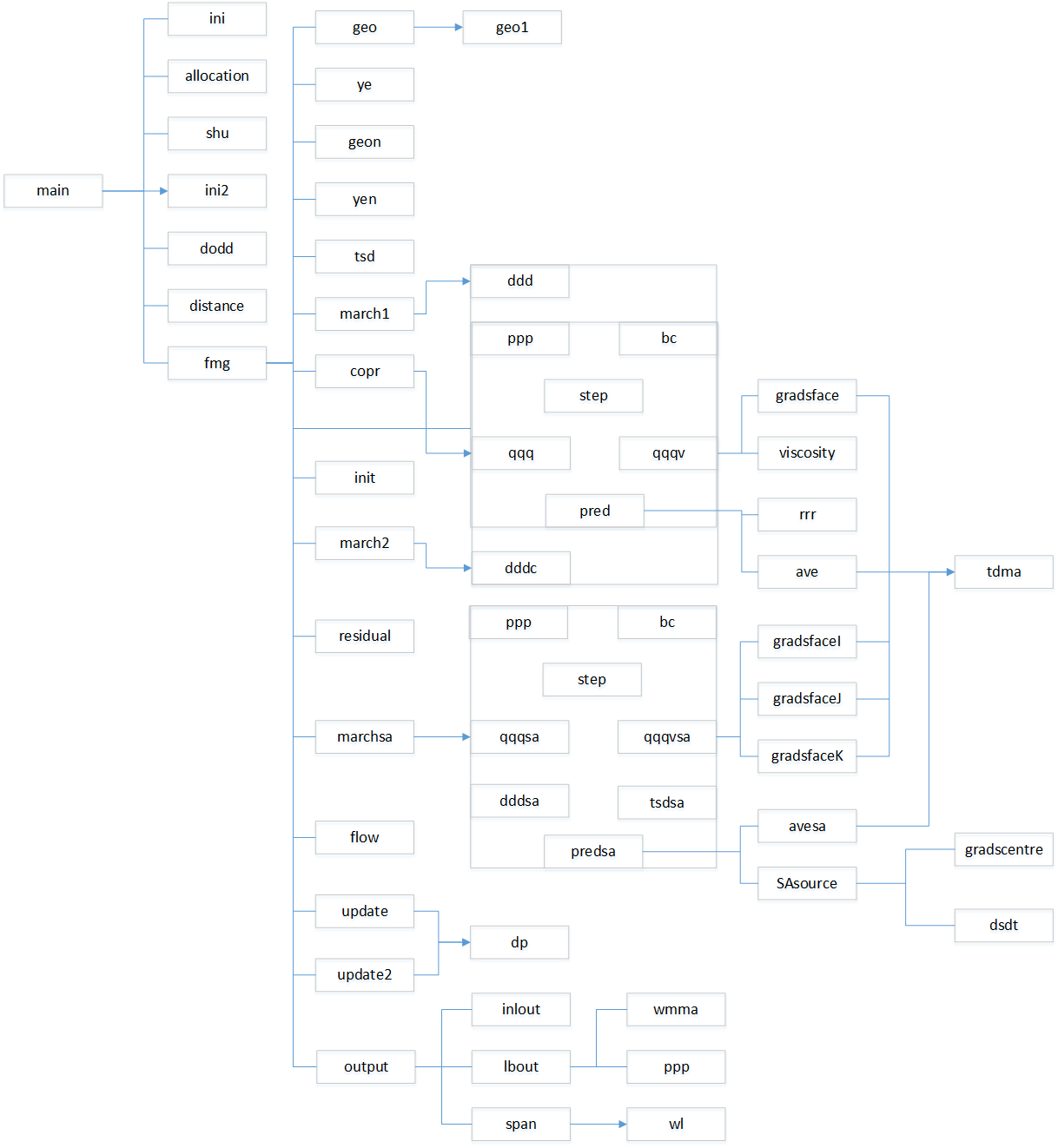


上图中最核心的计算是RK计算，其中一次RK计算过程如下图所示，一步RK计算式子为，图中的ppp与bc分别代表内部与边界的计算，step计算当地时间步长 ，ddd计算人工粘性项D，qqq计算对流项，qqqv计算粘性项，tsd计算源项I，pred为计算上述RK式子的过程。另外，人工粘性项ddd与粘性项qqqv以及源项tsd只在第一步计算，因此有些RK计算过程中没有这几项。S-A模型的计算也是如下过程，只是相应函数名变为xxxsa(eg. qqq->qqqsa)。



1. 各函数的调用关系与层次
2. C代码（函数调用关系）

函数调用关系图如下所示：



有部分函数(eg. ppp)被多个函数调用，但为了关系图的简洁，这部分函数在图中会重复出现。另外，为了关系图的美观，fmg函数的下层调用关系没有按照调用的顺序来排列，而是做了一些改动。

1. Cpp代码（类的继承与组合）
2. 框架图与代码的映射
3. 描述各函数的作用及实现（C/CPP）

|  |  |  |
| --- | --- | --- |
| 函数名 | 作用 | 实现 |
| main | 主函数 | 主要控制流程包括参数初始化与FMG多重网格求解，并使用MPI进行通信。 |
| ini | 读入控制参数与网格数据 | 直接赋值+读文件赋值 |
| allocation | 给变量分配内存空间 | 给代码中的数组用resize分配空间 |
| shu | 设置多重网格法的相关参数 | 以2的指数形式设置每层网格数等参数 |
| ini2 | 在最粗网格上初始化流场 | 为一些变量赋初值 |
| dodd | 时间谱系数 | 求解时间导数项时用到 |
| distance | 计算最细网格上到壁面的最短距离 | 求解SA源项时会用到 |
| fmg | 多重网格求解 | 具体实现流程见二中的程序流程图与1.2.2中多重网格法的图  在实现过程中，本代码只实现了多重网格图中细网格到粗网格的限制过程(h=>2h=>4h)，在粗网格中计算过后没有向上一层插值，而是直接对最细网格插值修正，在当前阶段迭代完成后直接回到最细网格层计算，如此循环往复。 |
| geo | 计算网格参数 | 将顶点坐标映射到当前网格(当前网格是此阶段迭代的最细网格，因此省去for循环)，调用geo1由顶点坐标计算面、体中心坐标等参数。再将这些网格参数映射到更粗的网格层(即所有网格层)。 |
| geo1 | 计算网格参数 | 由网格顶点坐标计算面、体中心等坐标。 |
| geon(nng) | 计算具体nng层网格参数 | 从存放所有网格数据的结构中取出来nng层数据放入当前网格层数据结构中 |
| ye | 边界值 | 计算当前网格的边界值，包括betax、betay、betaz、hat、pet(当前网格是此阶段迭代的最细网格，因此省去for循环)。 |
| yen(nnng) | nnng层网格边界值 | 从存放所有边界值的数据结构中取出nnng层的数据放入当前边界值数据结构中。 |
| tsd | 时间导数项计算 | 根据公式计算时间导数项ts，其中dm为前面dodd函数中求得。 |
| march1 | 3步RK推进 | 包含3步RK计算过程。RK计算流程中包含ppp、bc、step、ddd、qqq、qqqv、pred几步，如二中的程序流程图所示。其中step、qqqv均只在第一步计算，减少计算时间。并且在奇数步上pred参数为1，表示进行隐式残差光顺，而在偶数步上参数为0，表示不进行隐式残差光顺。 |
| residual | 计算残差 | 用上式计算残差，log函数是为了避免求平方根。 |
| ppp | 由守恒量计算原始量 | 计算原始量，主要包括vth、vre、p、t、q16等 |
| bc | 边界条件 | 计算边界值，利用MPI进行通信，更新边界值 |
| step | 当地时间步长计算 | 当地时间步长=，其中， |
| ddd | 人工粘性计算 | 分x,y,z三个方向分别计算，利用上式计算出x方向上的i-1与i之间的人工粘性量flu，并使i-1单元减去此人工粘性量flu，i单元加上此人工粘性量flu，意味着人工粘性量flu从i-1单元流向i单元。y、z方向同理。 |
| qqq | 对流通量计算 | 分x,y,z三个方向分别计算，根据相对流动控制方程组计算出x方向上的i-1与i之间的对流量flu，并使i-1单元减去此对流量flu，i单元加上此对流量flu，意味着对流量flu从i-1单元流向i单元。y、z方向同理。 |
| qqqv | 粘性通量计算 | 分x,y,z三个方向分别计算，根据相对流动控制方程组计算出x方向上的i-1与i之间的粘性量flu，并使i-1单元减去此此粘性量flu，i单元加上此粘性量flu，意味着粘性量flu从i-1单元流向i单元。y、z方向同理。 |
| pred(0/1) | 一步RK计算 | 根据公式进行一步RK推进，其中主要计算部分封装在了rrr函数中，隐式残差光顺部分封装在了ave函数中，pred函数主体只进行了对两个函数的计算结果结合来更新U。该函数有参数(0/1)，参数为0表示不进行ave函数计算，参数为1表示进行ave函数计算。 |
| marchsa | S-A湍流方程计算 | 同march1，包含4步RK推进。march1中计算方程中前五项(q01~q05)，而此函数中计算后一项(q06)。部分调用的函数改为计算湍流方程计算函数(eg. ddd->dddsa)。 |
| tsdsa | S-A湍流方程时间导数项计算 | 同tsd，计算SA方程的时间导数项。tsd中计算ts1~ts5，而此函数中计算ts6。 |
| dddsa | S-A湍流方程人工粘性计算 | 同ddd，计算S-A湍流方程人工粘性量。ddd中计算av1~av5，而此函数中计算av6。 |
| qqqsa | S-A湍流方程对流通量计算 | 同qqq，计算S-A湍流方程对流通量。qqq中计算qc1~qc5，而此函数中计算qc6。 |
| qqqvsa | S-A湍流方程粘性通量计算 | 同qqqv，计算S-A湍流方程粘性通量。qqqv中计算qv1~qv5，而此函数中计算qv6。 |
| predsa(0/1) | S-A湍流方程一步RK推进 | 同pred(0/1)，计算S-A湍流方程一步RK推进。和pred不同的是，在此函数中没有调用rrr函数，而是直接在该函数中进行公式的计算，而且去掉了驱动源项(因为该函数只在最细网格上计算)，并且相对应的进行隐式残差光顺的函数改为了avesa。 |
| avesa | S-A湍流方程上的隐式残差光顺 | 同ave，进行S-A湍流方程中的隐式残差光顺。ave中计算py1~py5，而此函数中计算py6。 |
| init | 限制算子 | 使用基于中心格式的有限体积法，采用体积加权的方式将细网格上的求解变量以及残差值限制到粗网格上，在此函数中是将8个相邻的小网格采用体积加权的方式限制到大网格上。 |
| copr | 粗网格上驱动源项计算 | 先利用除pred以外的所有函数进行一步RK推进(由于现在是粗网格，无需进行SA方程计算；另外人工粘性项由函数dddc计算)，得到qc、av、ts、qv等值，再根据init函数中限制得到的残差值rr，根据计算驱动源项。 |
| march2 | 粗网格上的3步RK推进 | 同march1，进行粗网格上的3步RK推进。同march1不同的是，该函数中人工粘性项采用dddc函数(只进行2阶项求解)。 |
| dddc | 粗网格上的人工粘性项计算 | 同ddd，计算粗网格上的人工粘性项。比ddd函数少计算了4阶项。 |
| update | 粗网格上的值插值到细网格上 | 利用插值算子将粗网格上的值插值到细网格上并计算修正量py，利用修正量对细网格上的值进行修正。 |
| dp | 插值算子 | 对参数a,b,c，返回值 |
| update2 | 插值跳阶段 | 在FMG每个阶段之间执行一次 |
| flow | 进出口流量计算 | 计算进出口流量，用MPI通信进行汇总，写到文件里 |
| viscosity | 求解动力粘度系数和导热系数 |  |
| gradsface | 界面导数值计算 | 调用gradsfaceI、gradsfaceJ、gradsfaceK |
| gradsfaceI | I方向界面导数值 | 计算gradfi |
| gradsfaceJ | J方向界面导数值 | 计算gradfj |
| gradsfaceK | K方向界面导数值 | 计算gradfk |
| rrr | 计算残差 | 根据式子计算残差rr，同时也是一步RK推进的主体部分。其中是驱动源项，在计算细网格时初始化为0，在向粗网格限制时计算。 |
| ave | 隐式残差光顺 | 根据式子，分别对x,y,z三个方向进行隐式残差光顺，计算过程中会要求解三对角矩阵方程组，调用tdma函数进行求解。 |
| tdma | 求解三对角矩阵 | 方程组的求解过程： |
| SAsource | 求网格中心的方程六的源项 | 根据计算SA湍流方程源项。其中需要求解一些偏微分的值，需要调用gradscentre和dsdt函数。 |
| gradscentre | 网格中心梯度计算 | 给定变量i和方向j，计算梯度 |
| dsdt | 旋转因子中的参数 | 在式子中计算其中的 |
| output | 输出 | 调用in1out、lbout、span函数进行文件输出 |
| in1out | 输出进口流量 | 计算要输出的量，通过MPI传递给0号进程，0号进程进行文件输出。 |
| lbout | 输出叶片计算结果 | 文件输出计算结果 |
| span(spa) | 输出沿叶高结果 | 文件输出计算结果 |
| wmma | 计算相对马赫数 |  |
| wl | 三次样条插值 | 在span中调用，用以计算要输出的值 |

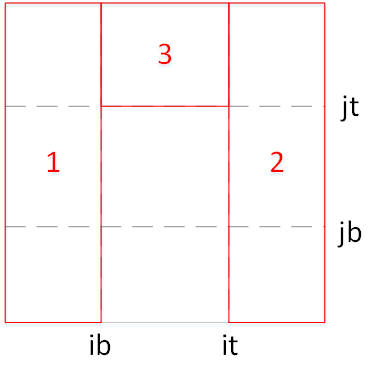
1. 描述各变量的作用(C/CPP)

|  |  |  |
| --- | --- | --- |
| 变量 | 类型 | 作用 |
| nb | Int | NULL |
| nt | Int | 物理时间步数 |
| ng | Int | 网格层数，例如在1.2.2多重网格图中为3 |
| nxm1,nym1  nzm1 | Int | 三个方向的网格数 |
| nng0 | Int | 值为1，表示最粗网格层数 |
| nitt | Int | 总迭代次数 |
| nng | Int | 当前多重网格迭代阶段最细网格层数，例如在1.2.2多重网格图中stage1~stage3分别对应1~3 |
| ign | Int | 在当前多重网格迭代阶段中当前计算的网格层，例如在1.2.2多重网格图中，假如当前处于stage2/3阶段的向下限制的阶段时，ign就代表当前计算的网格层 |
| it00 | Int | 在当前多重网格迭代阶段中的已迭代次数 |
| lbb | Int | 通道数量 |
| Nxm,nym,nzm | Int | 三个方向的网格数 |
| Ibm,itm  Jbm,jtm | Int | 初始化流场时网格层的叶片处网格数 |
| Cg | Int[] | 每层迭代步数，例如在1.2.2多重网格的图中表示stage1~stage3三个阶段的迭代次数。 |
| nx,ny,nz | Int | 给定网格层的三个方向上的网格数 |
| ib,it,jb,jt | Int | 给定网格层的叶片处的网格数 |
| n | Int | 物理时间步数迭代下标 |
| ci | Int | NULL |
| cycle\_by | Int | NULL |
| Vxx | Double | 求气流角时所用变量 |
| Vrr | Double | 求气流角时所用变量 vrr = vxx\*beta2 |
| Vtt | Double | 求气流角时所用变量 vtt = vxx\*beta1 |
| Vee | Double | 求气流角时所用变量 vee = |vxx,vrr,vtt| |
| Y1 | Double | 给定网格的yy0 |
| Z1 | Double | 给定网格的zz0 |
| Rr | Double | Rr=|y1-z1| |
| vx,vy,vz | Double | 三个方向上的绝对速度分量 |
| wx,wy,wz | Double | 三个方向上的相对速度分量 |
| Ama | Double | NULL |
| Dim | Double | 密度 |
| En | Double | 原始守恒量第五项 |
| Pp | Double | 压力p |
| Rpm | Double | 转速 |
| Ma | Double | 参数，用于初始化流场数据 |
| sir,cor | Double | 求气流角betax\y\zn时用到 |
| C2 | Double | 在利用湍流衰减函数收敛湍流运动粘度时的一个参数 |
| Cfl | Double | Courant数，计算当地时间步长时使用 |
| a2,a4 | Double | 用于计算人工粘性的参数 |
| beta1,beta2 | Double | 初始化流场时计算气流角所用参数 |
| Time\_begin  Time\_end  Time\_t | Double | Time\_begin与time\_end分别记录当前进程计算过程前后的时间  Time\_t在代码中未使用 |
| pi | Double |  |
| Nnx,Nny,Nnz | Int[] | 长度为ng，每层网格的三个方向的网格数 |
| Nib,nit  Njb,njt | Int[] | 长度为ng，每层网格的叶片处的网格数 |
| Q01~Q06 | Double[][][] | 存放上一次迭代Q11~Q16的值 |
| Q31~Q36 | Double[][][][] | 在一次迭代时，迭代完最细网格层后用此变量暂存Q11~Q16的值，在更粗网格层计算时对其进行插值修正，在一次迭代完成后再存回Q11~Q16，代表完成插值操作 |
| Q11~Q16 | Double[][][][] | 原始守恒量，可近似看作 |
| av1~av6 | Double[][][] | 人工粘性量，RK推进时用 |
| qc1~qc6 | Double[][][] | 对流量，RK推进时用 |
| qv1~qv6 | Double[][][] | 粘性量，RK推进时用 |
| ts1~ts6 | Double[][][] | 时间导数项，RK推进时用 |
| py1~py6 | Double[][][] | RK推进时计算出的修正量，可在插值时使用 |
| rr1~rr5 | Double[][][] | 残差值 |
| qp1~qp5 | Double[][][] | 驱动源项 |
| Gradfi  Gradfj  Gradfk | Double[][][][] | 16\*(nz+2)\*(ny+2)\*(nx+2)  Gradfi：  Gradfj、Gradfk同理，分别为i、j、k三个方向，求解粘性项时使用，交界面流动参量的导数 |
| Gradc,Gradcs | Double[][][][] | 13\*(nz+2)\*(ny+2)\*(nx+2)  Gradc:  10\*(nz+2)\*(ny+2)\*(nx+2)  Gradcs同理，求解SA源项时使用 |
| xf,yf,zf | Double[][][] | (nzm+2)\*(nym+2)\*(nxm+2) 读入的网格数据坐标 |
| x,y,z | Double[][][] | (nz+2)\*(ny+2)\*(nx+2) 网格顶点处坐标 |
| xx00,yy00,zz00 | Double[][][] | (nz+1)\*(ny+1)\*(nx+1) 网格中心点坐标 |
| Xx1,Yy1,Zz1 | Double[][][][] | (ng+1)\*(nz+2)\*(ny+1)\*(nx+1) 网格面中心值(周向) |
| Xx2,Yy2,Zz2 | Double[][][][] | (ng+1)\*(nz+1)\*(ny+1)\*(nx+2) 网格面中心值(轴向) |
| Xx3,Yy3,Zz3 | Double[][][][] | (ng+1)\*(nz+1)\*(ny+2)\*(nx+1) 网格面中心值(展向) |
| Xx,Yy,Zz | Double[][][][] | (ng+1)\*(nz+1)\*(ny+1)\*(nx+1) 网格体中心值 |
| Xx01,Yy01,Zz01 | Double[][][] | (nz+2)\*(ny+1)\*(nx+1) 给定网格层网格面中心值(周向) |
| Xx02,Yy02,Zz02 | Double[][][] | (nz+1)\*(ny+1)\*(nx+2) 给定网格层网格面中心值(轴向) |
| Xx03,Yy03,Zz03 | Double[][][] | (nz+1)\*(ny+2)\*(nx+1) 给定网格层网格面中心值(展向) |
| Xx0,Yy0,Zz0 | Double[][][] | (nz+1)\*(ny+1)\*(nx+1) 给定网格层网格体中心值 |
| S1xn,S1yn,S1zn | Double[][][][] | (ng+1)\*(nz+2)\*(ny+1)\*(nx+1) 沿z方向三个面的面积 |
| S2xn,S2yn,S2zn | Double[][][][] | (ng+1)\*(nz+1)\*(ny+1)\*(nx+2) 沿x方向三个面的面积 |
| S3xn,S3yn,S3zn | Double[][][][] | (ng+1)\*(nz+1)\*(ny+2)\*(nx+1) 沿y方向三个面的面积 |
| vvn | Double[][][][] | (ng+1)\*(nz+1)\*(ny+1)\*(nx+1) 网格体积 |
| S1x,S1y,S1z | Double[][][] | (nz+3)\*(ny+1)\*(nx+1) 给定网格层的s1xn~s1zn |
| S2x,S2y,S2z | Double[][][] | (nz+1)\*(ny+1)\*(nx+3) 给定网格层的s2xn~s2zn |
| S3x,S3y,S3z | Double[][][] | (nz+1)\*(ny+3)\*(nx+1) 给定网格层的s3xn~s3zn |
| vv | Double[][][] | (nz+1)\*(ny+1)\*(nx+1) 给定网格层网格体积 |
| Pvx,pvy,pvz | Double[][][] | nz\*ny\*nx 三个方向的绝对速度分量 |
| Vth,vre | Double[][][] | (nz+2)\*(ny+2)\*(nx+2) 周向速度、径向速度 |
| P | Double[][][] | 压力，nz\*ny\*nx |
| T | Double[][][] | 温度，nz\*ny\*nx |
| Time | Double[][][] | 存放tc+td，用于求解当地时间步长，nz\*ny\*nx |
| wma | Double[][][] | 马赫数，nz\*ny\*nx |
| Dmini | Double[][][] | (nz+1)\*(ny+1)\*(nx+1) 体中心点到固壁最短距离，即湍流模型耗散源项公式中的d |
| Sri,srj,srk | Double[][][] | (nz+2)\*(ny+2)\*(nx+2) 用来计算人工粘性项中的谱半径 |
| Dm | Double[][] | 时间谱的加权因子，长度为nt+1 |
| rms | Double[] | 长度为nt，存储当前进程中每步时间步迭代的残差值 |
| Betaxn  Betayn  Betazn | Double[][][][] | nt\*ng\*nz\*ny 三个方向的气流角beta |
| Petn | Double[][][][] | nt\*ng\*nz\*ny 进口总压 |
| Pebn | Double[][][][] | nt\*ng\*nz\*ny 出口背压 |
| Turin | Double[][][][] | nt\*ng\*nz\*ny 涡粘 |
| Hatn | Double[][][][] | nt\*ng\*nz\*ny 进口总焓 |
| Betax,Betay  Betaz | Double[][] | nz\*ny 确定物理时间步与网格层的三个方向的气流角beta |
| Pet | Double[][] | nz\*ny 确定物理时间步与网格层的进口总压 |
| Peb | Double[][] | nz\*ny 确定物理时间步与网格层的出口背压 |
| Turi | Double[][] | nz\*ny 确定物理时间步与网格层的涡粘 |
| Hat | Double[][] | nz\*ny 确定物理时间步与网格层的进口总焓 |
| Ta | Double | 用于隐式残差光顺的参数(三对角矩阵的参数) |
| Timl | Double | 用于求解当地时间步长的参数 |
| Pt | Double | 进口压 |
| Ht | Double | 进口焓 |
| Rout | Double | 初始化流场时的一个中间变量 |
| Pb0 | Double | NULL |
| Pb1 | Double | 出口压 |
| Period | Double | 周期 |
| Rmsm | Double | 某一进程当前的残差 |
| Rmsm0 | Double | 所设定的残差阈值 |
| rmsmmax | Double | 所有MPI进程求得的残差的最大值 |
| t0,ts,cvl0 | Double | 参数，用以计算 |
| cp,prl,prt | Double | 参数，用以计算热导率 |
| Rg | Double | 中的R |
| Cv1 | Double | 参数，用以计算 |
| Cv2 | Double | 参数，用以计算 |
| Kap | Double | 冯·卡门常数 |
| Sigmav,cb2 | Double | 参数，用以计算 |
| Cb1 | Double | 参数，用以计算 |
| Cw1 | Double | 参数，用以计算 |
| Cw2 | Double | 参数.，用以计算 |
| Cw3 | Double | 参数，用以计算 |
| cr1,cr2,cr3 | Double | 参数，用以计算 |
| Id\_m | string | Myid字符串形式 |
| F | Fstream | 处理文件error-+id\_m+myid.dat |
| F1 | Fstream | 处理文件convergence-inflow.dat |
| F2 | Fstream | 处理文件convergence-outflow.dat |
| Myid | Int | 进程号 |
| Myidl | Int | 上一个进程号 |
| Myidr | Int | 下一个进程号 |
| Numprocs | Int | 进程数 |
| Ierr | Int | NULL |
| Rc | Int | NULL |
| status | MPI\_Status | MPI通信时的参数 |

1. 通信模块bc的详细理解(C/CPP)

函数bc主要进行边界值的计算与通信，包含下面3个步骤：

1. 三个方向六个面上计算边界值，包括q11~q16,pvx~pvz,p,t
2. 分三块(1,2,3)两个方向(my\_idl , my\_idr)共六组与相邻进程通信，如下图所示。



1. 通信完成后对边界值进行更新