第五节 数值方法新技术

一、 概述

本章介绍数值方法中的新技术，多重网格法(MultiGrid Method)和区域分解法(Domain Decomposition Method)。其中多重网格法已有厂商发布相应的开发包，区域分解法尚未见到。

二、 多重网格法介绍

1、 多重网格法(MultiGrid Method)是偏微分常数值求解中的一种新技术，是对传统的定常迭代法的改进，是近 20 年发展起来的方法，且逐步走向成熟。本节对多重网格方法介绍，提供了一个 C++的展示程序与传统的定常迭代法比较。
2、 多重网格法的简单历史。多重网格法的思想在 30 年代就有人提出，但真正广泛应用于工程技术问题是 1979 年布朗特(Brandt)教授发表“边值问题多重网格适应解”开始。如今，多重网格法已广泛应用于各种工程技术问题，特别是流体力学中，目前已经有[1]的详细介绍
3、 多重网格法的优势。已从理论上证明至少对线性椭圆形问题是一种最优化的数值方法，其计算量仅与网格的节点数的一次方成正比，且收敛速度与网格的尺度大小无关，从而特别适合于超大型工程的数值计算问题。在具体计算时，可以将计算程序的速度提高 1-2 个数量级，使原来大型计算机不能胜任的数值模拟问题可以得到解决[1]。
4、 多重网格法方法的基本思想。是对亏损量(defective number)，也可称为残差，在不同的网格上进行校正，从而加速迭代的收敛。亏损量即$\mathbf{b}-\mathbf{Ax}$(其中$\mathbf{x}$是每步迭代产生的数值解)，用于描述的是每步迭代的解在经过矩阵$\mathbf{A}$变换后距离$\mathbf{b}$还有多远，是一个向量，如果此向量为 0，则解是真实解，但实际的迭代过程中，每次迭代都更接近真实解，因此残差不总是为 0。通过对此亏损量的分析，发现迭代后，出现高频分量衰减较快，低频分量衰减较慢的情况，再进一步分析，高频分量来自于网格的细分，低频分量来自网格的边界。为提高低频分量的衰减速度，可以人为重新划分网格，即将原离散的长度调整成不同的大小，原来在细网格上的低频分量在粗网格下就可以变成高频分量，这样，在不同的网格下，原来衰减速度慢的低频分量在另一个网格下就能快速衰减，通过这样的方法，迭代过程中的解距离真实解的偏差能否迅速减小，多重网格法的思想就是如此。
5、 多重网格法起源于对物理模型不同精度离散后的求解。即根据物理模型的方程，不同网格的划分就有不同大小的线性方程组，因此任何多重网格不同精度的划分严格对应到不同精度的物理模型，如果对原求解区域降低划分网格的密度(网格更粗)，则方程组的维数更低；但对于任意线性方程组，可能没有对应的物理模型，如何降低线性方程组的维度就成为代数多重网格法。

三、 多重网格法原理

1. 迭代法的收敛性分析。迭代过程中解的收敛性分析在文献[1][6]有较详细的分析。
2. 计算流程图。见图 6-1 所示，其中细网格上的矩阵，解，已知向量，亏损量分别为$\mathbf{A}$，$\mathbf{x}$，$\mathbf{b}$，$\mathbf{r}$；粗网格上的矩阵，解，已知向量，亏损量为$\mathbf{A2}$，$\mathbf{x2}$，$\mathbf{b2}$，$\mathbf{r2}$。具体

计算的流程为：(1)使用迭代法求解细网格，得到一个近似解**x**，计算亏损量**r**；(2)将亏损量粗化后，作为粗网格上的向量已知向量 **b2**，(3)在粗网格上求解得到**x2**，在插值成细网格上的修正量，用来修正细网格上的**x**，(4)在细网格上迭代求解，实际上就是加速了细网格上的迭代，让迭代过程中的解更快速接近真实解，(5)检查解的精度，如不满足则再次迭代。

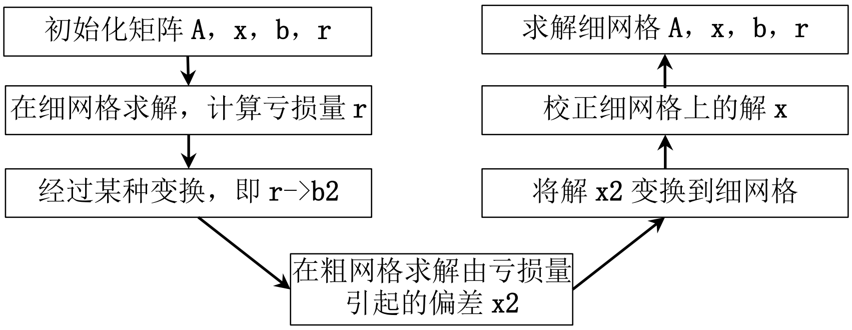


图 6-1

3. 多重网格在各层之间加速收敛的组织结构。一般有以下几种：V-cycle，W-cycle，Full Multigrid。见下图 6-2。实例代码中使用三层网格的 V-cvcel 方式进行迭代求解。此外各个网格层之间传递数据的算子也是一个重要内容。

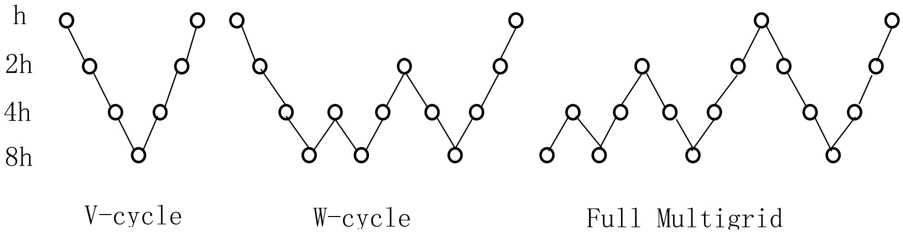


图 6-2

四、 多重网格法的实例代码讲解

1、 工程实例有 5 个模块，其中新增一个多重网格法的实现模块，其余为矩阵计算的模块。(1)CMGM.cpp 是多重网格法的实现模块。(2)Main.cpp 是实例程序的入口；(3)CMatrix.cpp 实现矩阵和向量类；(4)CMatrixCaculate.cpp 实现矩阵和向量之间的计算；(5)Solver.cpp 实现几种求解算法。
2、 核心模块的讲解。CMGM.cpp 实现了多重网格法的计算，具体步骤如下，(1)初始化各层网格的矩阵数据(本例子使用三层网格，名字分别称为细网格，中网格，粗网格)；然后在各个层之间计算，并通过函数传递各层之间需要的数据。
3、 网格到网格之间的数据传递。细网格计算得到的数据转换成粗网格使用，函数为 MeshhToMesh2h()完成，粗网格计算得到的数据转换成粗网格使用 Mesh2hToMeshh()函数。
4、 求解过程的迭代部分代码如下：其中每层的迭代法中使用 Gauss-Seidel 进行迭代，具体的执行过程请参见程序代码。

```
bool CMGM::Iteration( CVECTOR* pcOut , int iCount , float fErr )
  {
    for( int i = 0 ; i < iCount ; i = i + 1 )
      {
        // 在细网格上求解，获取亏损量，并将残差转换到中网格，具体就是将细网格残差平滑成
        // 作为中网格的已知量 b
```

```
        ResloveGS( &cm_vxh , &cm_Matrixh , &cm_vbh );
        GetResidual( &cm_vrh , &cm_Matrixh , &cm_vxh , &cm_vbh );
        MeshhToMesh2h( &cm_vb2h , &cm_vrh ) ;
        // 使用细网格计算得到的 b 在中网格中计算，再转换到粗网格，称为粗网格的已知量 b
        ResloveGS( &cm_vx2h , &cm_Matrix2h , &cm_vb2h );
        GetResidual( &cm_vr2h , &cm_Matrix2h , &cm_vx2h , &cm_vb2h );
        MeshhToMesh2h( &cm_vb4h , &cm_vr2h );
        // 在粗网格中计算解，在插值到中网格，在来校正原来中网格的解
        ResloveGS( &cm_vx4h , &cm_Matrix4h , &cm_vb4h );
        Mesh2hToMeshh( &cm_vc2h , &cm_vx4h );
        VectorAdd( &cm_vx2h , &cm_vx2h , &cm_vc2h , 1 );
        // 在中网格中求解，在插值到细网格，在来校正原细网格的解
        ResloveGS( &cm_vx2h , &cm_Matrix2h , &cm_vb2h ) ;
        Mesh2hToMeshh( &cm_vch , &cm_vx2h );
        VectorAdd( &cm_vxh , &cm_vxh , &cm_vch , 1 );
        // 精度检查
        float fNormal = GetNormal( &cm_vxh );
        if( fNormal < fErr ) break;
      }
    for( int i = 0 ; i < DIMH ; i = i + 1 ) // 将计算结果传递出函数
      {
        pcOut->SetElement( i , cm_vxh.GetData( i ));
      }
    return true;
  }
```

5、 计算结果比较。程序中同时使用了 Gauss-Seidel 求解同样的方程，在同样的次数下，多重网格法有更高的收敛速度，且在粗网格下计算量相对细网格更少。

五、 实例代码的一些思考

1. 网格粗化和细化。粗细网格之间亏损量的传递方式。在本实例的代码中，各个网格层之间的数据转换基本都是以代数平均值的方式传递。即细网格的亏损量在按相邻平均后作为粗网格的亏损量，粗网格的亏损量则平分后作为细网格的亏损量，是否有更有效的方法分配方法也是一个研究方向。
2. 迭代次数。各层循环次数。在代码中，多重网格法调用了 Gauss-Seidel 方法求解各个网格层上的值，Gauss-Seidel 内部迭代了 5 次，在多重网格法的循环中调用了 4 次 Gauss-Seidel 方法，迭代循环了 10 次，总共循环了 200 次。如何分配循环次数是一个改进方向。

六、 代数多重网格法和 GPU 上的运用

1、 代数多重网格(Algebraic Multigrid)简称为 AGM。在上述例子中，三个网格的矩阵的已经作为常量保存在代码中，此矩阵是根据具体的物理问题在离散过程中生成，但对于其他非物理问题的线性方程组，各个层次的网格的生成过程就无具体依据，因此通过某种方法生成矩阵，这种方法称之为代数多重网格方法，而相对上述居于物理问题的称为几何多重网格法，关于代数多重网格法的介绍请参见[6]。

2、 AGM 的改进。在本实例的代码中，各个网格层之间的数据转换基本都是以代数平均值的方式传递。即细网格的亏损量在按相邻平均后作为粗网格的亏损量，粗网格的亏损量则平分后作为细网格的亏损量。从文献[5]也反映出，在不同的平滑和插值方法下，迭代的收敛有不同。

3、 多重网格在 GPU 上的运用。多重网格在 GPU 上已有相应应用，见文献[3][4]。其中 Nvidia 提供了相应的 SDK 或 API 为 AmgX。

4、 经典多重网格并行计算的几个瓶颈。多重网格法属于 Gauss-Seidel 类型，层与层之间相互关联，对于分布式系统通信上的耗时原大于计算耗时，算法上的可扩展性较弱。在当前大量并行的情况下，前面两个问题更显凸出[1]。

5、 多重网格法的大规模并行计算。与区域分解法(参见本书的相应章节)一起处理大规模问题是另一个方向[1]。

七、 区域分解法介绍

区域分解法(Domain Decomposition Method)是近几十年发展起来的数值方法，因在算法上具备有并行性，特别适合在并行机上执行，因此广泛应用到天气预报、大型结构工程等各种数值场合，随 GPU 运用的不断深入，也有学者在 GPU 上实现有限元的求解上[7]。

区域分解法的简单历史、运用和现状。在 1870 年有德国数学家 Schwarz 提出的 Schwarz 交替法中，在 60 年代有学者用于数值计算，但未引起注意，随着并行机的运用，此种方法具有的优势随需求应运而生，在本世纪末，各种书籍、文章开始大量涌现。由于优势明显，已学者用于 Navier - Stokes 方程、Schrödinger 方程、Black - Scholes 方程、Lighthill - Whitham 方程的求解[10]，且国际上已有 http://www.ddm.org/，有学者在 GPU 上运用此方法来求解问题[7]。

区域分解法的优越性。(1)将大的计算问题分解成小的计算问题，缩小了计算规模，(2)子区域上可以利用各种不同的算法完成，(3)单个子区域中网格一致，不同网格间网格可以不一致，(4)各个子区域之间使用不同的数学模型更接近真实，(5)各个子区域之间算法独立，可以并行计算。通过了解下面的小实例就大致理解区域分解法的大致思想。

由于此技术仍处于发展之中，尚未找到此方法的综述性文献，通过以下文献[8][9]了解和运用。未提供语言代码(理解原理就可以使用前面各个章节的代码进行实现)。

八、 区域分解法原理

为方便讲解区域分解法，首先建立两个概念。一般工程要求解的区域较大，如图 6-3 中的整个 L 型求解区域，使用有限差分离散后区域不规则，使用分治法将各个待求部分分块，然后对分块进行求解。分块后各个区域中的变量分为两类，其中一类称为**子区域公共变量**，即分块后，各个区域之间相邻部分的待求变量。**子区域内部变量**，区域中除去相邻变量后的内部待求变量。可以说，在方程离散后，整个待求解的变量因分块后，待求解的变量分成子区域内部变量和子区域公共变量。以上概念对应到图 6-3 中，整个待求解的区域分块成 1～3 个，对区域 1 来说，子区域内部变量是圆圈内数字为 1、2 的离散点，子区域公共变量是圆圈内数字为 7、8、9、10 的离散点，方框内的数值是整个区域的已知边界条件。

区域分解法的基本思想。是将需要求解的区域分块后，使用代数方法中的消元，即消去子区域内部变量，求解出子区域公共变量，再使用此值作为分块区域的边界条件，再继续求解子区域内部变量，而各个分块的求解具备并行条件，相互独立，这也就是区域分解法的优势。

子区域内部变量和子区域公共变量与矩阵分块的联系。整个离散化的线性方程组从求解的角度可以将其划分成块，这些分块恰好可以和几何上的区域分块对应，即子区域内部变量划分成一个矩阵块，子区域公共部分划分成另一个块，剩余的也相应划块作为系数。这样，离散后的整个方程表示分块后成两个方程，两个方程都含有子区域内部变量和子区域公共变量，只不过系数矩阵不同。在求解中使用代数中的消元法，消去子区域内部变量，只求解子区域公共变量的值，待求出后，再求解子区域内部变量。

结合实例图形的计算步骤。按照以上的讲解，结合图 6-3 来叙述整个执行过程，第一步求解的值是子区域公共变量，是子区域内部变量的边界值，第二步，并行计算各个子区域内部变量。对应到下图就是先计算出子区域公共变量 7、8、9、10 的值，再计算子区域内部变量 1、2、3、4、5、6 的值，只不过在计算区域 1 时的 1、2 时可以使用已知的 7、8、9、10 和其他边界值，计算区域 2 时，使用 8、9、10 和其他边界值来计算 3、4，这样分块的计算就相互独立，互不干涉。

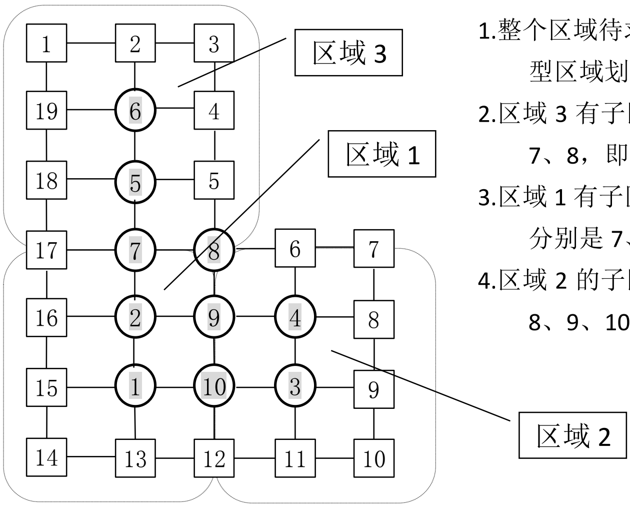
现在结合一个小实例来方便理解区域分解法的原理，具体讲解的步骤是：(1)将要求解的问题离散成代数方程，(2)将矩阵分块与待求的变量对应，(3)求解子区域的公共变量，(4)对分块的并行情况进行验证。图 6-3 是一个 L 型区域，满足式 Laplace 方程和边界条件式(6-1)。

$$\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}=0 \qquad x,y\in\Omega \qquad (6\text{-}1)$$
$$u=f(x,y) \qquad x,y\in\partial\Omega$$

为求解其数值解，将其离散后如图 6-3 所示，其中内部区域是待求解，由带圆圈的数字标注，从 1 到 10，外部边界已知，使用在方框的数字标注，从 1 到 19。

使用区域分解法求解，将此 L 型Ω区域划分成三个区域，如虚线框划分所示，各个区域之间的边界为 7 到 10。

注意，要让区域分解法能有效执行，变量命名的顺序需遵循以下原则，首先是区域内部变量，其次是区域公共变量，这样才能让矩阵分块后，保证要消元的变量能在一块中。下图中，区域内部变量编号连续取值，从 1 到 6，区域公共变量连续取值，7 到 10。



1.整个区域待求的点为：1～10，边界点为 1～19。将此 L 型区域划分成 3 个。
2.区域 3 有子区域内部变量：5、6，与区域 1 的公共点为 7、8，即子区域公共变量。
3.区域 1 有子区域内部变量：1、2，与区域 2、3 的公共点分别是 7、8 和 9、10，即子区域公共变量。
4.区域 2 的子区域内部变量：3、4，与区域 1 的公共点为 8、9、10，即子区域公共变量。

图 6-3

1、离散后的代数方程

按照有限差分格式的 5 点差分法列出各个未知点的代数方程，共 10 个，按求解区域内的未知点的顺序列出，其中x_i表示第 i 个待求点(i=1,2,⋯10)，bj 表示整个区域的边界(j=1,2,⋯19)，是已知条件。

$$4x_1-x_2-x_{10}=\mathrm{b15}+\mathrm{b13}$$
$$4x_2-x_1-x_7-x_9=\mathrm{b16} \qquad (6\text{-}2)$$
$$4x_3-x_4-x_{10}=\mathrm{b9}+\mathrm{b11}$$

$$4x_4 - x_3 - x_9 = \mathrm{b6} + \mathrm{b8}$$
$$4x_5 - x_6 - x_7 = \mathrm{b5} + \mathrm{b18}$$
$$4x_6 - x_5 = \mathrm{b2} + \mathrm{b4} + \mathrm{b19}$$
$$4x_7 - x_2 - x_5 - x_8 = \mathrm{b17}$$
$$4x_8 - x_7 - x_9 = \mathrm{b5} + \mathrm{b6}$$
$$4x_9 - x_2 - x_4 - x_8 - x_{10} = 0$$
$$4x_{10} - x_1 - x_3 - x_9 = \mathrm{b12}$$

整理以上方程，写成矩阵形式

$$\begin{bmatrix} 4 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\ -1 & 4 & 0 & 0 & 0 & 0 & -1 & 0 & -1 & 0 \\ 0 & 0 & 4 & -1 & 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & -1 & 4 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 4 & -1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 4 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & -1 & 0 & 4 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 4 & -1 & 0 \\ 0 & -1 & 0 & -1 & 0 & 0 & 0 & -1 & 4 & -1 \\ -1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & -1 & 4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \\ x_{10} \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \\ b_8 \\ b_9 \\ b_{10} \end{bmatrix} \tag{6-3}$$

2、矩阵分块

观察以上矩阵，有如下特点：对称、稀疏，可以分块。现在使用分块矩阵表示整个矩阵，其中分块按照子区域内部变量和子区域公共变量分块，待求变量从 1 行至 6 行，为子区域内部变量，7 行至 10 行为子区域公共变量，按照以上方式对式(6-3)分块有式(6-4)

$$\begin{bmatrix} \mathbf{M} & \mathbf{P} \\ \mathbf{P}^{\mathrm{T}} & \mathbf{N} \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix} = \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix} \tag{6-4}$$

其中矩阵$\mathbf{M}$、$\mathbf{M}$、$\mathbf{N}$、向量$\mathbf{x}$、$\mathbf{y}$、$\mathbf{a}$、$\mathbf{b}$分别表示见式(6-5)，式(6-6)，式(6-7)，式(6-8)。

$$\mathbf{M} = \begin{bmatrix} 4 & -1 & 0 & 0 & 0 & 0 \\ -1 & 4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 4 & -1 & 0 & 0 \\ 0 & 0 & -1 & 4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 4 & -1 \\ 0 & 0 & 0 & 0 & -1 & 4 \end{bmatrix} \tag{6-5}$$

$$\mathbf{P} = \begin{bmatrix} 0 & 0 & 0 & -1 \\ -1 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \\ 0 & 0 & -1 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{6-6}$$

$$\mathbf{N} = \begin{bmatrix} 4 & -1 & 0 & 0 \\ -1 & 4 & -1 & 0 \\ 0 & -1 & 4 & -1 \\ 0 & 0 & -1 & 4 \end{bmatrix} \tag{6-7}$$

$$\begin{gathered} \mathbf{x} = [x_1 \quad x_2 \quad x_3 \quad x_4 \quad x_5 \quad x_6]^{\mathrm{T}} \\ \mathbf{y} = [x_7 \quad x_8 \quad x_9 \quad x_{10}]^{\mathrm{T}} \\ \mathbf{a} = [b_1 \quad b_2 \quad b_3 \quad b_4 \quad b_5 \quad b_6]^{\mathrm{T}} \\ \mathbf{b} = [b_7 \quad b_8 \quad b_9 \quad b_{10}]^{\mathrm{T}} \end{gathered} \tag{6-8}$$

3、求解子区域公共变量

式(6-4)中的$\mathbf{x}$表示子区域内部变量，$\mathbf{y}$表示子区域公共变量，$\mathbf{a}$、$\mathbf{b}$都是边界已知条件，只不过分块而已，$\mathbf{M}$是子区域内部变量对应的矩阵，$\mathbf{P}$是子区域公共变量对应的矩阵，式(6-4)可以写成式(6-9)，式(6-10)。

$$\mathbf{Mx}+\mathbf{Py}=\mathbf{a} \tag{6-9}$$

$$\mathbf{P^T x}+\mathbf{Ny}=\mathbf{b} \tag{6-10}$$

通过消元$\mathbf{x}$，可以求出$\mathbf{y}$，具体步骤为将式(6-9)变换，其中$\mathbf{M}$非奇异

$$\mathbf{x}=\mathbf{M}^{-1}(\mathbf{a}-\mathbf{Py}) \tag{6-11}$$

带入式(6-10)，有

$$\mathbf{P^T M^{-1}}(\mathbf{a}-\mathbf{Py})+\mathbf{Ny}=\mathbf{b} \tag{6-12}$$

在$\mathbf{P^T M^{-1} P}-\mathbf{N}$非奇异的情况下，得到$\mathbf{y}$的表达式，具体表达式为

$$\mathbf{y}=(\mathbf{P^T M^{-1} P}-\mathbf{N})^{-1}(\mathbf{b}-\mathbf{P^T M^{-1} a}) \tag{6-13}$$

以上的思路就是通过代数方程的消元，计算出各个区域之间的相邻的边界值，有了边界值，将这些边界条件作为已知量，带入式(6-9)即可求出$\mathbf{x}$，这就是区域分解法的核心思想。现在来回头看下，再求解出$\mathbf{y}$后，在求解$\mathbf{x}$是否分块求解。

4、求解子区域内部变量

式(6-5)的代数方程同样可以表示成式(6-9)和式(6-10)的方式，这样分块的目的是将各个子区域体现出来，子区域公共变量依然作为单独块。即分块 1 的子区域内部变量为$\mathbf{x_1}$，矩阵为$\mathbf{A}$，边界条件为$\mathbf{a_1}$，区域 2 的内部变量为$\mathbf{x_2}$，矩阵为$\mathbf{B}$，边界条件为$\mathbf{a_2}$。子区域公共变量作为一个块存在，不区分，则有如下分块表示。

$$\begin{bmatrix} \mathbf{A} & 0 & 0 & \mathbf{E} \\ 0 & \mathbf{B} & 0 & \mathbf{F} \\ 0 & 0 & \mathbf{C} & \mathbf{G} \\ \mathbf{E}^\mathrm{T} & \mathbf{F}^\mathrm{T} & \mathbf{G}^\mathrm{T} & \mathbf{D} \end{bmatrix} \begin{bmatrix} \mathbf{x_1} \\ \mathbf{x_2} \\ \mathbf{x_3} \\ \mathbf{y} \end{bmatrix} = \begin{bmatrix} \mathbf{a_1} \\ \mathbf{a_2} \\ \mathbf{a_3} \\ \mathbf{b} \end{bmatrix} \tag{6-14}$$

其中，各个分块的表示见式(6-13)。

$$\begin{gathered}
\mathbf{A}=\mathbf{B}=\mathbf{C}=\begin{bmatrix} 4 & -1 \\ -1 & 4 \end{bmatrix} \\
\mathbf{E}=\begin{bmatrix} 0 & 0 & 0 & -1 \\ -1 & 0 & -1 & 0 \end{bmatrix} \\
\mathbf{F}=\begin{bmatrix} 0 & 0 & 0 & -1 \\ 0 & 0 & -1 & 0 \end{bmatrix} \\
\mathbf{G}=\begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \\
\mathbf{x_1}=[x_1 \quad x_2]^\mathrm{T} \\
\mathbf{x_2}=[x_3 \quad x_4]^\mathrm{T} \\
\mathbf{x_3}=[x_5 \quad x_6]^\mathrm{T} \\
\mathbf{y}=[x_7 \quad x_8 \quad x_9 \quad x_{10}]^\mathrm{T} \\
\mathbf{a_1}=[b_1 \quad b_2]^\mathrm{T} \\
\mathbf{a_2}=[b_3 \quad b_4]^\mathrm{T} \\
\mathbf{a_3}=[b_5 \quad b_6]^\mathrm{T} \\
\mathbf{b}=[b_7 \quad b_8 \quad b_9 \quad b_{10}]^\mathrm{T}
\end{gathered} \tag{6-15}$$

将式(6-12)展开

$$\begin{aligned} \mathbf{Ax_1} + \mathbf{Ey} = \mathbf{a_1} \\ \mathbf{Bx_2} + \mathbf{Fy} = \mathbf{a_2} \\ \mathbf{Cx_3} + \mathbf{Cy} = \mathbf{a_3} \end{aligned} \tag{6-16}$$

将式(6-12)移项有

$$\begin{aligned} \mathbf{Ax_1} = \mathbf{a_1} - \mathbf{Ey} \\ \mathbf{Bx_2} = \mathbf{a_2} - \mathbf{Fy} \\ \mathbf{Cx_3} = \mathbf{a_3} - \mathbf{Cy} \end{aligned} \tag{6-17}$$

因在式(6-13)中已经求解出$\mathbf{y}$，式(6-17)可以分别在不同的计算机上执行，各个区域之间无联系，可以并行完成，这就是区域分解法的核心思想。

九、 区域分解法方向

在当今超大规模计算的要求下，区域分解法的优势引起学者的极大关注，在 amazon 上关于此方法的图书不下 30 本。关于此方法的更进一步情况请参看链接。

十、 参考文献

[1]李晓梅，莫则尧，多重网格算法综述，中国科学，1996

[2]刘超群，多重网格法及其在计算流体力学中的运用，1993

[3]Godnight Nolan etal.A multigrid solver for boundary value problems using programmable graphics hardware

[4]Edmond Chowz etal. A Survey of Parallelization Techniques for Multigrid Solvers

[5]Q. S. Chang etal.New interpolation formulas of using geometric assumptions in the algebraic multigrid method 1992

[6]R.D.Faigout etal.A Introduction to Algebraic Multigrid 2006

[7]G. Stavroulakis.A GPU domain decomposition solution for spectral stochastic finite element method，2017

[8]吕涛，石济民，林振宝.区域分解算法-偏微分方程数值解新技术，科学出版社，1992

[9]Yousef Saad.Iteration Methods for Sparse Linear Systems，SIAM，2003

[10]Victorita Dolean etal. An Introduction to Domain Decomposition Methods:algorithms, theory and parallel implementation，SIAM,2015

十一、代码和一些说明

多重网格法代码 C++实现见 https://github.com/ljb1672/

区域分解法未提供代码。

由于非数学专业,国内也无通俗讲解此类新方法的通俗读物,因此在叙述和讲解上存在错误，恳请指出。