第十一章 通用水头边界子程序包

概念及程序化

通用水头边界（General-Head Boundary,或简称为GHB）子程序包的作用在数学形式上与河流、沟渠及蒸发蒸腾子程序包相似之处在于，从外部水源进入或流出计算单元(i,j,k)的水流量与该计算单元水头$h_{i,j,k}$和外部水源的水头$h_{bi,j,k}$之差成正比。据此，可确立计算单元水流量与计算单元水头间的线性关系，即

$$Q_{bi,j,k}=C_{bi,j,k}(h_{bi,j,k}-h_{i,j,k}) \tag{78}$$

其中$Q_{bi,j,k}$是从外部水源进入计算单元(i,j,k)的流量；$C_{bi,j,k}$是外部水源与计算单元(i,j,k)间的水力传导系数；$h_{bi,j,k}$是外部水源的水头；$h_{i,j,k}$是计算单元(i,j,k)的水头。计算单元(i,j,k)和外部水源的关系见图44。通用水头的水源由图44右边的具有固定水头的水箱表示，该水箱中的水源不受其它因素的影响，将水头维持在H_b的位置；水源与计算单元(i,j,k)的联系用多孔材料块体$C_{bi,j,k}$表示。注意图44中并没有任何机制限制水流因$h_{i,j,k}$上升或下降而向或左或右的方向流动。当$h_{i,j,k}$大于$h_{bi,j,k}$时，地下水由计算单元(i,j,k)向右流向水箱；而反过来，地下水向左流动进入计算单元(i,j,k)。

图45是公式(78)给出的$Q_{bi,j,k}$对$h_{i,j,k}$变量的曲线图解。与河流、沟渠和蒸发蒸腾子程序包相比，通用水头边界与含水层之间的流量仅与二者之间的水头差成正比，而不受其它条件限制。当计算单元(i,j,k)和外部水源水头之差增加时，进入或流出计算单元的水流不受限制地持续增加。因而在使用通用水头边界子程序包时必须注意，以保证不让进入或流出系统的流量随模拟过程变得不切实际。

因为公式(78)中$Q_{bi,j,k}$是定义为流进含水层的，所以必须将其加到公式(24)的左边。至于*HCOF*和*RHS*两项，则在联立求解矩阵方程时，从公式(26)的$HCOF_{i,j,k}$中减去$C_{bi,j,k}$，以及从$RHS_{i,j,k}$项中减去$C_{bi,j,k}h_{bi,j,k}$。

[译注：当水力传导系数很大时，GHB子程序包的作用与定水头边界相同。而且由于沿GHB边界的水位可以随时间变化，故可以在非稳定流计算中用来模拟地表水体水位随时间的变化。]

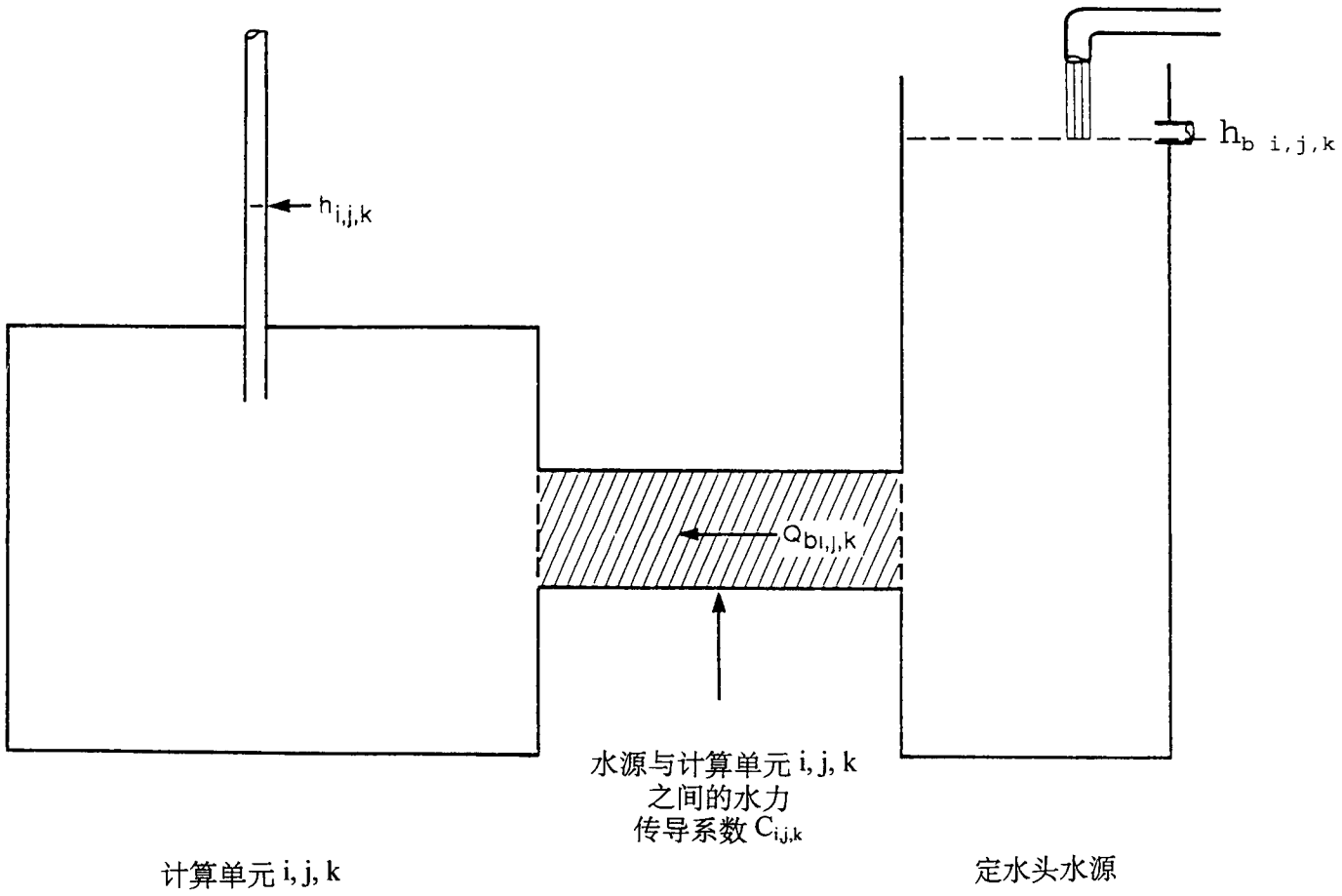


图44. 通用水头边界子程序包原理图。

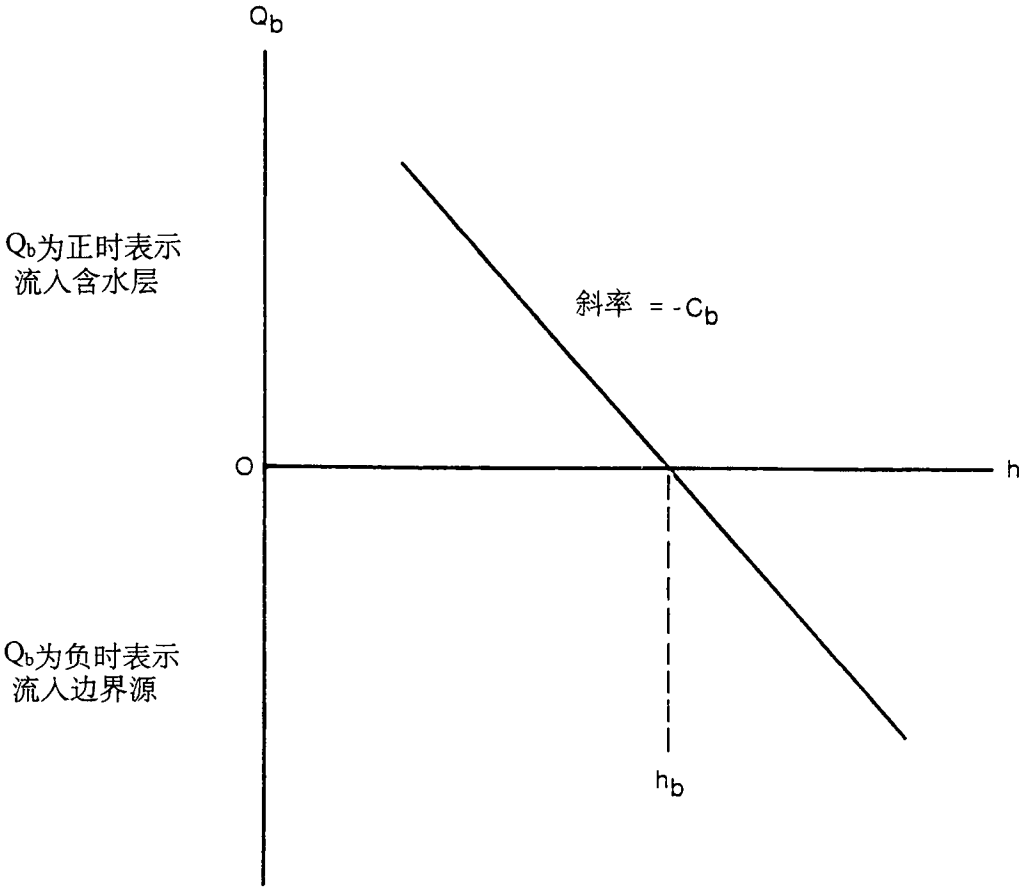


图45. 从通用水头边界水源流入某计算单元的流量Q_b与该计算单元水头h的函数关系示意图。h_b为水源之水头。

通用水头边界子程序包输入数据及格式

通用水头边界子程序包的输入从IUNIT(7)所指定的文件设备号读取。

对每次模拟:

由子程序GHB1AL读取的数据包括

1. 数据名称: MXBND IGHBCB
   输入格式: I10 I10

对每个应力期:

由子程序GHB1RP读取的数据包括

2. 数据名称: ITMP
   输入格式: I10
3. 数据名称: Layer Row Col Head Cond
   输入格式: I10 I10 I10 F10.0 F10.0
   (该项输入为每段通用水头边界一行记录。若ITMP是负或零，则不应输入本项。)

输入数据说明

MXBND: 各应力期中通用水头边界所在计算单元的最大数目。

IGHBCB: 标示符和文件设备号。
若IGHBCB>0，当ICBCFL（见输出控制）设定时，是记录计算单元间流量的文件设备号。
若IGHBCB=0，将不打印或记录计算单元间流量。
若IGHBCB<0，当ICBCFL设定时，将打印每个计算单元的边界渗流量。

ITMP: 标示符和计数器。

若ITMP<0，将重复使用上一应力期GHB的数据。

若ITMP≥0，ITMP是当前应力期通用水头边界单元的总数目。

Layer: 受通用水头边界影响的计算单元所在层号。

Row : 受通用水头边界影响的计算单元所在行号。

Col: 受通用水头边界影响的计算单元所在列号。

Head: 边界上的水头。

Cond: 含水层计算单元与通用水头边界之间界面的水力传导系数。

通用水头边界子程序包输入样单

数据项	说明	输入记录				
1	{MXBND, IGHBCB}	6	24			
2	第一应力期 {ITMP}	4				
3	第一段边界 {层号,行号,列号,水头,水力传导系数}	2	5	6	235.0	.0012
3	第二段边界 {层号,行号,列号,水头,水力传导系数}	2	4	6	230.0	.0012
3	第三段边界 {层号,行号,列号,水头,水力传导系数}	2	5	8	250.0	.0018
3	第四段边界 {层号,行号,列号,水头,水力传导系数}	2	7	6	235.0	.0012
2	第二应力期 {ITMP}	-1				
2	第三应力期 {ITMP}	-1				
2	第四应力期 {ITMP}	6				
3	第一段边界 {层号,行号,列号,水头,水力传导系数}	2	5	6	235.0	.0012
3	第二段边界 {层号,行号,列号,水头,水力传导系数}	2	4	6	230.0	.0012
3	第三段边界 {层号,行号,列号,水头,水力传导系数}	2	5	8	250.0	.0018
3	第四段边界 {层号,行号,列号,水头,水力传导系数}	2	7	6	235.0	.0018
3	第五段边界 {层号,行号,列号,水头,水力传导系数}	2	9	6	235.0	.0012
3	第六段边界 {层号,行号,列号,水头,水力传导系数}	2	10	6	250.0	.0012

[译注: 样单中的数据可能不符合格式要求，仅供参考。]

第十二章　强隐式法子程序包

概念及程序化

在本章，我们讨论强隐式法（Strongly Implicit Procedure，简称为SIP)。对于文中涉及到的一些矩阵代数和数值分析中的基本概念，读者可以阅读有关参考书，例如前面提到的Peaceman (1977), Crichlow(1977)或Remson, Hornberger和Molz(1971)。这些参考文献不仅可以帮助熟悉有关基本概念，它们对强隐式法的介绍还可以作为本书内容的补充。

强隐式法是一种对大型线性方程组联立迭代求解的方法。正如在第二章中讨论过的，一个计算单元（i, j, k)的有限差分方程具有下面的形式:

$$\begin{aligned} &CV_{i,j,k-\frac{1}{2}}h_{i,j,k-1}+CC_{i-\frac{1}{2},j,k}h_{i-1,j,k}+CR_{i,j-\frac{1}{2},k}h_{i,j-1,k}\\ &+(-CV_{i,j,k-\frac{1}{2}}-CC_{i-\frac{1}{2},j,k}-CR_{i,j-\frac{1}{2},k}-CR_{i,j+\frac{1}{2},k}-CC_{i+\frac{1}{2},j,k}-CV_{i,j,k+\frac{1}{2}}+HCOF_{i,j,k})h_{i,j,k}\\ &+CR_{i,j+\frac{1}{2},k}h_{i,j+1,k}+CC_{i+\frac{1}{2},j,k}h_{i+1,j,k}+CV_{i,j,k+\frac{1}{2}}h_{i,j,k+1}=RHS_{i,j,k} \end{aligned} \tag{79}$$

在每个时间段开始时,我们可以按上式对网格内每个有效计算单元写出一个这样的差分方程。该方程中包含有计算单元（i,j,k)本身以及与其相邻之六个计算单元的水头。由于每个差分方程中可能出现七个未知项，而且它们的值可能不完全相同，故所有计算单元的未知水头应在同一时间段内联立求解。联立的有限差分方程组的同步解，即为该时间段完成时，欲求的各个计算单元的水头预测值。

最早提出SIP方法的是 Weinstein, Stone 和 Kwan(1969)。在这里我们仍采用他们的标记法，将公式（79）改写为:

$$\begin{aligned} &Z_{i,j,k}h_{i,j,k-1}+B_{i,j,k}h_{i-1,j,k}+D_{i,j,k}h_{i,j-1,k}+E_{i,j,k}h_{i,j,k}\\ &+F_{i,j,k}h_{i,j+1,k}+H_{i,j,k}h_{i+1,j,k}+S_{i,j,k}h_{i,j,k+1}=Q_{i,j,k} \end{aligned} \tag{80}$$

公式（80）中各项系数的下标均记为（i,j,k)。因此，该式中的$Z_{i,j,k}$相当于(79)中的$CV_{i,j,k-1/2}$; $E_{i,j,k}$则相当公式（79）中的

$$(-CV_{i,j,k-\frac{1}{2}} - CC_{i-\frac{1}{2},j,k} - CR_{i,j-\frac{1}{2},k} - CR_{i,j+\frac{1}{2},k} - CC_{i+\frac{1}{2},j,k} - CV_{i,j,k+\frac{1}{2}} + HCOF_{i,j,k})$$

其它项则可依此类推。

正如在第二章中提到的，我们可以按照公式（80）的形式，对每个未知计算单元写出一个差分方程，并将它们记为:

$$[A]\{h\} = \{q\} \tag{81}$$

在这里[A]为水头的系数矩阵；{h}为水头矢量矩阵；{q}为（80）式中的右侧项矢量（已知量）。图46表示一个包含有三行、四列、两层的模型所对应的系数矩阵以及两个矢量矩阵。很显然，系数矩阵为一稀疏矩阵。这就是说，只有很少一部分的元素不为零；而且这些非零元素均位于七条对角线上（参见图47）。

从公式（79）和（80）中可以看出，（79）式中的$CV_{i,j,k-1/2}$项同时出现于（80）式中表示计算单元（i,j,k)的系数Z和表示计算单元（i,j,k－1)的系数S之中。即:

$$Z_{i,j,k} = S_{i,j,k-1} \tag{82}$$

同样,

$$B_{i,j,k} = H_{i-1,j,k} \tag{83}$$

$$D_{i,j,k} = F_{i,j-1,k} \tag{84}$$

如果按公式（82）至（84）的定义将图46中所示矩阵中的Z，B，D用与它们相当的S，H和F所替换，就可以得到如图48所示的矩阵形式。其对称性显而易见。所以我们说，公式（81）中系数矩阵为一稀疏对称矩阵。

如果[A]可以分解为两个子矩阵：$[L^*]$和$[U^*]$,由（81）所表示的方程组则可用直接的方法求解。这里，$[L^*]$为一下三角矩阵（即所有非零元素均出现于主对角线或主对角线以下）；$[U^*]$为一上三角矩阵（即所有非零元素均出现于主对角线上或主对角线以上且主对角线上的元素均为1）。图49表示一个3x3的矩阵以及相应的$[L^*]$和$[U^*]$。一旦系数矩阵分解为$[L^*]$和$[U^*]$之后，就可以用一种称为“前后替换”的方法对方程组进行求解。但这样做会有困难：虽然[A]为稀疏矩阵，但一般$[L^*]$和$[U^*]$并非稀疏矩阵。对其中非零元素的计算常需要大量的计算机内存和时间。除此之外，舍入误差可能会很大。

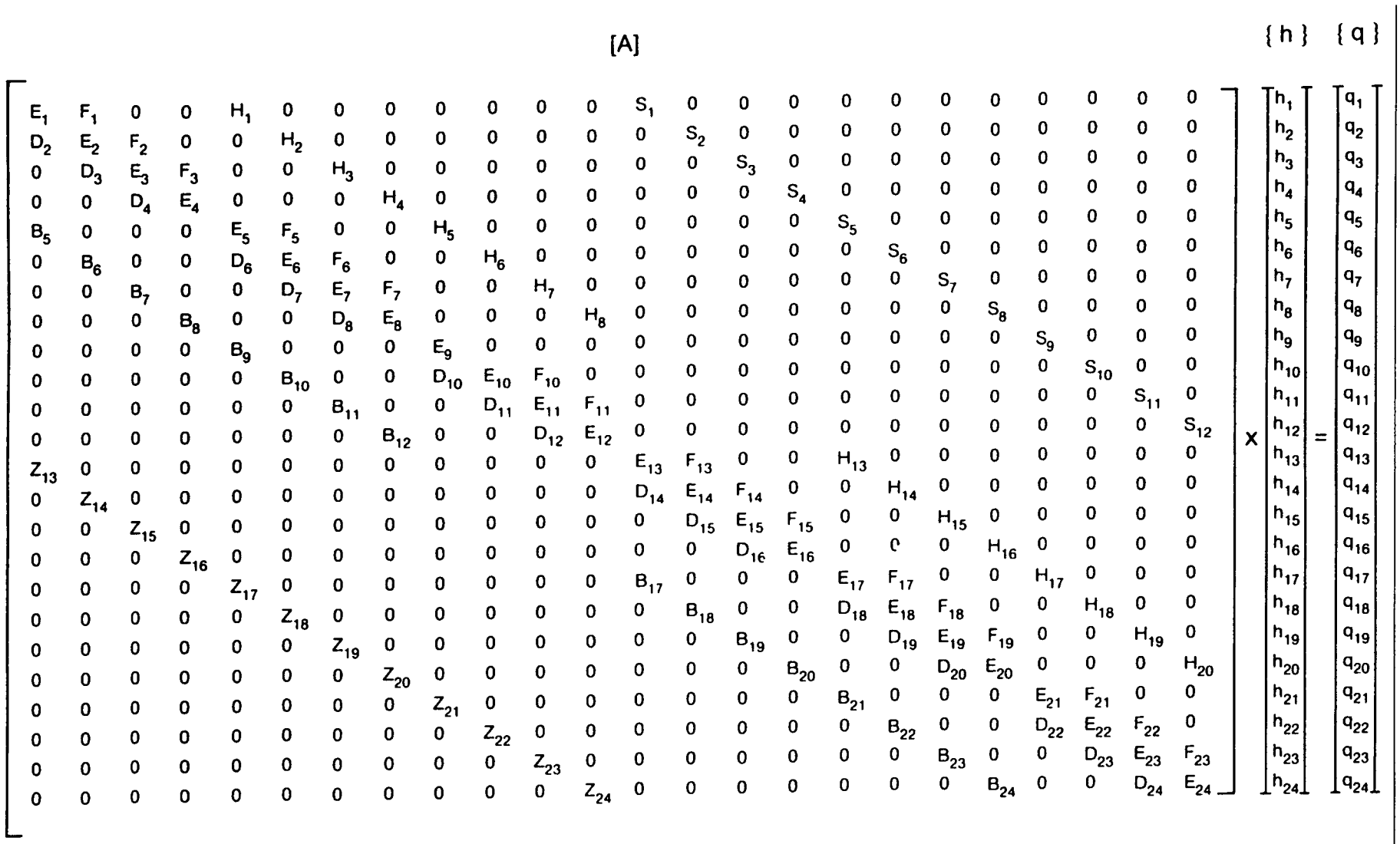
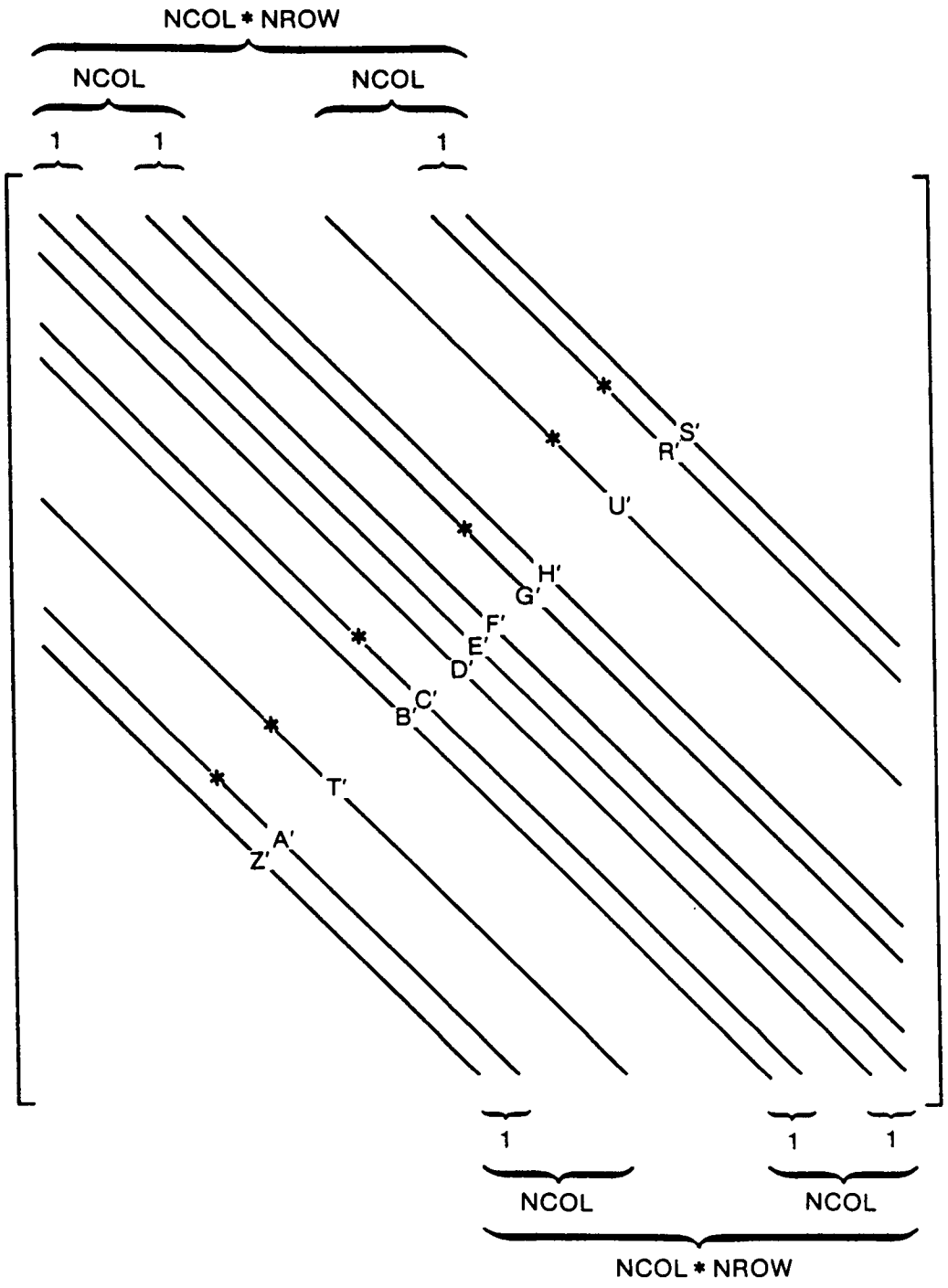


图46. 表示一个含有二行，三列，两层的网格的有限差分方程组与其矩阵方程的对应关系。



上下两端的花括弧表示非零对角线的水平距离，
并以列数来表示（例如，对角线E和F相邻）

图47. 系数矩阵的结构。斜线表示非零对角线。

$E_{1,1,1}$	$F_{1,1,1}$	0	$H_{1,1,1}$	0	0	$S_{1,1,1}$	0	0	0	0	0
$F_{1,1,1}$	$E_{1,2,1}$	$F_{1,2,1}$	0	$H_{1,2,1}$	0	0	$S_{1,2,1}$	0	0	0	0
0	$F_{1,2,1}$	$E_{1,3,1}$	0	0	$H_{1,3,1}$	0	0	$S_{1,3,1}$	0	0	0
$H_{1,1,1}$	0	0	$E_{2,1,1}$	$F_{2,1,1}$	0	0	0	0	$S_{2,1,1}$	0	0
0	$H_{1,2,1}$	0	$F_{2,1,1}$	$E_{2,2,1}$	$F_{2,2,1}$	0	0	0	0	$S_{2,2,1}$	0
0	0	$H_{1,3,1}$	0	$F_{2,2,1}$	$E_{2,3,1}$	0	0	0	0	0	$S_{2,3,1}$
$S_{1,1,1}$	0	0	0	0	0	$E_{1,1,2}$	$F_{1,1,2}$	0	$H_{1,1,2}$	0	0
0	$S_{1,2,1}$	0	0	0	0	$F_{1,1,2}$	$E_{1,2,2}$	$F_{1,2,2}$	0	$H_{1,2,2}$	0
0	0	$S_{1,3,1}$	0	0	0	0	$F_{1,2,2}$	$E_{1,3,2}$	0	0	$H_{1,3,2}$
0	0	0	$S_{2,1,1}$	0	0	$H_{1,1,2}$	0	0	$E_{2,1,2}$	$F_{2,1,2}$	0
0	0	0	0	$S_{2,2,1}$	0	0	$H_{1,2,2}$	0	$F_{2,1,2}$	$E_{2,2,2}$	$F_{2,2,2}$
0	0	0	0	0	$S_{2,3,1}$	0	0	$H_{1,3,2}$	0	$F_{2,2,2}$	$E_{2,3,2}$

图48. 一个含有二行，三列，两层的网格的对称系数矩阵。

$$\underset{[A]}{\begin{bmatrix} 1 & 2 & 1 \\ -1 & 1 & 2 \\ 3 & 2 & -2 \end{bmatrix}} \underset{\{h\}}{\begin{bmatrix} h_1 \\ h_2 \\ h_3 \end{bmatrix}} = \underset{\{q\}}{\begin{bmatrix} 1 \\ 2 \\ -3 \end{bmatrix}}$$

$$\underset{[L^*]}{\begin{bmatrix} 1 & 0 & 0 \\ -1 & 3 & 0 \\ 3 & -4 & -1 \end{bmatrix}} \underset{[U^*]}{\begin{bmatrix} 1 & 2 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}} \underset{\{h\}}{\begin{bmatrix} h_1 \\ h_2 \\ h_3 \end{bmatrix}} = \underset{\{q\}}{\begin{bmatrix} 1 \\ 2 \\ -3 \end{bmatrix}}$$

图49. 系数矩阵与由此分解出的上，下三角矩阵。

强隐式法子程序包（SIP）的工作原理是寻找一个矩阵[B]，使得[A+B]可以分解为两个子矩阵[L]和[U]。并且[A+B]，[L]和[U]满足下列条件:

（1）[A+B]“接近”于[A];

（2）[L]具有下三角矩阵的形式而[U]具有上三角矩阵的形式；[U]之主对角线上的元素均为1;

（3）[L]和[U]均为稀疏矩阵;

（4）[L]和[U]均仅含有四条非零对角线。

如果矩阵[B]能满足上述条件，则将[B]{h}加至公式（81）两侧后可得:

$$[A+B]\{h\}=\{q\}+[B]\{h\} \tag{85}$$

如果{h}是公式（85）的解，它也必然是（81）的解。但由于{h}出现于（85）式的两侧，使对公式（85）的求解并非易事。但如果采用迭代的办法（参见第二章）用前一次迭代后获得的值代替（85）式右侧的h值，则可将公式（85）写为下面的形式:

$$[A+B]\{h^{l}\}=\{q\}+[B]\{h^{l-1}\} \tag{86}$$

其中$\{h^{l}\}$为第l次迭代后的水头值；$\{h^{l-1}\}$为第$l-1$次迭代后的水头值。在公式（86）中，$\{h^{l-1}\}$实际上是$\{h^{l}\}$的一种近似值。如果矩阵[B]已知，则对公式（86）的求解是非常容易的。根据前面讲过的定义，我们可以将[A+B]分解成为两个稀疏矩阵[L]和[U],并使用前后替换法得到解。这样一来，对公式（86）的求解问题就转化为如何确定恰当的矩阵[B]的问题。但实际上，我们可利用矩阵[A]、[A+B]、[L]和[U]进行求解。从公式（86）的两侧同时减去[A+B]$\{h^{l-1}\}$后可得:

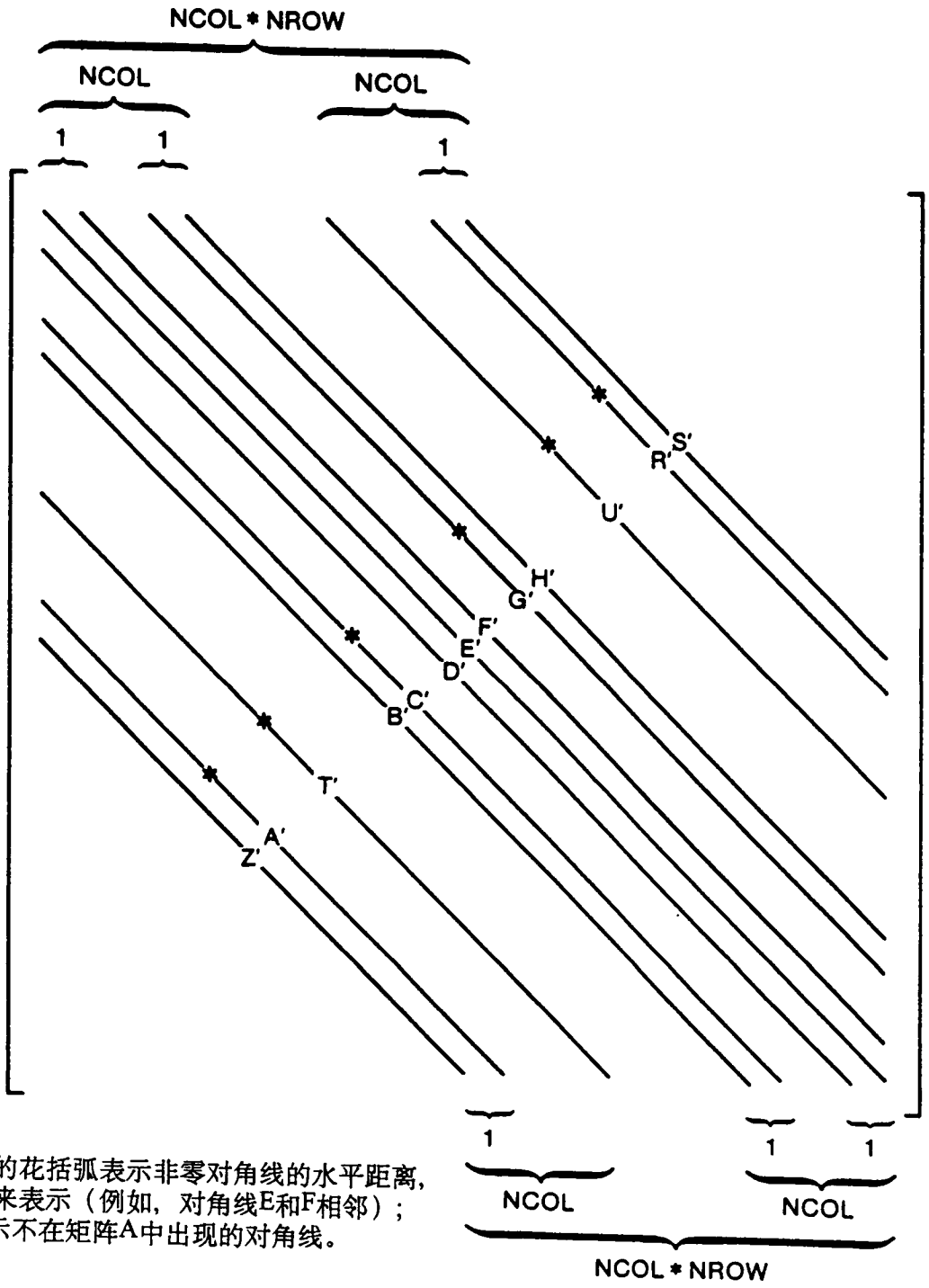
$$[A+B]\{h^{l}\}-[A+B]\{h^{l-1}\}=\{q\}-[A]\{h^{l-1}\} \tag{87}$$

或

$$[A+B]\{h^{l}-h^{l-1}\}=\{q\}-[A]\{h^{l-1}\} \tag{88}$$

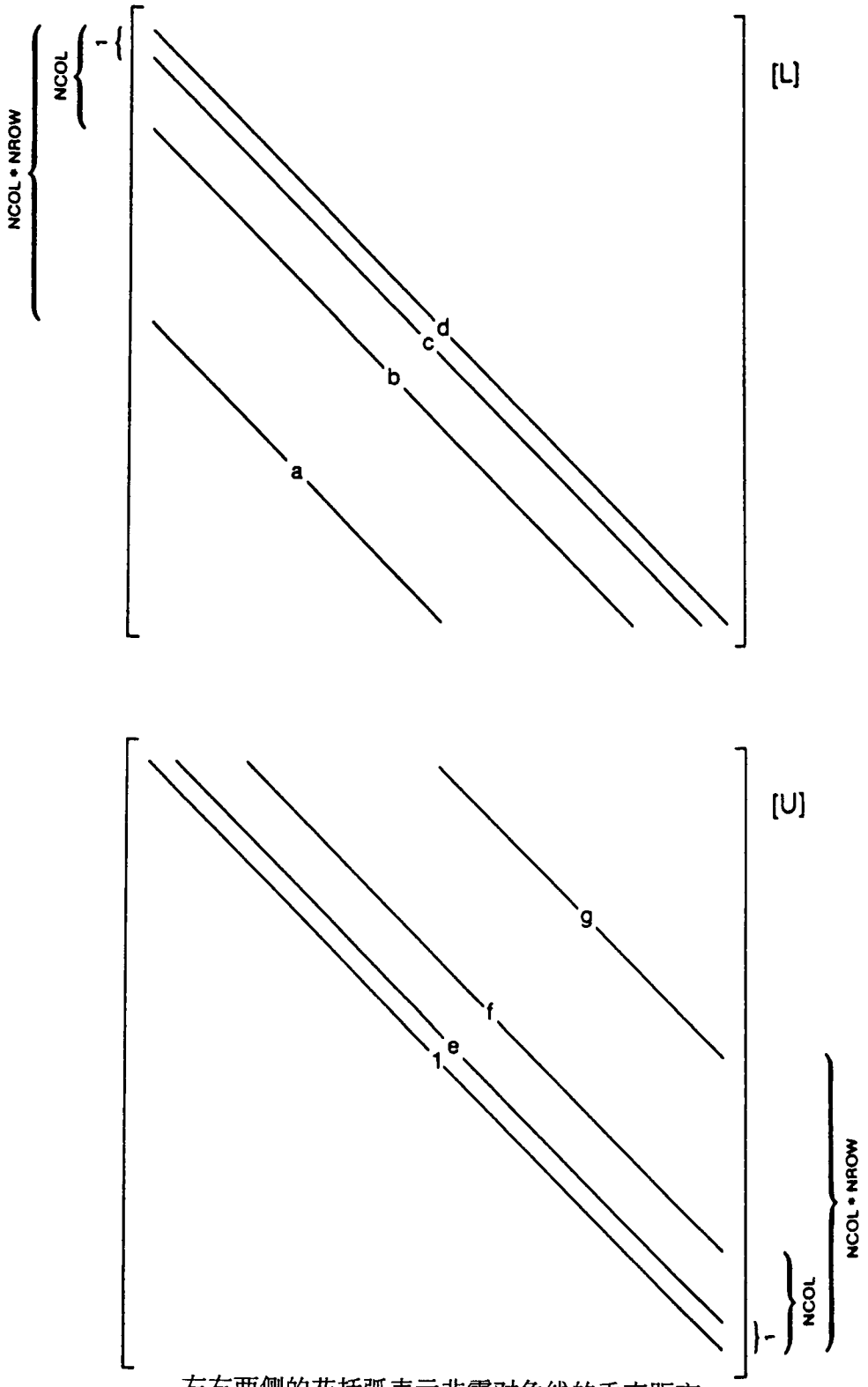
为了[L]和[U]以及[A+B]满足前面提到的条件，[A+B]必须包括矩阵[A]中所没有的六条非零对角线（见图50）。这些新非零对角线元素的增加，使得对计算单元（i,j,k)的求解公式中出现了与该单元不相邻的计算单元的水头项。矩阵[A+B]的元素与[L]和[U]的元素之间的关系可以由下列公式来定义。在图50和图51中，a、b、c和d为[L]的非零元素，而e、f和g为[U]中的非零元素。大写字母则表示矩阵[A+B]元素。

$$Z'_{i,j,k}=a_{i,j,k} \tag{89-a}$$



上下两端的花括弧表示非零对角线的水平距离，并以列数来表示（例如，对角线E和F相邻）；星号* 表示不在矩阵A中出现的对角线。

图50. 矩阵[A+B]之结构。 斜线表示非零对角线。



左右两侧的花括弧表示非零对角线的垂直距离，
并以行数来表示（例如，对角线 d 和 e 相邻）；

图51. 由矩阵[A+B]分解出的下三角矩阵[L]和上三角矩阵[U]之结构，
斜线表示非零对角线。

$$A'_{i,j,k} = a_{i,j,k} e_{i,j,k-1} \tag{89-b}$$

$$T'_{i,j,k} = a_{i,j,k} f_{i,j,k-1} \tag{89-c}$$

$$B'_{i,j,k} = b_{i,j,k} \tag{89-d}$$

$$C'_{i,j,k} = e_{i-1,j,k} b_{i,j,k} \tag{89-e}$$

$$D'_{i,j,k} = c_{i,j,k} \tag{89-f}$$

$$E'_{i,j,k} = a_{i,j,k} g_{i,j,k-1} + b_{i,j,k} f_{i-1,j,k} + e_{i,j-1,k} c_{i,j,k} + d_{i,j,k} \tag{89-g}$$

$$F'_{i,j,k} = d_{i,j,k} e_{i,j,k} \tag{89-h}$$

$$G'_{i,j,k} = f_{i,j-1,k} c_{i,j,k} \tag{89-i}$$

$$H'_{i,j,k} = f_{i,j,k} d_{i,j,k} \tag{89-j}$$

$$U'_{i,j,k} = b_{i,j,k} g_{i-1,j,k} \tag{89-k}$$

$$R'_{i,j,k} = g_{i,j-1,k} c_{i,j,k} \tag{89-l}$$

$$S'_{i,j,k} = g_{i,j,k} d_{i,j,k} \tag{89-m}$$

如果公式（89-a…m)中某元素的下标表示该元素位于模型之外，则其值为零。这十三个方程中共含有二十个未知数，它们都是矩阵[L]，[U]和[A+B]的元素。这表明满足这些条件的矩阵[B]的选择不是唯一的，很多不同的[B]矩阵与[A]相加后都可以分解为上三角矩阵[U]和下三角矩阵[L]。但[B]的选择还应满足另外一个条件，那就是[A+B]应当“接近”于[A]，或者说:

$$[A+B]\{h\} \approx [A]\{h\} \tag{90}$$

对于计算单元（i,j,k)而言，如果用矩阵[A+B]{h}和[A]{h}的元素来表达的话，（90）式可以看作为:

$$\begin{aligned} &Z'_{i,j,k} h_{i,j,k-1} + A'_{i,j,k} h_{i,j+1,k-1} + T'_{i,j,k} h_{i+1,j,k-1} + B'_{i,j,k} h_{i-1,j,k} + C'_{i,j,k} h_{i-1,j+1,k} \\ &+ D'_{i,j,k} h_{i,j-1,k} + E'_{i,j,k} h_{i,j,k} + F'_{i,j,k} h_{i,j+1,k} + G'_{i,j,k} h_{i+1,j-1,k} + H'_{i,j,k} h_{i+1,j,k} \\ &+ U'_{i,j,k} h_{i-1,j,k+1} + R'_{i,j,k} h_{i,j-1,k+1} + S'_{i,j,k} h_{i,j,k+1} \\ &\approx Z_{i,j,k} h_{i,j,k-1} + B_{i,j,k} h_{i-1,j,k} + D_{i,j,k} h_{i,j-1,k} + E_{i,j,k} h_{i,j,k} + F_{i,j,k} h_{i,j+1,k} \\ &+ H_{i,j,k} h_{i+1,j,k} + S_{i,j,k} h_{i,j,k+1} \end{aligned} \tag{91}$$

我们还可以把（91）式的形式改写如下：将不属于[A]矩阵六条非零对角线的元素移至右侧，而将矩阵[A]和[A+B]相应元素之差项移至方程的左侧，则有:

$$\begin{aligned}
&(Z_{i,j,k}-Z'_{i,j,k})h_{i,j,k-1}+(B_{i,j,k}-B'_{i,j,k})h_{i-1,j,k}\\
&+(D_{i,j,k}-D'_{i,j,k})h_{i,j-1,k}+(E_{i,j,k}-E'_{i,j,k})h_{i,j,k}\\
&+(F_{i,j,k}-F'_{i,j,k})h_{i,j+1,k}+(H_{i,j,k}-H'_{i,j,k})h_{i+1,j,k}\\
&+(S_{i,j,k}-S'_{i,j,k})h_{i,j,k+1}\approx A'_{i,j,k}h_{i,j+1,k-1}\\
&+T'_{i,j,k}h_{i+1,j,k-1}+C'_{i,j,k}h_{i-1,j+1,k}+G'_{i,j,k}h_{i+1,j-1,k}\\
&+U'_{i,j,k}h_{i-1,j,k+1}+R'_{i,j,k}h_{i,j-1,k+1}
\end{aligned}\tag{92}$$

公式（92）右侧项为矩阵[A+B]六条非零对角线上的元素，它们都不在矩阵[A]中出现。这些元素来自于矩阵[B]。与[A]中的非零元素相比较，这些元素都和不与计算单元(i,j,k)直接相连的计算单元的水头有关。公式（92）左侧各项则来自于矩阵[A]和[B]，并与计算单元(i,j,k)以及与其直接相连的计算单元的水头有关。

为了减少非相邻计算单元项的影响，我们引入三个计算参数（称为α，β和γ)作为公式（92）右侧项的乘因子。这些乘因子均取值于0和1之间。在矩阵求解(公式(85)和(86))过程中，这三个参数作为迭代参数使用。把它们用于（92）式后可得:

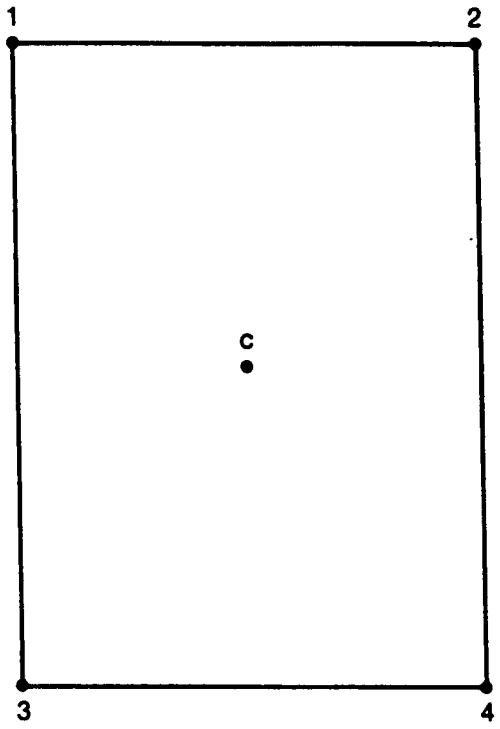
$$\begin{aligned}
&(Z_{i,j,k}-Z'_{i,j,k})h_{i,j,k-1}+(B_{i,j,k}-B'_{i,j,k})h_{i-1,j,k}\\
&+(D_{i,j,k}-D'_{i,j,k})h_{i,j-1,k}+(E_{i,j,k}-E'_{i,j,k})h_{i,j,k}\\
&+(F_{i,j,k}-F'_{i,j,k})h_{i,j+1,k}+(H_{i,j,k}-H'_{i,j,k})h_{i+1,j,k}\\
&+(S_{i,j,k}-S'_{i,j,k})h_{i,j,k+1}\approx \alpha A'_{i,j,k}h_{i,j+1,k-1}\\
&+\beta T'_{i,j,k}h_{i+1,j,k-1}+\gamma C'_{i,j,k}h_{i-1,j+1,k}+\gamma G'_{i,j,k}h_{i+1,j-1,k}\\
&+\beta U'_{i,j,k}h_{i-1,j,k+1}+\alpha R'_{i,j,k}h_{i,j-1,k+1}
\end{aligned}\tag{93}$$

下一步是将含有非相邻计算单元的水头项（即公式（93）的右侧项）用与计算单元(i,j,k)相邻的水头来表达。例如，计算单元（i,j+1,k-1)位于一个长方形的一个角，而其它三个角上分别为计算单元(i,j,k-1),(i,j+1,k)和(i,j,k)。用图52所示的插值的办法，$h_{i,j+1,k-1}$可近似表示为:

$$h_{i,j+1,k-1}=h_{i,j+1,k}+h_{i,j,k-1}-h_{i,j,k}\tag{94-a}$$

同理:

假定函数 f 在顶角2, 3, 4 的值已知:



通过插值,该函数在中心点 c 的值可近似表达为:

$$f_1(c) \approx \frac{f(2)+f(3)}{2}$$

或

$$f_2(c) \approx \frac{f(1)+f(4)}{2}$$

假定

$$f_1(c) \approx f_2(c)$$

则

$$\frac{f(2)+f(3)}{2} \approx \frac{f(1)+f(4)}{2}$$

所以

$$f(1) \approx f(2)+f(3)-f(4)$$

图52. 根据某函数在一个矩形三个顶点的已知值, 估计该函数在另一个顶点的值。

$$h_{i+1,j,k-1} = h_{i,j,k-1} + h_{i+1,j,k} - h_{i,j,k} \tag{94-b}$$

$$h_{i-1,j+1,k} = h_{i-1,j,k} + h_{i,j+1,k} - h_{i,j,k} \tag{94-c}$$

$$h_{i+1,j-1,k} = h_{i+1,j,k} + h_{i,j-1,k} - h_{i,j,k} \tag{94-d}$$

$$h_{i-1,j,k+1} = h_{i,j,k+1} + h_{i-1,j,k} - h_{i,j,k} \tag{94-e}$$

$$h_{i,j-1,k+1} = h_{i,j,k+1} + h_{i,j-1,k} - h_{i,j,k} \tag{94-f}$$

将公式（94－a … f）代入（93）式并整理后可得:

$$\begin{aligned}
&(Z'_{i,j,k} - Z_{i,j,k} + \alpha A'_{i,j,k} + \beta T'_{i,j,k})h_{i,j,k-1} \\
&+ (B'_{i,j,k} - B_{i,j,k} + \gamma C'_{i,j,k} + \beta U'_{i,j,k})h_{i-1,j,k} \\
&+ (D'_{i,j,k} - D_{i,j,k} + \gamma G'_{i,j,k} + \alpha R'_{i,j,k})h_{i,j-1,k} \\
&+ (E'_{i,j,k} - E_{i,j,k} - \alpha A'_{i,j,k} - \beta T'_{i,j,k} - \beta U'_{i,j,k} - \gamma C'_{i,j,k} - \gamma G'_{i,j,k} - \alpha R'_{i,j,k})h_{i,j,k} \\
&+ (F'_{i,j,k} - F_{i,j,k} + \alpha A'_{i,j,k} + \gamma C'_{i,j,k})h_{i,j+1,k} \\
&+ (H'_{i,j,k} - H_{i,j,k} + \beta T'_{i,j,k} + \gamma G'_{i,j,k})h_{i+1,j,k} \\
&+ (S'_{i,j,k} - S_{i,j,k} + \beta U'_{i,j,k} + \alpha R'_{i,j,k})h_{i,j,k+1} \approx 0
\end{aligned} \tag{95}$$

如果上式中各项系数接近于零，式（95）所表达的关系则可成立。为此，我们令这些系数等于零:

$$Z'_{i,j,k} - Z_{i,j,k} + \alpha A'_{i,j,k} + \beta T'_{i,j,k} = 0 \tag{96-a}$$

$$B'_{i,j,k} - B_{i,j,k} + \gamma C'_{i,j,k} + \beta U'_{i,j,k} = 0 \tag{96-b}$$

$$D'_{i,j,k} - D_{i,j,k} + \gamma G'_{i,j,k} + \alpha R'_{i,j,k} = 0 \tag{96-c}$$

$$E'_{i,j,k} - E_{i,j,k} - \alpha A'_{i,j,k} - \beta T'_{i,j,k} - \gamma C'_{i,j,k} - \gamma G'_{i,j,k} - \beta U'_{i,j,k} - \alpha R'_{i,j,k} = 0 \tag{96-d}$$

$$F'_{i,j,k} - F_{i,j,k} + \alpha A'_{i,j,k} + \gamma C'_{i,j,k} = 0 \tag{96-e}$$

$$H'_{i,j,k} - H_{i,j,k} + \beta T'_{i,j,k} + \gamma G'_{i,j,k} = 0 \tag{96-f}$$

$$S'_{i,j,k} - S_{i,j,k} + \beta U'_{i,j,k} + \alpha R'_{i,j,k} = 0 \tag{96-g}$$

公式（96－a … g）和（89－a … m）构成一个由20个方程组成的方程组，其中包含有20个未知数。它们的解就是矩阵[A+B]，[L]和[U]的各项元素，并满足[A+B]“接近”[A]，并可以很容易地分解为[L]和[U]的条件。[L]和[U]均为稀疏矩阵，并具有所要求的上下三角矩阵的形式。例如将公式（89－a，－b，－c）代入（96－a）后可得:

$$a_{i,j,k} = Z_{i,j,k} /(1+\alpha e_{i,j,k-1} + \beta f_{i,j,k-1}) \tag{97-a}$$

同理:

$$b_{i,j,k} = B_{i,j,k} /(1+\gamma e_{i-1,j,k} + \beta g_{i-1,j,k}) \tag{97-b}$$

$$c_{i,j,k} = D_{i,j,k} /(1+\gamma f_{i,j-1,k} + \alpha g_{i,j-1,k}) \tag{97-c}$$

$$A'_{i,j,k} = a_{i,j,k} e_{i,j,k-1} \tag{97-d}$$

$$C'_{i,j,k} = e_{i-1,j,k} b_{i,j,k} \tag{97-e}$$

$$G'_{i,j,k} = f_{i,j-1,k} c_{i,j,k} \tag{97-f}$$

$$R'_{i,j,k} = g_{i,j-1,k} c_{i,j,k} \tag{97-g}$$

$$T'_{i,j,k} = a_{i,j,k} f_{i,j,k-1} \tag{97-h}$$

$$U'_{i,j,k} = b_{i,j,k} g_{i-1,j,k} \tag{97-i}$$

$$\begin{aligned} d_{i,j,k} = {} & E_{i,j,k} + \alpha A'_{i,j,k} + \beta T'_{i,j,k} + \gamma C'_{i,j,k} + \gamma G'_{i,j,k} + \beta U'_{i,j,k} \\ & + \alpha R'_{i,j,k} - a_{i,j,k} g_{i,j,k-1} - b_{i,j,k} f_{i-1,j,k} - e_{i,j-1,k} c_{i,j,k} \end{aligned} \tag{97-j}$$

$$e_{i,j,k} = (F_{i,j,k} - \alpha A'_{i,j,k} - \gamma C'_{i,j,k}) / d_{i,j,k} \tag{97-k}$$

$$f_{i,j,k} = (H_{i,j,k} - \beta T'_{i,j,k} - \gamma G'_{i,j,k}) / d_{i,j,k} \tag{97-l}$$

$$g_{i,j,k} = (S_{i,j,k} - \alpha R'_{i,j,k} - \beta U'_{i,j,k}) / d_{i,j,k} \tag{97-m}$$

当[L]和[U]的元素确定之后，公式（8）中的[A+B]就可以由[L][U]来代替:

$$[L][U]\{h^l - h^{l-1}\} = \{q\} - [A]\{h^{l-1}\} \tag{98}$$

在这里，l 表示当前的迭代水平；$l-1$ 表示前一次迭代水平。下面我们定义矢量{RES^l}:

$$\{RES^l\} = \{q\} - [A]\{h^{l-1}\} \tag{99}$$

并就此将式（98）重写为:

$$[L][U]\{h^l - h^{l-1}\} = \{RES^l\} \tag{100}$$

到这里，公式（100）就可以用前后替换法求解了。第一步是利用向前替换法对矢量{v}求解:

$$[L]\{v\} = \{RES^l\} \tag{101}$$

其中$\{v\}=[U]\{h^l-h^{l-1}\}$。{v}确定之后，再利用向后替换法求出下式中的$\{h^l-h^{l-1}\}$:

$$[U]\{h^l - h^{l-1}\} = \{v\} \tag{102}$$

在前面的讨论中，我们采用计算单元的下标来表示矩阵的元素或公式中的系数（参见图53－a）。为了更清楚地演示向前替换的过程，在下面的讨论过程中，我们将采用单一下标来表示矩阵元素（图53－b）。因为[L]中主对角线以上的元素均为零，所以由公式（101）表示的第一个线性方程为:

$$d_1 v_1 = RES_1^l \tag{103}$$

在公式（103）中，d_1项已由公式（97－j）确定，并且$RES_1{}^l$已由公式（99）作为矢量（RES^l）中的一个元素求出。因此，v_1的值可从公式（103）中获得。由公式（101）表示的第二个方程为:

$$c_2 v_1 + d_2 v_2 = RES_2^l \tag{104}$$

同样，c_2和d_2已在公式（97）中解出，而$RES_2{}^l$则可从（99）式中得到。利用从(103)式得到的v_1的值，则可从公式（104）中得到v_2。

{v}中元素的一般表达式为:

$$v_n = (RES_n^l - a_n v_{n-NRC} - b_n v_{n-NCOL} - c_n v_{n-1})/d_n \tag{105}$$

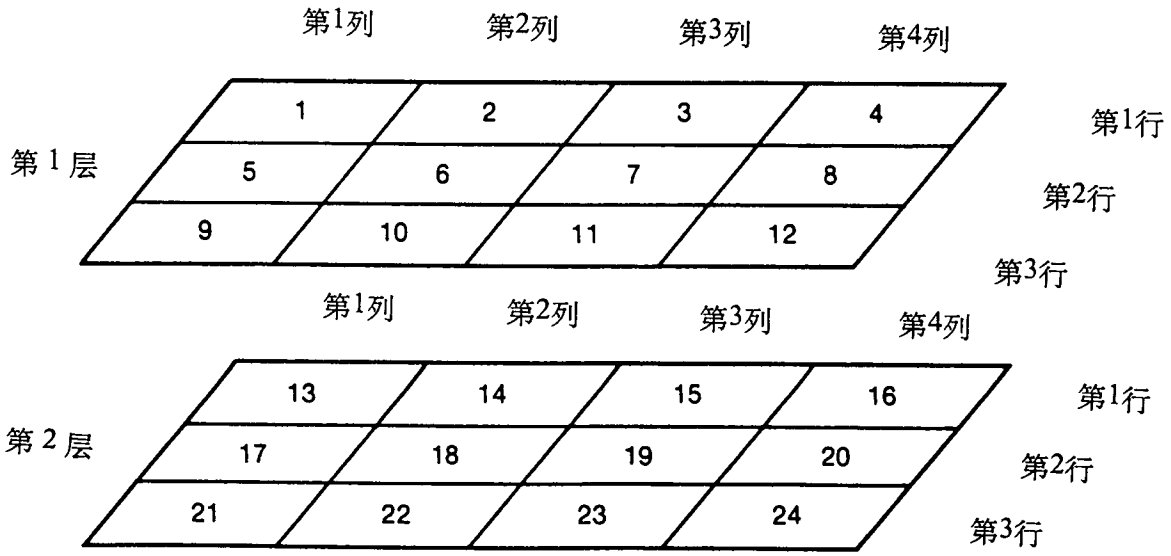
其中NRC为当前层中计算单元的数目；NCOL为模型的列数，系数a_n、b_n、c_n和d_n均可由（97）式求出，而$RES_n{}^l$则通过（99）式得到。在第一和第二个公式（式（103）和（104））中，系数a_n和b_n等于零。在计算过程中，所需{v}的元素则在前一步运算中求得。这个过程叫做向前替换。在向前替换过程中{v}的元素可依次求得，因为矩阵[L]是一个下三角矩阵，其右上侧的元素均为零。

下一步是采用向后替换法利用已知{v}的元素，计算矢量$\{h^l\text{-}h^{l-1}\}$，也就是对公式（102）求解。向后替换过程与向前替换相似。但在向后替换过程中，由于矩阵[U]为上三角矩阵，故计算顺序与向前替换相反。求出矢量$\{h^l\text{-}h^{l-1}\}$后，再与矢量$\{h^{l-1}\}$相加，就可以得到$\{h^l\}$，即本次迭代欲求的水头矢量。

总之，对下面方程组求解的问题:

$$[A]\{h\} = \{q\} \tag{106}$$

(a) 三元下标法



(b) 单一下标法

图53. 使用三元下标法和单一下标法分别表示计算单元在网格中的位置。

可以转化为下面的迭代过程: （1）由公式（97）确定矩阵[L]和[U]; （2）利用矢量{q}，系数矩阵[A]以及前一次迭代得到的水头值，计算出系统残差{RES^l}; （3）对公式（100）进行向前向后替换求出水头残差{h^l-h^{l-1}}; （4）将水头残差{h^l-h^{l-1}}与{h^{l-1}}相加从而得到水头矢量{h^l}。但是，这些仅仅是SIP方法的一些主要步骤，还有其它一些问题需要进一步讨论。

数组转换

前面谈到，系数矩阵[A]为一稀疏矩阵，其中只有七条对角线上的元素不为零。在计算过程中，我们只需要知道这七条对角线上的元素就行了，而不必将整个矩传递给SIP子程序包。又由于该矩阵的对称性，只要知道主对角线以及主对角线以下三条非零对角线上的元素就足够了。这三条非零对角线所相对应的水力传导系数数组分别为: CC，CR和CV。主对角线上的元素由这三个数组以及HCOF数组(参见第二章)构成。线性方程组的右侧项，矢量{q}，则对应于数组RHS（参见第二章）。前一次迭代后所得到的水头分布数组{h^{l-1}}则对应于数组HNEW。每次迭代完成之后，HNEW的内容也更新一次。根据这些，SIP子程序包所要求的数据包括: CC，CR，CV，RHS，HCOF和HNEW，而该子程序包的输出则包括新计算的结果HNEW。如在第三章中谈到的，迭代循环中包括“建立方程”的步骤，故这些输入数组的内容可能在每次迭代后发生变化。

计算顺序

实际经验表明，如果对有限差分方程组的迭代求解过程沿两种不同方向交替进行，便可以减少迭代次数。在这里，第一个方向的迭代从第一列，第一行，第一层开始，并沿列，行和层数增加的方向进行。另一个迭代方向可以选择从最底层的最后一行和第一列开始，并沿列增加而层数和行数减少的方向进行。按照图51所定义的对角线的名称，我们可以写出一组与公式（97－a …m）类似的方程:

$$a_{i,j,k} = Z_{i,j,k} /(1+\alpha e_{i,j,k+1} + \beta f_{i,j,k+1}) \tag{107-a}$$

$$b_{i,j,k} = B_{i,j,k} /(1+\gamma e_{i+1,j,k} + \beta g_{i+1,j,k}) \tag{107-b}$$

$$c_{i,j,k} = D_{i,j,k} /(1+\gamma f_{i,j-1,k} + \alpha g_{i,j-1,k}) \tag{107-c}$$

$$A'_{i,j,k} = a_{i,j,k} e_{i,j,k+1} \tag{107-d}$$

$$C'_{i,j,k} = e_{i+1,j,k} b_{i,j,k} \tag{107-e}$$

$$G'_{i,j,k} = f_{i,j-1,k} c_{i,j,k} \tag{107-f}$$

$$R'_{i,j,k} = g_{i,j-1,k} c_{i,j,k} \tag{107-g}$$

$$T'_{i,j,k} = a_{i,j,k} f_{i,j,k+1} \tag{107-h}$$

$$U'_{i,j,k} = b_{i,j,k} g_{i+1,j,k} \tag{107-i}$$

$$\begin{aligned} d_{i,j,k} = {} & E_{i,j,k} + \alpha A'_{i,j,k} + \beta T'_{i,j,k} + \gamma C'_{i,j,k} + \gamma G'_{i,j,k} + \beta U'_{i,j,k} \\ & + \alpha R'_{i,j,k} - a_{i,j,k} g_{i,j,k+1} - b_{i,j,k} f_{i+1,j,k} - e_{i,j-1,k} c_{i,j,k} \end{aligned} \tag{107-j}$$

$$e_{i,j,k} = (F_{i,j,k} - \alpha A'_{i,j,k} - \gamma C'_{i,j,k}) / d_{i,j,k} \tag{107-k}$$

$$f_{i,j,k} = (H_{i,j,k} - \beta T'_{i,j,k} - \gamma G'_{i,j,k}) / d_{i,j,k} \tag{107-l}$$

$$g_{i,j,k} = (S_{i,j,k} - \alpha R'_{i,j,k} - \beta U'_{i,j,k}) / d_{i,j,k} \tag{107-m}$$

在MODFLOW中，公式（107－a…m）和公式（97－a…m）的计算依次交替进行。事实上，MODFLOW中仅使用一组通用公式进行计算。这些公式中的变量名由单一下标表示。通过将值按顺序赋给该下标所代表的元素，公式（97）和（107）的求解顺序也就可以确定了。在这组通用公式中，下标n l l表示对应于刚计算过的层中与第n个元素具有相同行号和列号的元素；下标n r l和n c l也按这种方式定义。另外，在这些公式中，迭代参数α，β和γ均用ω表示。下面我们还将进一步讨论参数ω。注意下列的公式中有一个新的方程用来计算v n（矢量{v}中对应于计算单元n的元素）。一旦矩阵[L]和[U]的第n行计算完成之后，v n就可以求得了。这些公式为:

$$a_n = Z_n / (1 + \omega(e_{nll} + f_{nll})) \tag{108-a}$$

$$b_n = B_n / (1 + \omega(e_{nrl} + g_{nrl})) \tag{108-b}$$

$$c_n = D_n / (1 + \omega(f_{ncl} + g_{ncl})) \tag{108-c}$$

$$A'_n = a_n e_{nll} \tag{108-d}$$

$$C'_n = b_n e_{nrl} \tag{108-e}$$

$$G'_n = c_n f_{ncl} \tag{108-f}$$

$$R'_n = c_n g_{ncl} \tag{108-g}$$

$$T'_n = a_n f_{nll} \tag{108-h}$$

$$U'_n = b_n g_{nrl} \tag{108-i}$$

$$d_n = E_n + \omega(A'_n + T'_n + C'_n + G'_n + U'_n + R'_n) - a_n g_{nll} - b_n f_{nrl} - c_n e_{ncl} \tag{108-j}$$

$$e_n = (F_n - \omega(A'_n + C'_n))/d_n \tag{108-k}$$

$$f_n = (H_n - \omega(T'_n + G'_n))/d_n \tag{108-l}$$

$$g_n = (S_n - \omega(R'_n + U'_n))/d_n \tag{108-m}$$

$$v_n = (RES_n - a_n v_{nll} - b_n v_{nrl} - c_n v_{ncl})/d_n \tag{108-n}$$

由于向后替换时要用到e_n，f_n，g_n和v_n的值，SIP子程序包内专门设置了四个数组来存放它们的值。这四个数组的大小等于模型内所有计算单元的数目。

迭代参数

虽然Weinstein，Stone和Kwan（1969）在他们的文章中定义了三个迭代参数，但实际上他们仅使用了一个迭代参数。因此，公式（93）中的α，β，和γ可以用一个参数ω来代表。这个参数用来与公式（93）的右侧各项相乘。但为了达到一定的收敛速度，ω应取几个不同的值，并在迭代过程中循环使用这些值。在MODFLOW中，ω的值由下式确定:

$$\omega(\lambda) = 1 - (WSEED)^{(\lambda-1)/(NPARM-1)} \qquad \lambda=1,2,\cdots,\text{NPARM} \tag{109}$$

其中NPARM为欲使用的ω值的数目；λ为指标数，取值从1到NPARM；ω（λ）为与λ相对应的迭代参数；WSEED称为迭代参数之种子（SEED），它是确定迭代参数ω的基础。其取值按下面的规定进行。

下面我们讨论WSEED的确定。首先根据各个计算单元与其相邻单元之间水力传导系数求出三个参数：ρ_1，ρ_2，和ρ_3:

$$\rho_1 = \frac{CC_{\max} + CV_{\max}}{CR_{\min}} \tag{110}$$

$$\rho_2 = \frac{CR_{\max} + CV_{\max}}{CC_{\min}} \tag{111}$$

$$\rho_3 = \frac{CR_{\max} + CC_{\max}}{CV_{\min}} \tag{112}$$

其中：相对于计算单元（i，j，k），CC_{max}为沿列方向的水力传导系数$CC_{i-1/2,j,k}$和$CC_{i+1/2,j,k}$中间较大者，CC_{min}两者中间较小者；同样，CR_{max}为沿行方向的水力传导系数$CR_{i,j-1/2,k}$和$CR_{i,j+1/2,k}$中间较大者,CR_{min}为二者中间较小者；CR_{max}为$CV_{i,j,k-1/2}$和$CV_{i,j,k+1/2}$中间较大者,CV_{min}为二者中间较小者。利用这些值，我们可以对每个计算单元计算出下列值:

$$\frac{\pi^2}{2(NCOL)^2(1+\rho_1)}, \quad \frac{\pi^2}{2(NROW)^2(1+\rho_2)}, \quad \frac{\pi^2}{2(NLAY)^2(1+\rho_3)}$$

其中NCOL为模型的列数，NROW为行数，NLAY为层数。这三个数中最小者就是该单元之“种子”，而WSEED为所有单元之种子的平均值。对于不同的λ，可得到不同的ω值。迭代参数ω（λ）依次用于迭代计算，并循环使用（每NPARM次为一周期）。

上述对ω值的计算过程与Weinstein，Stone和Kwan（1969）所提出的方法略有不同，但得到的结果类似，并在许多实际问题中使用效果良好。但是，在迭代参数的选择过程中，我们还应当注意以下几点。第一，上述的计算方法完全是经验性的，我们还不清楚为什么某些参数的组合会优于另外一些组合。第二，所选择的参数可能会影响收敛速度，但并不影响最终计算结果（如果迭代收敛的话）。第三，所选参数对收敛速度的影响可能会很大。

MODFLOW中还采用了另外一个迭代参数。为区别于ω，称之为迭代加速因子，在程序中用ACCL来表示。它用作$\{RES^l\}$的乘积因子。当ACCL等于1时，它实际不起作用。虽然Weinstein，Stone和Kwan（1969）在他们的算法中并没有这样一个参数，但已有不少人在不同版的程序中引入这样的加速因子（如Peaceman，1977，第130页）。迭代加速因子并不象ω那样循环使用，而是由用户定义一个确定的值并预先输入给程序。一般来说，ACCL的取值在开始时为1，并通过调整种子的值来改善

收敛速度。下面我们还将进一步讨论这个问题。如果无法克服不收敛的问题，再通过调整ACCL的值进行试验。

经验表明，加速因子ACCL取1并使用程序计算的种子时，不一定能得到最好的收敛速度（即达到收敛的迭代次数为最少）。当每次迭代求得的水头变化的绝对值太大或太小时，收敛速度一般不会是最优。当水头变化太大时，计算的水头值会大大超过正确的水头值（这种现象称为过量,overshoot），因此水头会不断进行调整以补偿这种偏大的计算水头变化。这样就会出现振荡的情况（oscillation）。过量严重时可以造成迭代发散，而中等程度的过量则会降低收敛速度。当水头变化太小时，则可能出现相反的问题：计算的水头会单调地向正确值缓慢趋近。在严重的情况下，迭代过程可能因计算的水头变化小于收敛指标而停止，但事实上，所计算的水头仍远远偏离正确值。在这种情况下，水均衡计算会显示出巨大的误差。

Weinstein，Stone和Kwan（1969）建议使用试算法来确定种子的取值。实际工作中可以这样做：首先定ACCL为1，然后用程序计算的或根据经验估计的种子值进行初步计算。在同一个时间段内，注意观察每次迭代后水头变化的趋势。由于迭代参数的循环使用，一般情况下先后两次迭代所计算出的水头变化量会有所不同。但从每次迭代后得到的水头变化量仍不难看出其变化的总趋势，即水头变化值随着迭代次数的增加而上升或下降的趋势。这种总趋势正是我们所感兴趣的。这种趋势一般在迭代试验的后期比较明显。某种程度的振荡现象（即正负交替）是正常的。但持续性振荡则表明过量的情况，即计算的水头变化太大。另一方面，当水头变化太小时，则可观察到非常平缓的水头变化趋势。为适当评估所观察到的水头变化趋势，试算过程中的迭代次数至少应为所选用的种子数目的4至5倍，除非，在此之前迭代已经收敛。

初步试算完成之后则可根据水头变化的情况对种子加以调整：如果水头变化太大，则增加其值2至10倍，反之则减少2至10倍。如果水头变化的趋势不明显，则可试将种子乘以一个数或除以一个数。无论哪种情况，都应进行第二次试算。与第一次试算一样，注意观察计算水头变化情况，并与第一次试算的情况加以比较。如果两次试算都达到收敛则应比较各次的收敛次数。如果两次试算都不收敛，则可比较最后一次迭代后得到的水头变化量。试算过程可以持续下去，以对种子值进行微调。一般来说，每次试算时对种子的改变

量应当逐步减小。但应注意，一般不必进行太多次的试算。每次试算时对种子值的改变量不应小于2倍。

多数情况下，利用这种方法得到的种子值可以满足一个具体的数值模型的计算。即使边界条件，外应力，甚至差分网格发生了变化，也不必对已确定的种子值再做调整。但是，如果这些变化导致了模型不收敛，则应当重新选择种子。应当指出，系数矩阵的对角性（diagonal dominant)越强，解对种子选择的依赖程度也就越小。所以，源汇项（如蒸发蒸腾、河流渗流等）的使用仅影响主对角线上的元素，可增加解的稳定性。因此，源汇项的增加可提高解的稳定性。

每次迭代计算完成后，程序将$|\Delta h_l|_{max}$的值记录下来。这里$|\Delta h_l|_{max}$指的是在所有计算单元中，经过本次迭代计算后得到的水头变化的最大绝对值。每个应力期结束时，程序将各次迭代后得到的$|\Delta h_l|_{max}$打印在标准输出文件中供用户参考。用户也可以要求程序在每个时间段结束时输出$|\Delta h_l|_{max}$的值。除了$|\Delta h_l|_{max}$之外，MODFLOW还同时列出$|\Delta h_l|_{max}$所在的位置（即 i， j， k）。最大水头变化的方向则由正负号表示。这些信息可供用户在前面讲到的试算过程中参考。当然，我们假定这些$|\Delta h_l|_{max}$ 值能够反映所在点的水头变化 情况。

通过调整迭代加速因子，ACCL，还可以提高收敛速度。加大ACCL的值可以增加每次迭代计算出的水头变化量，而减小ACCL的值则可减小水头的变化量。以前谈到的试算法同样可以用来对ACCL的取值进行优化。但不能在同一次试算过程中同时调整种子和ACCL的值。

有些情况下，为避免由于计算水头变化太大而导致干枯计算单元的出现，我们希望减慢收敛速度。这时，最优的收敛速度并不能单纯以迭代次数最少来衡量，而是在不造成水头变化过量的前提之下的最小迭代次数。理解了这点差别之后，我们就可以使用前面提到的试算法对种子和ACCL的取值进行优化了。

SIP子程序包输入数据及格式

SIP子程序包所需要的各项数据从IUNIT(9)所指定的设备号读入。

对每次模拟:

由子程序SIP1AL读入的数据包括

1.数据名称:	MXITER	NPARM
输入格式:	I10	I10

由子程序SIP1RP读入的数据包括

2.数据名称:	ACCL	HCLOSE	IPCALC	WSEED	IPRSIP
输入格式:	F10.0	F10.0	I10	F10.0	I10

输入数据说明

MXITER: 最大迭代次数,通常取值50。

NPARM: 迭代参数的数目,通常取值5。

ACCL: 加速因子。大于或等于零。通常取值为1。如ACCL=0,则无水头变化。

HCLOSE: 收敛指标。当计算的最大水头变化的绝对值小于此值时,迭代结束。它的选值将影响解的精度。

IPCALC: 确定种子方式标识符:

IPCALC =0: 种子由用户输入;

IPCALC =1: 由程序自行计算种子值。

WSEED: 计算迭代参数之种子值。仅当IPCALC=0时需具体指定。

IPRSIP: 打印频率控制。当时间段数为IPRSIP的倍数时,MODFLOW将各次迭代后最大水头变化(正或负)的结果写入标准输出文件。如果IPSRIP=0,MODFLOW将重新赋值为999。无论IPRSIP取什么值,在每个应力期结束时,MODFLOW总会打印迭代的情况。

SIP子程序包输入样单 (由用户选择种子)

数据项	解释	输入记录
1	{MXITER, NPARM}	50 5
2	{ACCL,HCLOSE,IPCALC,WSEED,IPRSIP}	1. 0.01 0 .98 10

SIP子程序包输入样单 (由程序计算种子)

数据项	解释	输入记录
1	{MXITER, NPARM}	100 6
2	{ACCL,HCLOSE,IPCALC,WSEED,IPRSIP}	1. 0.01 1

[译注: 样单中的数据可能不符合格式要求, 仅供参考。]

第十三章 分层逐次超松弛法子程序包

概念及程序化

迭代法对大型线性方程组求解的另一方法是逐次超松弛法。MODFLOW使用了入分层逐次超松弛法(Slice Successive Overrelaxation) (简称为SSOR)子程序包。有关逐次超松弛法的介绍可从很多参考书中找到，包括Peaceman(1977), Crichlow (1977), 和Remson, Horberger and Molz(1971)等早期所著的论文。

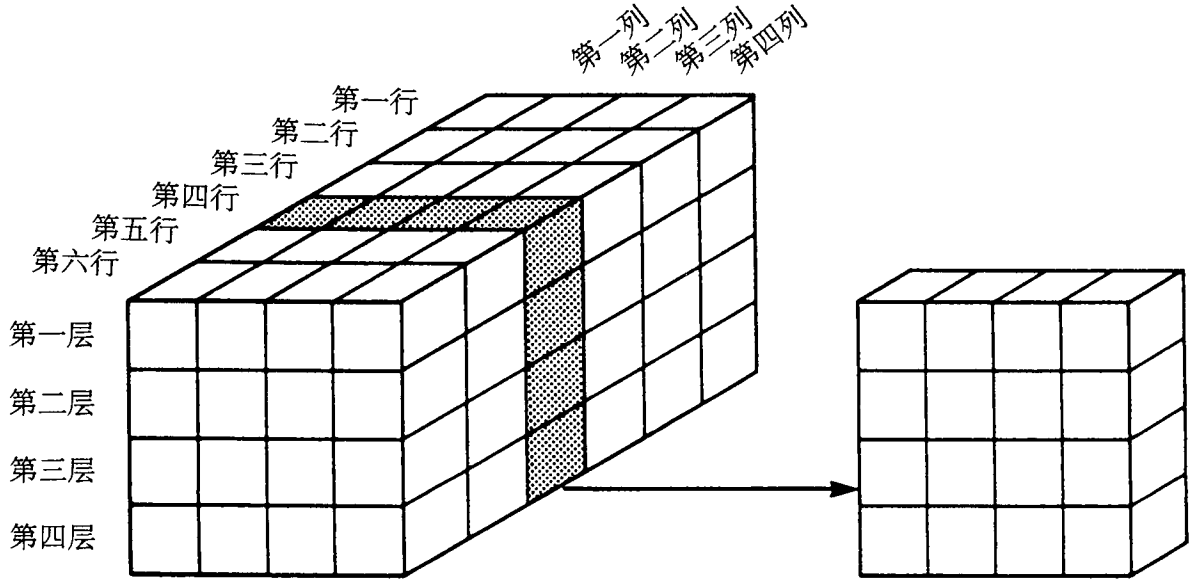
在SSOR子程序包中，逐次超松弛法是将有限差分网格分成“垂向”的“分层”，如图54所示，将各单元的差分方程分组归类，每一分层为一组。每次迭代时，轮流求解这些方程组，每分层都用估计的水头值形成新的一套方程组。求解时，每分层方程组首先用相继两次迭代计算所得的水头差表达。然后该分层的方程组用高斯消元法直接求解，将与其相邻的分层当成已知（即将计算所得的与其相邻的分层的最新水头值作为“已知”值代入正在求解的分层方程组)。然后，将从高斯消元法所求得的水头变化值乘以加速因子，ω（取值范围通常在1~2之间）；该结果被认为是该分层在那一次迭代的最终水头变化值。将这些水头变化值加到上一次迭代所得的各点的水头值上，便求得该分层的该次迭代的最终估计水头值。每分层都同一方法按顺序重复直至三维数组中所有的分层都已处理一遍，才完成一次迭代。然后，用同样的计算顺序逐次算遍各个分层，直至相继的两次迭代的水头之差均小于截止条件才结束。

应当注意，虽然各分层的方程组每次迭代都用一个直接的求解法(高斯消元法)进行求解，但是整个求解不是直接的而是迭代的。基于最新计算求得的与其相邻分层的水头值，每一直接解仅给出该分层水头变化的临时值或估计值；当逐个处理各分层时，计算值继续改变，直至满足截至条件。

上述求解过程可通过计算单元的差分方程加以更加详细的说明。下面是第二章中推导出的单个计算单元的差分方程，但其中添加了第二个上标以表示迭代的次序:

$$CV_{i,j,k-\frac{1}{2}}h^{m,l}_{i,j,k-1}+CC_{i-\frac{1}{2},j,k}h^{m,l}_{i-1,j,k}+CR_{i,j-\frac{1}{2},k}h^{m,l}_{i,j-1,k}$$

$$+(-CV_{i,j,k-\frac{1}{2}}-CC_{i-\frac{1}{2},j,k}-CR_{i,j-\frac{1}{2},k}-CR_{i,j+\frac{1}{2},k}-CC_{i+\frac{1}{2},j,k}-CV_{i,j,k+\frac{1}{2}}$$



完整的网格　　第四分层

图54. 在SSOR子程序包中，将三维模型分解为垂向分层进行处理。

$$+HCOF_{i,j,k})h_{i,j,k}^{m,l}+CR_{i,j+\frac{1}{2},k}h_{i,j+1,k}^{m,l}+CC_{i+\frac{1}{2},j,k}h_{i+1,j,k}^{m,l}$$

$$+CV_{i,j,k+\frac{1}{2}}h_{i,j,k+1}^{m,l}=RHS_{i,j,k} \tag{113}$$

在公式(113)中，上标m为当前时间段，而上标l是指迭代次数。若仿照公式(113)写出下一次迭代，l+1，的方程式，从这一新的公式的两边同时减去方程(113)的左边，结果可写成:

$$CV_{i,j,k-\frac{1}{2}}(h_{i,j,k-1}^{m,l+1}-h_{i,j,k-1}^{m,l})+CC_{i-\frac{1}{2},j,k}(h_{i-1,j,k}^{m,l+1}-h_{i-1,j,k}^{m,l})$$

$$+CR_{i,j-\frac{1}{2},k}(h_{i,j-1,k}^{m,l+1}-h_{i,j-1,k}^{m,l})+(-CV_{i,j,k-\frac{1}{2}}-CC_{i-\frac{1}{2},j,k}-CR_{i,j-\frac{1}{2},k}$$

$$-CV_{i,j,k+\frac{1}{2}}-CC_{i+\frac{1}{2},j,k}-CR_{i,j+\frac{1}{2},k}+HCOF_{i,j,k})(h_{i,j,k}^{m,l+1}-h_{i,j,k}^{m,l})$$

$$+CR_{i,j+\frac{1}{2},k}(h_{i,j+1,k}^{m,l+1}-h_{i,j+1,k}^{m,l})+CC_{i+\frac{1}{2},j,k}(h_{i+1,j,k}^{m,l+1}-h_{i+1,j,k}^{m,l})$$

$$+CV_{i,j,k+\frac{1}{2}}(h_{i,j,k+1}^{m,l+1}-h_{i,j,k+1}^{m,l})=RHS_{i,j,k}$$

$$-CV_{i,j,k-\frac{1}{2}}h_{i,j,k-1}^{m,l}-CC_{i-\frac{1}{2},j,k}h_{i-1,j,k}^{m,l}-CR_{i,j-\frac{1}{2},k}h_{i,j-1,k}^{m,l}$$

$$-(-CV_{i,j,k-\frac{1}{2}}-CC_{i-\frac{1}{2},j,k}-CR_{i,j-\frac{1}{2},k}-CR_{i,j+\frac{1}{2},k}-CC_{i+\frac{1}{2},j,k}-CV_{i,j,k+\frac{1}{2}}$$

$$+HCOF_{i,j,k})h_{i,j,k}^{m,l}-CR_{i,j+\frac{1}{2},k}h_{i,j+1,k}^{m,l}-CC_{i+\frac{1}{2},j,k}h_{i+1,j,k}^{m,l}-CV_{i,j,k+\frac{1}{2}}h_{i,j,k+1}^{m,l} \tag{114}$$

公式(114)中未知项当成是第l次迭代与第l+1次迭代之间的计算水头变化值—例如，$\left(h_{i,j,k}^{m,l+1}-h_{i,j,k}^{m,l}\right)$。注意，当第$l$次迭代完成时，(114)式的右边全由已知项组成－它包括 RHS 和联立求解得出的水力传导系数，以及第 l 次迭代后已获得的水头估计值。

现在假定我们按图54所示沿行将模型分成垂向的分层，将每个单独分层中所有计算单元的方程式分离出来－例如，从图54中将三维数组的第四行作为第四分层。用公式(114)来说，当处理第i行所对应的第i分层时，我们仍然将该分层各单元的水头变化当作未知项，但是认为与其相邻的分层的单元的水头变化为已知项。因此，公式(114)左边项中的两项，

$CC_{i-1/2,j,k}(h_{i-1,j,k}^{m,l+1}-h_{i-1,j,k}^{m,l})$ 和 $CC_{i+1/2,j,k}(h_{i+1,j,k}^{m,l+1}-h_{i+1,j,k}^{m,l})$，此时已作为已知量处理。如果将这两项移至方程的右边并重新整理，我们则已经消去 $h_{i-1,j,k}^{m,l}$ 和 $h_{i+1,j,k}^{m,l}$ 这两项，仅留下:

$$
\begin{aligned}
&CV_{i,j,k-\frac{1}{2}}(h_{i,j,k-1}^{m,l+1}-h_{i,j,k-1}^{m,l})+CR_{i,j-\frac{1}{2},k}(h_{i,j-1,k}^{m,l+1}-h_{i,j-1,k}^{m,l})\\
&+(-CV_{i,j,k-\frac{1}{2}}-CC_{i-\frac{1}{2},j,k}-CR_{i,j-\frac{1}{2},k}\\
&-CV_{i,j,k+\frac{1}{2}}-CC_{i+\frac{1}{2},j,k}-CR_{i,j+\frac{1}{2},k}+HCOF_{i,j,k})(h_{i,j,k}^{m,l+1}-h_{i,j,k}^{m,l})\\
&+CR_{i,j+\frac{1}{2},k}(h_{i,j+1,k}^{m,l+1}-h_{i,j+1,k}^{m,l})+CV_{i,j,k+\frac{1}{2}}(h_{i,j,k+1}^{m,l+1}-h_{i,j,k+1}^{m,l})=RHS_{i,j,k}\\
&-CV_{i,j,k-\frac{1}{2}}h_{i,j,k-1}^{m,l}-CC_{i-\frac{1}{2},j,k}h_{i-1,j,k}^{m,l+1}-CR_{i,j-\frac{1}{2},j,k}h_{i,j-1,k}^{m,l}\\
&-(-CV_{i,j,k-\frac{1}{2}}-CC_{i-\frac{1}{2},j,k}-CR_{i,j-\frac{1}{2},j,k}-CR_{i,j+\frac{1}{2},j,k}-CC_{i+\frac{1}{2},j,k}-CV_{i,j,k+\frac{1}{2}}\\
&+HCOF_{i,j,k})h_{i,j,k}^{m,l}-CR_{i,j+\frac{1}{2},k}h_{i,j+1,k}^{m,l}-CC_{i+\frac{1}{2},j,k}h_{i+1,j,k}^{m,l+1}-CV_{i,j,k+\frac{1}{2}}h_{i,j,k+1}^{m,l}
\end{aligned}
\tag{115}
$$

现在假定，分层的处理是以行数i增加的顺序进行的；对于每次迭代在第i分层的计算开始之前，第i-1分层的计算已经完成。这样说来，当第i分层的第i+1次迭代处理开始时，$h_{i-1,j,k}^{m,l+1}$ 的值是已知的，而 $h_{i+1,j,k}^{m,l+1}$ 的值则是未知。因此，处理第i分层时，$CC_{i-1/2,j,k}h_{i-1,j,k}^{m,l+1}$ 可直接作为已知项合并，但 $CC_{i+1/2,j,k}h_{i+1,j,k}^{m,l+1}$ 则是未知的。为了克服这一困难，将上一次迭代得到的 $h_{i+1,j,k}^{m}$ 值，即 $h_{i+1,j,k}^{m,l}$，代替公式(115)右边的 $h_{i+1,j,k}^{m,l+1}$。(这样一来，我们使用的相邻分层的值，是它们最新计算所得的水头值。)由此可得:

$$
\begin{aligned}
&CV_{i,j,k-\frac{1}{2}}(\tilde{h}_{i,j,k-1}^{m,l+1}-h_{i,j,k-1}^{m,l})+CR_{i,j-\frac{1}{2},k}(\tilde{h}_{i,j-1,k}^{m,l+1}-h_{i,j-1,k}^{m,l})\\
&+(-CV_{i,j,k-\frac{1}{2}}-CC_{i-\frac{1}{2},j,k}-CR_{i,j-\frac{1}{2},k}-CR_{i,j+\frac{1}{2},k}-CC_{i+\frac{1}{2},j,k}-CV_{i,j,k+\frac{1}{2}}\\
&+HCOF_{i,j,k})(\tilde{h}_{i,j,k}^{m,l+1}-h_{i,j,k}^{m,l})+CR_{i,j+\frac{1}{2},k}(\tilde{h}_{i,j+1,k}^{m,l+1}-h_{i,j+1,k}^{m,l})\\
&+CV_{i,j,k+\frac{1}{2}}(\tilde{h}_{i,j,k+1}^{m,l+1}-h_{i,j,k+1}^{m,l})=RHS_{i,j,k}-CV_{i,j,k-\frac{1}{2}}h_{i,j,k-1}^{m,l})
\end{aligned}
$$

$$-CC_{i-\frac{1}{2},j,k}h^{m,l+1}_{i-1,j,k}-CR_{i,j-\frac{1}{2},k}h^{m,l}_{i,j-1,k}$$
$$-(-CV_{i,j,k-\frac{1}{2}}-CC_{i-\frac{1}{2},j,k}-CR_{i,j-\frac{1}{2},k}-CR_{i,j+\frac{1}{2},k}-CC_{i+\frac{1}{2},j,k}-CV_{i,j,k+\frac{1}{2}}$$
$$+HCOF_{i,j,k})h^{m,l}_{i,j,k}-CR_{i,j+\frac{1}{2},k}h^{m,l}_{i,j+1,k}\quad -CC_{i+\frac{1}{2},j,k}h^{m,l}_{i+1,j,k}-CV_{i,j,k+\frac{1}{2}}h^{m,l}_{i,j,k+1}$$

(116)

方程(116)中，$\tilde{h}$表示第(l+1)次迭代时第i分层的水头项。使用这样的记法是为了便于清楚描述求解过程。分层中计算单元的数目为NC*NL，这里NC是模型中的列数，而NL是模型中的层数；每个计算单元都可写出一个与公式(116)类似的方程。因此对NC*NL个未知数，则形成一个包含有NC*NL个方程的方程组。因为层数一般说来是个比较小的数目，方程组中的方程的总数也比较少，通常用高斯消元法直接求解的效果也较为理想(注意用高斯消元法这种直接求解的办法对整个三维模型数组这样大的数组求解通常不可行)。

单个分层i的整套方程可写成矩阵的形式:

$$[\mathrm{A}]_i\{\Delta\tilde{\mathrm{h}}\}_i=\{\mathrm{R}\}_l \tag{117}$$

这里$[A]_i$ 是第i分层的系数矩阵；$\{\Delta\tilde{h}\}_i$是第l次迭代与第l+1次迭代间该分层的计算水头变化值矢量，用$\tilde{h}^{m,l+1}_{i,j,k}-h^{m,l}_{i,j,k}$的估计值表示；$\{R\}_i$是“常量”矢量，用它来表示第i分层的公式(116)右侧各项。

高斯消元法解矩阵式(117)便得到该分层的每个计算单元的水头残差$\tilde{h}^{m,l+1}_{i,j,k}-h^{m,l}_{i,j,k}$。将这些项作为从第$l$次迭代到第$l$+1次的计算水头的变化最初估计值。先将每个值乘以加速因子，ω，并将所得的结果与上一迭代所得到的水头值相加，以求得第l+1次迭代后的最终水头估计值；即:

$$h^{m,l+1}_{i,j,k}=h^{m,l}_{i,j,k}+\omega(\tilde{h}^{m,l+1}_{i,j,k}-h^{m,l}_{i,j,k}) \tag{118}$$

当第i分层中每个单元(j,k)的$h^{m,l+1}_{i,j,k}$值算出后，便开始下一分层i+1的计算处理。所有分层都经过处理后，这一次迭代便完成了。然后开始下一次迭代计算，直至精度达到所定要求。

从图55－a可知，公式(117)中的系数矩阵$[A]_i$ 具有对称性并且呈带状(banded)分布，其半带宽度等于模型的层数。因为矩阵对称性的缘故，只需要存储下三角部分就行了；程序中使用了一个二维数组来存储该矩阵，如图55-b所示。这个二维数组的大小为NL*NC和NL+1。在图示中的例子中，NL=NC=3。

为达最优收敛速度，SSOR中的加速因子常需调整。用户可使与用本书第十二章中所述的调整SIP“种子”值相似的试错法。

[译注：由于SSOR的计算速度较慢，故实际工作中很少使用。]

a_{11}	a_{12}		a_{14}					
a_{12}	a_{22}	a_{23}		a_{25}				
	a_{23}	a_{33}			a_{36}			
a_{14}			a_{44}	a_{45}		a_{47}		
	a_{25}		a_{45}	a_{55}	a_{56}			
		a_{36}		a_{56}	a_{66}		a_{68}	
			a_{47}			a_{77}		
					a_{68}		a_{88}	a_{89}
							a_{89}	a_{99}

(a) 单个分层的系数矩阵

a_{11}	a_{22}	a_{33}	a_{44}	a_{55}	a_{66}	a_{77}	a_{88}	a_{99}
a_{12}	a_{23}		a_{45}	a_{56}			a_{89}	
					a_{68}			
a_{14}	a_{25}	a_{36}	a_{47}					

(b) 存贮矩阵元素的二维数组

图55. 分层方程组的系数矩阵及其在计算机中存贮形式。

第十四章 工具子程序和数据读入

除了主程序以及各个子程序包，MODFLOW还附带有许多工具子程序。这些工具子程序不属于任何一个子程序包，主要用来协助不同的子程序包完成一些输入输出任务。这一类子程序的名称前总冠以“U”以表示工具类（Utility）。MODFLOW共含有八个这样的子程序。它们分别是:

UBUDSV: 将无格式实数数组写入指定的输出文件。该数组包括模型中所有计算单元。

ULASAV: 将无格式实数数组写入指定的输出文件。每次输出模型的一个分层所包含的计算单元。

ULAPRS和ULAPRW: 将一组二维实型数组写入指定的输出文件。每次输出模型的一个分层所包含的计算单元。ULAPRS输出时按列进行（参见图56）：首先输出第一行元素的前N列，然后输出第二行元素的前N列（这里N为一打印行中打印数值的数目）。当各行之前N列都输出之后，再输出第一行的第N+1至2N个元素，然后是第二行的N+1至2N个元素。如此循环下去，直到将该层所有的元素输出完毕为止。而ULAPRW输出时则按行进行：首先输出第一行的所有元素，然后输出第二行的元素，直到该层的所有元素输出完毕为止。数组打印输出的格式列于表2。ULAPRS和ULAPRW仅将数据写入MODFLOW之标准输出文件之中。它们所使用的格式已在第四章输出控制部分列出。

UCOLNO: 专门用来标示由ULAPRS和ULAPRW所输出数据之列号。

U2DREL: 读入二维实型数组。MODFLOW所用的数据多由此子程序读入。

U2DINT: 读入二维整型数组。

U1DREL: 读入一维实型数组。

数组输入说明

在输入数组时，MODFLOW要求在数据之前加一输入控制行。用于读入数组的子程序（包括用来读入二维实型数组的U2DREL、读入二维整型数组的U2DINT以及读入一维实型数组的U1DREL）首先读入该控制行，然后再按控制行的规定读入数组。控制行也从相应的设备号读入。例如，补给子程序包利用子程序U2DREL读入补给通量数组RECH。

```
      1       2       3       4       5       6       7       8       9      10
     11      12      13      14      15      16      17

1  1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79
   1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79

2  1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79
   1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79

3  1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79
   1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79

4  1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79
   1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79

5  1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79
   1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79

6  1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79
   1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79

7  1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79
   1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79
```

ULAPRW输出格式

```
      1       2       3       4       5       6       7       8       9      10

1  1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79

2  1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79

3  1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79

4  1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79

5  1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79

6  1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79

7  1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79

     11      12      13      14      15      16      17

1  1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79

2  1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79

3  1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79

4  1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79

5  1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79

6  1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79

7  1325.79 1325.79 1325.79 1325.79 1325.79 1325.79 1325.79
```

ULAPRS输出格式

图56. 工具子程序ULAPRW和ULAPRS的输出示例。该模型层含有七行十七列。

表2. 工具子程序ULAPRS 和 ULAPRW 使用的打印格式代码

IPRN	格 式
1	11G10.3
2	9G13.6
3	15F7.1
4	15F7.2
5	15F7.3
6	15F7.4
7	20F5.0
8	20F5.1
9	20F5.2
10	20F5.3
11	20F5.4
12	10G11.4

用户在IUNIT（8）的位置的选定整数，也是读入数组RECH的设备号。

输入控制行的格式如下:

(1) 实型数组（U2DREL和U1DREL）:

数据名称: LOCAT CNSTNT (FMTIN) IPRN

输入格式: I10 F10.0 5A4 I10

(2) 整型数组（U2DINT）:

数据名称: LOCAT ICONST (FMTIN) IPRN

输入格式: I10 I10 5A4 I10

LOCAT: 含有输入数据的文件设备号。

如果LOCAT<0: 从与LOCAT符号相反的设备号读入无格式数据;

如果LOCAT=0: 数组各元素为一常数。该常数的值由CNSTNT或ICONST给出。

如果LOCAT>0: 从设备号LOCAT 的输入文件中，按FMTIN的格式输入数据。

CNSTNT/ICONST: 为一常数。它们的作用取决于LOCAT是否等于零。

如果 LOCAT=0: 则该数组各元素的值均等于CNSTNT/ICONST的值;

如果LOCAT≠0且CNSTNT/ICONST不为零时，CNSTNT/ICONST的值作为一个因数。MODFLOW将读入的数据乘以该因数之后再存入相应的数组。如果CNSTNT/ICONST为零，则忽略不计。

FMTIN: 输入数据之格式。仅当LOCAT为正数时，FMTIN才有意义。FMTIN的定义完全按照FORTRAN语言的规定，例如F10.2，I4等。值得注意的是，格式说明符必须置于一对圆括号之中。例如: (15F5.0)，(15I5)等等。忘记将格式说明符放在括号之中，将会导致程序出错。另外，同一组数据应具有相同的输入格式。

IPRN: 数据打印标识符。用来表明是否将读入的数据写入标准输出文件，以供检查，同时，也作为打印输出时的格式代码。MODFLOW仅当LOCAT不为零时才读入该值。如果IPRN为一负数，MODFLOW将不打印读入的数据。当LOCAT≠0并且当IPRN大于零时，IPRN的值同时用作输出格式的代码。注意，各个子程序输出时的格式代码完全不同。此外，如果IPRN超出下列各代码的最大值时，程序将会按IPRN=0的格式输出。例如，如果用户输入的IPRN为15时，数据由子程序U2DREL读入时，MODFLOW将会以IPRN=0所指定的格式(10G11.4)将读入数据写入

标准输出文件。

下面将IPRN的值与其所表示的输出格式列出:

IPRN	U2DREL	U2DINT	U1DREL
0	10G11.4	10I11	10G12.5
1	11G10.3	60I1	
2	9G13.6	40I2	
3	15F7.1	30I3	
4	15F7.2	25I4	
5	15F7.3	20I5	
6	20F7.4		
7	20F5.0		
8	20F5.1		
9	20F5.2		
10	20F5.3		
11	20F5.4		
12	10G11.4		