The Index Mapping Theorems for Kronecker Product

Bu Zhou[*]

Abstract

This paper gives two theorems on the indexes of a series of matrices and their Kronecker product. Applications of the theorems in matrix algebra, spatial matrix calculations and voxel/3D array representation are discussed. R code for computation are provided in the appendix.

Index Terms — Kronecker product, index mapping, matrix algebra, spatial matrix, block matrix, octree, quadtree, spatial array indexing.

1 The Index Mapping Theorem

The first theorem shows the relationship between the elements indexes of a series of matrices and the elements indexes of the Kronecker product matrix.

Let $A = A_1 \otimes A_2 \otimes \cdots \otimes A_n$, where A is a $(l_1 \cdots l_n) \times (m_1 \cdots m_n)$ matrix, A_k is a $l_k \times m_k$ matrix, $k = 1, \cdots, n$, and $\otimes$ represents for the Kronecker product. Let $A(i,j)$ be the (i,j)-th element of A and $A_k(i,j)$ be the (i,j)-th element of A_k, $k = 1, \cdots, n$.

According to the definition of Kronecker product, any element of A can be uniquely factored into elements of A_ks. So for $A(i,j)$, there exists a series of index pairs $\{(i_k, j_k); k = 1, \cdots n\}$, satisfying

$$A(i,j) = A_1(i_1, j_1)A_2(i_2, j_2)\cdots A_n(i_n, j_n). \tag{1}$$

Then we have,

$$i = (i_1 - 1)l_2 \cdots l_n + (i_2 - 1)l_3 \cdots l_n + \cdots + (i_{n-1} - 1)l_n + i_n, \tag{2}$$

$$j = (j_1 - 1)m_2 \cdots m_n + (j_2 - 1)m_3 \cdots m_n + \cdots + (j_{n-1} - 1)m_n + j_n. \tag{3}$$

Proof 1 [Mathematical Induction]:

We only need to prove the result for row index because the result for the column index can be proved with the same argument.

Case $n = 1$:

$i = i_1$, trivial.

Case $n = k$:

[*]School of Finance and Statistics, East China Normal University, Shanghai, 200241, P. R. China. Email: `zhoubu1988@yahoo.com.cn`

[†]The Index Mapping Theorem is a generation of paper [1]'s Lemma 3, which is the special case for a 4×4 matrix. This paper is part of my Master Thesis [2].

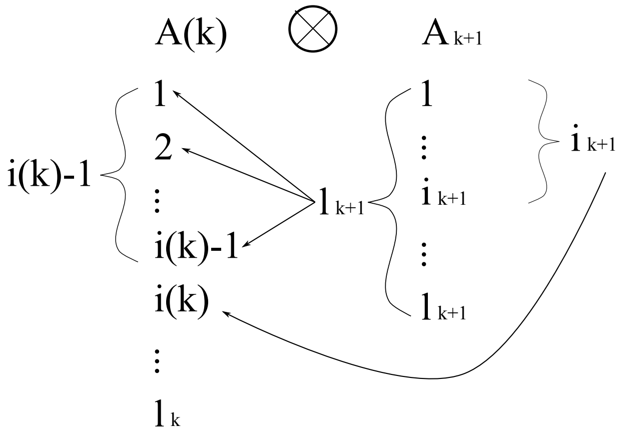

Figure 1:

for $A(k) = A_1 \otimes A_2 \otimes \cdots \otimes A_k$, let $i(k)$ be some row index of $A(k)$, and i_t be the corresponding row index of A_t, $t = 1, \cdots, k$, which satisfy (1). We assume that $i(k) = (i_1 - 1)l_2 \cdots l_k + (i_2 - 1)l_3 \cdots l_k + \cdots + (i_{k-1} - 1)l_k + i_k$.

Case $n = k + 1$:

for $A(k+1) = (A_1 \otimes A_2 \otimes \cdots \otimes A_k) \otimes A_{k+1} = A(k) \otimes A_{k+1}$, let $i(k+1)$ be some row index of $A(k+1)$, there exists an index i_{k+1} of the matrix A_{k+1} satisfying (1). Since $A(k+1) = A(k) \otimes A_{k+1}$, and according to the assumption in the case $n = k$, we have

$i(k+1) = (i(k) - 1)l_{k+1} + i_{k+1} = (i_1 - 1)l_2 \cdots l_{k+1} + (i_2 - 1)l_3 \cdots l_{k+1} + \cdots + (i_k - 1)l_{k+1} + i_{k+1}$.

The first equality above can be verified in Figure 1.

So the result for $n = k + 1$ holds.■

We easily proved the result using mathematical induction. However, this proof is almost mechanical and does not touch the essence of the problem. Use the idea of trees, we can have a direct and intuitive proof which sheds more light on the results.

Proof 2 [Trees]:

Denote the s_k-th row of matrix A_k as $A_k(s_k)$. The procedure of calculating Kronecker products $A(s) = A_1(s_1) \otimes A_2(s_2) \otimes \cdots \otimes A_n(s_n)$ can be viewed as a process of generating a tree with the l_1 parent branches (roots) indexed as

$$(1), \cdots (l_1),$$

and first level children branches indexed as

$$\underbrace{\underbrace{(1,1),(1,2),\cdots,(1,l_2)}_{l_2}, \underbrace{(2,1),(2,2),\cdots,(2,l_2)}_{l_2}, \cdots, \underbrace{(l_1,1),(l_1,2),\cdots,(l_1,l_2)}_{l_2}}_{l_1},$$

second level children branches indexed as

$$\underbrace{(1,1,1),(1,1,2),\cdots,(1,1,l_3)}_{l_3}, \underbrace{(1,2,1),(1,2,2),\cdots,(1,2,l_3)}_{l_3}, \cdots, \underbrace{(l_1,l_2,1),(l_1,l_2,2),\cdots,(l_1,l_2,l_3)}_{l_3},$$

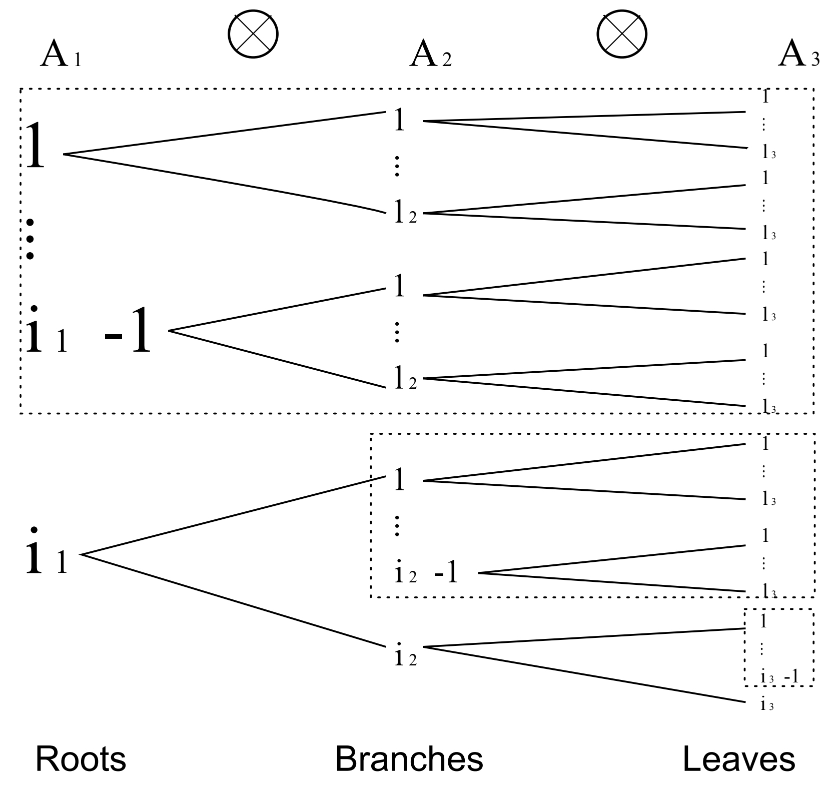

Figure 2: A three-level tree

$$\vdots$$

final level children branches (leaves) indexed as

$$\underbrace{(1,\cdots,1,1)}_{n},\underbrace{(1,\cdots,1,2)}_{n},\cdots,\underbrace{(l_1,\cdots,l_{n-1},l_n)}_{n}.$$

An example of a three-level tree is showed in Figure 2. Now given a leaf index $(i_1,\cdots,i_n)$ in the tree, we want to know its index i in the product $A(s)$. To find the value of i, we only need to count the number of the leaves before the leaf indexed as $(i_1,i_2,\cdots,i_n)$ in the tree.

In the first level, $k=1$, leaves generated from parent branches $(1),\cdots,(i_1-1)$, which will produce $(i_1-1)l_2\cdots l_n$ leaves, comes before the leaf $(i_1,i_2,\cdots,i_n)$.

For i_1, in the second level, $k=2$, leaves generated from branches $(2,1),\cdots,(2,i_2-1)$, which will produce $(i_2-1)l_3\cdots l_n$ leaves, comes before the leaf $(i_1,i_2,\cdots,i_n)$.

$\vdots$

In the final level, $k=n$, leaves $(i_1,i_2,\cdots,i_{n-1},1),\cdots,(i_1,i_2,\cdots,i_{n-1},i_n-1)$, comes before the leaf $(i_1,i_2,\cdots,i_n)$, that's (i_n-1) leaves.

So the total number of leaves before the leaf $(i_1,i_2,\cdots,i_n)$ is $(i_1-1)l_2\cdots l_n+(i_2-1)l_3\cdots l_n+\cdots+(i_{n-1}-1)l_n+i_n-1$,

that's to say, $i=(i_1-1)l_2\cdots l_n+(i_2-1)l_3\cdots l_n+\cdots+(i_{n-1}-1)l_n+i_n$.$\blacksquare$

For simplicity, in the above theorem, let's call matrix A the Kronecker product (matrix), and A_ks the factor matrices.

This theorem can be used to calculate the index of a Kronecker product's element from the indexes of the factor matrices elements.

2 The Inverse Index Mapping Theorem

Since the $A_k(i_k, j_k)$s in (1) are unique, give a index (i, j) in the Kronecker product A, we can also find all the corresponding indexes of the factor matrices A_k by the algorithm described below.

In (2) and (3), suppose $i, j, l_1, \cdots, l_n, m_1, \cdots, m_n$ are known, now we can also calculate $i_1, \cdots, i_n$ and $j_1, \cdots, j_n$.

Denote % as the mod operator: if $i_c \le l_c$, $i\%l_c = i_c$. Note that it's different from the common mod operator because for simplicity purpose, we set $i_c\%l_c = i_c \ne 0$ if $i_c = l_c$.

For rows, note that $i_c \le l_c$, for $c = 1, \cdots, n$, from (1), we have

$i_n = i\%l_n$,

$i_{n-1} = ((i - i_n)/l_n)\%l_{n-1} + 1$,

$i_{n-2} = ((i - i_n - (i_{n-1} - 1)l_n)/(l_n l_{n-1}))\%l_{n-2} + 1$,

$\vdots$

$i_2 = ((i - i_n - (i_{n-1} - 1)l_n - \cdots - (i_3 - 1)(l_n l_{n-1} \cdots l_4))/(l_n l_{n-1} \cdots l_3))\%l_2 + 1$

$i_1 = ((i - i_n - (i_{n-1} - 1)l_n - \cdots - (i_2 - 1)(l_n l_{n-1} \cdots l_3))/(l_n l_{n-1} \cdots l_2)) + 1$,

similarly, for columns,

$j_n = j\%l_n$,

$j_c = ((i - i_n - \Sigma_{v=c+1}^{n-1}[(i_v - 1)\Pi_{w=v+1}^{n} m_w])/\Pi_{v=c+1}^{n} m_v)\%l_c + 1$, for $c = 2, \cdots n - 1$,

$j_1 = ((i - i_n - \Sigma_{v=c+1}^{n-1}[(i_v - 1)\Pi_{w=v+1}^{n} m_w])/\Pi_{v=2}^{n} m_v) + 1$.■

3 Applications

3.1 Spatial Matrix

Let A_k be a $l_k \times m_k$ spatial matrix, $k = 1, \cdots, n$. Suppose the indexes for non-zero elements of A_k are $\{(i_k(p), j_k(q))\}$, where $0 \le p \le l_k$, $0 \le q \le m_k$, $k = 1, \cdots, n$.

According to the **The Index Mapping Theorem**, the non-zero elements of $A = A_1 \otimes A_2 \otimes \cdots \otimes A_n$ are determined by the non-zero elements of A_k, $k = 1, \cdots, n$, and we can find all the non-zero elements of A using the non-zero elements of A_ks by (2) and (3). Non-zero elements of the Kronecker product are the elements whose indexes have nothing to do with indexes of zero elements in the factor matrices. The non-zero elements of A can be listed by substituting $\{(i_k(p), j_k(q))\}$s into (2) and (3), and can be calculated by multiplying corresponding elements in A_ks.■

3.1.1 Example

Let $A_{1(4\times5)}, A_{2(5\times6)}, A_{3(9\times8)}$ be three spatial matrices. Suppose the non-zero elements of A_1, A_2, A_3 are as follows:

$A_1(1, 1) = 1$, $A_1(1, 2) = 2$, $A_1(3, 4) = 3$;

$A_2(2,2) = 4,\ A_2(3,5) = 5;$

$A_3(7,7) = 6.$

Then there are only $3 \times 2 \times 1 = 6$ non-zero elements of $A = A_1 \otimes A_2 \otimes A_3$, they are:

$A(i_1, j_1) = A_1(1,1) \times A_2(2,2) \times A_3(7,7) = 1 \times 4 \times 6 = 24$, where $i_1 = (1-1) \times 5 \times 9 + (2-1) \times 9 + 7 = 16$, $j_1 = (1-1) \times 6 \times 8 + (2-1) \times 8 + 7 = 15$.

$A(i_2, j_2) = A_1(1,2) \times A_2(2,2) \times A_3(7,7) = 2 \times 4 \times 6 = 48$, where $i_2 = 16$, $j_2 = (2-1) \times 6 \times 8 + (2-1) \times 8 + 7 = 63$.

$A(i_3, j_3) = A_1(3,4) \times A_2(2,2) \times A_3(7,7) = 3 \times 4 \times 6 = 72$, where $i_3 = (3-1) \times 5 \times 9 + (2-1) \times 9 + 7 = 106$, $j_3 = (4-1) \times 6 \times 8 + (2-1) \times 8 + 7 = 159$.

$A(i_4, j_4) = A_1(1,1) \times A_2(3,5) \times A_3(7,7) = 1 \times 5 \times 6 = 30$, where $i_4 = (1-1) \times 5 \times 9 + (3-1) \times 9 + 7 = 25$, $j_4 = (1-1) \times 6 \times 8 + (5-1) \times 8 + 7 = 39$.

$A(i_5, j_5) = A_1(1,2) \times A_2(3,5) \times A_3(7,7) = 2 \times 5 \times 6 = 60$, where $i_5 = 25$, $j_5 = (2-1) \times 6 \times 8 + (5-1) \times 8 + 7 = 87$.

$A(i_6, j_6) = A_1(3,4) \times A_2(3,5) \times A_3(7,7) = 3 \times 5 \times 6 = 90$, where $i_6 = (3-1) \times 5 \times 9 + (3-1) \times 9 + 7 = 115$, $j_6 = (4-1) \times 6 \times 8 + (5-1) \times 8 + 7 = 183$.

3.2 Block Matrix

Let B_n be a $(l_1 l_2 \cdots l_n) \times (m_1 m_2 \cdots m_n)$ block matrix with $l_n \times m_n$ blocks, and suppose the (i_k, j_k)-th block is a $(l_1 l_2 \cdots l_{k-1}) \times (m_1 m_2 \cdots m_{k-1})$ block matrix with $l_{k-1} \times m_{k-1}$ blocks, $k = 1, \cdots n$. Denote $[(i_1, \cdots, i_n), (j_1, \cdots, j_n)]$ as the index for the (i_k, j_k)-th element in the (i_{k-1}, j_{k-1})-th block of B_k, $k = 1, \cdots, n$. Figure 3 gives an illustration of the block structure.

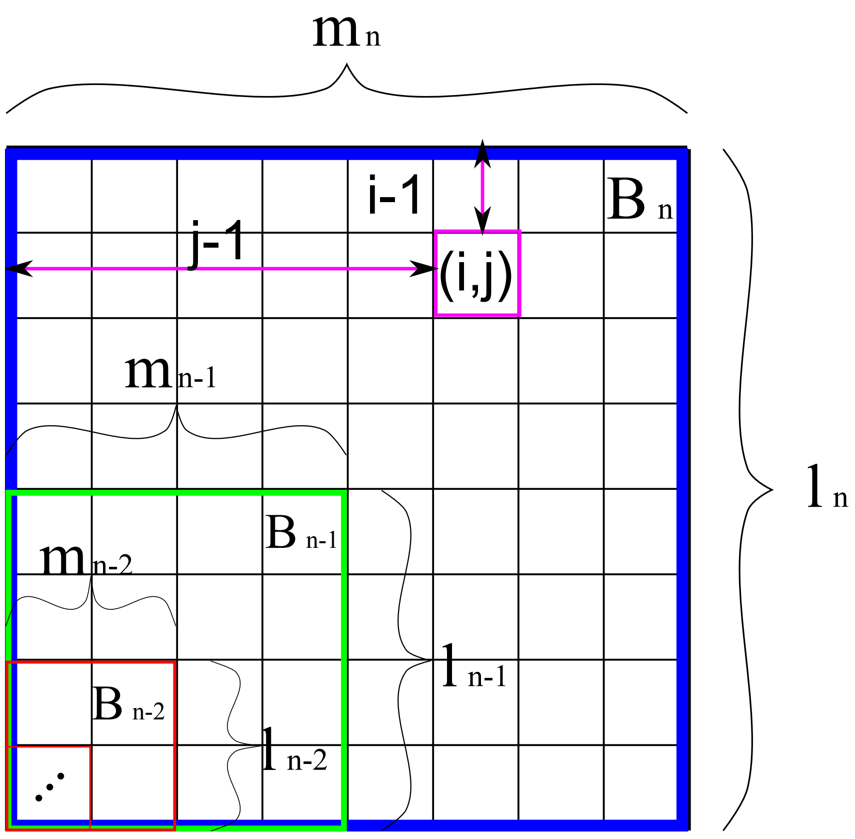

Figure 3: Block matrix structure

For an element b in B_n, let its index in B_n be $ind(n) = (i, j)$, and let its index in $Vec(B_n)$ be $t(n)$, where Vec is the vectorization operator. Suppose b's indexes in the blocks are $[(i_1, \cdots, i_n), (j_1, \cdots, j_n)]$.

Now we can find the relationship between $ind(n)$ and $t(n)$ easily using The Index Mapping Theorem.

It is easy to see that B_n can be decomposed as $B_n = B(1) \otimes B(2) \otimes \cdots \otimes B(n)$, where $B(k)$ is a $l_k \times m_k$ matrix, $k = 1, \cdots, n$.

According to The Index Mapping Theorem,

$i = (i_1-1)l_2 \cdots l_n + (i_2-1)l_3 \cdots l_n + \cdots + (i_{n-1}-1)l_n + i_n$, and $j = (j_1-1)m_2 \cdots m_n + (j_2-1)m_3 \cdots m_n + \cdots + (j_{n-1}-1)m_n + j_n$.

Let $m = m_1 \times m_2 \times \cdots \times m_n$, since $t(n) - 1$ is the number of elements before b in $Vec(B_n)$,

we have $t(n) = (j-1)m + i$.■

3.3 Octree

Example of 3.2 can also be extended to higher dimensions. Results in 3D, where the block matrix structure will be replaced by an octree (Figure 4), can be used to derive the algorithms for octree traversal and voxel searching.

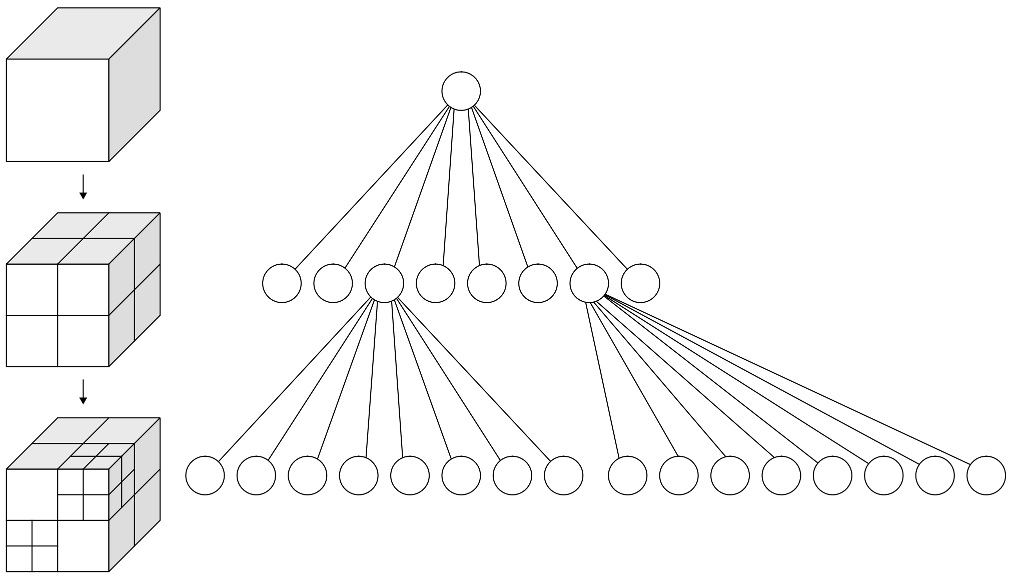

Figure 4: Octree (Courtesy of WhiteTimberwolf, licensed with Cc-by-sa-3.0-migrated, GFDL)

In 3D voxel rendering, the simplest way to represent voxel data is to use a 3D array - VoxelData[x][y][z], where the index (x, y, z) is the position of the voxel in 3D space and the value of the array member is the color for that voxel. However, for spatial voxel data, i.e., large part of the voxel data is empty space, 3D array is not efficient because it takes more computer memory than needed. A popular structure for representing spatial voxel data is the octree. Octrees can be used to partition a 3D space. In an octree, the 3D space is recursively divided into 8 cubes.

To encode a 3D array into an octree, we should give an one-to-one mapping of the 3D array indexes and the octree indexes. Note that for an octree of n levels, we have $2^n \times 2^n \times 2^n$ elements. For each dimension, X or Y or Z, we have 2^n leaves. The octree structure can be viewed as the Kronecker product of n vectors or levels $A_1, \cdots A_n$, where A_i is a 1×2 vector. So in each dimension, the index in the 3D

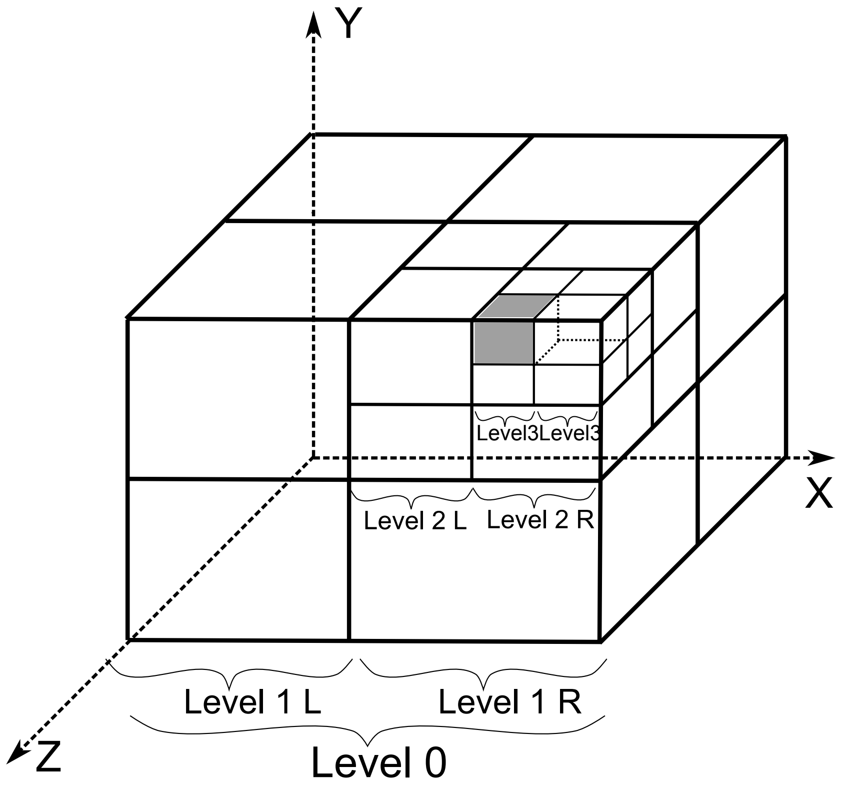

Figure 5: A 3-level octree

array can be mapped into the tree structure using The Inverse Index Mapping Theorem. Further more, since A_i is a 1×2 vector, if we use the number 0 to indicate that a voxel is in the left branch of the level, and number 1 to indicate that a voxel is in the right branch of the level, we immediately got a mapping from number of base 10 (the index in the 3D array, or equivalently, the position of the voxel in 3D space), to the number of base 2 (the indexes of the levels in the octree). This will be extremely useful because a fast algorithm for the mapping can be obtained using bitwise operation.

For example, in Figure 5, the shaded cube can be encoded as in the following table:

Dimension	3D Position	Level 0	Level 1	Level 2	Level 3
X	6	0	1	1	0
Y	7	0	1	1	1
Z	7	0	1	1	1

Note that the mapping is can be calculated using bitwise operators:

$6 = 0 << 3|1 << 2|1 << 1|0$,

$0_{6level3} = 6 >> 0\&2$,

$1_{6level2} = 6 >> 1\&2$,

$1_{6level1} = 6 >> 2\&2$,

$0_{6level0} = 6 >> 3\&2$;

$7 = 0 << 3|1 << 2|1 << 1|1$,

$1_{7level3} = 7 >> 0\&2$,

$1_{7level2} = 7 >> 1\&2$,

$1_{7level1} = 7 >> 2\&2$,

$1_{7level0} = 7 >> 3\&2$;

where $A << B$ denotes bitwise left shift A by B, $A >> B$ denotes bitwise right shift A by B, $|$ denotes bitwise OR operator, & denotes bitwise AND operator.

Another popular structure is quadtree. We can view a quadtree as an octree projected to the 2D XY plane - the 2D plane is recursively divided into 4 squares. A quadtree can also be seen as the special case of example 3.2, when the blocks are all 2×2. Similar to octree's usage in voxel representation and spatial 3D array indexing, quadtree can be used for image representation and spatial 2D array indexing. And the result is not limited to 2D and 3D. We can easily extend it to 4D, 5D, $\cdots$, because we treat each dimension independently.

4 Appendix

4.1 An implementation of the theorems in R language

```
#The source code is released under the MIT License
#<http://www.opensource.org/licenses/mit-license.php>
#*Copyright (c) <2011> <Bu Zhou>

#simple function to calculate product of the elements in a vector l
mul <- function(l)
{
len <- length(l);
returnVar <- 1;
for(i in 1:len)
{
returnVar <- returnVar*l[i];
}
return (returnVar);
}#end of mul

#The Index Mapping Function
#i: the row/column indexes of the factor matrices
#l: the length vector of the factor matrices
#return: the row/column index of kronecker product matrix
kronecker_index <- function(i,l)
{
returnVar <- 0;
len <- length(i);

for(j in 1:(len-1))
{
returnVar <- returnVar + (i[j]-1)*mul(l[(j+1):len]);
}

returnVar <- returnVar+i[len];
return (returnVar);
}#end of kronecker_index

#The Inverse Index Mapping Function
#c: the row/column index of kronecker product matrix
#l: the length vector of the factor matrices
#return: the row/column indexes of the factor matrices
kronecker_inv_index <- function(c,l)
{
n <- length(l);
i <- rep(0,n);
i[n] <- c%%l[n];
if(i[n]==0)
i[n] <- l[n];

if(n-1>1)
for(j in (n-1):1)
{

A <- 0;

if(j+1<n)
for(v in (j+1):(n-1))
{
A <- A+(i[v]-1)*mul(l[(v+1):n]);
}#end of if(j+1<n)for(v in (j+1):(n-1))

B <- mul(l[(j+1):n]);

if(j>1)
{
i[j] <- ((c-i[n]-A)/B)%%l[j]+1;
}
else
{
i[j] <- ((c-i[n]-A)/B)+1;
}

if(i[j]==0) i[j] <- l[j];

}#end of if(n-1>1)for(j in (n-1):1)
```

```
return (i);
}#end of kronecker_inv_index
```

4.2 R code for verifying the result in Example 3.1.1

```
A1<-matrix(0,4,5);
A2<-matrix(0,5,6);
A3<-matrix(0,9,8);
A1[1,1]<-1;
A1[1,2]<-2;
A1[3,4]<-3;
A2[2,2]<-4;
A2[3,5]<-5;
A3[7,7]<-6;
A<-kronecker(A1,A2);
A<-kronecker(A,A3);
> A[16,15]
[1] 24
> A[16,63]
[1] 48
> A[106,159]
[1] 72
> A[25,39]
[1] 30
> A[25,87]
[1] 60
> A[115,183]
[1] 90
library(Matrix);
> nnzero(A);#print the number of non-zero elements in A
[1] 6
```

References

[1] Shaosheng Zhou, Shizhou Fu, "Matrix representations for adjoint and anti-adjoint operators in multi-spin 1/2 systems," *Quantum Information Processin*, 2011.

[2] Bu Zhou,"An orthogonality-based estimation of moments for nonparametric linear mixed models (manuscript)," *Master Thesis*, 2011

[3] R. A. Horn and C. R. Johnson, "Topics in Matrix Analysis," *Cambridge University Press*, 2007.