# テンソル分解プログラミングの理解支援のための立体パズルの利用

山本 直樹
熊本高等専門学校人間情報システム工学科
naoki@kumamoto-nct.ac.jp

村上 純
熊本高等専門学校人間情報システム工学科
jun@kumamoto-nct.ac.jp

石田 明男
熊本高等専門学校共通教育科
ishida@kumamoto-nct.ac.jp

## 要旨

テンソル分解の一手法として高次特異値分解(HOSVD)があるが，これは行列の特異値分解(SVD)を高階テンソルの分解に拡張したものである．そのため，HOSVD は SVD と比べアルゴリズムが複雑となる．そこで，我々は，立体パズルを教材に用いた HOSVD アルゴリズムの理解支援の取り組みを行っている．本論文では，立体パズルとしてルービックキューブおよびインスタント・インサニティの 2 つを取り上げ，これらを高階テンソルで表現し，HOSVD アルゴリズムの *n*-モード行列展開を利用して展開図にする原理について述べる．さらに，その展開図のテンソル分解やそのプログラミング教育への応用として，本校学生によるルービックキューブの展開図作成の事例と，主に小中学生を対象として，インスタント・インサニティの展開図の理解度について調査した事例についても言及する．

## 1. はじめに

ビッグデータのような多次元データを扱いやすくするための低次元化手法にテンソル分解がある．テンソル分解におけるテンソルとは多次元配列で表現される高階テンソルのことを指し，多次元データを低次元の積や和に分解する．テンソル分解としては，Tucker 分解と CP(canonical or parallel factor)分解などがあり，これらの応用として，Fuzzy モデリング，信号処理，画像処理，画像分類，生体信号分析，テキストマイニング，ソーシャルネットワークおよび Web ハイパーリンクの分析等に用いられている[1].

我々は，これまでテンソル分解の数値計算法に関する研究[2, 3]や，それを用いた医療データ分析に関する研究[4, 5]を行ってきた．Tucker 分解を計算する手法としてよく利用されるものに高次特異値分解(HOSVD: higher-order SVD) [6]があるが，これは行列の特異値分解(SVD: singular value decomposition)を 3 階以上の高階テンソルの分解に拡張したものである．したがって，SVD よりもアルゴリズムが複雑となるため，このアルゴリズムの理解支援にも取り組んでいる．理解支援に関する最初の取り組みは，CG を利用したアルゴリズムの可視化[7]であった．次に，立体パズルに着目して，この求解に HOSVD のアルゴリズムを利用することで，アルゴリズムの理解に繋がると考え，研究を行っている[8, 9, 10, 11].

現在は，HOSVD を利用可能な立体パズルの収集を行っているところである．今回，本論文では，よく知られた立体パズルであるルービックキューブ[12]とインスタント・インサニティ[13]を取り上げ，このパズルの展開表現と，それをテンソル分解とそのプログラミングの教育に応用した事例について述べる．

## 2. 高階テンソル分解アルゴリズムとそのプログラミング教育への応用

### 2.1. 高階テンソルの概要と定義

高階テンソルとは，1章で述べたように多次元配列のことを指し，1 階テンソルはベクトル，2 階テンソルは行列，3 階テンソルは 3 次元配列にそれぞれ対応する．本論文では，立体パズルを 3 階テンソルおよび 4 階テンソルとして表現する．ここで，図 1 にサイズ$I \times J \times K$の 3 階テンソル$\boldsymbol{\mathcal{A}}$のイメージを示す．いま，$\boldsymbol{\mathcal{A}}$の$(i, j, k)$要素を$a_{ijk}$とする

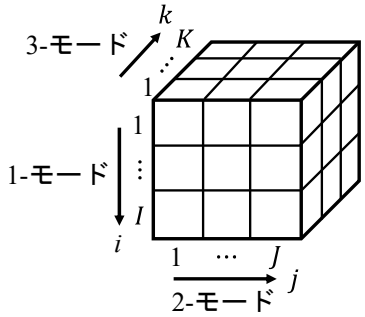



図 1. 3 階テンソルのイメージ

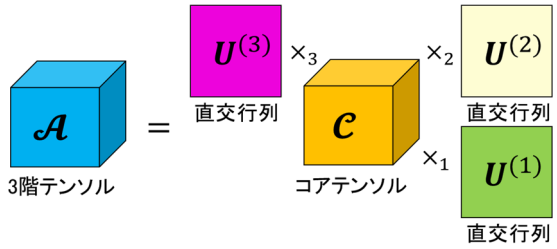



図 2.　3 階テンソルの HOSVD

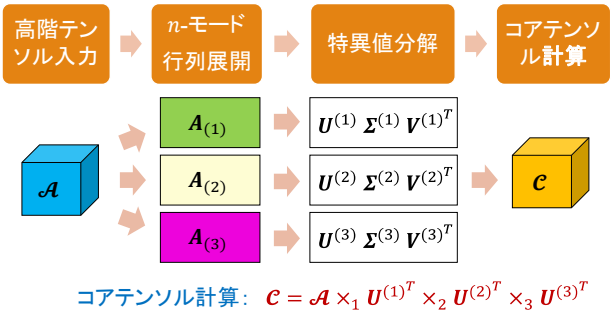



図 3.　3 階テンソルの HOSVD の流れ

と，$\mathcal{A}$は次式で表される．

$$\mathcal{A} = (a_{ijk}),$$
$$(i = 1,2,\dots,I; j = 1,2,\dots,J; k = 1,2,\dots,K) \quad (1)$$

また，図 1 におけるモードとは，図の矢印で示すような高階テンソルの方向を表す．本論文では，縦方向を 1-モード，横方向を 2-モード，奥方向を 3-モードとし，テンソル要素の添字$i, j, k$とそれぞれ対応させている．

さらに，4 階テンソルは，図 1 に示す 3 階テンソルがベクトル的に配置されたものと考える．すなわち，サイズ$I \times J \times K \times L$の 4 階テンソルを$\mathcal{B}$とすると，$\mathcal{B}$は$\mathcal{A}$と同じサイズの 3 階テンソル$\mathcal{A}_l, (l = 1,2,\dots,L)$を$L$個並べて$\mathcal{B} = (\mathcal{A}_1, \mathcal{A}_2, \dots, \mathcal{A}_L)$と構成されるものとする．

いま，$\mathcal{B}$の$(i, j, k, l)$要素を$b_{ijkl}$とすると，$\mathcal{B} = (b_{ijkl})$,
$(i = 1,2,\dots,I; j = 1,2,\dots,J; k = 1,2,\dots,K; l = 1,2,\dots,L)$
と表され，4 階テンソル要素の添字$l$が 4-モードに対応する．

## 2.2. テンソル分解の概要とアルゴリズム

本論文では，立体パズルの展開表現にテンソル分解の一手法である高次特異値分解（HOSVD）のアルゴリズムを利用する．以下に HOSVD の概要とアルゴリズムについて述べる．

図 1 に示す 3 階テンソル$\mathcal{A}$の HOSVD は次式で与えられる．

$$\mathcal{A} = \mathcal{C} \times_1 U^{(1)} \times_2 U^{(2)} \times_3 U^{(3)} \quad (2)$$

ここで，$U^{(n)}, (n = 1,2,3)$は正規直交行列を表し，SVD の左および右特異行列に対応し，$\mathcal{C}$はコアテンソルと呼ばれ，SVD の特異値行列に対応する．$\times_n, (n = 1,2,3)$は$n$-モード積と呼ばれ，高階テンソルと行列の積の演算を表す．例えば，$\times_3$は 3-モード積を表す．

以上より，元の 3 階テンソル$\mathcal{A}$は，HOSVD により 3 つの直交行列と 1 つのコアテンソルの$n$-モード積に分解されることが分かる．この分解のイメージを図 2 に示す．

次に，式(2)を計算する HOSVD アルゴリズムを示す．

**（アルゴリズム 1：3 階テンソルの HOSVD）**

入力：3 階テンソル$\mathcal{A}$.

出力：正規直交行列$U^{(n)}, (n = 1,2,3)$とコアテンソル$\mathcal{C}$.

ステップ 1：$\mathcal{A}$に$n$-モード行列展開（詳細は後述する）を適用し，展開された行列$A_{(n)}, (n = 1,2,3)$を求める．

ステップ 2：ステップ 1 で得られた行列$A_{(n)}, (n = 1,2,3)$の SVD をそれぞれ次式で計算する．

$$A_{(n)} = U^{(n)}\,\Sigma^{(n)}\,V^{(n)T}, (n = 1,2,3) \quad (3)$$

ただし，$U^{(n)}$および$V^{(n)}$は，各列に特異ベクトルを持つ左および右特異行列を表し，$\Sigma^{(n)}$は対角成分に特異値を持つ特異値行列であり，記号 $T$ は，行列の転置を表す．

ステップ 3：元のテンソル$\mathcal{A}$とステップ 2 で得られた左特異行列$U^{(n)}, (n = 1,2,3)$から，次式によりコアテンソル$\mathcal{C}$を計算する．

$$\mathcal{C} = \mathcal{A} \times_1 U^{(1)T} \times_2 U^{(2)T} \times_3 U^{(3)T} \quad (4)$$

ステップ 4：$U^{(n)}, (n = 1,2,3)$および$\mathcal{C}$を返す．

**（アルゴリズム終わり）**

## 2.3. テンソル分解プログラミング教育への応用

図 3 は，前節で述べたアルゴリズム 1 の流れを示しており，この図の流れに沿ってテンソル分解プログラミングを次のように教えることにする．

（1）まず，このアルゴリズムの全体像を学習者に掴ませるために，既に開発された CG によるアルゴリズムの可視化アプリ[7]を利用する．

（2）次に，図 3 の高階テンソル入力と行列展開の部分は，本論文で提案する立体パズルを利用して理解支援を行い，3.3 節および 3.5 節のスクリプト例に示すような行列展開を求めるプログラムを作成させる課題を与える．

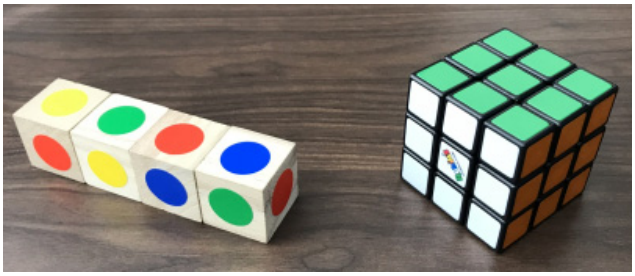
（3）図 3 の特異値分解の部分は，特に左特異行列$U^{(n)}, (n = 1,2,3)$は，高階テンソルの各モードの特徴を表すため，その理解が重要と考えられるので，（2）で作成された行列展開のデータを実際にプログラミング言語で特異値分解させ，$U^{(n)}$の各列ベクトルの変化を可視化させ

ることにより理解を深めさせる．

（4）最後に，コアテンソルの計算部分は，$n$-モード積の理解が重要であるが，この積の計算は高階テンソルと行列の積として直接計算するもので，理解が難しいと考えられる．そこで，高階テンソルの行列展開と左特異行列 $\boldsymbol{U}^{(n)}$ の行列積から得られた行列を再びテンソルとして畳み込むことと等価であることから，この計算法を用いることにする．そして，本論文の立体パズルを利用して，テンソルの行列展開と畳み込みの理解支援を行い，実際にコアテンソルを計算するプログラムを実装させる．

以上が立体パズルを用いたテンソル分解プログラミング教育の流れであるが，高階テンソルの取り扱いと行列展開および畳み込みがテンソル分解アルゴリズム全体に関連する重要なポイントであるといえる．この教育を実施する対象者としては，プログラミングおよび線形代数の知識があることが前提であり，高等専門学校の学生でいえば，4 年生以上がその対象と考えられる．この取り組みの実際の適用と教育効果の調査については，今後の課題であるが，対象学生を2つのグループに分け，立体パズルを利用するグループと，利用しないグループとの間で理解度の比較を行うことを考えている．

本論文では，実際に開始する前段階として，2種類の立体パズルを利用して2つの試みを行った．1つは，テンソル分解の知識を持つ高専専攻科生に対し，立体パズルの行列展開を CG で表現させるプログラミング課題を実施したもので，もう1つは，オープンキャンパスに参加した小中学生・高専生・一般に対し，立体パズルの行列展開表現を応用したマップ表示アプリと実物のパズルを利用して，マップ表示の理解調査を行ったものである．



（a）インスタント・インサニティ　　（b）ルービックキューブ

図 4.　立体パズル

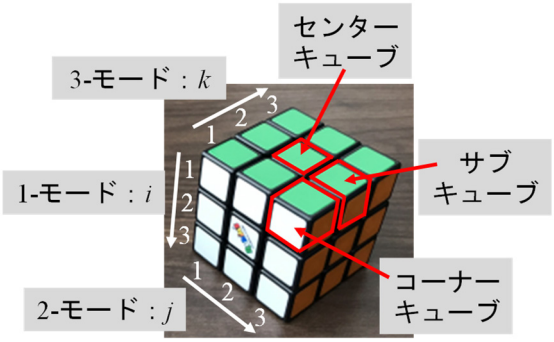



図 5.　ルービックキューブの 3 階テンソル表現

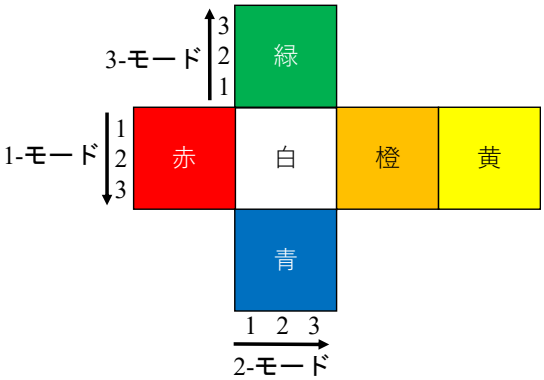



図 6.　ルービックキューブの展開図と配色

## 3. 立体パズルとその展開表現

### 3.1. 立体パズル

本論文で取り扱う立体パズルは，図 4 に示すようなインスタント・インサニティおよびルービックキューブである．インスタント・インサニティ（カラーダイスパズルとも呼ばれる[13]）は市販されてもいるが，図 4(a)は 4 個の木製の立方体にそれぞれ色シールを貼り，自作したものである．図 4(b)に示すルービックキューブは市販されているルービックキューブ ver2.0 で，このキューブの配色が世界基準となっている[14]．

### 3.2. ルービックキューブの高階テンソル表現

ルービックキューブは，サイズ3 × 3 × 3の 3 階テンソルにより表現できる．ルービックキューブにおける各モードの関係を図 5 に示す．

このパズルには，図 5 に示すように 1 色からなるセンターキューブ 6 個，2 色からなるサブキューブ 12 個，3 色からなるコーナーキューブ 8 個がある．さらに，本論文ではパズル内部に色を持たないキューブが 1 個あるとして，これをインナーキューブと呼ぶ．パズルは，このような 27 個のキューブから構成されるものとする．

図 6 に，白色の面を中心として描いたパズルの配色を展開図にして示す．ただし，図 6 で示されていないインナーキューブは無色とする．さらに，表 1 に，図 5 における

表 1. 各キューブの配色情報

| キューブ番号 | キューブ種類 | キューブ配色 | 図 5 での要素番号 |
|---|---|---|---|
| 1 | コーナー | 緑, 赤, 白 | (1,1,1) |
| 2 | サブ | 緑, 赤 | (1,1,2) |
| 3 | コーナー | 緑, 赤, 黄 | (1,1,3) |
| 4 | サブ | 緑, 白 | (1,2,1) |
| 5 | センター | 緑 | (1,2,2) |
| 6 | サブ | 緑, 黄 | (1,2,3) |
| 7 | コーナー | 緑, 橙, 白 | (1,3,1) |
| 8 | サブ | 緑, 橙 | (1,3,2) |
| 9 | コーナー | 緑, 橙, 黄 | (1,3,3) |
| 10 | サブ | 赤, 白 | (2,1,1) |
| 11 | センター | 赤 | (2,1,2) |
| 12 | サブ | 赤, 黄 | (2,1,3) |
| 13 | センター | 白 | (2,2,1) |
| 14 | インナー | なし | (2,2,2) |
| 15 | センター | 黄 | (2,2,3) |
| 16 | サブ | 橙, 白 | (2,3,1) |
| 17 | センター | 橙 | (2,3,2) |
| 18 | サブ | 橙, 黄 | (2,3,3) |
| 19 | コーナー | 青, 赤, 白 | (3,1,1) |
| 20 | サブ | 青, 赤 | (3,1,2) |
| 21 | コーナー | 青, 赤, 黄 | (3,1,3) |
| 22 | サブ | 青, 白 | (3,2,1) |
| 23 | センター | 青 | (3,2,2) |
| 24 | サブ | 青, 黄 | (3,2,3) |
| 25 | コーナー | 青, 橙, 白 | (3,3,1) |
| 26 | サブ | 青, 橙 | (3,3,2) |
| 27 | コーナー | 青, 橙、黄 | (3,3,3) |

各キューブの配置と配色の詳細な情報を示す.

このパズルを 3 階テンソル$\boldsymbol{\mathcal{A}}$で表し, $\boldsymbol{\mathcal{A}}$の$(i,j,k)$要素を$a_{ijk}$とすると, 次式のように表現できる.

$$\boldsymbol{\mathcal{A}} = (a_{ijk}),\ (i,j,k = 1,2,3) \quad (5)$$

ただし, $a_{ijk},(i,j,k = 1,2,3)$には, それぞれ対応する表 1 のキューブ番号を与える. 例えば, $a_{111}$では 1, $a_{123}$では 6 などとなる. パズルの各キューブの配色は, キューブ番号から, 表 1 を参照して得ることができる.

### 3.3. ルービックキューブの行列展開表現

次に, 式(5)の高階テンソルで表現されたパズルを展開図として表現する方法について説明する. 本論文では, 2.2 節で述べたアルゴリズム 1 のステップ 1 における $n$-モード行列展開を利用する. 行列展開とは, 図 3 に示すような元の高階テンソルを行列に変換する操作のことである.

この行列展開の仕方については, 諸論文で様々な方法があるが, ここでは文献[6, 15]の方法を用いた. 行列展開は $N$ 階テンソルまでに適用できるが, 次に示すのは 3 階テンソルの行列展開のアルゴリズムである.

**(アルゴリズム 2: 3 階テンソルの $n$-モード行列展開)**

入力: サイズ$I \times J \times K$の 3 階テンソル$\boldsymbol{\mathcal{A}}$.

出力: $n$-モード行列展開$\boldsymbol{A}_{(n)},(n = 1,2,3)$.

以下のステップ 1 から 3 の操作は, $i = 1,2,\dots,I;\ j = 1,2,\dots,J;\ k = 1,2,\dots,K$の全ての場合について行う.

ステップ 1 (1-モード行列展開): 3 階テンソル$\boldsymbol{\mathcal{A}}$の要素$a_{ijk}$を行列$\boldsymbol{A}_{(1)}$の$i$行$(j-1)K+k$列の要素に移す.

ステップ 2 (2-モード行列展開): $\boldsymbol{\mathcal{A}}$の要素$a_{ijk}$を行列$\boldsymbol{A}_{(2)}$の$j$行$(k-1)I+i$列の要素に移す.

ステップ 3 (3-モード行列展開): $\boldsymbol{\mathcal{A}}$の要素$a_{ijk}$を行列$\boldsymbol{A}_{(3)}$の$k$行$(i-1)J+j$列の要素に移す.

ステップ 4:行列$\boldsymbol{A}_{(n)},(n = 1,2,3)$を返す.

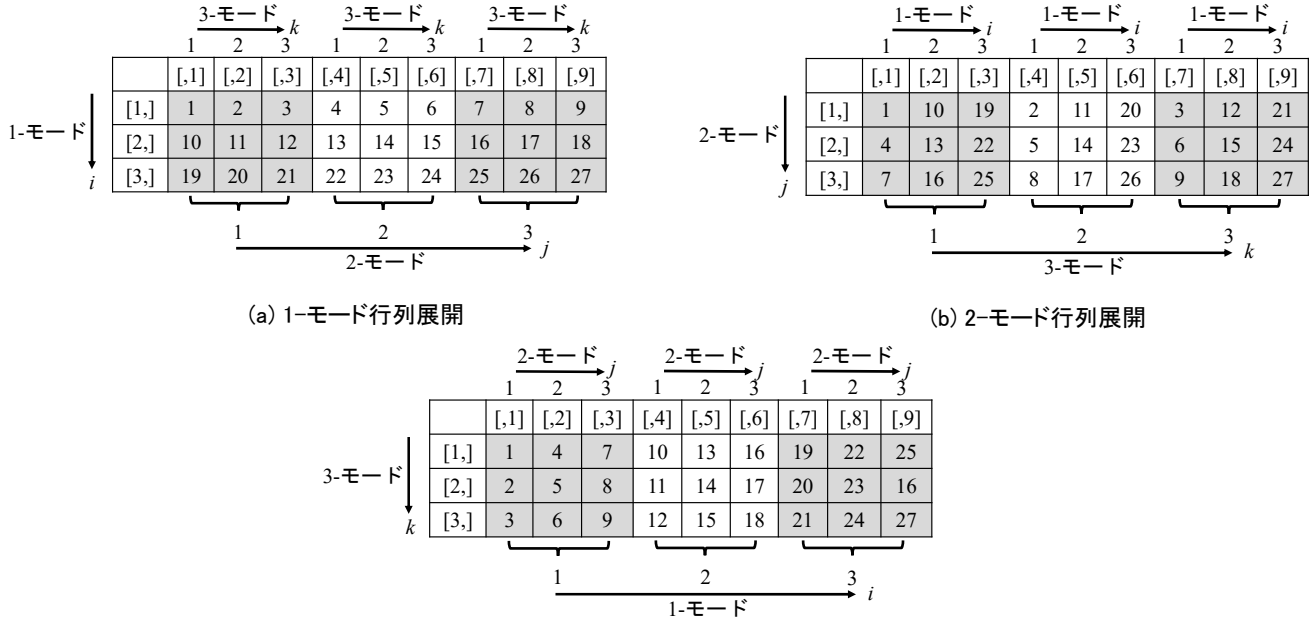
**(アルゴリズム終わり)**

このアルゴリズムを適用すると, 元のテンソル$\boldsymbol{\mathcal{A}}$の各モードが行列$\boldsymbol{A}_{(n)},(n = 1,2,3)$の各行に移される. 例えば, 1-モード行列展開$\boldsymbol{A}_{(1)}$の行には$\boldsymbol{\mathcal{A}}$の添字 $i$ が対応しており, $\boldsymbol{\mathcal{A}}$の 1-モードが移される. 3 階テンソルの場合のモード数は 3 であるので, 3 通りの行列展開が得られ, $N$ 階テンソルの場合は $N$ 通りとなる.

ルービックキューブの行列展開表現の原理をより具体的に示すために, 統計解析用のプログラミング言語 R[16]を利用し, テンソル計算用のパッケージである rTensor[17]を導入して, このパズルの展開図を作成する R スクリプトを次に示す. R スクリプトは, 実際に R 言語を用いた教育を行う際の参考のために載せたもので, 以降も同様である.

**(スクリプト1: ルービックキューブの展開図計算)**

```
# パッケージ rTensor の利用
 library( rTensor )
# テンソル A のサイズ指定
 I <- 3; J <- 3; K <- 3
# テンソル A の初期化
 A <- array( 0, c(I,J,K) ); A <- as.tensor( A )
# テンソル A の作成
 A[1, , ] <- 1:9; A[1, , ] <- t( A[1, , ]@data )
```




(a) 1−モード行列展開

(b) 2−モード行列展開

(c) 3−モード行列展開

図 7. $n$−モード行列展開の計算結果

```
  A[2, , ] <− 10:18; A[2, , ] <− t( A[2, , ]@data )
  A[3, , ] <− 19:27; A[3, , ] <− t( A[3, , ]@data )
# テンソル A の n-モード行列展開計算
  A_1mode <− unfold( A, 1, c(3,2) )
  A_2mode <− unfold( A, 2, c(1,3) )
  A_3mode <− unfold( A, 3, c(2,1) )
# n-モード行列展開の結果表示
  A_1mode@data     # 1-モード行列展開
  A_2mode@data     # 2-モード行列展開
  A_3mode@data     # 3-モード行列展開
```

（スクリプト終わり）

スクリプトの中の記号<−は代入操作を表している. 図 7(a)から(c)は, スクリプト 1 で計算されたパズルの $n$-モード行列展開の結果である.

スクリプト 1 において, 行列展開の計算にはパッケージ rTensor の関数 unfold を利用した. スクリプトでは, 関数 unfold の第 1 引数に元のテンソルを, 第 2 引数に行列展開の行に移すモード番号を, 第 3 引数には行列展開の列に移すモード番号を与えている. 例えば, 1-モード行列展開は unfold( A, 1, c(3,2) )で求められるが, 行列展開の行に移すモードとして 1-モードを指定し, 列に移すモードとして, 2-モードと 3-モードを指定している. 第 3 引数でベクトル c(3,2)としているのは, 図 7(a)から分かるように, 3-モードの添字は 2-モードの添字よりも先に変化させる必要があるためである.

図 7(a)の 1-モード行列展開は, 図 5 のテンソルの 2-モードに沿って輪切りして得られた行列を横に並べられたものと考えることができる. その他のモードの行列展開についても同様である.

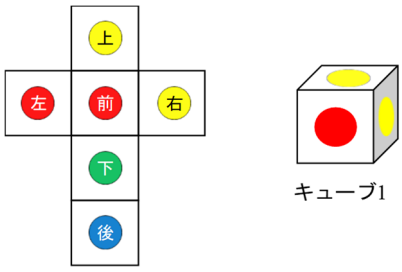
図 7 ではテンソルの要素値として, 表 1 のキューブ番号が格納されている. したがって, 表 1 を参照して各配置に対応するキューブ配色で塗れば, グラフィカルなパズルの展開図を作成することができる.

### 3.4. インスタント・インサニティの高階テンソル表現

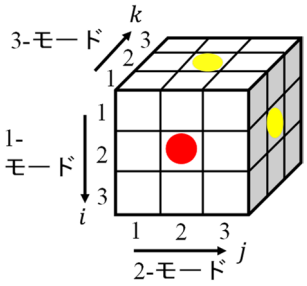
このパズルは, 図 4(a)に示すようにサイズ$1 \times 1 \times 1$の 4 つのキューブからなる. 本論文では, これらのキューブをそれぞれ$3 \times 3 \times 3$の 3 階テンソルにより表現する.

図 8(b)は図 4(a)の一番左のキューブを取り出したもので, これをキューブ 1 と呼ぶことにする. キューブ 1 の各面の配色は図 8(a)のようになっている. ここで, 図 8(b)に示される元のキューブのサイズを$3 \times 3 \times 3$に拡大して, 図 8(c)のような 3 階テンソルで表す. すなわち, 図 8(a)の配色は, この$3 \times 3 \times 3$のテンソルの各面の中央要素に塗られるものとする.

図 4(a)のようにキューブは横に 4 個並べられるため, 3 階テンソルを 4 個並べた$3 \times 3 \times 3 \times 4$ の 4 階テンソルとして, このパズルは表現される. 図 9 に, 図 4(a)の各キューブの配置を 4 階テンソル表現したイメージを示す. また, 図 10 には, 図 4(a)の左から 2〜4 番目のキューブ(キュ




(a) 元のキューブ 1 の配色　　(b) 元のキューブ 1




(c) 拡大されたキューブ 1

図 8. キューブ 1 の 3 階テンソル表現

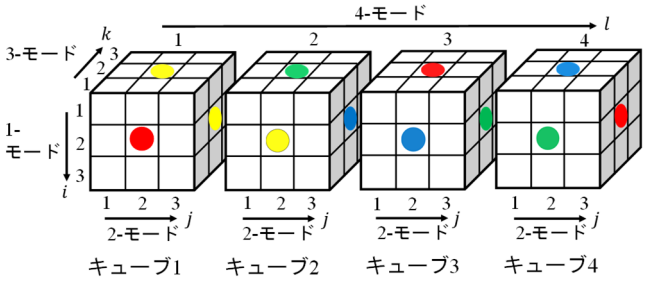



図 9. インスタント・インサニティの 4 階テンソル表現

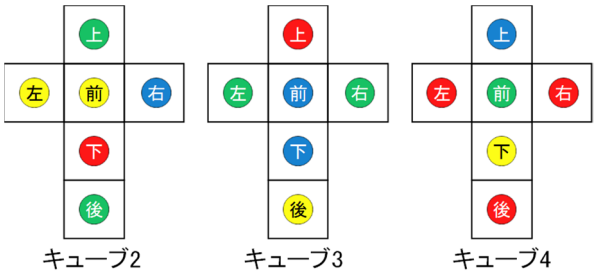



図 10. 元のキューブ 2 から 4 の配色

表 2. 図 9 での各キューブの配色情報

| キューブ番号 | テンソル要素番号 | キューブ配色 | キューブ番号 | テンソル要素番号 | キューブ配色 |
|---|---|---|---|---|---|
| 1 | (1,2,2,1) | 黄 | 3 | (1,2,2,3) | 赤 |
| | (2,1,2,1) | 赤 | | (2,1,2,3) | 緑 |
| | (2,2,1,1) | 赤 | | (2,2,1,3) | 青 |
| | (2,2,3,1) | 青 | | (2,2,3,3) | 黄 |
| | (2,3,2,1) | 黄 | | (2,3,2,3) | 緑 |
| | (3,2,2,1) | 緑 | | (3,2,2,3) | 青 |
| 2 | (1,2,2,2) | 緑 | 4 | (1,2,2,4) | 青 |
| | (2,1,2,2) | 黄 | | (2,1,2,4) | 赤 |
| | (2,2,1,2) | 黄 | | (2,2,1,4) | 緑 |
| | (2,2,3,2) | 緑 | | (2,2,3,4) | 赤 |
| | (2,3,2,2) | 青 | | (2,3,2,4) | 赤 |
| | (3,2,2,2) | 赤 | | (3,2,2,4) | 黄 |

ーブ 2～4)の配色を示した. 表 2 には, この 4 階テンソルの配色情報をまとめている.

いま, この 4 階テンソルを$\boldsymbol{\mathcal{B}}$, $\boldsymbol{\mathcal{B}}$の$(i,j,k,l)$要素を$b_{ijkl}$とすると, 次式のように表現できる.

$$\boldsymbol{\mathcal{B}} = \left(b_{ijkl}\right),\ (i,j,k = 1,2,3; l = 1,2,3,4) \qquad (6)$$

ただし, 表 2 に記載された各テンソル要素の配色が, 赤のとき$b_{ijkl} = 1$, 黄は 2, 緑は 3, 青は 4 を要素値として与え, 記載のないテンソル要素はすべて 0 とする.

### 3.5. インスタント・インサニティの行列展開表現

このパズルの展開図を作成するアルゴリズムを次に示す. ここでは, 1-モード行列展開のみを利用したが, 他のモードでも展開図を得ることができる.

**(アルゴリズム 3:インスタント・インサニティの展開図作成)**

入力: 表 2 の配色を持つサイズ$3 \times 3 \times 3 \times 4$の 4 階テンソル$\boldsymbol{\mathcal{B}}$.

出力: 1-モード行列展開$\boldsymbol{B}_{(1)}$.

ステップ 1: $\boldsymbol{\mathcal{B}}$から 3 階テンソル$\boldsymbol{\mathcal{B}}_l = (b_{***l}), (l = 1,2,3,4)$を取り出す. ただし, $b_{***l}$は, $l$だけを固定した$i, j, k = 1,2,3$の 3 階テンソルである.

ステップ 2: $\boldsymbol{\mathcal{B}}_l$, $(l = 1,2,3,4)$に, それぞれアルゴリズム 2 のステップ 1 を適用して, $\boldsymbol{B}_{l\,(1)}, (l = 1,2,3,4)$を求める.

ステップ 3: $\boldsymbol{B}_{l\,(1)}, (l = 1,2,3,4)$を横に並べて, $\boldsymbol{B}_{(1)} = \left(\boldsymbol{B}_{1\,(1)} | \boldsymbol{B}_{2\,(1)} | \boldsymbol{B}_{3\,(1)} | \boldsymbol{B}_{4\,(1)}\right)$とする.

ステップ 4: $\boldsymbol{B}_{(1)}$を返す.

**(アルゴリズム終わり)**

ここで, この立体パズルの行列展開表現の原理を具体的に示すために, 式(6)の 4 階テンソルに, アルゴリズム 3 を適用して展開図を得る R のスクリプトを次に示す.

**(スクリプト 2:インスタント・インサニティの展開図計算)**

```
library(rTensor)     # rTensor を利用する
# テンソル B の初期化
 B <- array( 0, c(3,3,3,4) ); B <- as.tensor( B )
# テンソル B の作成
# 色の設定( 赤:1, 黄:2, 緑:3, 青:4 )
 r <- 1; y <- 2; g <- 3; b <- 4
# 各キューブの前・後／上・下／左・右の面の
# 中央に配色
 B[2,2,c(1,3), ] <- rbind( c(r,y,b,g), c(b,g,y,r) )     # 前・後
 B[c(1,3),2,2, ] <- rbind( c(y,g,r,b), c(g,r,b,y) )     # 上・下
 B[2,c(1,3),2, ] <- rbind( c(r,y,g,r), c(y,b,g,r) )     # 左・右
# テンソル B の 1-モード行列展開計算
 B_1mode <- unfold( B, 1, c(3,2,4) )
 B_1mode@data     # 結果表示
```

**(スクリプト終わり)**

このスクリプトを実行して得られた図 9 のキューブ 1 部分の展開図を図 11 に示す. この図は, 図 8(c)の 2-モードに沿って輪切りにした行列を横に並べたものである. したがって, 一番左の$3 \times 3$の行列にはパズルの左面の赤色が, 左から 2 番目の$3 \times 3$の行列にはパズルの前, 後, 上, 下の色, すなわち赤, 青, 黄, 緑の色が, 一番右の$3 \times 3$の行列にはパズルの右面の黄色が現れていることが分かる. キューブ 2 から 4 部分についても, キューブ 1 の場合と同様にして展開図を得ることができる.

以上より, アルゴリズム3を実行すれば, 各キューブの 1-モード行列展開による展開図が得られるので, 実物のキューブを回転させて正しい並びを揃える際に, この展開図も参照させながら行わせることで, テンソルデータの行列展開や多次元データ処理の理解支援に繋げられるのではないかと考えている.

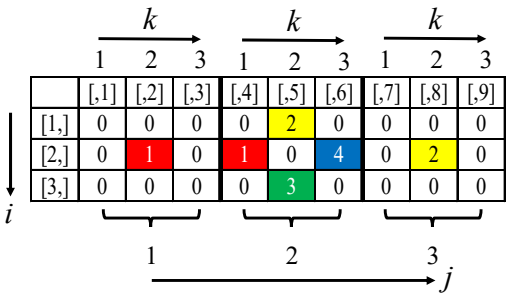



図 11. キューブ 1 部分($l = 1$)の 1-モード行列展開

## 4. 立体パズルの展開表現の教育への応用

### 4.1. ルービックキューブの行列展開表現の応用

3.3 節でルービックキューブの行列展開表現のアルゴリズムについて述べた. ここでは, 本校(熊本高等専門学校)においてこの展開表現を学生の教育に応用した事例について記す.

対象とする学生は平成 29 年度の専攻科 2 年生で, 実施期間は 2〜3 週間程であった. この学生は, 本科 5 年次の卒業研究と専攻科 1 年次および 2 年次のシステム工学特別研究を通して, HOSVD および Nonnegative Tucker Decomposition (NTD)などのテンソル分解法を利用した医療データ分析に関する研究に携わっていた. また, CG に関するプログラミング言語は, 専攻科 1 年次の実習科目で Unity や C#を使用してアプリケーションを開発した経験を有していた.

この学生へ与えたアプリ作成の課題は次のとおりである.「ルービックキューブを 3 階テンソルとして取り扱い, $n$-モード行列展開を求め, その配色を CG で表現すること」と,「ルービックキューブの回転に伴って $n$-モード行列展開の配置と配色も変化させること」というものであった. 学生には図 4(b)に示すような実物のルービックキューブを貸与した.

この課題の Unity と C#によるアプリの実行例を図 12 に示す. パズルの行列展開と配色の確認のために, 図 12 左上に示したルービックキューブは, 図 5 と同じ配置で, モードの取り方も同じにしている.

これより, 図 7 の計算結果および表 1 の配色情報と,

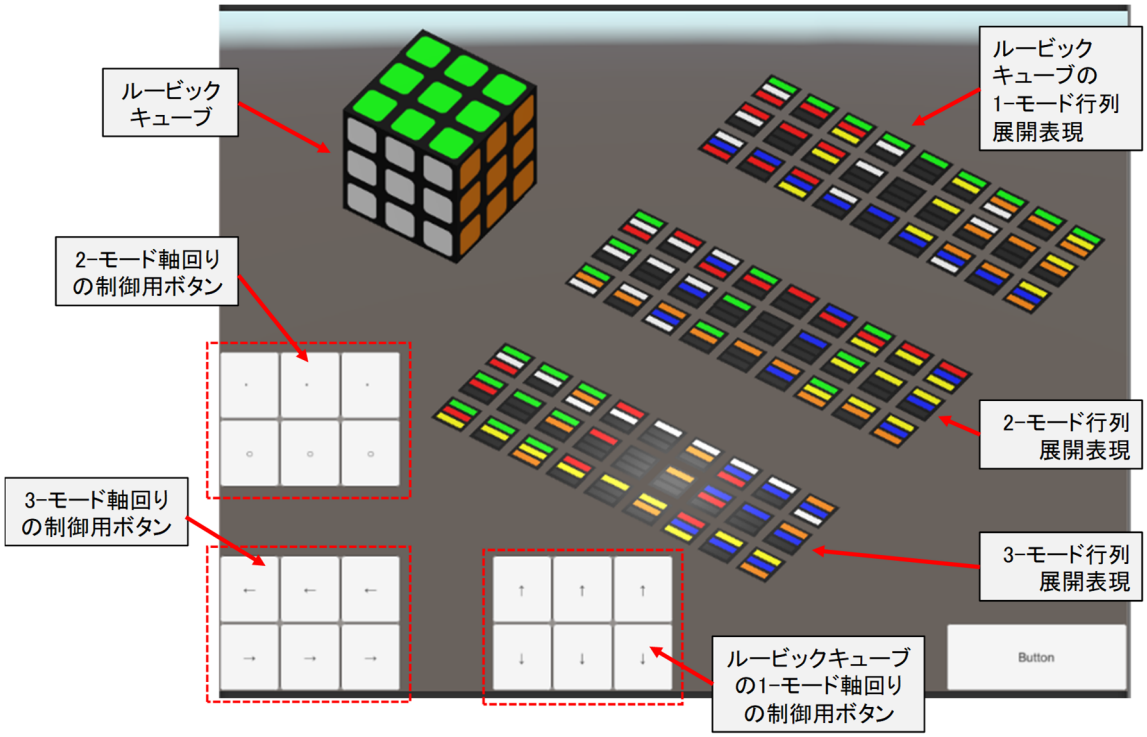



図 12. ルービックキューブのアプリの実行例

図 12 の各行列展開に塗られた色とを比較すると，一致していることが分かる．したがって，このアプリは行列展開の原理に基づいたパズルの展開図を正しく作成しているといえる．

さらに，このアプリでは，図 12 左下側にある制御用ボタンを用いて，各モード軸回りにルービックキューブを回転させることができる．キューブの回転に伴って，各行列展開の配色も変化するようになっている．

学生が作成したこのアプリの実行結果から，本学生は Unity および C#の CG プログラミングのスキルと，研究を通して学んだ高階テンソルの取り扱いや行列展開についてよく理解し，それをプログラミングして応用できる力を身につけたことが分かる．

### 4.2. インスタント・インサニティの行列展開表現の応用

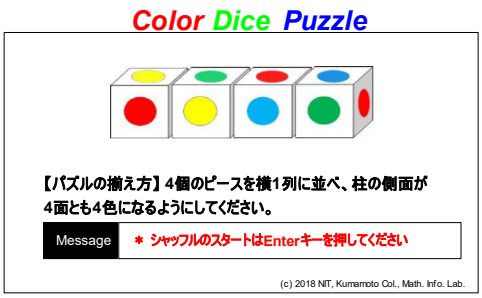
3.5 節で記したこのパズルの行列展開のアルゴリズムに基づいて R を用いて展開図を表示するアプリを作成した．このアプリ（図 13）と実物のパズル（図 4(a)）を利用して，本校のオープンキャンパス参加者に対し，展開図が理解できるかどうかを調査した事例[10]について述べる．

実施時期は平成 30 年 8 月上旬で，我々の研究室展示を見学した小中学生，高専本科生，保護者，教員の計 43 名を対象とした．このときの調査の様子は，次のとおりである．

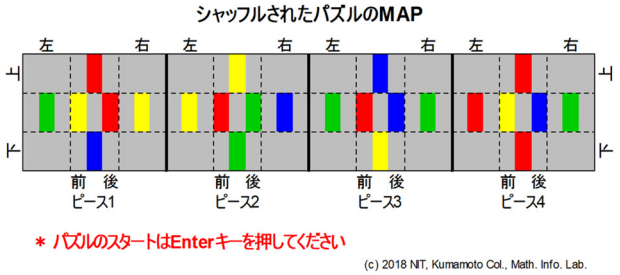
見学者が展示ブースを訪れると，説明者がこのパズルのアプリを起動して，図 13(a)のタイトル画面を表示させ，パズルの揃え方を説明する．その際，実物のパズルも併用する．

次に，アプリ上のパズルをシャッフルさせ，図 13(b)のような展開図を表示させる．このとき，展開図の見方も説明するが，キューブ 1 部分（アプリではピース 1 と表示）について，実物のキューブ 1 を用いて，具体的に図と合わせる．そして，残りのキューブ 2 から 4（アプリではピース 2 から 4 と表示）は，見学者が展開図を確認しながら実物のパズルと同じ色の配置になるように合わせる．図 13(c)は中学生 2 名が実施している様子である．

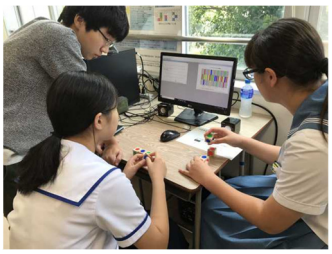
このアプリにはパズルをランダム試行で揃える機能が実装されており，これをスタートすると図 13(d)に示すように展開図の配色が次々に変化し，アプリがパズルを揃えている様子を確認できる．この際，見学者には実物のパズルを解くように指示し，アプリと対戦させる．



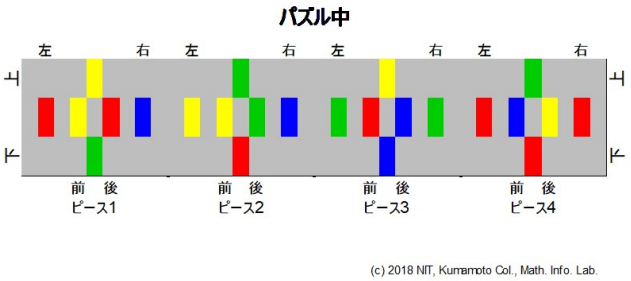

(a) タイトル画面



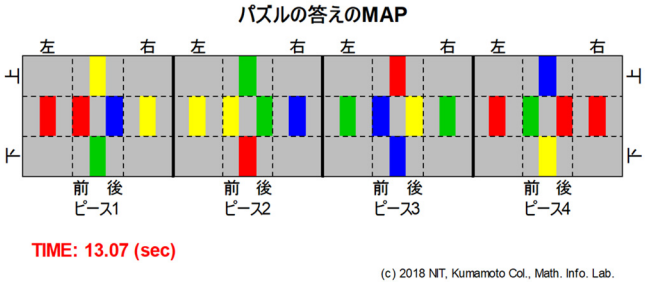

(b) パズルのシャッフル



(c) 展開図を確認しながら実物のパズルを配置




(d) パズルの求解途中




(e) パズルの解の表示

図 13. インスタント・インサニティのアプリの実行例

アプリが最長 2 万回(所要時間は約 2 分)までに解を求められれば，図 13(e)のようにその答えを表示するが，求められなかった場合でも他の解の例を表示する．見学者がアプリの解の表示時に，実物のパズルが解けていない場合は，見学者は図 13(e)の展開図を確認しながら，実物のパズルを配置し，解き終わってから終了する．

パズル終了後，各見学者に図 13 に示したパズルの展開図が理解できたかどうかを質問した．見学者の内訳は，小学生:1 名，中学生:31 名，高専本科生・教員・保護者:11 名であった．質問の回答をまとめると，「理解できた」が 29 名(67%)，「理解しづらかった・できなかった」が 14 名(33%)であり，概ねアプリで表示された展開図が理解できているものと考えられる．

図 13 の展開図は，各キューブの 1-モード行列展開から作成されたものである．アプリのユーザーは，パズルのシャッフル表示時と解の表示時に展開図を見ながら実物のパズルを配置することで，1-モード行列展開の並びを学習する効果があると考えられる．この事例のように，本アプリを利用することにより，テンソル分解や多次元データ処理への関心を持たせることが可能で，卒業研究・講義などに利用すればテンソル分解アルゴリズムやそのプログラミングの理解支援に繋がると考えており，今後も試用を行っていきたい．

## 5. まとめ

本論文では，立体パズルとしてポピュラーなルービックキューブとインスタント・インサニティを題材として取り上げ，まず各パズルの高階テンソル表現の例を示して，行列展開アルゴリズムにより展開図を作成する方法とそのスクリ

プトについて述べた.

そして, 本校専攻科生にルービックキューブを題材として与えて, このパズルの展開図作成アプリを実装させた事例について述べ, この学生が在学中の実習や研究を通して, テンソル分解を理解しプログラミングができる力のあることを確認した.

また, オープンキャンパスの研究室展示に訪れた小中学生および保護者等に対し, インスタント・インサニティの展開図を表示するアプリと実物のパズルを使ってもらい, 展開図が理解できるかどうか調査した事例を述べ, アプリに表示される展開図が概ね理解できることを確認した.

今後の課題としては, ルービックキューブについては, 同じ位置での同じキューブの回転が考慮されていないため, その高階テンソル表現と展開図表現の改善の必要があることと, インスタント・インサニティについては, より直感的に理解できるような展開図表示方法への改良などが挙げられる. さらに, 本校学生に対して, テンソル分解の教育に本格的に利用して, 効果を評価することも今後行わなければならないことである.

## 参考文献


[1] L. De Lathauwer “A Survey of Tensor Methods”, Proc. IEEE International Symposium on Circuits and Systems, pp.2773-2776, 2009.

[2] 村上純, 山本直樹, 田所嘉昭 “べき乗法を用いた 3 階テンソル積展開の高速計算法”, 電子情報通信学会論文誌 A, Vol. J82-A, No.8, pp.1351-1359, 1999.

[3] A. Ishida, T. Noda, J. Murakami, N. Yamamoto, and C. Okuma “Calculation of Fourth-Order Tensor Product Expansion by Power Method and Comparison of It with Higher-Order Singular Value Decomposition”, Applied Mechanics and Materials, Vols. 444-445, pp.703-711, 2013.

[4] N. Yamamoto, J. Murakami, K. Fujii, C. Okuma, S. Saito, T. Izumi, and N. Hayashida “Measurement and Analysis of the Functional Independence Measure Data by Using Nonnegative Matrix Factorization Method”, Advanced Materials Research, Vols. 718-720, pp.630-635, 2013.

[5] A. Ishida, K. Kawakami, D. Furushima, N. Yamamoto, and J. Murakami “Analysis of Relationship between Amount of Physical Activity of Patients in Rehabilitation and Their ADL Scores Using Multidimensional PCA”, Advances in Intelligent Systems and Computing, Vol. 690, pp.147-158, 2017.

[6] L. De Lathauwer, B. De Moor, and J. Vandewalle “A Multilinear Singular Value Decomposition”, SIAM Journal on Matrix Analysis and Applications, Vol.21, No.4, pp.1253-1278, 2000.

[7] 大隈千春, 長岡翔, 村上純, 山本直樹, 石田明男 “計算過程の可視化による高次特異値分解の理解支援システムの開発”, 論文集「高専教育」, Vol.38, pp.129-134, 2015.

[8] 山本直樹, 石田明男, 平田将大, 村上純 “立体パズルを利用した多次元データ分解手法の理解支援の試み”, 熊本高等専門学校研究紀要, Vol.9, No.1, pp.67-74, 2018.

[9] A. Ishida, N. Yamamoto, and J. Murakami, and N. Oishi “Solving 3-D Puzzles Using Tensor Decomposition and Application to Education of Multidimensional Data Analysis”, Intl. Journal of Machine Learning and Computing, Vol.8, No.5, pp.447-453, 2018.

[10] 山本直樹, 石田明男, 村上純, 大石信弘 “カラーダイスパズルを利用した *n*-モード行列展開の理解支援アプリの試作”, 熊本高等専門学校研究紀要, Vol.10, No.1, pp.75-78, 2019.

[11] 石田明男, 山本直樹, 大石信弘, 村上純 “多次元データ分解の手法を用いた立体パズルの解法”, 初等数学, Vol. 83, pp.18-22, 2018.（継続して掲載中）

[12] “Rubik’s Official Website”, Rubik’s, https://eu.rubiks.com/, Retrieved Feb. 27, 2019.

[13] 高木茂男 “PLAY PUZZLE パズルの百科”, pp.41-42, 平凡社, 1981.

[14] 日本ルービックキューブ協会, “ルービックキューブ ver.2.0 完全攻略公式ガイドブック”, p.5, 永岡書店, 2016.

[15] 村上純, 日野満司, 山本直樹, 石田明男 “統計ソフト R による 多次元データ処理入門－仮説検定・分散分析・主成分分析”, pp.223-227, 日新出版, 2017.

[16] R Core Team “R: A language and environment for statistical computing”, R Foundation for Statistical Computing, Vienna, Austria.

[17] J. Li, J. Bien, and M. Wells “Package ‘rTensor’”, https://cran.r-project.org/web/packages/rTensor/rTensor.pdf, 2018.