順序尺度の相関係数（ポリコリック相関係数）について *

Kosugitti の研究ノート

1　導入

社会調査において，三件法，五件法で得られたデータを因子分析していたりするけど，実は理論的にはマズイことがある。統計学者に言わせると，連続変数と見なしていいのは七件法からで (吉野・千野・山崎,2007)，それ以下のものはせいぜい順序尺度レベルであるからだ。順序尺度レベルということは，加減乗除の対象にならないってことで，平均をとるのは勿論，相関係数を求めて因子分析するなんて，もってのほかということになる。

しかしまぁ，今までは慣例的に行われてきている。心理学で調査研究をやった人であれば，むしろ五件法なら因子分析せよ，と教わったはずだ。しかしその背後には，「他に方法がないんだから仕方がないでしょ」という妥協があったことを知っている人は少ないのかもしれない。

例えば今でこそ，因子分析といえば最尤法プロマックス回転（あるいはクォーティミン回転。少なくとも斜交回転）が主流になってきたが，そうなると主因子法バリマックス回転で因子分析せよと教わったのは何だったのか，という気持ちになるだろう。あれも時代の要請で，当時は統計パッケージに斜交回転が入ってなかったから，仕方がなかった＝妥協の産物だったのだ（著者の記憶では，SPSSver8 ぐらいから斜交になったはずである。1997 年ぐらいの話。）。

同様に，今となっては五件法をそのまま因子分析するのは時代遅れである。統計学的には，早くから正しい方法が示されている。例えば狩野・三浦 (2002) によると，順序尺度を分析するには

1. 連続とみなす
2. 多分相関係数（polychoric correlation coefficient），多分系列相関係数 (polyserial correlation coefficient) を使う
3. 多項分布に基づく方法をとる

の三択になるとしている。

最初のは従来通りの妥協案。最後のは，柳井・繁桝・前川・市川 (1999) に書いてあるらしい。筆者も勉強中。

本稿では，二番目の多分相関係数（ポリコリック相関係数）や多分系列相関係数（ポリシリアル相関係数）について述べる。これらは最近，M-plus や R のパッケージの一つとして算出される**順序尺度の相関係数**であり，これを使えば冒頭に述べたような「妥協」について解決されるのである。

(余談) あくまでも厳密さの程度問題であり，ポリコリックを使ったからといって，劇的に因子構造が変わるというようなことはない。しかし，あくまでも統計的正当性をもとめるのであれば，徹底するべきだと筆者は考えます。

---

* written on 2008 年頃 / Last updated on 2019.08.20

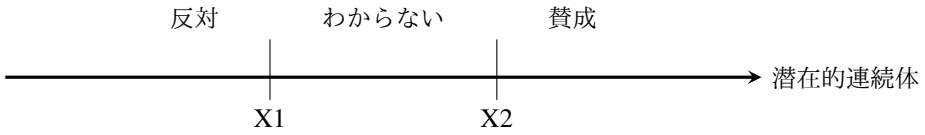

図 1　連続体と表現形。ある量 X1 を超えたら，態度は「反対」から「わからない」に変わる。さらに X2 を超えたら「賛成」に変わる。

1.1　名称について

聞き慣れない言葉が多く出てくると思われるので，ここで名称の整理をしておく。ポリコリック，ポリシリアル，テトラコリックという三つの言葉についての解説である。

ポリコリック相関係数 Polychoric Correlation は，上述の通り「**多分相関係数**」と訳される。順序尺度と順序尺度の相関係数である。

ポリシリアル相関係数 Polyserial Correlation は，同様に「**多分系列相関係数**」，あるいは「**重双相関係数**」と訳される。順序尺度と連続尺度の相関係数である。例えば項目反応理論の一部，段階反応モデル Graded Response Model や部分採点モデル Partial Credit Model で，順序尺度と見なされる各項目と，間隔尺度と見なされるテストの合計得点は，ポリシリアル相関係数で求める。

テトラコリック相関係数 Tetrachoric Correlation は**四分相関係数**と訳される。四分は 2 × 2，つまり二値データ同士の相関係数である。これはポリコリック相関係数の特殊な場合である，と考えればよい。基本的には項目反応理論の，正誤データの変数間相関はテトラコリック相関係数になる。各項目とテストの合計得点との相関は，各項目をダミー変数と見なしてピアソンの積率相関係数を求めればよいので，さほど問題にされることはない。

ポリコリック相関係数は，もとのデータを順序尺度と見なして，反応カテゴリ間の距離 (例えば「はい」と「どちらでもない」の距離) を閾値で調整することができる。このため，天井効果・床効果がみられるような反応項目であっても，相関係数を正しく見積もってくれる。実際に算出してみた所感としては，順序尺度を間隔尺度だと見なしてピアソンの積率相関係数を出すより，ポリコリック相関係数のほうが大きい値が得られることが多い。

2　基本的な考え方

基本的な考え方は，データの表現形が順序尺度であって，その背後に連続的なものがある，とするものである。社会的態度でも何でもいいが，$-\infty$ から ∞ までなめらかに変化する量があって，それが表に出てくるときに「賛成・わからない・反対」の三段階とか，「非常に賛成〜非常に反対」の五段階になってしまうという考え方である。

どこに意見の変わり目（＝閾値）があるのかはわからないけど，背後に連続量を仮定しようというわけである。

さて，変数 x と y がそれぞれ順序尺度で得られたとしよう。この二変数の相関係数を考えるにあたって，まず二変数のクロス集計表を書いてみるところから始めてみよう。x と y それぞれが，s 段階，r 段階に分割されているとしよう。このときクロス集計表は，表 2 のようになる。

		b_1	b_2	b_3	$\cdots$	b_{r-1}
	y / x	1	2	3	$\cdots$	r
a_1	1	n_{11}	n_{12}	n_{13}	$\cdots$	n_{1r}
a_2	2	n_{21}	n_{22}			
a_3	3	n_{31}		$\ddots$		
a_{s-1}	$\vdots$	$\vdots$				
	s	n_{s1}				n_{sr}

図 2　素データの一般的な形。x と y のクロス集計表。ここで a_i や b_j は閾値を表し，$a_0 = b_0 = -\infty$，$a_s = b_r = +\infty$ である。

さて，x と y の奥に潜在変数 ξ と η があり，それらは正規分布していると仮定するのであった。となれば，変数 x と ξ の関係は次のように書ける。

$$\begin{array}{lll} x = 1 & if & \xi < a_1 \\ x = 2 & if & a_1 \leq \xi < a_2 \\ x = 3 & if & a_2 \leq \xi < a_3 \\ \vdots & \vdots & \\ x = s & if & a_{s-1} \leq \xi \end{array} \tag{1}$$

y についても同様である。パラメータ a_1 が閾値，つまり意見の変わり目である。問題は，ξ と η の相関である ρ を，表 2 から求めたいということである。

3　二変数正規分布

ここで，我々が求めようとしている相関係数 ρ の姿をイメージしておこう。ある変数 X が正規分布する，というのは図にすると図 3 のようなものだ。これが二変数になると，図が立体になる。例えば，全く無相関であ

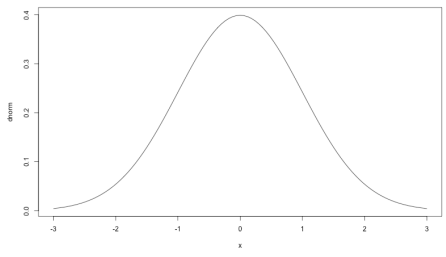

図 3　一変数の正規分布

れば，図 4 のようになる。徐々に相関係数が大きくなっていけば，立体部が細く尖っていくのが見て取れるだ

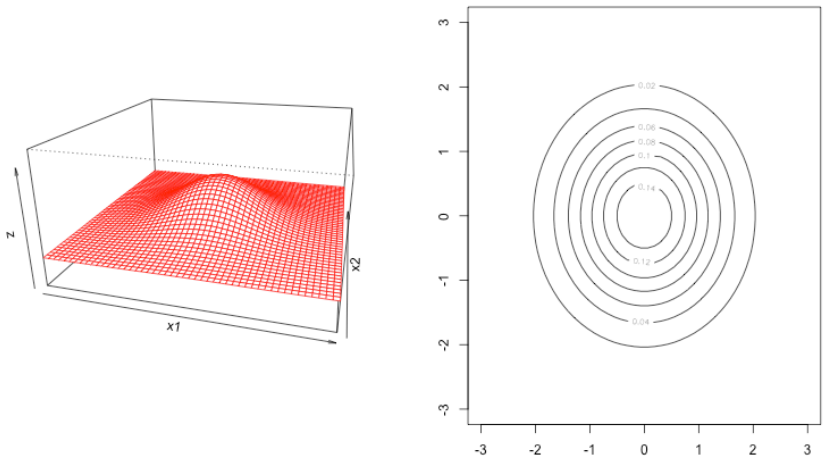

図 4　二変数正規分布，相関 $\rho = 0.0$

ろう。

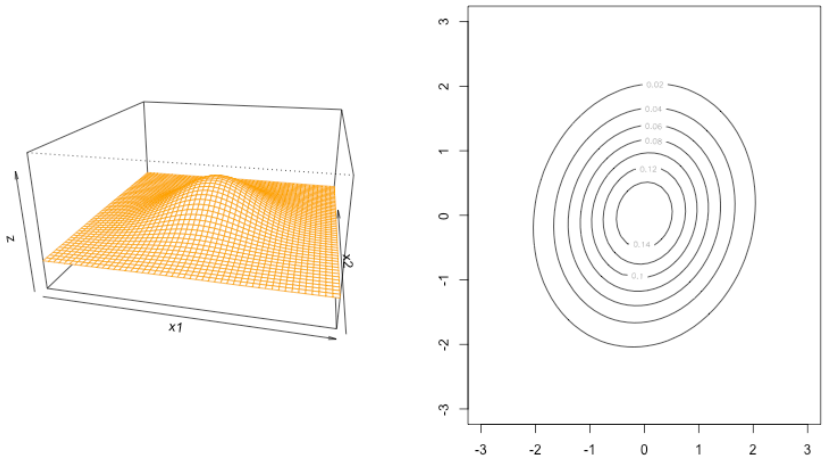

図 5　二変数正規分布，相関 $\rho = 0.1$

二つの変数が同時に正規分布するときの確率密度関数 (bivariate normal density function) は，次のような式

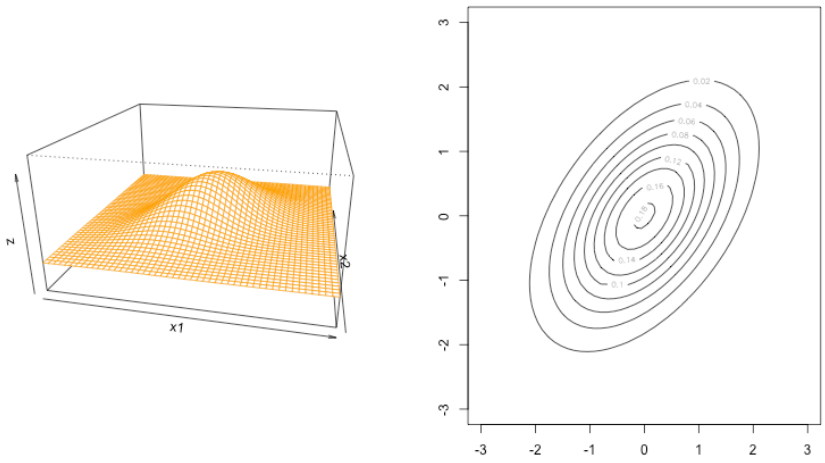

図 6　二変数正規分布，相関 $\rho = 0.5$

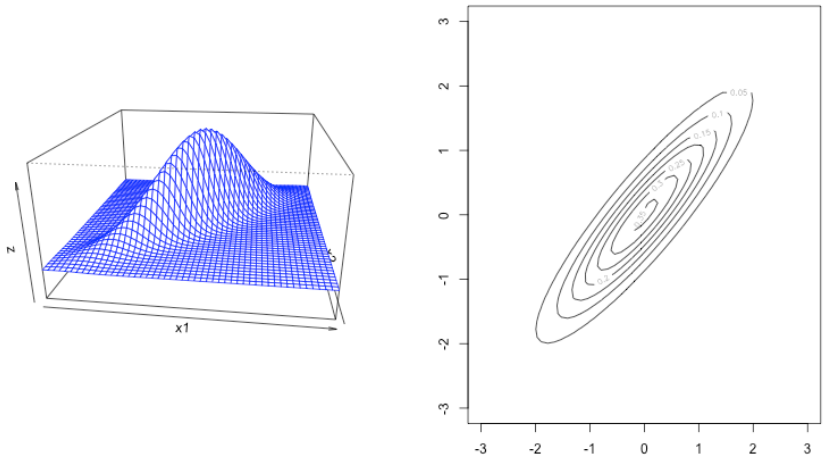

図 7　二変数正規分布，相関 $\rho = 0.9$

で表される。ちなみに図 4〜7 はこの式をプロットしたものである[*1]。

$$f(x, y; rho) = \frac{1}{2\pi\sqrt{1-\rho_{xy}^2}} exp\left\{-\frac{1}{2(1-\rho_{xy}^2)}(x^2 - 2\rho_{xy} + y^2)\right\} \tag{2}$$

イメージとしていえば，図 8 左にあるようなクロス集計表を，図 8 右のような 3 次元棒グラフにして，これが二次元正規分布に従っているとすると ρ の値はどれぐらいが妥当なのだろうか？と考えるのがポリコリック

[*1] 描画する R コードは付録を参照

相関係数の最尤推定量ということになる[*2]。

	A	B	C
1	10	20	9
2	26	58	21
3	8	12	9

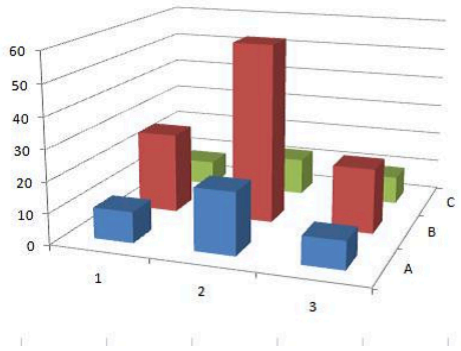

図 8　度数分布を二次元ヒストグラムにしたもの

4　最尤推定の二つの方法

ポリコリック相関係数を推定するとき，推定すべきパラメータは，相関係数 ρ だけではなく，変数 x に含まれる閾値 $a_1, a_2, ..., a_s$ と変数 y に含まれる閾値 $b_1, b_2, ..., b_r$ も，ということになる。

具体的な方法としては二種類ある。一つは，全てのパラメータを同時に求める方法で，これは理論的に正しいが偏微分連立方程式を解くことになるので面倒である。もう一つの方法は，閾値を周辺度数からもとめちゃって，ρ についての方程式一つにしちゃう，というものである。後者の方法を，特に二段階最尤推定twe-step ML 法という。実際，アプリケーションに実装する上では二段階最尤推定法が主に用いられているようである。

ところで，この最尤推定アプローチには，標準誤差が対数尤度の二次導関数の逆数として簡単に得られるというメリットもある。本稿では，以下 Olsson(1979) に沿って議論を進めていくが，標準誤差の推定など，より詳細は原典を当たって欲しい。

4.1　尤度方程式の導出

データが観測度数 n_{ij} から構成されているとする。ここで，表 1 の通り $i = 1, 2, \cdots, s, j = 1, 2, \cdots, r$ である。ここで，観測度数がセル (i,j) に落ち込む確率を π_{ij} とすれば，そのサンプルの尤度は

$$L = C \cdot \prod_{i}^{s} \cdot \prod_{j}^{r} \pi_{ij}^{n_{ij}} \tag{3}$$

である。ここで C は定数。対数をとると，

$$l = lnL = \ln C + \sum_{i=1}^{s} \sum_{j=1}^{r} n_{ij} \ln \pi_{ij}. \tag{4}$$

x における閾値は a_i で表され，$i = 0, 1, \cdots, s$ である。y における閾値は b_j で，$j = 0, 1, \cdots, r$ である。ここ

[*2] 本来 3D グラフはデータの記述としては適切な視覚化とは言えない。今回はイメージとして必要なので仕方なく採用したけど。

で，$a_0 = b_0 = -\infty$，$a_s = b_r = +\infty$ である。従って，

$$\pi_{ij} = \Phi_2(a_i, b_j) - \Phi_2(a_{i-1}, b_j) - \Phi_2(a_i, b_{j-1}) + \Phi_2(a_{i-1}, b_{j-1}) \tag{5}$$

となる。ここで Φ_2 は，相関係数 ρ のときの二変数正規分布関数 bivariate normal distribution function with ρ である。

4.2 手法 1:全てのパラメータを同時に推定する

推定すべきパラメータは，$\rho, a_1, \cdots, a_{s-1}, b_1, \cdots, b_{r-1}$ である。l についての偏微分をそれぞれのパラメタで行うと，

$$\frac{\partial l}{\partial \rho} = \sum_{i=1}^{s}\sum_{j=1}^{r} \frac{n_{ij}}{\pi_{ij}} \frac{\partial \pi_{ij}}{\partial \rho} \tag{6}$$

$$\frac{\partial l}{\partial a_k} = \sum_{i=1}^{s}\sum_{j=1}^{r} \frac{n_{ij}}{\pi_{ij}} \frac{\partial \pi_{ij}}{\partial a_k} \tag{7}$$

$$\frac{\partial l}{\partial b_m} = \sum_{i=1}^{s}\sum_{j=1}^{r} \frac{n_{ij}}{\pi_{ij}} \frac{\partial \pi_{ij}}{\partial b_m}. \tag{8}$$

となる。ここで，ϕ_2 を二変数正規密度関数とすると，$\partial\Phi_2(u,v)/\partial\rho = \phi_2(u,v)$ である。すると

$$\frac{\partial \pi_{ij}}{\partial \rho} = \phi_2(a_i, b_j) - \phi_2(a_{i-1}, b_j) - \phi_2(a_i, b_{j-1}) + \phi_2(a_{i-1}, b_{j-1}). \tag{9}$$

であり，式 6 は次のように書き直せる。

$$\frac{\partial l}{\partial \rho} = \sum_{i=1}^{s}\sum_{j=1}^{r} \frac{n_{ij}}{\pi_{ij}} \left\{\phi_2(a_i, b_j) - \phi_2(a_{i-1}, b_j) - \phi_2(a_i, b_{j-1}) + \phi_2(a_{i-1}, b_{j-1})\right\}. \tag{10}$$

式 7 においては，次のことが明らかである。

$$\frac{\partial \pi_{ij}}{\partial a_k} = \begin{cases} 0 & i \neq k \text{ かつ } i \neq k+1 \text{ であれば,} \pi_{ij} \text{の式に } a_k \text{を含まないので} \\ \frac{\partial \Phi_2(a_k, b_j)}{\partial a_k} - \frac{\partial \Phi_2(a_k, b_{j-1})}{\partial a_k} & k = i \text{ のとき} \\ -\frac{\partial \Phi_2(a_k, b_j)}{\partial a_k} + \frac{\partial \Phi_2(a_k, b_{j-1})}{\partial a_k} & k = i-1 \text{ のとき} \end{cases} \tag{11}$$

だから，式 7 は i を k から $k+1$ まで変化させれば十分である。故に，式 7 は

$$\begin{aligned} \frac{\partial l}{\partial a_k} &= \sum_{j=1}^{r} \frac{n_{ij}}{\pi_{kj}} \left\{\frac{\partial \Phi_2(a_k, b_j)}{\partial a_k} - \frac{\partial \Phi_2(a_k, b_{j-1})}{\partial a_k}\right\} \\ &+ \frac{b_{k+1j}}{\pi_{k+1j}} \left\{-\frac{\partial \Phi_2(a_k, b_j)}{\partial a_k} + \frac{\partial \Phi_2(a_k, b_{j-1})}{\partial a_k}\right\} \\ &= \sum_{j=1}^{r} \left(\frac{n_{kj}}{\pi_{kj}} - \frac{n_{k+1j}}{\pi_{k+1j}}\right) \left\{\frac{\partial \Phi_2(a_k, b_j)}{\partial a_k} - \frac{\partial \Phi_2(a_k, b_{j-1})}{\partial a_k}\right\}. \end{aligned} \tag{12}$$

また，もし ϕ_1 と Φ_1 を，それぞれ一変数についての生起確率密度と正規分布関数とすれば，

$$\frac{\partial \Phi_2(u,v)}{\partial u} = \phi_1(u) \cdot \Phi_1\left\{\frac{(v - \rho u)}{(1-\rho^2)^{1/2}}\right\} \tag{13}$$

となる (Tallis,1962,p.346)。さすれば式 7 は，次のようになる。

$$\frac{\partial l}{\partial a_k} = \sum_{j=1}^{r}\left(\frac{n_{kj}}{\pi_{kj}} - \frac{n_{k+1j}}{\pi_{k+1j}}\right)\cdot\phi_1(a_k)\cdot\left[\Phi_1\left\{\frac{(b_k-\rho a_k)}{(1-\rho^2)^{1/2}}\right\} - \Phi_1\left\{\frac{(b_{j-1}-\rho a_k)}{(1-\rho^2)^{1/2}}\right\}\right]. \tag{14}$$

同様に，

$$\frac{\partial l}{\partial b_m} = \sum_{j=1}^{r}\left(\frac{n_{im}}{\pi_{im}} - \frac{n_{im+1}}{\pi_{im+1}}\right)\cdot\phi_1(b_m)\cdot\left[\Phi_1\left\{\frac{(a_i-\rho b_m)}{(1-\rho^2)^{1/2}}\right\} - \Phi_1\left\{\frac{(a_{i-1}-\rho b_m)}{(1-\rho^2)^{1/2}}\right\}\right]. \tag{15}$$

これらの式 10,14,15 が，対数尤度の一次導関数を構成することになる。

4.3 方法 2:Two-Step 最尤推定法〜閾値を周辺度数から算出する場合〜

この場合，方程式は次のようにたてられる。

$$\frac{\partial l}{\partial \rho} = \sum_{i=1}^{s}\sum_{j=1}^{r}\frac{n_{ij}}{\pi_{ij}}\left\{\phi_2(a_i,b_j) - \phi_2(a_{i-1},b_j) - \phi_2(a_i,b_{j-1}) + \phi_2(a_{i-1},b_{j-1})\right\} = 0. \tag{16}$$

$$a_i = \Phi_1^{-1}(P_{i.}) \tag{17}$$

$$b_j = \Phi_1^{-1}(P_{.j}) \tag{18}$$

ここで P_{ij} はセル (i,j) の観測度数であり，$P_{i.}$ と $P_{.j}$ は観測された累積周辺分布である。つまり，

$$P_{i.} = \sum_{k=1}^{i}\sum_{j=1}^{r} P_{kj} \tag{19}$$

かつ

$$P_{.j} = \sum_{i=1}^{s}\sum_{k=1}^{j} P_{ik} \tag{20}$$

5 具体的例

ここでは，二段階最尤推定法によるポリコリック相関係数の求め方を具体的に論じる。コンピュータプログラムについては，C 言語で書いているが，「美しくなくともわかりやすく」をモットーに書いているので，簡単に多言語に翻訳できることと思う。

まず，式 17 や式 18 にみられる一変数正規分布の逆関数だが，これは数値計算的にはあるパーセンタイル点の上側確率として知られているものである。近似計算法としては次のようなものがある (パソコン統計解析ハンドブック 1 基礎統計編より。有効桁数下三桁)。

```
double Normal_Percent(double val){
        double b0,b1,b2,b3,b4,b5,b6,b7,b8,y,alpha,sum,v;
        if( val > 0.5 ){ v = 1 - val; }else{ v = val; }

        b0 =  0.1570796288*10;
        b1 =  0.3706987906e-1;
```

```
        b2 = -0.8364353589e-3;
        b3 = -0.2250947176e-3;
        b4 =  0.6841218299e-5;
        b5 =  0.5824238515e-5;
        b6 = -0.1045274970e-5;
        b7 =  0.8360937017e-7;
        b8 = -0.3231081277e-8;

        y = -log(4*v*(1-v));

        sum = b0 + (b1*y) + (b2*pow(y,2)) + (b3*pow(y,3)) + (b4*pow(y,4)) +
          (b5*pow(y,5)) + (b6*pow(y,6)) + (b7*pow(y,7)) + (b8*pow(y,8));
        sum = sum * y;

        alpha = sqrt(sum);
        if( val < 0.5 ){ alpha = -1 * alpha; }

        return alpha;

}
```

これに累積相対度数を入れてやることで，閾値の推定値が得られる。数値例を元に考えてみよう。次のようなデータがあったとする。

表 1　クロス集計表の例 (出典：Encyclopedia of Statistical Sciences;Polychoric and Polyserial Correlations より)

	A	B	C	合計
甲	58	52	1	111
乙	26	58	3	87
丙	8	12	9	29
合計	92	122	13	227

例えば，一列目と二列目の間にある閾値は，

$$\hat{a_1} = \Phi^{-1}(111/227) = \Phi^{-1}(0.489) = -0.027$$

である。以下同様に，

$$\hat{a_2} = \Phi^{-1}(198/227) = \Phi^{-1}(0.872) = 1.137$$

$$\hat{b_1} = \Phi^{-1}(92/227) = \Phi^{-1}(0.754) = -0.239$$

$$\hat{b_2} = \Phi^{-1}(214/227) = \Phi^{-1}(0.942) = -1.578$$

と推定される。これをもとに，式 16 を解くことになる。

式 16 をみてみると，どうやら二変量正規分布の確率密度関数 (ϕ_2;probability density function;PDF) と累積分布関数 (Φ_2;cumulative distribution function;CDF) が要りそうである。前者は式 2 で求められるが，後者は数値計算による近似計算となる。具体的なプログラムは，次のようなものである（ハル,1994)

```
//(一次元) 正規分布の累積分布関数の近似式
double N(double z) {
    double b1 =  0.31938153;
    double b2 = -0.356563782;
    double b3 =  1.781477937;
    double b4 = -1.821255978;
    double b5 =  1.330274429;
    double p  =  0.2316419;
    double c2 =  0.3989423;

    if (z >  6.0) { return 1.0; }; // this guards against overflow
    if (z < -6.0) { return 0.0; };
    double a=fabs(z);
    double t = 1.0/(1.0+a*p);
    double b = c2*exp((-z)*(z/2.0));
    double n = ((((b5*t+b4)*t+b3)*t+b2)*t+b1)*t;
    n = 1.0-b*n;
    if ( z < 0.0 ) n = 1.0 - n;
    return n;
}

//二次元正規分布の累積分布関数の近似式

//式中で使用する関数
double f( double x, double y, double aprime, double bprime ,double rho ){
    double r = aprime * ( 2 * x - aprime ) + bprime *( 2 * y - bprime )
        + 2 * rho * ( x - aprime ) * ( y - bprime );
    return exp(r);
}

double sgn( double x) {  // sign function
    if (x>=0.0)  return 1.0;
    return -1.0;
}

//メイン
```

```
double biNormalInt(double a, double b, double rho){
    double val;
    if( a <= 0 && b <= 0 && rho <= 0 ){
        double aprime = a/(sqrt(2 * ( 1 -rho * rho )));
        double bprime = b/(sqrt(2 * ( 1- rho * rho )));
        long double A[15],B[15];
        A[0] = 5.54433663102343E-02;
        A[1] = 1.24027738987730E-01;
        A[2] = 1.75290943892075E-01;
        A[3] = 1.91488340747342E-01;
        A[4] = 1.63473797144070E-01;
        A[5] = 1.05937637278492E-01;
        A[6] = 5.00270211534535E-02;
        A[7] = 1.64429690052673E-02;
        A[8] = 3.57320421428311E-03;
        A[9] = 4.82896509305201E-04;
        A[10]= 3.74908650266318E-05;
        A[11]= 1.49368411589636E-06;
        A[12]= 2.55270496934465E-08;
        A[13]= 1.34217679136316E-10;
        A[14]= 9.56227446736465E-14;

        B[0] = 2.16869474675590E-02;
        B[1] = 1.12684220347775E-01;
        B[2] = 2.70492671421899E-01;
        B[3] = 4.86902370381935E-01;
        B[4] = 7.53043683072978E-01;
        B[5] = 1.06093100362236E+00;
        B[6] = 1.40425495820363E+00;
        B[7] = 1.77864637941183E+00;
        B[8] = 2.18170813144494E+00;
        B[9] = 2.61306084533352E+00;
        B[10]= 3.07461811380851E+00;
        B[11]= 3.57140815113714E+00;
        B[12]= 4.11373608977209E+00;
        B[13]= 4.72351306243148E+00;
        B[14]= 5.46048893578335E+00;

        long double sum = 0;
        for( int i = 0 ; i < 15 ; i++ ){
```

```
            for( int j = 0; j < 15 ; j++ ){
                sum = sum + A[i] * A[j] * f( B[i], B[j],aprime,bprime,rho);
            }
        }
        sum = sum * ( sqrt( 1.0 - rho * rho )/ PI );
        return (double)sum;
    }else if( a* b * rho <= 0.0 ){
        if( ( a <= 0.0 ) && ( b >= 0.0 ) && ( rho >= 0.0 )){
            return N(a) - biNormalInt(a,-b,-rho);
        }else if (( a >= 0.0) && ( b <= 0.0 ) && ( rho >= 0.0)){
            return N(b) - biNormalInt(-a,b,-rho);
        }else if (( a >= 0.0) && ( b >= 0.0 ) && ( rho <= 0.0)){
            return N(a) + N(b) - 1.0 + biNormalInt(-a,-b,rho);
        }
    }else if ( a * b * rho >= 0.0 ){
        double denum = sqrt( a* a -2 * rho * a * b + b * b);
        double rho1 = (( rho * a - b ) * sgn(a))/denum;
        double rho2 = (( rho * b - a ) * sgn(b))/denum;
        double delta = ( 1.0-sgn(a)*sgn(b))/ 4.0;
        return biNormalInt( a,0.0,rho1)+biNormalInt(b,0.0,rho2)-delta;
    }
    return -99.9;
}
```

ちなみに，このプログラムのメインでは 10^{-15} の精度まで算出できるようになっているが，実際には 10^{-4} ぐらいで十分であろう。その場合は，次の数列を使えばよい。

```
    douuble A[4] ={ 0.3253030,0.4211071,0.1334425,0.006374323 };
    double B[4] ={ 0.1337764,0.6243247,1.3425378,2.2626645};
    double sum = 0;
    for( int i = 0 ; i < 4 ; i++ ){
        for( int j = 0; j < 4 ; j++ ){
            sum = sum + A[i] * A[j] * f( B[i], B[j],aprime,bprime,rho);
        }
    }
```

数値例を見てみよう。表 1 を用いた結果，相関係数 ρ は 0.419899 と推定される。その場合，この CDF ルーチンを使って算出される，各セル出現期待値 (式 5 の π_{ij}) は次の通りである。これらの関数を使って，式 16 の方程式の解を求めることになる。

非線形方程式を解くアルゴリズムはニュートン法（あるいはニュートン・ラフソン法）が有名だが，途中

I	j	π_{ij}
1	1	0.2651410
1	2	0.2139610
1	3	0.0098850
2	1	0.1199310
2	2	0.2373620
2	3	0.0259676
3	1	0.0202151
3	2	0.0861221
3	3	0.0214161

でさらに微分した式を必要とするので，筆者には力不足であった。しかし今回 ρ は -1 から 1 の間にあることが明らかなので，やろうと思えば -1 から 0.001 刻みぐらいで逐一方程式を解き，解がある程度小さくなる $f(x) < \epsilon$ であればよし，とすれば見つかるのである。勿論これは非常に面倒なやり方で，二分法や割線法 (セカント法) を用いればもっと早く収束させることができる。詳しくは戸川 (1992) などを参考にされたし。

6 付録

描画などの R コード

```
library(tidyverse)
library(rgl)
library(mvtnorm)
### 1 変量正規分布
png("gauss.png", width = 800, height = 500) # 描画デバイスを開く
plot(dnorm, -3, 3)
dev.off()
### 2 変量正規分布　相関ゼロ
x1 <- seq(-3, 3, length = 50)
x2 <- x1
f <- function(x1, x2) {
  dmvnorm(matrix(c(x1, x2), ncol = 2),
    mean = c(0, 0),
    sigma = matrix(c(1, 0, 0, 1), ncol = 2)
  )
}
z <- outer(x1, x2, f)
z[is.na(z)] <- 1
op <- par(bg = "white")
```

```
png("2gauss0.png", width = 800, height = 500) # 描画デバイスを開く
split.screen(figs = c(1, 2))
screen(1)
persp(x1, x2, z,
  theta = 15, phi = 20, expand = 0.5,
  zlim = c(-0.1, 0.35), border = "red"
)
screen(2)
contour(x1, x2, z)
close.screen(1, 2)
dev.off()

### 2 変量正規分布　相関 0.1
x1 <- seq(-3, 3, length = 50)
x2 <- x1
f <- function(x1, x2) {
  dmvnorm(matrix(c(x1, x2), ncol = 2),
    mean = c(0, 0),
    sigma = matrix(c(1, 0.1, 0.1, 1), ncol = 2)
  )
}
z <- outer(x1, x2, f)
z[is.na(z)] <- 1
op <- par(bg = "white")
png("2gauss1.png", width = 800, height = 500) # 描画デバイスを開く
split.screen(figs = c(1, 2))
screen(1)
persp(x1, x2, z,
  theta = 13, phi = 20, expand = 0.5,
  zlim = c(-0.1, 0.35), border = "orange"
)
screen(2)
contour(x1, x2, z)
close.screen(1, 2)
dev.off()

### 2 変量正規分布　相関 0.5
x1 <- seq(-3, 3, length = 50)
x2 <- x1
```

```
f <- function(x1, x2) {
  dmvnorm(matrix(c(x1, x2), ncol = 2),
    mean = c(0, 0),
    sigma = matrix(c(1, 0.5, 0.5, 1), ncol = 2)
  )
}
z <- outer(x1, x2, f)
z[is.na(z)] <- 1
op <- par(bg = "white")
png("2gauss5.png", width = 800, height = 500) # 描画デバイスを開く
split.screen(figs = c(1, 2))
screen(1)
persp(x1, x2, z,
  theta = 12, phi = 20, expand = 0.5,
  zlim = c(-0.1, 0.35), border = "orange"
)
screen(2)
contour(x1, x2, z)
close.screen(1, 2)
dev.off()
### 2 変量正規分布　相関 0.9
x1 <- seq(-3, 3, length = 50)
x2 <- x1
f <- function(x1, x2) {
  dmvnorm(matrix(c(x1, x2), ncol = 2),
    mean = c(0, 0),
    sigma = matrix(c(1, 0.9, 0.9, 1), ncol = 2)
  )
}
z <- outer(x1, x2, f)
z[is.na(z)] <- 1
op <- par(bg = "white")
png("2gauss9.png", width = 800, height = 500) # 描画デバイスを開く
split.screen(figs = c(1, 2))
screen(1)
persp(x1, x2, z,
  theta = 7, phi = 20, expand = 0.5,
  zlim = c(-0.1, 0.35), border = "blue"
)
screen(2)
```

```
contour(x1, x2, z)
close.screen(1, 2)
dev.off()
```

参考文献

[1] Hull,J. Options, futures, and other derivative securities 社団法人金融財政事情研究会 (訳)　フィナンシャルエンジニアリング, 株式会社きんざい.

[2] 狩野裕・三浦麻子　2002　グラフィカル多変量解析―AMOS，EQS，CALIS による 目で見る共分散構造分析 現代数学社

[3] Olsson,U. 1979 Maximum Likelihood Estimation of the Polychoric Correlation Coefficient. *Psychometrika*,44(4),443–460.

[4] 戸川隼人 1992 UNIX ワークステーションによる科学技術計算ハンドブック　サイエンス社

[5] 柳井晴夫・繁枡算男・前川眞一・市川雅教　1999　因子分析－その理論と方法－朝倉書店

[6] 吉野諒三・千野直仁・山岸侯彦　2007　数理心理学　培風館