3. 確率

3.1. 二項分布確率

ある事象が起こる確率にはさまざまなものがあります。たとえば，サイコロには{1, 2, 3, 4, 5, 6}という目があるので，1 回サイコロを投げるとき(「試行」trial と言います），それぞれの目「1」「2」…　「6」が出る確率は 1/6 ずつです。これらの目の中の 1 つ，たとえば「1」が出る確率は 1/6 なので，逆に「1」が出ない確率は 1 - 1/6 = 5/6 になります。両方の和は 1 です(1/6 + 5/6=1)。

次の表の T (True)は「１」が出ることを示し，F (False)は「１」が出ないことを示しています。確率の総和が 1 になることを確認してください(1/6 + 5/6 = 1)。

「１」	T の数	確率
T	1	1/6 = 0.167
F	0	5/6 = 0.833

次にサイコロを 2 回投げる場合(試行回数 n=2)，または 2 個のサイコロを投げる場合(サイコロの数 n=2)を考えましょう。たとえば 1 回目が F で，2 回目が T とすると，これを F, T と書きます。4 つの場合(T,T; T,F; F,T; F,F)のそれぞれの確率は，2 つのサイコロの T, F の確率の積になります[1]。この場合も確率の総和は 1 です(1/36 + 5/36 + 5/36 + 25/36 = 1)。

「１」	T の数	確率
T, T	2	(1/6) * (1/6) = 1/36 = 0.028
T, F	1	(1/6) * (5/6) = 5/36 = 0.139
F, T	1	(5/6) * (1/6) = 5/36 = 0.139
F, F	0	(5/6) * (5/6) = 25/36 = 0.694

さらに，サイコロを 3 回投げる場合(試行回数 n=3)を考えます。この場合も確率の総和は 1 になります。

[1] 互いに影響しない（独立な）複数の事象の確率はそれぞれの事象の確率の積になります。たとえば，ある趣味の会に，1/2 の確率で出席する A さんと 1/3 の確率で出席する B さんの 2 人が同時に出席する確率は(1/2) * (1/3) = 1/6 になります。もし，A さんと B さんが知り合いで誘いあってこの趣味の会に出席することがあるときは，互いに独立していないので，このような確率の積を使うことができません。

「１」	T の数	確率
T, T, T	3	(1/6) * (1/6) * (1/6) = 1/216=0.005
T, T, F	2	(1/6) * (1/6) * (5/6) = 5/216=0.023
T, F, T	2	(1/6) * (5/6) * (1/6) = 5/216=0.023
T, F, F	1	(1/6) * (5/6) * (5/6) = 25/216 = 0.116
F, T, T	2	(5/6) * (1/6) * (1/6) = 5/216 = 0.023
F, T, F	1	(5/6) * (1/6) * (5/6) = 25/216 = 0.116
F, F, T	1	(5/6) * (5/6) * (1/6) = 25/216 = 0.116
F, F, F	0	(5/6) * (5/6) * (5/6) = 125/216 = 0.579

ここで，たとえばサイコロを 3 回投げて順番を問題にせずに，全部で 2 回「１」が出る場合(T の数=2)の確率を求めると，上の表から

「１」	T の数	確率
T, T, F	2	(1/6) * (1/6) * (5/6) = 5/216 = 0.023
T, F, T	2	(1/6) * (5/6) * (1/6) = 5/216 = 0.023
F, T, T	2	(5/6) * (1/6) * (1/6) = 5/216 = 0.023

を合計した確率，つまり，(5/216) + (5/216) + (5/216) = 15/216=0.069 になります。これは「１」(T)が 2 回出る場合の確率(5/216)を 3 倍した数です。この倍数(=3)を求めるためには，このように少ない試行回数(3 回)ならばすぐ計算できますが，それが多くなると一般式を使わなければなりません。n 回の試行で T が x 回選ばれる場合の数は n**Cx** という「組み合わせ」(combination)の値になります(→**順列と組み合わせ**)。ここでは，T が 2 個で F が 1 個の組み合わせになるので $_3C_2$ で計算します。

$$nCx = {}_3C_2 = (3 * 2) / (2 * 1) = 6/2 = 3$$

そこで，3 回の試行で T が 2 回出る確率は[2]

$${}_3\mathbf{C}_2 * (1/6)^2 * (5/6) = (3 * 2) / (2 * 1) * (1/6)^2 * (5/6) = 15/216=0.069$$

この確率(二項確率 binomial probability: BinD)を一般式で示すと[3]

$$\begin{aligned}\mathrm{BinD}(x, n, e) &= nCx * e^x * (1–e)^{(n-x)}\\ &= n! / [x! (n - x)!] * e^x * (1–e)^{(n-x)}\end{aligned}$$

次が R で作成した関数 BinD と R 関数 dbinom の実行結果です(→プログ

[2] 上の表の「確率」の列中の(1/6)と(5/6)のそれぞれの回数(2, 1)を確認してください。
[3] 組み合わせ ${}_n\mathbf{C}_x$ の一般式は

$${}_n\mathbf{C}_x = {}_n\mathbf{P}_x / x! = [n (n– 1) (n – 2) \ldots (n – x + 1)] / x! = n! / [x! (n - x)!]$$

ラム)。

```
x=2; n=3; e=1/6; BinD(x,n,e); dbinom(x,n,e)
# 0.06944444 0.06944444
```

ここで n はサイコロを投げた総回数（試行数）, x は成功回数（T の数）, e は T の確率(成功確率：1/6)，1-e は F の確率(失敗確率：5/6)を示します。

二項確率 BinD は，確率 e で起こる現象が，n 回の試行中で x 回出現する確率を求めたものです。ここで「確率」という言葉が異なる意味で 2 回使われているので，前提とする確率(成功確率)を「期待確率」(e:expected probability: EP)と呼び[4]，出現する確率を「出現確率」(occurrent probability: OP)と呼んで区別しましょう[5]。また，この出現確率(OP)は n 回の中で x 回だけ出現する確率です。一方，n 回の試行中で 0 回から x 回まで出現する確率を「累積確率」(cumulative probability: CP)と呼びます。

二項分布を示すデータの平均と分散は次のように数理的に導出されます。

X	1	0	和
P	e	1 - e	1

n 回の試行での確率変数 $X_1, X_2, \ldots, X_n$ についてのそれぞれの平均 $M(X_i)$ と，それぞれの分散 $V(X_i)$を計算します。

$M(X_i)$ = Σ(i) x(i) * p(i)　←**●平均と分散**
= 1 * e + 0 * (1 - e)　←上の表
= e

$V(X_i)$ = Σ(i) [x(i) – m]^2 * p(i)　←**●平均と分散**, m:平均
= (1 - m)^2 * e + (0 - m)^2 * (1 - e)　←上の表
= (1 - e)^2 * e + (0 - e)^2 * (1 - e)　←m = e
= (1 - e)^2 * e + e^2 * (1 - e)
= e * (1 - e) * (1 - e + e)
= e * (1 - e)

この平均と分散が試行数 n 回の X について考えると

二項分布の平均(M)：$M(X) = n * M(X_i) = n * e$

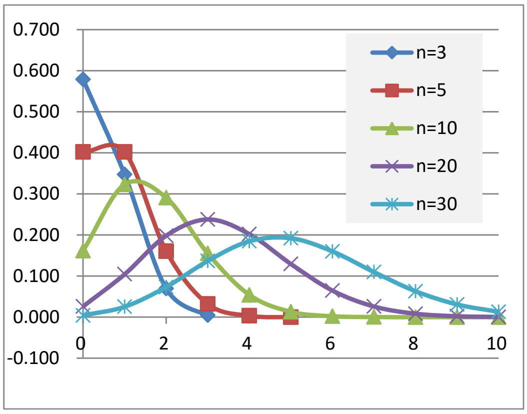
[4] 「期待確率」(expected probability)は「予想確率」という用語にしたほうがわかりやすいかもしれません。しかし統計学で expectation は「予想値」ではなく「期待値」と訳されているので，ここでも「期待確率」とします。
[5] ここで生起数(x)以下の確率を累積した確率を使い，生起数に対応する個別の確率そのものを使わない理由は，個別の確率はデータの個数が多くなるにつれて減少するので不都合だからです。累積生起確率は，当該の生起数以下すべてのケースの確率の総和です。つまり，その生起数以下の回数で起こる確率を示します。

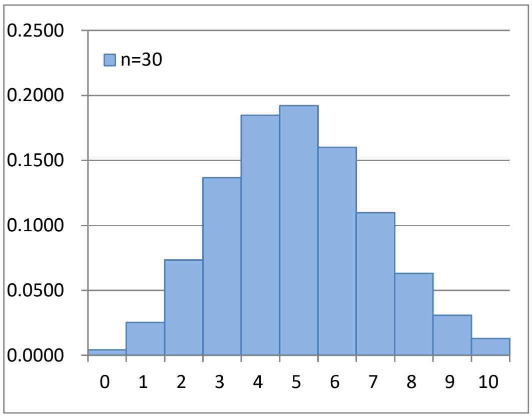
二項分布の分散(V)：$V(X) = n * V(X_i) = n * e * (1 - e)$

下左表は二項分布の試行数(N)，確率(P)，平均(M)，分散(V)を示します。平均は N*P，分散は N*P*(1-P)になります。下右表は試行回数を 3, 5, 10, 20, 30 にしたときの，それぞれの確率分布を示します。下図は，それらを結んだ平滑線です。試行数(N)が多くなるにつれ，左右対称の釣鐘状の分布（正規分布）に近づくことがわかります。

N	10
P	0.1667
M	1.6667
V	1.3889

x↓: n→	n=3	n=5	n=10	n=20	n=30
0	0.5787	0.4019	0.1615	0.0261	0.0042
1	0.3472	0.4019	0.3230	0.1043	0.0253
2	0.0694	0.1608	0.2907	0.1982	0.0733
3	0.0046	0.0322	0.1550	0.2379	0.1368
4		0.0032	0.0543	0.2022	0.1847
5		0.0001	0.0130	0.1294	0.1921
6			0.0022	0.0647	0.1601
7			0.0002	0.0259	0.1098
8			0.0000	0.0084	0.0631
9			0.0000	0.0022	0.0309
10			0.0000	0.0005	0.0130





上左図はそれぞれの試行数(N)の確率分布を比較するために曲線で示しましたが[6]，二項確率は N が離散的なので，本来ならば上右図のようにそれぞれの N の確率を間隔のない棒グラフで示すべきです[7]。

[6] Excel：折れ線を選択→右クリック→データ系列の書式設定→線のスタイル→スムージング

[7] Excel：棒グラフを選択→右クリック→データ系列の書式設定→系列のオプション→要素の間隔：なし

●プログラム

次の二項分布確率の計算式には多くの階乗(f!)があります。

$$\mathrm{Bin}(f, t, e) = {}_t\mathbf{C}_f * e\hat{\ }f * (1-e)\hat{\ }(t-f) = t! / [f!*(t-f)!]*e\hat{\ }f*(1-e)\hat{\ }(t-f)$$

階乗 x!の計算は単純ですが，x の値が大きくなると計算機はオーバーフローを起こします[8]。上式が示す計算では巨大な数を巨大な数で割った結果，最終的には[0, 1]の範囲になるので，分子と分母の計算をすべて対数を使って行い，その結果得られた数を指数とします。 (e)^f*(1 – e)^(t - f)の部分にも指数があって，e も 1-e も確率なので 1 以下の数字ですから，指数部の数が大きくなると(e)^x や(1 – e)^(t - f)が非常に小さな数になって計算ができなくなります(アンダーフロー)。対数を使えば階乗と積算が加算になり割り算が減算になるので巨大な数にはなりません。

そこで，プログラムでは上式の各項の積算・割り算・指数をそれぞれ対数の加算・減算・掛算にします。対数(log)によって指数の合計値を求め，最後にその合計値を関数 exp で自然対数の底 e の指数とします。その結果が個別の二項確率となります。次の式の対数(log)と指数(exp)の関係に注意してください。

A(k) = log(k!) とすると(階乗対数)[9],

BinD(f, t, e) = t! / [f! * (t - f)!] * e^f * (1 - e)^(t - f)
　　= exp(A(t) - A(f) - A(t- f) + f * log(e) + (t-f) * log(1-e))

次の関数 BinD では，次の関数 BinD では，生起回数 x を頻度 f とし，試行数 n を全数とします。はじめに n 個の階乗対数を計算しておき，次にそれを使って n!, f!, (t-f)!を含む対数を指数に変換しています。

```
BinD=function(f,t,e){ # f:出現数,t:標本数,e:期待確率
   A=NULL; for(i in 1:t) A[i]=ifelse(i==1,log(i),A[i-1]+log(i)) #階乗対数
   if(f==0) exp(t*log(1-e))
   else if(f==t) exp(t*log(e))
   else exp(A[t]-A[f]-A[t-f]+f*log(e)+(t-f)*log(1-e)) #二項分確率
} #二項分布確率
```

この関数 BinD は R 関数 dbinom と同じ値を返します。

```
BinD(3,10,.5)    # 0.1171875
dbinom(3,10,.5)  # 0.1171875
```

[8] R 関数 factorial(x)で実験すると x=170 が限界でした。
[9] log(k!) = log(1*2*…*k) = log(1)+log(2)+…+log(k)

次の関数 BinP では，はじめに n 個の階乗対数を計算しておき，繰り返し演算(for)の中で，これらの対数を使って，i に従って次々に二項分布確率を計算し，下側累積確率または上側累積確率を足し上げていきます。

```
BinP=function(f,t,e){ #f:出現数,t:標本数,e:期待確率
  r=0; for(i in 0:f) r=r+BinD(i,t,e); r
} #二項分布確率 (累積)
```

この関数 BinP は R 関数 pbinom と同じ値を返します。

```
BinP(3,10,.5)     # 0.171875
pbinom(3,10,.5) # 0.171875
```

このように, BinD と R 関数 dbinom が同等であり, BinP と R 関数 pbinom が同等であることを確認したので簡単化のため今後は R 関数 pbinom を使用します。こうして，たしかに R 関数はブラックボックスになりますが，納得して使用できます。

●順列と組み合わせ

異なる n 個のものから r 個を選んで一列に並べるとき，その並べ方の数は「順列」(permutation)と呼ばれます。たとえば，A, B, C, D, E という 5 個の文字から 3 個を選んで一列にする並べ方の数は

$$5 * 4 * 3 = 60$$

になります。この理由は，最初の文字が選ばれる場合の数(可能な数)は 5 個であり(A, B, C, D, E の中の 1 個)，2 番目の文字が選ばれる場合の数は可能性が 1 個減って(すでに 1 個は 1 番目の文字として使われているため) 5-1=4 個になり, 3 番目の文字が選ばれる場合の数は可能性が更に 1 個減って 4-1=3 個になるので，これらの積(5 * 4 * 3 個)が「異なる 5 個の文字の中から 3 個の文字を選んで一列に並べる場合の数(順列)」になります。

これを一般化して「n 個の文字の中から r 個の文字を選んで一列に並べる場合の数(順列)」nPr は[10]

$$\begin{aligned} nPr &= n * (n-1) * (n-2) * \ldots * (n-r+1) \\ &= [n * (n-1) * (n-2) * \ldots * (n-r+1) * (n-r) * (n-r-1) * \ldots * 2 * 1] \\ &\quad / [(n-r) * (n-r-1) * \ldots * 2 * 1] \end{aligned}$$

[10] 順列の数を求めるとき，n から始まって n-r+1 までの数をすべて掛け合わせるのは，それに続く n-r, n-r-1…は考慮しないためです。この例では 5-3=2 以下 2, 1 は考慮しません。しかし，公式を簡略に(エレガントに)示すために，はじめにすべての階乗 n!を計算し，その後で(n-r)!で割って考慮から外します。

$= n! / (n-r)!$

「順列」の数は「文字の並べかたの数」，つまり選んだ文字を一列に並べる，という条件をつけた場合の数を示しますが，「組み合わせ」(combination)ではその条件を外して「文字の選びかたの数」を示します。つまり，選ばれた 3 個の文字の並べ方の数，たとえば A, B, C の並べ方は ABC, ACB, BAC, ACA, CAB, CBA の 6 通りをすべて同一とします。この 3 個の文字の並べ方の数は 3 + 2 * 1 = 6 個です。

そして，A-B-C だけでなく，A-B-D, A-B-E, ..., C-D-E というすべての 3 文字についてそれぞれ 6 個の並べ方があるので，それらを全部同一化するために(つまり 6 個の並べかたをまとめた 1 個の選びかたとして)，先の順列の数 60 を 3 個の文字の並べ方の数 6 で割った数(60/6=10)が「異なる 5 個の文字の中から 3 個の文字を選ぶ場合の数(組み合わせ)」$_5C_3$ になります。

一般化すれば「異なる n 個の文字の中から r 個の文字を選ぶ場合の数(組み合わせ)」(nCr)は

$$nCr = nPr / r! = n! / [r! * (n-r)!]$$

組み合わせの数を求めるには R 関数 choose を使います。順列の数を求めるには choose 関数が返す値に factorial 関数で階乗を掛けます(nPr=nCr*r!)。

```
choose(5,3) # combination 10
choose(n,r)*factorial(r) # permutation 60
```

●平均と分散

データの中心を示す**平均**と，データの散らばり具合を示す**分散**は数値データを統計的に扱うときに重要な指標になります。このことは頻度分布のデータだけでなく, 確率分布のデータでも同じです。このセクションでは，頻度分布の平均・分散から出発して確率分布の平均・分散を理解し，その重要な性質を確認します。

はじめに次のような簡単な**数値分布**の平均と分散を求めます。

d	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
x	1	1	5	5	5	3	3	3	3	3	4	4	4	4	6

この平均(m)は

$$m = (\Sigma_i x_i) / n$$
$$= (1 + 1 + 5 + ... + 6) / 15 = 54 / 15 = 3.6$$

```
A=c(1,1,5,5,5,3,3,3,3,3,4,4,4,4,6)
sum(A) # 54
```

```
sum(A)/length(A) # 3.6
mean(A) # 3.6
```

分散(v)は

$$v = \Sigma_i (x_i - m)^2 / n$$
$$= (1 - 3.6)^2 + (1 - 3.6)^2 + (5 - 3.6)^2 + \ldots + (6 - 3.6)^2] / 15$$
$$= 1.844$$

```
Varp(A) # 1.84
```

```
varp=function(A) var(A)*(length(A)-1)/length(A) # Population variance
Varp=function(A) sum((A-mean(A))^2)/length(A) # Population variance
```

なお，次の**分散の別式**もよく使われます。

$$v = \Sigma_i x_i^2 / n - m^2$$

証明：

$$v = \Sigma_i (x_i - m)^2 / n$$
$$= \Sigma_i (x_i^2 - 2 m x_i + m^2) / n$$
$$= (\Sigma_i x_i^2 - \Sigma_i 2 m x_i + \Sigma_i m^2) / n$$
$$= (\Sigma_i x_i^2 - 2 m \Sigma_i x_i + \Sigma_i m^2) / n$$
$$= \Sigma_i x_i^2 / n - 2 m \Sigma_i x_i / n + \Sigma_i m^2 / n$$
$$= \Sigma_i x_i^2 / n - 2 m^2 + \Sigma_i m^2 / n$$
$$= \Sigma_i x_i^2 / n - 2 m^2 + n m^2 / n$$
$$= \Sigma_i x_i^2 / n - 2 m^2 + m^2$$
$$= \Sigma_i x_i^2 / n - m^2$$

この別式 $v = \Sigma_i x_i^2 / n - m^2$ は「分散＝2 乗の平均－平均の 2 乗」であることを示しています。

次に，上の数値ベクトル(A)から，それぞれのデータ(x)に頻度(f)があることを示す度数分布表を作成し，データの平均(m)と偏差(v)を求めます。

data	1	2	3	4	5	和
x	1	5	3	4	6	19
f	2	3	5	4	1	15

$$m = (\Sigma(i)\ x(i) * f(i)) / \Sigma(i)\ f(i)$$
$$= [(1 * 2) + (5 * 3) + (3 * 5) + (4 * 4) + (6 * 1) / 15$$
$$= (2 + 15 + 15 + 16 ＋ 6) / 15$$
$$= 54 / 15 = 3.6$$

$$v = [\Sigma(i)\ (x(i)-m) * f(i)] / \Sigma(i)\ f(i)$$
$$= ((1-3.6)^\wedge 2*2+(5-3.6)^\wedge 2*3+(3-3.6)^\wedge 2*5+(4-3.6)^\wedge 2*4+(6-3.6)^\wedge 2*1) / 15$$

= 1.84

```
A=c(1,1,5,5,5,3,3,3,3,3,4,4,4,4,6)
mean(A) # 3.6
varp(A) # 1.84
D=FreqDist(A); D        ←Function
   Data Freq
 1    1    2
 2    5    3
 3    3    5
 4    4    4
 5    6    1
FreqDistMean(D) # 3.6  →Function
FreqDistVar(D) # 1.84  →Function
FreqDistNum(D) # 15  →Function
```

```
FreqDist=function(A){
   Uq=unique(A); Fq=sapply(Uq, function(x) sum(A==x))
   data.frame(Data=Uq, Freq=Fq)
} # Frequency distribution (A: array)

FreqDistMean=function(D,v=1,f=2) sum(D[,v]*D[,f])/sum(D[,f])
# Mean in frequency distribution (v:value, f:frequency)

FreqDistVar=function(D,v=1,f=2) sum((D[,v]-m)^2*D[,f])/sum(D[,f])
# Variance in frequency distribution (v:value, f:frequency)

FreqDistNum=function(D,f=2) sum(D[,f])
# Number-of-data-in-frequency-distribution (f:frequency)
```

さらに，次は同じデータの確率分布表です。

d	1	2	3	4	5	和
x	1	5	3	4	6	19
p	2/15	3/15	5/15	4/15	1/15	1

m = Σ (i) [x(i) * p(i)]
= [(1 * 2 / 15) + (5 * 3 / 15) + (3 * 5 / 15) + (4 * 4 / 15) + (6 * 1 / 15)
= 3.6

v = Σ (i) [x(i) - m]^2 * p(i)
= [(1 - 3.6)^2 * 2 / 15 + (5 - 3.6)^2 * 3 / 15 + (3 - 3.6)^2 * 5 / 15
+ (4 - 3.6)^2 * 4 / 15 + (6 - 3.6)^2 * 1] / 15

= 1.84

```
D1[,2]=D1[,2]/sum(D1[,2]); D1
   Data       Freq
 1    1 0.13333333
 2    5 0.20000000
 3    3 0.33333333
 4    4 0.26666667
 5    6 0.06666667
sum(D1[,1]*D1[,2]) # 3.6
sum((D1[,1]-m)^2 * D1[,2]) # 1.84
```

このように，同じ原データは度数分布でも確率分布でも平均と分散が同じ結果になることを確認しました。平均(m)は期待値(expectation, expected value: E)とも呼ばれます。

```
m = E(X) = Σ(i) [x(i) * p(i)]
v = V(X) = Σ(i) {[x(i) - m]^2 * p(i)}
```

●二項分布確率の乱数実験

理論的に求められた二項分布個別確率と二項分布累積確率が実際に多数のケースで実現されることを乱数を使った実験で確かめます。

はじめに二項分布個別確率(dbinom)を見ます。次は R 関数が返す二項分布個別確率です(f=3, t=10, e=0.5)。

```
dbinom(3,10,.5)   #0.1171875
```

rbinom 関数は個数=100，サイズ=10，期待確率=.5 を受けて，次のベクトルを返します。目的の 3 にマークをつけました(f=3)。

```
set.seed(1); rbinom(100,10,.5)
[1] 4 4 5 7 4 7 7 6 6 3 4 4 6 5 6 5 6 9 5 6 7 4 6 3 4 5 2 5 7 4 5
[32] 5 5 4 6 6 6 3 6 5 6 6 6 5 5 6 2 5 6 6 5 7 5 4 3 3 4 5 6 5 7 4
[63] 5 4 6 4 5 6 3 7 4 7 4 4 5 7 7 5 6 8 5 6 5 4 6 4 6 3 4 3 4 3 6
[94] 7 6 6 5 5 6 5
```

この実験では，次のようにして実際に 0:9 (0, 1, 2, ..., 9) の中で 3 が現れる確率を求めます。rbinom 関数で 100 万個(10^6)の乱数を生成します。sum(R==3)は生成された数字ベクトル(R)の中で 3 である数字の数を返します。

```
set.seed(1); R=rbinom(10^6,10,.5); sum(R==3)/10^6 # 0.117163
```

このように，実験して求めた実際的個別確率(=0.117163)は先に R 関数で求めた理論的個別確率(=0.1171875)に近似しています。

累積確率でも，同様の実験をします。ここでは数字ベクトル(R)の中の 0, 1, 2, 3 の数を求めるために，sum は sum(R<=3)とします。

```
pbinom(3,10,.3) #0.6496107
set.seed(1); R=rbinom(10^6,10,.3); sum(R<=3)/10^6 #0.649448
```

ここでも, 実験して求めた実際的累積確率(=0. 649448)は先に R 関数で求めた理論的累積確率(=0.6496107)に近似しています。

■スペイン初等教育学童の使用語彙調査

次はスペイン初等教育学童の使用語彙調査(Fernando Justicia 1995, 3402 名，20 分の自由作文)の結果から語彙の音素数(N.fon)と年齢(E.1:6-7 歳，E.2:8-10 歳，E.3:11-13 歳)を集計したものです。

```
D=Ip(); D # Excel N.fon

   N.fon  E.1   E.2   E.3
1      1    0     0    89
2      2 1585  4878  6862
3      3 3795 14492 18423
4      4 9057 30146 31983
5      5 9101 35337 42843
6      6 6250 22856 25619
7      7 3072 13435 17931
8      8 1576  8124 12019
9      9  741  4611  6393
10    10  449  1964  3507
11    11   63   582  1063
12    12    7   111   464
13    13    3   107   331
14    14    0    46    99
15    15    2    11    37
16    16    0     7    23
17    17    0     0     3
18    18    0     3     2
```

ここで高学年の児童が音素数が多い語彙を使用することが予測されますが，次にその様子を確認します。

```
for(i in 1:3){
M[i,1]=FreqDistMean(D,1,i+1)
M[i,2]=sqrt(FreqDistVar(D,1,i+1))
M[i,3]=FreqDistNum(D,i+1)
```

```
}
rownames(M)=V('E.1,E.2,E.3');colnames(M)=V('Mean,Sd,N'); M

         Mean       Sd      N
 E.1 5.039215 2.189849  35701
 E.2 5.251357 2.431403 136710
 E.3 5.385972 2.643337 167691
```

この結果を見ると，確かに高学年になるほど音素数が僅かながら多い語彙を使用していることがわかります(Mean)。また，学年が上がるにつれて，使用語数(N)が大きく上昇しています。

次に，特に 11 音素以上の語数を比較します。

```
D=D[11:18,]; D
    N.fon E.1 E.2  E.3
 11    11  63 582 1063
 12    12   7 111  464
 13    13   3 107  331
 14    14   0  46   99
 15    15   2  11   37
 16    16   0   7   23
 17    17   0   0    3
 18    18   0   3    2

M=matrix(0,3,3)
> for(i in 1:3){
+     M[i,1]=FreqDistMean(D,1,i+1)
+     M[i,2]=sqrt(FreqDistVar(D,1,i+1))
+     M[i,3]=FreqDistNum(D,i+1)
+}
rownames(M)=V('E.1,E.2,E.3'); colnames(M)=V('Mean,Sd,N');M

         Mean       Sd    N
 E.1 11.28000 7.719067   75
 E.2 11.64937 8.126758  867
 E.3 11.84965 8.326994 2022
```

ここでも平均音素数(Mean)の差は小さいのですが，多音節語の数(N)は高学年で圧倒的に多くなります(75, 867, 2022)。

参考：Fernando Justicia. 1995. *El desarrollo del vocabulario. Diccionario de frecuencias*. Universidad de Granada.

3.2. 安全率

たとえば 10 個の問（正誤問題，正解の期待確率は 0.5: 50%）に 7 回正解

した成績(70%)よりも 100 個の問に 65 回正解した成績(65%)のほうが安全(重要)であり評価は高い，と考えられます。このような「安全（重要）である程度」を確率で示した数値を「安全率」(Security: S)と呼びます。

安全率(S)を計算するために二項分布確率を使います。次の表は試行数(n) = 1，期待確率(e) = 0.5 の事象, たとえばコイン投げについて, コインの「表」(おもて)が出現する回数(x = 0, 1)ごとの出現確率(occurrent probability: OP)，その累積確率(cumulative probability: CP)，安全率(S)の分布を示します。

(x, n:1, e:0.5)	場合	OP(x)	CP(x)	S(x)
x = 0	裏	1/2=0.5	0.5	0
x = 1	表	1/2=0.5	1	0.5

上表の出現確率(OP)はコインを 1 回投げて，「表」(おもて)にならないとき(x=0)の出現確率 OP(0)が 1/2 = 0.5 であり(OP(0)=0.5)，「表」になるときの出現確率 OP(1)が 1/2 = 0.5 であることを示します(OP(1)=0.5)。累積確率(CP)は, 出現数が 0 の場合(x=0)は出現確率(OP)と同じですが(CP(0) = 0.5)，出現数が 1 の場合(x=1)は，出現数が 0 の場合の出現確率(0.5)と出現数が 1 の場合の出現確率(0.5)を足した確率(1.0)になります(CP(1)=1)。

安全率(S)は該当するコインの「表」(おもて)の出現数 x の 1 つ前の出現数(x-1)までの累積確率(CP)に相当します[11]。

```
x=1; n=1; e=1/2; pbinom(x-1,n,e) # 0.5
```

出現数が 0 の場合は, その 1 つ前(-1)の累積確率(OP)は存在しないので，その安全率(S)を 0 とします(S(0)=0)。コインの表の出現数 1 の場合の安全率 S(1)は，出現数 x=0 の累積確率 CP(0)=0.5 になります(S(1)=CP(0)=0.5)。このように出現数 x の安全率 S(x)を直前の出現数(x - 1)の累積確率 CP(x-1)とする理由は，該当する出現数未満のすべての場合の確率の和，すなわち累積確率(CP)が該当する出現数の安全率を保証する確率，すなわち安全率であると考えるためです。コインを 1 度投げてそれが表であったとしても，それを保証する安全率は 0.5 に過ぎません。実際に出現数 0 にも出現確率 OP(0)として 0.5 の確率が存在するからです。この場合，累積確率 S(1)=CP(1-1)=CP(0)=0.5 が安全率です。

次の表は同じコイン投げの試行数を 2 回とした場合のそれぞれのコインの「表」(おもて)の出現数(x=0, 1, 2)の確率分布を示します。

S(x, n:2, e:0.5)	場合	OP(x)	CP(x)	S(x)
x = 0	裏裏	1/4=0.25	0.25	0
x = 1	裏表，表裏	2/4=0.50	0.75	0.25
x = 2	表表	1/4=0.25	1.00	0.75

---

[11] R 関数 pbinom の引数は x: 出現数，n: 試行数，e: 期待確率です。

今回もコインの表(おもて)が出ることの確率が 0.5 であることは同じです。出現確率(OP)の列は，表(おもて)が 0 回出現する出現確率 OP(0)が 0.25 (裏・裏)，表が 1 回出現する出現確率 OP(1)が 0.5(表・裏，裏・表)，そして表が 2 回出現する出現確率 OP(2)が 0.25（表・表）であることを示しています。CP の列はその累積確率を示し，S の列は安全率を示します。先のコイン投げ 1 回のケースで最後の安全率 S(1)が 0.5 (50%)であったのに対し，この場合の最後の安全率 S(2)は 0.75 (75%)に上昇しました。このことは 75%の安全率があり，同時に 25%の危険率(Risk: R)もあることを示しています。安全率(S)と危険率(R)の関係は[12]

安全率(S) + 危険率(R) = 1

この安全率 S(2) = 0.75 は，表の出現数 x=2 の出現確率 OP(2)の 1 の補数を示します。

S(2, 2, 0.5) = 1 - OP(2) = 1 - 0.25 = 0.75　　←上表

同様にして表の出現数 x=1 の安全率 S(1)は

S(1, 2, 0.5) = 1 - [OP(1) + OP(2)] = 1 - (0.5 + 0.25) = 0.25 ←上表

次の表で試行数 n=3 の場合も見ておきましょう。

S(x, n:3, e:0.5)	場合	OP(x)	CP(x)	S(x)
x = 0	裏裏裏	1/8=.125	.125	0
x = 1	表裏裏，裏表裏，裏裏表	3/8=.375	.500	.125
x = 2	表表裏，表裏表，裏表表	3/8=.375	.875	.500
x = 3	表表表	1/8=.125	1.000	.875

ここでコインの表の出現数 x=3 のときに，安全率 S(3)=.875 になりました。安全率が.875 であるとき，有意でない出現数が含まれる確率（危険率）が.125 (12.5%)もあることを示しています(1 - .875 = .125)。安全率は少なくとも 95%，理想的にはさらに 99%とし，危険率を 5%，理想的にはさらに 1%に抑えなければ，その出現数の重要性・有意性を正しく評価できません。

今度は試行数を 10 まで拡大しましょう。

S(x, n:10, e:0.5)	OP(x)	CP(x)	S(x)
x = 0	.001	.001	.000
x = 1	.010	.011	.001
x = 2	.044	.055	.011
x = 3	.117	.172	.055

[12]「危険率」(R)は一般に「P 値」と呼ばれています。

x = 4	.205	.377	.172
x = 5	.246	.623	.377
x = 6	.205	.828	.623
x = 7	.117	.945	.828
x = 8	.044	.989	.945
x = 9	.010	.999	.989
x = 10	.001	1.000	.999

このときコインの表の出現数(x)が 9 のとき，安全率(S)が 0.95 を超え(.989)，その出現数が 10 のとき安全率が 0.99 を超えました(.999)。よって 10 回中 9 回の出現数があれば，この頻度を 0.95 以上の安全率がある，として提示することができ，10 回中 10 回の出現数であれば，その安全率は 0.99 を超えることになります。実際に 10 回コインを投げて 9 回表が出現したならば，これは 9 に満たない出現数[0, 1, 2..., 8]の確率の和が 98.9%になるので，かなり有意（重要）な数値であると言えます。つまり，私たちは期待確率が 0.5 であることを 10 回試行して 9 回出現したことの安全率は 98.9%である，と理解します。

「有意である」という意味は，この数値以上になる，つまり例外となる確率(危険率)が 1.1%に過ぎないからです(.010 + .001 = .011)。これを先に見た，1 回の試行中 1 回の出現(安全率 s=0.500)，2 回の試行中 2 回の出現(安全率 s=0.750), 3 回の試行中 3 回の出現(安全率 s=0.875)と比べて見ると，その安全率の高さがわかります。一方，たとえば 10 回の試行で 5 回出現したことの安全率は 0.377 に過ぎません。6 回出現しても 0.623 で，0.95 や 0.99 には遠く及びません。これらの頻度は偶然でも十分に出現しうるのです[13]。

以下では基本的に安全率の基準を厳しく設定して 0.99 を前提にします。安全率の基準を 0.95 にすると比較的小さな数値でも安全率を超えてしまいます。逆にたとえば 0.999 のように非常に高い安全率を基準に設定すると，ほとんど可能性のない場合までも含めてしまうので実用的ではありません。

1 2 3 4 5 6 7 8 9 10

[13] ここで問題とするのは，個別の生起確率(OP)ではなく，直前の場合までを積み上げた累積確率(CP)です。個別の生起確率(OP)を問題にすると，たとえば 10 回コインを投げて，表が 5 回出現し裏が 5 回出現する確率は.246 でしかありません。そこで，0 回から 4 回までの累積確率ならば.377，0 回から 5 回までの累積確率ならば.623 であるので，その両者の間に，期待される確率である.500 が存在していることになります。また，累積確率を該当する場合までの累積ではなく，その直前までの累積とする理由は，その累積確率の補数（1-累積確率)を「危険率」として使うためです。たとえばコインを 10 回投げるとき，表が 9 回出るような累積確率(信頼係数)は.989 で，危険率は.011 です。1.1%の確率で起こることは稀なので，危険率が微小であると考えられます（→**二項検定**）。

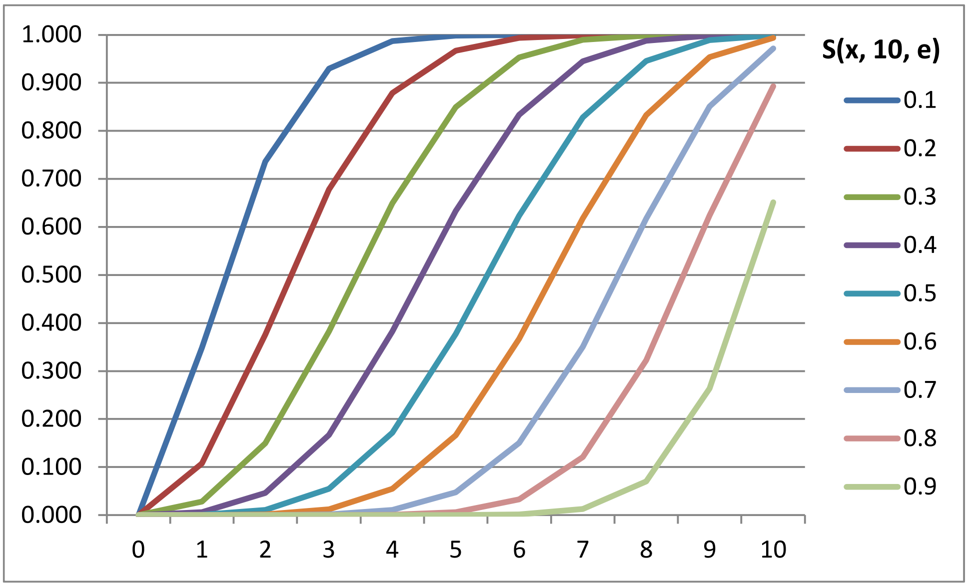
次の表は，上のように，1, 2, …, 10 という数字が書かれた 10 枚のカード群の中から，目を閉じて 1 枚のカードを引き，それが"1"であるときの場合(期待確率(e) = 1/10 = 0.1)の安全率(S)を示します。それを 10 回繰り返して，"1"が出た回数を x で示します。

S(x, n:10, e:0.1)	OP(x)	CP(x)	S(x)
x = 0	.349	.349	.000
x = 1	.387	.736	.349
x = 2	.194	.930	.736
x = 3	.057	.987	.930
x = 4	.011	.998	.987
x = 5	.001	1.000	.998
x = 6	.000	1.000	1.000
x = 7	.000	1.000	1.000
x = 8	.000	1.000	1.000
x = 9	.000	1.000	1.000
x = 10	.000	1.000	1.000

たとえば x:5, n:10, e:0.1 のとき安全率 S(5) = .998 になる，ということは x = 0, 1, 2, 3, 4 の累積出現確率が 0.998 になる，ということを意味します。ここで x=4 と x=5 の間で安全率が.99 を超えたことになります。よって，安全率 0.99 を境にすれば 0 から 4 までの範囲がほとんど確実に(99%)出現し，5 以上はほとんど出現しない(1%以下)ことになります。

次は横軸を期待確率(e: 0.1., 0.2., …, 0.9)，縦軸を出現数(x: 1, 2, …, 10)とし，それぞれのセルに対応する安全率(s)を計算した表です。

BinS(x, 10, e)	e = 0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
x = 0	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
1	0.349	0.107	0.028	0.006	0.001	0.000	0.000	0.000	0.000
2	0.736	0.376	0.149	0.046	0.011	0.002	0.000	0.000	0.000
3	0.930	0.678	0.383	0.167	0.055	0.012	0.002	0.000	0.000
4	0.987	0.879	0.650	0.382	0.172	0.055	0.011	0.001	0.000
5	0.998	0.967	0.850	0.633	0.377	0.166	0.047	0.006	0.000
6	1.000	0.994	0.953	0.834	0.623	0.367	0.150	0.033	0.002
7	1.000	0.999	0.989	0.945	0.828	0.618	0.350	0.121	0.013
8	1.000	1.000	0.998	0.988	0.945	0.833	0.617	0.322	0.070
9	1.000	1.000	1.000	0.998	0.989	0.954	0.851	0.624	0.264
10	1.000	1.000	1.000	1.000	0.999	0.994	0.972	0.893	0.651



たとえば，セル x=1:e=0.1 の.349 という安全率は次の関数 BinS を使います(→「プログラム」)。

s ← BinS(x,n,e)　　s:安全率; x:出現数; n:試行数; e:期待確率
.349 ← BinS(1, 10, 0.1)

これは期待確率 0.1 で起きることを 10 回試行して，1 回出現する安全率が.349 であることを示します。同様にして 2 回出現する安全率は

.736 ← BinS(2,10,0.1)

次の表は試行数(n)を 100 として，それぞれの出現数(x)を 10 ごとに上昇させた場合の安全率を示します。

S(x, n:100, e)	e = 0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
x = 0	.000	.000	.000	.000	.000	.000	.000	.000	.000
10	.451	.002	.000	.000	.000	.000	.000	.000	.000
20	.998	.460	.009	.000	.000	.000	.000	.000	.000
30	1.000	.989	.462	.015	.000	.000	.000	.000	.000
40	1.000	1.000	.979	.462	.018	.000	.000	.000	.000
50	1.000	1.000	1.000	.973	.460	.017	.000	.000	.000
60	1.000	1.000	1.000	1.000	.972	.457	.012	.000	.000
70	1.000	1.000	1.000	1.000	1.000	.975	.451	.006	.000
80	1.000	1.000	1.000	1.000	1.000	1.000	.984	.441	.001
90	1.000	1.000	1.000	1.000	1.000	1.000	1.000	.994	.417
100	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000

上の表で期待確率(e) = 0.1 の安全率 0.99 を超える出現数(x)は 20 になっ

ていますが，1 桁の数値まで正確に求めると 19 になります。このように数値の規模が 10 倍になったとしても，0.99 の点が 5 から 50 に変わるのではなく，19 という 50 よりもかなり小さな値になります。

参考サイト：
https://unit.aist.go.jp/mcml/rg-orgp/uncertainty_lecture/confidence.html
https://okumuralab.org/~okumura/stat/tests_and_CI.html

●プログラム: 安全率・危険率・P 値

次が安全率を求める関数 BinS です。安全率は pbinom(x-1,n,e)を使って求めることができますが，x=0 のときは計算できないので，そのときは強制的にゼロ(0)とします。危険率を求める関数 BinR は安全率の補数を返します。危険率は一般の統計学では P 値(P-value)と呼ばれます。一般に数値の大きさの安全率を調べることが多いのですが，逆に「数値の小ささ」の安全率を調べるときには sel=1 とします。

```
BinS=function(f,t,e=0.5,sel='g'){
   #f:freq.,t:total,e:expected probability,sel='l':less,'g':greater,'t':two-sided
   if(sel=='l') return(1-pbinom(f,t,e))
   if(sel=='g') return(ifelse(f==0,0,pbinom(f-1,t,e)))
   if(sel=='t') return(1-2*min(pbinom(f,t,e),1-pbinom(f-1,t,e)))
} # Security

BinR=function(f,t,e=.5,sel='g'){
   #f:freq.,t:total,e:expected probability,sel='l':less,'g':greater,'t':two-sided
   1-BinS(f,t,e,sel)
} # Risk, P-value
```

●安全率の乱数実験

次の 2 つの場合の理論的安全率を乱数を使った実験で確かめます。

```
BinS(5, 10, 0.15)         # 0.9901259 … (a)
BinS(5, 10, 0.50)         # 0.3769531 … (b)
```

次は乱数実験の結果です。

```
set.seed(1); R=rbinom(10^6, 10, 0.15); sum(R<5)/10^6 # 0.990252 …(1)
set.seed(1); R=rbinom(10^6, 10, 0.50); sum(R<5)/10^6 # 0.37679    …(2)
```

上の実験(1)で，期待確率 0.15 で 0:10 の乱数 100 万個(10^6)を発生させた

ベクトル R の安全率 P(x=0:4)(「x が 0~4 になる確率」)が確かに 99%になることが確かめられました。一方，実験(2)では，確率を 0.50 にすると，安全率が 38%に過ぎなくなることがわかります。これは数式で導かれた安全率とそれぞれほぼ一致しています((a), (b))。

●安全率のリスト

```
cl=.95; x=0; e=0.5; m=10^x
M=matrix(0,10,10); rownames(M)=colnames(M)=(1:10)*m
for(i in 1:10){
   for(j in 1:i) M[j,i]=Round(BinS(j*m,i*m,e,'t'),3)
}; M[M==0]=''; M=gsub('0¥¥.','.',M); noquote(M); Copy(M)
```

安全率のリスト **(右側検定)**

*	t:1	2	3	4	5	6	7	8	9	10
x:1	.500	.250	.125	.063	.031	.016	.008	.004	.002	.001
2		.750	.500	.313	.188	.109	.063	.035	.020	.011
3			.875	.688	.500	.344	.227	.145	.090	.055
4				.938	.813	.656	.500	.363	.254	.172
5					.969	.891	.773	.637	.500	.377
6						.984	.938	.855	.746	.623
7							.992	.965	.910	.828
8								.996	.980	.945
9									.998	.989
10										.999

*	t:10	20	30	40	50	60	70	80	90	100
f:10	.999	.412	.021	.000	.000	.000	.000	.000	.000	.000
20		1.000	.951	.437	.059	.003	.000	.000	.000	.000
30			1.000	.999	.899	.449	.094	.009	.000	.000
40				1.000	1.000	.993	.859	.456	.123	.018
50					1.000	1.000	1.000	.984	.829	.460
60						1.000	1.000	1.000	.999	.972
70							1.000	1.000	1.000	1.000
80								1.000	1.000	1.000
90									1.000	1.000
100										1.000

*	t:100	200	300	400	500	600	700	800	900	1000
f:100	1.000	.472	.000	.000	.000	.000	.000	.000	.000	.000
200		1.000	1.000	.480	.000	.000	.000	.000	.000	.000
300			1.000	1.000	1.000	.484	.000	.000	.000	.000
400				1.000	1.000	1.000	1.000	.486	.000	.000
500					1.000	1.000	1.000	1.000	1.000	.487
600						1.000	1.000	1.000	1.000	1.000
700							1.000	1.000	1.000	1.000
800								1.000	1.000	1.000
900									1.000	1.000
1000										1.000

Security, right-side test, f:frequency, t:total

(2) 安全率のリスト (両側検定)

*	t:1	2	3	4	5	6	7	8	9	10
f:1	.000	.000	.000	.375	.625	.781	.875	.930	.961	.979
2		.500	.000	.000	.000	.313	.547	.711	.820	.891
3			.750	.375	.000	.000	.000	.273	.492	.656
4				.875	.625	.313	.000	.000	.000	.246
5					.938	.781	.547	.273	.000	.000
6						.969	.875	.711	.492	.246
7							.984	.930	.820	.656
8								.992	.961	.891
9									.996	.979
10										.998

*	t:10	20	30	40	50	60	70	80	90	100
f:10	.998	.000	.901	.998	1.000	1.000	1.000	1.000	1.000	1.000
20		1.000	.901	.000	.797	.987	1.000	1.000	1.000	1.000
30			1.000	.998	.797	.000	.718	.967	.998	1.000
40				1.000	1.000	.987	.718	.000	.657	.943
50					1.000	1.000	1.000	.967	.657	.000
60						1.000	1.000	1.000	.998	.943
70							1.000	1.000	1.000	1.000
80								1.000	1.000	1.000
90									1.000	1.000
100										1.000

*	t:100	200	300	400	500	600	700	800	900	1000
f:100	1.000	.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
200		1.000	1.000	.000	1.000	1.000	1.000	1.000	1.000	1.000
300			1.000	1.000	1.000	.000	1.000	1.000	1.000	1.000
400				1.000	1.000	1.000	1.000	.000	.999	1.000
500					1.000	1.000	1.000	1.000	.999	.000
600						1.000	1.000	1.000	1.000	1.000
700							1.000	1.000	1.000	1.000
800								1.000	1.000	1.000
900									1.000	1.000
1000										1.000

Security, both-side test, f:frequency, t:total

■中世子音字の変異<u>, <b>, <v>の安全率

現代スペイン語の文字<b>と<v>はどちらも有声両唇音を示し，音韻的な区別はありません。中世スペイン語では唇歯摩擦音(または両唇摩擦音)と両唇閉鎖音が区別され，書き方にも<b>, <u>, <v>のバリエーションがありました。

下左図は資料中の絶対頻度を示します。このような頻度分布表から，統計的な安全率を求めます。たとえば *uoz*「声」の 1250 年(1250-1299)の頻度は 8 でバリエーション(*boz,uoz, voz*)の全体は 12 (=3+8+1) なので，12 の中の 8 という数値の安全率を求めることになります。そのときの期待確率は 1/3 とします。これは *uoz, boz, voz* という 3 種の変種がランダムに分布するならば，それぞれ 1/3 の確率になる，と期待（予想）できるからです。そこで，*uoz* の 1250 年(1250-1299)の頻度の安全率(S)は次の通りです。

```
S(uoz, 1250): pbinom(8-1, 12, 1/3, 1) = 0.981
```

AF	1200	1250	1300	1350	1400
boz	0	3	**8**	18	35
uoz	**3**	**8**	3	11	6
voz	0	1	1	**23**	**53**
Sum	3	12	12	52	94

S.	1200	1250	1300	1350	1400
boz	0.000	0.181	**0.981**	0.526	0.758
uoz	**0.963**	**0.981**	0.181	0.019	0.000
voz	0.000	0.008	0.008	**0.934**	**1.000**

安全率が 0.95 以上の値を太字にしました。これらが統計的に有意である，と判断されます($s > .95$; $p < .05$)。

3.3. 期待確率

たとえばコイン投げで「表」(おもて)が出ることの確率(「期待確率」expected probability: e と呼びます)は 1/2 = 0.5 であり，サイコロを投げてその目が「1」になる期待確率は 1/6 = 0.167 になります。しかし，実際の言語データ分析では，コインやサイコロを投げるときとは違って，初めから期待確率がわかっていないことが普通です。そこで，既知の出現数(f)，全数(t)，安全率(s)から未知の期待確率(e)を求める関数 BinE を作成します(→プログラム)。

e = BinE(f,t,s)

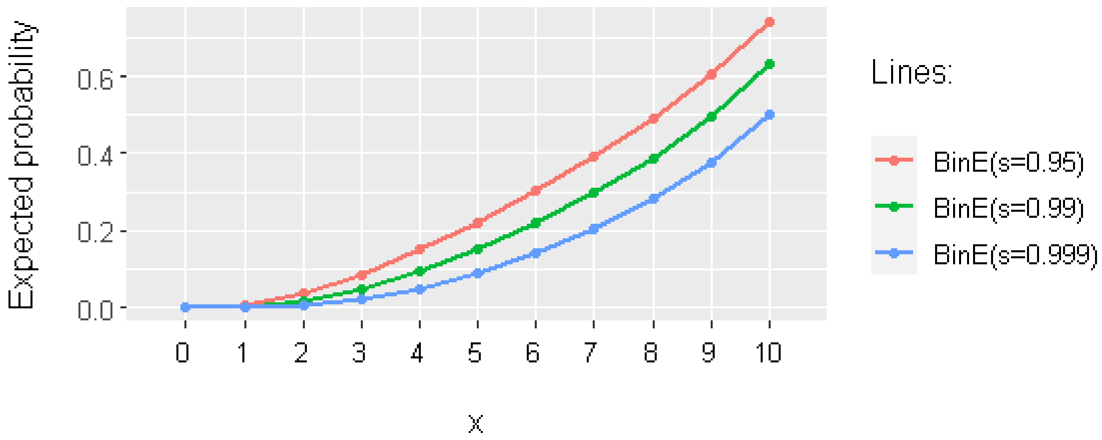
期待確率(e)は，t 回試行して一定の安全率(s)で f 回出現するときの前提となる事象の生起確率です。この期待確率は，たとえば標本数 10 個のデータの中で 5 個出現すれば，そのようなことが一定の安全率(たとえば 99%)で起こることが予想される確率です。これを関数 BinE で求めると，期待確率(e) = 0.2224411 が返されます。この期待確率(e)を使って，安全率(s)を

求めると確かに s=0.95 が返されます。これは先に BinE で用いた安全率(s)と一致します。

```
e=BinE(5,10,0.95); e # 0.2224411
s=BinS(5, 10, e); s # 0.95
```

期待確率は出現数(x)，標本数(n)，安全率(s)によって変わります。次の表は標本数 n を 10 回とし，出現数 x を 0, 1, 2, ..., 10 とし，安全率を 95%, 99%, 99.9%としたときのそれぞれの期待確率(e)を示します。この表を見ると出現数(x)が上がるほど期待確率(e)が全体的に上昇することがわかります。たとえば，安全率(s)を 0.99 にすると 10 回の試行(n)をして 1 回の出現(x)があれば，その事象の期待確率(e)は.001 という低い数値になりますが，出現数が 10 であれば,その事象の期待確率(e)は.631 となって上昇します。これは期待確率が小さな事象の出現数が小さく，逆に期待確率が大きな事象の出現数が大きくなる，という自明のことを示しています。確かにこのことは自明なのですが，ここでの目的はそれぞれのケースの正確な期待確率を求めることです。

```
A=0:10; D=data.frame(BinE(A,10,.95),BinE(A,10,.99),BinE(A,10,.999))
rownames(D)=A
colnames(D)=V('BinE(s=0.95),BinE(s=0.99),BinE(s=0.999)')
round(D,4)

   BinE(s=0.95) BinE(s=0.99) BinE(s=0.999)
0        0.0000       0.0000        0.0000
1        0.0051       0.0010        0.0001
2        0.0368       0.0155        0.0048
3        0.0873       0.0475        0.0210
4        0.1500       0.0932        0.0496
5        0.2224       0.1504        0.0898
6        0.3035       0.2183        0.1413
7        0.3934       0.2971        0.2046
8        0.4931       0.3883        0.2815
9        0.6058       0.4956        0.3763
10       0.7411       0.6310        0.5012
```

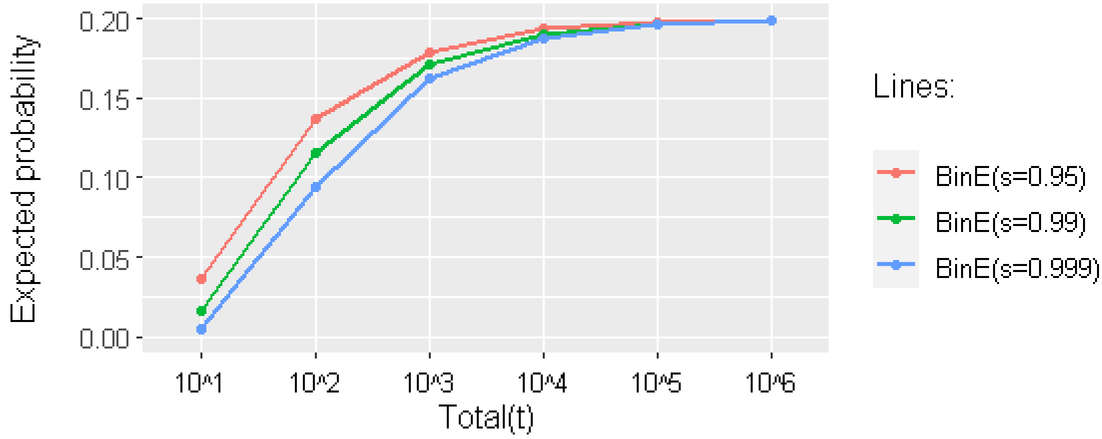


次に，安全率(s)が上がるほど(0.95, 0.99, 0.999)，逆に期待確率(e)が下がることを確認しましょう。たとえば BinE(5, 10, 0.95)は.222 になりますが，同じ事象を安全率 s=.99 で見ると期待確率 e = .150 になり，さらに安全率 s=.999 で見ると期待確率は e = .090 に下がります。

```
e=BinE(5, 10, 0.95); e #0.2224411
e=BinE(5, 10, 0.99); e #0.1504428
e=BinE(5, 10, 0.999); e #0.08981319
```

そして，試行数 n (total: t)を 10, 100, 1000, 10000, ...のように 10 倍ずつ増やして，期待確率がどのように増えるのかを見ましょう。

```
A=10^(1:6); A; B=10^(1:6)*.2; B
D=data.frame(BinE(B,A,.95),BinE(B,A,.99),BinE(B,A,.999))
rownames(D)='10^'&(1:6)
colnames(D)=V('BinE(s=0.95),BinE(s=0.99),BinE(s=0.999)')
round(D,4)

     BinE(s=0.95) BinE(s=0.99) BinE(s=0.999)
10^1       0.0368       0.0155        0.0048
10^2       0.1367       0.1156        0.0945
10^3       0.1794       0.1713        0.1625
10^4       0.1934       0.1908        0.1878
10^5       0.1979       0.1971        0.1961
10^6       0.1993       0.1991        0.1988
```



上の表と図を見ると，安全率(s)を 95%としたとき(s = 0.95), 10 回の試行で 2 回成功しても，その期待確率(e)は 0.0368 (3.6%)に過ぎず，私たちが期待する e = 0.20 (20%)の確率には遠く及んでいないことがわかります。実質的に e = 0.2 (20%)に近くなるのは試行数 n = 10,000 (10^4)で，e = 0.1934 (19.34%) になりました。n を 10,000 以上に増やしても，期待確率の変化は微増です。最終的にほとんど完全に近く 20%に達するときの n は 1,000,000 です(n=10^6; e = 0.1993)。

よって，比率の母数(標本数=n)が小さいとき(1000 に満たないとき)は，想定される期待確率に達していないので，相対頻度・正規化頻度(相対頻度*乗数)は取扱いに注意が必要です。また，相対頻度や正規化頻度を比較するとき，当然比較する比率のそれぞれの母数(n)が異なります(まさに母数が異なるので，相対頻度・正規化頻度が使われます)。そのとき，母数(n)が大きく異なると，期待確率も大きく異なります。そのような頻度や比率を適切に正しく比較するときには期待確率を用いるとよいでしょう。比較する相対頻度の母数が小さいとき(n<1000)，そして母数の大きさが大きく異なるときは(n1>>n2, n1<<n2)，単純な比率・相対頻度・正規化頻度による比較はできません。そのときに期待確率が役立ちます。逆に，比較する相対頻度の母数が大きいときや母数の大きさにあまり差がないときに，期待確率を用いてもとくに問題はありません。したがって，ほとんどの比率や割り算を使うさまざまな統計的数値に，期待確率の導入の可能性が考えられます。

●プログラム

先で見たように，出現数(x)と標本数(n)が一定であれば，一般に期待確率(e)が高くなるほど安全率(s)が低くなります。この期待確率と安全率が連続して相反する関係を利用して，二分検索法によって成功数(出現数・頻度:f), 総数(t)，信頼水準(s)から期待確率を導出するプログラム BinE を作成しました。

```
BinE.z=function(f,t,s=.95){#f:frequency,t:total,s:security
   if(f==0) return(0) else {p=10^8; mx=p; mn=0; lw=s-1/p; up=s+1/p}
   #p:precision : mx:max, mn:min,lw:lower, up:upper
   for (i in 1:1000){
      mid=(mx+mn)/2; e=mid/p; s=pbinom(f-1,t,e)
       #mid; expected prob.; security
      if(s<lw) mx=mid          #raising of mx to mid
      else if(s>up) mn=mid #lowering of mn to mid
      else break    # if s-1/p <= s <= s+1/p], break
   }; e
} # Expected binomial probability (binary search)
```

はじめに出現数 f=0 のときの期待確率を 0 とします。f > 0 のとき，この関数 BinE は先に見た安全率(s)と期待確率(e)の間の単調増加・単調減少で相反する関係を使い，一定の精度(ps)で二分探索をしながら順次期待確率を作成し[14]，その期待確率で計算した安全率の候補が指定された安全率 s と差が±1/ps の範囲内にあって実質的に同値になった時点の期待確率を返します。

一方, 次は R の二項検定のパッケージ stats::binom.test が返す信頼区間下限を使って期待確率を求める関数 BinE です。

```
BinE=function(f,t,s=.95){ #    f:frequency,t:total,s:security
   qbeta(1-s,f,t-f+1)
   # library(stats); binom.test(f,t,0,'greater',s)$conf.int[1]
   # lower-b. of greater interval of confidence
   # the smallest plausible value for the true proportion of successes
} # Expected probability (binom.test / qbeta)
```

二分検索法によって求めた期待確率(BinE.z)が R 統計パッケージ stats::binom.test (または qbeta 関数)を使って求めた期待確率(BinE)と一致することを確かめます。

```
f=3; t=10; s=.95        #f:freq.=3, t:total=10, s:security=.95
BinE.z(f,t,s); BinE(f,t,s)
[1] 0.08726443
[1] 0.08726443
```

この BinE 関数はブラックボックスですが，その機能は BinE.z と同一であることを確認しましたので，今後は BinE を使用して期待確率を計算し

[14] For ... Next を使って連続的に探索すると多くの場合多大な時間がかかります。

ます。

●期待確率のリスト

次は度数(f)と全数(t)に応じて，それぞれの場合の期待確率のリストです。このリストは関数 BinE()が使用できない環境で有用です。また，このリストによって，頻度(f)と総数(t)から見た期待確率の全体的な様子が観察できます。

```
s=0.95; m=10^2; M=matrix(0,10,10); rownames(M)=colnames(M)=(1:10)*m
#s=.95, .99; m=10^0,10^1,10^2
for(i in 1:10){
   for(j in 1:i) M[j,i]=Round(BinE(j*m,i*m,s),3)
}; M[M==0]=''; M=gsub('0¥¥.','.',M); noquote(M); Copy(M)
BinE(2*m,3*m,s) # Example
```

```
Copy=function(D,rn=T,cn=T) {#rn: row-names, cn: col-names
   D=as.data.frame(D)
   Rn=c('*',rownames(D))
   if(cn) D=rbind(colnames(D),D)
   if(rn) D=cbind(Rn,D)
   P=D[,1]
   if(ncol(D)>1) {for(i in 2:ncol(D)){P=paste(P,D[,i],sep='¥t')}}; D=P
   #insert tab
   writeClipboard(as.character(D),format=13)
} #Copy to clipboard
```

(1) 期待確率 (安全率:0.95)

s:.95	*t*:1	2	3	4	5	6	7	8	9	10
f:1	.050	.025	.017	.013	.002	.009	.007	.006	.006	.005
2		.100	.135	.098	.033	.063	.053	.046	.041	.037
3			.368	.249	.106	.153	.129	.111	.098	.087
4				.473	.222	.271	.225	.193	.169	.150
5					.398	.418	.341	.289	.251	.222
6						.607	.479	.400	.345	.304
7							.652	.529	.450	.393
8								.688	.571	.493
9									.717	.606
10										.741

s:.95	*t*:10	20	30	40	50	60	70	80	90	100
f:10	.741	.302	.193	.142	.113	.093	.080	.069	.062	.055
20		.861	.501	.361	.283	.233	.198	.172	.152	.137
30			.905	.613	.474	.387	.328	.284	.251	.225
40				.928	.684	.554	.466	.403	.355	.318
50					.942	.734	.612	.527	.463	.414
60						.951	.770	.658	.576	.513
70							.958	.797	.694	.616
80								.963	.819	.723
90									.967	.836
100										.970

s:.95	*t*:100	200	300	400	500	600	700	800	900	1000
f:100	.970	.440	.288	.215	.171	.142	.122	.106	.094	.085
200		.985	.619	.458	.363	.302	.258	.225	.200	.179
300			.990	.712	.563	.466	.397	.347	.307	.276
400				.993	.768	.634	.540	.470	.417	.374
500					.994	.806	.685	.596	.528	.474
600						.995	.834	.724	.640	.574
700							.996	.854	.754	.675
800								.996	.870	.778
900									.997	.883
1000										.997

Expected probability with Security (*s*) = 0.95

(2) 期待確率 (安全率:0.99)

s:.99	*t*:1	2	3	4	5	6	7	8	9	10
f:**1**	.010	.005	.003	.003	.002	.002	.001	.001	.001	.001
2		.100	.059	.042	.033	.027	.023	.020	.017	.016
3			.215	.141	.106	.085	.071	.061	.053	.048
4				.316	.222	.173	.142	.121	.105	.093
5					.398	.294	.236	.198	.171	.150
6						.464	.357	.293	.250	.218
7							.518	.410	.344	.297
8								.562	.456	.388
9									.599	.496
10										.631

s:.99	*t*:10	20	30	40	50	60	70	80	90	100
f:**10**	.631	.239	.151	.110	.087	.072	.061	.053	.047	.042
20		.794	.439	.312	.243	.199	.168	.146	.129	.116
30			.858	.559	.426	.346	.291	.252	.222	.198
40				.891	.637	.509	.426	.367	.322	.287
50					.912	.692	.572	.489	.428	.381
60						.926	.733	.620	.540	.479
70							.936	.764	.659	.582
80								.944	.789	.691
90									.950	.809
100										.955

s:.99	*t*:100	200	300	400	500	600	700	800	900	1000
f:**100**	.955	.416	.271	.201	.160	.133	.113	.099	.088	.079
200		.977	.600	.441	.349	.289	.247	.215	.191	.171
300			.985	.696	.547	.452	.385	.335	.297	.267
400				.989	.755	.620	.527	.458	.406	.364
500					.991	.795	.673	.584	.516	.463
600						.992	.824	.713	.629	.563
700							.993	.845	.744	.665
800								.994	.862	.769
900									.995	.876
1000										.995

Expected probability with Security (*s*) = 0.99

次は, 上の表から BinE(200,900)=0.1907 と BinE(300,900)=0.2970 を求め, 比例補完を行って, BinE(235,900)を計算した結果を示します(0.2279)。この値は関数 BinE(235,900)が返す正確な値 0.2276 に近似します

```
BinE(235,900) #0.2276
BinE(200,900) #0.1907
BinE(300,900) #0.2970
0.1907+(0.2970-0.1907)*35/100 #0.2279
```

●確率頻度

期待確率は確率の一種なので[0,1]の範囲をとります。すべて 1 以下の小数になりますが, 場合によってかなり小さな値を示します。そこで, 期待確率に適切な乗数(10^2, 10^3, ...)を掛けた数値を使用します。これを「確率頻度」(Probabilistic frequency:Pf)と呼びます。

Pf = 期待確率(e) * 乗数(m)

次が確率頻度を返す関数 Pf です。

```
Pf=function(f,t,c=0.95,m=1000,r=0){
   #f:freq,t:total,m:multiplier,c:confidence,r:round
   round(BinE(f,t,c)*m,r)
} # Probabilistic frequency
Pf(f=3,t=10) # 87 (not 300)
```

次の出力は f=0:11, t=10, m=1000 の確率頻度を示します。

```
A1=A2=A3=NULL
for(i in 0:10) {
   A1[i+1]=i; A2[i+1]=round(BinE(i,10),4); A3[i+1]=Pf(i,10)}
D=data.frame(A1,t=rep(10,11),A2,t=rep(1000,11),A3);
colnames(D)=V('f,t,e,m,Pf'); D

     f  t      e    m  Pf
 1   0 10 0.0000 1000   0
 2   1 10 0.0051 1000   5
 3   2 10 0.0368 1000  37
 4   3 10 0.0873 1000  87
 5   4 10 0.1500 1000 150
 6   5 10 0.2224 1000 222
 7   6 10 0.3035 1000 304
 8   7 10 0.3934 1000 393
 9   8 10 0.4931 1000 493
 10  9 10 0.6058 1000 606
 11 10 10 0.7411 1000 741
```

最終行(度数 f=10，全数 t=10)の期待確率(e)は.741，乗数 m=1000 の確率頻度は 741 であることを示します。このことは，10 回の試行で 10 回出現したことが 1000 回の試行で 741 回起こることが 95%の安全率をもって予想できることを示します。

3.4. 期待頻度

これまで見てきたように，頻度(f)，全数(t)，期待確率(e)，安全率(s)の 4 つの値は相互に関係し，これらの中の 3 つの値がわかれば，それらを使って残りの 1 つの値を求めることができます。安全率(s)は頻度(f)，標本数(全数)(t)，期待確率(e)を使って求め(関数: BinS)，期待確率(e)は頻度(f)，全数(t)，安全率(s)を使って求めます(関数: BinE)。そこで，次のように全数(t)，期待確率(e)，安全率(s)から期待される頻度(f)を求める関数 BinF を作成し，この関数が返す値を「期待頻度」(expected frequency)と呼ぶことにします。

```
BinF=function(t,e,s=.95){if(e==0) return(0); f=qbinom(s,t,e); IE(f<t,f+1,f)}
# Expected frequency. (t:total,e:expected probability,s:security=>f:freq)
```

期待確率(e)と期待頻度(f)は互いに逆の関係(逆関数)になることを確かめましょう。

```
e=BinE(f=3,t=10,s=.95); e #e:expected prob. 0.08726443
f=BinF(t=10,e,s=.95); f #f:expected freq. 3 (inverse function)
```

上の実験では，f:度数=3,t:全数=10,s:安全率=.95 を設定して，BinE で期待確率(e)を求め，その期待確率を使って，BinF で期待頻度を求めて，それが最初に設定した度数(=3)と一致することを確かめました。

次に，，全数(t)の変化によって期待頻度がどのように変わるかを観察します。

```
BinF(t=10^1,e=.5,s=.95) #期待頻度 9    (a1)
BinF(t=10^2,e=.5,s=.95) #期待頻度 59   (a2)
BinF(t=10^3,e=.5,s=.95) #期待頻度 527  (a3)
BinF(t=10^4,e=.5,s=.95) #期待頻度 5083 (a4)
```

最初の実験では，全数(t)が 10 のとき，期待頻度(f)が 9 以上であれば，安全率が.95 を超えることを示します(a1)。このことは，10 回のコイン投げで 9 回「表」が出なければ安全率が 95%にならないことを示しています。この全数(コイン投げの回数)が 100，1000，10000 のように増加させていくと，期待頻度は次第に全数の 50%に近づいていきます(a2, a3, a4)。

●安全頻度

期待確率(e)と期待頻度(f)を使って，標本数をたとえば 1000 とした場合に高い安全率で予想される頻度を求めることができます。初めに先の期待頻度を求める関数 BinF を使って次の実験をします。

```
e=BinE(3,10,s=.95); e     #expected prob.: 0.08726443 (a1)
f=BinF(t=10,e,s=.95); f   #expected freq.: 3          (a2)
f=BinF(t=1000,e,s=.95); f #expected freq.: 103        (a3)
s=BinS(f,t=1000,e); s     #s:security=0.9535415       (a4)
s=BinS(f-1,1000,e); s     #s:security=0.94218         (a5)
```

この実験では，初めに f=3; t=10; s=.99 で期待確率(e)を求め(e=0.08726443)，次にその期待確率を使って，全数(t)が 10 であるときの期待頻度(f)を求めます(f=3) (a1,a2)。次にその同じ期待確率(e)を使って，全数(t)が 1000 であるときの期待頻度(f)を求めます(f=103) (a3)。このように 3/10=.30 (30%)であっても，全数が 1000 になると，300 という頻度(=30%)は期待できず，BinF はかなり下方修正した 103 (103/1000=10.3%)を返します。そして, 上で求めた期待確率(e)と期待頻度(f)を使って, 安全率(security)を求めると 0.9535415 となり，信頼水準の 0.95 を超えることを確認しました(a4)[15]。しかし，この f を下回ると(f-1)，そのときの安全率は.95 を超えません(0.94218) (a5)。このことは，期待確率が e=0.08726443 であるとき, f=0:103 の全確率の累積が 95%を超えることを示しています。よって 95%の信頼を置くことができます。

度数(f)，全数(t)，基準数(b)，安全率(s)から求めた期待頻度を「安全頻度」(Secure frequency: Sf)と呼びます。次は安全頻度を求める関数 Sf です。

```
Sf=function(f,t,s=.95,b=10^3){
   #Secure frequency. f:frequency,t:total,s:security,b:base
   if(t==0) return(0); e=BinE(f,t,s); BinF(b,e,s)}
```

この関数 Sf を使って，さまざまな実験をします(安全率 s=.95)。

```
Sf(f=1,t=10)  #f:freq.=1,t:total=10 SF: 10 (b1)
SF(f=2,t=10)  #f:freq.=2,t:total=10 SF: 48 (b2)
SF(f=3,t=10)  #f:freq.=3,t:total=10 SF: 103 (b3)
SF(f=10,t=10) #f:freq.=10,t:total=10 SF: 765 (b4)
```

上の実験では, 最初に全数 10 の中で 1 回起こることは 1000 回(規定の基準数 base)の中では 100 回ではなく, せいぜい 9 回しか起きないことを示しています(b1)。当然，度数(f)が増えれば(f=2,3)，安全頻度は上昇しますが，全数(t)が小さいため，安全頻度はあまり大きくなりません(b2, b3)。10 回(t)

[15] ここで安全率(BinS)を使わない理由は，期待頻度(f)までの「下側」の累積確率を求めることが必要だからです。

の中で 10 回(f)起きたことでも 1000 回の中では 764 回の出現にすぎません(b4)。

```
Sf(f=10,t=100)    #f:freq.=10^1,t:total=10^2 Sf: 67  (c1)
Sf(f=100,t=1000)  #f:freq.=10^2,t:total=10^3 Sf: 100 (c2)
Sf(f=1000,t=10000)#f:freq.=10^3,t:total=10^4 Sf: 111 (c3)
```

上の実験を見ると，たしかに 1000 回の中で 100 回起きることは，基準数(b)が 1000 であれば(デフォルト b=1000)，当然 100 回の出現が予想されますが(c2)，100 回の中で 10 回起きることは 1000 回で予想される回数は 100 回ではなく，67 回だけになります(c1)。逆に，10000 回の中で 1000 回起きることは，1000 回の中では 100 回を超えて，111 回起きることが 95%の安全率で予想されます(c3)。

```
Sf(f=5,t=10,,b=10)  #f:freq.=5,t:total=10,b:base=10 Sf: 5     (d1)
Sf(f=5,t=10,,b=100) #f:freq.=5,t:total=10,b:base=100 Sf: 29   (d2)
Sf(f=5,t=10,,b=1000)#f:freq.=5,t:total=10,b:base=1000 Sf: 244 (d3)
```

上の実験によれば，10 回の中で 5 回起きることは 100 回の中では 29 回(d2)，1000 回の中では 244 回起きることが予想されます(d3)。一般にパーセント(%)やパーミル(‰)は，基準数(b)を 100, 1000 にしたときの数値を示すので，それぞれ 50%や 500‰となりますが，上のように全数(t)が基準数(b)より小さな安全頻度(Sf)はパーセントやパーミルよりかなり小さな数値になります。たとえば 10 回中 3 回起きたことを簡単に 30%とし，これを 100 回中 25 回起きたときの 25%と比較して，それよりも多いと判断することはできません。次の出力を見ると，それぞれの安全頻度は 10, 25 になるので，10 回中 3 回起きたことよりも 100 回中 25 回起きたことのほうが高い数値(安全頻度)を示すからです。

```
Sf(3,10,100) # 10
Sf(25,100,100) # 25
```

このように，パーセント，パーミル，比率などのように「割合」を示す数値は単純な割り算ではなく，本来その分母である全数の大きさによって異なる扱いが必要です。

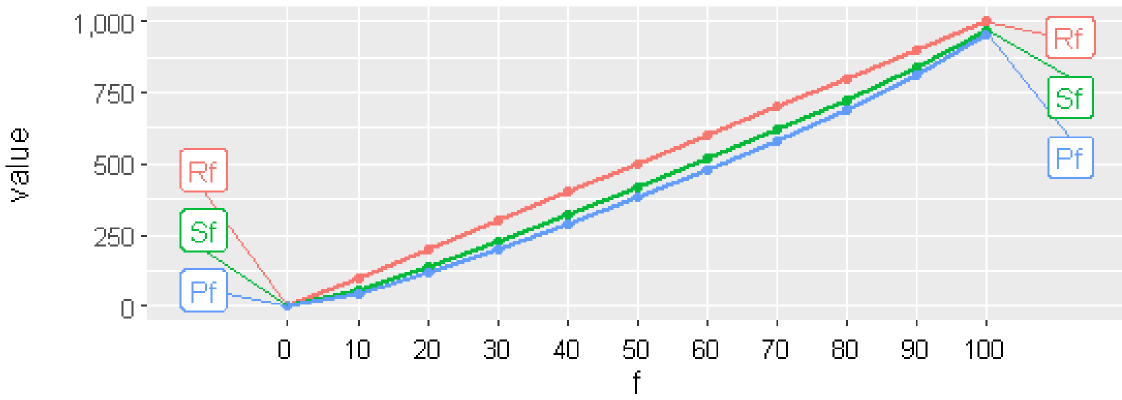
●相対頻度・確率頻度・安全頻度の比較

確率頻度(Pf)は度数(f)と全数(t)から求めた期待確率(e)に適当な乗数(m)を掛けた値であり，安全頻度(Sf)は度数(f)と全数(t)から求めた期待確率(e)を使って基準数(b)の中で予想される度数です。よって，確率頻度(Pf)は基本的に期待確率(e)と同じであり，期待確率との違いは乗数の大きさだけになります。一方，安全頻度は第 3 引数として乗数ではなく，基準数(base)を使っています。そこで，確率頻度を求めるときに使う乗数を安全頻度の

基準数に対応すると見なして確率頻度(Pf)と安全頻度(Sf)を比較します。参考までに相対頻度(Rf, ここではパーミル)も比較しましょう。

```
#1
V1=V2=V3=V4=V5=V6=V7=NULL
for(i in 0:10) {
  j=i+1; V1[j]=i*10; V2[j]=100;
  V3[j]=round(BinE(V1[j],V2[j]),4)
  V4[j]=1000; V5[j]=round(V1[j]/V2[j]*1000,0)
  V6[j]=Pf(V1[j],V2[j]); V7[j]=Sf(V1[j],V2[j])
}
D=data.frame(V1,V2,V3,V4,V5,V6,V7)
colnames(D)=V('f,t,e,m,Rf,Pf,Sf'); D
```

```
          f   t      e    m   Rf  Pf  Sf
1         0 100 0.0000 1000    0   0   1
2        10 100 0.0553 1000  100  42  68
3        20 100 0.1367 1000  200 116 156
4        30 100 0.2249 1000  300 198 248
5        40 100 0.3175 1000  400 287 343
6        50 100 0.4136 1000  500 381 440
7        60 100 0.5130 1000  600 479 540
8        70 100 0.6158 1000  700 582 642
9        80 100 0.7228 1000  800 691 747
10       90 100 0.8363 1000  900 809 856
11      100 100 0.9705 1000 1000 955 980
```



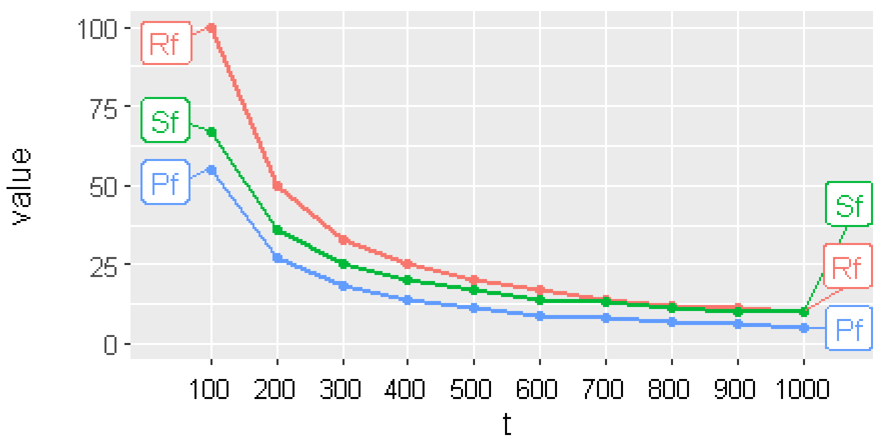
このように，同じ基準数(=1000)で比較すると安全頻度(Sf)は確率頻度(Pf)よりやや数値が高くなります。そして，度数(f)が両サイドの最小値・最大値に近いときは両者の差は小さいのですが中央の値ではかなり大きくなります。

次は度数(f)を一定にして(=10), 全数 t を 100,200,...,1000 に変化させたときの 3 つの頻度を比較します。

```
# 2
V1=V2=V3=V4=V5=V6=V7=NULL
for(i in 1:10) {
 V1[i]=10; V2[i]=100*i;
```

```
 V3[i]=round(BinE(V1[i],V2[i]),4); V4[i]=1000
 V5[i]=round(V1[i]/V2[i]*1000,0);
 V6[i]=Pf(V1[i],V2[i]); V7[i]=Sf(V1[i],V2[i])
}
D=data.frame(V1,V2,V3,V4,V5,V6,V7)
colnames(D)=V('f,t,e,m,Rf,Pf,Sf'); D
```

```
        f    t      e    m  Rf Pf Sf
1      10  100 0.0553 1000 100 42 68
2      10  200 0.0274 1000  50 21 37
3      10  300 0.0182 1000  33 14 26
4      10  400 0.0136 1000  25 10 21
5      10  500 0.0109 1000  20  8 18
6      10  600 0.0091 1000  17  7 15
7      10  700 0.0078 1000  14  6 14
8      10  800 0.0068 1000  12  5 12
9      10  900 0.0060 1000  11  5 11
10 10 1000 0.0054 1000  10  4 10
```



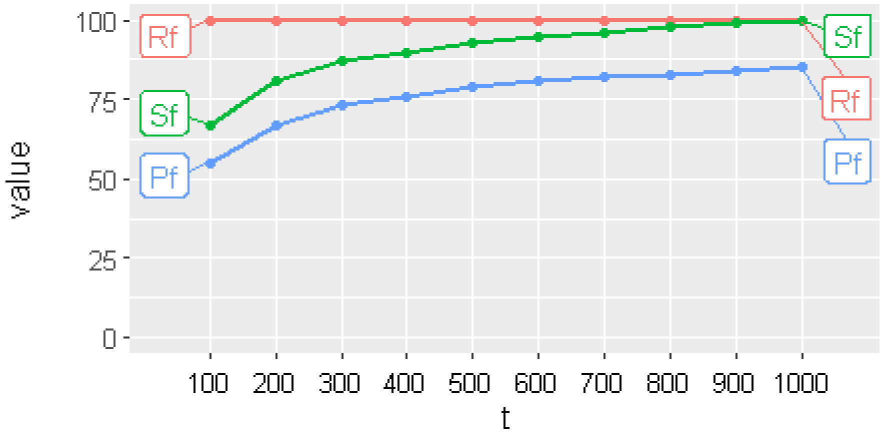
全体的に三者とも減少しますが，全数が小さいときは三者間に大きな差があり，全数が大きくなると差が小さくなります。

次に度数(f)と全数(t)の割合を一定にして(10%)，両者を増加させたときの 3 つの頻度を比較します。

```
# 3
V1=V2=V3=V4=V5=V6=V7=NULL
for(i in 1:10) {
 V1[i]=10*i; V2[i]=100*i;
 V3[i]=round(BinE(V1[i],V2[i]),4); V4[i]=1000
 V5[i]=round(V1[i]/V2[i]*1000,0);
 V6[i]=Pf(V1[i],V2[i]); V7[i]=Sf(V1[i],V2[i])
}
D=data.frame(V1,V2,V3,V4,V5,V6,V7)
colnames(D)=V('f,t,e,m,Rf,Pf,Sf'); D
```

```
    f    t      e    m  Rf Pf  Sf
1   10  100 0.0553 1000 100 42  68
```

```
2    20   200 0.0673 1000 100 57   82
3    30   300 0.0729 1000 100 64   88
4    40   400 0.0763 1000 100 68   91
5    50   500 0.0787 1000 100 71   94
6    60   600 0.0805 1000 100 73   96
7    70   700 0.0819 1000 100 75   97
8    80   800 0.0830 1000 100 77   99
9    90   900 0.0840 1000 100 78  100
10  100  1000 0.0848 1000 100 79  100
```

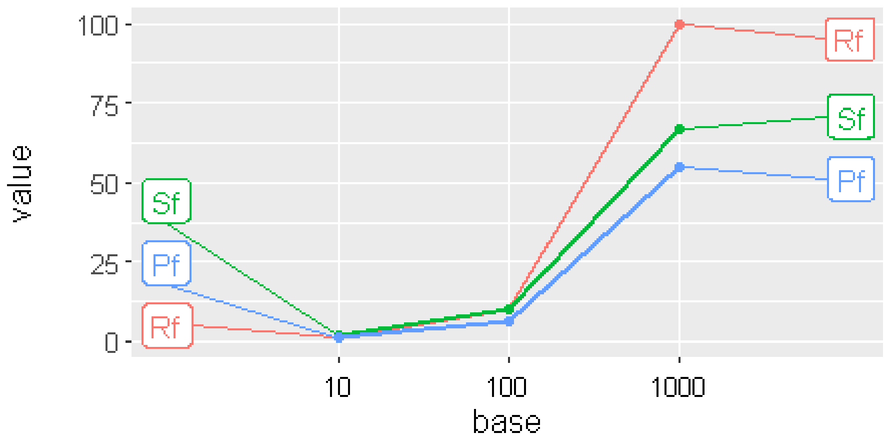


このように，相対頻度(Rf)はコンスタントに 100‰になりますが，安全頻度(Sf)と確率頻度(Pf)は，とくに全数が小さいときに，相対頻度との差が大きくなります(グラフの左側)。

これまでの観察では，すべて Rf > Sf > Pf という大小関係を見ました。そして，全数が小さいときに 3 つの頻度の差が大きくなることを見ました。最後に基準数(b)の変化(10,100,1000)との関係を見ます。

```
#4
V1=V2=V3=V4=V5=V6=V7=NULL
for(i in 1:3) {
  V1[i]=10; V2[i]=100; V3[i]=round(BinE(V1[i],V2[i]),4);
  V4[i]=10^i
  V5[i]=round(V1[i]/V2[i]*10^i,0);
  V6[i]=Pf(V1[i],V2[i],m=10^i)
  V7[i]=Sf(V1[i],V2[i],10^i)
}; D=data.frame(V1,V2,V3,V4,V5,V6,V7)
colnames(D)=V('f,t,e,base,Rf,Pf,Sf'); rownames(D)=V4; D
```

```
   f   t      e base  Rf Pf Sf
1 10 100 0.0424   10   1  0  3
2 10 100 0.0424  100  10  4 11
3 10 100 0.0424 1000 100 42 59
```



相対頻度(Rf)と確率頻度(Pf)は乗数(m)の大きさをそのまま反映しますが，安全頻度(Sf)のときの base は単なる乗数ではなく基準数となるので，かなり異なる動きを見せています。t=base=100 のように t と base が一致するとき，相対頻度と同じく，安全頻度はデータの度数(f=10)と一致します。これは全体が 100 のときに 10 回起きることは,基準数が同じ 100 であるとき予想される度数はやはり同じ 10 になることを示しています。このことは当然であるので，相対頻度(=10)と安全頻度(=10)は正しく，確率頻度(=4)は信頼できません。一方，全体が 100 のときに 10 回起きることは，10 回では 2 回起こることが安全頻度で予想されます。これは 10% (=1)でなく，それを超えています。しかし，全数が比較的大きいときの確率の安全率は高いので，全数が小さくなったときに安全頻度が高くなることは納得できます。

このような安全率に基づく予想頻度は相対頻度では考慮されず，一律に 10%が適用され，m=10, 100, 1000 のときの予想度数は 1, 10, 100 になります。一方，確率頻度は期待確率 0.0424 がコンスタントに適用され，それに乗数(m＝10，100，1000)をかけて，0.424, 4.24, 42.4 のように同じ倍率になります。それらは，全数が 10 のとき 0.424 (Pf=0)，全数が 100 のとき 4.24 (Pf=4)，全数が 1000 のとき.42.4 (Pf=42)となるので本質的にすべて同等の数値を示しています。

以上の観察から，安全頻度(Sf) > 確率頻度(Pf) > 相対頻度(Rf)の順で合理性・信頼性が確保される，と思われます。相対頻度はあまり信頼できません。

●安全頻度の乱数実験

次は安全頻度の妥当性を確かるために行った乱数を使った実験の結果です。

```
Sf(8,10,100) #secure frequency 58
f=8; t=10; s=.95; b=100  #f:freq.=8,t:total=10,s:security,b:base
e=BinE(f,t,s); e    #expected prob.: 0.4930987
sf=BinF(b,e,s); sf  #secure freq.: 58
rbinom(100,b,e)     #random number in normal dist.
```

```
 set.seed(1); R=rbinom(10^6,b,e); sum(R<=sf)/10^6    # security
(+): 0.967331 > 0.95
 set.seed(1); R=rbinom(10^6,b,e); sum(R<=sf-1)/10^6 # security
(-): 0.949616 < 0.95
```

この実験では，10 回の中で 8 回起きたことが 100 回の中では 58 回起こる，ということを確かめます(安全頻度=58)。はじめに，基準数 b=100，期待確率 e=0.4930987 として，乱数を 100 個生成します。

```
 rbinom(100,b,e)
 [1] 41 39 38 43 44 38 42 31 39 42 45 37 29 48 36 37 48 34 43 41 45
 [22] 40 41 40 42 38 43 44 38 42 45 42 41 40 42 47 48 33 32 40 38 40
 [43] 39 39 38 36 42 43 32 35 34 42 37 43 40 41 32 28 39 37 39 46 38
 [64] 39 36 41 32 39 51 47 41 33 35 33 40 47 39 38 43 35 36 48 35 37
 [85] 39 37 41 37 35 39 38 52 37 47 40 34 42 52 38 41
```

このような乱数を 100 万個生成し(10^6)，その中で安全頻度(sf=58)以下の数値が全体の中で占める確率を求めると，その確率はたしかに最初に設定した安全率(=信頼水準)=.95 に近似しています(0.967331)。求められた安全頻度以下であれば(sf-1=57)，安全率は.99 以下になります(0.949616)。よって，乱数を使った実験によって安全率(=0.95)を確保する頻度(安全頻度)は 58 であることが確かめられました。

●安全頻度のリスト

次は安全頻度(Sf)のリストです(基準数(b)=1000; 安全率(s):0.95/0.99)。関数 Sf()が使用できない環境で有用です。

```
 s=0.95; m=1; SF=matrix(0,10,10);
 rownames(SF)=colnames(SF)=(1:10)*m
 for(i in 1:10){
   for(j in 1:i) SF[j,i]=Sf(j*m,i*m,1000,s)
 }; SF[SF==0]=''; SF; C(SF)
```

(1) 安全頻度のリスト (基準:1000; 安全率:0.95)

*	1	2	3	4	5	6	7	8	9	10
1	63	35	25	20	17	15	13	12	11	10
2		246	154	114	91	77	66	59	53	48
3			395	272	211	173	147	129	114	103
4				500	368	296	248	215	189	170
5					576	445	367	314	275	245
6						633	506	427	371	329
7							678	556	477	420
8								713	598	520
9									741	632
10										765

*	10	20	30	40	50	60	70	80	90	100
10	765	327	215	162	130	110	95	84	75	68
20		880	528	387	308	256	220	193	172	156
30			921	639	501	414	353	309	275	248
40				942	709	580	493	430	381	343
50					955	758	639	554	490	440
60						963	792	683	603	540
70							969	819	719	642
80								974	840	747
90									977	856
100										980

*	100	200	300	400	500	600	700	800	900	1000
100	980	466	313	237	192	161	140	123	111	100
200		992	645	485	390	326	282	248	222	200
300			996	736	589	493	424	372	332	300
400				998	791	660	567	497	443	400
500					999	828	710	622	555	500
600						999	854	748	666	600
700							1000	873	777	700
800								1000	888	800
900									1000	900
1000										1000

Secure frequency with Base = 1000 and Security (s) = 0.95

(2) 安全頻度のリスト (基準:1000; 安全率:0.99)

*	1	2	3	4	5	6	7	8	9	10
1	19	12	9	8	7	6	6	5	5	5
2		124	78	58	47	40	35	32	29	26
3			247	168	130	107	91	80	72	65
4				352	254	202	169	146	129	116
5					435	329	269	229	200	178
6						502	393	328	283	250
7							556	447	380	332
8								600	494	425
9									636	533
10										667

*	10	20	30	40	50	60	70	80	90	100
10	667	272	179	135	109	92	80	71	64	59
20		825	476	347	276	230	197	174	155	141
30			884	596	464	382	326	285	254	229
40				915	673	547	463	403	358	322
50					933	727	609	527	465	418
60						946	766	657	578	517
70							955	796	695	619
80								961	819	725
90									966	838
100										971

*	100	200	300	400	500	600	700	800	900	1000
100	971	453	305	232	188	159	138	123	110	100
200		988	637	478	385	324	280	247	221	200
300			994	731	585	489	422	371	332	300
400				997	787	657	565	496	443	400
500					998	825	708	621	554	500
600						999	852	747	665	600
700							1000	872	777	700
800								1000	888	800
900									1000	900
1000										1000

Secure frequency with Base = 1000 and Security (*s*) = 0.99

次は，上の表から Sf(200,900)=222 と Sf(300,900)=332 を求め，比例補完を行って，Sf(235,900)を計算した結果を示します(256)。この値は関数 Sf(235,900)が返す正確な値 260 に一致しています。このように正確に一致しないときでも近似した値を得ることができます。

```
Sf(235,900) #260
Sf(200,900) #222
Sf(300,900) #332
222+(332-222)*35/100 #260
```

■スペイン語の疑問符と感嘆符

スペイン語の疑問文や感嘆文には，次のように文末だけでなく文頭にも逆転した疑問符と感嘆符がつけられます。

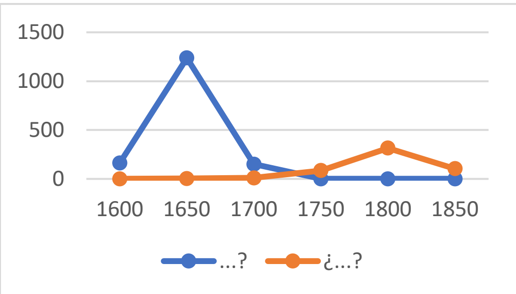
- ¿Qué es esto?「これは何ですか？」
- ¡Qué frío!「寒い！」

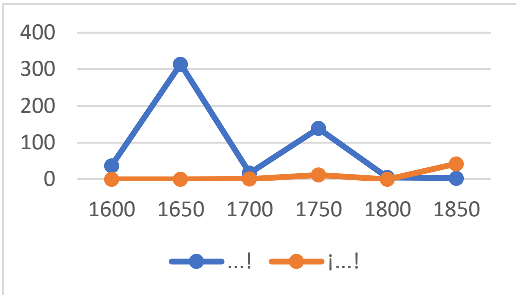
このような書き方の規範は 1754 年のスペイン王立アカデミーの正書法で規定されました(Real Academia Española / Asociación de Academias de la Lengua Española. 2010. *Ortografía de la lengua española*. Madrid. Espasa Libros. p. 387)。このような正書法の変革は突然に実行されるのではなく，先行する習慣が尊重されることが多いので，それぞれの年代に印刷された資料を使ってその歴史的変遷を辿ってみたいと思います[16]。

次の表と図は，それぞれの絶対頻度(AF)を示します。

AF	1600	1650	1700	1750	1800	1850
...?	162	1238	150	0	0	0
¿...?	0	3	10	84	315	103

AF	1600	1650	1700	1750	1800	1850
...!	37	314	17	139	5	3
¡...!	0	0	1	12	0	42



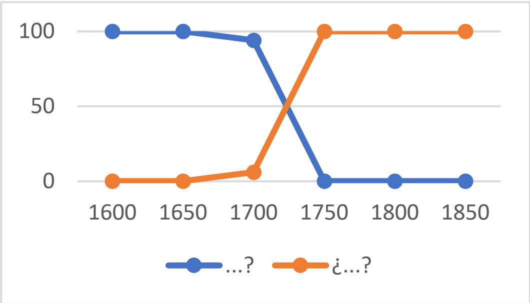


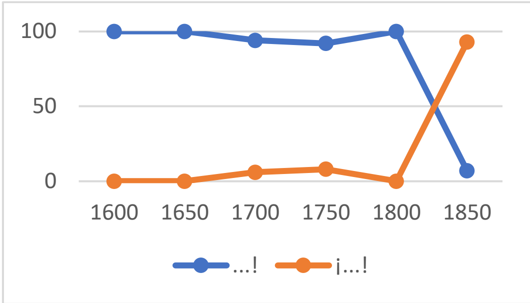
このように絶対頻度では，それぞれの年代の総頻度が異なるので，疑問符と感嘆符の出現の相対的な変化がよくわかりません。そこで次のように，それぞれの年代で相対化した頻度(RF:%)を計算しました。

[16] Programes en Lyneal: https://h-ueda.sakura.ne.jp/lyneal/programes.htm

RF	1600	1650	1700	1750	1800	1850
...?	100	100	94	0	0	0
¿...?	0	0	6	100	100	100

RF	1600	1650	1700	1750	1800	1850
...!	100	100	94	92	100	7
¡...!	0	0	6	8	0	93

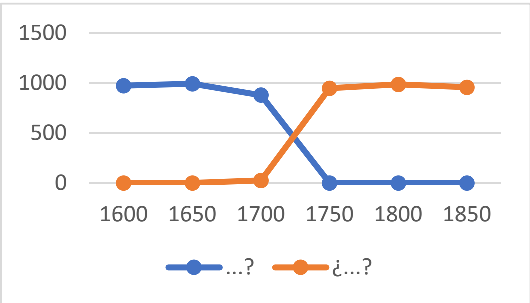


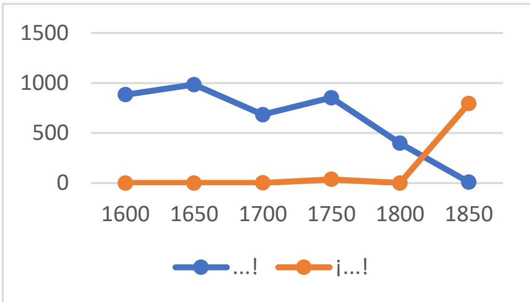


このように相対頻度(RF)はそれぞれの形態の歴史的変化をわかりやすく示します。しかし 1800 年代の「...!」の絶対頻度(AF)はわずかに 5 であるので，この数字が 100%を示すことにどれほどの有意性があるかのは疑問です。1600 年代や 1650 年代の「...!」も同じように 100%を示しますが，これら高頻度の例と同列に扱うことはできません。また，1600 年代と 1650 年代は!の絶対頻度(AF)が大きく異なりますが，相対頻度ではどちらも 100(%)になっています。そこで，次に統計的有意性を考慮した確率頻度(PF)を計算しました。

PF	1600	1650	1700	1750	1800	1850
...?	972	992	878	0	0	0
¿...?	0	0	260	947	985	956

PF	1600	1650	1700	1750	1800	1850
!	883	985	684	854	398	10
¡!	0	0	1	370	0	794



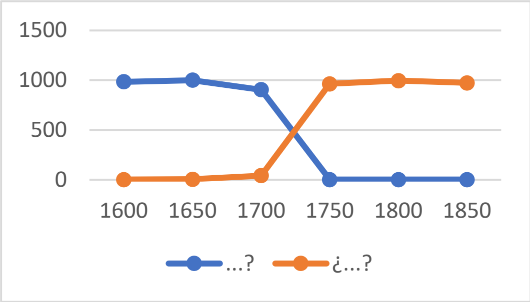


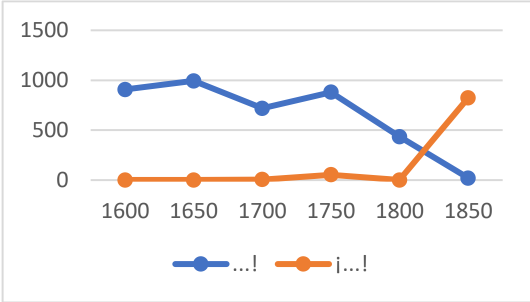
このように，確率頻度は 1800 年の「...!」をかなり小さく評価しているので，相対頻度よりも信頼できます。

さらに信頼性を高めるために，最後に安全頻度(SF)を計算します。

SF	1600	1650	1700	1750	1800	1850
...?	984	999	903	0	0	0
¿...?	0	3	40	963	994	972

SF	1600	1650	1700	1750	1800	1850
...!	907	994	719	881	435	19
¡...!	0	0	4	52	0	824



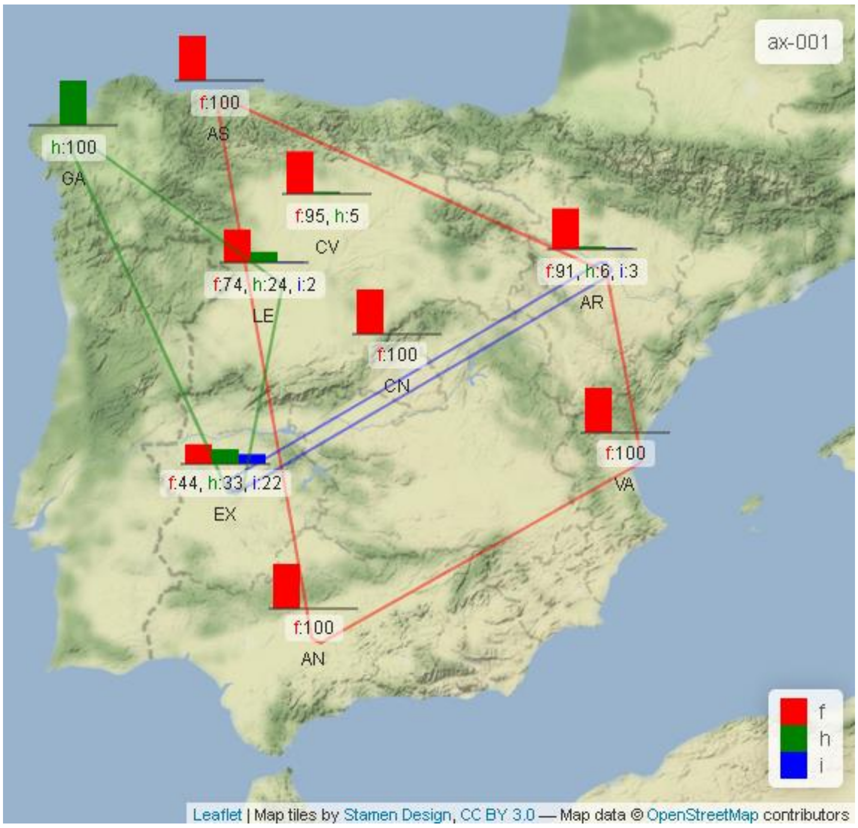


安全頻度(SF)は，先の確率頻度(PF)とよく似た傾向を示します。しかし，確率頻度(PF)は期待確率だけに基づいてそれを機械的に 1000 倍していますが，安全頻度は，期待確率に基づいて母数を 1000 にしたときに予想される頻度を正確に求めています。また，全体的に安全頻度のほうが確率頻度よりも高い数値を示します。

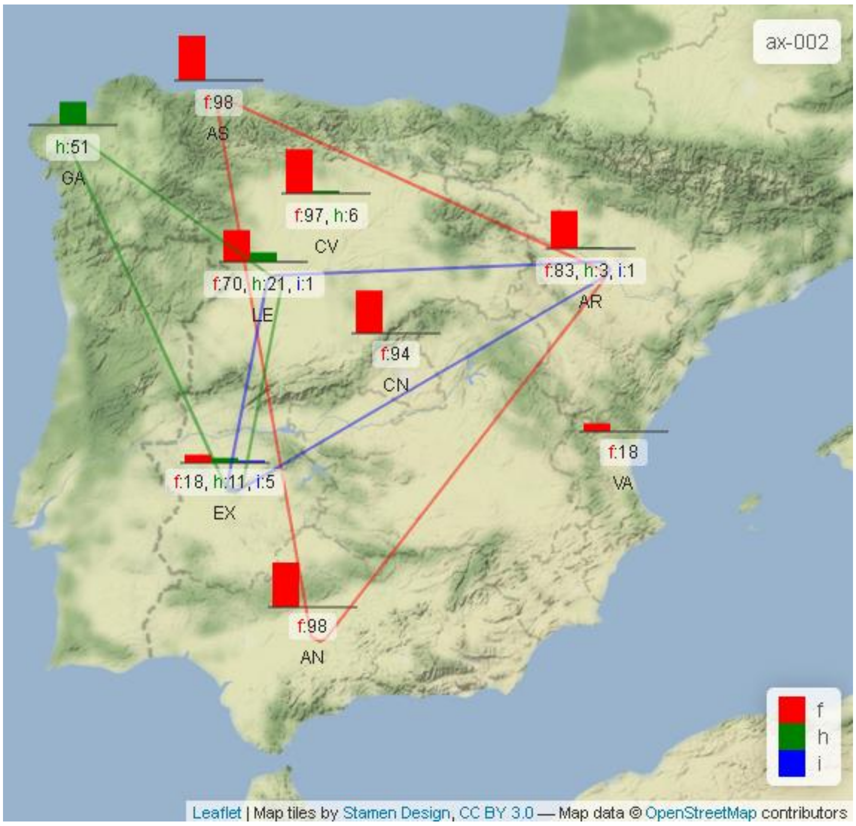
この資料の分析から，スペイン語に特有の逆転疑問符「¿...?」はスペイン王立アカデミーの規範(1754)に従って 1750 年代から急速に使用率が高まったことがわかりました。しかし，それは完全にゼロから出発したのではなく，半世紀前の 1700 年代にすでに存在していました。一方，逆転感嘆符「¡...!」の普及は 1 世紀遅く，1850 年代まで待たなくてはなりません。

■スペイン語 fijo > hijo の歴史・地理

ラテン語の語頭の F-が F > h > [ゼロ]になったことは他のロマンス諸語にはない，スペイン語の特徴です。次の 2 つのグラフは FILIU/A(M) > hijo/a「息子/娘」(単複)の語頭文字の 1200 年代の地理分布を示します。地図(1)ではパーセント(%)，地図(2)では安全頻度(SF)を使っています。赤が f-を示し，緑が h-を示し，青が i-を示します。



(1) fi-, hi- i- 1200 (%)



(2) fi-, hi- i- 1200 (SF)

地図(1)で，とくに左上の GA (Galicia)の h-と右下の VA (Valencia)の f-が目立ちます(どちらも 100%を示します)。しかし，それぞれの母数は 5, 2 に過ぎません。そこで，安全頻度を示す地図(2)を見ると，それぞれの値は次の式で計算されて適切な数値になります(%はどちらも 100)。

```
Sf(5,5,100) # 64
Sf(2,2,100) # 30
```

パーセントは母数の大小にかかわらず，すべてそのまま母数を分母とするので，このような不都合が起こります。このパーセントの問題はすべての相対頻度に共通しています。このように相対頻度と安全頻度による分析は結果が大きく異なるので注意が必要です。

3.5. 二項検定

私たちは言語データ分析で得られた頻度や比率が統計的に有意に大きな値であるか否かを判断しなければならないときがあります。たとえば，ある年代の文書に *voz*「声」という単語が<voz>または<uoz>で書かれる語形を数えて，それぞれ 7 語と 3 語が見つかったとき「voz が有意に多数である」と言えるでしょうか？たしかに voz が全体の半数以上になりますが(7 > 5)，このような差ならば偶然でも起こるかもしれません。そこで統計的に有意に多数であるのかをテスト(検定)する方法を先の安全率(security: S)を応用して説明します。ここでは，事前に設定する「信頼水準」(confidence level; 例: 0.95)と，検定の結果得られる安全率(security; 例: 0.97)を区別します。一般に，それぞれの補数である，事前に設定する「有意水準」(significance level; 例: 0.05)，検定の結果得られる P 値(p value, p; 例: 0.03)が区別されています。P 値は「危険率」(Risk, R=P)を示します。ここでは「信頼水準」・「安全率」を使い，その補数である「有意水準」・「危険率」を使わないで「二項検定」(binomial test)の方法を説明します[17]。さらに，検出力と信頼区間も扱います。

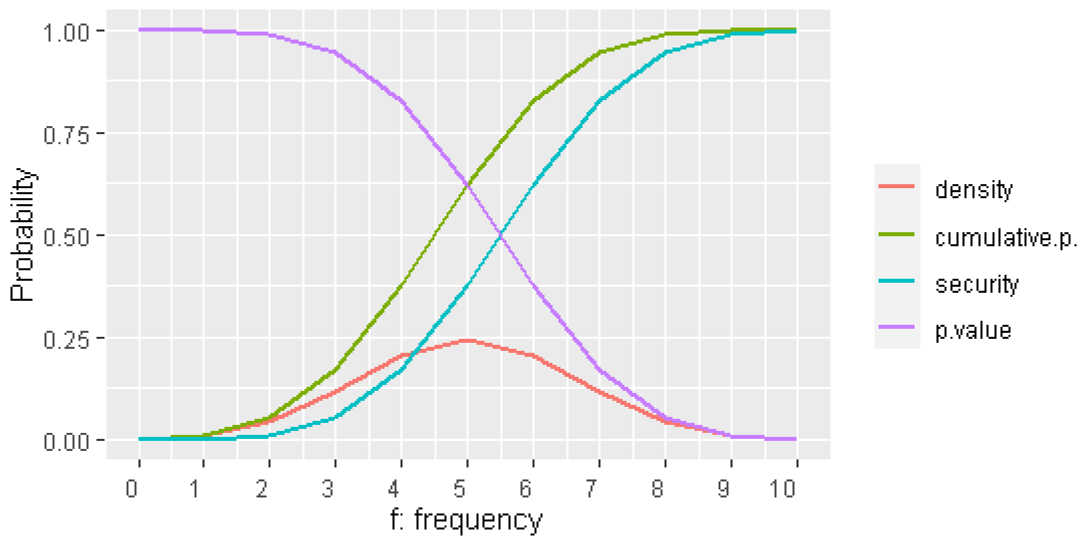
3.5.1. 安全率と P 値

次の表とグラフはコインを 10 回投げて(n=10)表が出る回数(f)が 0 ~ 10 回であるときの確率密度(density), 累積確率(cumulative.p.), 安全率(security), 危険率(P 値: p-value)それぞれの確率を示します。以下の表でわかるように P 値(p.value)は安全率(security)の補数です。

[17] 「信頼水準」(confidence level)は R の統計パッケージ stats::bonom.test で使用されています。「信頼係数」(confidence coefficient)のように「係数」(coefficent)よりも「水準」(level)の方が，事前に設定された値であることをよく示しています。有意水準(significance level, level of significance)ではなく，その補数である信頼水準を使う理由は以下で説明します。

P + S = 1, P = 1 - S　　(P: P 値, S: 安全率, P = R)

```
M=mapply; t=10; X=0:t; e=0.5
D=data.frame(f=X,density=M(BinD,X,t,e),cumulative.p.=M(BinP,X,t,e),
                  security=M(BinS,X,t,e),p.value=M(BinR,X,t,e))
print(round(D,3),row.names=F)

  f density cumulative.p. security p.value
  0   0.001         0.001    0.000   1.000
  1   0.010         0.011    0.001   0.999
  2   0.044         0.055    0.011   0.989
  3   0.117         0.172    0.055   0.945
  4   0.205         0.377    0.172   0.828
  5   0.246         0.623    0.377   0.623
  6   0.205         0.828    0.623   0.377
  7   0.117         0.945    0.828   0.172
  8   0.044         0.989    0.945   0.055
  9   0.010         0.999    0.989   0.011
 10   0.001         1.000    0.999   0.001
```



このように，累積確率(cumulative.p.)と安全率(security)は近接していますが，安全率は累積確率の f-1 に対応します。これは，f に対応する確率が危険率の領域に入るためです。

上の表によれば，期待確率(e) = 0.5 の事象(たとえばコイン投げ)が 10 の試行回数(n)の中で 7 回起こることの安全率(S)が 0.828 (82.8%)であることになります。一方，統計的検定では直接「数値が有意に多数であるか」を考えるのではなく，「数値が有意に多数でない」という「帰無仮説」(null hypothesis)を設定し，その帰無仮説に対して対立仮説(alternative hypothesis;「数値が有意に多数である」)が正しい確率(P)はどの程度か，を問います。ここでは x = 0, 1, 2, ..., 7 まで p 値は下がり続け，x = 7 のときに p = 0.172 になることがわかります。しかし x = 7 では 17.2%も帰無仮説を支持する確率があるので帰無仮説を棄却できません。x = 7 は偶然に左右されるコイン

投げのようなことでも 17.2%の確率で出現する数値だからです。

棄却する基準となる確率は「有意水準」(sl: significance level: α)と呼ばれ，一般に緩い 5%(sl=0.05)，または厳しい 1%(sl=0.01)が設定されています。有意水準を 5%に設定すれば x = 9 で有意水準を下に向けて超えます。有意水準を 1%に設定すれば x = 10 のときに初めて有意水準を下に超えます。

P 値は出現数(x)，標本数(n)，期待確率(e)によって変化します。たとえば，全文書中 35%の確率(e)で見出される言語現象が 10 文書で 7 回見つかったとすれば，その P 値は次の計算から 0.026 となるので，統計的検定（二項検定）によって有意に多数であると判定され(有意水準:5%)，「有意に多数でない」という帰無仮説を棄却して「有意に多数である」という「対立仮説」を採ります。ここでは関数 BinR(x, n, e) (x 出現数,n 試行数,e 期待確率)を使います[18]。

```
BinR(7, 10, 0.35) # 0.02602428
```

このような帰無仮説の設定は「背理法」「帰謬(きびゅう)法」(proof by contradiction, lat. reductio ad absurdum)の方法に似ています。背理法では目的の仮説(対立仮説)を直接証明するのではなく，目的の仮説が偽であると仮定し(帰無仮説)，そこから矛盾を導くことにより帰無仮説が偽である，つまり目的の仮説(対立仮説)が真であると結論づける方法です。統計的検定では帰無仮説の「偽である」ことの正しさを P 値で問います。その P 値が有意水準より低ければ(下に超えたとき)，偽であること（帰無仮説）が間違えている可能性が高いので，帰無仮説を退けて（棄却して）対立仮説を採用します。しかし，このことは帰無仮説が間違えている，ということではなく，間違えている可能性が高い，対立仮説が正しい可能性が高いと判断することになります。この方法を「否定的検定」(negative test)と呼びます。

先に見たように，P 値(P)と安全率(S)は補数の関係になるので，P 値の代わりに安全率を使って二項検定をするならば，仮説の正しさは，安全率(BinS)を使って次のように直接計算できます。

```
BinS(7, 10, 0.35) # 0.9739757
```

この検定法を「肯定的検定」(positive test)と呼びます。肯定的検定では帰無仮説を設定せず，証明しようとする仮説は帰無仮説に対する「対立仮説」ではなく，「作業仮説」(working hypothesis)と呼び，作業仮説が正しいことの確率を安全率(S)で直接求めます。あらかじめ肯定的検定における

[18] 検定の結果は「p = 0.026 < α: 0.05」のように P 値(p)と有意水準(α)を共に示し，有意水準を下回るか否かを不等号(<, >)で示すとよいでしょう。R 関数 *pbinom* を用いれば P 値は次のようになります。

安全率：pbinom(7-1, 10, 0.35) # 0.974
P 値：1 - pbinom(7-1, 10, 0.35) # 0.026

信頼水準を 0.95 (95%)，または 0.99 (99%)と設定しておき，安全率がこの信頼水準を上に超えたときに仮説を採用します。このとき作業仮説を採択できる確率は 97.4%になります(S=0,974 > cl=0.95, cl:信頼水準)。

帰無仮説を設定する否定的検定における 5%または 1%の有意水準や，帰無仮説を設定しない肯定的検定における 95%または 99%の信頼水準はガイドラインにすぎません。たとえば有意水準 = 0.10 (10%)や信頼水準 = 0.90 (90%)にすれば x=8 であっても有意であると判定されます。しかし合意による一定の基準がなければ，統計的な判断がすべて自由になり，恣意的なものになってしまいます。統計的検定の結果を報告するときは，(対立)仮説の採択の有無だけでなく，P 値(否定的検定)または安全率(肯定的検定)も示すべきです。

否定的検定をした結果，P 値が有意水準(たとえば 5%)以上であれば，帰無仮説を棄却できません。しかし，このことは帰無仮説が正しい，ということを示すのではなく，帰無仮説が誤りであったとは言えない，という意味であり，その判断は保留されます。この論理はやや複雑です。そして，P 値が有意水準(5%)以下であれば帰無仮説を棄却し，目的の対立仮説を採用する，という手続きもかなりの思考の回り道が必要です。

肯定的検定では，安全率をゼロ(0%)から 1(100%)に連続して漸進するスケールの中で評価します。そのとき，信頼水準(たとえば 95%)は 1 つの目安・ガイドラインに過ぎません。否定的検定のように，有意水準を絶対視して，それを元に帰無仮説の棄却・対立仮説の採択を決定することはしません。たとえば，有意水準を 0.05 (5%)としたとき，P 値が 0.0499 (4.99%)ならば帰無仮説を棄却し，対立仮説を採択され，P 値が 0.050 (5.00%)ならば帰無仮説を棄却せず、統計的な有意性が得られないと結論づけることになります。しかし，0.0499 と 0.0500 の間に実質的な差はほとんどありません[19]。そのようなわずかの差に対立仮説採択の可否がかかっていることは不合理です。肯定的検定における信頼水準は，対立仮説採択の可否を決める基準ではなく,有意性の大きさを評価するときの目安として使用します。このように，肯定的検定の論理は単純で，作業仮説を採用する手続きも直接的・直感的です。そのとき，否定的検定のようにとくに採択・棄却という，二者択一的な結論を導きません。むしろ，安全率をゼロ(0%)から 1(100%)の範囲にあるスケールの中で評価します[20]。

対立仮説の採択をめぐる二者択一的な結論は「可否の判断・決定」のた

[19] むしろ, 0.0500 と 0.2000 の差, 0.0499 と 0.0100 の差のほうが重要です。しかし，否定的検定では，前者はどちらも帰無仮説を棄却しないケースであり，後者はどちらも帰無仮説を棄却するケースであり，区別しません。否定的検定では「A は有意差があり，B は有意差がない」というように有意水準を元に表明しますが，肯定的検定では，「A は B よりもが有意差がある」という評価をします。

[20] 安全率はゼロになることも完全になることもないので，その範囲は 0 と 1 を含みません。よって，範囲は[0, 1]ではなく，(0, 1)で表記します。

めの手続きであり，一方，有意性の連続漸進的評価は数値情報の知見を得るための手段です。本来，研究は後者を本務とするはずです[21]。

一般に，「多重検定」(multiple tests: 一度に多数の検定を行うこと)は否定的検定では注意が必要です。問題となるのは、第 1 種の誤り(帰無仮説が真であるにも関わらず、帰無仮説を誤って棄却してしまう誤り)の確率(危険率)が増えてしまうことです。たとえば、有意水準=0.05 で 5 つの仮説検定を行うと、全体として誤りが起こる確率が 5 * 0.05 = 0.25 となるからです。このように誤りの確率が増加するため、統計的な有意性を正しく評価することが難しくなります。そのようなときは，有意水準を厳しくして，検定の回数で割った値 0.05/5=0.01 を使うことが考えられます。しかし，そうすると，たとえば 10 回の検定をするときは，0.005 という非常に厳しい有意水準となり，どの検定でも有意性が見つけることが困難になります。一方，肯定的検定は，有意水準を使って対立仮説の採択の可否を決めることを目標にすることはないので，すべて同じ信頼水準をガイドラインとして安全率の多寡を評価するのでとくに問題となりません。

次の図式で否定的検定と肯定的検定を比較します。

*	否定的検定	肯定的検定
水準	有意水準(5%, 1%)	信頼水準(95%, 99%)
仮説	帰無仮説・対立仮説	作業仮説
論理	背理法	直接法
対象	P 値(危険率)	安全率
判断	消極的(保留可)	積極的(保留しない)
結論	二者択一的(棄却/採択)	連続漸進的(0→1)
多重検定	有意水準を調整する	同じ信頼水準を使う

肯定的検定でも否定的検定でも，仮説の絶対的真偽を決定しているわけではありません。このことを肯定的検定で説明します。肯定的検定の安全率は仮説の正しさの程度を示しているのであって，決して 100%にはなりません[22]。安全率は一定の基準(=信頼水準)を設けて作業仮説の採択の可能性を検討しています。たとえば, 95%の信頼水準で安全率が 96%であれば, 作業仮説を採択する可能性がありますが, 4%の確率で仮説が偽であることがありえます。一方, 安全率が 94%であるということで仮説を棄却すると, 6%の確率で仮説が真であることがあります。

はじめに否定的検定の例を示します。R の統計パッケージ binom.test で求めた P 値と BinR, BinP, BinT で求めた P 値を比較したものです。

[21] Prieto Valiente y Herranz Tejedor, 2004, *¿Qué significa "estadísticamente sifnificativo"? La falacia del criterio del 5% en la investigación científica.* Madrid. Díaz Santos.

[22] コインを 10 回投げて 10 回とも表が出るときの安全率は 0.9990234375 になりますが，それでも 1 (100%)にはなりません

```
binom.test(7,10,0.5,'g')$p.value # greater: 0.171875
BinR(7,10,0.5) # 0.171875

binom.test(2,10,0.5, 'l')$p.value # less: 0.0546875
BinP(2,10,0.5) # 0.0546875

binom.test(9,10,0.5,'t')$p.value # two-sided: 0.02148438
BinT(9,10,0.5) # 0.02148438
```

```
BinT=function(f,t,e) 2*min(pbinom(f,t,e), 1-pbinom(f-1,t,e))
# Binomial test (two-sided)
```

このように，binom.test は上側検定(P 値: x 以上になる確率)と下側検定(x 以下になる確率)の区別を"greater"と"less"で区別し，それぞれ BinR と BinP に対応します。両側検定(two-sided)の結果を示す BinT は下側の累積確率と上側の累積確率の小さい方を 2 倍にした値を返します。

次は肯定的検定の結果です。ここでは安全率 BinS を使用します。

```
BinS(7,10,.5,'g') # greater: 0.828125
BinS(2,10,.5,'l') # less: 0.9892578
BinS(9,10,.5,'t') # two-sided: 0.9892578
```

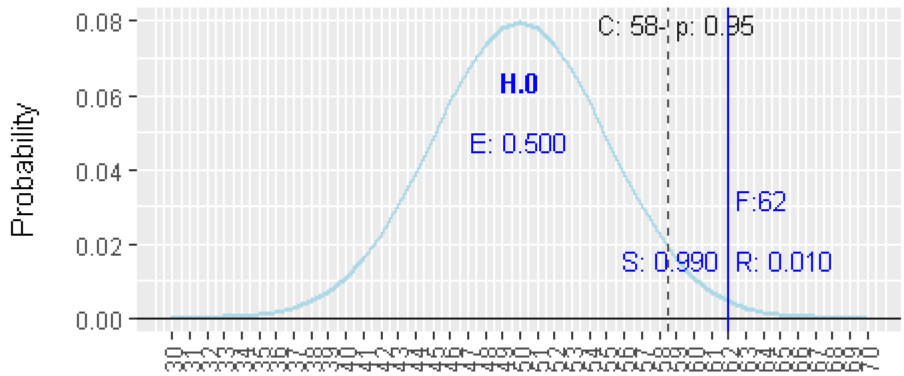
上側検定では安全率を使って判定します。安全率が 95%以上であれば有意とされます。BinS の仕組みは少し複雑ですが，関数を作ればその使い方は簡単です。

●片側検定と両側検定

右側検定・左側検定・両側検定の順で安全率・危険率(P 値)を計算する方法を説明します。

(1) 右側検定 (f:度数, t:和, e:期待確率)

```
f=62; t=100; e=.5
ifelse(f==0,1,pbinom(f-1,t,e,F)) #right-side: 0.01048937
binom.test(f,t,e,'g',0.95)$p.value #greater: 0.01048937
gSecurity(f,t,side='g',A=30:70,B=30:70,a=90) # Graph
```

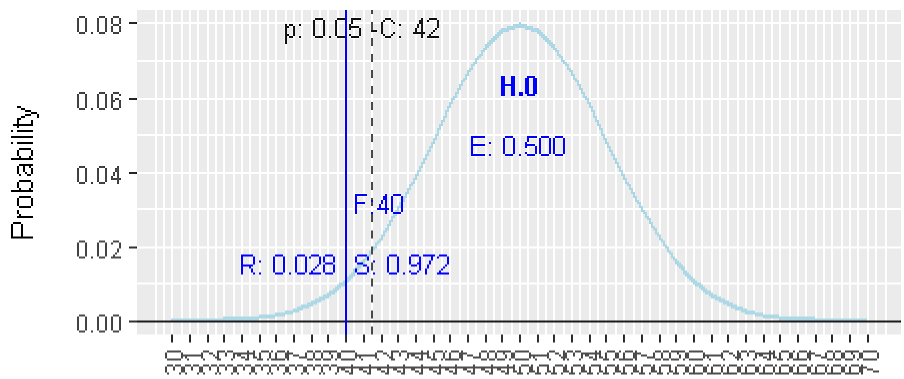


f: 62, t: 100, e: 0.5, c.l.: 0.95, side: g, test: b

P 値は少し複雑な計算になります。pbinom(f-1,t,e,F)は f から t までの全確率の和を返します(f=0 では pbinom(-1,t,e,F)が計算できないので，ifelse 関数を使って，このときの P 値をゼロとします)。R 関数 binom.test を使っても同じ結果になります。安全率・危険率のグラフ(gSecurity)を見ると，安全率(S):0,990，危険率(P 値)R: 0.010 があります。データの度数(F=62)は信頼水準 95%の境界(C:58，p:0.96)を超えています。つまり安全率(S:0.990)が信頼水準 95%を超えているので，否定的検定で有意と判定されます。

(2) 左側検定 (f:度数, t:和, e:期待確率)

```
f=40; t=100; e=.5
pbinom(f,t,e) #0.02844397
binom.test(f,t,e,'l')$p.value #less: 0.02844397
gSecurity(f,t,side='l',A=30:70,B=30:70,a=90) # Graph
```

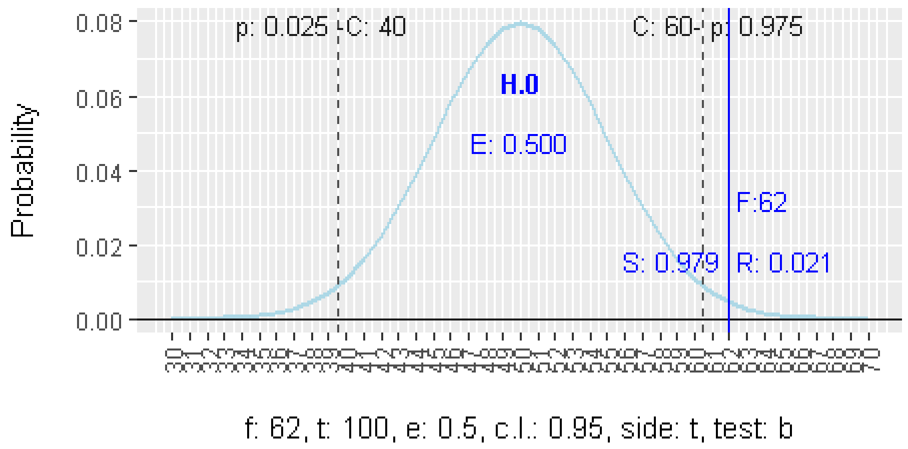


f: 40, t: 100, e: 0.5, c.l.: 0.95, side: l, test: b

左側検定では，対象の数値(f=40)が有意に小さいか否かを判断します。よって，中間値(t/2=50)より小さい値が対象になります。このときの P 値は pbinom(f,t,e) で計算し，0.02844397 が得られます。この P 値は f～t の範囲のすべての確率の和です。R 関数 binom.test(f,t,e,'l')$p.value も同じ値を返します。

(3) 両側検定 (f:度数, t:和, e:期待確率)

```
f=62; t=100; e=.5
p1=pbinom(f,t,e); p1 #p-value (left-side): 0.9939835
p2=ifelse(f==0,1,pbinom(f-1,t,e,F)); p2; #p-value (right-side): 0.01048937
mn=min(p1,p2); mn #mn:min(left-side, right-side) 0.01048937
p=2*mn; p #p:p-value (risk) 0.02097874
binom.test(f,t,e,'t')$p.value #two-sided: 0.02097874
gSecurity(f,t,A=30:70,B=30:70,a=90) # Graph
```

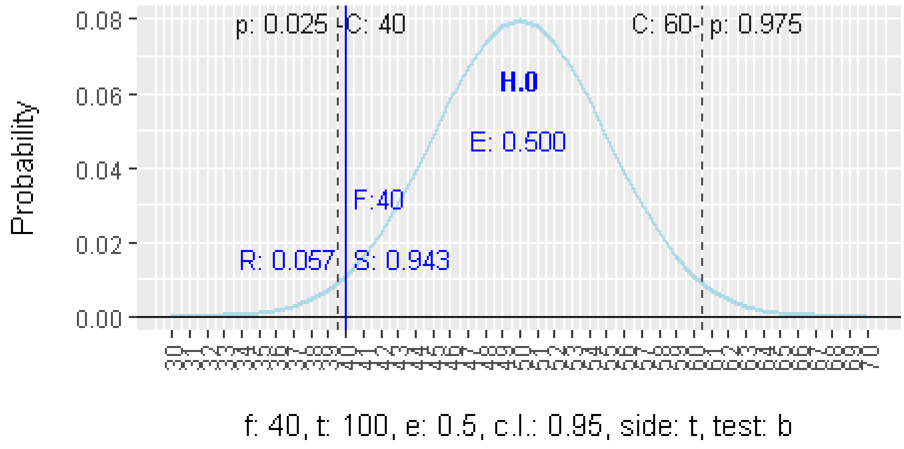


先の右側検定と同じ f, t, e を使って，両側検定の P 値を計算します。左側と右側の P 値(p1, p2)を計算し，それぞれ，0.994, 0.010 を得ます。上の図を見ると，両側検定の帰無仮説 H0 の棄却域は左右に，それぞれ 0.025 を設定し，両者を合わせて 0.05 (5%)としています。先の 2 つの P 値(p1, p2)のうち，小さなほう(関数 min を使う: mn=min(p1,p2))，右側の P 値(p2)が棄却域に入っています。f の値が小さいと棄却域に入らないときもありますが，それでも小さなほうを使って，P 値を計算します。そして，小さなほうの P 値を 2 倍した値 が両側検定の P 値です(p=2*mn: 0.021)。小さなほうの P 値を 2 倍とする理由は，両側検定の棄却域の境界値の確率(0.975)が先の右側検定の棄却域の境界値の確率(0.950)よりもさらに右に行き，棄却域の面積(確率の和)が半分になっているためです。先の右側検定の危険率 R=0.010 の 2 倍になります。最後の桁はわずかに異なりますが(R: 0.021)，これは小数点以下の丸めの誤差によるものです。このように，2 倍にした P 値は棄却域に入らなくなる可能性がありますが，今回の場合は，2 倍化しても境界値を超えていますから，帰無仮説は棄却されます。当然ですが，安全率(S=0.979)も信頼水準(0.95, 95%)を超えています。

参考までに，対象の数値がデータの中間値以下の場合(f: 40)の両側検定の結果も見ておきましょう。

```
f=40; t=100; e=.5
p1=pbinom(f,t,e); p1 #p-value (left-side): 0.02844397
p2=ifelse(f==0,1,pbinom(f-1,t,e,F)); p2; #id.: 0.9823999
mn=min(p1,p2); mn #mn:min(left-side, right-side) 0.02844397
p=2*mn; p #p:p-value (risk)   0.05688793
```

```
binom.test(f,t,e,'t')$p.value #two-sided: 0.05688793
gSecurity(f,t,A=30:70,B=30:70,a=90) # Graph
```



このように，数値(F: 40)は左側に位置して，検定を受けます。危険率・P 値は R=0.057 なので，帰無仮説の棄却域に入りません。否定的検定では，この時点で分析を断念しますが，肯定的検定では安全率 0.943 を評価します。この安全率は否定的検定の棄却域を決定する信頼水準 0.95 に近接していますから，その差は本質的な違いというより可能な誤差の範囲にあると考えられます。安全率は連続的な尺度であり，交通信号のように対立的な基準にはなりません。たとえば，新しい学習法を導入した結果，100 人の生徒の答案で誤答者が 40 人いたとしても，十分に有効な学習法である可能性があります。

言語のバリエーション研究（地理的変異や歴史的変化を扱う）では右側検定(頻度の高さを検定)を使うことが多いのですが，文体論研究や言語教育データ分析では左側検定(頻度の低さを検定)が必要になることがあります。前者では対象の数値が有意に大きいか否かを検定します。逆に，後者では対象の数値が有意に小さいか否かを検定します。一般の統計学の本で説明されている両側検定(対象の数値が有意であるか否かを検定)は，工業製品の品質管理(計測機の精度)などで用いられます。

次に同じ条件(f:60, t:100,e:0.5)で片側検定(右側: 'g' greater)と両側検定('t': two-sided)が示す P 値(p.value)を比較します。

```
binom.test(60,100,0.5,'g',0.95)$p.value # greater:        0.02844397
binom.test(60,100,0.5,'t',0.95)$p.value # two-sided:      0.05688793
```

このように，片側検定の P 値は両側検定の P 値よりも小さくなり(1/2 になる)，有意性をよく示しています。一方，両側検定の P 値は片側検定の P 値の 2 倍になるので検定が厳しくなります(有意になりにくい)。

しかし，検定が厳しくても両側検定をすれば，頻度(f)が全数(t)の 1/2 以上では右側検定，1/2 以下では左側検定，という区別する必要はなく，すべて同じ条件で統一して検定できるようになります。たしかに，頻度が中

央の値の上にあるか，下にあるかはデータの状態によるので，はじめから決めることはできません。一方，両側検定の難点は，検定に関わらない棄却域(対象の数値が中間値よりも上のときの左側の棄却域)までも設定して，それを考慮した判定をしていることです。先の両側検定の 2 つのグラフを見ると，2 本の境界線が引かれていますが，実際にはそのうちの 1 本だけが使われているのです。このそれぞれの境界線は有意水準(5%)ではなく，有意水準の半分の領域(2.5%)です。このことが，両側検定において，数値の有意性を認めることが困難にしています(難度を 2 倍にしている)。両側検定のもう一つの難点は，否定的検定において対立仮説が採択されても，その有意性の意味が「有意である」「有意な相違がある」というだけで，有意に大きいのか，有意に小さいのかがわからないことです。

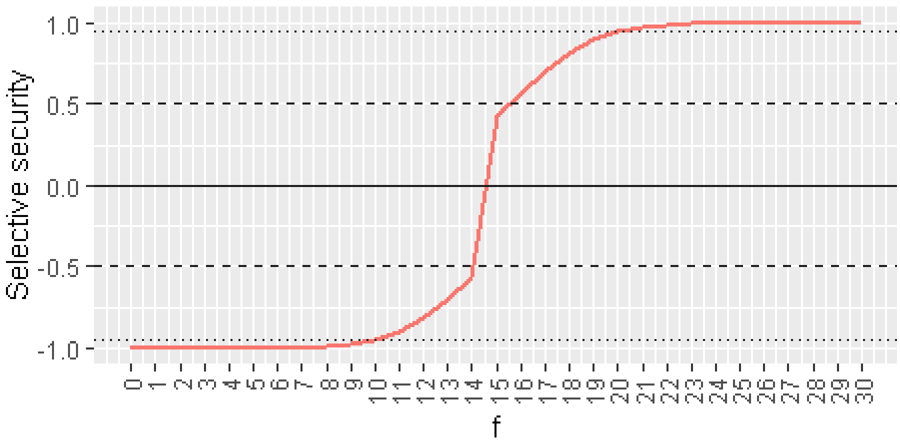
一方，検定対象が比較する値よりも明らかに大きい(または小さい)ことがデータ全体を通して言えるならば，それぞれのケースで右側検定(または左側検定)をすることができます。すべてのケースで右側検定(または左側検定)をすれば，その検定結果を比較できます。そして，右側の肯定的検定であれば，対象の数値(f)が全数(t)の 1/2 以下であっても，安全率(S)を 0 ~ 1 の範囲の中で評価することが可能です。そのとき，安全率(S)が 0.5 (50%) 以上であれば，その安全率が信頼性(fiability)を示していると考えられます。

●選択的安全率

以上の議論では右側検定・左側検定の選択を対象の数値とデータの中間値と比較して検定の前提にしています。そこで，度数(f)が中間値(t/2)より小さいときは左側検定の安全率の負値を返し，それ以外のときは左側検定の安全率の正値を返す「選択的安全率」(Selective security)の関数 Selsecurity を作成し，次のように実験しました。

```
Selsecurity=function(f,t,e=0.5){
  s=ifelse(f<t/2,'l','g'); m=ifelse(f<t/2,-1,1); BinS(f,t,e,s)*m
} # Selective security (plus/minus)

Selsecurity(15,20) #0.9793053
Selsecurity(7,20) #-0.868412
t=30; X=0:t; D=Df(X,mapply(Selsecurity,X,t))
colnames(D)=c('f','Selective security')
print(D[11:21,],row.names=F)

  f Selective security
 10         -0.9506314
 11         -0.8997558
 12         -0.8192027
 13         -0.7076676
 14         -0.5722322
 15          0.4277678
```

```
 16           0.5722322
 17           0.7076676
 18           0.8192027
 19           0.8997558
 20           0.9506314

G=gLines(D,lx='f', ly='Selective security', a=90)
G=G+geom_hline(yintercept=0,linetype=1,color='gray10')
G=G+geom_hline(yintercept=.5,linetype=2,color='gray10')
G=G+geom_hline(yintercept=-.5,linetype=2,color='gray10')
G=G+geom_hline(yintercept=.95,linetype=3,color='gray10')
G=G+geom_hline(yintercept=-.95,linetype=3,color='gray10')
G
```



Selsecurity(15,20)のように，f >= t/2 のときは正の安全率を返し(0.9793053)，Selsecurity(7,20)のように，f<t/2 のときは負の安全率を返します。グラフを見ると，f の変化に従った正負の安全率の動きがわかります。0.95, 0.5, -0.5, -0.95 に点線を引いて基準線を示しました。これは，目安となる参照値であって,作業仮説の採択を決定するものではありません。

3.5.2. 検出力

統計的検定(否定的検定)では「帰無仮説」(null hypothesis)と「対立仮説」(alternative hypothesis)を区別します。

帰無仮説：検定対象の数値は有意でない。
対立仮説：検定対象の数値は有意である。

ここで，「有意」というのは「偶然では起こらない」という意味です。そこで，次のように書き換えると分かりやすくなります。

帰無仮説：検定対象の数値は偶然でも起きる。
対立仮説：検定対象の数値は偶然では起きない。

そして，統計的検定には次の 3 種があります。

両側検定：検定対象の数値は有意に(偶然では起きないほど)大きい，または小さい。
右側検定：検定対象の数値は有意に(偶然では起きないほど)大きい。
左側検定：検定対象の数値は有意に(偶然では起きないほど)小さい。

検定の目的は帰無仮説(「有意な差がない」)を棄却して対立仮説(「有意な差がある」)が採択される可能性があるか否かを判定することです。しかし，対立仮説の採択については次の二種の誤りがあり得ます。

(1)「第一種の誤り」：可能な帰無仮説なのに棄却してしまう。
(2)「第二種の誤り」：可能な対立仮説なのに採択しない。

「第一種の誤り」の「可能な帰無仮説を棄却してしまう」ということは直ちには理解しにくいので，簡単な例を使って説明します。たとえば 20 枚のコインを投げる実験を考えましょう。コインを投げて表(おもて)が出る期待確率は 0.5(50%)です。そのとき，15 枚が表になったことを想像します。その帰無仮説を「期待確率(0.5)との差は有意ではない」「偶然でも起こる」と設定します。たしかに，下の出力(H0.r: risk)が示すように 15 枚以上のコインが表になる確率，つまり，2 項分布確率の和 P(X>=15)は 0.021 (2.1%)となり，有意水準(5%)以下になります。言い換えれば，安全率(H0.s)は 0,979 (97.9%)であり，信頼水準(95%)以上になります。このようなことは「偶然でも起こる」(帰無仮説)とは考えられません。よって，「偶然でも起こる」という帰無仮説が棄却され，「偶然では起こらない」という対立仮説が採択されます。しかし，それでもわずかですが(2.1%の確率)，この帰無仮説が正しいことがあり得ます。そうすると帰無仮説を棄却して対立仮説を採用したことが誤りであることになります。その誤りを「第一種の誤り」(type I error)と呼びます。これは「可能な帰無仮説を棄却してしまう」「可能性を無視する」ことを意味します。その確率(1.2%)は可能性を無視する危険を犯す確率なので「危険率」と呼びます[23]。危険率は一般に「P 値」と呼ばれます。

```
t=20; A=0:t; r=0.75; e=0.5
D=data.frame(X=A,H0=dbinom(A,t,e),H1=dbinom(A,t,r),
           H0.r=BinR(A,t,e),H0.s=BinS(A,t,e),H1.ac=pbinom(A,t,r))
print(round(D,3), row.names=F)

  X    H0    H1  H0.r  H0.s H1.ac
  0 0.000 0.000 1.000 0.000 0.000
```

[23] 一般の統計学では「危険率」を「有意水準」の意味で用いられますが，「危険」と「安全」の意味は相反するので，安全率の補数として危険率を使うことにします。

```
     1 0.000 0.000 1.000 0.000 0.000
     2 0.000 0.000 1.000 0.000 0.000
     3 0.001 0.000 1.000 0.000 0.000
     4 0.005 0.000 0.999 0.001 0.000
     5 0.015 0.000 0.994 0.006 0.000
     6 0.037 0.000 0.979 0.021 0.000
     7 0.074 0.000 0.942 0.058 0.000
     8 0.120 0.001 0.868 0.132 0.001
     9 0.160 0.003 0.748 0.252 0.004
    10 0.176 0.010 0.588 0.412 0.014
    11 0.160 0.027 0.412 0.588 0.041
    12 0.120 0.061 0.252 0.748 0.102
    13 0.074 0.112 0.132 0.868 0.214
    14 0.037 0.169 0.058 0.942 0.383
    15 0.015 0.202 0.021 0.979 0.585
    16 0.005 0.190 0.006 0.994 0.775
    17 0.001 0.134 0.001 0.999 0.909
    18 0.000 0.067 0.000 1.000 0.976
    19 0.000 0.021 0.000 1.000 0.997
    20 0.000 0.003 0.000 1.000 1.000
```

このように，安全率・危険率・P 値は「第一種の誤り」と関係しますが，他方で，「検出力」(Power)は「第二種の誤り」と関係します。ここでは，先と同じ実験で，20 個のコインを投げて 15 個が表になったときに想定される確率 15 / 20 = 0.75 をコイン投げ一般の期待確率(0.5)と比較します。このとき検定の対象となる事態(f=15, t=20)で想定した確率(0.75)を「対象確率」と呼びましょう。対象確率の確率分布が証明したい対立仮説(=作業仮説)に対応するので「H1:対立仮説確率分布」(H1: probability distribution of alternative hypothesis)と呼びます。それに対し，期待確率(0.5)の確率分布を「H0:帰無仮説確率分布」(H0 probability distribution of null hypothesis)と呼びます。

上の出力によれば，X=14 までの H0 が示す危険率(P 値, H0.r)は 0.058 なので，有意水準(0.05, 5%)以下になっていません。X=15 のときになって初めて，危険率(P 値, H0.r)は 0.021 (P(X>=15))となり，有意水準(0.05, 5%)以下になります。そこで，X=14 が帰無仮説を棄却しない領域 (X=0:14)の「限界値」(c: critical value)とされます。問題の数値がこの限界値以上(X=15:20)ならば危険率・P 値が有意水準以下となるので(0.021)，帰無仮説 H0 が棄却されます(危険率・P 値: 0.021, 2.1%)。一方，X=0:14 の範囲では帰無仮説 H0 が棄却されないので，対立仮説 H1 が採択されないことになります。その限界値 c=14 で対立仮説が占める確率は，出力(H1.ac: 対立仮説確率分布累積確率)を見ると 0.383 (38.3 %)になっています。つまり，X が 0 から 14 までの範囲にある H1 の確率の和は帰無仮説が棄却されず，よって(帰無仮説が保たれているので)対立仮説 H1 が採択されない確率ですが，38.3%の確率で起きる可能性があることです。これが「第二種の誤り」(「可能な対立仮説を採択しない」)を犯す確率(beta)です。そして，その補数 1 - beta = 1

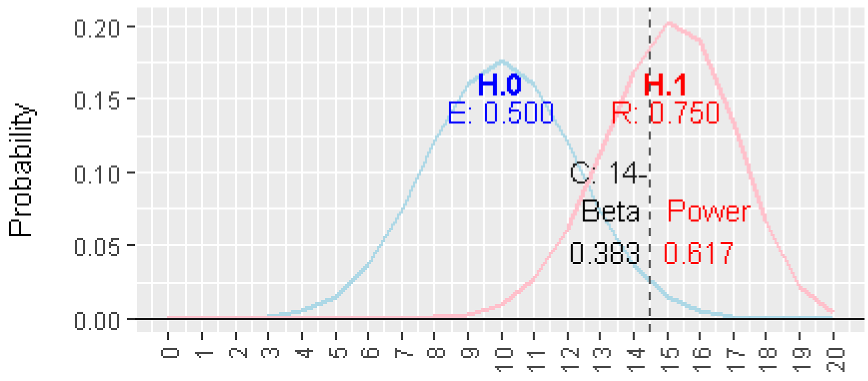
- 0.383 = 0.617 (61.7 %) が「検出力」が占める確率です。検出力は「第二種の誤り」を犯していない確率(「可能な対立仮説を正しく採択する確率」)を示す数値なので重要です。一般に，検出力の目安は 0.8 が使われることが多く, 今回のように標本数が少ない(t=20)ときは 0.7 が最低ラインとされています[24]。よって，今回の検定(f=15, t=20, e=0.5)では，確かに P 値は 0.05 以下なので，有意と判定されますが，検出力は弱く 61.7%なので(最低ライン以下)，検定結果を慎重に扱うならば検定をパスしません。第二種の誤りを犯す確率(0.383=38.3%)が高いためです。

一方，肯定的検定ではすべてのケースの安全率と検出力を計算し，信頼水準 (95%)に基づく限界値を参照しながら，その結論を導きます。たとえ安全率・検出力が低くても結論を控えることはありません。比較的低い安全率と検出力であっても報告の価値があるからです。このように，否定的検定と肯定的検定は同じ事実を分析し，かなり異なる結論を導きます。しかし，次に見るように，限界値の設定・検出力の算出方法・グラフ出力は同じです。はじめに右側検定の限界値の設定方法を説明します。

```
f=15; t=20; r=f/t; e=.5; cl=.95; r #(1) r: 0.75
c=qbinom(cl,t,e); c #(2) critical value 14
pw=pbinom(c,t,r,F); round(pw,3) #(3) pw: power 0.617
gPower(f,t,e,cl=.95,A=0:t,B=0:t,side='g',test='b',a=90) #(4) graph
```

#(1)で頻度(f), 和(t), 期待確率(e), 信頼水準(cl)を設定し，「データの比率 r=f/t=0.75 は比較する期待確率(e)よりも有意に大きい」という作業仮説(H1)を検証します。#(2)で，R 関数 *qbinom* を使って，信頼水準(cl)，和(t)，期待確率(e)から限界値(c)を求めます。R 関数 *qbinom* は，和(t=10)と期待確率(e=0.5)で設定される確率分布の中で，信頼水準(cl)が 0.95 の位置にある度数を返します。これが限界値(c)=14 になります。#3: この限界値(c)を使って, pbinom(c,t,r,F)で検出力(pw: power)を求めます(#3): 0.617。pbinom の最後の引数(F: FALSE)は限界値(c)以上のすべての確率の和を計算することを示します。以上のことを理解するためには次のグラフが役立ちます(#4)。

[24] 豊田秀樹(2009)『検出力分析入門：R で学ぶ最新データ分析』東京図書 p.36.



f: 15, t: 20, e: 0.5, c.l.: 0.95, side: g, test: b

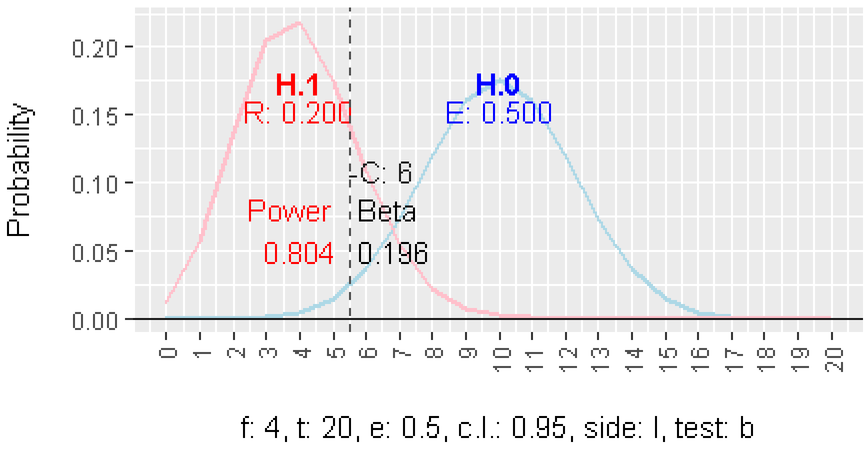
はじめに，限界値(c=14)は縦の点線で示されています。このとき限界値(c)は 14 なのですが，信頼水準以下の領域に入るので，14 と 15 の間に境界があることを示すために，f=14.5 の位置に線を引きました。安全率は境界線の左側の H0(青)の領域の確率の総和になります。「第二種の誤り」を犯す確率は同じ境界値(14)で対立仮説の領域(H1:赤のグラフ)を区分して，その左部分の確率の和で示されます(Beta)。H1 の境界値を超えた右側が「正しく対立仮説を採択する・評価する確率」に対応します。この確率の和は「検出力」(Power)と呼ばれ，統計の信頼性を示す重要な指標とされています。ベータと検出力は次の関係になります。

beta + power = 1, power = 1 - beta

肯定的検定では検出力は信頼水準とともに参考にするに留め，たとえば，「参考までに 95%信頼水準で検出力は….になる」というような評価をします。

次は左側検定の場合です。

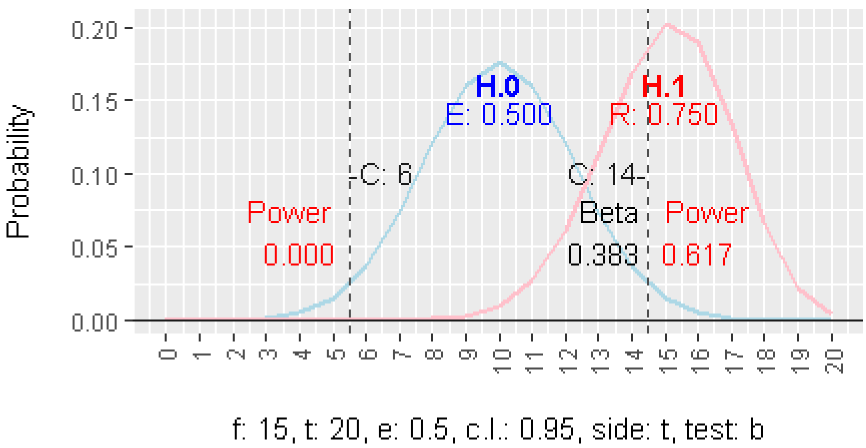
```
#(b) less: left-side H0: r>=e, #H1: r<e
f=4; t=20; e=.5; r=f/t; r; cl=.95 #(1) r=0.2
c=qbinom(1-cl,t,e); c #(2) c:critical value 6
pw=pbinom(c-1,t,r); pw #(3) pw:power: 0.8042078
gPower(f,t,e,cl,A=0:t,B=0:t,side='l',a=90) #(4) graph
```



ここでは頻度(f)=4，計(t)=20 のデータの比率(rt)=f/t=0.2 の有意性を信頼水準(cl)=0.95 で検討します。境界値(c)は *qbinom* 関数(分位数)で求めます。このときの期待確率(e)を 0.5 とします。つまり，データの確率 r=0.2 が期待確率(e=0.5)と比べて有意に小さいかを見ます。*qbinom*(1-cl.t.e)は境界値(c)=6 を返します。検出力の領域の全確率は 0:c の範囲の全確率の和になるので，これを *pbinom*(c-1,t,r)で求めます(pw=0.804)。

最後に両側検定の場合を見ます。

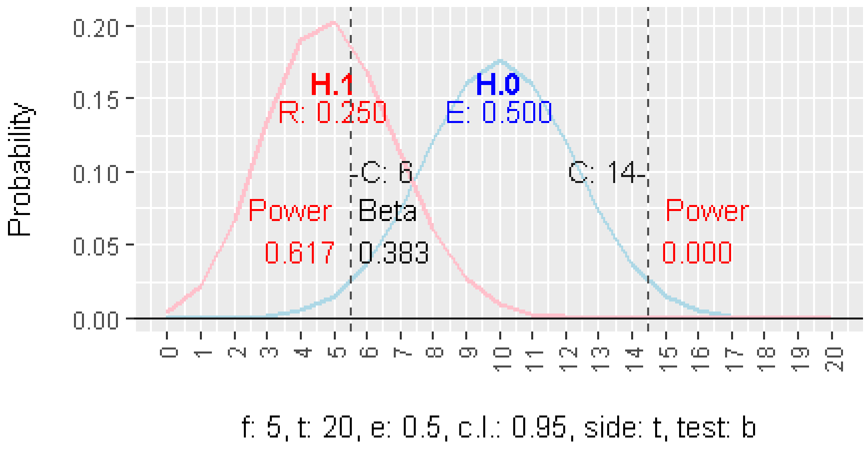
```
f=15; t=20; r=f/t; r; e=.5; cl=.95; side='t'; test='b' #(1) r: 0.75
c1=qbinom((1-cl)/2,t,e); c1 #(2) s:0.025, c1: lower critical value: 6
c2=qbinom((1+cl)/2,t,e); c2 #(3) s:0.975, c2: upper critical value: 14
pw=pbinom(c1-1,t,r)+pbinom(c2,t,r,F); pw #(4) pw: 0.6171765
```



比較する H0 (e:0.05)のグラフ(青)の両側で帰無仮説を棄却する領域で，左側境界の確率を(1-cl)/2=0.025 ，右側境界の確率を(1+cl)/2=0.975 として，それぞれの境界値 c1=14, c2=26 を求めます。この 2 つの境界値を使って，対立仮説・作業仮説(H1)の確率の中で検出力(pw: Power)を計算します。左側の pw は pbinom(c1-1,t,rt)となり，右側の pw は 1-pbinom(c2,t,rt)で計算し，両者を足して全体の検出力を求めます(pw=0.441)。グラフを見ると，この場合左側の pw はゼロ(0)

に近いのですが，計算式は f が t の半数以下の場合もあるので，この式を使います。

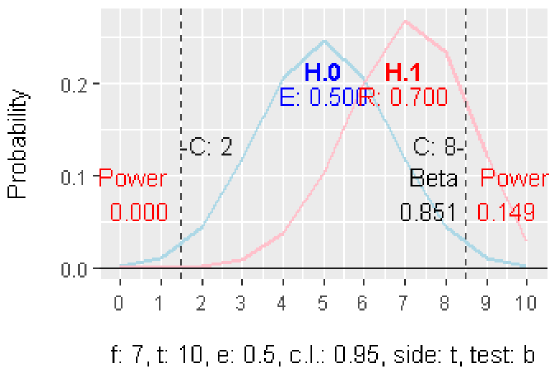
次の図は f=5 としたときのグラフです。このように，対立仮説・作業仮説(H1)のグラフは左側になります。

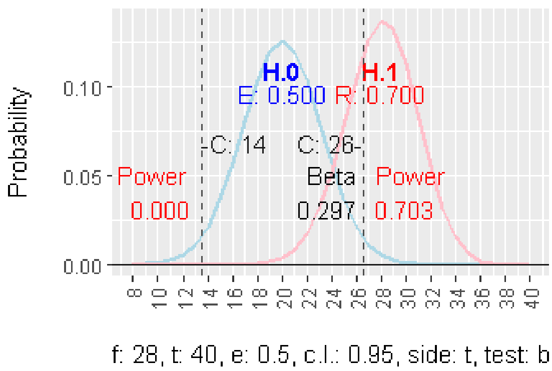


●検出力の変化

検出力(pw)は頻度(f)，和(t)，期待確率(e)，信頼水準(cl)，検定の種類(両側，右側，左側)によって変化します。それぞれを変化させて，右側検定で比較します。

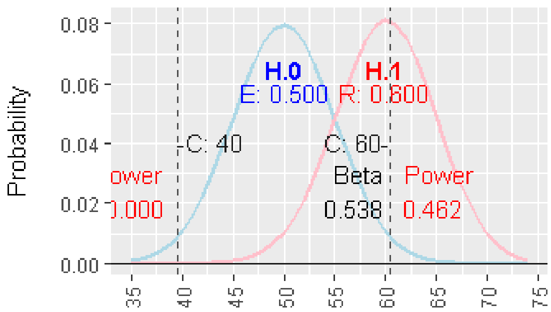
初めに 7/10=r1 と(7*4)/(10*4)=r2 を e=0.5, cl=.95 で比較します(両側検定)。



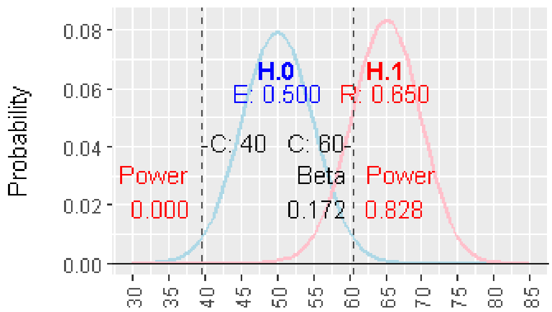


左図(f=7, t=10)では，安全率(S:.656)も検出力(Power:.149)も高い信頼性がありませんが，右図(f=28, t=40)とすると，どちらも高い信頼性を獲得しています(S:.983, Power:.703)。このように比率は同じでも(.7)，和(標本数)が大きくなると，信頼性が高くなります。

次は，f=60; t=100 と f=65, t=100 を比較した結果です。後者のほうが，期待確率(e=0.5)との差が大きいので，信頼性が高くなっています。

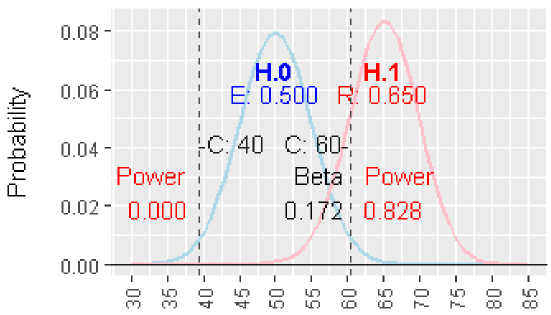


f: 60, t: 100, e: 0.5, c.l.: 0.95, side: t, test: b

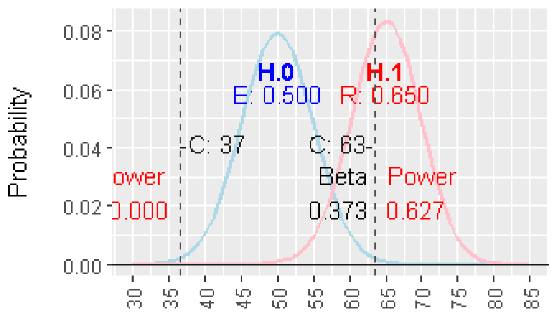


f: 65, t: 100, e: 0.5, c.l.: 0.95, side: t, test: b

次に，f=65, t=100 という条件で，信頼水準(c.l.)を 0.95, 0.99 とした場合を比較します。下の図が示すように，信頼水準の相違は検出力を変化させ，信頼水準を上げると検出力は低下します(Power = .828 →.627)。この理由は下の図を比較するとわかります。信頼水準(c.l.) = 0.95 のときは，限界値の境界線が中央に寄っていますが，信頼水準(c.l.) = 0.99 のときは，限界値の境界線は外に開き([.025, .975] → [.005, .995])，その分だけ対立仮説の左側領域(ベータの領域)が拡大し，右側領域(検出力の領域)が狭くなるためです。

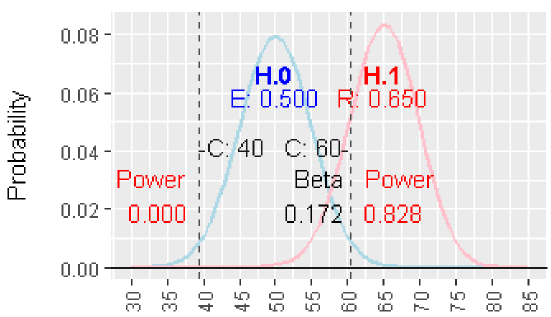


f: 65, t: 100, e: 0.5, c.l.: 0.95, side: t, test: b

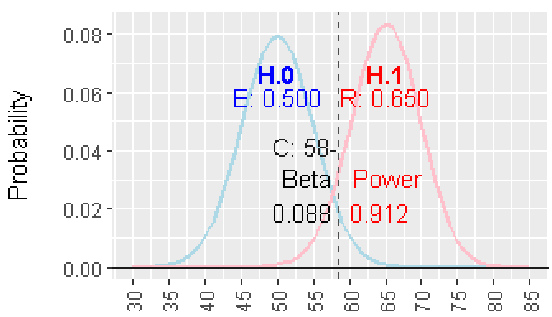


f: 65, t: 100, e: 0.5, c.l.: 0.99, side: t, test: b

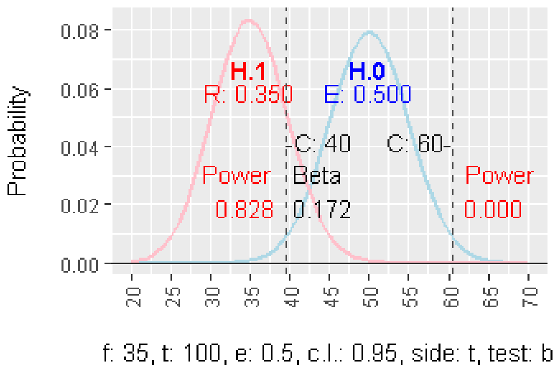
最後に検定の種類の違いによる検定力の変化を見ましょう。ここでは両側検定と片側(右側・左側)検定を比較します。結果は両側検定が片側検定よりも検出力が低くなっています。つまり，両側検定のほうが検定が厳しいということです。

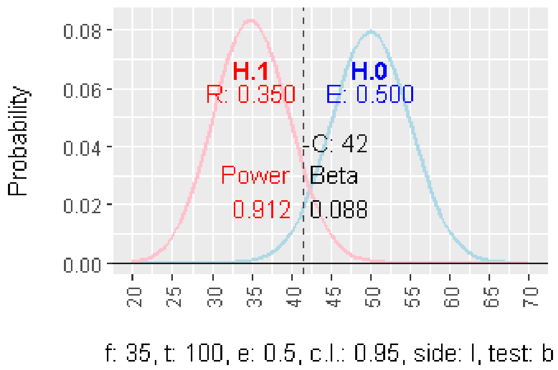


f: 65, t: 100, e: 0.5, c.l.: 0.95, side: t, test: b



f: 65, t: 100, e: 0.5, c.l.: 0.95, side: g, test: b





●検出力のリスト

```
 cl=.99;  x=2;  e=0.5;  m=10^x;  M=matrix(0,10,10);
rownames(M)=colnames(M)=(1:10)*m
 for(i in 1:10){
    for(j in 1:i) M[j,i]=Round(BinPower(j*m,i*m,e,cl,'g'),3)
 }; M[M==0]=''; M=gsub('0¥¥.','.',M); noquote(M); Copy(M)
```

(1) 検出力のリスト (信頼水準:0.95)

*	t:1	2	3	4	5	6	7	8	9	10
f:1	.000	.000	.000	.000	.000	.000	.000	.000	.000	.000
2		.000	.000	.000	.010	.001	.000	.000	.000	.000
3			.000	.000	.078	.016	.003	.006	.001	.000
4				.000	.328	.088	.020	.035	.008	.002
5					1.000	.335	.095	.135	.041	.011
6						1.000	.340	.367	.143	.046
7							1.000	.736	.372	.149
8								1.000	.736	.376
9									1.000	.736
10										1.000

*	t:10	20	30	40	50	60	70	80	90	100
f:10	1.000	.021	.000	.000	.000	.000	.000	.000	.000	.000
20		1.000	.585	.040	.001	.000	.000	.000	.000	.000
30			1.000	.946	.336	.046	.001	.000	.000	.000
40				1.000	.997	.831	.274	.046	.002	.000
50					1.000	1.000	.974	.720	.230	.044
60						1.000	1.000	.999	.925	.623
70							1.000	1.000	1.000	.993
80								1.000	1.000	1.000
90									1.000	1.000
100										1.000

*	t:100	200	300	400	500	600	700	800	900	1000
f:100	1.000	.038	.000	.000	.000	.000	.000	.000	.000	.000
200		1.000	1.000	.049	.000	.000	.000	.000	.000	.000
300			1.000	1.000	.998	.047	.000	.000	.000	.000
400				1.000	1.000	1.000	.982	.048	.000	.000
500					1.000	1.000	1.000	1.000	.950	.047
600						1.000	1.000	1.000	1.000	1.000
700							1.000	1.000	1.000	1.000
800								1.000	1.000	1.000
900									1.000	1.000
1000										1.000

Power with confidence level = 0.95, f:frequency, t:total

(2) 検出力のリスト (信頼水準:0.99)

*	t:1	2	3	4	5	6	7	8	9	10
f:1	.000	.000	.000	.000	.000	.000	.000	.000	.000	.000
2		.000	.000	.000	.000	.000	.000	.000	.000	.000
3			.000	.000	.000	.000	.003	.000	.000	.000
4				.000	.000	.000	.020	.004	.001	.000
5					.000	.000	.095	.023	.005	.001
6						.000	.340	.100	.026	.006
7							1.000	.344	.104	.028
8								1.000	.346	.107
9									1.000	.349
10										1.000

*	t:10	20	30	40	50	60	70	80	90	100
f:10	1.000	.006	.000	.000	.000	.000	.000	.000	.000	.000
20		1.000	.286	.008	.000	.000	.000	.000	.000	.000
30			1.000	.821	.156	.007	.000	.000	.000	.000
40				1.000	.986	.560	.091	.009	.000	.000
50					1.000	1.000	.882	.458	.083	.006
60						1.000	1.000	.991	.784	.307
70							1.000	1.000	.999	.947
80								1.000	1.000	1.000
90									1.000	1.000
100										1.000

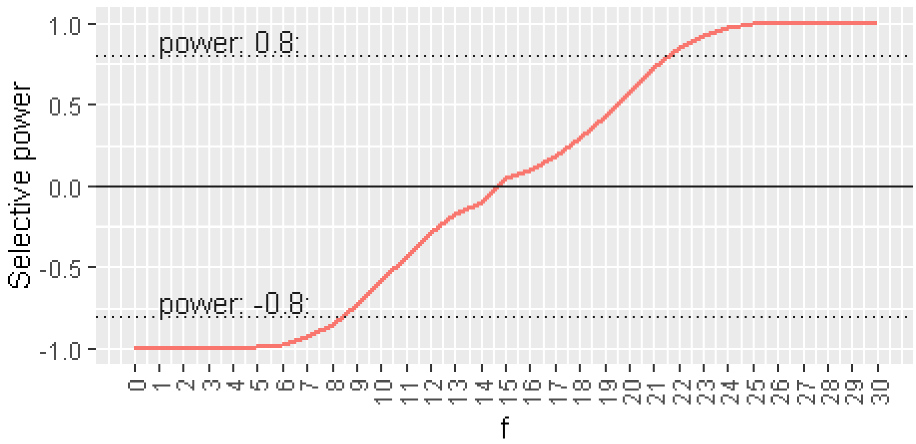
*	t:100	200	300	400	500	600	700	800	900	1000
f:100	1.000	.010	.000	.000	.000	.000	.000	.000	.000	.000
200		1.000	1.000	.009	.000	.000	.000	.000	.000	.000
300			1.000	1.000	.984	.010	.000	.000	.000	.000
400				1.000	1.000	1.000	.921	.009	.000	.000
500					1.000	1.000	1.000	1.000	.835	.009
600						1.000	1.000	1.000	1.000	1.000
700							1.000	1.000	1.000	1.000
800								1.000	1.000	1.000
900									1.000	1.000
1000										1.000

Power with confidence level = 0.99, f:frequency, t:total

●選択的検出力

次は度数(f)が和の中間値(t/2)より小さいときは左側の検出力を計算し，度数(f)が和の中間値(t/2)以上のときは左側の検出力を計算する「選択的検出力」(Selective power)のプログラム Selpower とその出力です。

```
Selpower=function(f,t,e=0.5,cl=.95){
   s=ifelse(f<t/2,'l','g'); m=ifelse(f<t/2,-1,1); Power(f,t,e,cl,s)*m
} # Selective power (plus/minus)

Power=function(f,t,e=.5,cl=.95,side='g'){
   r=f/t #f: freq, t:total, e:exp.probability, cl:conf.level, side:g,l,t
   if(side=='g') {c=qbinom(cl,t,e); pw=pbinom(c,t,r,F)}
   if(side=='l') {c=qbinom(1-cl,t,e); pw=pbinom(c-1,t,r)}
   if(side=='t') {
      c1=qbinom((1-cl)/2,t,e); c2=qbinom((1+cl)/2,t,e)
      pw=pbinom(c1-1,t,r)+pbinom(c2,t,r,F)
   }; pw
} #Power in binomial distribution

f=60; t=100; Selpower(f,t); B.test(f,t,side='g') #0.6225327
f=40; t=100; Selpower(f,t); B.test(f,t,side='l') #-0.6225327
t=30; X=0:t; D=Df(X,mapply(Selpower,X,t))
colnames(D)=c('f','Selective power')
print(D[11:21,],row.names=F)

  f Selective power
  10      -0.58475960
  11      -0.43158632
  12      -0.29147186
  13      -0.17896469
  14      -0.09926032
  15       0.04936857
  16       0.09926032
  17       0.17896469
  18       0.29147186
  19       0.43158632
  20       0.58475960

G=gLines(D,lx='f', ly='Selective power', a=90)
G=G+geom_hline(yintercept=0,linetype=1,color='gray10')
G=G+geom_hline(yintercept=.8,linetype=3,color='gray10')
G=G+geom_hline(yintercept=-.8,linetype=3,color='gray10')
G=G+geom_text(x=01,y=.9,label='power: 0.8: ',color='gray10',hjust=0)
G=G+geom_text(x=01,y=-.7,label='power: -0.8: ',color='gray10',hjust=0)
G
```



●効果量

標本の比率(r=f/t)と帰無仮説の比率(e)の大きさを示す数値として，リスク比(RR=risk ratio)が使われます。

RR = r / e = f / t / e

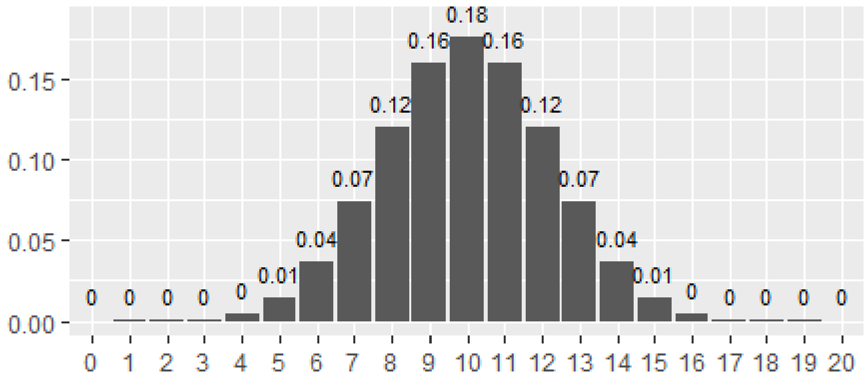
リスク比は r が e の何倍になるのか，何分の一になるのかを示す数値です。

●プログラム

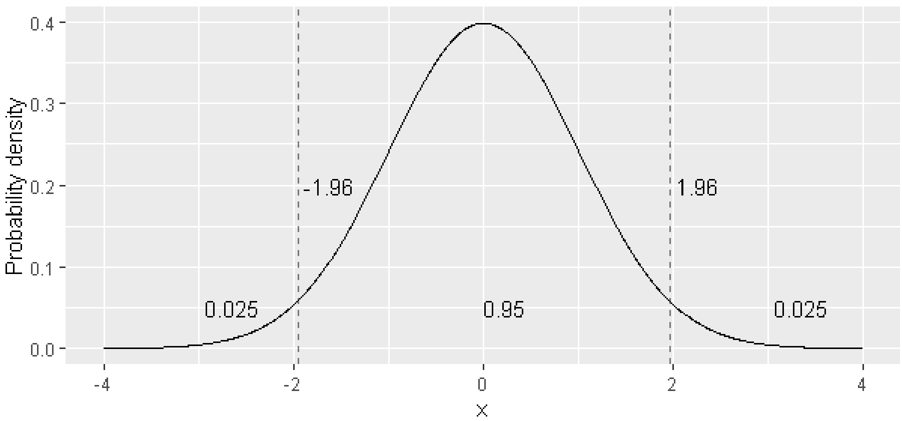
```
BinPower=function(f,t,e=.5,cl=.95,side='g'){
   r=f/t #f: freq, t:total, e:exp.probability, cl:conf.level, side:g,l,t
   if(side=='g') {c=qbinom(cl,t,e); pw=pbinom(c,t,r,F)}
   if(side=='l') {c=qbinom(1-cl,t,e); pw=pbinom(c-1,t,r)}
   if(side=='t') {
      c1=qbinom((1-cl)/2,t,e); c2=qbinom((1+cl)/2,t,e)
      pw=pbinom(c1-1,t,r)+pbinom(c2,t,r,F)
   }; pw
} #Power in binomial distribution
```

3.5.3. 信頼区間

二項検定では，1 つの P 値や検出力だけでなく，一定の幅をもつ「信頼区間」(confidence interval)も使われます。はじめに，標本数(n)が十分に大きい場合の信頼区間の求め方を説明します。次の図は二項分布の確率の分布を示します(p=0.5, n=20)。



このグラフではそれぞれの棒は離散的に示されていますが，標本数(n)を次第に大きくすると，次のような正規分布確率の連続的なグラフに次第に近づきます。



後述する「標準正規分布」に従う次の T 値の絶対値が 1.96 以内である確率の和は全体の 95%を占めます。

$$T = (M - m) / sd / \sqrt{n}$$

(M: 標本平均 x/n, m: 母集団の平均，sd: 母集団の標準偏差，n: 標本数)

$$P(|T| < 1.96) = 0.95$$

上の式の

$$|T| = |(M - m) / sd / \sqrt{n}| < 1.96$$

を m について解くと次の式が得られます。

$$[M - k] < m < [M + k], \quad k = 1.96 * sd / \sqrt{n}$$

先に見たように，二項分布の標準偏差(sd)は母比率(p)を使って次のように

求められます[25]。

sd = √[p * (1-p)]

ここで, n が十分に大きいことから母比率(p)=標本比率(x/n)と見なして, 母比率を標本比率で代用します。よって次のようにして, 母比率の安全率99%の信頼区間を求めます。

p - k < m < p + k, p = x/n, k = 1.96 * √[p*(1-p) / n]

この式は, 母集団から取り出した標本から母比率の 95%信頼区間を求めることを多数回行ったとき、その 95%程度はこの区間の中に母平均 m を含むことを表します。信頼水準を 99%に設定するときは, 1.96 の係数を 2.576 にします。上の不等式の左側の項は「下界」(lower-b.), 右側の項は「上界」(upper-b.)と呼ばれます。

以上を R を使って実験します。

```
q=qnorm(.975) #1.959964
f=300; t=1000; p=f/t; k=q*sqrt(p*(1-p)/t); p+c(-k,k)
#0.2715974 0.3284026
f=300; t=1000; binom.test(f,t,0,"t",.95)$conf.int[1:2]
#0.2717211 0.3294617
```

しかし, このように二項分布を正規分布に見立てて求めた信頼区間は, 本来の信頼区間の近似に過ぎません。たしかに, 次のようにして R の統計パッケージ *stats::binom.test* を使って求めた信頼区間と比較すると, かなり近似しているものの, 正確には一致していません。そして, 次のようにして t を小さくすると(t=10), 両者が示す信頼区間はかなり異なります。よって, 標本数(t)が小さいとき(t<30), この信頼区間は使えません。

```
q=qnorm(.975) #1.959964
f=3; t=10; p=f/t; k=q*sqrt(p*(1-p)/t); p+c(-k,k)
# 0.01597423 0.58402577
f=3; t=10; binom.test(f,t,0,"t",.95)$conf.int[1:2]
# 0.06673951 0.65245285
```

そこで, 先に説明した期待確率を求める *BinE* を使って, 二項分布→正規分布の近似法ではなく, 二項分布から信頼区間を直接に計算することを提案します。この信頼区間は, 後で確認するように, R 標準関数 binom.test()

---

[25] [0, 1]の集合の中で 1 が起こる確率を p とすると, :

P(X=1) = p, P(X=0) = 1 - p (P:確率)
E(X) = 1*p + 0*(1-p) = p (E:期待値, 平均)
E(X^2) = 1^2*p + 0^2*q = p
V(X) = E(X^2) - [E(X)]^2 = p -p^2 = p*(1-p) (V: 分散)
sd = √[p * (1-p)] (sd: 標準偏差)

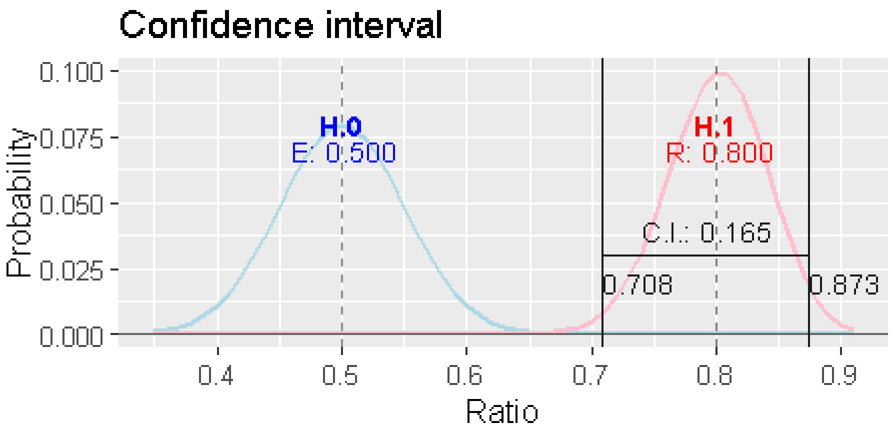
が返す信頼区間と一致します。この方法で求めた信頼区間は標本数の大小に拘わらず使用できます。

```
f=80; t=100
#(a) two-sided, conf.level=.975, .025
e=BinE(f,t,.975); e # 0.7081573 lower-bound
BinS(f,t,e) #security: 0.975
pbinom(f-1,t,e) #security: 0.975
s=BinS(f,t,e+.001); s # 0.973727 => less than 0.975

e=BinE(f+1,t,.025); e # 0.8733444 upper-bound
BinS(f+1,t,e) #security: 0.025
pbinom(f,t,e) #security: 0.025
s=BinS(f+1,t,e-.001); s # 0.02693882 => more than 0.025
binom.test(f,t,e,'t',cl)$conf.int[1:2] #0.7081573 0.8733444

#(b) greater, conf.level=.95
e=BinE(f,t,.95);e #security: s=cl:.95, lw:0.7227998
BinS(f,t,e) #security: 0.95
pbinom(f-1,t,e)   #security: 0.95
binom.test(f,t,,'g',cl)$conf.int[1:2] #0.7227998 1.0000000

#(c) less, conf.level= .05
e=BinE(f+1,t,.05); e #security:.05 up:0.8633387
BinS(f+1,t,e) #security: 0.05
pbinom(f,t,e) #security: 0.05
binom.test(f,t,e,'l',cl)$conf.int[1:2] #0.0000000 0.8633387
```

上の実験(a)では，f=80; t=100 (f:出現数，t:標本数)とします。(a)では両側の信頼区間を計算します。信頼下界(lw: lower-b.)は BinE(f,t,.975)で計算します。よって，安全率(s)=0.975 で標本数(t)=100 のうち出現数(f)=80 であったときの期待確率(e)を信頼下界(lw)とします。同様にして，信頼上界(up: upper-b.)は BinE(f,t,.025)で計算します。このとき，安全率の設定が逆のように見えますが，安全率と期待確率の大小関係は逆になるので，このように安全率を設定します。信頼下界の計算では BinE の第 1 引数は f ですが，信頼上界の場合は f+1 としてあります。この理由は安全率(BinS)の確認の後の対応する pbinom の引数を見ればわかります。信頼下界の安全率.975 は 0:(f-1)の累積確率になり，信頼上界の安全率.0255 は 0:f の累積確率になるからです。最後に BinE で求めた信頼区間と R 関数 binom.test が返す信頼区間が一致していることを確認します。

上の実験(b)では右側検定(greater)の信頼区間を求めています。このときの安全率を 0.95 とします。信頼上界は計算するまでもなく 1 となります。実験(c)で左側検定(less)の信頼区間を求めます。安全率は 0.05 とします。信頼下界は 0 とします。R 標準関数 binom.test が返す信頼区間を見ると，BinE 関数で求めた信頼区間と一致していることがわかります。

この binom.test 関数はブラックボックスになりますが，実験によって BinE で求めた信頼区間と一致することを確認したので，以下の説明では，binom.test 関数を使用します。



上の図は f=80, t=100, e=0.5, conf.level=0.95 の場合の信頼区間を示します。このように，データの比率 f/t=80/100=0.8 の 95%信頼区間[.708, .873](幅: .163)を示しています。

次は成功率=30%を保って，データの規模を 10, 100, 1000 倍にしたときの信頼区間を示します(両側検定('t': two-sided)，信頼水準: 95%)。

```
binom.test(3,10,0,'t',0.95)$conf.int[1:2]
# 0.06673951 0.65245285
binom.test(30,100,0,'t',0.95)$conf.int[1:2]
# 0.2124064 0.3998147
binom.test(300,1000,0,'t',0.95)$conf.int[1:2]
# 0.2717211 0.3294617
```

上の結果から，データの規模(f, t)の上昇が信頼区間の幅を狭めていくことがわかります。信頼区間の幅が狭いほど，成功率の信頼性は高まります。

次は信頼水準を 95%と 99%としたときの信頼区間を示します。

```
A=binom.test(30,100,0,'t',0.95)$conf.int[1:2]; c(A[1],A[2],A[2]-A[1])
# 0.2124064 0.3998147 0.187408
A=binom.test(30,100,0,'t',0.99)$conf.int[1:2]; c(A[1],A[2],A[2]-A[1])
# 0.1890148 0.4306136 0.2415988
```

このように，信頼水準を上げると信頼区間の幅は広くなります(幅: .187 → .242)。信頼区間の幅が広いときはその比率の信頼性が低いことを意味します。このように，信頼水準を上げると検定の水準を厳しく設定するので，幅が広い信頼区間を使って信頼性を低くして見積もることになります。

二項検定では P 値のほかに，たとえば，「95%信頼区間=[0.068, 0.652](幅:0.584)」のようにして結果を示すとよいでしょう。P 値はポイント(点)で評価し，信頼区間は幅(線)で評価するので両者は相補的な関係になります。

●プログラム

二項検定を実行するするために次の自作関数 B.test.b では自作 BinE を使って信頼区間を計算しています。

```
f=62;t=100;e=0.5;side='t'; B.test.b(f,t,e,side); binom.test(f,t,e,side)

       #ratio  effect-size     security      p-value         beta
        0.620        1.240        0.979        0.021        0.376
        power          cv1          cv2     lower-b.     upper-b.
        0.624       40.000       60.000        0.517        0.715
        width confidence
        0.198        0.775

binom.test(f,t,e,side)

        Exact binomial test

data:   f and t
number of successes = 62, number of trials = 100,
p-value = 0.02098
alternative hypothesis: true probability of success is not equal to 0.5
95 percent confidence interval:
 0.5174607 0.7152325
sample estimates:
probability of success
                  0.62
```

上では最初に B.test.b を使って，各種の検定量を出力し，次に R 標準関数 binom.test を使った結果を示しています。P 値と信頼区間は小数点以下の桁数が異なりますが，実質的に一致しています。B.test.b は，安全率(security)と信頼区間の幅(width)も加えて出力します。2 項検定の信頼区間は比率の幅を示すので，その最大値は 1 です。よって，幅の範囲は[0, 1]となるので，そのまま相対的な指標として使用できます。暫定的にその基準値を 0.2 とすれば(ガイドライン), 基準値 0.2 以下であれば信頼区間の「信頼性」は比較的高い，と評価できます。

最後に，安全率(sc)・検出力(pw)・信頼区間の幅(wd)の補数(1-wd)の調和平均(harmonic mean)を求め，これを「信頼性」(cf: confidence)として出力します。頻度や試験の点数などのように，同じ単位・性質をもつ集団であれば算術平均が使えますが, 安全率(sc)・検出力(pw)・信頼区間の幅は確率・比率であり，単位はありません。そのような数値の平均として調和平均が適しています。たとえば，時速 40 km/h と時速 100 km/h の平均値を

(40+100)/2=70 (km/h)とするわけにはいきません。以下にその理由を説明します。距離=k，時間=h1, h2 とすると，それぞれの時速は k/h1=40, k/h2=100 となります。全体の走行距離は 2*k です。よって，2*k/(h1+h2)が平均時速となります。

$$k/h1 = 40,\ k/h2 = 100 \rightarrow k = 40*h1,\ k = 100*h2 \rightarrow h1=k/40,\ h2=k/100$$

$$2*k / (h1+h2) = 2*k / (k/40+k/100) = 2 / (1/40 + 1/100) = 57.14286$$

上の式を一般化すると, 調和平均の式は n / Σ(i=1:n) [1/x(i)]となります。R で関数化すると(Hm),

```
Hm = function(A) length(A)/sum(1/A)    #harmonic mean Hm(c(40,100))
```

先の例の調和平均値(57.1 km/h)は単純な算術平均値(70 km/h)よりかなり小さくなっています。とくに，c(40, 100)のように大きな偏差(平均からの距離)があるベクトルでは調和平均値が小さくなります。算術平均値は大きな値と小さな値が互いに補う合う数値になって相殺され，一方が非常に大きな値であれば，他方が小さくても平均値がある程度大きくなります。一方、調和平均は全体の偏差が大きいときに低く抑えられ，全体の偏差が小さいときに高くなります。偏差が小さくなると調和平均は算術平均に近くなり，c(40, 40, 40)のように，すべて同数であれば，算術平均と一致して 40 になります。引数の偏差が小さい，ということは引数の間でバランスが保たれていることを示すので信頼性が高まる，と考えられます。そこで，安全率(sc)・検出力(pw)・信頼区間の幅(補数: 1-wd)がそれぞれバランスよく高い数値であることがデータの高い信頼性を示していることになるので，3 者の調和平均を使用します。次が B.test の中の「信頼性」(cf)を導く式です。

```
cf = Hm(c(abs(sc), abs(pw), (1-wd)))
```

その第 1,2 引数(sc, pw)に絶対値(abs)を使う理由は選択的安全率(sc)・選択的検出力(pw)が負になることがあるからです。信頼区間の幅(wd)の補数(1-wd)を使う理由は，信頼区間の幅が小さいほど信頼性が増すためです。

次は内部で R 関数 binom.test を使用した自作関数 B.test を使った結果です。

```
f=62;t=100;e=0.5;side='t'; B.test.b(f,t,e,side)

    #ratio  effect-size    security     p-value        beta
     0.620        1.240       0.979       0.021       0.376
     power          cv1         cv2    lower-b.    upper-b.
     0.624       40.000      60.000       0.517       0.715
     width significance
     0.198        0.775
```

このように，*B.test.b* と *B.test* は同じ結果を示します。こうして，pbinom と BinE を使って求めた P 値と信頼区間が binom.test と同じ結果となることを確かめたので，これからは後者を使用します。(1), (2)は選択的安全率・選択的検出力を使った検定です。(1)は f >= 2/t の場合，(2)は f < t/2 の場合です。(3)は両側検定(t: two sided)，(4)は右側検定(g: greater)，(5)は左側検定(l: less)の結果です。両側検定は右側検定よりも安全率(security)，P 値(p-value)，信頼区間の幅(width)，検出力(power)において，すべて厳しい評価になっています。

●選択的二項検定

「二項検定」を実行する自作関数 B.test は「右側検定」「左側検定」「両側検定」のほかに「選択的検定」(selective test)を含みます。選択的検定の安全率と検出力として先に扱った「選択的安全率」と「選択的検出力」を採用します。信頼区間は検定の種類(右側・左側・両側)によって異なります。右側検定の信頼区間の上界は常に最大の 1 になるため，信頼区間の幅が非常に大きくなります。逆に，下側検定の信頼区間の下界は常に最小の 0 になるため，信頼区間の幅がやはり非常に大きくなります。両側検定の信頼区間は上界と下界の数値がともに確定されて信頼区間は狭まります。数値の信頼区間の幅が狭いときは，その数値に信頼性がある，と考えられるので，極端に広がる片側検定の信頼区間を使うことができません。とくに，ゼロや 1 に近いときの確率は非常に小さいので，そのような確率に対応する数値までも含めることは不都合です。そこで選択的検定の信頼区間は両側検定の信頼区間を採用します。

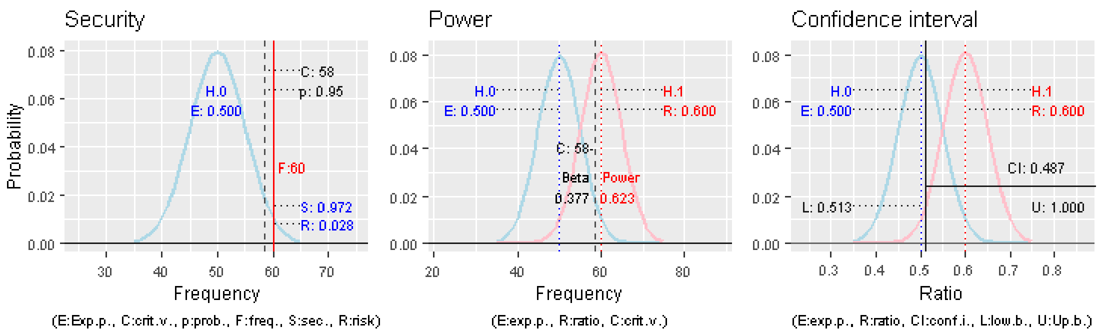
次は，(1)片側検定(右側), (2)片側検定(左側), (3)両側検定, (4)選択的検定(右側), (5)選択的検定(左側)の結果です。

```
> t=100; e=0.5

> f=60; s='g'; B.test(f,t,e,s) #(1) Unilateral test (greater)
ratio effect-size    security     p-value        beta       power
0.600       1.200       0.972       0.028       0.377       0.623
cv1          cv2    lower-b.    upper-b.       width   confidence
0.000      58.000       0.513       1.000       0.487       0.654

> f=42; s='l'; B.test(f,t,e,s) #(2) Unilateral test (less)
ratio effect-size    security     p-value        beta       power
0.420       0.840       0.933       0.067       0.538       0.462
cv1          cv2    lower-b.    upper-b.       width   confidence
42.000     100.000       0.000       0.507       0.507       0.570

> f=60; s='t'; B.test(f,t,e,s) #(3) Bilateral test (two-sided)
ratio effect-size    security     p-value        beta       power
0.600       1.200       0.943       0.057       0.538       0.462
```

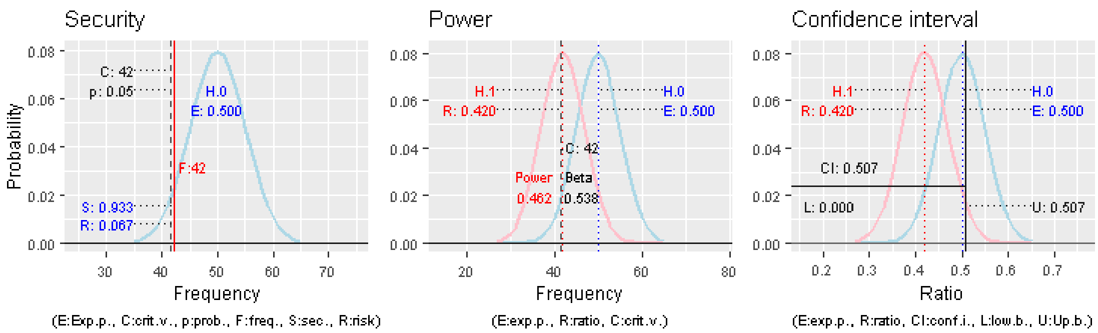
```
cv1          cv2     lower-b.     upper-b.       width   confidence
40.000       60.000        0.497        0.697       0.199        0.671

> f=67; s='s'; B.test(f,t,e,s) #(4) Selective test (greater)
ratio effect-size     security      p-value         beta        power
0.670        1.340        1.000        0.000       0.037        0.963
cv1          cv2     lower-b.     upper-b.       width   confidence
0.000       58.000        0.569        0.761       0.192        0.916

> f=42; s='s'; B.test(f,t,e,s) #(5) Selective test (less)
ratio effect-size     security      p-value         beta        power
0.420        0.840       -0.933        0.067       0.538       -0.462
cv1          cv2     lower-b.     upper-b.       width   confidence
42.000      100.000        0.322        0.523       0.201        0.668
```

t=100; e=0.5

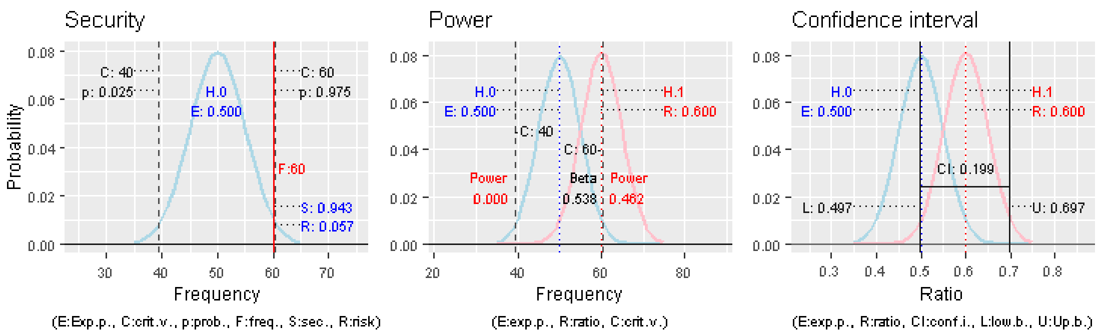
f=60; s='g'; B.test(f,t,e,s) #(1) Unilateral test (greater)



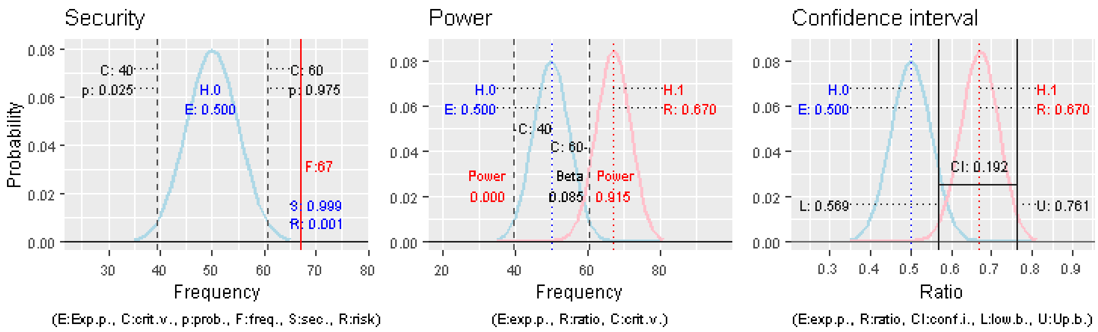
f=42; s='l'; B.test(f,t,e,s) #(2) Unilateral test (less)



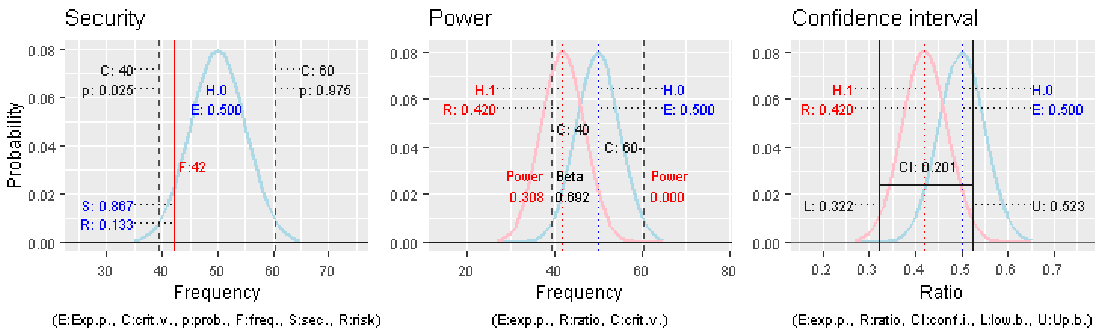
f=60; s='t'; B.test(f,t,e,s) #(3) Bilateral test (two-sided)



f=67; s='s'; B.test(f,t,e,s) #(4) Selective test (greater)



f=42; s='s'; B.test(f,t,e,s) #(5) Selective test (less)

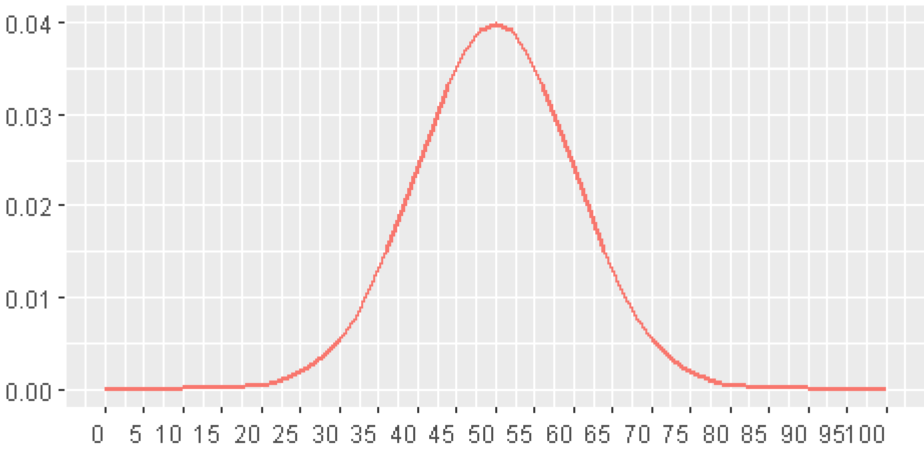


3.6. 正規分布

二項分布確率は頻度・比率(割合・パーセント・パーミル)の統計的検定に使われます。一方，この章で扱う「正規分布確率」(probability of normal distribution)は 1 つの平均値や 2 つの平均値の差の統計的検定に使われます。頻度・比率・平均値はどれも使われる機会が多いので重要です。

はじめに正規分布の確率密度の動きを観察しましょう。

```
m=50; sd=10 #mean; standard deviation
NormD=function(x,m,sd) (1/(sd*sqrt(2*pi)))*exp(-((x-m)^2)/(2*sd^2))
#Normal dist. density
NormD(65,m,sd); dnorm(65,m,sd) #confirm
B=seq(0,100,by=2); D=data.frame(B, dnorm(B,m,sd)); D #breaks, density
gLines(D,brk=seq(0,100,by=5)) #graph
```



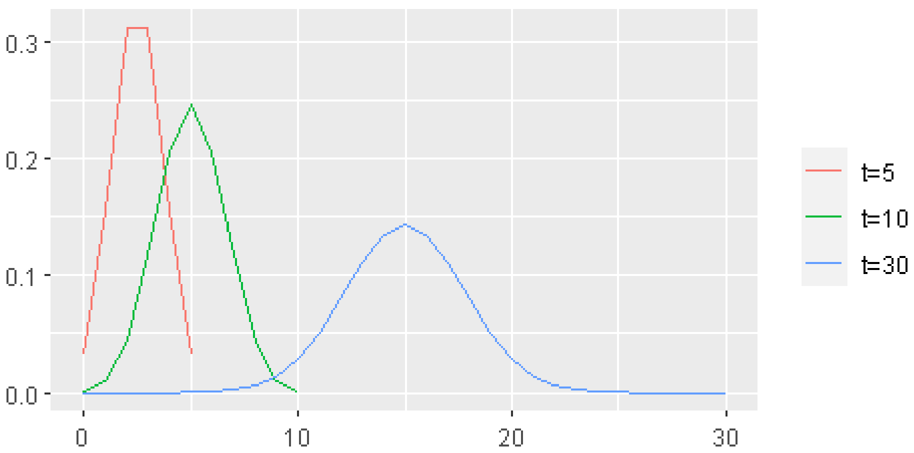
上のコードで示したように,正規分布確率密度は次の式で定義されます:

(1 / (sd * sqrt(2 * pi))) * exp(-((x-m)^2) / (2 * sd^2))

ここで sd は標準偏差値，sqrt は平方根，pi は円周率，exp は指数，x は x 軸の数値， m は平均値です。上で示した正規分布確率密度を返す自作関数 NormD は x， m, sd を引数とします。これは R 関数 dinorm と同じ確率を返します：0.01295176．次に，[0，100]の範囲で 2 ごとに分割したベクトル B とそれに対応する正規分布確率密度からなるデータフレーム D を用意し，自作関数 gLines でグラフを出力します。

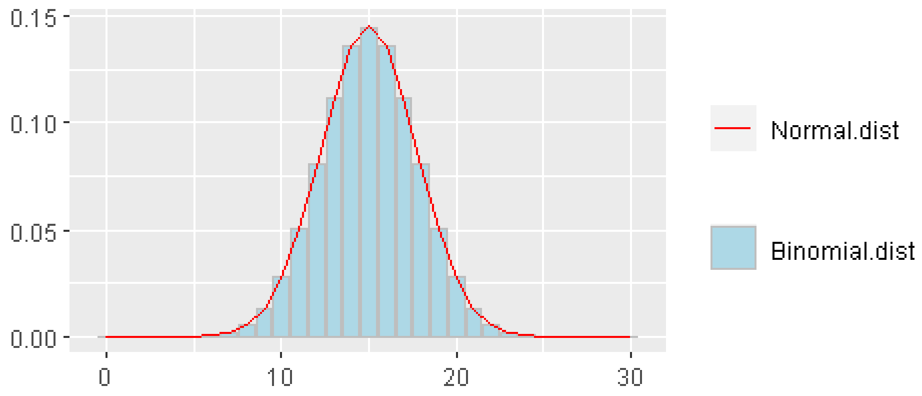
正規分布確率密度の上の式を理解することは困難ですが，ここで上のグラフを見て二項分布確率密度の形とよく似ていることを確認することが重要です。実際に，次の実験によれば，和(t)を大きくするごとに二項分布確率密度に次第に近づくことがわかります。

```
t=5;    D1=data.frame(0:t, dbinom(0:t,t,0.5),'t='&t); D1
t=10; D2=data.frame(0:t, dbinom(0:t,t,0.5),'t='&t); D2
t=30; D3=data.frame(0:t, dbinom(0:t,t,0.5),'t='&t); D3
D=rbind(D1,D2,D3); D; gLineR(D) # graph
```



上の実験によれば，和 t=30 で完全に近いくらい二項分布確率密度と正規分布確率密度が一致しています。そこで，t=30 以上であれば，正規分布確率を二項分布確率の近似値として使うことができます。実際に，次のように，二項分布確率密度と正規分布確率密度をそれぞれ棒グラフと線グラフにして比較すると，一致していることがわかります。

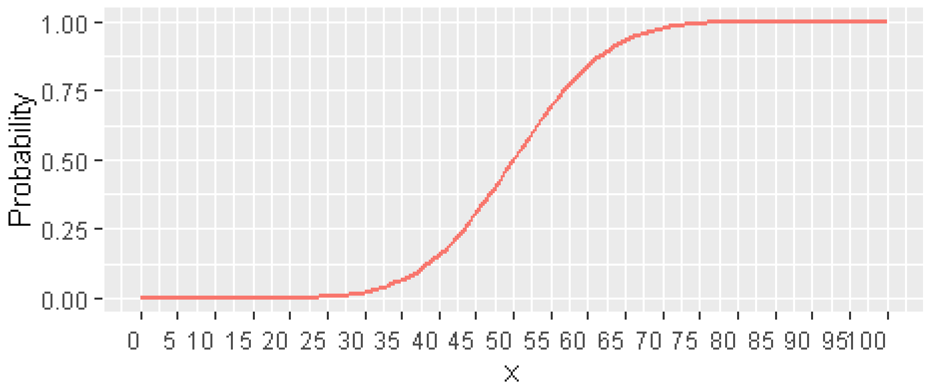
```
t=30; e=.5; X=0:t; A=dbinom(f,t,e); B=dnorm(f,t*e,sqrt(t*e*(1-e)))
D=data.frame(f,Binomial.dist=A,Normal.dist=B); round(D,4)
gBarLine(D) # graph
```



上の A は f:頻度，t:和，e:期待確率で定義される二項分布確率密度を示します。一方，B は f:頻度，m:平均，sd:標準偏差で定義される正規分布確率密度です。平均値は m=t*e，標準偏差値は sqrt(t*e*(1-e))で計算することができます(多くの統計学書に証明が載せられています)。そうすると，上図が示すように，二項分布確率密度と正規分布確率密度が一致します。

次に，正規分布確率を使って，数値(f)の安全率・危険率(P 値)を計算します。はじめに，次の実験で R 関数 pnorm を使って累積正規分布確率の動きを見ます。

```
t=100; X=0:t; D=data.frame(x=X, p=pnorm(X,50,10)); D
gLines(D,brk=seq(0,t,5))
```



このように，累積正規分布確率は滑らかな「S 字型」カーブを描きます。

このカーブの特徴は x が平均値(50)の近くで大きく変化し，平均値から離れるに従って，変化が少なくなることです。

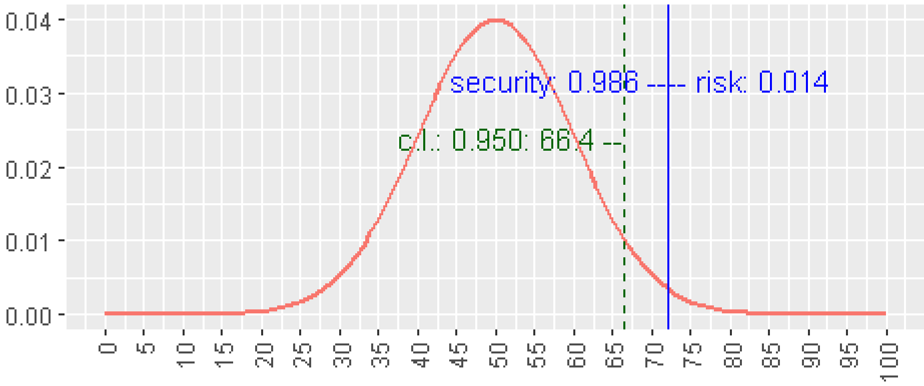
この R 関数 pnorm を使って安全率(security)を計算します。

```
f=72; m=50; sd=10
sc=pnorm(f,m,sd); sc # security 0.9860966
p=1-sc; p # p-value, risk 0.01390345
qnorm(sc,m,sd) # qnorm is inverse function of pnorm 72
qnorm(.95,m,sd) # 66.44854
qnorm(.99,m,sd) # 73.26348
```

ここでは,平均値 50，標準偏差値 10 のときの 72 という数値の安全率(sc)は 98.6%，危険率(p)は 1.4%であることがわかりました。これは非常に高い数値であることを示しています。次に pnorm の逆関数 qnorm を使えば，安全率 0.9860966 のときの数値(72)が求められます。さらに，qnorm を使って安全率 95%の数値(66.4)と，安全率 99%の数値(73.3)が求められます。

R 関数 pnorm と qnorm を使って，次のグラフを出力します。

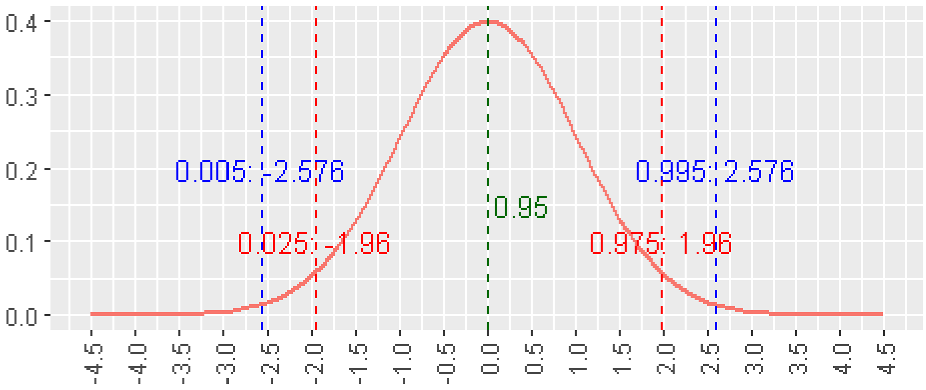
```
qcl=qnorm(0.950,m,sd); qcl #qcl: 66.44854
pcl=pnorm(qcl,m,sd); pcl # pcl: cummulative-prob. = 0.95
B=seq(0,100,1); R=dnorm(B,m,sd); R
D=data.frame(B,R); D
m=max(D[,2])
gLines(D,'Normal dist.',brk=seq(0,100,5),a=90)+
   geom_vline(xintercept=f, linetype=1, color="blue")+
   geom_vline(xintercept=qcl, linetype=2, color="darkgreen")+
   geom_text(aes(x=f,y=.8*m, label='security: '&Fm(sc,3)&' --'),
                  color="blue",hjust=1)+
   geom_text(aes(x=f,y=.8*m, label='-- risk: '&Fm(1-sc,3)),
                  color="blue", hjust=0)+
   geom_text(aes(x=qcl,y=.6*m, label='c.l.: '&Fm(pcl,3)&': '
                        &round(qcl,1)&' --'), color="darkgreen",hjust=1)
#s: security, r: risk (p-value), c.l. confidence level
```



グラフの c.l は信頼水準(confidence level)を示します。信頼水準を 95%とすれば，それに対応する数値は 66.4 なので，この数値以上であれば，安全率は 95%以上であることと示しています。この例では，数値は 72 なので信頼水準を超えています。

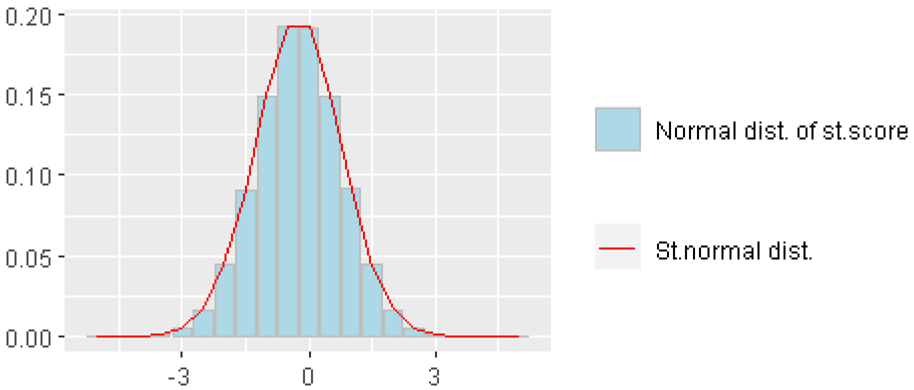
平均値(m)と標準偏差(sd)で定義される正規分布確率と並んで，平均値=0，標準偏差=1 とした「標準正規分布確率」(standard nomal probability)もさまざまな統計分析で使われます。

```
q1=qnorm(0.025); q1 #q1: inverse fun. of pnorm = -1.959964
p1=pnorm(q1); p1 # p1: cummulanormive-prob. = 0.025
q2=qnorm(0.975); q2 #q2: inverse fun. of pnorm = 1.959964
p2=pnorm(q2); p2 # p2: cummulanormive-prob. = 0.975
q991=qnorm(0.005); q991 #q1: inverse fun. of pnorm = -2.575829
p991=round(pnorm(q991),3); p991 # p1: cummulative norm.prob. = 0.005
q992=qnorm(0.995); q992 #q2: inverse fun. of pnorm = 2.575829
p992=round(pnorm(q992),3); p992 # p2: cummulative norm.prob. = 0.995
B=seq(-4.5,4.5,.1); Dn=dnorm(B); D=data.frame(B,Dn); D #breaks, normal prob. density
gLines(D,'St. norm. dist.',brk=seq(-4.5,4.5,.5), a=90)+
   geom_vline(xintercept=q1, linetype="dashed", color="red")+
   geom_vline(xintercept=q2, linetype="dashed", color="red")+
   geom_vline(xintercept=q991, linetype="dashed", color="blue")+
   geom_vline(xintercept=q992, linetype="dashed", color="blue")+
   geom_vline(xintercept=0, linetype="dashed", color="darkgreen")+
   geom_text(aes(x=q1,y=.1, label=p1&': '&round(q1,3)), color = "red")+
   geom_text(aes(x=q2,y=.1, label=p2&': '&round(q2,3)), color = "red")+
   geom_text(aes(x=q991,y=.2, label=p991&': '&round(q991,3)), color = "blue")+
   geom_text(aes(x=q992,y=.2, label=p992&': '&round(q992,3)), color = "blue")+
   geom_text(aes(x=0,y=.15, label='0.95'), color = "darkgreen",hjust=-0.1)
```



この標準正規分布確率密度は，正規分布をするベクトル A の平均(m)と標準偏差(s)を使って，(A-m)/s のように標準化したベクトルの確率分布です(FD1)。この標準化ベクトルは，平均(m)=0 と標準偏差(s)=1 を使って発生させた乱数の確率分布と一致します。このことは以下のグラフで確認できます。

```
B=seq(-5,5,.5)
set.seed(1); A=rnorm(10^5,50,10); m=mean(A); s=sd(A); A=(A-m)/s
mean(A); sd(A) #-0.000000000000000002125033, 1
FD1=FreqDistV(A,B,T); FD1
A=rnorm(10^5,0,1); FD2=FreqDistV(A,B,T); FD2
D=cbind(FD1,FD2); D=D[,-3]
colnames(D)=V('x,Normal dist. of st.score, St.normal dist.'); D
gBarLine(D)
```



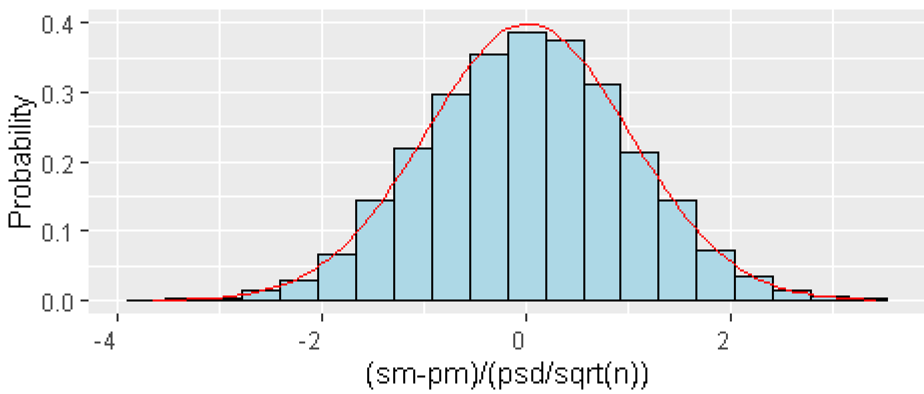
次の実験で，統計学の重要な定理「中心極限定理」(central limit theorem)が実現されることを確認します。

中心極限定理： 母平均 pm，母標準偏差 psd の任意の母集団について，標本サイズ n が大きいとき， z = (sm-pm)/(psd/sqrt(n)) は近似的に標準正規分布 N(0,1)に従う。

ここで「任意の母集団」ということは，どのような分布の母集団でも，その平均と標準偏差がわかれば，そこから抽出したサンプルの平均(sm)を含む z 値は標準正規分布ということなので，この定理を利用すれば，サンプルの平均の有意性・信頼性を検定することが可能になります。この中心極限定理が示すことを次の一様乱数実験で確かめましょう。

```
set.seed(1); R=runif(10^6,min=0,max=100); R[1:20] # random uniform-dist.U(0,100)
pm=mean(R); psd=sd(R); n=30; ns=10^4; Z=NULL
#pm:pop.mean; psd:pop.sd; n:sample size; ns:number of samples; Z: vector
for(i in 1:ns){
   S=sample(R,size=n,replace=T) #random sampling
   Z[i]=(mean(S)-pm)/(psd/sqrt(n)) # z-value
}
Z[1:10]; min(Z); max(Z)
gNormal(Z,lx='(sm-pm)/(psd/sqrt(n))')
#Graph for conformation to standard normal dist.
```

はじめに，R 関数 runif で 100 万個(10^6)の 0~100 の範囲の一様分布の乱数ベクトル R を生成します。このベクトルを母集団として，その母平均(pm)，母標準偏差(psd)を求めます。n は標本サイズ，ns は計算する z 値の数になります。for 文の中で，サイズ n=30 の標本ベクトル S を sample 関数で抽出し，その S の z 値を Z ベクトルに代入します。最後に gNormal でグラフを出力させると，その確率分布は次のように標準正規分布を示します(棒グラフ: z 値の分布，線グラフ: 標準正規分布)。



このようにして確認された「中心極限定理」を使って，「1 標本の平均の検定」を実行します（後述）。

次の統計値 z2 も標準正規分布 N(0,1)に従います[26]。

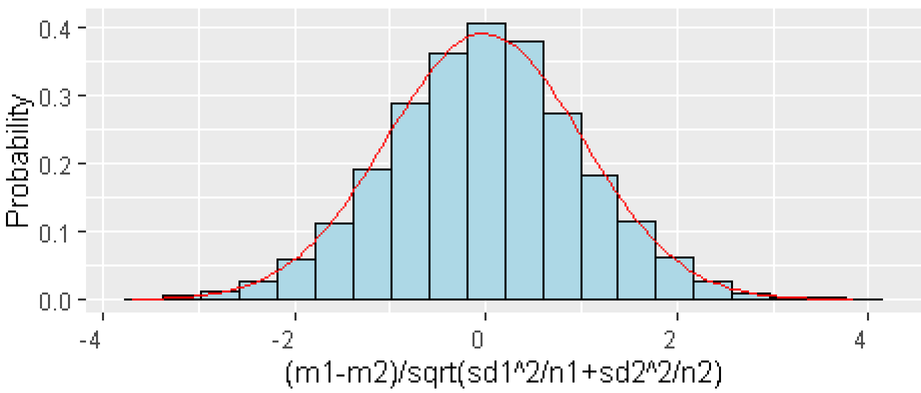
[26] Kin (2009)『テキストデータの統計科学入門』東京：岩波書店, p.121.

z2 = (m1-m2) / sqrt(sd1^2/n1 + sd2^2/n2)

(m1: 標本-1 の平均, sd1: 標本-1 の標準偏差, n1: 標本-1 のサイズ)
(m2: 標本-2 の平均, sd2: 標本-2 の標準偏差, n2: 標本-2 のサイズ)

この事実を次の実験で確かめましょう。

```
set.seed(1); R1=runif(10^6,min=0,max=100); R1[1:20] # 1.random uniform-dist.U(0,100)
set.seed(2); R2=runif(10^6,min=0,max=100); R2[1:20] # 2.random uniform-dist.U(0,100)
#pm:pop.mean; psd:pop.sd; n:sample size; ns:number of samples; A: vector
ns=10^4; n1=5; n2=5; Z=NULL
ns=10^4; n1=25; n2=25; Z=NULL
for(i in 1:ns){
   S1=sample(R1,size=n1,replace=T) #random sampling.1
   S2=sample(R2,size=n2,replace=T) #random sampling.2
   m1=mean(S1); sd1=sd(S1); m2=mean(S2); sd2=sd(S2)
   Z[i]=(m1-m2)/sqrt(sd1^2/n1+sd2^2/n2) # z-value
}
Z[1:10]; min(Z); max(Z)
#NormDistConform(Z,.5) #Mean of difference
#gNormDistConform(Z,.5,lx='(m1-m2)/sqrt(sd1^2/n1+sd2^2/n2)')
gNormal(Z,lx='(m1-m2)/sqrt(sd1^2/n1+sd2^2/n2)')
#Graph for conformation to standard normal dist.
```



このように，棒グラフが示す z 値の分布と線グラフが示す標準正規分布がほぼ一致します。z2 値の分子が(m1-m2)なので，標準正規分布を使って 2 つの標本の平均値の差を検定します(後述：「1 標本の平均の検定」)。

●Z 検定の準備

標本の平均値を評価する Z 検定では，(1)安全率・危険率(P 値)，(2)検出力，(3)信頼区間を分析します。はじめに，(1)安全率・危険率(P 値)を次の例を使って説明します。

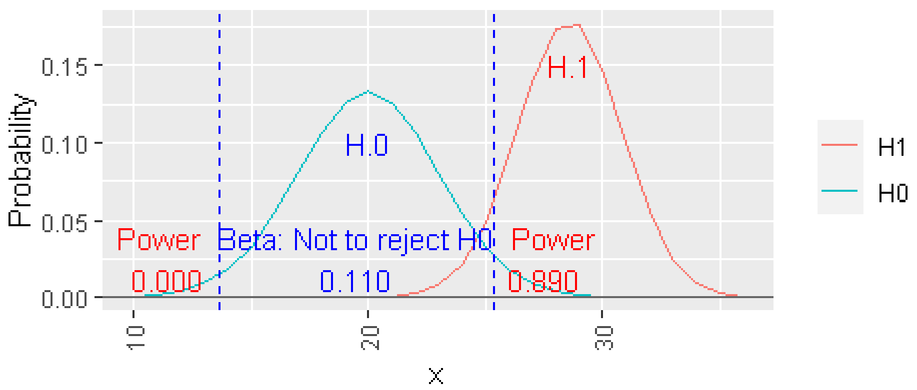
```
A=c(25,28,30,29,32,27,31,26,28,30); pm=26; psd=3; cl=.95; r=4
sm=mean(A); ssd=sd(A); n=length(A); sm; ssd; n #sm:28.6, ssd=2.22, n=10
z=(sm-pm)/(psd/sqrt(n)) #z-value
#(1) two-sided
2*pnorm(-abs(z));                    z.test(A,NULL,'t',pm,psd,NULL,cl)$p.value
#p:0.006131953
#(2) greater
1-pnorm(z); z.test(A,NULL,'g',pm,psd,NULL,cl)$p.value #p: 0.003065977
#(3) less
pnorm(z); z.test(A,NULL,'l',pm,psd,NULL,cl)$p.value #p:0.996934
```

検定の対象はベクトル A の標本平均値(sm)です。目的はこの標本平均値(sm)が母集団の平均値(pm)と比べて，有意に異なるか(両側検定: two-sided)，有意に大きいか(右側検定: greater)，有意に小さいか(左側検定: less)を判断します。両側検定の P 値は 2*pnorm(-abs(z))で求めます。z 値に対応する標準正規分布累積確率で左側の値を求めるために，z 値の絶対値を abs で計算し，改めてマイナス(-)をつけて左側の累積確率を求め，両側にするため 2 倍にします。右側検定では全体の確率 1 から z に対応する累積確率(z までの確率の和)を引いて，右側の確率の和を求めます。左側検定では，単純に z に対応する累積確率(z までの確率の和)を P 値とします。以上の計算結果を R 関数 z.test が返す P 値(p.value)と比較して確認します。二項分布確率を使った P 値では，当該の数値を含めるか否かを指定しなければなりませんが，正規分布は連続しているので，pnorm が返す値をそのまま使用できます。

(2a) 検出力 (両側検定)

```
A=c(25,28,30,29,32,27,31,26,28,30); pm=20; psd=3; cl=.95; r=4
sm=mean(A); ssd=sd(A); sm;ssd #sm:28.6,ssd=2.221111
cv1=qnorm((1-cl)/2,pm,psd); cv1 #cv1: lower critical value: 14.12011
cv2=qnorm((1+cl)/2,pm,psd); cv2 #cv2: upper critical value: 25.87989
pw1=pnorm(cv1,sm,ssd); pw2=pnorm(cv2,sm,ssd,F); pw1; pw2
#0.0000000000353389; 0.8896485
pw=pw1+pw2; pw #pw: 0.8896485
b=1-pw; b # beta: 0.1103515
X=10:36; P1=dnorm(X,sm,ssd); P2=dnorm(X,pm,psd) #probability density
```

```
D=data.frame(x=rep(X,2),prob=c(P1,P2),
                          Dist=rep(c('H1','H0'),each=length(X)));
x1=cv1-.5; x2=cv2-.5 #points of critical value
gLineR(D,lx='x',ly='Probability')+
    scale_fill_manual(values=c('H1'='steelblue','H0'='gray60'))+
    theme(axis.text.x=element_text(angle=90,hjust=1,vjust=.5))+
    geom_hline(yintercept=0,linetype=1,color='gray40')+
    geom_vline(xintercept=x1,linetype=2,color='blue')+
    geom_vline(xintercept=x2,linetype=2,color='blue')+
    geom_text(x=pm,y=0.1,label='H.0',color='blue',hjust=.5)+
    geom_text(x=sm,y=0.15,label='H.1',color='red',hjust=.5)+
    geom_text(x=(x1+x2)/2,y=0.025,label='Beta: Not to reject H0¥n'&Fm(b,3),
                      color='blue',hjust=.5)+

geom_text(x=x1,y=0.025,label='Power¥n'&Fm(pw1,3),color='red',hjust=1.2)+

geom_text(x=x2,y=0.025,label='Power¥n'&Fm(pw2,3),color='red',hjust=-0.2)
```

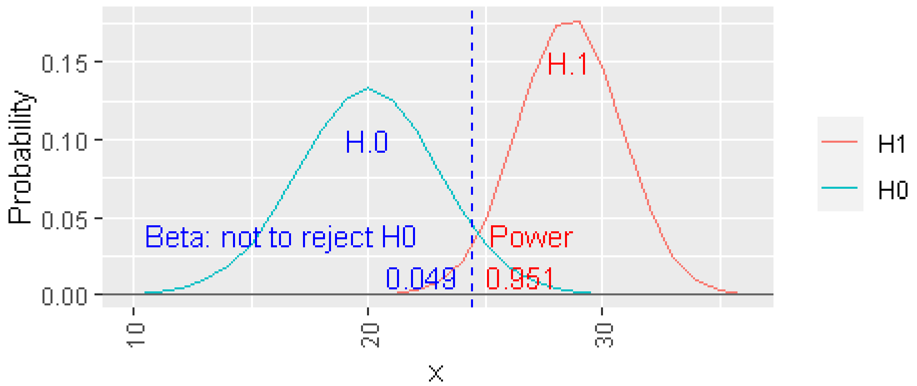


上のグラフを見ると，母集団の帰無仮説(H.0)で棄却されない領域で対立仮説(H.1)で実現されている領域(Beta)が 0.110 であり， これは第二種の誤りを犯す確率になります。この領域では帰無仮説(H.0)で棄却されず，よって対立仮説の成立が可能なのに, 対立仮説が採択されないことになります。その補数が検出力(Power)であり， 89%です。左側の検出力の領域は非常に小さく， 小数点以下 3 桁では表示されません(0.0000000000353389)。

(2b) 検出力 (右側検定)

```
A=c(25,28,30,29,32,27,31,26,28,30); pm=20; psd=3; cl=.95; r=4
sm=mean(A); ssd=sd(A); sm; ssd #sm:28.6,ssd=2.221111
cv=qnorm(cl,pm,psd); cv #critical value 24.93456
b=pnorm(cv,sm,ssd); b #beta: 0.04944356
pw=1-b; pw #power 0.9505564
```

```
X=10:36; P1=dnorm(X,sm,ssd); P2=dnorm(X,pm,psd) #probability density
D=data.frame(x=rep(X,2),prob=c(P1,P2),
                    Dist=rep(c('H1','H0'),each=length(X)))
x1=cv-.5 #point of critical value
gLineR(D,lx='x',ly='Probability')+
   labs(x='x',y='Probability')+
   scale_fill_manual(values=c('H1'='steelblue','H0'='gray60'))+
   theme(axis.text.x=element_text(angle=90,hjust=1,vjust=.5))+
   geom_hline(yintercept=0,linetype=1,color='gray40')+
   geom_vline(xintercept=x1,linetype=2,color='blue')+
   geom_text(x=pm,y=0.1,label='H.0',color='blue',hjust=.5)+
   geom_text(x=sm,y=0.15,label='H.1',color='red',hjust=.5)+
   geom_text(x=x1,y=0.025,label='Beta: not to reject H0¥n'&Fm(b,3),
                  color='blue',hjust=1.2)+
   geom_text(x=x1,y=0.025,label='Power¥n'&Fm(pw,3),color='red',hjust=-0.2)
```

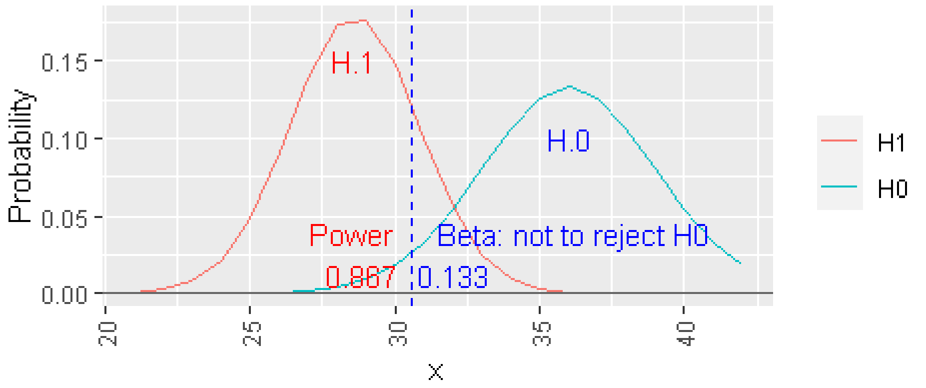


右側検定の結果は同じデータを使ったため結果が類似しています。違いは右側検定の場合のほうが検出力が高くなっていることです。このように両側検定の検出力は片側検定よりも厳しくなります。

(2c) 検出力 (左側検定)

```
A=c(25,28,30,29,32,27,31,26,28,30); pm=36; psd=3; cl=.95; r=4
sm=mean(A); ssd=sd(A); sm;ssd #sm:28.6,ssd=2.221111
cv=qnorm(1-cl,pm,psd); cv #critical value 31.06544
pw=pnorm(cv,sm,ssd); pw #power: 0.8665011
b=1-pw; b #beta: 0.1334989
X=21:42; P1=dnorm(X,sm,ssd); P2=dnorm(X,pm,psd) #probability density
D=data.frame(x=rep(X,2),prob=c(P1,P2),
                    Dist=rep(c('H1','H0'),each=length(X)))
x1=cv-.5 #point of critical value
gLineR(D,lx='x',ly='Probability')+
```

```
    labs(x='x',y='Probability')+
    scale_fill_manual(values=c('H1'='steelblue','H0'='gray60'))+
    theme(axis.text.x=element_text(angle=90,hjust=1,vjust=.5))+
    geom_hline(yintercept=0,linetype=1,color='gray40')+
    geom_vline(xintercept=x1,linetype=2,color='blue')+
    geom_text(x=pm,y=0.1,label='H.0',color='blue',hjust=.5)+
    geom_text(x=sm,y=0.15,label='H.1',color='red',hjust=.5)+
    geom_text(x=x1,y=0.025,label='Beta: not to reject H0¥n'&Fm(b,3),
              color='blue',hjust=-0.1)+
    geom_text(x=x1,y=0.025,label='Power¥n'&Fm(pw,3),color='red',hjust=1.2)
```



左側検定の検出力は対立仮説の確率分布が左側にあるのが普通で，このとき，帰無仮説(H0)の棄却域が左側にあり，その右側が Beta の領域になります。Beta の領域では帰無仮説が棄却されず，対立仮説が採択されないにも拘わらず，対立仮説が成立する可能性が 13.3%あることを示しています(第二種の誤り)。ここでは検出力(正しく対立仮説の可能性を認める確率)は 86.7%であることを示しています。

(3) 信頼区間

```
A=c(25,28,30,29,32,27,31,26,28,30); pm=26; psd=3; cl=.95; r=4
sm=mean(A); n=length(A); se=psd/sqrt(n); sm; n; se
#sm:28.6, n:10, se:0.9486833
#(1) two-sided
q=qnorm((1+cl)/2); em=q*se; c(sm-em, sm+em)#c.i.:26.74061 30.45939
z.test(A,NULL,'t',pm,psd,NULL,cl)$conf.int[1:2]#c.i.:26.74061 30.45939
#(2) greater
q=qnorm(cl); em=q*se; c(sm-em,Inf)#c.i.:27.03955 Inf
z.test(A,NULL,'g',pm,psd,NULL,cl)$conf.int[1:2]#c.i.:27.03955 NA
#(3) less
q=qnorm(cl); em=q*se; c(-Inf,sm+em)#c.i.:-Inf 30.16045
z.test(A,NULL,'l',pm,psd,NULL,cl)$conf.int[1:2]#c.i.:NA 30.16045
```

はじめに，例として標本のベクトル A を用意し，母平均(pm)，母標準偏差(psd)，信頼水準(cl)を設定します。次に標本平均(sm)，標本標本(ssd)，標本サイズ(n)を計算します。se=psd/sqrt(n)は標準誤差(standard error)と呼ばれます。両側検定(two-sided)の信頼区間は確率 0.975 に対応する数値 q に標準誤差を掛けて誤差範囲(em: error margin)を計算し，標本平均±誤差範囲を信頼区間とします。右側検定(greater)と左側検定では，信頼水準(0.95)に対応する数値 q を使って，信頼区間を計算します。それぞれ R 関数 z.test が返す信頼区間と比較して検算します。右側検定では上界，左側検定では下界の値を無限(Inf, -Inf)とします。R 関数 z.test は NA (not available)を返します。

●1 標本の平均値の検定

```
A=c(10,23,33,38,40,40,44,45,46,47,48,49,49,50,50,
   51,51,54,54,55,58,59,61,62,63,67,69,80,95,100)
n=length(A); sm=mean(A); ssd=sd(A); n; sm; ssd #30, 53.0, 18,1
pm=47.8; psd=16.1
side='t'; cl=0.95

> Z1.test(n,sm,ssd,pm,psd,side,cl,r=4)
sm:sample mean   pm:pop. mean       md:sm-pm     md/(sm+pm)
       53.0333        47.8000         5.2333         0.0519
           ssd            psd              n              z
       18.0907        16.1000        30.0000         1.7804
      security        p-value           beta          power
        0.9250         0.0750         0.9062         0.0938
      lower-b.       upper-b.      rel.width     confidence
       47.2721        58.7945         0.6533         0.2052

> library(BSDA); z.test(A,NULL,side,pm,psd,NULL,cl)
     One-sample z-Test
data:  A
z = 1.7804, p-value = 0.07501
alternative hypothesis: true mean is not equal to 47.8
95 percent confidence interval:
 47.27213 58.79454
sample estimates:
mean of x
 53.03333
```

A は 30 人のクラスの英語模擬テストの得点です(作例)。この平均値(sm)は 53.0，標準偏差(ssd)は 18.1 です。一方，全国の平均値と標準偏差はそれ

ぞれ pm=48.8, psd であったとします。そこで，このクラスの成績(sm)は全国平均(pm)とは有意に異なるかどうかを検定します(両側検定:'t': two-sided)。自作関数 Z1.test の結果を見ると，確かにクラス平均は全国平均よりも 5.2 点高く，相対的に 5.2%(0.519)高いことがわかります。一方，かなりばらついています(標準偏差 ssd=18.1)。全国の標準偏差(psd:16.1)よりも大きい値です。この標準偏差も勘案した安全率(security)は 0.925 なので，信頼水準 0.95 を超えていません。よって，否定的検定では帰無仮説を棄却できません。

次に「全国平均値より有意に高い」かどうかを知るために右側検定をします。

```
> side='g'; cl=0.95
> Z1.test(n,sm,ssd,pm,psd,side,cl,r=4)
sm:sample mean   pm:pop. mean      md:sm-pm     md/(sm+pm)
       53.0333        47.8000        5.2333         0.0519
           ssd            psd             n              z
       18.0907        16.1000       30.0000         1.7804
      security        p-value          beta          power
        0.9625         0.0375        0.8799         0.1201
      lower-b.       upper-b.     rel.width     confidence
       48.1984            Inf           Inf         0.3203

> library(BSDA); z.test(A,NULL,side,pm,psd,NULL,cl)
     One-sample z-Test
data:  A
z = 1.7804, p-value = 0.03751
alternative hypothesis: true mean is greater than 47.8
95 percent confidence interval:
 48.19838       NA
sample estimates:
mean of x
 53.03333
```

右側検定では安全率は 96.3%を示し，信頼水準を超えています。否定的検定では P 値が 3.8%なので 5%以下となり，帰無仮説を棄却して，対立仮説「クラスの平均値は全国の平均値よりも有意に高い」を採択します。このように両側検定は片側検定よりも厳しい判断になります。

次は選択的検定(side:'s')の結果です。(R 関数 z.test では実行できません。)

```
> side='s'; cl=0.95
> Z1.test(n,sm,ssd,pm,psd,side,cl,r=4)
sm:sample mean   pm:pop. mean      md:sm-pm     md/(sm+pm)
```

```
     53.0333         47.8000          5.2333          0.0519
         ssd             psd               n               z
     18.0907         16.1000         30.0000          1.7804
    security         p-value            beta           power
      0.9625          0.0375          0.8799          0.1201
    lower-b.        upper-b.       rel.width      confidence
     48.1984         57.8683          0.5483          0.2591
```

選択的検定では対象の平均値が比較する平均値以上のときは右側検定をし,そうでなければ左側検定をします。信頼区間は両側検定で算出します。この例のように，平均値の差が小さく，標準偏差がかなり大きいときは，確率分布は横に広がり，重なる領域が大きいので，検出力は比較的小さくなります。標準偏差が大きなデータでは信頼区間も大きくなります。このデータの信頼区間の相対値(rel.width)も大きいので，全体の信頼性(confidence)の数値を下げています。

●2 標本の平均値の差の検定

先の実験で確認した，次の z2 値が標準正規分布に従う，という性質を利用して，次のように 2 標本の平均値の差の検定を行います。

z2 = (m1-m2) / sqrt(sd1^2/n1 + sd2^2/n2)

(m1: 標本-1 の平均, sd1: 標本-1 の標準偏差，n1: 標本-1 のサイズ)
(m2: 標本-2 の平均, sd2: 標本-2 の標準偏差，n2: 標本-2 のサイズ)

2 標本の平均値の差の検定では，2 標本の平均値の差(m1-m2)の安全率・危険率(P 値)を検討します。帰無仮説として「2 標本の平均値の有意な差はない」ということを想定します。

```
#(2) Z2.test, of two samples (sample.1::sample.2)
A=c(10,23,33,38,40,40,44,45,46,47,48,49,49,50,50,
      51,51,54,54,55,58,59,61,62,63,67,69,80,95,100)
B=c(35,40,41,42,42,43,54,55,55,55,55,55,56,56,56,
      56,57,57,64,67,68,69,70,85,89,90,98,99)
n1=length(A); m1=mean(A); sd1=sd(A)
n2=length(B); m2=mean(B); sd2=sd(B)
side='t'; cl=.95
Z2.test(n1,m1,sd1,n2,m2,sd2,side,cl)
z.test(A,B,side,0,sd1,sd2,cl)

> Z2.test(n1,m1,sd1,n2,m2,sd2,side,cl)
```

```
 m1:mean.1  m2:mean.2   md:m1-m2 md/(m1+m2)        sd1        sd2
    53.033     61.036     -8.002     -0.070     18.091     17.403
    sd1/m1     sd2/m2         n1         n2          z   security
     0.341      0.285     30.000     28.000     -1.717      0.914
   p-value       beta      power   lower-b.   upper-b.  rel.width
     0.086      0.081      0.919    -17.138      1.133      0.275
confidence
     0.842

> z.test(A,B,side,0,sd1,sd2,cl)
      Two-sample z-Test
data:  A and B
z = -1.7169, p-value = 0.086
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -17.13784   1.13308
sample estimates:
mean of x mean of y
 53.03333  61.03571
```

上の A と B はそれぞれ A クラス(30 人), B クラス(28 人)の英語テストの得点です(作例)。比較すると，B クラスの方が成績がよかったようですが，それは有意な違い(差)でしょうか？はじめに両側検定(side='t')で「A クラスと B クラスの得点に有意な差はない」という帰無仮説が棄却できるか・否かを確かめました。両側検定の結果は安全率(security)が 91.4%で信頼水準(cl)=95%を超えていません。同じことですが，P 値を見ると 8.6%で有意水準(5%)を超えています。そこで，否定的検定では帰無仮説は棄却されず，有意な差がある，とは言えないことになります。しかし，検出力は高く(0.919)，信頼区間の相対的間隔(rel.width)はかなり狭いので，さらに分析を続けます。

次に，A クラスの平均値が B クラスの平均値よりも低いので，これが有意に低いか・否かを見るために左側検定を実行します。

```
> side='l'; cl=.95
> Z2.test(n1,m1,sd1,n2,m2,sd2,side,cl)
 m1:mean.1  m2:mean.2   md:m1-m2 md/(m1+m2)        sd1        sd2
    53.033     61.036     -8.002     -0.070     18.091     17.403
    sd1/m1     sd2/m2         n1         n2          z   security
     0.341      0.285     30.000     28.000     -1.717      0.957
   p-value       beta      power   lower-b.   upper-b.  rel.width
     0.043      0.086      0.914       -Inf     -0.336      0.231
```

```
confidence
     0.872

> z.test(A,B,side,0,sd1,sd2,cl)
      Two-sample z-Test
data:  A and B
z = -1.7169, p-value = 0.043
alternative hypothesis: true difference in means is less than 0
95 percent confidence interval:
         NA -0.3356606
sample estimates:
mean of x mean of y
 53.03333  61.03571
```

このように，安全率は 95.7%であり(P 値は 4.3%)，信頼水準を超えているので，帰無仮説を棄却して，対立仮説(「A クラスは B クラスよりも有意に平均値が低い」)を採択します。検出力の高さ(91.4%)と信頼区間の相対的間隔の狭さがこの結論を支持しています。総合的な信頼性(confidence)は高く 87.2%となりました。P 値と信頼区間は R 関数 z.test と一致します。

片側検定では信頼区間の下界が負の無限(-Inf)となるので，最後に選択的検定(side='s')を実行します。

```
> side='s'; cl=.95
> Z2.test(n1,m1,sd1,n2,m2,sd2,side,cl)
 m1:mean.1  m2:mean.2   md:m1-m2 md/(m1+m2)        sd1        sd2
    53.033     61.036     -8.002     -0.070     18.091     17.403
    sd1/m1     sd2/m2         n1         n2          z   security
     0.341      0.285     30.000     28.000     -1.717      0.957
   p-value       beta      power   lower-b.   upper-b.  rel.width
     0.043      0.086      0.914    -17.138      1.133      0.275
confidence
     0.852
```

上の Z2.test が示す Z 検定の結果を見ると, 平均値の差がかなりあるので, 検定するまでもなく，この差は有意であるかのようにも思われます。しかし，両側検定では有意な差にはなりませんでした。その理由は標準偏差を見るとわかります。データ A もデータ B も標準偏差(sd)が比較的大きいことは，sd1, sd2 の数値よりも，sd1/m1, sd2/m2 を見た方がわかりやすいでしょう。これは標準偏差値を平均値で割った値で「変動係数」(coefficent of variation)と呼ばれます。標準偏差値はデータの数値の規模に左右されますが，変動係数はデータの規模に左右されません。変動係数を使えば 100 点

満点のテストの標準偏差値と 50 点満点のテストの標準偏差値を比べることも可能です。このデータでは，平均値の大きさと比べた標準偏差の相対的な数値がかなり高いことが，安全率を下げ，危険率・P 値を上げている原因です。標準偏差が小さければ，A と B の分布曲線が細くなるので，次の実験結果を見るとわかるように，検定結果は全体的に有意差を示すようになります。

```
> Z2.test(n1,m1,18,n2,m2,17,side,cl)
  m1:mean.1  m2:mean.2   md:m1-m2 md/(m1+m2)        sd1        sd2
     53.033     61.036     -8.002     -0.070     18.000     17.000
     sd1/m1     sd2/m2         n1         n2          z   security
      0.339      0.279     30.000     28.000     -1.741      0.959
    p-value       beta      power   lower-b.   upper-b.  rel.width
      0.041      0.083      0.917    -17.010      1.005      0.275
 confidence
      0.854

> Z2.test(n1,m1,10,n2,m2,10,side,cl)
  m1:mean.1  m2:mean.2   md:m1-m2 md/(m1+m2)        sd1        sd2
     53.033     61.036     -8.002     -0.070     10.000     10.000
     sd1/m1     sd2/m2         n1         n2          z   security
      0.189      0.164     30.000     28.000     -3.045      0.999
    p-value       beta      power   lower-b.   upper-b.  rel.width
      0.001      0.008      0.992    -13.153     -2.852      0.252
 confidence
      0.897
```

データ数(n1, n2)は統計的検定の方法を選択するときに重要な指標になります。Z 検定はサイズが 25 以上で「大標本」と見なします。サイズがそれ以下であれば，後述する T 検定が使われます。

3.7. t 分布

3.8. ポアソン分布

たとえば，国内で 1 日に起こる交通事故数や，多くの文字を含む 1 頁の中に見られる特定の文字の数などのように，ランダムに生起する現象ではその生起確率(p)が低くても母数(n)が大きければ，かなりの平均値(m = p*n)になることがあります。このような場合に二項分布をそのまま使うと多く

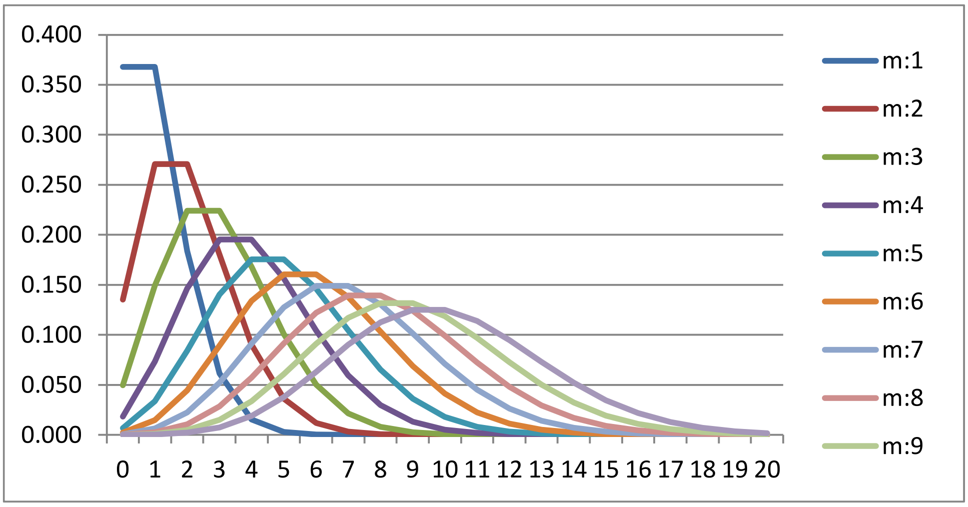
の階乗を含むので計算が膨大になります。そこで n→∞(無限大), p→0 に近づけたときの二項分布の数式を理論的に導き，その式を使って二項分布の近似式とします。そのような近似式によって示される確率分布は発見者の名 Siméon Denis Poisson (1781-1840) から「ポアソン分布」(Poisson distribution)と呼ばれ，次の式(Po)で示されます(→**●ポアソン分布の数式の導出**)。

Po(X) = e^(-m) * m^x / x!, (x: 0, 1, 2, …; 平均 m=n*p)

```
x=0;m=1; exp(1)^(-m)*m^x/factorial(x) # 0.3678794
ppois(x,m) # 0.3678794
```

次の表は，横軸に平均(Mean: m)を設定し，縦軸に生起回数(x)を設定して，平均と生起回数から，個別のポアソン確率を計算した結果です。

Mean	1	2	3	4	5	6	7	8	9	10
x	m:1	m:2	m:3	m:4	m:5	m:6	m:7	m:8	m:9	m:10
0	0.368	0.135	0.050	0.018	0.007	0.002	0.001	0.000	0.000	0.000
1	0.368	0.271	0.149	0.073	0.034	0.015	0.006	0.003	0.001	0.000
2	0.184	0.271	0.224	0.147	0.084	0.045	0.022	0.011	0.005	0.002
3	0.061	0.180	0.224	0.195	0.140	0.089	0.052	0.029	0.015	0.008
4	0.015	0.090	0.168	0.195	0.175	0.134	0.091	0.057	0.034	0.019
5	0.003	0.036	0.101	0.156	0.175	0.161	0.128	0.092	0.061	0.038
6	0.001	0.012	0.050	0.104	0.146	0.161	0.149	0.122	0.091	0.063
7	0.000	0.003	0.022	0.060	0.104	0.138	0.149	0.140	0.117	0.090
8	0.000	0.001	0.008	0.030	0.065	0.103	0.130	0.140	0.132	0.113
9	0.000	0.000	0.003	0.013	0.036	0.069	0.101	0.124	0.132	0.125
10	0.000	0.000	0.001	0.005	0.018	0.041	0.071	0.099	0.119	0.125
11	0.000	0.000	0.000	0.002	0.008	0.023	0.045	0.072	0.097	0.114
12	0.000	0.000	0.000	0.001	0.003	0.011	0.026	0.048	0.073	0.095
13	0.000	0.000	0.000	0.000	0.001	0.005	0.014	0.030	0.050	0.073
14	0.000	0.000	0.000	0.000	0.000	0.002	0.007	0.017	0.032	0.052
15	0.000	0.000	0.000	0.000	0.000	0.001	0.003	0.009	0.019	0.035
16	0.000	0.000	0.000	0.000	0.000	0.000	0.001	0.005	0.011	0.022
17	0.000	0.000	0.000	0.000	0.000	0.000	0.001	0.002	0.006	0.013
18	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.001	0.003	0.007
19	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.001	0.004
20	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.001	0.002



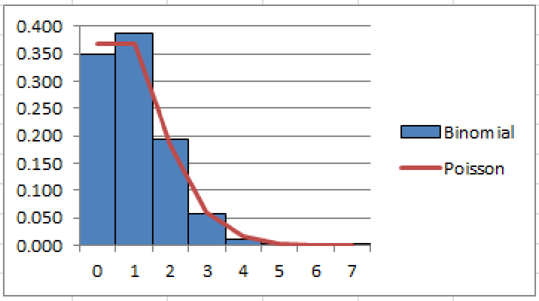
上の表によって，たとえば 60 分間に平均 3 回電話を受けている事務所が実際に 60 分間にランダムに電話を受ける回数が x=0, 1, 2, 3, 4, ...となるそれぞれの確率は，m:3 の列を見て，それぞれ 5.0% (x:0), 14.9% (x:1), 22.4% (x:2), 22.4% (x:3), 16.8% (x:4), ...になることが予想されます。このことは 1 時間を多くの区間，たとえば 20 個，30 個，60 個，...などの区間に分けて，各区間に起こる確率を 3/20, 3/30, 3/60, ...と考えます。そうすると，このように n を増やしていくと確率はどんどん小さくなります。そして n*p = m を一定にしながら n→∞，p→0 という極限に近づけたときの確率の分布がポアソン分布になります。

次の図は，n=10, 20, 30 にしたときの二項分布確率(Bin)とポアソン分布確率(Po)を比較したものです。ここで

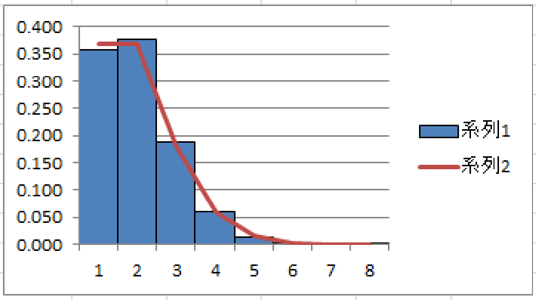
Bin: B77=BINOMDIST(A77,10,1/10,0)
Po: C77=POISSON(A77,1,0)

試行回数 n が上昇するにしたがって二項分布の確率の変数 p を 1/10, 1/20, 1/50 と下げていきます。ポアソン分布の平均の変数 m = n*p は常に 1 とします。下の表とグラフによって，試行回数 n が上昇し，確率 p が下降するに従って，ポアソン分布が二項分布に近似する精度が高まっていることが確かめられます。

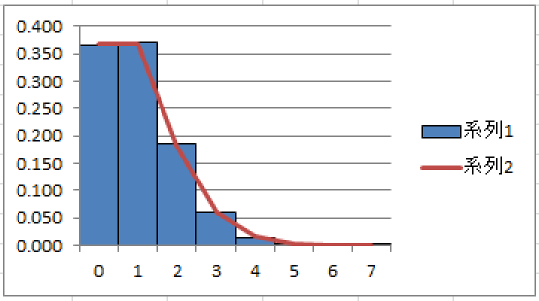
75	N:10			
76	x	Binomial	Poisson	B-P
77	0	0.349	0.368	-0.019
78	1	0.387	0.368	0.020
79	2	0.194	0.184	0.010
80	3	0.057	0.061	-0.004
81	4	0.011	0.015	-0.004
82	5	0.001	0.003	-0.002
83	6	0.000	0.001	0.000
84	7	0.000	0.000	0.000
85				



87	N:20			
88	x	Binomial	Poisson	B-P
89	0	0.358	0.368	-0.009
90	1	0.377	0.368	0.009
91	2	0.189	0.184	0.005
92	3	0.060	0.061	-0.002
93	4	0.013	0.015	-0.002
94	5	0.002	0.003	-0.001
95	6	0.000	0.001	0.000
96	7	0.000	0.000	0.000
97				

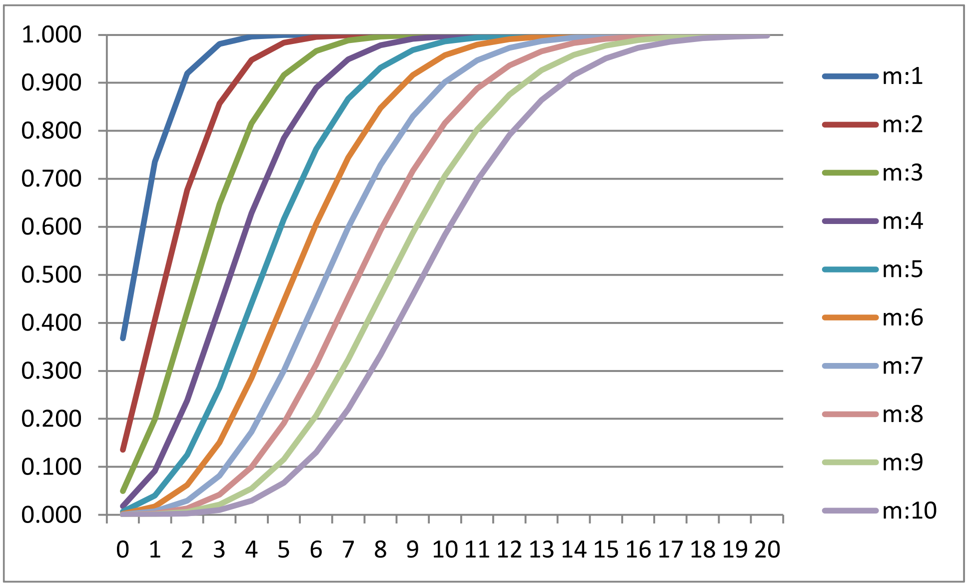


99	N:50			
100	x	Binomial	Poisson	B-P
101	0	0.364	0.368	-0.004
102	1	0.372	0.368	0.004
103	2	0.186	0.184	0.002
104	3	0.061	0.061	-0.001
105	4	0.015	0.015	-0.001
106	5	0.003	0.003	0.000
107	6	0.000	0.001	0.000
108	7	0.000	0.000	0.000
109				



次の表と図は累積ポアソン確率を示します。

Mean→	1	2	3	4	5	6	7	8	9	10
x↓	m:1	m:2	m:3	m:4	m:5	m:6	m:7	m:8	m:9	m:10
0	0.368	0.135	0.050	0.018	0.007	0.002	0.001	0.000	0.000	0.000
1	0.736	0.406	0.199	0.092	0.040	0.017	0.007	0.003	0.001	0.000
2	0.920	0.677	0.423	0.238	0.125	0.062	0.030	0.014	0.006	0.003
3	0.981	0.857	0.647	0.433	0.265	0.151	0.082	0.042	0.021	0.010
4	0.996	0.947	0.815	0.629	0.440	0.285	0.173	0.100	0.055	0.029
5	0.999	0.983	0.916	0.785	0.616	0.446	0.301	0.191	0.116	0.067
6	1.000	0.995	0.966	0.889	0.762	0.606	0.450	0.313	0.207	0.130
7	1.000	0.999	0.988	0.949	0.867	0.744	0.599	0.453	0.324	0.220
8	1.000	1.000	0.996	0.979	0.932	0.847	0.729	0.593	0.456	0.333
9	1.000	1.000	0.999	0.992	0.968	0.916	0.830	0.717	0.587	0.458
10	1.000	1.000	1.000	0.997	0.986	0.957	0.901	0.816	0.706	0.583
11	1.000	1.000	1.000	0.999	0.995	0.980	0.947	0.888	0.803	0.697
12	1.000	1.000	1.000	1.000	0.998	0.991	0.973	0.936	0.876	0.792
13	1.000	1.000	1.000	1.000	0.999	0.996	0.987	0.966	0.926	0.864
14	1.000	1.000	1.000	1.000	1.000	0.999	0.994	0.983	0.959	0.917
15	1.000	1.000	1.000	1.000	1.000	0.999	0.998	0.992	0.978	0.951
16	1.000	1.000	1.000	1.000	1.000	1.000	0.999	0.996	0.989	0.973
17	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.998	0.995	0.986
18	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.999	0.998	0.993
19	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.999	0.997
20	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.998



上の表と図で用いられた平均 m は整数を使っていますが，これは分数や小数になることもあります。たとえば，ある病気の全国の発生率が 1/2550 であって，ある地域の人口が 30000 人であれば，平均の発生数は 30000/2550=11.765...となります。このときたとえば個別の発生数，たとえば 10 件の確率や(第 3 引数=0)，10 件以下の発生数の累積確率(第 3 引数=1)を求めるときには，平均 m として 30000/2550 を先のそれぞれの式に代入します。

```
dpois(10, 30000/2550) # 0.108818
ppois(10, 30000/2550) # 0.3723706
```

●ポアソン分布の数式の導出

次のポアソン分布の式(Po)を二項分布の式(Bin)から導出します。

$Po(X) = e^{-m} * m^x / x!$, (x: 0, 1, 2, …; 平均 $m = np > 0$)

上の式を次の二項分布の式から導きます。

$Bin(X) = {}_nC_x * p^x * (1 - p)^{n-x} \quad (n \to \infty)$

ここで

$n * p = m \quad \to \quad p = m / n$

とすると

$Bin(X)$
$= {}_nC_x * p^x * (1 - p)^{n-x}$

$= n! / [(n - x)!\ x!] * (m/n)^x * (1 - m/n)^{n-x}$　　←C の定義, p = m/n

$= n! / [(n - x)!\ x!] * m^x/n^x * (1 - m/n)^{n-x}$ ←指数 x を分配

$= \underline{n! / [(n - x)!\ n^x]} * m^x/\underline{x!} * \underline{(1 - m/n)^{n-x}}$ ←x!と n^x の位置を交換

$= \underline{n! / [(n - x)!\ n^x]} * \underline{m^x/x!} * \underline{(1 - m/n)^n} * \underline{(1 - m/n)^{-x}}$ ←指数 n-x を分解

上の式を下線の(a), (b), (c), (d)に分け，n→∞とします。

(a) $n! / [(n - x)!\ n^x]$

$= [n * (n - 1) * \ldots * (n - x +1)] / n^x$　　←階乗(!)を整理

$= \text{ñ}n * (n - 1)/n * \ldots * (n - x +1)/n$　　←各項に n を分配[27]

$= 1 * (1 - 1/n) * \ldots * [1 - (x +1)/n]$　　←各項を計算

ここで n→∞とすると

$= 1 * 1 * \ldots * 1 = 1$　　←(n→∞)

(b) $m^x/x!$　←このままにする

(c) $(1 - m/n)^n$

$= [1 - 1 / (\text{ñ}m)]^n$　　←分子と分母に ñm を掛ける

$= \{[1 - 1 / (\text{ñ}m)]^{-\text{ñ}m}\}^{-m}$　←指数部に-ñm を導入

$= \{[1 + 1/(-\text{ñ}m)]^{-\text{ñ}m}\}^{-m}$　←分子と分母に-1 を掛ける

ここで x = - ñm とし，n→∞，よって，(x→-∞)とすると

$= \underline{[(1 + 1/x)^x]}^{-m}$　　(x→-∞)

$= e^{-m}$　←ネイピア数 e の定義：$e = (1 + 1/x)^x$, x→±∞ (→後述)

(d) $= (1 - m/n)^{-x}$　　((n→∞))

$= 1$

よって

Po(X)

= Bin(X), n→∞

$= {}_nC_x * p^x * (1 - p)^{n-x}$　　(n→∞)

= (a) * (b) * (c) * (d)　　(n→∞)

$= 1 * m^x / x! * e^{-m} * 1$

$= e^{-m} * m^x / x!$

---

[27] 分母 $n^x = n * n * \ldots$の各項(n)を，それぞれの分子に対応する分母にします。

●ポアソン分布の平均と分散

ポアソン分布の期待値 E(x)と分散 V(x)は二項分布の期待値 E(x)と分散 V(x)の式の中で p = m / n (←m = n * p)とし，n→∞とすることにより導かれます。

E(x) = n * p
= n * m / n ← p = m / n
= m

V(x) = n * p * (1 - p) (n→∞)
= n * m / n (1 - m / n) (n→∞) ← p = m / n
= m * (1 - m / n) (n→∞)
= m

このようにポアソン分布は期待値（平均）と分散が一致することが特徴です。

＊参考：一石(2004:60-61)，倉田・星野(2009:136-137).

●ネイピア数の定義

ネイピア数(e)は次のように定義されます（John Napier, 1550–1617: 高校数学 III）。

$$e = \lim(h \to 0)\ (1 + h)^{1/h}$$

上の式は x = 1/h とすると次のように書き換えられます。

$$e = \lim(x \to \pm\infty)\ (1 + 1/x)^{x} \qquad \leftarrow\ x = 1/h,\ h = 1/x$$

このとき，h→0 は x→±∞になります。h→0 の h はプラスでもマイナスでも成り立つので，x→±∞とします。このことを Excel で実験すると次のようになり，すべて同じ結果(2.71828...)に収束していくことがわかります。

h	$(1+h)^{1/h}$	h	$(1+h)^{1/h}$	x	$(1+1/x)^{x}$	x	$(1+1/x)^{x}$
0.1	2.59374	-0.1	2.86797	10	2.59374	-10	2.86797
0.01	2.70481	-0.01	2.73200	100	2.70481	-100	2.73200
0.001	2.71692	-0.001	2.71964	1000	2.71692	-1000	2.71964
0.0001	2.71815	-0.0001	2.71842	10000	2.71815	-10000	2.71842
0.00001	2.71827	-0.00001	2.71830	100000	2.71827	-100000	2.71830
0.000001	2.71828	-0.000001	2.71828	1000000	2.71828	-1000000	2.71828

●二項分布確率・ポアソン分布確率・正規分布確率の比較

二項分布の分散 V = np(1-p)と正規確率 p の値によって，次のような場合分けがされています[28]。

(1) V = n*p*(1-p) < 10
  (1a) p ≧ 0.1 → 二項分布による検定
  (1b) p < 0.1 → ポアソン分布による検定

(2) V = n*p*(1-p) ≧ 10 → 正規分布による検定

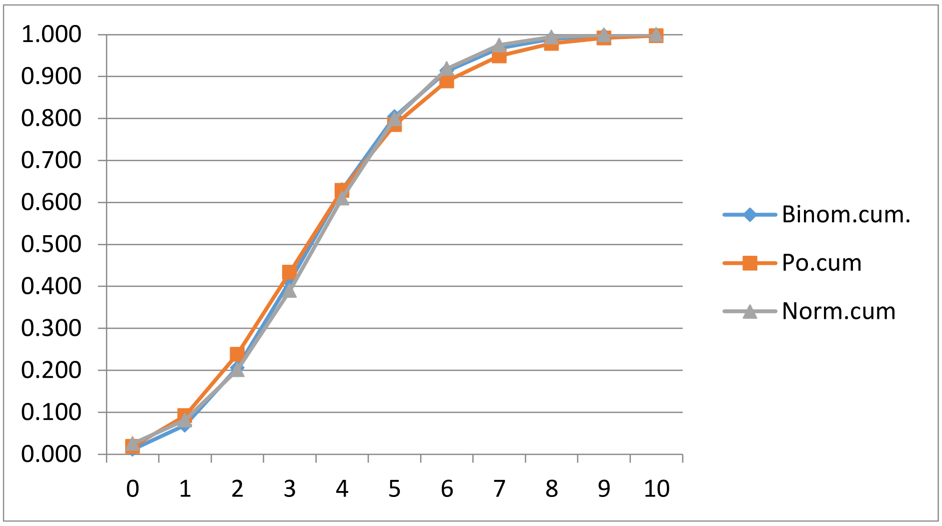
次の表と図は(1a) V=3.2, p = 0.2 の場合の二項分布，ポアソン分布，正規分布の累積確率を示します。

N	P	M	V
20	0.2	4	3.2

x	Binom.cum.	Po.cum	Norm.cum
0	0.012	0.018	0.025
1	0.069	0.092	0.081
2	0.206	0.238	0.201
3	0.411	0.433	0.390
4	0.630	0.629	0.610
5	0.804	0.785	0.799
6	0.913	0.889	0.919
7	0.968	0.949	0.975
8	0.990	0.979	0.994
9	0.997	0.992	0.999
10	0.999	0.997	1.000

```
x=0; n=20; p=.2; m=4; v=3.2
pbinom(x,n,p) # 0.01152922
ppois(x,m) # 0.01831564
pnorm(x+0.5,m,sqrt(v)) # 0.02519964
```

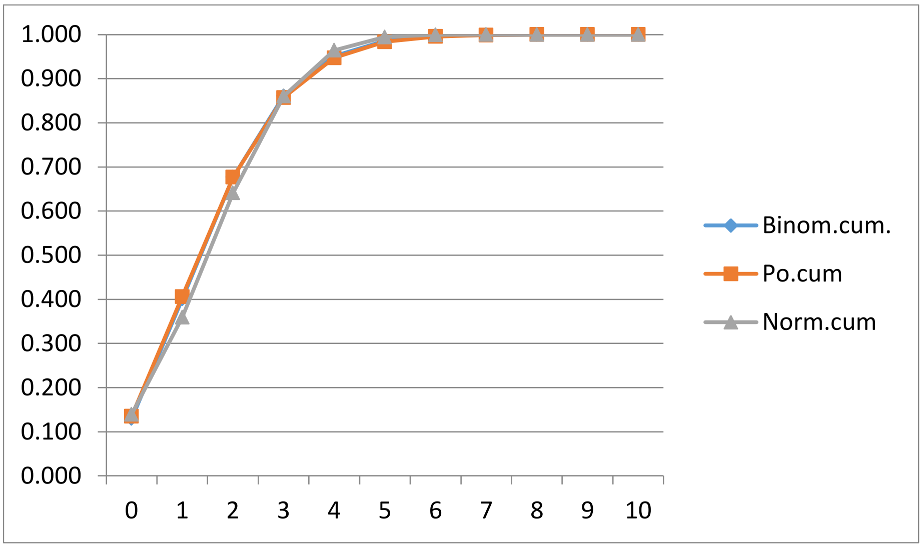
28 市原清志(1990)『バイオサイエンスの統計学』（南江堂）p.116-121.



ポアソン分布も正規分布もかなり二項分布に近似していますが，すこし誤差があることがわかります（とくに正規分布の誤差が大きい）。

次の表と図は(1b) V=1.92 < 10, p = 0.04 < 0.1 の場合の二項分布，ポアソン分布，正規分布を累積確率を示します。

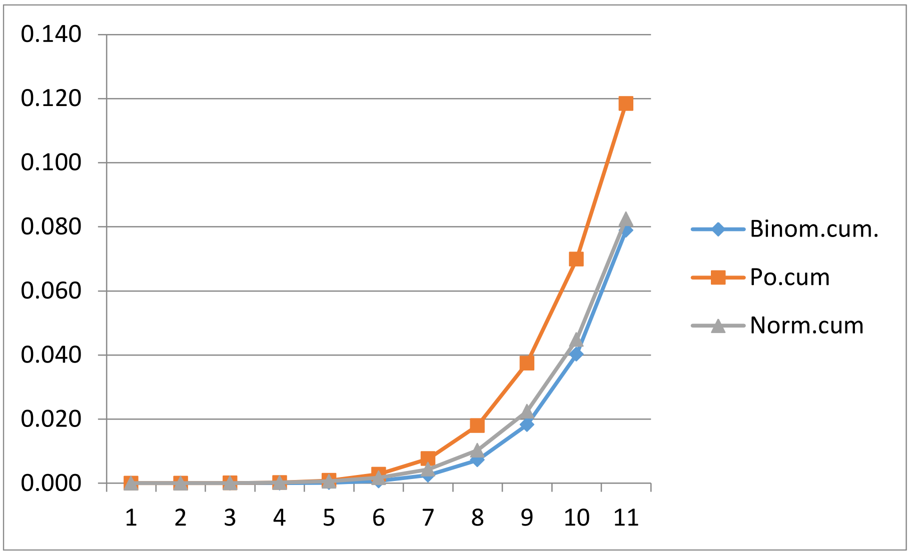
N	P	M	V
50	0.04	2	1.92
x	Binom.cum.	Po.cum	Norm.cum
0	0.130	0.135	0.140
1	0.400	0.406	0.359
2	0.677	0.677	0.641
3	0.861	0.857	0.860
4	0.951	0.947	0.964
5	0.986	0.983	0.994
6	0.996	0.995	0.999
7	0.999	0.999	1.000
8	1.000	1.000	1.000
9	1.000	1.000	1.000
10	1.000	1.000	1.000



たしかにポアソン分布は二項分布によく近似していますが，正規分布の近似があまりよくありません。

最後に，次の表と図は(2) V=10.5 > 10, p = 0.3 > 0.1 の場合の二項分布，ポアソン分布，正規分布を確率を示します。

N	P	M	V
50	0.3	15	10.5
x	Binom.cum.	Po.cum	Norm.cum
0	0.000	0.000	0.000
1	0.000	0.000	0.000
2	0.000	0.000	0.000
3	0.000	0.000	0.000
4	0.000	0.001	0.001
5	0.001	0.003	0.002
6	0.002	0.008	0.004
7	0.007	0.018	0.010
8	0.018	0.037	0.022
9	0.040	0.070	0.045
10	0.079	0.118	0.082



このように，ポアソン確率分布も正規確率分布も二項確率分布の近似であるので，できるかぎり二項確率分布を使うべきです。そのうえ二項確率分布はわかりやすいのですが，ポアソン確率分布や正規確率分布を理解するにはかなりの数学的準備が必要です。従来，巨大なサンプル数(N)や微小の期待確率(P)のときに，二項確率の計算は困難であったのですが，R 関数 pbinom は次のように問題なく対応する累積確率を返します。

```
n=10^8; p=.5; m=n*p; v=n*e*(1-e); x=n*.5
pbinom(x,n,p) # 0.5000399
ppois(x,m) # 0.5000376
pnorm(x+0.5,m,sqrt(v)) # 0.5000418
```

3.9. 乱数

先に見た投げたサイコロの目や，円盤に投げた針が示す角度のように，それぞれの数値が次の数値を予測することができず，それぞれの数値の範囲に対応する度数が均等になるような乱数は「一様分布乱数」(ramdam numbers of uniform distribution)と呼ばれます。

はじめに，一様分布乱数を生成する R 関数 runif()を使って，(0, 100)の範囲内で 20 個の乱数を出力させ確認します[29]。

```
set.seed(1); A=runif(20,min=0,max=100); round(A,0)
27 37 57 91 20 90 94 66 63  6 21 18 69 38 77 50 72 99 38 78
```

階級度数分布の関数 FreqDist を作成して，それを使い 10 万個の一様分布乱数のそれぞれの階級の範囲内に入る乱数の数が等しいかどうかを確か

[29] runif(x,min=0,max=100)で指定する min, max の数値は含みません。ここでは round で丸めているので，0 や 100 が現われる可能性があります。なお，round は正確には「一番近い偶数に丸める」rounding to the nearest even value ので，たとえば round(0.5)=0, round(1.5)=2 になります。

めます。ここでは 10 個の階級(0-, 1-, …, 10-)を用意します。

```
> set.seed(1); A=runif(10^2,min=0,max=100); FreqDist(A,10)
   class freq
1     0-    7
2    10-    6
3    20-   11
4    30-   14
5    40-   14
6    50-    5
7    60-   11
8    70-   15
9    80-   11
10   90-    6
```

上の実験では,それぞれの階級に入る乱数の数がかなりばらついています。そこで，乱数の数を 10 万個に増やして，同じ実験をします。

```
set.seed(1); A=runif(10^5,min=0,max=100); FreqDist(A,10)
   class  freq
1     0- 10156
2    10-  9986
3    20- 10008
4    30-  9928
5    40-  9930
6    50-  9958
7    60-  9959
8    70- 10034
9    80- 10014
10   90- 10027
```

このように，それぞれの階級に入る乱数の数がそれぞれ 1 万個にかなりよく近似しています。

```
FreqDist=function(A,it=10,s=0){
  f=ifelse(s==0,floor,Rnd); Id=f(A/it)+1; mx=max(Id); Fr=rep(0,mx)
  for(i in 1:length(Id)) Fr[Id[i]]=Fr[Id[i]]+1
  st=ifelse(s==0,'','-'); data.frame(class=st&0:(mx-1)*it&'-', freq.=Fr)
} #度数分布  Frequency distribution (A:array, it: interval,s=0:floor, 1:Rnd)
```

●四捨五入

数値を丸める R 関数 round は，次のように，丸める桁が 5 かつ，それ以下がゼロのときは一番近い偶数を返します(round to nearest even)。

```
round(1250,-2); round(2.5,0); round(3.5,0)
#丸め-1: 四捨五入しない
[1] 1200  [1] 2  [1] 4
```

一方，丸める桁が 5 でもそれ以下がゼロでなければ四捨五入します。

```
round(1251,-2); round(2.51,0); round(3.51,0)
#丸め-2: 四捨五入する
[1] 1300  [1] 3  [1] 4
```

自作関数 Rnd はすべてのケースで四捨五入します[30]。

```
Rnd(1250,-2); Rnd(2.5,0); Rnd(2.25,1) #四捨五入
[1] 1300
[1] 3
[1] 2.3
```

小数桁が多い乱数では，round と Rnd の違いはほとんど問題になりませんが，たとえば，度数分布から相対頻度などを求めるとき，度数を 2, 4, … などの偶数で割ると，丸める桁の数値が偶数になり，その直下の桁の数値が 5 になり, それ以下の桁に数値がないケースがしばしば発生します(たとえば 5/2=2.5，18/4=4.5)。このとき，小数 1 桁で四捨五入すと，round(2.5), round(4.5)は 2，4 を返すので注意が必要です。自作関数 Rnd では，次のように, 受けた数値に微小な数値(1/10^10)を加えてから round に渡すことで, これを回避しています。

```
Rnd=function(x,d=0) round(x+1/10^10,d)
# 四捨五入 (d:小数桁) cf round(2.5)=>2
```

●乱数の平均

R 関数で乱数(0, 1)を多数発生させ，その平均(=.5)を実験的に確かめます。たとえば 10 万個の乱数を発生させると，それらの乱数の平均はおよそ 0.5 となり，その分散はおよそ 0.0833 になりました

```
set.seed(1); A=runif(10^6,min=0,max=1); mean(A)
# 0.4999223
```

一様分布乱数の範囲が(0, 1)ですから，この平均がおよそ 0.5 になることは想像できますが，次に数理的にそれを一般化します。

次のような数値(x)と，その頻度(f)からなる頻度分布の例を見ましょう。

x	0	0.1	0.2	…	0.9	和
f	100	100	100	…	100	1000

[30] Excel 関数 ROUND もすべてのケースで四捨五入します。

この頻度分布表を使って平均(m)を求めると次のようになります。

$$m = [(0 * 100) + (0.1 * 100) + (0.2 * 100) + \dots + (0.9 * 100)] / 1000$$

次に，この頻度(f)を確率(P)に変えて，次の確率分布にします。

X	0	0.1	0.2	…	0.9	和
P	1 / 10	1 / 10	1 / 10	…	1 / 10	1

上表のように確率(p)の和は必ず 1 になります。

[1] $\Sigma_i p_i = 1$

この確率分布を使って平均(m)を求めます。分数の分母のゼロの連続を避けるためにマイナスの指数を使います。下の第一式が先の頻度分布表と同じであることを確かめてください。

$$\begin{aligned}
m &= \Sigma_i x_i p_i \qquad (i = 0, 1, 2, \dots, 9)\\
&= (0 * 10^{-1}) + (10^{-1} * 10^{-1}) + (0.2 * 10^{-1}) + \dots + (0.9 * 10^{-1})\\
&= (0 + 0.1^{1} + 0.2 + \dots + 0.9) * 10^{-1} && \text{←各項の } 10^{-1} \text{ を外へ}\\
&= (0 + 1 + 2 + \dots + 9) * 10^{-1} * 10^{-1} && \text{←(*)内の各項の } 10^{-1} \text{ を外絵}\\
&= (0 + 1 + 2 + \dots + 9) * 10^{-2} && \text{←分母を整理}\\
&= (9 * 10 / 2) * 10^{-2} && \text{←脚注}^{31}\\
&= 45 * 10^{-2} = 0.45
\end{aligned}$$

次に小数点以下 2 桁までの乱数の平均(m')は

$$\begin{aligned}
m' &= \Sigma_i x_i p_i \qquad (i = 0, 1, 2, \dots, 99)\\
&= (0 * 10^{-2}) + (0.01 * 10^{-2}) + (0.02 * 10^{-2}) + \dots + (0.99 * 10^{-2})\\
&= (0 + 0.01 + 0.02 + \dots + 0.99) * 10^{-2}\\
&= (0 + 1 + 2 + \dots + 99) * 10^{-2} * 10^{-2}\\
&= (0 + 1 + 2 + \dots + 99) * 10^{-4}\\
&= (99 * 100 / 2) * 10^{-4}\\
&= 4950 * 10^{-4} = 0.495
\end{aligned}$$

さらに，小数点以下 3 桁までの乱数の平均(m")は

$$\begin{aligned}
m'' &= \Sigma_i x_i p_i \qquad (i = 0, 1, 2, \dots, 999)\\
&= (0 * 10^{-3}) + (0.001 * 10^{-3}) + (0.002 * 10^{-3}) + \dots + (0.999 * 10^{-3})\\
&= (0 + 0.01 + 0.02 + \dots + 0.999) * 10^{-3}\\
&= (0 + 1 + 2 + \dots + 999) * 10^{-3} * 10^{-3}
\end{aligned}$$

[31] 数列(1, 2, …, n)の和＝$n(n + 1) / 2$，よって $n = 9$ のときの和は 45 (←高校数学 B)．わかりやすいようにこの部分を括弧(…)で囲みます。

$= (0 + 1 + 2 + ... + 999) * 10^{-6}$
$= (999 * 1000 / 2) * 10^{-6}$
$= 499500 * 10^{-6} = 0.4995$

このように乱数の間隔を次第に小さくし,乱数の種類を多くしていくと,乱数の平均は次第に 0.5 に近づくことがわかります。後述するように，乱数の間隔を無限にゼロ(0)に近づければ，平均は無限に 0.5 に近づくことが予想できます。そして，範囲が[0, 1)の乱数の平均が 0.5 に近づくことは，私たちが直感で納得できることです。

●乱数の分散

次のようにして，一様分布乱数の分散を実験的に求めます。

```
set.seed(1); A=runif(10^6,min=0,max=1); varp(A)
# 0.08330679
```

先に見たように，確率分布の分散は

$$V(X) = E(X^2) - [E(X)]^2$$

そこで,分散 V(X)を求めるには,先に平均 E(X) = 0.5 を求めてあるので,あとは $E(X^2)$がわかればよいことになります。

X	0	0.1	0.2	…	0.9	和
X^2	0^2	$(0.1)^2$	$(0.2)^2$		$(0.9)^2$	
P	0.1	0.1	0.1	…	0.1	1

$E(X^2) = \Sigma_i X_i^2 p_i \quad (i = 0, 1, 2, ..., 9)$
$= 0^2*10^{-1} + (10^{-1})^2*10^{-1} + (0.2)^2*10^{-1} + (0.3)^2*10^{-1} + ... + (0.9)^2*10^{-1}$
$= [0^2 + (0.1)^2 + (0.2)^2 + (0.3)^2 + ... + (0.9)^2] * 10^{-1}$
$= [0 + 1^2 + 2^2 + 3^2 + ... + 9^2] * 10^{-2} * 10^{-1}$
$= [0 + 1^2 + 2^2 + 3^2 + ... + 9^2] * 10^{-3}$
$= [9 * (9 + 1) * (2 * 9 + 1) / 6] * 10^{-3}$ ←注[32]
$= 285 * 10^{-3} = 0.285$

よって，分散 V(X)は

$$V(X) = E(X^2) - [E(X)]^2 = 0.285 - (0.5)^2 = 0.285 - 0.25 = 0.035$$

次に，小数点以下 2 桁までの乱数の分散 V(X)'を求めます。

[32] 数列$(1^2, 2^2, ..., n^2)$の和＝$n*(n+1)*(2n+1) / 6$，よって $n = 9$ のときの和は 285.←高校数学 B. わかりやすいようにこの部分を括弧[...]で囲みます。

$E(X^2)' = \Sigma_i X_i^2 p_i \quad (i = 0, 1, 2, ..., 99)$

$= 0^2*10^{-2}+(0.01)^2*10^{-2}+(0.02)^2*10^{-2}+(0.03)^2*10^{-2}+ ... + (0.99)^2*10^{-2}$

$= [0 + (10^{-2})^2 + (0.02)^2 + (0.03)^2 + ... + (0.99)^2] * 10^{-2}$

$= [0 + 1^2 + 2^2 + 3^2 + ... + 99^2] * 10^{-4} * 10^{-2}$

$= [0 + 1^2 + 2^2 + 3^2 + ... + 99^2] * 10^{-6}$

$= [99 * (99 + 1) * (2 * 99 + 1) / 6] * 10^{-6}$

$= 328350 * 10^{-6} = 0.32835$

$V(X)' = E(x^2) - [E(x)]^2 = 0.328 - (0.5)^2 = 0.32835 - 0.25 = 0.07835$

さらに，小数点以下 3 桁までの乱数の分散 V"(X)は

$E(x^2)'' = \Sigma_i x_i^2 p_i \quad (i = 0, 1, 2, ..., 999)$

$= 0^2*10^{-3}+(0.001)^2*10^{-3}+(0.002)^2*10^{-3}+(0.003)^2*10^{-3}+...+ (0.999)^2*10^{-3}$

$= [0 + (0.001)^2 + (0.002)^2 + (0.003)^2 + ... + (0.999)^2] * 10^{-3}$

$= [0 + 1^2 + 2^2 + 3^2 + ... + 999^2] * 10^{-6} * 10^{-3}$

$= [0 + 1^2 + 2^2 + 3^2 + ... + 999^2] * 10^{-9}$

$= [999 * (999 + 1) * (2 * 999 + 1) / 6] * 10^{-12}$

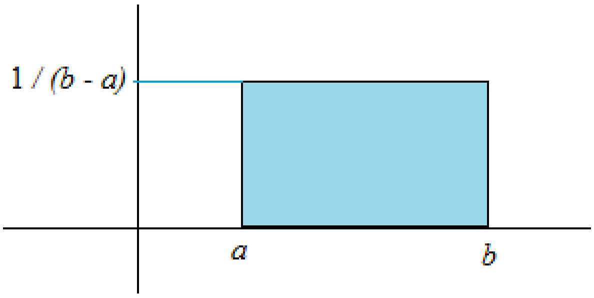
$= 332833500 * 10^{-12} = 0.3328335$

$V(X)'' = E(x^2)'' - [E(x)]^2 = 0.3328335 - (0.5)^2 = 0.3328335 - 0.25 = 0.0828335$

この段階まで求めた分散 0.0828335 が，先に実験的に確かめた乱数の分散 0.0833...に近似することがわかりました。以上で，それぞれの数値に対応する確率を個別に区切ってその平均と分散を求めました。そのような個別の確率は「離散的確率」(discrete probability)と呼ばれます。ここでは i = 9, 99, 999 まで増やしながら乱数の分散を求めましたが，さらに i = 9999, 99999, ...のように精度を高めていき，i → ∞のときの乱数の分散を求めることを，次に考えます。

たとえば，{1, 2, 3, 4, 5, 6}という目をもつサイコロを次々に投げたとき，次に出る目は[1, 6]の範囲内でまったく予測できませんが，それぞれの確率はすべて 1/6 で同じになります。

先に離散的確率変数の平均と分散を求めましたが，実は乱数の小数点以下の桁数は非常に大きく理論的には無限にあると考えられるので，厳密に言えば，確率分布表の p ではなく，次のようなグラフと式で示される**一様分布**(uniform distribution)の**確率密度**(probability density)の関数 f(x)を使わなければなりません。



$$f(x) = \begin{cases} 1 / (b - a) & [a \sim b] \\ 0 & [-\infty \sim a,\ b \sim +\infty] \end{cases}$$

ここで，a, b はそれぞれ区間の下端（開始点）と上端（終了点）を示します。[0, 1)の区間にある乱数では，a = 0, b = 1 になります。x が 0 以下または 1 以上のときは f(x)はゼロ(0)になります。

$$f(x)' = 1 / (1 - 0) = 1 \quad [0 \sim 1]$$

はじめに，このような一様分布の確率密度関数の全体の値(総和：S)を積分を使って求めます。先の離散的な確率ではシグマ（Σ：和）を使って，個別の確率を掛けて足し合わせていきましたが，ここでは連続的な確率になるので，次のような定積分を使います(←高校数学 II)。

$$S = \int_0^1 f(x)\ dx = \int_0^1 1\ dx = [x]_0^1 = 1 - 0 = 1$$

上式では[0 ~ 1]の区間で計算していますが，特定の点での積分値はゼロになるので[33]，乱数の区間[0 ~ 1)でも同じです。また，[0 ~ 1]の区間以外の f(x)の値はゼロなので，区間[-∞ ~ +∞]にしても同じように結果は 1 になり，このことは一様分布の確率の総和が 1 になることを示しています。

さて，この f(x) = 1 を用いて連続的確率変数の平均を求めると

$$E(x) = \int_0^1 x\, f(x)\ dx = \int_0^1 x\, 1\ dx = \left[\frac{x^2}{2}\right]_0^1 = (1^2 / 2) - (0^2 / 2) = 1 / 2$$

よって，連続的な乱数の平均値は 1 / 2 = 0.5 になります。このことは先の実験で確かめました。

次に分散を求めるために，二乗の平均 $E(x^2)$を計算します。

$$E(x^2) = \int_0^1 x^2 f(x)\ dx = \int_0^1 x^2\, 1\ dx = \left[\frac{x^3}{3}\right]_0^1 = (1^3 / 3) - (0^3 / 3) = 1 / 3$$

よって分散 V(x)は

$$\begin{aligned} V(x) &= E(x^2) - [E(x)]^2 = (1 / 3) - (1 / 2)^2 = (1 / 3) - (1 / 4) \\ &= (4 / 12) - (3 / 12) = 1 / 12 = 0.0833\ldots \end{aligned}$$

[33] $\int_k^k f(x)\ dx = [F(x)]_k^k = F(k) - F(k) = 0$ ($F(x)$は $f(x)$の原始関数)

以上で，先に乱数の実験で求めた分散 0.0833...と，小数点以下 3 桁の離散的確率変数で求めた分散 0.0828 が，連続的確率変数を使って数理的に求めた分散 1 / 12 = 0.0833...と近似することを確かめました。

＊一様分布の平均と分散については永田(2005: 61, 66)を参照しました。

●正規分布乱数

平均値を中心にして度数が多く集中し，平均値から離れるにつれて度数が次第に少なくなる正規分布を示す乱数は「正規分布乱数」(ramdam numbers of normal distribution)と呼ばれます。階級度数分布の自作関数 FreqDist を使って 100 万個の正規分布乱数のそれぞれの階級の範囲内に入る乱数の分布を確かめます。同時にその平均値と分散も計算します。

```
set.seed(1);  A=rnorm(10^6,mean=5,sd=1);    FreqDist(A,1);
mean(A); varp(A) #正規分布乱数
   class  freq.
1     0-     37
2     1-   1293
3     2-  21425
4     3- 136084
5     4- 340974
6     5- 341357
7     6- 135927
8     7-  21589
9     8-   1278
10    9-     36
[1] 5.000047
[1] 1.00037
```

確かに，このように，それぞれの階級に入る乱数の数がそれぞれ正規分布にかなりよく近似しています。しかし，階級が平均値(=5)の下(=4)と上(=6)の度数が大きく違っています。正規分布によく近似するならば，両者は上下対称で差があまりないはずです。これは度数分布の関数 FreqDist で階級値を切り捨てて(関数: floor)，その数値から始まる数値を数えているためです。そこで，階級の間隔を 1 でなく，0.1 にすると次の結果になります。

```
(…)
49  4.8- 39600
50  4.9- 39794
51    5- 40035
52  5.1- 39302
53  5.2- 38564
(…)
```

このように，確かに今度は平均値(=5)を中心に上下対称で近似していま

す。

別の解決法は，階級値を度数の開始値ではなく，FreqDist の第 3 引数を 1 にして，階級値を，数値を四捨五入した値にする方法です(関数:Rnd)。

```
set.seed(1); A=rnorm(10^6,mean=5,sd=1);  FreqDist(A,1,1);
mean(A); varp(A) #正規分布乱数
   class  freq.
1    -0-      5
2    -1-    223
3    -2-   5855
4    -3-  60948
5    -4- 241605
6    -5- 382845
7    -6- 241772
8    -7-  60645
9    -8-   5870
10   -9-    228
11  -10-      4
[1] 5.000047
[1] 1.00037
```

このように，平均値を中心にした上下対称の分布に非常に近似しています。

(終)