## ■ Wilcoxon 検定の棄却確率の推定

```
In[1]:= m[n_, k_] := 0 /; k < 0
        m[n_, k_] := 0 /; k > n (n + 1) / 2
        m[1, 0] = m[1, 1] = 1;
        m[n_, k_] := m[n, n (n + 1) / 2 - k] /; k > n (n + 1) / 4
        m[n_, k_] := m[n, k] = m[n - 1, k] + m[n - 1, k - n]

In[6]:= cW[n_, a_] := Module[{i, p, d},
          i = n (n + 1) / 2;
          d = 2^n;
          p = m[n, i] / d;
          While[p < a, i = i - 1; p = p + m[n, i] / d];
          {i, p}]

In[7]:= cW[16, 0.10]

Out[7]= {93, 433/4096}

In[8]:= N[%[[2]]]

Out[8]= 0.105713

In[9]:= N[Sum[m[16, i] / (2^16), {i, 93, 8 * 17}]]

Out[9]= 0.105713

In[10]:= cutW = 93

Out[10]= 93

In[11]:= <<Statistics`ContinuousDistributions`

In[12]:= dataC[n_, a_, b_] := RandomArray[CauchyDistribution[a, b], n]

In[13]:= x = dataC[5, 0, 1]

Out[13]= {-3.41631, 0.875673, 0.12594, -3.24563, 0.929477}

In[14]:= y = Abs[%]

Out[14]= {3.41631, 0.875673, 0.12594, 3.24563, 0.929477}

In[15]:= z = Ordering[y]

Out[15]= {3, 2, 5, 4, 1}

In[16]:= Sort[y]

Out[16]= {0.12594, 0.875673, 0.929477, 3.24563, 3.41631}

In[17]:= s[x_] := 1 /; x > 0
         s[x_] := 0 /; x ≤ 0

In[19]:= Table[s[x[[i]]], {i, 5}]

Out[19]= {0, 1, 1, 0, 1}
```

```
In[20]:= wilcoxon[x_List] := Module[{z},
           z = Ordering[Abs[x]];
           Sum[i * s[x[[z[[i]]]]], {i, Length[x]}]]

In[21]:= x

Out[21]= {-3.41631, 0.875673, 0.12594, -3.24563, 0.929477}

In[22]:= Sort[y]

Out[22]= {0.12594, 0.875673, 0.929477, 3.24563, 3.41631}

In[23]:= wilcoxon[x]

Out[23]= 6

In[24]:= wilcoxon[dataC[16, 0, 1]]

Out[24]= 87

In[25]:= wilcoxon[dataC[16, 1, 1]]

Out[25]= 77

In[26]:= s[x_, c_] := 1 /; x ≥ c
         s[x_, c_] := 0 /; x < c

In[28]:= ? s

       Global`s

        s[x_] := 1 /; x > 0

        s[x_] := 0 /; x ≤ 0

        s[x_, c_] := 1 /; x ≥ c

        s[x_, c_] := 0 /; x < c

In[29]:= Table[s[wilcoxon[dataC[16, 1, 1]], cutW], {20}]

Out[29]= {0, 0, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1}

In[30]:= Sim[k_, n_, a_, b_, c_] :=

           Module[{p, sig}, p = Sum[s[wilcoxon[dataC[n, a, b]], c], {k}] / k; {p, 2 Sqrt[p (1 - p) / k]}]

In[31]:= N[Sim[10000, 16, 0, 1, cutW]]

Out[31]= {0.107, 0.00618226}

In[32]:= dataN[n_, a_, b_] := RandomArray[NormalDistribution[a, b], n]

       General::spell1 : スペル間違いの可能性があります. 新規シンボル"dataN"はすでにあるシンボル"dataC"に似ています. 詳細

In[33]:= student[x_List] := Module[{n, b, s2}, n = Length[x]; b = Sum[x[[i]], {i, n}] / n;

           s2 = Sum[x[[i]]^2, {i, n}] / n; s2 = (s2 - b^2) n / (n - 1); b Sqrt[n / s2]]
```

In[34]:= Quantile[NormalDistribution[0, 1], 0.95]

Out[34]= 1.64485

In[35]:= CDF[NormalDistribution[0, 1], %]

Out[35]= 0.95

In[36]:= cS[n_, a_] := Quantile[StudentTDistribution[n - 1], a]

In[37]:= cutS = cS[16, 1 - 0.105]

General::spell1 : スペル間違いの可能性があります. 新規シンボル"cutS"はすでにあるシンボル"cutW"に似ています. 詳細

Out[37]= 1.30971

In[38]:= Sim[k_, n_, a_, b_, c_, test_, data_] :=
Module[{p, sig}, p = Sum[s[test[data[n, a, b]], c], {k}] / k; {p, 2 $\sqrt{\frac{p\,(1-p)}{k}}$}]

In[39]:= N[Sim[10000, 16, 0, 1, cutW, wilcoxon, dataC]]

Out[39]= {0.111, 0.00628264}

In[40]:= N[Sim[10000, 16, 0, 1, cutS, student, dataN]]

Out[40]= {0.1086, 0.00622273}

In[41]:= N[Sim[10000, 16, 0, 1, cutW, wilcoxon, dataN]]

Out[41]= {0.1049, 0.00612849}

In[42]:= N[Sim[10000, 16, 0.5, 1, cutS, student, dataN]]

Out[42]= {0.7509, 0.00864984}

In[43]:= cB[n_, a_] := Module[{i, p, d},
i = n;
d = 2^n;
p = Binomial[n, i] / d;
While[p < a, i = i - 1; p = p + Binomial[n, i] / d];
{i, p}]

In[44]:= cB[16, 0.105]

Out[44]= $\left\{11, \frac{6885}{65536}\right\}$

In[45]:= N[%]

Out[45]= {11., 0.105057}

In[46]:= N[Sum[Binomial[16, i], {i, 11, 16}] / $2^{16}$]

Out[46]= 0.105057

In[47]:= cutT = 11

General::spell : スペル間違いの可能性があります. 新規シンボル"cutT"はすでにあるシンボル{cutS, cutW}に似ています. 詳細

Out[47]= 11

```
In[48]:= tukey[x_List] := Sum[s[x[[i]]], {i, Length[x]}]

In[49]:= x = dataN[16, 0, 1]

Out[49]= {-0.856295, 1.6623, -1.76112, -0.575696, 1.05413, 0.574969, -0.108474, 0.465186,
          1.01578, 1.16251, 1.98214, 0.540122, 0.416425, 0.441777, 0.456553, -0.41737}

In[50]:= tukey[x]

Out[50]= 11

In[51]:= N[Sim[10000, 16, 0, 1, cutT, tukey, dataN]]

Out[51]= {0.1073, 0.00618989}

In[52]:= N[Sim[10000, 16, 0.5, 1, cutT, tukey, dataN]]

Out[52]= {0.6226, 0.00969473}
```