

HTMLアプリケーションを用いた簡易SASツールの開発事例 - 臨床試験における症例数設計用ツールの構築 -

舟尾 暢男* 高浪 洋平*

* 武田薬品工業株式会社 医薬開発本部 日本開発センター 統計解析部

Some cases of developing simple SAS tools with HTML Application - Construction of tools for sample size calculation in clinical trial -

Funao Nobuo* Yohei Takanami*
*Takeda Pharmaceutical Company, Ltd.

SASプログラムを実行するための簡易ツール作成方法を紹介する。GUIを実装した本ツールはHTML/JavaScriptのみで構成され、作成とメンテナンスが容易である。実装例として症例数設計ツールを紹介する。

キーワード：アプリケーション構築、HTML Application(HTA)、JavaScript、Jscript、SAS、レポートの自動出力、他のソフトウェアの実行

1. はじめに

臨床試験の統計解析業務では、臨床試験が終わった後、SASによりデータ解析を行い、データ解析結果に関する報告書を作成することが一般的であるが、最近の臨床試験における解析報告書の分量は少なくなく、個々の臨床試験ごとにSASプログラムを作成するのは非常に労力を要する。この問題を解消する方法として、SASプログラムの汎用マクロを作成する方法があり、その先にはSASの汎用マクロを自動で実行するアプリケーションを開発する方法がある。さらに、GUIで操作することが出来るアプリケーションを開発してしまえば、SASの知識が無い人でも解析報告書を作成することが出来るようになり、解析報告書を作成する工数確保が容易になる。

さて、アプリケーション開発言語としては、C++、Java、Microsoft® Visual Basic® for Applications (VBA)、Microsoft® ASP.NET®、アプリケーション開発環境としては、SAS® AppDev Studio™、Java Eclipse、Microsoft® Visual Studio® などがあるが、これらを用いて開発を行うデメリットとして、新しい言語を覚える労力がかかり過ぎる、開発費用がかかり過ぎる、場合によってはサーバを立てる必要がある、などの問題がある。

本稿では、本ツールの入力画面からWindows版SASを動作させ、SASプログラムを実行するための簡易ツールの作成方法を紹介する。本稿におけるツール作成方法の特徴を以下に挙げる。

- 必要とする知識は「ホームページ作成のためのHTML言語」「JavaScriptとJscriptの限られた知識」であり、本稿に出てくる知識だけでアプリケーションを作成することが出来る。
- 本稿で紹介した方法により作成したツールは、大規模なウェブアプリケーションとは異なり、JavaScript等で記述されたHTMLアプリケーションのみで構成されるため、作成およびメンテナンスが容易であることに加えて、PC SASのライセンスを除くウェブサーバ等の費用を必要としない。
- 本稿で紹介した方法を適用することで、SASを実行することもWindowsのアプリケーションを実行することも出来るため、SASとその他のアプリケーションを組み合わせるレポートを作成することも出来る。

本稿では、アプリケーションの開発例として、症例数設計に関するアプリケーション「例数設計くん」の作成手順およびその概要を紹介する。

2. 使用するツールの概要

まず、本稿で使用するツールの概要を紹介する。

HTML

HTML とは、ウェブページを記述するための言語である。HTML で記述されたファイルを開覧するには Internet Explorer などのウェブブラウザを使用する。

JavaScript と JScript

JavaScript とは、主に HTML に埋め込んで動作させるスクリプト言語であり、コンパイラを必要としない手軽さが特徴である。ウェブページ上で「文字を入力」「ボタンをクリック」「統計量を算出する計算を実行」などが出来るが、JavaScript にはアプリケーションを操作する機能やファイルを操作する機能が用意されていない。そこで、Microsoft 社がこれらの機能を搭載するなどの拡張を JavaScript に加えたものが JScript である。本稿ではこの2つを併用してツールを作成する。

HTML Applications

HTML Applications (HTA) とは、HTML ファイル (拡張子「.html」) を拡張子「.hta」として保存し直したものである。HTA はウェブページで実行できる機能 (html, Java Script などのスクリプト言語, CSS など) をサポートしている。また、実行可能ファイル (拡張子「.exe」) と同じようなアプリケーションとして動作させることが出来、ダブルクリックで起動させることが出来る。

R

R とは、オープンソースかつフリーの統計解析用ソフトである。関数電卓、数値計算、プログラミング、統計解析などの機能がある。特に、グラフィックスの作成が容易に行えることが特徴である。

3. HTML アプリケーションのサンプル集

本節では、「例数設計くん」を開発するための基礎的なテクニックを紹介する目的で、HTML アプリケーションのサンプルをいくつか紹介する。

単純な HTML アプリケーション

まず、最もシンプルな HTML アプリケーションの例を挙げる。まず、作成した HTML アプリケーションを「sample1.hta」という名前で適当なフォルダに保存する。この「sample1.hta」をダブルクリックするか、「マウスを右クリック」「プログラムを実行する」から「Microsoft (R) HTML Application host」を選択することで、HTML アプリケーションを実行することが出来る。実行すると、図 3.1 の真ん中に示したようなウィンドウが表示される (ツール とする)。このツールの「Run」ボタンをクリックすると、Windows のメモ帳が表示される。

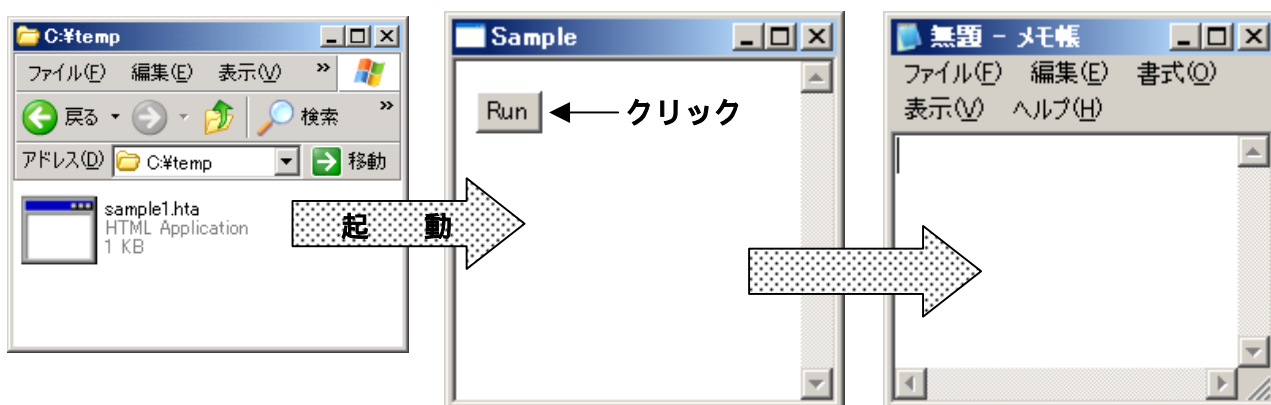


図 3.1 : sample1.hta

本ツールのプログラムを修正することで、SAS や他の Windows アプリケーションを実行するようなツールを作成することが出来る。このツール はテキストファイルで構成されており、適当なテキストエディタで編集することが出来る。ツール の中身は以下のとおり。

プログラム 3.1 : ツール 「sample1.hta」

```
01: <html>
02: <title>Sample</title>
03: <head>
04: <script type="text/javascript">
05:   resizeTo(400,300);
06:   function MyRun() {
07:     MyShell = new ActiveXObject("WScript.Shell") ;
08:     MyShell.Exec('notepad') ;
09:     MyShell = null ;
10:   }
11: </script>
12: </head>
13: <body>
14: <input type="button" value="Run" onclick="MyRun()">
15: </body>
16: </html>
```

以下にプログラムの中身に関する解説を付ける。

01～16 行目 : HTML アプリケーションは、最初と最後を <html> と </html> でくくる。

02 行目 : <title> と </title> の間に HTML アプリケーションのタイトルを入れる。

03～12 行目 : <head> <script ...> と </script> </head> の間に、Javascript や Jscript の命令を定義する。

05 行目 : 関数 resizeTo(横の幅,縦の幅) でウインドウの大きさをピクセル単位で指定する。

06～10 行目 : 関数 MyRun() を実行すると、メモ帳が起動される。

07 行目 : 08 行目でシェルを動作させるための準備をする。

08 行目 : 07 行目で準備した MyShell を使用し、notepad を実行する。イメージとしては、コマンドプロンプトから 'notepad' を実行するというイメージである。

もし、SAS プログラムを実行する場合は、8 行目を以下のように書き換えればよい。

```
08:     MyShell.Exec("C:/Program Files/SAS/SAS 9.1/sas.exe" -sysin
           "C:/temp/sample1.sas" -log "C:/temp/sample1.log"
           -print "C:/temp/sample1.txt" -nosplash -icon');
```

また、R プログラムを実行する場合は、8 行目を以下のように書き換えればよい。

```
08:     MyShell.Exec("C:/Program Files/R/R-2.6.2/bin/R.exe"
           --no-restore --no-save < "C:/temp/sample3.r");
```

09 行目 : 後処理。

13～15 行目 : アプリケーションの見栄え (文字の表示、ボタンやメニューの配置) を定義する。

14 行目 : <input type="button" ...> で ボタンを配置する。「value="Run"」でボタンのラベルを「Run」とし、「onclick="MyRun()」で、ボタンを押したときの動作 (この場合は「MyRun()」を実行する) を定義する。

HTML ファイルと Javascript の部分の分離

「sample1.hta」は HTML の命令と Javascript の命令が一緒になったファイルであるが、大規模なアプリケーションを作成する場合は、HTML の命令と Javascript の命令を分離するのが効率的である。以下に「sample1.hta」を「sample2_1.hta」と「sample2_2.js」に分離する例を示す。

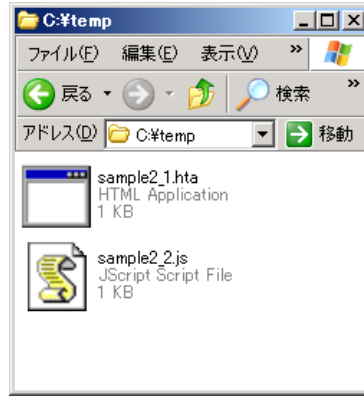


図 3.2 : sample2_1.hta、sample2_2.js

「sample2_1.hta」の 04 行目の「src="./sample2_2.js"」で「sample2_2.js」を呼び出しており、「sample2_2.js」には、Javascript の命令が定義されている。

プログラム 3.2.1 : sample2_1.hta

プログラム 3.2.2 : sample2_2.js

01:	<html>	01:	resizeTo(200,200);
02:	<title>Sample</title>	02:	function MyRun() {
03:	<head>	03:	MyShell = new ActiveXObject("WScript.Shell") ;
04:	<script language="JavaScript" src="./sample2_2.js"></script>	04:	MyShell.Exec('notepad') ;
05:	</head>	05:	MyShell = null ;
06:	<body>	06:	}
07:	<input type="button" value="Run" onclick="MyRun()">		
08:	</body>		
09:	</html>		

SAS ファイルに文字を追記する HTML アプリケーション

SAS ファイルにテキストを追記する HTML アプリケーション「sample3.hta (ツール として)」を作成する (図 3.3)。

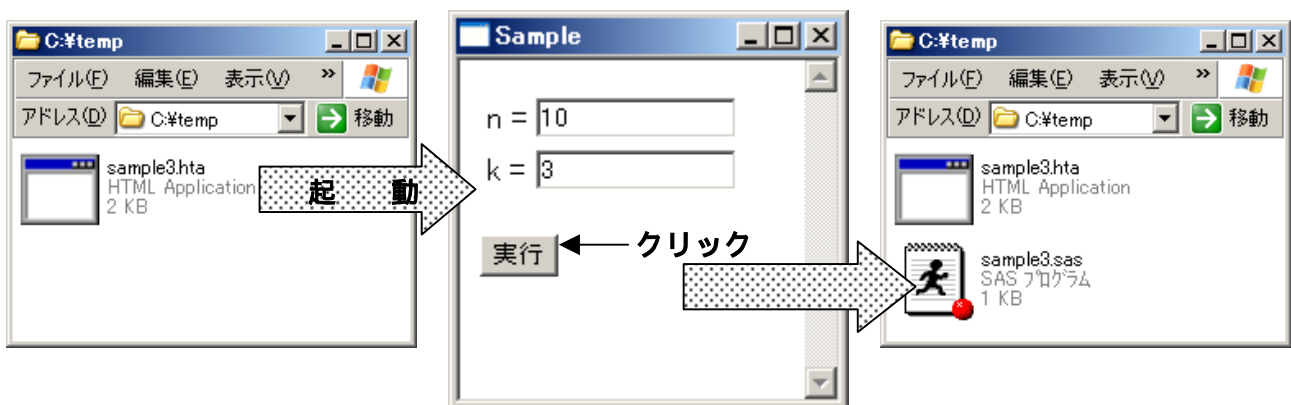


図 3.3 : sample1.hta

ツールのテキストボックスに数値を入力した後、「実行」ボタンをクリックすると、実行終了を知らせるダイアログが表示され、SAS ファイル「sample3.sas」にマクロパラメータに関するテキストを追記する (図 3.3 右)。生成された SAS プログラム「sample3.sas」の中身は以下のとおり。

プログラム 3.3.1 : sample3.sas

```
01: %let n = 10 ;
02: %let k = 3 ;
```

ツール の中身は以下のとおり。

プログラム 3.3.2 : ツール 「sample3.hta」

```
01: <html>
02: <title>Sample</title>
03: <head>
04: <script type="text/javascript">
05:   resizeTo(300,200);
06:   function MyGetParameter() {
07:     var MyVariable01, MyVariable02, MyScript, MyText ;
08:     MyVariable01 = document.MyForm.Mytextbox01 ;
09:     MyVariable02 = document.MyForm.Mytextbox02 ;
10:     if (MyVariable01 != "" & MyVariable02 != "") {
11:       MyScript = new ActiveXObject("Scripting.FileSystemObject");
12:       MyText = MyScript.CreateTextFile("C:/temp/sample5.sas", true);
13:       MyText.WriteLine("%let n = " + MyVariable01.value + " ;");
14:       MyText.WriteLine("%let k = " + MyVariable02.value + " ;");
15:       MyText.Close();
16:     }
17:     window.alert("セーブ完了 : %n n = " + MyVariable01.value + "%n k = " + MyVariable02.value) ;
18:   }
19: </script>
20: </head>
21: <body>
22: <form name="MyForm">
23: <table border=0>
24: <tr>
25: <td width=0><p>n = </p></td>
26: <td width=100><input name="Mytextbox01" type="text" value="10" style="width:100%;"></td>
27: </tr>
28: <tr>
29: <td width=0><p>k = </p></td>
30: <td width=100><input name="Mytextbox02" type="text" value="3" style="width:100%;"></td>
31: </tr>
32: </table>
33: <br>
34: <input name="Mygetbutton" type="button" value="実行" onclick="MyGetParameter()">
35: </form>
36: </body>
37: </html>
```

ツール 「sample3.hta」に出てくる主な命令について解説を付ける。

06～18 行目：後に出てくるテキストボックス「Mytextbox01」「Mytextbox02」から値を取得して、フォルダ「C:/temp/」に SAS ファイル「sample3.sas」を生成してマクロパラメータを書き込み、書き込みが終了したらその旨を知らせるダイアログを表示する関数 MyGetParameter() を定義する。

11 行目：ファイルを開くためのオブジェクト MyScript を定義する。

12 行目："C:/temp/sample3.sas" を開く。

13、14 行目：テキスト「%let n = 」を「sample3.sas」に書き出すことで、マクロパラメータを作成する。

15 行目："C:/temp/sample3.sas" を閉じる。

22～40 行目：<input name="Mytextbox0 " type="text" ... > でテキストボックスを表示する。見栄えの関係上、HTML の表に関する命令を使用している。

42 行目：<input name="Mygetbutton" type="button" ...> でボタンを配置する。「value="実行"」でボタンのラベルを「実行」とし、「onclick=" MyGetParameter ()"」で、ボタンを押したときの動作（この場合は「MyGetParameter ()」を実行する）を定義する。

メニュー選択を実装した HTML アプリケーション

HTML アプリケーションを複数用意する。ラジオボタンとドロップダウンリストを選択した後、「OK」ボタンをクリック（実行）すると、別の HTML アプリケーションを起動する「sample4_1.hta（ツール とする）」を作成する（図 3.4）。

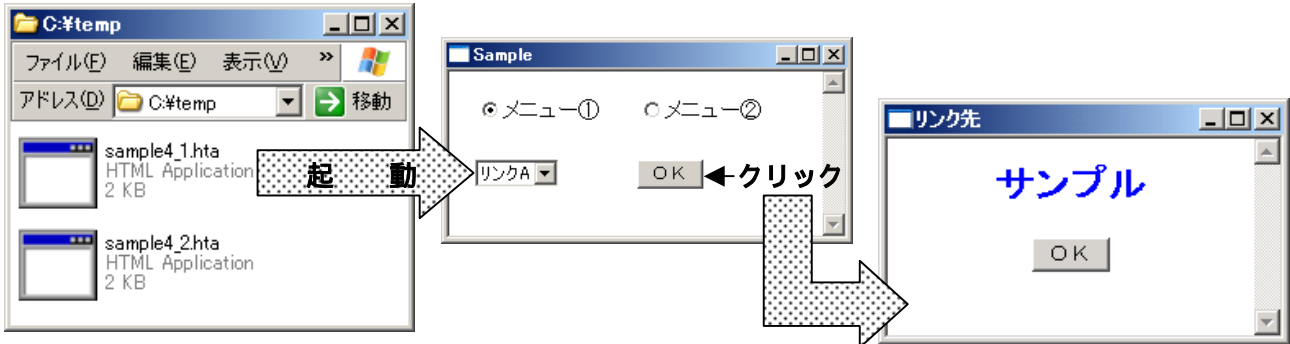


図 3.4 : sample1.hta

ツールの「メニュー、メニュー」をいずれかが選択し、プルダウンメニューからリンク先を選択して「OK」ボタンをクリックすると、別の HTML アプリケーションが起動される（簡単のため、HTML アプリケーションの種類は「sample4_2.hta」のみとした）。ツールの中身は以下のとおり。

プログラム 3.4 : ツール 「sample4_1.hta」

```

01: <html>
02: <title>Sample</title>
03: <head>
04: <script type="text/javascript">
05:   resizeTo(300,150);
06:   var MyArray1   = new Array("リンク A", "リンク B");
07:   var MyArray2   = new Array("リンク C", "リンク D");
08:   var MyValue1   = new Array("sample4_2.hta", "sample4_2.hta");
09:   var MyValue2   = new Array("sample4_2.hta", "sample4_2.hta");
10:   var ListNumber = 2;
11:   function $(_id) {
12:     return document.getElementById(_id) ;
13:   }
14:   function Switch (nType) {
15:     for (var i=0 ; i < ListNumber; i++) {
16:       if (nType == 1) {
17:         $("MySelect").options[i].text = MyArray1[i] ;
18:         $("MySelect").options[i].value = MyValue1[i] ;
19:       }
20:       else if (nType == 2) {
21:         $("MySelect").options[i].text = MyArray2[i] ;
22:         $("MySelect").options[i].value = MyValue2[i] ;
23:       }
24:     }
25:   }
26:   function Go() {
27:     MyShell = new ActiveXObject("WScript.Shell");
28:     MyShell.Run($("MySelect").value, 1, false);
29:   }
30: </script>
31: </head>
32: </html>
32: <body onLoad="Switch(1)">
33:   <table align="center" border="0" width="95%">
34:     <tr>
35:       <td width="50%"><input type="radio" name="MyRadio" id="MyRadio" onClick="Switch(1)" checked>
        メニュー </td>
36:       <td width="50%"><input type="radio" name="MyRadio" id="MyRadio" onClick="Switch(2)">
        メニュー </td>

```


- 例数設計くん.hta：本体となる HTML アプリケーション
- 「files」フォルダ：「例数設計くん」を動作させるための部品が入ったフォルダ。HTML アプリケーション、SAS プログラム、バナーや背景用の画像などが含まれている。

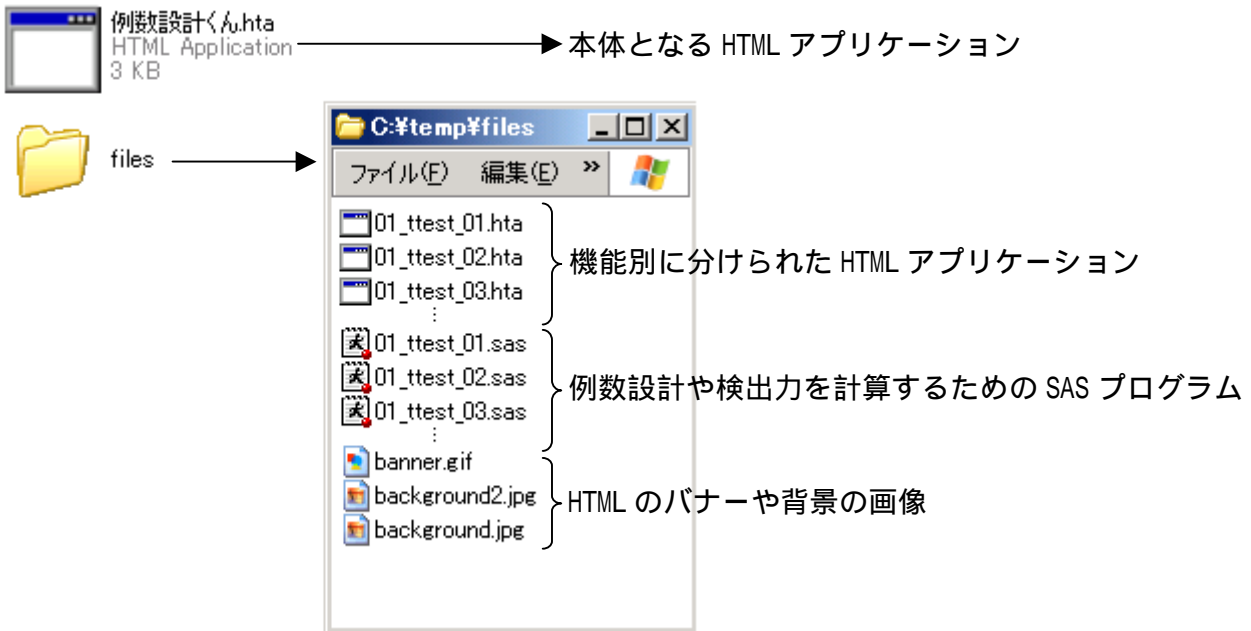


図 4.2：設定ファイルが入っている「files」フォルダの中身

「例数設計くん.hta」をクリックすると、図 4.3 のようなアプリケーションが起動される²。「例数の算出」「検出力の算出」「検出力カーブ」のいずれかを選択した後、例数設計等を行いたい検定手法を選択して「OK」ボタンをクリックする。

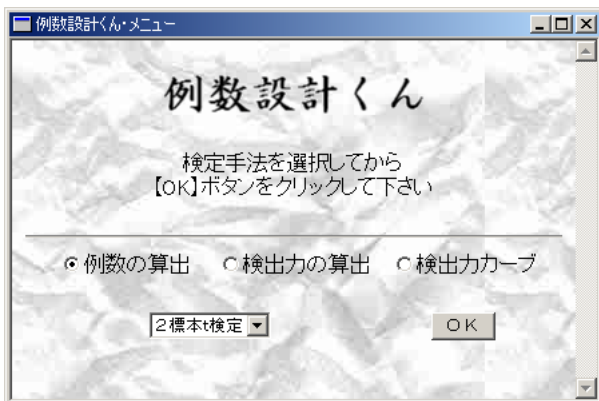


図 4.3：「例数設計くん」本体画面

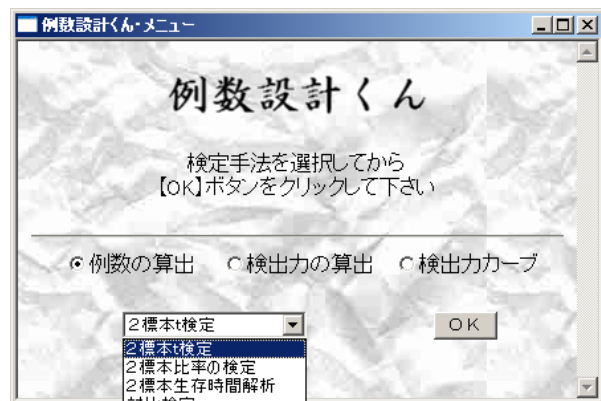


図 4.4：「例数設計くん」メニュー選択例

例数の算出

「例数設計くん.hta」から「例数の算出」「2標本t検定」を選択すると、「2標本t検定・例数設計 (01_ttest_01.hta)」が立ち上がる³。やなどの各パラメータの値を設定して「OK」ボタンをクリックすると、「2標本t検定の例数設計を実行します」のダイアログが出た

² 「例数設計くん.hta」のプログラムは「参考プログラム1」を参照のこと。

³ 「01_ttest_01.hta」のプログラムは「参考プログラム2」を参照のこと。

後、例数の算出プログラム (proc power) が実行され、結果が HTML ファイルとして表示される。

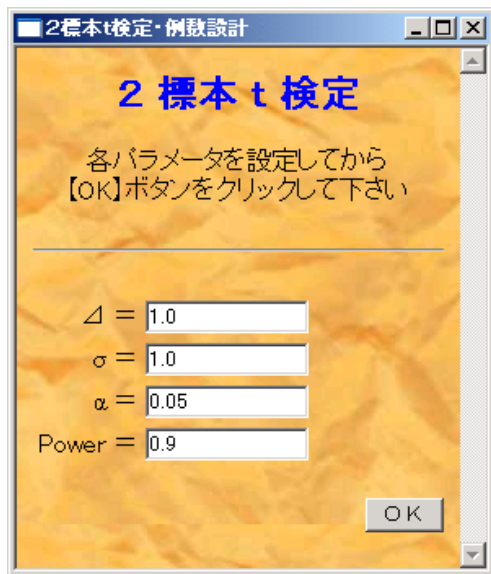


図 4.5 : 2 標本 t 検定・例数設計

SAS システム

The POWER Procedure
Two-sample t Test for Mean Difference

Fixed Scenario Elements	
Distribution	Normal
Method	Exact
Alpha	0.05
Mean Difference	1
Standard Deviation	1
Nominal Power	0.9
Number of Sides	2
Null Difference	0
Group 1 Weight	1
Group 2 Weight	1

Computed N Total	
Actual Power	N Total
0.912	46

図 4.6 : 実行結果

実行の流れを図 4.7 に示す。まず、「01_ttest_01.hta (図 4.5)」を実行すると、計算結果の出力先や検定手法の種類を SAS のマクロパラメータに格納する SAS プログラム「_EXE.sas」と、 Δ や σ などの各パラメータの値を SAS のマクロパラメータに格納する SAS プログラム「parameter.sas」が生成される。その後、「_EXE.sas」から例数の算出を行う SAS プログラム「01_ttest_01.sas」を実行し、例数の算出を行い、結果を「result.html」に出力する。ちなみに、本プログラムでは proc power を用いて計算を行ったが、data ステップで例数設計の公式を計算しても、シミュレーションにより例数設計を行っても構わない。

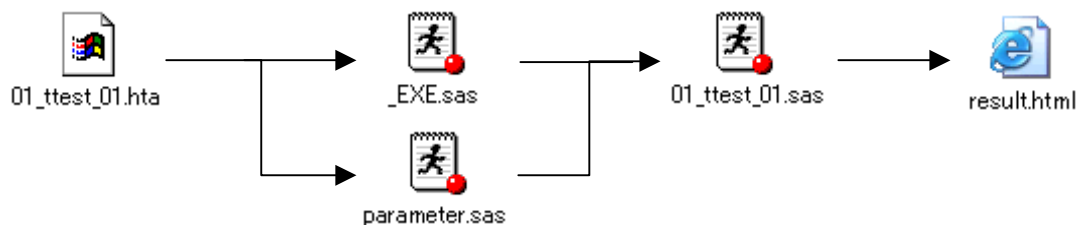


図 4.7 : 動作の流れ

参考までに、「_EXE.sas」と「parameter.sas」の中身を紹介する。

プログラム 4.1 : _EXE.sas

```
01: options source source2 symbolgen ;
02: %let _path1 = C:\temp\例数設計\ ;
03: %let _path2 = %sysfunc(tranwrd(&_path1,¥,/)) ;
04: %inc "./files/parameter.sas" ;
05: %inc "./files/01_ttest_01.sas" ;
```

プログラム 4.2 : parameter.sas

```
01: %let _meandiff = 1.0 ;
02: %let _stddev = 1.0 ;
03: %let _alpha = 0.05 ;
04: %let _power = 0.9 ;
```

検出力の算出

「例数設計くん.hta」から「検出力の算出」「2標本t検定」を選択すると、「2標本t検定・例数設計(01_ttest_02.hta)」が立ち上がる⁴。

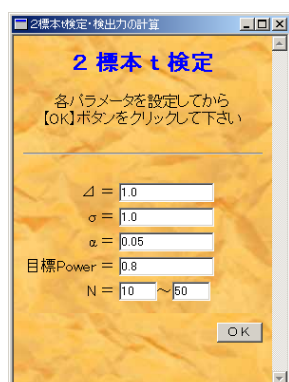


図 4.8 : 2 標本 t 検定・例数設計

The SAS POWER Procedure window shows the following configuration and results:

Fixed Scenario Elements	
Distribution	Normal
Method	Exact
Alpha	0.05
Mean Difference	1
Standard Deviation	1
Number of Sides	2
Null Difference	0
Group 1 Weight	1
Group 2 Weight	1

Computed Power		
Index	N Total	Power
1	20	0.562
2	22	0.607
3	24	0.649
4	26	0.687

図 4.9 : 実行結果

動作の流れは図 4.7 とほぼ同様である。「OK」ボタンをクリックすると、SAS プログラム「_EXE.sas」と「parameter.sas」が生成される。その後、「_EXE.sas」から例数の算出を行う SAS プログラム「01_ttest_02.sas」を実行し、例数の算出を行い、結果を「result.html」に出力する。

検出力カーブの作成

「例数設計くん.hta」から「検出力カーブ」「2標本t検定」を選択すると、「2標本t検定・検出力カーブ(01_ttest_03.hta)」が立ち上がる⁵。検出力カーブを SAS で描いてもよいが、ここでは SAS とその他のアプリケーションを組み合わせる例として、検出力の計算を SAS で行った後、結果を CSV ファイルに出力し、これを R に読み込ませて検出力カーブを描くことにする。

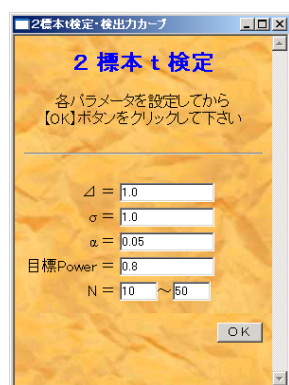


図 4.10 : 2 標本 t 検定・検出力カーブ

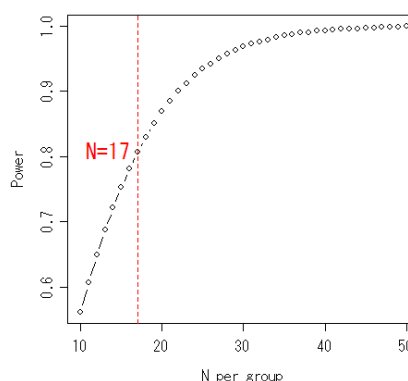


図 4.11 : 実行結果

実行の流れを図 4.12 に示す。まず、「01_ttest_03.hta (図 4.10)」を実行すると、計算結果の出力先や検定手法の種類を SAS のマクロパラメータに格納する SAS プログラム「_EXE.sas」と、
や などの各パラメータの値を SAS のマクロパラメータに格納する SAS プログラム「parameter.sas」が生成される。その後、「_EXE.sas」から例数の算出を行う SAS プログラム「01_ttest_03.sas」を実行すると、各例数における検出力の算出を行った後、結果を

⁴ 01_ttest_02.hta のプログラムは「参考プログラム 3」を参照のこと。

⁵ 01_ttest_03.hta のプログラムは「参考プログラム 4」を参照のこと。

「mypower.csv」に出力すると同時に、検出力カーブを描くための R プログラム「mypower.R」を生成する。最後に「mypower.R」を実行し、検出力カーブ「mypower.bmp」を出力する。

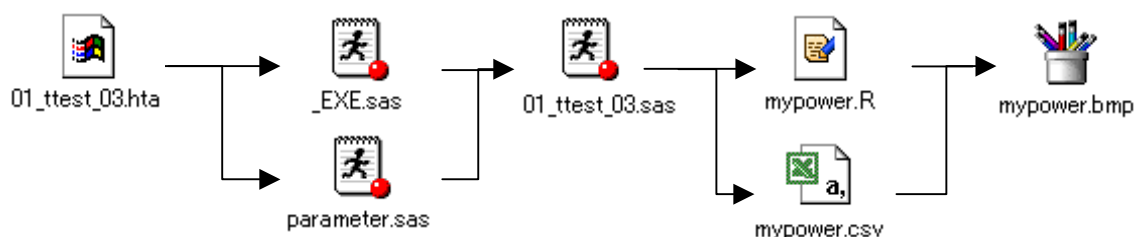


図 4.12：動作の流れ

参考までに、「mypower.R」と「mypower.csv」の中身を紹介する。

プログラム 4.3：mypower.csv

01:	10	,0.562
02:	11	,0.607
03:	12	,0.649
04:	13	,0.687
05:	14	,0.721
...
41:	50	,0.999

プログラム 4.4：mypower.R

01:	x <- read.csv("C:/temp/例数設計くん/files/mypower.csv", head=F)
02:	bmp(filename="C:/temp/例数設計くん/files/mypower.bmp")
03:	plot(V2~ V1, data=x, type="b", xlab="N per group", ylab="Power")
04:	abline(v=17, lty=2, col=2)
05:	text(17, 0.8070367151, labels="N=17", pos=2, cex=1.5, col=2)
06:	dev.off()

5. おわりに

HTML アプリケーションを使用することで、GUI を備えたアプリケーションツールを簡単に作成することが出来、本ツールを使用することで GUI 上で SAS プログラムを実行してレポートを自動出力することが実現出来る。本稿で紹介した方法を適用するメリットとして、ツールの作成およびメンテナンスが容易であり、ウェブサーバ等を必要としないため開発・保守費用がかからない、作成されたツールは SAS を習得していない人にも操作が出来るため、運用コストも抑えられることが挙げられる。デメリットとして、多言語に比べて HTML アプリケーションに関する資料が少ない、OS (Windows) バージョンアップによるプログラムの仕様変更が生じる可能性があることが挙げられる。ただ、メリットとデメリットを天秤にかけた場合、メリットの方が大きいと発表者は考える。ちなみに、本稿では説明を簡単にするため、例数設計用のツールの作成方法を例に挙げたが、臨床試験に関するデータを解析するためのツールを構築するときも、同様の方法が適用できることを最後に申し添える。

謝辞

株式会社 JIEC の長谷川 伸弘さんには、HTML の命令と Java 全般の内容について、貴重な助言を幾度となく頂戴した。この場をお借りしてお礼申し上げあげる。

連絡先

Funao_Nobuo@takeda.co.jp

Takanami_Yohei@takeda.co.jp

参考文献

- MSDN ライブラリ : Jscript , ActiveXObject の解説
<http://msdn2.microsoft.com/ja-jp/library/default.aspx>
- とほほの WWW 入門
<http://www.tohoho-web.com/www.htm>
- 佐藤 信正 (2007) 「JavaScript 完全マスター 再入門編」
(メディア・テック出版)
- 佐藤 信正 (2007) 「JavaScript+Ajax プログラミング・テクニック」
(メディア・テック出版)
- 浜田知久馬、安藤英一 (2005) 「POWER プロシジャによる症例数設計」
SAS Forum ユーザー会 学術総会 2005
- 舟尾 暢男、高浪 洋平 (2005) 「データ解析環境『R』」 (工学社)

【参考】「例数設計くん」で出てきたプログラム

参考プログラム 1 : 例数設計くん.hta

(簡単のため 2 標本 t 検定に関するもののみ選択する仕様とした)

<pre><html> <title>例数設計くん・メニュー</title> <head> <script type="text/javascript"> resizeTo(450,300); var MyArray1 = new Array("2 標本 t 検定"); var MyArray2 = new Array("2 標本 t 検定"); var MyArray3 = new Array("2 標本 t 検定"); var MyValue1 = new Array("01_ttest_01"); var MyValue2 = new Array("01_ttest_02"); var MyValue3 = new Array("01_ttest_03"); var ListNumber = 1; function \$(id) { return document.getElementById(id); } function Switch (nType) { for (var i=0 ; i < ListNumber ; i++) { if (nType == 1) { \$("MySelect").options[i].text = MyArray1[i]; \$("MySelect").options[i].value = MyValue1[i]; } else if (nType == 2) { \$("MySelect").options[i].text = MyArray2[i]; \$("MySelect").options[i].value = MyValue2[i]; } else if (nType == 3) { \$("MySelect").options[i].text = MyArray3[i]; \$("MySelect").options[i].value = MyValue3[i]; } } } function Go() { MyScript = new ActiveXObject("Scripting.FileSystemObject"); MyPath = MyScript.GetFolder(".").Path; MyText = MyScript.CreateTextFile("./files/_EXE.sas", true); MyText.WriteLine("options source source2 symbolgen;"); MyText.WriteLine("%let _path1 = " + MyPath + " ;"); MyText.WriteLine("%let _path2 = %sysfunc(tranwrd(&_path1,¥¥,¥));"); MyText.WriteLine("%inc ./files/parameter.sas;"); MyText.WriteLine("%inc ./files/' + \$("MySelect").value + '.sas";"); MyText.Close(); MyShell = new ActiveXObject("WScript.Shell"); MyShell.Run("./files/' + \$("MySelect").value + ".hta", 1, false); }</pre>	<pre></script> </head> <body background="files/background.jpg" onLoad="Switch(1)"> <p align="center"></p> <p align="center">検定手法を選択してから
【OK】ボタンをクリック して下さい</p> <hr> <table align="center" border="0" width="95%"> <tr> <table align="center" border="0" width="95%"> <tr> <td width="33%" align="center"><input type="radio" name="MyRadio" id="MyRadio" onClick="Switch(1)" checked>例数の算出</td> <td width="34%" align="center"><input type="radio" name="MyRadio" id="MyRadio" onClick="Switch(2)">検出力の算出</td> <td width="33%" align="center"><input type="radio" name="MyRadio" id="MyRadio" onClick="Switch(3)">検出力カーブ</td> </tr> <tr> <td colspan="3">&nbsp;&nbsp;&nbsp;</td> </tr> <tr> <td width="60%" align="center" colspan="2"> <select size="1" name="MySelect" id="MySelect"> <option>2 標本 t 検定</option> </select> </td> <td width="40%" align="center"><input type="button" name="btnGo" value=" O K " onclick="Go()"></td> </tr> </table> </body> </html></pre>
--	--

参考プログラム 2 : 01_ttest_01.hta

<pre> <html> <title>2標本t検定・例数設計</title> <head> <script type="text/javascript"> resizeTo(300,350); function MyGetParameter() { var MyVariable01, MyVariable02, MyVariable03, MyVariable04, MyScript, MyText, MyShell ; MyVariable01 = document.MyForm.Mytextbox01 ; MyVariable02 = document.MyForm.Mytextbox02 ; MyVariable03 = document.MyForm.Mytextbox03 ; MyVariable04 = document.MyForm.Mytextbox04 ; if (MyVariable01 != "" & MyVariable02 != "") { MyScript = new ActiveXObject("Scripting.FileSystemObject"); MyText = MyScript.CreateTextFile("./files/parameter.sas", true); MyText.WriteLine("%let _meandiff = " + MyVariable01.value + " ;"); MyText.WriteLine("%let _stddev = " + MyVariable02.value + " ;"); MyText.WriteLine("%let _alpha = " + MyVariable03.value + " ;"); MyText.WriteLine("%let _power = " + MyVariable04.value + " ;"); MyText.Close(); } window.alert("2標本t検定の例数設計を実行します"); MyShell = new ActiveXObject("WScript.Shell"); MyShell.Exec("C:/Program Files/SAS/SAS 9.1/sas.exe -sysin ./files/_EXE.sas -log ./files/_EXE.log -nosplash -icon"); } </script> </head> <body background="background2.jpg"> <p align="center"> 2 標本 t 検定</p> <p align="center">各パラメータを設定してから
 【OK】ボタンをクリックして下さい</p> <hr> </pre>	<pre> <form name="MyForm"> <table border=0> <tr><td align="right">Δ</td><td width=0>=</td> <td width=100> <input name="Mytextbox01" type="text" value="1.0" style="width:100%;"> </td> </tr> <tr> <td align="right">σ</td><td width=0>=</td> <td width=100> <input name="Mytextbox02" type="text" value="1.0" style="width:100%;"> </td> </tr> <tr> <td align="right">α</td><td width=0>=</td> <td width=100> <input name="Mytextbox03" type="text" value="0.05" style="width:100%;"> </td> </tr> <tr> <td align="right">Power</td><td width=0>=</td> <td width=100> <input name="Mytextbox04" type="text" value="0.9" style="width:100%;"> </td> </tr> </table> <p align="right"> <input name="Mygetbutton" type="button" value=" O K " onclick="MyGetParameter()"> </p> </form> </body> </html> </pre>
---	--

参考プログラム 3 : 01_ttest_02.hta

<pre> <html> <title>2標本t検定・検出力の計算</title> <head> <script type="text/javascript"> resizeTo(300,400); function MyGetParameter() { var MyVariable01, MyVariable02, MyVariable03, MyVariable04, MyVariable05, MyVariable06, MyScript, MyText, MyShell ; MyVariable01 = document.MyForm.Mytextbox01 ; MyVariable02 = document.MyForm.Mytextbox02 ; MyVariable03 = document.MyForm.Mytextbox03 ; MyVariable04 = document.MyForm.Mytextbox04 ; MyVariable05 = document.MyForm.Mytextbox05 ; MyVariable06 = document.MyForm.Mytextbox06 ; if (MyVariable01 != "" & MyVariable02 != "") { MyScript = new ActiveXObject("Scripting.FileSystemObject"); MyText = MyScript.CreateTextFile("./files/parameter.sas", true); MyText.WriteLine("%let _meandiff = " + MyVariable01.value + " ;"); MyText.WriteLine("%let _stddev = " + MyVariable02.value + " ;"); MyText.WriteLine("%let _alpha = " + MyVariable03.value + " ;"); MyText.WriteLine("%let _Power = " + MyVariable04.value + " ;"); MyText.WriteLine("%let _Nmin = " + MyVariable05.value*2 + " ;"); MyText.WriteLine("%let _Nmax = " + MyVariable06.value*2 + " ;"); MyText.Close(); } window.alert("2標本t検定の検出力を計算します"); MyShell = new ActiveXObject("WScript.Shell"); MyShell.Exec("C:/Program Files/SAS/SAS 9.1/sas.exe -sysin ./files/_EXE.sas -log ./files/_EXE.log -nosplash -icon"); } </script> </head> <body background="background2.jpg"> <p align="center"> 2 標本 t 検定</p> <p align="center">各パラメータを設定してから
【OK】ボタンを クリックして下さい</p> <hr> <form name="MyForm"> <table border=0> </pre>	<pre> <tr><td align="right">Δ</td><td width=0>=</td> <td width=100> <input name="Mytextbox01" type="text" value="1.0" style="width:100%;"> </td> </tr> <tr><td align="right">σ</td><td width=0>=</td> <td width=100> <input name="Mytextbox02" type="text" value="1.0" style="width:100%;"> </td> </tr> <tr><td align="right">α</td><td width=0>=</td> <td width=100> <input name="Mytextbox03" type="text" value="0.05" style="width:100%;"> </td> </tr> <tr><td align="right">目標 Power</td> <td width=0>=</td> <td width=100> <input name="Mytextbox04" type="text" value="0.8" style="width:100%;"> </td> </tr> <tr><td align="right">N</td><td width=0>=</td> <td width=100> <input name="Mytextbox05" type="text" value="10" style="width:40%;">~ <input name="Mytextbox06" type="text" value="50" style="width:40%;"> </td> </tr> </table> <p align="right"> <input name="Mygetbutton" type="button" value=" O K " onclick="MyGetParameter()"> </p> </form> </body> </html> </pre>
--	--

参考プログラム 4 : 01_ttest_03.hta

<pre> <html> <title>2標本t検定・検出力カーブ</title> <head> <script type="text/javascript"> resizeTo(300,400); <!-- function MyGetParameter() { var MyVariable01, MyVariable02, MyVariable03, MyVariable04, MyVariable05, MyVariable06, MyScript, MyText, MyShell ; MyVariable01 = document.MyForm.Mytextbox01 ; MyVariable02 = document.MyForm.Mytextbox02 ; MyVariable03 = document.MyForm.Mytextbox03 ; MyVariable04 = document.MyForm.Mytextbox04 ; MyVariable05 = document.MyForm.Mytextbox05 ; MyVariable06 = document.MyForm.Mytextbox06 ; if (MyVariable01 != "" & MyVariable02 != "") { MyScript = new ActiveXObject("Scripting.FileSystemObject"); MyText = MyScript.CreateTextFile("./files/parameter.sas", true); MyText.WriteLine("%let _meandiff = " + MyVariable01.value + " ;"); MyText.WriteLine("%let _stddev = " + MyVariable02.value + " ;"); MyText.WriteLine("%let _alpha = " + MyVariable03.value + " ;"); MyText.WriteLine("%let _Power = " + MyVariable04.value + " ;"); MyText.WriteLine("%let _Nmin = " + MyVariable05.value*2 + " ;"); MyText.WriteLine("%let _Nmax = " + MyVariable06.value*2 + " ;"); MyText.Close(); } window.alert("2標本t検定の検出力カーブを描きます"); MyShell = new ActiveXObject("WScript.Shell"); MyShell.Exec("C:/Program Files/SAS/SAS 9.1/sas.exe -sysin ./files/_EXE.sas -log ./files/_EXE.log -nosplash -icon"); } --> </script> </head> <body background="background2.jpg"> <p align="center"> 2 標本 t 検定</p> <p align="center">各パラメータを設定してから
【OK】ボタンを クリックして下さい</p> <hr> <form name="MyForm"> <table border=0> <tr> </pre>	<pre> <td width=0 align="right"></td><td width=0>=</td> <td width=100> <input name="Mytextbox01" type="text" value="1.0" style="width:100%;"> </td> </tr> <tr> <td width=0 align="right"></td><td width=0>=</td> <td width=100> <input name="Mytextbox02" type="text" value="1.0" style="width:100%;"> </td> </tr> <tr> <td width=0 align="right"></td><td width=0>=</td> <td width=100> <input name="Mytextbox03" type="text" value="0.05" style="width:100%;"> </td> </tr> <tr> <td width=0 align="right">目標 Power</td><td width=0>=</td> <td width=100> <input name="Mytextbox04" type="text" value="0.8" style="width:100%;"> </td> </tr> <tr> <td width=0 align="right"></td><td width=0>=</td> <td width=100> <input name="Mytextbox05" type="text" value="10" style="width:40%;">~ <input name="Mytextbox06" type="text" value="50" style="width:40%;"> </td> </tr> </table> <p align="right"> <input name="Mygetbutton" type="button" value=" O K " onclick="MyGetParameter()"> </p> </form> </body> </html> </pre>
--	---

参考プログラム 5 : 01_ttest_01.sas

```

ods html file = 'result.html'
      path = "&_path1" ;

proc power;
  twosamplemeans
  test      = diff
  meandiff  = &_meandiff.
  stddev    = &_stddev.
  alpha     = &_alpha.
  power     = &_power.
  ntotal    = . ;
run ;

ods html close;

x "&_path1¥result.html" ;

```

参考プログラム 6 : 01_ttest_02.sas

```

ods html file = 'result.html'
      path = "&_path1" ;

proc power ;
  twosamplemeans
  test      = diff
  meandiff  = &_meandiff
  stddev    = &_stddev
  alpha     = &_alpha
  power     = .
  ntotal    = &_Nmin to &_Nmax by 2 ;
run ;

ods html close;

x "&_path1¥result.html" ;

```

参考プログラム 7 : 01_ttest_03.sas

```

%let _RPGM = &_path2/files/mypower.R ;
%let _REMF = &_path2/files/mypower.bmp ;
%let _RCSV = &_path2/files/mypower.csv ;

ods output Output=Tmp ;
ods listing close ;

proc power ;
  twosamplemeans
  test = diff
  meandiff = &_meandiff
  stddev = &_stddev
  alpha = &_alpha
  power = .
  ntotal = &_Nmin to &_Nmax by 2 ;
run ;

ods output close ;
ods listing ;

filename RCSV "&_RCSV" ;

data MyPower ;
  set Tmp ;
  file RCSV ;
  Ntotal = Ntotal / 2 ;
  put Ntotal " " Power ;
run ;

data _NULL_ ;
  set MyPower end=_EOF ;
  retain _PFLG 0 ;
  if ( (Ntotal = &_Nmin/2 and Power > &_Power)
    or (Ntotal = &_Nmax/2 and Power < &_Power) ) then do ;
    _PFLG = 2 ;
  end ;
  else if Power > &_Power and _PFLG = 0 then do ;
    _PFLG = 1 ;
    call symput("_POWER",compress(put(Power,best)));
    call symput("_NTOT",compress(put(Ntotal,best)));
  end ;
  if _EOF then call symput("_FLG",compress(put(_PFLG,best)));
run ;

options noxwait xsync ;
filename RPGM "&_RPGM" ;

%macro drawgraph() ;
data _null_ ;
  file RPGM ;
  put "x <- read.csv("&_RCSV", head=F) ;
  put "bmp(filename="&_REMF) ;
  put "plot(V2~ V1, data=x, type="b", xlab="N per group", ylab="Power") ;
  %if &_FLG = 1 %then %do ;
    put "abline(v=&_NTOT, lty=2, col=2)" ;
    put "text(&_NTOT, &_POWER, labels="N=&_NTOT", pos=2, cex=1.5, col=2)" ;
  %end ;
  put 'dev.off()' ;
run ;

x "C:\Program Files\%R\R-2.6.2\pat\bin\R.exe" --no-restore --no-save < "&_RPGM" ;
%mend ;

%drawgraph() ;

x "&_REMF" ;

```

以上