

<http://cran.r-project.org/doc/manuals/R-intro.html> 의 Appendix B 에 대한 내용입니다.

## Appendix B Invoking R

윈도우와 Mac OS X 환경의 R 유저들은 OS 관련 설명 부분을 우선 읽어볼 것을 권장합니다. command-line 은 지원됩니다.

command line 에서 Invoking R

윈도우 환경에서의 Invoking R

MAC OS X 환경에서의 Invoking R

R 스크립트

### B.1 Invoking R from the command line

#### 1. command line 에서의 Invoking R

UNIX 혹은 윈도우 환경의 command line 작업 시, R 은 **R [options] [<infile] [>outfile]**의 형태 혹은 '직접적'이란 의미를 지니지 않는 다양한 R 툴에 대한 **Wrapper**의 역할로서, **R CMD** 인터페이스를 통해 사용이 될 수 있습니다.

윈도우 상의 command-line 에서, **Rterm.exe** 는 R 에 해당합니다.

환경 변수 **TMPDIR** 이 결정되지 않거나, 일시적 파일과 디렉토리 생성을 위한 공간을 해당 변수가 지정할 수 있도록 합니다.

대부분의 옵션들은 R 의 시작과 끝을 제어합니다. 시작 메커니즘은 아래와 같이 설명이 되어 있습니다. (추가 정보를 원하시면, 'startup'이라는 주제에 대한 온라인 도움말(help)을 활용해 보세요. 윈도우 환경 관련 구체적인 설명 또한 아래 부분에 기술되어 있습니다.)

- **-no-environ** 이 주어지지 않는다면, R 은 유저 환경변수설정을 위해서 **site files** 과 유저를 위해서 탐색합니다. 사이트 파일의 이름은 환경변수 **R\_ENVIRON**에 의해 지정됩니다. 이름이 결정되지 않은 경우, 존재한다면, **R\_HOME/etc/Renviron.site** 를 활용할 수 있습니다. 유저 파일의 경우, 결정이 되

었다면, 환경변수 **R\_ENVIRON\_USER** 에 의해 지정이 됩니다. 그렇지 않은 경우, 현재 혹은 유저의 홈 디렉토리(home directory)에 있는 **file.Renviron** 이 탐색됩니다. 이 파일들은 'name=value' 형태의 라인들을 포함해야 합니다. (좀 더 구체적인 이해를 위해서 "**Startup**" 도움말을 참조하세요) 지정하고 싶은 변수들은 **R\_PAPERSIZE**(default paper size), **R\_PRINTCMD** (default print command)와 **R\_LIBS**(추가 패키지를 위해 마련된 R library 트리 리스트의 세분화)를 포함해야 한다.

- 그 다음, command line 옵션 (**-no-site-file**)이 주어지지 않았다면, R 이 **site-wide startup profile** 을 탐색합니다. 이 파일의 이름은 **R\_PROFILE** 환경 변수의 벨류(value)에서 얻을 수 있습니다. 만약 그 변수가 정해지지 않았다면, default **R\_HOME/etc/Rprofile.site** 가 존재한다면 사용될 수 있습니다.

- 그 다음 **-no-init-file** 이 없다면, R 은 **user profile** 을 탐색 및 활용합니다. 이 파일의 이름은 **R\_PROFILE\_USER** 환경변수에서 얻을 수 있습니다. 그리고, 지정되지 않았다면, 현재 디렉토리나 유저의 홈 디렉토리(해당 목록)의 **.Rprofile** 파일을 탐색할 수 있습니다.

- 현재 디렉토리 상의 **.RData file** 이 존재한다면, 해당 디렉토리에 있는 **.RData** 파일로부터 저장된 작업공간(workspace)을 로딩할 수 있습니다. (**-no-restore** 혹은 **-no-restore-data** 가 구분되어 있지 않은 경우)

- 마지막으로, **.First()** 함수가 존재한다면, 실행됩니다. 이 함수는 (R 설명 마지막 부분에서 실행되었던 **.Last()** 포함) 적절한 **startup profile** 정의(define)될 수 있고 혹은 **.RData** 에 상주할 수가 있습니다.

추가적으로, R 처리과정에서 이용 가능한 메모리 제어를 위한 옵션이 있습니다. (추가 정보를 원하신다면, '**Memory**' 토픽 부분의 온라인 도움말을 참고하시기 바랍니다.) R 에 의해 사용되는 메모리 양을 제한하려 하지 않는 이상 이 옵션들을 사용해야 할 필요는 굳이 없습니다.

R 에서 아래의 **command-line** 옵션들을 사용할 수 있습니다.

```
--help
-h
```

표준출력과 **exit** 에 대한 짧은 도움말 메시지를 출력(Print)

--version

표준출력과 **exit** 에 대한 버전 정보를 출력

--encoding=*enc*

콘솔 혹은 **stdin** 의 **input** 을 위해서 인코딩(**encoding**)을 구체화해야 합니다. 이는 **iconv** 라고 불리는 인코딩이 필요합니다. 도움말 페이지를 참조하세요. (**--encoding enc** 도 사용될 수 있습니다.)

RHOME

**R home directory** 를 표준출력과 **exit** 을 위해서 **R "home directory"**에 대한 경로 출력을 합니다. **front-end shell** 스크립트 와 매인 페이지를 제외하고, **R installation** 은 모든 것을 이 디렉토리 안에 포함시킵니다.

--save

--no-save

데이터가 **R** 마지막 부분에 저장되어야 할지 여부에 대해서 결정합니다. **non-interactive use** 상의 **q()**와 **session** 을 끝낼 때, **interactive session** 에 주어지지 않거나, 필요한 조치가 요구되지 않는다면, 다른 옵션들에 의해서 구체화되거나 전달되어야 합니다.

--no-environ

환경변수 설정을 위한 유저 파일 불러오기는 하지 않습니다.

--no-site-file

**startup** 상에서 **site-wide** 프로파일을 불러오지 않습니다.

--no-init-file

**startup** 상에서 **user** 프로파일을 불러오지 않습니다.

--restore

--no-restore

--no-restore-data

저장된 이미지 파일이 **startup** 상에서의 **restore** 여부를 결정합니다.(**R** 이 시작된 디렉토리 안에 **file .RData** 존재) 디폴트는 **restore** 되어야 합니다. (**--no-restore** 는 모든 **--no-restore-\*** 옵션들을 의미합니다.

--no-restore-history

히스토리 파일이 **restore** 되어야 하는지 여부 결정합니다. (**R** 이 시작된 디렉토리에 일반적으로 **file .Rhistory** 가 존재하며, 환경변수 **R\_HISTFILE** 에 의해 설정될 수 있습니다.) 디폴트는 **restore** 되어야 합니다.

--no-Rconsole

(윈도우 환경)**startup** 시, **Rconsole** 파일 로딩 예방

--vanilla

**-no-save, -no-environ, -no-site-file, -no-init-file** 과 **-no-restore** 를 결합합니다.

윈도우 상에서, 이는 또한 **-no-Rconconsole** 포함합니다.

**-f file** 

**--file=file**

**file** 로부터 **input** 을 가져오세요. '.'은 **stdin** 을 의미하며, **--save** 가 설정되지 되어 있지 않다면, **-no-save** 를 나타냅니다. **Unix** 와 같은 플랫폼 상에서는, **file** 상의 **shell metacharacters** 은 피해야 하는 요소입니다. (하지만, **R2.14.0** 에서와 같이, 공간이 허용됩니다.)

**-e expression**

(**Not Rgui.exe**) **input line** 의 형태로 **expression** 을 활용하세요. 하나 혹은 그 이상의 **-e options** 이 사용될 수 있지만, **-f** 혹은 **-file** 과 함께 사용될 순 없습니다. 그리고, 이는 **-save** 가 설정되지 않는 한 **-no-save** 를 의미합니다. (이와 같은 방식으로, 전체 **expression** 가 **10,000** 바이트 정도가 최대라고 할 수 있습니다.

**expression** 은 **space** 혹은 **shell metacharacters** 를 가지고 있으며, 인용될 필요가 있습니다.)

**--no-readline**

**readline** 을 활용한 **commend-line editing** 을 끄시기 바랍니다. (유닉스 환경). 이는 **ESS(Emacs Speaks Statistics)**를 사용하는 **Emacs** 안에서 **R** 이 활용될 때 유용합니다. 추가 정보를 원하시면, **commend-line editor** 를 참조하시기 바랍니다. **commend-line editing** 은 디폴트 인터랙티브 유즈(**default interactive use**)를 통해 가능합니다. (**-interactive** 참조) 이 옵션은 **tilde-expansion** 에 영향을 끼칩니다. **path.expand** 에 대한 내용은 도움말을 참조하세요.

**--min-vsize=N**

**--min-nspace=N**

고급사용을 위해서, 각각 **cons cells** 과 **vector heap**(바이트 상)의 **garbage collection** 에 대한 초기 **trigger size** 를 설정합니다. 접미사 "**M**"은 각각 메가바이트(**megabyte**) 혹은 수백만 개의 **cell** 을 구체화합니다. 디폴트는 각각 **6Mb, 350k** 입니다.

**--max-ppsize=N**

**N** 로케이션에 해당하는 **pointer protection stack** 의 최대 사이즈를 구체화합니다. 이는 **10000** 에 디폴트 됩니다. 하지만, 대규모의 복잡한 산술을 위해서 증가될 수 있습니다. 현재 최대 밸류 사이즈는 **100000** 입니다.

**--max-mem-size=N**

(윈도우 상에서) **R** 객체들과 작업 영역을 위해 사용되는 양의 메모리에 대한 한계를 정합니다. 이는 **machine** 에서 그리고 **32-bit R** 을 위해서 **physical RAM** 의 상대

적으로 작은 양에 디폴트가 설정되어 있습니다. 그리고, 반드시 **32MB** 사이에 위치해야 하며, 윈도우 버전에서 할당된 최대량입니다.

--quiet

--silent

-q

저작권 및 환영 메시지가 포함된 부분은 출력하지 않습니다.

--slave

**R** 을 최대한 차분히 실행합니다. 이 옵션은 프로그램의 산술결과를 위해 **R** 을 사용하는 프로그램을 지원합니다. 이는 **-quiet** 을 의미하며, **-no-save** 를 의미합니다.

--interactive

(유닉스 상에서) **input** 이 재설정된다 하더라도 **R** 이 쌍방향으로 실행된다는 것을 보여줍니다. **input** 이 **FIFO** 혹은 **pipe** 혹은 쌍방향 프로그램으로부터 사용이 될 때, **-interactive** 를 사용 합니다. (디폴트는 **stdin** 이 터미널(**terminal**) 혹은 **pty** 에 연결이 되어 있을 때, **R** 이 쌍방향으로 실행된다는 상황에서 **deduce** 로 설정되어 있습니다.) **-e,f** 혹은 **-file** 의 활용은 **-interactive** 가 주어진 상황에서 **non-interactive** 사용을 하고자 합니다.

--ess

(윈도우 상에서) **command-line editor** 없이 **interactive** 사용을 포함한, **ESS** 내에서 **R-inferior-mode** 를 통해, **Rterm up** 을 설정합니다.

--verbose

진행과정에 대한 추가 정보를 프린트합니다. 그리고, **R** 의 옵션을 **TRUE** 에 **verbose** 합니다. **R** 코드는 진단 메시지의 프린팅 제어를 위해서 이 옵션을 사용합니다.

--debugger=*name*

-d *name*

(유닉스상에서) **name** 이라는 **debugger** 를 통해서 **R** 을 실행하세요. (**valgrind** 와 최근 **gdb** 버전은 예외) 대부분의 **debugger** 들을 위해서, 추가 **command line** 옵션은 무시될 것이며, 대신, **debugger** 내부를 통해서 **R** 이 실행될 때, 주어집니다.

--gui=*type*

-g *type*

(유닉스 환경) **GUI** 로서 **type** 을 사용합니다. (이 옵션은 쌍방향 그래픽을 포함하고 있습니다. ) 현재, **type** 을 위한 가능한 밸류는 '**Tcl/Tk**' 지원이 가능하다는 전제 하에, '**X11**'입니다. (**back-compatibility** 를 위해서, '**x11**'과 '**tk**'가 사용될 수 있습니다. )

--arch=*name*

(유닉스 환경) 구체화된 **sub-architecture** 를 실행합니다. 가능한 밸류가 'i386', 'x86\_64' 그리고 'ppc'를 포함하는 Mac OS X 환경에서 가장 일반적으로 쓰입니다.

--args

이 플래그(flag)는 나머지 **command line** 이 넘어가게 만드는 걸 제외하면 다른 기능을 포함하진 않습니다. 이는 **commandArgs(TRUE)**와 함께 **args** 로 부터 밸류를 되찾는데 있어 유용한 옵션입니다.

**input** 과 **output** 이 일반적으로는 재설정된다는 것을 주지합니다. ('<'와 '>'사용) 하지만, **4095** 바이트로 제한되어 있는 라인 길이는 여전히 적용됩니다. 경고와 에러 메시지는 에러 채널(**stderr**)로 보내집니다.

R CMD *command args*

**R CMD** 는 **R** 과 함께, 유용한 다양한 툴의 **invocation** 을 허용합니다. 하지만, 직접적 이진 않습니다. 일반적인 형태는 **R CMD command args** 이고, **command** 는 그 툴의 이름이며 **args** 는 그 이름에 삽입되어 있는 아규먼트(argument)입니다.

현재, 아래와 같은 툴들을 사용할 수 있습니다.

BATCH

**batch** 모드에서의 **R** 실행. 가능한 옵션들과 함께 **R -restore -save** 를 실행

COMPILE

(유닉스 환경) **R** 과의 활용을 위해서 **C**, **C++**, 포트란 파일들을 편집

SHLIB

다이나믹 로딩을 위한 공유하고 있는 **library** 를 구축

INSTALL

추가 패키지 설치

REMOVE

추가 패키지 제거

build

추가 패키지 구축

check

추가 패키지 점검

LINK

(유닉스 환경) 실행가능한 프로그램 생성을 위한 **Front-end**

Rprof

**Post-process R** 프로파일링 파일

Rdconv

Rd2txt

**HTML, LaTeX, 일반 text** 를 포함, 다양한 포맷으로 **Rd** 포맷을 전환 그리고, 예를 추출. **Rd2txt** 는 **Rd2conv -t txt** 를 위한 **shorthand** 로서 사용됩니다.

Rd2pdf

**Rd** 포맷을 **PDF** 로 전환

Stangle

**Sweave** 문서로부터 **S/R** 코드를 추출

Sweave

**Sweave** 문서 프로세스

Rdiff

**headers** 를 무시하는 **R output** 을 diff

config

**Configuration information** 입수

javareconf

(유닉스 환경) 자바 **configuration** 변수 업데이트

rtags

(유닉스 환경) **C,R,Rd** 파일로 부터 **Emacs-style tag** 파일을 생성

open

(윈도우 환경) 윈도우 파일 **association** 을 통한 파일 오픈

texify

(윈도우 환경) **R** 형태의 파일을 활용한 **(La)Tex** 파일 프로세스

Use

R CMD *command* --help

**R CMD command--help**

**R CMD** 인터페이스를 통한 접근 가능한 각각의 툴을 위한 사용정보

추가로, **-arch-**, **-no-envron**, **-no-init-file**, **-no-site-file**, **-vanilla** 를 **R** 과 **CMD** 에서 사용할 수 있습니다. 이 옵션들은 툴에 의해 실행되는 **R** 에 영향을 끼칩니다. (**-vanilla** 는 **-no-envron**, **-no-site-file**, **-no-init-file** 과 동일 합니다.) 하지만, **R CMD** 는 **R startup file** 을 사용하지 않습니다. (특히, 유저 그리고 **site Renvron** 파일에 적용은 되지 않습니다.) 그리고, 모든 **R** 프로세스들은 이러한 툴들에 의해 실행됩니다. (**BATCH** 제외) **-no-restore** 를 사용합니다. 대부분은 **-vanilla** 를 사용하고 **R startup file** 을 **invoke** 하지 않습니다. 현

재 예외라고 할 수 있는 부분은 **INSTALL.REMOVE**, **Sweave** 그리고 **SHLIB** 입니다.  
(**SHLIB** 는 **-no-site-file -no-init-file** 을 사용합니다.)

R CMD *cmd args*

경로에 위치해 있는 다른 실행 가능한 **cmd** 와 **absolute filepath** 를 고려하여, 이 옵션은 R 과 같은 환경을 위해서 유용합니다. 그리고 특정 **command** 는 예를 들어, **1dd** 혹은 **pdflatex** 실행을 위해서 실행됩니다. 윈도우 상에서 **cmd** 는 실행가능 혹은 **batch** 파일의 형태입니다. 그리고, 확장성을 가지고 있다면, **.sh** 혹은 **.pl** 이 실행을 위해 **call** 됩니다.

## B.2 Invoking R under Windows

### B. 2 윈도우 상에서의 Invoking R

윈도우 상에서 R 을 실행하는 방법은 두 가지가 있습니다. **cmd.exe** 혹은 더 강력한 **shell** 과 같은, 터미널 윈도우 내에서, 전 장에서 설명 드렸던 방법이 사용될 수 있습니다. 바로, **R.exe** 그리고 **Rterm.exe** 을 실행하는 것입니다. 쌍방향 사용목적을 위해서, 콘솔 기반의 **GUI(Rgui.exe)**도 있습니다.

윈도우 상에서 초기 프로시저는 유닉스 상에서의 방식과 유사합니다. 하지만, '**home directory**'에 대한 레퍼런스는 좀 더 명확할 필요가 있습니다. 윈도우 상에서 항상 정의가 되지 않기 때문입니다. 만약, 환경변수 **R\_USER** 가 정의되면, **home directory** 를 제공합니다. 다음으로, 만약 환경변수 **HOME** 이 정의되면, 이는 또한 **home directory** 를 제공합니다. 이 두 유저 컨트롤링 셋팅 완료 후, R 은 **home directory** 를 정의한 시스템을 파악합니다. 우선, 윈도우 "**personal**" **directory** 를 사용코자 합니다, (일반적으로, **C:\Documents and**

**Settings\username\My Documents in Windows XP**) 만약, 실패 시, 환경변수 **HOMEDRIVE** 와 **HOMEPATH** 가 정의되며, 이들은 일반적으로 **home directory** 를 정의하는 변수들입니다. 이 모든 과정이 실패할 경우, **home directory** 는 **starting directory** 로서, 사용됩니다.

환경변수 **TMPDIR**, **TMP** 혹은 **TEMP** 가 결정되지 않거나, 이 둘 중 하나가 일시적인 파일 혹은 디렉토리 생성을 위한 유효한 공간을 지정하게끔 설정해야 합니다.

환경변수들은 **command line** 에서 '**name=value**' 페어로서 제공될 수 있습니다.

**.RDATA** 로 끝나는 아큐먼트가 존재할 경우, 이는 되찾기 위한 작업공간에 대한 경로로서, 인식됩니다. 이는 **-restore** 를 의미하며 해당 파일의 상위에 작업 디렉토리를 설정합니다. (이 메커니즘은 **RGui.exe** 와 관련 있는 파일 혹은 **drag-and-drop** 을 위해 사용됩니다. 하지



만, 또한 **Rterm.exe** 를 위해 활용되며, 해당 파일이 존재하지 않고, 상위 디렉토리가 존재할 경우, **working directory** 를 설정합니다.

추가 **command-line** 옵션은 **RGui.exe** 를 **invoking** 할 때, 이용 가능합니다.

--mdi

--sdi

--no-mdi

**Rgui** 가 **MDI** 프로그램(하나의 주 윈도우 상에 여러 개의 작은 윈도우 창) 혹은 **SDI** 어플리케이션(콘솔, 그래픽 혹은 페이지를 위한 다수의 **top-level** 윈도우)으로서, 실행되는지 여부 판단. 커맨드라인 설정은 유저의 **Rconsole** 파일 상의 설정보다 우선합니다.

--debug

**Rgui** 에서 "**Break to debugger**" 메뉴 아이템을 이용할 수 있게 제공 하고 **command line** 처리하는 동안 **break** 를 **debugger** 를 위해서 작동합니다.

윈도우 상에서, **R CMD** 를 활용하여, **.bat**, **exe**, **.sh** or **.pl** 파일들을 구체화할 수 있습니다. **R\_HOME**, **R\_OSTYPE**, **PATH**, **BSTINPUTS**, **TEXINPUTS** 포함해, 이미 설정된 일부 환경변수들과 함께, 적절한 **interpreter** (**Perl** for **.pl**)하에서 실행될 수 있습니다. 예를 들어, 만약 자신의 경로에 이미 **latex.exe** 가 존재한다면, 그 다음으로,

```
R CMD latex.exe mydoc
```

**TEXINPUTS** 에 **macros appended** 되어 있는 **R** 의 **share/texmf** 경로와 함께 **mydoc.tex** 상에서 **LaTeX** 를 실행할 수 있습니다. (안타깝게도, **MiKTeX LaTeX** 는 적용이 되지 않습니다. 하지만, **R CMD texify mydoc** 은 이 상황에 작동할 수 있습니다.)

## B.3 Invoking R under Mac OS X

### Mac OS X 상에서의 R invoking

**Mac OS X** 상에서 **R** 을 실행할 수 있는 방법에는 두 가지가 있습니다. 첫 번째, **subsection** 에서 설명된 것처럼 터미널 안에서, **R invoking** 을 통한 **.app window** 가 그것이며, 콘솔기반의 **GUI(R.app)**가 또 다른 방법입니다. 이는 시스템 상의 어플리케이션 폴더 안에 디폴트로 설치가 되어 있습니다. 이는 표준 더블 클릭(**standard double-clickable**) **Mac OS X** 어플리케이션입니다.

Mac OS X 상에서 **startup** 프로시저는 유닉스 상의 그것과 비교해서 유사합니다. **home directory** 는 **R.framework** 내부에 존재합니다. 하지만, **GUI** 내에서 **preference window accessible** 에 다른 **startup directory** 가 없는 경우, **startup** 과 현재 **working directory** 는 유저의 **home directory** 로서 설정됩니다.

## B.4 Scripting with R

### R Scripting

R 커맨드의 **foo.R** 파일을 실행하고자 한다면, **R CMD BATCH foo.R** 을 활용하는 방법을 생각해볼 수 있습니다. 이 파일을 표면적으로 드러나지 않게 실행하길 원하거나, **batch** 작업으로 하길 원한다면, **OS-specific facilities** 를 사용합니다. 예를 들어, 유닉스와 같은 **OSes** 상에서 **R CMD BATCH foo.R** 혹은 **background job** 으로 실행합니다.

커맨드 라인에서 추가 아규먼트들을 통해서 스크립트를 위해서 파라미터를 넘겨 줄 수 있습니다. 예를 들어,(where the exact quoting needed will depend on the shell in use)

```
R CMD BATCH "--args arg1 arg2" foo.R &  
args <- commandArgs(TRUE)
```

**R CMD BATCH "--args arg1 arg2" foo.R &** 는 **args <- commandArgs(TRUE)**에 의한 **character vector** 로서, 되찾을 수 있는 스크립트에 아규먼트를 넘겨 줄 수 있습니다.

```
Rscript foo.R arg1 arg2
```

이는 **Rscript foo.R arg1 arg2** 에 의해 **invoked** 될 수 있는 **alternative front-end Rscript** 에 의해 간소화 될 수 있습니다.

```
#!/path/to/Rscript  
args <- commandArgs(TRUE)  
...  
q(status=<exit status code>)
```

그리고 (적어도 유닉스와 같은 환경에서, 그리고 일부 **Window shell** 에서) 실행가능한 스크립트를 생성하기 위해서 사용될 수 있습니다.

```
runfoo arg1 arg2
```

만약 `runfoo text file` 으로 들어간다면, 이는 `chmod 755 runfoo` 에 의해 실행 가능하게 되며, `runfoo arg1 arg2` 에 의해서 다른 아규먼트들을 대한 `invoke` 를 실행할 수 있습니다.

추가 옵션을 위해서, `help("Rscript")`를 찾아 봅니다. 이는 `R output` 을 `stdout` 그리고 `stderr` 로 `write` 할 수 있으며, 그 커맨드를 실행시키는 `shell` 을 위해서 일반적인 방식으로 재설정 될 수 있습니다.

```
#!/usr/bin/env Rscript
```

...

**Rscript** 로의 경로를 **hardcode** 하지 않고, 경로 상에만 위치하게끔 하고자 한다면, `#!/usr/bin/env Rscript` 을 사용합니다. (이 상황은 윈도우를 제외한 일반적인 경우이며, 하지만, **Mac OS X** 유저들은 `/usr/local/bin` 을 경로에 추가할 필요가 있습니다. )

적어도, **Bourne** 와 **bash shell** 에서, `#!` 메커니즘은 `#!/usr/bin/env Rscript` **--vanilla** 와 같은 추가 아규먼트들을 허용하지 않습니다.

한가지 고려할 사항은 `stdin()`이 무엇을 의미하는지 여부입니다.

```
chem <- scan(n=24)
  2.90 3.10 3.40 3.40 3.70 3.70 2.80 2.50 2.40 2.40 2.70 2.20
  5.28 3.37 3.03 3.03 28.95 3.77 3.40 2.20 3.50 3.60 3.70 3.70
```

와 같은 세그먼트들과 함께 **R** 스크립트를 생성하는 것은 일반적인 것입니다.

그리고 `stdin()`은 이러한 기존의 사용방식을 허용하기 위해서 스크립트 파일을 의미합니다. `stdin` 을 참조하고 싶다면, 이를 **file connection** 으로 사용합니다 (예, `scan("stdin"..)`)

**Francois Pinard** 에 의해 마련된 스크립트 파일을 만들기 위한 다른 방법은

```
#!/bin/sh
[environment variables can be set here]
R --slave [other options] <<EOF
```

**R program goes here...**

**EOF**

와 같은 다큐먼트를 사용하는 것입니다. 하지만하지만, `stdin()`은 프로그램 소스를 의미하며, `stdin` 을 사용할 수 없습니다. **e-flag** 를 통해서 **command line** 상에서 짧은 스크립트들

은 **Rscript** 로 보내질 수 있습니다. 유닉스와 같은 환경에서 **foo.R** 과 같은 **input filename** 은 공간을 포함하거나, **shell metacharacter** 를 포함해서는 안됩니다.