R FAQ

KRUG

원문출처: http://cran.r-project.org/doc/FAQ/R-FAQ.html

R FAQ (Frequently Asked Questions on R) test-faq-wiki

Version 2.14.2012-02-23 ISBN 3-900051-08-9 Kurt Hornik

목차

Introduction

이 문서는 R 을 사용함에 있어서 사용자들이 많이 묻는 질문들을 정리해 놓은 문서입니다.

Legalese


```
http://www.R-project.org/Licenses/
```

Obtaining this document

본 문서에 대한 최신버전은 아래의 주소로부터 확인할 수 있습니다.

```
http://CRAN.R-project.org/doc/FAQ/
```

이 문서는 GNU Texinfo 시스템을 통하여 Texinfo 소스로부터 plain ASCII 텍스트, GNU info, HTML, PDF 와 같은 다양한 형식의 문서들을 생성할 수 있으며, CRAN 사이트내에서 doc/FAQ 라는 하위디렉토리로부터 얻을 수도 있습니다.

Citing this document

만약 이 문서를 출판의 목적으로 이용되게 된다면 위에서 제공된 URL 과 ISBN 3-900051-08-9 라는 정보를 Hornik (2012), “The R FAQ” 와 함께 참고목록에 나열해 주시면 됩니다. 이용의 편의를 위하여 아래의 BibTeX 를 제공합니다.

```
@Misc{,
  author       = {Kurt Hornik},
  title        = {The {R} {FAQ}},
  year         = {2011},
  note         = {{ISBN} 3-900051-08-9},
  url          = {http://CRAN.R-project.org/doc/FAQ/R-FAQ.html}
}
```

Notation

이 문서에서는 'R>' 은 R 프롬프트를 나타내고, '$' 은 쉘프롬프트를 의미합니다.

Feedback

이 문서에 대한 제안 및 의견사항은 Kurt.Hornik@r-project.org 로 보내주시길 바랍니다. 그러나, 이 문서의 작성자는 Windows 와 Mac OS X 시스템에 대한 엑세스가 없으므로, 이들에 대한 자세한 사항은 "R for Windows FAQ" 와 "R for Mac OS X FAQ" 를 참고하시길 바랍니다. 만약, Macintosh 또는 Windows 시스템에 관계된 정보중 이 문서에 반드시 필요하다고 생각된다면 알려주시길 바랍니다.

한국어 사용자님들께서는 이 문서에 대한 문의 및 제안사항을 R Development Translation Team 의 한국어 담당 이철희 (Chel Hee Lee, gnustats@r-project.kr) 으로 보내주시면 됩니다.

R Basics

What is R?

R 은 통계 계산 및 그래픽을 위한 시스템으로서, 프로그래밍 언어, 그래픽 실행환경, 디버거, 특정시스템에 대한 접근과 함께 스크립트 파일로 저장된 프로그램등을 실행시키는 요소들로 구성되어 있습니다. R 시스템 디자인은 크게 Becker, Chambers & Wilks' S 언어와 Sussman's Scheme 언어으로부터 큰 영향을 받았습니다. (S 언어에 대해서 알고 싶으시다면 What is S? 섹션을 보시길 바랍니다). 결과적으로 보이는 R 언어는 S 언어와 매우 흡사하지만, 근본적인 구현방법과 시맨틱은 Scheme 언어로부터 유래되었습니다. 더 자세한 사항에 What are the differences between R and S? (R 과 S 언어가 다른 점은 무엇인가?)라는 섹션을 참고하세요.

R 언어의 핵심요소라 할 것 같으면 분기와 반복, 그리고 함수를 이용한 모듈 프로그래밍을 지원하는 인터프리터 형식의 컴퓨터 언어라는 것입니다. R 시스템에서 사용자가 사용하는 대부분의 함수들은 R 로 작성된 것입니다. 또한 R 은 사용자가 C, C++, 혹은 Fortran 언어로 작성된 프로시져들과 효율적으로 인터페이스하는 것을 가능하게 해줍니다. 그리고, R 배포판은 많은 양의 통계프로시져들에 대한 기능들을 함께 포함하고 있습니다. 예를들면, 선형과 일반선형모델, 비선형회귀분석, 시계열분석, 모수와 비모수 테스트, 클러스터링과 스무딩을 들수 있습니다. 또한 데이터에 대한 다양한 시각화를 위한 방대하고도 유연한 그래픽 환경을 제공해 줍니다. 더 나아가, 다양한 목적으로 개발되어지는 "add-on packages" (애드온 패키지) 라고 불리는 추가적인 모듈들도 제공됩니다. (R Add-on Package 섹션을 참고해주세요).

가장 최초 R 은 뉴질랜드의 Auckland 시에 있는 Auckland 대학의 통계학과 소속의 Ross Ihaka 와 Robert Gentleman 에 의해서 개발되었습니다. 그리고, 많은 개인사용자 그룹이 코드를 보내고, 버그를 리포트 함으로서 R 에 기여하고 있습니다. 1997년 중반부터 R 소스코드 변경할 수 있는 권한을 가진 "R Core Team" 이라고 불리는 코어그룹이 활동해 오고 있습니다. 현재 이 그룹은 Doug Bates, John Chambers, Peter Dalgaard, Seth Falcon, Robert Gentleman, Kurt Hornik, Stefano Iacus, Ross Ihaka, Friedrich Leisch, Uwe Ligges, Thomas Lumley, Martin Maechler, Duncan Murdoch, Paul Murrell, Martyn Plummer, Brian Ripley, Deepayan Sarkar, Duncan Temple Lang, Luke Tierney, 그리고 Simon Urbanek 입니다. R 은 현재 http://www.r-project.org/ 라는 공식 홈페이지를 가지고 있으며, GNU 를 따르는 카피라이트 프리하에 배포되는 자유 소프트웨어입니다 (실은 "GNU S" 라고 불리는 GNU 공식 프로젝트 중의 하나입니다).

What machines does R run on?

R 은 유닉스에 기반을 둔 운영체제 및 Windows, 그리고 Mac 을 기반으로 개발되어졌습니다. 그러나, Mac OS 클래식에 대한 지원은 R-1.7.1 이후 중단되었습니다.

현재 배포되는 R 버전은 cpu-linux-gnu 를 포함하여, i386, amd64, alpha, arm/armel, hppa, ia64, m68k, mips/mipsel, powerpc, s390 를 지원하는 많은 종류의 공통 유닉스와 같은 플랫폼 (e.g., http://en.wikipedia.org/wiki/Unix-like)과 i386, amd64, powerpc-apple-darwin, mips-sgi-irix, i386-freebsd, rs6000-ibm-aix, sparc-sun-solaris 를 지원하는 sparc CPUs (e.g., http://buildd.debian.org/build.php?&pkg=r-base), i386-hurd-gnu, cpu-kfreebsd-gnu 에서 설정되고 구축될 수 있습니다. 만약, 사용자가 위에 명시된 것 외의 다른 플랫폼에 대해서 알고 계신다면 저희에게 알려주시길 부탁드립니다.

What is the current version of R?

현재 릴리즈된 버전은 2.14.2 입니다. 우리는 현재 'major.minor.patchlevel' 이라는 개발스킴을 따르고 있는데, R 은 두가지 개발버전을 가지고 있습니다. 하나는 현재 릴리즈의 패치된 버전 ('r-patched') 이고, 또 다른 하나는 추후에 minor 혹은 최종적으로는 major 버전이 될 릴리즈 ('r-devel') 버전입니다. 일반적으로 r-patched 의 경우 버그수정에 중점을 두고 있고, r-devel 은 주로 새로운 기능의 추가에 촛점을 맞추고 있습니다.

2.4 How can R be obtained?

R 에 대한 소스, 바이너리, 그리고 문서들은 "Comprehensive R Archive Network" 라고 불리는 CRAN 을 통하여 얻을 수 있습니다. (What is CRAN? 섹션을 살펴보세요).

소스는 또한 R 서브버전 저장소 (subversion repository)인 https://svn.r-project.org/R/ 를 통하여 얻을 수 있습니다. 그러나, rsync 와 cvs 는 지원하지 않습니다.

r-devel 과 r-patched 개발진행 버전의 R 은 타르볼의 형태로 ftp://ftp.stat.math.ethz.ch/Software/R 에서 확인할 수 있습니다.

How can R be installed?

How can R be installed (Unix-like)

만약 R 이 이미 설치되어 있다면, 쉘 프롬프트에서 R 이라고 입력한다면 실행할 수 있습니다.

만약 사용자의 플랫폼에 바이너리 (Are there Unix-like binaries for R? 이라는 섹션을 살펴보세요)가 지원된다면, 사용자는 이와 함께 제공되는 지시사항을 따른다면 사용할 수 있습니다.

그렇지 않다면, 사용자는 R 을 직접 컴파일하여 설치하여야 합니다. 그러나 이 과정은 일반적인 Unix-like 와 같은 플랫폼에서는 매우 쉽습니다. R 의 배포판에 함께 제공되는 INSTALL 이라는 파일은 간단한 설치 지침을 포함하고 있으면 "R Installation and Administration" 이라는 문서에는 설치와 관련한 자세한 세부사항들이 기록되어 있습니다. (What documentation exists for R? 이라는 섹션을 참고하세요).

참고로, 사용자는 R 을 설치하기 위해서는 C 컴파일러 외에 FORTRAN 컴파일러 혹은 아마도 f2c 라는 것이 필요할 것입니다.

가장 단순한 방법은, R 소스코드의 압축을 풀고 난뒤, 압축이 풀린 디렉토리로 이동한 후, 쉘프롬프트에서 아래와 같은 명령어들을 입력합니다.

```
$ ./configure
$ make
```

만약 이러한 명령어들이 성공적으로 수행된다면, R 바이너리와 R 이라고 불리는 R 쉘스크립트 프론트-엔드가 생성될 것이고, 이들은 bin 디렉토리로 복사되어 집니다. 사용자는 스크립트를 단지 사용자가 실행시키고자 하는 장소로 스크립트를 복사해 넣으면 됩니다. 예를들면, /usr/local/bin 디렉토리입니다. 추가적으로 HTML, LaTeX, 혹은 텍스트 버전의 문서들도 함께 설치됩니다.

refman.dvi (R 객체 레퍼런스 인덱스문서) 혹은 R-exts.dvi (doc/manual 디렉토리에 있는 "R Extension Writers Guide")와 같은 R 매뉴얼들의 DVI 버전을 생성하기 위해서는 make dvi 를 사용하시길 바랍니다. 이러한 파일들은 xdvi 혹은 dvips 와 같은 기본 프로그램을 이용하여 미리보거나 프린트를 할 수 있습니다. 또한, 매뉴얼들의 PDF (Portable Document Format) 과 같은 문서를 생성하기 위해서는 make pdf 를 사용하세요. 이들은 Arobat 과 같은 프로그램을 이용하여 볼 수 있습니다. GNU Texinfo 시스템으로 작성된 매뉴얼들은 Emacs 혹은 독립적인 GNU Info 에서 온라인으로 읽을 수 있도록 하는 info 파일들로 변환이 가능합니다. 이러한 버전들을 생성하기 위해서는 make info 파일을 이용하시길 바랍니다. (단, 이경우에는 Makeinfo 의 버전이 4.5 이상이 요구됩니다).

마지막으로 R 시스템이 올바르게 작동하는지 확인하기 위해서는 make check 를 사용하세요.

사용자는 또한 make install 을 이용하여 "system-wide" 설치를 수행할 수 있습니다. 기본적으로, 이 설치는 아래와 같은 디렉토리에 설치될 것입니다.

```
${prefix}/bin
    프론트엔드 쉘스크립트
${prefix}/man/man1
    도움말 페이지
${prefix}/lib/R
    위의 내용을 제외한 나머지 모두 (라이브러리, 온라인 도움말, 등등). 이것을 우리는 설치된 R의 홈디렉토리 (R_HOME) 이라고 부릅니다.
```

위에서 prefix 는 주로 설치환경을 확인하는 동안 결정되어지며 (주로 /usr/local 입니다), 이는 보통 아래와 같은 옵션을 통하여 변경할 수도 있습니다.

```
$ ./configure --prefix=/where/you/want/R/to/go
```

즉, R 은 /where/you/want/R/to/go/bin 이라는 곳에 설치되어지고 실행이 가능해질 것입니다.

매뉴얼들의 DVI, info, 그리고 PDF 버전들을 설치하기 위해서는 make install-dvi, make install-info, make install-pdf 를 실행하도록 하세요.

How can R be installed (Windows)

CRAN 사이트의 bin/windows 디렉토리는 ix86 과 x86_64 칩상에서 돌아가는 Windows 2000 과 그 이후의 버전들 (64비트 포함)에서 돌아갈 수 있는 R base 배포판과 CRAN으로 부터 얻을 수 있는 방대한 양의 애드온 패키지들에 대한 바이너리들을 포함하고 있습니다. 윈도우 버전의 R 은 Robert Gentleman 과 Guido Masarotto 에 의해서 만들어 졌으고, 현재 Duncan Murdoch 와 Brian D. Ripley 에 의해서 개발 및 유지보수가 되고 있습니다.

대부분의 경우, Windows 인스톨러 프로그램은 사용하기에 가장 쉬운 도구 일 것입니다. 보다 더 자세한 사항에 대해서는 "R for Windows FAQ" 를 참고하세요.

How can R be installed (Macintosh)

CRAN 사이트내의 bin/macosx 라는 디렉토리는 표준 Apple 인스톨러 패키지가 R.dmg 라는 디스크 이미지 안에 포함되어 있습니다. 이것을 다운로드 받아 실행시킨다면, 인스톨러는 현 비개발자 배포판을 설치하게 될 것입니다. RAqua 는 네이브(naive) Mac OS X Darwin 버전의 R 과 R.app Mac OS X GUI 을 지칭합니다. bin/macosx/powerpc/contrib/x.y 디렉토리 내에는, R의 "x.y" 릴리즈에 대응되는 RAqua 와 함께 사용될 (Mac OS X 의 powerpc 버전을 위하여) 미리 빌드된 바이너리 패키지들이 있을 것입니다. 이러한 패키지들의 설치는 R.app GUI 에 있는 "Package" 메뉴를 통하여 가능합니다. Mac OS X 에 대한 이러한 R 포트 (port) 는 Stefano Iacus 에 의하여 유지 보수 됩니다. 더 자세한 사항에 대해서는 "R for Mac OS X FAQ" 를 참고하길 바랍니다.

Are there Unix-like binaries for R?

CRAN 사이트의 bin/linux 디렉토리는 아래와 같은 패키지들을 포함하고 있습니다.

CPU	Versions	Provider	
Debian	i386/amd64	etch/lenny/squeeze	Johannes Ranke
Red Hat	i386/x86_64	fedora10/fedora11	Martyn Plummer
Ubuntu	i386/amd64	hardy/lucid/maverick/natty	Michael Rutter

Dirk Eddelbuettel 에 의해 유지보수되는 Debian 패지들은 오랜시간에 걸쳐 데비안 배포판의 일부분이 되어가고 있으며, 데비안 패키지 관리도구인 APT 를 통하여 쉽게 엑세스 될 수 있습니다. 즉, R 베이스 기본 환경 구축 및 추천패키지들을 설치하기 위해서는 apt-get install r-base r-recommended 이라고 하시면 됩니다. 만약, 소스로부터 R 패키지를 구축하기 원한다면, 이에 필요한 도구들을 설치하기 위해서 apt-get install r-base-dev 를 실행하면 됩니다. 안정적인 데비안 버전을 기반으로 하여 "backports" (백포트)라고 불리는 현재의 R 패키지들은 Johannes Ranke 에 의하여 제공되며, CRAN 에서 엑세스가 가능합니다. 이와 관련하여 R 데비안 패키지들과 백포트를 설치하는 방법에 대해서는 http://CRAN.r-project.org/bin/linux/debian/README 를 읽어보시길 바랍니다. 중요한 점은 데비안으로 파생된 다른 배포판에서도 안정적으로 작동하게끔 해야합니다. 우분투에 대한 네이티브 백포트 (native backports)는 Michael Rutter 에 의해서 제공되어졌습니다.

Tom "Spot" Callaway 에 의해서 유지보수되는 Fedora (페도라)에 이용되는 R 바이너리는 페도라 배포판의 일부로서 제공되며 RPM 설치 및 업데이트 관리자인 yum 에 의해서 엑세스될 수 있습니다. 페도라 R RPM 은 사용자와 개발자에게 필요한 모든 구성요소들을 다 함께 설치해주는 메타패키지("meta-package")인데, 이는 R-core 와 R-devel 으로 구분되어집니다. 또한, 독립적인 R math library (libRmath 와 libRmath-devel)를 설치할 수도 있으며, R 패키지들을 선택하여 설치하는 기능을 제공하고 있습니다. Enterprise Linux (EPEL) 프로젝트 (http://fedoraproject.org/wiki/EPEL) 에 대한 추가적인 패키지들은 RedHat Enterprise Linux 와 이에 호환되는 배포판들에 대한 포트를 제공해줍니다. 새로운 버전의 R이 릴리즈될때, 페도라 RPM 에서 새버전을 사용하기 위해서는 최대 2주까지 지연될 가능성이 있습니다. 이는 반드시 페도라내에 지정된 검증프로세스를 반드시 통과해야하기 때문입니다.

openSUSE 에서 이용되는 RPM에 관한 정보는 http://cran.r-project.org/bin/linux/suse/README.html 에서 살펴보시길 바랍니다.

현재로서는 CRAN 에서는 다른 바이너리 배포에 대해서 지원하지 않습니다.

What documentation exists for R?

R 에서 사용할 수 있는 대부분의 함수들과 변수명에 대해서는 온라인 문서에 대한 지원이 이루어지고 있으며, 이들 문서에 대한 열람은 R 프롬프트에서 help(검색어) 혹은 ?검색어 라는 기능을 이용함으로서 확인해 볼 수 있습니다. 여기에서 사용되는 검색어는 도움을 얻고자하는 주제어입니다. (단항과 이항 연산자, 그리고 분기문과 같은 특수한 형태의 경우에는 검색어를 입력할때 따옴표를 함께 넣어주어야 할 경우도 있습

니다).

본 문서 역시 온라인으로 HTML 과 PDF 을 통하여 읽을 수 있으며 LaTeX 를 이용하여 출력물으로 만들 수도 있습니다. How can R be installed? 라는 섹션을 참고하세요. 가장 최신버전의 HTML 문서들은 항상 http://stat.ethz.ch/R-manual/ 에서 확인해 보실수 있습니다.

R reference manual 의 몇몇 버전들은 Network Theory Ltd (http://www.network-theory.co.uk/R/base/) 에서 출판물로 간행되었습니다. 한 권의 매뉴얼이 판매될 때마다, 출판사는 R Foundation 에 USD 10 (미국돈으로 10달러)를 기부합니다 (What is the R Foundation? 이라는 섹션을 참고하세요).

R 은 아래와 같은 매뉴얼과 함께 배포됩니다.

- "An Introduction to R" (R-intro) 문서는 데이터형, 프로그래밍에 필요한 기초요소들, 통계 모델링과 그래픽에 대한 기본적인 정보를 포함하고 있습니다. 이 문서는 본래 Bill Venables 과 David Smith 가 작성한 "Notes on S-Plus" 라는 문서를 기초로 하여 작성되었습니다.
- "Writing R Extensions" (R-exts) 문서는 R 애드온 패키지들을 제작하는 과정, R 문서를 작성하는 요령, R 시스템과 다른 프로그래밍 언어들과 인터페이싱, 그리고 R API 에 대해서 기술하고 있습니다.
- "R Data Import/Export" (R-data) 는 데이터 입출력에 대한 가이드입니다.
- "The R Language Definition" (R-lang) 은 "Kernighan & Ritchie of R" 이라는 문서의 가장 처음버전이며, 이 문서는 이밸류에이션 (evaluation), 파싱 (parsing, 구문해석), 객체지향 프로그래밍 (object oriented programming), 컴퓨팅 (computing) 과 같은 내용들을 다루고 있습니다.
- "R Installation and Administration" (R-admin)
- "R Internals" (R-ints) 는 R의 내부구조에 대한 안내서이며 R-2.4.0 버전부터 추가되었습니다.

R 과 관련된 출판물들에 대한 정보는 BibTex 형식으로 아래의 주소에서 제공됩니다.

```
http://www.R-project.org/doc/bib/R.bib
```

R Core Team 멤버들에 의해서 작성된 책들은 다음과 같습니다.

- John M. Chambers (2008), “Software for Data Analysis: Programming with R”. Springer, New York, ISBN 978-0-387-75935-7, http://stat.stanford.edu/~jmc4/Rbook/.
- Peter Dalgaard (2008), “Introductory Statistics with R”, 2nd edition. Springer, ISBN 978-0-387-79053-4, http://www.biostat.ku.dk/~pd/ISwR.html.
- Robert Gentleman (2008), “R Programming for Bioinformatics”. Chapman & Hall/CRC, Boca Raton, FL, ISBN 978-1-420-06367-7, http://www.bioconductor.org/pub/RBioinf/.
- Stefano M. Iacus (2008), “Simulation and Inference for Stochastic Differential Equations: With R Examples”. Springer, New York, ISBN 978-0-387-75838-1.
- Deepayan Sarkar (2007), “Lattice: Multivariate Data Visualization with R”. Springer, New York, ISBN 978-0-387-75968-5.

- W. John Braun and Duncan J. Murdoch (2007), "A First Course in Statistical Programming with R". Cambridge University Press, Cambridge, ISBN 978-0521872652.
- P. Murrell (2005), "R Graphics", Chapman & Hall/CRC, ISBN: 1-584-88486-X, http://www.stat.auckland.ac.nz/~paul/RGraphics/rgraphics.html.
- William N. Venables and Brian D. Ripley (2002), "Modern Applied Statistics with S" (4th edition). Springer, ISBN 0-387-95457-0, http://www.stats.ox.ac.uk/pub/MASS4/.
- Jose C. Pinheiro and Douglas M. Bates (2000), "Mixed-Effects Models in S and S-Plus". Springer, ISBN 0-387-98957-0.

마지막으로, 최근것은 아니지만, Ross 와 Robert 가 R 을 디자인하고 구현하면서 겪은 경험들에 대해서는 다음의 문서에 기록되어 있습니다.

- Ihaka & Gentleman (1996), "R: A Language for Data Analysis and Graphics", Journal of Computational and Graphical Statistics, 5, 299 – 314.

Citing R

R 을 출판물에 인용하고자 한다면 다음의 BibTeX 정보를 이용하세요.

```
@Manual{,
  title        = {R: A Language and Environment for Statistical
                  Computing},
  author       = {{R Development Core Team}},
  organization = {R Foundation for Statistical Computing},
  address      = {Vienna, Austria},
  year         = 2011,
  note         = {{ISBN} 3-900051-07-0},
  url          = {http://www.R-project.org}
}
```

R 과 R 패키지들에 대한 BibTeX 엔트리 혹은 인용정보는 citation() 을 통하여 얻을 수 있습니다.

What mailing lists exist for R?

우리는 R 에 아래의 네가지 메일링 리스트를 제공해준 Martin Maechler 에 감사를 드립니다.

- R-announce: R 의 개발과 새로운 코드들에 대한 이용가능성에 대한 중요한 공지사항들에 대한 메일링 리스트입니다.
- R-packages: 새롭게 개발되거나, 주요기능들이 강화된 패키지들에 대한 공지사항들에 대한 소식을 접할 수 있습니다.
- R-help: R 을 사용함에 있어서 겪는 문제점들과 이들에 대한 해법에 대하여 논의하는 가장 중요한 메일링 리스트라고 할 수 있습니다. R-announce와 R-packages 에서는 공지되지 않았지만 R 의 개발 및 새코드의 이용가능성에 대한 내용을 접할 수 있습니다.
- R-devel: R 의 코드개발에 대한 질문과 토론내용을 담고 있는 리스트입니다.

위에 있는 어느 메일링 리스트에 내용을 보내기전에 먼저 포스팅 안내를 읽어보시길 바랍니다.

R-help 는 본래 R 을 이용하여 본인의 문제를 해결하고자 하지만, 프로그래밍에 대한 지식이 부족한 분들

을 위해서 개설되었습니다. 그래서, 프로그래머가 아닌 사람들이 이해하기 어려운 질문들 (즉, C 혹은 C++ 과 같은 내용)은 R-devel 에 올리셔야 합니다.

이러한 메일링 리스트들에 대한 구독 및 보관된 자료들에 대한 열람은 http://stat.ethz.ch/mailman/listinfo/ 에서 손쉽게 하실 수 있으며, 이메일을 통한 구독 (혹은 비구독)을 할 수 도 있습니다. R-help 에 대한 메일링을 구독하기 위해서는 r-help-request@lists.r-project.org 로 이메일을 보낼때 이메일 제목이 아닌 이메일 본문에 단순히 'subscribe'(구독) 혹은 'unsubscribe' (구독안함) 이라고 보내주세요.

R-help 메일링 리스트에 있는 모든 사람들에게 메시지를 보내기 위해서는 r-help@lists.r-project.org 로 이메일을 보내세요. 다른 메일링 리스트에 대하여 구독 및 포스팅 요령은 r-help 와 동일합니다. 단순히, 'R-help' 대신에 'R-announce', 'R-packages', 그리고 'R-devel' 이라고만 변경해주시면됩니다. 여기에서 알아두어야 할 점은 R-announce 와 R-packages 메일링 리스트들은 R-help 를 구독하기 위한 일종의 게이트웨이이므로, 만약 R-help 를 구독하지 않는 상태라면 둘중에 하나는 구독해야 합니다.

사용자가 R 코어멤버들에게 이메일을 보내기 보다는 R-help 메일링을 통하여 커뮤니케이션을 하기를 권장합니다 (R 코어멤버들 역시 메일링을 받고 있습니다). 그 이유는 지속적으로 R을 향상시키는데 있어서 많은 시간을 줄여주기도 할 뿐더러, 질문자가 가장 빠른시간내에 답변을 받을 수 있는 길이기도 하기 때문입니다.

당연히, 버그를 정확하게 재구현할 수 있도록 코드를 올려주는 형식의 버그리포팅은 매우 도움이 됩니다. 또한, 이러한 버그를 보고할 때 사용자가 어떤 시스템과 어떤 버전의 R 을 사용하는지 꼭 알려주십시요. 더 자세한 사항에 대해서는 R Bugs 섹션을 참고 하세요.

R 메일링 리스트들에 대한 좀 더 많은 정보를 구독하기 위해서는 http://www.r-project.org/mail.html 을 확인하기 부탁바랍니다.

R 코어팀 멤버들은 코멘트와 보고사항에 대해서 r-core@lists.r-project.org 메일링 리스트에 접속할 수 있습니다.

R project 의 메일링 리스트들의 대부분은 또한 웹브라우저에서 NNTP 뉴스리더 혹은 RSS 피드를 통하여 읽을 수 있는 Gmane 을 통해서 볼 수 있습니다. 더 구체적인 사항은 http://dir.gmane.org/index.php?prefix=gmane.comp.lang.r. 에서 확인하시길 바라고, RSS 피드에 대한 내용은 http://www.gmane.org/rss.php 에서 확인해주세요.

What is CRAN?

"Comprehensive R Archive Network" (CRAN) 은 R 배포판(들), 기여된 확장요소들 (즉, 패키지들), R 에 관련된 문서들, 그리고 바이너리들을 동일하게 제공할 수 있는 사이트들의 모음을 의미합니다.

Austria 의 Wirtschaftsuniversität Wien 에 있는 CRAN 마스터 사이트는 아래의 주소로부터 찾을 수 있습니다.

```
http://CRAN.R-project.org/
```

Daily 미러들은 아래와 같습니다.

```
http://cran.at.R-project.org/       (WU Wien, Austria)
http://cran.au.R-project.org/       (PlanetMirror, Australia)
http://cran.br.R-project.org/       (Universidade Federal do Paraná, Brazil)
http://cran.ch.R-project.org/       (ETH Zürich, Switzerland)
```

```
http://cran.dk.R-project.org/       (SunSITE, Denmark)
http://cran.es.R-project.org/       (Spanish National Research Network, Madrid, Spain)
http://cran.fr.R-project.org/       (INRA, Toulouse, France)
http://cran.pt.R-project.org/       (Universidade do Porto, Portugal)
http://cran.uk.R-project.org/       (U of Bristol, United Kingdom)
http://cran.za.R-project.org/       (Rhodes U, South Africa)
```

미러들에 대한 목록에 대해서는 http://cran.r-project.org/mirrors.html 페이지를 참고 해주세요. 네트워크의 부담을 줄이기 위해서 사용자가 현재 접속하고 있는 위치에서 가장 가까운 CRAN 사이트를 이용해주세요.

CRAN 으로부터 사용자는 tar.gz 와 tar.bz 의 형식으로 압축된 가장 최근의 공식 릴리즈 R과 데일리 스냅샷 (즉, 현재 소스트리의 복사본), 추가적으로 기여된 코드들 뿐만 아니라 다양한 운영체제 (Linux, Mac OS Classic, Mac OS X, 그리고 MS Windows) 를 위해 미리 빌드된 바이너리들을 얻을 수 있습니다. CRAN은 또한 R 다큐멘테이션, 메일링 리스트, 그리고 버그트랙킹 시스템에 대한 엑세스를 제공하고 있습니다.

CRAN 에 "서브밋"을 하기 위해서는 사용자의 서브미션이 CRAN 저장소의 정책 (Repository Policy) 에 부합되는지 먼저 확인을 하고, ftp://cran.r-project.org/incoming/ 에 올려주시고 cran@r-project.org 로 이메일을 보내주세요. CRAN은 일반적으로 보안적인 문제로 인하여 미리 컴파일된 바이너리를 수용하지 않습니다. 특히, Windows 와 Mac OS X 에 대한 바이너리 패키지들은 바이너리 패키지 유지자들에 의해서 각각 제공되어집니다.

특히 주의할 점은 반드시 저작권 (라이센스) 정보 (GPL-2, GPL-3, BSD, Artistic, ...) 과 함께 제공하는 것입니다. 그리고, CRAN 을 지칭하기 위해서는 꼭 마스터 사이트의 URL 을 사용하도록 하세요.

Can I use R for commercial purposes?

R 은 GNU 일반공중 사용허가서 (General Public License, GPL) 버전 2 를 기본으로 릴리즈됩니다. 만약, 사용자가 어떤 특정한 상황에서 R 을 사용하는 것이 합법적인가에 대한 의구심이 생길경우, 사용자는 반드시 사용자 자신의 전문 법률상담가와 상의해야합니다. 실제로 우리는 법적 조언을 줄 수 있는 그 어떠한 위치에 있지 않습니다.

R Core Team 은 R 을 상업적 목적 (즉, 비지니스 혹은 컨설팅)으로 사용할 수 있다고 생각합니다. 모든 종류의 오픈소스의 라이센스들과 같이 GPL 은 모든 패키지들에 대하여 어떠한 용도로 사용되어도 무방합니다. 오로지, R 자체 혹은 R 의 코드 일부분을 포함하는 프로그램을 배포하는 것만이 제한됩니다. 이것은 오픈소스 데피니션 (the Open Source Definition)의 제 6항 (clause 6) "No Discrimination Against Fields of Endeavor" 에서 찾아볼 수 있습니다.

라이센스는 특정분야에서 프로그램을 사용하고자 하는 그 어떠한 사람들을 제약해서는 안됩니다. 예를들면, 비지니스 혹은 유전자 연구를 목적으로 하는 프로그램에서의 사용에 제한을 두지 않습니다.

이것 또한 GPL 의 제 0항 (clause 0) 에 아래와 같이 분명하게 명시되어 있습니다.

```
복사, 배포, 및 수정을 제외한 그 외의 활동들에 대해서는 본 라이센스 정책에 영향을 받지 않습니다.
프로그램을 실행하는 행위는 제약받지 아니하고, 프로그램으로부터 생성된 결과물에 대해서도
만약 프로그램에 기초로한 결과물들로 구성된 경우에만 그 제약을 받습니다.
```

모든 추천패키지들을 포함하여 대부분의 애드온 패키지들 또한 이러한 방법으로 상업적인 목적으로 이용될 수 있음을 명시하고 있습니다. 단지, 특정 패키지들만이 "비상업용적 용도"로서 그 사용을 제한하고 있습니다. 이러한 경우에는 사용자는 반드시 저작권자에게 상업적용도로의 활용여부를 문의하거나, 법률상담

가로부터의 조언을 찾아야 합니다.

이 섹션에 언급된 모든 내용은 법적조언이 아니며, R Core Team 은 어떠한 경우에도 법적조언을 하지 않을 것입니다.

Why is R named R?

R 이라는 명칭은 최초로 R 을 개발한 두 개발자인 Robert Gentleman 과 Ross Ihaka 의 이름에서 기인하기도 하며, 동시에 Bell 연구소의 'S' 라는 언어에서 영감을 받기도 하였습니다 (What is S? 라는 섹션을 참고하세요).

What is the R Foundation?

The R Foundation 은 공익을 위한 비영리단체입니다. R Core Team 의 멤버들에 의해 설립된 이 단체의 주목적은 R project 와 statistical computing 에 기여할 기술들을 지원하는 것입니다. 이것은 R 개발커뮤니티를 지원하거나 협력관계를 맺고자 하는 개인, 기관 및 영리추구의 사업체들에게 그 기준을 제공하며, R 소프트웨어와 문서에 대한 저작권을 관리하기도 합니다. 더 자세한 사항은 http://www.R-project.org/foundation/ 을 참고하기 바랍니다.

What is R-Forge?

R-Forge (http://R-Forge.R-project.org/) 는 R 패키지 및 연관소프트웨어 개발, 그리고 다양한 프로젝트들을 자유롭게 발전할 수 있도록 해주는 중요한 개발환경을 제공하는 공간입니다. SVN 에 최적화된 손쉬운 접근, 나날이 구축되고 체크되어지는 패키지들, 메일링 리스트, 버그트래킹, 그리고 메시지보드와 포럼공간, 사이트 호스팅, 영구파일저장공간, 안정적인 백업, 그리고 웹기반의 관리 등의 기능을 제공하는 GForge 를 기초로 운영됩니다. 더 자세한 사항은 R-Forge 웹사이트와 Stefan Theusl 과 Achim Zeileis (2009) "Collaborative software development using R-Forge", The R Journal, 1(1):9-14 를 참고해주세요.

R and S

What is S?

S is a very high level language and an environment for data analysis and graphics. In 1998, the Association for Computing Machinery (ACM) presented its Software System Award to John M. Chambers, the principal designer of S, for the S system, which has forever altered the way people analyze, visualize, and manipulate data ...

S 는 데이터 분석과 그래픽스를 위한 하이레벨 언어 (high level language) 이며 환경입니다. Association for Computing Machinery (ACM) 은 S 언어(S 시스템은 오랜 시간동안 사람들이 데이터를 분석하고, 시각화하며, 다양화 하는 방법을 변화시켜오고 있다) 의 설계자인 John M. Chambers 에게 아래와 같이 치하하고 상을 수여하였습니다.

S is an elegant, widely accepted, and enduring software system, with conceptual integrity, thanks to the insight, taste, and effort of John Chambers. S는 개념적으로 완전한 상태로써 멋있고, 넓은 영역에 적합하며, 내구성 높은 소프트웨어 시스템입니다. John Chambers의 노력, 기호, 통찰력에 감사드립니다.

The evolution of the S language is characterized by four books by John Chambers and coauthors, which are also the primary references for S. S을 중요하게 추천하고 있는 Chambers 와 coauthors에 의해 발간된 네 가지의 책들은 S 언어의 혁명을 특징적으로 묘사하고 있습니다.

Richard A. Becker and John M. Chambers (1984), "S. An Interactive Environment for Data Analysis and Graphics," Monterey: Wadsworth and Brooks/Cole. This is also referred to as the "Brown Book", and of historical interest only.

Richard A. Becker and John M. Chambers (1984), "S. An Interactive Environment for Data Analysis and Graphics," Monterey: Wadsworth and Brooks/Cole. 이 책은 또한 "Brown Book"와 역사에대한 궁금증을 해결 하기 위할 때만 참조되어 집니다.

Richard A. Becker, John M. Chambers and Allan R. Wilks (1988), "The New S Language," London: Chapman & Hall. This book is often called the "Blue Book", and introduced what is now known as S version 2.

Richard A. Becker, John M. Chambers and Allan R. Wilks (1988), "The New S Language," London: Chapman & Hall. 이 책은 종종 "Blue Book"로 불려지면 어떤것이 S version 2로써 알려졌는지는 소개 합니다.

John M. Chambers and Trevor J. Hastie (1992), "Statistical Models in S," London: Chapman & Hall. This is also called the "White Book", and introduced S version 3, which added structures to facilitate statistical modeling in S.

John M. Chambers and Trevor J. Hastie (1992), "Statistical Models in S," London: Chapman & Hall. 이 책은 또한 "White Book" 불려지면, S 안에서 통계적인 모델링을 가능하게 하는 structures 을 포함한, S version 3를 소개합니다.

John M. Chambers (1998), "Programming with Data," New York: Springer, ISBN 0-387-98503-4 (http://cm.bell-labs.com/cm/ms/departments/sia/Sbook/). This "Green Book" describes version 4 of S, a major revision of S designed by John Chambers to improve its usefulness at every stage of the programming process.

John M. Chambers (1998), "Programming with Data," New York: Springer, ISBN 0-387-98503-4 (http://cm.bell-labs.com/cm/ms/departments/sia/Sbook/). 이 "Green Book"은 S의 중요 수정판 (John Chambers이 모든 프로그래밍 과정에서 효율성을 상승 시킨 버전), S version4 를 설명합니다. See http://cm.bell-labs.com/cm/ms/departments/sia/S/history.html for further information on "Stages in the Evolution of S". 보다 많은 "Stages in the Evolution of S"의 정보는 http://cm.bell-labs.com/cm/ms/departments/sia/S/history.html 에서 볼 수 있습니다.

There is a huge amount of user-contributed code for S, available at the S Repository at CMU. CMU에 있는 S Repository 에서 방대한 user-contributed code for S 이용 할 수 있습니다.

What is S-Plus?

S-Plus is a value-added version of S sold by Insightful Corporation, which in 2008 was acquired by TIBCO Software Inc. See the Insightful S-Plus page and the TIBCO Spotfire S+ Products page for further information. S-Plus 는 통찰력 있는 회사, TIBCO Software Inc.(2008)에 의해 판매되어지는 추가

적인 가치가 더해진 버전입니다. 보다 많은 정보를 위해 the Insightful S-Plus page 와 the TIBCO Spotfire S+ Products page 을 보십시오.

What are the differences between R and S?

We can regard S as a language with three current implementations or "engines", the "old S engine" (S version 3; S-Plus 3.x and 4.x), the "new S engine" (S version 4; S-Plus 5.x and above), and R. Given this understanding, asking for "the differences between R and S" really amounts to asking for the specifics of the R implementation of the S language, i.e., the difference between the R and S engines.

우리는 S 를 세가지 결과물들 혹은 엔진들; old S engine" (S version 3; S-Plus 3.x and 4.x), the "new S engine" (S version 4; S-Plus 5.x and above), and R 들과 함께 언어로써 취급합니다. R 과 S의 차이점에 대해 질문하는 것에 대한 이해는 S 언어의 R 결과물의 특정함에 대해 질문하는 것과 결과적으로 비슷합니다.

For the remainder of this section, "S" refers to the S engines and not the S language. 이 세션의 상기를 위해, S는 S엔진들을 참고하지만, S 언어는 참고하지 않습니다.

Lexical scoping Models Others

Lexical scoping =

Contrary to other implementations of the S language, R has adopted an evaluation model in which nested function definitions are lexically scoped. This is analogous to the evaluation model in Scheme.

다른 S언어의 실행과 대조적으로, R은 평가 모델 적응해오고 있고, 평가모델 안의 nested function definitions들은 사전적으로 살펴지고 있습니다. 이 것은 계획 안 에서의 평가모델과 유사합니다.

This difference becomes manifest when free variables occur in a function. Free variables are those which are neither formal parameters (occurring in the argument list of the function) nor local variables (created by assigning to them in the body of the function). In S, the values of free variables are determined by a set of global variables (similar to C, there is only local and global scope). In R, they are determined by the environment in which the function was created.

자유로운 변수들이 하나의 function안에서 발생할 때, 이 차이점을 다양해 집니다. 자유로운 변수는 가매개변수(occurring in the argument list of the function) 혹은 국부적 변수(created by assigning to them in the body of the function)가 아닙니다. S안에서, 자유변수의 가치는 전역 변수의 set(C와 비슷하고, 오직 지역, 전역 범위만 있습니다) 에 의해서 결정되어 집니다. R에서 환경안에서 어떠한 function이 생성되었는지에 의해 자유변수들은 결정되어집니다.

Consider the following function: 아래에 나와있는 function 들을 생각해 보십시오.

```
cube <- function(n) {
  sq <- function() n * n
```

```
  n * sq()
}
```

Under S, sq() does not "know" about the variable n unless it is defined globally: S통제 하에, sq()가 넓게 정의되지만 않는다면, 그것은 변수 n에대해 알지 못합니다.

```
S> cube(2)
Error in sq():  Object "n" not found
Dumped
S> n <- 3
S> cube(2)
[1] 18
```

In R, the "environment" created when cube() was invoked is also looked in:

```
R> cube(2)
[1] 8
```

As a more "interesting" real-world problem, suppose you want to write a function which returns the density function of the r-th order statistic from a sample of size n from a (continuous) distribution. For simplicity, we shall use both the cdf and pdf of the distribution as explicit arguments. (Example compiled from various postings by Luke Tierney.)

현실세계의 문제인 호기심 때문에, 당신이 r번째 명령통계의 density function을 되돌리는function을 적기 원하는 것을 추정합니다. r 번째 명령통계는 distribution 으로부터의 하나의 사이즈 'n'의 샘플로 부터 옵니다. 보다 간편하게 하기 위해, 우리는 distribution의 두가지 형식, cdf 와 pdf, 을 분명한 arguments로써 사용할 것입니다. (Luke Tierney에 의한 다양한 포스팅들로부터 다음의 예를 컴파일 했었습니다.)

The S-Plus documentation for call() basically suggests the following:

```
dorder <- function(n, r, pfun, dfun) {
  f <- function(x) NULL
  con <- round(exp(lgamma(n + 1) - lgamma(r) - lgamma(n - r + 1)))
  PF <- call(substitute(pfun), as.name("x"))
  DF <- call(substitute(dfun), as.name("x"))
  flength(f) <-
    call("*", con,
         call("*", call("^", PF, r - 1),
              call("*", call("^", call("-", 1, PF), n - r),
                   DF)))
  f
}
```

Rather tricky, isn't it? The code uses the fact that in S, functions are just lists of special mode with the function body as the last argument, and hence does not work in R (one could make the idea work, though).

그 코드는 S 안 에서 functions들을 function body와 함께 특별한 모드의 목록이라는 사실을 최근의 argument로써 사용하지만, 결론적으로 R에서는 function들을 수행하지 않습니다.(한 가지는 function을 수행할수도 있습니다.)

A version which makes heavy use of substitute() and seems to work under both S and R is substitute의 과도한 사용을 만들고, S와 R 모두에서 수행되는 하나의 버전입니다.

```
dorder <- function(n, r, pfun, dfun) {
  con <- round(exp(lgamma(n + 1) - lgamma(r) - lgamma(n - r + 1)))
  eval(substitute(function(x) K * PF(x)^a * (1 - PF(x))^b * DF(x),
                  list(PF = substitute(pfun), DF = substitute(dfun),
                       a = r - 1, b = n - r, K = con)))
}
```

(the eval() is not needed in S).

However, in R there is a much easier solution: 하지만, R에서, 훨씬 쉬운 해결책이 존재합니다.

```
dorder <- function(n, r, pfun, dfun) {
  con <- round(exp(lgamma(n + 1) - lgamma(r) - lgamma(n - r + 1)))
  function(x) {
    con * pfun(x)^(r - 1) * (1 - pfun(x))^(n - r) * dfun(x)
  }
}
```

This seems to be the "natural" implementation, and it works because the free variables in the returned function can be looked up in the defining environment (this is lexical scope). 이것은 자연스럽게 실행되는것 같고, returned function 안의 자유변수들은 정의되고있는 환경(사전적 범위)안에서 선호되어 지기 때문에, 그 것은 수행합니다.

Note that what you really need is the function closure, i.e., the body along with all variable bindings needed for evaluating it. Since in the above version, the free variables in the value function are not modified, you can actually use it in S as well if you abstract out the closure operation into a function MC() (for "make closure"):

정말로 사용자가 필요한 것을 function 폐쇄라는 것을 주목하세요. 예를 들면, body는 모든 다양한 bindings와 함께 그 것을 평가하는 것을 위해 필요했었습니다.

```
dorder <- function(n, r, pfun, dfun) {
  con <- round(exp(lgamma(n + 1) - lgamma(r) - lgamma(n - r + 1)))
  MC(function(x) {
       con * pfun(x)^(r - 1) * (1 - pfun(x))^(n - r) * dfun(x)
     },
     list(con = con, pfun = pfun, dfun = dfun, r = r, n = n))
}
```

Given the appropriate definitions of the closure operator, this works in both R and S, and is much "cleaner" than a substitute/eval solution (or one which overrules the default scoping rules by using explicit access to evaluation frames, as is of course possible in both R and S).

For R, MC() simply is

```
MC <- function(f, env) f
```

(lexical scope!), a version for S is

```
MC <- function(f, env = NULL) {
  env <- as.list(env)
  if (mode(f) != "function")
    stop(paste("not a function:", f))
  if (length(env) > 0 && any(names(env) == ""))
    stop(paste("not all arguments are named:", env))
```

```
    fargs <- if(length(f) > 1) f[1:(length(f) - 1)] else NULL
    fargs <- c(fargs, env)
    if (any(duplicated(names(fargs))))
      stop(paste("duplicated arguments:", paste(names(fargs)),
           collapse = ", "))
    fbody <- f[length(f)]
    cf <- c(fargs, fbody)
    mode(cf) <- "function"
    return(cf)
  }
```

Similarly, most optimization (or zero-finding) routines need some arguments to be optimized over and have other parameters that depend on the data but are fixed with respect to optimization. With R scoping rules, this is a trivial problem; simply make up the function with the required definitions in the same environment and scoping takes care of it. With S, one solution is to add an extra parameter to the function and to the optimizer to pass in these extras, which however can only work if the optimizer supports this.

비슷하게, 대부분의

Nested lexically scoped functions allow using function closures and maintaining local state. A simple example (taken from Abelson and Sussman) is obtained by typing demo("scoping") at the R prompt. Further information is provided in the standard R reference "R: A Language for Data Analysis and Graphics" (see What documentation exists for R?) and in Robert Gentleman and Ross Ihaka (2000), "Lexical Scope and Statistical Computing", Journal of Computational and Graphical Statistics, 9, 491 – 508.

Nested lexically scoped functions also imply a further major difference. Whereas S stores all objects as separate files in a directory somewhere (usually .Data under the current directory), R does not. All objects in R are stored internally. When R is started up it grabs a piece of memory and uses it to store the objects. R performs its own memory management of this piece of memory, growing and shrinking its size as needed. Having everything in memory is necessary because it is not really possible to externally maintain all relevant "environments" of symbol/value pairs. This difference also seems to make R faster than S.

The down side is that if R crashes you will lose all the work for the current session. Saving and restoring the memory "images" (the functions and data stored in R's internal memory at any time) can be a bit slow, especially if they are big. In S this does not happen, because everything is saved in disk files and if you crash nothing is likely to happen to them. (In fact, one might conjecture that the S developers felt that the price of changing their approach to persistent storage just to accommodate lexical scope was far too expensive.) Hence, when doing important work, you might consider saving often (see How can I save my workspace?) to safeguard against possible crashes. Other possibilities are logging your sessions, or have your R commands stored in text files which can be read in using source().

Note: If you run R from within Emacs (see R and Emacs), you can save the contents of the interaction buffer to a file and conveniently manipulate it using ess-transcript-mode, as well as save source copies of all functions and data used.

Next: Others, Previous: Lexical scoping, Up: What are the differences between R and S? 3.3.2 Models
There are some differences in the modeling code, such as

Whereas in S, you would use lm(y ~ x^3) to regress y on x^3, in R, you have to insulate powers of numeric vectors (using I()), i.e., you have to use lm(y ~ I(x^3)). The glm family objects are implemented differently in R and S. The same functionality is available but the components have different names. Option na.action is set to "na.omit" by default in R, but not set in S. Terms objects are stored differently. In S a terms object is an expression with attributes, in R it is a formula with attributes. The attributes have the same names but are mostly stored differently. Finally, in R y ~ x + 0 is an alternative to y ~ x - 1 for specifying a model with no intercept. Models with no parameters at all can be specified by y ~ 0.

Previous: Models, Up: What are the differences between R and S? 3.3.3 Others Apart from lexical scoping and its implications, R follows the S language definition in the Blue and White Books as much as possible, and hence really is an “implementation” of S. There are some intentional differences where the behavior of S is considered “not clean”. In general, the rationale is that R should help you detect programming errors, while at the same time being as compatible as possible with S.

Some known differences are the following.

In R, if x is a list, then x[i] <- NULL and xi <- NULL remove the specified elements from x. The first of these is incompatible with S, where it is a no-op. (Note that you can set elements to NULL using x[i] <- list(NULL).) In S, the functions named .First and .Last in the .Data directory can be used for customizing, as they are executed at the very beginning and end of a session, respectively. In R, the startup mechanism is as follows. Unless --no-environ was given on the command line, R searches for site and user files to process for setting environment variables. Then, R searches for a site-wide startup profile unless the command line option --no-site-file was given. This code is loaded in package base. Then, unless --no-init-file was given, R searches for a user profile file, and sources it into the user workspace. It then loads a saved image of the user workspace from .RData in case there is one (unless --no-restore-data or --no-restore were specified). Next, a function .First() is run if found on the search path. Finally, function .First.sys in the base package is run. When terminating an R session, by default a function .Last is run if found on the search path, followed by .Last.sys. If needed, the functions .First() and .Last() should be defined in the appropriate startup profiles. See the help pages for .First and .Last for more details.

In R, T and F are just variables being set to TRUE and FALSE, respectively, but are not reserved words as in S and hence can be overwritten by the user. (This helps e.g. when you have factors with levels "T" or "F".) Hence, when writing code you should always use TRUE and FALSE. In R, dyn.load() can only load shared objects, as created for example by R CMD SHLIB. In R, attach() currently only works for lists and data frames, but not for directories. (In fact, attach() also works for R data files created with save(), which is analogous to attaching directories in S.) Also, you cannot attach at position 1. Categories do not exist in R, and never will as they are deprecated now in S. Use factors instead. In R, For() loops are not necessary and hence not supported. In R, assign() uses the argument envir= rather than where= as in S. The random number generators are different, and the seeds have different length. R passes integer objects to C as int * rather than long * as in S. R has no single precision storage mode. However, as of version 0.65.1, there is a single precision interface to C/FORTRAN subroutines. By default, ls() returns the names of the objects in the current (under R) and global (under S) environment, respectively. For example, given

```
x <- 1; fun <- function() {y <- 1; ls()}
```

then fun() returns "y" in R and "x" (together with the rest of the global environment) in S.

R allows for zero-extent matrices (and arrays, i.e., some elements of the dim attribute vector can be 0). This has been determined a useful feature as it helps reducing the need for special-case tests for empty subsets. For example, if x is a matrix, x[, FALSE] is not NULL but a "matrix" with 0 columns. Hence, such objects need to be tested for by checking whether their length() is zero (which works in both R and S), and not using is.null(). Named vectors are considered vectors in R but not in S (e.g., is.vector(c(a = 1:3)) returns FALSE in S and TRUE in R). Data frames are not considered as matrices in R (i.e., if DF is a data frame, then is.matrix(DF) returns FALSE in R and TRUE in S). R by default uses treatment contrasts in the unordered case, whereas S uses the Helmert ones. This is a deliberate difference reflecting the opinion that treatment contrasts are more natural. In R, the argument of a replacement function which corresponds to the right hand side must be named 'value'. E.g., f(a) <- b is evaluated as a <- "f<-"(a, value = b). S always takes the last argument, irrespective of its name. In S, substitute() searches for names for substitution in the given expression in three places: the actual and the default arguments of the matching call, and the local frame (in that order). R looks in the local frame only, with the special rule to use a "promise" if a variable is not evaluated. Since the local frame is initialized with the actual arguments or the default expressions, this is usually equivalent to S, until assignment takes place. In S, the index variable in a for() loop is local to the inside of the loop. In R it is local to the environment where the for() statement is executed. In S, tapply(simplify=TRUE) returns a vector where R returns a one-dimensional array (which can have named dimnames). In S(-Plus) the C locale is used, whereas in R the current operating system locale is used for determining which characters are alphanumeric and how they are sorted. This affects the set of valid names for R objects (for example accented chars may be allowed in R) and ordering in sorts and comparisons (such as whether "aA" < "Bb" is true or false). From version 1.2.0 the locale can be (re-)set in R by the Sys.setlocale() function. In S, missing(arg) remains TRUE if arg is subsequently modified; in R it doesn't. From R version 1.3.0, data.frame strips I() when creating (column) names. In R, the string "NA" is not treated as a missing value in a character variable. Use as.character(NA) to create a missing character value. R disallows repeated formal arguments in function calls. In S, dump(), dput() and deparse() are essentially different interfaces to the same code. In R from version 2.0.0, this is only true if the same control argument is used, but by default it is not. By default dump() tries to write code that will evaluate to reproduce the object, whereas dput() and deparse() default to options for producing deparsed code that is readable. In R, indexing a vector, matrix, array or data frame with [using a character vector index looks only for exact matches (whereas [[and $ allow partial matches). In S, [allows partial matches. S has a two-argument version of atan and no atan2. A call in S such as atan(x1, x2) is equivalent to R's atan2(x1, x2). However, beware of named arguments since S's atan(x = a, y = b) is equivalent to R's atan2(y = a, x = b) with the meanings of x and y interchanged. (R used to have undocumented support for a two-argument atan with positional arguments, but this has been withdrawn to avoid further confusion.) Numeric constants with no fractional and exponent (i.e., only integer) part are taken as integer in S-Plus 6.x or later, but as double in R. There are also differences which are not intentional, and result from missing or incorrect code in R. The developers would appreciate hearing about any deficiencies you may find (in a written report fully documenting the difference as you see it). Of course, it would be useful if you were to implement the change yourself and make sure it works.

Next: What is R-plus?, Previous: What are the differences between R and S?, Up: R and S 3.4 Is there anything R can do that S-Plus cannot? Since almost anything you can do in R has source code that you could port to S-Plus with little effort there will never be much you can do in R that you

couldn't do in S-Plus if you wanted to. (Note that using lexical scoping may simplify matters considerably, though.)

R offers several graphics features that S-Plus does not, such as finer handling of line types, more convenient color handling (via palettes), gamma correction for color, and, most importantly, mathematical annotation in plot texts, via input expressions reminiscent of TeX constructs. See the help page for plotmath, which features an impressive on-line example. More details can be found in Paul Murrell and Ross Ihaka (2000), "An Approach to Providing Mathematical Annotation in Plots", Journal of Computational and Graphical Statistics, 9, 582 - 599.

Previous: Is there anything R can do that S-PLUS cannot?, Up: R and S 3.5 What is R-plus? For a very long time, there was no such thing.

XLSolutions Corporation is currently beta testing a commercially supported version of R named R+ (read R plus).

REvolution Computing has released REvolution R, an enterprise-class statistical analysis system based on R, suitable for deployment in professional, commercial and regulated environments.

Random Technologies offers RStat, an enterprise-strength statistical computing environment which combines R with enterprise-level validation, documentation, software support, and consulting services, as well as related R-based products.

See also http://en.wikipedia.org/wiki/R_programming_language#Commercialized_versions_of_R for pointers to commercialized versions of R.

Next: R Add-On Packages, Previous: R and S, Up: Top

What is a bug?

버그란 무엇인가?

If R executes an illegal instruction, or dies with an operating system error message that indicates a problem in the program (as opposed to something like "disk full"), then it is certainly a bug. 만약 R이 비정상적인 명령을 실행하거나 혹은 프로그램 문제를 나타내는 실행 시스템 에러 메세지("disk full"과 다른)와 함께 종료된다면,그 것은 버그가 발생한 것 이다. If you call .C(), .Fortran(), .External() or .Call() (or .Internal()) yourself (or in a function you wrote), you can always crash R by using wrong argument types (modes). This is not a bug. 사용자가 .C(), .Fortran(), .External() or .Call() (or .Internal()) yourself (or in a function you wrote)등의 명령을 입력하였다면, 사용자의 잘못된 사용에 의한 충동이므로, 그것은 버그가 아니다. Taking forever to complete a command can be a bug, but you must make certain that it was really R's fault. Some commands simply take a long time. If the input was such that you know it should have been processed quickly, report a bug. If you don't know whether the command should take a long time, find out by looking in the manual or by asking for assistance. 명령어가 수행되는데 엄청나게 오랜 시간이 소요되는 것은 버그이지만, 사용자는 어떤문제가 R에 의해서 발생되었는지 명확히 알고 있어야만 하다. 몇몇 명령어들은 오랜시간을 소요한다. 만약, 사용자가 입력한

명령이 짧을 시간안에 실행되야만 하지만, 오랜시간을 소요할 시, 그것은 버그이므로 보고하여라. 사용자가 사용한 명령에 대해서 얼마나 오랜시간을 소요되어야 하는지 알지 모른다면, manual을 통해 해결하거나, 도움을 요청하라. If a command you are familiar with causes an R error message in a case where its usual definition ought to be reasonable, it is probably a bug. If a command does the wrong thing, that is a bug. But be sure you know for certain what it ought to have done. If you aren't familiar with the command, or don't know for certain how the command is supposed to work, then it might actually be working right. For example, people sometimes think there is a bug in R's mathematics because they don't understand how finite-precision arithmetic works. Rather than jumping to conclusions, show the problem to someone who knows for certain. Unexpected results of comparison of decimal numbers, for example 0.28 * 100 != 28 or 0.1 + 0.2 != 0.3, are not a bug. See Why doesn't R think these numbers are equal?, for more details. 사용자가 잘 알고있는 명령어가 목정에 타당한 경우 R 오류 메세지의 원인이 된다면, 이 것을 대체적으로 버그이다. 만약, 명령어가 잘못된 형식 수행한다면, 그 또한 버그이다. 하지만 사용자는 명령어에 대한 어떤 것들이 수행되어져야 하는지 정확히 알고 있어야 한다. 사용작 명령어에 대한 지식이 없거나, 어떠한 형식으로 작업을 수행하지는 모를 경우, 명령어는 정확히 작동하고 있을 지도 모른다. 예를 들어, 사람들은 R이 정확이 어떠한 과정을 거쳐서 명령을 수행하는지 모르기 때문에, 종종 R의 mathematics에버그가 발생한다고 생각한다. 바로 결론을 내리기 보다는, 그 문제를 정확히 하는 사람에게 보여주어라. 예를 들면 0.28 * 100 != 28 or 0.1 + 0.2 != 0.3 십진수의 비교의 예상되지 않은 결과들은 버그가 아니다.

Finally, a command's intended definition may not be best for statistical analysis. 결론적으로, 명령어의 의도된 정의는 어쩌면 통계적인 분석에 대한 최선이 아닐지도 모른다.

This is a very important sort of problem, but it is also a matter of judgment.

이것은 매우 중요한 문제이지기도 하지만, 판결의 요소이기도하다.

Also, it is easy to come to such a conclusion out of ignorance of some of the existing features.

그 것은 또한 존재하고있는 특성을 알지 못하는 쉽게 결론지을

It is probably best not to complain about such a problem until you have checked the documentation in the usual ways, feel confident that you understand it, and know for certain that what you want is not available. 사용자가 일반적이 방법으로 그 문서를 확인하거나, 확실이 그 것을 이해하고 있거나, 사용자가 원하지만 이용할수 없는 것을 알고 있을 때 까지, 문제를 제기하지 않는 것이 좋습니다. If you are not sure what the command is supposed to do after a careful reading of the manual this indicates a bug in the manual. 만약 그 manual을 신중하게 읽은후에도 사용자가 어떻게 그 명령어가 수행되어지는지 잘 모른다며, 이 것은 manual 자체의 문제입니다.

The manual's job is to make everything clear.

manual은 모든것을 정확히 알려주기위해 만들어 졌습니다.

It is just as important to report documentation bugs as program bugs.

버그들을 문서화 하여 보고하는 것은, 버그를 프로그램화해서 고쳐내는 것 만큼 중요합니다.

```
However, we know that the introductory documentation is seriously inadequate, so you don't need to report this.
```

하지만, 처음 문서화되는 것은 상당히 불충분함으로, 사용자는 이 것을 보고할 필요는 없습니다. If the online argument list of a function disagrees with the manual, one of them must be wrong, so report the bug. 만약 the online argument list of a function가 그 manual과 일치하지 않는 다면, 그것들중 하나는 잘 못된 것이니, 보고해야 합니다.

How to report a bug

When you decide that there is a bug, it is important to report it and to report it in a way which is useful. 사용자가 버그의 존재에 대한 확신이 생겼을 때, 유용한 방법으로 그 버그를 보고하는 것은 중요합니다. What is most useful is an exact description of what commands you type, starting with the shell command to run R, until the problem happens. 가장좋은 것은 사용자가 그 문제가 발생하기까지어떤 명령어들을(R을 실행하기 위한 shell command들과 함께 시작하는) 입력하였는지 에대한 정확한 설명입니다.

```
Always include the version of R, machine, and operating system that you are using; type version in R to print this.
```

그리고 항상 사용하고 있는 R, 머신, os의 버전을 포함시켜야 합니다.(이것을 프리트하기위해 버전을 R에 입력하세요) The most important principle in reporting a bug is to report facts, not hypotheses or categorizations. 버그 보고에 있어 가장 중요한 원리는 요소들은 보고하는 것이지, hypotheses 혹은 categorizations 이 아닙니다.

```
It is always easier to report the facts, but people seem to prefer to strain to posit explanations and report them instead.
```

항상 그 요소들을 보고하는 것이 쉽지만, 사람들은 보고하는 것 보다 설명들을 이해하기위해 안간 힘을 쓰고 있습니다.

```
If the explanations are based on guesses about how R is implemented, they will be useless; others will have to try to figure out what
```

만약 그 설명들이 R이 어떻게 실행되어지는지에 대한 가정들에 근거하고 있다면, 그것들은 유용한 정보가 아닙니다. 다른 설명들은 어떤 요소들이 가정들은 이끄는지 생각해야만 하고, 종종 이것은 불가능 합니다.

```
But in any case, it is unnecessary work for the ones trying to fix the problem.
```

하지만, 어떠한 경우에서든 한 명이 이 문제를 고치려고 하는 것은 필요로 하지 않습니다. For example, suppose that on a data set which you know to be quite large the command 예를 들어, 당신은 전적으로 큰 data set 추정하다

```
   R> data.frame(x, y, z, monday, tuesday)
```

never returns. 결코 돌아오지 않는다 Do not report that data.frame() fails for large data sets. large data sets 에 대한 data.frame()의 실패는 보고하지 말아라. Perhaps it fails when a variable name is a day of the week. 어떠면, 그 것은 다양한 이름이 한주의 하루 일 때 실패할지도 모르다.

```
If this is so then when others got your report they would try out the data.frame() command on a large data set, probably with no day o
```

만약, 이것이 사실이라면, 다른 사람들이 당신의 보고를 얻었을 때, 그들은 large data set 을 기반으로한 the data.frame 명령어를 다양한 요일이름 없이 시도하거나 아무런 문제도 없다고 본다.. There is no way in the world that others could guess that they should try a day of the week variable name. 다른사람들 요일의 이름을 시도해야만 한다는 가정의 시대속에서는 아무런 방법이 없습니다. Or perhaps the command fails because the last command you used was a method for "["() that had a bug causing R's internal data structures to be corrupted and making the data.frame() command fail from then on. This is why others need to know what other commands you have typed (or read from your startup file). 혹은, 그 명령은 실패합니다. 왜냐하면 사용자가 사용했던 최근의 명령어가 R의 내부적인 data 구성이 붕괴되고 data.frame() 명령어 만들기가 실패되는 데 원인이 되는 버그를 포함한 "["()에 대한 방법이기 때문입니다. 이 것은 왜 다른사람들이 어떤 명령어를 입력했었는지가 필요한지를 말해줍니다.

It is very useful to try and find simple examples that produce apparently the same bug, and somewhat useful to find simple examples that might be expected to produce the bug but actually do not. If you want to debug the problem and find exactly what caused it, that is wonderful. You should still report the facts as well as any explanations or solutions. Please include an example that reproduces (e.g., http://en.wikipedia.org/wiki/Reproducibility) the problem, preferably the simplest one you have found.

이 것은 같은 버그를 만들어내는 간단한 예들을 시도하고 찾아내는데 유용합니다. 그리고 버그를 실제로 만들지는 않지만, 그렇다고 예상되어지는 간단한 예를 찾아내는데 약간은 유용합니다. 만약 사용자가 문제들을 debug 하기는 원하고 어떤 것이 정확히 원인이 되는지를 찾기는 원한다면, 가장 바람직한 것이다. 사용자는 여전히 요소들과 어떠한 설명 혹은 해결법을 보고해야합니다. 보고할때 사용자들이 발견해오고 있는 문제들을 해설해주는 가장 간단한 예들을 포함시켜 주시기 바랍니다.

Invoking R with the --vanilla option may help in isolating a bug. This ensures that the site profile and saved data files are not read.

--vanilla option와 함께 R 에 대해 언급하였던것은 독립된 하나의 버그를 도울지도 모릅니다. 이 것은 그 홈페이지 프로필과 저장된 데이터 파일들이 읽혀지지 않는 것을 보장합니다.

Before you actually submit a bug report, you should check whether the bug has already been reported and/or fixed. First, try the “Show open bugs new-to-old” or the search facility on http://bugs.R-project.org/. Second, consult https://svn.R-project.org/R/trunk/doc/NEWS.Rd, which records changes that will appear in the next release of R, including bug fixes that do not appear on the Bug Tracker. (Windows users should additionally consult https://svn.R-project.org/R/trunk/src/gnuwin32/CHANGES.Rd.) Third, if possible try the current r-patched or r-devel version of R. If a bug has already been reported or fixed, please do not submit further bug reports on it. Finally, check carefully whether the bug is with R, or a contributed package. Bug reports on contributed packages should be sent first to the package maintainer, and only submitted to the R-bugs repository by package maintainers, mentioning the package in the subject line.

실제적으로 버그 리포트를 제출하기 이전에, 사용자는 그 버그가 이미 보고되었거나 혹은 정정되었는지 확인하여야 합니다. 첫번째로, “Show open bugs new-to-old”를 시도하거나, 혹은 "http://bugs.R-project.org/"에서 시설을 검색하세요. 다음으로, R의 다음버전의 기능을 보여주는 "https://svn.R-project.org/R/trunk/doc/NEWS.Rd" 와 상당하십시오. 다음버전의 R은 Bug Tracker에 나타나지 않았떤 버그 수정을 포함하고 있습니다 (윈도우 유저들은 추가적으러 "https://svn.R-project.org/R/trunk/src/gnuwin32/CHANGES.Rd" 와 상담해야합니다.) 세번째로, 가능하다면 현재 패치되었거나 발전된 R

버전을 시도해보세요. 만약, 버그나 이미 보고되었거나, 수정되었다면, 추가적인 보고는 필요하지 않습니다. 마지막으로, 그 버그가 R에 대한것인지, 팩키지에 대한 것인지 신중하게 확인해주시기 바랍니다. 공헌된 팩키지에 대한 Bug reports 는 먼저 팩키지 maintainer에게 보내져야 하고, 그 보고서는 팩키지 maintainers (같은 맥락에서 그 팩키지를 언급하고 있는) 에 의해 R-bugs repository 에게 보고되어야 합니다.

A bug report can be generated using the function bug.report(). For reports on R this will open the Web page at http://bugs.R-project.org/: for a contributed package it will open the package's bug tracker Web page or help you compose an email to the maintainer.

버그 보고서는 bug.report()의 사용하서 발생될 수 있습니다. 이 것은 R에대한 보고서들을 위해 홈페이지 "http://bugs.R-project.org/"에 열려있습니다. 그 것은 팩키지의 Bug Tracker Web page를 열거나 혹은 사용자들이 maintainers에게 이메일 보내는 것을 도와 줍니다.

There is a section of the bug repository for suggestions for enhancements for R labelled ‘wishlist’. Suggestions can be submitted in the same ways as bugs, but please ensure that the subject line makes clear that this is for the wishlist and not a bug report, for example by starting with ‘Wishlist:’.

'wishlist' 는 R의 발전을 위한 제안들의 큰 저장소 입니다. 사용자들은 버그와 같은 방법으로 제안들을 제출할 수 있습니다. 그리고, 제목란에 wishlist를 위한 것이고, 버그 보고서가 아닌것을 명시해 주어야 합니다. 예를 들어, 제목을 "Wishlist"란 글자와 함께 시작하십시오.

Comments on and suggestions for the Windows port of R should be sent to R-windows@R-project.org.

윈도우 R에 대한 제안들과 의견들은 R-windows@R-project.org로 보내져야 합니다.

Corrections to and comments on message translations should be sent to the last translator (listed at the top of the appropriate ‘.po’ file) or to the translation team as listed at http://developer.R-project.org/TranslationTeams.html. 메세지 변역에 대한 정확성과 의견들은 http://developer.R-project.org/TranslationTeams.html에 명시된 최근의 번역자(".po" 란 약어로 상위에 명시된) 혹은 변역팀에게 보내져야 합니다.

R Web Interfaces

Rweb is developed and maintained by Jeff Banfield. The Rweb Home Page provides access to all three versions of Rweb—a simple text entry form that returns output and graphs, a more sophisticated JavaScript version that provides a multiple window environment, and a set of point and click modules that are useful for introductory statistics courses and require no knowledge of the R language. All of the Rweb versions can analyze Web accessible datasets if a URL is provided.

Rweb은 Jeff Banfield 에 의해 발전되고 유지되고 있습니다. Rweb 홈 페이지는 Rweb의 모든 세가지 버전에 대한 엑세스를 제공합니다. 첫번째 버전은 간단한 텍스트 입장 형식으로써 산출과 그래프들을 되돌립니다. 두번째는 더욱 정교한 JavaScript 버전으로, 이것은 다양한 윈도우 환경들을 제공합니다. 마지막은 a set of point 와 클릭 모듈로써, 이것들은 입문적인 통계코스에 유용하고, R 언어에 대한 지식을 요구하지 않습니다. URL이 제공되기만 한다면, 모든 Rweb 버전들은 Web accessible datasets 들을 분석할수 있습니다.

The paper “Rweb: Web-based Statistical Analysis”, providing a detailed explanation of the different versions of Rweb and an overview of how Rweb works, was published in the Journal of Statistical Software (http://www.jstatsoft.org/v04/i01/).

“Rweb: Web-based Statistical Analysis”(다른 버전의 Rweb의 세부적인 설명과 Rweb이 어떻게 수행되는지에 대한 overview를 제공함)은 the Journal of Statistical Software (http://www.jstatsoft.org/v04/i01/)안에서 출간되었습니다.

Ulf Bartel has developed R-Online, a simple on-line programming environment for R which intends to make the first steps in statistical programming with R (especially with time series) as easy as possible. There is no need for a local installation since the only requirement for the user is a JavaScript capable browser. See http://osvisions.com/r-online/ for more information.

Ulf Bartel는 R-Oline을 발전시켜 오고 있습니다. R-Online은 편리한 온라인 프로그램 환경으로서, R을 이용해 통계적인 프로그래밍의 첫 번째 단계(특별히 연속적인 시간과 함께)를 가능한한 쉽게 만들 목적으로 합니다.

Rcgi is a CGI WWW interface to R by MJ Ray. It had the ability to use “embedded code”: you could mix user input and code, allowing the HTML author to do anything from load in data sets to enter most of the commands for users without writing CGI scripts. Graphical output was possible in PostScript or GIF formats and the executed code was presented to the user for revision. However, it is not clear if the project is still active. Currently, a modified version of Rcgi by Mai Zhou (actually, two versions: one with (bitmap) graphics and one without) as well as the original code are available from http://www.ms.uky.edu/~statweb/.

Rchi는 MJ Ray에 의한CGI WWW 인터페이스 입니다. 그 것은 "embedded code"(사용자는 유저입력과 코드를 조합할수 있다는 뜻이고, 이는 HTML 저자에게 data sets load 부터 CGI script를 입력할 필요없이 유저들이 대부분의 명령어에 입력하는 것 까지, 무엇이든지 할수 있게 해줌) 사용 할 수 있었습니다. graphical output은 PostScript 혹은 GIF formats에서 가능했고, 실행된 코드는 수정을 위해 유저에게 보여집니다. 하지만, 그 프로젝트가 여전히 활동적이라면, 그 것은 명백하지 않습니다. 최근에, Mai Zhou이 조정한 Rcgi 버전(사실상, 두가지 버전임, 비트맵 그래픽 버전과 비트맵 그래픽이 없는 버전) 뿐만아니라 오리지날 코드는 http://www.ms.uky.edu/~statweb/ 로 부터 이용될 수 있습니다.

CGI-based web access to R is also provided at http://hermes.sdu.dk/cgi-bin/go/. There are many additional examples of web interfaces to R which basically allow to submit R code to a remote server, see for example the collection of links available from http://biostat.mc.vanderbilt.edu/twiki/bin/view/Main/StatCompCourse. R의 웹 엑세스를 기초로한 CGI는 또한 http://hermes.sdu.dk/cgi-bin/go/ 에서 제공되어 집니다. 다음은 R에 대한 웹 인터페이스의 많은 추가적인 예들입니다. 사용자는 기본적으로 remote server에 R 코드를 제출할 수 있습니다. 예를 위해 http://biostat.mc.vanderbilt.edu/twiki/bin/view/Main/StatCompCourse 에서 이용할 수 있는 link 모음를 보십시오

David Firth has written CGIwithR, an R add-on package available from CRAN. It provides some simple extensions to R to facilitate running R scripts through the CGI interface to a web server, and allows submission of data using both GET and POST methods. It is easily installed using Apache under Linux and in principle should run on any platform that supports R and a web server provided that the installer has the necessary security permissions. David's paper “CGIwithR: Facilities for Processing Web Forms Using R” was published in the Journal of Statistical Software (http://www.jstatsoft.org/v08/i10/). The package is now maintained by Duncan Temple Lang and has a web page at http://www.omegahat.org/CGIwithR/.

David Firth CGIwithR을 작성해오고 있다. R add-on package는 CRAN 으로부터 이용할 수 있습니다. 그 것은 R이 웹 서버를 위한 CGI 인터페이스를 통해, R scripts가 수행되는 것을 가능하게 하는 몇몇의 간단한 확장기능을 제공하고, GET 과 POST 메소드를 사용하는 data 제출을 가능하게 합니다. 그 것은 리눅스의 Apache를 이용하여 쉽게 설치될 수 있고, R과 필수적 security permissions을 가지고 있는 인스톨러를 제공하는 웹 서버를 제공하는 어떠한 플랫폼에서 든지 실행되어야 합니다. David의 paper: Facilities for Processing Web Forms Using R" Journal of Statistical Software(http://www.jstatsoft.org/v08/i10/). 에서 출간되었습니다. 그 패키지는 웹 페이지 http://www.omegahet.org/CGIwithR/ 를 가지고 있고, Duncan Temple에 의해서 유지되고 있습니다.

Rpad, developed and actively maintained by Tom Short, provides a sophisticated environment which combines some of the features of the previous approaches with quite a bit of JavaScript, allowing for a GUI-like behavior (with sortable tables, clickable graphics, editable output), etc.

Tom Short 에 의해서 활동적으로 유지되어지고 발전되오고 있는 Rpad는 상당한 양의 JavaScript와 함께 몇몇의 초기 접근법의 특징을 조합하는 정교한 환경을 제공합니다. 앞의 JavaScript는 GUI-like behavior(sortable tables, clickable graphics, editable output 등등) 를 가능하게 합니다.

Jeff Horner is working on the R/Apache Integration Project which embeds the R interpreter inside Apache 2 (and beyond). A tutorial and presentation are available from the project web page at http://biostat.mc.vanderbilt.edu/twiki/bin/view/Main/RApacheProject.

Jeff horner는 R interpreter를 Apache 2 (그리고 상위 버전들도 함께) 에 끼워넣는 R/Apache Integration Project에서 일하고 있습니다. 튜토리얼과 프레젠테이션을 웹페이지 http://biostat.me.vanderbilt.edu /twiki/bin/view/Main/RApacheProject 에서 이용할 수 있습니다.

Rserve is a project actively developed by Simon Urbanek. It implements a TCP/IP server which allows other programs to use facilities of R. Clients are available from the web site for Java and C++ (and could be written for other languages that support TCP/IP sockets).

Rserve는 Simon Urbanek 에 의해서 활동적으로 발전되 하나의 프로젝트 입니다. 그것을 다른 프로그램들이 R 의 시설들은 이용할수 있도로 해주는 TCP/IP 를 실행시킵니다. 클라이언트는 Java 와 C++(TCP/IP 를 지원하는 다른 언어도 포함) 에 대해서 그 웹사이트에서 이용할 수 있습니다.

OpenStatServer is being developed by a team lead by Greg Warnes; it aims “to provide clean access to computational modules defined in a variety of computational environments (R, SAS, Matlab, etc) via a single well-defined client interface” and to turn computational services into web services.

OpenStatServer 는 Greg Warnes가 이끄는 팀에 의해 발전되고 있습니다. 그 것은 다양한 computational environments(R, SAS, Matlab, ect)라고 정의된 computational modules에 대한 명백한 엑세스를 하나의 클라이언트 인터페이를 경유해서 제공하는 것을 목적으로 하고, computational services를 웹 서비스 안으로 돌리는 것을 목적으로 합니다.

Two projects use PHP to provide a web interface to R. R_PHP_Online by Steve Chen (though it is unclear if this project is still active) is somewhat similar to the above Rcgi and Rweb. R-php is actively developed by Alfredo Pontillo and Angelo Mineo and provides both a web interface to R and a set of pre-specified analyses that need no R code input.

두 프로젝트는 Stecve Chen에 의한 R. R_PHP_Oline(이 프로젝트가 여전히 활동적인지 명백하지 않음에도 불구하고)에 하나의 웹 인터페이스를 제공하기 위해, PHP 사용합니다. R. R_PHP_Oline 위의 Rcgi와 Rweb과 비슷합니다.

webbioc is “an integrated web interface for doing microarray analysis using several of the Bioconductor packages” and is designed to be installed at local sites as a shared computing resource.

webbioc은 몇몇의 Bioconductor packages 를 사용하여 미세 분석을 하기 위한 통합된 하나의 웹 인터페이스 이고, 하나의 공유된 computing resource로써 local sites에 설치되도록 설계되었습니다.

Rwui is a web application to create user-friendly web interfaces for R scripts. All code for the web interface is created automatically. There is no need for the user to do any extra scripting or learn any new scripting techniques. Rwui can also be found at http://rwui.cryst.bbk.ac.uk. Rwui 는 유저가 R script에 대해 친근해 질 수 있도록 해주는 웹 어플리케이션입니다. 그 웹 인터페이스에 대한 모든 코드는 자동적으로 생성되어집니다. 유저는 추가적인 scripting 이나 새로운 scripting 기술을 배울 필요가 없습니다. 사용자는 http://rwui.cryst.bbk.ac.uk 에서 Rwui를 찾을 수 있습니다.

The R.rsp package by Henrik Bengtsson introduces “R Server Pages”. Analogous to Java Server Pages, an R server page is typically HTML with embedded R code that gets evaluated when the page is requested. The package includes an internal cross-platform HTTP server implemented in Tcl, so provides a good framework for including web-based user interfaces in packages. The approach is similar to the use of the brew package with Rapache with the advantage of cross-platform support and easy installation.

Henrik Bengtsson 에 의한 R.rsp 패키지는 "R Server Pages"를 소개합니다. Jana Server Pages와 유사하게, 웹 페이지가 요청되었을 때, R server page는 전형적으로 평가되어지는 삽입된 R 코드를 가지는 HTML입니다. 그 패키지는 Tcl 에서 실행되어진 internal cross-platform HTTP server를 포함합니다. 그래서, R.rsp 패키지는 웹기반 유저 인터페이스를 포함하는 잘 설계된 체계를 제공합니다. 이 접근방법은 쉬운 설치와 cross-platform support의 장점을 가진 Rapache와 함께 brew package의 사용과 비슷합니다.

The Rook package by Jeffrey Horner provides a web server interface borrowing heavily from Ruby's Rack project.

Jeffrey Horner에 의한 Rook package는 하나의 웹 서버 인터페이스(Ruby의 Rack 프로젝트에서 많은 정보를 가져옴)를 제공합니다.

Finally, Concerto is a user friendly open-source Web Interface to R developed at the Psychometrics Centre of Cambridge University. It was designed as an online platform to design and run Computerized Adaptive Tests, but can be also used as a general-purpose R Web Interface. It allows R users with no programming or web designing background to quickly develop flexible and powerful online applications, websites, and psychometrics tests. To maximize its reliability, security, and performance, Concerto relies on the popular and reliable open-source elements such as MySQL server (exchange and storage of the data), Rstudio (R code designing and testing, file management), CKEditor (HTML Layer design), and PHP.

마지막으로, Concerto 는 the Psychometrics Centre of Cambridge University에서 발전된 R 유저 친화 웹 인터페이스 오픈 소스 입니다. 이것은 Computerized Adaptive Tests을 설계하고 실행하기 위한 온라인 플랫폼으로써 디자인 되었지만, 일반적인 목적으로 하는 R Web 인터페이스로도 이용되어 질 수 있습니다. 이 것은 프로그래밍이나 디자인 백그라운드가 없는 R 유저들도 빠르게 강력하고 유연한 어플리케이션, 웹 사이트, psychometrics tests 를 발전시킬수 있도록 해줍니다. 그것의 신뢰성, 보안, 퍼포먼스를 최대화 하기 위해, Concerto는 MySQL server(data의 교환과 저장), Rstudio(R 코드 디자인과 실험, 파일 관리), CKEditor(HTML 레이어 디자인), PHP등과 같은 유명하고 믿을수 있는 오픈 소스 요소들을 신뢰하고 있습니다.

See http://rwiki.sciviews.org/doku.php?id=faq-r#web_interfaces for additional information.

추가적인 정보는 http://rwiki.sciviews.org/doku.php?id=faq-r#web_interfaces 에서 볼수 있습니다.

R Miscellanea

어떻게 해야 list 의 구성요소들을 NULL 값으로 초기화할 수 있나요?

리스트 x 의 구성요소 i 혹은 이름이 주어진 구성요소를 NULL 로 초기화 하기 위해서는 아래와 같이 하시면 됩니다.

```
x[ i ] <- list(NULL)
```

그러나, x [i] 혹은 x [[i]] 자체를 NULL 값을 할당하지 마세요. 그 이유는 해당 구성요소 자체를 리스트로부터 삭제하게 되기 때문입니다.

행렬 x 의 행이름들을 없애기 위해서는 rownames(x) <- NULL 과 같이 하는 것이 쉬울 것입니다. 이는 열의 이름을 삭제하는데도 동일하게 적용될 수 있습니다.

어떻게 나의 워크스페이스 (workspace)를 저장할 수 있나요?

save.image() 함수는 사용자의 .GlobalEnv 내에 있는 모든 객체들을 스타트업(startup) 디렉토리에 있는 .RData 라는 파일로 모두 저장합니다. (이것은 또한 q("yes") 라고 하는 것과 동일합니다). save.image(file) 을 이용하여 사용자는 다른 이름으로도 저장이 가능합니다.

어떻게 하면 나의 워크스페이스를 지울 수 있나요?

현재 활성화되어 있는 환경 (주로 .GlobalEnv) 에 있는 모든 객체들을 삭제하기 위해서는 아래와 같이 하면 됩니다.

```
rm(list=ls(all=TRUE))
```

(all=TRUE 이라는 옵션을 사용하지 않는다면 '.'으로 시작하지 않는 이름을 가진 모든 객체가 삭제됩니다).

어떻게 eval() 과 D() 를 이용하나요?

Strange things will happen if you use eval(print(x), envir = e) or D(x^2, "x"). The first one will either tell you that "x" is not found, or print the value of the wrong x. The other one will likely return zero if x exists, and an error otherwise.

This is because in both cases, the first argument is evaluated in the calling environment first. The result (which should be an object of mode "expression" or "call") is then evaluated or differentiated. What you (most likely) really want is obtained by “quoting” the first argument upon surrounding it with expression(). For example,

```
R> D(expression(x^2), "x")
2 * x
```

Although this behavior may initially seem to be rather strange, is perfectly logical. The “intuitive” behavior could easily be implemented, but problems would arise whenever the expression is contained in a variable, passed as a parameter, or is the result of a function call. Consider for instance the semantics in cases like

```
D2 <- function(e, n) D(D(e, n), n)
```

or

```
g <- function(y) eval(substitute(y), sys.frame(sys.parent(n = 2)))
g(a * b)
```

See the help page for deriv() for more examples.

왜 나의 행렬이 행과 열에 대한 차원정보를 잃어버리게 되나요?

When a matrix with a single row or column is created by a subscripting operation, e.g., row <- mat[2,], it is by default turned into a vector. In a similar way if an array with dimension, say, 2 x 3 x 1 x 4 is created by subscripting it will be coerced into a 2 x 3 x 4 array, losing the unnecessary dimension. After much discussion this has been determined to be a feature.

To prevent this happening, add the option drop = FALSE to the subscripting. For example,

```
rowmatrix <- mat[2, , drop = FALSE]  # creates a row matrix
colmatrix <- mat[, 2, drop = FALSE]  # creates a column matrix
a <- b[1, 1, 1, drop = FALSE]        # creates a 1 x 1 x 1 array
```

The drop = FALSE option should be used defensively when programming. For example, the statement

```
somerows <- mat[index, ]
```

will return a vector rather than a matrix if index happens to have length 1, causing errors later in the code. It should probably be rewritten as

```
somerows <- mat[index, , drop = FALSE]
```

어떻게 해야 자동으로 나의 작업을 불러올 수 있나요?

R has a special environment called .AutoloadEnv. Using autoload(name, pkg), where name and pkg

are strings giving the names of an object and the package containing it, stores some information in this environment. When R tries to evaluate name, it loads the corresponding package pkg and reevaluates name in the new package's environment.

Using this mechanism makes R behave as if the package was loaded, but does not occupy memory (yet).

See the help page for autoload() for a very nice example.

어떻게 옵션(options) 들을 지정해 두어야 하나요?

The function options() allows setting and examining a variety of global "options" which affect the way in which R computes and displays its results. The variable .Options holds the current values of these options, but should never directly be assigned to unless you want to drive yourself crazy—simply pretend that it is a "read-only" variable.

For example, given

```
test1 <- function(x = pi, dig = 3) {
  oo <- options(digits = dig); on.exit(options(oo));
  cat(.Options$digits, x, "\n")
}
test2 <- function(x = pi, dig = 3) {
  .Options$digits <- dig
  cat(.Options$digits, x, "\n")
}
```

we obtain:

```
R> test1()
3 3.14
R> test2()
3 3.141593
```

What is really used is the global value of .Options, and using options(OPT = VAL) correctly updates it. Local copies of .Options, either in .GlobalEnv or in a function environment (frame), are just silently disregarded.

윈도우즈에서는 어떻게 파일이름들이 작동하는 것인가요?

As R uses C-style string handling, '₩' is treated as an escape character, so that for example one can enter a newline as '₩n'. When you really need a '₩', you have to escape it with another '₩'.

Thus, in filenames use something like "c:₩₩data₩₩money.dat". You can also replace '₩' by '/' ("c:/data/money.dat").

플랏(plotting) 을 할때 왜 색상지정에러 (color allocation error) 가 발생하나요?

On an X11 device, plotting sometimes, e.g., when running demo("image"), results in “Error: color allocation error”. This is an X problem, and only indirectly related to R. It occurs when applications started prior to R have used all the available colors. (How many colors are available depends on the X configuration; sometimes only 256 colors can be used.)

One application which is notorious for “eating” colors is Netscape. If the problem occurs when Netscape is running, try (re)starting it with either the -no-install (to use the default colormap) or the -install (to install a private colormap) option.

You could also set the colortype of X11() to "pseudo.cube" rather than the default "pseudo". See the help page for X11() for more information.

어떻게 해야 요인 (factor) 를 수치형 (numeric) 으로 변경할 수 있나요?

It may happen that when reading numeric data into R (usually, when reading in a file), they come in as factors. If f is such a factor object, you can use

```
as.numeric(as.character(f))
```

to get the numbers back. More efficient, but harder to remember, is

```
as.numeric(levels(f))[as.integer(f)]
```

In any case, do not call as.numeric() or their likes directly for the task at hand (as as.numeric() or unclass() give the internal codes).

R 에서 트렐리스 디스플레이 (trellis display) 가 구현되나요?

The recommended package lattice (which is based on base package grid) provides graphical functionality that is compatible with most Trellis commands.

You could also look at coplot() and dotchart() which might do at least some of what you want. Note also that the R version of pairs() is fairly general and provides most of the functionality of splom(), and that R's default plot method has an argument asp allowing to specify (and fix against device resizing) the aspect ratio of the plot.

(Because the word “Trellis” has been claimed as a trademark we do not use it in R. The name “lattice” has been chosen for the R equivalent.)

부모환경 (parent environment) 와 인클로징 (enclosing) 이란 무엇인가요?

Inside a function you may want to access variables in two additional environments: the one that the function was defined in (“enclosing”), and the one it was invoked in (“parent”).

If you create a function at the command line or load it in a package its enclosing environment is the

global workspace. If you define a function f() inside another function g() its enclosing environment is the environment inside g(). The enclosing environment for a function is fixed when the function is created. You can find out the enclosing environment for a function f() using environment(f).

The "parent" environment, on the other hand, is defined when you invoke a function. If you invoke lm() at the command line its parent environment is the global workspace, if you invoke it inside a function f() then its parent environment is the environment inside f(). You can find out the parent environment for an invocation of a function by using parent.frame() or sys.frame(sys.parent()).

So for most user-visible functions the enclosing environment will be the global workspace, since that is where most functions are defined. The parent environment will be wherever the function happens to be called from. If a function f() is defined inside another function g() it will probably be used inside g() as well, so its parent environment and enclosing environment will probably be the same.

Parent environments are important because things like model formulas need to be evaluated in the environment the function was called from, since that's where all the variables will be available. This relies on the parent environment being potentially different with each invocation.

Enclosing environments are important because a function can use variables in the enclosing environment to share information with other functions or with other invocations of itself (see the section on lexical scoping). This relies on the enclosing environment being the same each time the function is invoked. (In C this would be done with static variables.)

Scoping is hard. Looking at examples helps. It is particularly instructive to look at examples that work differently in R and S and try to see why they differ. One way to describe the scoping differences between R and S is to say that in S the enclosing environment is always the global workspace, but in R the enclosing environment is wherever the function was created.

어떻게 해야 플랏라벨 (plot label) 을 변경할 수 있나요?

Often, it is desired to use the value of an R object in a plot label, e.g., a title. This is easily accomplished using paste() if the label is a simple character string, but not always obvious in case the label is an expression (for refined mathematical annotation). In such a case, either use parse() on your pasted character string or use substitute() on an expression. For example, if ahat is an estimator of your parameter a of interest, use

```
title(substitute(hat(a) == ahat, list(ahat = ahat)))
```

(note that it is '==' and not '='). Sometimes bquote() gives a more compact form, e.g.,

```
title(bquote(hat(a) = .(ahat)))
```

where subexpressions enclosed in '.()' are replaced by their values.

There are more worked examples in the mailing list archives.

유효한 이름들은 무엇이 있나요?

When creating data frames using data.frame() or read.table(), R by default ensures that the variable names are syntactically valid. (The argument check.names to these functions controls whether variable names are checked and adjusted by make.names() if needed.)

To understand what names are “valid”, one needs to take into account that the term “name” is used in several different (but related) ways in the language:

A syntactic name is a string the parser interprets as this type of expression. It consists of letters, numbers, and the dot and (for version of R at least 1.9.0) underscore characters, and starts with either a letter or a dot not followed by a number. Reserved words are not syntactic names. An object name is a string associated with an object that is assigned in an expression either by having the object name on the left of an assignment operation or as an argument to the assign() function. It is usually a syntactic name as well, but can be any non-empty string if it is quoted (and it is always quoted in the call to assign()). An argument name is what appears to the left of the equals sign when supplying an argument in a function call (for example, f(trim=.5)). Argument names are also usually syntactic names, but again can be anything if they are quoted. An element name is a string that identifies a piece of an object (a component of a list, for example.) When it is used on the right of the ‘$’ operator, it must be a syntactic name, or quoted. Otherwise, element names can be any strings. (When an object is used as a database, as in a call to eval() or attach(), the element names become object names.) Finally, a file name is a string identifying a file in the operating system for reading, writing, etc. It really has nothing much to do with names in the language, but it is traditional to call these strings file “names”.

R 에서 GAMs 이 구현되나요?

Package gam from CRAN implements all the Generalized Additive Models (GAM) functionality as described in the GAM chapter of the White Book. In particular, it implements backfitting with both local regression and smoothing splines, and is extendable. There is a gam() function for GAMs in package mgcv, but it is not an exact clone of what is described in the White Book (no lo() for example). Package gss can fit spline-based GAMs too. And if you can accept regression splines you can use glm(). For Gaussian GAMs you can use bruto() from package mda.

source()를 이용하여 파일을 읽어들일때 결과물이 출력되지 않나요?

Most R commands do not generate any output. The command

```
1+1
```

computes the value 2 and returns it; the command

```
summary(glm(y~x+z, family=binomial))
```

fits a logistic regression model, computes some summary information and returns an object of class "summary.glm" (see How should I write summary methods?).

If you type ‘1+1’ or ‘summary(glm(y~x+z, family=binomial))’ at the command line the returned value is automatically printed (unless it is invisible()), but in other circumstances, such as in a source()d file or inside a function it isn't printed unless you specifically print it.

To print the value use

```
print(1+1)
```

or

```
print(summary(glm(y~x+z, family=binomial)))
```

instead, or use source(file, echo=TRUE).

왜 outer() 가 내가 정의한 함수와 올바르게 작동하지 않는가요?

As the help for outer() indicates, it does not work on arbitrary functions the way the apply() family does. It requires functions that are vectorized to work elementwise on arrays. As you can see by looking at the code, outer(x, y, FUN) creates two large vectors containing every possible combination of elements of x and y and then passes this to FUN all at once. Your function probably cannot handle two large vectors as parameters.

If you have a function that cannot handle two vectors but can handle two scalars, then you can still use outer() but you will need to wrap your function up first, to simulate vectorized behavior. Suppose your function is

```
foo <- function(x, y, happy) {
  stopifnot(length(x) == 1, length(y) == 1) # scalars only!
  (x + y) * happy
}
```

If you define the general function

```
wrapper <- function(x, y, my.fun, ...) {
  sapply(seq_along(x), FUN = function(i) my.fun(x[i], y[i], ...))
}
```

then you can use outer() by writing, e.g.,

```
outer(1:4, 1:2, FUN = wrapper, my.fun = foo, happy = 10)
```

왜 anova() 로부터 나오는 결과물은 모델에 있는 요인의 순서에 따라 다른가요?

In a model such as ~A+B+A:B, R will report the difference in sums of squares between the models ~1, ~A, ~A+B and ~A+B+A:B. If the model were ~B+A+A:B, R would report differences between ~1, ~B, ~A+B, and ~A+B+A:B . In the first case the sum of squares for A is comparing ~1 and ~A, in the second case it is comparing ~B and ~B+A. In a non-orthogonal design (i.e., most unbalanced designs) these comparisons are (conceptually and numerically) different.

Some packages report instead the sums of squares based on comparing the full model to the models

with each factor removed one at a time (the famous `Type III sums of squares' from SAS, for example). These do not depend on the order of factors in the model. The question of which set of sums of squares is the Right Thing provokes low-level holy wars on R-help from time to time.

There is no need to be agitated about the particular sums of squares that R reports. You can compute your favorite sums of squares quite easily. Any two models can be compared with anova(model1, model2), and drop1(model1) will show the sums of squares resulting from dropping single terms.

어떻게 해야 PNG 그래픽스를 배치모드에서 생성할 수 있나요?

Under a Unix-like, if your installation supports the type="cairo" option to the png() device there should be no problems, and the default settings should just work. This option is not available for versions of R prior to 2.7.0, or without support for cairo. From R 2.7.0 png() by default uses the Quartz device on Mac OS X, and that too works in batch mode.

Earlier versions of the png() device uses the X11 driver, which is a problem in batch mode or for remote operation. If you have Ghostscript you can use bitmap(), which produces a PostScript or PDF file then converts it to any bitmap format supported by Ghostscript. On some installations this produces ugly output, on others it is perfectly satisfactory. Many systems now come with Xvfb from X.Org (possibly as an optional install), which is an X11 server that does not require a screen; and there is the GDD package from CRAN, which produces PNG, JPEG and GIF bitmaps without X11.

어떻게 해야 커맨드라인에서 작업이 가능한가요?

The Unix-like command-line interface to R can only provide the inbuilt command line editor which allows recall, editing and re-submission of prior commands provided that the GNU readline library is available at the time R is configured for compilation. Note that the `development' version of readline including the appropriate headers is needed: users of Linux binary distributions will need to install packages such as libreadline-dev (Debian) or readline-devel (Red Hat).

어떻게 해야 스트링 (string) 을 변수 (variable) 로 변경할 수 있나요?

If you have

```
varname <- c("a", "b", "d")
```

you can do

```
get(varname[1]) + 2
```

for

```
a + 2
```

or

```
assign(varname[1], 2 + 2)
```

for

```
a <- 2 + 2
```

or

```
eval(substitute(lm(y ~ x + variable),
                list(variable = as.name(varname[1]))))
```

for

```
lm(y ~ x + a)
```

At least in the first two cases it is often easier to just use a list, and then you can easily index it by name

```
vars <- list(a = 1:10, b = rnorm(100), d = LETTERS)
vars"a"
```

without any of this messing about.

왜 lattice/trellis 그래픽스가 작동하지 않나요?

The most likely reason is that you forgot to tell R to display the graph. Lattice functions such as xyplot() create a graph object, but do not display it (the same is true of ggplot2 graphics, and Trellis graphics in S-Plus). The print() method for the graph object produces the actual display. When you use these functions interactively at the command line, the result is automatically printed, but in source() or inside your own functions you will need an explicit print() statement.

어떻게 해야 데이터프레임 (data frame) 의 열들을 정렬할 수 있나요?

To sort the rows within a data frame, with respect to the values in one or more of the columns, simply use order() (e.g., DF[order(DF$a, DF"b"),] to sort the data frame DF on columns named a and b).

왜 help.start() 검색엔진이 작동하지 않나요?

The browser-based search engine in help.start() utilizes a Java applet. In order for this to function properly, a compatible version of Java must installed on your system and linked to your browser, and both Java and JavaScript need to be enabled in your browser.

There have been a number of compatibility issues with versions of Java and of browsers. For further details please consult section “Enabling search in HTML help” in R Installation and Administration. This manual is included in the R distribution, see What documentation exists for R?, and its HTML version is linked from the HTML search page.

R 을 업데이트하면 왜 .Rprofile 이 작동하지 않나요?

Did you read the NEWS file? For functions that are not in the base package you need to specify the correct package namespace, since the code will be run before the packages are loaded. E.g.,

```
ps.options(horizontal = FALSE)
help.start()
```

needs to be

```
grDevices::ps.options(horizontal = FALSE)
utils::help.start()
```

(graphics::ps.options(horizontal = FALSE) in R 1.9.x).

어디로 모든 메소드들(methods) 가 가게 되나요?

Many functions, particularly S3 methods, are now hidden in namespaces. This has the advantage that they cannot be called inadvertently with arguments of the wrong class, but it makes them harder to view.

To see the code for an S3 method (e.g., [.terms) use

```
getS3method("[", "terms")
```

To see the code for an unexported function foo() in the namespace of package "bar" use bar:::foo. Don't use these constructions to call unexported functions in your own code—they are probably unexported for a reason and may change without warning.

어떻게 해야 좌표축에 있는 라벨을 회전시킬 수 있나요?

To rotate axis labels (using base graphics), you need to use text(), rather than mtext(), as the latter does not support par("srt").

```
## Increase bottom margin to make room for rotated labels
par(mar = c(7, 4, 4, 2) + 0.1)
## Create plot with no x axis and no x axis label
plot(1 : 8, xaxt = "n",  xlab = "")
## Set up x axis with tick marks alone
axis(1, labels = FALSE)
## Create some text labels
labels <- paste("Label", 1:8, sep = " ")
## Plot x axis labels at default tick marks
text(1:8, par("usr")[3] - 0.25, srt = 45, adj = 1,
     labels = labels, xpd = TRUE)
## Plot x axis label at line 6 (of 7)
mtext(1, text = "X Axis Label", line = 6)
```

When plotting the x axis labels, we use srt = 45 for text rotation angle, adj = 1 to place the right end of text at the tick marks, and xpd = TRUE to allow for text outside the plot region. You can adjust the value of the 0.25 offset as required to move the axis labels up or down relative to the x axis. See ?par

for more information.

Also see Figure 1 and associated code in Paul Murrell (2003), "Integrating grid Graphics Output with Base Graphics Output", R News, 3/2, 7–12.

왜 read.table() 이 비효율적인가요?

By default, read.table() needs to read in everything as character data, and then try to figure out which variables to convert to numerics or factors. For a large data set, this takes considerable amounts of time and memory. Performance can substantially be improved by using the colClasses argument to specify the classes to be assumed for the columns of the table.

패키지 (package) 와 라이브러리 (library) 의 차이점은 무엇인가요?

A package is a standardized collection of material extending R, e.g. providing code, data, or documentation. A library is a place (directory) where R knows to find packages it can use (i.e., which were installed). R is told to use a package (to "load" it and add it to the search path) via calls to the function library. I.e., library() is employed to load a package from libraries containing packages.

See R Add-On Packages, for more details. See also Uwe Ligges (2003), "R Help Desk: Package Management", R News, 3/3, 37–39.

패키지를 설치했음에도 불구하고 함수를 찾을 수 없습니다

To actually use the package, it needs to be loaded using library().

See R Add-On Packages and What is the difference between package and library? for more information.

다음과 같은 숫자들이 왜 R 에서는 같지 않나요?

The only numbers that can be represented exactly in R's numeric type are integers and fractions whose denominator is a power of 2. Other numbers have to be rounded to (typically) 53 binary digits accuracy. As a result, two floating point numbers will not reliably be equal unless they have been computed by the same algorithm, and not always even then. For example

```
R> a <- sqrt(2)
R> a * a == 2
[1] FALSE
R> a * a - 2
[1] 4.440892e-16
```

The function all.equal() compares two objects using a numeric tolerance of .Machine$double.eps ^ 0.5. If you want much greater accuracy than this you will need to consider error propagation carefully.

For more information, see e.g. David Goldberg (1991), "What Every Computer Scientist Should Know About Floating-Point Arithmetic", ACM Computing Surveys, 23/1, 5–48, also available via http://www.validlab.com/goldberg/paper.pdf.

To quote from “The Elements of Programming Style” by Kernighan and Plauger:

10.0 times 0.1 is hardly ever 1.0.

오랜 시간이 걸리는 시뮬레이션 스터디를 할때 어떻게 해야 에러들을 잡거나 혹은 무시할 수 있나요?

Use try(), which returns an object of class "try-error" instead of an error, or preferably tryCatch(), where the return value can be configured more flexibly. For example

```
beta[i,] <- tryCatch(coef(lm(formula, data)),
                     error = function(e) rep(NaN, 4))
```

would return the coefficients if the lm() call succeeded and would return c(NaN, NaN, NaN, NaN) if it failed (presumably there are supposed to be 4 coefficients in this example).

왜 음수의 자승 (powers of negative numbers) 가 잘못되나요?

You are probably seeing something like

```
R> -2^2
[1] -4
```

and misunderstanding the precedence rules for expressions in R. Write

```
R> (-2)^2
[1] 4
```

to get the square of −2.

The precedence rules are documented in ?Syntax, and to see how R interprets an expression you can look at the parse tree

```
R> as.list(quote(-2^2))
1
`-`


2
2^2
```

반복문을 사용할때 매 반복으로부터 나오는 결과를 서로 다른 파일에 어떻게 저장해야 하나요?

One way is to use paste() (or sprintf()) to concatenate a stem filename and the iteration number while file.path() constructs the path. For example, to save results into files result1.rda, ..., result100.rda in the subdirectory Results of the current working directory, one can use

```
for(i in 1:100) {
```

```
    ## Calculations constructing "some_object" ...
    fp <- file.path("Results", paste("result", i, ".rda", sep = ""))
    save(list = "some_object", file = fp)
  }
```

lmer() 을 사용할 때 왜 p-값들이 표시되지 않나요?

Doug Bates has kindly provided an extensive response in a post to the r-help list, which can be reviewed at https://stat.ethz.ch/pipermail/r-help/2006-May/094765.html.

PS 혹은 PDF 파일에 플랏을 저장한뒤 볼때 원하지 않는 경계선 혹은 선들이 보이거나 혹은 격자와 같은 것들이 보이지 않나요?

This can occur when using functions such as polygon(), filled.contour(), image() or other functions which may call these internally. In the case of polygon(), you may observe unwanted borders between the polygons even when setting the border argument to NA or "transparent".

The source of the problem is the PS/PDF viewer when the plot is anti-aliased. The details for the solution will be different depending upon the viewer used, the operating system and may change over time. For some common viewers, consider the following:

Acrobat Reader (cross platform) There are options in Preferences to enable/disable text smoothing, image smoothing and line art smoothing. Disable line art smoothing. Preview (Mac OS X) There is an option in Preferences to enable/disable anti-aliasing of text and line art. Disable this option. GSview (cross platform) There are settings for Text Alpha and Graphics Alpha. Change Graphics Alpha from 4 bits to 1 bit to disable graphic anti-aliasing. gv (Unix-like X) There is an option to enable/disable anti-aliasing. Disable this option. Evince (Linux/GNOME) There is not an option to disable anti-aliasing in this viewer. Okular (Linux/KDE) There is not an option in the GUI to enable/disable anti-aliasing. From a console command line, use:

```
      $ kwriteconfig --file okularpartrc --group 'Dlg Performance' \
                     --key TextAntialias Disabled
```

Then restart Okular. Change the final word to ‘Enabled’ to restore the original setting.

왜 백스래쉬가 스트링에서 이상하게 작동하나요?

This question most often comes up in relation to file names (see How do file names work in Windows?) but it also happens that people complain that they cannot seem to put a single ‘₩’ character into a text string unless it happens to be followed by certain other characters.

To understand this, you have to distinguish between character strings and representations of character strings. Mostly, the representation in R is just the string with a single or double quote at either end, but there are strings that cannot be represented that way, e.g., strings that themselves contains the quote character. So

```
> str <- "This \"text\" is quoted"
> str
[1] "This \"text\" is quoted"
> cat(str, "\n")
This "text" is quoted
```

The escape sequences '₩"' and '₩n' represent a double quote and the newline character respectively. Printing text strings, using print() or by typing the name at the prompt will use the escape sequences too, but the cat() function will display the string as-is. Notice that '"₩n"' is a one-character string, not two; the backslash is not actually in the string, it is just generated in the printed representation.

```
> nchar("\n")
[1] 1
> substring("\n", 1, 1)
[1] "\n"
```

So how do you put a backslash in a string? For this, you have to escape the escape character. I.e., you have to double the backslash. as in

```
> cat("\\n", "\n")
\n
```

Some functions, particularly those involving regular expression matching, themselves use metacharacters, which may need to be escaped by the backslash mechanism. In those cases you may need a quadruple backslash to represent a single literal one.

In versions of R up to 2.4.1 an unknown escape sequence like '₩p' was quietly interpreted as just 'p'. Current versions of R emit a warning.

어떻게 해야 내가 작성한 플랏에서 신뢰구간 (confidence bands) 혹은 에러바 (error bars) 를 표시할 수 있나요?

Some functions will display a particular kind of plot with error bars, such as the bar.err() function in the agricolae package, the plotCI() function in the gplots package, the plotCI() and brkdn.plot() functions in the plotrix package and the error.bars(), error.crosses() and error.bars.by() functions in the psych package. Within these types of functions, some will accept the measures of dispersion (e.g., plotCI), some will calculate the dispersion measures from the raw values (bar.err, brkdn.plot), and some will do both (error.bars). Still other functions will just display error bars, like the dispersion function in the plotrix package. Most of the above functions use the arrows() function in the base graphics package to draw the error bars.

The above functions all use the base graphics system. The grid and lattice graphics systems also have specific functions for displaying error bars, e.g., the grid.arrow() function in the grid package, and the geom_errorbar(), geom_errorbarh(), geom_pointrange(), geom_linerange(), geom_crossbar() and geom_ribbon() functions in the ggplot2 package. In the lattice system, error bars can be displayed with Dotplot() or xYplot() in the Hmisc package and segplot() in the latticeExtra package.

어떻게 해야 플랏에서 두개의 y 축들을 생성할 수 있나요?

Creating a graph with two y-axes, i.e., with two sorts of data that are scaled to the same vertical size

and showing separate vertical axes on the left and right sides of the plot that reflect the original scales of the data, is possible in R but is not recommended. The basic approach for constructing such graphs is to use par(new=TRUE) (see ?par); functions twoord.plot() (in the plotrix package) and doubleYScale() (in the latticeExtra package) automate the process somewhat. See http://rwiki.sciviews.org/doku.php?id=tips:graphics-base:2yaxes for more information, including strong arguments against this sort of graph.

어떻게 해야 함수의 소스코드를 살펴 볼 수 있나요?

대부분의 경우에는 함수의 이름을 타이핑하면 소스코드를 볼 수 있습니다. 그러나, 어떤 경우에 네임스페이스 (namespace) 내에 코드가 가려져 있거나 컴파일되어 있어 볼 수 없는 경우가 있습니다. 이런 경우에 대해서 소스코드를 확인하고자 한다면, Uwe Ligges (2006), "Help Desk: Accessing the sources", R News, 6/4, 43-45 (http://cran.r-project.org/doc/Rnews/Rnews_2006-4.pdf) 문서를 확인하시길 바랍니다.

Why does summary() report strange results for the R^2 estimate when I fit a linear model with no intercept?

As described in ?summary.lm, when the intercept is zero (e.g., from y ~ x - 1 or y ~ x + 0), summary.lm() uses the formula R^2 = 1 - Sum(R[i]^2) / Sum((y[i])^2) which is different from the usual R^2 = 1 - Sum(R[i]^2) / Sum((y[i] - mean(y))^2). There are several reasons for this:

Otherwise the R^2 could be negative (because the model with zero intercept can fit worse than the constant-mean model it is implicitly compared to). If you set the slope to zero in the model with a line through the origin you get fitted values y*=0 The model with constant, non-zero mean is not nested in the model with a line through the origin. All these come down to saying that if you know a priori that E[Y]=0 when x=0 then the `null' model that you should compare to the fitted line, the model where x doesn't explain any of the variance, is the model where E[Y]=0 everywhere. (If you don't know a priori that E[Y]=0 when x=0, then you probably shouldn't be fitting a line through the origin.)

Why is R apparently not releasing memory?

This question is often asked in different flavors along the lines of “I have removed objects in R and run gc() and yet ps/top still shows the R process using a lot of memory”, often on Linux machines.

This is an artifact of the way the operating system (OS) allocates memory. In general it is common that the OS is not capable of releasing all unused memory. In extreme cases it is possible that even if R frees almost all its memory, the OS can not release any of it due to its design and thus tools such as ps or top will report substantial amount of resident RAM used by the R process even though R has released all that memory. In general such tools do not report the actual memory usage of the process but rather what the OS is reserving for that process.

The short answer is that this is a limitation of the memory allocator in the operating system and there is nothing R can do about it. That space is simply kept by the OS in the hope that R will ask for it later. The following paragraph gives more in-depth answer with technical details on how this happens.

Most systems use two separate ways to allocate memory. For allocation of large chunks they will use mmap to map memory into the process address space. Such chunks can be released immediately when they are completely free, because they can reside anywhere in the virtual memory. However, this is a relatively expensive operation and many OSes have a limit on the number of such allocated chunks, so this is only used for allocating large memory regions. For smaller allocations the system can expand the data segment of the process (historically using the brk system call), but this whole area is always contiguous. The OS can only move the end of this space, it cannot create any “holes”. Since this operation is fairly cheap, it is used for allocations of small pieces of memory. However, the side-effect is that even if there is just one byte that is in use at the end of the data segment, the OS cannot release any memory at all, because it cannot change the address of that byte. This is actually more common than it may seem, because allocating a lot of intermediate objects, then allocating a result object and removing all intermediate objects is a very common practice. Since the result is allocated at the end it will prevent the OS from releasing any memory used by the intermediate objects. In practice, this is not necessarily a problem, because modern operating systems can page out unused portions of the virtual memory so it does not necessarily reduce the amount of real memory available for other applications. Typically, small objects such as strings or pairlists will be affected by this behavior, whereas large objects such as long vectors will be allocated using mmap and thus not affected. On Linux (and possibly other Unix-like systems) it is possible to use the mallinfo system call (also see the mallinfo package) to query the allocator about the layout of the allocations, including the actually used memory as well as unused memory that cannot be released.

R Programming

8.1 How should I write summary methods?

Suppose you want to provide a summary method for class "foo". Then summary.foo() should not print anything, but return an object of class "summary.foo", and you should write a method print.summary.foo() which nicely prints the summary information and invisibly returns its object. This approach is preferred over having summary.foo() print summary information and return something useful, as sometimes you need to grab something computed by summary() inside a function or similar. In such cases you don't want anything printed.

Next: How can I inspect R objects when debugging?, Previous: How should I write summary methods?, Up: R Programming 8.2 How can I debug dynamically loaded code?

Roughly speaking, you need to start R inside the debugger, load the code, send an interrupt, and then set the required breakpoints.

See section “Finding entry points in dynamically loaded code” in Writing R Extensions. This manual is included in the R distribution, see What documentation exists for R?.

Next: How can I change compilation flags?, Previous: How can I debug dynamically loaded code?, Up: R Programming 8.3 How can I inspect R objects when debugging?

The most convenient way is to call R_PV from the symbolic debugger.

See section “Inspecting R objects when debugging” in Writing R Extensions.

Next: How can I debug S4 methods?, Previous: How can I inspect R objects when debugging?, Up: R Programming 8.4 How can I change compilation flags?

Suppose you have C code file for dynloading into R, but you want to use R CMD SHLIB with compilation flags other than the default ones (which were determined when R was built).

Starting with R 2.1.0, users can provide personal Makevars configuration files in $HOME/.R to override the default flags. See section "Add-on packages" in R Installation and Administration.

For earlier versions of R, you could change the file R_HOME/etc/Makeconf to reflect your preferences, or (at least for systems using GNU Make) override them by the environment variable MAKEFLAGS. See section "Creating shared objects" in Writing R Extensions.

Previous: How can I change compilation flags?, Up: R Programming 8.5 How can I debug S4 methods?

Use the trace() function with argument signature= to add calls to the browser or any other code to the method that will be dispatched for the corresponding signature. See ?trace for details.

Next: Acknowledgments, Previous: R Programming, Up: Top

R Bugs

What is a bug? How to report a bug Next: How to report a bug, Previous: R Bugs, Up: R Bugs 9.1 What is a bug?

If R executes an illegal instruction, or dies with an operating system error message that indicates a problem in the program (as opposed to something like "disk full"), then it is certainly a bug. If you call .C(), .Fortran(), .External() or .Call() (or .Internal()) yourself (or in a function you wrote), you can always crash R by using wrong argument types (modes). This is not a bug.

Taking forever to complete a command can be a bug, but you must make certain that it was really R's fault. Some commands simply take a long time. If the input was such that you know it should have been processed quickly, report a bug. If you don't know whether the command should take a long time, find out by looking in the manual or by asking for assistance.

If a command you are familiar with causes an R error message in a case where its usual definition ought to be reasonable, it is probably a bug. If a command does the wrong thing, that is a bug. But be sure you know for certain what it ought to have done. If you aren't familiar with the command, or don't know for certain how the command is supposed to work, then it might actually be working right. For example, people sometimes think there is a bug in R's mathematics because they don't understand how finite-precision arithmetic works. Rather than jumping to conclusions, show the problem to someone who knows for certain. Unexpected results of comparison of decimal numbers, for example 0.28 * 100 != 28 or 0.1 + 0.2 != 0.3, are not a bug. See Why doesn't R think these numbers are equal?, for more details.

Finally, a command's intended definition may not be best for statistical analysis. This is a very important sort of problem, but it is also a matter of judgment. Also, it is easy to come to such a conclusion out of ignorance of some of the existing features. It is probably best not to complain about such a problem until you have checked the documentation in the usual ways, feel confident that you

understand it, and know for certain that what you want is not available. If you are not sure what the command is supposed to do after a careful reading of the manual this indicates a bug in the manual. The manual's job is to make everything clear. It is just as important to report documentation bugs as program bugs. However, we know that the introductory documentation is seriously inadequate, so you don't need to report this.

If the online argument list of a function disagrees with the manual, one of them must be wrong, so report the bug.

Previous: What is a bug?, Up: R Bugs 9.2 How to report a bug

When you decide that there is a bug, it is important to report it and to report it in a way which is useful. What is most useful is an exact description of what commands you type, starting with the shell command to run R, until the problem happens. Always include the version of R, machine, and operating system that you are using; type version in R to print this.

The most important principle in reporting a bug is to report facts, not hypotheses or categorizations. It is always easier to report the facts, but people seem to prefer to strain to posit explanations and report them instead. If the explanations are based on guesses about how R is implemented, they will be useless; others will have to try to figure out what the facts must have been to lead to such speculations. Sometimes this is impossible. But in any case, it is unnecessary work for the ones trying to fix the problem.

For example, suppose that on a data set which you know to be quite large the command

```
R> data.frame(x, y, z, monday, tuesday)
```

never returns. Do not report that data.frame() fails for large data sets. Perhaps it fails when a variable name is a day of the week. If this is so then when others got your report they would try out the data.frame() command on a large data set, probably with no day of the week variable name, and not see any problem. There is no way in the world that others could guess that they should try a day of the week variable name.

Or perhaps the command fails because the last command you used was a method for "["() that had a bug causing R's internal data structures to be corrupted and making the data.frame() command fail from then on. This is why others need to know what other commands you have typed (or read from your startup file).

It is very useful to try and find simple examples that produce apparently the same bug, and somewhat useful to find simple examples that might be expected to produce the bug but actually do not. If you want to debug the problem and find exactly what caused it, that is wonderful. You should still report the facts as well as any explanations or solutions. Please include an example that reproduces (e.g., http://en.wikipedia.org/wiki/Reproducibility) the problem, preferably the simplest one you have found.

Invoking R with the --vanilla option may help in isolating a bug. This ensures that the site profile and saved data files are not read.

Before you actually submit a bug report, you should check whether the bug has already been reported and/or fixed. First, try the “Show open bugs new-to-old” or the search facility on http://bugs.R-

project.org/. Second, consult https://svn.R-project.org/R/trunk/doc/NEWS.Rd, which records changes that will appear in the next release of R, including bug fixes that do not appear on the Bug Tracker. (Windows users should additionally consult https://svn.R-project.org/R/trunk/src/gnuwin32/CHANGES.Rd.) Third, if possible try the current r-patched or r-devel version of R. If a bug has already been reported or fixed, please do not submit further bug reports on it. Finally, check carefully whether the bug is with R, or a contributed package. Bug reports on contributed packages should be sent first to the package maintainer, and only submitted to the R-bugs repository by package maintainers, mentioning the package in the subject line.

A bug report can be generated using the function bug.report(). For reports on R this will open the Web page at http://bugs.R-project.org/: for a contributed package it will open the package's bug tracker Web page or help you compose an email to the maintainer.

There is a section of the bug repository for suggestions for enhancements for R labelled ‘wishlist’. Suggestions can be submitted in the same ways as bugs, but please ensure that the subject line makes clear that this is for the wishlist and not a bug report, for example by starting with ‘Wishlist:’.

Comments on and suggestions for the Windows port of R should be sent to R-windows@R-project.org.

Corrections to and comments on message translations should be sent to the last translator (listed at the top of the appropriate ‘.po’ file) or to the translation team as listed at http://developer.R-project.org/TranslationTeams.html.

Acknowledgment

Of course, many many thanks to Robert and Ross for the R system, and to the package writers and porters for adding

이 문서에 기여하신 분들

- 박승순, listed by Gnustats 2012년 4월 5일 (목) 02:49 (KST)

TODO

- 한국어 사용자의 실정에 맞도록 추가, 수정, 및 개작의 자유를 허용하며, 이에 대한 문서의 수정에 대해서는 Gnustats 에게 문의 및 수정내용에 대한 통보를 부탁드립니다.

원본 주소 "http://r-project.kr/w/index.php?title=R_FAQ&oldid=2733"

- 이 문서는 2012년 8월 9일 (목) 02:22에 마지막으로 바뀌었습니다.
- 이 문서는 2,282번 읽혔습니다.