1 General Information: README

This is the README file for the distribution of ESS version 13.09

ESS is a GNU Emacs and XEmacs mode for interactive statistical programming and data analysis. Languages supported: the S family (S, S-PLUS and R), SAS, BUGS/JAGS and Stata. ESS grew out of the desire for bug fixes and extensions to S-mode and SAS-mode as well as a consistent union of their features in one package.

Installation instructions are provided in sections for both Unix and Windows; see below.

The current development team is led by Martin Maechler since August 2004. Former project leader A.J. (Tony) Rossini (rossini@blindglobe.net) did the initial port to XEmacs and has been the primary coder. Martin Maechler (maechler@stat.math.ethz.ch) and Kurt Hornik (Kurt.Hornik@R-project.org) have assisted with the S family and XLispStat. Stephen Eglen (stephen@gnu.org) has worked mostly on R support. Richard M. Heiberger (rmh@temple.edu) has assisted with S/S-PLUS development for Windows. Richard and Rodney A. Sparapani (rsparapa@mcw.edu) have done much of the work improving SAS batch and interactive support. Rodney has also extended ESS to support BUGS/JAGS and has an interest in improving Stata support.

We are grateful to the previous developers of S-mode (Doug Bates, Ed Kademan, Frank Ritter, David M. Smith), SAS-mode (Tom Cook) and Stata-mode (Thomas Lumley).

1.1 License

1.2 Stability

All recent released versions are meant to be release-quality versions. While some new features are being introduced, we are cleaning up and improving the interface. We know that there are many remaining opportunities for documentation improvements, but all contributors are volunteers and time is precious. Patches or suggested fixes with bug reports are much appreciated!

1.3 Requirements

ESS is most likely to work with current/recent versions of the following statistical packages: R/S-PLUS, SAS, Stata, OpenBUGS and JAGS.

ESS supports current, and recent, stable versions of GNU Emacs (currently, specifically, the 23.x and 24.x series; alpha/beta/pre-release versions are NOT SUPPORTED). Non-Windows users beware: GNU Emacs 24.3 is preferable to 24.1 or 24.2: these broken builds

suffer from bug 12463 http://debbugs.gnu.org/cgi/bugreport.cgi?bug=12463 which will cause emacs and ESS to get progressively slower over time.

Due to XEmacs lacking some features that ESS requires, ESS support of XEmacs ends with ESS 12.04-4. This decision will be re-visited in the future as XEmacs continues to sync with GNU Emacs.

To build the PDF documentation, you will need a version of TeX Live or texinfo that includes texi2dvi (BEWARE: recent TeX Live, and some texinfo RPMs, do NOT include texi2dvi).

1.4 Getting the Latest Version

The latest released version of ESS is always available on the web at: ESS web page or StatLib

1.4.1 ESS subversion repository

The latest development version of ESS is available via https://svn.R-project.org/ESS/, the ESS Subversion repository. If you have a Subversion client (see http://subversion.tigris.org/), you can download the sources using:

```
% svn checkout https://svn.r-project.org/ESS/trunk path
```

which will put the ESS files into directory *path*. Later, within that directory, 'svn update' will bring that directory up to date. Windows-based tools such as TortoiseSVN are also available for downloading the files. Alternatively, you can browse the sources with a web browser at: ESS SVN site. However, please use a subversion client instead to minimize the load when retrieving.

If you remove other versions of ESS from your emacs load-path, you can then use the development version by adding the following to .emacs:

```
(load "/path/to/ess-svn/lisp/ess-site.el")
```

Note that https is required, and that the SSL certificate for the Subversion server of the R project is

```
Certificate information:
 - Hostname: svn.r-project.org
 - Valid: from Jul 16 08:10:01 2004 GMT until Jul 14 08:10:01 2014 GMT
 - Issuer: Department of Mathematics, ETH Zurich, Zurich, Switzerland, CH
 - Fingerprint: c9:5d:eb:f9:f2:56:d1:04:ba:44:61:f8:64:6b:d9:33:3f:93:6e:ad
```

(currently, there is no "trusted certificate"). You can accept this certificate permanently and will not be asked about it anymore.

1.4.2 Git for development

For development and experimentation on new features, there is now a GitHub branch for ESS, available at https://github.com/emacs-ess/ESS.

1.5 Installation

1.6 Unix (Linux / MacOS / Solaris / ...) Installation

For a **Unix or Unix-like installation**, please follow the next steps. Retrieve the latest tgz file ('`ess-VERSION.tgz`') from ESS downloads area.

GNU Emacs Simple Instructions: for recent versions of Emacs (23.x or higher) and ESS, the installation process is simple.

1. Extract all the files from '`ess-VERSION.tgz`' into the directory '`PREFIX/site-lisp`' where '`PREFIX`' is appropriate for GNU Emacs on your system; '`PREFIX`' will most likely be either '`/usr/share/emacs`' or '`/usr/local/share/emacs`' (on Mac OS X, '`PREFIX`' will most likely be something like '`/Applications/Emacs.app/Contents/Resources`'):

   ```
   GNU tar  % gtar zxf ess-VERSION.tgz -C PREFIX/site-lisp
   Unix tar % gunzip < ess-VERSION.tgz | tar xf - -C PREFIX/site-lisp
   ```

2. Then, add the line

   ```
   (require 'ess-site)
   ```

   to '`~/.emacs`' and restart Emacs.

3. If you see a buffer named '`*ESS*`', then the simple instructions were most likely successful. If not, then read further.

4. It could be that you have an older version of Emacs, some other problem with your installation or you are not a sysadmin. Whatever the case, you may need to edit '`ess-site.el`' manually. If that is the case, then you should create a directory just for ESS like '`~/ess`' or '`/usr/local/ess`' and unpack ESS there. That way, your changes to '`ess-site.el`' will not be lost if you update Emacs later.

5. Replace the line above with

   ```
   (load "~/ess/ess-VERSION/lisp/ess-site")
   ```

   in '`~/.emacs`' and restart Emacs.

6. If you see a buffer named '`*ESS*`', then the manual instructions were most likely successful. If not, then send a query to `ess-help@r-project.org` explicitly describing your problem and your environment including operating system, Emacs version, ESS version, etc.

Due to XEmacs lacking some features that ESS requires, ESS support of XEmacs ends with ESS 12.04-4. This decision will be re-visited in the future as XEmacs continues to sync with GNU Emacs.

XEmacs Simple Instructions: for recent versions of XEmacs (21.4.x or higher) and ESS, the installation process is simple.

1. Extract all the files from '`ess-VERSION.tgz`' when you are in the current working directory of '`/usr/local/LOCATION/xemacs/site-packages`' which exists for packages like ESS where '`LOCATION`' is '`lib`' for legacy installations and '`share`' now (and for Mac OS X create a link to it from the directory '`/Applications/XEmacs.app/Contents/Resources/site-lisp`'):

   ```
   %prompt gtar zxf ess-VERSION.tgz            # for GNU tar
   %prompt gunzip < ess-VERSION.tgz | tar xf - # for Unix tar
   ```

2. Then, add the line

   ```
   (require 'ess-site)
   ```

   to '`~/.xemacs/init.el`' and restart XEmacs.

3. If you see a buffer named '`*ESS*`', then the simple instructions were most likely successful. If not, then read further.
4. It could be that you have an older version of XEmacs, some other problem with your installation or you are not a sysadmin. Whatever the case, you may need to edit '`ess-site.el`' manually. If that is the case, then you should create a directory just for ESS like '`~/ess`' or '`/usr/local/ess`' and unpack ESS there. That way, your changes to '`ess-site.el`' will not be lost if you update XEmacs later.
5. Replace the line above with

   ```
   (load "~/ess/ess-VERSION/lisp/ess-site")
   ```

   in '`~/.xemacs/init.el`' and restart XEmacs.
6. If you see a buffer named '`*ESS*`', then the manual instructions were most likely successful. If not, then send a query to ess-help@r-project.org explicitly describing your problem and your environment including operating system, XEmacs version, ESS version, etc.

1. **(OPTIONAL) COMPILING E-LISP**:

   Edit the default locations of `LISPDIR`, `INFODIR` and `ETCDIR` in Section 1 of '`Makeconf`' (if you are using XEmacs, then edit the XEmacs subsection in Section 1).

   You can compile those files by:

   ```
   make all
   ```

   When that completes successfully, install the compiled files:

   ```
   make install
   ```

1.7 Microsoft Windows installation

For **Microsoft Windows installation**, please follow the next steps. Retrieve the latest zip file ('`ess-VERSION.zip`') from ESS downloads area.

GNU Emacs Simple Instructions: for recent versions of Emacs (23.x or higher) and ESS, the installation process is simple.

1. Extract all the files from '`ess-VERSION.zip`' (by double clicking on it and selecting "Extract all files" which launches the Folders Extraction Wizard) into an '`ESS`' subdirectory of the '`site-lisp`' directory that exists for packages like ESS. If GNU Emacs was installed in the default location, then this directory can be found somewhere like '`C:\Program Files\GNU Emacs\emacs-24.x\site-lisp`'
2. Add the line

   ```
   (require 'ess-site)
   ```

   to '`~/.emacs`' and restart Emacs.
3. If you see a buffer named '`*ESS*`', then the simple instructions were most likely successful. If not, then read further.
4. It could be you have an older version of Emacs or some other problem with your installation. Either way, you may need to edit '`C:\ess\ess-VERSION\lisp\ess-site.el`' manually. If that is the case, then you should create a directory just for ESS like '`C:\ess`' and unpack ESS there. That way, your changes to '`C:\ess\ess-VERSION\lisp\ess-site.el`' will not be lost if you update Emacs later.

5. Replace the line above with

   ```
   (load "C:/ess/ess-VERSION/lisp/ess-site")
   ```

   in '`~/.emacs`' and restart Emacs.
6. If you see a buffer named '`*ESS*`', then the manual instructions were most likely successful. If not, then send a query to ess-help@r-project.org explicitly describing your problem and your environment including operating system, Emacs version, ESS version, etc.

Due to XEmacs lacking some features that ESS requires, ESS support of XEmacs ends with ESS 12.04-4. This decision will be re-visited in the future as XEmacs continues to sync with GNU Emacs.

XEmacs Simple Instructions: for recent versions of XEmacs (21.x or higher), the installation process is much simpler. Hopefully, these simple instructions will work for you. If not, then more detailed, manual instructions follow.

1. Extract all the files from '`ess-VERSION.zip`' (by double clicking on it and selecting "Extract all files" which launches the Folders Extraction Wizard) into the '`site-packages`' directory that exists for packages like ESS. If XEmacs was installed in the default location, then this directory can be found at '`C:\Program Files\XEmacs\site-packages`'.
2. XEmacs requires the `HOME` environment variable to be defined. You can create it by visiting the following dialog: `My Computer->Control Panel->System->Advanced->Environment Variables` In the User variables window, press New. And create a variable named `HOME` with a value something like (you must use forward slashes `/` rather than backslashes `\`) `c:/Documents and Settings/%USERNAME%/Application Data`. Then press OK for that window and press OK for the main window. *If you also have GNU Emacs installed, GNU Emacs will recognize* `HOME` *and expand* `~` *accordingly.*
3. Now launch XEmacs and do a `C-x C-f` followed by a `~`. From the Subdir menu, select Create Directory, and enter `.xemacs`
4. Add the line

   ```
   (require 'ess-site)
   ```

   to '`~/.xemacs/init.el`' and restart XEmacs.
5. If you see a buffer named '`*ESS*`', then the simple instructions were most likely successful. If not, then read further.
6. It could be you have an older version of XEmacs or some other problem with your installation. Either way, you may need to edit '`C:\ess\ess-VERSION\lisp\ess-site.el`' manually. If that is the case, then you should create a directory just for ESS like '`C:\ess`' and unpack ESS there. That way, your changes to '`C:\ess\ess-VERSION\lisp\ess-site.el`' will not be lost if you update XEmacs later.
7. Replace the line above with

   ```
   (load "C:/ess/ess-VERSION/lisp/ess-site")
   ```

   in '`~/.xemacs/init.el`' and restart XEmacs.
8. If you see a buffer named '`*ESS*`', then the manual instructions were most likely successful. If not, then send a query to ess-help@r-project.org explicitly describing your problem and your environment including operating system, XEmacs version, ESS version, etc.

Now, you should be ready to use ESS. For example, to edit statistical programs, load the files with the requisite extensions (".sas" for SAS, ".S" or "s" or "q" or "Q" for S-PLUS, ".r" or ".R" for R, and ".lsp" for XLispStat). One further step is needed if you wish to run statistical processes, see below.

To run statistical processes under ESS, Windows users will need to make sure that the directories for the software they will be using is in the PATH environment variable.

On Windows NT/2000/XP, add the directories to the PATH using the `My Computer->Control Panel->System->Advanced->Environment Variables` menu. Note that the directory containing the program is added to the PATH, not the program itself. One such line is needed for each software program. Be sure to use the abbreviation `progra~1` and not the long version with embedded blanks as this may cause problems. Also, make sure to use backslashes \ since Windows requires them.

An alternative, for R users, is that rather than adjusting the PATH variable, you can add the following to your emacs initialization file (and restart emacs):

```
(setq inferior-R-program-name "c:/progra~1/R/R-2.15.1/bin/Rterm.exe")
```

This assumes that you have installed R-2.15.1 in the default location. Change the path otherwise to point to other locations.

Windows users who place S-PLUS anywhere other than the default location will also need to add the following three lines (properly adjusted for their location):

```
(setq-default inferior-S+6-program-name
    "c:/progra~1/Insightful/SPLUS70/cmd/Splus")
(setq-default inferior-Sqpe+6-SHOME-name
    "c:/progra~1/Insightful/SPLUS70")
(setq-default inferior-Sqpe+6-program-name
    "c:/progra~1/Insightful/SPLUS70/cmd/Sqpe.exe")
```

The above example uses the default location of S-PLUS in `c:\progra~1\Insightful`. Please note that ESS considers S-PLUS 6, 7, and 8 to be variants of S+6.

These users may also need to modify the emacs variable `ess-SHOME-versions` to match their installation in order to get the full set of S-PLUS versions on their machine into the ESS menu.

To start the S-PLUS [678].x GUI from ESS under emacs:

1. If you use Cygwin bash as your primary shell, then

   ```
   M-x S
   (or M-x S+6).
   ```

2. If you use the MSDOS prompt window as your primary shell, then

   ```
   M-x S+6-msdos
   ```

You will then be asked for a pathname ("S starting data directory?"), from which to start the process. The prompt will propose your current directory as the default. ESS will start the S-PLUS GUI. There will be slight delay during which emacs is temporarily frozen. ESS will arrange for communication with the S-PLUS GUI using the DDE protocol. Send lines or regions from the emacs buffer containing your S program (for example, '`myfile.s`') to the S-PLUS Commands Window with the `C-c C-n` or `C-c C-r` keys. (If you are still using S-PLUS 4.x or 2000,\ then use `M-x S+4` or `M-x S+4-msdos`.)

To start an S-PLUS [678].x session inside an emacs buffer—and without the S-PLUS GUI:

```
M-x Sqpe
(or M-x Sqpe+6).
```

This works with both the bash and msdos shells. You will then be asked for a pathname ("S starting data directory?"), from which to start the process. The prompt will propose your current directory as the default. You get Unix-like behavior, in particular the entire transcript is available for emacs-style search commands. Send lines or regions from the emacs buffer containing your S program (for example, '`myfile.s`') to the *S+6* buffer with the `C-c C-n` or `C-c C-r` keys. Interactive graphics are available with Sqpe by using the java library supplied with S-PLUS 6.1 and newer releases. Enter the commands:

```
library(winjava)
java.graph()
```

Graphs can be saved from the `java.graph` device in several formats, but not PostScript. If you need a PostScript file you will need to open a separate `postscript` device. (If you are still using S-PLUS 4.x or 2000, then use `M-x Sqpe+4`.)

To connect to an already running S-PLUS GUI (started, for example, from the S-PLUS icon):

```
M-x S+6-existing
```

or

```
M-x S+6-msdos-existing
```

You will then be asked for a pathname ("S starting data directory?"), from which to start the process. The prompt will propose your current directory as the default. ESS will arrange for communication with the already running S-PLUS GUI using the DDE protocol. Send lines or regions from the emacs buffer containing your S program (for example, '`myfile.s`') to the S-PLUS Commands Window with the `C-c C-n` or `C-c C-r` keys. (If you are still using S-PLUS 4.x or 2000, then use `M-x S+4-existing` or `M-x S+4-msdos-existing`.)

If you wish to run R, you can start it with:

```
M-x R
```

1.8 Starting an ESS process

To start an S session on Unix or on Windows when you use the Cygwin bash shell, simply type *M-x S RET*.

To start an S session on Windows when you use the MSDOS prompt shell, simply type *M-x S+6-msdos RET*.

1.9 Current Features

- Languages Supported:
  - S family (R and S+ AKA S-PLUS)
  - SAS
  - OpenBUGS/JAGS
  - Stata

    - Julia
- Editing source code (S family, SAS, OpenBUGS/JAGS, Stata, Julia)
    - Syntactic indentation and highlighting of source code
    - Partial evaluation of code
    - Loading and error-checking of code
    - Source code revision maintenance
    - Batch execution (SAS, OpenBUGS/JAGS)
    - Use of imenu to provide links to appropriate functions
- Interacting with the process (S family, SAS, Stata, Julia)
    - Command-line editing
    - Searchable Command history
    - Command-line completion of S family object names and file names
    - Quick access to object lists and search lists
    - Transcript recording
    - Interface to the help system
- Transcript manipulation (S family, Stata)
    - Recording and saving transcript files
    - Manipulating and editing saved transcripts
    - Re-evaluating commands from transcript files
- Interaction with Help Pages and other Documenation (R)
    - Fast Navigation
    - Sending Examples to running ESS process.
    - Fast Transfer to Further Help Pages
- Help File Editing (R)
    - Syntactic indentation and highlighting of source code.
    - Sending Examples to running ESS process.
    - Previewing

1.10 New Features

Changes/New Features in 13.09:

- font-lock in process buffers doesn't "spill" over prompts. Missing closing string delimiters should not cause wrong fontification of the following command input.
- ESS[julia]: full features M-TAB completion and auto-complete support, which now works for modules, structures and data types.
- ESS[julia]: a much better eldoc showing arguments of methods and data type constructors
- ESS-developer:
    - ESS-developer work-flow pattern has been streamlined: ESS-developer is now automatically activated on per-file basis if the file is part of a developed package `ess-developer-packages`. The old behavior (activation on per-process basis) is still available on `M-x ess-developer` in a process buffer.

  - integration with `devtools` package. New command `ess-developer-load-package` calls `load_all` on the package containing current file. `ess-developer-add-package` now offers IDO menu completions with available loading methods, currently `library`, and `load_all`. Loading command can be customized with `ess-developer-load-on-add-commands`.

- *TAB* now indents region if region is active (a contribution of Matthew Fidler in pull #41)
- `M-x ess-version` now reports full loading path and recognizes git and ELPA versions.
- warning and error keyword are now highlighted with `font-lock-warning-face` as they should be, (for quite some time these keywords have been hijacked by compilation mode fontification).
- eldoc: Eldoc now recognizes multiple processes. If current process is busy, or current buffer is not associated with a process, eldoc picks its completions from the first available free process.
- org-babel: evaluation is now org-friendly
- help: new help buffers now try to reuse ess-help buffers. This behavior is controlled by `ess-help-reuse-window` custom variable.
- help: ?foo pops IDO menu on multiple help files (so far it worked only for *C-c C-v*)
- remote evaluation is considerably faster now on slow connections
- ESS[R] tracebug R source references regular expressions are (mostly) language agnostic.
- `ess-function-call-face` inherits from `font-lock-function-name-face` rather than `font-lock-builtin-face`.
- `ess-inject-source` now accepts `function-and-buffer` option.
- Documentation: The "New Features" section (and '`NEWS`') now represent recent changes: within the last year or so. All changes can be found in the new news.html (or '`NEWS`' and '`ONEWS`').
- ESS[R] `ess-rep-regexp` should no longer inf.loop (rarely!), and hence `M-x ess-fix-miscellaneous` should neither.

Changes/New Features in 13.05:

- ESS[gretl]: Support for `gretl` (both editing and sub-process interaction). A contribution of Ahmadou Dicko.
- ESS: process output display is 4-10 times faster due to new caching and only occasional emacs re-display (for the moment this functionality is available only when `ess-tracebug` is active).
- ESS: *C-c ‘* is now bound to `ess-show-traceback` and *C-c ~* is bound to `ess-show-call-stack`.
- ESS[R]: ESS stores function in 'ESSR' environment to avoid kludging users' global environment and accidental deletion.
- ESS[R]: new variable `ess-swv-processing-command` to control weaving and tangling.
- ESS[R]: `ess-default-style` has been changed (from `DEFAULT`) to `RRR`. Use something like `(setq ess-default-style 'DEFAULT)` or `(setq ess-indent-level 2)` in your '`~/.emacs`' equivalent *before* loading ESS, if you do not like this new "incompatible" default style.

- ESS[julia]: ESS stores its functions in 'ESS' module.
- ESS[julia]: Eldoc is now supported in julia modes
- ESS[julia]: Adjusted error reference detection and interactive help to julia internal changes
- ESS[R]: `ess-use-tracebug`'s default has been changed to `t`. Set it to nil if you want to keep the previous behavior.
- ESS[tracebug]: Electric debug keys have been removed [breaking change] The functionality was replaced with `ess-debug-minor-mode` and `ess-debug-minor-mode-map`.
- ESS[tracebug]: `ess-tracebug-map` is an alias to `ess-dev-map` *C-c C-t*.
- ESS[tracebug]: `ess-bp-toggle-state` (*C-c C-t o*) can now be used during the debug session to toggle breakpoints on the fly (suggestion by Ross Boylan).
- ESS[tracebug]: `ess-debug-flag-for-debugging` and `ess-debug-unflag-for-debugging` work correctly from the debugging contexts. These commands also recognize non-exported functions for the packages listed in `ess-developer-packages` (*C-c C-t C-a*).
- ESS[R]: Eldoc (activated by `ess-use-eldoc`) has become more sophisticated, and hence also more intruding in the interface between the Statistics softare, e.g., R, and the user.

  Note that you can turn off ElDoc, by placing `(setq ess-use-eldoc nil)` in your '`~/.emacs`' file, prior to loading ESS,
- ESS[SAS]: long over-looked `SAS-mode-hook` appears!
- ESS[SAS]: `ess-sas-edit-keys-toggle` now defaults to `t` since `sas-indent-line` is still broken, i.e. *TAB* is now bound to `ess-sas-tab-to-tab-stop` by default

Changes/Bug Fixes in 12.09-2:

- ESS: new `ess-switch-to-end-of-proc-buffer` variable that controls whether *C-c C-z* switches to the end of process buffer. The default is `t`. You can use prefix argument to *C-c C-z* to toggle this variable.
- ESS: fix in `ess-eval-linewise` that was causing emacs to hang during R debugging with `ess-eval-visibly` equal to `t`.
- ESS: fix in `ess-eval-linewise` that was causing emacs to recenter the prompt in visible window
- ESS[tracebug]: A better handling of "Selection" prompts and debug related singlekey commands.
- ESS: fix a bug in `ess-switch-process` that was causing `*new*` selection to fail.
- ESS[R]: Solve missing `ess-local-process-name` bug in R-dired.
- ESS[SWV]: `ess-swv-PDF` doesn't ask for a command to run if there is only one command in `ess-swv-pdflatex-commands`.
- ESS[SWV]: `ess-swv-weave` gained an universal argument to allow for an interactive choice between available weavers (sweave, knitr).
- ESS: `ess-eval-*-and-step` functions go to next empty line at eob, instead of staying at the last line.

Changes/New Features in 12.09-1:

- ESS *Breaking Changes in Keys*:
  - New keymaps: `ess-doc-map` bound to *C-c C-d*; `ess-extra-map` bound to *C-c C-e*; `ess-dump-object-into-edit-buffer` was moved on *C-c C-e C-d*
  - roxygen map was moved on *C-c C-o* and *ess-roxy-update-entry* now resides on *C-c C-o C-o*
  - ess-handy-commands is not bound anymore
  - `ess-dev-map` (including `ess-tracebug` and `ess-developer`) moved on *C-c C-t*
  - `C-c C-y` is deprecated in favor of `C-c C-z C-z`
- ESS[R] new command `ess-describe-object-at-point` bound to *C-c C-d C-e* (repeat *C-e* or *e* to cycle). It was inspired by Erik Iverson's `ess-R-object-tooltip`. Customize `ess-describe-at-point-method` to use tooltip instead of an electric buffer.
- ESS: New command `ess-build-tags-for-directory` bound to *C-c C-e C-t* for building dialect specific tag tables. After building tags use *M-.* to navigate to function and objects definitions. By default *C-c C-e C-t* builds tags based on imenu regular expressions and also include other common languages `.c, .o, .cpp` etc. But it relies on external `find` and `etags` commands. If `ess-build-tags-command` is defined (for `R`), the inferior process is asked to build tags instead.
- ESS: `ess-switch-process` offers `*new*` alternative to start a new process instead of switching to one of the currently running processes.
- ESS: Switching between processes (*C-c C-s*) uses buffer names instead of the internal process names. Use `M-x rename-buffer` command to conveniently rename your process buffers.
- ESS: Process buffers can be automatically named on process creation according to user specified scheme. Default schemes are *proc*, *proc:dir* and *proc:abbr-long-dir* where `proc` stands for the internal process name and `dir` stands for the directory where the process was started in. The default is *proc*. For customization see `ess-gen-proc-buffer-name-function`.
- ESS: `ess-eval-visibly-p` is deprecated in favor of `ess-eval-visibly`.
- ESS: New evaluation pattern `nowait`. In addition to old `nil` and `t` values, `ess-eval-visibly` accepts `nowait` for a visible evaluation with no waiting for the process. See `ess-eval-visibly` for details on evaluation patterns.
- ESS: New "Process" menu entry with process related commands and configuration
- iESS: Process buffer is now automatically shown on errors
- ESS: New `ess-switch-to-inferior-or-script-buffer` command bound to *C-c C-z* in both script and process buffers. If invoked form process buffer it switches to the most recent buffer of the same dialect. It is a single key command.
- ESSR-help: On multiple help pages with the same name, *C-c C-v* now asks for user resolution directly in emacs.
- ESS[R] ess-roxy: new variable `ess-roxy-re` for fontification of cases where the number of leading `#` differs from `ess-roxy-str`.
- ESS[R] Eldoc was considerably enhanced. It now finds hidden default S3 methods and displays non-default methods' arguments after trailing `||`.
- ESS[R]: New `ess-display-demos` command bound to *C-c C-d o* and *C-c C-d C-o*

- ESS: New `ess-help-web-search` command bound to *C-c C-d w* and *C-c C-d C-w* to facilitate interactive search of web resources. Implemented for `R`, `Stata` and `Julia`. See also `ess-help-web-search-command`.
- ESS: ess-pdf-viewer-pref accepts now command line arguments
- ESS[Rnw]: Add knitr support. Customize `ess-swv-processor` for the default processor.
- ESS[Rnw]: More thorough renaming of remaining `noweb-*` to `ess-noweb-*`.
- ESS[Rnw] new commands `ess-eval-chunk-and-step` and `ess-eval-chunk` bound to *M-n C-c* and *M-n C-M-x* to mirror standard ess commands in C-c map.
- ESS[R] Auto-completion: new variable `ess-ac-R-argument-suffix` to customize the insertion of trailing `"="`. Defaults to " = ".
- ESS[Julia]: Added index, apropos and web-search to julia.
- ESS help: More evaluation commands were added to ess-help mode (*C-c C-c*, *C-M-x* etc)

Bug Fixes in 12.09-1:

- iESShelp: Multiple help pages with the same name are properly handled on *C-c C-v*
- iESSremote: Evaluation with ESS remote no longer freezes emacs.
- iESS: `comint-previous-prompt` *C-c C-p* no longer stops on secondary prompt "`+`".
- iESS[R], iESS(Sqpe) [S] on Windows: The `options("editor")` is now initialized to `emacsclient` instead of the previous `gnuclient`. The user may need to add the line `(server-start)` to the emacs initialization file. `emacsclient` has been included with emacs since GNU Emacs 22.1.
- ESS[Rnw] Fixed "connection to R" bug (in 12.09 only).
- ESS[Rnw] Explicit `ess-swv-stangle` and `ess-swv-sweave` functions.
- ESS[Rnw] Fixed completion and smart underscore problems cause by unmatched "\"'
- ESS[R] is more careful with the `R` code injection. It now happens only once at the start of the session.
- ESS[R]: Fixed auto-scrolling the comint buffer on evaluation.
- ESS[Julia]: Solve several indentation and word navigation problems.
- ESS[Julia]: Help system works again.

Changes/New Features in 12.09:

- **Due to XEmacs lacking some features that ESS requires, ESS support of XEmacs ends with ESS 12.04-4. This decision will be re-visited in the future as XEmacs continues to sync with GNU Emacs.**
- ESS[R]: On Windows, there is now a new customizable variable (currently called `ess-directory-containing-R`) to tell ESS where to look for the '`Rterm.exe`' executables. The name of the variable and the values it can take are both in beta and subject to change. Prior to this variable, ESS searched only in the default installation directory. Setting this variable now tells ESS how to find '`Rterm.exe`' executables when they are installed somewhere else.
- ESS[julia]: *new* mode for editing julia code ('`*.jl`'). Start with `M-x julia`.

  Full interaction interface, imenu and basic error referencing are available.

- ESS[R] noweb: `noweb-mode` and `noweb-font-lock-mode` have been renamed to `ess-noweb-mode` and `ess-noweb-font-lock-mode` to avoid conflicts with the "real" `noweb-mode`.
- ESS[R] noweb: The long standing font-lock bug has been solved in `ess-noweb` interface.
- ESS: Basic evaluation keys are now bound to `ess-eval-region-*-` functions:
  - *C-M-x* is bound to `ess-eval-region-or-function-or-paragraph`
  - *C-c C-c* is bound to `ess-eval-region-or-function-or-paragraph-and-step`
  - *C-RET* is bound to `ess-eval-region-or-line-and-step`

  Each of these functions first evaluates the region whenever the region is active.
- ESS: *C-M-a*/*C-M-e* now step to beginning/end of paragraph if no function has been detected.
- ESS: `ess-eval-*-and-step` family of functions are now smarter, and don't step to end of buffer or end of chunk code (`@`) when at the end of the code.
- ESS: `ess-handy-commands` function is bound to *C-c h*
- ESS: ESS is now *blinking* the evaluated region. Set `ess-blink-region` to nil to deactivate; `ess-blink-delay` gives the duration of the blink. Evaluated region is "blinked" in `highlight` face.
- ESS[R-help] New key `a` for "apropos()" in help buffers. Also available through *C-c h*.
- ESS[R-help] All R commands of type foo?bar and foo??bar are recognized and redirected into appropriate *ESS-help* buffers.
- ESS[R]: New customization interface for *font-lock*.

  ESS font-lock operates with predefined keywords. Default keywords are listed in `ess-R-font-lock-keywords` and `inferior-R-font-lock-keywords`, which see. The user can easily customize those by adding new keywords. These variables can also be interactively accessed and saved through *ESS/Font-lock* submenu.

  Several new fontification keywords have been added. Most notably the keywords for highlighting of function calls, numbers and operators.
- ESS[R]: auto-complete is now activated by default whenever auto-complete package is detected. Set `ess-use-auto-complete` to nil to deactivate.
- ESS[R]: R AC sources are no longer auto-starting at 0 characters but at the default `ac-auto-start` characters.
- ESS no longer redefines default ac-sources, but only appends `ac-source-filename` to it.
- ESS: `ac-source-R` now concatenates " = " to function arguments.
- ESS: Menus for ESS and iESS have been reorganized and enriched with *Tracebug* and *Developer* submenus.
- ESS[R]: `ess-developer` and `ess-tracebug` commands are available by default in `ess-dev-map` which is bound to *C-c d* in ESS and iESS maps.
- ESS[R]: `eldoc` truncates long lines whenever `eldoc-echo-area-use-multiline-p` is non-nil (the default). Set this variable to t if you insist on multiline eldoc. See also `ess-eldoc-abbreviation-style`.

- ESS[R]: completion code pre-caches arguments of heavy generics such as `plot` and `print` to eliminated the undesirable delay on first request.
- iESS: Prompts in inferior buffers are now highlighted uniformly with `comint-highlight-prompt` face.
- ESS[R]: R process no longer wait for the completion of input in inferior buffer. Thus, long running commands like `Sys.sleep(5)` no longer stall emacs.
- ESS: [R, S, Stata, Julia] have specialized `ess-X-post-run-hook`s, which are run at the end of subprocess initialization.
- ESS[Stata]: All interactive evaluation commands work as expected. On-line comments are removed before the evaluation and multiline comments are skipped on *C-c C-c* and other interactive commands.
- ESS no longer auto-connects to a subprocess with a different dialect than the current buffer's one.
- ESS: `ess-arg-function-offset-new-line` is now a list for all the ESS indentation styles, which results in the following indentation after an open "(":

```
a <- some.function(other.function(
   arg1,
   arg2)
```

- ESS[SAS]: Improved MS RTF support for GNU Emacs; try `ess-sas-rtf-portrait` and `ess-sas-rtf-landscape`.

Changes/Bug Fixes in 12.04-3:

- ESS: basic support for package.el compatibility
- ESS[R]: correct indentation of & and | continuation lines
- `M-x ess-version` shows the svn revision even after `make install`
- ESS[SAS]: improved XEmacs support
- iESS[R]: better finding of previous prompt
- ESS[Stata]: adjusted prompt for mata mode
- ESS[R]: resolved name clashes with cl.el
- ESS[R]: removed dependence on obsolete package assoc
- New `make` target `lisp`, to build the lisp-only part, i.e., not building the docs.

Changes/New Features in 12.04-1:

- iESS[Stata]: New interactive help invocation.
- iESS[Stata]: New custom variable `inferior-STA-start-file`.
- iESS[Stata]: `inferior-STA-program-name` is now "stata" and can be customized.
- ESS[Stata] New sections in stata help files Syntax(*s-S*), Remarks(*r*), Title(*t*).

Bug Fixes in 12.04-1:

- ESS[R]: Better `ess-tracebug` error handling.
- ESS[R]: Corrected `ess-eldoc` help string filtering and improved argument caching.
- ESS[R]: Indentation of non-block if/else/for/while lines fixed.
- `M-x ess-version` should work better.

- ESS: Filename completion now again works inside strings.
- iESS[Stata]: Fixed prompt detection issue.
- ESS[Rd]: R is autostarted also from here, when needed.

Changes/New Features in 12.04:

- ESS: Reverting new behavior of 12.03, *TAB* in `ess-mode` no longer completes by default. If you want smart *TAB* completion in R and S scripts, similarly to iESS behavior, set the variable `ess-tab-complete-in-script` to `t`. Also see `ess-first-tab-never-complete` for how to customize where first *TAB* is allowed to complete.
- ESS: completion is consistently bound to *M-TAB* (aka *M-C-i*) in both Emacs23 and Emacs24.
- ESS: The variable `ess-arg-function-offset-new-line` introduced in ESS(12.03) now accepts a list with the first element a number to indicate that the offset should be computed from the indent of the previous line. For example setting it to '(2) results in:

```
a <- some.function(
  arg1,
  arg2)
```

Changes/New Features in 12.03:

- ESS indentation: new offset variable `ess-arg-function-offset-new-line` controlling for the indentation of lines immediately following open '('. This is useful to shift backwards function arguments after a long function call expression:

```
a <- some.function(
       arg1,
       arg2)
```

instead of the old

```
a <- some.function(
                    arg1,
                    arg2)
```

If '(' is not followed by new line the behavior is unchanged:

```
a <- some.function(arg1,
                    arg2)
```

This variable should be set as part of indentation style lists, or in ess-mode hook.

- ESS[R]: *C-c .* sets (indentation) style.
- ESS: In ESS buffers `yank`(*C-y*) command accepts double argument *C-u C-u* to paste commands only. It deletes any lines not beginning with a prompt, and then removes the prompt from those lines that remain. Useful to paste code from emails, documentation, inferior ESS buffers or transcript files.
- Documentation: ESS user manual has been rearranged and completed with several new chapters and sections to reflect newly added features ("Completion", "Developing with ESS", "ESS tracebug", "ESS developer", "ESS ElDoc", "IDO Completion" and "Evaluating Code")
- RefCard: Reference card was updated to include new features.

- Eldoc: Eldoc was rewritten and is activated by default. See `ess-use-eldoc`, `ess-eldoc-show-on-symbol`, `ess-eldoc-abbreviation-style` variables for how to change the default behavior. *Note:* `skeleton-pair-insert-maybe` prohibits eldoc display, on `(` insertion.
- ESS[R]: Eldoc shows arguments of a generic function whenever found.
- ESS: *TAB* in `ess-mode` now indents and completes, if there is nothing to indent. Set `ess-first-tab-never-completes-p` to `t` to make *TAB* never complete on first invocation. Completion mechanism is similar to the completion in the `inferior-ess-mode` – a filename expansion is tried, if not found ESS completes the symbol by querying the process.
- ESS for emacs version 24 or higher: ESS is fully compatible with the emacs 24 completion scheme, i.e. all the completion is done by `completion-at-point`. Also in accordance with emacs conventions, ESS doesn't bind *M-TAB* for emacs 24 or higher. *M-TAB* calls the default `complete-symbol`.
- ESS[R]: Out of the box integration with `Auto Completion` mode (http://cx4a.org/software/auto-complete/). Three AC sources `ac-source-R-args`, `ac-source-R-objects` and `ac-source-R` are provided. The last one combines the previous two and makes them play nicely together. Set `ess-use-auto-complete` to `t` to start using it. Refer to documentation string of `ac-use-auto-complete` for further information.
- ESS[R]: New unified and fast argument completion system, comprised of `ess-funname.start`, `ess-function-arguments`, `ess-get-object-at-point`. Eldoc and auto-completion integration are using this system.
- ESS: `ess-switch-to-end-of-ESS`(*C-c C-z*), and `ess-switch-to-ESS`(*C-c C-y*): Automatically start the process whenever needed.
- ESS[R]: `roxy` knows about previewing text version of the documentation. Bound to *C-c C-e t*.
- ESS[R]: Solved the "nil filename" bug in roxygen support.
- ESS[R]: `ess-tracebug` is now part of ESS:

  New Features:

  - Source injection: Tracebug now can inject source references on the fly during code evaluation, i.e. you don't have to source your file, but just evaluate your code in normal fashion. Variable `ess-tracebug-inject-source-p` controls this behavior - if t, always inject source reference, if `'function`, inject only for functions (this is the default), if `nil`, never inject.

    During the source injection the value of `ess-eval-visibly` is ignored.
  - Org-mode support: Visual debugger is now aware of the temporary org source editing buffer (*C-c '*) and jumps through this buffers if still alive, or in original org buffer otherwise.
  - New keys in watch mode: `?` and *d*
  - Two new hooks: ess-tracebug-enter-hook and ess-tracebug-exit-hook
- ESS[R]: New package `ess-developer` to evaluate `R` code directly in the package environment and namespace. It can be toggled on and off with *C-c d t*. When `ess-developer` is on all ESS evaluation commands are redefined to evaluate code in appropriate environments. Add package names to the list of your development packages with *C-d* `a`,

and remove with `C-d r`. Source the current file with `C-d s`.Evaluation function which depend on `'ess-eval-region'` ask for the package to source the code into, `ess-eval-function` and alternatives search for the function name in the development packages' environment and namespace and insert the definition accordingly. See the documentation section "Developing with ESS/ESS developer" for more details.

- ESS[R] help system:

  New Features:

  - `q` quits window instead of calling `ess-switch-to-end-of-ESS`. This is consistent with emacs behavior help and other special buffers (*breaking change*).
  - `k` kills window without asking for the name (pointed by Sam Steingold)
  - Help map inherits from `special-mode-map` as sugested by Sam Steingold.
  - Package index: new function `ess-display-index` bound to `i` in help mode map.
  - Package vignettes: new function `ess-display-vignettes` bound to `v` in help mode map.
  - Display help in HTML browser: new function `ess-display-help-in-browser` bound to `w` in help mode map. It depends on `R`'s `browser` option.
  - New custom variable `ess-help-pop-to-buffer`: if non-nil ESS help buffers are given focus on display. The default is `t` (*breaking change*).
  - New menu entries for the above functions.
  - Bogus help buffers are no longer generated by default, i.e. buffers of the form "No documentation for 'foo' in specified packages and libraries: you could try '??foo' ". `ess-help-kill-bogus-buffers` now defaults to `t`. Beware, there may be instances where the default is unsatisfactory such as debugging and/or during R development. Thanks to Ross Boylan for making the suggestion, Sam Steingold for reminding us of this variable and Martin Maechler for the warning.

- ESS now uses `IDO` completing read functionality for all the interactive requests. It uses ido completion mechanism whenever available, and falls back on classical completing-read otherwise. You can set `ess-use-ido` to nil if you don't want the IDO completion. See the documentation string of `ess-use-ido` for more information about `IDO` and ESS configuration.

- ESS[S]: "`,`" is bound to ess-smart-comma: If comma is invoked at the process marker of an ESS inferior buffer, request and execute a command from `'ess-handy-commands'` list. If `ess-R-smart-operators` is t `'ess-smart-comma` also inserts " " after comma.

- ESS[S], notably `R`: Variable `'ess-handy-commands'` stores an alist of useful commands which are called by `ess-smart-comma` in the inferior buffer.

  Currently containing:

  change-directory
  : `ess-change-directory`

  help-index
  : `ess-display-index`

  help-object
  : `ess-display-help-on-object`

  vignettes
  : `ess-display-vignettes`

objects[ls] `ess-execute-objects`

search `ess-execute-search`

set-width `ess-execute-screen-options`

install.packages
`ess-install.packages`

library `ess-library`

setRepos `ess-setRepositories`

sos `ess-sos`

Handy commands: `ess-library`, `ess-install.packages`, etc - ask for item with completion and execute the correspond command. `ess-sos` is a interface to `findFn` function in package `sos`. If package `sos` is not found, ask user for interactive install.

- ESS: New dynamic mode line indicator: Process status is automatically reflected in all mode-lines of associated with the process buffers. Particularly useful for displaying debug status of `ess-tracebug` and developer status of `ess-developer` in all associated buffers.
- ESS: New `ess-completing-read` mechanism: ESS uses `ido` completions whenever possible. Variable `ess-use-ido` controls whether to use ido completion or not. Active by default.
- ESS now supports comint fields for output and input detection. This feature is not used by default, but might be useful in the future.
- ESS[S]: New custom variable `inferior-ess-S-prompt` to customize prompt detection regular expression in the inferior ESS buffers. You can customize this variable to enhance comint navigation (`comint-previous-prompt` and `comint-next-prompt`) the inferior buffers.
- ESS[R]: Internal `R` completion retrieval (`ess-R-complete-object-name`) was rewritten and is faster now.
- ESS is using process plist to store process specific variables, as opposed to buffer local variables as it was using before. The use of buffer local variables to store process variables is discouraged.
- ESS: new functions to manipulate process plists: `ess-process-get` and `ess-process-set`.
- ESS: Internal process waiting mechanism was completely rewritten. ESS no more relies on prompt regular expressions for the prompt detection. The only requirement on the primary process prompt is to end in `>` . This could be overwritten by setting `inferor-ess-primary-prompt`.
- ESS[S], notably `R`: Saved command history: *ess-history-file* now accepts `t` (default), `nil`, or a file name. By setting it to `nil` no command line history is saved anymore. *ess-history-directory* now allows to have the history all saved in one "central" file.
- ESS[R]: more Roxygen improvements.
- ESS[R]: *C-c .* to set (indentation) style.
- ESS[R]: Functions with non-standard names (for example 'aaa-bbb:cc') are properly handled by font-lock and evaluation routines.

- ESS[R]:Several regexp bugs (described in etc/R-ESS-bugs.el) were fixed in `ess-get-words-from-vector` and `ess-command`.

1.11 Reporting Bugs

Please send bug reports, suggestions etc. to ESS-bugs@stat.math.ethz.ch

The easiest way to do this is within Emacs by typing

M-x ess-submit-bug-report

This also gives the maintainers valuable information about your installation which may help us to identify or even fix the bug.

If Emacs reports an error, backtraces can help us debug the problem. Type "M-x set-variable RET debug-on-error RET t RET". Then run the command that causes the error and you should see a *Backtrace* buffer containing debug information; send us that buffer.

Note that comments, suggestions, words of praise and large cash donations are also more than welcome.

1.12 Mailing Lists

There is a mailing list for discussions and announcements relating to ESS. Join the list by sending an e-mail with "subscribe ess-help" (or "help") in the body to ess-help-request@stat.math.ethz.ch; contributions to the list may be mailed to ess-help@stat.math.ethz.ch. Rest assured, this is a fairly low-volume mailing list.

The purposes of the mailing list include

- helping users of ESS to get along with it.
- discussing aspects of using ESS on Emacs and XEmacs.
- suggestions for improvements.
- announcements of new releases of ESS.
- posting small patches to ESS.

1.13 Authors

- A.J. Rossini
- Richard M. Heiberger
- Kurt Hornik
- Martin Maechler
- Rodney A. Sparapani
- Stephen Eglen
- Sebastian P. Luque
- Henning Redestig
- Vitalie Spinu