

# Package ‘apsrtable’

May 12, 2009

**Version** 0.7-4

**Date** 2009-05-12

**Title** apsrtable model-output formatter for social science

**Author** Michael Malecki

**Maintainer** Michael Malecki <malecki@wustl.edu>

**Depends** R (>= 2.8.0), methods

**Description** Formats latex tables from one or more model objects side-by-side with standard errors below, not unlike tables found in such journals as the American Political Science Review.

**Encoding** UTF-8

**License** GPL (>= 2)

**LazyLoad** yes

**Repository** CRAN

**Date/Publication** 2009-05-12 19:05:39

## R topics documented:

apsrtable . . . . .	2
apsrtableSummary . . . . .	5
modelInfo . . . . .	6
notefunctions . . . . .	8

<b>Index</b>	<b>9</b>
--------------	----------

apsrtable

*APSR-style latex tables with multiple models***Description**

Produce well-formatted LaTeX tables of multiple models side-by-side. Requires `\usepackage{Dcolumn}` in LaTeX/Sweave preamble.

**Usage**

```
apsrtable (... ,
            se=c("robust", "vcov", "both"),
            model.names=NULL, model.counter=1, digits=2,
            stars=1, lev=.05,
            align=c("left", "center", "right"),
            order=c("lr", "rl", "longest"),
            notes=list(se.note(), stars.note() ),
            omitcoef=NULL, coef.names=NULL,
            coef.rows=2,
            multicolumn.align=c("center", "left", "right"),
            col.hspace=NULL,
            Sweave=FALSE, float="table",
            Minionfig=FALSE)
```

**Arguments**

- `...` One or more fitted model objects of a supported class such as `lm` or `glm`. The model-object (a `list`) may also optionally contain an item named `se`: `model$se` may be a vector of standard errors, or a variance-covariance matrix, in which case the square root of the diagonal is used as the “robust” standard errors in the output. See the `se` argument.
- `se` Print the default standard errors (“vcov”), or those supplied by the user (“robust”) in the model, or both? In the case of “both”, user-supplied errors are printed in (parentheses) and the default are printed in [square brackets.] If any model in `...` contains an `se` element and “robust” is chosen (the default), output is labeled as “robust;” if no models have an `se` element (all use `model$vcov`) but `se="robust"`, labeling is simply “Standard errors in parentheses.” Default = “robust”
- `model.names` Optional vector of names to use as column headings in the table. If more models than names are supplied, unnamed models are numbered (starting at one more than the number of names).
- `model.counter` Change the number to start counting from when using automatically numbered models. Default = 1.
- `digits` Number of decimal places to report. Default = 2

stars	Show statistical significance “stars”, either “1” or “default” where “default” is based on the output of <code>summary.lm</code> , except that a superscript dagger is used instead of a dot for $p < .10$ . Here “default” means “the R default”, not to be confused with the function’s (perhaps confusing) <code>Default=1</code>
lev	When <code>stars=1</code> , what level should be used for the test to reject statistical insignificance and bestow the glittering star? Disable decoration entirely by specifying <code>lev=0</code> . <code>Default=.05</code> .
align	How should columns be aligned in the output? Model summaries are always decimal-aligned using <code>Dcolumn</code> (and therefore also set in math mode), but <code>Dcolumn</code> also provides for decimal-point centering. Model names are set in <code>\multicolumn</code> spans with alignment given here, as are model terms (leftmost column of table). <code>Default = “left”</code> .
order	Determines the order in which terms (rows) are included in the output when more than one model (column) is present. “lr” and “rl” take the order of terms from the first or last (leftmost or rightmost) model and appends new terms as they are encountered. “longest” uses the order of terms in the model with the most terms. <code>Default = “lr”</code> .
notes	A list to be evaluated and placed, one item per full-width (multicolumn) line, in footnote size. The default uses two functions, <code>se.note</code> and <code>stars.note</code> to generate notes about the standard errors and indicators of statistical significance. Other notes can be named function calls or simple character strings.
omitcoef	An optional positive-integer, or character, vector of coefficient indices to exclude from the output.
coef.names	An optional vector of names for coefficients. It is recommended to establish the <code>omitcoef</code> and <code>order</code> settings with automatic symbolic naming before supplying a vector of “pretty” variable names.
coef.rows	The number of rows in the table given to each coefficient: by default each coefficient’s standard error is printed in a row beneath it, but setting <code>coef.rows</code> to 1 places it in a new column to the right instead.
multicolumn.align	Alignment for the table’s <code>multicolumn</code> spans: typically only the model names at the top, but, in the case of <code>coef.rows=1</code> , the <code>model.info</code> is also aligned beneath both columns. <code>Default=“center”</code>
col.hspace	Optional <code>hspace</code> (number+tex units such as <code>em</code> ) to insert between each model column(s). Intended mainly to separate models from each other when <code>coef.rows=1</code> . <code>Default=NULL</code>
Sweave	Toggle whether to include <code>\begin{table}... \end{table}</code> , <code>empty \label{}</code> and <code>\caption{}</code> , or only the <code>\begin{tabular} ... \end{tabular}</code> . When called from within an <code>Sweave</code> document one would typically write such elements in the “documentation” (latex-part) rather than inside the code chunk. When called from an <code>Sweave</code> document, make sure to set the code chunk option <code>results=tex</code> . <code>Default = FALSE</code>
float	if <code>Sweave</code> is false – that is, if <code>apsrtable</code> is supposed to wrap the output in the float environment, <code>float</code> allows you to specify an arbitrary custom float environment. Some useful ones include “sidewaystable” (latex package <code>rotating</code> ), or “longtable”

`Minionfig` Include latex command to change the figure style to “tabular” and back to “proportional”, specifically for the `MinionPro` latex package. Default = FALSE

## Details

Given one or several model objects of various types, `apsrtable()` will produce side-by-side output in well-formatted LaTeX using either automated numbering or user-supplied model names and `Dcolumn` decimal-aligned columns. Terms are matched across rows, with options for determining the order of terms. Nuisance terms (e.g. controls, or other quantities not of primary interest) may be omitted. Standard errors of parameter estimates are placed below estimates and in parentheses, with the option for the user to supply a replacement vector of standard errors or a replacement variance-covariance matrix, such as one estimated using the `sandwich` package. By default a single star denotes statistical significance at the .05 level, with the option to employ further decorations or specify another arbitrary level for the test. Finally, some model diagnostics are included along with a (somewhat) flexible means to program or include different items depending on model object class.

Model diagnostic information (“model info”) is handled by formal `modelInfo` methods defined for model summaries. These methods return lists of S3 class `model.info`, named formatted (character) elements. To include fit (or other) information that is available from fitted model objects but *not their summaries*, write an `apsrtableSummary` method to prepare a summary with the items needed for your own `modelInfo` method.

Included are `modelInfo` functions for `lm`, `glm`, and `tobit` and a skelton (incomplete `modelInfo`) for `gee` objects. Please email the author any `modelInfo` functions you write for different model objects for inclusion in future releases.

## Value

A character vector containing lines of latex code. It can be written out using `writeLines` for inclusion via `\input{ }` in latex documents.

## Author(s)

Michael Malecki <malecki at wustl.edu>

## See Also

`modelInfo` for changing the model diagnostic summary information presented and how to include it for different classes of model objects; `notefunctions` for functions to produce dynamic “notes” beneath tables; and `apsrtableSummary` for creating model summaries that produce results compatible with what `apsrtable` expects.

## Examples

```
## Use the example from lm() to show both models:
## Annette Dobson (1990) "An Introduction to Generalized Linear Models".
## Page 9: Plant Weight Data.
ctl <- c(4.17, 5.58, 5.18, 6.11, 4.50, 4.61, 5.17, 4.53, 5.33, 5.14)
trt <- c(4.81, 4.17, 4.41, 3.59, 5.87, 3.83, 6.03, 4.89, 4.32, 4.69)
group <- gl(2, 10, 20, labels=c("Ctl", "Trt"))
weight <- c(ctl, trt)
lm.D9 <- lm(weight ~ group)
```

```
glm.D9 <- glm(weight~group)
lm.D90 <- lm(weight ~ group - 1) # omitting intercept
apsrtable(lm.D90, lm.D9, glm.D9, digits=1, align="center",
          stars="default", model.counter=0, order="r1")
## Not run:
apsrtable(lm.D90, lm.D9, glm.D9, digits=1, align="l",
          stars=1, model.counter=0, order="r1",
          coef.rows=1, col.hspace="3em", float="sidewaystable")
## End (Not run)
```

---

apsrtableSummary    *Custom summary functions for output tables*

---

## Description

Provide alternative model summaries specifically for use with `apsrtable`

## Usage

```
## S3 method for class 'gee':
apsrtableSummary(x)
```

## Arguments

`x`                    A model object to be summarized in a format suitable for `apsrtable` output.

## Details

When preparing model objects for output, `apsrtable` uses primarily the representation of the model provided by its `summary` method. However, some packages return summaries with information that can be confusing to `apsrtable`.

In such an event, you have two options: provide a custom `apsrtableSummary` method, or work with the package maintainers to produce a suitable `summary` object. Ideally, the former is a stopgap for the latter.

## Value

A `summary` representation of a model object, probably derived from the object's own `summary` method.

## Author(s)

Michael Malecki <malecki at wustl.edu>

## Examples

```
### summary.gee produces z scores but not Pr(z). This converts the relevant columns
### to Pr(z) so that apsrstars() works on it, and places the vector of robust se's in
### an $se position which apsrtable expects.

apsrtableSummary.gee <- function(x) {
  s <- summary(x)
  newCoef <- coef(s)
  ## which columns have z scores? (two of them in robust case)
  zcols <- grep("z", colnames(newCoef))
  newCoef[, zcols] <- pnorm(abs(newCoef[, zcols]), lower.tail=FALSE)
  colnames(newCoef)[zcols] <- "Pr(z)"
  s$coefficients <- newCoef
  ## put the robust se in $se so that notefunction works automatically
  ## the se checker will overwrite [,4] with pt, but this doesn't matter
  ## because the last column Pr(z) is used by apsrstars() anyway
  ## and the se are pulled from $se.
  if( class(x) == "gee.robust" ) {
    s$se <- coef(s)[,4]
  }
  return(s)
}
```

---

modelInfo

*Model fit and diagnostic functions for output*

---

## Description

Model diagnostic / summary information to be included in apsrtable output.

## Usage

```
modelInfo(x)
## S4 method for signature 'summary.lm':
modelInfo(x)
## S4 method for signature 'summary.glm':
modelInfo(x)
## S4 method for signature 'summary.tobit':
modelInfo(x)
## S4 method for signature 'summary.gee':
modelInfo(x)
```

## Arguments

x                    A summary object.

## Details

Returns a list containing model diagnostic information, with an interface described here to allow the user to change the information returned and thus presented. The method is called by `apsrtable` within an `lapply` on a list of model summaries. The `modelInfo` methods for a given model summary object simply return a list of arbitrary name-value pairs and give themselves the S3 class `modelInfo`. The `modelInfo` method dispatch uses formal S4 classes, however.

The example shows how one can change the summary for `lm` objects to include only the  $N$  and residual  $\sigma$ .

It is highly likely you will have to call `setOldClass` in order to register new `modelInfo` methods for most types of model summary objects. S4 method dispatch makes changing and overriding these functions much more straightforward than the alternative.

## Value

A list of named character objects representing the lines of model diagnostic information to be included for a given class of model. For example, the default for `lm` reports the  $N$ ,  $R^2$ , adjusted  $R^2$ , and residual  $\sigma$ . The default for `glm` includes the  $N$ , AIC, BIC, and log-likelihood. Common names across model classes in the same table – e.g., the  $N$  – are matched by name, exactly like the model coefficients (indeed, the same functions aggregate terms and order across models.)

## Author(s)

Michael Malecki <malecki at wustl.edu>

## See Also

[sys.frame](#)

## Examples

```
setMethod("modelInfo", "summary.lm", function(x) {
  env <- sys.parent()
  digits <- evalq(digits, env)
  model.info <- list(
    "$N$"=formatC(sum(x$df[1:2]), format="d"),
    "Resid. sd" = formatC(x$sigma, format="f", digits=digits)
  )
  class(model.info) <- "model.info"
  return(model.info)
})

example(apsrtable)

### Switch back to the default
setMethod("modelInfo", "summary.lm", apsrtable:::modelInfo.summary.lm)
## Not run: example(apsrtable)
```

**Description**

Prepare notes about standard errors and statistical significance

**Usage**

```
se.note()
stars.note()
```

**Details**

Table notes are part of the tabular environment and may be based on the content of the table itself. For example, the `stars` argument to `apsrtable` determines whether one or many levels of statistical significance are indicated in the output. The `stars.note` function creates text to place in such a note.

By default the output uses the notation  $*p < .05$  and the example below shows a replacement function that states, “significant at `lev` percent.”.

Thanks to lazy evaluation, you can access variables in the call to `apsrtable` from functions in `notes`. The appropriate environment is 3 levels up the call stack (see the example).

Remember, to escape characters in Latex output, backslashes have to be doubled in R character strings.

**Value**

A character string to place within the tabular environment in `footnotesize` beneath other output.

**Author(s)**

Michael Malecki <malecki at wustl.edu>

**Examples**

```
### Custom note function

signif.pct <- function() {
  env <- sys.frame(-3);
  paste("$^*$ significant at", evalq(lev,env)*100, "percent")
}

### Continue the example from apsrtable
## Not run:
apsrtable(lm.D90, lm.D9, glm.D9, digits=1, align="left",
          stars=1, lev=0.05, model.counter=0, order="rl",
          notes=list(se.note(), signif.pct(),
                    "Plant weight data from the lm() example" )
          )
## End(Not run)
```

# Index

apsrtable, [1](#), [8](#)  
apsrtableSummary, [4](#), [5](#)

gee, [4](#)  
glm, [7](#)

lm, [7](#)

modelInfo, [4](#), [6](#)  
modelInfo,summary.ggee-method  
    (*modelInfo*), [6](#)  
modelInfo,summary.glm-method  
    (*modelInfo*), [6](#)  
modelInfo,summary.lm-method  
    (*modelInfo*), [6](#)  
modelInfo,summary.tobit-method  
    (*modelInfo*), [6](#)

notefunctions, [4](#), [7](#)

se.note, [3](#)  
se.note(*notefunctions*), [7](#)  
setOldClass, [6](#)  
stars.note, [3](#)  
stars.note(*notefunctions*), [7](#)  
sys.frame, [7](#)

tobit, [4](#)