

# Package ‘OVtool’

July 21, 2025

**Type** Package

**Title** Omitted Variable Tool

**Version** 1.0.3

**Description** This tool was designed to assess the sensitivity of research findings to omitted variables when estimating causal effects using propensity score (PS) weighting. This tool produces graphics and summary results that will enable a researcher to quantify the impact an omitted variable would have on their results. Burgette et al. (2021) describe the methodology behind the primary function in this package, `ov_sim`. The method is demonstrated in Griffin et al. (2020) <[doi:10.1016/j.jsat.2020.108075](https://doi.org/10.1016/j.jsat.2020.108075)>.

**License** GPL-3

**Encoding** UTF-8

**LazyData** true

**Imports** Amelia, EnvStats, devtools, dplyr, ggplot2, ggrepel, glue, magrittr, metR, purrr, progress, rlang, survey, stats, tibble, tidyselect, varhandle

**Suggests** rmarkdown, knitr

**Depends** R (>= 2.10), twang

**VignetteBuilder** knitr

**RoxygenNote** 7.1.1

**NeedsCompilation** no

**Author** Lane Burgette [aut, cre],  
Joseph Pane [aut],  
Beth Ann Griffin [aut],  
Daniel McCaffrey [aut]

**Maintainer** Lane Burgette <[burgette@rand.org](mailto:burgette@rand.org)>

**Repository** CRAN

**Date/Publication** 2021-11-02 08:10:07 UTC

## Contents

add_pvals_plot . . . . .	2
add_reps . . . . .	3
es_plot . . . . .	4
es_point_plot . . . . .	5
find_esgrid . . . . .	6
gen_a_finish . . . . .	7
gen_a_start . . . . .	8
outcome_model . . . . .	9
ov_sim . . . . .	10
plot.ov . . . . .	12
prep_for_plots . . . . .	13
sud . . . . .	14
summary.ov . . . . .	16

<b>Index</b>	<b>18</b>
--------------	-----------

---

add_pvals_plot	<i>add_pvals_plot</i>
----------------	-----------------------

---

### Description

Plot of effect size contours with pvalue overlay

### Usage

```
add_pvals_plot(prepare, col = "color")
```

### Arguments

prepare	Input from prep_for_plots
col	If user wants color or black and white. Specify color with "color" or black and white "bw"

### Value

a list of class gg and ggplot

### Examples

```
data(sud)
sud = data.frame(sud[sample(1:nrow(sud),100),])
sud$treat = ifelse(sud$treat == "A", 1, 0)
sud$wts = sample(seq(1, 10, by=.01), size=nrow(sud), replace = TRUE)
outcome_mod = outcome_model(data = sud,
                             weights = "wts",
                             treatment = "treat",
                             outcome = "eps7p_6",
```

```

        model_covariates = c("sfs8p_0", "eps7p_0",
                             "ada_0"),
        estimand = "ATE")

ovtool_results = ov_sim(model_results=outcome_mod,
                        plot_covariates=c("sfs8p_0", "ada_0"),
                        es_grid = 0,
                        rho_grid = 0,
                        n_reps = 2,
                        progress=FALSE)
prep = prep_for_plots(ovtool_results, p_contours=.05)
plot = add_pvals_plot(prepare = prep)

```

---

add_reps	<i>add_reps</i>
----------	-----------------

---

## Description

This function will run additional simulations of the unobserved confounder and add the results to the object returned from the previous call to `OVtool::ov_sim`

## Usage

```
add_reps(OVtool_results, model_results, more_reps)
```

## Arguments

`OVtool_results` The object returned from `OVtool::ov_sim()`

`model_results` The object returned from `OVtool::outcome_model()`

`more_reps` The number of additional repetitions the user wants to simulate the unobserved confounder

## Value

`add_reps` returns an updated object returned from `OVtool::ov_sim()`

## Examples

```

data(sud)
sud = data.frame(sud[sample(1:nrow(sud),100),])
sud$treat = ifelse(sud$treat == "A", 1, 0)
sud$wts = sample(seq(1, 10, by=.01), size=nrow(sud), replace = TRUE)
outcome_mod = outcome_model(data = sud,
                             weights = "wts",
                             treatment = "treat",
                             outcome = "eps7p_6",
                             model_covariates = c("sfs8p_0", "eps7p_0"),
                             estimand = "ATE")

```

```

ovtool_results = ov_sim(model_results=outcome_mod,
                        plot_covariates=c("sfs8p_0"),
                        es_grid = .1,
                        rho_grid = .1,
                        n_reps = 2,
                        progress=FALSE)

additional = add_reps(OVtool_results = ovtool_results,
                    model_results = outcome_mod,
                    more_reps = 2)

```

---

es\_plot

*es\_plot*

---

## Description

Plot of the effect size contours

## Usage

```
es_plot(prepare, col="bw")
```

## Arguments

prepare	Input from prepare_for_plots
col	If user wants color (a heat map layered with contours) or black and white (contours only). Specify the heat map with "color" or black and white "bw".

## Value

a list of class gg and ggplot

## Examples

```

data(sud)
sud = data.frame(sud[sample(1:nrow(sud),100),])
sud$treat = ifelse(sud$treat == "A", 1, 0)
sud$wts = sample(seq(1, 10, by=.01), size=nrow(sud), replace = TRUE)
outcome_mod = outcome_model(data = sud,
                             weights = "wts",
                             treatment = "treat",
                             outcome = "eps7p_6",
                             model_covariates = c("sfs8p_0", "eps7p_0",
                                                    "ada_0"),
                             estimand = "ATE")

ovtool_results = ov_sim(model_results=outcome_mod,
                        plot_covariates=c("sfs8p_0", "ada_0"),
                        es_grid = 0,

```

```

                                rho_grid = 0,
                                n_reps = 2,
                                progress=FALSE)
prep = prep_for_plots(ovtool_results, p_contours=.05)
plot = es_plot(prepare = prep)

```

---

es\_point\_plot

*es\_point\_plot*


---

## Description

Plot of effect size contours with pvalue contours and observed covariate points overlaid

## Usage

```
es_point_plot(prepare, col = "color")
```

## Arguments

prepare	Input from prep_for_plots
col	If user wants color or black and white. Specify color with "color" or black and white "bw".

## Value

a list of class gg and ggplot

## Examples

```

data(sud)
sud = data.frame(sud[sample(1:nrow(sud),100),])
sud$treat = ifelse(sud$treat == "A", 1, 0)
sud$wts = sample(seq(1, 10, by=.01), size=nrow(sud), replace = TRUE)
outcome_mod = outcome_model(data = sud,
                             weights = "wts",
                             treatment = "treat",
                             outcome = "eps7p_6",
                             model_covariates = c("sfs8p_0", "eps7p_0",
                                                  "ada_0"),
                             estimand = "ATE")

ovtool_results = ov_sim(model_results=outcome_mod,
                        plot_covariates=c("sfs8p_0", "ada_0"),
                        es_grid = 0,
                        rho_grid = 0,
                        n_reps = 2,
                        progress=FALSE)
prepare = prep_for_plots(ovtool_results, p_contours=.05)
plot = es_point_plot(prepare = prepare)

```

---

find_esgrid	<i>find_esgrid</i>
-------------	--------------------

---

## Description

Finds a reasonable effect size grid to simulate over.

## Usage

```
find_esgrid(my_data, my_cov, treatment, outcome, my_estimand)
```

## Arguments

my_data	Data
my_cov	vector of covariates
treatment	column name of treatment indicator in my_data
outcome	column name of outcome in my_data
my_estimand	Relevant estimand ("ATE" or "ATT")

## Value

a data frame with three columns, "Cor\_Outcome", "es", and "cov". "Cor\_Outcome" represents rho grid values, "ES" represents the range of grid values to represent the association between the unobserved confounder and the treatment indicator on the effect size scale, and "cov" is a vector of all the covariates used in the propensity score model

## Examples

```
data(sud)
sud = data.frame(sud)
sud$treat = ifelse(sud$treat == "A", 1, 0)
sud$wts = sample(seq(1, 10, by=.01), size=nrow(sud), replace = TRUE)
outcome_mod = outcome_model(data = sud,
                             weights = "wts",
                             treatment = "treat",
                             outcome = "eps7p_3",
                             model_covariates = c("sfs8p_0"),
                             estimand = "ATE")

find_es = find_esgrid(sud, my_cov="sfs8p_0", treatment="treat",
                      outcome="eps7p_3",
                      my_estimand="ATE")
```

---

gen_a_finish	<i>gen_a_finish</i>
--------------	---------------------

---

## Description

This function will generate a at finish.

## Usage

```
gen_a_finish(a_res, my_estimand, wts)
```

## Arguments

a_res	A list of values returned by gen_a_start
my_estimand	"ATE" or "ATT"
wts	A vector of the original weights

## Value

a	used to control the strength of the relationship between the omitted variable and the treatment
---	---

## Examples

```
data(sud)
sud = data.frame(sud)
sud$treat = ifelse(sud$treat == "A", 1, 0)
sud$wts = sample(seq(1, 10, by=.01), size=nrow(sud), replace = TRUE)
outcome_mod = outcome_model(data = sud,
                             weights = "wts",
                             treatment = "treat",
                             outcome = "eps7p_3",
                             model_covariates = c("sfs8p_0"),
                             estimand = "ATE")

start = gen_a_start(y=sud$eps7p_3, tx=sud$treat,
                   residuals=residuals(outcome_mod$mod_results),
                   es = .01,
                   rho = .01,
                   my_estimand = "ATE")

finish = gen_a_finish(a_res = start, my_estimand = "ATE", wts = sud$wts)
```

---

<code>gen_a_start</code>	<i>gen_a_start</i>
--------------------------	--------------------

---

### Description

This function is a wrapper to `ov_simgrid`. It generates the `a`. `a` is used to control the strength of the relationship between the unobserved confounder, `U`, and the treatment indicator

### Usage

```
gen_a_start(y, tx, residuals, es, rho, my_estimand)
```

### Arguments

<code>y</code>	A vector that represents the outcome.
<code>tx</code>	A vector for the treatment indicator (must be 0s and 1s).
<code>residuals</code>	A vector of residuals from regressing <code>Y</code> on <code>X</code> and controlling for treatment.
<code>es</code>	An effect size value to simulate over.
<code>rho</code>	A rho (correlation) value to simulate over.
<code>my_estimand</code>	"ATE" or "ATT"

### Value

`gen_a_start` returns a list containing the following components:

<code>n1</code>	scalar representing sample size of treatment group ( <code>treat == 1</code> )
<code>ve1</code>	$1 - b1^2$ multiplied by the variance of <code>Ystar1</code>
<code>b1</code>	bounded parameter for treatment group ( <code>treat == 1</code> ) that it with <code>b0</code> are selected to set the correlation of the omitted variable and the outcome equal to <code>rho</code>
<code>es</code>	
<code>pi</code>	proportion of population that is in the treatment group ( <code>treat == 1</code> )
<code>n0</code>	scalar representing sample size of control group ( <code>treat == 0</code> )
<code>ve0</code>	$1 - b0^2$ multiplied by the variance of <code>Ystar0</code>
<code>b0</code>	bounded parameter for control group ( <code>treat == 0</code> ) that it with <code>b1</code> are selected to set the correlation of the omitted variable and the outcome equal to <code>rho</code>
<code>n</code>	scalar representing the total sample size
<code>ind</code>	vector of positions in data that represent treatment group ( <code>treat == 1</code> )
<code>Rstar_1</code>	Residuals in treatment group
<code>Rstar_0</code>	Residuals in control group



**Examples**

```

data(sud)
sud = data.frame(sud)
sud$treat = ifelse(sud$treat == "A", 1, 0)
sud$wts = sample(seq(1, 10, by=.01), size=nrow(sud), replace = TRUE)
outcome_mod = outcome_model(data = sud,
                             weights = "wts",
                             treatment = "treat",
                             outcome = "eps7p_3",
                             model_covariates = c("sfs8p_0"),
                             estimand = "ATE")

start = gen_a_start(y=sud$eps7p_3, tx=sud$treat,
                    residuals=residuals(outcome_mod$mod_results),
                    es = .01,
                    rho = .01,
                    my_estimand = "ATE")

```

---

outcome\_model

*outcome\_model*


---

**Description**

This function will run the outcomes model for your analysis. Upon completion, use the model object returned from this function and call `ov_simgrid` to check the sensitivity of your findings.

**Usage**

```
outcome_model(ps_object = NULL, stop.method=NULL, data, weights=NULL, treatment,
             outcome, model_covariates, estimand = "ATE")
```

**Arguments**

<code>ps_object</code>	A ps object exported from TWANG
<code>stop.method</code>	If the user specifies <code>ps_object</code> , <code>stop.method</code> should be used to export the weights (e.g "ks.max")
<code>data</code>	A data frame containing the data
<code>weights</code>	A column name in data that represents the relevant weights
<code>treatment</code>	A column name in data for the treatment indicator
<code>outcome</code>	A column name in data indicating the outcome vector
<code>model_covariates</code>	A vector of column names representing the covariates in your final outcome's model
<code>estimand</code>	"ATE" or "ATT"

**Value**

outcome\_model returns a list containing the following components:

ps_object	The ps_object from TWANG specified in the function call. If ignored, this component will be NULL
stop.method	The stop method, if applicable, specified in the function call
data	the updated data frame
weights	the original vector of weights
tx	a character name in data indicating the treatment indicator
y	a character name in data indicating the outcome
outcome_mod_fm1a	the final outcome model formula
estimand	The estimand specified in the function call
mod_results	an object of class "svyglm"

**References**

Lumley T (2020). "survey: analysis of complex survey samples." R package version 4.0.

**Examples**

```
data(sud)
sud = data.frame(sud)
sud$treat = ifelse(sud$treat == "A", 1, 0)
sud$wts = sample(seq(1, 10, by=.01), size=nrow(sud), replace = TRUE)
outcome_mod = outcome_model(data = sud,
                             weights = "wts",
                             treatment = "treat",
                             outcome = "eps7p_3",
                             model_covariates = c("sfs8p_0"),
                             estimand = "ATE")
```

---

 ov\_sim

 ov\_sim
 

---

**Description**

This function will create the simulation grid. The simulation will iterate over effects sizes and absolute correlations with the outcome ( $\rho$ ) and see how the treatment effect and relevant p-value changes

**Usage**

```
ov_sim(model_results, plot_covariates, es_grid = seq(-.4, .4, by = 0.05),
       rho_grid = seq(0, .4, by = 0.05), n_reps = 50, progress = TRUE, add = FALSE,
       sim_archive = NULL)
```

**Arguments**

model_results	object returned from outcome_model
plot_covariates	vector of column names representing the covariates that will be plotted on the graphic as observed covariates. Most users will include the variables on the right-hand side of the propensity score model
es_grid	Not required. A grid of effect sizes to simulate over
rho_grid	Not required. A grid of correlations to simulate over; rho relates the correlation to the effect size.
n_reps	Number of repetitions to simulate over
progress	Whether or not the function progress should print to screen. The default value is TRUE. If the user does not want the output to print to screen, they should set to FALSE.
add	Default is FALSE. This is set to true if the user is running additional repetitions after the first call to ov_sim
sim_archive	Default is NULL

**Value**

ov\_sim returns a list containing the following components:

p_val	matrix of pvalues for each grid point
trt_effect	matrix of effect sizes for each grid point
es_grid	vector of the effect size grid
rho_grid	vector of the rho grid
cov	vector of covariates used to estimate propensity score weights
data	the initial data frame containing data with new weights
tx	column name in data representing the treatment indicator
y	column name in data representing the outcome
estimand	estimand used
n_reps	number of repetitions to simulate over
std.error	matrix of standard errors for each grid point
es_se_raw	matrix that stores each repetitions results at every grid point

**Examples**

```
data(sud)
sud = data.frame(sud)
sud$treat = ifelse(sud$treat == "A", 1, 0)
sud$wts = sample(seq(1, 10, by=.01), size=nrow(sud), replace = TRUE)
outcome_mod = outcome_model(data = sud,
                             weights = "wts",
                             treatment = "treat",
                             outcome = "eps7p_3",
```

```

        model_covariates = c("sfs8p_0"),
        estimand = "ATE")

ovtool_results = ov_sim(model_results=outcome_mod,
                        plot_covariates=c("sfs8p_0"),
                        es_grid = NULL,
                        rho_grid = NULL,
                        n_reps = 2,
                        progress=FALSE)

```

---

plot.ov

*plot*


---

## Description

Plots the user specified graphic(s)

## Usage

```

## S3 method for class 'ov'
plot(x, col="color", print_graphic="1", p_contours = c(0.01, 0.05, 0.1), ...)

```

## Arguments

x	Object returned from the call to ov_sim
col	If user wants color or black and white. Specify color with "color" or black and white "bw"
print_graphic	Takes values "1", "2", or "3", depending what graphics the user wants
p_contours	P-value countours to plot. The default plots: 0.01, 0.05, and 0.1. We only recommend changing this if the raw effect p-value is very close to one of these values. Do not specify more than four p-value contours.
...	Additional arguments.

## Value

This function will print the plot to screen that the use specifies with print\_graphic.

## Examples

```

data(sud)
sud = data.frame(sud[sample(1:nrow(sud),100),])
sud$treat = ifelse(sud$treat == "A", 1, 0)
sud$wts = sample(seq(1, 10, by=.01), size=nrow(sud), replace = TRUE)
outcome_mod = outcome_model(data = sud,
                             weights = "wts",
                             treatment = "treat",
                             outcome = "eps7p_6",
                             model_covariates = c("sfs8p_0", "eps7p_0",

```

```

                                "ada_0"),
    estimand = "ATE")

ovtool_results = ov_sim(model_results=outcome_mod,
                        plot_covariates=c("sfs8p_0", "ada_0"),
                        es_grid = 0,
                        rho_grid = 0,
                        n_reps = 2,
                        progress=FALSE)
plot = plot.ov(ovtool_results, print_graphic="3", p_contours=.05)

```

---

```

prep_for_plots      prep_for_plots

```

---

## Description

Data preparation for producing the graphics and summary results.

## Usage

```
prep_for_plots(r1, p_contours)
```

## Arguments

<code>r1</code>	An object returned from <code>ov_sim</code>
<code>p_contours</code>	P-value countours to plot. The default plots: 0.01, 0.05, and 0.1. We only recommend changing this if the raw effect p-value is very close to one of these values. Do not specify more than four p-value contours.

## Value

`prep_for_plots` returns a list containing the following components:

<code>r1</code>	a list with the components returned from <code>ov_simgrid</code>
<code>r1_df</code>	a data frame with components used to create the contour graphic
<code>obs_cors</code>	a data frame with components used to plot the observed covariates on <code>plot_graphic = "2"</code> and <code>plot_graphic = "3"</code>
<code>text_high</code>	a character noting the covariates whose absolute correlation with the outcome is greater than the grid allows
<code>text_high_es</code>	a character noting the covariates with effect sizes greater than the maximum the plot will allow
<code>pvals</code>	a vector of p-value thresholds to be plotted on the graphics
<code>pval_lines</code>	a vector of line types to represent pvals
<code>raw</code>	a character with the raw effect and pvalue from the outcome model

## Examples

```

data(sud)
sud = data.frame(sud[sample(1:nrow(sud),100),])
sud$treat = ifelse(sud$treat == "A", 1, 0)
sud$wts = sample(seq(1, 10, by=.01), size=nrow(sud), replace = TRUE)
outcome_mod = outcome_model(data = sud,
                             weights = "wts",
                             treatment = "treat",
                             outcome = "eps7p_6",
                             model_covariates = c("sfs8p_0", "eps7p_0",
                                                    "ada_0"),
                             estimand = "ATE")

ovtool_results = ov_sim(model_results=outcome_mod,
                        plot_covariates=c("sfs8p_0", "ada_0"),
                        es_grid = 0,
                        rho_grid = 0,
                        n_reps = 2,
                        progress=FALSE)
prep = prep_for_plots(ovtool_results, p_contours=.05)

```

---

sud	<i>Longitudinal observational data from adolescents receiving SUD treatment.</i>
-----	--

---

## Description

A dataset containing substance use disorder and mental health measures for adolescents who had one of two substance use treatments.

## Usage

```
data("sud")
```

## Format

A data frame with 4000 observations on the following 29 variables.

```

treat treatment indicator
tss_0 Traumatic Stress Scale, baseline
tss_3 Traumatic Stress Scale, recorded at 3-months
tss_6 Traumatic Stress Scale, recorded at 6-months
sfs8p_0 Substance Frequency Scale, baseline
sfs8p_3 Substance Frequency Scale, recorded at 3-months
sfs8p_6 Substance Frequency Scale, recorded at 6-months
eps7p_0 Emotional Problems Scale, baseline

```

eps7p\_3 Emotional Problems Scale, recorded at 3-months  
 eps7p\_6 Emotional Problems Scale, recorded at 6-months  
 ias5p\_0 Illegal Activity Scale, baseline  
 dss9\_0 Depressive Symptom Scale-9 Item, baseline  
 mhtrt\_0 MH treatment, past 90 days, baseline  
 sati\_0 Substance Abuse Tx Index, baseline  
 sp\_sm\_0 Substance Problem Scale, Past Month, baseline  
 sp\_sm\_3 Substance Problem Scale, Past Month, recorded at 3-months  
 sp\_sm\_6 Substance Problem Scale, Past Month, recorded at 6-months  
 gvs General Victimization Scale  
 ers21\_0 Environment Risk Scale, baseline  
 nproc Count of Treatment A procedures delivered to client  
 ada\_0 Adjusted Days Abstinent-Any, baseline  
 ada\_3 Adjusted Days Abstinent-Any, recorded at 3-months  
 ada\_6 Adjusted Days Abstinent-Any, recorded at 6-months  
 recov\_0 Binary indicator indicating if in recovery, baseline  
 recov\_3 Binary indicator indicating if in recovery, recorded at 3-months  
 recov\_6 Binary indicator indicating if in recovery, recorded at 6-months  
 subsgtps\_n Categorical variable where: 1="Alcohol and/or marijuana disorder/weekly use; 2="Other drugs"; 3="Opiate disorder/weekly use"  
 sncnt Total number of sessions for Treatment A  
 engage Binary indicator indicating initiated treatment and had 4+ sessions within 45 days for Treatment A

### Source

Global Appraisal of Individual Needs biopsychosocial assessment instrument - GAIN - Dennis, Titus et al. 2003

### Examples

```
data(sud)
```

---

summary.ov

*summary.ov*


---

## Description

Produces summary information that contains a recommendation for reporting the sensitivity analyses

## Usage

```
## S3 method for class 'ov'
summary(object, model_results, sig_level = 0.05, progress = TRUE, ...)
```

## Arguments

object	The object returned from <code>OVtool::ov_simgrid()</code>
model_results	The object returned from <code>OVtool::outcome_model()</code>
sig_level	The alpha level with default 0.05
progress	Whether or not the function progress should print to screen. The default value is TRUE. If the user does not want the output to print to screen, they should set to FALSE.
...	Additional arguments.

## Value

This function will print a recommendation for reporting the sensitivity analyses.

## Examples

```
data(sud)
sud = data.frame(sud)
sud$treat = ifelse(sud$treat == "A", 1, 0)
sud$wts = sample(seq(1, 10, by=.01), size=nrow(sud), replace = TRUE)
outcome_mod = outcome_model(data = sud,
                             weights = "wts",
                             treatment = "treat",
                             outcome = "eps7p_6",
                             model_covariates = c("sfs8p_0", "eps7p_0"),
                             estimand = "ATE")

ovtool_results = ov_sim(model_results=outcome_mod,
                        plot_covariates=c("sfs8p_0"),
                        es_grid = NULL,
                        rho_grid = NULL,
                        n_reps = 2,
                        progress=FALSE)
summary = summary.ov(object = ovtool_results,
                     model_results = outcome_mod,
```



```
sig_level=0.05,  
progress = FALSE)
```

# Index

## \* datasets

sud, [14](#)

add\_pvals\_plot, [2](#)

add\_reps, [3](#)

es\_plot, [4](#)

es\_point\_plot, [5](#)

find\_esgrid, [6](#)

gen\_a\_finish, [7](#)

gen\_a\_start, [8](#)

outcome\_model, [9](#)

ov\_sim, [10](#)

plot.ov, [12](#)

prep\_for\_plots, [13](#)

sud, [14](#)

summary.ov, [16](#)