

How can I create a forest plot in R?

Authored by
stats writer

April 28, 2024

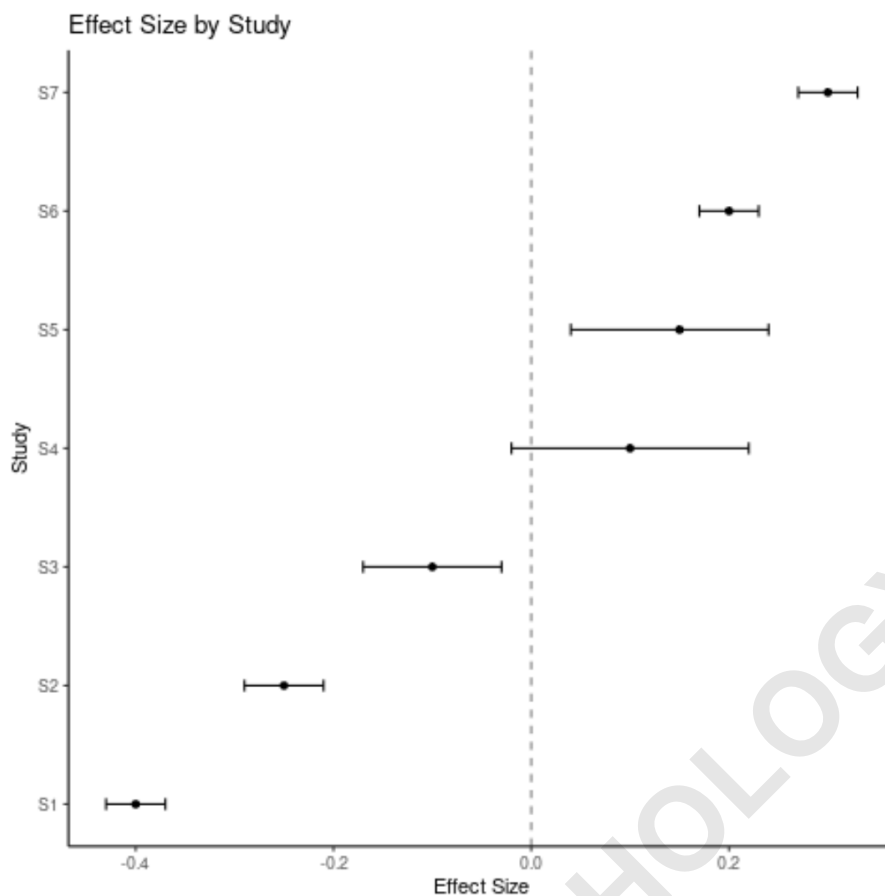
RECOMMENDED CITATION

stats writer (2024). *How can I create a forest plot in R?*. PSYCHOLOGICAL SCALES.
Retrieved from <https://scales.arabpsychology.com/?p=140640>

A forest plot is a graphical representation commonly used in meta-analyses to display the results of multiple studies and their pooled effect estimates. In order to create a forest plot in R, one can use the "metafor" package which allows for the easy creation and customization of forest plots. This package provides functions for inputting the necessary data and producing the plot, as well as options for adding markers, labels, and other features to enhance the visual representation of the data. Additionally, the "ggplot2" package can also be utilized for creating forest plots, providing more advanced options for customization. With the use of these packages, R users can efficiently and effectively create forest plots to visually summarize and interpret the findings of meta-analyses.

Create a Forest Plot in R

A forest plot (sometimes called a "blobbogram") is used in a meta-analysis to visualize the results of several studies in one plot.



The x-axis displays the value of interest in the studies (often an odds ratio, effect size, or mean difference) and the y-axis displays the results from each individual study.

This type of plot offers a convenient way to visualize the results of several studies all at once.

The following example shows how to create a forest plot in R.

Example: Forest Plot in R

To create a forest plot in R, we need to first create a data frame to hold the effect size (or whatever value of interest) and the upper and lower confidence intervals for each study:

```
#create data
```

```
df <- data.frame(study=c('S1', 'S2', 'S3', 'S4', 'S5', 'S6',  
'S7'),  
index=1:7,  
effect=c(-.4, -.25, -.1, .1, .15, .2, .3),  
lower=c(-.43, -.29, -.17, -.02, .04, .17, .27),  
upper=c(-.37, -.21, -.03, .22, .24, .23, .33))
```

```
#view data
```

```
head(df)
```

```
study index effect lower upper
```

```
1 S1 1 -0.40 -0.43 -0.37
```

```
2 S2 2 -0.25 -0.29 -0.21
```

```
3 S3 3 -0.10 -0.17 -0.03
```

```
4 S4 4 0.10 -0.02 0.22
```

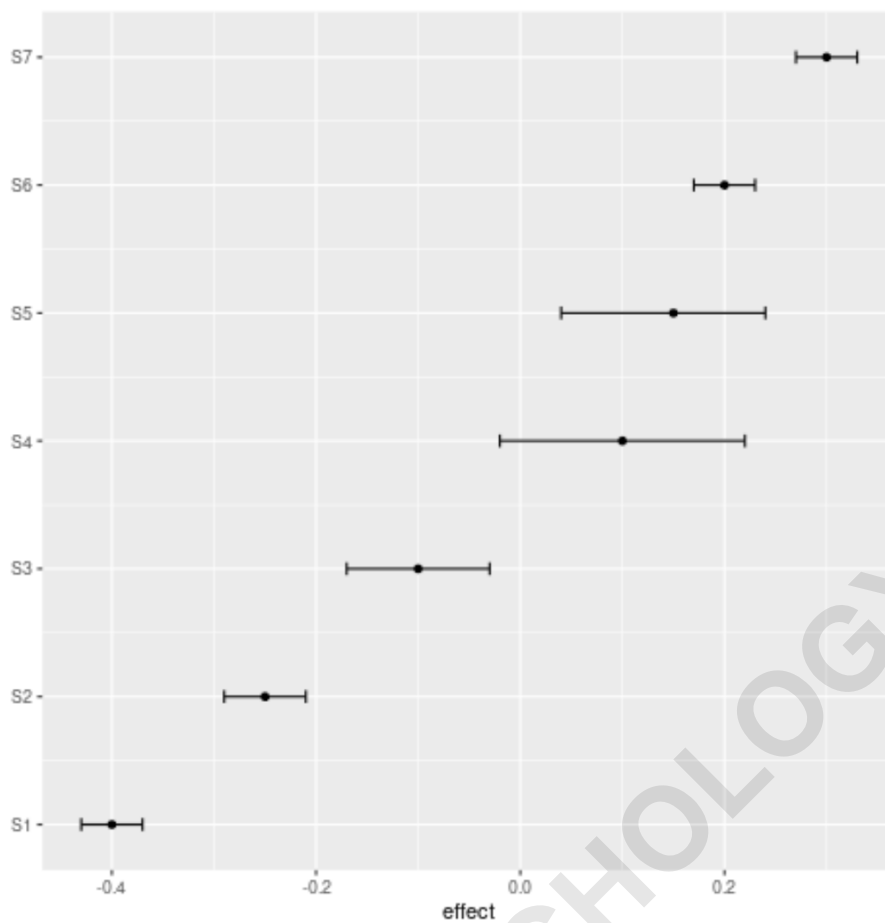
```
5 S5 5 0.15 0.04 0.24
```

```
6 S6 6 0.20 0.17 0.23
```

7 S7 7 0.30 0.27 0.33

Next, we can use functions from the `ggplot2` data visualization package to create the following forest plot:

```
#load ggplot2  
library(ggplot2)  
  
#create forest plot  
ggplot(data=df, aes(y=index, x=effect, xmin=lower,  
xmax=upper)) +  
geom_point() +  
geom_errorbarh(height=.1) +  
scale_y_continuous(name = "", breaks=1:nrow(df),  
labels=df$study)
```

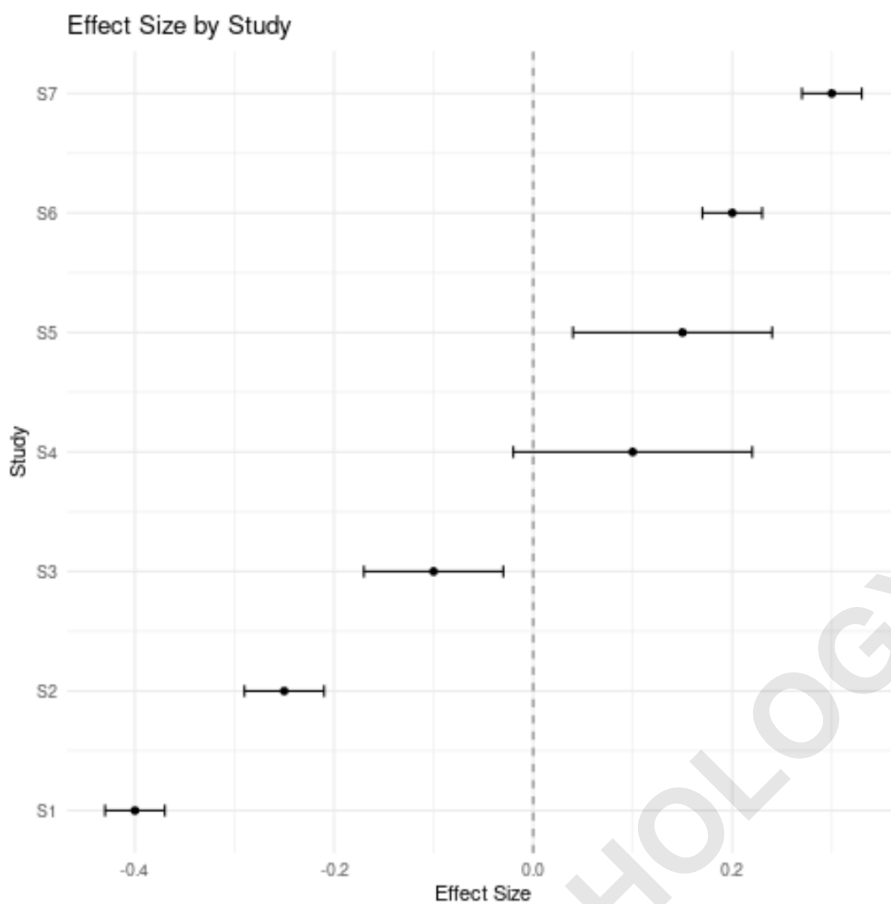


The x-axis displays the effect size for each study and the y-axis displays the name of each study.

The points in the plot displays the effect size for each study and the error bars show the confidence interval bounds.

Note that we can also add a title, modify the axis labels, and add a vertical line at an effect size of zero to make the chart more aesthetically pleasing:

```
#load ggplot2  
library(ggplot2)  
  
#create forest plot  
ggplot(data=df, aes(y=index, x=effect, xmin=lower,  
xmax=upper)) +  
geom_point() +  
geom_errorbarh(height=.1) +  
scale_y_continuous(breaks=1:nrow(df),  
labels=df$study) +  
labs(title='Effect Size by Study', x='Effect Size', y =  
'Study') +  
geom_vline(xintercept=0, color='black',  
linetype='dashed', alpha=.5) +  
theme_minimal()
```



Feel free to modify the theme of the plot to make it look however you'd like. For example, we could also use `theme_classic()` for an even more classic appearance:

```
#load ggplot2
```

```
library(ggplot2)
```

```
#create forest plot
```

```
ggplot(data=df, aes(y=index, x=effect, xmin=lower, xmax=upper)) +
```

```
geom_point() +
```

```
geom_errorbarh(height=.1) +  
scale_y_continuous(breaks=1:nrow(df),  
labels=df$study) +  
labs(title='Effect Size by Study', x='Effect Size', y =  
'Study') +  
geom_vline(xintercept=0, color='black',  
linetype='dashed', alpha=.5) +  
theme_classic()
```

