# The regplane3D package

The regplane3D package is a convenience wrapper for Karline Soetaert’s plot3D package. regplane3D uses several plot3D functions to produce visually appealing three-dimensional displays of regression estimates with confidence intervals. For example, the package can be used to plot conditional expected values of an outcome variable \(Z\) over the joint distribution of two continuous predictors, \(X\) and \(Y\), i.e., \(\mathbb{E}(Z|X,Y)\).

regplane3D (development version 0.1.0) consists of the following functions:

1. plane3D: Plot a three-dimensional regression prediction with confidence intervals.
2. twoplanes3D: Plot a three-dimensional regression prediction with two planes, typically separated at a cut point in one of the two horizontal dimensions, with their respective confidence intervals.
3. heatmap3D: Auxiliary function for adding three-dimensional heatmaps to plots produced by either plane3D or twoplanes3D. These heatmaps show the joint frequency/density distribution of the model predictors represented on the horizontal axes of the plots.
4. pretty\_axis\_inputs: Auxiliary function for generating inputs for prediction and plotting to ensure that the grid lines of the perspective box and the lines of the grid lines of the regression planes match.

**Installation**

To install the latest development version of regplane3D from GitHub, run:

## devtools

if (!("devtools" %in% installed.packages())) install.packages("devtools")

library(devtools)

## regplane3D

if (!("regplane3D" %in% installed.packages())) devtools::install\_github("denis-cohen/regplane3D")

library(regplane3D)

**Prerequisites**

The use of regplane3D functions requires that users provide the following inputs:

1. A vector containing a sequence of values for the first predictor, \(X\).
2. A vector containing a sequence of values for the second predictor, \(Y\).
3. A matrix containing the expected values of \(Z\), or an array containing the expected values as well as their lower and upper confidence interval bounds, for all combinations of the specified values for \(X\) and \(Y\).
4. *Optional*: A matrix containing the discretized joint density or joint frequency of \(X\) and

\(Y\).

We illustrate how these inputs can be generated in some applied examples below. For all

illustrations, we will use the regplane3D’s internal data set us, a small data set containing information on incumbent vote shares, approval ratings, and economic growth rates in US Presidential Elections between 1948 and 2004. For a documentation of the data, see

?regplane3D::us.

# Motivation

A staple of introductory statistics classes is the notion that model predictions from a linear model are no longer represented by line in two dimensions but by a plane in three dimensions once we extend the simple bivariate regression model to include an additional continuous predictor.

Graphical representations of model predictions are widespread for the bivariate case. Most commonly, these appear in the form of conditional expected values of an outcome variable as a function of a key predictor while holding other covariates constant. The Figure below exemplifies this by showing the effect of economic growth on incumbent vote shares in US presidential elections, holding incumbents’ approval rating constant at its sample mean.

R code: Estimation, prediction, and visualization

## ---- Estimation ----

mod <- lm(vote ~ growth + approval, dat = us

## ---- Prediction ----

growth\_vals <- seq(-4, 6, length.out = 101L) pred <- predict.lm(

mod,

newdata = data.frame(growth = growth\_vals,

approval = mean(us$approval)),

se.fit = TRUE

)

## ---- Visualization ----

x\_lines <- seq(-4, 6, 2)

y\_lines <- seq(40, 60, 5)

## Canvas plot(

1,

1,

type = 'n',

main = "Economic Growth and Incumbent Vote Shares", axes = F,

xlab = "Economic Growth",

ylab = "E[Incumbent Vote Share]", ylim = range(y\_lines),

xlim = range(x\_lines)

)

axis(2, outer = FALSE) axis(1)

rect(

min(x\_lines), min(y\_lines), max(x\_lines),

max(y\_lines), col = "gray95", border = NA

)

for (y in y\_lines) segments(min(x\_lines),

y, max(x\_lines), y,

col = "white", lwd = 2)

for (x in x\_lines) segments(x,

min(y\_lines), x, max(y\_lines), col = "white", lwd = 2)

## Prediction polygon(

c(growth\_vals, rev(growth\_vals)), c(

pred$fit + qnorm(.025) \* pred$se.fit, rev(pred$fit + qnorm(.975) \* pred$se.fit)

),

col = adjustcolor("gray30", alpha.f = .2), border = NA

)

lines(growth\_vals,

pred$fit, lty = 1,

col = "gray10", lwd = 2)

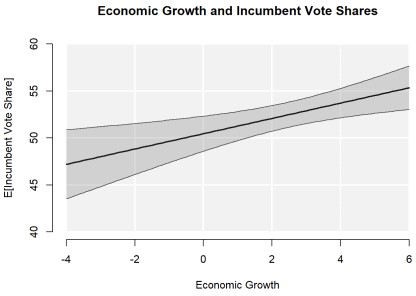
lines(growth\_vals,

pred$fit + qnorm(.025) \* pred$se.fit, lty = 1,

col = "gray30") lines(growth\_vals,

pred$fit + qnorm(.975) \* pred$se.fit, lty = 1,

col = "gray30")



This display of conditional expected values allows readers to grasp at a glance how the expected value of an outcome \(Z\) changes along a range of values \(\overrightarrow{x}\) of a given predictor \(X\). As such, it is both highly informative and easily accessible even to non- technical audiences. Yet, graphically conveying the same information when we want to know how expected values vary as a joint function of two predictors, i.e. \(\mathbb{E}[Z | X, Y]\), remains a difficult enterprise. This is especially true when researchers wish to include inferential uncertainty in the form confidence intervals. Among the most common alternatives, researchers typically

1. separately report marginal effects of \(X\) and \(Y\) on \(Z\), i.e. \(\frac{\partial

\mathbb{E}[Z | X, Y]}{\partial X}\) and \(\frac{\partial \mathbb{E}[Z | X, Y]}{\partial Y}\),

1. selectively report point estimates of conditional expected at characteristic value pairs

\(\{x^\prime, y^\prime\}\) of \(X\) and \(Y\), i.e. \(\mathbb{E}[Z | X = x^\prime, Y = y^\prime]\), or

1. selectively report conditional expected values of \(Z\) along a range values

\(\overrightarrow{x}\) of \(X\) while fixing \(Y\) at some characteristic value(s) \(y^\prime\) (or vice versa), i.e. \(\mathbb{E}[Z|X=\overrightarrow{x}, Y=y^\prime]\).

While all of these quantities of interest yield valuable insights into the model-based relationships between the three variables, none of them allows researchers to grasp how the prediction of

\(\mathbb{E}[Z | X, Y]\) changes over the full grid of plausible value combinations of \(X\) and

\(Y\). This is particularly true if the function that maps \(X\) and \(Y\) onto \(\mathbb{E}[Z | X, Y]\) yields a curved surface.1 In this case, readers cannot interpolate the full structure of

\(\mathbb{E}[Z | X, Y]\) from the selective information included in displays generated akin to

approaches (2) and (3).

regplane3D offers a flexible toolbox that allows users to overcome these limitations. regplane3D can plot any prediction surface over a two-dimensional grid with two predictors with the corresponding confidence or credible intervals. Users must supply these inputs in the form of an array containing the expected values of \(Z\) as well as their lower and upper confidence interval bounds over a grid of pre-specified values of \(X\) and \(Y\). While this means that users must perform the steps of estimation and prediction before using regplane3D, it also offers users full flexibility with respect to model types, quantities of interest, and method of obtaining uncertainty estimates.

# Using regplane3D functions

For the sake of illustration, the following sections will demonstrate the use of regplane3D in the context of conditional expected values from an OLS model with confidence intervals obtained via normal approximation.

**plane3D()**

Our first example uses the regplane3D::plane3D() function to illustrate the us in the context of the OLS regression of incumbent vote shares in US Presidential Elections on incumbent approval ratings and economic growth.

**Definition of axis inputs**

When using regplane3D plotting functions, it is recommended that users use regplane3D::pretty\_axis\_inputs() for defining axis inputs that should be used for *both* prediction *and* plotting. The reason for this requires some explanation. regplane3D plotting functions use plot3D::perspbox() to generate the perspective box inside which the plots are drawn. plot3D::perspbox() in turn depends on graphics::persp(), which uses the base R function base::pretty() to determine the number of ticks and reference lines of the perspective box.

As a result of these dependencies, users have limited control over the exact number and placement of ticks and reference lines. For instance, if users were to provide a variable ranging from 1.89 to 24.31 and request four ticks, this suggestion will be overridden with a rounded value range from 0 to 25 and a total of six ticks in integer steps of 5:

pretty(c(1.89, 24.31), n = 4)

## [1] 0 5 10 15 20 25

Therefore, we recommend that users anticipate this particularity early on and define their axis inputs such that predictions and their visualization eventually conform to the grid lines of the perspective box. The function regplane3D::pretty\_axis\_inputs() performs these tasks. It rounds value ranges to a custom base and provides the number and positions of the grid lines in the perspective box. These can then be used *before* the plot is generated to predict the regression plane at the corresponding values. To provide for smoother curves in plots involving curvilinear planes, the option multiply ensure that values of the plane are not only calculated at the intersections of the grid but at finer gradations.

The example below illustrates the functionality. The function extends the range of the variable us$growth (-3.5969999, 5.914) to a base of 2, such that the coarsened range is (-4, 6). We suggest that this range be split into 7 equally spaced intervals. This is rejected by the function, as such a division would not yield pretty values. Instead, the function returns nlines = 6, which means that the lines should be drawn at the reported linevals of -4, -2, 0, 2, 4, and 6. When plotting non-linear relationships, grid lines evaluated at such few values may look a little jerky. To obtain smoother predictions, we can compute our expected values not just at these coarse values but also at finer gradations in between. To accomplish this, we can for instance specify multiply = 4, which means that the linevals sequence in steps of two will be divided into a finer sequence in steps of 0.5, which is returned as seq.

## Find range of variable growth\_range <- range(us$growth)

growth\_range

## [1] -3.597 5.914

## Determine axis inputs growth\_axis <- pretty\_axis\_inputs(

axis\_range = growth\_range, base = 2,

nlines\_suggest = 7L, multiply = 4

)

growth\_axis

|  |  |  |
| --- | --- | --- |
| ## | $range |  |
| ## | [1] -4 | 6 |
| ## |  |  |
| ## | $seq |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ## | [1] -4.0 -3.5 -3.0 | -2.5 -2.0 -1.5 | -1.0 -0.5 | 0.0 | 0.5 | 1.0 | 1.5 |
| 2.0 | 2.5 3.0 |  |  |  |  |  |  |

## [16] 3.5 4.0 4.5 5.0 5.5 6.0 ##

## $linevals

## [1] -4 -2 0 2 4 6 ##

## $nlines ## [1] 6

**Estimation and prediction**

To obtain the required inputs, we first run a linear regression model of the form \(\texttt{vote} =

\beta\_1 + \beta\_2 \texttt{growth} + \beta\_3 \texttt{approval} +\epsilon\) and save the estimation results to an object named mod.

## ---- Estimation ----

mod <- lm(vote ~ growth + approval, dat = us

We then use regplane3D::pretty\_axis\_inputs() to define the inputs for both axes. This gives us the sequence of values for each growth and approval.

## ---- Axis inputs ----

## Growth

growth\_axis <- pretty\_axis\_inputs( axis\_range = range(us$growth), base = 2,

nlines\_suggest = 6L, multiply = 1

)

## Approval

approval\_axis <- pretty\_axis\_inputs( axis\_range = range(us$approval), base = 10,

nlines\_suggest = 6L, multiply = 1

)

For every combination of the values of these two sequences, we subsequently calculate the expected value and the lower and upper bounds of its 95% confidence interval using the predict.lm() function with option se.fit = TRUE. At each iteration of the nested loop, expected values are temporarily stored in pred\_tmp$fit and standard errors are temporarily stored in pred\_tmp$se.fit. We can extract the expected value and calculate the lower and upper bounds of the 95% confidence interval at each iteration using pred\_tmp$fit + qnorm(.025) \* pred\_tmp$se.fit and pred\_tmp$fit + qnorm(.975) \* pred\_tmp$se.fit, respectively. We subsequently store the estimate of a given iteration in the appropriate cell of the array pred. The array is of dimensions dim = c(length(growth\_axis$seq), length(approval\_axis$seq), 3L). The first dimension represents the values of growth\_axis$seq, the second dimension represents the values of approval\_axis$seq, and the third dimension represents the point estimates, lower confidence bounds, and upper confidence bounds.

## ---- Prediction ----

pred <-

array(NA, dim = c(length(growth\_axis$seq), length(approval\_axis$seq), 3L))

for (growth in seq\_along(growth\_axis$seq)) {

for (approval in seq\_along(approval\_axis$seq)) { pred\_tmp <- predict.lm(

mod,

newdata = data.frame(growth = growth\_axis$seq[growth],

approval = approval\_axis$seq[approval]),

se.fit = TRUE

)

pred[growth, approval, ] <- c( pred\_tmp$fit,

pred\_tmp$fit + qnorm(.025) \* pred\_tmp$se.fit, pred\_tmp$fit + qnorm(.975) \* pred\_tmp$se.fit

)

}

}

*Note:* The prediction step can (and should) be adopted to fit the requirements of a given empirical application. For instance, the calculation of expected values from generalized linear models requires the specification of scenarios for the covariate values, the application of an inverse link function, and the use of bootstrapping or parameter simulation for the construction of confidence intervals (though the last step may be skipped in favor of analytical confidence intervals based on normal approximation if the sampling distribution of the quantity of interest is approximately normal). For more information on the calculation of quantities of interest in generalized linear models, see the Further reading section.

**Plotting**

Using these estimates, we can then plot our regression plane using plane3D(). We pass the inputs z = pred, x = growth\_axis$seq, and y = approval\_axis$seq to the function, which contain all required information to plot the regression plane. The point estimate of the regression line is plotted by default. Confidence intervals are added per the option cis = TRUE. For additional options, see ?regplane3D::plane3D.

## ---- Plot ----

par(mar = c(2.1, 2.1, 4.1, 0.1))

plane3D(

z = pred,

x = growth\_axis$seq,

y = approval\_axis$seq,

zlab = "Predicted Vote Share", xlab = "Economic Growth",

ylab = "Approval Rating", zlim = c(35, 75),

xlim = growth\_axis$range, ylim = approval\_axis$range, cis = TRUE,

xnlines = growth\_axis$nlines, ynlines = approval\_axis$nlines,

main = "Incumbent Vote Shares, Economic \n Growth, and Approval Ratings",

theta = -45,

phi = 9

)



**twoplanes3D()**

The regplane3D::twoplanes3D() function extends the functionality of regplane3D::plane3D() to accommodate two separate planes. These are typically required when the model prediction is distinct to specific value ranges separated by a cut point in one of the two horizontal dimensions, akin to a discontinuity or binary spline.

We showcase the function by replicating the empirical example introduced above, now with distinct predictions for incumbent presidents with above-average and below-average approval ratings, respectively.

**Axis inputs, estimation and prediction**

Axis inputs, estimation and prediction are now slightly more intricate than in the previous example. First, we interact both model predictors with the binary indicator approval\_above\_mean. We then define the axis input for the cut axis (i.e., centered\_approval) from its minimum value up to the cut\_pointof 0. We store the expected values and confidence bounds across the values of growth\_axis$range and approval values ranging from min(centered\_approval) up to the cut point of 0 in pred[,

, 1, ] for the prediction with below-average approval ratings. Analogously, we store this information for the prediction with above-average approval ratings in pred[, , 2, ], where the values of centered\_approval now range from the cut point of up to abs(min(centered\_approval)) to provide for a symmetrical value range and display.

R code: Estimation and prediction

## ---- Estimation ----

mod2 <

lm(vote ~

growth + centered\_approval + approval\_above\_mean +

growth:approval\_above\_mean + centered\_approval:approval\_above\_mean,

dat = us)

## ---- Axis inputs ----

## Cut point cut\_point <- 0

approval\_above\_mean\_vals <- c(0, 1)

## Growth

growth\_axis2 <- pretty\_axis\_inputs( axis\_range = range(us$growth), base = 2,

nlines\_suggest = 6L, multiply = 1

)

## Approval

approval\_axis2 <- pretty\_axis\_inputs(

axis\_range = c(min(us$centered\_approval), cut\_point), base = 10,

nlines\_suggest = 3L, multiply = 1

)

## ---- Prediction ----

pred2 <-

array(NA, dim = c(length(growth\_axis2$seq), length(approval\_axis2$seq), 2L, 3L))

for (growth in seq\_along(growth\_axis2$seq)) {

for (centered\_approval in seq\_along(approval\_axis2$seq)) {

for (approval\_above\_mean in seq\_along(approval\_above\_mean\_vals)) { pred\_tmp <- predict.lm(

mod2,

newdata = data.frame(

growth = growth\_axis2$seq[growth],

centered\_approval = approval\_axis2$seq[centered\_approval] - min(approval\_axis2$seq) \* approval\_above\_mean\_vals[approval\_above\_mean],

approval\_above\_mean = approval\_above\_mean\_vals[ approval\_above\_mean]

),

se.fit = TRUE

)

pred2[growth, centered\_approval, approval\_above\_mean,] <- c(

pred\_tmp$fit,

pred\_tmp$fit + qnorm(.025) \* pred\_tmp$se.fit, pred\_tmp$fit + qnorm(.975) \* pred\_tmp$se.fit

)

}

}

}

**Plotting**

Plotting with regplane3D::twoplanes3D() works the same way as with regplane3D::plane3D(), except we now must provide the \(x\), \(y\) and \(z\) values separately for both planes. For this, we use the inputs x and x2, y and y2, as well as z and z2.

## ---- Plot ----

par(mar = c(2.1, 2.1, 4.1, 0.1))

twoplanes3D(

z = pred2[, , 1,],

x = growth\_axis2$seq, y = approval\_axis2$seq, z2 = pred2[, , 2,],

x2 = growth\_axis2$seq,

y2 = approval\_axis2$seq - min(approval\_axis2$seq), zlim = c(35, 75),

xlim = growth\_axis2$range,

ylim = c(min(approval\_axis2$seq),-min(approval\_axis2$seq)), zlab = "Predicted Vote Share",

xlab = "Economic Growth",

ylab = "Approval Rating \n Above & Below Average", cis = TRUE,

xnlines = growth\_axis2$nlines, ynlines = approval\_axis2$nlines,

main = "Incumbent Vote Shares, Economic \n Growth, and Approval Ratings",

theta = -55,

phi = 9

)



# Extensions

Plot produced with either regplane3D::plane3D() or regplane3D::twoplanes3D() can be extended in numerous ways. For one, we can use regplane3D::heatmap3D() to add a three-dimensional histogram that shows the joint frequency or density distribution of the predictor variables growth and approval. Toward this end, we must first compute a matrix of the joint frequency along discrete intervals of the two continuous predictors. For appealing visuals, it is recommended that the partition of the discrete intervals corresponds to the grid lines of the main plot. These values are returned in the linevals entry of the output returned by regplane3D::pretty\_axis\_inputs().

## Heatmap values

growth\_cat <- cut(us$growth, breaks = growth\_axis$linevals) approval\_cat <- cut(us$approval, breaks = approval\_axis$linevals) joint\_frequency <- table(growth\_cat, approval\_cat)

We can then add the three-dimensional heatmap by adding the option heatmap = joint\_frequency to our regplane3D::plane3D() command:

## ---- Plot ----

par(mar = c(2.1, 2.1, 4.1, 0.1))

par(mar = c(2.1, 2.1, 4.1, 0.1))

plane3D(

z = pred,

x = growth\_axis$seq,

y = approval\_axis$seq,

zlab = "Predicted Vote Share", xlab = "Economic Growth",

ylab = "Approval Rating", zlim = c(35, 75),

xlim = growth\_axis$range, ylim = approval\_axis$range, cis = TRUE,

xnlines = growth\_axis$nlines, ynlines = approval\_axis$nlines,

main = "Incumbent Vote Shares, Economic \n Growth, and Approval Ratings",

theta = -45,

phi = 9,

heatmap = joint\_frequency

)



As the regplane3D package is a convenience wrapper for the plot3D package, plots produced by regplane3D plotting functions can be supplemented with output from plot3D functions (using the option add = TRUE). For instance, we can add the observed values of the outcome variable using plot3D::points3D() and add text labels using plot3D::text3D().

## ---- Plot ----

par(mar = c(2.1, 2.1, 4.1, 0.1))

plane3D(

z = pred,

x = growth\_axis$seq,

y = approval\_axis$seq,

zlab = "Predicted Vote Share", xlab = "Economic Growth",

ylab = "Approval Rating", zlim = c(35, 75),

xlim = growth\_axis$range, ylim = approval\_axis$range,

cis = TRUE,

xnlines = growth\_axis$nlines, ynlines = approval\_axis$nlines,

main = "Incumbent Vote Shares, Economic \n Growth, and Approval Ratings",

theta = -45,

phi = 9,

heatmap = joint\_frequency

)

plot3D::points3D( z = us$vote,

x = us$growth,

y = us$approval, add = TRUE,

col = adjustcolor("black", alpha.f = .3), pch = 19

)

plot3D::text3D(

z = us$vote + 2.5, x = us$growth,

y = us$approval, labels = us$incumbent, add = TRUE,

cex = 0.6

)



# Conclusion

An increasing number of empirical applications in quantitative social science focuses on the interplay of two predictors in determining the expected levels of an outcome. Making sense of such analyses requires interpreting the expected values of an outcome variable over the joint distribution of two predictors. Existing visualizations, however, are typically limited to bivariate displays which show the expected values of the outcome variable as a function of a single predictor, fixing the respective other predictor at some characteristic value and holding background covariates constant. To overcome this limitation, this blog post has introduced the regplane3D package and showcased its functionality. Practitioners can now use this tool to produce visually appealing three-dimensional displays of regression estimates with confidence intervals.