Plotting model residuals with \texttt{plotres}

Stephen Milborrow

October 25, 2019

Contents

1 Introduction 1
2 Some examples 4
3 Limitations 5
4 Generating the residuals and calling predict 5
5 The which=1 plot and the \texttt{“w1.”} prefix 5
6 Notes on \texttt{glmnet} and \texttt{glmnetUtils} models 7
7 Notes on \texttt{gbm} models 9
8 Comparison to \texttt{plot.lm} 10
9 FAQ and common error messages 10

1 Introduction

Residual plots are important for checking linear models, but they also are useful for other types of model. They can be used for an overview of the model’s performance, to check for outliers, and to check if the response should be transformed.

The \texttt{plotres} function in the \texttt{plotmo} R package \cite{plotmo} makes it easy to plot residuals for a wide variety of R models. Figure 1 shows an example. It was produced with the following code:

\begin{verbatim}
library(rpart); library(plotmo) # plotres is in the plotmo package
library(earth); data(ozone1) # get the ozone data
rpart.mod <- rpart(O3 ~ ., data = ozone1) # generate an rpart model
plotres(rpart.mod) # plot its residuals
\end{verbatim}
Since this is an \texttt{rpart} model \cite{rpart}, plotres draws the model tree at the top left \cite{plotres}. Different figures will be drawn in the top left for other types of model (Section 5).

The bottom left plot is a standard Residuals vs Fitted plot of the training data. In this example we see the quantized fits characteristic of \texttt{rpart} models. Each vertical line of points corresponds to a leaf of the tree. (We could use plotres’s \texttt{jitter} argument to jitter the points to reduce overplotting caused by quantization here.) The red line is a loess smooth.

In case 220, for example, we see that although the model estimates a high ozone level (the fitted value is 23), the observed level was actually a lot higher. From the QQ plot (bottom right) we see that this residual is indeed a bit unusual. Since it’s the biggest absolute residual, it determines the right bound of the Cumulative Distribution plot (top right).

A variety of other residual plots can be generated by plotres as described on its help page.

Figure 1: An example plotres plot. This example is for an \texttt{rpart} model.
Residual plots reveal an often-overlooked issue in the popular Boston housing dataset [3].

We build a model which estimates house prices given various characteristics of the neighborhood. The left side of Figure 2 shows the model residuals i.e. the difference between the predicted and actual house prices in the training data. This example uses a gbm model, although any popular non-parametric modeling technique could be used:

```r
data(Boston, package="MASS") # boston housing data
library(gbm)
mod <- gbm(medv ~ ., data=Boston, n.trees=5000, interact=4)
library(plotmo)
plotres(mod, which=3) # left figure (which=3 for residuals plot)
```

In the residuals plot the largest residual is for case 373, and we see an artificial line which starts at this residual and extends diagonally downwards to the right. \(^1\) Investigation of these cases reveals that there are 16 houses with a price of 50 (right of Figure 2). It’s likely that this price is artificial, i.e., the maximum price of a house was clipped to 50 (in units of $1000, the data is from a 1970 US census).

If we remove these cases and rebuild the model we see a more healthy residual plot (middle of Figure 2):

```r
ok <- which(Boston$medv < 50)
mod <- gbm(medv ~ ., data=Boston[ok,], n.trees=5000, interact=4)
plotres(mod, which=3) # middle figure
```

This isn’t to say that these artificial cases should be removed, only that we should be aware of them when deciding how to model the data. The well-known StatLib California housing data (e.g. [4] Section 10.14) exhibits the same issue in a more pronounced form (965 of the house values are clipped at $500001).

\(^1\)Such lines are common for quantized responses, but not for continuous values like house prices.
2 Some examples

Here are some examples which illustrate plotres on various models.

```r
library(earth) # for ozone1 data
data(ozone1)

lm.mod <- lm(O3 ~ ., data=ozone1)  # linear model
plotres(lm.mod)

earth.mod <- earth(O3 ~ ., data=ozone1, degree=2)  # earth
plotres(earth.mod)  # equivalent to plot.earth

library(rpart)  # rpart
rpart.mod <- rpart(O3 ~ ., data=ozone1)
plotres(rpart.mod)

library(tree)  # tree
tree.mod <- tree(O3~., data=ozone1)
plotres(tree.mod)

library(randomForest)  # randomForest
set.seed(2015)
rf.mod <- randomForest(O3~., data=ozone1)
plotres(rf.mod)

library(gbm)  # gbm
set.seed(2015)
gbm.mod <- gbm(O3~., data=ozone1, dist="gaussian",
               interact=2, shrink=.01, n.trees=1000)
plotres(gbm.mod)

library(nnet)  # nnet
set.seed(2015)
nnet.mod <- nnet(O3~., data=scale(ozone1), size=2, decay=0.01, trace=FALSE)
plotres(nnet.mod, type="raw")

library(neuralnet)  # neuralnet
set.seed(2015)
nn.mod <- neuralnet(O3~humidity+temp, data=scale(ozone1), hidden=2)
plotres(nn.mod)

library(caret)  # caret
set.seed(2015)
caret.earth.mod <- train(O3~., data=ozone1, method="earth",
                        tuneGrid=data.frame(degree=2, nprune=10))
plotres(caret.earth.mod, type="raw")
```

This definitely isn’t an exhaustive list of models supported by plotres. The packages used in the above code are \([2,5,6,11–15]\).
3 Limitations

Plotres is designed primarily for displaying standard response - fitted residuals for regression models with a single continuous response. For some model types it supports multiple responses and other kinds of residuals.

In general, it won’t work on models that don’t save the call or data with the model in a standard way. For further discussion please see Accessing the model data in the plotmo vignette.

4 Generating the residuals and calling predict

Plotres first tries to get the residuals by calling the residuals method for the model. If the call fails (which it will for models that don’t have a residuals method), plotres must figure out the residuals manually. It does that using predict.

Plotres tries to use sensible default arguments for predict, but they won’t always be correct (plotres can’t know about the predict method for every kind of model). Change the defaults if necessary using plotres arguments with a predict prefix. Plotres passes any argument prefixed with predict directly to predict, after removing the prefix.

For example, predict.gbm has a n.trees argument, which plotres defaults to the total number of trees. But that can be changed, for example:

```r
library(gbm); library(plotmo)
exmple(gbm) # create gbm1, a gbm model
plotres(gbm1) # call predict.gbm with total n.trees
nbest <- gbm.perf(gbm1, method = "OOB", plot.it = FALSE) # get "best" number of trees
plotres(gbm1, predict.n.trees = nbest) # pass n.trees = nbest to predict.gbm
```

Use trace = 1 with plotres to see the arguments passed to predict and friends.

5 The which=1 plot and the “w1.” prefix

The top left plot is a model-specific plot. We call this plot the “which=1” plot. What gets plotted here depends on the model class. For example, for earth models this is a model selection plot and for glmnet models it’s a coefficient profile plot.

Nothing will be displayed for some models. This isn’t really an issue. You will see three instead of four plots when you call plotres with the default arguments.

For some models, the which=1 plot is called with default arguments programmed into plotres. Use trace = 1 to see those arguments.

---

2This plot is included by default, since by default which = 1:4.
Change the arguments passed to this plot by using plotres arguments with a \texttt{w1} prefix. Plotres passes any argument prefixed with \texttt{w1} directly to the \texttt{which=1} plot, after removing the prefix. For example, \texttt{w1 ylim=c(-10,20)}. Note the \texttt{w1} prefix; plain \texttt{ylim=c(-10,20)} affects the residual plots, not this plot. See also the example below. It’s very easy to forget the \texttt{w1} prefix.

It may happen that the \texttt{which=1} plot is only partially plotted, or messes up the page for further plots. For example, plots that use grid graphics will cause problems. In that case, call plotres with a \texttt{which} argument that excludes 1 (don’t use the default \texttt{which}). We would be interested in hearing about models that cause this kind of bad behavior.

**An example**

Here’s a \texttt{which=1} plot showing \texttt{glmnet} taming the \texttt{longley} data (Figure 3):

```r
library(glmnet); library(plotmo); data(longley)
mod <- glmnet(data.matrix(longley[,1:6]), longley[,7])
plotres(mod, which=1) # left side of the figure
```

Change the parameters passed to the plot using the \texttt{w1} prefix:

```r
plotres(mod, which=1,
    w1.xvar="norm",
    w1.col=1:3) # right side of the figure
```

![Figure 3: A \texttt{glmnet [1]} model of the \texttt{longley} data [7]. Just the \texttt{which=1} plot is shown. Left plotres(glmnet.mod) Right plotres(glmnet.mod, w1.xvar="norm", w1.col=1:3)](image)
6 Notes on glmnet and glmnetUtils models

The following code plots an example glmnet model (Figure 4). The previous page also has some examples.

```r
library(glmnet)
library(plotmo)
x <- matrix(rnorm(100 * 10), 100, 10) # n=100 p=10
y <- x[,1] + x[,2] + 3 * rnorm(100) # y depends only on x[,1] and x[,2]
mod <- glmnet(x, y)
plotres(mod)
```

For the which=1 plot (Figure 4 top left), plotres makes an internal call to plot.glmnet (in the plotmo package). This function is much like plot.glmnet in the glmnet package. It has an extra xvar option "rlambda", which plotres uses by default. (This allows s to be plotted and variable names on the right of the plot.) Override that default by passing say w1.xvar="norm" to plotres.

The plot is annotated with a vertical gray line showing the s parameter. This is the penalty lambda that plotres passes internally to predict.glmnet when generating the residuals for the residual plots (Section 4). By default, plotres calls predict.glmnet with type="response" and s=0 (no penalty). Change that by passing say predict.s=.02 to plotres. (In fact, you can pass any argument to predict.glmnet by prefixing the argument with predict, as described in Section 4.)

When using plotres, arguments such as ylim can be passed on to plot.glmnet using a w1. prefix as described in Section 5, for example w1.ylim=c(-3,3). When calling plot.glmnet directly, don’t use the w1. prefix.

Use w1.s.col=0 to remove the s vertical gray line, or w1.s.col="green" to make it green (for example).

Plotres can also be used with cv.glmnet models (it will invoke plot.cv.glmnet for the which=1 plot).

For multiple response models, use plotres’s nresponse argument to select which response is plotted. The type.coef argument of plot.multnet isn’t supported.
In this example, all variables except V1 and V2 are noise variables.

To calculate the residuals, \( s=0 \) was passed to `predict.glmnet`, as indicated by the vertical gray line in the top left plot.

The \( x \) matrix has no column names, so the variables have been automatically named with a V prefix. The variable names are spread out to prevent overplotting, but we can still make out which curve is associated with which variable.
7 Notes on gbm models

Plotres uses the `plot.gbm` function (in the `plotmo` package) for plotting `gbm` models. Figure 5 shows an example, generated with the following code. (Only the `which=1` `plot.gbm` plot is shown; the residual plots are the same as for other models.)

```r
library(earth); data(ozone1) # get the ozone data
oz <- ozone1[sample.int(n=nrow(ozone1)),] # random row order for train.fraction
gbm.mod <- gbm(O3~., data=oz, dist="gaussian", interaction.depth=2,
                shrinkage=.01, train.fraction=.8, cv.folds=10, n.trees=3000)
plotres(gbm.mod)
```

The vertical dotted lines and the corresponding colored numbers along the top of the plot show the number of trees selected by various criteria.

The OOB curve is the out-of-bag improvement, similar to the `gbm.perf` plot. The OOB curve is on a different scale to the other curves, so shouldn’t really be shown on the same plot. We force it in by rescaling and shifting it. Although helpful for comparing model-selection criteria, this can also lead to confusion: the scale on the left doesn’t apply to the OOB curve. A dashed curve is used as a reminder.

The vertical gray line shows the `n.trees` parameter. This is the number of trees that plotres passes internally to `predict.gbm` when generating the residuals (Section 4), although the residual plots aren’t shown here. In the residual plots, only `train.fraction` of the residuals will be shown if `train.fraction` was specified in the original call to `gbm`.

When generating the residuals, by default plotres calls `predict.gbm` with `type="response"` and `n.trees` set to all the trees. Change that by passing say `predict.n.trees=200` to plotres. (In fact, you can pass any argument to `predict.gbm` by prefixing the argument with `predict.` as described in Section 4.)

When using plotres, arguments such as `ylim` can be passed on to `plot.gbm` using a `w1.` prefix as described in Section 5, for example `w1.ylim=c(-3,3)`. When calling `plot.gbm` directly, don’t use the `w1.` prefix.

![Figure 5: A gbm model.](image)

Just the plotres `which=1` plot is shown (same as calling `plot.gbm` directly).

The plot is a little crowded because both the `train.fraction` and `cv.folds` arguments were used in the call to `gbm` when building the model.
8 Comparison to plot.lm

The function `plot.lm` automatically standardizes residuals for some of the plots. In contrast, with plotres we must explicitly specify when the residuals should be standardized. This is because standardization isn’t possible or appropriate for most of the models that plotres is designed for.

The different `which` numbering scheme used by plotres is mostly an historical legacy (from the days when this function was part of the `earth` package).

As a somewhat academic exercise, here’s how `plot.lm` can be emulated with plotres (Figure 9):

```r
plotlm <- function(object) { # similar to plot.lm
  plotres(object, which=3, center=FALSE, # residuals vs fitted
          caption=paste(deparse(object$call), collapse=" "))

  plotres(object, which=4, standardize=TRUE) # QQ plot

  plotres(object, which=6, standardize=TRUE, main="Scale-Location")

  plotres(object, which=3, versus=4, standardize=TRUE) # leverage plot
}

fit <- lm(Volume ~ ., data = trees) # simple linear model
par(mfrow = c(2,2), oma = c(0,0,3,0)) # four plots on page, space for caption
plot(fit) # call plot.lm
plotlm(fit) # call our version of plot.lm
```

9 FAQ and common error messages

Please see the FAQ and Common error messages in the plotmo vignette.

See also the Notes on some packages in that vignette.

Most of the difficulties associated with plotting residuals by functions like plotres arise because the model-building function neglected to include some essential fields in the model, for example the call. See the vignette Guidelines for S3 Regression Models [9].

Why is nothing displayed for `which=1`?

See Section 5. For some model classes, nothing will displayed for `which=1`. By default, plotres will plot three instead of four plots. (Package authors may contact the maintainer of plotres to add `which=1` support for their model plot; this is usually a trivial extension to plotres.)
Figure 6: `plot.lm` and `plotres`.

---

**plot.lm**

- **Residuals vs Fitted**
- **Normal Q–Q**
- **Scale–Location**
- **Residuals vs Leverage**

**plot.lm emulated with plotres**

- **Residuals vs Fitted**
- **Standardized Residual QQ**
- **Scale–Location**
- **Residuals vs Leverage**

---

---

---

---

---

---

---

---

---

---

---

---

---

---
References


