# REM Tutorial 3: Regression I - Hedonic – Workbook

Prof. Dr. Kathleen Kürschner Rauck

AT 2023

## Contents

Remember to **organise your workspace** (create a working folder and store the **Lucas_County_data** I uploaded on Canvas), **set up a new RScript**, **set your working directory** and save the R Script in your working directory.

Now we can solve some practical problems provided in *Tutorial 3 - Problem Set*.

## Practical 1: Jarque-Bera Test

### a. Generate a variable for the age of buildings in the Lucas_County_data. Compute and plot the kernel density of variable *age*.

We use again the **Lucas_County_data** to generate a variable for the age of the buildings contained in it. We start by loading the data into data frame **dat1** and determine the most recent year of property construction in the data, using the `max()` function:

```
# load data
dat1 <- read.table("Lucas_County_data.txt", header = TRUE)
max(dat1$yrbuilt) # displays maximum building year
```

```
## [1] 1998
```

The most recent year of construction is 1998. Here, we use this year as point of reference in time to calculate variable *age* in relation to the most recently constructed buildings.

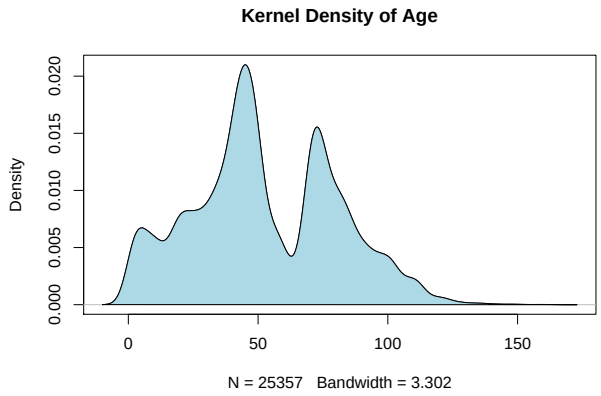
```
dat1$age <- 1998 - dat1$yrbuilt # adds a column for age to the data frame
summary(dat1$age) # summarises variable age
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.00   34.00   48.00   52.66   74.00  163.00
```

The average building age is 52.66 years and ranges from 0 to 163 years.

We next want to take a look at the density function of variable *age*. To do this, we store the results from the function `density()`, which computes estimates of the kernel density of a specified variable, in R object **d** and plot it as follows:

```
d <- density(dat1$age) # computes kernel density estimates
plot(d, main = "Kernel Density of Age") # plots the kernel density estimates
polygon(d, col = "lightblue") # fills a polygon with colour
```

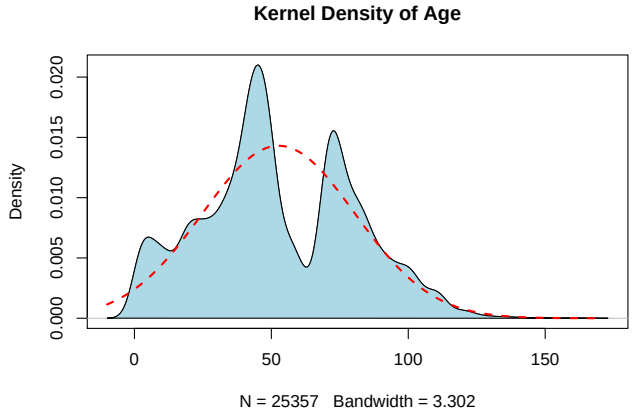



Function `polygon()` allows to fill the area within the boundary of the plotted *d*-line with colour.

### b. Add the density function of the normal distribution to visually inspect if *age* is normally distributed.

We want to check if *age* is normally distributed. Therefore, we want to add a dashed line in red colour to our plot, which depicts the density of a hypothetical normally distributed variable with a mean and standard deviation (sd) that equals the mean and sd of variable *age*, respectively.

```
# add density function of normal distribution
curve(dnorm(x, mean = mean(dat1$age), sd = sd(dat1$age)), from = -10, to = 170,
      add = TRUE, col = "red", lwd = 2, lty = 2)
```




Note that we have used a different function, called `curve()`, to add the dashed line to the existing plot. To add a curve, we specify `add = TRUE` among several other options in the command. `dnorm()` is the probability density function (PDF) of the normal distribution. `lwd` specifies the line width, `col` the line colour and `lty` the line type. We specify line colour "red" and line type "2" to obtain a dashed line in red colour, as stated in the problem set. Visual inspection indicates that the estimated kernel density of *age* hardly coincides with the PDF of the normal distribution.

Side note - similar commands are: `pnorm` for the cumulative distribution function (CDF), `qnorm` for quantiles, and `rnorm` for random numbers.

### c. Run a Jarque-Bera test for normality of variable *age*.

We can also formally test whether variable *age* is normally distributed. There is an R function for the *Jarque-Bera test for normality* of a variable, which requires the `tseries` package to be loaded.

```
# load package
library(tseries)

jarque.bera.test(dat1$age) # runs the test on variable age
```

```
##
##  Jarque Bera Test
##
## data:  dat1$age
## X-squared = 487.57, df = 2, p-value < 2.2e-16
```

We test the null hypothesis $H_0$: "Age is normally distributed". Here the null is rejected. The $\chi^2$-statistic is very large and the corresponding $p$-value is very close to "0".

## Practical 2: The Vintage Effect in Real Estate

### a. *Simple linear regression model:* run an ordinary least squares (OLS) regression of the dependent variable, *price*, on one explanatory variable, *age*, and a constant term. Store the results from this regression in an R object, called fit1.

To run this bivariate linear regression, we can use the `lm()` function in R (lm means linear model). The ~ symbol means "on" as in: we regress the dependent variable (*price*) on the independent variable (*age*). The constant term is automatically included.

```
fit1 <- lm(price ~ age, data = dat1) # lm = linear model (OLS regression)
summary(fit1) # to take a look at the regression output
```

```
## Call:
## lm(formula = price ~ age, data = dat1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -138077  -24792   -6507   14887  732818
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 147374.81     636.36   231.6   <2e-16 ***
## age          -1298.11      10.68  -121.5   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 47420 on 25355 degrees of freedom
## Multiple R-squared:  0.3682, Adjusted R-squared:  0.3681
## F-statistic: 1.477e+04 on 1 and 25355 DF,  p-value: < 2.2e-16
```

The linearity of the model implies that a one-unit change in *age* has the same effect on *price*, regardless of the initial value of *age*, i.e., the partial effect of age on price is a constant (-1.3k dollars price penalty for each additional year of building age). Whether this linearity assumption is appropriate depends on the application at hand. For example, one might want to allow for decreasing or increasing penalties/returns of an additional year of building age, or other types of non-linearities in the relationship between the two variables. We will consider an example, in parts c. and d., below.

Remember that one of the assumptions of the linear regression model is that the error term has a population mean of zero. This assumption is uncontroversial as along as an intercept is included in the model. Why? Because we can always redefine the intercept in our model, such that this assumption is satisfied. Hence, the assumption essentially defines the intercept. Here the intercept is about 147k dollars for a new house ($age = 0$); however, in part b., we will produce a plot from which we can assess that the fit of the price-age relationship is actually not that well.

**b. Produce a scatter plot of *price* against *age*, which shows both the actual data points and the fitted line from the simple regression.**

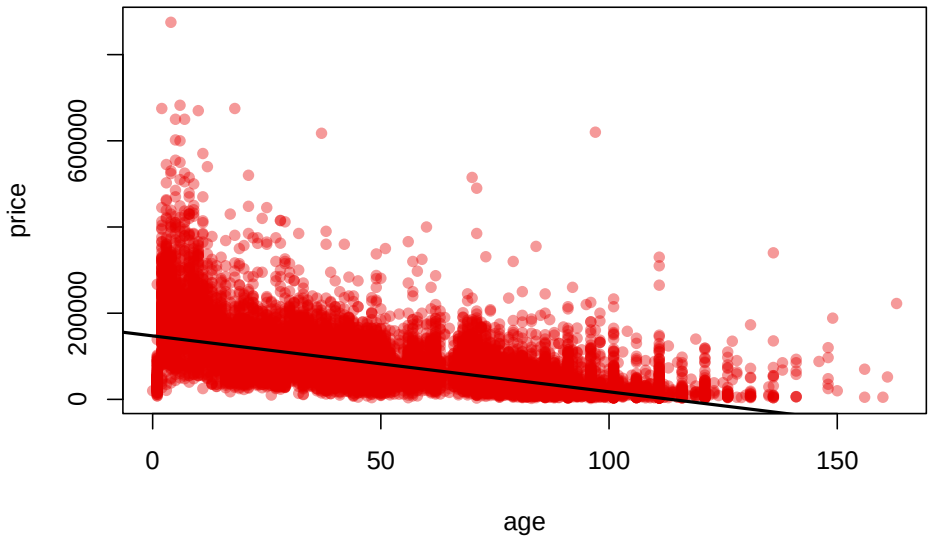
We start by plotting *price* against *age*, using the full estimation sample, i.e., all data contained in **dat1**, and remove the scientific notation from the y-axis.

```
options(scipen = 10) # remove scientific notation
plot(price ~ age, data = dat1, pch = 16, col = rgb(0.9,0,0,0.4))
```

Option `pch` specifies the plotting character to use. For example, `pch = 17` produces a plot with solid triangles (pointing-up) for the data points. See `?pch` for further examples.

Next, we can add a straight line with the estimated slope parameter from the simple regression analysis (in part a.) to the plot. To accomplish this, we can use the function `abline()`, which adds straight lines to a plot using information on intercept and slope that we need to specify. Here it suffices to tell your computer to use the corresponding coefficient estimates from the OLS regression, which we stored in R object **fit1**. To do this, we can simply specify `fit1` instead of manually supplying the values of the estimated parameters for the constant term and slope parameter/coefficient estimate on variable *age*, respectively.

```
abline(fit1, lwd = 2)
```




Two observations from this regression line. It:

- does not account for high prices of new houses
- predicts negative prices for houses older than:

  ```
  -fit1$coefficients[1]/fit1$coefficients[2]
  ```

  ```
  ## (Intercept)
  ##    113.5301
  ```

  114 years.

**c. *Functional form:* extend the model to allow for a potential non-linear relationship between the price and age of a house, including *age* and *age squared* as explanatory variables. Store the results from this regression in an R object, called fit2.**

So far, we have focused on linear relationships between the dependent and independent variable. However, such linear relationships are not general enough for most hedonic house pricing applications. Fortunately, it is easy to incorporate many non-linearities into linear regression analyses by appropriately defining the dependent and independent variables. Common examples of model transformations that often appear in hedonic (house) pricing models, are the log-level (semi-elasticity) and log-log (constant elasticity) model. In both cases, the dependent variable is measured in logarithmic form, i.e., a non-linear transformation of the dependent variable). In the log-level case, i.e., the independent variable in levels (not log-transformed), this gives rise to a constant percentage interpretation of the regression coefficient. For example, in the simple regression of the natural logarithm of price on age, the slope regression coefficient ($\times 100$) gives (approximately) a constant percentage effect of each additional year of building age on property price. In the log-log case, i.e., independent variable is also log-transformed, the interpretation of the slope regression coefficient in our simple regression example would change again, to a constant elasticity effect. That is, the percentage change in house price, associated with a one percent change in housing vintage, or the elasticity of *price* w.r.t. *age*.

Quadratic functions are also quite often applied in (hedonic) regression models to capture decreasing or increasing marginal effects. We start again, by considering the simplest case, in which the dependent variable, *price*, depends on a single observed factor, *age*, but it does so in a quadratic fashion. So, this model falls outside of simple regression analysis but is easily handled with multiple regression. Note, that we model a potential non-linearity in the relationship between the two variables *price* and *age* by inclusion of a second order polynomial in variable *age*, but the regression model is still linear in the parameters we estimate, i.e., *price* is a linear combination of *age* and *age squared*. Thus, we run the same regression as before, adding, however, *age squared* as an explanatory variable. We use again the `lm()` function to apply the OLS estimator.

```
fit2 <- lm(price ~ age + I(age^2), data = dat1)# I() fctn means treat an object 'as is'
summary(fit2)
```

```
## Call:
## lm(formula = price ~ age + I(age^2), data = dat1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -165972  -22290   -6318   15418  708998
##
## Coefficients:
##               Estimate  Std. Error t value Pr(>|t|)
## (Intercept) 176679.5789    942.9412  187.37   <2e-16 ***
## age          -2720.6723     36.1577  -75.25   <2e-16 ***
## I(age^2)        12.8451      0.3128   41.06   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 45920 on 25354 degrees of freedom
## Multiple R-squared:  0.4076, Adjusted R-squared:  0.4075
```

```
## F-statistic:  8721 on 2 and 25354 DF,  p-value: < 2.2e-16
```

The regression results are stored in R object **fit2**. Note that a quadratic relationship indeed prevails between the two variables: the estimated coefficient on age is still negatively signed, respectively, the coefficient estimate on its squared term is positive, and both are highly statistically significant. The fit of the regression model has improved somewhat ($R$-squared increased from 0.37 to 0.41).

### d. Plot the regression results analogously to the ones in part b. Where is the tipping point?

Next, we would like to plot the results from the regression with *age* and *age squared* as explanatory variables. To do this, we step-wise prepare a data frame, called **dat2**, and fill it with the necessary information to predict house prices for a sequence of potential values of property vintage. The goal is to plot:

i. the true data points (observations) on property prices (y-axis) against age (x-axis)

ii. the fitted line, using coefficient estimates from regression model 2 (stored in **fit2**)

**Step 1:** we construct age as a running variable with increments of equal size (to use these later for prediction of prices along the scale of variable *age* in data set, **dat1**) and store this variable as a data frame, called **pred.fit2**.

```
pred.fit2 <- data.frame(age =
                          seq(min(dat1$age), max(dat1$age), # sequence bounds
                                  length.out = nrow(dat1))) # no. of elements
head(pred.fit2) # see increments of 0.0064 yrs
```

```
##           age
## 1 0.000000000
## 2 0.006428459
## 3 0.012856917
## 4 0.019285376
## 5 0.025713835
## 6 0.032142294
```

Within the bounds of the observed data on property age, we have generated a sequence of equal length as the number of rows in **dat1** (note on the `sequence()` function in general: we specify `seq(from = , to = , length of sequence)`).

**Step 2:** add to data frame **pred.fit2** the predicted price data for the values of running variable *age*, we just generated. To do this, we use the `predict()` function, specifying the R object that contains the regression coefficients to be employed for the prediction, `fit2`, and the data set, containing the variable occurrences to use for the prediction, `newdat = pred.fit2`. The last two options tell your computer to calculate prediction (tolerance) intervals at the specified level.

```
pred.fit2[,2:4] <- as.data.frame(predict(fit2, newdat = pred.fit2,
                                         interval = "prediction", level = 0.25))
head(pred.fit2,4)
```

```
##           age      fit      lwr      upr
## 1 0.000000000 176679.6 162044.9 191314.3
## 2 0.006428459 176662.1 162027.4 191296.8
## 3 0.012856917 176644.6 162009.9 191279.3
```

```
## 4 0.019285376 176627.1 161992.4 191261.8
```

**pred.fit2** now contains the running variable *age* within the bounds of the actual data, predicted house prices (variable *fit*) for the values of running variable *age*, using coefficient estimates from regression model 2 (stored in **fit2**), and lower (variable *lwr*)/upper (variable *upr*) bounds of prediction tolerance intervals (here: 25 %).

We can change the name of variable/column *fit* to *pred.price*:

```
colnames(pred.fit2)[2] <- "pred.price"
head(pred.fit2)
```

```
##           age pred.price      lwr      upr
## 1 0.000000000   176679.6 162044.9 191314.3
## 2 0.006428459   176662.1 162027.4 191296.8
## 3 0.012856917   176644.6 162009.9 191279.3
## 4 0.019285376   176627.1 161992.4 191261.8
## 5 0.025713835   176609.6 161974.9 191244.3
## 6 0.032142294   176592.1 161957.4 191226.9
```

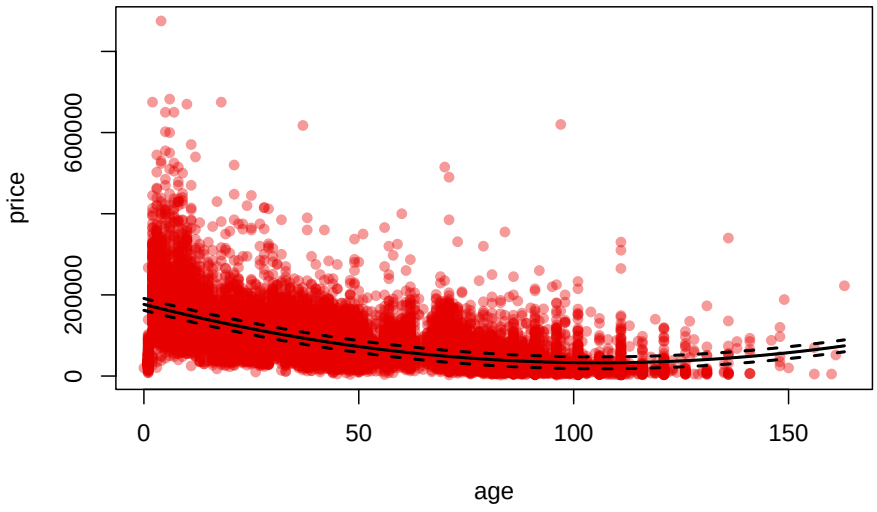
**Step 3:** generate the plot in two further steps.

First, we plot again (from **dat1**) the actual data points, corresponding to house price observations against properties' age.

Second, we consecutively add three lines to the plot, which indicate the fitted/predicted line from the regression along with the lower and upper bounds of the 25 % prediction-tolerance interval.

```
plot(price ~ age, data = dat1, pch = 16, col = rgb(0.9,0,0,0.4)) # same as in b.

# add line plot of predicted price against values of running variable "age"
  using pred.fit2 data
lines(pred.price ~ age, data = pred.fit2, lwd = 2)
# add lower range (prediction interval (alternatively can use CIs))
lines(lwr ~ age, data = pred.fit2, lwd = 2, lty = 2)
# add upper range (prediction interval)
lines(upr ~ age, data = pred.fit2, lwd = 2, lty = 2)
```



**Where is the tipping point?**

Recap: interpretation of coefficients from quadratic equation by taking partial derivative of regression equation w.r.t. *age* $\Rightarrow \beta_1$ (i.e., coefficient on age) $+2\times\beta_2$ (i.e., coefficient on age squared) $\times age$, to obtain the partial effect of *age* on *price* (for infinitesimal changes in *age*). Hence, we can determine the turning (minimum) point at:

```
abs(fit2$coefficients[2]/(2*fit2$coefficients[3]))
```

```
##      age
## 105.9034
```

Careful with the interpretation: *age* still has a discounting effect on *price* (the curve is still below the intercept for all non-zero building ages) but the discount becomes smaller for older houses beyond the 'tipping age' (i.e., the vintage effect of older property, which exhibits often certain period character and features that appeal to many buyers).

# Practical 3: Hedonic Regression Analysis

## a. Hedonic regression analysis involves a multitude of regressors. Add three further hedonic items to the model: the number of bed- and bathrooms (*beds, baths*) as well as information on wall material (*wall*).

Running this multivariate regression works either way using the `lm()` function, but if we want to take a look at the factor levels of factor variables beforehand, we may rather convert character strings such as *wall* to factor variables (see *Tutorial 2*).

```
class(dat1$wall) # format as factor var
```

```
## [1] "character"
```

```
dat1$wall <- as.factor(dat1$wall)
levels(dat1$wall) # to look at the levels
```

```
## [1] "Aluminum, vinyl, or steel siding"
## [2] "Concrete block or tile"
## [3] "Full brick exterior"
## [4] "half or less brick exterior"
## [5] "Horizontal or vertical wood siding or shakes"
## [6] "Stone exterior"
## [7] "Stucco or Dryvit plaster"
```

Run the regression:

```
fit3 <- lm(price ~ age + I(age^2) + beds + baths + wall, data = dat1)
summary(fit3)
```

```
## Call:
## lm(formula = price ~ age + I(age^2) + beds + baths + wall, data = dat1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -286244  -18167   -1999   15017  698264
##
## Coefficients:
##                                                  Estimate Std. Error t value
## (Intercept)                                     20958.6877  1830.2730  11.451
## age                                              -914.5956    36.3388 -25.169
## I(age^2)                                            2.0301     0.2928   6.934
## beds                                            12237.3276   377.1119  32.450
## baths                                           45135.1342   641.4648  70.363
## wallConcrete block or tile                       -199.3360  3436.4583  -0.058
## wallFull brick exterior                         20183.4806   887.3090  22.747
## wallhalf or less brick exterior                 18490.0092   851.8245  21.706
## wallHorizontal or vertical wood siding or shakes -2195.0215   698.7498  -3.141
## wallStone exterior                              13041.6756  4187.9820   3.114
## wallStucco or Dryvit plaster                    15954.4080  2759.0616   5.783
##                                                        Pr(>|t|)
## (Intercept)                                             < 2e-16 ***
## age                                                     < 2e-16 ***
## I(age^2)                                        0.00000000000418 ***
## beds                                                    < 2e-16 ***
## baths                                                   < 2e-16 ***
## wallConcrete block or tile                              0.95374
## wallFull brick exterior                                 < 2e-16 ***
## wallhalf or less brick exterior                         < 2e-16 ***
## wallHorizontal or vertical wood siding or shakes        0.00168 **
## wallStone exterior                                      0.00185 **
## wallStucco or Dryvit plaster                    0.00000000744409 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 38410 on 25346 degrees of freedom
## Multiple R-squared:  0.5855, Adjusted R-squared:  0.5854
## F-statistic:  3580 on 10 and 25346 DF,  p-value: < 2.2e-16
```

Interpretation of:

- categorical variable *wall*: the first level "Aluminum, vinyl, or steel siding" is missing and is the base group, i.e., the other wall types give rise to price premia or discounts relative to this base group.
- intercept: no longer meaningful because this is the estimated price for a new house ($age = 0$) with an "Aluminium, vinyl, or steel siding" but no bed- or bath rooms. There are no observations for this case: to see this you can query whether there is an observation that displays the variable occurrences of interest using the `which()` function.

```
which(dat1$beds == 0, dat1$baths == 0,
dat1$wall ==  "Aluminum, vinyl, or steel siding", dat1$age== 0)
# get an error message that last argument is unused, i.e., there are no obs.
```

More generally you can produce contingency tables to get an overview of the number of observations in certain categories/cells, e.g.:

```
# a contingency table for beds and baths
table(dat1$beds, dat1$baths, dnn = c("Beds","Baths"))
```

```
##     Baths
## Beds     0     1     2     3     4     5     6     7
##    0     1     0     0     0     0     0     0     0
##    1     3   200     3     0     0     0     0     0
##    2     4  5301   192     2     0     0     0     0
##    3     0 12204  2340    60     6     0     0     0
##    4     0  1916  2384   244    15     6     1     1
##    5     0   118   189    75    22     2     3     1
##    6     0    13    24     9     3     2     0     0
##    7     0     3     2     2     2     1     0     0
##    8     0     0     0     1     1     0     0     0
##    9     0     0     0     1     0     0     0     0
```

- Think about/discuss: Why is the effect of *age* now smaller? Which result seems more reliable?

### b. Which price would you predict for a ten year old, three bed-, two bathroom house with full brick exterior?

Example of counterfactual analysis:

Start by writing the variable occurrences into a data frame, called **cf1** and use them for the prediction by specifying `newdat = cf1` in the terms of the `predict()` function. We also specify `fit3` for the regression output, in which your computer should search for the coefficients to be employed in the prediction.

```
cf1 <- data.frame(age = 10, beds = 3, baths = 2, wall = "Full brick exterior")
predict(fit3, newdat = cf1, interval = "prediction", level = 0.25) # roughly $159k
```

```
##        fit      lwr      upr
## 1 159181.5 146937.9 171425.1
```

### c. Can we improve the prediction by adding the total living area (*tla*) to the model? Why is the effect of *beds* now negative?

We add *tla* to the model, run the regression, storing the results in R object **dat4**, and take a look at the output:

```
fit4 <- lm(price ~ age + I(age^2) + beds + baths + wall + tla, data = dat1)
summary(fit4)
```

```
## Call:
## lm(formula = price ~ age + I(age^2) + beds + baths + wall + tla,
##     data = dat1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -420756  -14626     170   14098  601093
##
```

```
## Coefficients:
##                                                  Estimate Std. Error t value
## (Intercept)                                     25668.060   1535.536  16.716
## age                                              -465.410     30.782 -15.120
## I(age^2)                                           -1.609      0.248  -6.488
## beds                                            -6681.823    365.340 -18.289
## baths                                           13575.528    618.453  21.951
## wallConcrete block or tile                      -8685.124   2882.971  -3.013
## wallFull brick exterior                          6850.943    755.179   9.072
## wallhalf or less brick exterior                  4557.590    726.928   6.270
## wallHorizontal or vertical wood siding or shakes -3021.037    586.024  -5.155
## wallStone exterior                               4963.432   3512.896   1.413
## wallStucco or Dryvit plaster                      888.597   2318.321   0.383
## tla                                               632.951      6.120 103.424
##                                                       Pr(>|t|)
## (Intercept)                                            < 2e-16 ***
## age                                                    < 2e-16 ***
## I(age^2)                                       0.0000000000888 ***
## beds                                                   < 2e-16 ***
## baths                                                  < 2e-16 ***
## wallConcrete block or tile                             0.00259 **
## wallFull brick exterior                                < 2e-16 ***
## wallhalf or less brick exterior                0.0000000003677 ***
## wallHorizontal or vertical wood siding or shakes 0.0000002553382 ***
## wallStone exterior                                     0.15769
## wallStucco or Dryvit plaster                           0.70151
## tla                                                    < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 32210 on 25345 degrees of freedom
## Multiple R-squared:  0.7085, Adjusted R-squared:  0.7084
## F-statistic:  5601 on 11 and 25345 DF,  p-value: < 2.2e-16
```

Note that this is a ceteris paribus effect: everything else is held constant, so increasing bedrooms while holding the total living area constant may well have a negative effect.

### d. Which price would you predict for a ten year old, three bed-, two bathroom house with full brick exterior and a total living area of 130 square feet?

Same as b., adding tla of 130 sqft:

```
cf2 <- data.frame(age = 10, beds = 3, baths = 2, wall = "Full brick exterior",
tla = 130)
predict(fit4, newdat = cf2, interval = "prediction", level = 0.25) # roughly $117k
```

```
##        fit    lwr      upr
## 1 117093.2 106825 127361.5
```