SOC-5811 Week 4: Linear regression

Nick Graetz

University of Minnesota, Department of Sociology

9/22/2025

Load data

- R is an **object-oriented** programming language.

LOAD DATA

- R is an **object-oriented** programming language.
- We assign objects with the **<-** operator.

Load data

- R is an **object-oriented** programming language.
- We assign objects with the **<-** operator.
- We apply functions to objects: **function(object)**.

LOAD DATA

- R is an **object-oriented** programming language.
- We assign objects with the **<-** operator.
- We apply functions to objects: **function(object)**.
- All objects have a **class**.

LOAD DATA

- R is an **object-oriented** programming language.
- We assign objects with the **<-** operator.
- We apply functions to objects: **function(object)**.
- All objects have a **class**.
- All functions take inputs of certain classes and return outputs of certain classes.

LOAD DATA

- R is an **object-oriented** programming language.
- We assign objects with the **<-** operator.
- We apply functions to objects: **function(object)**.
- All objects have a **class**.
- All functions take inputs of certain classes and return outputs of certain classes.

- **Scripts are run in computing environments.**

Set up filepaths

- Assign my filepath to an object called "dropbox"
- Test different basic R functions

```r
dropbox <- 'C:/Users/ngraetz/Dropbox/'
class(dropbox)
```

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

```r
length(dropbox)
```

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

```r
nchar(dropbox)
```

```
## [1] 25
```

Set up filepaths

```r
sum(dropbox)
```

```
## Error in sum(dropbox): invalid 'type' (character) of
```

Set up filepaths

- I'm going to set up a few filepaths.

```
fig_dir <- paste0(dropbox,
                  'Minnesota/repos/soc5811/figures/')
data_dir <- paste0(dropbox,
                   'Minnesota/repos/soc5811/data/')
```

Load data

- We are going to look at population and housing data from the 2000/2010 Census.

```
census <- read_dta(paste0(data_dir,
                          'state_pophouse.dta'))
```

Load data

```r
class(census)
```

```
## [1] "tbl_df"     "tbl"        "data.frame"
```

```r
dim(census)
```

```
## [1] 51  9
```

```r
names(census)
```

```
## [1] "state"     "statefp"   "a00aa2000" "a00aa2010" "a41aa2000" "a41aa2010"
## [7] "pctpop"    "pcthouse"  "onepct"
```

```r
head(census)
```

```
## # A tibble: 6 x 9
##   state   statefp a00aa2000 a00aa2010 a41aa2000 a41aa2010 pctpop pcthouse onepct
##   <chr>   <chr>       <dbl>     <dbl>     <dbl>     <dbl>  <dbl>    <dbl>  <dbl>
## 1 Alabama 01        4447100   4779736   1963711   2171853   7.48     10.6      0
## 2 Alaska  02         626932    710231    260978    306967  13.3      17.6      1
## 3 Arizona 04        5130632   6392017   2189189   2844526  24.6      29.9      2
## 4 Arkans~ 05        2673400   2915918   1173043   1316299   9.07     12.2      3
## 5 Califo~ 06       33871648  37253956  12214549  13680081   9.99     12.0      4
## 6 Colora~ 08        4301261   5029196   1808037   2212898  16.9      22.4      5
```

Examine data

We can use different functions like select() and slice() to look at specific rows and columns:

```
census %>%
  select(state,a00aa2000) %>%
  slice(1:5)
```

```
## # A tibble: 5 x 2
##   state      a00aa2000
##   <chr>          <dbl>
## 1 Alabama      4447100
## 2 Alaska        626932
## 3 Arizona      5130632
## 4 Arkansas     2673400
## 5 California  33871648
```

EXAMINE DATA

We can use other packages like "data.table" with different functions:

```
census <- as.data.table(census)
class(census)
```

```
## [1] "data.table" "data.frame"
```

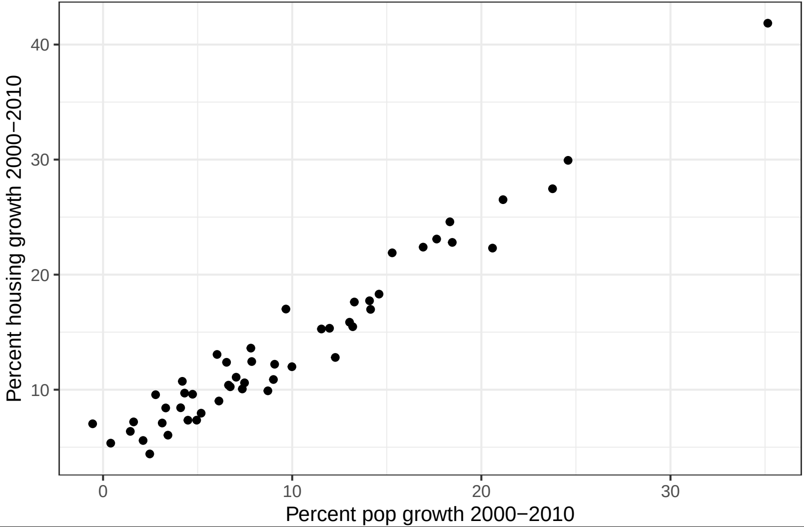
```
census[1:5, c('state','a00aa2000')]
```

```
##         state a00aa2000
##        <char>     <num>
## 1:    Alabama   4447100
## 2:     Alaska    626932
## 3:    Arizona   5130632
## 4:   Arkansas   2673400
## 5: California  33871648
```

Examine data

```
plot <- ggplot(data=census,
               aes(x=pctpop,
                   y=pcthouse)) +
  geom_point() +
  labs(x='Percent pop growth 2000-2010',
       y='Percent housing growth 2000-2010') +
  theme_bw()
```

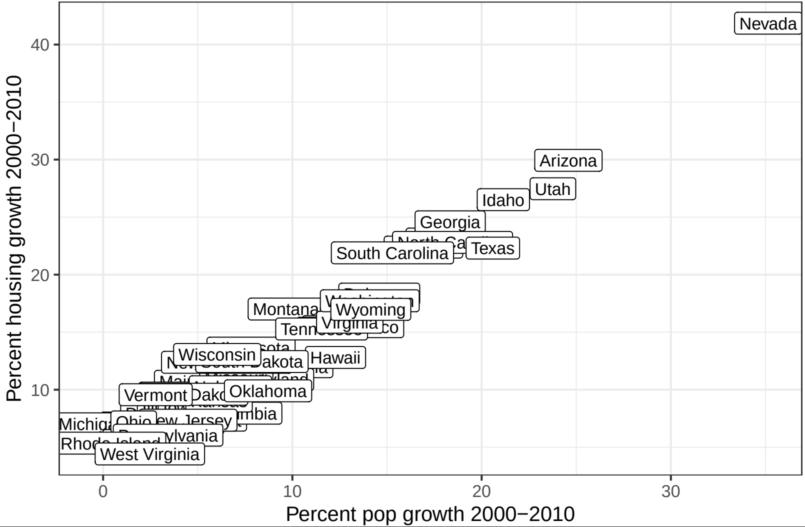
Examine data



Examine data

```r
plot <- ggplot(data=census,
               aes(x=pctpop,
                   y=pcthouse,
                   label=state)) +
  geom_label(size=3) +
  labs(x='Percent pop growth 2000-2010',
       y='Percent housing growth 2000-2010') +
  theme_bw()
```

Examine data



Population regression functions

Let's think about creating a **model** for housing growth:

$$pcthouse = f(pctpop)$$

- What is a model?

Population regression functions

Let's think about creating a **model** for housing growth:

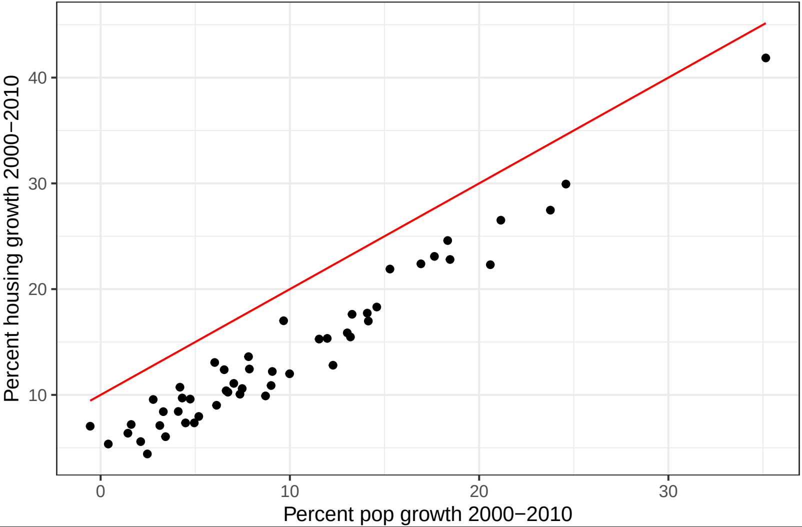
$$pcthouse = 10 + pctpop$$

- Models are defined by *coefficients* (or more generally, *parameters*).

Examine data

```r
census <- census %>%
  mutate(pcthouse_mod1 = 10 + pctpop)
```

Examine data



Population regression functions

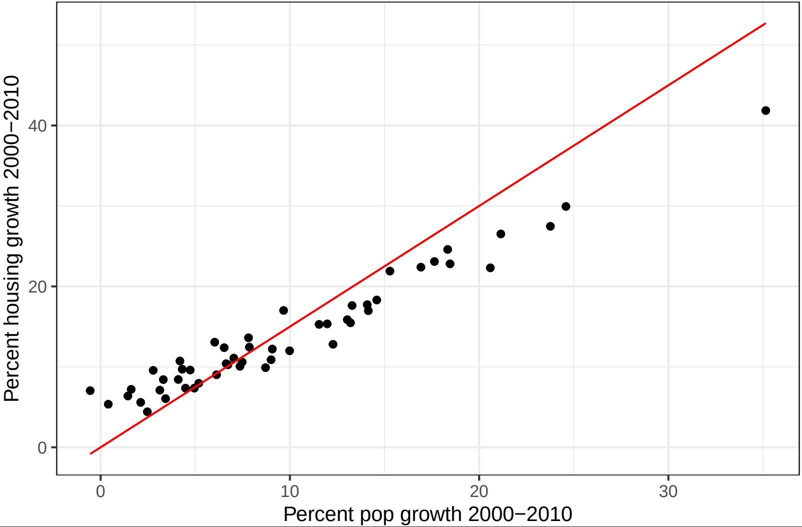
Let's think about creating a **model** for housing growth:

$$pcthouse = 1.5 \times pctpop$$

Examine data

```r
census <- census %>%
  mutate(pcthouse_mod1 = 1.5 * pctpop)
```

Examine data



Population regression functions

- How do I pick a good model?
- What makes a model good?
- What is my goal?

$$pcthouse = f(pctpop)$$

Population regression functions

Let's think about creating a model for housing growth:

$$pcthouse = f(pctpop)$$

$$pcthouse_i = \beta_0 + \beta_1 pctpop_i + \epsilon_i$$

Fitting a linear regression with data:

```r
model <- lm(pcthouse~pctpop,
            data=census)
summary(model)
```

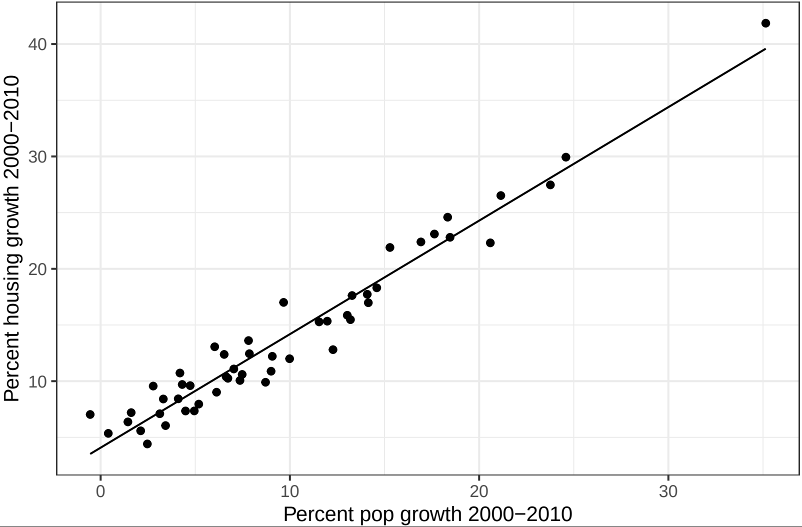
```
##
## Call:
## lm(formula = pcthouse ~ pctpop, data = census)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.6830 -1.3132 -0.1364  1.2039  3.5126
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.08125    0.40793   10.01 1.98e-13 ***
## pctpop       1.01030    0.03371   29.97  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.719 on 49 degrees of freedom
## Multiple R-squared:  0.9483, Adjusted R-squared:  0.9472
## F-statistic: 898.1 on 1 and 49 DF,  p-value: < 2.2e-16
```

Making predictions

```r
census <- census %>%
  mutate(pcthouse_pred=predict(model))
census %>%
  select(state,pctpop,pcthouse,pcthouse_pred) %>%
  head()
```

```
##         state    pctpop pcthouse pcthouse_pred
##        <char>     <num>    <num>         <num>
## 1:    Alabama  7.479841 10.59942      11.63810
## 2:     Alaska 13.286768 17.62179      17.50481
## 3:    Arizona 24.585373 29.93515      28.91974
## 4:   Arkansas  9.071520 12.21234      13.24616
## 5: California  9.985662 11.99825      14.16972
## 6:   Colorado 16.923758 22.39230      21.17924
```

Making predictions



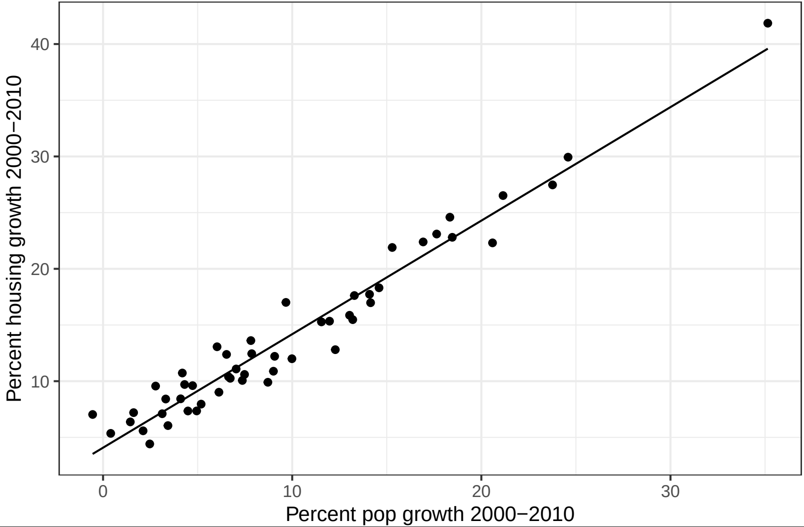
Model coefficients

- What does it mean to "fit" a regression model?
- How did R come up with the coefficients 4.08 and 1.01?

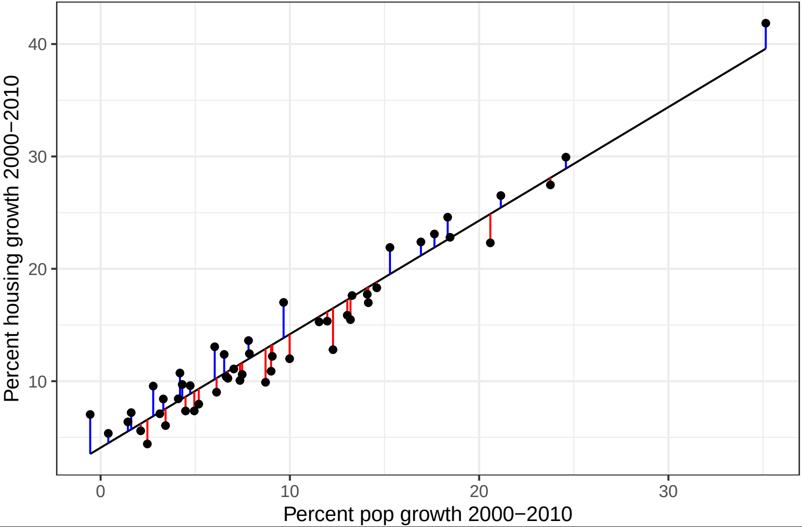
```r
summary(model)
```

```
##
## Call:
## lm(formula = pcthouse ~ pctpop, data = census)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.6830 -1.3132 -0.1364  1.2039  3.5126
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.08125    0.40793   10.01 1.98e-13 ***
## pctpop       1.01030    0.03371   29.97  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.719 on 49 degrees of freedom
## Multiple R-squared:  0.9483, Adjusted R-squared:  0.9472
## F-statistic: 898.1 on 1 and 49 DF,  p-value: < 2.2e-16
```

Making predictions



Making predictions



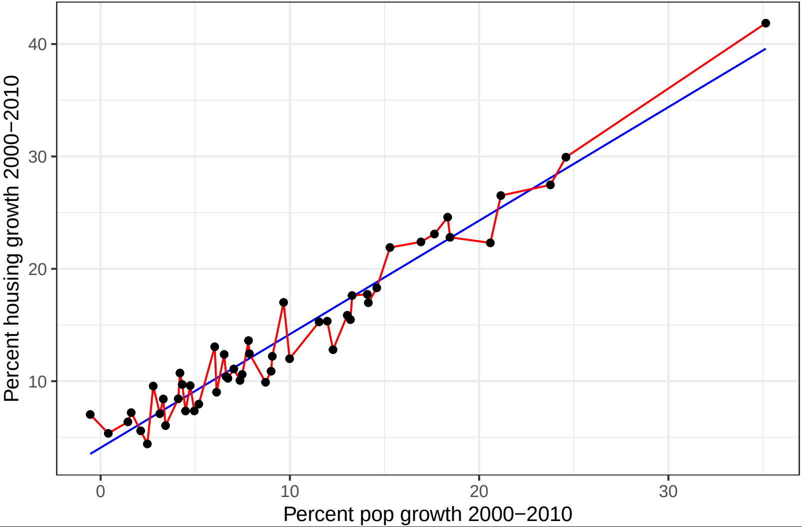
Using models to predict

Just looking *within* my sample... why is my model always wrong?

```
census %>%
  select(state,pctpop,pcthouse,pcthouse_pred) %>%
  head()
```

```
##         state    pctpop pcthouse pcthouse_pred
##        <char>     <num>    <num>         <num>
## 1:    Alabama  7.479841 10.59942      11.63810
## 2:     Alaska 13.286768 17.62179      17.50481
## 3:    Arizona 24.585373 29.93515      28.91974
## 4:   Arkansas  9.071520 12.21234      13.24616
## 5: California  9.985662 11.99825      14.16972
## 6:   Colorado 16.923758 22.39230      21.17924
```

Using models to predict



Coefficients

$$pcthouse_i = \beta_0 + \beta_1 pctpop_i + \epsilon_i$$

$$pcthouse_i = 4.08 + 1.01 pctpop_i + \epsilon_i$$

- Coefficients represent **average comparisons**.

Coefficients

$$pcthouse_i = \beta_0 + \beta_1 pctpop_i + \epsilon_i$$

$$pcthouse_i = 4.08 + 1.01 pctpop_i + \epsilon_i$$

- Coefficients represent **average comparisons**.
- Interpreting the coefficient on *pctpop* (e.g., x):

Coefficients

$$pcthouse_i = \beta_0 + \beta_1 pctpop_i + \epsilon_i$$

$$pcthouse_i = 4.08 + 1.01 pctpop_i + \epsilon_i$$

- Coefficients represent **average comparisons**.
- Interpreting the coefficient on *pctpop* (e.g., x):
  - On average, a 1-point increase in x is associated with a 1.01-point increase in y.

Coefficients

$$pcthouse_i = \beta_0 + \beta_1 pctpop_i + \epsilon_i$$

$$pcthouse_i = 4.08 + 1.01 pctpop_i + \epsilon_i$$

- Coefficients represent **average comparisons**.
- Interpreting the coefficient on *pctpop* (e.g., x):
  - On average, a 1-point increase in x is associated with a 1.01-point increase in y.
  - Across all values of x, the average difference in y at x and x+1 is 1.01.

Coefficients

$$pcthouse_i = \beta_0 + \beta_1 pctpop_i + \epsilon_i$$

$$pcthouse_i = 4.08 + 1.01 pctpop_i + \epsilon_i$$

- Coefficients represent **average comparisons**.
- Interpreting the coefficient on *pctpop* (e.g., x):
  - On average, a 1-point increase in x is associated with a 1.01-point increase in y.
  - Across all values of x, the average difference in y at x and x+1 is 1.01.
  - The slope of the predicted line of y across all values of x is 1.01.

Using models to compare

- Regression is a mathematical tool for making predictions.

Using models to compare

- Regression is a mathematical tool for making predictions.
- Regression coefficients can *sometimes* be interpreted as effects.

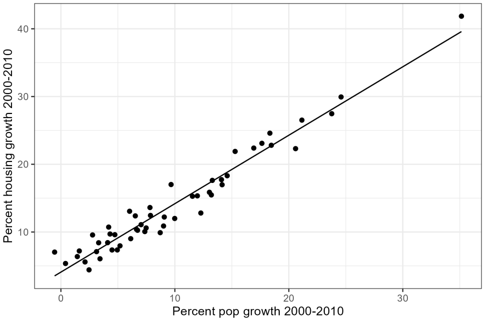
Using models to compare

- Regression is a mathematical tool for making predictions.
- Regression coefficients can *sometimes* be interpreted as effects.
- Regression coefficients can *always* be interpreted as average comparisons.

Building models

What can we do with this model?

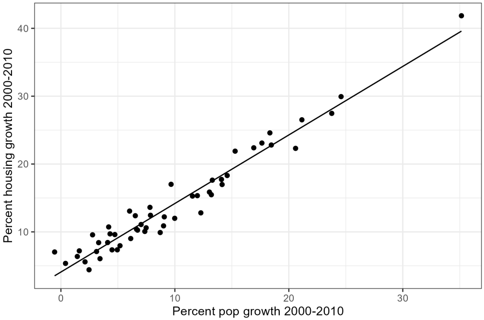
1. Generalizing from sample to population.
2. Measurement.
3. Forecasting.
4. Causal inference.



Building models

What can we do with this model?

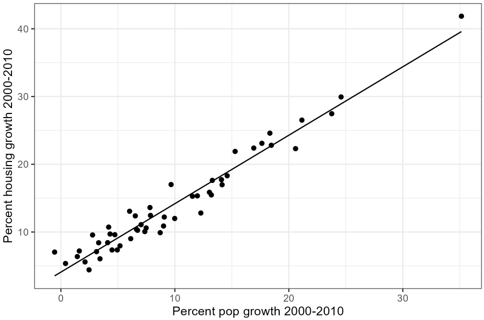
1. **Generalizing from sample to population:** Is this coefficient the same one I would estimate with the entire population?



Building models

What can we do with this model?

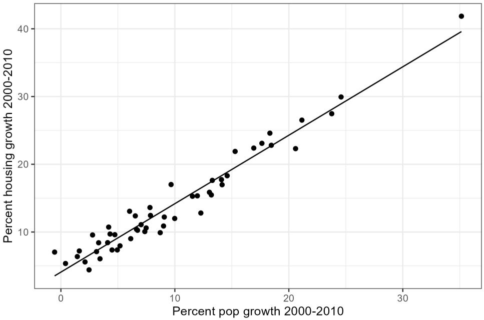
1. Generalizing from sample to population
2. **Measurement:** Can I generalize to all types of housing growth?



Building models

What can we do with this model?

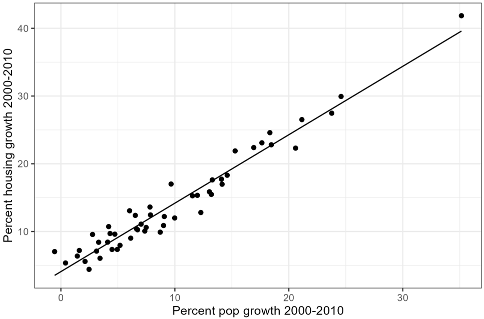
1. Generalizing from sample to population
2. Measurement
3. **Forecasting:** Can I use this model to predict out-of-sample?



Building models

What can we do with this model?

1. Generalizing from sample to population
2. Measurement
3. Forecasting
4. **Causal inference:** Can I say pop growth **causes** housing growth?



Review

- How do we calculate linear regression coefficients?

Review

- How do we calculate linear regression coefficients?
- Minimize the sum of squared residuals.
- This is why linear regression is called Ordinary Least Squares (OLS).

$$\sum_{i=1}^{n}(y_i - \widehat{\beta_0} - \widehat{\beta_1}x_i)^2$$

Review

$$\text{Residual}_i = \text{Observed}_i - \text{Prediction}_i$$

Review

$$\text{Residual}_i = y_i - \text{Prediction}_i$$

Review

$$\text{Residual}_i = y_i - (\widehat{\beta_0} + \widehat{\beta_1} x_i)$$

Review

$$\text{Residual}_i = y_i - \widehat{\beta_0} - \widehat{\beta_1} x_i$$

Review

$$\text{Squared residual}_i = (y_i - \widehat{\beta_0} - \widehat{\beta_1} x_i)^2$$

Review

$$\text{Sum of squared residuals} = \sum_{i=1}^{n}(y_i - \widehat{\beta_0} - \widehat{\beta_1}x_i)^2$$

Review

- What do I need to know to calculate the sum of squared residuals?

$$\text{Sum of squared residuals} = \sum_{i=1}^{n} (y_i - \widehat{\beta_0} - \widehat{\beta_1} x_i)^2$$

Review

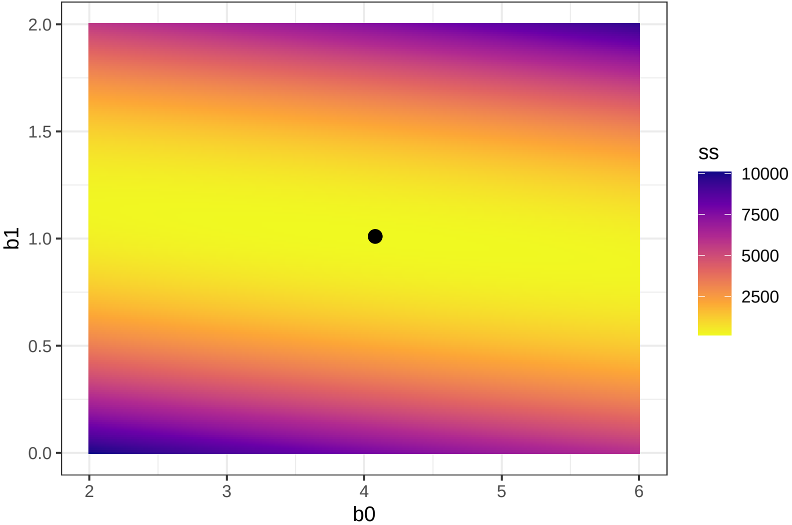
- What do I need to know to calculate the sum of squared residuals?
- For every possible value of the coefficients, there is a single sum of squared residuals.

$$\text{Sum of squared residuals} = \sum_{i=1}^{n}(y_i - \widehat{\beta_0} - \widehat{\beta_1}x_i)^2$$

Review

```
beta_grid <- as.data.table(expand.grid(b0=seq(2,6,0.01),b1=seq(0,2,0.01)))
ss <- function(i) {
  out <- data.table(b0=as.numeric(beta_grid[i,1]),
                    b1=as.numeric(beta_grid[i,2]),
                    ss=census[, sum((pcthouse-
                                    as.numeric(beta_grid[i,1])-
                                    as.numeric(beta_grid[i,2])*pctpop)^2)])
  return(out)
}
out <- rbindlist(lapply(1:nrow(beta_grid), ss))
```

Review



Review

```r
out %>%
  filter(ss==min(ss))
```

```
## Index: <ss>
##        b0    b1       ss
##     <num> <num>    <num>
## 1:   4.08  1.01 144.8543
```

Review

```r
## Outcome
Y <- as.matrix(census$pcthouse)
## Design matrix
X <- cbind(1, census$pctpop)
## Matrix algebra
solve(t(X) %*% X) %*% t(X) %*% Y
```

```
##              [,1]
## [1,] 4.081249
## [2,] 1.010295
```

Review

```r
model <- lm(pcthouse~pctpop,
            data=census)
summary(model)
```

```
##
## Call:
## lm(formula = pcthouse ~ pctpop, data = census)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.6830 -1.3132 -0.1364  1.2039  3.5126
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.08125    0.40793   10.01 1.98e-13 ***
## pctpop       1.01030    0.03371   29.97  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.719 on 49 degrees of freedom
## Multiple R-squared:  0.9483, Adjusted R-squared:  0.9472
## F-statistic: 898.1 on 1 and 49 DF,  p-value: < 2.2e-16
```

How do we create a model based on empirical data?

$$pcthouse = f(pctpop)$$

Review

How do we create a model based on empirical data?

$$pcthouse = f(pctpop)$$

1. Choose a functional form for the model.

$$pcthouse_i = \beta_0 + \beta_1 pctpop_i + \epsilon_i$$

Review

How do we create a model based on empirical data?

$$pcthouse = f(pctpop)$$

1. Choose a functional form for the model.

$$pcthouse_i = \beta_0 + \beta_1 pctpop_i + \epsilon_i$$

2. Choose an estimator (i.e., objective function) that relates the model to real observed data (e.g., OLS).

$$\text{Sum of squared residuals} = \sum_{i=1}^{n} (y_i - \widehat{\beta_0} - \widehat{\beta_1} x_i)^2$$

Review

How do we create a model based on empirical data?

$$pcthouse = f(pctpop)$$

1. Choose a functional form for the model.

$$pcthouse_i = \beta_0 + \beta_1 pctpop_i + \epsilon_i$$

2. Choose an estimator (i.e., objective function) that relates the model to real observed data (e.g., OLS).

$$\text{Sum of squared residuals} = \sum_{i=1}^{n} (y_i - \widehat{\beta_0} - \widehat{\beta_1} x_i)^2$$

3. Fit the model (e.g., estimate parameters) by optimizing the objective function.

Review

How do we create a model based on empirical data?

$$pcthouse = f(pctpop)$$

1. Choose a functional form for the model.

$$pcthouse_i = \beta_0 + \beta_1 pctpop_i + \epsilon_i$$

2. Choose an estimator (i.e., objective function) that relates the model to real observed data (e.g., OLS).

$$\text{Sum of squared residuals} = \sum_{i=1}^{n}(y_i - \widehat{\beta_0} - \widehat{\beta_1}x_i)^2$$

3. I want to maximize the likelihood that my observed data came from one particular model (e.g., set of parameters) relative to all other models.

Review

Why Language Models Hallucinate

Adam Tauman Kalai* — OpenAI
Ofir Nachum — OpenAI
Santosh S. Vempala† — Georgia Tech
Edwin Zhang — OpenAI

September 4, 2025

Abstract

Like students facing hard exam questions, large language models sometimes guess when uncertain, producing plausible yet incorrect statements instead of admitting uncertainty. Such "hallucinations" persist even in state-of-the-art systems and undermine trust. We argue that language models hallucinate because the training and evaluation procedures reward guessing over acknowledging uncertainty, and we analyze the statistical causes of hallucinations in the modern training pipeline. Hallucinations need not be mysterious—they originate simply as errors in binary classification. If incorrect statements cannot be distinguished from facts, then hallucinations in pretrained language models will arise through natural statistical pressures. We then argue that hallucinations persist due to the way most evaluations are graded—language models are optimized to be good test-takers, and guessing when uncertain improves test performance. This "epidemic" of penalizing uncertain responses can only be addressed through a socio-technical mitigation: modifying the scoring of existing benchmarks that are misaligned but dominate leaderboards, rather than introducing additional hallucination evaluations. This change may steer the field toward more trustworthy AI systems.

Properties of a good estimator

- How do we choose an objective function?

PROPERTIES OF A GOOD ESTIMATOR

- How do we choose an objective function?
- **Unbiased**: An unbiased estimator has an expected value equal to the "true" population parameter.

Properties of a good estimator

- How do we choose an objective function?
- **Unbiased**: An unbiased estimator has an expected value equal to the "true" population parameter.
- **Consistency**: A consistent estimator "collapses" around the true value with variance zero as the sample size gets larger and moves towards infinity.

Properties of a good estimator

- How do we choose an objective function?
- **Unbiased**: An unbiased estimator has an expected value equal to the "true" population parameter.
- **Consistency**: A consistent estimator "collapses" around the true value with variance zero as the sample size gets larger and moves towards infinity.
- **Efficiency**: An efficient estimator is one that has a small sampling variance, relative to another estimator.

Properties of a good estimator

- There are entire classes on statistical inference that focus on deriving the proofs for these properties.

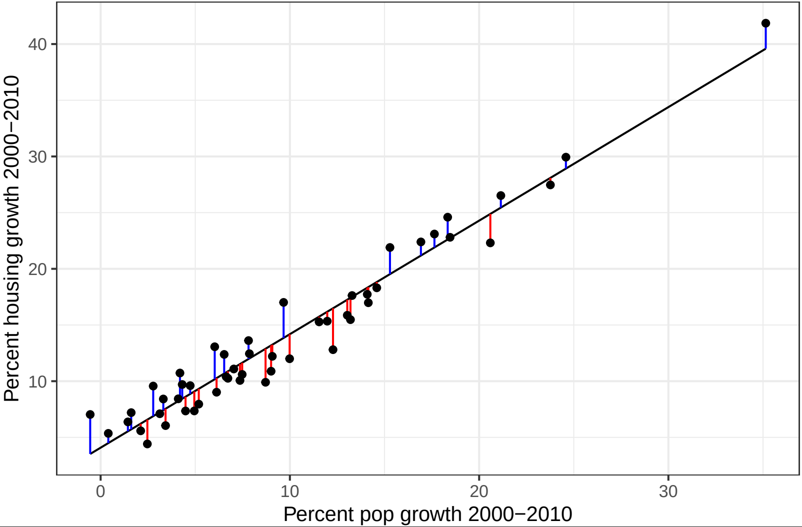
Properties of a good estimator

- There are entire classes on statistical inference that focus on deriving the proofs for these properties.
- The **Gauss-Markov theorem** states that for a linear regression model, the ordinary least squares (OLS) estimator provides the Best Linear Unbiased Estimator (BLUE), meaning it is the most precise (has the minimum variance) among all linear, unbiased estimators. This holds true when the model's error terms are uncorrelated, have equal variances, and a zero expectation, a set of assumptions known as **Gauss-Markov assumptions**.

Properties of a good estimator

- There are entire classes on statistical inference that focus on deriving the proofs for these properties.
- The **Gauss-Markov theorem** states that for a linear regression model, the ordinary least squares (OLS) estimator provides the Best Linear Unbiased Estimator (BLUE), meaning it is the most precise (has the minimum variance) among all linear, unbiased estimators. This holds true when the model's error terms are uncorrelated, have equal variances, and a zero expectation, a set of assumptions known as **Gauss-Markov assumptions**.
- **We will discuss model diagnostics in Week 12.**

Residuals

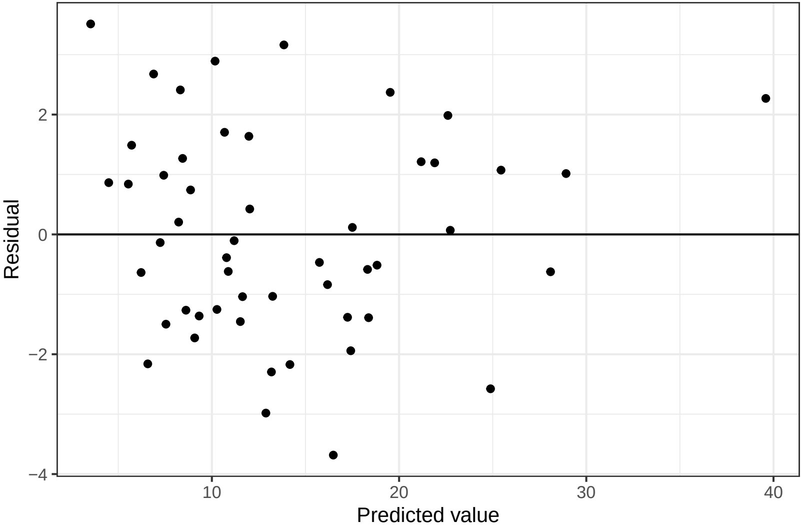


Percent housing growth 2000−2010
40
30
20
10
0
10
20
30
Percent pop growth 2000−2010

Residuals

```
census <- census %>%
  mutate(redisual=pcthouse-pcthouse_pred)

plot <- ggplot(data=census,
               aes(x=pcthouse_pred,
                   y=residual)) +
  geom_point() +
  geom_hline(yintercept=0) +
  labs(x='Predicted value',y='Residual') +
  theme_bw()
```



Residual
2
0
−2
−4
10
20
30
40
Predicted value

Residuals

```
census %>%
  summarize(mean_residual=round(mean(residual),4),
            total_residual=round(sum(residual),4))
```

```
##   mean_residual total_residual
## 1             0              0
```

Properties of OLS

- OLS produces residuals which are uncorrelated with predicted values.
- OLS produces residuals that sum to zero.

Goodness of fit

```r
summary(model)
```

```
##
## Call:
## lm(formula = pcthouse ~ pctpop, data = census)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.6830 -1.3132 -0.1364  1.2039  3.5126
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.08125    0.40793   10.01 1.98e-13 ***
## pctpop       1.01030    0.03371   29.97  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.719 on 49 degrees of freedom
## Multiple R-squared:  0.9483, Adjusted R-squared:  0.9472
## F-statistic: 898.1 on 1 and 49 DF,  p-value: < 2.2e-16
```

Interpreting coefficients

$$pcthouse_i = \beta_0 + \beta_1 pctpop_i + \epsilon_i$$

$$pcthouse_i = 4.08 + 1.01 pctpop_i + \epsilon_i$$

- How do we interpret the **intercept** coefficient?

INTERPRETING COEFFICIENTS

$$pcthouse_i = \beta_0 + \beta_1 pctpop_i + \epsilon_i$$

$$pcthouse_i = 4.08 + 1.01 pctpop_i + \epsilon_i$$

- How do we interpret the **intercept** coefficient?
- How do we interpret the **slope** coefficient?

What if I divide my independent variable by 10?

```
census <- census %>%
  mutate(pctpop_10 = pctpop/10)
```

Interpreting coefficients

What if I divide my independent variable by 10?

```
lm(pcthouse~pctpop_10, data=census)
```

```
##
## Call:
## lm(formula = pcthouse ~ pctpop_10, data = census
##
## Coefficients:
## (Intercept)    pctpop_10
##       4.081       10.103
```

Interpreting coefficients

What if I shift my independent variable by 10?

```
census <- census %>%
  mutate(pctpop_10 = pctpop+10)
lm(pcthouse~pctpop_10, data=census)
```

```
##
## Call:
## lm(formula = pcthouse ~ pctpop_10, data = census
##
## Coefficients:
## (Intercept)    pctpop_10
##      -6.022        1.010
```

Linear transformations of the independent variable

- **Rescaling**

Linear transformations of the independent variable

- **Rescaling**
- **Shifting**

LINEAR TRANSFORMATIONS OF THE INDEPENDENT VARIABLE

- **Rescaling**
- **Shifting**
- Rescaling and shifting are useful for **interpretation** of regression coefficients. For example, say my dependent variable is life expectancy and my independent variable is income. How would I interpret the slope coefficient?

Linear transformations of the independent variable

- **Rescaling**
- **Shifting**
- Rescaling and shifting are useful for **interpretation** of regression coefficients. For example, say my dependent variable is life expectancy and my independent variable is income. How would I interpret the slope coefficient?
- As we will see, they don't fundamentally change properties of the regression model or statistical tests.

Linear transformations of the independent variable

- R-squared is the same.
- p-values are the same.

```
##
## Call:
## lm(formula = pcthouse ~ pctpop_10, data = census)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.6830 -1.3132 -0.1364  1.2039  3.5126
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -6.02170    0.70859  -8.498 3.34e-11 ***
## pctpop_10    1.01030    0.03371  29.968  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.719 on 49 degrees of freedom
## Multiple R-squared:  0.9483, Adjusted R-squared:  0.9472
## F-statistic: 898.1 on 1 and 49 DF,  p-value: < 2.2e-16
```

LINEAR TRANSFORMATIONS OF THE INDEPENDENT VARIABLE

```r
census <- census %>%
  mutate(pctpop_rescaled = scale(pctpop))
summary(lm(pcthouse~pctpop_rescaled, data=census))
```

```
##
## Call:
## lm(formula = pcthouse ~ pctpop_rescaled, data = census)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.6830 -1.3132 -0.1364  1.2039  3.5126
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)      13.9499     0.2408   57.94   <2e-16 ***
## pctpop_rescaled   7.2868     0.2432   29.97   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.719 on 49 degrees of freedom
## Multiple R-squared:  0.9483, Adjusted R-squared:  0.9472
## F-statistic: 898.1 on 1 and 49 DF,  p-value: < 2.2e-16
```