Unit 8a

Agenda

- Unit 8a linear regression
- Unit 8b: start applying what we learned to **explore data**
- Bootstrap methods
- lab unit 8

Last two classes July 30/Aug 6

`Clustering methods`

`Network/Graph theory and applied problems`

`image recognition`

`? Pizza party ?`

Least Squares Linear Regression

The idea is that given n data points

$\{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\}$

we want to **best fit model** the data as lying on some line

$$\hat{y} = \beta_0 + \beta_1 x$$

data points won't lie exactly on a line so we define residuals

$$y_i - \hat{y}_i = \epsilon_i \qquad \text{so} \qquad y = \beta_0 + \beta_1 x + \epsilon$$

$$\text{Data} = \text{Fit} + \text{Residual}$$

least squares regression minimizes sum of squared residuals

Different way of thinking

- You may be used to thinking of x and y as random variables
- here in linear regression, **Data = Fit + Randomness**
- Here, x**, is fixed data (not a random variable)** whereas the residuals are where the randomness lies
- y is a random variable ONLY through ϵ, $\text{Var}(y) = \text{Var}(\epsilon)$
- We assume **normal residuals** $\epsilon_i \sim N(0, \sigma^2)$
- usually we don't know σ^2. It is estimated from data

$$\hat{\sigma}^2 = \frac{1}{n-2}\sum_i (y_i - \hat{y})^2 = \frac{1}{n-2}\sum_i \epsilon_i^2$$

The values β_0, β_1 that minimize the residuals are

$\hat{\beta}_0 = \overline{y} - \hat{\beta}_1 \overline{x}$

$\hat{\beta}_1 = \frac{s_y}{s_x} R = \frac{\text{Cov}(xy)}{s_x^2}$

where the correlation and ***sample*** *covariance* are

$R = \frac{\text{Cov}(xy)}{s_x s_y}$

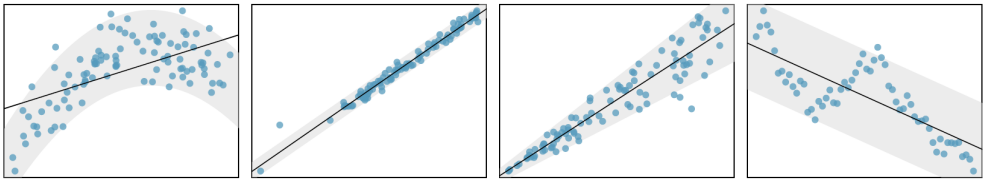
$\text{Cov}(xy) = \frac{1}{n-1}\sum_i (x_i - \overline{x})(y_i - \overline{y})$

Correlation, R, measures linear relationship on scale -1 to +1

correlation coefficient, R^2 instead of R sometimes also used

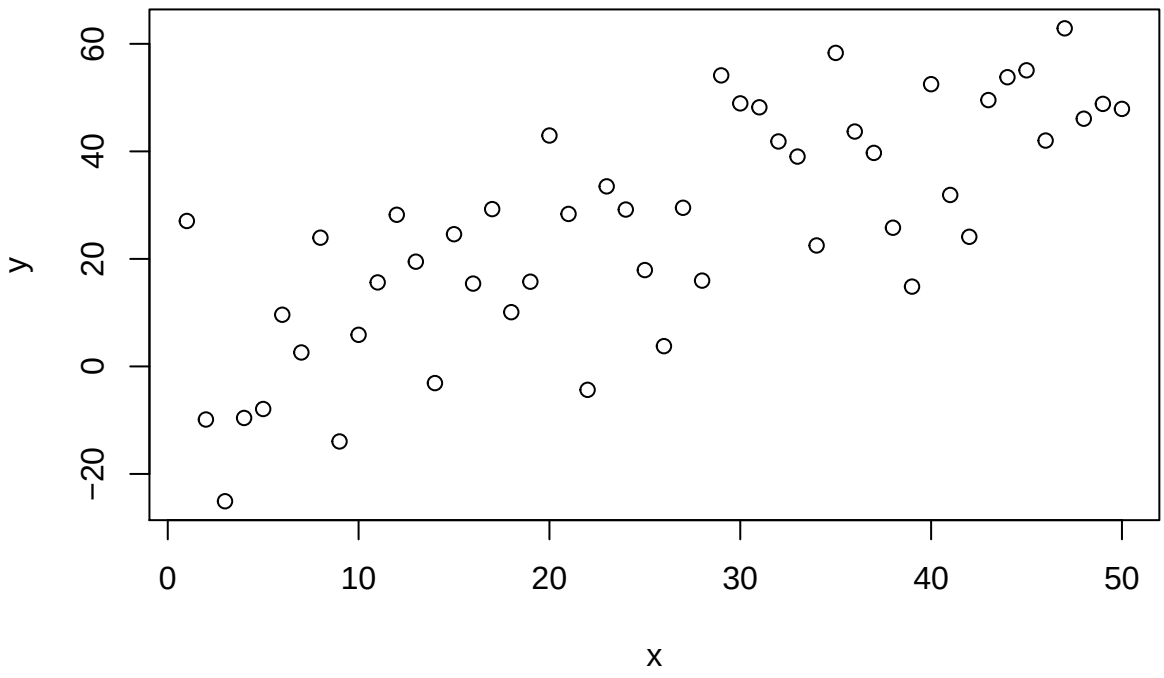
Conditions on doing LSLR

1. **Linearity** The data when plotted should show a linear trend, if not, you will need a more advanced method!!
2. **normal residuals** You can't have any crazy outliers
3. **homoscedasticity of residuals** The variance of the residuals must be a constant
4. **Independence** The observations must be independent. Time series is often has data strongly correlated to previous



Inference for LR

```
x = 1:50     # Notice X is fixed (NOT RANDOM VARIABLE)
y = x + rnorm(50,sd=15)
```



We think about our n samples as being drawn from a population which is perfectly described by the model

H_0: the *population* slope $\beta_1 = 0$

H_A: the *population* slope $\beta_1 \neq 0$

The null hypothesis says:

There is no linear relationship between the dependent response, y, and the explanatory variable, x

To assess the hypotheses, we identify a standard error for the point estimate, compute a test statistic, and get p-value

$\hat{\beta}_1 = \frac{\text{Cov}(xy)}{s_x^2}$ is a **point estimate** for the population slope β_1

Since our estimate for the population slope is a ***statistic*** that means it has some **sampling distribution**

$\frac{\hat{\beta}_1 - \beta_1}{SE} \sim$ normal or t-distribution, but what is SE?

$$\text{Var}(\hat{\beta}_1) = \text{Var}\left(\frac{\text{Cov}(xy)}{s_x^2}\right) = \text{Var}\left(\frac{\sum_i (x_i - \overline{x})(y_i - \overline{y})}{\sum_i (x_i - \overline{x})^2}\right)$$

Note that $\sum_i (x_i - \overline{x})\overline{y} = 0$ so

$$\text{Var}(\hat{\beta}_1) = \text{Var}\left(\frac{\sum_i (x_i - \overline{x}) y_i}{\sum_i (x_i - \overline{x})^2}\right) = \text{Var}\left(\frac{\sum_i (x_i - \overline{x})(\beta_1 x_i + \beta_0 + \epsilon_i)}{\sum_i (x_i - \overline{x})^2}\right)$$

take the attitude that only ϵ_i is a random variable and the data x are fixed, This is a bit ***counterintuitive*** since we are computing s_x on the data - but bear with it - x is NOT a random variable!

---

then since the variance of constants is zero,

$\text{Var}(\hat{\beta}_1) = \text{Var}\left(\frac{\sum_i (x_i - \overline{x})\epsilon_i}{\sum_i (x_i - \overline{x})^2}\right)$

again use $\text{Var}(\text{kZ}) = k^2\text{Var}(\text{Z})$ to get $\text{Var}(\hat{\beta}_1) = \frac{\sigma^2}{\sum_i (x_i - \overline{x})^2}$

or $SE = \frac{\sigma}{\sqrt{\sum_i (x_i - \overline{x})^2}} = \frac{\sigma}{s_x\sqrt{n-1}}$

when estimating the population variance of the residuals is

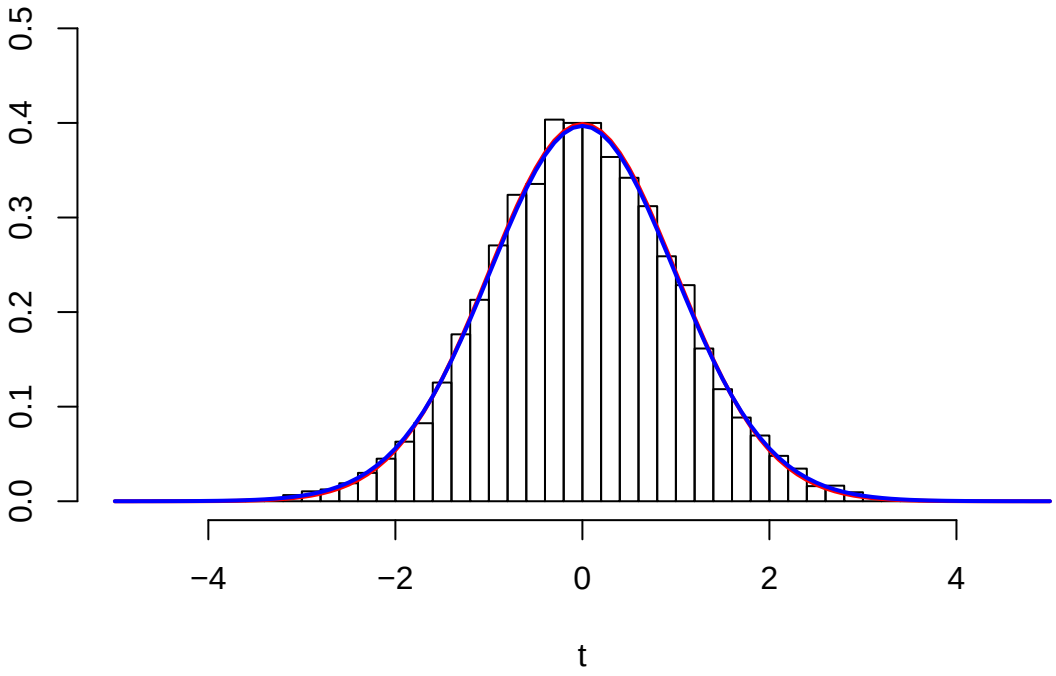
$SE = \sqrt{\frac{1}{n-2}\frac{\sum_i \epsilon_i}{\sum_i (x_i - \overline{x})^2}}$

that was rough!

- Since we will estimate σ^2 from the data we will say that our sample statistic should be t distributed instead of normal
- The test statistic t is equal to $\frac{\hat{\beta}_1 - 0}{SE} = \frac{\hat{\beta}_1}{\sqrt{\frac{1}{n-2}\frac{\sum_i \epsilon_i}{\sum_i (x_i - \overline{x})^2}}}$

```
    t_statistics <- vector()
    x = 1:50
for (m in 1:10000) {
    population_slope = 1  #could set to any number (including 0)
    y = population_slope*x + rnorm(50,sd=15)
    hat_beta_1 = cov(x,y)/cov(x,x)
    hat_beta_0 = mean(y) - hat_beta_1*mean(x)
    SE = sqrt((1.0/48)*sum((y-(hat_beta_0+hat_beta_1*x))^2)
              /sum((x-mean(x))^2))
    t_statistics[m] = (hat_beta_1-population_slope)/SE
}
```

The takeaway is that when we use this weird definition of the SE, we get a sampling distribution for the slope, $\hat{\beta}_1$ that works!



running `lm` in R

We don't need to do all this math, we can just run linear models in R

```
lm(y ~ x)
```

```
##
## Call:
## lm(formula = y ~ x)
##
## Coefficients:
## (Intercept)            x
##     -0.5616       1.0558
```

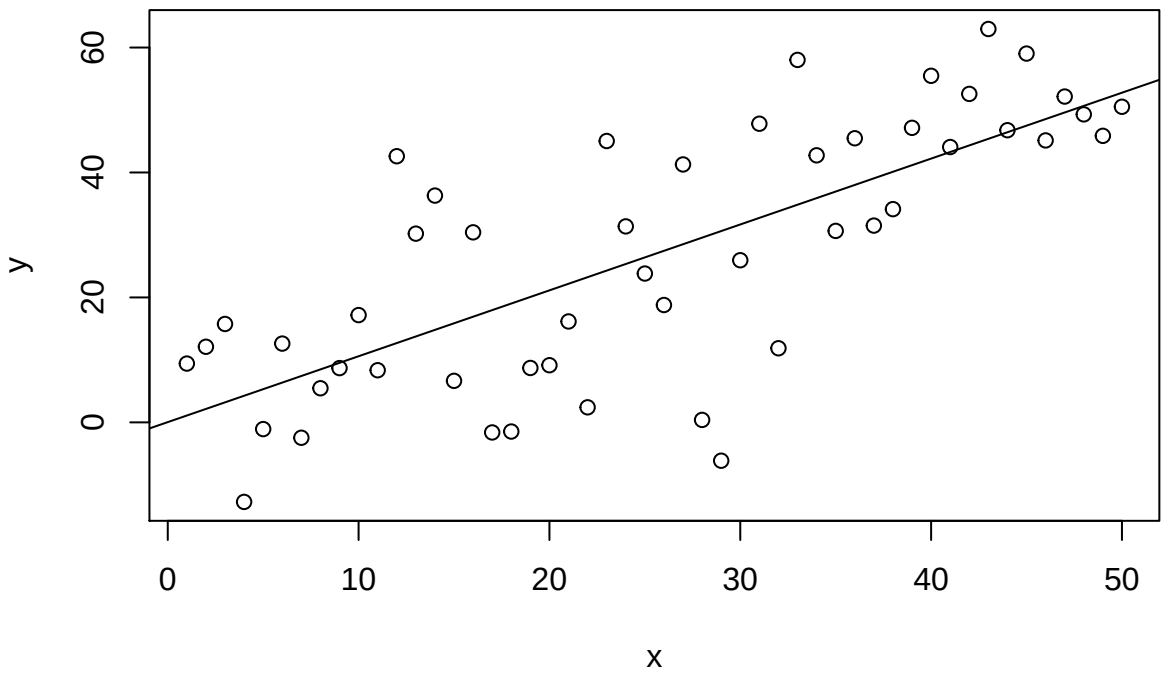
- side note: the null hypothesis for $\hat{\beta}_0$ is usually also that the population intercept $\beta_0 = 0$. So a large p value for the intercept for our simulated model would make sense.

```
summary(lm(y ~ x))
```

```
##
## Call:
## lm(formula = y ~ x)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
```

```
## -46.325  -9.593   0.706   9.508  34.613
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -0.5616     4.6611  -0.120    0.905
## x             1.0558     0.1591   6.637 2.65e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 16.23 on 48 degrees of freedom
## Multiple R-squared:  0.4786, Adjusted R-squared:  0.4677
## F-statistic: 44.05 on 1 and 48 DF,  p-value: 2.65e-08
```

```
x = 1:50    # Notice X is fixed (NOT RANDOM VARIABLE)
y = x + rnorm(50,sd=15)
fit = lm(y~x)
```



- This gives us confidence intervals for population parameters based on point estimates with the usual approach:

  CI $=$ point estimate $\pm\, t^*_{df} SE$

- note that the standard error of $\hat{\beta}_0$ can be easily determined from $\hat{\beta}_0 = \overline{y} - \hat{\beta}_1 \overline{x} \rightarrow SE_{\hat{\beta}_0} = \sqrt{\frac{\hat{\sigma}^2}{n} + (-\overline{x})^2 SE^2_{\hat{\beta}_1}}$

- The report of the F statistic can be understood in terms of the full and reduced model, where $\hat{\beta}_1 = 0$
- You can think of the residuals as having model components and then errors within the model
- Let's try to **fully** understand the connection between ANOVA and linear regression

---

$$SSE_{n-2} = \sum_i (y_i - \hat{y})^2 = \sum_i \epsilon_i^2$$

where

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x = \overline{y} - \hat{\beta}_1(\overline{x} - x) = \overline{y} - \frac{\text{Cov}(xy)}{s_x^2}(\overline{x} - x)$$

$SSE_{n-2} = \sum_i (y_i - \hat{\beta}_0 - \hat{\beta}_1 x)^2 =$

$\sum_i (y_i - \overline{y} + \frac{\text{Cov}(xy)}{s_x^2}(\overline{x} - x))^2$

For the 1 parameter model, with $\hat{\beta}_1 = 0$,

$SSE_{n-1} = \sum_i (y_i - \hat{y})^2 = \sum_i (y_i - \overline{y})^2$

$F = \frac{SSE_{n-1} - SSE_{n-2}}{(n-1)-(n-2)} \div \frac{SSE}{n-2} \sim \frac{SSR}{SSE}$

residuals **always be larger or the same** in reduced model (why?) numerator tells us how much variance explained by model

$F \sim \frac{SSG}{SSE} = \frac{SST - SSE}{SSE}$

ANOVA compare grouped factor model SSE to ungrouped SST LR, compare modelled 2-param SSE_{n-2} to SSE_{n-1}

Linear Regression:

$$SSE_{n-1} = SSE_{n-2} + SSR$$

ANOVA:

$$SST = SSE + SSG$$

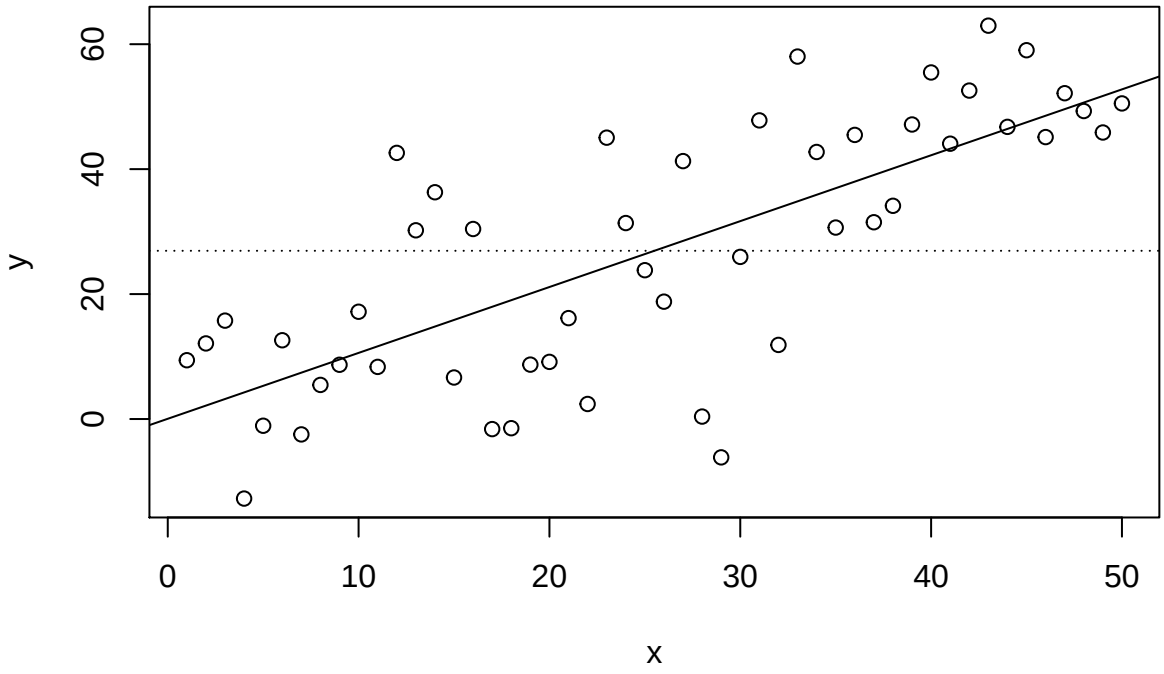
Reduced model

For the 1 parameter model, with $\hat{\beta}_1 = 0$, $y = \overline{y} + \epsilon$

```
fit_reduced = lm(y ~ 1)  # y = mean(y) + noise
```

```
##
## Call:
## lm(formula = y ~ 1)
##
## Coefficients:
## (Intercept)
##       26.93
```

Reduced and full models



Graphical interpretation

1. SSE_{n-2} residuals: distances from data points to solid line
2. SST/SSE_{n-1} residuals: distances from data points to dotted line
3. (SSG/SSR) residuals: the difference between modelled and unmodeled, or the **distance between the solid and dotted lines**

Prediction Interval

Let's say we fit our model on the data x and y

If we are given a new value of x, call it x^*, the pt est for y is

$$\hat{y}^* = \hat{\beta}_0 + \hat{\beta}_1 x^*$$

with **prediction interval**

point estimate $\pm t^*_{df}\hat{\sigma}\sqrt{1 + \frac{1}{n} + \frac{(x^*-\overline{x})^2}{\sum_i (x_i-\overline{x})^2}}$

The last term in square root reflects that we are less confident the farther away x^* is from $\overline{x}$

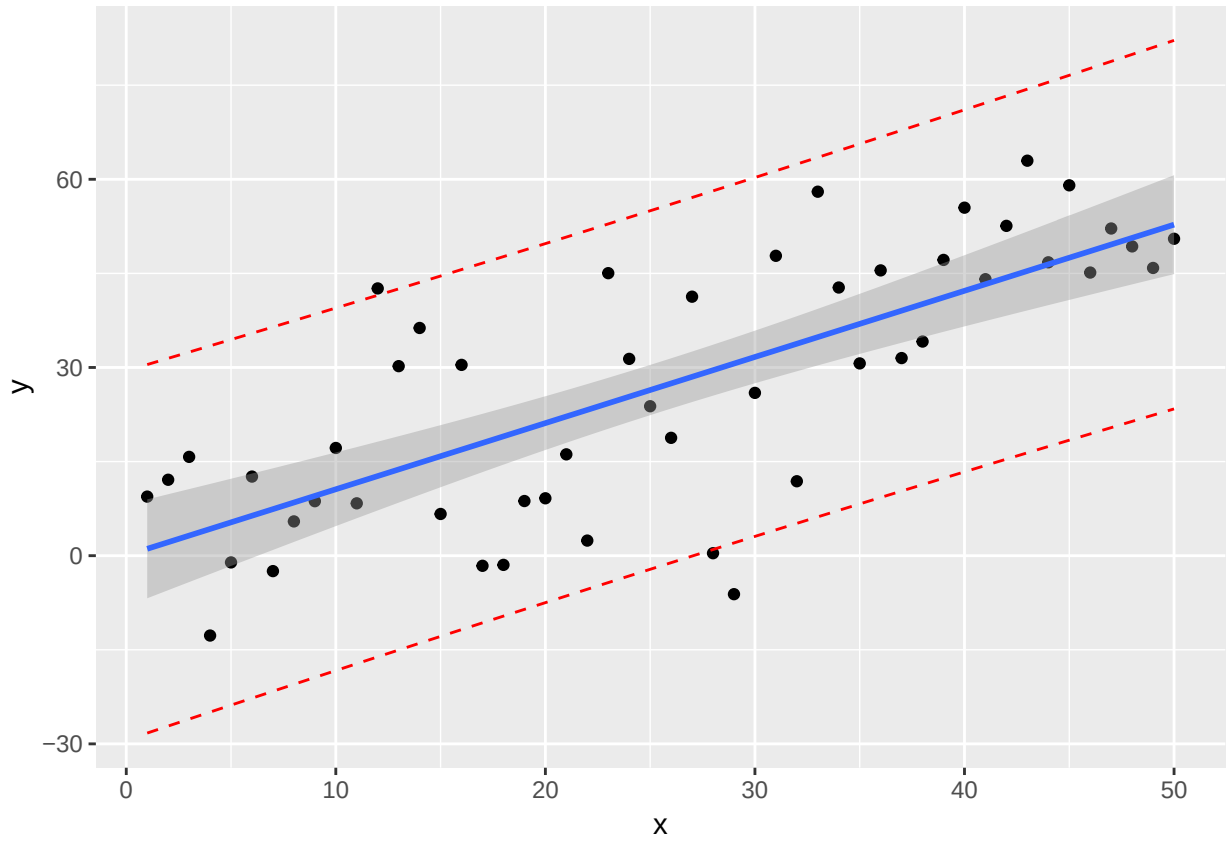
Note that prediction intervals don't go to zero as $n \to \infty$ (whereas CIs do!) there is always underlying uncertainty!

Confidence interval for mean response

Say we want an interval for the average of the response, $\mathrm{E}(\hat{y}^*)$, to some new data x^*, the point estimate is the same as before, but now the confidence interval is

point estimate $\pm t^*_{df}\hat{\sigma}\sqrt{\frac{1}{n} + \frac{(x^*-\overline{x})^2}{\sum_i (x_i-\overline{x})^2}}$

- Note that the interval is bigger the farther you are away from the center, $\overline{x}$
- In general, linear regression is kind of centered about $\overline{x}$,$\overline{y}$ so if you go away, you lose confidence
- This is reflected in the wedge shape on the plot (Next Page)



Notice how the prediction interval is alot larger than the confidence interval, because the prediction interval retains the effect of the residuals

Prediction accuracy

It's possible to save some of your data for testing purposes.

```
x = 1:1000
y = x + rnorm(1000,sd=50)
```

```
model_x = x[1:800]
model_y = y[1:800]
test_x = x[801:1000]
test_y = y[801:1000]
model_dat = data.frame(x=model_x,y=model_y)
test_dat = data.frame(x=test_x,y=test_y)
fit = lm(y ~ x,data=model_dat)
pred <- predict(fit,test_dat)
cor(test_y,pred)
```

```
## [1] 0.807001
```

Min max accuracy and % difference (absolute relative)

```
actuals_pred <-  data.frame(cbind(test_y,pred))
mean(apply(actuals_pred, 1, min) / apply(actuals_pred, 1, max))
```

```
## [1] 0.9598019
```

```
mean(abs((pred - test_y))/test_y)
```

```
## [1] 0.04131519
```

k fold cross-validation

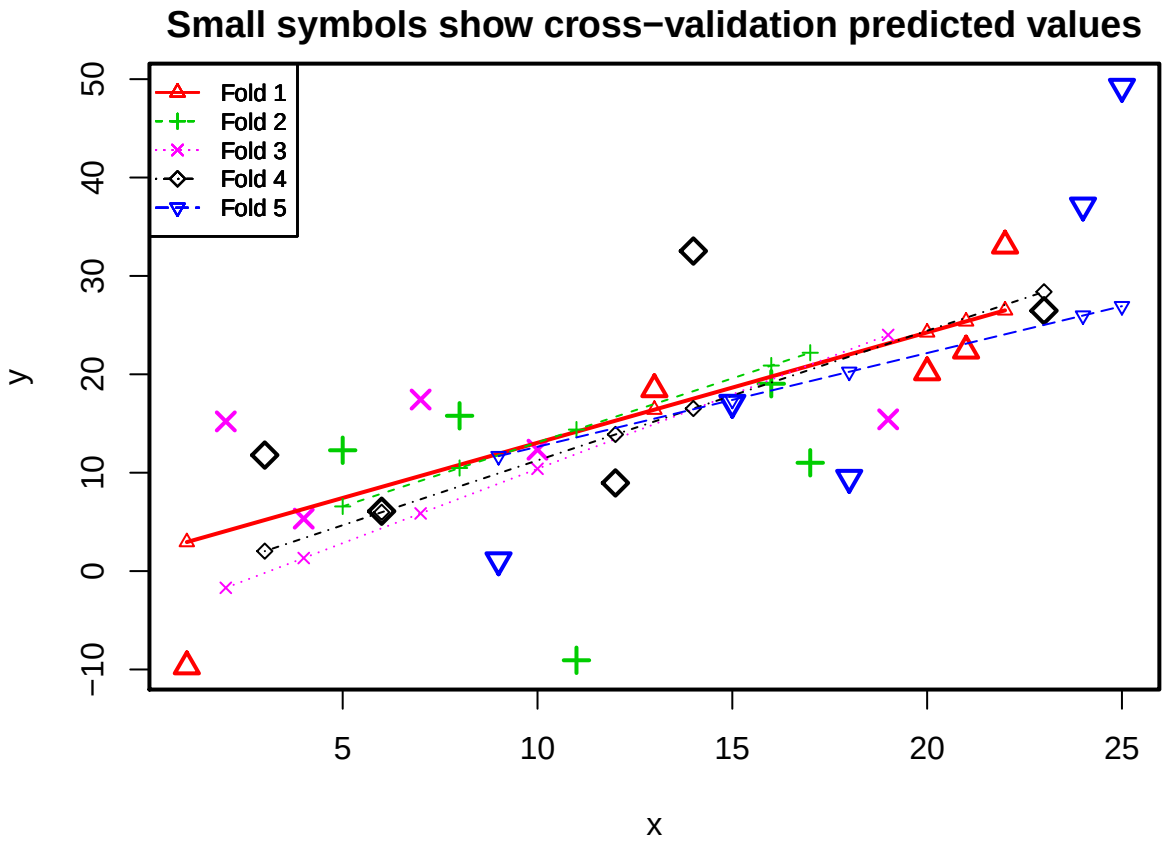
1. Partition the data in k sets randomly
2. choosing the test set to be one of these sets
3. Fits the model on the rest
4. The average of the squares of residuals for different partiononings is recorded (MSE)

```
x = 1:25
y = x + rnorm(25,sd=10)
require('DAAG')
cv_results <- CVlm(data=my_dat,form.lm= y ~ x,m=5)
```

```
attr(cv_results, 'ms')
```

```
## [1] 109
```

```
cv_results <- CVlm(data=my_dat,form.lm= y ~ x,m=5,dots=FALSE,
              legend.pos="topleft",plotit='Observed',printit=F)
```



Multivariate LR

$$y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \epsilon$$

- model your data as lying on a *plane*
- find parameters of *plane of best fit*, $\beta = (\beta_0, \beta_1, \beta_2)$
- linear regression can include non-linearities in the features. The linearity requirement is for the coefficients
- We won't get into the theory/math of multivariate regression, but all the same stuff you did before you can also do with multiple covariates and also with factors as well!!

We will now start doing some examples on real data, since it's **really about time we started applying things we have learned**