

## 432 Class 05

<https://thomaseLove.github.io/432-2023/>

2023-01-31

# Today's Agenda

- The HELP study (today's data) and preliminaries
- Using `ols` to fit a linear model
  - Obtaining coefficients and basic summaries
  - Validating summary statistics like  $R^2$
  - ANOVA in `ols`
  - Plot Effects with `summary` and `Predict`
  - Building and using a nomogram
  - Evaluating Calibration
  - Influential points and `dfbeta`
- Spending Degrees of Freedom on Non-Linearity
  - The Spearman  $\rho^2$  (rho-squared) plot
- Building Non-Linear Predictors in `ols`
  - Polynomial Functions
  - Restricted Cubic Splines
  - Restricting Interaction (Product) Terms

# Today's R Setup

```
knitr::opts_chunk$set(comment = NA)

library(mosaic)           ## auto-loads mosaicData
library(here)
library(janitor)
library(knitr)
library(broom)
library(patchwork)
library(GGally)           ## for scatterplot matrix
library(rms)              ## auto-loads Hmisc
library(tidyverse)

theme_set(theme_bw())
```

# Section 1

Today's Data, from the HELP study

## Today's Data (helpdat, from the HELP study)

Today's data set comes from the Health Evaluation and Linkage to Primary Care trial, and is stored as HELPrct in the mosaicData package.

HELP was a clinical trial of adult inpatients recruited from a detoxification unit. Patients with no primary care physician were randomized to receive a multidisciplinary assessment and a brief motivational intervention or usual care, with the goal of linking them to primary medical care.

We will look at 453 subjects with complete data today.

```
helpdat <- tibble(mosaicData::HELPrct) |>
  select(id, cesd, age, sex, subst = substance,
         mcs, pcs, pss_fr)

dim(helpdat)
```

```
[1] 453    8
```

# Key Variables for Today

Variable	Description
id	subject identifier
cesd	Center for Epidemiologic Studies Depression measure (higher scores indicate more depressive symptoms)
age	subject age (in years)
sex	female (n = 107) or male (n = 346)
subst	primary substance of abuse (alcohol, cocaine or heroin)
mcs	SF-36 Mental Component Score (lower = worse status)
pcs	SF-36 Physical Component Score (lower = worse status)
pss_fr	perceived social support by friends (higher = more support)

- All measures from baseline during the subjects' detoxification stay.
- More data and details at <https://nhorton.people.amherst.edu/help/>.

## helpdat categorical variables

```
helpdat |> tabyl(sex, subst) |>
  adorn_totals(where = c("row", "col")) |>
  adorn_percentages() |>
  adorn_pct_formatting() |>
  adorn_ns(position = "front") |>
  adorn_title(placement = "combined") |>
  kable(align = 'lrrrr')
```

sex/subst	alcohol	cocaine	heroin	Total
female	36 (33.6%)	41 (38.3%)	30 (28.0%)	107 (100.0%)
male	141 (40.8%)	111 (32.1%)	94 (27.2%)	346 (100.0%)
Total	177 (39.1%)	152 (33.6%)	124 (27.4%)	453 (100.0%)

## helpdat quantitative variables

```
helpdat |> select(cesd, age, mcs, pcs, pss_fr) |>  
  inspect(digits = 2)
```

quantitative variables:

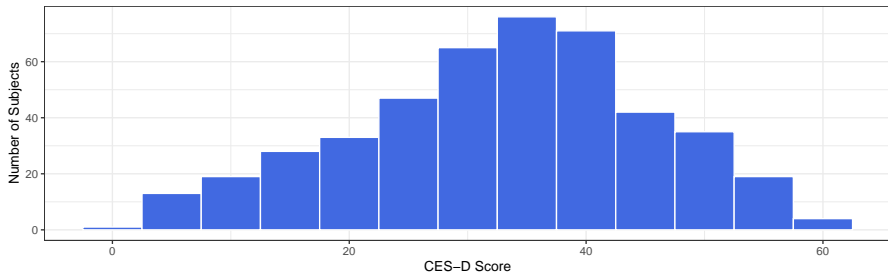
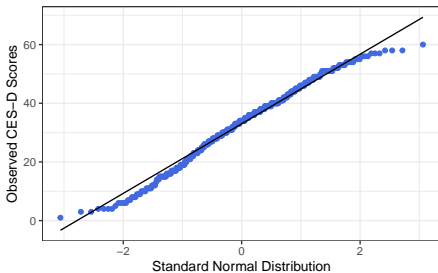
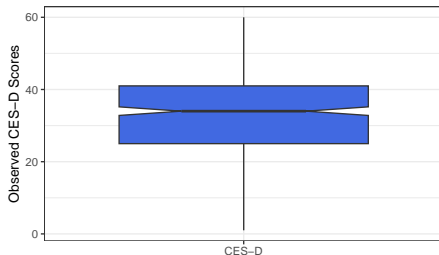
	name	class	min	Q1	median	Q3	max	mean	sd	n	missing
1	cesd	integer	1.0	25	34	41	60	32.8	12.5	453	0
2	age	integer	19.0	30	35	40	60	35.7	7.7	453	0
3	mcs	numeric	6.8	22	29	41	62	31.7	12.8	453	0
4	pcs	numeric	14.1	40	49	57	75	48.0	10.8	453	0
5	pss_fr	integer	0.0	3	7	10	14	6.7	4.0	453	0



# Our Outcome (CES-Depression score)

CES-D Depression Scores from helpdat data

Higher CES-D indicates more depressive symptoms



n = 453, no missing data

# Describing our outcome CES-D

```
describe(helpdat$cesd)
```

```
helpdat$cesd : CESD at baseline
```

n	missing	distinct	Info	Mean	Gmd	.05
453	0	58	0.999	32.85	14.23	10.0
.25	.50	.75	.90	.95		
25.0	34.0	41.0	49.0	52.4		

```
lowest : 1 3 4 5 6, highest: 55 56 57 58 60
```

- Info measures the variable's information between 0 and 1: the higher the Info, the more continuous the variable is (the fewer ties there are.)
- Gmd = Gini's mean difference, a robust measure of variation. If you randomly selected two of the 453 subjects many times, the mean difference in cesd would be 14.23 points.

## We have some labels in our data

```
str(helpdat)
```

```
tibble [453 x 8] (S3: tbl_df/tbl/data.frame)
 $ id      : int [1:453] 1 2 3 4 5 6 7 8 9 10 ...
 ..- attr(*, "label")= chr "subject ID"
 $ cesd    : int [1:453] 49 30 39 15 39 6 52 32 50 46 ...
 ..- attr(*, "label")= chr "CESD at baseline"
 $ age     : int [1:453] 37 37 26 39 32 47 49 28 50 39 ...
 ..- attr(*, "label")= chr "age (years)"
 $ sex     : Factor w/ 2 levels "female","male": 2 2 2 1 2 1 1 2 ...
 ..- attr(*, "label")= chr "sex"
 $ subst   : Factor w/ 3 levels "alcohol","cocaine",...: 2 1 3 3 ...
 ..- attr(*, "label")= chr "primary substance of abuse"
 $ mcs     : num [1:453] 25.11 26.67 6.76 43.97 21.68 ...
 ..- attr(*, "label")= chr "SF-36 Mental Component Score"
 $ pcs     : num [1:453] 58.4 36 74.8 61.9 37.3 ...
 ..- attr(*, "label")= chr "SF-36 Physical Component Score"
```

## Scatterplot Matrix (code)

```
temp <- helpdat |>
  select(age, mcs, pcs, pss_fr, sex, subst, cesd)

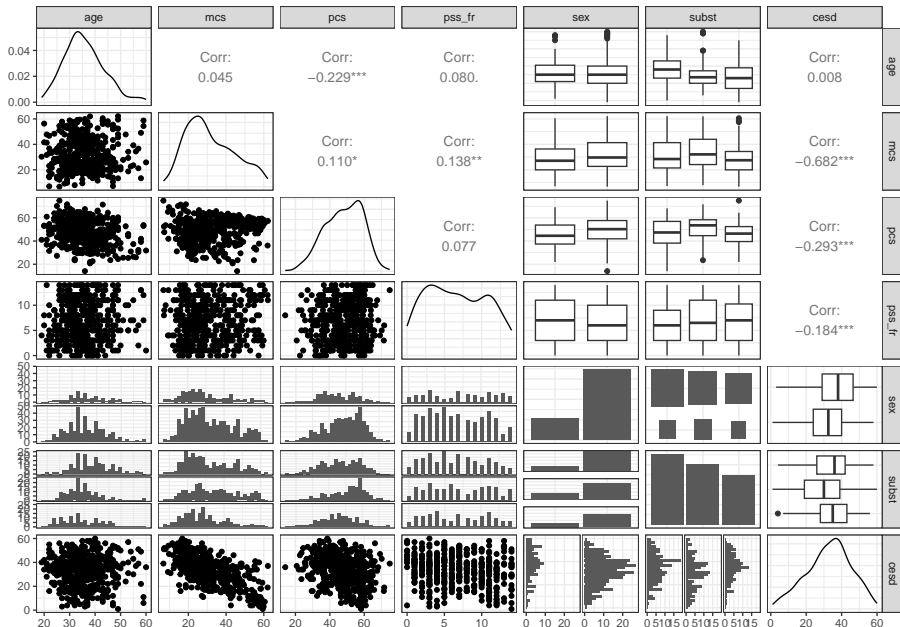
ggpairs(temp) ## ggpairs from the GGally package
```

Note that we're placing the outcome (cesd) last (result on next slide.)

## Saving the Data Set

```
write_rds(helpdat, "c05/data/helpdat.Rds")
```

# Scatterplot Matrix (result)



## Section 2

Using `ols` to fit a linear regression model

## Fitting using `ols`

The `ols` function stands for ordinary least squares and comes from the `rms` package, by Frank Harrell and colleagues. Any model fit with `lm` can also be fit with `ols`.

- To predict `var_y` using `var_x` from the `my_tibble` data, we would use the following syntax:

```
dd <- datadist(my_tibble)
options(datadist = "dd")

model_name <- ols(var_y ~ var_x, data = my_tibble,
                  x = TRUE, y = TRUE)
```

This leaves the following questions:

- 1 What's the `datadist` stuff doing?
- 2 Why use `x = TRUE`, `y = TRUE` in the fit?

# What is datadist?

Before we fit any `ols` model to data from `my_tibble`, we'll use:

```
dd <- datadist(my_tibble)
options(datadist = "dd")
```

*Run (the `datadist` code above) once before any models are fitted, storing the distribution summaries for all potential variables. Adjustment values are 0 for binary variables, the most frequent category (or optionally the first category level) for categorical (factor) variables, the middle level for ordered factor variables, and medians for continuous variables.*

- excerpted from the `datadist` documentation



## Why use `x = TRUE`, `y = TRUE` in the fit?

Once we've set up the distribution summaries with the `datadist` code, we fit linear regression models using the same fitting routines as `lm` with `ols`:

```
model_name <- ols(var_y ~ var_x, data = my_tibble,  
                  x = TRUE, y = TRUE)
```

- `ols` stores additional information beyond what `lm` does
- `x = TRUE` and `y = TRUE` save even more expanded information that we'll need in building plots and summaries of the fit.
- The defaults are `x = FALSE`, `y = FALSE`, but in 432, we'll always want these to be saved.

## Using `ols` to fit a Two-Predictor Model

Now, we'll fit an `ols` model predicting our outcome (`cesd`) using two predictors (`mcs` and `subst`) using the `helpdat` tibble.

- Start with setting up the `datadist`
- Then fit the model, including `x = TRUE`, `y = TRUE`

```
dd <- datadist(helpdat)
options(datadist = "dd")

mod1 <- ols(cesd ~ mcs + subst, data = helpdat,
            x = TRUE, y = TRUE)
```

# Contents of mod1?

```
> mod1
Linear Regression Model

ols(formula = cesd ~ mcs + subst, data = helpdat, x = TRUE, y = TRUE)


```

		Model Likelihood		Discrimination	
		Ratio Test		Indexes	
Obs	453	LR chi2	295.10	R2	0.479
sigma	9.0657	d.f.	3	R2 adj	0.475
d.f.	449	Pr(> chi2)	0.0000	g	9.827

```


Residuals


```

	Min	1Q	Median	3Q	Max
	-25.43696	-6.74592	0.09334	6.16212	24.24842

```



```

	Coef	S.E.	t	Pr(> t )
Intercept	55.3026	1.2724	43.46	<0.0001
mcs	-0.6570	0.0337	-19.48	<0.0001
subst=cocaine	-3.4440	1.0055	-3.43	0.0007
subst=heroin	-1.7791	1.0681	-1.67	0.0965

- Likelihood Ratio Test?
- What is the discrimination index  $g$ ?

## New elements in `ols`

For our `mod1`,

- Model Likelihood Ratio test output includes `LR chi2 = 295.10`, `d.f. = 3`, `Pr(> chi2) = 0.0000`

The log of the likelihood ratio, multiplied by -2, yields a test against a  $\chi^2$  distribution. Interpret this as a goodness-of-fit test that compares `mod1` to a null model with only an intercept term. In `ols` this is similar to a global (ANOVA) F test.

- Under the  $R^2$  values, we have `g = 9.827`.
- This is the *g*-index, based on Gini's mean difference. If you randomly selected two of the subjects in the model, the average difference in predicted `cesd` will be 9.827.
- This can be compared to the Gini's mean difference for the original `cesd` values, from `describe`, which was `Gmd = 14.23`.

# Validate the summary statistics of an `ols` fit

- Can we validate summary statistics by resampling?

```
> set.seed(432)
> validate(mod1)
```

	index.orig	training	test	optimism	index.corrected	n
R-square	0.4787	0.4874	0.4737	0.0137	0.4650	40
MSE	81.4606	79.7851	82.2361	-2.4510	83.9116	40
g	9.8272	9.9133	9.8038	0.1095	9.7177	40
Intercept	0.0000	0.0000	0.2793	-0.2793	0.2793	40
Slope	1.0000	1.0000	0.9894	0.0106	0.9894	40

- The data used to fit the model provide an over-optimistic view of the quality of fit.
- We're interested here in assessing how well the model might work in new data, and to do so, we can use a resampling approach.
- Consider  $R^2$  here...

## Interpreting the Resampling Validation Results

	index.orig	training	test	optimism	index.corrected	n
$R^2$	0.4787	0.4874	0.4737	0.0137	0.4650	40

- `index.orig` for  $R^2$  is 0.4787. That's what we get from the data we used to fit the model, and is what we see in our standard output.
- With `validate` we create 40 (by default) bootstrapped resamples of the data and then split each of those into training and test samples.
  - For each of the 40 splits, R refits the model (same predictors) in the training sample to obtain  $R^2$ : mean across 40 splits is 0.4874
  - Check each model in its test sample: average  $R^2$  was 0.4737
- `optimism` = training result - test result = 0.0137
- `index.corrected` = `index.orig` - `optimism` = 0.4650

While our *nominal*  $R^2$  is 0.4787 for `mod1`, correcting for optimism yields a *validated*  $R^2$  of 0.4650

- $R^2 = 0.4650$  better estimates how `mod1` will perform in new data.

## ANOVA for mod1 fit by ols

```
> anova(mod1)
```

Analysis of Variance					Response: cesd
Factor	d.f.	Partial SS	MS	F	P
mcs	1	31182.7237	31182.72373	379.42	<.0001
subst	2	968.7563	484.37816	5.89	0.003
REGRESSION	3	33886.8359	11295.61195	137.44	<.0001
ERROR	449	36901.6542	82.18631		

- This adds a line for the complete regression model (both terms) which can be helpful, but is otherwise the same as `anova` after `lm`.
- As with `lm`, this is a sequential ANOVA table, so if we had included `subst` in the model first, we'd get a different SS, MS, F and p for `mcs` and `subst`, but the same REGRESSION and ERROR results.

## summary for mod1 fit by ols

```
> summary(mod1, conf.int = 0.90)
```

Effects	Response : cesd						
Factor	Low	High	Diff.	Effect	S.E.	Lower 0.9	Upper 0.9
mcs	21.676	40.941	19.266	-12.6580	0.64984	-13.7290	-11.587000
subst - cocaine:alcohol	1.000	2.000	NA	-3.4440	1.00550	-5.1013	-1.786700
subst - heroin:alcohol	1.000	3.000	NA	-1.7791	1.06810	-3.5396	-0.018654

- How do we interpret the subst effects estimated by this model?
  - Effect of subst being cocaine instead of alcohol on ces\_d is -3.44 assuming no change in mcs, with 90% CI (-5.10, -1.79).
  - Effect of subst being heroin instead of alcohol on ces\_d is -1.78 assuming no change in mcs, with 90% CI (-3.54, -0.02).

But what about the mcs effect?



## summary for mod1 fit by ols

```
> summary(mod1, conf.int = 0.90)
      Effects      Response : cesd

Factor      Low      High      Diff.      Effect      S.E.      Lower 0.9      Upper 0.9
mcs          21.676  40.941  19.266  -12.6580  0.64984  -13.7290  -11.587000
subst - cocaine:alcohol  1.000   2.000    NA   -3.4440  1.00550   -5.1013   -1.786700
subst - heroin:alcohol   1.000   3.000    NA   -1.7791  1.06810   -3.5396   -0.018654
```

- Effect of mcs: -12.66 is the estimated change in cesd associated with a move from  $mcs = 21.68$  (see Low value) to  $mcs = 40.94$  (the High value) assuming no change in subst.
- ols chooses the Low and High values from the interquartile range.

```
quantile(helpdat$mcs, c(0.25, 0.75))
```

25%	75%
21.67575	40.94134

## Plot the summary to see effect sizes

- Goal: plot effect sizes for similar moves within predictor distributions.

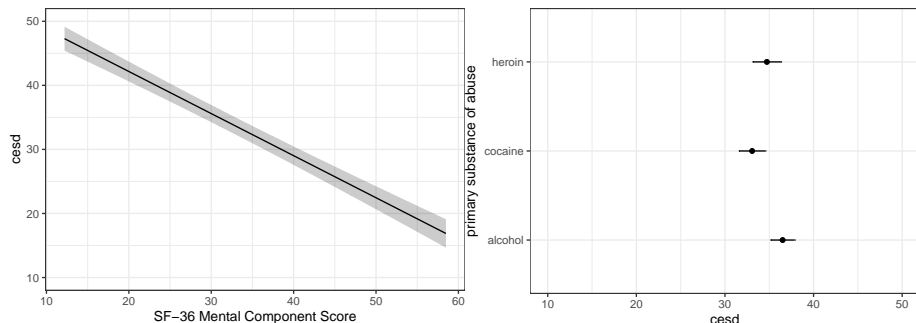
```
plot(summary(mod1))
```



- The triangles indicate the point estimate, augmented with confidence interval bars.
  - The 90% confidence intervals are plotted with the thickest bars.
  - The 95% CIs are then shown with thinner, more transparent bars.
  - Finally, the 99% CIs are shown as the longest, thinnest bars.

# What do the individual effects look like?

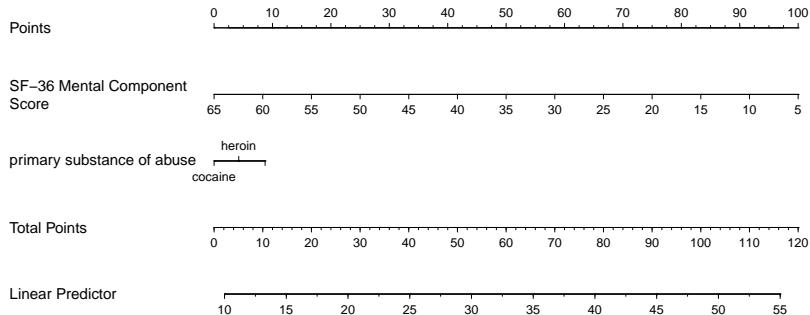
```
ggplot(Predict(mod1, conf.int = 0.95), layout = c(1,2))
```



- The left plot shows the impact of changing `mcs` on `cesd` holding `subst` at its baseline level (alcohol).
- The right plot shows the impact of changing `subst` on `cesd` holding `mcs` at its median value which is 28.602417.
- Defaults: add 95% CI bands and layout tries for a square.

# Build a nomogram for the ols fit

```
plot(nomogram(mod1))
```



# Nomograms

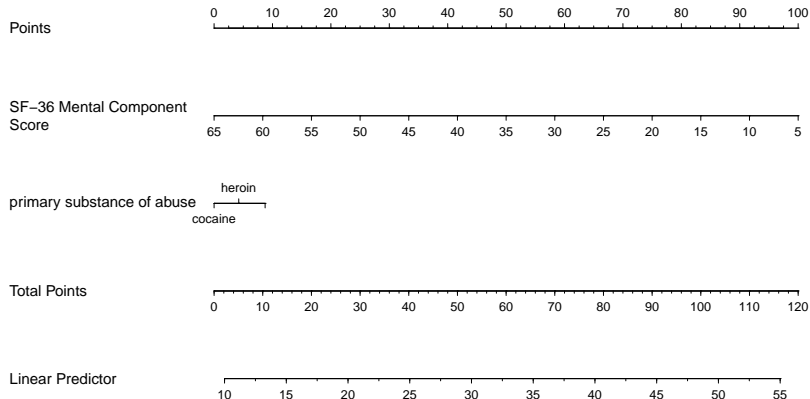
For complex models (this model isn't actually very complex) it can be helpful to have a tool that will help you see the modeled effects in terms of their impact on the predicted outcome.

A *nomogram* is an established graphical tool for doing this.

- Find the value of each predictor on its provided line, and identify the “points” for that predictor by drawing a vertical line up to the “Points”.
- Then sum up the points over all predictors to obtain “Total Points”.
- Draw a vertical line down from the “Total Points” to the “Linear Predictor” to get the predicted *cesd* for this subject.

# Using the nomogram for the mod1 fit

Predicted cesd for a subject with  $mcs = 35$  and  $subst = \text{heroin}$ ?



## Actual Prediction for such a subject...

- The `predict` function for our `ols` fit provides fitted values.

```
predict(mod1,  
        newdata = tibble(mcs = 35, subst = "heroin"))
```

1

30.52766

- The `broom` package can also support `rms` fits

```
augment(mod1,  
        newdata = tibble(mcs = 35, subst = "heroin"))
```

# A tibble: 1 x 3

	mcs	subst	.fitted
	<dbl>	<chr>	<dbl>
1	35	heroin	30.5

# Assessing the Calibration of mod1

We would like our model to be well-calibrated, in the following sense...

- Suppose our model assigns a predicted outcome of 6 to several subjects. If the model is well-calibrated, then we expect the mean of those subjects' actual outcomes to be very close to 6.

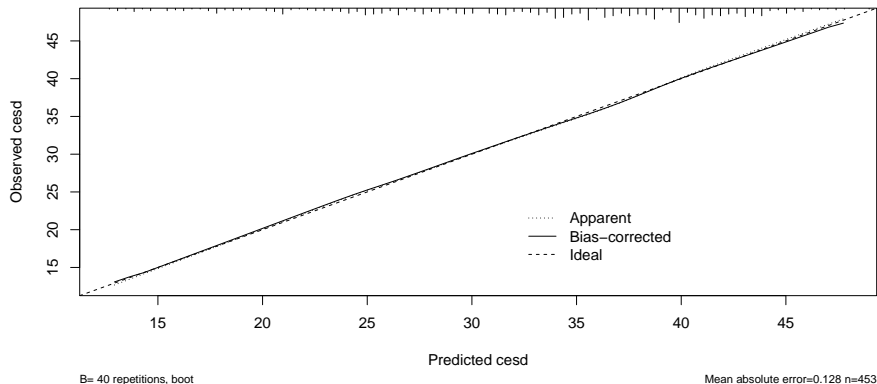
We'd like to look at the relationship between the observed `cesd` outcome and our predicted `cesd` from the model.

- The calibration plot we'll create provides two estimates (with and without bias-correction) of the predicted vs. observed values of our outcome, and compares these to the ideal scenario (predicted = observed).
- The plot uses resampling validation to produce bias-corrected estimates and uses lowess smooths to connect across predicted values.
- Calibration plots require `x = TRUE`, `y = TRUE` in the `ols` fit.



# Calibration Plot for mod1

```
set.seed(43299); plot(calibrate(mod1))
```



n=453 Mean absolute error=0.128

Mean squared error=0.02428

## Influential Points for mod1?

The `dfbeta` value for a particular subject and coefficient  $\beta$  is the change in the coefficient that happens when the subject is excluded from the model.

```
which.influence(mod1, cutoff = 0.2)
```

```
$Intercept
```

```
[1] 8 351 405 433
```

```
$mcs
```

```
[1] 351 402 450
```

```
$subst
```

```
[1] 351
```

- These are the subjects that have absolute values of `dfbetas` that exceed the specified cutoff (default is 0.2 but it's an arbitrary choice.)

## Show the influential points more directly?

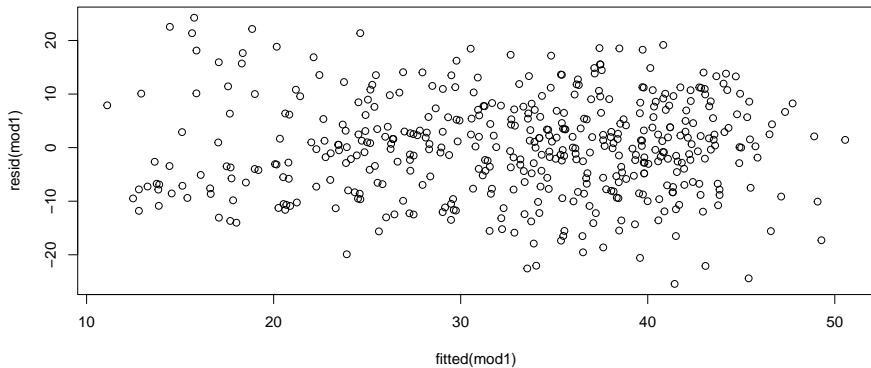
```
w <- which.influence(mod1, cutoff = 0.2)
d <- helpdat |> select(mcs, subst, cesd) |> data.frame()
show.influence(w, d)
```

	Count	mcs	subst
8	1	9.16053	alcohol
351	3	*57.48944	*heroin
402	1	*55.47938	alcohol
405	1	15.07887	alcohol
433	1	18.59431	alcohol
450	1	*62.17550	alcohol

- Count = number of coefficients where this row appears influential.
- Use `helpdat |> slice(351)` to see row 351 in its entirety.
- Use residual plots (with an `lm` fit) to check Cook's distances.

# Residuals vs. Fitted Values is easy from `ols`

```
plot(resid(mod1) ~ fitted(mod1))
```



## Fitting all Residual Plots for mod1

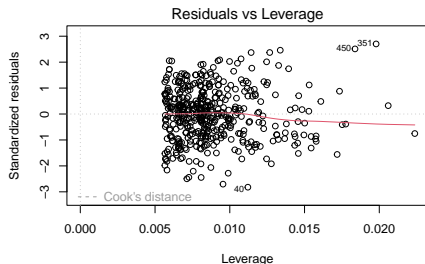
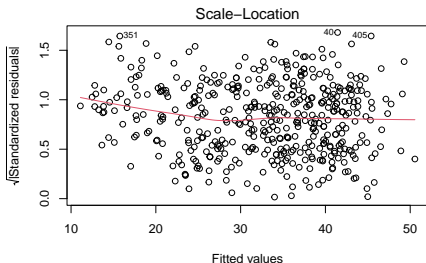
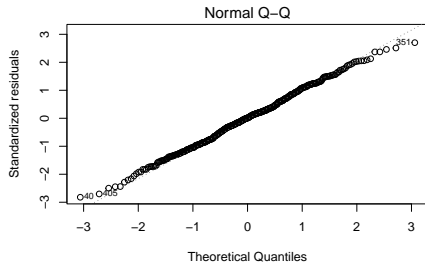
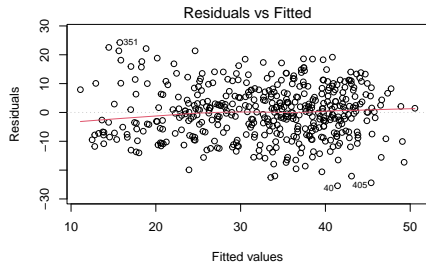
To fit more complete residual plots (and other things) we can fit the `lm` version of this same model...

```
mod1_lm <- lm(cesd ~ mcs + subst, data = helpdat)

par(mfrow = c(2,2)); plot(mod1_lm); par(mfrow = c(1,1))
```

- Plots are shown on the next slide. While the subject in row 351 is more influential than most other points, it doesn't reach the standard of a problematic Cook's distance.

# Residual Plots for mod1



## Section 3

### Thinking about Non-Linear Terms?

# Non-Linear Terms

In building a linear regression model, we're most often going to be thinking about:

- for quantitative predictors, some curvature...
  - perhaps polynomial terms
  - but more often restricted cubic splines
- for any predictors, possible interactions
  - between categorical predictors
  - between categorical and quantitative predictors
  - between quantitative predictors



# Polynomial Regression

A polynomial in the variable  $x$  of degree  $D$  is a linear combination of the powers of  $x$  up to  $D$ . Fitting such a model creates a **polynomial regression**.

- Linear:  $y = \beta_0 + \beta_1 x$
- Quadratic:  $y = \beta_0 + \beta_1 x + \beta_2 x^2$
- Cubic:  $y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3$
- Quartic:  $y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \beta_4 x^4$
- Quintic:  $y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \beta_4 x^4 + \beta_5 x^5$

An **orthogonal polynomial** sets up a model design matrix and then scales those columns so that each column is uncorrelated with the previous ones.

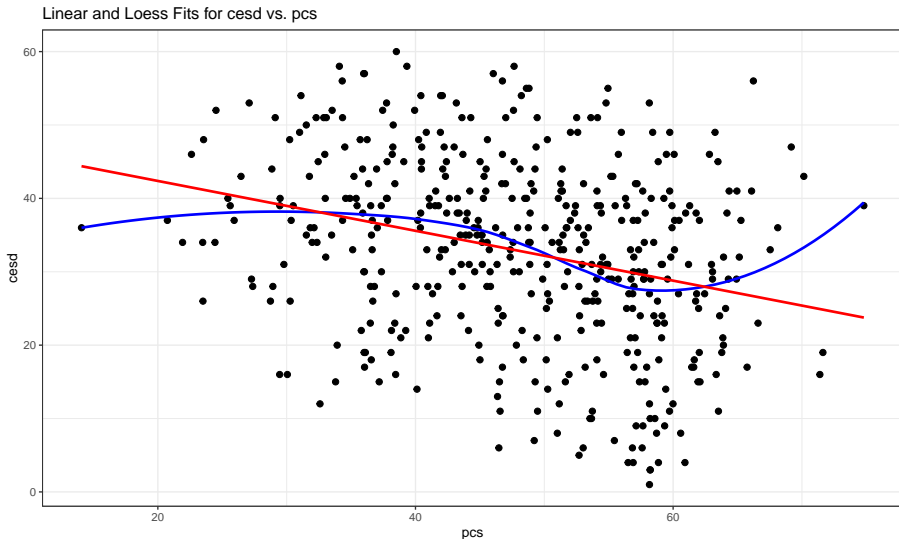
- This reduction in collinearity (correlation between predictors) lets us gauge whether the addition of any particular polynomial term improves model fit.

## A new predictor: use pcs to predict cesd?

- Let's look at both a linear fit and a loess smooth to see if they indicate meaningfully different things about the association between pcs and cesd

```
ggplot(helpdat, aes(x = pcs, y = cesd)) +  
  geom_point(size = 2) +  
  geom_smooth(method = "loess", formula = y ~ x,  
              se = FALSE, col = "blue") +  
  geom_smooth(method = "lm", formula = y ~ x,  
              se = FALSE, col = "red") +  
  labs(title = "Linear and Loess Fits for cesd vs. pcs")
```

# Linear and Loess Fits for cesd with pcs



## Fitting polynomial regressions with `ols`

```
dd <- datadist(helpdat)
options(datadist = "dd")

mod_B1 <- ols(cesd ~ pcs,
              data = helpdat, x = TRUE, y = TRUE)
mod_B2 <- ols(cesd ~ pol(pcs, 2),
              data = helpdat, x = TRUE, y = TRUE)
mod_B3 <- ols(cesd ~ pol(pcs, 3),
              data = helpdat, x = TRUE, y = TRUE)
```

- Note the use of `pol()` from the `rms` package here to fit orthogonal polynomials, rather than `poly()` which we used for an `lm` fit.

## Model B1 (linear in pcs)

```
> mod_B1
Linear Regression Model

ols(formula = cesd ~ pcs, data = helpdat, x = TRUE, y = TRUE)

              Model Likelihood      Discrimination
              Ratio Test              Indexes
Obs          453    LR chi2      40.57    R2        0.086
sigma11.9796    d.f.           1    R2 adj     0.084
d.f.          451    Pr(> chi2) 0.0000    g         4.177

Residuals

      Min       1Q   Median       3Q      Max
-28.4116  -7.8036   0.6846   8.7917  29.3281

      Coef      S.E.      t      Pr(>|t|)
Intercept 49.1673  2.5728  19.11 <0.0001
pcs       -0.3396  0.0522  -6.50 <0.0001
```

## Model B2 (quadratic polynomial in pcs)

```
> mod_B2
Linear Regression Model

ols(formula = cesd ~ pol(pcs, 2), data = helpdat, x = TRUE, y = TRUE)
```

		Model Likelihood		Discrimination	
		Ratio Test		Indexes	
Obs	453	LR chi2	40.68	R2	0.086
sigma	11.9915	d.f.	2	R2 adj	0.082
d.f.	450	Pr(> chi2)	0.0000	g	4.199

Residuals

	Min	1Q	Median	3Q	Max
	-28.387	-7.750	0.591	8.634	29.697

	Coef	S.E.	t	Pr(> t )
Intercept	46.4007	8.7967	5.27	<0.0001
pcs	-0.2136	0.3867	-0.55	0.5809
pcs^2	-0.0014	0.0041	-0.33	0.7424

## Model B3 (cubic polynomial in pcs)

```
> mod_B3
Linear Regression Model

ols(formula = cesd ~ pol(pcs, 3), data = helpdat, x = TRUE, y = TRUE)

              Model Likelihood      Discrimination
              Ratio Test              Indexes
Obs          453    LR chi2      48.70    R2          0.102
sigma11.8991    d.f.              3    R2 adj       0.096
d.f.          449    Pr(> chi2) 0.0000    g           4.556

Residuals

      Min       1Q   Median       3Q      Max
-27.5245  -8.2651   0.7988   8.9004  27.4480

      Coef      S.E.      t      Pr(>|t|)
Intercept -13.4076  22.8605  -0.59  0.5578
pcs         4.1323   1.5825   2.61  0.0093
pcs^2      -0.1010   0.0354  -2.85  0.0046
pcs^3       0.0007   0.0003   2.83  0.0049
```

## Store the polynomial fits

First, we need to store the values. Again broom doesn't play well with `ols` fits, so I'll just add the predictions as columns

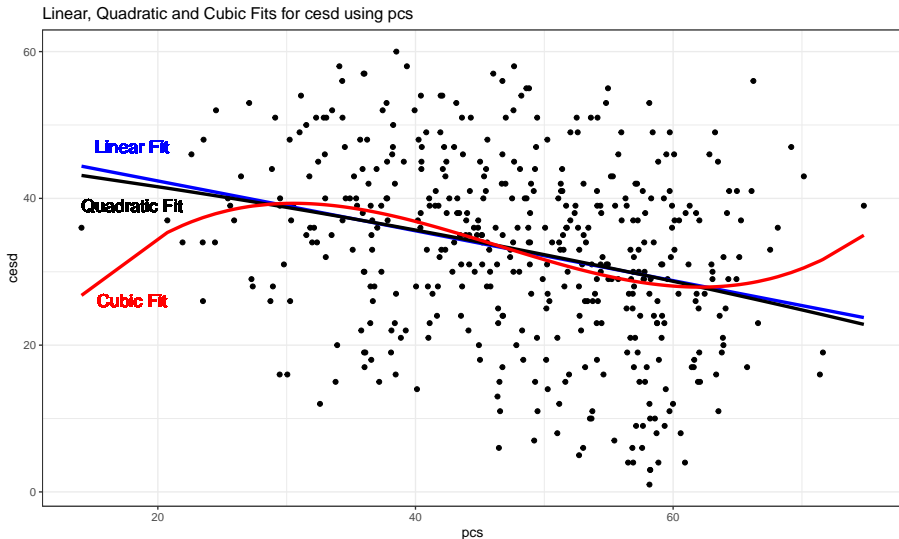
```
cesd_fits <- helpdat |>
  mutate(fitB1 = predict(mod_B1),
         fitB2 = predict(mod_B2),
         fitB3 = predict(mod_B3))
```



## Code to plot polynomial fits

```
ggplot(cesd_fits, aes(x = pcs, y = cesd)) +  
  geom_point() +  
  geom_line(aes(x = pcs, y = fitB1),  
            col = "blue", size = 1.25) +  
  geom_line(aes(x = pcs, y = fitB2),  
            col = "black", size = 1.25) +  
  geom_line(aes(x = pcs, y = fitB3),  
            col = "red", size = 1.25) +  
  geom_text(x = 18, y = 47, label = "Linear Fit",  
            size = 5, col = "blue") +  
  geom_text(x = 18, y = 39, label = "Quadratic Fit",  
            size = 5, col = "black") +  
  geom_text(x = 18, y = 26, label = "Cubic Fit",  
            size = 5, col = "red") +  
  labs(title = "Linear, Quadratic and Cubic Fits for cesd us")
```

# The Polynomial Fits, plotted

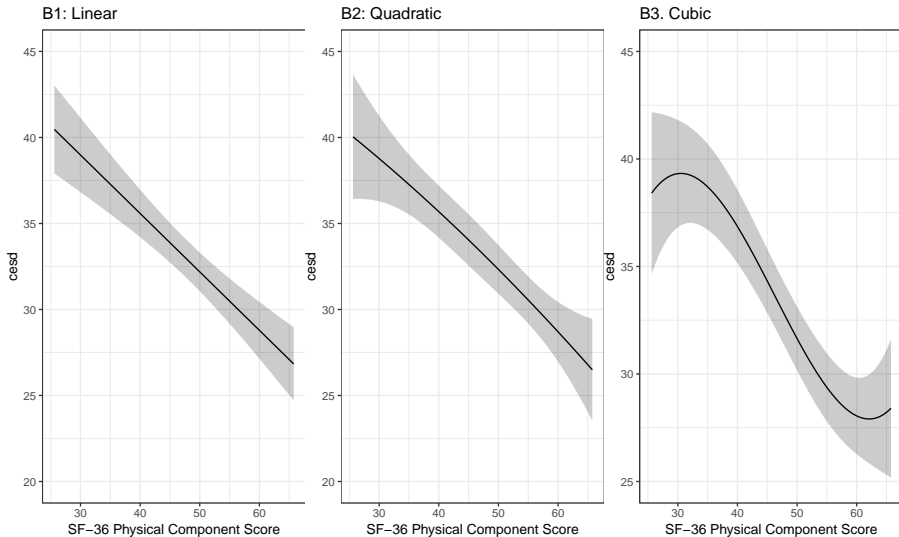


## Code to plot polynomial fits with Predict

```
p1 <- ggplot(Predict(mod_B1)) + ggtitle("B1: Linear")
p2 <- ggplot(Predict(mod_B2)) + ggtitle("B2: Quadratic")
p3 <- ggplot(Predict(mod_B3)) + ggtitle("B3. Cubic")

p1 + p2 + p3
```

# Visualizing the polynomial fits with Predict



# Splines

- A **linear spline** is a continuous function formed by connecting points (called **knots** of the spline) by line segments.
- A **restricted cubic spline** is a way to build highly complicated curves into a regression equation in a fairly easily structured way.
- A restricted cubic spline is a series of polynomial functions joined together at the knots.
  - Such a spline gives us a way to flexibly account for non-linearity without over-fitting the model.
  - Restricted cubic splines can fit many different types of non-linearities.
  - Specifying the number of knots is all you need to do in R to get a reasonable result from a restricted cubic spline.

The most common choices are 3, 4, or 5 knots.

- 3 Knots, 2 degrees of freedom, allows the curve to “bend” once.
- 4 Knots, 3 degrees of freedom, lets the curve “bend” twice.
- 5 Knots, 4 degrees of freedom, lets the curve “bend” three times.

## Fitting Restricted Cubic Splines with `ols`

Let's consider a restricted cubic spline model for `cesd` based on `pcs` with:

- 3 knots in `modC3`, 4 knots in `modC4`, and 5 knots in `modC5`

```
dd <- datadist(helpdat)
options(datadist = "dd")

mod_C3 <- ols(cesd ~ rcs(pcs, 3),
              data = helpdat, x = TRUE, y = TRUE)
mod_C4 <- ols(cesd ~ rcs(pcs, 4),
              data = helpdat, x = TRUE, y = TRUE)
mod_C5 <- ols(cesd ~ rcs(pcs, 5),
              data = helpdat, x = TRUE, y = TRUE)
```

## Model C3 (3-knot spline in pcs)

```
> mod_C3
```

```
Linear Regression Model
```

```
ols(formula = cesd ~ rcs(pcs, 3), data = helpdat, x = TRUE, y = TRUE)
```

		Model	Likelihood	Discrimination
			Ratio Test	Indexes
Obs	453	LR chi2	40.79	R2 0.086
sigma11	0.9901	d.f.	2	R2 adj 0.082
d.f.	450	Pr(> chi2)	0.0000	g 4.206

```
Residuals
```

	Min	1Q	Median	3Q	Max
	-28.3462	-7.7005	0.5098	8.6376	29.8454

	Coef	S.E.	t	Pr(> t )
Intercept	47.3631	4.7053	10.07	<0.0001
pcs	-0.2908	0.1187	-2.45	0.0146
pcs'	-0.0624	0.1363	-0.46	0.6471

## Model C4 (4-knot spline in pcs)

```
> mod_C4
Linear Regression Model

ols(formula = cesd ~ rcs(pcs, 4), data = helpdat, x = TRUE, y = TRUE)


```

		Model Likelihood		Discrimination	
		Ratio Test		Indexes	
Obs	453	LR chi2	51.31	R2	0.107
sigma	11.8648	d.f.	3	R2 adj	0.101
d.f.	449	Pr(> chi2)	0.0000	g	4.590

```


Residuals


```

	Min	1Q	Median	3Q	Max
	-28.3147	-8.2830	0.8559	8.8866	26.5458

```



```

	Coef	S.E.	t	Pr(> t )
Intercept	33.3298	6.5742	5.07	<0.0001
pcs	0.1464	0.1856	0.79	0.4308
pcs'	-1.4383	0.4497	-3.20	0.0015
pcs''	6.2561	1.9076	3.28	0.0011



## Model C5 (5-knot spline in pcs)

```
> mod_C5
Linear Regression Model

ols(formula = cesd ~ rcs(pcs, 5), data = helpdat, x = TRUE, y = TRUE)


```

		Model Likelihood		Discrimination	
		Ratio Test		Indexes	
Obs	453	LR chi2	54.64	R2	0.114
sigma	1.8345	d.f.	4	R2 adj	0.106
d.f.	448	Pr(> chi2)	0.0000	g	4.744

```

Residuals


```

	Min	1Q	Median	3Q	Max
	-29.396	-7.928	1.016	8.762	26.974

```


```

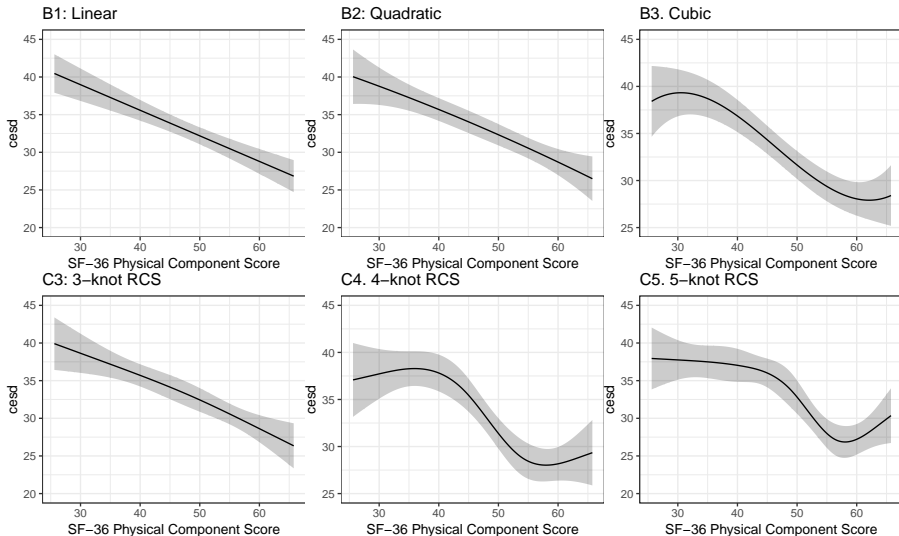
	Coef	S.E.	t	Pr(> t )
Intercept	39.0631	7.8282	4.99	<0.0001
pcs	-0.0436	0.2332	-0.19	0.8517
pcs'	-0.2952	1.0079	-0.29	0.7697
pcs''	-3.1835	4.8079	-0.66	0.5082
pcs'''	14.4216	8.3721	1.72	0.0857

## Code to plot all six fits

```
p1 <- ggplot(Predict(mod_B1)) + ggtitle("B1: Linear")
p2 <- ggplot(Predict(mod_B2)) + ggtitle("B2: Quadratic")
p3 <- ggplot(Predict(mod_B3)) + ggtitle("B3: Cubic")
p4 <- ggplot(Predict(mod_C3)) + ggtitle("C3: 3-knot RCS")
p5 <- ggplot(Predict(mod_C4)) + ggtitle("C4: 4-knot RCS")
p6 <- ggplot(Predict(mod_C5)) + ggtitle("C5: 5-knot RCS")

(p1 + p2 + p3) / (p4 + p5 + p6)
```

# Visualizing the fits better?



## Which of these models looks better?

- Compare our six models for the `cesd` to `pcs` association
- I used `set.seed(432)` then `validate(mod_B1)` etc.

Model	Index-Corrected $R^2$	Corrected MSE
B1 (linear)	0.0848	143.25
B2 (quadratic)	0.0752	142.49
B3 (cubic)	0.0909	143.73
C3 (3-knot RCS)	0.0732	143.31
C4 (4-knot RCS)	0.0870	144.00
C5 (5-knot RCS)	0.0984	141.44

- So which model has the best (validated) summaries?
- We'd need to look at residual plots, too, of course.

## Section 4

### Data Spending: Non-Linearity Prior to Fits

## Spending degrees of freedom wisely

- Suppose we have a data set with many possible predictors, and minimal theory or subject matter knowledge to guide us.
- We might want our final inferences to be as unbiased as possible. To accomplish this, we have to pay a penalty (in terms of degrees of freedom) for any “peeks” we make at the data in advance of fitting a model.
- So that rules out a lot of decision-making about non-linearity based on looking at the data, if our sample size isn't much larger than 15 times the number of predictors we're considering including in our model.
- In our case, we have  $n = 453$  observations on 6 candidate predictors.
- In addition, adding non-linearity to our model costs additional degrees of freedom.
- What can we do?

## Spearman's $\rho^2$ plot: A smart first step?

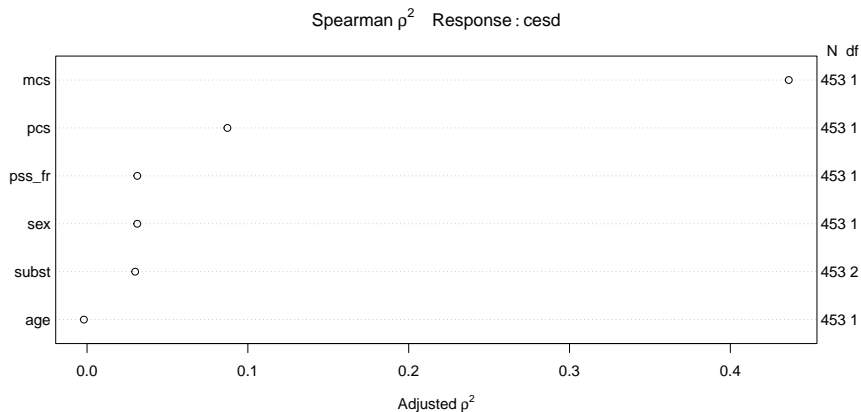
Spearman's  $\rho^2$  is an indicator (not a perfect one) of potential predictive punch, but doesn't give away the game.

- Idea: Perhaps we should focus our efforts re: non-linearity on predictors that score better on this measure.

```
spear_cesd <- spearman2(cesd ~ mcs + subst + pcs +  
                        age + sex + pss_fr,  
                        data = helpdat)
```

# Spearman's $\rho^2$ Plot

```
plot(spear_cesd)
```





## Conclusions from Spearman $\rho^2$ Plot

- `mcs` is the most attractive candidate for a non-linear term, as it packs the most potential predictive punch, so if it does turn out to need non-linear terms, our degrees of freedom will be well spent.
  - This **does not** mean that `mcs` actually needs a non-linear term, or will show meaningfully better results if a non-linear term is included. We'd have to fit a model with and without non-linearity in `mcs` to know that.
  - Non-linearity will often take the form of a product term, a polynomial term, or a restricted cubic spline.
- `pcs`, also quantitative, has the next most potential predictive punch
- these are followed by `pss_fr` and `sex`.

# Grim Reality

With 453 observations (452 df) we should be thinking about models with modest numbers of regression inputs.

- Non-linear terms (polynomials, splines) just add to the problem, as they need additional df to be estimated.

In this case, we might choose to include non-linear terms in just two or three variables (and that's it) and even that would be tough to justify with this modest sample size.

# Contents of spear\_cesd

spear\_cesd

Spearman rho<sup>2</sup>      Response variable: cesd

	rho2	F	df1	df2	P	Adjusted rho2	n
mcs	0.438	350.89	1	451	0.0000	0.436	453
subst	0.034	7.97	2	450	0.0004	0.030	453
pcs	0.089	44.22	1	451	0.0000	0.087	453
age	0.000	0.12	1	451	0.7286	-0.002	453
sex	0.033	15.56	1	451	0.0001	0.031	453
pss_fr	0.033	15.57	1	451	0.0001	0.031	453

# Proposed New Model

Fit a model to predict `cesd` using:

- a 5-knot spline on `mcs`
- a 3-knot spline on `pcs`
- a linear term on `pss_fr`
- a linear term on `age`
- an interaction of `sex` with the main effect of `mcs` (restricting our model so that terms that are non-linear in both `sex` and `mcs` are excluded), and
- a main effect of `subst`

Perhaps more than we can reasonably do with 453 observations, but let's see how it looks.

## Our new model mod2

```
dd <- datadist(helpdat)
options(datadist = "dd")

mod2 <- ols(cesd ~ rcs(mcs, 5) + rcs(pcs, 3) + sex +
            mcs %ia% sex + pss_fr + age + subst,
            data = helpdat, x = TRUE, y = TRUE)
```

- %ia% tells R to fit an interaction term with sex and the main effect of mcs.
- We have to include sex as a main effect for the interaction term (%ia%) to work here.

# Our new, more complex model mod2

```
> mod2
Linear Regression Model

ols(formula = cesd ~ rcs(mcs, 5) + rcs(pcs, 3) + sex + mcs %ia%
     sex + pss_fr + age + subst, data = helpdat, x = TRUE, y = TRUE)


```

		Model Likelihood		Discrimination
		Ratio Test		Indexes
Obs	453	LR chi2	349.44	R2 0.538
sigma	8.6248	d.f.	12	R2 adj 0.525
d.f.	440	Pr(> chi2)	0.0000	g 10.439

```

Residuals


```

	Min	1Q	Median	3Q	Max
	-26.7893	-5.9000	0.1545	5.5884	26.1304

```


```

	Coef	S.E.	t	Pr(> t )
Intercept	76.3346	6.2540	12.21	<0.0001
mcs	-0.9306	0.2315	-4.02	<0.0001
mcs'	1.6607	2.5040	0.66	0.5075
mcs''	-2.8854	8.3945	-0.34	0.7312
mcs'''	0.2942	7.9390	0.04	0.9705
pcs	-0.2341	0.0883	-2.65	0.0083
pcs'	-0.0151	0.1000	-0.15	0.8797
sex=male	-2.0330	2.5456	-0.80	0.4249
mcs * sex=male	-0.0129	0.0783	-0.17	0.8690
pss_fr	-0.2569	0.1046	-2.46	0.0144
age	-0.0466	0.0569	-0.82	0.4139
subst=cocaine	-2.6999	0.9965	-2.71	0.0070
subst=heroin	-2.1741	1.0677	-2.04	0.0423

# ANOVA for this model

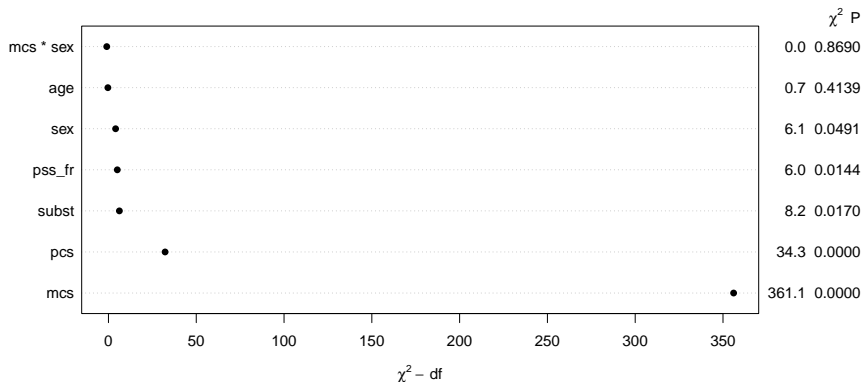
```
> anova(mod2)
```

Analysis of Variance		Response: cesd				
Factor	d.f.	Partial SS	MS	F	P	
mcs (Factor+Higher Order Factors)	5	26857.364670	5371.472934	72.21	<.0001	
All Interactions	1	2.026255	2.026255	0.03	0.8690	
Nonlinear	3	293.502251	97.834084	1.32	0.2688	
pcs	2	2548.388579	1274.194290	17.13	<.0001	
Nonlinear	1	1.705031	1.705031	0.02	0.8797	
sex (Factor+Higher Order Factors)	2	451.578352	225.789176	3.04	0.0491	
All Interactions	1	2.026255	2.026255	0.03	0.8690	
mcs * sex (Factor+Higher Order Factors)	1	2.026255	2.026255	0.03	0.8690	
pss_fr	1	448.812293	448.812293	6.03	0.0144	
age	1	49.758786	49.758786	0.67	0.4139	
subst	2	611.625952	305.812976	4.11	0.0170	
TOTAL NONLINEAR	4	293.512204	73.378051	0.99	0.4146	
TOTAL NONLINEAR + INTERACTION	5	294.601803	58.920361	0.79	0.5558	
REGRESSION	12	38058.315322	3171.526277	42.64	<.0001	
ERROR	440	32730.174744	74.386761			

- Remember that this ANOVA testing is sequential, other than the TOTALs.
- We can also plot the ANOVA results, for example...

# Plotting ANOVA results for mod2

```
plot(anova(mod2))
```





# Validation of Summary Statistics

```
> set.seed(432); validate(mod2)
```

	index.orig	training	test	optimism	index.corrected	n
R-square	0.5376	0.5513	0.5233	0.0280	0.5096	40
MSE	72.2520	69.8358	74.4984	-4.6627	76.9147	40
g	10.4392	10.5053	10.2718	0.2335	10.2056	40
Intercept	0.0000	0.0000	0.7893	-0.7893	0.7893	40
Slope	1.0000	1.0000	0.9751	0.0249	0.9751	40

## summary results for mod2

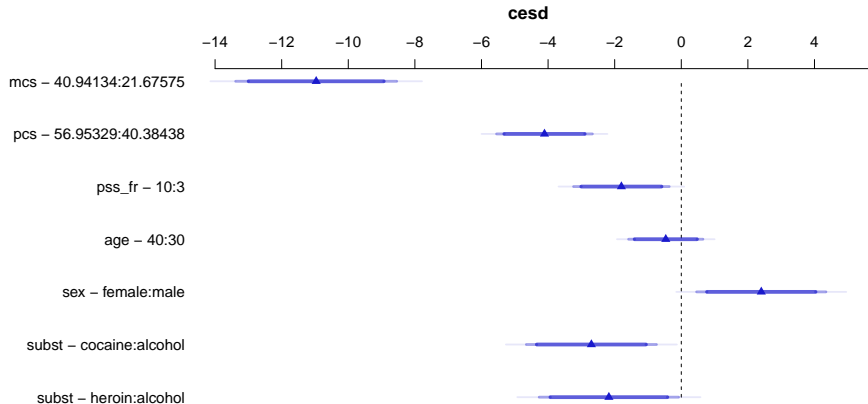
```
> summary(mod2)
```

Effects	Response : cesd						
Factor	Low	High	Diff.	Effect	S.E.	Lower 0.95	Upper 0.95
mcs	21.676	40.941	19.266	-10.96400	1.23340	-13.38800	-8.539800
pcs	40.384	56.953	16.569	-4.10790	0.73381	-5.55010	-2.665700
pss_fr	3.000	10.000	7.000	-1.79860	0.73225	-3.23780	-0.359500
age	30.000	40.000	10.000	-0.46552	0.56918	-1.58420	0.653130
sex - female:male	2.000	1.000	NA	2.40260	0.99054	0.45577	4.349300
subst - cocaine:alcohol	1.000	2.000	NA	-2.69990	0.99647	-4.65830	-0.741430
subst - heroin:alcohol	1.000	3.000	NA	-2.17410	1.06770	-4.27250	-0.075632

Adjusted to: mcs=28.60242 sex=male

# Plot of summary results for mod2

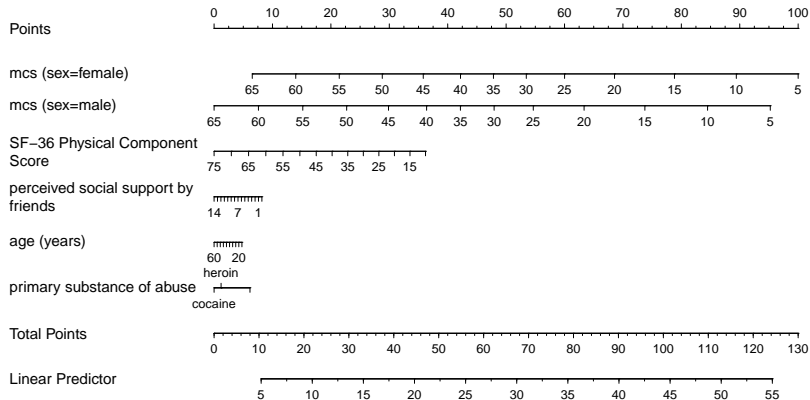
```
plot(summary(mod2))
```



Adjusted to:mcs=28.60242 sex=male

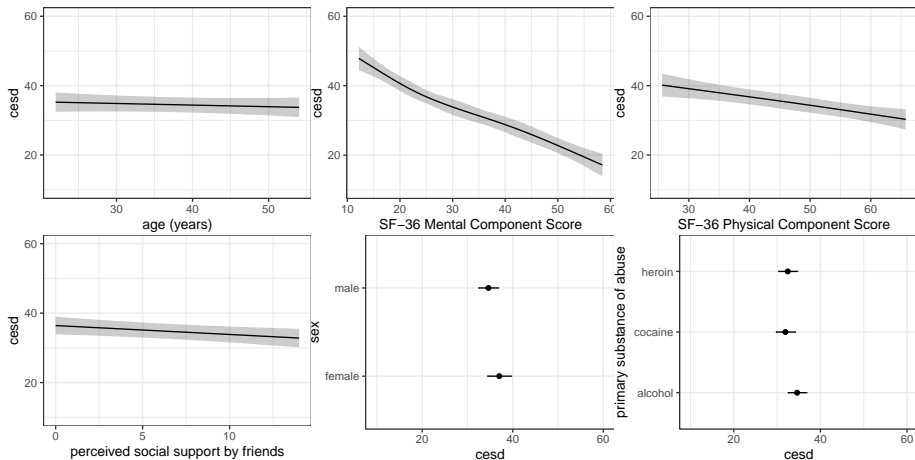
# Nomogram for mod2

```
plot(nomogram(mod2))
```



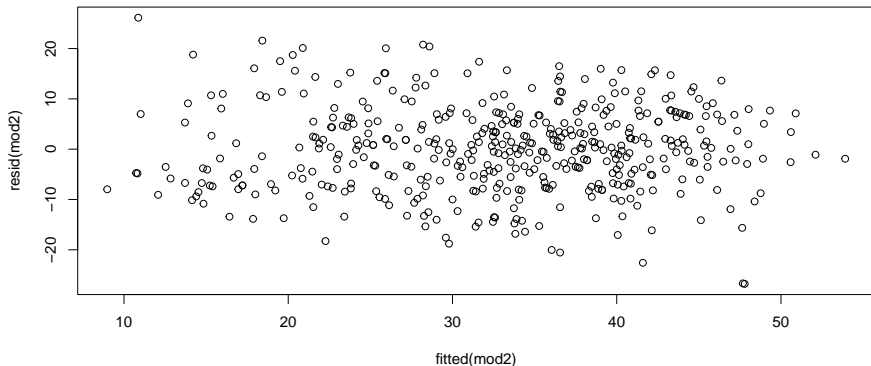
# Seeing the impact of the modeling another way

```
ggplot(Predict(mod2))
```



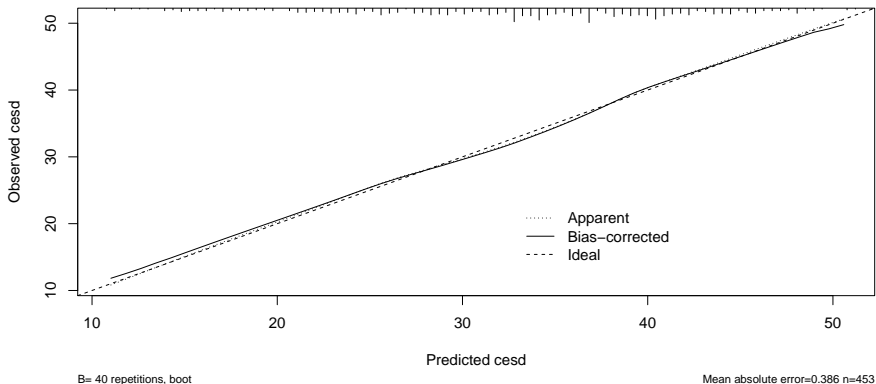
# Residuals vs. Fitted Values to check assumptions

```
plot(resid(mod2) ~ fitted(mod2))
```



# Checking the model's calibration

```
set.seed(432); plot(calibrate(mod2))
```



n=453 Mean absolute error=0.386

Mean squared error=0.19775

# Limitations of `lm` for fitting complex linear models

We can certainly assess this big, complex model using `lm`, too:

- with in-sample summary statistics like adjusted  $R^2$ , AIC and BIC,
- we can assess its assumptions with residual plots, and
- we can also compare out-of-sample predictive quality through cross-validation,

But to really delve into the details of how well this complex model works, and to help plot what is actually being fit, we'll probably want to fit the model using `ols`.

- In Project A, we expect some results that are most easily obtained using `lm` and others that are most easily obtained using `ols`.

# Next Time

- The HERS data
- Fitting a more complex linear regression model
- Adding missing data into all of this, and using multiple imputation