432 Class 05

Thomas E. Love, Ph.D.

2026-01-27

Today's Agenda

- The HELP trial, again
- Incorporating Non-Linearity into our models
  - Polynomial terms
  - Restricted Cubic Splines

Today's R Setup

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

library(janitor)
library(naniar)
library(broom); library(gt); library(patchwork)

library(haven)              ## for zapping labels
library(mosaic)             ## auto-loads mosaicData - data source

library(rms)                ## auto-loads Hmisc
library(easystats)
library(tidyverse)

theme_set(theme_bw())
```


Reminders: The HELP Study

Health Evaluation and Linkage to Primary Care (HELP) was a clinical trial of adult inpatients recruited from a detoxification unit.

- We have baseline data for each subject on several variables, including two outcomes:

Variable	Description
`cesd`	Center for Epidemiologic Studies-Depression
`cesd_hi`	`cesd` above 15 (indicates high risk)

Potential Predictors in `help1`

Variable	Description
`age`	subject age (in years)
`sex`	female (n = 107) or male (n = 346)
`subst`	substance abused (alcohol, cocaine, heroin)
`mcs`	SF-36 Mental Component Score
`pcs`	SF-36 Physical Component Score
`pss_fr`	perceived social support by friends

- See https://nhorton.people.amherst.edu/help/ for more.

`help1` data load

```
help1 <- tibble(mosaicData::HELPrct) |>
  select(id, cesd, age, sex, subst = substance, mcs, pcs, pss_fr) |>
  zap_label() |>
  mutate(across(where(is.character), as_factor),
         id = as.character(id),
         cesd_hi = factor(as.numeric(cesd >= 16)))

dim(help1); n_miss(help1)
```

```
[1] 453   9
[1] 0
```

```
head(help1, 5)
```

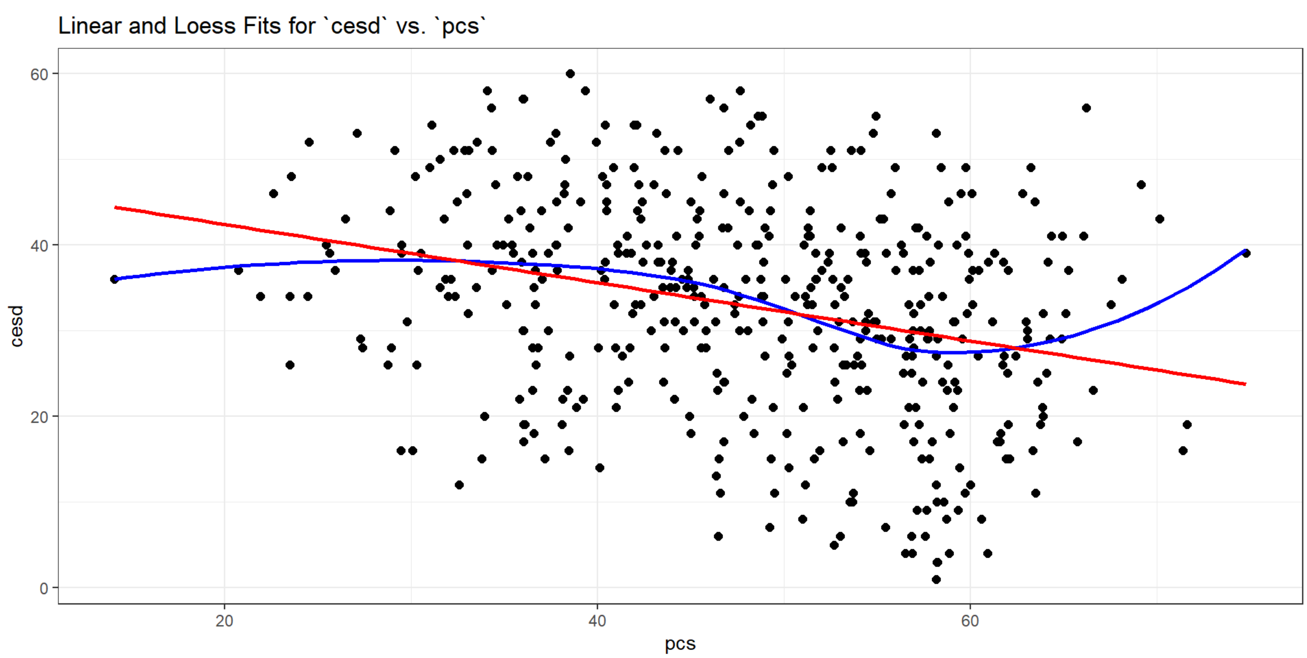
```
# A tibble: 5 × 9
  id     cesd   age sex    subst     mcs   pcs pss_fr cesd_hi
  <chr> <int> <int> <fct>  <fct>   <dbl> <dbl>  <int> <fct>
1 1        49    37 male   cocaine 25.1   58.4      0 1
2 2        30    37 male   alcohol 26.7   36.0      1 1
3 3        39    26 male   heroin   6.76  74.8     13 1
4 4        15    39 female heroin  44.0   61.9     11 0
5 5        39    32 male   cocaine 21.7   37.3     10 1
```

Can we use pcs to predict cesd?

Does the loess smooth match up well with the linear fit?

```
ggplot(help1, 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`")
```


Can we use pcs to predict cesd?



A simple linear regression: `fitA`

```
dd <- datadist(help1); options(datadist = "dd")

fitA <- ols(cesd ~ pcs, data = help1, x = TRUE, y = TRUE)

fitA$coefficients
```

```
 Intercept        pcs 
49.1673458 -0.3396495 
```

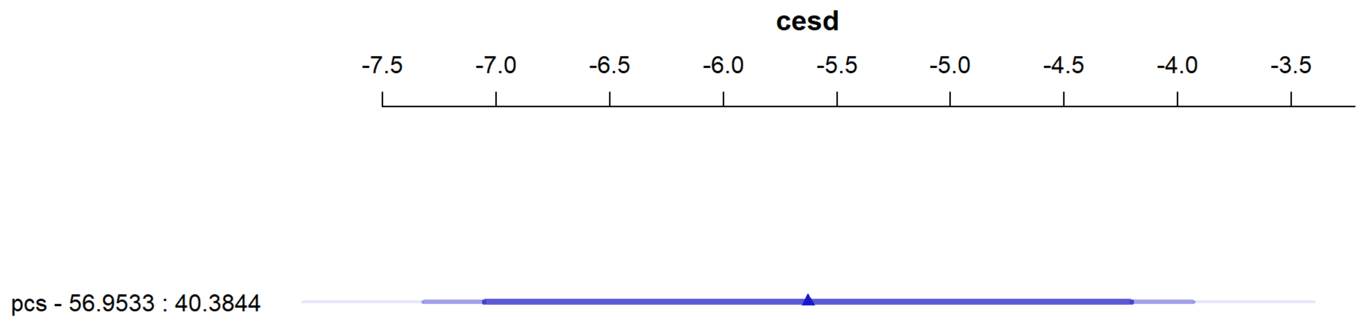

Our simple linear regression

```
fitA
```

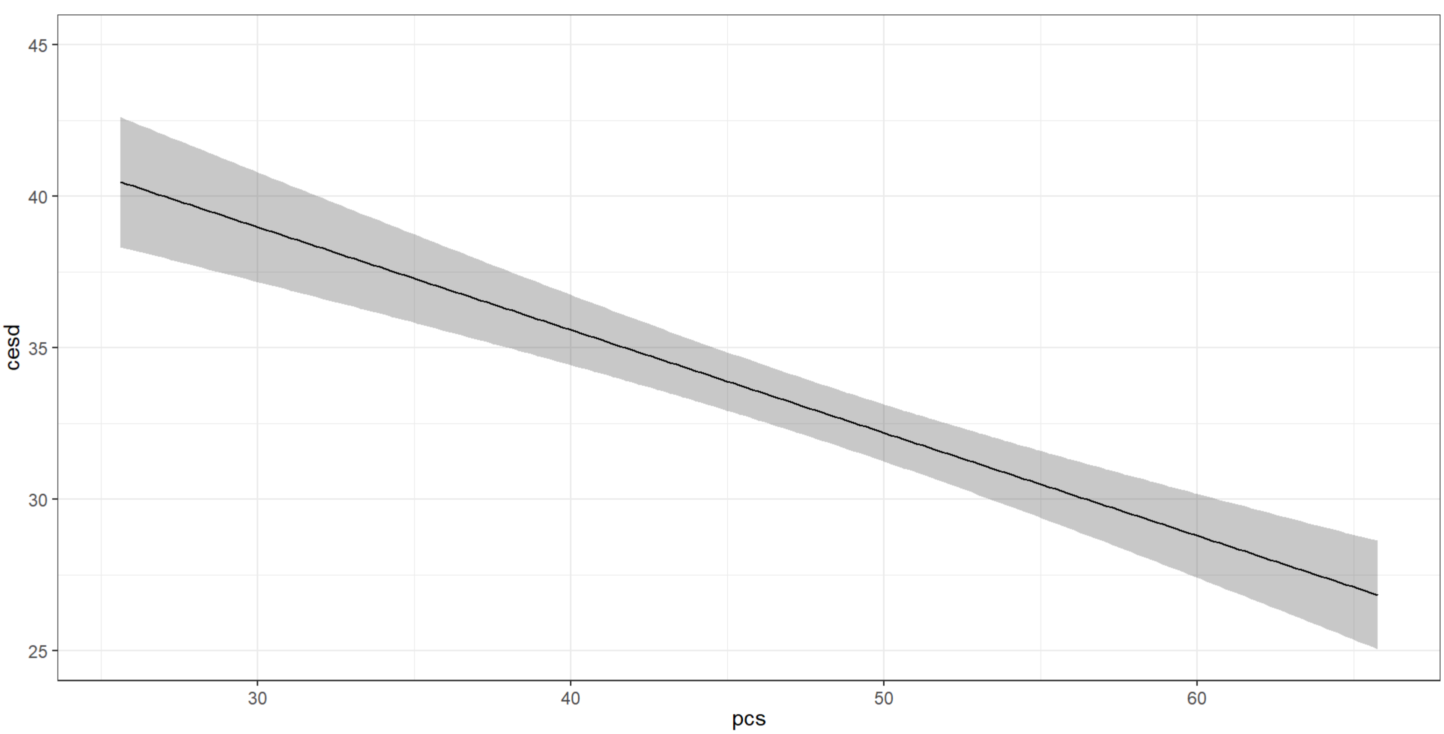
```
Linear Regression Model

ols(formula = cesd ~ pcs, data = help1, 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 
```

Effect Sizes in `fitA`

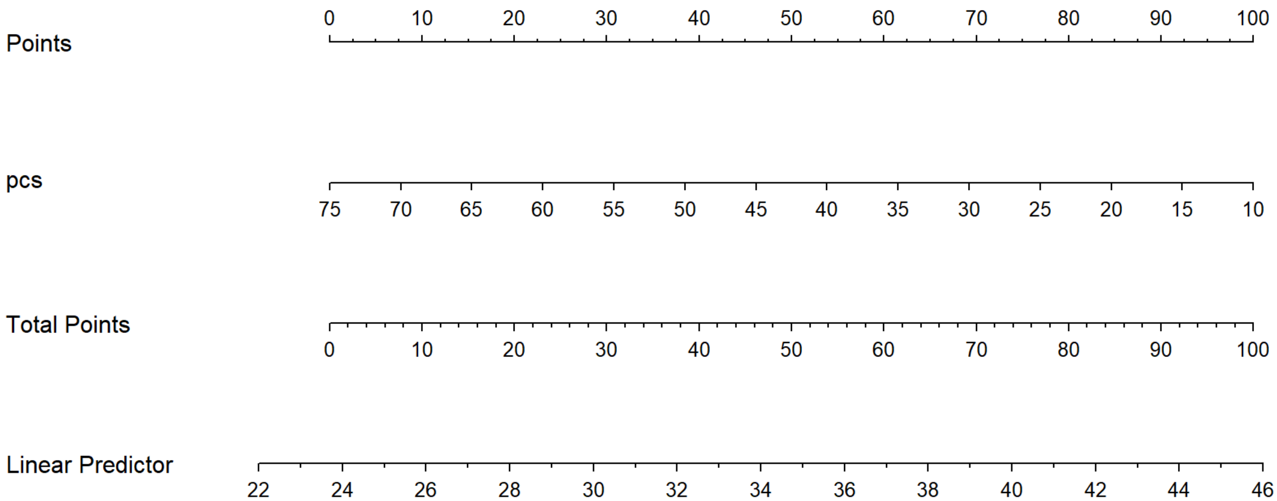
```r
plot(summary(fitA))
```




```r
ggplot(Predict(fitA, conf.int = 0.90))
```



```r
plot(nomogram(fitA))
```



Using `ols` to fit a larger model

```r
dd <- datadist(help1)
options(datadist = "dd")

fitB <- ols(cesd ~ pcs + subst + pss_fr + sex,
            data = help1, x = TRUE, y = TRUE)

fitB$coefficients
```

```
  Intercept          pcs subst=cocaine  subst=heroin       pss_fr 
 53.7511151   -0.2574023    -3.8664109     0.2322071   -0.5370221 
   sex=male 
 -4.8446977 
```

- Can use `model_parameters()` and `model_performance()` with `fitB` or other `ols()` fits.
- We could also fit this model, naturally, using `lm()` instead.

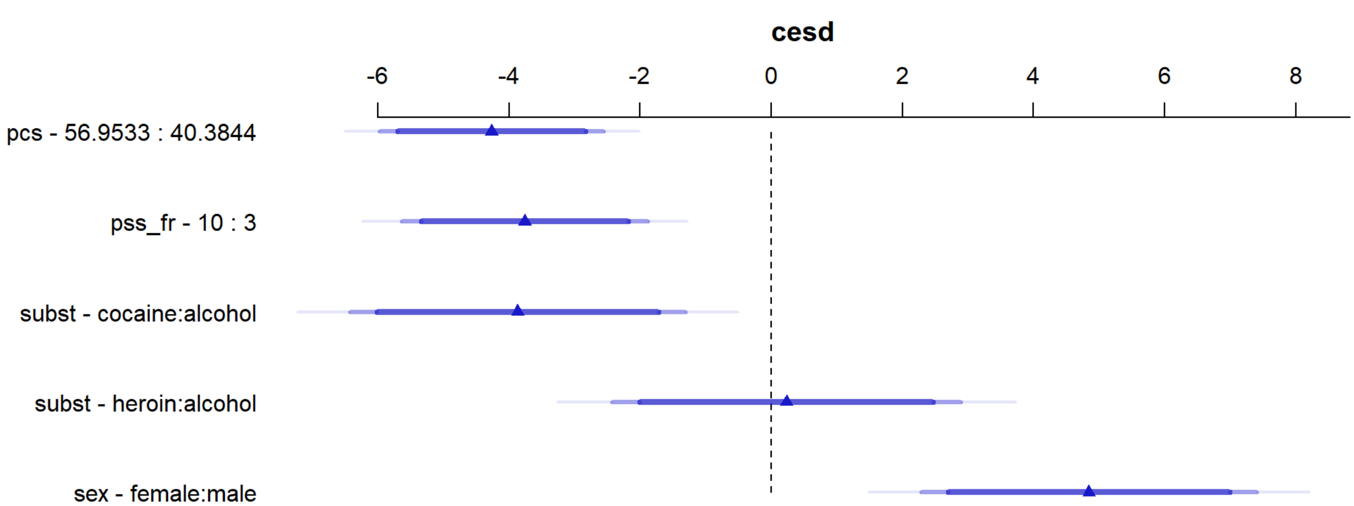
Contents of `fitB`?

```
1  fitB
```

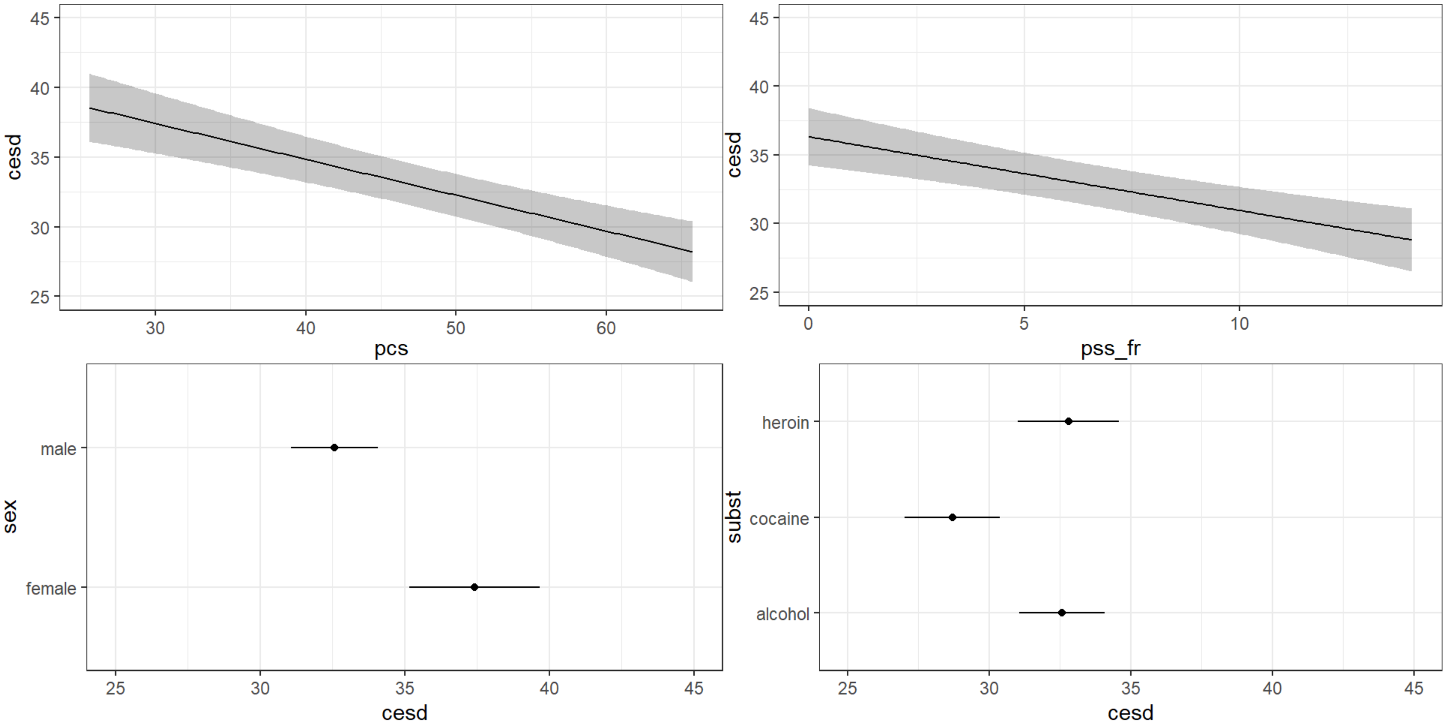
```
Linear Regression Model

ols(formula = cesd ~ pcs + subst + pss_fr + sex, data = help1,
    x = TRUE, y = TRUE)

                 Model Likelihood    Discrimination
                       Ratio Test           Indexes
Obs       453    LR chi2      76.43    R2       0.155
sigma11.5662    d.f.             5    R2 adj   0.146
d.f.      447    Pr(> chi2) 0.0000    g        5.625

Residuals

     Min       1Q   Median       3Q      Max
```


Effect Sizes in `fitB`

```
1  plot(summary(fitB))
```

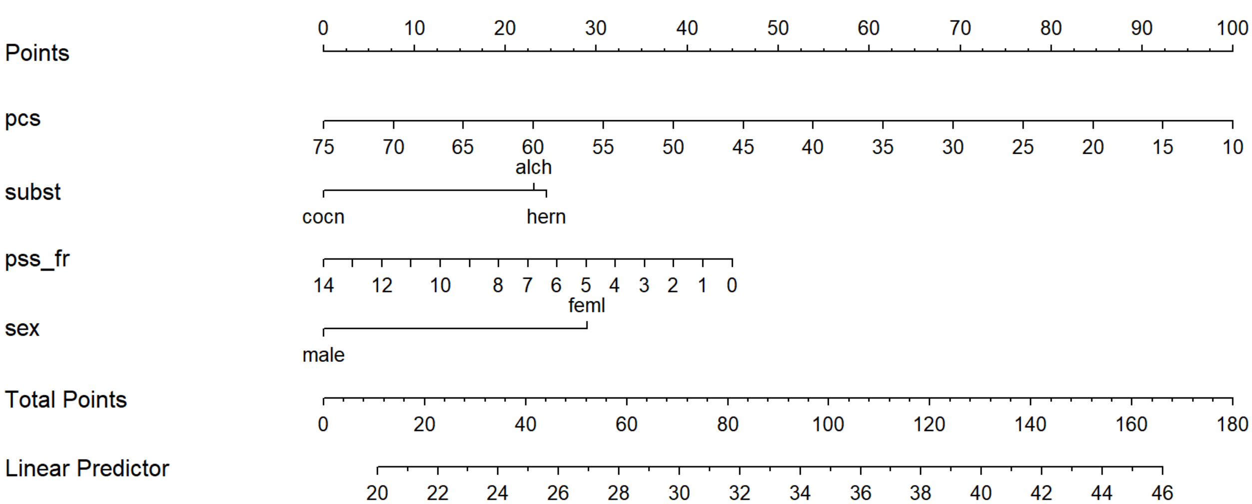


```r
ggplot(Predict(fitB, conf.int = 0.90))
```



A Nomogram for `fitB`

```r
plot(nomogram(fitB, abbrev = TRUE))
```



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

Non-Linear Terms: Polynomials

Polynomial Regression

A polynomial in the variable x of degree D is a linear combination of the powers of x up to D. For example:

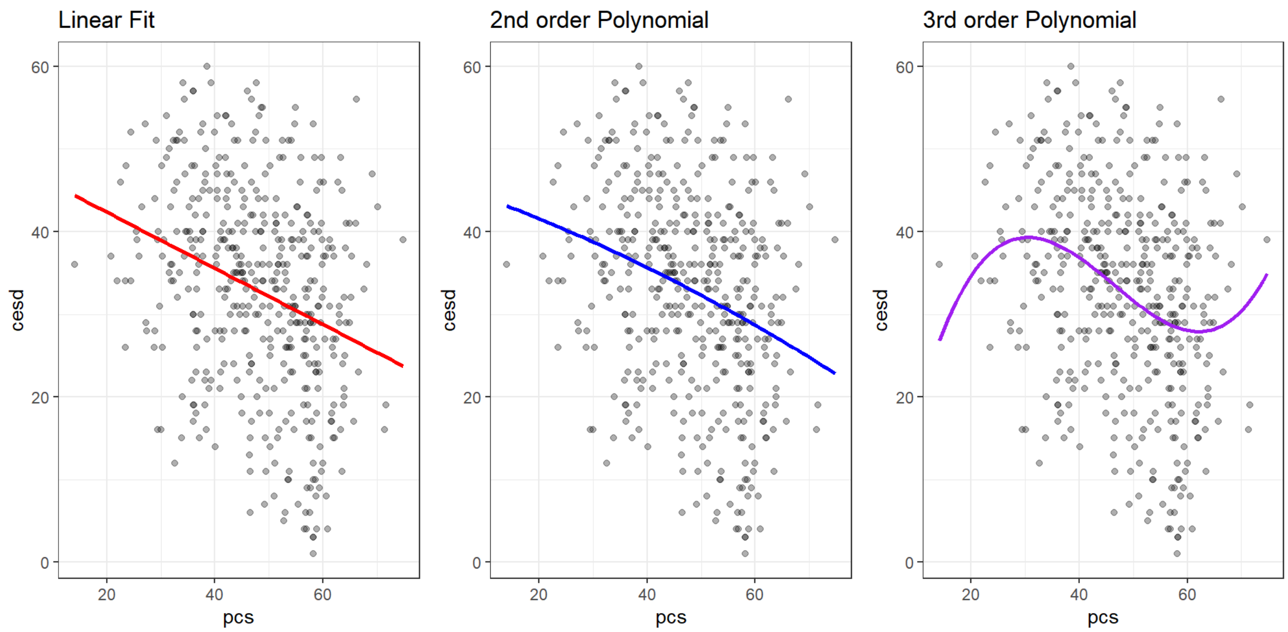
- Linear: $y = \beta_0 + \beta_1 x$
- Quadratic: $y = \beta_0 + \beta_1 x + \beta_2 x^2$
- Cubic: $y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3$
- Quartic: $y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \beta_4 x^4$

Fitting such a model creates a **polynomial regression**.

Plotting the Polynomials

```
p1 <- ggplot(help1, aes(x = pcs, y = cesd)) +
    geom_point(alpha = 0.3) +
    geom_smooth(formula = y ~ x, method = "lm",
                col = "red", se = FALSE) +
    labs(title = "Linear Fit")

p2 <- ggplot(help1, aes(x = pcs, y = cesd)) +
    geom_point(alpha = 0.3) +
    geom_smooth(formula = y ~ poly(x, 2), method = "lm",
                col = "blue", se = FALSE) +
    labs(title = "2nd order Polynomial")

p3 <- ggplot(help1, aes(x = pcs, y = cesd)) +
    geom_point(alpha = 0.3) +
    geom_smooth(formula = y ~ poly(x, 3), method = "lm",
                col = "purple", se = FALSE) +
    labs(title = "3rd order Polynomial")

```

Plotting the Polynomials



Adding a polynomial in `pcs`

Can we predict `cesd` with a polynomial in `pcs`?

Yes, with `ols()` and `pol()`, as follows:

```
fitA <- ols(cesd ~ pcs, data = help1, x = TRUE, y = TRUE)
fitA_2 <- ols(cesd ~ pol(pcs,2), data = help1, x = TRUE, y = TRUE)
fitA_3 <- ols(cesd ~ pol(pcs,3), data = help1, x = TRUE, y = TRUE)
```

With `lm()`, we use `poly()` instead of `pol()`…

```
lmfitA <- lm(cesd ~ pcs, data = help1)
lmfitA_2 <- lm(cesd ~ poly(pcs,2), data = help1)
lmfitA_3 <- lm(cesd ~ poly(pcs,3), data = help1)
```

Raw vs. Orthogonal Polynomials

Predict `cesd` using `pcs` with a “raw polynomial of degree 2.”

```
(temp1 <- lm(cesd ~ pcs + I(pcs^2), data = help1))
```

```
Call:
lm(formula = cesd ~ pcs + I(pcs^2), data = help1)

Coefficients:
(Intercept)          pcs     I(pcs^2)
  46.400713    -0.213627    -0.001356
```

Predicted `cesd` for `pcs` = 40 is

```
cesd = 46.400713 - 0.213627 (40) - 0.001356 (40^2)
     = 46.400713 - 8.545080 - 2.169600
     = 35.686
```


Does the raw polynomial match our expectations?

```
temp1 <- lm(cesd ~ pcs + I(pcs^2), data = help1)

augment(temp1, newdata = tibble(pcs = 40)) |>
  gt() |> tab_options(table.font.size = 24)
```

pcs	.fitted
40	35.6856

This matches our “by hand” calculation.

- But it turns out most regression models use *orthogonal* rather than raw polynomials…

Fitting an Orthogonal Polynomial

Predict `cesd` using `pcs` with an *orthogonal* polynomial of degree 2.

```
(temp2 <- lm(cesd ~ poly(pcs,2), data = help1))
```

```
Call:
lm(formula = cesd ~ poly(pcs, 2), data = help1)

Coefficients:
  (Intercept)  poly(pcs, 2)1  poly(pcs, 2)2
       32.848        -77.876         -3.944
```

This looks very different from our previous version of the model. What happens when we make a prediction, though?

Orthogonal Polynomial Model Prediction

Remember that in our raw polynomial model, our "by hand" and "using R" calculations each predicted `cesd` for a subject with `pcs` = 40 to be 35.686.

What happens with the orthogonal polynomial model `temp2`?

```
augment(temp2, newdata = data.frame(pcs = 40)) |>
  gt() |> tab_options(table.font.size = 24)
```

pcs	.fitted
40	35.6856

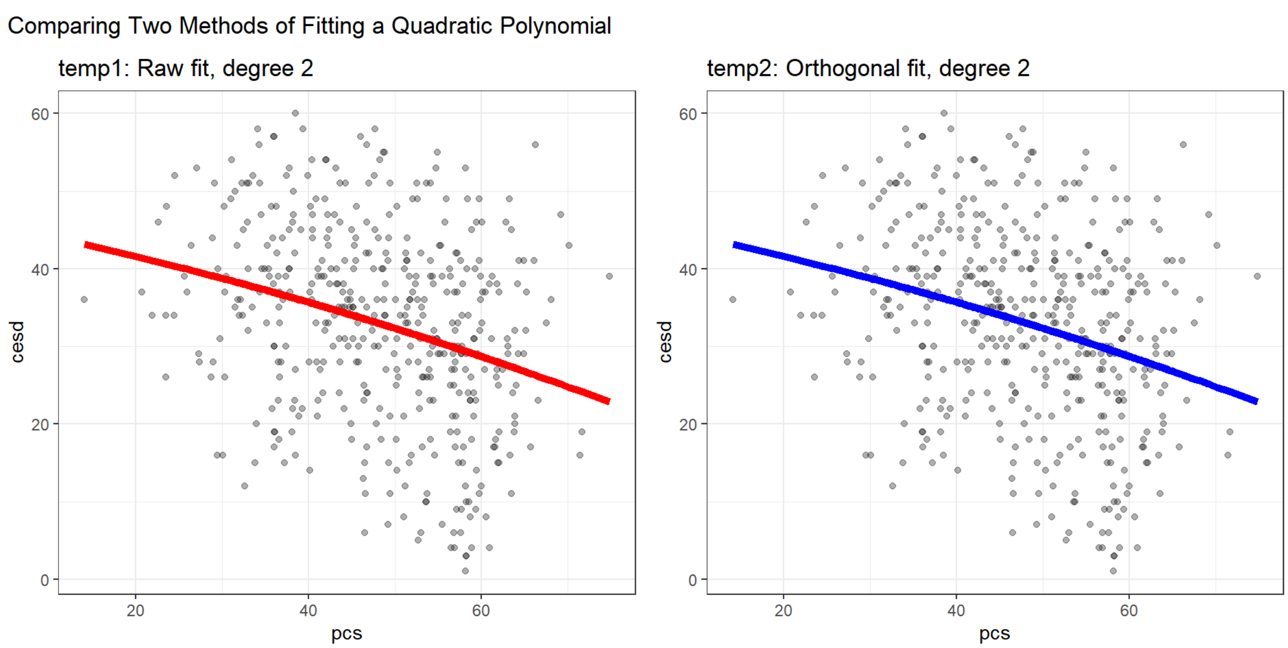
- No change in the prediction.

Fits of raw vs orthogonal polynomials

```r
temp1_aug <- augment(temp1, help1)
temp2_aug <- augment(temp2, help1)

p1 <- ggplot(temp1_aug, aes(x = pcs, y = cesd)) +
    geom_point(alpha = 0.3) +
    geom_line(aes(x = pcs, y = .fitted), col = "red", linewidth = 2) +
    labs(title = "temp1: Raw fit, degree 2")

p2 <- ggplot(temp2_aug, aes(x = pcs, y = cesd)) +
    geom_point(alpha = 0.3) +
    geom_line(aes(x = pcs, y = .fitted), col = "blue", linewidth = 2) +
    labs(title = "temp2: Orthogonal fit, degree 2")

p1 + p2 +
    plot_annotation(title = "Comparing Two Methods of Fitting a Quadratic Polynomi
```

- The two models are, in fact, identical.

Fits of raw vs orthogonal polynomials



Why use orthogonal polynomials?

- The main reason is to avoid having to include powers of our predictor that are highly collinear.
- Variance Inflation Factor assesses collinearity…

```r
rms::vif(temp1)         ## from rms package
```

```
     pcs I(pcs^2) 
54.66793 54.66793 
```

- Orthogonal polynomial terms are uncorrelated…

```r
rms::vif(temp2)
```

```
poly(pcs, 2)1 poly(pcs, 2)2 
            1             1 
```


Why orthogonal polynomials?

An **orthogonal polynomial** sets up a model design matrix and then scales those columns so that each column is uncorrelated with the others. The tradeoff is that the raw polynomial is a lot easier to explain in terms of a single equation in the simplest case.

Actually, we'll often use splines instead of polynomials, which are more flexible and require less maintenance, but at the cost of pretty much requiring you to focus on visualizing their predictions rather than their equations.

fitA with a cubic polynomial

```
dd <- datadist(help1); options(datadist = "dd")

fitA_3 <- ols(cesd ~ pol(pcs,3), data = help1, x = TRUE, y = TRUE)

fitA_3$coefficients
```

```
    Intercept           pcs         pcs^2         pcs^3 
-1.340758e+01  4.132348e+00 -1.009667e-01  7.268386e-04 
```

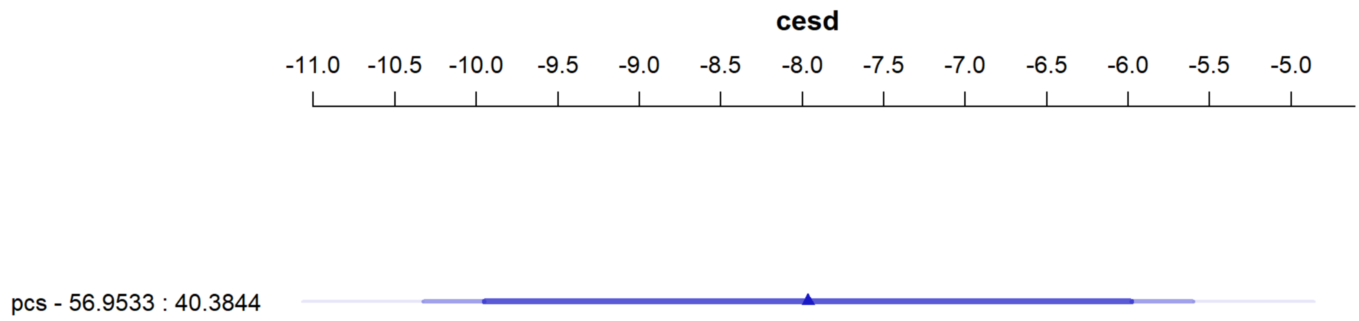

Our model fitA_3

```
fitA_3
```

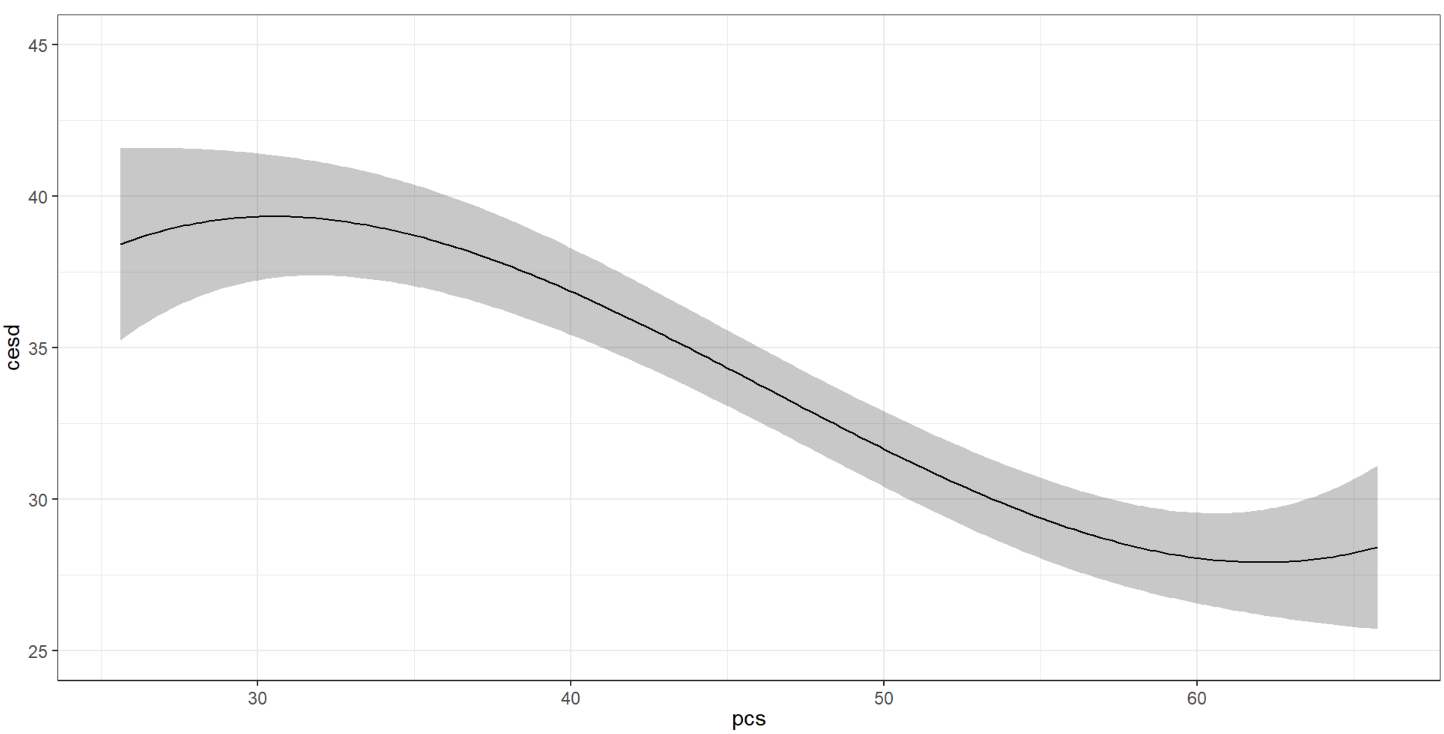
```
Linear Regression Model

ols(formula = cesd ~ pol(pcs, 3), data = help1, 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 
```

Effect Sizes in `fitA_3`

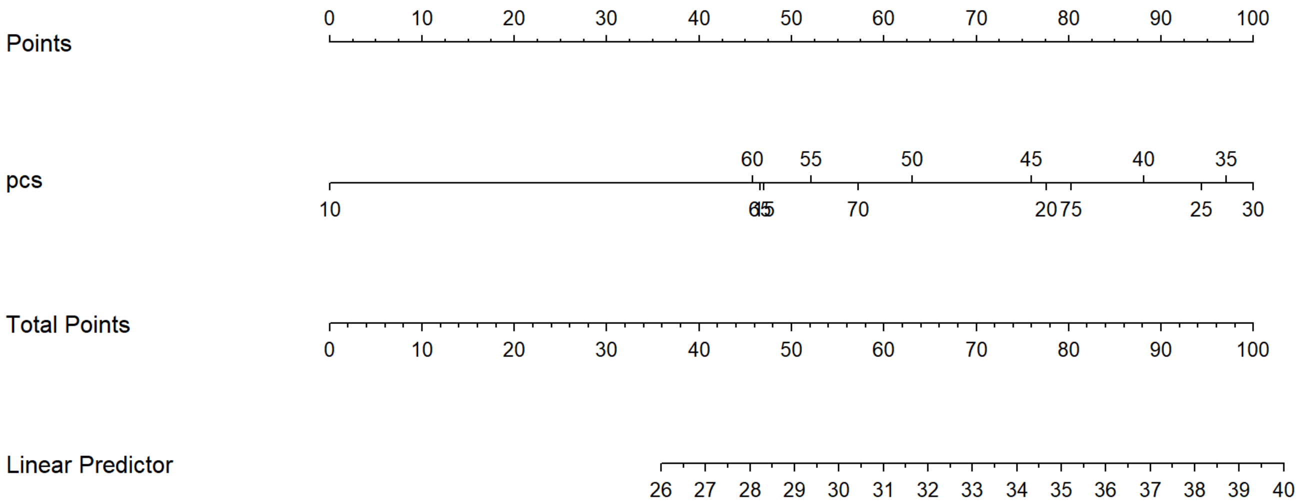
```r
plot(summary(fitA_3))
```




```r
ggplot(Predict(fitA_3, conf.int = 0.90))
```



```r
plot(nomogram(fitA_3))
```



Fitting `fitB` including a polynomial

```r
dd <- datadist(help1)
options(datadist = "dd")

fitB_3 <- ols(cesd ~ pol(pcs,3) + subst + pss_fr + sex,
              data = help1, x = TRUE, y = TRUE)

fitB_3$coefficients
```

```
   Intercept            pcs          pcs^2          pcs^3 subst=cocaine
5.2983256376  3.2271532761 -0.0794837993  0.0005770243 -3.8581390102
subst=heroin         pss_fr       sex=male
0.0455051022 -0.5127744954 -4.5981834492
```

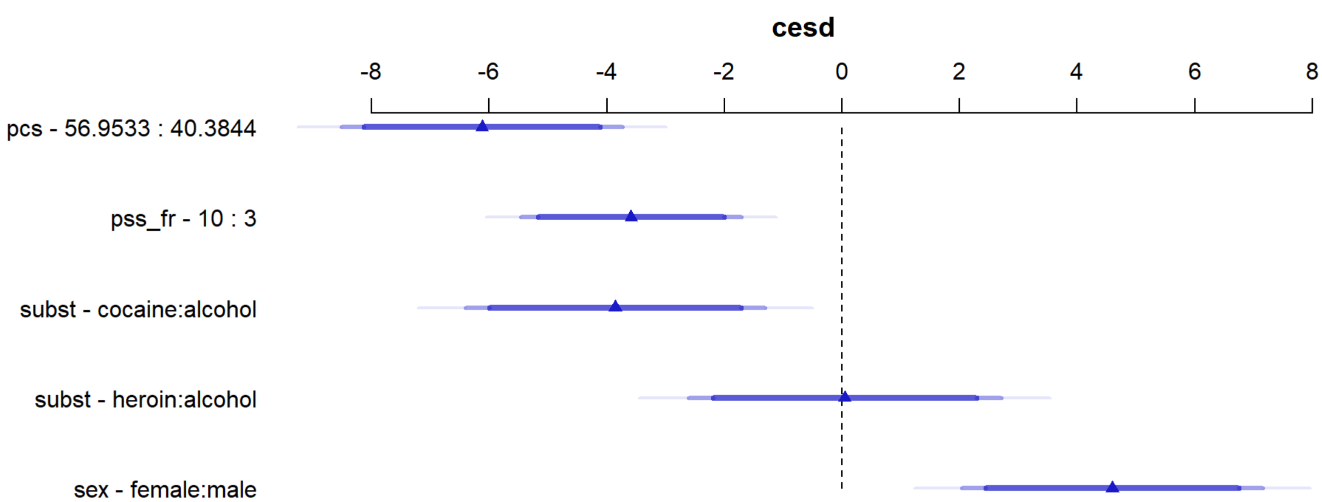
Contents of fitB_3?

```
1  fitB_3
```

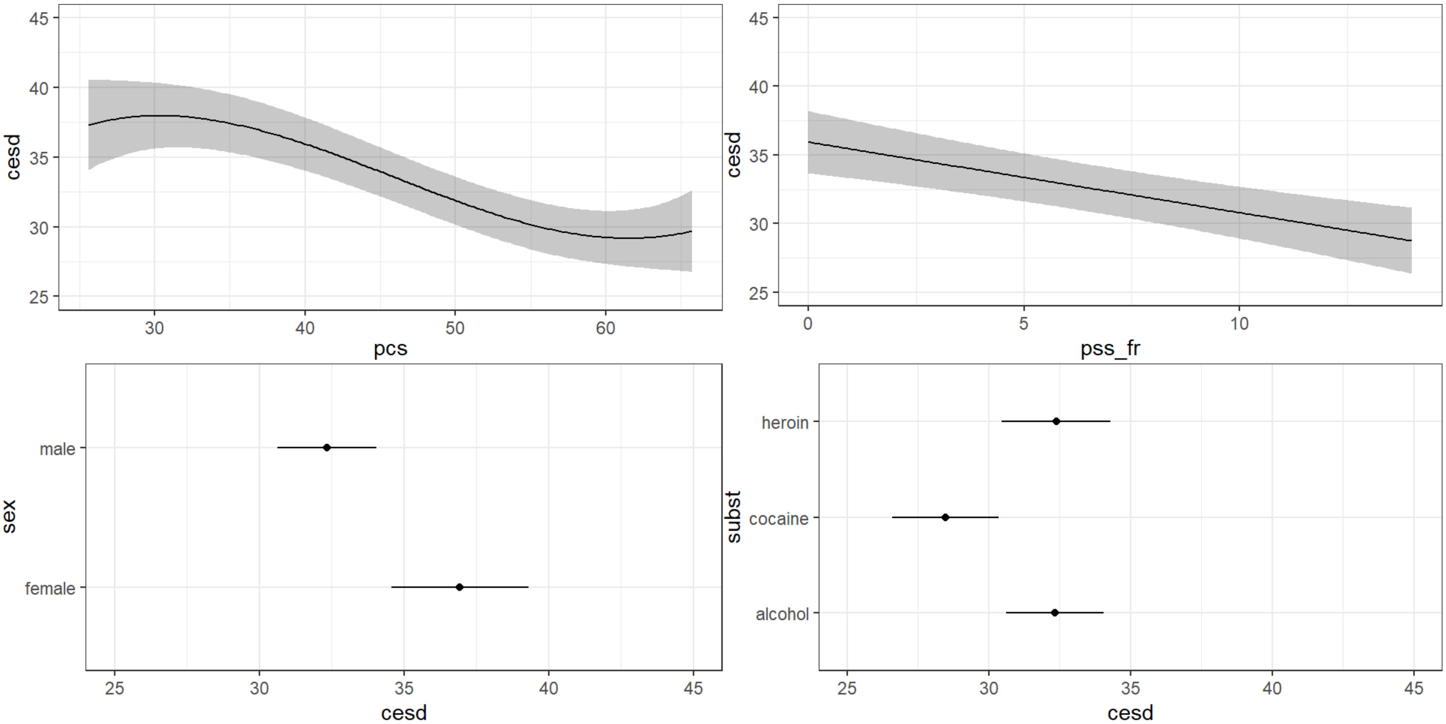
```
Linear Regression Model

ols(formula = cesd ~ pol(pcs, 3) + subst + pss_fr + sex, data = help1,
    x = TRUE, y = TRUE)

                   Model Likelihood    Discrimination
                         Ratio Test           Indexes
Obs       453     LR chi2      81.80    R2        0.165
sigma11.5236     d.f.              7    R2 adj    0.152
d.f.      445     Pr(> chi2) 0.0000    g         5.808

Residuals

     Min       1Q   Median       3Q      Max
```


Effect Sizes in fitB_3

```
1  plot(summary(fitB_3))
```

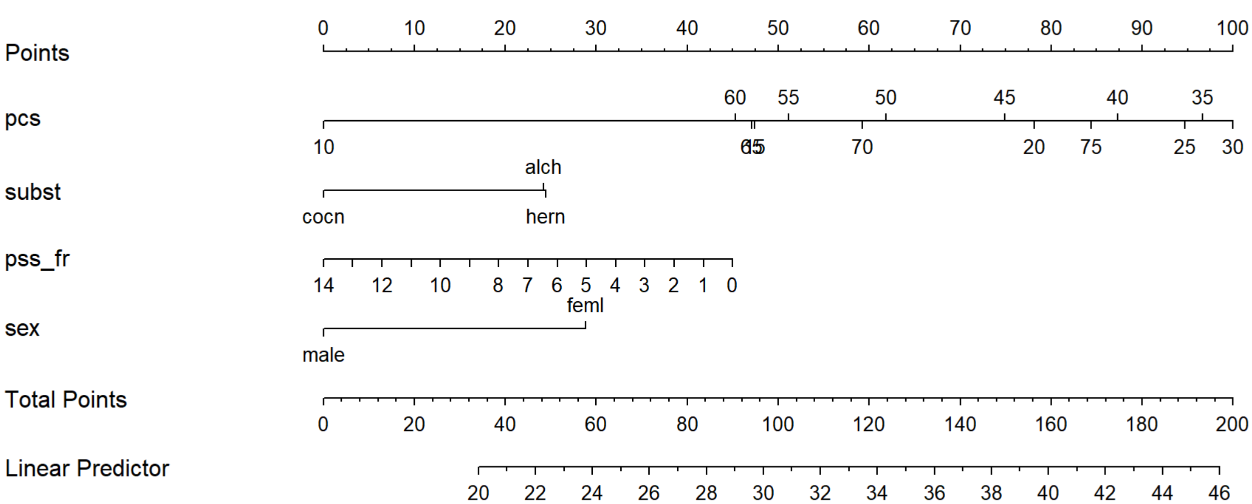


```r
ggplot(Predict(fitB_3, conf.int = 0.90))
```



A Nomogram for `fitB_3`

```r
plot(nomogram(fitB_3, abbrev = TRUE))
```



Standing Break

Non-Linear Terms: Splines

Types of 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.

How complex should our spline be?

Restricted cubic splines can fit many types of non-linearity. Specifying the number of knots (usually 3, 4 or 5) is all you must do in R to get a reasonable result from a restricted cubic spline.

- 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.

Restricted Cubic Splines with `ols`

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

- 3 knots in `fitC3`, 4 knots in `fitC4`, and 5 knots in `fitC5`

```
dd <- datadist(help1)
options(datadist = "dd")

fitC3 <- ols(cesd ~ rcs(pcs, 3),
             data = help1, x = TRUE, y = TRUE)
fitC4 <- ols(cesd ~ rcs(pcs, 4),
             data = help1, x = TRUE, y = TRUE)
fitC5 <- ols(cesd ~ rcs(pcs, 5),
             data = help1, x = TRUE, y = TRUE)
```

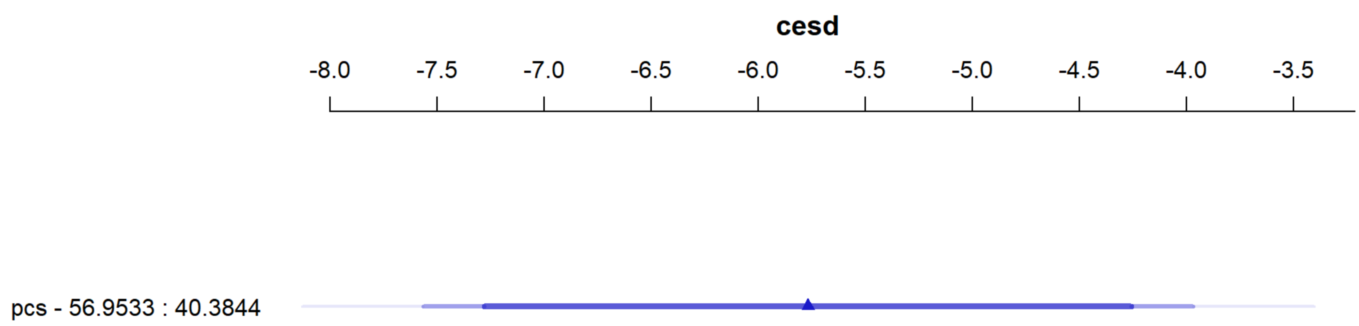

Model `fitC3` (3-knot spline in `pcs`)

```
fitC3
```

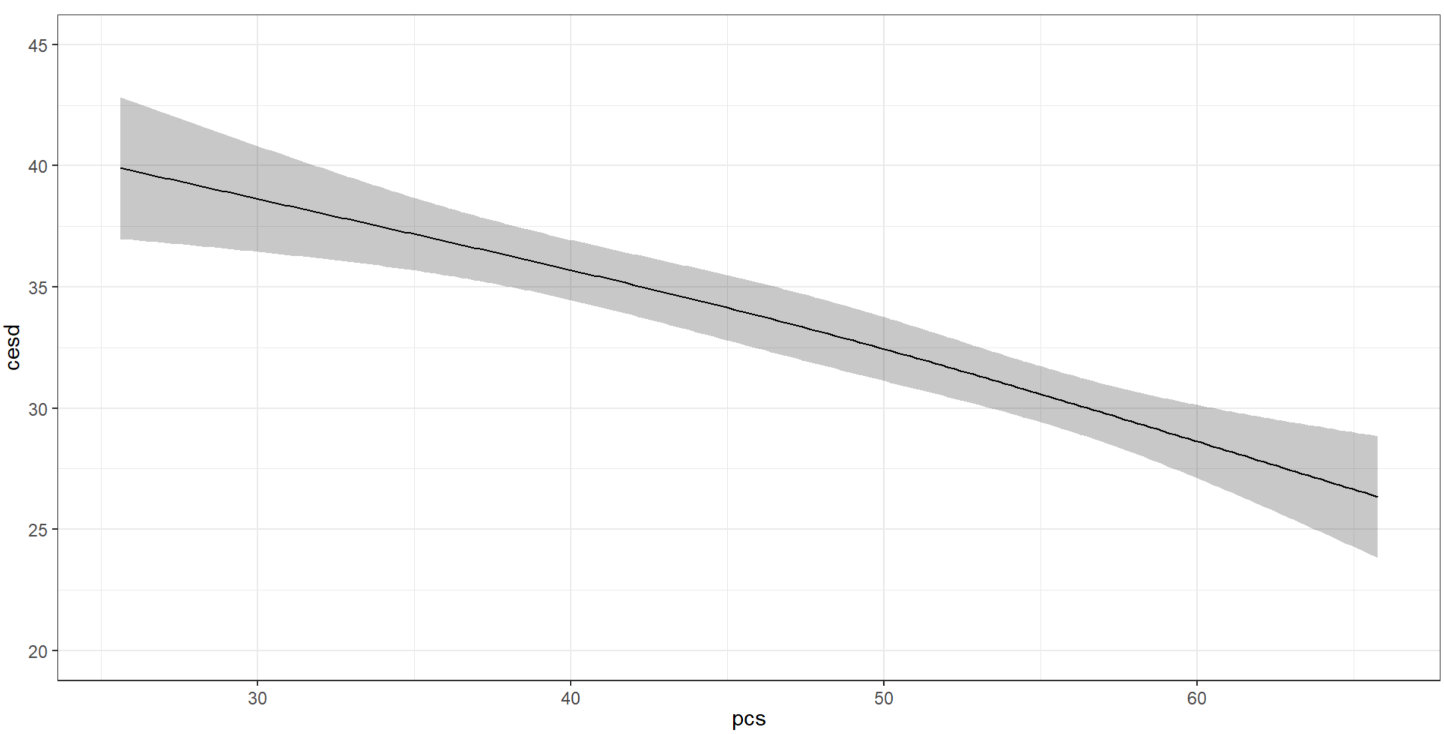
```
Linear Regression Model

ols(formula = cesd ~ rcs(pcs, 3), data = help1, x = TRUE, y = TRUE)

                 Model Likelihood    Discrimination
                       Ratio Test           Indexes
Obs       453    LR chi2     40.79    R2       0.086
sigma11.9901    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
```

Effect Sizes in `fitC3`

```
plot(summary(fitC3))
```

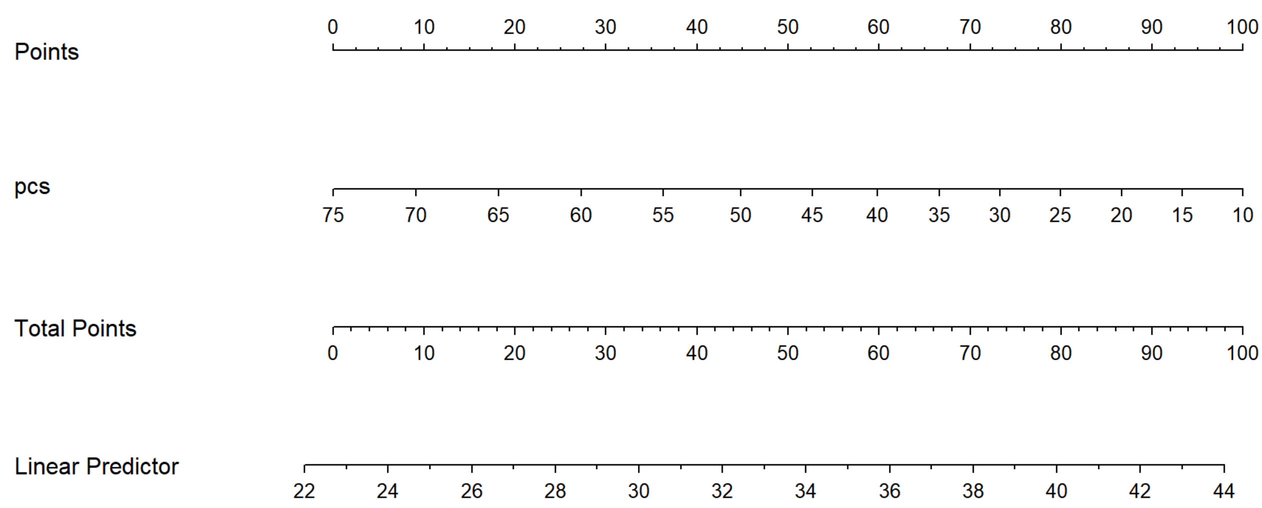



```
ggplot(Predict(fitC3, conf.int = 0.90))
```



A Nomogram for `fitC3`

```
plot(nomogram(fitC3, abbrev = TRUE))
```



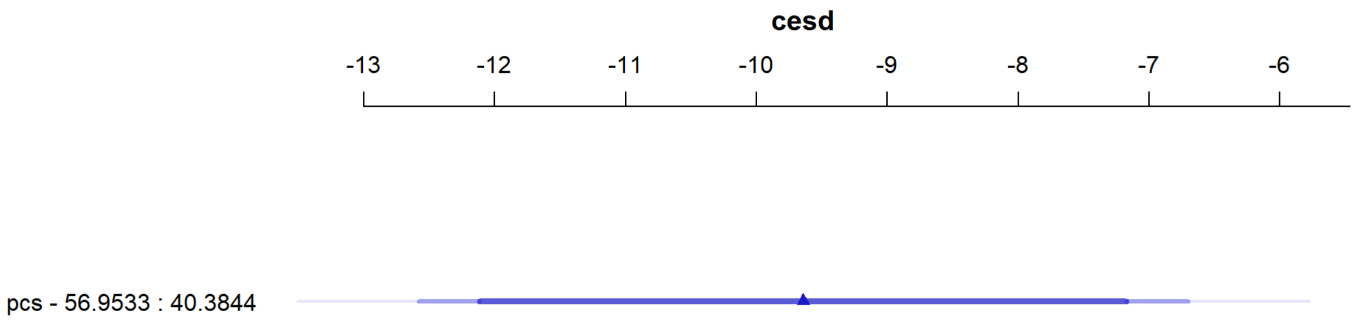
Model `fitC4` (4-knot spline in `pcs`)

```
fitC4
```

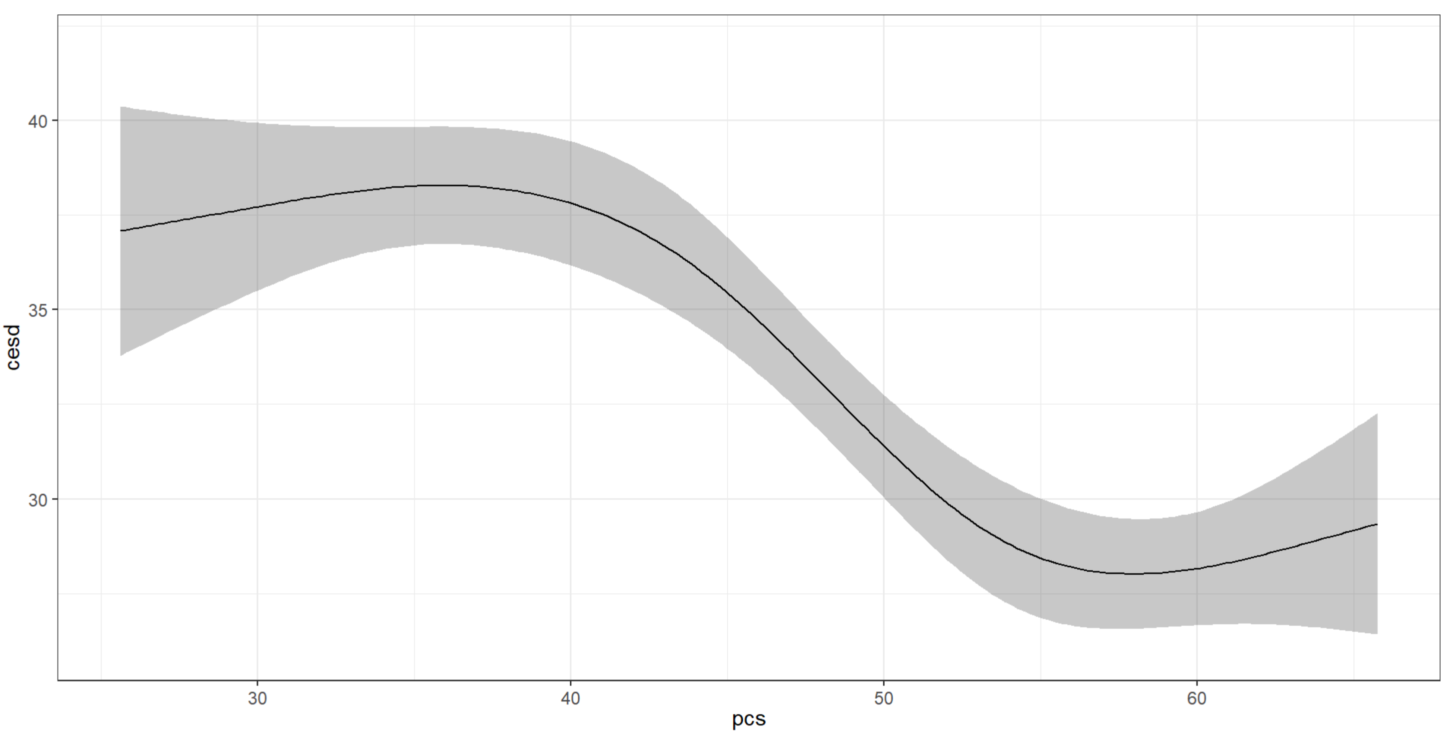
```
Linear Regression Model

ols(formula = cesd ~ rcs(pcs, 4), data = help1, x = TRUE, y = TRUE)

                  Model Likelihood    Discrimination
                        Ratio Test           Indexes
Obs       453     LR chi2      51.31    R2       0.107
sigma11.8648     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
```

Effect Sizes in `fitC4`

```r
plot(summary(fitC4))
```

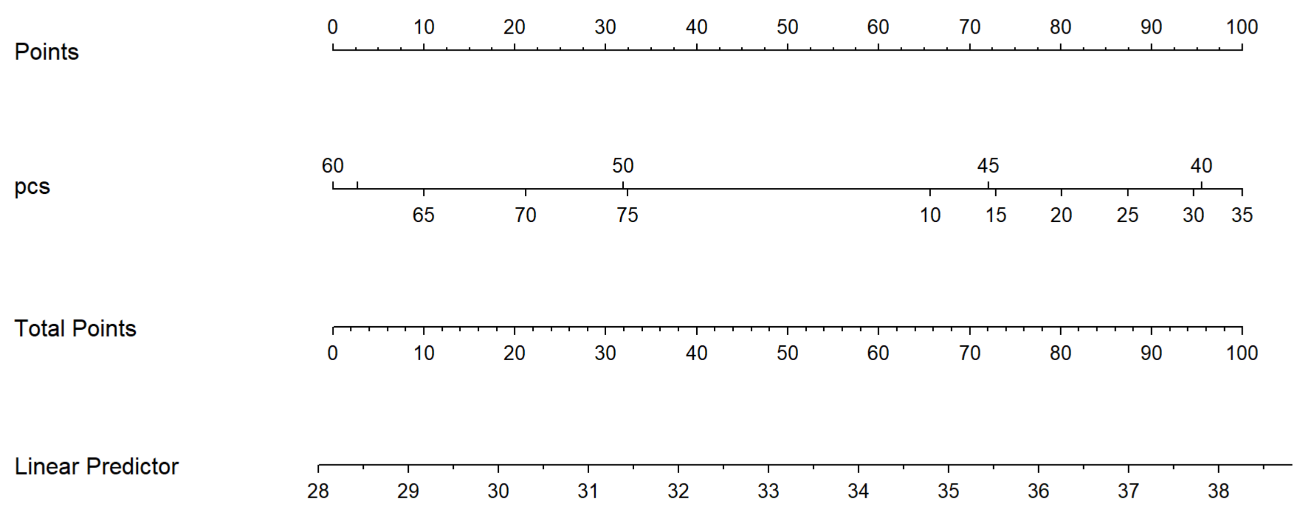



```r
ggplot(Predict(fitC4, conf.int = 0.90))
```



A Nomogram for `fitC4`

```
plot(nomogram(fitC4, abbrev = TRUE))
```



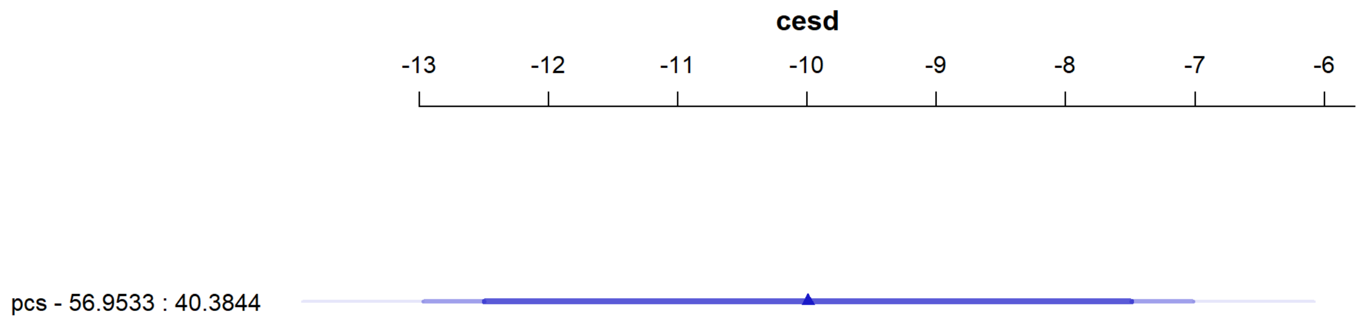
Model `fitC5` (5-knot spline in `pcs`)

```
fitC5
```

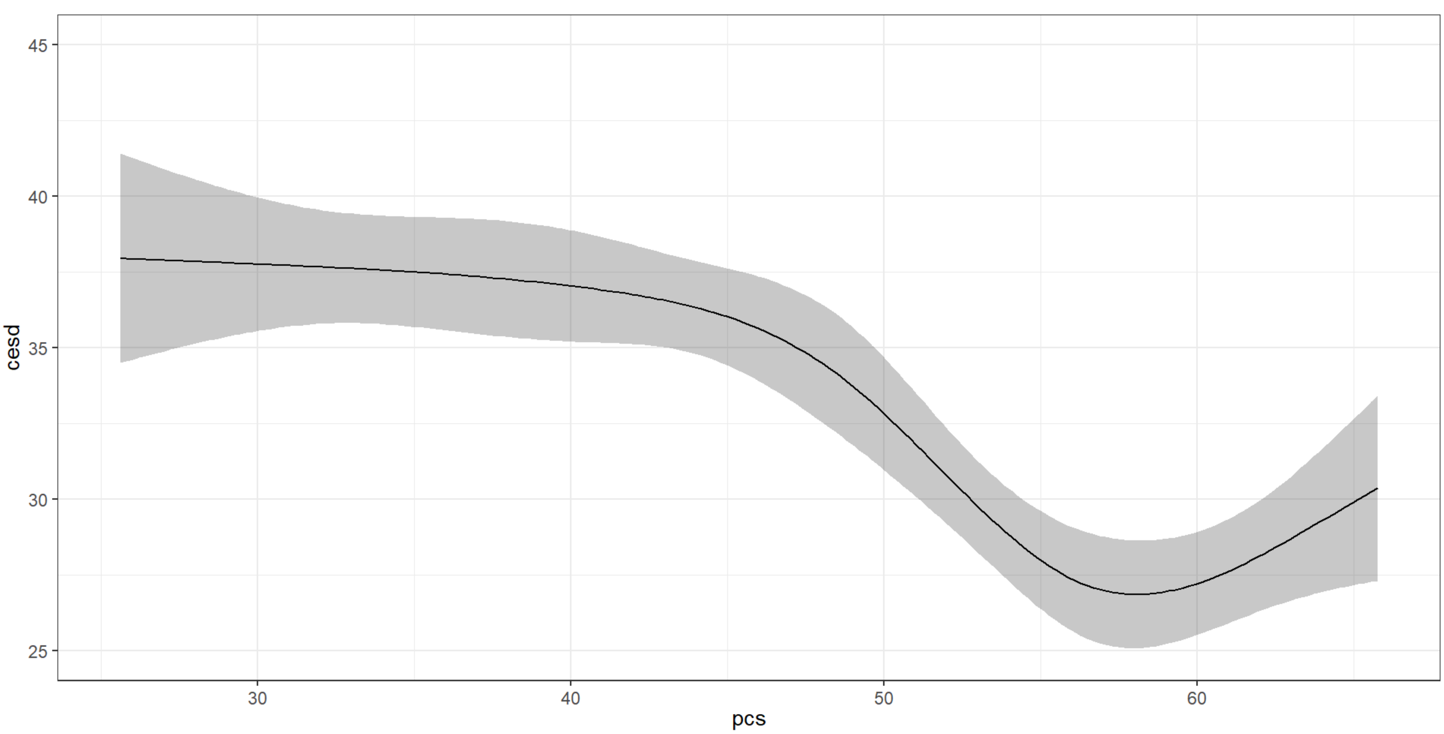
```
Linear Regression Model

ols(formula = cesd ~ rcs(pcs, 5), data = help1, x = TRUE, y = TRUE)

                 Model Likelihood    Discrimination
                       Ratio Test           Indexes
Obs       453    LR chi2     54.64    R2       0.114
sigma11.8345    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
```

Effect Sizes in `fitC5`

```r
plot(summary(fitC5))
```

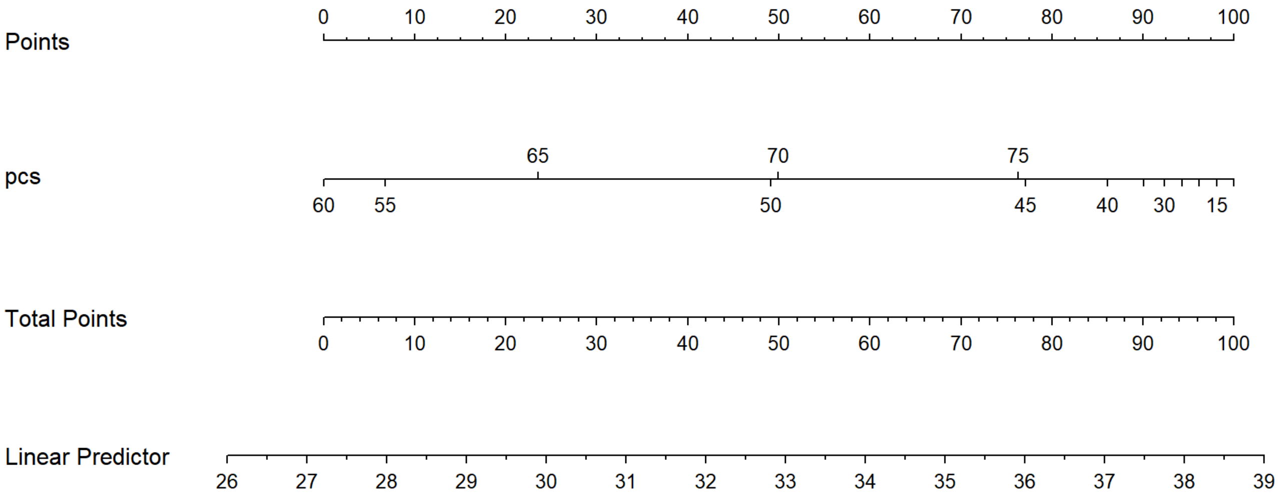



```r
ggplot(Predict(fitC5, conf.int = 0.90))
```



A Nomogram for `fitC5`

```r
plot(nomogram(fitC5, abbrev = TRUE))
```



Fitting `fitB` including a 5-knot RCS

```r
dd <- datadist(help1)
options(datadist = "dd")

fitB5 <- ols(cesd ~ rcs(pcs,5) + subst + pss_fr + sex,
             data = help1, x = TRUE, y = TRUE)

fitB5$coefficients
```

```
    Intercept           pcs          pcs'         pcs''        pcs'''
   48.4263870    -0.1151961     0.2519557    -4.9683530    15.6487400
subst=cocaine  subst=heroin        pss_fr      sex=male
   -3.7580871    -0.1528491    -0.4985050    -4.7527659
```

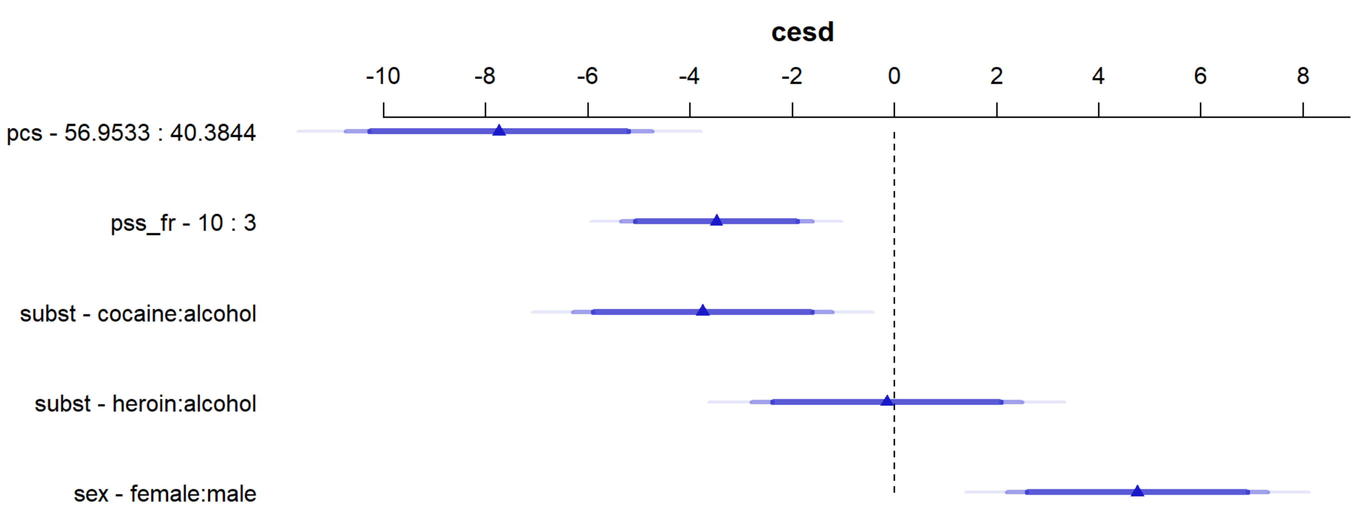
Contents of `fitB5`?

```
fitB5
```

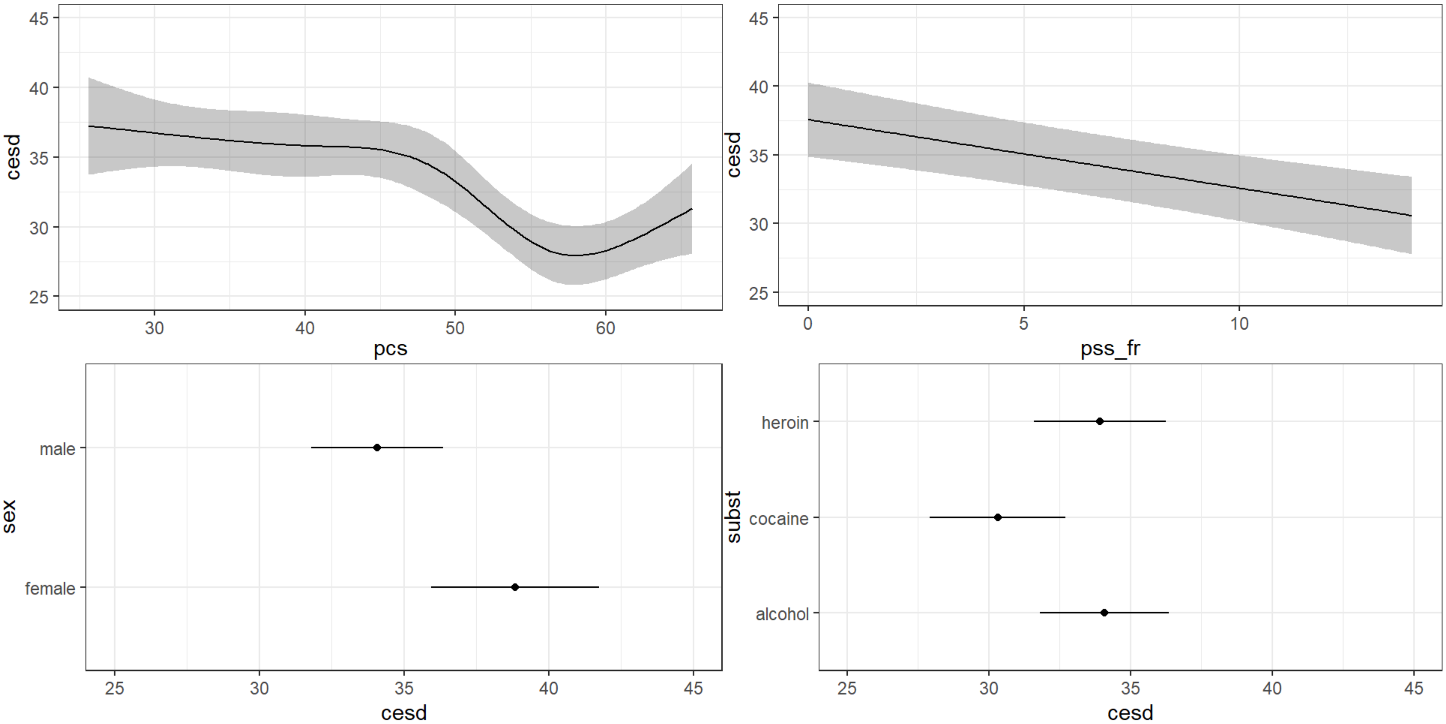
```
Linear Regression Model

ols(formula = cesd ~ rcs(pcs, 5) + subst + pss_fr + sex, data = help1,
    x = TRUE, y = TRUE)

                    Model Likelihood    Discrimination
                          Ratio Test           Indexes
Obs       453     LR chi2      86.99    R2       0.175
sigma11.4707     d.f.             8    R2 adj   0.160
d.f.      444     Pr(> chi2) 0.0000    g        5.991

Residuals

    Min      1Q  Median      3Q     Max
```


Effect Sizes in `fitB5`

```
plot(summary(fitB5))
```

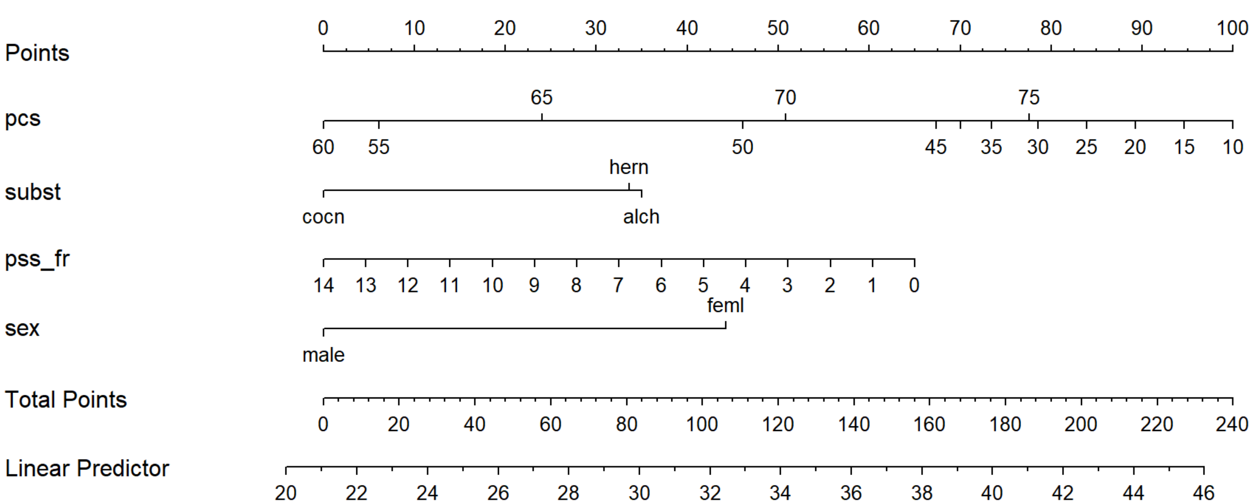


```r
ggplot(Predict(fitB5, conf.int = 0.90))
```



A Nomogram for `fitB5`

```r
plot(nomogram(fitB5, abbrev = TRUE))
```



What if you're doing a logistic regression?

Predicting Pr(CESD>15) with a spline

```
dd <- datadist(help1)
options(datadist = "dd")

fitD5 <- lrm(cesd_hi ~ rcs(pcs,5) + subst + pss_fr + sex,
             data = help1, x = TRUE, y = TRUE)

fitD5$coefficients
```

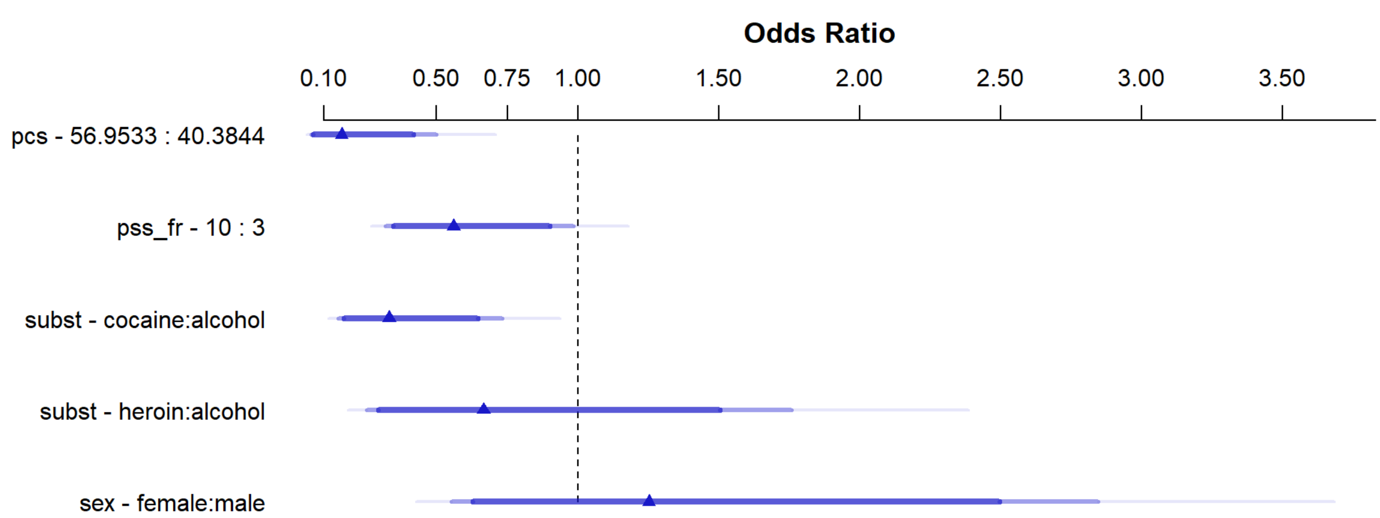
```
    Intercept           pcs          pcs'         pcs''        pcs'''
   6.62332347   -0.05399131    0.03371332   -0.87238631    3.36265431
subst=cocaine  subst=heroin        pss_fr      sex=male
  -1.10386515   -0.40683747   -0.08278496   -0.22546869
```

Contents of `fitD5`?

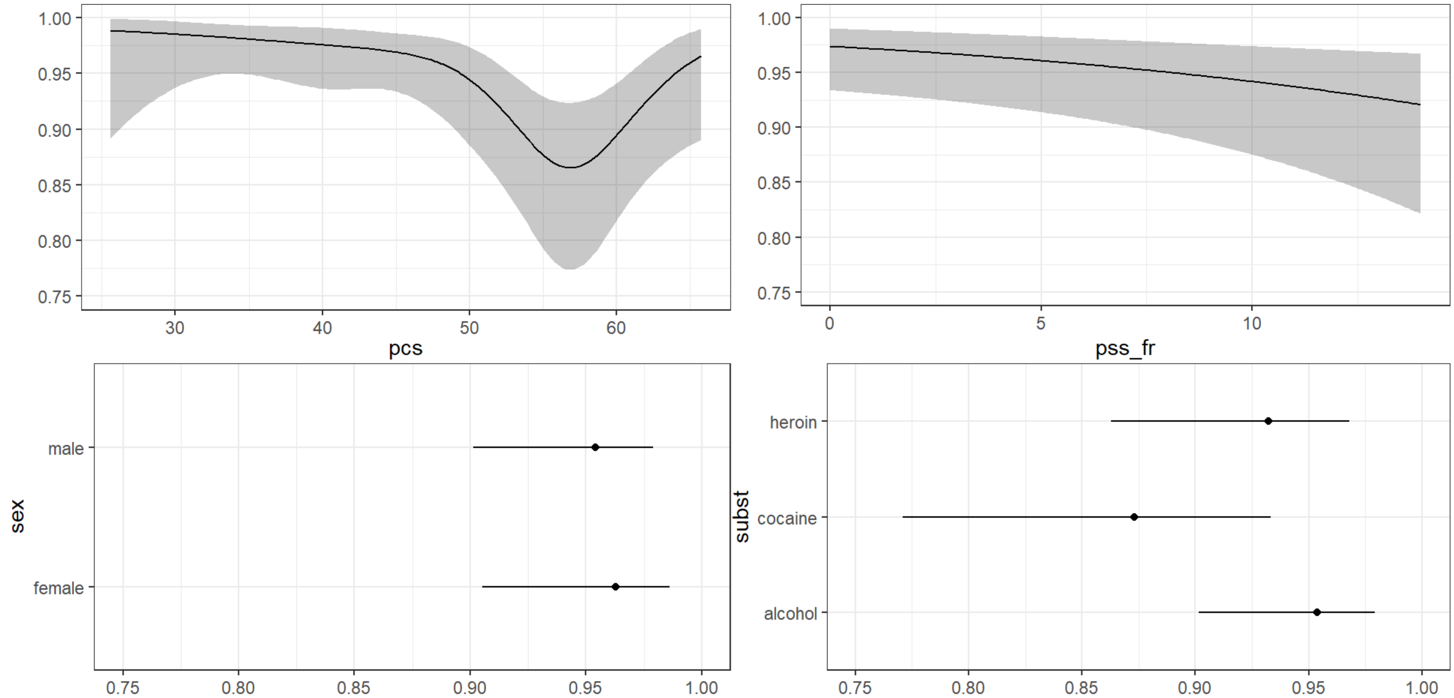
```
1  fitD5
```

```
Logistic Regression Model

lrm(formula = cesd_hi ~ rcs(pcs, 5) + subst + pss_fr + sex, data = help1,
    x = TRUE, y = TRUE)

                       Model Likelihood    Discrimination    Rank Discrim.
                             Ratio Test           Indexes          Indexes
Obs            453    LR chi2      42.01    R2        0.184    C       0.778
 0              46    d.f.             8    R2(8,453)0.072    Dxy     0.555
 1             407    Pr(> chi2) <0.0001    R2(8,124)0.240    gamma   0.555
max |deriv| 3e-05                           Brier     0.083    tau-a   0.102

              Coef    S.E.   Wald Z Pr(>|Z|)
Intercept      6.6233 4.8970  1.35  0.1762
```


Effect Sizes in `fitD5`

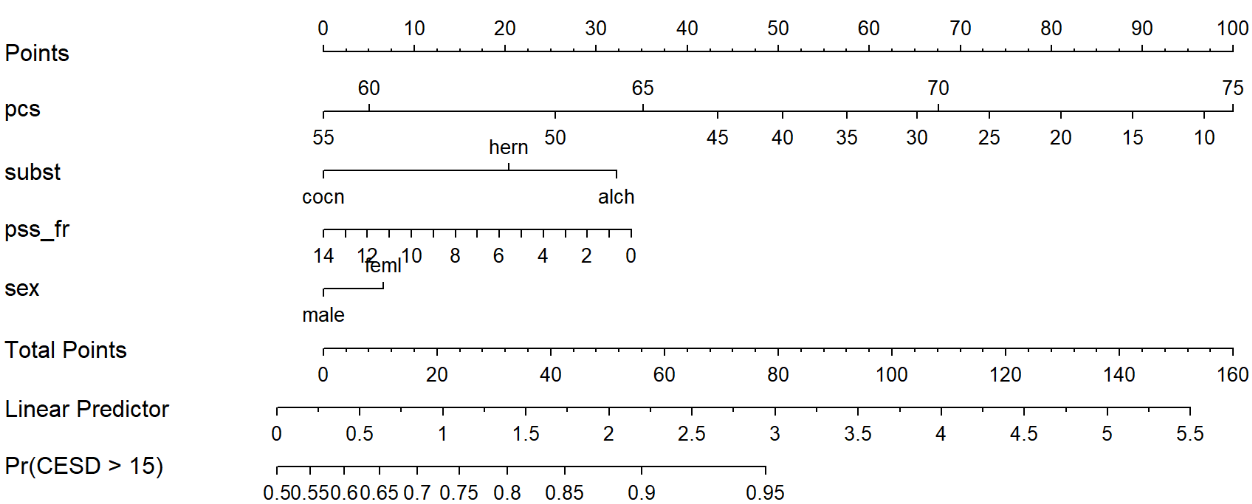


```r
ggplot(Predict(fitD5, conf.int = 0.90, fun = plogis))
```



A Nomogram for `fitD5`

```r
plot(nomogram(fitD5, abbrev = TRUE, fun = plogis, funlabel = "Pr(CESD > 15)"))
```



What's next?

- Spearman's ρ^2 plot: exploring non-linearity
  - Spending degrees of freedom on non-linearity wisely
- Model Calibration
- Making Predictions with `ols()` models
- Using both `ols()` and `lm()` as fitters