Multiple Linear Regression: Least squares and non-linearity

Author: Nicholas G Reich, Jeff Goldsmith

This material is part of the **statsTeachR** *project*

Today's topics

- least squares for MLR: geometry, "hat matrix"
- collinearity and non-identifiability
- introduction to modeling non-linear relationships

Example predicting respiratory disease severity ("lung" dataset)
Holding off on inference/diagnostics for another week...

Multiple linear regression model

- Observe data $(y_i, x_{i1}, \ldots, x_{ip})$ for subjects $1, \ldots, n$. Want to estimate $\beta_0, \beta_1, \ldots, \beta_p$ in the model

$$y_i = \beta_0 + \beta_1 x_{i1} + \ldots + \beta_1 x_{ip} + \epsilon_i; \ \epsilon_i \overset{iid}{\sim} (0, \sigma^2)$$

Déjà vu: MLR assumptions

Assumptions

- Residuals have mean zero, constant variance, are independent
- Often assuming linearity
- Our primary interest will be $E(y|\mathbf{x})$
- Estimation using least squares

Déjà vu: Least squares

As in simple linear regression, we want to find the $\boldsymbol{\beta}$ that minimizes the residual sum of squares.

$$RSS(\boldsymbol{\beta}) = \sum_i \epsilon_i^2 = \epsilon^T \epsilon$$

After taking the derivative, setting equal to zero, we obtain:

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

Déjà vu: Sampling distribution of $\hat{\boldsymbol{\beta}}$

If our usual assumptions are satisfied and $\epsilon \overset{iid}{\sim} \mathrm{N}\left[0, \sigma^2\right]$ then

$$\hat{\boldsymbol{\beta}} \sim \mathrm{N}\left[\boldsymbol{\beta}, \sigma^2(\mathbf{X}^T\mathbf{X})^{-1}\right].$$

- This will be used later for inference.
- Even without Normal errors, asymptotic Normality of LSEs is possible under reasonable assumptions.

Déjà vu: Definitions

- *Fitted values*: $\hat{\mathbf{y}} = \mathbf{X}\hat{\boldsymbol{\beta}} = \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} = \mathbf{H}\mathbf{y}$
- *Residuals / estimated errors*: $\hat{\epsilon} = \mathbf{y} - \hat{\mathbf{y}}$
- *Residual sum of squares*: $\sum_{i=1}^{n} \hat{\epsilon_i}^2 = \hat{\boldsymbol{\epsilon}}^T\hat{\boldsymbol{\epsilon}}$
- *Residual variance*: $\hat{\sigma^2} = \frac{RSS}{n-p-1}$
- *Degrees of freedom*: $n - p - 1$

Déjà vu: R^2 and sums of squares

- Regression sum of squares $SS_{reg} = \sum(\hat{y}_i - \bar{y})^2$
- Residual sum of squares $SS_{res} = \sum(y_i - \hat{y}_i)^2$
- Total sum of squares $SS_{tot} = \sum(y_i - \bar{y})^2$
- Coefficient of determination

$$R^2 = 1 - \frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \bar{y})^2} = \frac{\sum(\hat{y}_i - \bar{y})^2}{\sum(y_i - \bar{y})^2}$$

Not so Déjà vu: the "Hat matrix"

Some properties of the hat matrix:

- It is a projection matrix: $\mathbf{HH} = \mathbf{H}$
- It is symmetric: $\mathbf{H}^T = \mathbf{H}$
- The residuals are $\hat{\epsilon} = (\mathbf{I} - \mathbf{H})\mathbf{y}$
- The inner product of $(\mathbf{I} - \mathbf{H})\mathbf{y}$ and $\mathbf{Hy}$ is zero (predicted values and residuals are uncorrelated).

Projection space interpretation

The hat matrix projects $\mathbf{y}$ onto the column space of $\mathbf{X}$.
Alternatively, minimizing the $RSS(\beta)$ is equivalent to minimizing the Euclidean distance between $\mathbf{y}$ and the column space of $\mathbf{X}$.

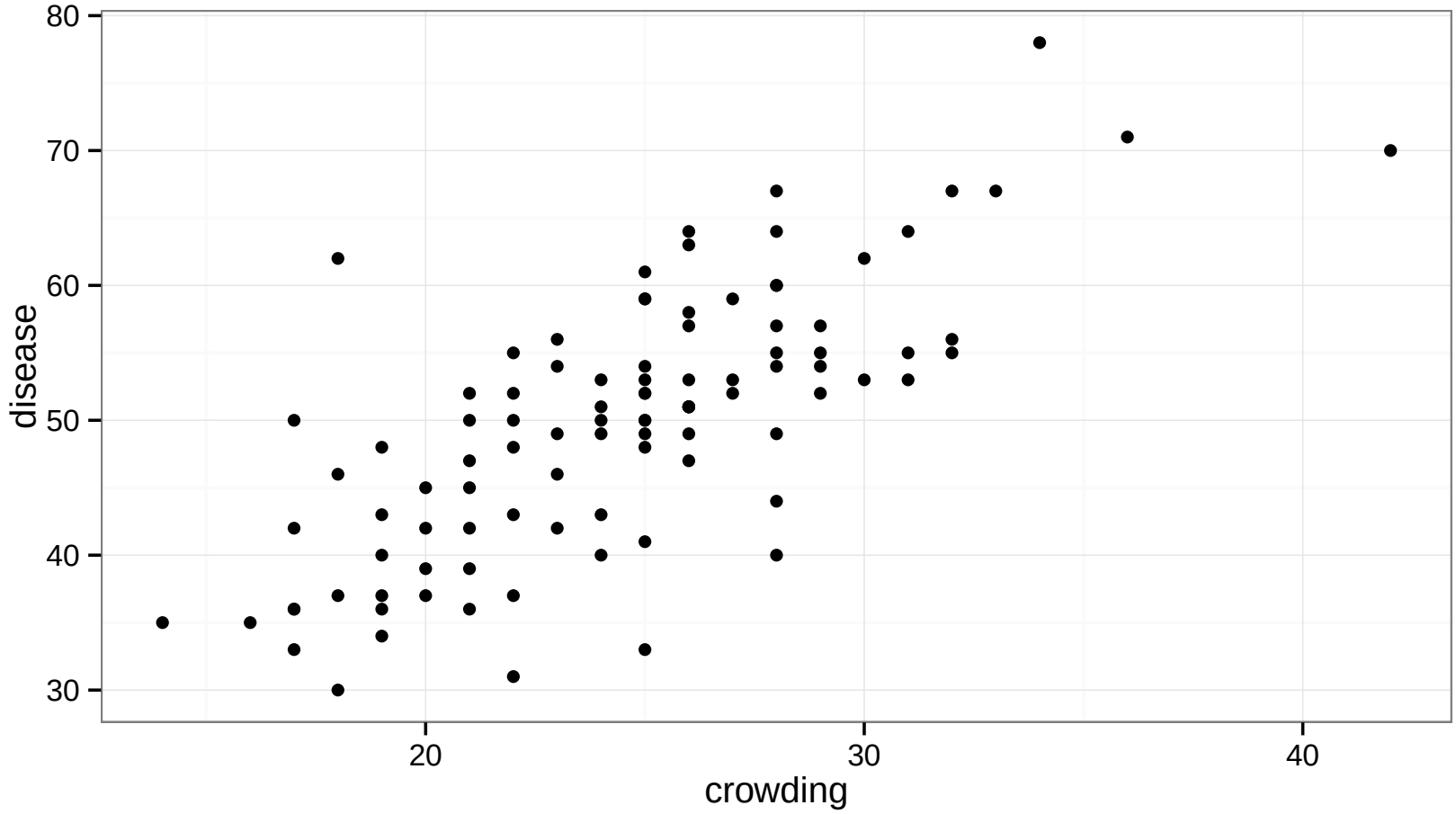
Lung Data Example

99 observations on patients who have sought treatment for the relief of respiratory disease symptoms. The dependent variable, The variables are:

- `disease` measure of disease severity (larger values indicates more serious condition).
- `education` highest grade completed
- `crowding` measure of crowding of living quarters (larger values indicate more crowding)
- `airqual` measure of air quality at place of residence (larger number indicates poorer quality)
- `nutrition` nutritional status (larger number indicates better nutrition)
- `smoking` smoking status (1 if smoker, 0 if non-smoker)

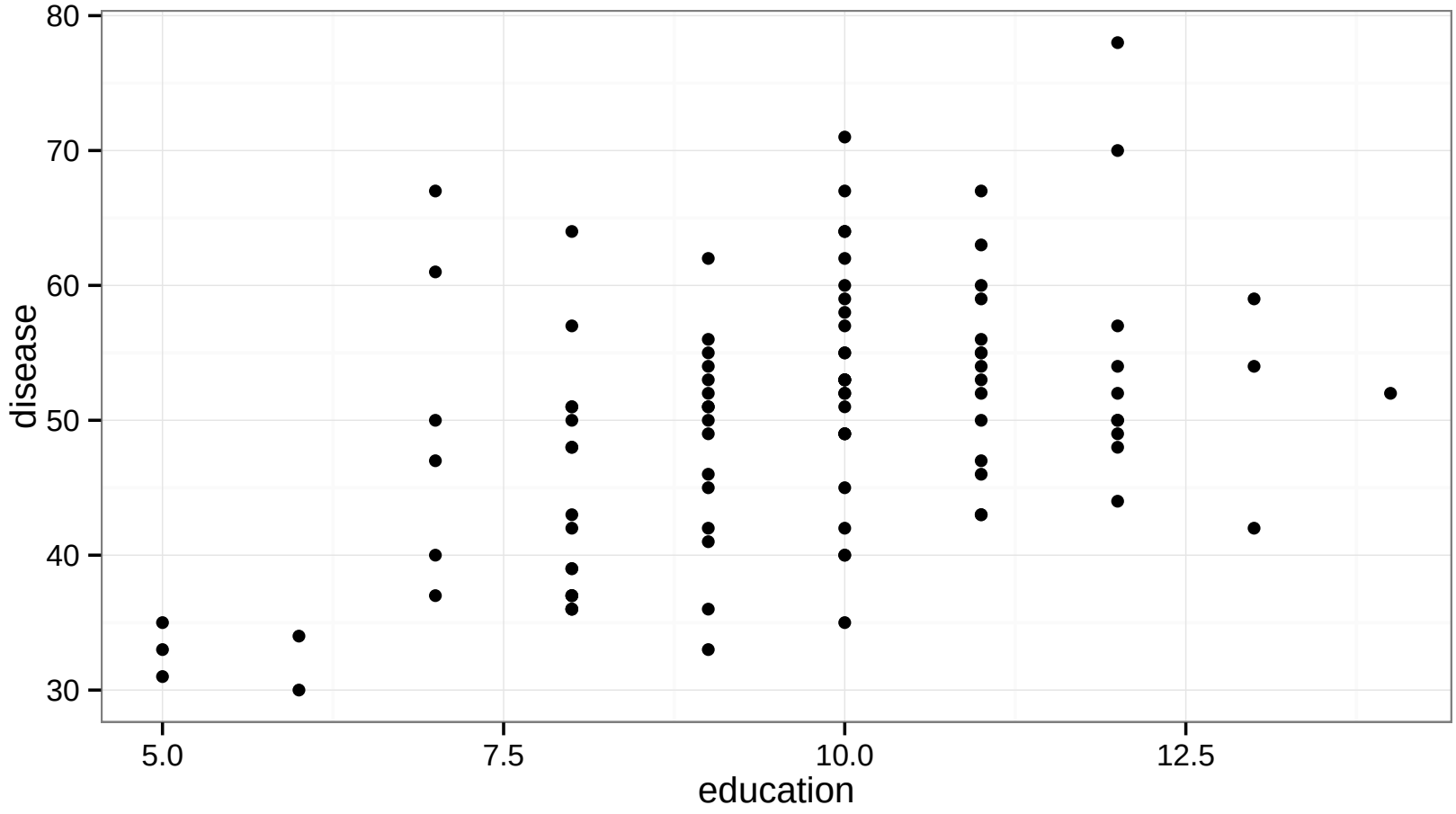
Lung Data Example

```
qplot(crowding, disease, data=dat)
```



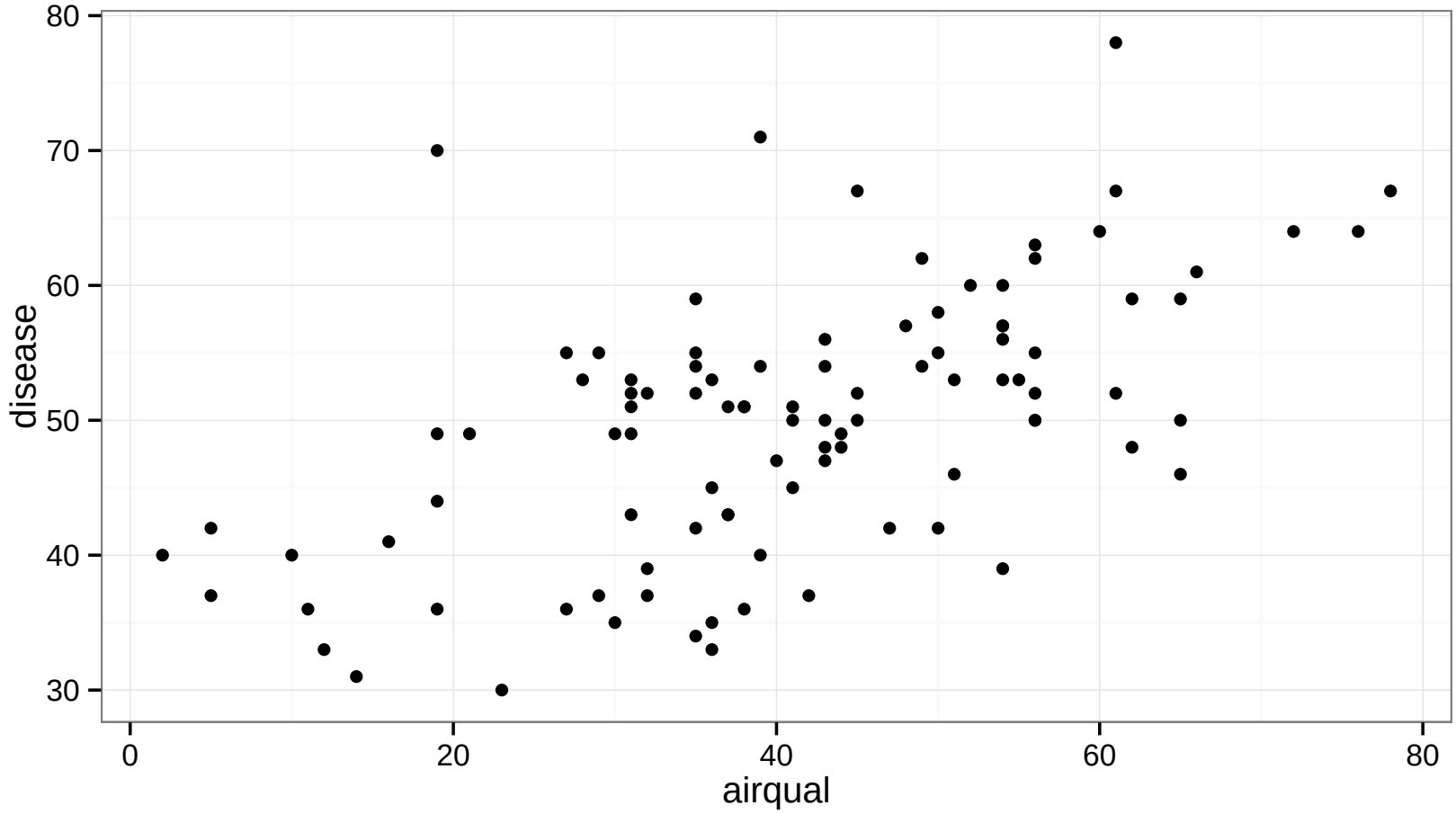
Lung Data Example

```
qplot(education, disease, data=dat)
```



Lung Data Example

```
qplot(airqual, disease, data=dat)
```



Lung Data Example

```
mlr1 <- lm(disease ~ crowding + education + airqual,
           data=dat, x=TRUE, y=TRUE)
coef(mlr1)

## (Intercept)    crowding   education     airqual
##  -7.7505215   1.3127837   1.4376563   0.2880687

X = mlr1$x
y = mlr1$y
(beta_hat = solve(t(X)%*%X) %*% t(X) %*% y )

##                    [,1]
## (Intercept) -7.7505215
## crowding     1.3127837
## education    1.4376563
## airqual      0.2880687
```

Least squares estimates: identifiability

$$\hat{\boldsymbol{\beta}} = \left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T\mathbf{y}$$

A condition on $\left(\mathbf{X}^T\mathbf{X}\right)$: must be invertible

- If $\left(\mathbf{X}^T\mathbf{X}\right)$ is singular, there are infinitely many least squares solutions, making $\hat{\boldsymbol{\beta}}$ non-identifiable (can't choose between different solutions)
- In practice, true **non-identifiability** (there really are infinite solutions) is rare.
- More common, and perhaps more dangerous, is **collinearity**.

Causes of non-identifiability

- Can happen if $\mathbf{X}$ is not of full rank, i.e. the columns of $\mathbf{X}$ are linearly dependent (for example, including weight in Kg and lb as predictors)
- Can happen if there are fewer data points than terms in $\mathbf{X}$: $n < p$ (having 100 predictors and only 50 observations)
- Generally, the $p \times p$ matrix $\left(\mathbf{X}^T\mathbf{X}\right)$ is invertible if and only if it has rank p.

Infinite solutions

Suppose I fit a model $y_i = \beta_0 + \beta_1 x_{i1} + \epsilon_i$.

- I have estimates $\hat{\beta}_0 = 1, \hat{\beta}_1 = 2$
- I put in a new variable $x_2 = x_1$
- My new model is $y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \epsilon_i$
- Possible least squares estimates that are equivalent to my first model:
  - $\hat{\beta}_0 = 1, \hat{\beta}_1 = 2, \hat{\beta}_2 = 0$
  - $\hat{\beta}_0 = 1, \hat{\beta}_1 = 0, \hat{\beta}_2 = 2$
  - $\hat{\beta}_0 = 1, \hat{\beta}_1 = 1002, \hat{\beta}_2 = -1000$
  - $\ldots$

Non-identifiability example: lung data

```
mlr3 <- lm(disease ~ airqual, data=dat)
coef(mlr3)

## (Intercept)     airqual
##  35.4444812   0.3537389

dat$x2 <- dat$airqual/100
mlr4 <- lm(disease ~ airqual + x2, data=dat, x=TRUE)
coef(mlr4)

## (Intercept)     airqual          x2
##  35.4444812   0.3537389          NA

X = mlr4$x
solve( t(X) %*% X)

## Error in solve.default(t(X) %*% X): system is computationally
singular:  reciprocal condition number = 3.57906e-20
```

Non-identifiablity: causes and solutions

- Often due to data coding errors (variable duplication, scale changes)
- Pretty easy to detect and resolve
- Can be addressed using *penalties* (might come up much later)
- A bigger problem is near-unidentifiability (collinearity)

Diagnosing collinearity

- Arises when variables are highly correlated, but not exact duplicates
- Commonly arises in data (perfect correlation is usually there by mistake)
- Might exist between several variables, i.e. a linear combination of several variables exists in the data
- A variety of tools exist (correlation analyses, multiple R^2, eigen decompositions)

Effects of collinearity

Suppose I fit a model $y_i = \beta_0 + \beta_1 x_{i1} + \epsilon_i$.

- I have estimates $\hat{\beta}_0 = 1, \hat{\beta}_1 = 2$
- I put in a new variable $x_2 = x_1 + \textit{error}$, where *error* is pretty small
- My new model is $y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \epsilon_i$
- Possible least squares estimates that are nearly equivalent to my first model:
  - $\hat{\beta}_0 = 1, \hat{\beta}_1 = 2, \hat{\beta}_2 = 0$
  - $\hat{\beta}_0 = 1, \hat{\beta}_1 = 0, \hat{\beta}_2 = 2$
  - $\hat{\beta}_0 = 1, \hat{\beta}_1 = 1002, \hat{\beta}_2 = -1000$
  - $\ldots$
- A unique solution exists, but it is hard to find

Effects of collinearity

- Collinearity results in a "flat" RSS
- Makes identifying a unique solution difficult
- Dramatically inflates the variance of LSEs

Collinearity example: lung data

```
dat$crowd2 <- dat$crowding + rnorm(nrow(dat), sd=.1)
mlr5 <- lm(disease ~ crowding, data=dat)
summary(mlr5)$coef

##              Estimate Std. Error   t value     Pr(>|t|)
## (Intercept) 12.991536  3.4750250  3.738544 3.130355e-04
## crowding     1.508806  0.1393709 10.825836 2.231686e-18

mlr6 <- lm(disease ~ crowding + crowd2, data=dat)
summary(mlr6)$coef

##              Estimate Std. Error    t value     Pr(>|t|)
## (Intercept) 13.019617   3.490168  3.7303699 0.0003236495
## crowding    -1.510590   6.883739 -0.2194432 0.8267707628
## crowd2       3.017039   6.876945  0.4387180 0.6618516741
```

Some take away messages

- Collinearity can (and does) happen, so be careful
- Often contributes to the problem of variable selection, which we'll touch on later

Non-linear relationships: polynomial regression

Many relationships between X and Y are non-linear. A simple (not necessarily the best) way to account for this is using polynomial forms of X.

- Model of the form

$$y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \ldots + \beta_p x_i^p + \epsilon_i;\ \epsilon_i \overset{iid}{\sim} (0, \sigma^2)$$

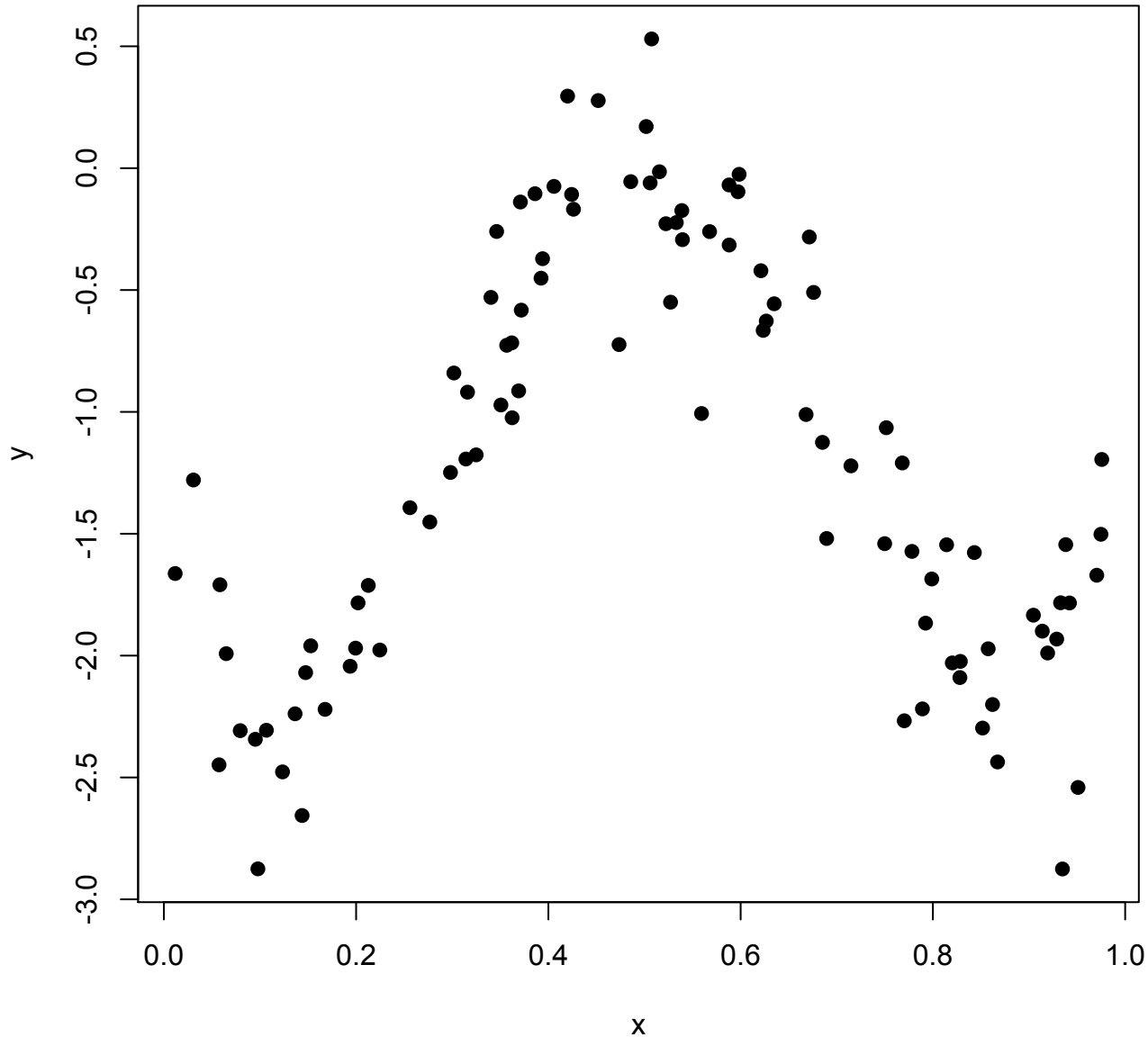
- p is the polynomial order
- More polynomial terms can lead to a better approximation of $E(y|x)$, but also higher variability in the fit
- Conversely, smaller p can lead to inability to capture $E(y|x)$, but is often more stable
- Quadratic and cubic fits are relatively common

Non-linear relationships

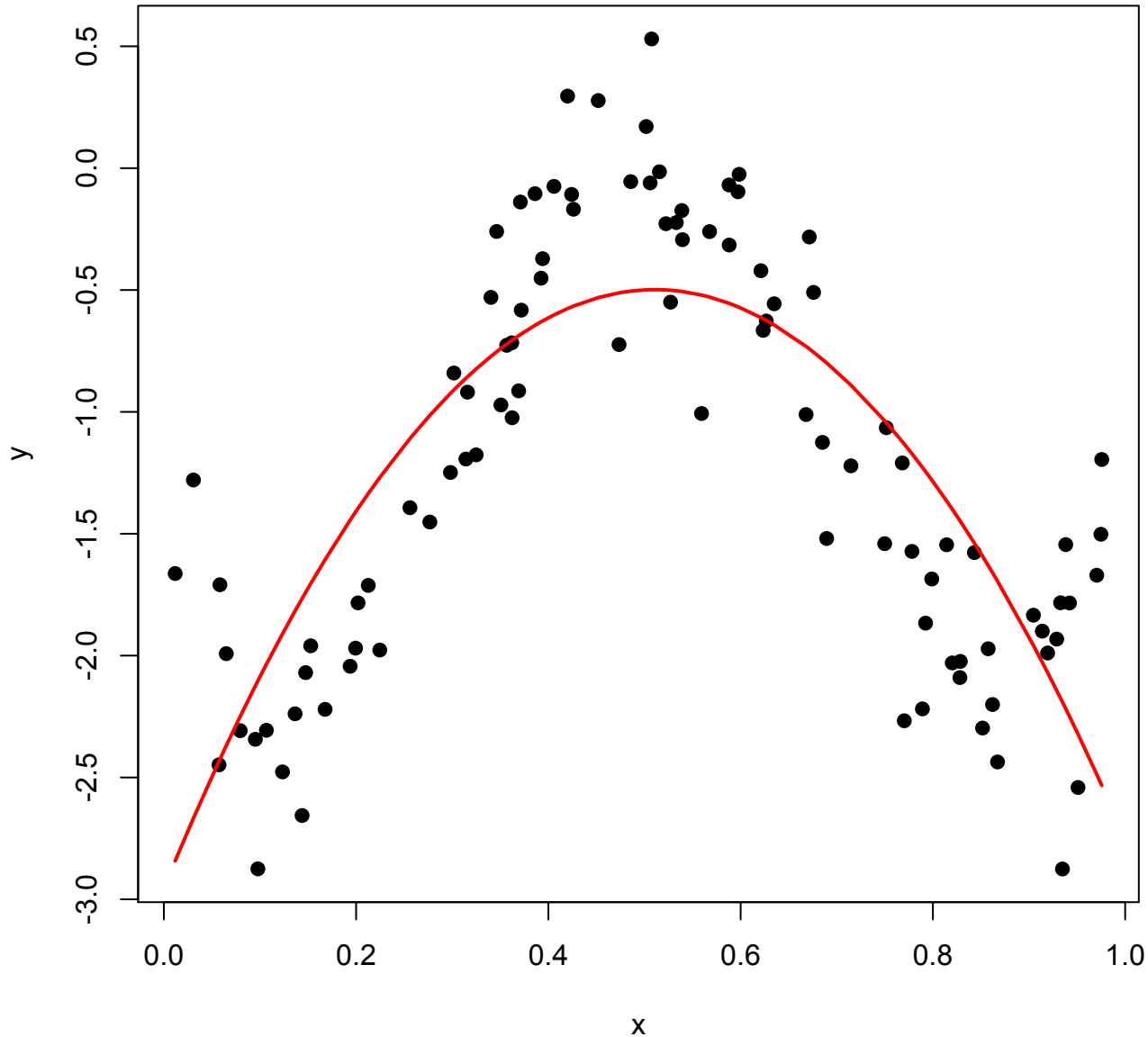
Some tips on non-linear relationships

- You can go as high as $p = n$, but don't do it! "Overfitting" data is common practice (unfortunately).
- Coefficients become harder to interpret – you can't increase x_2 without changing every other x_p
- Better (maybe) to think of the model as an estimated curve, whose interpretation is related to the derivative
- The literal formulation above is numerically unstable. Better to use orthogonal polynomials (R's `poly` function)

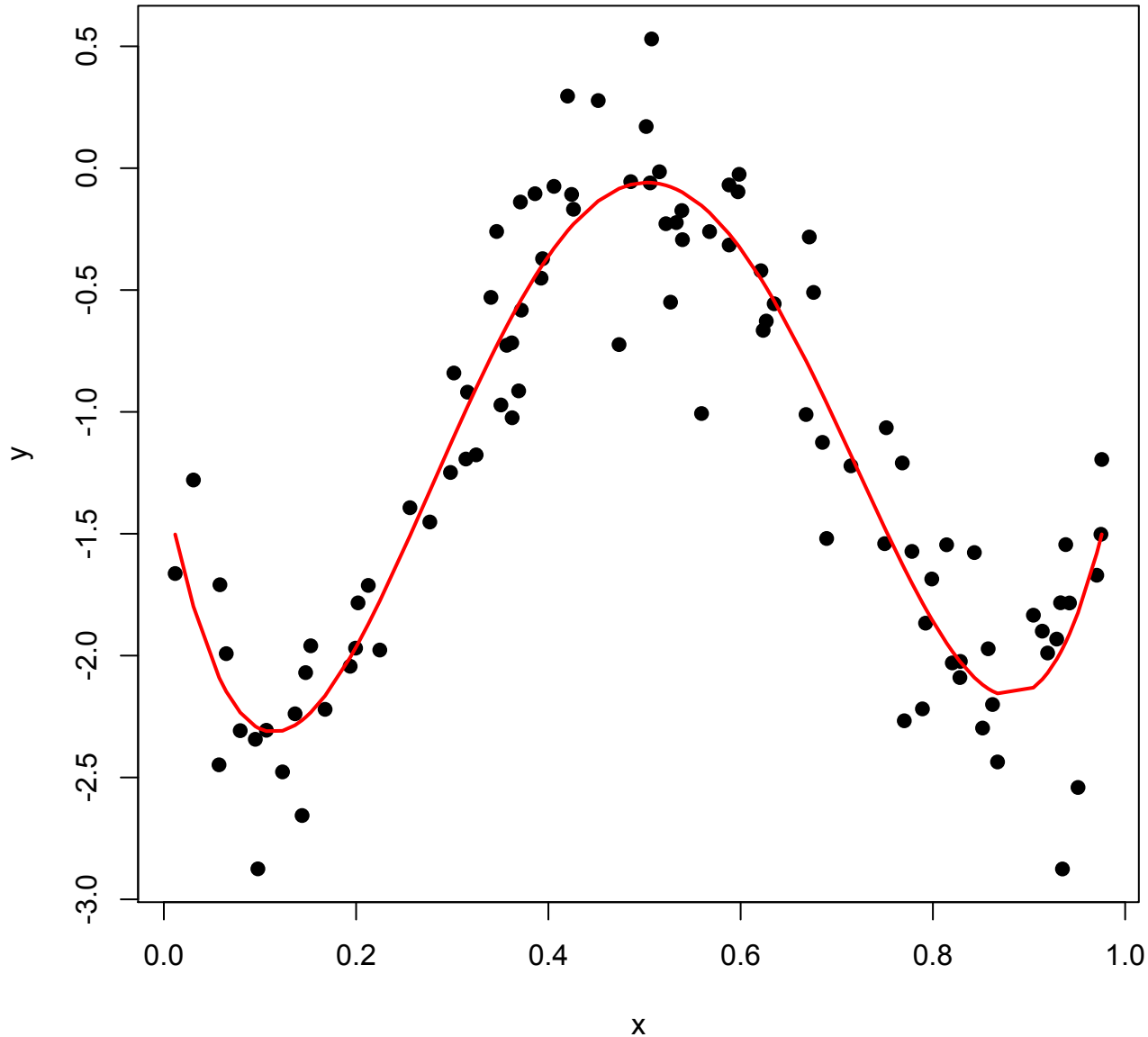
Non-linear relationships



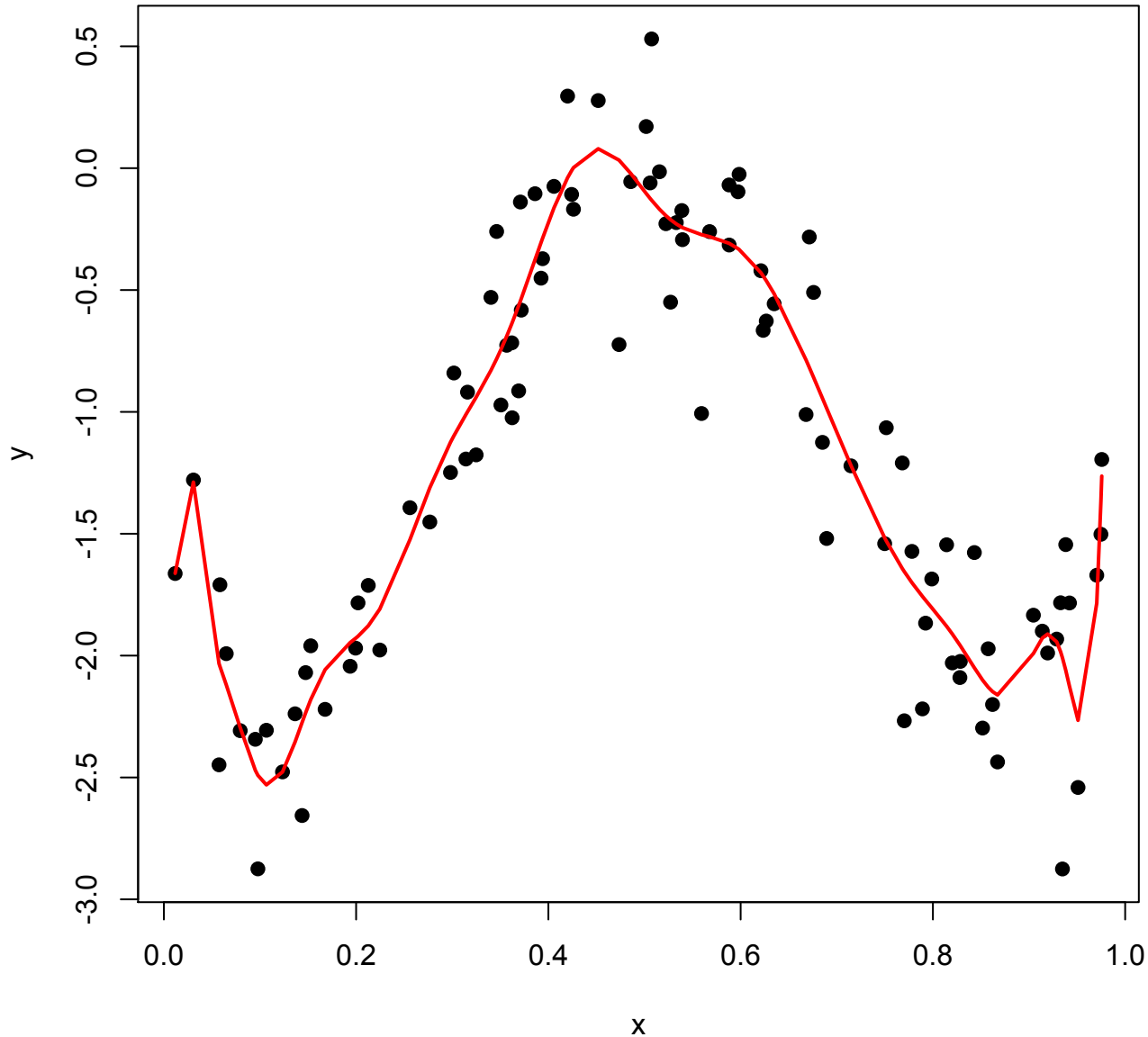
Non-linear relationships



Non-linear relationships

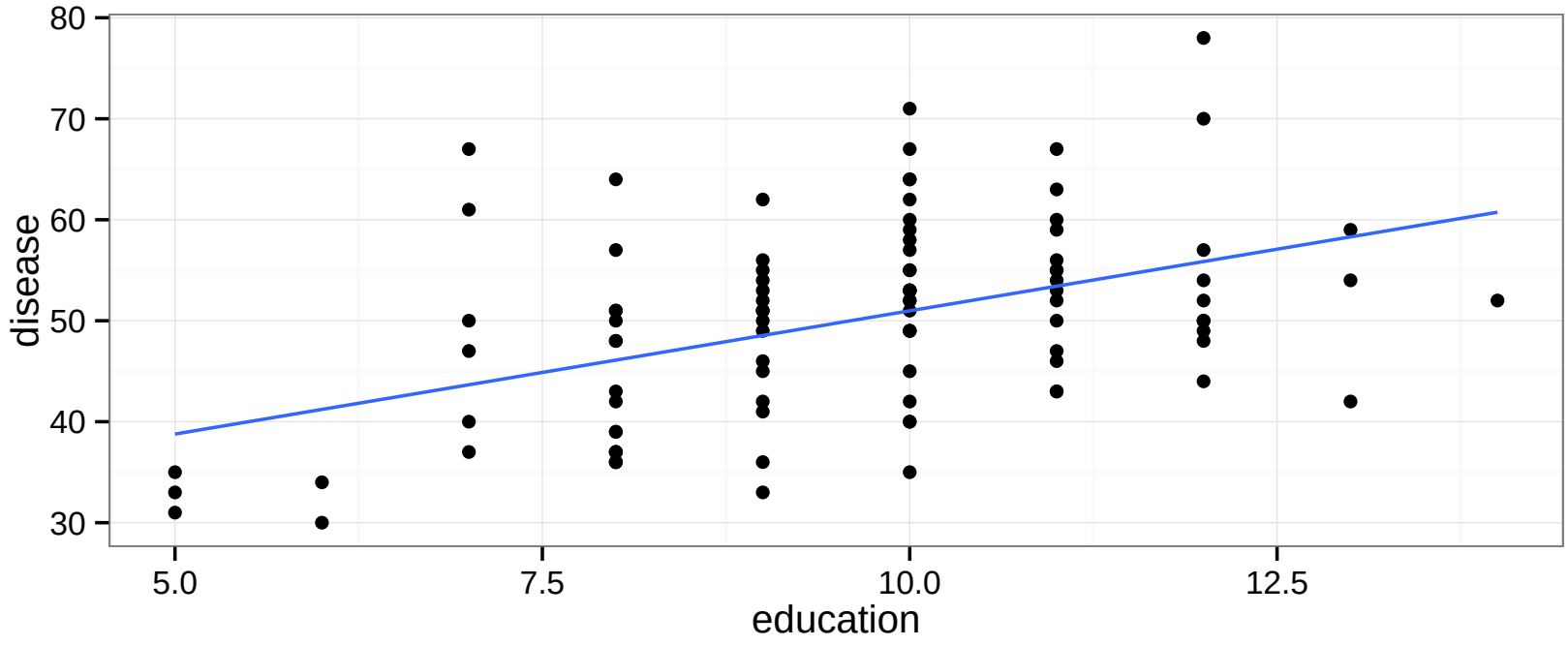


Non-linear relationships



Non-linear relationships

```
(p <- ggplot(dat, aes(x=education, y=disease)) + geom_point() +
    geom_smooth(method="lm", se=FALSE) )
```

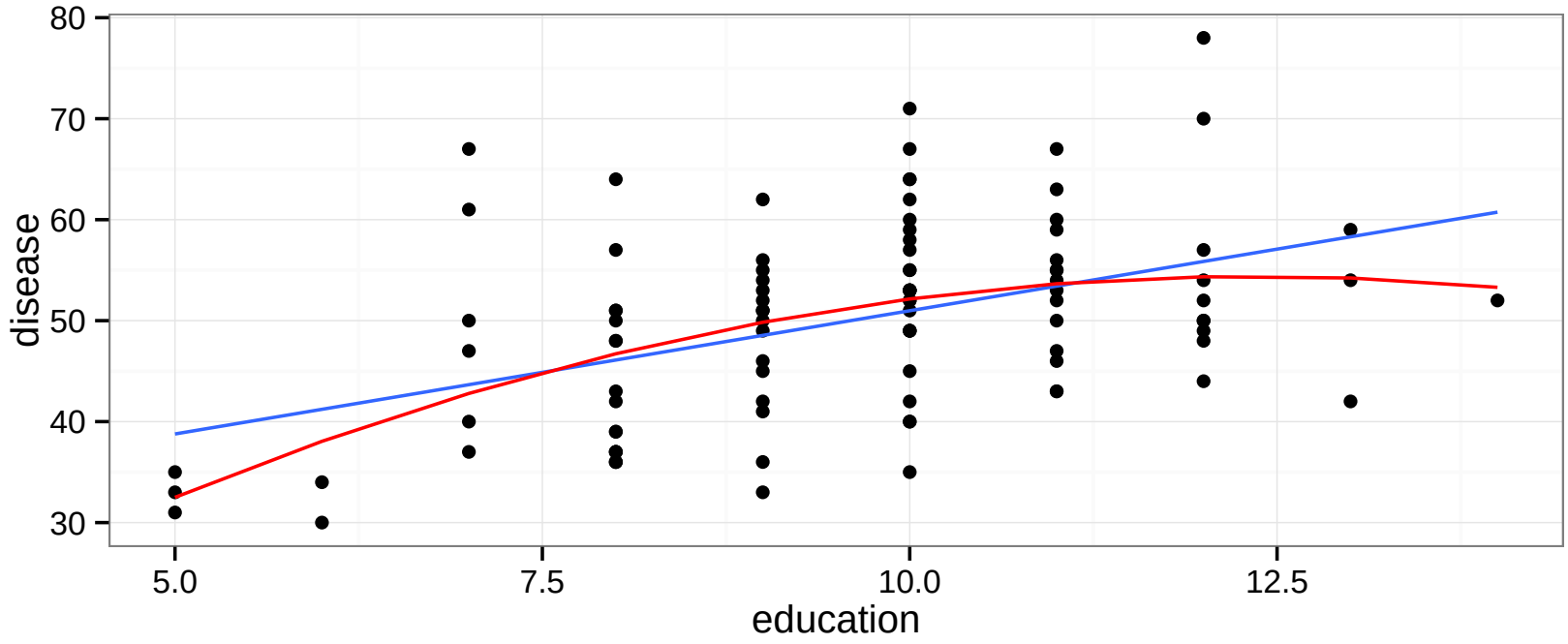


Non-linear relationships

```
mlr3 <- lm(disease ~ poly(education, 2), data=dat)
coef(mlr3)

##          (Intercept) poly(education, 2)1 poly(education, 2)2
##              49.91919             43.95171            -18.46921

(p <- p + geom_line(aes(y=predict(mlr3)), color="red") )
```

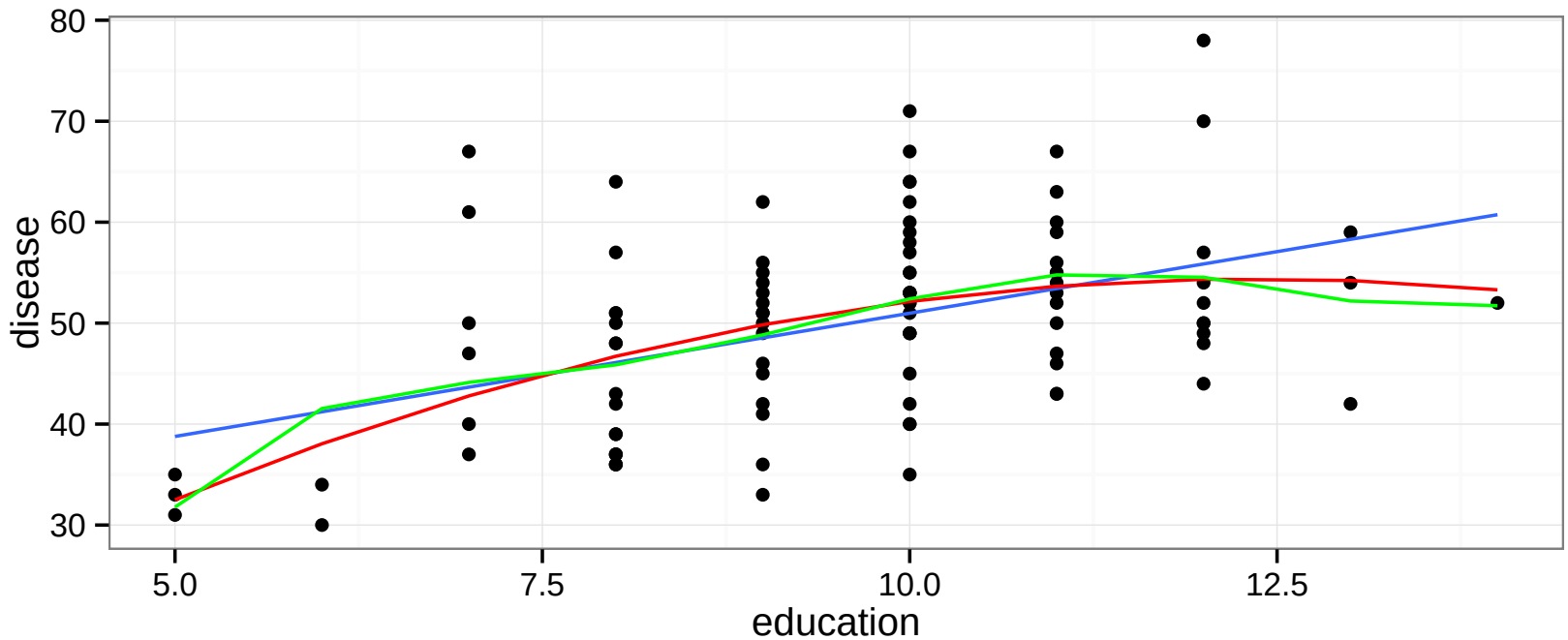


Non-linear relationships

```
mlr4 <- lm(disease ~ poly(education, 5), data=dat)
coef(mlr4)

##          (Intercept) poly(education, 5)1 poly(education, 5)2
##            49.919192           43.951707          -18.469208
## poly(education, 5)3 poly(education, 5)4 poly(education, 5)5
##            -4.131932           -4.651902            7.896361

(p <- p + geom_line(aes(y=predict(mlr4)), color="green"))
```



Smoothing and splines

Turns out there's a lot of work on estimating smooth $E(y|x) = f(x)$

- Rather than polynomials, use smooth spline basis functions with nice properties (stable, smooth, flexible, smooth derivatives)
- These are piecewise polynomials
- How many to use governs how smooth or wiggly the final fit is
- Can introduce explicit penalties for smoothness, which gets you into semi-parametric regression ...

Today's big ideas

- least squares geometry, "hat matrix"
- dangers of collinearity and non-identifiability
- polynomial regression to model non-linear relationships

Lab

Analyze the NHANES dataset. Create a model with the outcome variable of cholesterol (`chol`) that estimates relationships with other variables in the dataset.

```
library(NHANES)
data(NHANES)
?NHANES
```