Nonlinear, Non-normal models

Lecture 05.1: Nonlinear Models

Lauren Sullivan

Module: Linear, Nonlinear, and Mixed Effects Models

Readings

Required for class:

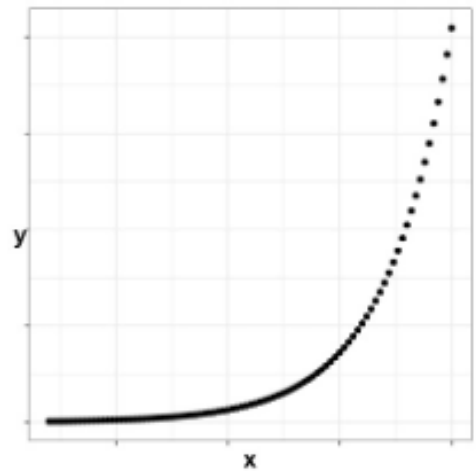
- NA

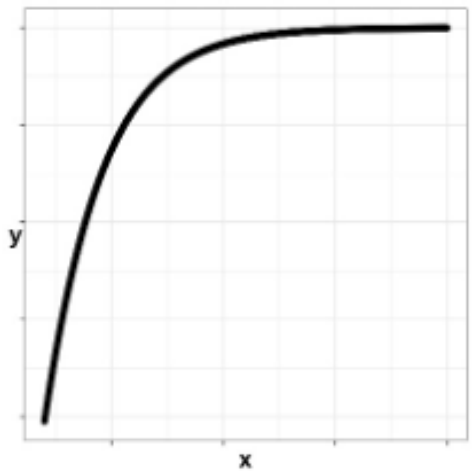
Optional:

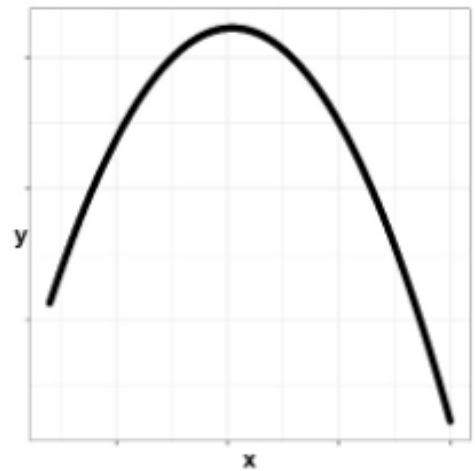
- Crawley, M. *Statistics: An Introduction Using R*
- Bolker, B. *Ecological Models and Data in R - Ebook version*
- R-tutorials, Nonlinear Regressions
- Helpful non-linear equations and associated R code

Nonlinear Regression

A nonlinear regression occurs when your data do not follow a linear trend (think: an exponential function, a saturating function, a parabolic function, etc.)







What is the shape of your data?

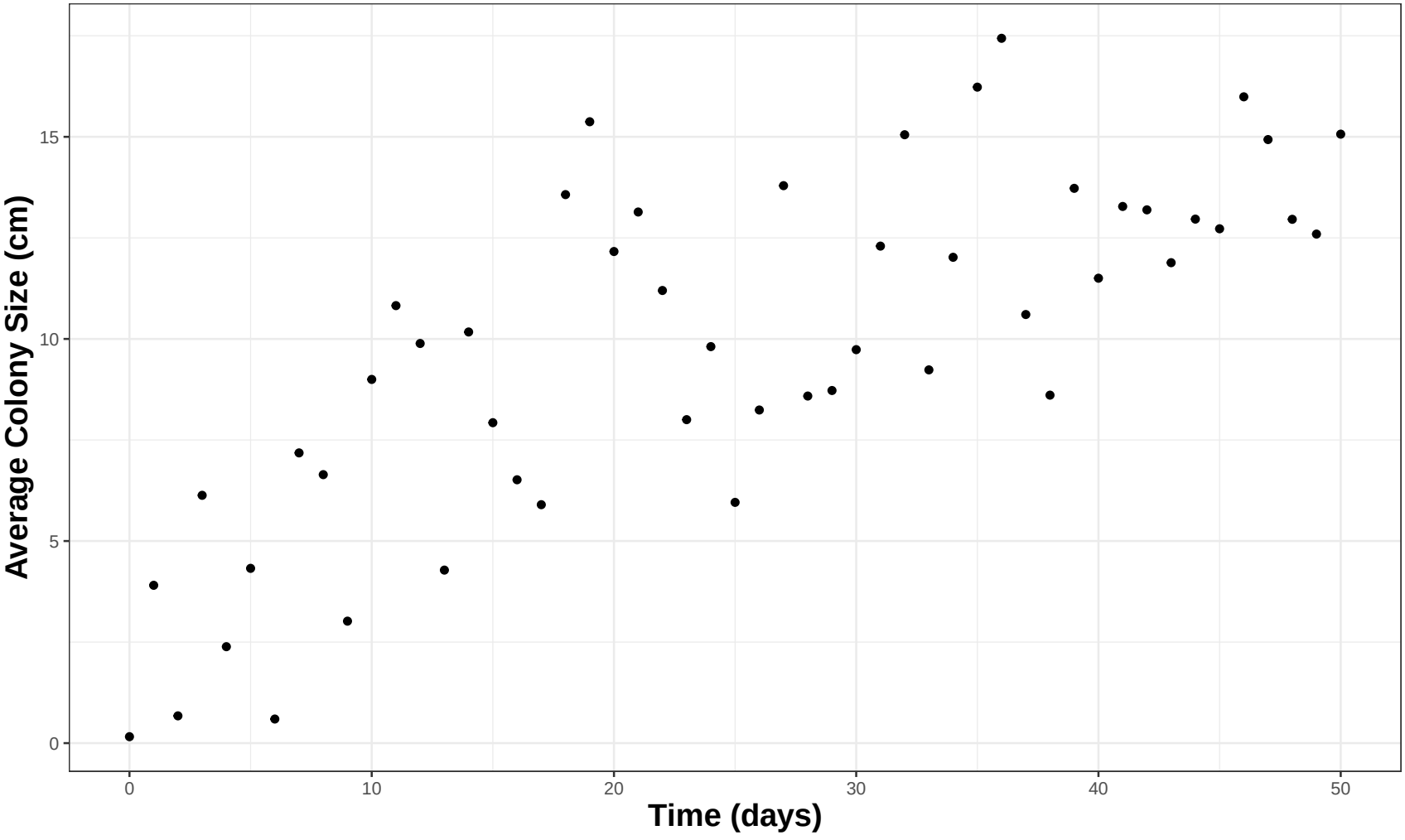
1. Sometimes you know you have nonlinear or non-normal data and you a-priori know you need to use a non-standard linear model. For example:

- Exponential growth.
- Binary response data (survived till the next time point yes/no)
- Count data

2. Sometimes you have data, and you just do not know if you should use a linear or nonlinear model.

Data

Bacterial colony growth on a medium through time.



Linear? Saturating? Hump-shaped?

How do you determine the shape of your data?

You can fit Y as a function of X, trying several different potential functional types.

- Then compare your models with AIC to determine which fits best.

Y as a function of X

Say we have three potential functions we think might determine the shape of X on Y. (You can often find these on Wikipedia!)

- Linear: $Y \sim aX + \epsilon$
- Saturating: $Y \sim \left(\frac{aX}{b+X}\right) + \epsilon$
- Quadratic: $Y \sim aX^2 + bX + \epsilon$

To do this we need the Nonlinear Least Squares function `nls()`. This determines the nonlinear (weighted) least-squares estimates of the parameters of a nonlinear model.

Running the models

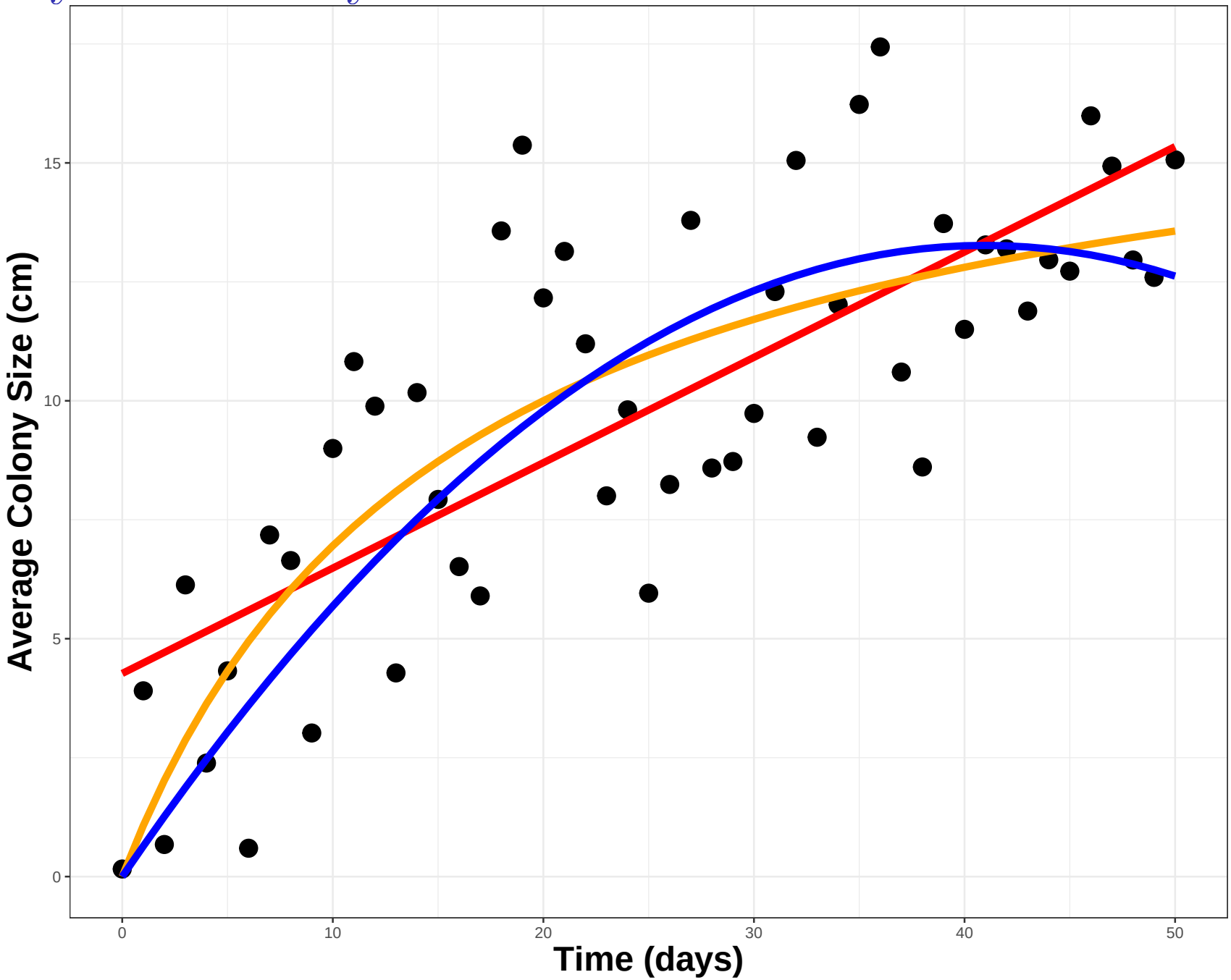
We then run all the models and compare them with AIC.

```
fit_linear <- lm(avg_colony_size ~ time, data = colony)

fit_saturating <- nls(avg_colony_size ~ (a * time)/(b + time),
                      data = colony, start = list(a = 1 , b = .5))

fit_quad <- nls(avg_colony_size ~ a * time^2 + b * time,
                data = colony, start = list(a = 0 , b = 0))

AICtab(fit_linear, fit_saturating, fit_quad)
```

```
##                dAIC df
## fit_saturating 0.0  3
## fit_quad       5.8  3
## fit_linear     8.0  3
```

So it appears that **of the models we have tried,** the saturating function fits the data the best.

Always visualize your results



Running Significance tests

You can pull out the coefficients of your model (here: `a` and `b`).

```
coefficients(fit_saturating)
```

```
##        a        b
## 17.80015 15.59772
```

Running Significance tests

You can test to make sure the coefficients from your `nls()` are different from zero.

```
summary(fit_saturating)
```

```
##
## Formula: avg_colony_size ~ (a * time)/(b + time)
##
## Parameters:
##   Estimate Std. Error t value Pr(>|t|)
## a   17.800      2.099   8.480 3.56e-11 ***
## b   15.598      5.034   3.098  0.00322 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.626 on 49 degrees of freedom
##
## Number of iterations to convergence: 8
## Achieved convergence tolerance: 8.317e-07
```

Running Significance tests

You can fit an `lm()` by defining the equation with your parameters, and test for the effect of time on average colony size. But you will have to back transform your results when interpreting (solve for X).

```
coeffs <- coefficients(fit_saturating)
colony$a <- as.numeric(coeffs[1])
colony$b <- as.numeric(coeffs[2])

fit_saturating_lm <- lm(avg_colony_size ~ (a * time)/(b + time),
                        data = colony)

anova(fit_saturating_lm)
```

```
## Analysis of Variance Table
##
## Response: avg_colony_size
##           Df Sum Sq Mean Sq F value    Pr(>F)
## time       1 541.95  541.95  67.139 9.671e-11 ***
## Residuals 49 395.53    8.07
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```