Fitting models to data using Non-linear Least-Squares

Samraat Pawar

Department of Life Sciences

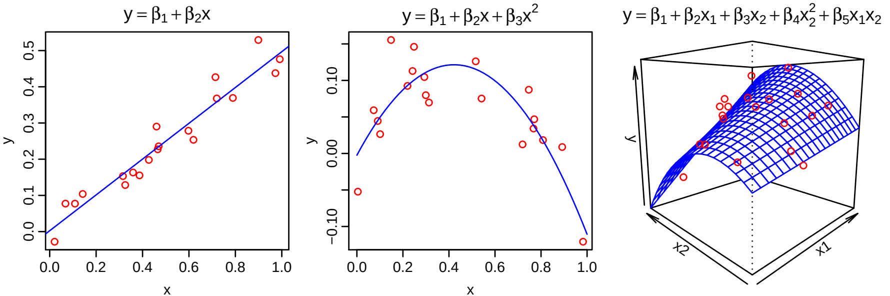
(Silwood Park)

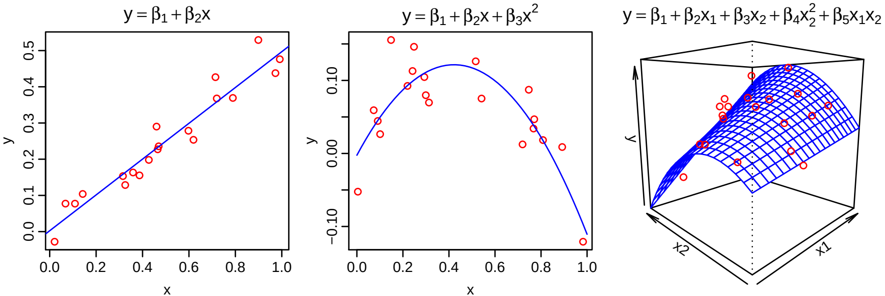
Imperial College London

January 23, 2017

OUTLINE

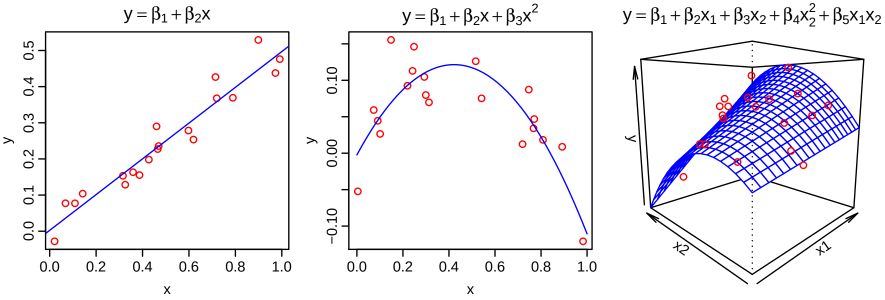
- Why Non-Linear Least Squares regression / fitting?
- The NLLS fitting method
- NLLS in `R`
- Afternoon practicals overview (two examples)


$y = \beta_1 + \beta_2 x$
y
x
0.0
0.1
0.2
0.3
0.4
0.5
0.0
0.2
0.4
0.6
0.8
1.0
$y = \beta_1 + \beta_2 x + \beta_3 x^2$
y
x
−0.10
0.00
0.10
0.0
0.2
0.4
0.6
0.8
1.0
$y = \beta_1 + \beta_2 x_1 + \beta_3 x_2 + \beta_4 x_2^2 + \beta_5 x_1 x_2$
y
x2
x1



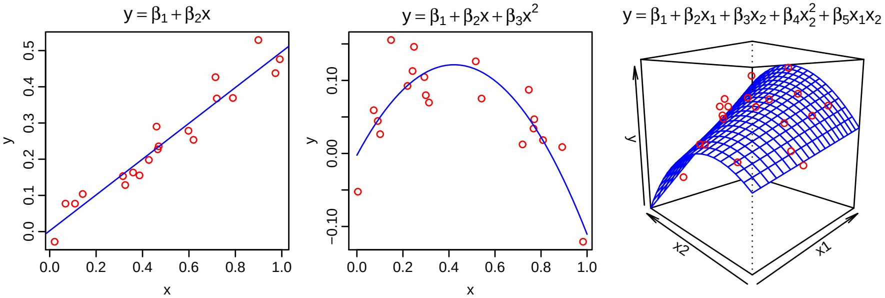
- These are *all* good linear models (huh?!)

LINEAR MODELS ARE GREAT



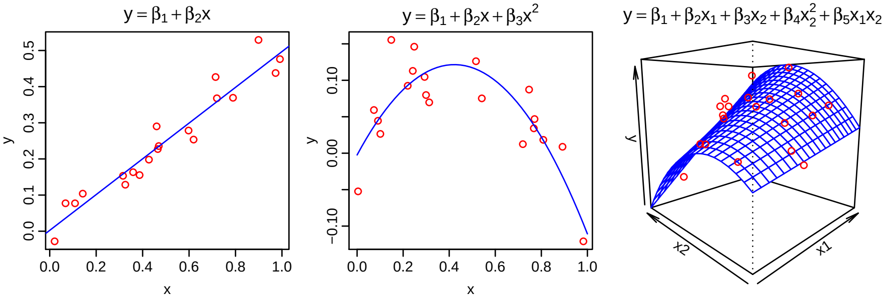
- These are *all* good linear models (huh?!)
- The data can be modelled as *linear combination* of variables and coefficients

Linear models are great

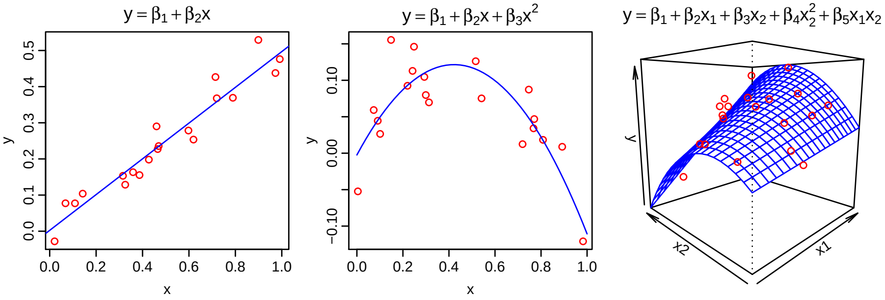


- These are *all* good linear models (huh?!)
- The data can be modelled as *linear combination* of variables and coefficients
- Easily fitted using Ordinary Least Squares (OLS) regression

LINEAR MODELS ARE GREAT



- These are *all* good linear models (huh?!)
- The data can be modelled as *linear combination* of variables and coefficients
- Easily fitted using Ordinary Least Squares (OLS) regression
- Linear models can include curved relationships (e.g. polynomials)



- These are *all* good linear models (huh?!)
- The data can be modelled as *linear combination* of variables and coefficients
- Easily fitted using Ordinary Least Squares (OLS) regression
- Linear models can include curved relationships (e.g. polynomials)
- *OK, so then why Non-Linear Least Squares (NLLS) fitting?*

SO WHAT — WHY IS INTRINSIC NON-LINEARITY A PROBLEM?

Recall what the Least Squares method does:

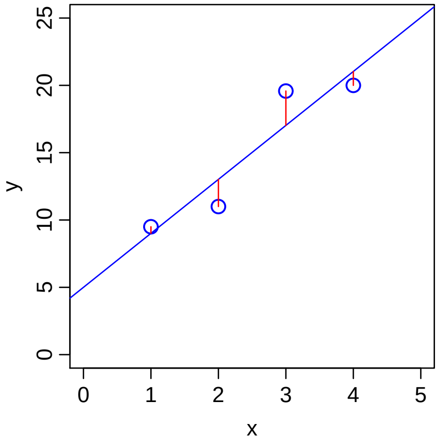
- Consider a predictor x, data y, n observations, and a model that we want to fit to the data:
  $f(x_i, \boldsymbol{\beta}) + \varepsilon_i$
  where $\boldsymbol{\beta} = (\beta_1, \beta_2, \ldots, \beta_k)$ are the model's k parameters

SO WHAT — WHY IS INTRINSIC NON-LINEARITY A PROBLEM?

Recall what the Least Squares method does:

- Consider a predictor x, data y, n observations, and a model that we want to fit to the data:
  $f(x_i, \boldsymbol{\beta}) + \varepsilon_i$
  where $\boldsymbol{\beta} = (\beta_1, \beta_2, \ldots, \beta_k)$ are the model's k parameters
- The objective is to find estimates of values of the k parameters ($\hat{\beta}_j$) that minimize the sum (S) of squared residuals (r_i) (AKA RSS):
  $S = \sum_{i=1}^{n} [y_i - f(x_i, \boldsymbol{\beta})]^2 = \sum_{i=1}^{n} r_i^2$

So what — why is intrinsic non-linearity a problem?



$$y_1 = \beta_0 + \beta_1 x_1 + \varepsilon_i$$

$$9.50 = 5 + 4 \times 1 + 0.50$$
$$11.00 = 5 + 4 \times 2 - 2.00$$
$$19.58 = 5 + 4 \times 3 + 2.58$$
$$20.00 = 5 + 4 \times 4 - 1.00$$

$$\beta_0 = 5; \beta_1 = 4$$

So what — why is intrinsic non-linearity a problem?

That's all well and good, but we can use maths instead of brute-force computation!

- Our model is $f(x_i, \boldsymbol{\beta}) + \varepsilon_i$

So what — why is intrinsic non-linearity a problem?

That's all well and good, but we can use maths instead of brute-force computation!

- Our model is $f(x_i, \boldsymbol{\beta}) + \varepsilon_i$
- We want to find estimates of values of the parameters ($\hat{\beta}_j$) that *minimize* the sum (S) of squared residuals (r_i) (AKA "RSS")
  $S = \sum_{i=1}^{n} [y_i - f(x_i, \boldsymbol{\beta})]^2 = \sum_{i=1}^{n} r_i^2$

So what — why is intrinsic non-linearity a problem?

That's all well and good, but we can use maths instead of brute-force computation!

- Our model is $f(x_i, \boldsymbol{\beta}) + \varepsilon_i$
- We want to find estimates of values of the parameters ($\hat{\beta}_j$) that *minimize* the sum (S) of squared residuals (r_i) (AKA "RSS")
  $S = \sum_{i=1}^{n} \left[y_i - f(x_i, \boldsymbol{\beta})\right]^2 = \sum_{i=1}^{n} r_i^2$
- For this we can solve $\frac{\partial S}{\partial \beta_j} = 0, j = 0, 1, 2, \ldots, k$ to find the *minimum*

So what — why is intrinsic non-linearity a problem?

That's all well and good, but we can use maths instead of brute-force computation!

- Our model is $f(x_i, \boldsymbol{\beta}) + \varepsilon_i$
- We want to find estimates of values of the parameters ($\hat{\beta}_j$) that *minimize* the sum (S) of squared residuals (r_i) (AKA "RSS")
  $S = \sum_{i=1}^{n} \left[y_i - f(x_i, \boldsymbol{\beta})\right]^2 = \sum_{i=1}^{n} r_i^2$
- For this we can solve $\frac{\partial S}{\partial \beta_j} = 0, j = 0, 1, 2, \ldots, k$ to find the *minimum*
- That is, we need to solve $\frac{\partial \sum_{i=1}^{n} r_i^2}{\partial \beta_j} = 0$

SO WHAT — WHY IS INTRINSIC NON-LINEARITY A PROBLEM?

That's all well and good, but we can use maths instead of brute-force computation!

- Our model is $f(x_i, \boldsymbol{\beta}) + \varepsilon_i$
- We want to find estimates of values of the parameters ($\hat{\beta}_j$) that *minimize* the sum (S) of squared residuals (r_i) (AKA "RSS") $S = \sum_{i=1}^{n} \left[y_i - f(x_i, \boldsymbol{\beta})\right]^2 = \sum_{i=1}^{n} r_i^2$
- For this we can solve $\frac{\partial S}{\partial \beta_j} = 0, j = 0, 1, 2, \ldots, k$ to find the *minimum*
- That is, we need to solve $\frac{\partial \sum_{i=1}^{n} r_i^2}{\partial \beta_j} = 0$
- Or, $2 \sum_{i=1}^{n} r_i \frac{\partial r_i}{\partial \beta_j} = 0$

So — Enter NLLS!

But we can use a computer!

- Choose initial values for the β_j's

So — Enter NLLS!

But we can use a computer!

- Choose initial values for the β_j's
- Then, "refine" the parameters *iteratively* by calculating $\frac{\partial r_i}{\partial \beta_j}$ *approximately* — this approximation is the *Jacobian* (the gradient), which is a matrix of the $\frac{\partial r_i}{\partial \beta_j}$'s

So — Enter NLLS!

But we can use a computer!

- Choose initial values for the β_j's
- Then, "refine" the parameters *iterative*ly by calculating $\frac{\partial r_i}{\partial \beta_j}$ *approximately* — this approximation is the *Jacobian* (the gradient), which is a matrix of the $\frac{\partial r_i}{\partial \beta_j}$'s
- Whether a refinement has taken place in any step of the iteration is determined by re-calculating the residuals at that step

So — Enter NLLS!

But we can use a computer!

- Choose initial values for the β_j's
- Then, "refine" the parameters *iteratively* by calculating $\frac{\partial r_i}{\partial \beta_j}$ *approximately* — this approximation is the *Jacobian* (the gradient), which is a matrix of the $\frac{\partial r_i}{\partial \beta_j}$'s
- Whether a refinement has taken place in any step of the iteration is determined by re-calculating the residuals at that step
- Eventually, if it all goes well, we find a combination of β_j's that is *very close* to the desired solution $\frac{\partial S}{\partial \beta_j} = 0, j = 0, 1, 2, \ldots, k$

OK, fine, why would *I* ever need to worry about NLLS?

OK, FINE, WHY WOULD *I* EVER NEED TO WORRY ABOUT NLLS?

- Many observations in biology are just not well-fitted by a linear model

OK, FINE, WHY WOULD *I* EVER NEED TO WORRY ABOUT NLLS?

- Many observations in biology are just not well-fitted by a linear model
- That is, the underlying biological phenomena are not well-described by a linear model

OK, fine, why would *I* ever need to worry about NLLS?

- Many observations in biology are just not well-fitted by a linear model
- That is, the underlying biological phenomena are not well-described by a linear model
- Examples:

OK, FINE, WHY WOULD *I* EVER NEED TO WORRY ABOUT NLLS?

- Many observations in biology are just not well-fitted by a linear model
- That is, the underlying biological phenomena are not well-described by a linear model
- Examples:
    - Logistic growth model

OK, FINE, WHY WOULD *I* EVER NEED TO WORRY ABOUT NLLS?

- Many observations in biology are just not well-fitted by a linear model
- That is, the underlying biological phenomena are not well-described by a linear model
- Examples:
  - Logistic growth model
  - Michaelis-Menten biochemical kinetics (two parameters $V_{\max}$ and K_m: $v = \frac{V_{\max}[S]}{K_m+[S]}$

OK, fine, why would *I* ever need to worry about NLLS?

- Many observations in biology are just not well-fitted by a linear model
- That is, the underlying biological phenomena are not well-described by a linear model
- Examples:
  - Logistic growth model
  - Michaelis-Menten biochemical kinetics (two parameters $V_{\max}$ and K_m: $v = \frac{V_{\max}[S]}{K_m+[S]}$
  - Responses of metabolic rates to changing temperature (practical 1)

OK, FINE, WHY WOULD *I* EVER NEED TO WORRY ABOUT NLLS?

- Many observations in biology are just not well-fitted by a linear model
- That is, the underlying biological phenomena are not well-described by a linear model
- Examples:
  - Logistic growth model
  - Michaelis-Menten biochemical kinetics (two parameters $V_{\max}$ and K_m: $v = \frac{V_{\max}[S]}{K_m+[S]}$
  - Responses of metabolic rates to changing temperature (practical 1)
  - Consumer-Resource (e.g., predator-prey) functional responses (practical 2)

OK, FINE, WHY WOULD *I* EVER NEED TO WORRY ABOUT NLLS?

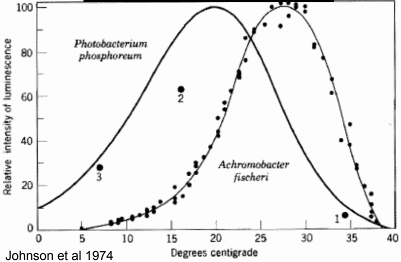
- Many observations in biology are just not well-fitted by a linear model
- That is, the underlying biological phenomena are not well-described by a linear model
- Examples:
    - Logistic growth model
    - Michaelis-Menten biochemical kinetics (two parameters $V_{\max}$ and K_m: $v = \frac{V_{\max}[S]}{K_m+[S]}$
    - Responses of metabolic rates to changing temperature (practical 1)
    - Consumer-Resource (e.g., predator-prey) functional responses (practical 2)
    - Time-series data (e.g., fitting a sinusoidal function)

OK, FINE, WHY WOULD *I* EVER NEED TO WORRY ABOUT NLLS?

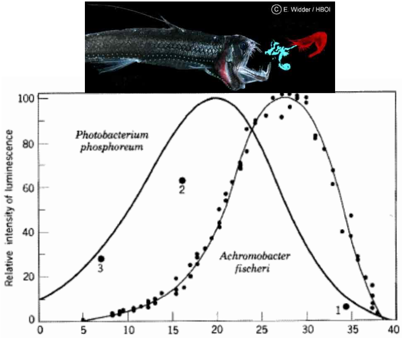
- Many observations in biology are just not well-fitted by a linear model
- That is, the underlying biological phenomena are not well-described by a linear model
- Examples:
  - Logistic growth model
  - Michaelis-Menten biochemical kinetics (two parameters $V_{\max}$ and K_m: $v = \frac{V_{\max}[S]}{K_m+[S]}$
  - Responses of metabolic rates to changing temperature (practical 1)
  - Consumer-Resource (e.g., predator-prey) functional responses (practical 2)
  - Time-series data (e.g., fitting a sinusoidal function)
  - *Can you think of some examples?*

Example 1: Temperature and metabolism

Example 1: Temperature and metabolism

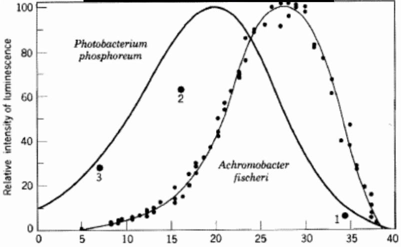


Johnson et al 1974



$$B = B_0 \boxed{e^{-\frac{E}{kT}}} f(T, T_{pk}, E_D)$$

T = temperature (K)
k = Boltzmann constant (eV K^{-1})
E = Activation energy (eV)
T_{pk} = Temperature of peak performance
E_D = Deactivation energy (eV)
(J H vant Hoff 1884, S Arrhenius 1889)

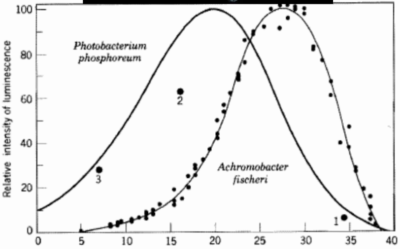


Johnson et al 1974

$$B = B_0 \boxed{e^{-\frac{E}{kT}}} f(T, T_{pk}, E_D)$$

T = temperature (K)
k = Boltzmann constant (eV K^{-1})
E = Activation energy (eV)
T_{pk} = Temperature of peak performance
E_D = Deactivation energy (eV)
(J H vant Hoff 1884, S Arrhenius 1889)

- Surely there is more to thermal responses?
  - Oxygen limitation
  - Complexity of metabolic network
  - Hormonal regulation



Johnson et al 1974

$$B = B_0 \boxed{e^{-\frac{E}{kT}}} f(T, T_{pk}, E_D)$$

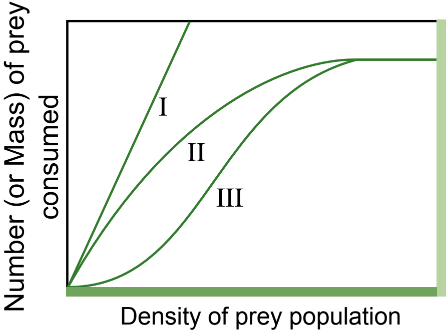
T = temperature (K)
k = Boltzmann constant (eV K^{-1})
E = Activation energy (eV)
T_{pk} = Temperature of peak performance
E_D = Deactivation energy (eV)
(J H vant Hoff 1884, S Arrhenius 1889)

- Surely there is more to thermal responses?
  - Oxygen limitation
  - Complexity of metabolic network
  - Hormonal regulation
- *What about alternative models?*

EXAMPLE 2: FUNCTIONAL RESPONSES

$$f(x_R) = \frac{ax_R^{q+1}}{1+hax_R^{q+1}} \text{ (Holling, 1959)}$$

x_R = Resource density (Mass / Area or Volume)
a = Search rate (Area or Volume / Time)
h = Handling time
q = Shape parameter (dimensionless)



Note that:

- NLLS fitting can yield $h < 0$, $q < 0$, or both
- $h < 0$ is biologically impossible but indicates an upward curving response
- $q < 0$ is biologically unlikely as it indicates a decline in search rate with resource density (but is useful as a measure of deviation away from a type III response)

NLLS FITTING IN R

So the general procedure is:

1. Start with an initial value for each parameter in the model

NLLS FITTING IN R

So the general procedure is:

1. Start with an initial value for each parameter in the model
2. Generate the curve defined by the initial values

NLLS FITTING IN R

So the general procedure is:

1. Start with an initial value for each parameter in the model
2. Generate the curve defined by the initial values
3. Calculate the residual sum-of-squares (rss)

NLLS FITTING IN R

So the general procedure is:

1. Start with an initial value for each parameter in the model
2. Generate the curve defined by the initial values
3. Calculate the residual sum-of-squares (rss)
4. Adjust the parameters to make the curve come closer to the data points. This the tricky part — more on this in the next slide

So the general procedure is:

1. Start with an initial value for each parameter in the model
2. Generate the curve defined by the initial values
3. Calculate the residual sum-of-squares (rss)
4. Adjust the parameters to make the curve come closer to the data points. This the tricky part — more on this in the next slide
5. Adjust the parameters again so that the curve comes even closer to the points (rss decreases)

NLLS FITTING IN R

So the general procedure is:

1. Start with an initial value for each parameter in the model
2. Generate the curve defined by the initial values
3. Calculate the residual sum-of-squares (rss)
4. Adjust the parameters to make the curve come closer to the data points. This the tricky part — more on this in the next slide
5. Adjust the parameters again so that the curve comes even closer to the points (rss decreases)
6. Repeat 4–5

NLLS fitting in R

So the general procedure is:

1. Start with an initial value for each parameter in the model
2. Generate the curve defined by the initial values
3. Calculate the residual sum-of-squares (rss)
4. Adjust the parameters to make the curve come closer to the data points. This the tricky part — more on this in the next slide
5. Adjust the parameters again so that the curve comes even closer to the points (rss decreases)
6. Repeat 4–5
7. Stop simulations when the adjustments make virtually no difference to the rss

NLLS FITTING IN R

The *tricky part — adjust parameters to make curve come closer to the data points* (step 4) has at least two algorithms:

- The Gauss-Newton algorithm is the default in the `nls` package (part of the `stats` base package) — good in many cases, but doesn't work very well if the model is mathematically weird (the optimization landscape is difficult) and the starting values for parameters are far-off-optimal

NLLS FITTING IN R

The *tricky part — adjust parameters to make curve come closer to the data points* (step 4) has at least two algorithms:

- The Gauss-Newton algorithm is the default in the `nls` package (part of the `stats` base package) — good in many cases, but doesn't work very well if the model is mathematically weird (the optimization landscape is difficult) and the starting values for parameters are far-off-optimal

- The Levenberg-Marquardt (LM) switches switches between Gauss-Newton and "gradient descent" and is more robust against starting values that are far-off-optimal — available in `R` through the the `minpack.lm` package
  `http://cran.r-project.org/web/packages/minpack.lm`

NLLS FITTING IN R

The *tricky part — adjust parameters to make curve come closer to the data points* (step 4) has at least two algorithms:

- The Gauss-Newton algorithm is the default in the `nls` package (part of the `stats` base package) — good in many cases, but doesn't work very well if the model is mathematically weird (the optimization landscape is difficult) and the starting values for parameters are far-off-optimal

- The Levenberg-Marquardt (LM) switches switches between Gauss-Newton and "gradient descent" and is more robust against starting values that are far-off-optimal — available in `R` through the the `minpack.lm` package
  `http://cran.r-project.org/web/packages/minpack.lm`

- The command is `nlsLM`

NLLS FITTING IN R

- Once the algorithm as converged (hopefully – but you may be surprised how well it usually works), you need to get the goodness of fit measures

NLLS FITTING IN R

- Once the algorithm as converged (hopefully – but you may be surprised how well it usually works), you need to get the goodness of fit measures
- First, of course, examine the fits visually

NLLS FITTING IN R

- Once the algorithm as converged (hopefully – but you may be surprised how well it usually works), you need to get the goodness of fit measures
- First, of course, examine the fits visually
- Also, report the best-fit results, including:

NLLS FITTING IN R

- Once the algorithm as converged (hopefully – but you may be surprised how well it usually works), you need to get the goodness of fit measures
- First, of course, examine the fits visually
- Also, report the best-fit results, including:
    - Sums of deviations of the data points from the final model fit (final RSS)

NLLS FITTING IN R

- Once the algorithm as converged (hopefully – but you may be surprised how well it usually works), you need to get the goodness of fit measures
- First, of course, examine the fits visually
- Also, report the best-fit results, including:
  - Sums of deviations of the data points from the final model fit (final RSS)
  - R^2

NLLS FITTING IN R

- Once the algorithm as converged (hopefully – but you may be surprised how well it usually works), you need to get the goodness of fit measures
- First, of course, examine the fits visually
- Also, report the best-fit results, including:
    - Sums of deviations of the data points from the final model fit (final RSS)
    - R^2
    - Estimated coefficients

NLLS fitting in R

- Once the algorithm as converged (hopefully – but you may be surprised how well it usually works), you need to get the goodness of fit measures
- First, of course, examine the fits visually
- Also, report the best-fit results, including:
    - Sums of deviations of the data points from the final model fit (final RSS)
    - R^2
    - Estimated coefficients
    - For each coefficient, standard error (can be used for CI's), t-statistic and corresponding (two-sided) p-value

NLLS FITTING IN R

- Once the algorithm as converged (hopefully – but you may be surprised how well it usually works), you need to get the goodness of fit measures
- First, of course, examine the fits visually
- Also, report the best-fit results, including:
  - Sums of deviations of the data points from the final model fit (final RSS)
  - R^2
  - Estimated coefficients
  - For each coefficient, standard error (can be used for CI's), t-statistic and corresponding (two-sided) p-value
- The function `summary.nls` will give you all these measures

NLLS FITTING IN R

- Once the algorithm as converged (hopefully – but you may be surprised how well it usually works), you need to get the goodness of fit measures
- First, of course, examine the fits visually
- Also, report the best-fit results, including:
  - Sums of deviations of the data points from the final model fit (final RSS)
  - R^2
  - Estimated coefficients
  - For each coefficient, standard error (can be used for CI's), t-statistic and corresponding (two-sided) p-value
- The function `summary.nls` will give you all these measures
- Remember, the precise parameter values you obtain will depend in part on the initial values chosen and the convergence criteria

NLLS FITTING IN R

- Once the algorithm as converged (hopefully – but you may be surprised how well it usually works), you need to get the goodness of fit measures
- First, of course, examine the fits visually
- Also, report the best-fit results, including:
  - Sums of deviations of the data points from the final model fit (final RSS)
  - R^2
  - Estimated coefficients
  - For each coefficient, standard error (can be used for CI's), t-statistic and corresponding (two-sided) p-value
- The function `summary.nls` will give you all these measures
- Remember, the precise parameter values you obtain will depend in part on the initial values chosen and the convergence criteria
- You may also want to compare multiple models...

NLLS ASSUMPTIONS

NLLS-regression has all the assumptions of OLS-regression:

- No (in practice, minimal) measurement error in explanatory variable (x-axis variable)

NLLS Assumptions

NLLS-regression has all the assumptions of OLS-regression:

- No (in practice, minimal) measurement error in explanatory variable (x-axis variable)
- Data have constant normal variance — errors in the y-axis are homogeneously distributed over the x-axis range

NLLS ASSUMPTIONS

NLLS-regression has all the assumptions of OLS-regression:

- No (in practice, minimal) measurement error in explanatory variable (x-axis variable)
- Data have constant normal variance — errors in the y-axis are homogeneously distributed over the x-axis range
- The measurement/observation error distribution is Gaussian — for example, what would the error distribution of this non-linear model be: $y_i = \beta_0 e^{\beta_2 x_i} + \varepsilon_i$

NLLS ASSUMPTIONS

NLLS-regression has all the assumptions of OLS-regression:

- No (in practice, minimal) measurement error in explanatory variable (x-axis variable)
- Data have constant normal variance — errors in the y-axis are homogeneously distributed over the x-axis range
- The measurement/observation error distribution is Gaussian — for example, what would the error distribution of this non-linear model be: $y_i = \beta_0 e^{\beta_2 x_i} + \varepsilon_i$
- What if the errors are not normal? — use maximum likelihood instead! (e.g., using `nlm` for optimizing/fitting)

COMPARING MODELS

- It's all about the "Likelihood" of a model:
- That is, the likelihood of a set of parameter values (of a model), θ, given outcomes x, equals the probability of those observed outcomes given those parameter values, that is,

$$\mathcal{L}(\theta|x) = P(x|\theta)$$

Comparing models

The easiest thing to do for you is to use information theory (including AIC and BIC) to compare models.
Both use estimated likelihood of a model

Comparing models

The easiest thing to do for you is to use information theory (including AIC and BIC) to compare models.

Both use estimated likelihood of a model

This is how you can calculate these (using `R` syntax):

COMPARING MODELS

The easiest thing to do for you is to use information theory (including AIC and BIC) to compare models.
Both use estimated likelihood of a model
This is how you can calculate these (using `R` syntax):

- residuals = Observations - Predictions
- `rss = sum(residuals ^ 2)`
- Then, AIC is `n * log((2 * pi) / n) + n + 2 + n * log(rss) + 2 * k`
  (*what is* `n` *and* `k`*?*)
- And BIC is `n + n * log(2 * pi) + n * log(rss / n) + (log(n)) * (k + 1)`
- For both AIC and BIC, If model **A** has AIC lower by 2-3 or more than model **B**, its better — Differences of less than 2-3 don't really matter

COMPARING MODELS

The easiest thing to do for you is to use information theory (including AIC and BIC) to compare models.
Both use estimated likelihood of a model
This is how you can calculate these (using `R` syntax):

- residuals = Observations - Predictions
- `rss = sum(residuals ^ 2)`
- Then, AIC is `n * log((2 * pi) / n) + n + 2 + n * log(rss) + 2 * k`
  (*what is* `n` *and* `k`*?*)
- And BIC is `n + n * log(2 * pi) + n * log(rss / n) + (log(n)) * (k + 1)`
- For both AIC and BIC, If model **A** has AIC lower by 2-3 or more than model **B**, its better — Differences of less than 2-3 don't really matter

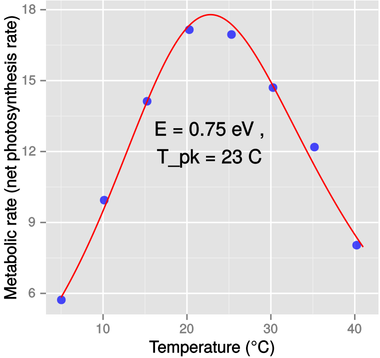
Also note that:

- R^2 = 1 - (rss/tss), where tss is total sum of squares:
  `tss = sum((Observations - mean(Predictions)) ^ 2)`

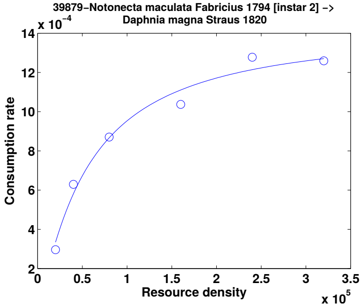
PRACTICALS: INSTRUCTIONS

- We shall start off with some simple examples (`NLSFitEx1.R`, `NLSFitEx2.R`, etc.)
- You can then choose either Practical 1 or 2, or do both!
- In each, make sure you have a good look at the data first by plotting them up in a loop
- Keep workflow organized in `Code`, `Results`, `Data` !
- You may also find your own model to fit some data related to your interests or project
- Your demonstrators and I will help you get started in any case

- Use `nls` (or `nlsLM`) to fit the `ThermRespData.csv` dataset to the model: $B = B_0 \boxed{e^{-\frac{E}{kT}}} f(T, T_{pk}, E_D)$
- Plot the data and output coefficient estimates and fit stats to a file
- Here's an example of how a fitted curve looks (only two fitted parameters are shown):



- Use `nls` (or `nlsLM`) to fit `CRat.csv` dataset to the model:
  $f(x_R) = c = \frac{a x_R^{q+1}}{1 + h a x_R^{q+1}}$
  (c is consumption rate)
- Plot the data and output coefficient estimates and fit stats to a file
- Here's an example of how a fitted curve looks :



You can also use mixed-effects models in NLLS

- You can use mixed-effects modelling with NLLS in R

You can also use mixed-effects models in NLLS

- You can use mixed-effects modelling with NLLS in R
- The package is `nlme` `https://stat.ethz.ch/R-manual/R-devel/library/nlme/html/nlme.html`

You can also use mixed-effects models in NLLS

- You can use mixed-effects modelling with NLLS in R
- The package is `nlme` `https://stat.ethz.ch/R-manual/R-devel/library/nlme/html/nlme.html`
- You are probably stuck with the Gauss-Newton algorithm with `nlme` though...

READINGS

- Motulsky, Harvey, and Arthur Christopoulos. Fitting models to biological data using linear and nonlinear regression: a practical guide to curve fitting. OUP USA, 2004.
- Johnson, J. B. & Omland, K. S. 2004 Model selection in ecology and evolution. Trends Ecol. Evol. 19, 101–108.