Nonlinear Regression

30.11.2016

Goals of Today's Lecture

- Understand the difference between linear and nonlinear regression models.
- See that not all functions are linearizable.
- Get an understanding of the fitting algorithm in a statistical sense (i.e. fitting many linear regressions).
- Know that tests etc. are based on approximations and be able to interpret computer output, profile t-plots and profile traces.

Nonlinear Regression Model

The **nonlinear regression model** is

$$\begin{aligned} Y_i &= h(x_i^{(1)}, x_i^{(2)}, \ldots, x_i^{(m)} \,;\, \theta_1, \theta_2, \ldots, \theta_p) + E_i \\ &= h(\underline{x}_i; \underline{\theta}) + E_i. \end{aligned}$$

where

- E_i are the error terms, $E_i \sim \mathcal{N}(0, \sigma^2)$ independent
- $x^{(1)}, \ldots, x^{(m)}$ are the predictors
- $\theta_1, \ldots, \theta_p$ are the parameters
- h is the regression function, "any" function.
  h is a function of the predictors and the parameters.

Comparison with linear regression model

- In contrast to the linear regression model we now have a **general function** h.

  In the linear regression model we had

  $$h(\underline{x}_i; \underline{\theta}) = \underline{x}_i^T \underline{\theta}$$

  (there we denoted the parameters by $\underline{\beta}$).

- Note that in linear regression we required that the **parameters** appear in **linear form**.

- In nonlinear regression, we don't have that restriction anymore.

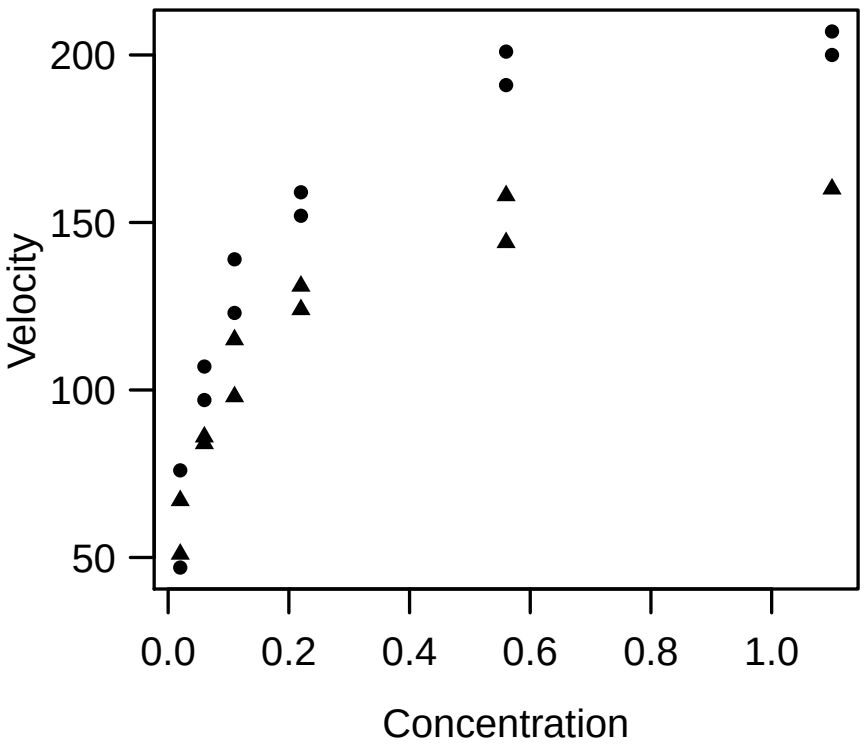
Example: Puromycin

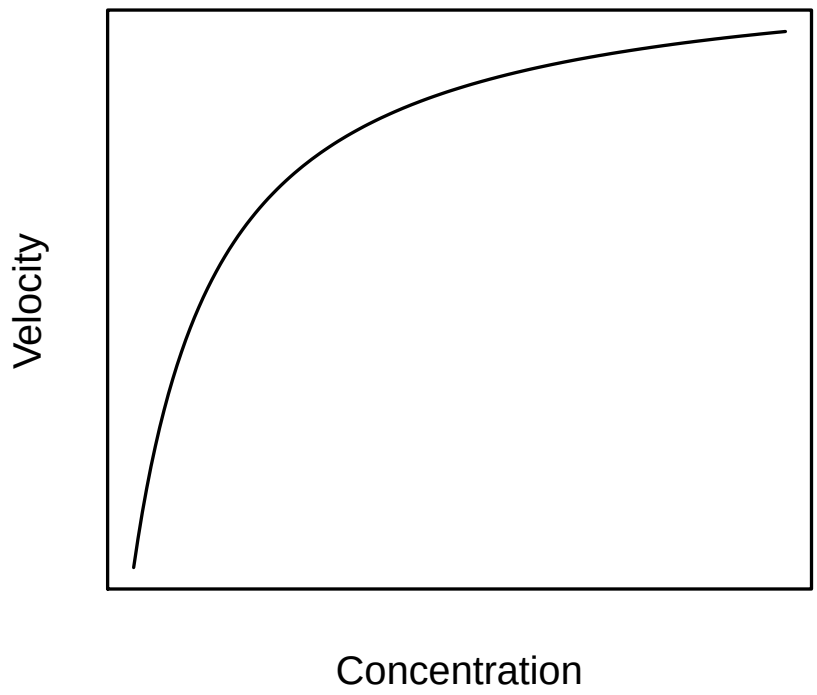
- The speed of an enzymatic reaction depends on the concentration of a substrate.
- The initial speed is the response variable (Y). The concentration of the substrate is used as predictor (x). Observations are from different runs.
- Model with *Michaelis-Menten* function

$$h(x; \underline{\theta}) = \frac{\theta_1 x}{\theta_2 + x}.$$

- Here we have one predictor x (the concentration) and two parameters: θ_1 and θ_2.
- Moreover, we observe two groups: One where we treat the enzyme with Puromycin and one without treatment (control group).

Illustration: Puromycin (two groups)





Left:

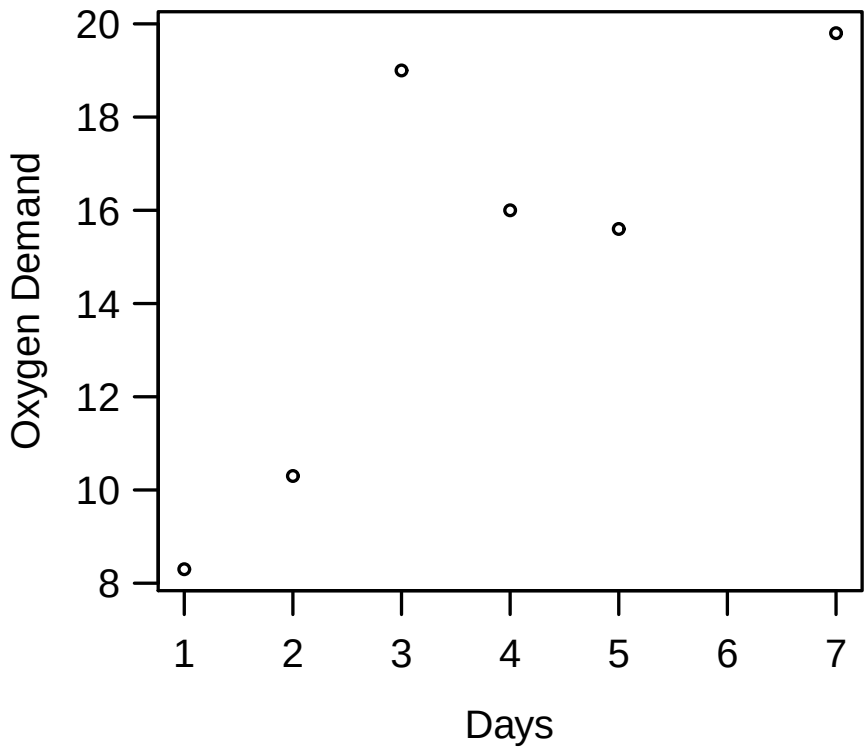
Data (• treated enzyme; △ untreated enzyme)

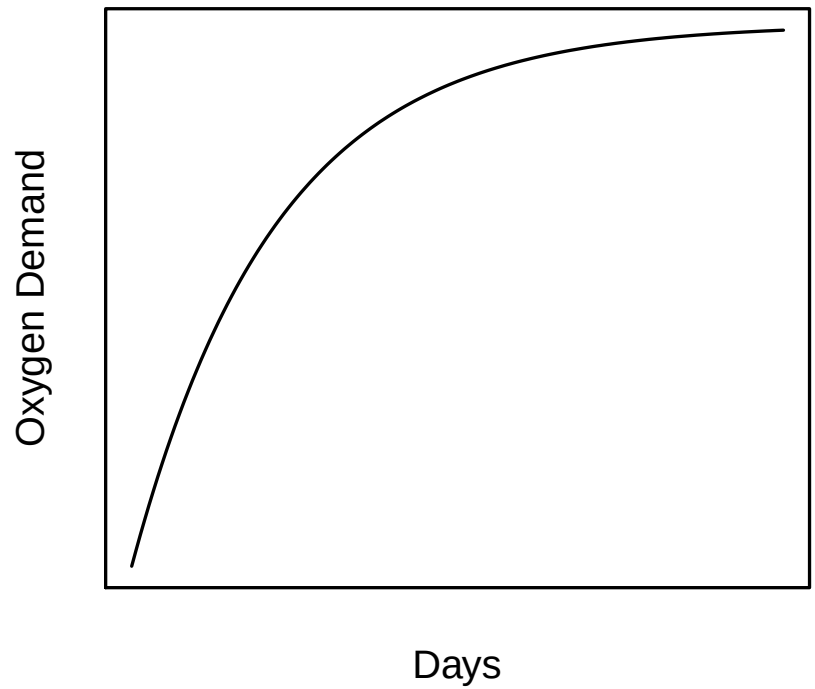
Right: Typical shape of the regression function.

Example: Biochemical Oxygen Demand (BOD)

Model the biochemical oxygen demand (Y) as a function of the incubation time (x)

$$h(x;\underline{\theta}) = \theta_1 \left(1 - e^{-\theta_2 x}\right).$$





Linearizable Functions

Sometimes (but **not always**), the function h is **linearizable**.

Example

- Let's forget about the error term E for a moment. Assume we have

$$\begin{aligned} y = h(x;\underline{\theta}) &= \theta_1 \exp\{\theta_2/x\} \\ &\Longleftrightarrow \\ \log(y) &= \log(\theta_1) + \theta_2 \cdot (1/x) \end{aligned}$$

- We can rewrite this as

$$\widetilde{y} = \widetilde{\theta}_1 + \widetilde{\theta}_2 \cdot \widetilde{x},$$

  where $\widetilde{y} = \log(y)$, $\widetilde{\theta}_1 = \log(\theta_1)$, $\widetilde{\theta}_2 = \theta_2$ and $\widetilde{x} = 1/x$.

- If we use this linear model, we assume **additive errors** E_i

$$\widetilde{Y}_i = \widetilde{\theta}_1 + \widetilde{\theta}_2 \widetilde{x}_i + E_i.$$

- This means that we have **multiplicative errors** on the original scale

$$Y_i = \theta_1 \exp\{\theta_2/x_i\} \cdot \exp\{E_i\}.$$

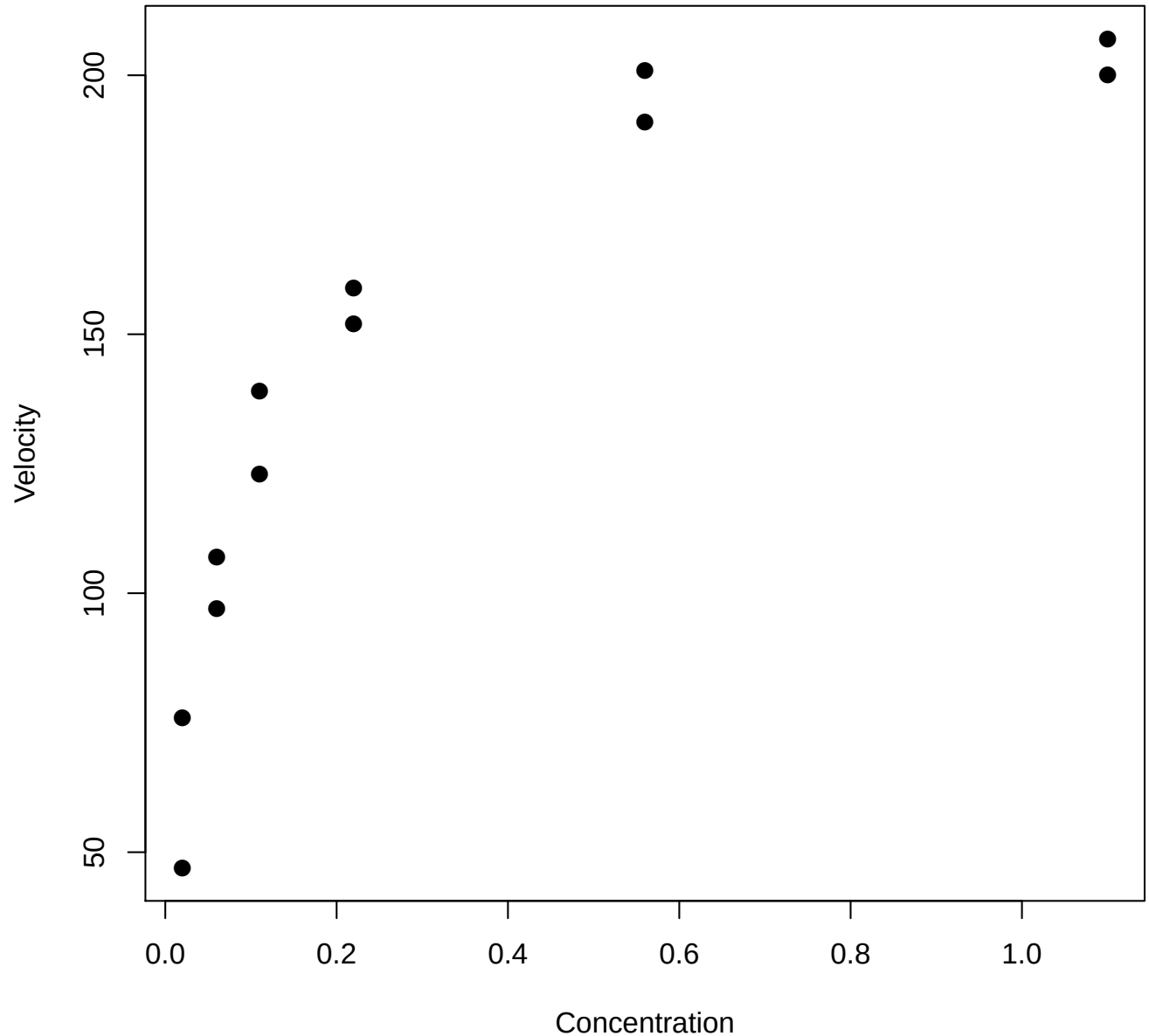
- This is **not** the same as using a nonlinear model on the original scale (it would have additive errors!).

- Hence, **transformations of Y modify the model with respect to the error term**.

- In the Puromycin example: Do not linearize because error term would fit worse (see next slide).

- Hence, for those cases where h is linearizable, it depends on the data if it's advisable to do so or to perform a nonlinear regression.

Puromycin: Treated enzyme



Parameter Estimation

Let's now assume that we really want to fit a **nonlinear** model.

Again, we use **least squares**. Minimize

$$S(\underline{\theta}) := \sum_{i=1}^{n} (Y_i - \eta_i(\underline{\theta}))^2 ,$$

where

$$\eta_i(\underline{\theta}) := h(\underline{x}_i; \underline{\theta})$$

is the fitted value for the ith observation ($\underline{x}_i$ is fixed, we only vary the parameter vector $\underline{\theta}$).

Geometrical Interpretation

First we recall the situation for **linear regression**.

- By applying least squares we are looking for the parameter vector $\underline{\theta}$ such that

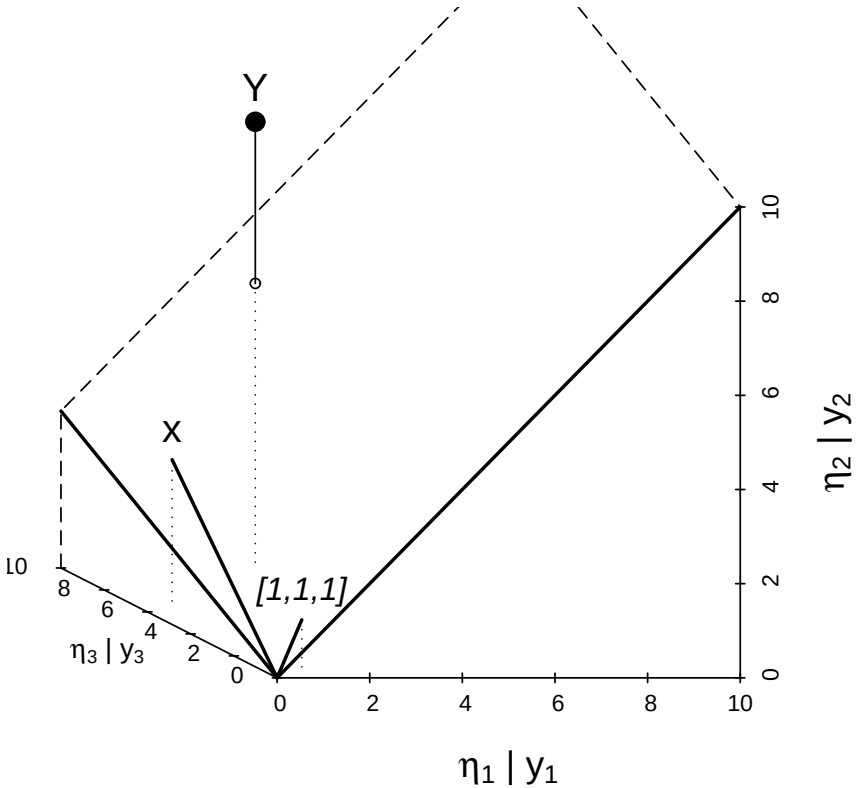
$$\|\underline{Y} - X\underline{\theta}\|_2^2 = \sum_{i=1}^{n} \left(Y_i - \underline{x}_i^T \underline{\theta}\right)^2$$

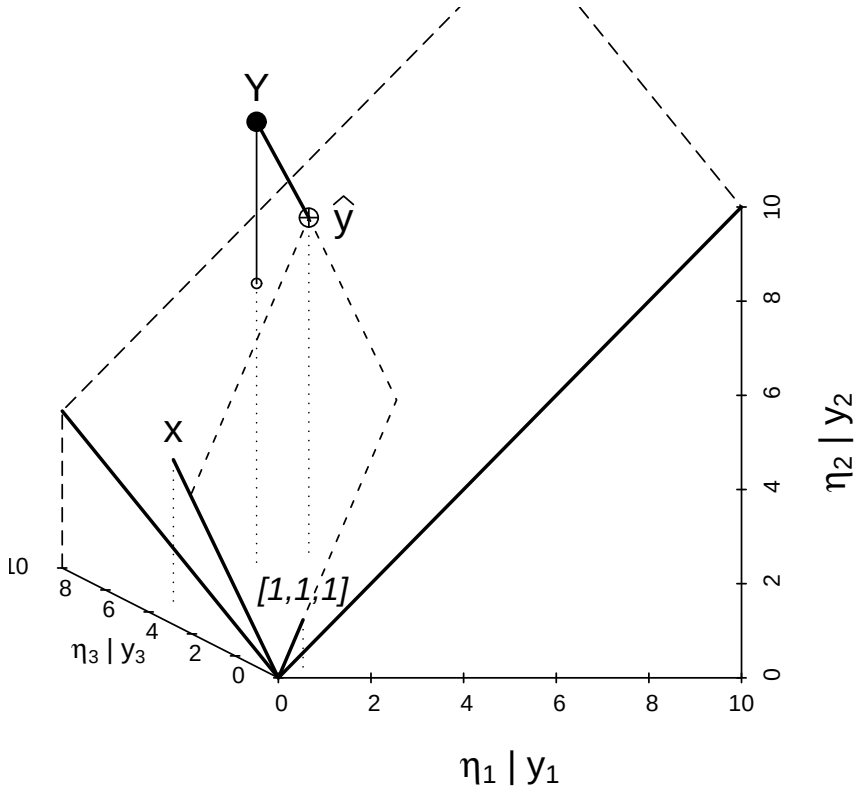
  is minimized.

- Or in other words: We are looking for the point on the plane spanned by the columns of X that is **closest** to $\underline{Y} \in \mathbb{R}^n$.

- This is nothing else than **projecting** $\underline{Y}$ on that specific plane.

Linear Regression: Illustration of Projection





Situation for **nonlinear regression**

- Conceptually, the same holds true for nonlinear regression.

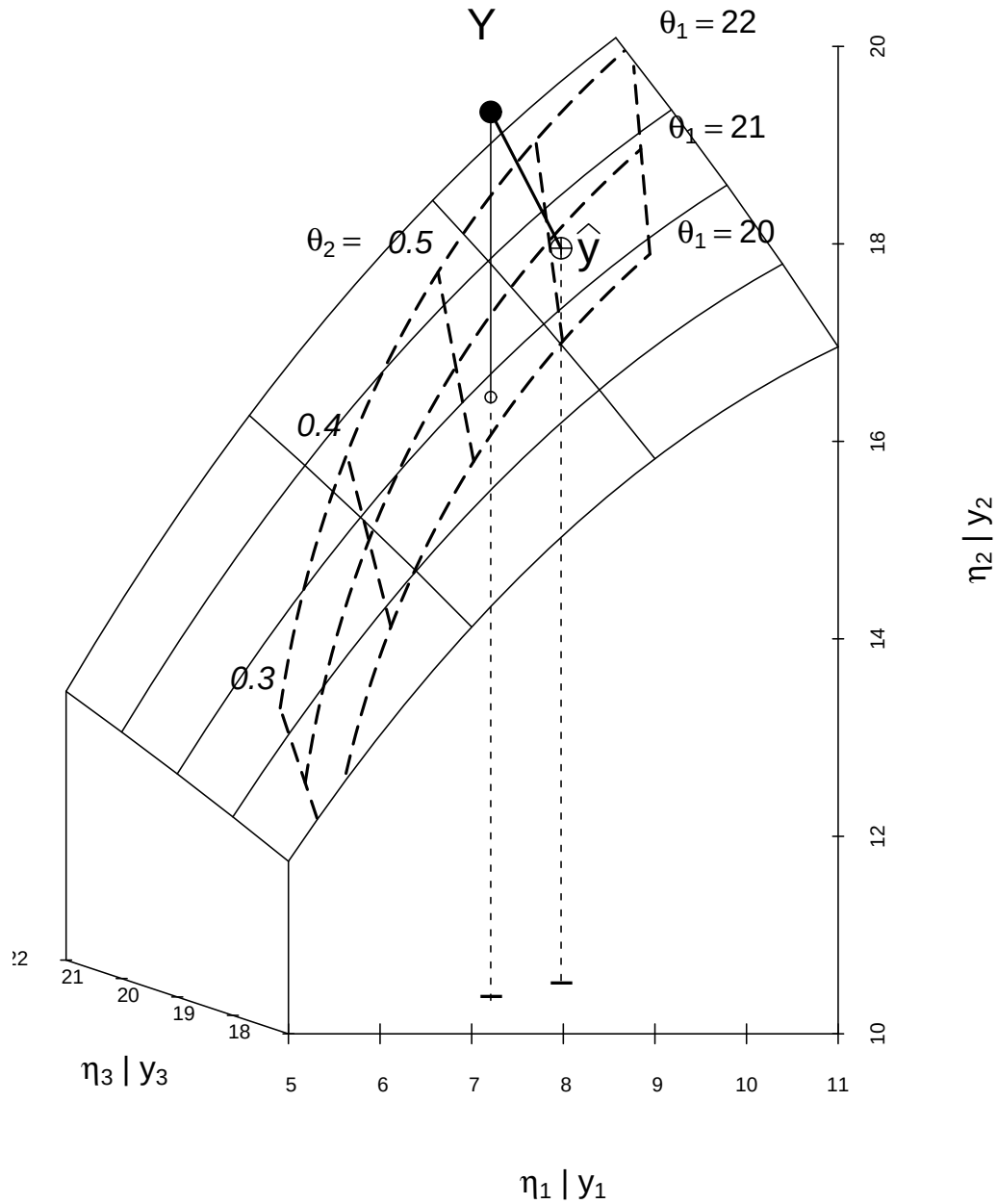
- The difference is: All possible points do **not** lie on a plane anymore, but on a **curved surface**, the so called **model surface** defined by

$$\underline{\eta}(\underline{\theta}) \in \mathbb{R}^n$$

  when varying the parameter vector $\underline{\theta}$.

- This is a p-dimensional surface because we parameterize it with p parameters.

Nonlinear Regression: Projection on Curved Surface



Computation

- Unfortunately, we can **not** derive a closed form solution for the parameter estimate $\widehat{\underline{\theta}}$.
- Iterative procedures are therefore needed.
- We use a **Gauss-Newton** approach.
- Starting from an **initial value** $\underline{\theta}^{(0)}$, the idea is to **approximate** the model surface by a **plane**, to perform a projection on that plane and to iterate many times.
- Remember $\eta : \mathbb{R}^p \to \mathbb{R}^n$. Define $n \times p$ matrix

$$A_i^{(j)}(\underline{\theta}) = \frac{\partial \eta_i(\underline{\theta})}{\partial \theta_j}.$$

This is the **Jacobi-matrix** containing all partial derivatives.

Gauss-Newton Algorithm

More formally, the Gauss-Newton algorithm is as follows

- Start with **initial value** $\underline{\widehat{\theta}}^{(0)}$
- For $l = 1, 2, \ldots$

  Calculate tangent plane of $\underline{\eta}(\underline{\theta})$ in $\underline{\widehat{\theta}}^{(l-1)}$:

  $$\underline{\eta}(\underline{\theta}) \approx \underline{\eta}(\underline{\widehat{\theta}}^{(l-1)}) + A(\underline{\widehat{\theta}}^{(l-1)}) \cdot (\underline{\theta} - \underline{\widehat{\theta}}^{(l-1)})$$

  Project $\underline{Y}$ on tangent plane $\rightsquigarrow \underline{\widehat{\theta}}^{(l)}$
  Projection is a linear regression problem, see blackboard.

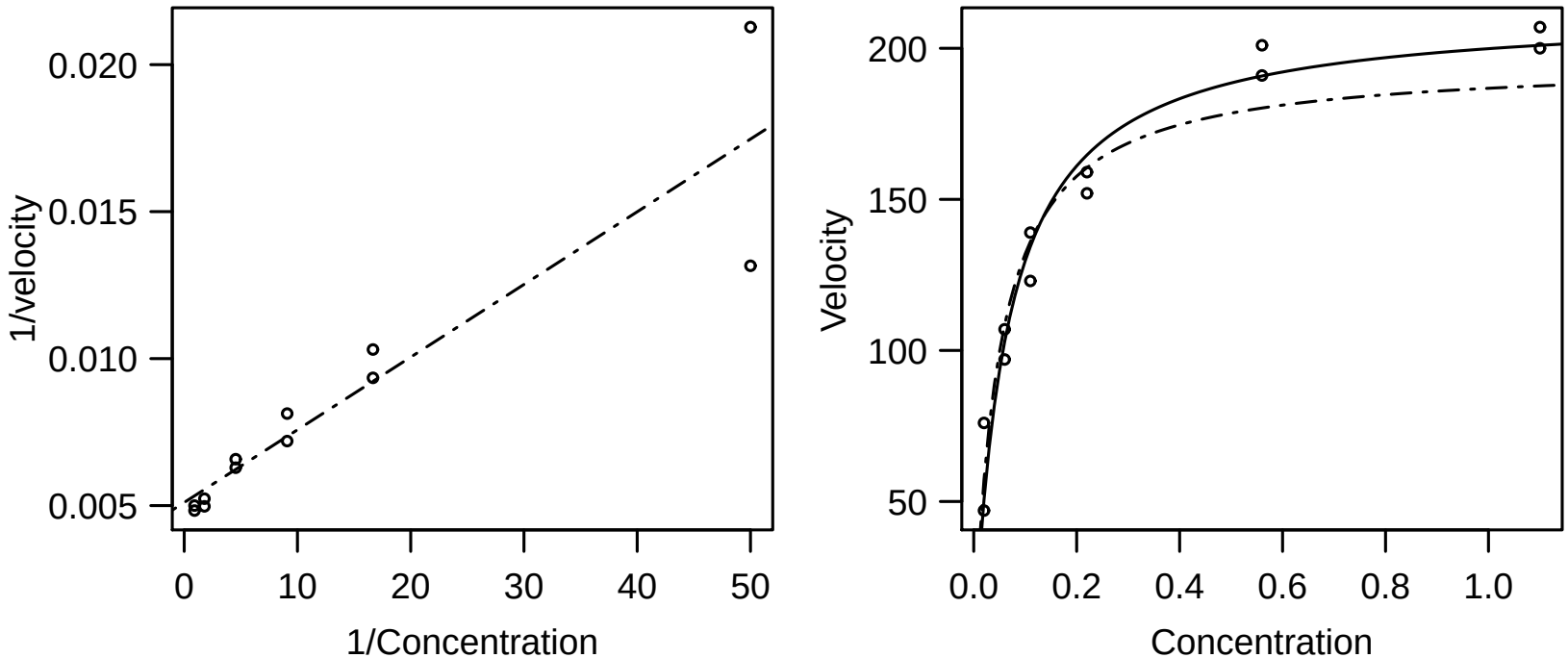
  Next l
- Iterate until convergence

Initial Values

How can we get initial values?

- Available knowledge
- Linearized version (see Puromycin)
- Interpretation of parameters (asymptotes, half-life, . . .), "fitting by eye".
- Combination of these ideas (e.g., conditional linearizable functions)

Example: Puromycin (only treated enzyme)



Dashed line: Solution of linearized problem.

Solid line: Solution of the nonlinear least squares problem.

Approximate Tests and Confidence Intervals

- Algorithm "only" gives us $\underline{\widehat{\theta}}$.
- How accurate is this estimate in a statistical sense?
- In linear regression we knew the (exact) distribution of the estimated parameters (remember animation!).
- In nonlinear regression the situation is more complex in the sense that we only have **approximate results**.
- It can be shown that

$$\widehat{\theta}_j \overset{approx.}{\sim} \mathcal{N}(\theta_j, V_{jj})$$

for some matrix V (V_{jj} is the jth diagonal element).

- Tests and confidence intervals are then constructed as in the linear regression situation, i.e.

$$\frac{\widehat{\theta}_j - \theta_j}{\sqrt{\widehat{V}_{jj}}} \overset{approx.}{\sim} t_{n-p}.$$

- The reason why we basically have the same result as in the linear regression case is because the algorithm is based on (many) linear regression problems.

- Once converged, the solution is not only the solution to the nonlinear regression problem but also for the linear one of the last iteration.

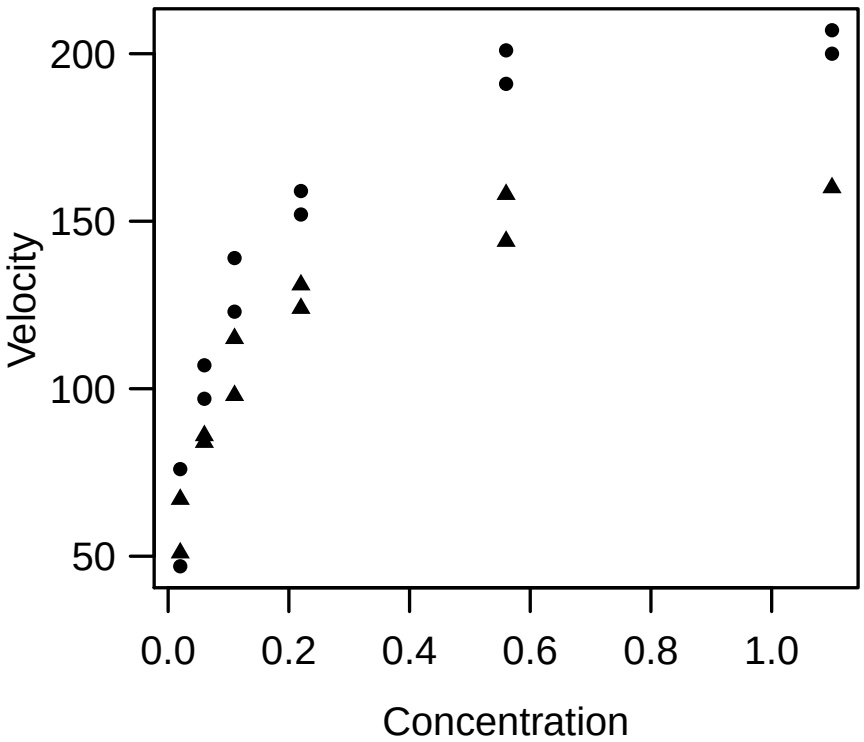
  In fact

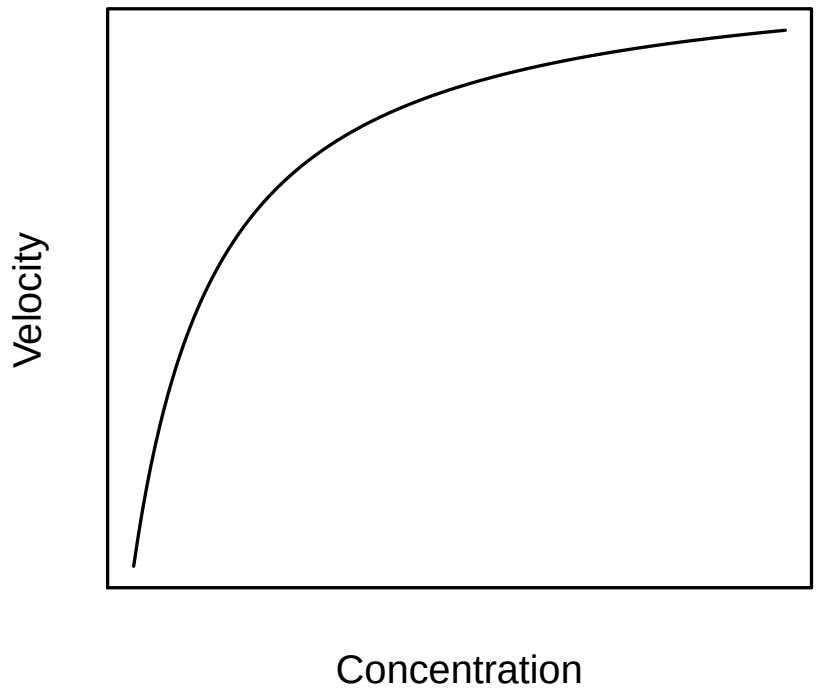
$$\widehat{V} = \widehat{\sigma}^2 (\widehat{A}^T \widehat{A})^{-1},$$

  where $\widehat{A} = A(\widehat{\theta})$.

Example Puromycin (two groups)

Remember, we originally had **two groups** (treatment and control)





Question: Do the two groups need different regression parameters?

- To answer this question we set up a model of the form

$$Y_i = \frac{(\theta_1 + \theta_3 z_i)x_i}{\theta_2 + \theta_4 z_i + x_i} + E_i,$$

  where z is the **indicator variable** for the treatment ($z_i = 1$ if treated, $z_i = 0$ otherwise).

- E.g., if θ_3 is nonzero we have a different asymptote for the treatment group ($\theta_1 + \theta_3$ vs. only θ_1 in the control group).

- Similarly for θ_2, θ_4.

- Let's fit this model to data.

Computer Output

```
Formula: velocity ~ (T1 + T3 * (treated == T)) * conc/(T2 +
T4 * (treated == T) + conc)

  Parameters:
       Estimate  Std.Error  t value  Pr(>|t|)
  T1    160.280      6.896   23.242  2.04e-15
  T2      0.048      0.008    5.761  1.50e-05
  T3     52.404      9.551    5.487  2.71e-05
  T4      0.016      0.011    1.436     0.167
```

- We only get a significant test result for θ_3 ($\rightsquigarrow$ **different asymptotes**) and not θ_4.
- A 95%-**confidence interval** for θ_3 (=difference between asymptotes) is

$$52.404 \pm q_{0.975}^{t_{19}} \cdot 9.551 = [32.4,\, 72.4],$$

where $q_{0.975}^{t_{19}} \approx 2.09$.

More Precise Tests and Confidence Intervals

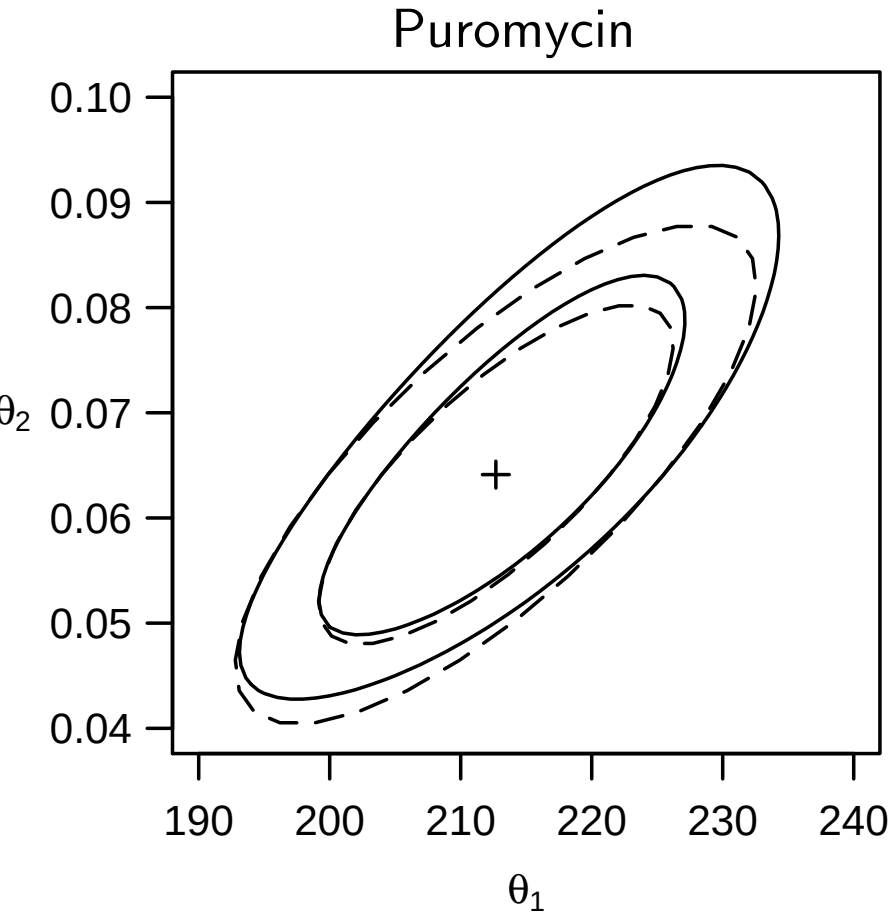
- Tests etc. that we have seen so far are only "usable" if linear approximation of the problem around the solution $\widehat{\underline{\theta}}$ is good.
- We can use another approach that is better (but also more complicated).
- In linear regression we had a quick look at the F-test for testing simultaneous null-hypotheses. This is also possible here.
- Say we have the null hypothesis $H_0 : \underline{\theta} = \underline{\theta}^*$ **(whole vector)**.

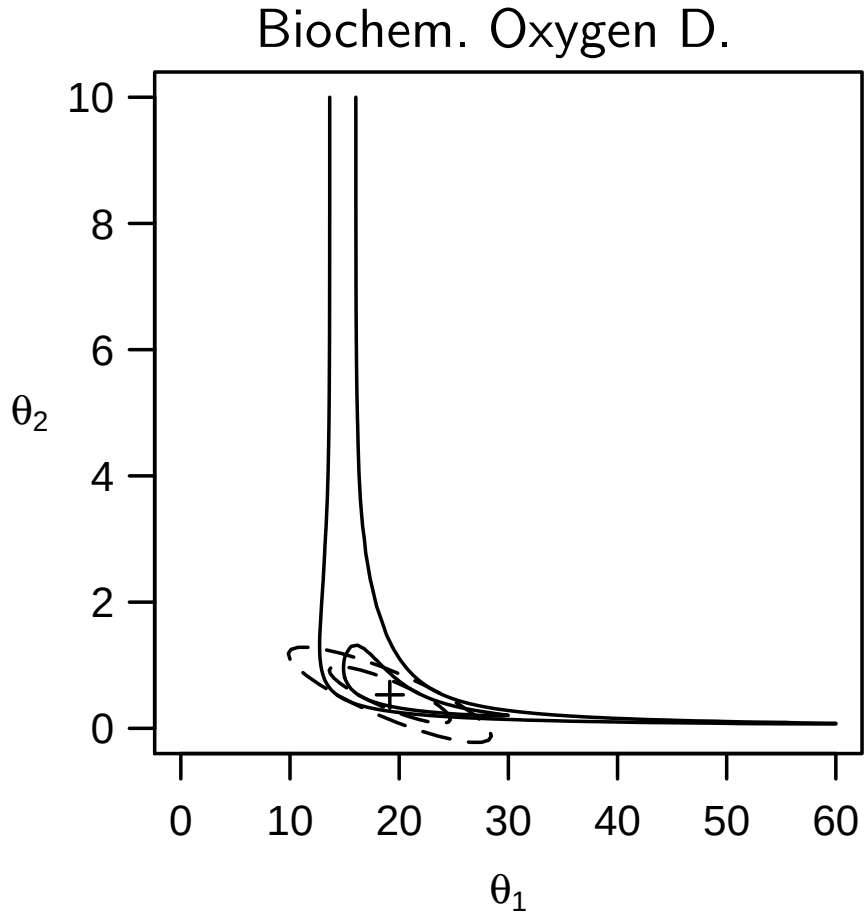
  Fact: Under H_0 it holds

$$T = \left(\frac{n-p}{p}\right) \frac{S(\underline{\theta}^*) - S(\widehat{\underline{\theta}})}{S(\widehat{\underline{\theta}})} \overset{approx.}{\sim} F_{p,n-p}.$$

- We still have only an "approximate" result. But this approximation is **(much) better** (more accurate) than the one that is based on the linear approximation.
- This can now be used to construct **confidence regions** by searching for all **vectors** $\underline{\theta}^*$ that are **not** rejected using this test (as before).
- If we only have two parameters it's easy to illustrate these confidence regions.
- Using linear regression it's also possible to derive confidence regions (for several parameters). We haven't seen this in detail.
- This approach can also be used here (because we use a linear approximation in the algorithm, see also later).

Confidence Regions: Examples





- Dashed: Confidence Region (80% and 95%) based on linear approx.
- Solid: Approach with F-test from above (more accurate).
- "+" is parameter estimate.

What if we only want to test a **single component** θ_k?

- Assume we want to test $H_0 : \theta_k = \theta_k^*$.
- Now fix $\theta_k = \theta_k^*$ and minimize $S(\underline{\theta})$ with respect to θ_j, $j \neq k$.
- Denote the minimum by $\widetilde{S}_k(\theta_k^*)$.
- **Fact:** Under H_0 it holds that

$$\widetilde{T}_k(\theta_k^*) = (n-p)\,\frac{\widetilde{S}_k(\theta_k^*) - S(\widehat{\underline{\theta}})}{S(\widehat{\underline{\theta}})} \overset{approx.}{\sim} F_{1,n-p},$$

or similarly

$$T_k(\theta_k^*) = \text{sign}(\widehat{\theta}_k - \theta_k^*)\,\frac{\sqrt{\widetilde{S}_k(\theta_k^*) - S(\widehat{\underline{\theta}})}}{\widehat{\sigma}} \overset{approx.}{\sim} t_{n-p}.$$

- Our first approximation was based on the linear approximation and we got a test of the form

$$\delta_k(\theta_k^*) = \frac{\widehat{\theta}_k - \theta_k^*}{\widehat{\text{s.e.}}(\widehat{\theta}_k)} \overset{\text{approx.}}{\sim} t_{n-p},$$

  where $\widehat{\text{s.e.}}(\widehat{\theta}_k) = \sqrt{\widehat{V}_{jj}}$.

  This is what we saw in the computer output.

- The new approach with $T_k(\theta_k^*)$ answers the same question (i.e., we do a test for a single component).

- The approximation of the new approach is (typically) much **more accurate**.

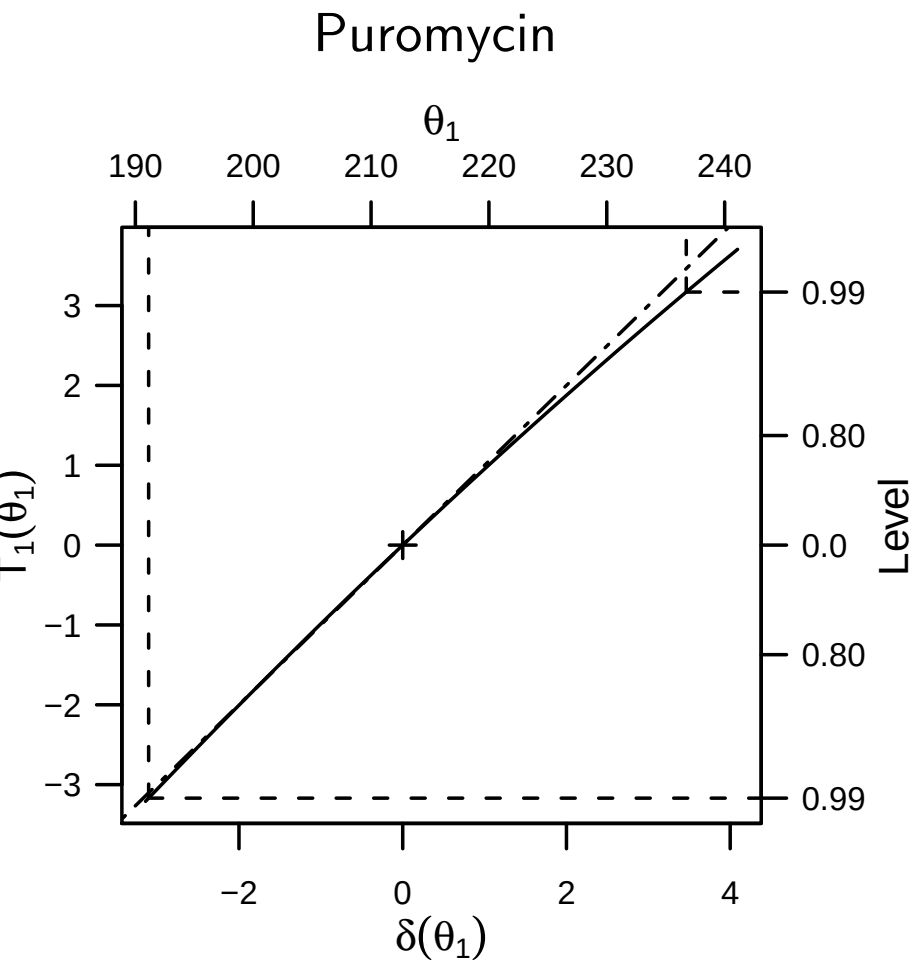
- We can compare the different approaches using plots.

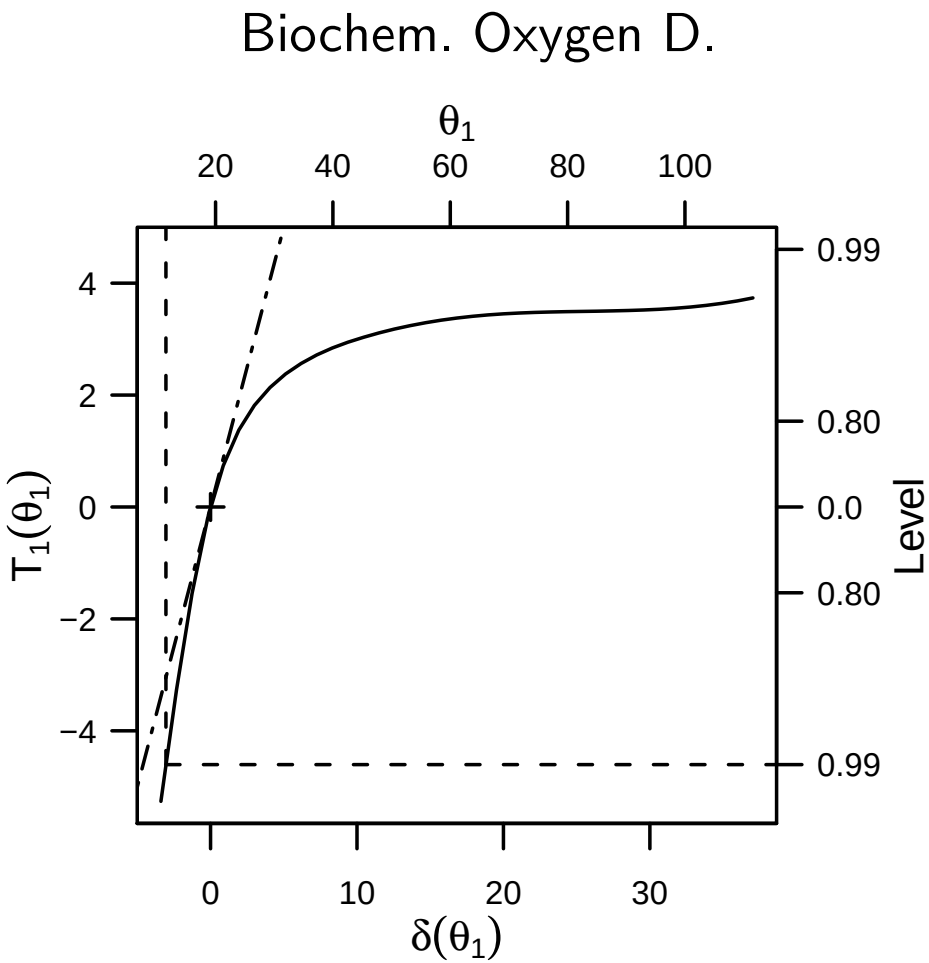
Profile t-Plots and Profile Traces

The **profile t-plot** is defined as the plot of $T_k(\theta_k^*)$ against $\delta_k(\theta_k^*)$ (when varying θ_k^*).

- Remember: The two tests (T_k and δ_k) test the **same thing**.
- If they behave similarly, we would expect the same answers, hence the plot should show a **diagonal** (intercept 0, slope 1).
- Strong deviations from the diagonal indicate that the linear approximation at the solution is not suitable and that the problem is very **non-linear** in a neighborhood of $\widehat{\theta}_k$.

Profile t-Plots: Examples

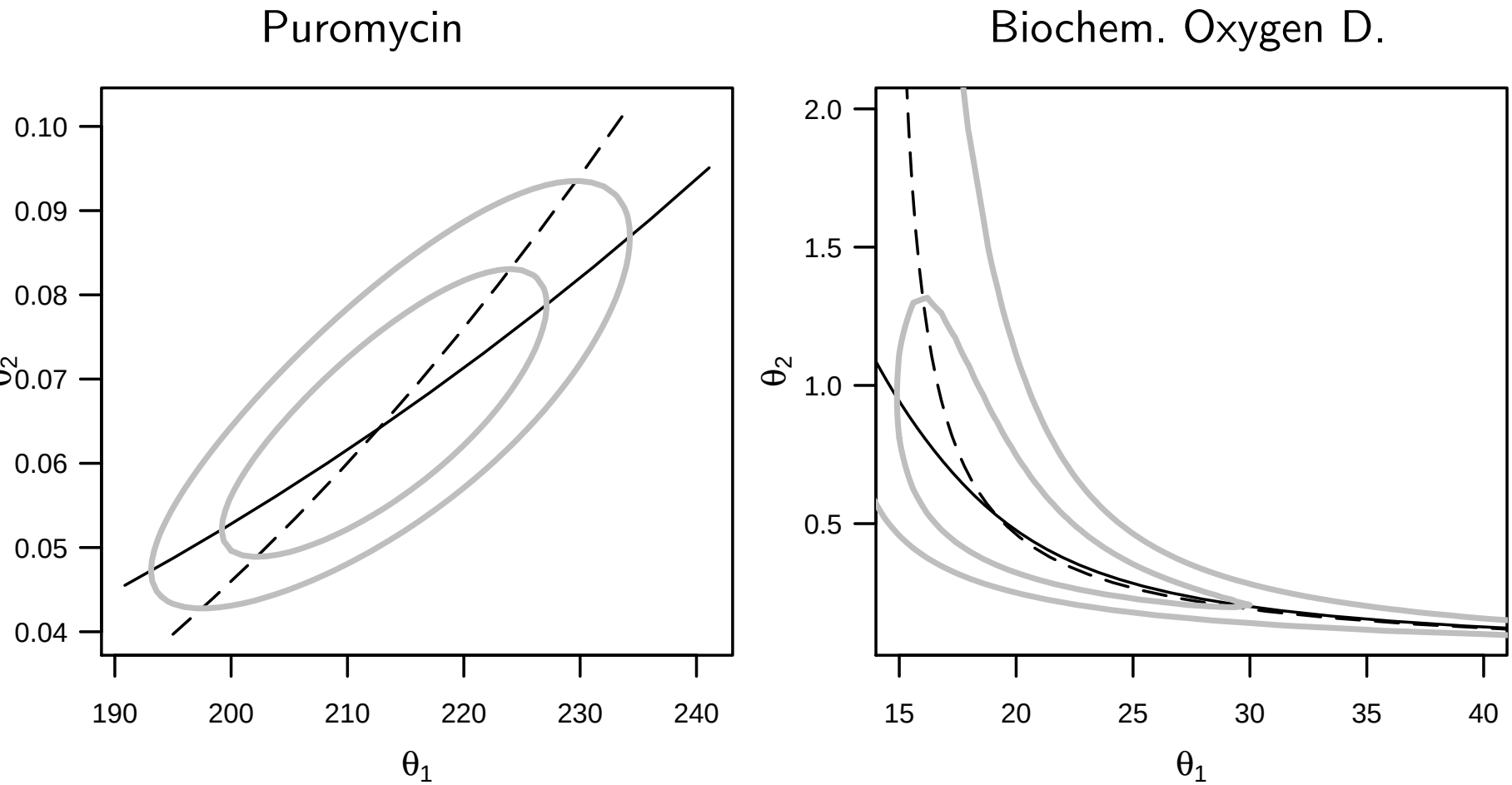




Profile Traces

- Select a **pair of parameters**: θ_j, θ_k; $j \neq k$.
- Keep θ_k **fixed**, estimate remaining parameters: $\widetilde{\theta}_j(\theta_k)$.
- This means: When varying θ_k we can plot the estimated $\widetilde{\theta}_j$ (and vice versa)
- Illustrate these two curves on a **single plot**.
- What can we learn from this?
  - The **angle between the two curves** is a measure for the **correlation between estimated parameters**. The smaller the angle, the higher the correlation.
  - In the linear case we would see **straight lines**. Deviations are an indication for **nonlinearities**.
- Correlated parameter estimates influence each other strongly and make estimation difficult.

Profile Traces: Examples



Grey lines indicate confidence regions (80% and 95%).

Parameter Transformations

In order to improve the linear approximation (and therefore improve convergence behaviour) it can be useful to transform the parameters.

- Transformations of **parameters** do **not** change the model, but
  - the **quality of the linear approximation**, influencing the **difficulty of computation** and the **validity of approximate confidence regions**.
  - the **interpretation** of the parameters.
- Typically, finding good transformations is hard.
- Results can be transformed back to original parameters. Then, transformation is just a technical step to solve the problem.

- Use parameter transformations to **avoid side constraints**, e.g.

$$\theta_j > 0 \quad \longrightarrow \quad \text{Use } \theta_j = \exp\{\phi_j\},\ \phi_j \in \mathbb{R}$$

$$\theta_j \in (a, b) \quad \longrightarrow \quad \text{Use } \theta_j = a + \frac{b-a}{1+\exp\{-\phi_j\}},\ \phi_j \in \mathbb{R}$$

Summary

- Nonlinear regression models are widespread in chemistry.
- Computation needs **iterative procedure**.
- Simplest tests and confidence intervals are based on **linear approximations** around solution $\widehat{\underline{\theta}}$.
- If linear approximation is not very accurate, problems can occur. Graphical tools for checking linearities are **profile t-plots** and **profile traces**.
- Tests and confidence intervals based on F-test are more accurate.
- **Parameter transformations** can help reducing these problems.