Math 340 • Fall 2026 • Victor Matveev

Section 4.1: Least Squares Parameter Estimation

Simplest Case: Linear Model (Linear Regression)

Suppose our data comes from an underlying linear model (process): $\boldsymbol{y_{model}(t;p,q) = p \cdot t + q}$

This time-dependent process also depends on two parameters (p, q) representing the slope and the intercept of this linear relationship, which may be unknown, and which we want to determine. This is a general problem of **parameter estimation**, more colloquially termed **data fitting**.

If we don't know the model parameters, the only way to determine them is to use experimental measurements of $y(t)$ for a set of values of argument *t*, which represents our data: $\boldsymbol{y_{data}(t_k) \equiv y_k}$ (we suppress subscript "data" for brevity of notation). Thus, our data is described by *N* pairs $\{(t_k, y_k)\}$ (note: there could be several values of *y* for the same t_k).

Now, any real-world data includes random measurement noise (or other sources of statistical error), therefore it will not achieve perfect fit to the underlying model process that produced this data: in other words, $\boldsymbol{y_{model}(t_k;p,q) \neq y_{data}(t_k)}$ for any values of *p* and *q*, but includes an error (a residual) which we denote as $\boldsymbol{r_k}$:

$$y_{\text{model}}(t_k;p,q) = y_{\text{data}}(t_k) + r_k$$

So how do we determine parameters in this case? A typical approach (not obvious but justified more in later lectures) is to **minimize the sum of squares of these residuals** (deviations r_k between model and data):

$$L(p,q) \equiv \sum_{k=1}^{N=3} r_k^2 \equiv \sum_{k=1}^{N} \underbrace{\left[y_{\text{model}}(t_k;p,q) - y_{\text{data}}(t_k) \right]}_{\text{"Residuals" (measurement errors)}}^2$$

$$= \sum_{k=1}^{N} \left[\underbrace{p \cdot t_k + q}_{\text{Model}} - \underbrace{y_k}_{\text{Data}} \right]^2$$

The textbook starts with the geometric approach on how to achieve this. However, geometric approach may look abstract initially, so let us start with a direct approach from calculus: a minimum of a smooth function requires that all partial derivatives are zero (note that $L(p, q)$ is a paraboloid in p and q, with a single unique minimum). Thus, we need to solve:

$$\begin{cases} \dfrac{\partial L}{\partial p} = \dfrac{\partial}{\partial p} \displaystyle\sum_{k=1}^{N} \left[p t_k + q - y_k \right]^2 = 2 \sum_{k=1}^{N} t_k \left[p t_k + q - y_k \right] = 2 \left(p \sum_{k=1}^{N} t_k^2 + q \sum_{k=1}^{N} t_k - \sum_{k=1}^{N} t_k y_k \right) = 0 \\ \dfrac{\partial L}{\partial q} = \dfrac{\partial}{\partial q} \displaystyle\sum_{k=1}^{N} \left[p t_k + q - y_k \right]^2 = 2 \sum_{k=1}^{N} \left[p t_k + q - y_k \right] = 2 \left(p \sum_{k=1}^{N} t_k + q \sum_{k=1}^{N} 1 - \sum_{k=1}^{N} y_k \right) = 0 \end{cases}$$

Here $\sum_{k=1..N} 1 = N$, but we will keep writing this as sum of 1's, for pedagogical reasons. Cancelling the unnecessary factor of 2 and moving the parameter-independent term to the right-hand side, we obtain a linear system:

$$\begin{cases} p \displaystyle\sum_{k=1}^{N} t_k^2 + q \sum_{k=1}^{N} t_k = \sum_{k=1}^{N} t_k y_k \\ p \displaystyle\sum_{k=1}^{N} t_k + q \sum_{k=1}^{N} 1 = \sum_{k=1}^{N} y_k \end{cases}$$

This is a system of 2 equations for 2 unknowns. In matrix form, we obtain:

$$\begin{pmatrix} \displaystyle\sum_{k=1}^{N} t_k^2 & \displaystyle\sum_{k=1}^{N} t_k \\ \displaystyle\sum_{k=1}^{N} t_k & \displaystyle\sum_{k=1}^{N} 1 \end{pmatrix} \begin{pmatrix} p \\ q \end{pmatrix} = \begin{pmatrix} \displaystyle\sum_{k=1}^{N} t_k y_k \\ \displaystyle\sum_{k=1}^{N} y_k \end{pmatrix}$$

Further below, we will also obtain this result without any calculus, using only vector algebra (see textbook section 4.1)

Example 1: Find parameters *p* and *q* for a linear model fit to *N*=3 data points **(0,0), (1,1), (2,1)**:

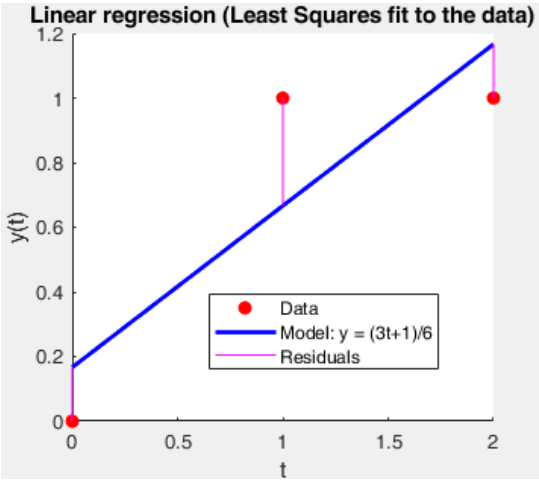
$$\begin{cases} \sum_{k=1}^{N} t_k^2 = 0+1+2^2 = 5 \\ \sum_{k=1}^{N} t_k = 0+1+2 = 3 \\ \sum_{k=1}^{N} 1 = N = 3 \\ \sum_{k=1}^{N} t_k y_k = 0\cdot 0+1\cdot 1+2\cdot 1 = 3 \\ \sum_{k=1}^{N} y_k = 0+1+1 = 2 \end{cases} \Rightarrow \begin{pmatrix} \sum_{k=1}^{N} t_k^2 & \sum_{k=1}^{N} t_k \\ \sum_{k=1}^{N} t_k & \sum_{k=1}^{N} 1 \end{pmatrix} \begin{pmatrix} p \\ q \end{pmatrix} = \begin{pmatrix} \sum_{k=1}^{N} t_k y_k \\ \sum_{k=1}^{N} y_k \end{pmatrix} \Rightarrow \begin{pmatrix} 5 & 3 \\ 3 & 3 \end{pmatrix} \begin{pmatrix} p \\ q \end{pmatrix} = \begin{pmatrix} 3 \\ 2 \end{pmatrix}$$

We need only 2 row operations to solve this:

Start with $R_1 - R_2 \to R_1$: $\begin{pmatrix} 2 & 0 \\ 3 & 3 \end{pmatrix} \begin{pmatrix} p \\ q \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$. Thus, we know that $\boldsymbol{p = \frac{1}{2}}$

Now, perform $2R_2 - 3R_1 \to R_2$: $\begin{pmatrix} 2 & 0 \\ 0 & 6 \end{pmatrix} \begin{pmatrix} p \\ q \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$. Therefore, the slope is $\boldsymbol{q = \frac{1}{6}}$

We conclude that the model underlying our data is $\boldsymbol{y_{model}(t) = \frac{t}{2} + \frac{1}{6} = \frac{3t+1}{6}}$



This fit definitely looks reasonable. We can now calculate the corresponding residuals, if we like (note: the sign convention is not important, since we are interested in the squared residuals):

$$r_1 = \frac{1}{6} - 0 = \frac{1}{6}; \quad r_2 = \frac{3+1}{6} - 1 = -\frac{1}{3}; \quad r_3 = \frac{6+1}{6} - 1 = \frac{1}{6}.$$

Note: the fact that the sum of residuals is zero is not a coincidence: this is in general true for models with a constant additive term (here, *q*). This will be clear when we learn the geometric visualization of the least-squares method.

Measures of Fit Quality

RSS (residual sum of squares) or **SE (squared error):** the sum of squared residuals equals $\frac{1}{36} + \frac{1}{9} + \frac{1}{36} = \frac{1}{6}$. As we have proven above, you can never make it smaller by choosing some other line.

Root mean squared error (RMSE) is directly related to above, and is self-explanatory: $\mathbf{RMSE} = \sqrt{\frac{\sum r_k^2}{N}} = \sqrt{\frac{1}{18}} = \frac{1}{3\sqrt{2}}$

Geometric Approach

Remarkably, we can obtain the same result without using any calculus and partial differentiation. Instead, we can view the least squares approach geometrically (this is done in the textbook section 4.1, for instance).

Let's examine the same **example** as on the previous page: find parameters *p* and *q* for a linear model fit to *N*=3 data points **(0,0), (1,1), (2,1)**.

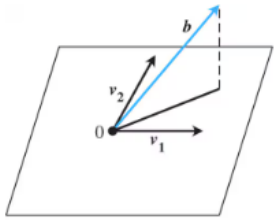
Let's write down the system of equations defining our residuals (one equation per data point):

$$y_{model}(t_k) = p \cdot t_k + q = y_{data}(t_k) + r_k : \quad \begin{cases} p \cdot 0 + q = 0 + r_1 \\ p \cdot 1 + q = 1 + r_2 \\ p \cdot 2 + q = 1 + r_3 \end{cases}$$

In vector form this becomes

$$\underbrace{p\begin{pmatrix}0\\1\\2\end{pmatrix} + q\begin{pmatrix}1\\1\\1\end{pmatrix}}_{\text{Model}} = \underbrace{\begin{pmatrix}0\\1\\1\end{pmatrix}}_{\text{Data, y}(t_k)} + \begin{pmatrix}r_1\\r_2\\r_3\end{pmatrix}$$

The left-hand side of this equation is a linear combination of two vectors, $v_1 = \begin{pmatrix}t_1\\t_2\\t_3\end{pmatrix} = \begin{pmatrix}0\\1\\2\end{pmatrix}$ and $v_2 = \begin{pmatrix}1\\1\\1\end{pmatrix}$, which defines a plane spanned by these two vectors. Now, the reason the **residuals are not zero** is that the vector of data values $b \equiv y_{data} = \begin{pmatrix}y_1\\y_2\\y_3\end{pmatrix} = \begin{pmatrix}0\\1\\1\end{pmatrix}$ does **not** lie in this plane (see vector in blue): the model does **not** match the data, for **any** *p* and *q*:



Sum of squared residuals is the square of the Euclidean distance from vector $b \equiv y_{data}$ to the plane formed by vectors v_1 and v_2. We know that this distance is minimized when the residual vector (dashed line in the figure) is orthogonal to both v_1 and v_2. Now, vectors are orthogonal when their inner product (dot product) is zero. In linear algebra notation, dot (inner) product of vectors ***v*** and ***u*** is $\boldsymbol{u^T v}$. Therefore, we have:

$$\begin{cases} v_1 \perp r \\ v_2 \perp r \end{cases} \Rightarrow \begin{cases} v_1^T r = 0 \\ v_2^T r = 0 \end{cases} \quad \text{where} \quad r = \underbrace{pv_1 + qv_2}_{\text{Model}} - \underbrace{y}_{\text{Data, y}(t_k)}$$

This system is simplified by introducing matrix **A** whose **columns** are **vectors** $\boldsymbol{v_1}$ and $\boldsymbol{v_2}$: $A = \begin{pmatrix}| & |\\ v_1 & v_2\\ | & |\end{pmatrix} = \begin{pmatrix}t_1 & 1\\ t_2 & 1\\ t_3 & 1\end{pmatrix} = \begin{pmatrix}0 & 1\\ 1 & 1\\ 2 & 1\end{pmatrix}$

Then the above system becomes: $\begin{cases} v_1^T r = 0 \\ v_2^T r = 0 \end{cases} \Rightarrow \boxed{A^T r = 0}$ where $r = \underbrace{(pv_1 + qv_2)}_{\text{Model}} - y = A\underbrace{\begin{pmatrix}p\\q\end{pmatrix}}_{x} - y$

Note about last step: a linear combination of two vectors with multipliers ***p*** and ***q*** is a product of a matrix containing these two vectors as its columns, times the vector of factors ***p*** and ***q***.

Finally, plugging $\boldsymbol{r = Ax - y}$ (where $\boldsymbol{x}$ is the vector of unknown model parameters $\boldsymbol{p}$ and $\boldsymbol{q}$) into $\boldsymbol{A^T r = 0}$ yields:

$$\boldsymbol{A^T(Ax - y) = 0}$$

or, moving the two terms to the opposite sides of the equation:

$$\boldsymbol{A^T Ax = A^T y.}$$

This equation is called the **normal equations**.

Now, let's show that we obtained the same linear algebra system as with analytic (calculus) approach:

$$A^T A x = \begin{pmatrix} t_1 & t_2 & t_3 \\ 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} t_1 & 1 \\ t_2 & 1 \\ t_3 & 1 \end{pmatrix} \begin{pmatrix} p \\ q \end{pmatrix} = \begin{pmatrix} \sum_{k=1}^{N} t_k^2 & \sum_{k=1}^{N} t_k \\ \sum_{k=1}^{N} t_k & \sum_{k=1}^{N} 1 \end{pmatrix} \begin{pmatrix} p \\ q \end{pmatrix} =$$

$$A^T y = \begin{pmatrix} t_1 & t_2 & t_3 \\ 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} = \begin{pmatrix} \sum_{k=1}^{N} t_k y_k \\ \sum_{k=1}^{N} y_k \end{pmatrix}$$

We obtain the same system as with our calculus approach! ☺

Example 2: model non-liner in state variables, but linear in parameters

As it turns out, very little changes in this method when fitting an arbitrarily complicated non-linear function of time to the data, instead of a linear curve: for our approach above, it only matters that the model is **linear in parameters**.

For example, suppose we are performing the following complicated fit to the same model data set **(0,0), (1,1), (2,1)**:

$$y_{model}(t_k) = p \cdot e^{t_k} + q \cdot \cos(t_k) = y_{data}(t_k) + r_k : \quad \begin{cases} p \cdot 1 + q \cdot 1 = 0 + r_1 \\ p \cdot e + q \cdot \cos(1) = 1 + r_2 \\ p \cdot e^2 + q \cdot \cos(2) = 1 + r_3 \end{cases}$$

This linear system is very similar to the linear regression case, but with different vectors: $v_1 = \begin{pmatrix} e^{t_1} \\ e^{t_2} \\ e^{t_3} \end{pmatrix}$ and $v_2 = \begin{pmatrix} \cos(t_1) \\ \cos(t_2) \\ \cos(t_3) \end{pmatrix}$.

If we use the geometric approach, our matrix is $A = \begin{pmatrix} | & | \\ v_1 & v_2 \\ | & | \end{pmatrix} = \begin{pmatrix} e^{t_1} & \cos(t_1) \\ e^{t_2} & \cos(t_2) \\ e^{t_3} & \cos(t_3) \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ e & \cos(1) \\ e^2 & \cos(2) \end{pmatrix}$

And our "normal equations", $\boldsymbol{A^T Ax = A^T y}$, now assume the form

$$A^T A x = \begin{pmatrix} e^{t_1} & e^{t_2} & e^{t_3} \\ \cos(t_1) & \cos(t_2) & \cos(t_3) \end{pmatrix} \begin{pmatrix} e^{t_1} & \cos(t_1) \\ e^{t_2} & \cos(t_2) \\ e^{t_3} & \cos(t_3) \end{pmatrix} \begin{pmatrix} p \\ q \end{pmatrix} = \begin{pmatrix} \sum_{k=1}^{N} e^{2t_k} & \sum_{k=1}^{N} e^{t_k} \cos(t_k) \\ \sum_{k=1}^{N} e^{t_k} \cos(t_k) & \sum_{k=1}^{N} \cos^2(t_k) \end{pmatrix} \begin{pmatrix} p \\ q \end{pmatrix} =$$

$$A^T y = \begin{pmatrix} e^{t_1} & e^{t_2} & e^{t_3} \\ \cos(t_1) & \cos(t_2) & \cos(t_3) \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} = \begin{pmatrix} \sum_{k=1}^{N} e^{t_k} y_k \\ \sum_{k=1}^{N} \cos(t_k) y_k \end{pmatrix}$$

Alternatively, you can follow the partial differentiation approach on page 1, to obtain the same system. I will leave the solution of this system for $\boldsymbol{x}$ to determine model parameters up to you (this will be the task in your lab assignment)

Generalization: models with arbitrary components (basis functions), linear in parameters

At this point, we can generalize to the case of an arbitrary model which is linear in parameters. To keep things simple, we will consider the case of 2 parameters only, but this is easily extended to any number of parameters.

General model which is linear in two parameters can be written in terms of the so-called basis functions which describe the individual components of the model:

$$y_{model}(t_k) = p_1 \cdot \phi_1(t_k) + p_2 \cdot \phi_2(t_k)$$

For example, the **linear regression** case corresponds to $\boldsymbol{\phi_1(t) = t, \phi_2(t) = 1}$, while the 2nd example on the previous page corresponds to $\boldsymbol{\phi_1(t) = e^t, \phi_2(t) = cos(t)}$.

Our model, given by the equation above, is a linear combination of vectors $v_1 = \begin{pmatrix} \phi_1(t_1) \\ \phi_1(t_2) \\ \phi_1(t_3) \end{pmatrix}$ and $v_2 = \begin{pmatrix} \phi_2(t_1) \\ \phi_2(t_2) \\ \phi_2(t_3) \end{pmatrix}$ (note: the number of rows equals the number of measurements, and could be greater than 3 of course).

Our normal equations, $\boldsymbol{A^T A x = A^T y}$, now assume the form

$$A^T A x = \begin{pmatrix} \phi_1(t_1) & \phi_1(t_2) & \phi_1(t_3) \\ \phi_2(t_1) & \phi_2(t_2) & \phi_2(t_3) \end{pmatrix} \begin{pmatrix} \phi_1(t_1) & \phi_2(t_1) \\ \phi_1(t_2) & \phi_2(t_2) \\ \phi_1(t_3) & \phi_2(t_3) \end{pmatrix} \begin{pmatrix} p_1 \\ \\ p_2 \end{pmatrix} = \begin{pmatrix} \sum_{k=1}^{N} \phi_1(t_k)^2 & \sum_{k=1}^{N} \phi_1(t_k)\phi_2(t_k) \\ \sum_{k=1}^{N} \phi_1(t_k)\phi_2(t_k) & \sum_{k=1}^{N} \phi_2(t_k)^2 \end{pmatrix} \begin{pmatrix} p_1 \\ \\ p_2 \end{pmatrix} =$$

$$A^T y = \begin{pmatrix} \phi_1(t_1) & \phi_1(t_2) & \phi_1(t_3) \\ \phi_2(t_1) & \phi_2(t_2) & \phi_2(t_3) \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} = \begin{pmatrix} \sum_{k=1}^{N} \phi_1(t_k) y_k \\ \sum_{k=1}^{N} \phi_2(t_k) y_k \end{pmatrix}$$

Generalization to ***M*** basis functions (*M* parameters) and ***N*** data points (measurements) is straightforward, but it is important that $\boldsymbol{N \geq M}$, and that the columns of ***A*** are linearly independent (i.e. $rank(A) = M$); otherwise, this approach (or any other approach) fails to find a unique solution, and at least some (or all) of the parameters are said to be **non-identifiable** from the data.

Note that for ***M*** basis functions (and *M* parameters), the system matrix $\boldsymbol{A^T A}$ is an *M*×*M* matrix.

As our **final example**, consider fitting a **quadratic polynomial** instead of line to our data:

$$\boldsymbol{y_{model}(t; p_k) = p_1 t^2 + p_2 t + p_3.}$$

Here basis functions are $\boldsymbol{\phi_1(t) = t^2, \phi_2(t) = t, \phi_3(t) = 1,}$ and our system matrix is $A = \begin{pmatrix} | & | & | \\ v_1 & v_2 & v_3 \\ | & | & | \end{pmatrix} = \begin{pmatrix} t_1^2 & t_1 & 1 \\ t_2^2 & t_2 & 1 \\ t_3^2 & t_2 & 1 \end{pmatrix}$.

Therefore, **normal equations** $\boldsymbol{A^T A x = A^T y}$ become $\begin{pmatrix} \sum_{k=1}^{N} t_k^4 & \sum_{k=1}^{N} t_k^3 & \sum_{k=1}^{N} t_k^2 \\ \sum_{k=1}^{N} t_k^3 & \sum_{k=1}^{N} t_k^2 & \sum_{k=1}^{N} t_k \\ \sum_{k=1}^{N} t_k^2 & \sum_{k=1}^{N} t_k & \sum_{k=1}^{N} 1 \end{pmatrix} \begin{pmatrix} p_1 \\ p_2 \\ p_3 \end{pmatrix} = \begin{pmatrix} \sum_{k=1}^{N} t_k^2 y_k \\ \sum_{k=1}^{N} t_k y_k \\ \sum_{k=1}^{N} y_k \end{pmatrix}$

Note: numerically, fitting monomials is a **bad idea**. Choose instead **Legendre orthogonal polynomials** as basis functions for a better-conditioned matrix.

Models non-linear in parameters

If the model being fitted is **not** linear in parameters, derivatives of the loss function will not be linear in parameters either, so equating the gradient of the loss function with respect to parameters to zero does not give a linear system. The method does not fail however: in this case one would numerically descend in the direction opposite to this gradient (in the space of parameters), until one reaches the minimum of the loss function. In fact, much of machine learning (AI) involves gradient descent in very high-dimensional parameter space (with billions or trillions of parameters).

However, in some specific cases, a model non-linear in parameters can be linearized. Consider the following example:

Example: simple pharmacokinetics

Suppose we measure the amount of medication in the bloodstream, as a function of time after administration, and obtain the following data set: $\{(t_k, y_k)\} = \{(\mathbf{2}, \mathbf{20}), (\mathbf{4}, \mathbf{12}), (\mathbf{8}, \mathbf{7}), (\mathbf{12}, \mathbf{4}), (\mathbf{16}, \mathbf{3})\}$

We seek to estimate the total absorbed amount $\boldsymbol{y_0}$ and characteristic elimination time $\boldsymbol{\tau}$ (related to half-life via $\tau = T_{1/2}/\ln 2$). To do that, we have to fit the data to the following model:

$$\boldsymbol{y_{model}(t_k; y_0, \tau) = y_0 \exp\left(-\frac{t_k}{\tau}\right)}$$

If you differentiate the loss of function with respect to $\boldsymbol{\tau}$, the resulting equation is nonlinear in τ. However, we can linearize this model using a logarithm:

$$\boldsymbol{z_{model} \equiv ln(y_{model}) = ln(y_0) - \frac{t_k}{\tau}} \;\equiv\; q + p\, t_k$$

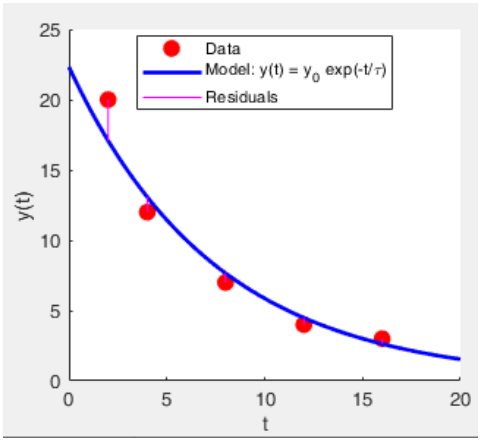
Thus, we obtained a line with slope $\boldsymbol{p = -\frac{1}{\tau}}$ and constant offset $\boldsymbol{q = \ln(y_0)}$**,** which we determine using our familiar linear system:

$$\begin{pmatrix} \sum_{k=1}^{N} t_k^2 & \sum_{k=1}^{N} t_k \\ \sum_{k=1}^{N} t_k & \sum_{k=1}^{N} 1 \end{pmatrix} \begin{pmatrix} p \\ q \end{pmatrix} = \begin{pmatrix} \sum_{k=1}^{N} t_k z_k \\ \sum_{k=1}^{N} z_k \end{pmatrix} = \begin{pmatrix} \sum_{k=1}^{N} t_k \ln(y_k) \\ \sum_{k=1}^{N} \ln(y_k) \end{pmatrix}$$

After we find the best-fit parameters p and q, we can find the original model parameters y_0 and τ:

$$\boldsymbol{\tau = -\frac{1}{p}, \quad y_0 = \exp(q)}$$

For the given data set, we obtain: $\begin{pmatrix} 484 & 42 \\ 42 & 5 \end{pmatrix} \begin{pmatrix} p \\ q \end{pmatrix} = \begin{pmatrix} 65.712 \\ 9.911 \end{pmatrix} \Rightarrow \begin{cases} p = -0.1337 \\ q = 3.105 \end{cases} \Rightarrow \begin{cases} \tau = -\frac{1}{p} \approx 7.48 \text{ hrs} \\ y_0 = e^q \approx 22.3 \end{cases}$



NOTE: we minimized the sum of squared residuals in $\log(y)$, not in the original measurements y. However, the best-fit parameter values we obtained can then be used as initial guess for minimizing the original loss function, numerically.