Maximum Likelihood

Nuno Carvalho

November 21, 2025

A lot of this text is adapted from the textbook [Lynch & Walsh, 1998], particularly Appendix 4, as well as some back-and-forth with ChatGPT to help consolidate ideas. The point of this is to re-explain (restricted) maximum likelihood in a way that makes more sense to me, and hopefully to you. Here, I focus more on the theory and abstraction of ML/REML, and in other entries, I plan to apply it to a particular class of problem (e.g. variance component estimation).

1 Maximum Likelihood (ML)

1.1 Likelihood

When we construct a model of some real world phenomenon, we define some process that describes how data, $\mathbf{y}$, is generated according to some set of parameters, $\boldsymbol{\Theta}$. Typically, our model is not deterministic since we do not understand every single cause of the real world phenomenon with perfect accuracy. Instead, our data is generated probabilistically, which means given some set of parameter values, different possible data values can be generated with differing probabilities. We can also think of the reverse: Given our observed data, there are many different combinations of parameter values that could have generated that data, each with differing likelihoods.

We can define a **likelihood function** that tells us the probability density of some data $\mathbf{y}$ being generated by some set of parameter values $\boldsymbol{\Theta}$:

$$\ell(\boldsymbol{\Theta}|\mathbf{y})$$

Note that because we actually do know what the data is (it is observed), we are asking the question: "How well does an input set of parameter values, $\boldsymbol{\Theta}$, explain the observed data, $\mathbf{y}$. This does not necessarily tell us the **probability** of the parameter values given our data, since we are not computing the likelihood across all possible parameters (weighted by some prior), as we would in Bayesian statistics.

Because our data is (usually) composed of multiple data points, and our model often describes the probability of a single data point being generated, our likelihood for the

entire data will often be the product of multiple individual likelihoods:

$$\ell(\mathbf{\Theta}|\mathbf{y}) = \prod_{i}^{N} \ell(\mathbf{\Theta}|y_i)$$

Note that this makes the crucial assumption that each data point is independent of each other, such that the likelihood of one data point is not affected by the values of the other datapoints. Because sums are typically easier to work with than products, it's useful to instead compute the natural logarithm of the likelihood, which per the properties of logarithms, turns those products into sums. This defines the **log-likelihood**:

$$\mathrm{L}(\mathbf{\Theta}|\mathbf{y}) = \sum_{i}^{N} \ln(\ell(\mathbf{\Theta}|y_i))$$

Because logarithms are a monotonic function, it means that whatever parameter values maximize the likelihood also maximize the log-likelihood, and same for the minimum.

1.1.1 Example of the likelihood of a function

(Adapted from Example 1 of Appendix 4 of [Lynch & Walsh, 1998]). Let's say we have N data points: $\mathbf{y} = [y_1, y_2, ..., y_N]$, which we believe were generated from some normal distribution with a variance of $\sigma^2 = 1$ but an unknown mean. We know the probability density function (PDF) for a normal distribution, which is a function of the mean μ and variance σ.

$$\Pr(y_i, \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} \exp[-\frac{1}{2}(\frac{y_i - \mu}{\sigma})^2]$$

In our case, we know $\sigma = 1$ and we also observe the data $\mathbf{y}$. What we don't know is the mean, μ, which is the only non-fixed parameter. We can plug in our known parameter σ, compute the product of the PDF for all our data points, and take the logarithm. This results in a log-likelihood function that describes how well an inputted μ value explains our observed data:

$$\mathrm{L}(\mu|\mathbf{y}) = -\frac{n}{2}\ln(2\pi) - \frac{1}{2}\sum_{i}^{N}(y_i - \mu)^2$$

1.2 Maximum Likelihood Estimation

Once we have described the likelihood of some set of parameter values given our observed data, we can then ask the question: What set of parameter values maximizes the likelihood of our data? The rationale is that if we want to estimate these unknown parameters, our best bet for a single point estimate is the combination of parameter values that are most likely to produce the observed data. The set of parameter values $\hat{\mathbf{\Theta}}$ that maximizes the likelihood of our observed data is called the **maximum likelihood estimate (MLE)**.

From calculus, we know that the maximum of a multivariable function can be identified by taking the partial derivatives of the function with respect to each parameter and then

solving for the parameter values that make them equal to zero. We define this first partial derivative of a log-likelihood function to be the **Score** (**S**) of the likelihood function:

$$\mathbf{S}(\boldsymbol{\Theta}) = \frac{\partial \mathrm{L}(\boldsymbol{\Theta})}{\partial \boldsymbol{\Theta}}$$

Note that if $\boldsymbol{\Theta}$ is a vector of M parameters, then $\mathbf{S}(\boldsymbol{\Theta})$ is also a vector of length M. The MLE, $\hat{\boldsymbol{\Theta}}$, is therefore the solution to the following:

$$\mathbf{S}(\hat{\boldsymbol{\Theta}}) = 0$$

1.2.1 Example of the MLE of a function

Continuing the previous example, we can compute the score of the likelihood function, which in this case is just a single derivative:

$$\begin{aligned}
\mathbf{S}(\mu) &= \frac{\partial \mathrm{L}(\mu)}{\partial \mu} \\
&= \frac{\partial}{\partial \mu}[-\frac{n}{2}\ln(2\pi) - \frac{1}{2}\sum_{i}^{N}(y_i - \mu)^2] \\
&= \frac{\partial}{\partial \mu}[-\frac{n}{2}\ln(2\pi) - \frac{1}{2}\sum_{i}^{N}(y_i^2 - 2y_i\mu + \mu^2)] \\
&= -\frac{1}{2}\sum_{i}^{N}(-2y_i + 2\mu) \\
&= N(\bar{y} - \mu)
\end{aligned}$$

where $\bar{y} = \frac{1}{N}\sum_{i}^{N}(y_i)$ (i.e. the sample mean). We now set $\mathbf{S}(\hat{\mu}) = 0$ and solve:

$$\begin{aligned}
0 &= \mathbf{S}(\hat{\mu}) \\
&= N(\bar{y} - \hat{\mu})
\end{aligned}$$

The solution is: $\hat{\mu} = \bar{y} = \frac{1}{N}\sum_{i}^{N}(y_i)$. Matching our intuitions, the mean parameter that maximizes the likelihood of our data is the arithmetic mean of our data sample.

1.2.2 Asymptotic efficiency of the MLE

What's nice about ML estimation is that as the sample size increases, the MLE converges to the true parameter value θ_0 (if the likelihood function matches that of the true data-generating function). This is called **asymptotic efficiency** and can be written formally as:

$$\hat{\theta}_N \rightarrow^p \theta_0$$

This should make intuitive sense, as the true data-generating function has some true distribution of observed data that it produces, which in most cases, is unique to the specific parameter value of that data-generating function. Consequently, the parameter that maximizes the likelihood of producing the true parameter's distribution of data is the true parameter itself. The more samples we collect, the more our observed data's distribution should match the true distribution of the possible data. Therefore, the MLE should get closer to the true parameter value.

1.3 The Curvature of the Likelihood

While the first derivative describes the rate of change of the likelihood function, the second derivative describes its curvature. At the MLE, a strong curvature (high magnitude) means the likelihood peaks very sharply at the MLE. Therefore, a small change in the paramater value(s) from the MLE leads to a sharp drop in the likelihood. In contrast, if there is weak curvature (low magnitude) at the MLE, then there is a wide region of parameter estimates that yield similar (albeit slightly lower) likelihoods of producing the data.

1.3.1 The Hessian

For M parameters, the second derivative of the log-likelihood function is the **Hessian** ($\mathbf{H}$), an $M \times M$ symmetric matrix where:

$$H_{ij} = \frac{\partial^2 \mathrm{L}(\Theta|\mathbf{y})}{\partial \Theta_i \partial \Theta_j}$$

This is also called the gradient and can be evaluated at any set of parameter values, describing the multi-dimensional curvature of the log-likelihood at that set of parameters.

1.3.2 The Fisher information Matrix

Of mathematical interest to us is the negative expected value of the Hessian, which is called the **Fisher information matrix**:

$$\mathcal{I}(\boldsymbol{\Theta}) = -\mathrm{E}[\mathbf{H}(\boldsymbol{\Theta})]$$

Note that the expected value here is taken by averaging across all the data that is potentially generated by the inputted set of parameters, weighted by the probability of generating that data. This means that while the Hessian is a function that depends on both the parameters and the data that is observed, the Fisher information matrix only depends on the parameters. That is why $-\mathbf{H}(\boldsymbol{\Theta})$ is often called the **observed information matrix**, while $\mathcal{I}(\boldsymbol{\Theta})$ is the **expected information matrix**.

As sample size increases, the Fisher information matrix evaluated at the MLE converges to the negative of the Hessian evaluated at the MLE, since the distribution of the observed data increasingly resembles the true distribution of the possible data that is generated:

$$\mathcal{I}(\hat{\boldsymbol{\Theta}}) \approx -\mathbf{H}(\hat{\boldsymbol{\Theta}})$$

1.3.3 Asymptotic normality of the MLE

If our likelihood function follows some regularity conditions, as sample size increases, the sampling distribution of the MLE (that is, the distribution of the MLE for different samples of observed data from the same underlying data-generating process) converges to normality. This is called **asymptotic normality** and can be written formally as:

$$\sqrt{N}(\hat{\theta}_N - \theta_0) \rightarrow^d \mathcal{N}(0, \sigma^2)$$

It can be proven that the variance (σ^2 in the above equation) of the MLE converges to the inverse of the Fisher information at the true parameter value:

$$\hat{\theta}_N \rightarrow^d \mathcal{N}(\theta_0, \mathcal{I}(\theta_0)^{-1})$$

This also applies when there are multiple parameters, such that the inverse of the Fisher information matrix evaluated at the MLE is the covariance matrix of the MLE. The i$^{\text{th}}$ diagonal element is the variance of the MLE for the i$^{\text{th}}$ parameter. The element (i,j) is the covariance of the MLE for the i$^{\text{th}}$ and j$^{\text{th}}$ parameters.

2 Restricted Maximum Likelihood (REML)

Maximum likelihood estimation is often applied in the context of a linear mixed model:

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \mathbf{u} + \boldsymbol{\epsilon}$$

where $\mathbf{X}\boldsymbol{\beta}$ encodes fixed effects on $\mathbf{y}$, while $\mathbf{u}$ is a random effect drawn from a multivariate normal distribution, $\mathbf{u} \sim MVN(0, \sigma_A^2\mathbf{A})$. Furthermore, $\boldsymbol{\epsilon}$ can also be thought of as a random effect representing the random residual that is distributed from a single normal distribution, $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \sigma_\epsilon^2\mathbf{I})$. The problem with ML is that when estimating the parameter σ_A^2, it must assume that its estimate of the fixed effect is exactly correct, $\hat{\boldsymbol{\beta}} = \boldsymbol{\beta}$. This often leads to bias in the estimate of the other parameters (see [Lynch & Walsh, 1998, pg 781] for an example). Restricted Maximum Likelihood (REML) resolves this by transforming the equation definining y such that it does not depend on the fixed effects. The transformation can be thought of as multiplying both sides of the equation by a transformation matrix $\mathbf{K}$ that when multiplied by $\mathbf{X}$ equals $\mathbf{0}$, $\mathbf{KX} = \mathbf{0}$:

$$\mathbf{Ky} = \mathbf{Ku} + \mathbf{K}\boldsymbol{\epsilon}$$

The way this changes downstream ML equations depends on the application. For the application of REML in variance component decomposition, as shown here, see the article on Heritability.

3 Methods of solving for the MLE

In practice, solving for the set of parameters that makes the score of the likelihood function equal to zero is very difficult. Instead, there are algorithms that start with a guess for the parameter estimates, and then iteratively move closer and closer to the parameter values that maximize the likelihood of the observed data. Eventually, the algorithm gets close enough to the MLE that the parameter estimate at each iteration no longer changes by much, and thus we decide on a cutoff point where we declare that the algorithm has converged on the MLE.

Broadly, the strategy of these methods is to randomly choose (or have the user specify) a location in the parameter landscape, where the height of the landscape at that set of parameter values is the log-likelihood of those values generating the observed data. The

algorithm can't look very far from its position with confidence, but it can look at how the landscape height (the log-likelihood function) changes at its location. This can be simply the rate of change (first derivative) or also the curvature (second derivative). It then uses this information to predict the direction and distance of where the the landscape height should reach its maximum (the MLE) and goes to it. In the first few iterations, it's likely that this new predicted MLE location turns out to not explain the data very well, since the landscape is not flat at that location. The algorithm repeats this jumping procedure until it is no longer moving very much between its prediction for how the landscape changes around the true peak has become pretty accurate.

Note, however, if there are local maxima (peaks in the likelihood landscape that are not the tallest in the entire landscape), the algorithm may fail to converge on the true parameter values, instead converging on said local maxima. By re-running the algorithm under multiple starting spots, one can usually avoid this.

3.1 Quadratic iterative methods

From calculus, we know that a function can be approximated around some point through a Taylor series expansion:

$$f(x) \approx f(x_0) + f'(x_0)(x - x_0) + \frac{1}{2}f''(x_0)(x - x_0)^2 + ... + \frac{1}{n!}f^{(n)}(x_0)(x - x_0)^n + ...$$

Through some algebra, if we set $f(x) = 0$ and solve for x using the first-order approxmation, we get:

$$f(x) = 0 \approx f(x_0) + f'(x_0)(x - x_0)$$
$$f'(x_0)(x - x_0) \approx -f(x_0)$$
$$(x - x_0) \approx -\frac{f(x_0)}{f'(x_0)}$$
$$x \approx x_0 - \frac{f(x_0)}{f'(x_0)}$$

We can replace the above function with the score, which is the first derivative of the log-likelihood: $f(x) = \mathbf{S}(\boldsymbol{\Theta}) = \frac{\partial L(\boldsymbol{\Theta})}{\partial \boldsymbol{\Theta}}$. And we can replace its derivative with the second derivative of the log-likelihood: $f'(x) = \frac{\partial \mathbf{S}(\boldsymbol{\Theta})}{\partial \boldsymbol{\Theta}} = \frac{\partial L^2(\boldsymbol{\Theta})}{\partial^2 \boldsymbol{\Theta}}$. Furthermore, the x_0 in this equation becomes an arbitrary set of parameter values we are starting from, $\hat{\boldsymbol{\Theta}}^{(0)}$, while the x represents our approximation of the MLE from that starting point, since $f(x) = 0$ and the MLE is what makes $\mathbf{S}(\boldsymbol{\Theta}) = 0$. We can then use that resulting MLE, $\hat{\boldsymbol{\Theta}}^{(1)}$, as our starting point to again approximate where the MLE actually is, $\hat{\boldsymbol{\Theta}}^{(2)}$, repeating the process that can be generalized as:

$$\hat{\boldsymbol{\Theta}}^{(k+1)} = \hat{\boldsymbol{\Theta}}^{(k)} - (\frac{\partial L^2(\boldsymbol{\Theta})}{\partial^2 \boldsymbol{\Theta}})\Big|_{\hat{\boldsymbol{\Theta}}^{(k)}})^{-1} \frac{\partial L(\boldsymbol{\Theta})}{\partial \boldsymbol{\Theta}}\Big|_{\hat{\boldsymbol{\Theta}}^{(k)}}$$

We can rewrite this using the derivative functions of the likelihood we've already defined: the Score $\mathbf{S}(\boldsymbol{\Theta})$ and the Hessian $\mathbf{H}(\boldsymbol{\Theta})$:

$$\hat{\boldsymbol{\Theta}}^{(k+1)} = \hat{\boldsymbol{\Theta}}^{(k)} - \mathbf{H}^{-1}(\hat{\boldsymbol{\Theta}}^{(k)})\mathbf{S}(\hat{\boldsymbol{\Theta}}^{(k)})$$

Connecting back to the earlier analogy, using the first and second derivatives of the likelihood landscape at the algorithm's current spot, it predicts where the landscape peak should be. This becomes the algorithm's MLE for its next iteration. However, once at this new estimate, if there is still curvature in the landscape, then we must not yet be at the true MLE of the observed. The algorithm iteratively does this, and in theory, after infinite iterations, the algorithm lands on the true MLE, where $\mathbf{S}(\boldsymbol{\Theta})^{(K)} = 0$ and therefore our iterative formula becomes:

$$\hat{\boldsymbol{\Theta}}^{(K+1)} = \hat{\boldsymbol{\Theta}}^{(K)}$$

The algorithm has converged. Of course, in practice, we don't have time to do an infinite number of iterations, so we decide on some small distance between subsequent estimates for which we declare that the algorithm has converged.

The following three quadratic methods discussed here differ in what they compute to use as the second derivative of the log-likelihood.

3.1.1 Newton-Raphson

The Newton-Raphson method uses the observed information matrix evaluated the current iteration's MLE as the second derivative of the log-likelihood. This is just the negative of the Hessian evaluated at the current MLE, so it looks identical to how we defined the iterative process earlier:

$$\hat{\boldsymbol{\Theta}}^{(k+1)} = \hat{\boldsymbol{\Theta}}^{(k)} - \mathbf{H}^{-1}(\hat{\boldsymbol{\Theta}}^{(k)})\mathbf{S}(\hat{\boldsymbol{\Theta}}^{(k)})$$

3.1.2 Fisher-Scoring

The Fisher-Scoring method uses the expected information matrix evaluated the current iteration's MLE as the second derivative of the log-likelihood. So, instead of the negative Hessian, it uses the Fisher information matrix. Noting the change in sign, the iterative formula looks like:

$$\begin{aligned}\hat{\boldsymbol{\Theta}}^{(k+1)} &= \hat{\boldsymbol{\Theta}}^{(k)} + \mathcal{I}^{-1}(\hat{\boldsymbol{\Theta}}^{(k)})\mathbf{S}(\hat{\boldsymbol{\Theta}}^{(k)}) \\ &= \hat{\boldsymbol{\Theta}}^{(k)} - \mathrm{E}[\mathbf{H}^{-1}(\hat{\boldsymbol{\Theta}}^{(k)})]\mathbf{S}(\hat{\boldsymbol{\Theta}}^{(k)})\end{aligned}$$

The Fisher-Scoring method offers some advantages compared to the Newton-Raphson method. First, it is often easier to compute the Fisher information matrix than the Hessian. The Hessian will usually be a function of both the unknown parameters we are trying to estimate (which are set to the current iteration's MLE when evaluating it) as well as unbiased estimates of said parameters that are computed directly from the observed data. Since these estimates from the data are unbiased, on expectation across different data samples (generated from the current MLE), those estimates will exactly equal their respective parameters. This often leads to terms cancelling out, simplifying the equation significantly. Second, the Fisher-Scoring method also appears to be better at converging correctly from different starting points compared to Newton-Raphson (according to [Jenrich and Sampson, 1986]).

3.1.3 Average Information

The Average Information method uses the average of the expected and observed information matrices evaluated at the current iteration's MLE as the second derivative of the log-likelihood. Defining this matrix as **AI**, it is:

$$\mathbf{AI}(\boldsymbol{\Theta}) = \frac{1}{2}(\mathcal{I}(\boldsymbol{\Theta}) - \mathbf{H}(\boldsymbol{\Theta}))$$

Such that the iterative equation is:

$$\begin{aligned}\hat{\boldsymbol{\Theta}}^{(k+1)} &= \hat{\boldsymbol{\Theta}}^{(k)} + \mathbf{AI}^{-1}(\hat{\boldsymbol{\Theta}}^{(k)})\mathbf{S}(\hat{\boldsymbol{\Theta}}^{(k)}) \\ &= \hat{\boldsymbol{\Theta}}^{(k)} + (\frac{\mathcal{I}(\hat{\boldsymbol{\Theta}}^{(k)}) - \mathbf{H}(\hat{\boldsymbol{\Theta}}^{(k)})}{2})^{-1}\mathbf{S}(\hat{\boldsymbol{\Theta}}^{(k)})\end{aligned}$$

In some scenarios, taking the average of the expected and observed information matrices leads to terms cancelling out, resulting in a simpler matrix to calculate. So, using the Average Information method can be faster and converge better than the Newton-Raphson and Fisher-Scoring methods.

4 Variance component estimation through REML

Here, I'll define a common application of REML: estimating the variances of random effect components. For example, in statistical genetics, a phenotype can be modeled as a combination of fixed effects (such as covariate effects, like sex and age) and random effects, like an additive genetic component and shared environmental component.

4.1 Model definition

We define a mixed model for an outcome $\mathbf{y}$ composed of fixed effects and M random effects as:

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \sum_{i=1}^{M+1} \mathbf{Z}_i\mathbf{u}_i$$

Where:

- $\mathbf{y}$: phenotypes. $N \times 1$ vector. Centered so that $\mathrm{E}[\mathbf{y}] = 0$.
  - N: number of samples.
- $\mathbf{X}$: covariate design matrix. $N \times C$ matrix.
  - C: number of covariates.
- $\boldsymbol{\beta}$: covariate effects. $C \times 1$ vector.
- M: number of random effect components. Does not include the random residual component. Indexed by i.
  - $i = M + 1$ refers to the random residual component, often denoted by ϵ
- $\mathbf{Z}_i$: incidence matrix for the i^{th} random effect component. $N \times N_i$ matrix. Maps each individual to their respective cluster in the i^{th} random effect component.
  - N_i: number of clusters in i^{th} random effect component.

  - $\mathbf{Z}_{M+1} = \mathbf{I}_N$
- $\mathbf{u}_i$: the i^{th} set of random effects. $N_i \times 1$ vector. Assumed to be drawn from a multivariate normal distribution: $\mathbf{u}_i \sim \mathcal{N}(0, \sigma_i^2 \mathbf{A}_i)$
  - σ_i^2: variance of the i^{th} random effect component. This is what we are often interested in estimating.
  - $\mathbf{A}_i$: relationship matrix for the i^{th} random effect component. $N_i \times N_i$ matrix. This encodes the similarity between clusters for some component of interest.
    - $\mathbf{A}_{M+1} = \mathbf{I}_N$

We can write $\mathbf{y}$ as a multivariate normal:

$$\mathbf{y} = \mathcal{N}(\mathbf{X}\boldsymbol{\beta}, \mathbf{V})$$

where:

$$V = \sum_{i=1}^{M+1} \sigma_i^2 \mathbf{Z}_i \mathbf{A}_i \mathbf{Z}_i^\intercal$$

4.2 Solving ML equations through quadratic methods

The ML equations that need to be solved to obtain parameter estimates are not solvable through analytical methods. Instead, iterative methods should be utilized, such as the quadratic method, where we wrote earlier:

$$\hat{\boldsymbol{\Theta}}^{(k+1)} = \hat{\boldsymbol{\Theta}}^{(k)} - \left(\frac{\partial \mathrm{L}^2(\boldsymbol{\Theta})}{\partial^2 \boldsymbol{\Theta}}\right)\Big|_{\hat{\boldsymbol{\Theta}}^{(k)}}^{-1} \frac{\partial \mathrm{L}(\boldsymbol{\Theta})}{\partial \boldsymbol{\Theta}}\Big|_{\hat{\boldsymbol{\Theta}}^{(k)}}$$

Here, $\boldsymbol{\Theta}$ denotes the variances of the random components, $\boldsymbol{\Theta} = [\sigma_1^2, \sigma_2^2, ..., \sigma_K^2, \sigma_\epsilon^2]^\intercal$. All we need is an equation for the first and second derivatives of the likelihood function. As the derivation is lengthy, I will simply write the equations that yield such derivatives. Refer to [Lynch & Walsh, 1998, pg 784] for derivations. Note that these derivatives are evaluated at the current set of parameter estimates. In an iterative algorithm, it starts with a random (or user-specified) set of parameter estimates, and iteratively improves on them until the estimates change very little. We'll first define two sets of matrices that are used for both first and second derivatives.

4.2.1 Variance component matrices

The first is the derivative of the overall variance $\mathbf{V}$ with respect to the variance of an individual component:

$$\begin{aligned}
\mathbf{V}_i &= \frac{\partial \mathbf{V}}{\partial \sigma_i^2} \\
&= \frac{\partial}{\partial \sigma_i^2} \sum_{i=1}^{M+1} \sigma_i^2 \mathbf{Z}_i \mathbf{A}_i \mathbf{Z}_i^\intercal \\
&= \mathbf{Z}_i \mathbf{A}_i \mathbf{Z}_i^\intercal
\end{aligned}$$

This also applies for the random residual component, $\mathbf{V}_\epsilon = \mathbf{I}$. Note that this only needs to be computed once, prior to any iterations. However, the full variance matrix $\mathbf{V}^{(k)}$ does need to be evaluated at every iteration, since it is a function of the current parameter estimates, $\boldsymbol{\Theta}^{(k)}$.

4.2.2 Projection matrix

The second matrix is crucial to REML:

$$\mathbf{P} = \mathbf{V}^{-1} - \mathbf{V}^{-1}\mathbf{X}(\mathbf{X}^\intercal\mathbf{V}^{-1}\mathbf{X})^{-1}\mathbf{X}^\intercal\mathbf{V}^{-1}$$

This projection matrix has the neat property that:

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

It projects the outcomes $\mathbf{y}$ onto a residual space that has subtracted out the fixed effects (i.e., it is orthogonal to $\mathbf{X}$), while also accounting for the correlation of those residuals that is implied by $\mathbf{V}$. By considering simpler scenarios, $\mathbf{P}$ simplifies to perhaps more intuitive or familiar forms. For example, if there are fixed effects and the random components are independent from each other, e.g. $\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$, then $\mathbf{V}^{-1} = \mathbf{I}$, and so $\mathbf{P} = \mathbf{I} - \mathbf{X}(\mathbf{X}^\intercal\mathbf{X})^{-1}\mathbf{X}^\intercal$. This is the residual maker matrix in ordinary least squares (OLS).

Note that this matrix needs to be computed at every iteration of parameter estimation, since $\mathbf{P}^{(k)}$ is a function of the current parameter estimates, $\boldsymbol{\Theta}^{(k)}$.

4.2.3 First derivative of the likelihood

With these two sets of matrices defined, we can now write the equation for the first derivative of the log-likelihood function with respect to a variance parameter, σ_i^2.

$$\frac{\partial \mathrm{L}}{\partial \sigma_i^2} = -\frac{1}{2}\mathrm{tr}[\mathbf{P}\mathbf{V}_i] + \frac{1}{2}\mathbf{y}^\intercal\mathbf{P}\mathbf{V}_i\mathbf{P}\mathbf{y}$$

Here, tr[$\mathbf{A}$] denotes the **trace** of the matrix $\mathbf{A}$, which is simply the sum of the diagonal elements of $\mathbf{A}$. This also needs to be computed at every iteration.

4.2.4 Second derivative of the likelihood

As discussed above, different algorithms use different matrices as their second derivative of the log-likelihood differently. Using the notation from above:

$$\mathbf{H}_{ij} = \frac{1}{2}\mathrm{tr}[\mathbf{P}\mathbf{V}_i\mathbf{P}\mathbf{V}_j] - \mathbf{y}^\intercal\mathbf{P}\mathbf{V}_i\mathbf{P}\mathbf{V}_j\mathbf{P}\mathbf{y}$$

$$\mathcal{I}_{ij} = \frac{1}{2}\mathrm{tr}[\mathbf{P}\mathbf{V}_i\mathbf{P}\mathbf{V}_j]$$

$$\mathbf{AI}_{ij} = \frac{1}{2}\mathbf{y}^\intercal\mathbf{P}\mathbf{V}_i\mathbf{P}\mathbf{V}_j\mathbf{P}\mathbf{y}$$

As you may have noticed, they are closely related to each other:

$$\mathbf{H}_{ij} = \mathcal{I}_{ij} - 2\mathbf{AI}_{ij}$$

This also needs to be computed at every iteration.