1 Nonlinear Least Squares

All the models we've seen so far are *linear* in the parameters we're trying to learn. That is, our prediction $\hat{y} = f(\mathbf{x}; \boldsymbol{\theta})$ is some linear function of the parameters $\boldsymbol{\theta}$. For example, in OLS, $\boldsymbol{\theta} = \mathbf{w}$ and the residuals r_i are computed by $y_i - \mathbf{w}^\top \mathbf{x}_i$, which is linear in the components of $\mathbf{w}$. In the case of least-squares polynomial regression, the predicted value is not a linear function of the input $\mathbf{x}$, but it is still a linear function of the augmented input $\phi(\mathbf{x}_i)$.

However, we may have models which are nonlinear functions of their parameters. Let's consider a motivating example. Suppose we want to estimate the 2D position $\boldsymbol{\theta} = (\theta_1, \theta_2)$ of some entity, for example a robot. The information we have to work with are noisy distance estimates $Y_i \in \mathbb{R}$ from n sensors whose positions $\mathbf{x}_i \in \mathbb{R}^2$ are fixed and known. If we assume i.i.d. Gaussian noise as usual, our statistical model has the form

$$Y_i = \|\mathbf{x}_i - \theta\|_2 + Z_i, \quad Z_i \overset{\text{iid}}{\sim} \mathcal{N}(0, \sigma^2), \quad i = 1, \ldots, n$$

where

$$\|\mathbf{x}_i - \theta\|_2 = \sqrt{(x_{i1} - \theta_1)^2 + (x_{i2} - \theta_2)^2}$$

Our prediction is

$$\hat{y} = f(\mathbf{x}; \boldsymbol{\theta}) = \|\mathbf{x} - \boldsymbol{\theta}\|_2$$

which is clearly not linear in $\boldsymbol{\theta}$.

1.1 MLE Formulation

More generally, let us assume a model where f is now some arbitrary differentiable function and $\boldsymbol{\theta} \in \mathbb{R}^d$:

$$Y_i = f(\mathbf{x}_i; \boldsymbol{\theta}) + Z_i, \quad Z_i \overset{\text{iid}}{\sim} \mathcal{N}(0, \sigma^2), \quad i = 1, \ldots, n$$

Note that this implies $Y_i \mid \mathbf{x}_i \sim \mathcal{N}(f(\mathbf{x}_i; \boldsymbol{\theta}), \sigma^2)$. The maximum likelihood estimator is given by

$$\begin{aligned}
\hat{\boldsymbol{\theta}}_{\text{MLE}} &= \arg\max_{\boldsymbol{\theta}} \ \ell(\boldsymbol{\theta}; \mathbf{X}, \mathbf{y}) \\
&= \arg\max_{\boldsymbol{\theta}} \ \sum_{i=1}^{n} \log p(y_i \mid \mathbf{x}_i, \boldsymbol{\theta}) \\
&= \arg\max_{\boldsymbol{\theta}} \ \sum_{i=1}^{n} \log \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y_i - f(\mathbf{x}_i; \boldsymbol{\theta}))^2}{2\sigma^2} \right)
\end{aligned}$$

$$= \arg\max_{\boldsymbol{\theta}} \sum_{i=1}^{n} \left[-\frac{1}{2}\log\left(2\pi\sigma^2\right) - \frac{1}{2\sigma^2}(y_i - f(\mathbf{x}_i;\boldsymbol{\theta}))^2 \right]$$
$$= \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{n} (y_i - f(\mathbf{x}_i;\boldsymbol{\theta}))^2$$

Observe that the objective function is a sum of squared residuals as we've seen before, even though the function f is nonlinear in general. For this reason this method is called **nonlinear least squares**.

Motivated by the MLE formulation above, we consider the following optimization problem:

$$\min_{\boldsymbol{\theta}} \epsilon(\boldsymbol{\theta}) = \min_{\boldsymbol{\theta}} \sum_{i=1}^{n} (y_i - f(\mathbf{x}_i;\boldsymbol{\theta}))^2$$

One way to solve this optimization problem is to find all of its critical points and choose the point that minimizes the objective. From **first-order optimality conditions**, the gradient of the objective function at any minimum must be zero:

$$\nabla_{\boldsymbol{\theta}} \epsilon = 2\sum_{i=1}^{n} (y_i - f(\mathbf{x}_i;\boldsymbol{\theta}))\nabla_{\boldsymbol{\theta}} f(\mathbf{x}_i;\boldsymbol{\theta}) = \mathbf{0}$$

In compact matrix notation:

$$\nabla_{\boldsymbol{\theta}} \epsilon = J(\boldsymbol{\theta})^\top(\mathbf{y} - F(\boldsymbol{\theta})) = \mathbf{0}$$

where

$$F(\boldsymbol{\theta}) = \begin{bmatrix} f(\mathbf{x}_1;\boldsymbol{\theta}) \\ \vdots \\ f(\mathbf{x}_n;\boldsymbol{\theta}) \end{bmatrix}, \quad J(\boldsymbol{\theta}) = \begin{bmatrix} \nabla_{\boldsymbol{\theta}} f(\mathbf{x}_1;\boldsymbol{\theta})^\top \\ \vdots \\ \nabla_{\boldsymbol{\theta}} f(\mathbf{x}_n;\boldsymbol{\theta})^\top \end{bmatrix}$$

J is also referred to as the **Jacobian** of F. Observe that in the special case when f is linear in $\boldsymbol{\theta}$ (i.e. $f(\mathbf{x}_i;\boldsymbol{\theta}) = \boldsymbol{\theta}^\top \mathbf{x}_i$), the gradient $\nabla_{\boldsymbol{\theta}} \epsilon$ will only have $\boldsymbol{\theta}$ in one place because the term $\nabla_{\boldsymbol{\theta}} f(\mathbf{x}_i;\boldsymbol{\theta})$ will only depend on $\mathbf{x}_i$:

$$\nabla_{\boldsymbol{\theta}} \epsilon = 2\sum_{i=1}^{n} (y_i - \boldsymbol{\theta}^\top \mathbf{x}_i)\nabla_{\boldsymbol{\theta}} (\boldsymbol{\theta}^\top \mathbf{x}_i) = 2\sum_{i=1}^{n} (y_i - \boldsymbol{\theta}^\top \mathbf{x}_i)\mathbf{x}_i$$

and we can derive a closed-form solution for $\boldsymbol{\theta}$, arriving at the OLS solution:

$$\begin{aligned} 2\mathbf{X}^\top(\mathbf{y} - \mathbf{X}\boldsymbol{\theta}) &= \mathbf{0} \\ 2\mathbf{X}^\top\mathbf{y} - 2\mathbf{X}^\top\mathbf{X}\boldsymbol{\theta} &= \mathbf{0} \\ \mathbf{X}^\top\mathbf{y} &= \mathbf{X}^\top\mathbf{X}\boldsymbol{\theta} \\ \boldsymbol{\theta} &= (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{y} \end{aligned}$$

However, in the general case where f is nonlinear in $\boldsymbol{\theta}$, it would not be possible to write out a closed-form solution for $\boldsymbol{\theta}$.

Remark: Without more assumptions on f, the NLS objective is not convex in general. This means that the first-order optimality condition is a *necessary* but not *sufficient* condition for a local minimum. That is, it is possible that the derivative is zero for some value of $\boldsymbol{\theta}$, but that value is not a local minimum. It could be a saddle point, or worse, a local maximum! Even if it is a minimum, it may not be the global minimum.

1.2 Gauss-Newton Algorithm

Since there is no closed-form solution to the nonlinear least squares optimization problem, we must resort an iterative algorithm instead. One such algorithm is the **Gauss-Newton algorithm**. At each iteration, this method linearly approximates the function F about the current iterate and solves a least-squares problem involving the linearization in order to compute the next iterate.

Let's say that we have a "guess" for $\boldsymbol{\theta}$ at iteration k, which we denote $\boldsymbol{\theta}^{(k)}$. We consider the first-order approximation of $F(\boldsymbol{\theta})$ about $\boldsymbol{\theta}^{(k)}$:

$$\begin{aligned} F(\boldsymbol{\theta}) \approx \tilde{F}(\boldsymbol{\theta}) &= F(\boldsymbol{\theta}^{(k)}) + \nabla_{\boldsymbol{\theta}} F(\boldsymbol{\theta}^{(k)})(\boldsymbol{\theta} - \boldsymbol{\theta}^{(k)}) \\ &= F(\boldsymbol{\theta}^{(k)}) + J(\boldsymbol{\theta}^{(k)})\Delta\boldsymbol{\theta} \end{aligned}$$

where $\Delta\boldsymbol{\theta} := \boldsymbol{\theta} - \boldsymbol{\theta}^{(k)}$.

Now since $\tilde{F}$ is linear in $\Delta\boldsymbol{\theta}$ (the Jacobian and F are just constants: functions evaluated at $\boldsymbol{\theta}^{(k)}$), we can use the closed form solution for $\Delta\boldsymbol{\theta}$ from the optimality condition to *update* our current guess $\boldsymbol{\theta}^{(k)}$. Applying the first-order optimality condition to the objective $\tilde{F}$ yields the following equation:

$$\mathbf{0} = J_{\tilde{F}}(\boldsymbol{\theta})^\top(\mathbf{y} - \tilde{F}(\boldsymbol{\theta})) = J(\boldsymbol{\theta}^{(k)})^\top \left(\mathbf{y} - \left(F(\boldsymbol{\theta}^{(k)}) + J(\boldsymbol{\theta}^{(k)})\Delta\boldsymbol{\theta}\right)\right)$$

Note that the Jacobian of the linearized function $\tilde{F}$, evaluated at any $\boldsymbol{\theta}$, is precisely $J(\boldsymbol{\theta}^{(k)})$. Writing $\mathbf{J} = J(\boldsymbol{\theta}^{(k)})$ for brevity, we have

$$\begin{aligned} \mathbf{J}^\top\mathbf{y} &= \mathbf{J}^\top(F(\boldsymbol{\theta}^{(k)}) + \mathbf{J}\Delta\boldsymbol{\theta}) \\ \mathbf{J}^\top(\mathbf{y} - F(\boldsymbol{\theta}^{(k)})) &= \mathbf{J}^\top\mathbf{J}(\Delta\boldsymbol{\theta}) \\ \Delta\boldsymbol{\theta} &= (\mathbf{J}^\top\mathbf{J})^{-1}\mathbf{J}^\top(\mathbf{y} - F(\boldsymbol{\theta}^{(k)})) \\ &= (\mathbf{J}^\top\mathbf{J})^{-1}\mathbf{J}^\top\Delta\mathbf{y} \end{aligned}$$

where $\Delta\mathbf{y} := \mathbf{y} - F(\boldsymbol{\theta}^{(k)})$. By comparing this solution to OLS, we see that it is effectively solving

$$\Delta\boldsymbol{\theta} = \arg\min_{\delta\boldsymbol{\theta}} \|\mathbf{J}\delta\boldsymbol{\theta} - \Delta\mathbf{y}\|^2$$

Since $\delta F \approx \mathbf{J}\delta\boldsymbol{\theta}$ is close to $\boldsymbol{\theta}^{(k)}$, this is saying that we choose a change to the weights that corrects for the current error in the function values, but it bases this calculation on the linearization of F. Recalling that $\Delta\boldsymbol{\theta} = \boldsymbol{\theta} - \boldsymbol{\theta}^{(k)}$, we can improve upon our current guess $\boldsymbol{\theta}^{(k)}$ with the update

$$\begin{aligned} \boldsymbol{\theta}^{(k+1)} &= \boldsymbol{\theta}^{(k)} + \Delta\boldsymbol{\theta} \\ &= \boldsymbol{\theta}^{(k)} + (\mathbf{J}^\top\mathbf{J})^{-1}\mathbf{J}^\top\Delta\mathbf{y} \end{aligned}$$

Note that the solution will depend on the initial value $\boldsymbol{\theta}^{(0)}$ in general. Here is the entire process laid out in steps:

Algorithm 1 Gauss-Newton

Initialize $\boldsymbol{\theta}^{(0)}$ with some guess
while $\boldsymbol{\theta}^{(k)}$ has not converged **do**
  Compute Jacobian with respect to the current iterate, $\mathbf{J} = J(\boldsymbol{\theta}^{(k)})$
  Compute $\Delta\mathbf{y} = \mathbf{y} - F(\boldsymbol{\theta}^{(k)})$
  Update: $\boldsymbol{\theta}^{(k+1)} = \boldsymbol{\theta}^{(k)} + (\mathbf{J}^\top\mathbf{J})^{-1}\mathbf{J}^\top\Delta\mathbf{y}$
end while

The choice for measuring convergence is up to the practitioner. Some common choices include testing changes in the objective value:

$$\left|\frac{\epsilon^{(k+1)} - \epsilon^{(k)}}{\epsilon^{(k)}}\right| \leq \text{threshold}$$

or in the iterates themselves:

$$\max_j \left|\frac{\Delta\theta_j}{\theta_j^{(k)}}\right| \leq \text{threshold}$$