

Backpropagation as a Linear Solve

Mikkel Paltor (mpasc@dtu.dk), 5 October 2026

Department of Applied Mathematics and Computer Science, Technical University of Denmark

In the previous post on backpropagation¹, we showed that backpropagation is a special case of the adjoint method. In this post we skip the adjoint description and instead derive the same result directly from the chain rule.

1. The forward relation

We denote the data by u_0 and the parameters by $\theta \in \mathbb{R}^p$. The network has N layers, each of which is a function Φ_i that takes as input outputs from previous layers $(u_{i-1}, \dots, u_0)$ as well as the parameters (θ) , and produces the next layer output u_i . The full forward pass, including all layer outputs, can be expressed as

$$\underbrace{\begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_N \end{bmatrix}}_{u(\theta)} = \underbrace{\begin{bmatrix} \Phi_1(u_0, \theta) \\ \Phi_2(u_1, u_0, \theta) \\ \vdots \\ \Phi_N(u_{N-1}, \dots, u_0, \theta) \end{bmatrix}}_{\Phi(u(\theta), \theta)} \in \mathbb{R}^d, \quad (1)$$

where $d = \sum_{i=1}^N n_i$ and n_i is the size of u_i .

We introduce the projection $P \in \mathbb{R}^{r \times d}$ that takes out relevant layer-outputs, i.e.²

$$u_P(\theta) = Pu(\theta). \quad (2)$$

Using this the loss function can be written as

$$\mathcal{J}(\theta) = \ell(u_P(\theta)) + \mu \mathcal{R}(\theta), \quad (3)$$

where ℓ is the data loss and $\mathcal{R}$ is a regularization term.

2. Differentiating the network

Applying the chain rule to Equation 1, we have

$$\frac{du}{d\theta} = \frac{\partial \Phi}{\partial u} \frac{du}{d\theta} + \frac{\partial \Phi}{\partial \theta}, \quad \text{hence} \quad \frac{du}{d\theta} = \left(\underbrace{I - \frac{\partial \Phi}{\partial u}}_L \right)^{-1} \underbrace{\frac{\partial \Phi}{\partial \theta}}_{M^\top}. \quad (4)$$

Applying the projection P , it further follows that

$$\frac{du_P}{d\theta} = PL^{-1}M^\top. \quad (5)$$

At this point it is worth looking at the structure of the matrices. As the i th layer depends only on previous outputs, the matrix L is block lower-triangular with identity blocks on the diagonal³. The triangular structure makes it efficient to solve the linear system

$$L^\top \Lambda = P^\top, \quad \Lambda \in \mathbb{R}^{d \times r}. \quad (6)$$

Using Λ it follows that

Contents

1. The forward relation
2. Differentiating the network
3. A scalar loss

¹ Earlier post: *Backpropagation is the Adjoint Method*.
mipals.github.io/blog/2025/10-25-adjoint-method/

² Often P just picks out the last layer, i.e. $u_P = u_N = Pu(\theta)$, $P = \begin{pmatrix} 0 & \dots & 0 & I_r \end{pmatrix}$.

³ The sparsity pattern of the strictly lower/upper part of a (block) triangular matrix is a directed acyclic graph (DAG), which is often how backpropagation is introduced.

$$\frac{du_P}{d\theta} = \Lambda^\top M^\top. \quad (7)$$

In practice, as seen in the next section, one rarely needs to form $\frac{du_P}{d\theta}$ as a full matrix; it suffices to define products with it.

3. A scalar loss

Applying the chain rule to Equation 3, we have

$$\nabla_\theta \mathcal{J} = \left(\frac{du_P}{d\theta} \right)^\top \nabla_{u_P} \ell + \mu \nabla_\theta \mathcal{R} = M L^{-\top} P^\top \nabla_{u_P} \ell + \mu \nabla_\theta \mathcal{R}. \quad (8)$$

Given that $\nabla_{u_P} \ell$ is a vector of length r we have that the product $P^\top \nabla_{u_P} \ell$ is a vector of length d . It is therefore of interest to first form $P^\top \nabla_{u_P} \ell$ and then solve $L^\top \lambda = P^\top \nabla_{u_P} \ell$, so that

$$\nabla_\theta \mathcal{J} = M \lambda + \mu \nabla_\theta \mathcal{R}, \quad (9)$$

which is equivalent to the final result of the adjoint method (see the previous post on backpropagation⁴). Note that λ here is the negative of the adjoint variable of the previous post, which absorbs the sign of $\frac{\partial f}{\partial \theta} = -M^\top$ there.

⁴ Earlier post: *Backpropagation is the Adjoint Method*.
mipals.github.io/blog/2025/10-25-adjoint-method/