Consider an input space $X = \mathbb{R}^n$. Suppose you have two scalar-valued constraints, h_1 and h_2, which are satisfied if $h_i = 0$.

$$h_1(\mathbf{x}) \quad J_{h_1}(\mathbf{x}) = \nabla^T_{h_1}(\mathbf{x}) = \left[\begin{array}{cccc} \frac{\partial h_1}{\partial x_1} & \frac{\partial h_1}{\partial x_2} & \dots & \frac{\partial h_1}{\partial x_n} \end{array} \right]$$

$$h_2(\mathbf{x}) \quad J_{h_2}(\mathbf{x}) = \nabla^T_{h_2}(\mathbf{x}) = \left[\begin{array}{cccc} \frac{\partial h_2}{\partial x_1} & \frac{\partial h_2}{\partial x_2} & \dots & \frac{\partial h_2}{\partial x_n} \end{array} \right]$$

We want to combine these into one constraint, and then use Newton's method to project an input point $\mathbf{x}_{in}$ onto the combined constraint.

One option is to stack the constraints into a vector-valued constraint function:

$$\mathbf{h}(\mathbf{x}) = \left[\begin{array}{c} h_1(\mathbf{x}) \\ h_2(\mathbf{x}) \end{array} \right]$$

Then, apply Newton's method directly on $\mathbf{h}$:

$$\mathbf{h}_{\mathbf{p}}(\mathbf{x}) = \mathbf{h}(\mathbf{p}) + J_{\mathbf{h}}(\mathbf{p})\,(\mathbf{x} - \mathbf{p}) = \mathbf{0}$$

$$\mathbf{x} = \mathbf{p} - J^{\dagger}_{\mathbf{h}}(\mathbf{p})\,\mathbf{h}(\mathbf{p})$$

with $J^{\dagger}$ a left-inverse of J.

Alternatively, we could consider various ways of representing our constraint as a scalar-valued function $s(\mathbf{x})$. One common way is to use the ℓ^2 norm:

$$s(\mathbf{x}) = ||\mathbf{h}(\mathbf{x})||$$

$$J_s(\mathbf{x}) = \nabla^T_s(\mathbf{x}) = \frac{\mathbf{h}^T(\mathbf{x})}{||\mathbf{h}(\mathbf{x})||} J_{\mathbf{h}}(\mathbf{x})$$

Applying Newton's method to $\mathbf{s}$ yields:

$$\mathbf{x} = \mathbf{p} - \frac{1}{||\nabla_s(\mathbf{x})||^2} J^T_{\mathbf{h}}(\mathbf{p})\,\mathbf{h}(\mathbf{x})$$

In other words, scalarizing the constraint using the ℓ^2 norm is equivalent to a Jacobian-transpose approach to root-finding.