A. PyTorch implementation

Our proposed algorithm can be easily integrated into existing deep learning architectures. Here, we provide our PyTorch [30] implementation of this module, see Fig. 8. The forward pass is the standard Sinkhorn matrix scaling algorithm [9] with a robust log-space implementation. The backward function contains an implementation of our Algorithm 1.

B. Additional experiments

B.1. Gradient accuracy

Theorem 5 states that the error of our backward pass in Algorithm 1 can be bounded by the error of the forward pass in Eq. (6). We now assess the magnitude of this error in practice by revisiting the experiments on image barycenter computation and number sorting from Sec. 5. Since the ground truth gradients are unknown in general, we define the (approximate) ground truth gradients as $\nabla_{\boldsymbol{C}} \ell^* := \nabla_{\boldsymbol{C}} \ell^{(\tau_{\max})}, \nabla_{\boldsymbol{a}} \ell^* := \nabla_{\boldsymbol{a}} \ell^{(\tau_{\max})}$ for $\tau_{\max} := 10,000$. In Fig. 9, we compare the error of our approach to AD, averaged over all settings and iterations from the experiments in Fig. 4 and Fig. 6 respectively. These results show clearly that, on average, our approach produces more accurate gradients than AD. Intuitively, ours yields optimal gradients for a suboptimal forward pass $\boldsymbol{P}^* \approx \boldsymbol{S}_\lambda^{(\tau)}$, see Theorem 5, whereas AD yields approximate gradients for an approximate forward pass. To further illustrate this point, we show a qualitative comparison of the gradients of both approaches on the task of image barycenter computation in Fig. 10. These results demonstrate the usefulness of our custom backward pass. Our gradients and the ground truth start disagreeing in certain regions for $t \geq 50$, but only after the barycenter optimization converges. Compared to the vanilla AD approach, the gradients are much closer to the ground truth and therefore more useful.

B.2. MNIST image clustering

We mainly consider the barycenter problem described in Sec. 5.2 a useful toy example to assess the stability of implicit differentiation in comparison to AD. On the other hand, the interpretation as a generic optimization problem allows us to trivially extend it to related tasks like image clustering. To that end, we apply the k-means algorithm which alternates between computing cluster centroids (by minimizing Eq. (14)) and assigning all images $\boldsymbol{b}_i$ to their closest centroid (defined as the centroid with minimum distance d, Eq. (15)). We show results on the 60k images from the MNIST dataset [24] in Fig. 11.

B.3. Details on permutation baselines

Gumbel-Sinkhorn networks. As outlined in Sec. 5.3, the goal of the Gumbel-Sinkhorn method [28] is to learn how to sort a set of input elements $\{x_1, \dots, x_n\}$. To this end, the cost matrix $\boldsymbol{C}$ is parameterized via a permutation-invariant neural network architecture (set encoder), conditioned on the input elements $\{x_1, \dots, x_n\}$. The matrix $\boldsymbol{C}$, together with marginals $\boldsymbol{a} = \boldsymbol{b} = \mathbb{1}_n$ are then passed through a differentiable Sinkhorn layer.[2] The final output $\boldsymbol{P}$ is a bistochatic matrix which encodes an approximate permutation. The training objective is a geometric loss, minimizing the distance of the sorted elements x_i to their natural ground-truth ordering. At test time, the Hungarian algorithm [23] is applied to the learned cost matrix $\boldsymbol{C}$ to obtain an exact, hard permutation.

In Sec. 5.3, we show the concrete application of sorting n real numbers, sampled randomly from the uniform distribution $x_i \sim \mathcal{U}(0, 1)$. More specifically, we consider separate training and test sets of 4096 and 128 random sequences each and report the error, defined as the proportion of incorrectly placed numbers in the test set, see Fig. 6. To provide a more complete picture, we report quantitative results on the task of generalizing to different test sets here. Specifically, we train the vanilla GS method and our approach for $\tau = 200$ (the maximum number for which GS is below the GPU memory limit) for $n = 200$ numbers sampled from $\mathcal{U}(0, 1)$. We then investigate the error on test sets sampled from different distributions $x_i \sim \mathcal{U}(s, t)$ with $s < t$ in Tab. 3. Even though the variance is quite high, these results indicate that our method evidently helps to reduce overfitting. Note, that the improved generalization is observed for the setting $\tau = 200$, $n = 200$ where the performance of both methods on the standard test set is almost identical, see Fig. 6.

RPM-Net. A number of recent works use the Sinkhorn operator as a differentiable matching module for deep learning [13, 25, 39, 46, 47]. The standard strategy of such methods is to use a learnable feature extractor to obtain descriptors $\boldsymbol{F}^X \in \mathbb{R}^{m \times p}, \boldsymbol{F}^Y \in \mathbb{R}^{n \times p}$ on pairs of input objects X and Y in a siamese manner. We can then define the cost matrix $\boldsymbol{C} := \boldsymbol{D}$ as the squared pairwise distance matrix $\boldsymbol{D}_{i,j} = \left\| \boldsymbol{F}^X_{:,i} - \boldsymbol{F}^Y_{:,j} \right\|_2^2$. For fixed, uniform marginals $\boldsymbol{a}$ and $\boldsymbol{b}$, the Sinkhorn operator then yields a soft matching $\boldsymbol{P} = S_\lambda(\boldsymbol{C}, \boldsymbol{a}, \boldsymbol{b}) \in \mathbb{R}^{m \times n}$ between the two input objects. The baseline method RPM-Net we consider in Sec. 5.3 uses this methodology to obtain a matching between pairs of input point clouds. As a feature extractor, it uses PointNet [35] with a custom input task point cloud, see [47, Sec. 5.1]

[2] Strictly speaking, the choice of marginals $\boldsymbol{a} = \boldsymbol{b} = \mathbb{1}_n$ does not fit in our framework, since we require $\boldsymbol{a}, \boldsymbol{b} \in \Delta_n$, see Eq. (1). However, we can simply use $\boldsymbol{a} = \boldsymbol{b} = \frac{1}{n}\mathbb{1}_n$ and scale the resulting transportation plan $\boldsymbol{P}$ by a constant factor of n.

```
import torch

class Sinkhorn(torch.autograd.Function):
   @staticmethod
   def forward(ctx, c, a, b, num_sink, lambd_sink):
      log_p = -c / lambd_sink
      log_a = torch.log(a).unsqueeze(dim=1)
      log_b = torch.log(b).unsqueeze(dim=0)
      for _ in range(num_sink):
         log_p -= (torch.logsumexp(log_p, dim=0, keepdim=True) - log_b)
         log_p -= (torch.logsumexp(log_p, dim=1, keepdim=True) - log_a)
      p = torch.exp(log_p)

      ctx.save_for_backward(p, torch.sum(p, dim=1), torch.sum(p, dim=0))
      ctx.lambd_sink = lambd_sink
      return p

   @staticmethod
   def backward(ctx, grad_p):
      p, a, b = ctx.saved_tensors
      m, n = p.shape

      grad_p *= -1 / ctx.lambd_sink * p
      K = torch.cat((
         torch.cat((torch.diag(a), p), dim=1),
         torch.cat((p.T, torch.diag(b)), dim=1)),
         dim=0)[:-1, :-1]
      t = torch.cat((
         grad_p.sum(dim=1),
         grad_p[:, :-1].sum(dim=0)),
         dim=0).unsqueeze(1)
      grad_ab, _ = torch.solve(t, K)
      grad_a = grad_ab[:m, :]
      grad_b = torch.cat((grad_ab[m:, :], torch.zeros([1, 1], device=device,
          dtype=torch.float32)), dim=0)
      U = grad_a + grad_b.T
      grad_p -= p * U
      grad_a = -ctx.lambd_sink * grad_a.squeeze(dim=1)
      grad_b = -ctx.lambd_sink * grad_b.squeeze(dim=1)

      return grad_p, grad_a, grad_b, None, None
```

Figure 8. A PyTorch implementation of Algorithm 1.

	$\mathcal{U}(1,2)$	$\mathcal{U}(10,11)$	$\mathcal{U}(100,101)$	$\mathcal{U}(-1,0)$
Ours	0.2964 ± 0.0744	0.3340 ± 0.3059	0.3531 ± 0.1380	0.3552 ± 0.2116
Gumbel-Sinkhorn	0.3163 ± 0.0976	0.3620 ± 0.3179	0.3955 ± 0.1478	0.4678 ± 0.2526

Table 3. **Number sorting generalization.** We assess the capability of our approach (first row) and the vanilla Gumbel-Sinkhorn method (second row) [28] to generalize to various test sets $\mathcal{U}(s,t)$ with $s < t$. We train both methods to sort sets of $n = 200$ numbers x_i from the uniform distribution on the unit interval $\mathcal{U}(0,1)$ with $\tau = 200$ Sinkhorn iterations. For each test set, we show the mean proportion of false predictions, as well as the empirical standard deviation, obtained from 50 test runs per setting.

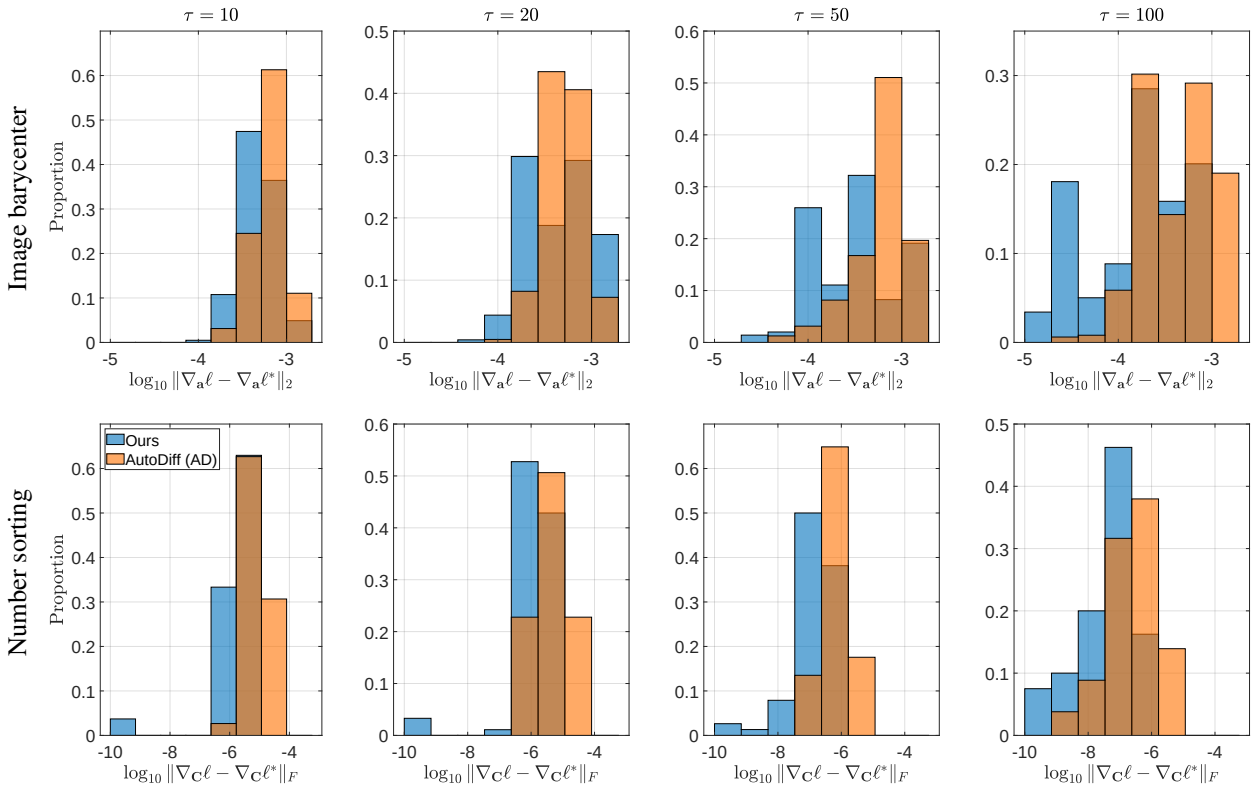

Figure 9. **Gradient accuracy.** We empirically assess the accuracy of the error bound discussed in Theorem 5. Specifically, we show the accuracy of the gradients $\nabla_{\boldsymbol{a}}\ell$ for the image barycenter experiment in Sec. 5.2 (top row) and $\nabla_{\boldsymbol{C}}\ell$ for the number sorting experiment in Sec. 5.3 (bottom row). While both distributions have a large overlap, the gradients from our approach (blue) are noticeably more accurate than AD (orange) on average. Note that all comparisons show histograms on a log scale.

for more details. Using the obtained soft correspondences, a simple SVD transformation then yields the optimal rigid transformation between the two input point clouds. In order to train their model, RPM-Net uses automatic differentiation of the Sinkhorn layer. We can now demonstrate that replacing AD with our backward algorithm improves the performance. To that end, we revisit the experiments from [47, Sec. 6]: We use 20 separate object identities of the ModelNet40 dataset [44] as our train and test set respectively. In Fig. 12, we now show a qualitative comparison corresponding to the results in Tab. 2 in the main paper. On the standard test set, both approaches produce comparable, high-quality results. On the other hand, our method significantly improves the robustness when generalizing to noisy data or partial views.

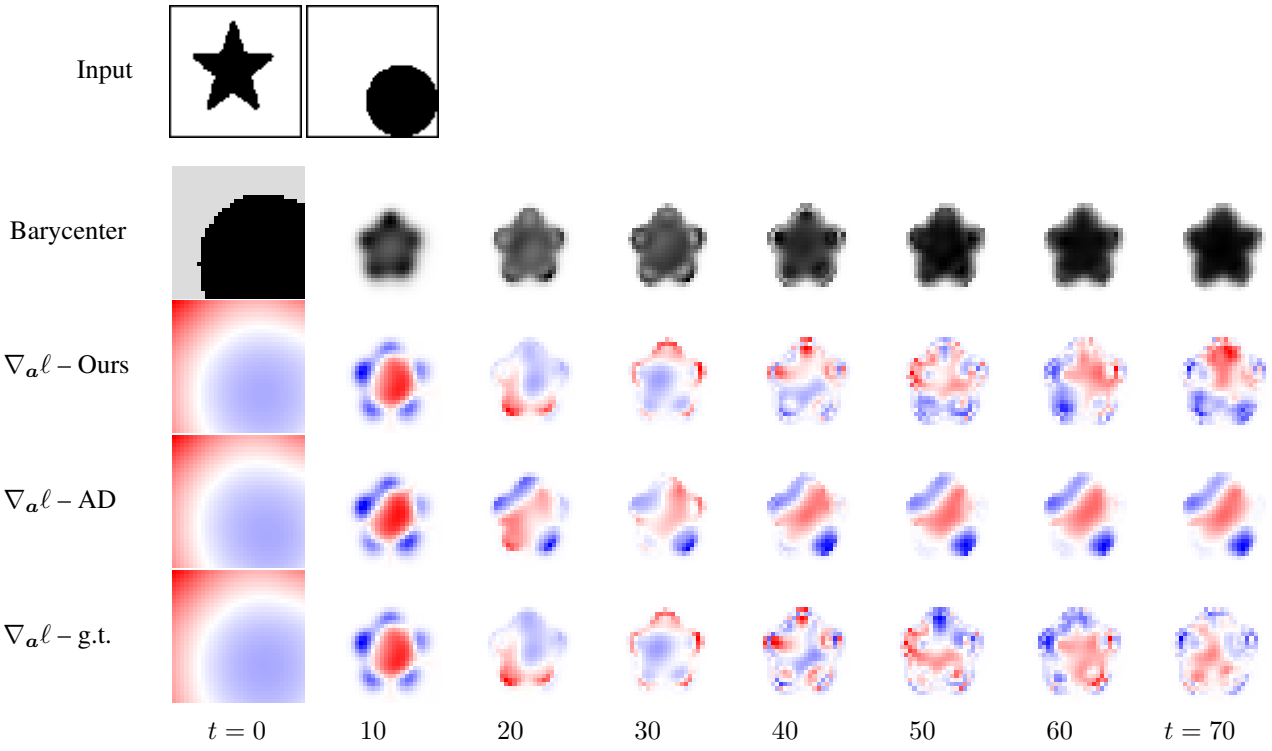

Figure 10. **Image barycenter gradients.** A qualitative comparison of our gradients (3rd row), the AD gradients (4th row), and the ground truth gradients (last row) for the image barycenter experiment from Sec. 5.2. Specifically, we consider the task of interpolating between two input images (1st row) with uniform interpolation weights $w_1 = w_2 = 0.5$. We show intermediate snapshots of the obtained barycenter image (2nd row) for different numbers of gradient descent iterations $t \in \{0, \dots, 70\}$ that result from minimizing the energy in Equation 14.

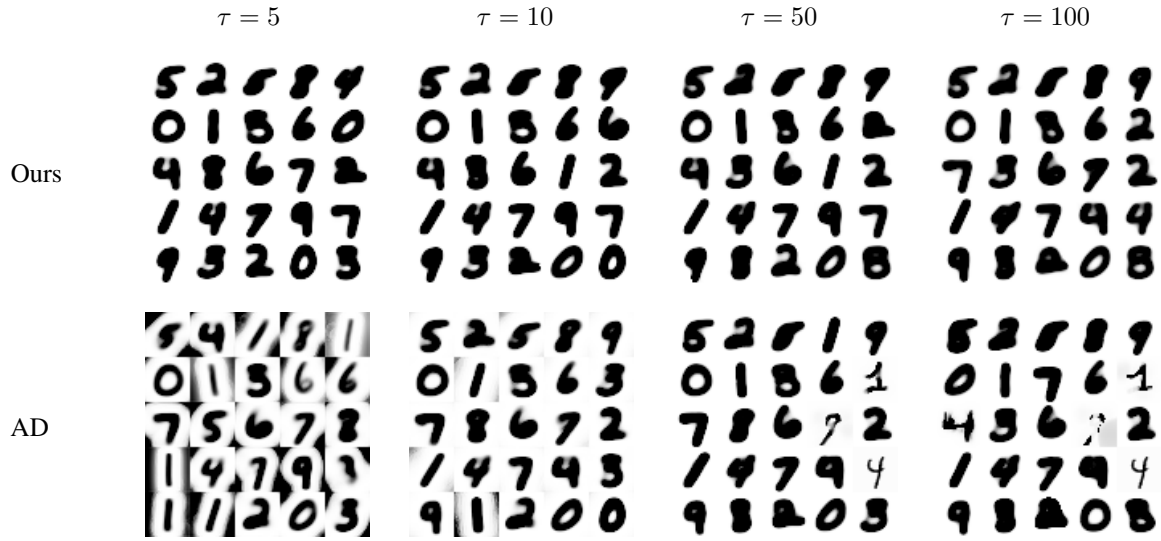

Figure 11. **MNIST k-means clustering**. A comparison of implicit differentiation and AD on the task of k-means image clustering. For both approaches, we show the predicted $k = 25$ clusters for $\tau \in \{5, 10, 50, 100\}$ Sinkhorn iterations. We choose more than 10 clusters to capture several different appearances and styles for each digit. To make individual results more comparable, we use an identical random initialization of the cluster centroids for all settings. For AD, the maximum permissible batch sizes for the 4 settings are $512, 256, 64, 32$, whereas the implicit differentiation algorithm consistently allows for a batch size of 1024.

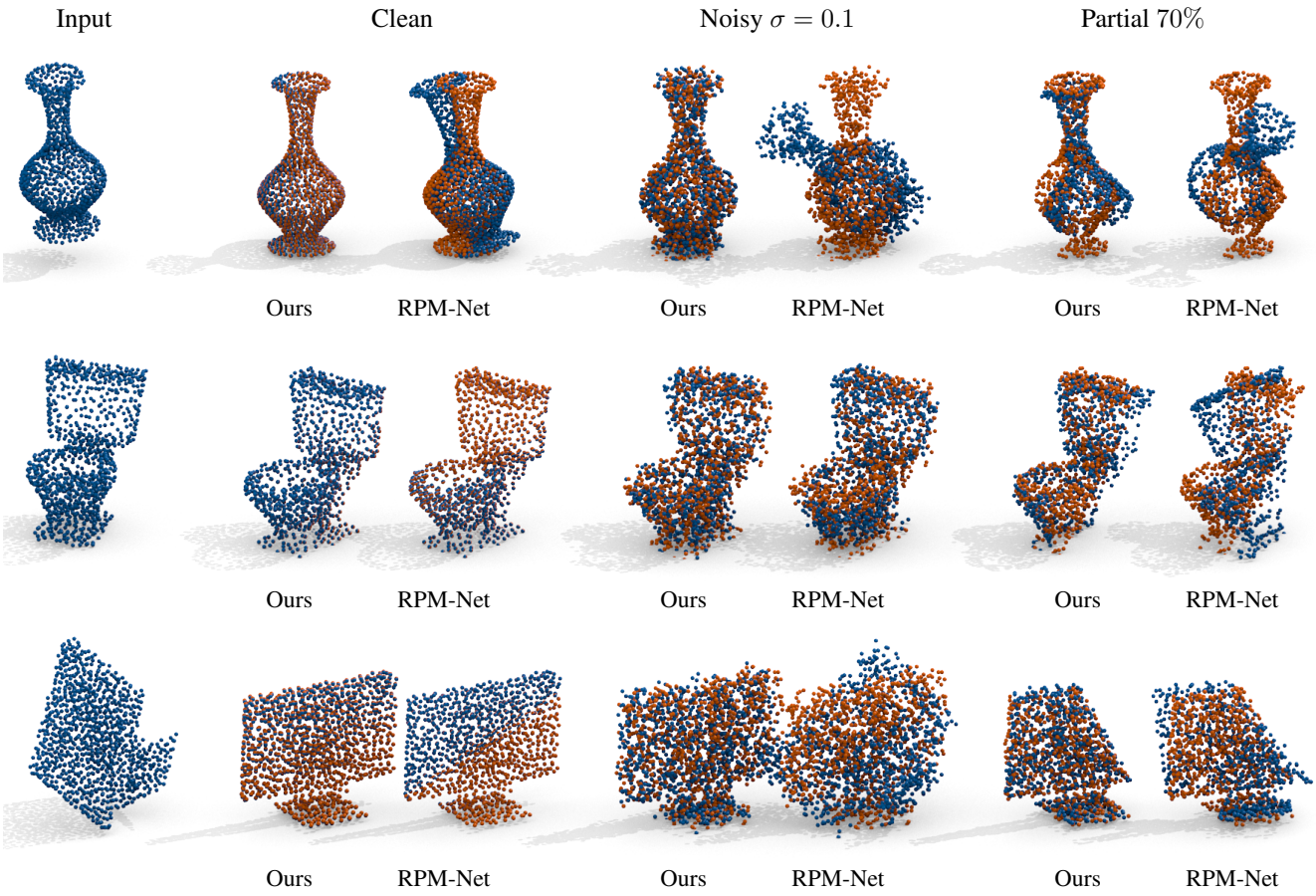

Figure 12. **Point cloud registration.** Additional qualitative examples of RPM-Net [47] and the augmented version with our custom backward pass, see Algorithm 1. The increased stability of the gradients predicted by our algorithm directly translates into more robust generalization results: Both methods are trained and tested on separate subsets of the 40 object categories in ModelNet40 [44], see [47, Section 6] for more details. Both methods yield accurate predictions for the clean test data, as indicated by the corresponding quantitative results in Tab. 2. On the other hand, our approach shows significant improvements when generalizing to noisy test data and partial inputs.

C. Proofs

In the following, we provide proofs of Lemma 1, Lemma 2, Theorem 3, Theorem 4 and Theorem 5 from the main paper.

C.1. Proof of Lemma 1

Proof. The function $\mathcal{K}$ contains the KKT conditions corresponding to the optimization problem in Eq. (4). The proposed identity therefore follows directly from the (strict) convexity of Eq. (4), see [15]. Apart from the two equality constraints, Eq. (3) contains additional inequality constraints $P_{i,j} \geq 0$. Those are however inactive and can be dropped, because the entropy term in Eq. (4) invariably yields transportation plans in the interior of the positive orthant $P_{i,j} > 0$, see [33, p. 68]. □

C.2. Proof of Lemma 2

Proof. The key for proving this statement is applying the implicit function theorem. We start by (a) showing that this indeed yields the identity from Eq. (9), and then (b) justify why removing the last $(m+n)$-th equality condition from $\boldsymbol{E}$ is necessary.

(a) First of all, we can verify by direct computation that the matrix $\boldsymbol{K}$ from Eq. (9) corresponds to the partial derivatives of the KKT conditions from Eq. (8)

$$\boldsymbol{K} = \left(\frac{\partial \mathcal{K}(\boldsymbol{c}, \boldsymbol{a}, \boldsymbol{b}, \boldsymbol{p}, \boldsymbol{\alpha}, \boldsymbol{\beta})}{\partial \left[\boldsymbol{p}; \boldsymbol{\alpha}; \boldsymbol{\beta}\right]} \right)_{-l,-l}, \tag{16}$$

where the notation $(\cdot)_{-l,-l}$ means that the last row and column is removed and where $l = mn + m + n$. Furthermore, $\boldsymbol{K}$ is invertible (since the solution lies in the interior $\boldsymbol{P}_{i,j} > 0$, see [33, p. 68]), and the $m + n - 1$ columns of $\tilde{\boldsymbol{E}}$ are linearly dependent, see (b). Consequently, the implicit function theorem states that $\mathcal{K}$ implicitly defines a mapping $(\boldsymbol{c}, \boldsymbol{a}, \boldsymbol{b}) \mapsto (\boldsymbol{p}, \boldsymbol{\alpha}, \boldsymbol{\beta})$ whose Jacobian is

$$\boldsymbol{J} = \frac{\partial \left[\boldsymbol{p}; \boldsymbol{\alpha}; \tilde{\boldsymbol{\beta}}\right]}{\partial \left[\boldsymbol{c}; -\boldsymbol{a}; -\tilde{\boldsymbol{b}}\right]} = -\left(\frac{\partial \mathcal{K}}{\partial \left[\boldsymbol{p}; \boldsymbol{\alpha}; \boldsymbol{\beta}\right]} \right)^{-1}_{-l,-l} \underbrace{\left(\frac{\partial \mathcal{K}}{\partial \left[\boldsymbol{c}; -\boldsymbol{a}; -\boldsymbol{b}\right]} \right)_{-l,-l}}_{=\boldsymbol{I}_{l-1}} = -\boldsymbol{K}^{-1}. \tag{17}$$

(b) As part of the proof in (a), we use the fact that the columns of $\tilde{\boldsymbol{E}}$ are linearly independent. Verifying this statement also provides insight as to why removing the last row of $\tilde{\boldsymbol{b}}, \tilde{\boldsymbol{\beta}}$ and the last column of $\tilde{\boldsymbol{E}}$ is necessary. Intuitively, the columns of $\boldsymbol{E}$ contain one redundant condition: The identity

$$\sum_{i=1}^{m} \boldsymbol{P}_{i,n} = 0 \iff \boldsymbol{E}_{:,m+n}^{\top} \boldsymbol{p} = 0, \tag{18}$$

follows directly from the other $m + n - 1$ conditions. More formally, we can take the kernel

$$\ker(\boldsymbol{E}^{\top}) = \left\{ \boldsymbol{P} \in \mathbb{R}^{m \times n} | \boldsymbol{E}^{\top} \boldsymbol{p} = 0 \right\}, \tag{19}$$

of $\boldsymbol{E}^{\top} \in \mathbb{R}^{m+n \times mn}$ and observe that $\dim\big(\ker(\boldsymbol{E}^{\top})\big) = (m-1)(n-1)$, see [6, Sec. 1]. The rank–nullity theorem then implies that the dimension of the subspace spanned by the columns of $\boldsymbol{E}$ is of dimension $mn - (m-1)(n-1) = m + n - 1$. Consequently, removing the redundant condition in Eq. (19) from $\boldsymbol{E}$ yields the reduced $\tilde{\boldsymbol{E}} \in \mathbb{R}^{mn \times m+n-1}$ with $= m + n - 1$ linearly independent columns.

□

C.3. Proof of Theorem 3

Proof. This identity follows trivially from Lemma 2 by applying the chain rule

$$\nabla_{[\boldsymbol{c};-\boldsymbol{a};-\tilde{\boldsymbol{b}}]} \ell = \left(\frac{\partial \left[\boldsymbol{p}; \boldsymbol{\alpha}; \tilde{\boldsymbol{\beta}}\right]}{\partial \left[\boldsymbol{c}; -\boldsymbol{a}; -\tilde{\boldsymbol{b}}\right]} \right)^{\top} \nabla_{[\boldsymbol{p};\boldsymbol{\alpha};\tilde{\boldsymbol{\beta}}]} \ell = -\boldsymbol{K}^{-1} \nabla_{[\boldsymbol{p};\boldsymbol{\alpha};\tilde{\boldsymbol{\beta}}]} \ell = \boldsymbol{K}^{-1} \begin{bmatrix} -\nabla_{\boldsymbol{p}} \ell \\ \boldsymbol{0} \end{bmatrix}. \tag{20}$$

The last equality holds, since the loss ℓ does not depend on the dual variables $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, see Fig. 2. □

C.4. Proof of Theorem 4

Proof. We want to show that the gradients $\nabla_{\boldsymbol{C}}\ell, \nabla_{\boldsymbol{a}}\ell, \nabla_{\boldsymbol{b}}\ell$ obtained with Algorithm 1 are equivalent to the solution of Eq. (11). To that end, we start by applying the Schur complement trick to the block matrix $\boldsymbol{K}$. This yields the expression

$$\left(\boldsymbol{K}^{-1}\right)_{:,1:mn} = \begin{bmatrix} \lambda^{-1}\operatorname{diag}(\boldsymbol{p})\big(\boldsymbol{I}_{mn} + \tilde{\boldsymbol{E}}\big(\tilde{\boldsymbol{E}}^\top \operatorname{diag}(\boldsymbol{p})\tilde{\boldsymbol{E}}\big)^{-1}\tilde{\boldsymbol{E}}^\top \operatorname{diag}(\boldsymbol{p})\big) \\ \big(\tilde{\boldsymbol{E}}^\top \operatorname{diag}(\boldsymbol{p})\tilde{\boldsymbol{E}}\big)^{-1}\tilde{\boldsymbol{E}}^\top \operatorname{diag}(\boldsymbol{p}) \end{bmatrix}, \tag{21}$$

for the first mn columns of its inverse. In the next step, we can insert this expression in Eq. (11) and invert the linear system of equations

$$\begin{bmatrix} \nabla_{\boldsymbol{c}}\ell \\ -\nabla_{[\boldsymbol{a};\tilde{\boldsymbol{b}}]}\ell \end{bmatrix} = \boldsymbol{K}^{-1}\begin{bmatrix} -\nabla_{\boldsymbol{p}}\ell \\ \boldsymbol{0} \end{bmatrix} = -\left(\boldsymbol{K}^{-1}\right)_{:,1:mn}\nabla_{\boldsymbol{p}}\ell. \tag{22}$$

Further simplification yields the following identities for the gradients of $\boldsymbol{C}, \boldsymbol{a}$ and $\boldsymbol{b}$

$$\begin{aligned} \begin{bmatrix} \nabla_{\boldsymbol{c}}\ell \\ \nabla_{[\boldsymbol{a};\tilde{\boldsymbol{b}}]}\ell \end{bmatrix} &= \begin{bmatrix} -\lambda^{-1}\operatorname{diag}(\boldsymbol{p})\big(\boldsymbol{I}_{mn} - \tilde{\boldsymbol{E}}\big(\tilde{\boldsymbol{E}}^\top \operatorname{diag}(\boldsymbol{p})\tilde{\boldsymbol{E}}\big)^{-1}\tilde{\boldsymbol{E}}^\top \operatorname{diag}(\boldsymbol{p})\big)\nabla_{\boldsymbol{p}}\ell \\ \big(\tilde{\boldsymbol{E}}^\top \operatorname{diag}(\boldsymbol{p})\tilde{\boldsymbol{E}}\big)^{-1}\tilde{\boldsymbol{E}}^\top \operatorname{diag}(\boldsymbol{p})\nabla_{\boldsymbol{p}}\ell \end{bmatrix} \\ &= \begin{bmatrix} -\lambda^{-1}\big(\operatorname{diag}(\boldsymbol{p})\nabla_{\boldsymbol{p}}\ell - \operatorname{diag}(\boldsymbol{p})\tilde{\boldsymbol{E}}\nabla_{[\boldsymbol{a};\tilde{\boldsymbol{b}}]}\ell\big) \\ \big(\tilde{\boldsymbol{E}}^\top \operatorname{diag}(\boldsymbol{p})\tilde{\boldsymbol{E}}\big)^{-1}\tilde{\boldsymbol{E}}^\top \operatorname{diag}(\boldsymbol{p})\nabla_{\boldsymbol{p}}\ell \end{bmatrix}, \end{aligned} \tag{23}$$

where the latter equality results from substituting the obtained expression for $\nabla_{[\boldsymbol{a};\tilde{\boldsymbol{b}}]}\ell$ in the first block row. In the remainder of this proof, we can show line by line that these expressions yield Algorithm 1. The main idea is to first compute the second block row identity in Eq. (23) and then use the result to eventually obtain $\nabla_{\boldsymbol{c}}\ell$ from the first block row:

Line 1 The first line defines the matrix $\boldsymbol{T} := \boldsymbol{P} \odot \nabla_{\boldsymbol{P}}\ell$ via the Hadamard product $\odot$. In vectorized form it corresponds to the expression

$$\boldsymbol{t} = \operatorname{diag}(\boldsymbol{p})\nabla_{\boldsymbol{p}}\ell. \tag{24}$$

Line 2 As detailed in Lemma 2, we remove the last equality condition from $\boldsymbol{E}$ to obtain $\tilde{\boldsymbol{E}}$. Equivalent considerations require us to introduce the truncated versions $\tilde{\boldsymbol{T}}, \tilde{\boldsymbol{P}}$ of $\boldsymbol{T}, \boldsymbol{P}$.

Line 3 The operator $\boldsymbol{E}^\top$ then maps $\boldsymbol{t}$ to the vector

$$\boldsymbol{E}^\top\boldsymbol{t} = \begin{bmatrix} \mathbb{1}_n \otimes \boldsymbol{I}_m & \boldsymbol{I}_n \otimes \mathbb{1}_m \end{bmatrix}^\top \boldsymbol{t} = \begin{bmatrix} (\mathbb{1}_n^\top \otimes \boldsymbol{I}_m)\boldsymbol{t} \\ (\boldsymbol{I}_n \otimes \mathbb{1}_m^\top)\boldsymbol{t} \end{bmatrix} = \begin{bmatrix} \boldsymbol{T}\mathbb{1}_n \\ \boldsymbol{T}^\top\mathbb{1}_m \end{bmatrix}, \tag{25}$$

that contains its row and column sums. In terms of the truncated $\tilde{\boldsymbol{E}}$, the last row of Eq. (25) gets removed, thus

$$\tilde{\boldsymbol{E}}^\top\boldsymbol{t} = \begin{bmatrix} \boldsymbol{T}\mathbb{1}_n \\ \tilde{\boldsymbol{T}}^\top\mathbb{1}_m \end{bmatrix} = \begin{bmatrix} \boldsymbol{t}^{(a)} \\ \tilde{\boldsymbol{t}}^{(b)} \end{bmatrix}. \tag{26}$$

Line 4 A direction computation reveals that

$$\begin{aligned} \boldsymbol{E}^\top \operatorname{diag}(\boldsymbol{p})\boldsymbol{E} &= \begin{bmatrix} (\mathbb{1}_n^\top \otimes \boldsymbol{I}_m)\operatorname{diag}(\boldsymbol{p})(\mathbb{1}_n \otimes \boldsymbol{I}_m) & (\mathbb{1}_n^\top \otimes \boldsymbol{I}_m)\operatorname{diag}(\boldsymbol{p})(\boldsymbol{I}_n \otimes \mathbb{1}_m) \\ (\boldsymbol{I}_n \otimes \mathbb{1}_m^\top)\operatorname{diag}(\boldsymbol{p})(\mathbb{1}_n \otimes \boldsymbol{I}_m) & (\boldsymbol{I}_n \otimes \mathbb{1}_m^\top)\operatorname{diag}(\boldsymbol{p})(\boldsymbol{I}_n \otimes \mathbb{1}_m) \end{bmatrix} \\ &= \begin{bmatrix} \operatorname{diag}(\boldsymbol{P}\mathbb{1}_n) & \boldsymbol{P} \\ \boldsymbol{P}^\top & \operatorname{diag}(\boldsymbol{P}^\top\mathbb{1}_m) \end{bmatrix} = \begin{bmatrix} \operatorname{diag}(\boldsymbol{a}) & \boldsymbol{P} \\ \boldsymbol{P}^\top & \operatorname{diag}(\boldsymbol{b}) \end{bmatrix}. \end{aligned} \tag{27}$$

The linear system in Line 4 of Algorithm 1 therefore yields the gradients $\nabla_{[\boldsymbol{a};\tilde{\boldsymbol{b}}]}\ell$ by inserting Eq. (24), Eq. (26) and (the reduced version of) Eq. (27) into the second block row identity from Eq. (23).

Line 5 We can expand $\nabla_{\tilde{\boldsymbol{b}}}\ell$ to $\nabla_{\boldsymbol{b}}\ell$ by appending zero $\nabla_{b_n}\ell = 0$ as the last entry. Since $\boldsymbol{b}$ is constrained to the probability simplex Δ_n this gradient is exact for all entries, see the discussion in Sec. 4.4.

Line 6 Having computed the gradients $\nabla_{[\boldsymbol{a};\boldsymbol{b}]}\ell$, we can now insert them in the first row of Eq. (23). Here, the reduced and the original expressions are equivalent

$$\tilde{\boldsymbol{E}}\nabla_{[\boldsymbol{a};\tilde{\boldsymbol{b}}]}\ell = \boldsymbol{E}\nabla_{[\boldsymbol{a};\boldsymbol{b}]}\ell \tag{28}$$

because Line 5 specifies $\nabla_{b_n}\ell = 0$. Thus,

$$\tilde{\boldsymbol{E}}\nabla_{[\boldsymbol{a};\tilde{\boldsymbol{b}}]}\ell = \begin{bmatrix}\mathbb{1}_n \otimes \boldsymbol{I}_m & \boldsymbol{I}_n \otimes \mathbb{1}_m\end{bmatrix}\nabla_{[\boldsymbol{a};\boldsymbol{b}]}\ell = \mathbb{1}_n \otimes \nabla_{\boldsymbol{a}}\ell + \nabla_{\boldsymbol{b}}\ell \otimes \mathbb{1}_m =: \boldsymbol{u}, \tag{29}$$

defines the vectorized version of $\boldsymbol{U}$ from Line 6.

Line 7 Putting everything together, we insert the identities from Eq. (24) and Eq. (28) into the first block row of Eq. (23)

$$\nabla_{\boldsymbol{c}}\ell = -\lambda^{-1}\big(\operatorname{diag}(\boldsymbol{p})\nabla_{\boldsymbol{p}}\ell - \operatorname{diag}(\boldsymbol{p})\tilde{\boldsymbol{E}}\nabla_{[\boldsymbol{a};\tilde{\boldsymbol{b}}]}\ell\big) = -\lambda^{-1}\big(\boldsymbol{t} - \operatorname{diag}(\boldsymbol{p})\boldsymbol{u}\big), \tag{30}$$

which is equivalent to the matrix-valued expression in Line 7.

□

C.5. Proof of Theorem 5

Proof. The key for constructing the error bounds in Eq. (13a) and Eq. (13b) is finding a bound for the first-order derivatives $\frac{\partial\nabla_{\boldsymbol{c}}\ell}{\partial\boldsymbol{p}}$ and $\frac{\partial\nabla_{[\boldsymbol{a};\boldsymbol{b}]}\ell}{\partial\boldsymbol{p}}$. For brevity, we introduce the short-hand notation $\bar{\boldsymbol{P}} := \operatorname{diag}(\boldsymbol{p})$. Furthermore, we define the projection of $\boldsymbol{x}$ onto the column space of $\tilde{\boldsymbol{E}}$ as

$$\boldsymbol{\Pi}_{\boldsymbol{E}}\boldsymbol{x} := \operatorname*{arg\,min}_{\boldsymbol{y}\in\operatorname{span}(\tilde{\boldsymbol{E}})}\|\boldsymbol{x}-\boldsymbol{y}\|_{\bar{\boldsymbol{P}}}^2 = \tilde{\boldsymbol{E}}\operatorname*{arg\,min}_{\boldsymbol{z}\in\mathbb{R}^{m+n-1}}\|\boldsymbol{x}-\tilde{\boldsymbol{E}}\boldsymbol{z}\|_{\bar{\boldsymbol{P}}}^2, \tag{31}$$

where $\langle\cdot,\cdot\rangle_{\bar{\boldsymbol{P}}} := \langle\bar{\boldsymbol{P}}^{\frac{1}{2}}\cdot,\bar{\boldsymbol{P}}^{\frac{1}{2}}\cdot\rangle_2$. In matrix notation, Eq. (31) reads

$$\boldsymbol{\Pi}_{\boldsymbol{E}} = \tilde{\boldsymbol{E}}\big(\tilde{\boldsymbol{E}}^\top\bar{\boldsymbol{P}}\tilde{\boldsymbol{E}}\big)^{-1}\tilde{\boldsymbol{E}}^\top\bar{\boldsymbol{P}}. \tag{32}$$

Using Eq. (12a) and Eq. (12b) from the proof of Theorem 4, we can then rewrite the backward pass compactly as

$$\nabla_{[\boldsymbol{a};\tilde{\boldsymbol{b}}]}\ell = \tilde{\boldsymbol{E}}^\dagger\boldsymbol{\Pi}_{\boldsymbol{E}}\nabla_{\boldsymbol{p}}\ell, \quad \text{and} \tag{33a}$$

$$\nabla_{\boldsymbol{c}}\ell = -\lambda^{-1}\big(\bar{\boldsymbol{P}}(\boldsymbol{I}-\boldsymbol{\Pi}_{\boldsymbol{E}})\nabla_{\boldsymbol{p}}\ell\big), \tag{33b}$$

The first identity follows from $\tilde{\boldsymbol{E}}^\dagger\tilde{\boldsymbol{E}} = \boldsymbol{I}$, since the columns of $\tilde{\boldsymbol{E}}$ are linearly independent (see part (b) of the proof of Lemma 2 in Appendix C.2). Direct substitution of Eq. (12a) into Eq. (12b) immediately yields Eq. (33b). To differentiate $\nabla_{[\boldsymbol{a};\tilde{\boldsymbol{b}}]}\ell$ and $\nabla_{\boldsymbol{c}}\ell$, we apply the chain rule which in turn requires a closed-form expression for the derivative of the projection operator $\boldsymbol{\Pi}_{\boldsymbol{E}}$. Since it is defined as the solution of an optimization problem, we apply the implicit function theorem to the gradient of the objective in Eq. (31), i.e.

$$\nabla_{\boldsymbol{z}}\left(\frac{1}{2}\|\boldsymbol{x}-\tilde{\boldsymbol{E}}\boldsymbol{z}\|_{\bar{\boldsymbol{P}}}^2\right) = \tilde{\boldsymbol{E}}^\top\bar{\boldsymbol{P}}\tilde{\boldsymbol{E}}\boldsymbol{z} - \tilde{\boldsymbol{E}}^\top\bar{\boldsymbol{P}}\boldsymbol{x} = \boldsymbol{0}. \tag{34}$$

The Jacobian of the mapping $\boldsymbol{p}\mapsto\boldsymbol{\Pi}_{\boldsymbol{E}}\boldsymbol{x}$ can therefore be written in terms of the IFT as

$$\frac{\partial\boldsymbol{\Pi}_{\boldsymbol{E}}\boldsymbol{x}}{\partial\boldsymbol{p}} = \tilde{\boldsymbol{E}}(\tilde{\boldsymbol{E}}^\top\bar{\boldsymbol{P}}\tilde{\boldsymbol{E}})^{-1}\tilde{\boldsymbol{E}}^\top\operatorname{diag}\big(\boldsymbol{x}-\boldsymbol{\Pi}_{\boldsymbol{E}}\boldsymbol{x}\big), \tag{35}$$

This auxiliary result implies that the Jacobians of the mappings $\boldsymbol{p}\mapsto\nabla_{[\boldsymbol{a};\tilde{\boldsymbol{b}}]}\ell$ and $\boldsymbol{p}\mapsto\nabla_{\boldsymbol{c}}\ell$ defined in Eq. (33a) and Eq. (33b) are

$$\frac{\partial\nabla_{[\boldsymbol{a};\tilde{\boldsymbol{b}}]}\ell}{\partial\boldsymbol{p}} = \tilde{\boldsymbol{E}}^\dagger\frac{\partial}{\partial\boldsymbol{p}}\boldsymbol{\Pi}_{\boldsymbol{E}}\nabla_{\boldsymbol{p}}\ell = (\tilde{\boldsymbol{E}}^\top\bar{\boldsymbol{P}}\tilde{\boldsymbol{E}})^{-1}\tilde{\boldsymbol{E}}^\top\operatorname{diag}\big((\boldsymbol{I}-\boldsymbol{\Pi}_{\boldsymbol{E}})\nabla_{\boldsymbol{p}}\ell\big) + \tilde{\boldsymbol{E}}^\dagger\boldsymbol{\Pi}_{\boldsymbol{E}}\nabla_{\boldsymbol{p}}^2\ell, \quad \text{and} \tag{36a}$$

$$\begin{aligned}\frac{\partial\nabla_{\boldsymbol{c}}\ell}{\partial\boldsymbol{p}} &= -\lambda^{-1}\left(\operatorname{diag}\big((\boldsymbol{I}-\boldsymbol{\Pi}_{\boldsymbol{E}})\nabla_{\boldsymbol{p}}\ell\big) - \boldsymbol{\Pi}_{\boldsymbol{E}}^\top\operatorname{diag}\big((\boldsymbol{I}-\boldsymbol{\Pi}_{\boldsymbol{E}})\nabla_{\boldsymbol{p}}\ell\big) + \bar{\boldsymbol{P}}(\boldsymbol{I}-\boldsymbol{\Pi}_{\boldsymbol{E}})\nabla_{\boldsymbol{p}}^2\ell\right)\\ &= -\lambda^{-1}\left(\big(\boldsymbol{I}-\boldsymbol{\Pi}_{\boldsymbol{E}}^\top\big)\operatorname{diag}\big((\boldsymbol{I}-\boldsymbol{\Pi}_{\boldsymbol{E}})\nabla_{\boldsymbol{p}}\ell\big) + \bar{\boldsymbol{P}}(\boldsymbol{I}-\boldsymbol{\Pi}_{\boldsymbol{E}})\nabla_{\boldsymbol{p}}^2\ell\right).\end{aligned} \tag{36b}$$

In order to bound the errors of these two gradients, we first derive an upper bound for the norm of the operator $\boldsymbol{\Pi}_{\boldsymbol{E}}$. An important insight is that we can precondition $\boldsymbol{\Pi}_{\boldsymbol{E}}$ via $\bar{\boldsymbol{P}}^{\frac{1}{2}}$

$$\bar{\boldsymbol{P}}^{\frac{1}{2}}\boldsymbol{\Pi}_{\boldsymbol{E}}\bar{\boldsymbol{P}}^{-\frac{1}{2}}\boldsymbol{x} = \underset{\boldsymbol{y}\in\operatorname{span}(\bar{\boldsymbol{P}}^{\frac{1}{2}}\tilde{\boldsymbol{E}})}{\arg\min} \|\boldsymbol{x}-\boldsymbol{y}\|_2^2, \tag{37}$$

which results in an orthogonal projection $\bar{\boldsymbol{P}}^{\frac{1}{2}}\boldsymbol{\Pi}_{\boldsymbol{E}}\bar{\boldsymbol{P}}^{-\frac{1}{2}}$. Since such projections have a spectral radius of at most 1, we can bound the norm of $\boldsymbol{\Pi}_{\boldsymbol{E}}$ as

$$\big\|\boldsymbol{\Pi}_{\boldsymbol{E}}\big\|_2 \leq \big\|\bar{\boldsymbol{P}}^{-\frac{1}{2}}\big\|_2\big\|\bar{\boldsymbol{P}}^{\frac{1}{2}}\boldsymbol{\Pi}_{\boldsymbol{E}}\bar{\boldsymbol{P}}^{-\frac{1}{2}}\big\|_2\big\|\bar{\boldsymbol{P}}^{\frac{1}{2}}\big\|_2 \leq \big\|\bar{\boldsymbol{P}}^{-\frac{1}{2}}\big\|_2\big\|\bar{\boldsymbol{P}}^{\frac{1}{2}}\big\|_2, \tag{38}$$

and equivalently show for the complementary projector $\boldsymbol{I}-\boldsymbol{\Pi}_{\boldsymbol{E}}$ that

$$\big\|\boldsymbol{I}-\boldsymbol{\Pi}_{\boldsymbol{E}}\big\|_2 \leq \big\|\bar{\boldsymbol{P}}^{-\frac{1}{2}}\big\|_2\big\|\bar{\boldsymbol{P}}^{\frac{1}{2}}(\boldsymbol{I}-\boldsymbol{\Pi}_{\boldsymbol{E}})\bar{\boldsymbol{P}}^{-\frac{1}{2}}\big\|_2\big\|\bar{\boldsymbol{P}}^{\frac{1}{2}}\big\|_2 \leq \big\|\bar{\boldsymbol{P}}^{-\frac{1}{2}}\big\|_2\big\|\bar{\boldsymbol{P}}^{\frac{1}{2}}\big\|_2. \tag{39}$$

The Jacobians of the backward pass can then be bounded as

$$\begin{aligned}
\left\|\frac{\partial\nabla_{[\boldsymbol{a};\tilde{\boldsymbol{b}}]}\ell}{\partial\boldsymbol{p}}\right\|_F \leq& \big\|(\tilde{\boldsymbol{E}}^\top\bar{\boldsymbol{P}}\tilde{\boldsymbol{E}})^{-1}\tilde{\boldsymbol{E}}^\top\big\|_2\big\|(\boldsymbol{I}-\boldsymbol{\Pi}_{\boldsymbol{E}})\nabla_{\boldsymbol{p}}\ell\big\|_2 + \big\|\tilde{\boldsymbol{E}}^\dagger\boldsymbol{\Pi}_{\boldsymbol{E}}\nabla^2_{\boldsymbol{p}}\ell\big\|_F \\
=& \big\|\tilde{\boldsymbol{E}}^\dagger\boldsymbol{\Pi}_{\boldsymbol{E}}\bar{\boldsymbol{P}}^{-1}\big\|_2\big\|(\boldsymbol{I}-\boldsymbol{\Pi}_{\boldsymbol{E}})\nabla_{\boldsymbol{p}}\ell\big\|_2 + \big\|\tilde{\boldsymbol{E}}^\dagger\boldsymbol{\Pi}_{\boldsymbol{E}}\nabla^2_{\boldsymbol{p}}\ell\big\|_F \\
\leq& \big\|\tilde{\boldsymbol{E}}^\dagger\big\|_2\big\|\bar{\boldsymbol{P}}^{-\frac{1}{2}}\big\|_2^2\big\|\boldsymbol{I}-\boldsymbol{\Pi}_{\boldsymbol{E}}\big\|_2\big\|\nabla_{\boldsymbol{p}}\ell\big\|_2 + \big\|\tilde{\boldsymbol{E}}^\dagger\big\|_2\big\|\boldsymbol{\Pi}_{\boldsymbol{E}}\big\|_2\big\|\nabla^2_{\boldsymbol{p}}\ell\big\|_F \\
\leq& \big\|\tilde{\boldsymbol{E}}^\dagger\big\|_2\big\|\bar{\boldsymbol{P}}^{-\frac{1}{2}}\big\|_2\big\|\bar{\boldsymbol{P}}^{\frac{1}{2}}\big\|_2\left(\big\|\bar{\boldsymbol{P}}^{-\frac{1}{2}}\big\|_2^2\|\nabla_{\boldsymbol{p}}\ell\big\|_2 + \big\|\nabla^2_{\boldsymbol{p}}\ell\big\|_F\right) \\
\leq& \kappa\sqrt{\frac{\sigma_+}{\sigma_-}}\left(\frac{1}{\sigma_-}C_1 + C_2\right), \quad \text{and}
\end{aligned} \tag{40a}$$

$$\begin{aligned}
\left\|\frac{\partial\nabla_{\boldsymbol{c}}\ell}{\partial\boldsymbol{p}}\right\|_F \leq& \lambda^{-1}\big\|\boldsymbol{I}-\boldsymbol{\Pi}_{\boldsymbol{E}}^\top\big\|_2\big\|\boldsymbol{I}-\boldsymbol{\Pi}_{\boldsymbol{E}}\big\|_2\big\|\nabla_{\boldsymbol{p}}\ell\big\|_2 + \lambda^{-1}\big\|\bar{\boldsymbol{P}}(\boldsymbol{I}-\boldsymbol{\Pi}_{\boldsymbol{E}})\big\|_2\big\|\nabla^2_{\boldsymbol{p}}\ell\big\|_F \\
\leq& \lambda^{-1}\big\|\bar{\boldsymbol{P}}^{-\frac{1}{2}}\big\|_2^2\big\|\bar{\boldsymbol{P}}^{\frac{1}{2}}\big\|_2^2\big\|\nabla_{\boldsymbol{p}}\ell\big\|_2 + \lambda^{-1}\big\|\bar{\boldsymbol{P}}^{\frac{1}{2}}\big\|_2^2\big\|\nabla^2_{\boldsymbol{p}}\ell\big\|_F \\
\leq& \lambda^{-1}\sigma_+\left(\frac{1}{\sigma_-}C_1 + C_2\right),
\end{aligned} \tag{40b}$$

where the constants $\sigma_-, \sigma_+, C_1, C_2 > 0$ are as defined in Theorem 5, and where we use the identity

$$\big\|\boldsymbol{A}\operatorname{diag}(\boldsymbol{b})\big\|_F \leq \big\|\boldsymbol{A}\big\|_2\big\|\operatorname{diag}(\boldsymbol{b})\big\|_F = \big\|\boldsymbol{A}\big\|_2\big\|\boldsymbol{b}\big\|_2. \tag{41}$$

As a direct consequence, we obtain the bounds from Eq. (13a) and Eq. (13b), since the bounded derivatives imply the Lipschitz continuity of the differentiable map $\boldsymbol{p} \mapsto \nabla_{[\boldsymbol{c};\boldsymbol{a};\boldsymbol{b}]}\ell$.

□