Sparse Activity and Sparse Connectivity in Supervised Learning

Markus Thom   MARKUS.THOM@UNI-ULM.DE
driveU / Institute of Measurement, Control and Microtechnology
Ulm University
89081 Ulm, Germany

Günther Palm   GUENTHER.PALM@UNI-ULM.DE
Institute of Neural Information Processing
Ulm University
89081 Ulm, Germany

Editor: Aapo Hyvärinen

Abstract

Sparseness is a useful regularizer for learning in a wide range of applications, in particular in neural networks. This paper proposes a model targeted at classification tasks, where sparse activity and sparse connectivity are used to enhance classification capabilities. The tool for achieving this is a sparseness-enforcing projection operator which finds the closest vector with a pre-defined sparseness for any given vector. In the theoretical part of this paper, a comprehensive theory for such a projection is developed. In conclusion, it is shown that the projection is differentiable almost everywhere and can thus be implemented as a smooth neuronal transfer function. The entire model can hence be tuned end-to-end using gradient-based methods. Experiments on the MNIST database of handwritten digits show that classification performance can be boosted by sparse activity or sparse connectivity. With a combination of both, performance can be significantly better compared to classical non-sparse approaches.

Keywords: supervised learning, sparseness projection, sparse activity, sparse connectivity

1. Introduction

Sparseness is a concept of efficiency in neural networks, and exists in two variants in that context (Laughlin and Sejnowski, 2003). The *sparse activity* property means that only a small fraction of neurons is active at any time. The *sparse connectivity* property means that each neuron is connected to only a limited number of other neurons. Both properties have been observed in mammalian brains (Hubel and Wiesel, 1959; Olshausen and Field, 2004; Mason et al., 1991; Markram et al., 1997) and have inspired a variety of machine learning algorithms. A notable result was achieved through the sparse coding model of Olshausen and Field (1996). Given small patches from images of natural scenes, the model is able to produce Gabor-like filters, resembling properties of simple cells found in mammalian primary visual cortex (Hubel and Wiesel, 1959; Vinje and Gallant, 2000). Another example is the optimal brain damage method of LeCun et al. (1990), which can be used to prune synaptic connections in a neural network, making connectivity sparse. Although only a small fraction of possible connections remains after pruning, this is sufficient to achieve equivalent classification results. Since then, numerous approaches on how to measure sparseness have been proposed, see Hurley and Rickard (2009) for an overview, and how to achieve sparse solutions of classical machine learning problems.

The L_0 pseudo-norm is a natural sparseness measure. Its computation consists of counting the number of non-vanishing entries in a vector. Using it rather than other sparseness measures has been shown to induce biologically more plausible properties (Rehn and Sommer, 2007). However, finding of optimal solutions subject to the L_0 pseudo-norm turns out to be NP-hard (Natarajan, 1995; Weston et al., 2003). Analytical properties of this counting measure are very poor, for it is non-continuous, rendering the localization of approximate solutions difficult. The Manhattan norm of a vector is a convex relaxation of the L_0 pseudo-norm (Donoho, 2006), and has been employed in a vast range of applications. This sparseness measure has the significant disadvantage of not being scale-invariant, so that an intuitive notion of sparseness cannot be derived from it.

1.1 Hoyer's Normalized Sparseness Measure

A normalized sparseness measure σ based on the ratio of the L_1 or Manhattan norm and the L_2 or Euclidean norm of a vector has been proposed by Hoyer (2004),

$$\sigma\colon \mathbb{R}^n \setminus \{0\} \to [0,\ 1], \qquad x \mapsto \frac{\sqrt{n} - \|x\|_1/\|x\|_2}{\sqrt{n} - 1},$$

where higher values indicate more sparse vectors. σ is well-defined because $\|x\|_2 \leq \|x\|_1 \leq \sqrt{n}\,\|x\|_2$ holds for all $x \in \mathbb{R}^n$ (Laub, 2004). As $\sigma(\alpha x) = \sigma(x)$ for all $\alpha \neq 0$ and all $x \in \mathbb{R}^n \setminus \{0\}$, σ is also scale-invariant. As composition of differentiable functions, σ is differentiable on its entire domain.

This sparseness measure fulfills all criteria of Hurley and Rickard (2009) except for Dalton's fourth law, which states that the sparseness of a vector should be identical to the sparseness of the vector resulting from multiple concatenation of the original vector. This property, however, is not crucial for a proper sparseness measure. For example, sparseness of connectivity in a biological brain increases quickly with its volume, so that connectivity in a human brain is about 170 times more sparse than in a rat brain (Karbowski, 2003). It follows that σ features all desirable properties of a proper sparseness measure.

A sparseness-enforcing projection operator, suitable for projected gradient descent algorithms, was proposed by Hoyer (2004) for optimization with respect to σ. For a pre-defined target degree of sparseness $\sigma^* \in (0,\ 1)$, the operator finds the closest vector of a given scale that has sparseness σ^* given an arbitrary vector. This can be expressed formally as Euclidean projection onto parameterizations of the sets

$$S^{(\lambda_1, \lambda_2)} := \left\{\, s \in \mathbb{R}^n \mid \|s\|_1 = \lambda_1 \text{ and } \|s\|_2 = \lambda_2 \,\right\} \text{ and } S_{\geq 0}^{(\lambda_1, \lambda_2)} := S^{(\lambda_1, \lambda_2)} \cap \mathbb{R}_{\geq 0}^n.$$

The first set is for achieving unrestricted projections, whereas the latter set is useful in situations where only non-negative solutions are feasible, for example in non-negative matrix factorization problems. The constants $\lambda_1, \lambda_2 > 0$ are target norms and can be chosen such that all points in these sets achieve a sparseness of σ^*. For example, if λ_2 was set to unity for yielding normalized projections, then λ_1 can be easily derived from the definition of σ.

Hoyer's original algorithm for computation of such a projection is an alternating projection onto a hyperplane representing the L_1 norm constraint, a hypersphere representing the L_2 norm constraint, and the non-negative orthant. A slightly modified version of this algorithm has been proved to be correct by Theis et al. (2005) in the special case when exactly one negative entry emerges that is zeroed out in the orthant projection. However, there is still no mathematically satisfactory proof for the general case.

1.2 Contributions of this Paper

This paper improves upon previous work in the following ways. Section 2 proposes a simple algorithm for carrying out sparseness-enforcing projections with respect to Hoyer's sparseness measure. Further, an improved algorithm is proposed and compared with Hoyer's original algorithm. Because the projection itself is differentiable, it is the ideal tool for achieving sparseness in gradient-based learning. This is exploited in Section 3, where the sparseness projection is used to obtain a classifier that features both sparse activity and sparse connectivity in a natural way. The benefit of these two key properties is demonstrated on a real-world classification problem, proving that sparseness acts as regularizer and improves classification results. The final sections give an overview of related concepts and conclude this paper.

On the theoretical side, a first rigorous and mathematically satisfactory analysis of the properties of the sparseness-enforcing projection is provided. This is lengthy and technical and therefore deferred into several appendixes. Appendix A fixes the notation and gives an introduction to general projections. In Appendix B, certain symmetries of subsets of the Euclidean space and their effect on projections onto such sets is studied. The problem of finding projections onto sets where Hoyer's sparseness measure attains a constant value is addressed in Appendix C. Ultimately, the algorithms proposed in Section 2 are proved to be correct. Appendix D investigates analytical properties of the sparseness projection and concludes with an efficient algorithm that computes its gradient. The gradients for optimization of the parameters of the architecture proposed in Section 3 are collected in the final Appendix E.

2. Algorithms for the Sparseness-Enforcing Projection Operator

The projection onto a set is a fundamental concept, for example see Deutsch (2001):

Definition 1 *Let $x \in \mathbb{R}^n$ and $\varnothing \neq M \subseteq \mathbb{R}^n$. Then every point in*

$$\mathrm{proj}_M(x) := \{ y \in M \mid \|y - x\|_2 \leq \|z - x\|_2 \text{ for all } z \in M \}$$

is called Euclidean projection *of x onto M. When there is exactly one point y in* $\mathrm{proj}_M(x)$, *then* $y = \mathrm{proj}_M(x)$ *is used as an abbreviation.*

Because $\mathbb{R}^n$ is finite-dimensional, $\mathrm{proj}_M(x)$ is nonempty for all $x \in \mathbb{R}^n$ if and only if M is closed, and $\mathrm{proj}_M(x)$ is a singleton for all $x \in \mathbb{R}^n$ if and only if M is closed and convex (Deutsch, 2001). In the literature, the elements from $\mathrm{proj}_M(x)$ are also called *best approximations* to x from M.

Projections onto sets that fulfill certain symmetries are of special interest in this paper and are formalized and discussed in Appendix B in greater detail. It is notable that projections onto a *permutation-invariant* set M, that is a set where membership is stable upon coordinate permutation, are *order-preserving*. This is proved in Lemma 9(a). As a consequence, when a vector is sorted in ascending or descending order, then its projection onto M is sorted accordingly. If M is *reflection-invariant*, that is when the signs of arbitrary coordinates can be swapped without violating membership in M, then the projection onto M is *orthant-preserving*, as shown in Lemma 9(b). This means that a point and its projection onto M are located in the same orthant. By exploiting this property, projections onto M can be yielded by recording and discarding the signs of the coordinates of the argument, projecting onto $M \cap \mathbb{R}^n_{\geq 0}$, and finally restoring the signs of the coordinates of the result using the signs of the argument. This is formalized in Lemma 11.

As an example for these concepts, consider the set $Z := \{x \in \mathbb{R}^n \mid \|x\|_0 = \kappa\}$ of all vectors with exactly $\kappa \in \mathbb{N}$ non-vanishing entries. Z is clearly both permutation-invariant and reflection-invariant. Therefore, the projection with respect to an L_0 pseudo-norm constraint must be both order-preserving and orthant-preserving. In fact, the projection onto Z consists simply of zeroing out all entries but the κ that are greatest in absolute value (Blumensath and Davies, 2009). This trivially fulfills the aforementioned properties of order-preservation and orthant-preservation.

Permutation-invariance and reflection-invariance are closed under intersection and union operations. Therefore, the unrestricted target set $S^{(\lambda_1, \lambda_2)}$ for the σ projection is permutation-invariant and reflection-invariant. It is hence enough to handle projections onto $S^{(\lambda_1, \lambda_2)}_{\geq 0}$ in the first place, as projections onto the unrestricted target set can easily be recovered.

In the remainder of this section, let $n \in \mathbb{N}$ be the problem dimensionality and let $\lambda_1, \lambda_2 > 0$ be the fixed target norms, which must fulfill $\lambda_2 \leq \lambda_1 \leq \sqrt{n}\lambda_2$ to avoid the existence of only trivial solutions. In the applications of the sparseness projection in this paper, λ_2 is always set to unity to achieve normalized projections, and λ_1 is adjusted as explained in Section 1.1 to achieve the target degree of sparseness σ^*. The related problem of finding the best approximation to a point x regardless of the concrete scaling, that is computing projections onto $\{s \in \mathbb{R}^n \setminus \{0\} \mid \sigma(s) = \sigma^*\}$, can be solved by projecting x onto $S^{(\lambda_1, \lambda_2)}$ and rescaling the result p such as to minimize $\|x - \alpha p\|_2$ under variation of $\alpha \in \mathbb{R}$, which yields $\alpha = \langle x, p \rangle / \|p\|_2^2$. This method is justified theoretically by Remark 5.

2.1 Alternating Projections

First note that the target set can be written as an intersection of simpler sets. Let $e_1, \ldots, e_n \in \mathbb{R}^n$ be the canonical basis of the n-dimensional Euclidean space $\mathbb{R}^n$. Further, let $e := \sum_{i=1}^n e_i \in \mathbb{R}^n$ be the vector where all entries are identical to unity. Then $H := \{a \in \mathbb{R}^n \mid e^T a = \lambda_1\}$ denotes the target hyperplane where the coordinates of all points sum up to λ_1. In the non-negative orthant $\mathbb{R}^n_{\geq 0}$, this is equivalent to the L_1 norm constraint. Further, define $K := \{q \in \mathbb{R}^n \mid \|q\|_2 = \lambda_2\}$ as the target hypersphere of all points satisfying the L_2 norm constraint. This yields the following factorization:

$$S^{(\lambda_1, \lambda_2)}_{\geq 0} = \mathbb{R}^n_{\geq 0} \cap H \cap K =: D.$$

For computation of projections onto an intersection of a finite number of closed and convex sets, it is enough to perform alternating projections onto the members of the intersection (Deutsch, 2001). As K is clearly non-convex, this general approach has to be altered to work in this specific setup.

First, consider $L := H \cap K$, which denotes the intersection of the L_1 norm target hyperplane and the L_2 norm target hypersphere. L essentially possesses the structure of a hypercircle, that is, all points in L lie also in H and there is a central point $m \in H$ and a real number $\rho \geq 0$ such that all points in L have squared distance ρ from m. It will be shown in Appendix C that $m = \lambda_1/n \cdot e \in \mathbb{R}^n$ and $\rho = \lambda_2^2 - \lambda_1^2/n$. The intersection of the non-negative orthant with the L_1 norm hyperplane, $C := \mathbb{R}^n_{\geq 0} \cap H$, is a scaled canonical simplex. Its barycenter coincides with the barycenter m of L. Finally, for an index set $I \subseteq \{1, \ldots, n\}$ let $L_I := \{a \in L \mid a_i = 0 \text{ for all } i \notin I\}$ denote the subset of points from L, where all coordinates with index not in I vanish. Its barycenter is given by $m_I = \lambda_1/d \cdot \sum_{i \in I} e_i \in \mathbb{R}^n$. With these preparations, a simple algorithm can be proposed; it computes the sparseness-enforcing projection with respect to a constraint induced by Hoyer's sparseness measure σ.

Theorem 2 *For every $x \in \mathbb{R}^n$, Algorithm 1 computes an element from $\mathrm{proj}_D(x)$. If $r \neq m$ after line 1 and $r \neq m_I$ after line 4 in all iterations, then $\mathrm{proj}_D(x)$ is a singleton.*

Algorithm 1: Proposed algorithm for computing the sparseness-enforcing projection operator for Hoyer's sparseness measure σ.

Input: $x \in \mathbb{R}^n$ and $\lambda_1, \lambda_2 \in \mathbb{R}_{>0}$ with $\lambda_2 \leq \lambda_1 \leq \sqrt{n}\lambda_2$.
Output: $s \in \text{proj}_D(x)$ where $D = S_{\geq 0}^{(\lambda_1, \lambda_2)}$.

// Project onto target hyperplane `H` and target hypercircle `L`.
1 $r := \text{proj}_H(x)$;
2 $s \in \text{proj}_L(r)$;
   // Perform alternating projections until feasible solution is found.
3 **while** $s \notin \mathbb{R}_{\geq 0}^n$ **do**
       // Project onto scaled canonical simplex `C`.
4   $\quad r := \text{proj}_C(s)$;
       // Project onto `L` keeping already vanished coordinates at zero.
5   $\quad s \in \text{proj}_{L_I}(r)$ where $I := \{ i \in \{1, \ldots, n\} \mid r_i \neq 0 \}$;
6 **end**

As already pointed out, the idea of Algorithm 1 is that projections onto D can be computed by alternating projections onto the geometric structures just defined. The rigorous proof of correctness from Appendix C proceeds by showing that the set of solutions is not tampered by projection onto the intermediate structures H, C, L and L_I. Because of the non-convexity of L and L_I, the relation between these sets and the simplex C is non-trivial and needs long arguments to be described further, see especially Lemma 26 and Corollary 27.

The projection onto the hyperplane H is straightforward and discussed in Section C.1.1. As L is essentially a hypersphere embedded in a subspace H of $\mathbb{R}^n$, projections of points from H onto L are achieved by shifting and scaling, see Section C.1.2. The alternating projection onto H and L in the beginning of Algorithm 1 make the result of the projection onto D invariant to positive scaling and arbitrary shifting of the argument, as shown in Corollary 19. This is especially useful in practice, alleviating the need for certain pre-processing methods. The formula for projections onto L can be generalized for projections onto L_I for an index set $I \subseteq \{1, \ldots, n\}$, by keeping already vanished coordinates at zero, see Section C.3.

Projections onto the simplex C are more involved and discussed at length in Section C.2. The most relevant result is that if $x \in \mathbb{R}^n \setminus C$, then there exists a separator $\hat{t} \in \mathbb{R}$ such that $p := \text{proj}_C(x) = \max(x - \hat{t} \cdot e, 0)$, where the maximum is taken element-wise (Chen and Ye, 2011). In the cases considered in this paper it is always $\hat{t} \geq 0$ as shown in Lemma 28. This implies that all entries in x that are less than $\hat{t}$ do not survive the projection, and hence the L_0 pseudo-norm of x is strictly greater than that of p. The simplex projection therefore enhances sparseness.

The separator $\hat{t}$ and the number of nonzero entries in the projection onto C can be computed with Algorithm 2, which is an adapted version of the algorithm of Chen and Ye (2011). In line 1, S_n denotes the symmetric group and P_τ denotes the permutation matrix associated with a permutation $\tau \in S_n$. The algorithm works by sorting its argument x and then determining $\hat{t}$ as the mean value of the largest entries of x minus the target L_1 norm λ_1. The number of relevant entries for computation of $\hat{t}$ is equal to the L_0 pseudo-norm of the projection and is found by trying all feasible values, starting with the largest ones. The computational complexity of Algorithm 2 is dominated by sorting the input vector and is thus quasilinear.

Algorithm 2: Computation of information for performing projections onto C, which is a scaled canonical simplex. This is an adapted version of the algorithm of Chen and Ye (2011).

Input: $x \in \mathbb{R}^n \setminus C$ and $\lambda_1 \in \mathbb{R}_{>0}$.
Output: $(\hat{t}, d) \in \mathbb{R} \times \mathbb{N}$ such that $\text{proj}_C(x) = \max(x - \hat{t} \cdot e, 0)$ and $\|\text{proj}_C(x)\|_0 = d$.

```
// Sort the input vector in descending order.
```
1 Let $\tau \in S_n$ such that $x_{\tau(1)} \geq \cdots \geq x_{\tau(n)}$ and $y := P_\tau x \in \mathbb{R}^n$;

```
// Find the only feasible separator t̂.
```
2 $s := 0$;
3 **for** $i := 1$ **to** $n-1$ **do**
4 $\quad$ $s := s + y_i$; $\; t := \frac{s - \lambda_1}{i}$;
5 $\quad$ **if** $t \geq y_{i+1}$ **then return** (t, i);
6 **end**
7 $s := s + y_n$; $\; t := \frac{s - \lambda_1}{n}$; **return** (t, n);

Algorithm 3: Explicit and optimized variant of Algorithm 1.

Input: $x \in \mathbb{R}^n$ and $\lambda_1, \lambda_2 \in \mathbb{R}_{>0}$ with $\lambda_2 \leq \lambda_1 \leq \sqrt{n}\lambda_2$.
Output: $s \in \text{proj}_D(x)$ where $D = S_{\geq 0}^{(\lambda_1, \lambda_2)}$.

1 **procedure** `proj_L`$(y \in \mathbb{R}^d)$
2 $\quad$ $\rho := \lambda_2^2 - \lambda_1^2/d$; $\hspace{2.5em}$ `// Compute squared radius of ` L_I ` (Lemma 15).`
3 $\quad$ $\varphi := \sum_{i=1}^d y_i^2 - \lambda_1^2/d$; $\hspace{2.5em}$ `// ` $\varphi := \|y - m_I\|_2^2$ ` (Remark 14).`
4 $\quad$ **if** $\varphi = 0$ **then**
5 $\quad\quad$ $(y_1, \ldots, y_{d-1})^T := \lambda_1/d + \sqrt{\rho}/\sqrt{d(d-1)}$; $\hspace{1em}$ `// ` y ` equals the barycenter of ` L_I`,`
6 $\quad\quad$ $y_d := \lambda_1/d - \sqrt{\rho(d-1)}/\sqrt{d}$; $\hspace{1em}$ `// pick a sorted projection (Remark 18).`
7 $\quad$ **else** $y := \lambda_1/d \cdot e + \sqrt{\rho/\varphi} \cdot (y - \lambda_1/d \cdot e)$; $\hspace{0.5em}$ `// Pick unique projection (Lemma 17).`
8 **end**

```
// Beginning of main body.
```
9 Let $\tau \in S_n$ such that $x_{\tau(1)} \geq \cdots \geq x_{\tau(n)}$ and $y := P_\tau x \in \mathbb{R}^n$; $\hspace{1em}$ `// Sort the input vector.`
10 $y := y + 1/n \cdot (\lambda_1 - \sum_{i=1}^n y_i) e$; $\hspace{2.5em}$ `// Project onto ` H ` (Lemma 13).`
11 `proj_L`$(y_1, \ldots, y_n)$; $\hspace{2.5em}$ `// Project in-place onto ` L`.`

```
// Perform alternating projections until feasible solution is found.
```
12 $d := n$; $\hspace{2.5em}$ `// Store current number of relevant entries of ` y`.`
13 **while** $(y_1, \ldots, y_d)^T \notin \mathbb{R}_{\geq 0}^d$ **do**
14 $\quad$ $(\hat{t}, d) := $ `proj_C`$(y_1, \ldots, y_d)$; $\hspace{1em}$ `// This is carried out by Algorithm 2.`
15 $\quad$ $(y_1, \ldots, y_d)^T := (y_1, \ldots, y_d)^T - \hat{t}$; $\hspace{1em}$ `// Project onto ` C ` (Proposition 24).`
16 $\quad$ `proj_L`$(y_1, \ldots, y_d)$; $\hspace{1em}$ `// Project onto ` L_I ` where ` $I = \{1, \ldots, d\}$ ` (Lemma 30).`
17 **end**

```
// Undo sorting permutation and set remaining entries to zero.
```
18 $s \in \{0\}^n$; **for** $i := 1$ **to** d **do** $s_{\tau(i)} := y_i$;

2.2 Optimized Variant

Because of the permutation-invariance of the sets involved in the projections, it is enough to sort the vector that is to be projected onto D once. This guarantees that the working vector that emerges from subsequent projections is sorted also. No additional sorting has then to be carried out when using Algorithm 2 for projections onto C. This additionally has the side effect that the non-vanishing entries of the working vector are always concentrated in its first entries. Hence all relevant information can always be stored in a small unit-stride array, to which access is more efficient than to a large sparse array. Further, the index set I of non-vanishing entries in the working vector is always of the form $I = \{1, \ldots, d\}$, where d is the number of nonzero entries.

Algorithm 3 is a variant of Algorithm 1 where these optimizations were applied, and where the explicit formulas for the intermediate projections were used. The following result, which is proved in Appendix C, states that both algorithms always compute the same result:

Theorem 3 *Algorithm 1 is equivalent to Algorithm 3.*

Projections onto C increase the amount of vanishing entries in the working vector, which is of finite dimension n. Hence, at most n alternating projections are carried out, and the algorithm terminates in finite time. Further, the complexity of each iteration is at most linear in the L_0 pseudo-norm of the working vector. The theoretic overall computational complexity is thus at most quadratic in problem dimensionality n.

2.3 Comparison with Hoyer's Original Algorithm

The original algorithm for the sparseness-enforcing projection operator proposed by Hoyer (2004) is hard to understand, and correctness has been proved by Theis et al. (2005) in a special case only. A simple alternative has been proposed with Algorithm 1 in this paper. Based on the symmetries induced by Hoyer's sparseness measure σ and by exploiting the projection onto a simplex, an improved method was given in Algorithm 3.

The improved algorithm proposed in this paper always requires at most the same number of iterations of alternating projections as the original algorithm. The original algorithm uses a projection onto the non-negative orthant $\mathbb{R}^n_{\geq 0}$ to achieve vanishing coordinates in the working vector. This operation can be written as $\mathrm{proj}_{\mathbb{R}^n_{\geq 0}}(x) = \max(x, 0)$. In the improved algorithm, a simplex projection is used for this purpose, expressed formally as $\mathrm{proj}_C(x) = \max(x - \hat{t} \cdot e, 0)$ with $\hat{t} \in \mathbb{R}$ chosen accordingly. Due to the theoretical results on simplex geometry from Section C.2 and their application in Lemma 28 in Section C.3, the number $\hat{t}$ is always non-negative. Therefore, at least the same amount of entries is set to zero in the simplex projection compared to the projection onto the non-negative orthant, see also Corollary 29. Hence with induction for the number of non-vanishing entries in the working vector, the number of iterations the proposed algorithm needs to terminate is bounded by the number of iterations the original method needs to terminate given the same input.

The experimental determination of an estimate of the number of iterations required was carried out as follows. Random vectors with sparseness 0.15 were sampled and their sparse projections were computed using the respective algorithms, to gain the best normalized approximations with a target sparseness degree of $\sigma^* := 0.90$. For both algorithms the very same vectors were used as input. During the run-time of the algorithms, the number of iterations that were necessary to compute the result were counted. Additionally, the number of nonzero entries in the working vector was
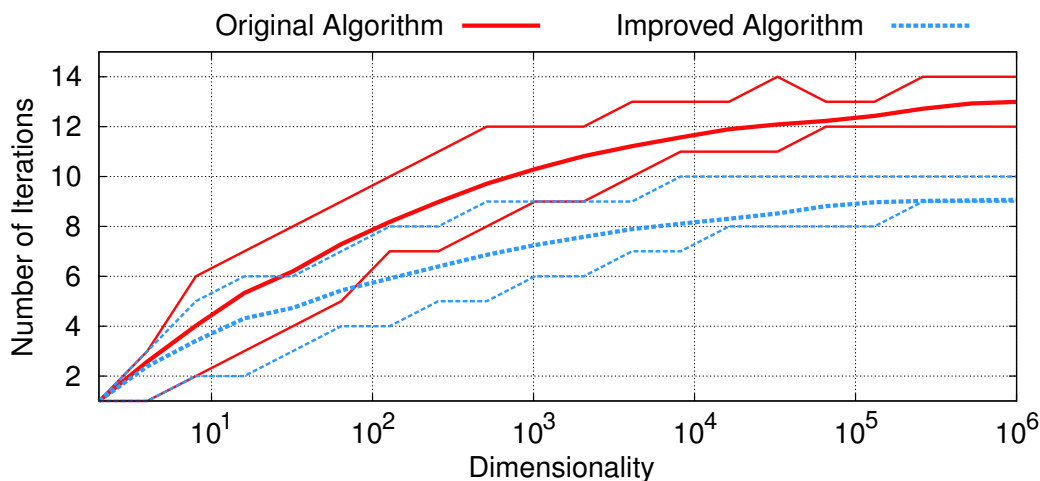
Figure 1: Comparison of the number of iterations of the original algorithm for the projection onto D with the improved version as proposed in this paper. The sparseness-enforcing projection with target sparseness 0.90 was carried out for input vectors of sparseness 0.15. The thick lines indicate the mean number of iterations required for the projection, and the thin lines indicate the minimum and maximum number of iterations, respectively. Even for input vectors with a million entries, less than 14 iterations are required to find the projection. With the improved algorithm, this reduces to at most 10 iterations.

recorded in each iteration. This was done for different dimensionalities, and for each dimensionality 1000 vectors were sampled.

Figure 1 shows statistics on the number of iterations the algorithms needed to terminate. As was already observed by Hoyer (2004), the number of required iterations grows very slowly with problem dimensionality. For $n = 10^6$, only between 12 and 14 iterations were needed with the original algorithm to compute the result. With Algorithm 3, this can be improved to requiring 9 to 10 iterations, which amounts to roughly 30% less iterations. Due to the small slope in the number of required iterations, it can be conjectured that this quantity is at most logarithmic in problem dimensionality n. If this applies, the complexity of Algorithm 3 is at most quasilinear. Because the input vector is sorted in the beginning, it is also not possible to fall below this complexity class.

The progress of working dimensionality reduction for problem dimensionality $n = 1000$ is depicted in Figure 2, averaged over the 1000 input vectors from the experiment. After the first iteration, that is after projecting onto H and L, the working dimensionality still matches the input dimensionality. Starting with the second iteration, dimensions are discarded by projecting onto $\mathbb{R}^n_{\geq 0}$ in the original algorithm and onto C in the improved variant, which yields vanishing entries in the working vectors. With the original algorithm, in the mean 54% of all entries are nonzero after the second iteration, while with the improved algorithm only 27% of the original 1000 dimensions remain in the mean. This trend continues in subsequent iterations such that the final working dimensionality is reached more quickly with the algorithm proposed in this paper. Although using Algorithm 2 to perform the simplex projection is more expensive than just setting negative entries to zero in the orthant projection, the overhead quickly amortizes because of the boost in dimensionality reduction.
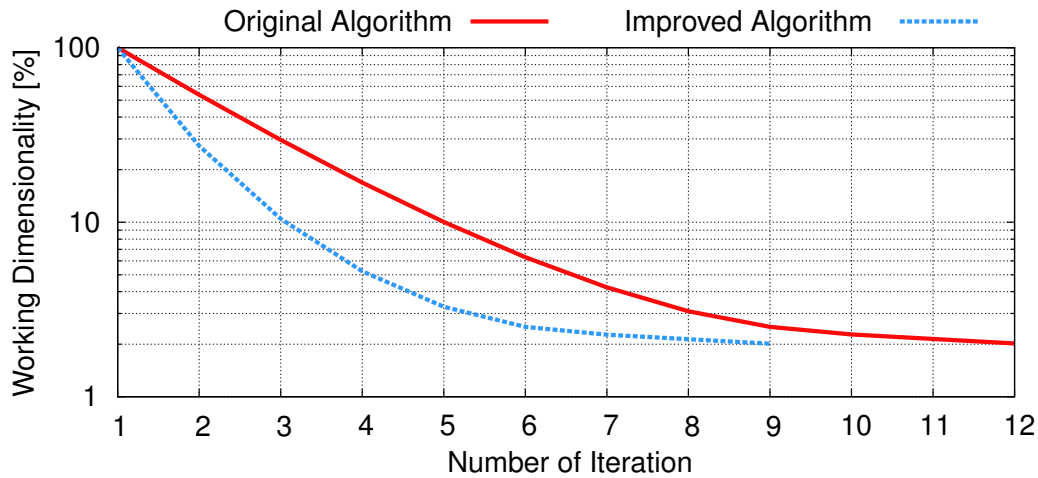
Figure 2: Comparison of the number of non-vanishing entries in the working vectors of the original algorithm and the improved algorithm during run-time. The algorithms were run with input vectors of dimensionality 1000 and initial sparseness 0.15 to compute projections with sparseness 0.90. Standard deviations were always less than 1%; they were omitted in the plot to avoid clutter. The algorithm proposed in this paper reduces dimensionality more quickly and terminates earlier than the original algorithm.
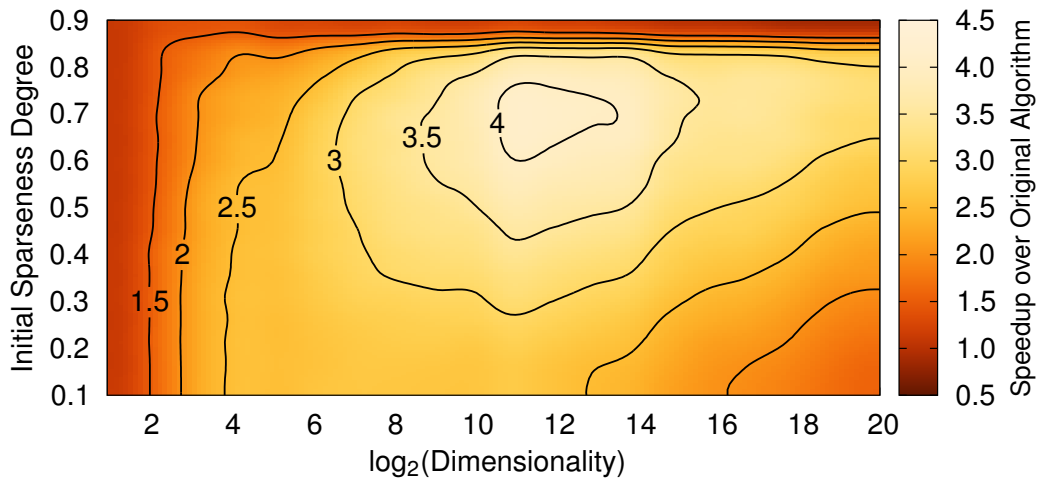

Figure 3: Ratio of the computation time of the original algorithm and the improved algorithm for a variety of input dimensionality and initial vector sparseness. Numbers greater than one indicate parameterizations where the proposed algorithm is more efficient than the original one. There is a large region where the speedup is decent.

For determination of the relative speedup incorporated with both the simplex projection and the access to unit-stride arrays due to the permutation-invariance, both algorithms were implemented as C++ programs using an optimized implementation of the BLAS library for carrying out the vector operations. The employed processor was an Intel Core i7-990X. For a range of different dimensionalities, a set of vectors with varying initial sparseness were sampled. The number of the vectors for every pair of dimensionality and initial sparseness was chosen such that the processing time of the algorithms was several orders of magnitudes greater than the latency time of the operation system. Then the absolute time needed for the algorithms to compute the projections with a target sparseness of 0.90 were measured, and their ratio was taken to compute the relative speedup. The results of this experiment are depicted in Figure 3. It is evident that the maximum speedup is achieved for vectors with a dimensionality between 2^9 and 2^{15}, and an initial sparseness greater than 0.40. For low initial sparseness, as is achieved by randomly sampled vectors, a speedup of about 2.5 can be achieved for a broad spectrum of dimensionality between 2^4 and 2^{13}.

The improvements to the original algorithm are thus not only theoretical, but also noticeable in practice. The speedup is especially useful when the projection is used as a neuronal transfer function in a classifier as proposed in Section 3, because then the computational complexity of the prediction of class membership of unknown samples can be reduced.

2.4 Function Definition and Differentiability

It is clear from Theorem 2 that the projection onto D is unique almost everywhere. Therefore the set $R := \{ x \in \mathbb{R}^n \mid |\mathrm{proj}_D(x)| \neq 1 \}$ is a null set. However, $R \neq \varnothing$ as for example the projection is not unique for vectors where all entries are identical. In other words, for $x := \xi e \in \mathbb{R}^n$ for some $\xi \in \mathbb{R}$ follows $\mathrm{proj}_H(x) = m$ and $\mathrm{proj}_L(m) = L$. If $n = 2$ a possible solution is given by $(\alpha, \beta)^T \in \mathrm{proj}_D(x)$ with α and β given as stated in Remark 18, as in this case α and β are positive. Additionally, another solution is given by $(\beta, \alpha)^T \in \mathrm{proj}_D(x)$ which is unequal to the other solution because of $\alpha \neq \beta$. A similar argument can be used to show non-uniqueness for all $n \geq 2$. As R is merely a small set, non-uniqueness is not an issue in practical applications.

The sparseness-enforcing projection operator that is restricted to non-negative solutions can thus be cast almost everywhere as a function

$$\pi_{\geq 0} \colon \mathbb{R}^n \setminus R \to D, \qquad x \mapsto \mathrm{proj}_D(x).$$

Exploiting reflection-invariance implies that the unrestricted variant of the projection

$$\pi \colon \mathbb{R}^n \setminus R \to S^{(\lambda_1, \lambda_2)}, \qquad x \mapsto s \circ \pi_{\geq 0}\left(|x|\right),$$

is well-defined, where $s \in \{\pm 1\}^n$ is given as described in Lemma 11. Note that computation of $\pi_{\geq 0}$ is a crucial prerequisite to computation of the unrestricted variant π. It will be used exclusively in Section 3 because non-negativity is not necessary in the application proposed there.

If π or $\pi_{\geq 0}$ is employed in an objective function that is to be optimized, the information whether these functions are differentiable is crucial for selecting an optimization strategy. As an example, consider once more projections onto $Z := \{ x \in \mathbb{R}^n \mid \|x\|_0 = \kappa \}$ where $\kappa \in \mathbb{N}$ is a constant. It was already mentioned in Section 2 that the projection onto Z consists simply of zeroing out the elements that are smallest in absolute value. Let $x \in \mathbb{R}^n$ be a point and let $\tau \in S_n$ be a permutation such that $\left|x_{\tau(1)}\right| \geq \cdots \geq \left|x_{\tau(n)}\right|$. Clearly, if $\left|x_{\tau(\kappa)}\right| \neq \left|x_{\tau(\kappa+1)}\right|$ then $\mathrm{proj}_Z(x) = y$ where $y_i = x_i$ for $i \in \{\tau(1), \ldots, \tau(\kappa)\}$ and $y_i = 0$ for $i \in \{\tau(\kappa+1), \ldots, \tau(n)\}$. Moreover, when $\left|x_{\tau(\kappa)}\right| \neq \left|x_{\tau(\kappa+1)}\right|$ then

there exists a neighborhood U of x such that $\text{proj}_Z(s) = \sum_{i=1}^{\kappa} s_{\tau(i)} e_{\tau(i)}$ for all $s \in U$. With this closed-form expression, $s \mapsto \text{proj}_Z(s)$ is differentiable in x with gradient $\partial \text{proj}_Z(x)/\partial x = \text{diag}\left(\sum_{i=1}^{\kappa} e_{\tau(i)}\right)$, that is the identity matrix where the entries on the diagonal belonging to small absolute values of x have been zeroed out. If the requirement on x is not fulfilled, then a small distortion of x is sufficient to find a point in which the projection onto Z is differentiable.

In contrast to the L_0 projection, differentiability of π and $\pi_{\geq 0}$ is non-trivial. A full-length discussion is given in Appendix D, and concludes that both π and $\pi_{\geq 0}$ are differentiable almost everywhere. It is more efficient when only the product of the gradient with an arbitrary vector needs to be computed, see Corollary 36. Such an expression emerges in a natural way by application of the chain rule to an objective function where the sparseness-enforcing projection is used. In practice this weaker form is thus mostly no restriction and preferable for efficiency reasons over the more general complete gradient as given in Theorem 35.

The derivative of $\pi_{\geq 0}$ is obtained by exploiting the structure of Algorithm 1. Because the projection onto D is essentially a composition of projections onto H, C, L and L_I, the overall gradient can be computed using the chain rule. The gradients of the intermediate projections are simple expressions and can be combined to yield one matrix for each iteration of alternating projections. Since these iteration gradients are basically sums of dyadic products, their product with an arbitrary vector can be computed by primitive vector operations. With matrix product associativity, this process can be repeated to efficiently compute the product of the gradient of $\pi_{\geq 0}$ with an arbitrary vector. For this, it is sufficient to record some intermediate quantities during execution of Algorithm 3, which does not add any major overhead to the algorithm itself. The gradient of the unrestricted variant π can be deduced in a straightforward way from the gradient of $\pi_{\geq 0}$ because of their close relationship.

3. Sparse Activity and Sparse Connectivity in Supervised Learning

The sparseness-enforcing projection operator can be cast almost everywhere as vector-valued function π, which is differentiable almost everywhere, see Section 2.4. This section proposes a hybrid of an auto-encoder network and a two-layer neural network, where the sparseness projection is employed as a neuronal transfer function. The proposed model is called supervised online auto-encoder (SOAE) and is intended for classification by means of a neural network that features sparse activity and sparse connectivity. Because of the analytical properties of the sparseness-enforcing projection operator, the model can be optimized end-to-end using gradient-based methods.

3.1 Architecture

Figure 4 depicts the data flow in the proposed model. There is one module for reconstruction capabilities and one module for classification capabilities. The *reconstruction module*, depicted on the left of Figure 4, operates by converting an input sample $x \in \mathbb{R}^d$ into an *internal representation* $h \in \mathbb{R}^n$, and then computing an approximation $\tilde{x} \in \mathbb{R}^d$ to the original input sample. In doing so, the product $u \in \mathbb{R}^n$ of the input sample with a *matrix of bases* $W \in \mathbb{R}^{d \times n}$ is computed, and a transfer function $f \colon \mathbb{R}^n \to \mathbb{R}^n$ is applied. For sparse activity, f can be chosen to be the sparseness-enforcing projection operator π or the projection with respect to the L_0 pseudo-norm. This guarantees that the internal representation is sparsely populated and close to u. The reconstruction is achieved like in a linear generative model, by multiplication of the matrix of bases with the internal representation. Hence the same matrix W is used for both encoding and decoding, rendering the reconstruction module symmetric, or in other words with tied weights. This approach is similar to principal com-
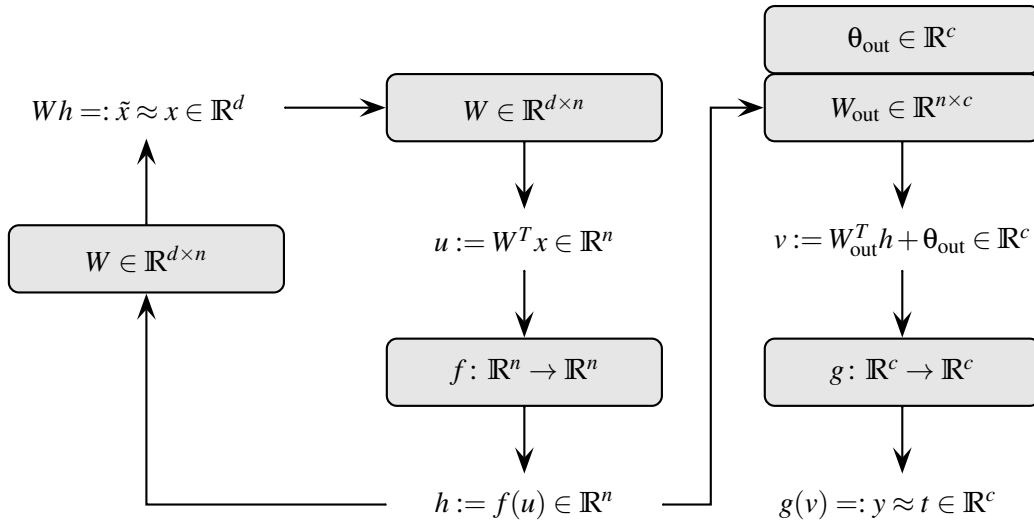
$$Wh =: \tilde{x} \approx x \in \mathbb{R}^d \longrightarrow \boxed{W \in \mathbb{R}^{d \times n}}$$

$$\boxed{W \in \mathbb{R}^{d \times n}}$$

$$\boxed{\theta_{\text{out}} \in \mathbb{R}^c}$$

$$\boxed{W_{\text{out}} \in \mathbb{R}^{n \times c}}$$

$$u := W^T x \in \mathbb{R}^n$$

$$v := W_{\text{out}}^T h + \theta_{\text{out}} \in \mathbb{R}^c$$

$$\boxed{f : \mathbb{R}^n \to \mathbb{R}^n}$$

$$\boxed{g : \mathbb{R}^c \to \mathbb{R}^c}$$

$$h := f(u) \in \mathbb{R}^n$$

$$g(v) =: y \approx t \in \mathbb{R}^c$$

Figure 4: Architecture and data flow of supervised online auto-encoder (SOAE). The circle on the left, mapping from an input sample x to its approximation $\tilde{x}$ comprises the reconstruction module. The classification module consists of the mapping from x to the classification decision y. The matrix of bases W shall be sparsely populated to account for the sparse connectivity property. If the transfer function f is set to the sparseness projection, the internal representation h will be sparsely populated, fulfilling the sparse activity property.

ponent analysis (Hotelling, 1933), restricted Boltzmann machines for deep auto-encoder networks (Hinton et al., 2006) and to sparse encoding symmetric machine (Ranzato et al., 2008).

By enforcing W to be sparsely populated, the sparse connectivity property holds as well. More formally, the aim is that $\sigma(We_i) = \sigma_W$ holds for all $i \in \{1, \dots, n\}$, where $\sigma_W \in (0, 1)$ is the target degree of connectivity sparseness and We_i is the i-th column of W. This condition was adopted from non-negative matrix factorization with sparseness constraints (Hoyer, 2004). In the context of neural networks, the synaptic weights of individual neurons are stored in the columns of the weight matrix W. The interpretation of this formal sparseness constraint is then that each neuron is only allowed to be sparsely connected with the input layer.

The *classification module* is shown on the right-hand side of Figure 4. It computes a *classification decision* $y \in \mathbb{R}^c$ by feeding h through a one-layer neural network. The network output y is yielded through computation of the product with a matrix of weights $W_{\text{out}} \in \mathbb{R}^{n \times c}$, addition of a threshold vector $\theta_{\text{out}} \in \mathbb{R}^c$ and application of a transfer function $g : \mathbb{R}^c \to \mathbb{R}^c$. This module shares the inference of the internal representation with the reconstruction module, which can also be considered a one-layer neural network. Therefore the entire processing path from x to y forms a two-layer neural network (Rumelhart et al., 1986), where W stores the synaptic weights of the hidden layer, and W_{out} and θ_{out} are the parameters of the output layer.

The input sample x shall be approximated by $\tilde{x}$, and the target vector for classification $t \in \mathbb{R}^c$ shall be approximated by y. This is achieved by optimization of the parameters of SOAE, that is the quantities W, W_{out} and θ_{out}. The goodness of the approximation $x \approx \tilde{x}$ is estimated using

a differentiable similarity measure $s_R \colon \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$, and the approximation $y \approx t$ is assessed by another similarity measure $s_C \colon \mathbb{R}^c \times \mathbb{R}^c \to \mathbb{R}$. For minimizing the deviation in both approximations, the objective function

$$E_{\text{SOAE}}(W,\, W_{\text{out}},\, \theta_{\text{out}}) := (1 - \alpha) \cdot s_R(\tilde{x},\, x) + \alpha \cdot s_C(y,\, t)$$

shall be optimized, where $\alpha \in [0,\, 1]$ controls the trade-off between reconstruction and classification capabilities. To incorporate sparse connectivity, feasible solutions are restricted to fulfill $\sigma(We_i) = \sigma_W$ for all $i \in \{1, \ldots, n\}$. If $\alpha = 0$, then SOAE is identical to a symmetric auto-encoder network with sparse activity and sparse connectivity. In the case of $\alpha = 1$, SOAE forms a two-layer neural network for classification with a sparsely connected hidden layer and where the activity in the hidden layer is sparse. The parameter α can also be used to blend continuously between these two extremes. Note that $\tilde{x}$ only depends on W but not on W_{out} or θ_{out}, but y depends on W, W_{out} and θ_{out}. Hence W_{out} and θ_{out} are only relevant when $\alpha > 0$, whereas W is essential for all choices of α.

An appropriate choice for s_R is the correlation coefficient (see for example Rodgers and Nicewander, 1988), because it is normed to values in the interval $[-1,\, 1]$, invariant to affine-linear transformations, and differentiable. If f is set to π, then a model that is invariant to the concrete scaling and shifting of the occurring quantities can be yielded. This follows because π is also invariant to such transformations, see Corollary 19. The similarity measure for classification capabilities s_C is chosen to be the cross-entropy error function (Bishop, 1995), which was shown empirically by Simard et al. (2003) to induce better classification capabilities than the mean squared error function. The softmax transfer function (Bishop, 1995) is used as transfer function g of the output layer. It provides a natural pairing together with the cross-entropy error function (Dunne and Campbell, 1997) and supports multi-class classification.

3.2 Learning Algorithm

The proposed optimization algorithm for minimization of the objective function E_{SOAE} is projected gradient descent (Bertsekas, 1999). Here, each update to the degrees of freedom is followed by application of the sparseness projection to the columns of W to enforce sparse connectivity. There are theoretical results on the convergence of projected gradient methods when projections are carried out onto convex sets (Bertsekas, 1999), but here the target set for projection is non-convex. Nevertheless, the experiments described below show that projected gradient descent is an adequate heuristic in the situation of the SOAE framework to tune the network parameters. For completeness, the gradients of E_{SOAE} with respect to the network parameters are given in Appendix E. Update steps are carried out after every presentation of a pair of an input sample and associated target vector. This online learning procedure results in faster learning and improves generalization capabilities over batch learning (Wilson and Martinez, 2003; Bottou and LeCun, 2004).

A learning set with samples from $\mathbb{R}^d$ and associated target vectors from $\{0, 1\}^c$ as one-of-c-codes is input to the algorithm. The dimensionality of the internal representation n and the target degree of sparseness with respect to the connectivity $\sigma_W \in (0,\, 1)$ are parameters of the algorithm. Sparseness of connectivity increases for larger σ_W, as Hoyer's sparseness measure is employed in the definition of the set of feasible solutions.

Two possible choices for the hidden layer's transfer function f to achieve sparse activity were discussed in this paper. One possibility is to carry out the projection with respect to the L_0 pseudo-norm. The more sophisticated method is to use the unrestricted sparseness-enforcing projection

operator π with respect to Hoyer's sparseness measure σ, which can be carried out by Algorithm 3. In both cases, a target degree for sparse activity is a parameter of the learning algorithm. In case of the L_0 projection, this sparseness degree is denoted by $\kappa \in \{1, \ldots, n\}$, and sparseness increases with smaller values of it. For the σ projection, $\sigma_H \in (0, 1)$ is used, where larger values indicate more sparse activity.

Initialization of the columns of W is achieved by selecting a random subset of the learning set, similar to the initialization of radial basis function networks (Bishop, 1995). This ensures significant activity of the hidden layer from the very start, resulting in strong gradients and therefore reducing training time. The parameters of the output layer, that is W_{out} and θ_{out}, are initialized by sampling from a zero-mean Gaussian distribution with a standard deviation of $1/100$.

In every epoch, a randomly selected subset of samples and associated target vectors from the learning set is used for stochastic gradient descent to update W, W_{out} and θ_{out}. The results from Appendix E can be used to efficiently compute the gradient of the objective function. There, the gradient for the transfer function f only emerges as a product with a vector. The gradient for the L_0 projection is trivial and was given as an example in Section 2.4. If f is Hoyer's sparseness-enforcing projection operator, it is possible to exploit that only the product of the gradient with a vector is needed. In this case, it is more efficient to compute the result of the multiplication implicitly using Corollary 36 and thus avoid the computation of the entire gradient of π.

After every epoch, a sparseness projection is applied to the columns of W. This guarantees that $\sigma(We_i) = \sigma_W$ holds for all $i \in \{1, \ldots, n\}$, and therefore the sparse connectivity property is fulfilled. The trade-off variable α which controls the weight of the reconstruction and the classification term is adjusted according to $\alpha(\nu) := 1 - \exp(-\nu/100)$, where $\nu \in \mathbb{N}$ denotes the number of the current epoch. Thus α starts at zero, increases slowly and asymptotically reaches one. The emphasis at the beginning of the optimization is thus on reconstruction capabilities. Subsequently, classification capabilities are incorporated slowly, and in the final phase of training classification capabilities exclusively are optimized. This continuous variant of unsupervised pre-training (Hinton et al., 2006) leads to parameters in the vicinity of a good minimizer for classification capabilities before classification is preferred over reconstruction through the trade-off parameter α. Compared to the choice $\alpha \equiv 1$ this strategy helps to stabilize the trajectory in parameter space and makes the objective function values settle down more quickly, such that the termination criterion is satisfied earlier.

3.3 Description of Experiments

To assess the classification capabilities and the impact of sparse activity and sparse connectivity, the MNIST database of handwritten digits (LeCun and Cortes, 1998) was employed. It is a popular benchmark data set for classification algorithms, and numerous results with respect to this data set are reported in the literature. The database consists of 70 000 samples, divided into a learning set of 60 000 samples and an evaluation set of 10 000 samples. Each sample represents a digit of size 28×28 pixels and has a class label from $\{0, \ldots, 9\}$ associated with it. Therefore the input and output dimensionalities are $d := 28^2 = 784$ and $c := 10$, respectively. The classification error is given in percent of all 10 000 evaluation samples, hence 0.01% corresponds to a single misclassified digit.

For generation of the original data set, the placement of the digits has been achieved based on their barycenter (LeCun and Cortes, 1998). Because of sampling and rounding errors, the localization uncertainty can hence be assumed to be less than one pixel in both directions. To account for this uncertainty, the learning set was augmented by jittering each sample in each of eight possible

directions by one pixel, yielding 540 000 samples for learning in total. The evaluation set was left unchanged to yield results that can be compared to the literature. As noted by Hinton et al. (2006), the learning problem is no more permutation-invariant due to the jittering, as information on the neighborhood of the pixels is implicitly incorporated in the learning set.

However, classification results improve dramatically when such prior knowledge is used. This was demonstrated by Schölkopf (1997) using the virtual support vector method, which improved a support vector machine with polynomial kernel of degree five from an error of 1.4% to 1.0% by jittering the support vectors by one pixel in four principal directions. This result was extended by DeCoste and Schölkopf (2002), where a support vector machine with a polynomial kernel of degree nine was improved from an error of 1.22% to 0.68% by jittering in all possible eight directions. Further improvements can be achieved by generating artificial training samples using elastic distortions (Simard et al., 2003). This reduced the error of a two-layer neural network with 800 hidden units to 0.7%, compared to the 1.1% error yielded when training on samples created by affine distortions. Very big and very deep neural networks possess a large number of adaptable weights. In conjunction with elastic and affine distortions such neural networks can yield errors as low as 0.35% (Cireşan et al., 2010). The current record error of 0.23% is held by an approach that combines distorted samples with a committee of convolutional neural networks (Cireşan et al., 2012). This is an architecture that has been optimized exclusively for input data that represents images, that is where the neighborhood of the pixels is hard-wired in the classifier. To allow for a plain evaluation that does not depend on additional parameters for creating artificial samples, the jittered learning set with 540 000 samples is used throughout this paper.

The experimental methodology was as follows. The number of hidden units was chosen to be $n := 1000$ in all experiments that are described below. This is an increased number compared to the 800 hidden units employed by Simard et al. (2003), but promises to yield better results when an adequate number of learning samples is used. As all tested learning algorithms are essentially gradient descent methods, an initial step size had to be chosen. For each candidate step size, five runs of a two-fold cross validation were carried out on the learning set. Then, for each step size the median of the ten resulting classification errors was computed. The winning step size was then determined to be the one that achieved a minimum median of classification errors.

In every epoch, 21 600 samples were randomly chosen from the learning set and presented to the network. This number of samples was chosen as it is $1/25$-th of the jittered learning set. The step size was multiplicatively annealed using a factor of 0.999 after every epoch. Optimization was terminated once the relative change in the objective function became very small and no more significant progress on the learning set could be observed. The resulting classifiers were then applied to the evaluation set, and misclassifications were counted.

3.4 Experimental Results

Two variants of the supervised online auto-encoder architecture as proposed in this section were trained on the augmented learning set. In both variants, the target degree of sparse connectivity was set to $\sigma_W := 0.75$. This choice was made because 96% of all samples in the learning set possess a sparseness which is less than 0.75. Therefore, the resulting bases are forced to be truly sparsely connected compared to the sparseness of the digits.

The first variant is denoted by SOAE-σ. Here, the sparseness-enforcing projection operator π was used as transfer function f in the hidden layer. Target degrees of sparse activity σ_H with respect

to Hoyer's sparseness measure σ were chosen from the interval [0.20, 0.95] in steps of size 0.05. This variant was then trained on the jittered learning set using the method described in Section 3.2. For every value of σ_H, the resulting sparseness of activity was measured after training using the L_0 pseudo-norm. For this, each sample of the learning set was presented to the networks, and the number of active units in the hidden layer was counted. Figure 5 shows the resulting mean value and standard deviation of sparse activity. If $\sigma_H = 0.20$ is chosen, then in the mean about 800 of the total 1000 hidden units are active upon presentation of a sample from the learning set. For $\sigma_H = 0.80$ only one hundred units are active at any one time, and for $\sigma_H = 0.95$ there are only eleven active units. The standard deviation of the activity decreases when sparseness increases, hence the mapping from σ_H to the resulting number of active units becomes more accurate.

The second variant, denoted SOAE-L_0, differs from SOAE-σ in that the projection with respect to the L_0 pseudo-norm as transfer function f was used. The target sparseness of activity is given by a parameter $\kappa \in \{1, \ldots, n\}$, which controls the exact number of units that are allowed to be active at any one time. For the experiments, the values for κ were chosen to match the mean activities from the SOAE-σ experiments. This way the results of both variants can be compared based on a unified value of activity sparseness. The results are depicted in Figure 6. Usage of the σ projection consequently outperforms the L_0 projection for all sparseness degrees. Even for high sparseness of activity, that is when only about ten percent of the units are allowed to be active at any one time, good classification capabilities can be obtained with SOAE-σ. For $\kappa \in [242, 558]$, the classification results of SOAE-L_0 reach an optimum. SOAE-σ is more robust, as classification capabilities first begin to collapse when sparseness is below 5%, whereas SOAE-L_0 starts to degenerate when sparseness falls below 20%. For $\sigma_H \in [0.45, 0.85]$, roughly translating to between 5% and 50% activity, about equal classification performance is achieved using SOAE-σ.

It can thus be concluded that using the sparseness-enforcing projection operator as described in this paper yields better results than when the simple L_0 projection is used to achieve sparse activity. To assess the benefit more precisely and to investigate the effect of individual factors, several comparative experiments have been carried out. A summary of these experiments and their outcome is given in Table 1. The variants SOAE-σ and SOAE-L_0 denote the entirety of the respective experiments where sparseness of activity lies in the intervals described above, that is $\sigma_H \in [0.45, 0.85]$ and $\kappa \in [242, 558]$, respectively. Using these intervals, SOAE-σ and SOAE-L_0 achieved a median error of 0.75% and 0.82% on the evaluation set, respectively. Variant SOAE-σ-conn is essentially equal to SOAE-σ, except for sparse connectivity not being incorporated. Sparseness of activity here was also chosen to be $\sigma_H \in [0.45, 0.85]$, which resulted in about equal classification results over the entire range. Dropping of sparse connectivity increases misclassifications, for the median error of SOAE-σ-conn is 0.81% and thereby greater than the median error of SOAE-σ.

The other five approaches included in the comparison are multi-layer perceptrons (MLPs) with the same topology and dynamics as the classification module of supervised online auto-encoder, with two exceptions. First, the transfer function of the hidden layer f was set to a hyperbolic tangent, thus not including explicit sparse activity. Second, in all but one experiment sparse connectivity was either not incorporated, or achieved through other means than by performing a σ projection after each learning epoch. Besides the variation in sparseness of connectivity, the experiments differ in the initialization of the network parameters.

For each variant, 55 runs were carried out and the resulting classifiers were applied to the evaluation set to compute the classification error. Then, the best four and the worst four results were discarded and not included in further analysis. Hence a random sample of size 47 was achieved,
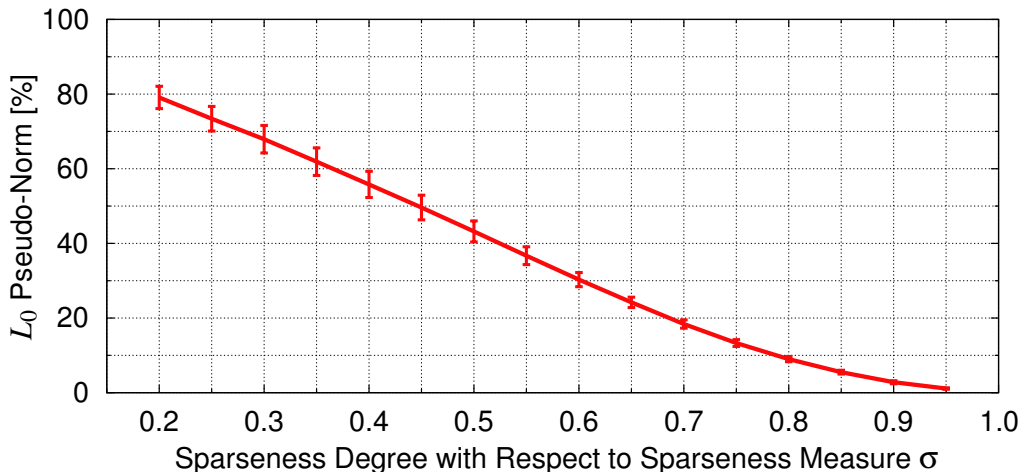
Figure 5: Resulting amount of nonzero entries in an internal representation h with 1000 entries, depending on the target degree of sparseness for activity σ_H with respect to Hoyer's sparseness measure σ. For low values of σ_H, about 80% of the entries are nonzero, whereas for very high sparseness degrees only 1% of the entries do not vanish. The error bars indicate $\pm$ one standard deviation distance from the mean value. Standard deviation shrinks with increasing sparseness degree, making the mapping more accurate.
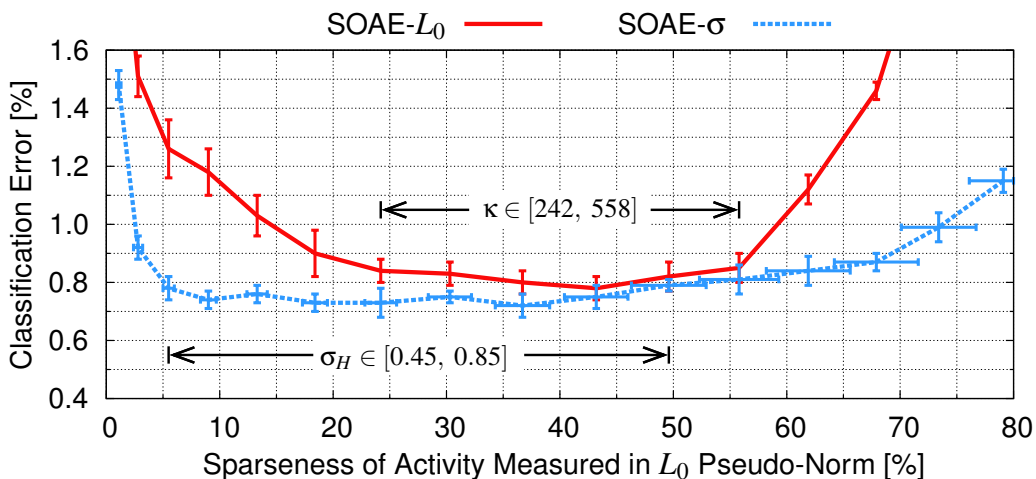

Figure 6: Resulting classification error on the MNIST evaluation set for the supervised online auto-encoder network, in dependence of sparseness of activity in the hidden layer. The projection onto an L_0 pseudo-norm constraint for variant SOAE-L_0 and the projection onto a constraint determined by sparseness measure σ for variant SOAE-σ were used as transfer functions. The error bars indicate $\pm$ one standard deviation difference from the mean.

Approach	Sparse Connectivity	Sparse Activity	Result (W, p) of Shapiro-Wilk Test	Evaluation Error [%]
SOAE-σ	$\sigma_W = 0.75$	$\sigma_H \in [0.45,\ 0.85]$	$(0.9802,\ 0.60)$	0.75 ± 0.04
SOAE-L_0	$\sigma_W = 0.75$	$\kappa \in [242,\ 558]$	$(0.9786,\ 0.53)$	0.82 ± 0.05
SOAE-σ-conn	none	$\sigma_H \in [0.45,\ 0.85]$	$(0.9747,\ 0.40)$	0.81 ± 0.04
SMLP-SCFC	$\sigma_W = 0.75$	none	$(0.9770,\ 0.47)$	0.81 ± 0.05
MLP-OBD	$\gamma = 12.5\%$	none	$(0.9807,\ 0.62)$	0.89 ± 0.04
MLP-random	none	none	$(0.9798,\ 0.58)$	0.88 ± 0.03
MLP-samples	none	none	$(0.9773,\ 0.49)$	0.91 ± 0.05
MLP-SCFC	none	none	$(0.9794,\ 0.57)$	0.91 ± 0.06

Table 1: Overview of comparative experiments. The second and third columns indicate whether sparse connectivity or sparse activity was incorporated, respectively. The fourth column reports the result of a statistical test for normality, which is interpreted in Section 3.5. The final column gives the median $\pm$ one standard deviation of the achieved classification error on the MNIST evaluation set. The results for each experiment were trimmed to gain a sample of size 47, allowing for statistical robust estimates.

where 15% of the original data were trimmed away. This procedure was also applied to the results of SOAE-σ, SOAE-σ-conn, and SOAE-L_0, to obtain a total of eight random samples of equal size for comparison with another.

The most basic variant, denoted the baseline in this discussion, is MLP-random, where all network parameters were initialized randomly. This achieved a median error of 0.88% on the evaluation set, being considerably worse than SOAE-σ. For variant MLP-samples, the hidden layer was initialized by replication of n randomly chosen samples from the learning set. This did decrease the overall learning time. However, the median classification error was slightly worse with 0.91% compared to MLP-random.

For variant MLP-SCFC, the network parameters were initialized in an unsupervised manner using the sparse coding for fast classification (SCFC) algorithm (Thom et al., 2011a). This method is a precursor to the SOAE proposed in this paper. It also features sparse connectivity and sparse activity but differs in some essential parts. First, sparseness of activity is achieved through a latent variable that stores the optimal sparse code words of all samples simultaneously. Using this matrix of code words, the activity of individual units was enforced to be sparse over time on the entire learning set. SOAE achieves sparseness over space, as for each sample only a pre-defined fraction of units is allowed to be active at any one time. A second difference is that sparse activity is achieved only indirectly by approximation of the latent matrix of code words with a feed-forward representation. With SOAE, sparseness of activity is guaranteed by construction. MLP-SCFC achieved a median classification error of 0.91% on the MNIST evaluation set, rendering it slightly worse than MLP-random and equivalent to MLP-samples.

The first experiment that incorporates only sparse connectivity is SMLP-SCFC. Initialization was done as for MLP-SCFC, but during training sparseness of connectivity was yielded by application of the sparseness-enforcing projection operator to the weights of the hidden layer after

every learning epoch. Hence the sparseness gained from unsupervised initialization was retained. MLP-SCFC features sparse connectivity only after initialization, but loses this property when training proceeds. With this slight modification, the median error of SMLP-SCFC decreases to 0.81%, which is significantly better than the baseline result.

The effect of better generalization due to sparse connectivity has also been observed by LeCun et al. (1990) in the context of convolutional neural networks. It can be explained by the bias-variance decomposition of the generalization error (Geman et al., 1992). When the effective number of the degrees of freedom is constrained, overfitting will be less likely and hence classifiers produce better results on average. The same argument can be applied to SOAE-σ, where additional sparse activity further improves classification results.

The last variant is called MLP-OBD. Here, the optimal brain damage (OBD) algorithm (LeCun et al., 1990) was used to prune synaptic connections in the hidden layer that are irrelevant for the computation of the classification decision of the network. The parameters of the network were first initialized randomly and then optimized on the learning set. Then the impact for each synaptic connection on the objective function was estimated using the Taylor series of the objective function, where a diagonal approximation of the Hessian was employed and terms of cubic or higher order were neglected. Using this information, the number of connections was halved by setting the weight of connections with low impact to zero. The network was then retrained with weights of removed connections kept at zero. This procedure was repeated until a target percentage γ of active synaptic connections in the hidden layer was achieved. For the results reported here, $\gamma = 12.5\%$ was chosen as this reflects the sparse connectivity $\sigma_W = 0.75$ of the other approaches best. MLP-OBD achieved a median classification error of 0.89%, which is comparable to the baseline result.

3.5 Statistical Analysis and Conclusions

A statistical analysis was carried out to assess the significance of the differences in the performance of the eight algorithms. The procedure follows the proposals of Pizarro et al. (2002) and Demšar (2006) for hypothesis testing, and is concluded by effect size estimation as proposed by Grissom (1994) and Acion et al. (2006). For each algorithm, a sample of size 47 was available, allowing for robust analysis results.

First, all results were tested for normality using the test developed by Shapiro and Wilk (1965). The resulting test statistics W and p-values are given in Table 1. As all p-values are large, it cannot be rejected that the samples came from normally distributed populations. Thus normality is assumed in the remainder of this discussion. Next, the test proposed by Levene (1960) was applied to determine whether equality of variances of the groups holds. This resulted in a test statistic $F = 2.7979$ with 7 and 368 degrees of freedom, and therefore a p-value of 0.0075. Hence the hypothesis that all group variances are equal can be rejected with very high significance. Consequently, parametric omnibus and post-hoc tests cannot be applied, as they require the groups to have equal variance.

As an alternative, the nonparametric test by Kruskal and Wallis (1952) which is based on rank information was employed to test whether all algorithms produced classifiers with equal classification errors in the mean. The test statistic was $H = 214.44$ with 7 degrees of freedom, and the p-value was less than 10^{-15}. There is hence a statistically significant difference in the mean classification results. To locate this deviation, a critical difference for comparing the mean ranks of the algorithms was computed. A Tukey-Kramer type modification applied to Dunn's procedure yields this critical difference, which is less conservative than Nemenyi's procedure for the Kruskal-Wallis
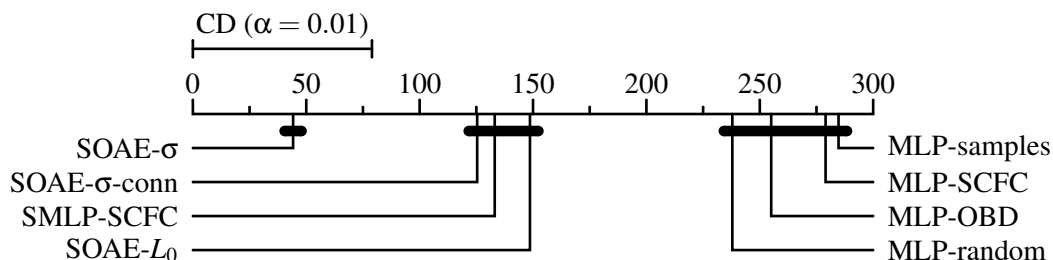
Figure 7: Diagram for multiple comparison of algorithms following Demšar (2006). For each algorithm, the mean rank was computed during the Kruskal-Wallis test. Then, a critical difference (CD) was computed at the $\alpha = 0.01$ significance level. Two algorithms produce classification results that are statistically not equal if the difference between their mean ranks is greater than the critical difference. This induced three groups of algorithms that produced statistically equivalent results, which are marked with black bars.

test (Hochberg and Tamhane, 1987). Note that this approach is nevertheless similar to the post-hoc procedure proposed by Demšar (2006) for paired observations, such that the diagrams proposed there can be adapted to the case for unpaired observations. The result is depicted in Figure 7, where the critical difference for statistical significance at the $\alpha = 0.01$ level is given. This test induces a highly significant partitioning of the eight algorithms, namely three groups A, B and C given by

$$A := \{\text{SOAE-}\sigma\}, \ B := \{\text{SOAE-}\sigma\text{-conn, SOAE-}L_0, \text{SMLP-SCFC}\},$$
$$\text{and } C := \{\text{MLP-OBD, MLP-random, MLP-samples, MLP-SCFC}\}.$$

This partition in turn induces an equivalence relation. Statistical equivalence is hence unambiguous and well-defined at $\alpha = 0.01$. Moreover, the p-value for this partition is 0.007. If the significance level α would have been set lower than this, then groups A and B would blend together.

To assess the benefit when an algorithm from one group is chosen over an algorithm from another group, the probability of superior experiment outcome was estimated (Grissom, 1994; Acion et al., 2006). For this, the classification errors were pooled with respect to membership in the three groups. It was then tested whether these pooled results still come from normal distributions. As group A is a singleton, this is trivially fulfilled with the result from Table 1. For group B, the Shapiro-Wilk test statistic was $W = 0.9845$ and the p-value was 0.11. Group C achieved a test statistic of $W = 0.9882$ and a p-value of 0.12. If a standard significance level of $\alpha = 0.01$ is chosen, then B and C can be assumed to be normally distributed also.

Let E_G be the random variable modeling the classification results of the algorithms from group $G \in \{A, B, C\}$. It is assumed that E_G is normally distributed with unknown mean and unknown variance for all G. Then $E_G - E_{\tilde{G}}$ is clearly normally distributed also for two groups $G, \tilde{G} \in \{A, B, C\}$. Therefore, the probability $P(E_G < E_{\tilde{G}})$ that one algorithm produces a better classifier than another could be computed from the Gaussian error function if the group means and variances were known. However, using Rao-Blackwell theory a minimum variance unbiased estimator $\hat{R}_2$ of this probability can be computed easily (Downton, 1973). Evaluation of the expression for $\hat{R}_2$ shows that $P(E_A < E_B)$ can be estimated by 0.87, $P(E_B < E_C)$ can be estimated by 0.88, and $P(E_A < E_C)$

can be estimated by 0.99. Therefore, the effect of choosing SOAE-σ over any of the seven other algorithms is dramatic (Grissom, 1994).

These results can be interpreted as follows. When neither sparse activity nor sparse connectivity is incorporated, then the worst classification results are obtained regardless of the initialization of the network parameters. The exception is MLP-OBD which incorporates sparse connectivity, although, as its name says, in a destructive way. Once a synaptic connection has been removed, it cannot be recovered, as the measure for relevance of LeCun et al. (1990) vanishes for synaptic connections of zero strength. The statistics for SMLP-SCFC shows that when sparse connectivity is obtained using the sparseness-enforcing projection operator, then superior results can be achieved. Because of the nature of projected gradient descent, it is possible here to restore deleted connections if it helps to decrease the classification error during learning. For SOAE-σ-conn only sparse activity was used, and classification results were statistically equivalent to SMLP-SCFC.

Therefore, using either sparse activity or sparse connectivity improves classification capabilities. When both are used, then results improve even more as variant SOAE-σ shows. This does not hold for SOAE-L_0 however, where the L_0 projection was used as transfer function. As Hoyer's sparseness measure σ and the according projection possess desirable analytical properties, they can be considered smooth approximations to the L_0 pseudo-norm. It is this smoothness which seems to produce this benefit in practice.

4. Related Work

This section reviews work related with the contents of this paper. First, the theoretical foundations of the sparseness-enforcing projection operator are discussed. Next, its application as neuronal transfer function to achieve sparse activity in a classification scenario is put in context with alternative approaches, and possible advantages of sparse connectivity are described.

4.1 Sparseness-Enforcing Projection Operator

The first major part of this paper dealt with improvements to the work of Hoyer (2004) and Theis et al. (2005). Here, an algorithm for the sparseness-enforcing projection with respect to Hoyer's sparseness measure σ was proposed. The technical proof of correctness is given in Appendix C. The set that should be projected onto is an intersection of a simplex C and a hypercircle L, which is a hypersphere lying in a hyperplane. The overall procedure can be described as performing alternating projections onto C and certain subsets of L. This approach is common for handling projections onto intersections of individual sets. For example, von Neumann (1950) proposed essentially the same idea when the investigated sets are closed subspaces, and has shown that this converges to a solution. A similar approach can be carried out for intersections of closed, convex cones (Dykstra, 1983), which can be generalized to translated cones that can be used to approximate any convex set (Dykstra and Boyle, 1987). For these alternating methods, it is only necessary to know how projections onto individual members of the intersection can be achieved.

Although these methods exhibit great generality, they have two severe drawbacks in the scenario of this paper. First, the target set for projection must be an intersection of convex sets. The scaled canonical simplex C is clearly convex, but the hypercircle L is non-convex if it contains more than one point. The condition that generates L cannot easily be weakened to achieve convexity. If the original hypersphere were replaced with a closed ball, then L would be convex. But this changes the meaning of the problem dramatically, as now virtually any sparseness below the original target

degree of sparseness can be obtained. This is because when the target L_1 norm λ_1 is fixed, the sparseness measure σ decreases whenever the target L_2 norm decreases. In geometric terms, the method proposed in this paper performs a projection from within a circle onto its boundary to increase the sparseness of the working vector. This argument is given in more detail in Figure 11 and the proof of Lemma 28(f).

The second drawback of the general methods for projecting onto intersections is that a solution is only achieved asymptotically, even when the convexity requirements are fulfilled. Due to the special structure of C and L, the number of alternating projections that have to be carried out to find a solution using Algorithm 3 is bounded from above by the problem dimensionality. Thus an exact projection is always found in finite time. Furthermore, the solution is guaranteed to be found in time that is at most quadratic in problem dimensionality.

A crucial point is the computation of the projection onto C and certain subsets of L. Due to the nature of the L_2 norm, the latter is straightforward. For the former, efficient algorithms have been proposed recently (Duchi et al., 2008; Chen and Ye, 2011). When only independent solutions are required, the projection of a point x onto a scaled canonical simplex of L_1 norm λ_1 can also be carried out in linear time (Liu and Ye, 2009), without having to sort the vector that is to be projected. This can be achieved by showing that the separator $\hat{t}$ for performing the simplex projection is the unique zero of the monotonically decreasing function $t \mapsto \|\max(|x| - t \cdot e, \, 0)\|_1 - \lambda_1$. The zero of this function can be found efficiently using the bisection method, and exploiting the special structure of the occurring expressions (Liu and Ye, 2009).

In the context of this paper an explicit closed-form expression for $\hat{t}$ is preferable as it permits additional insight into the properties of the projected point. The major part in proving the correctness of Algorithm 1 is the interconnection between C and L, that is that the final solution has zero entries at the according positions in the working vector and thus a chain monotonically decreasing in L_0 pseudo-norm is achieved. This result is established through Lemma 26, which characterizes projections onto certain faces of a simplex, Corollary 27 and their application in Lemma 28.

Analysis of the theoretical properties of the sparseness-enforcing projection is concluded with its differentiability in Appendix D. The idea is to exploit the finiteness of the projection sequence and to apply the chain rule of differential calculus. It is necessary to show that the projection chain is robust in a neighborhood of the argument. This reduces analysis to individual projection steps which have already been studied in the literature. For example, the projection onto a closed, convex set is guaranteed to be differentiable almost everywhere (Hiriart-Urruty, 1982). Here non-convexity of L is not an issue, as the only critical point is its barycenter. For the simplex C, a characterization of critical points is given with Lemma 32 and Lemma 33, and it is shown that the expression for the projection onto C is invariant to local changes. An explicit expression for construction of the gradient of the sparseness-enforcing projection operator is given in Theorem 35. In Corollary 36 it is shown that the computation of the product of the gradient with an arbitrary vector can be achieved efficiently by exploiting sparseness and the special structure of the gradient.

Similar approaches for sparseness projections are discussed in the following. The iterative hard thresholding algorithm is a gradient descent algorithm, where a projection onto an L_0 pseudo-norm constraint is performed (Blumensath and Davies, 2009). Its application lies in compressed sensing, where a linear generative model is used to infer a sparse representation for a given observation. Sparseness here acts as regularizer which is necessary because observations are sampled below the Nyquist rate. In spite of the simplicity of the method, it can be shown that it achieves a good approximation to the optimal solution of this NP-hard problem (Blumensath and Davies, 2009).

Closely related with the work of this paper is the generalization of Hoyer's sparseness measure by Theis and Tanaka (2006). Here, the L_1 norm constraint is replaced with a generalized L_p pseudo-norm constraint, such that the sparseness measure becomes $\sigma_p(x) := \|x\|_p/\|x\|_2$. For $p = 1$, Hoyer's sparseness measure up to a constant normalization is obtained. When p converges decreasingly to zero, then $\sigma_p(x)^p$ converges point-wise to the L_0 pseudo-norm. Hence for small values of p a more natural sparseness measure is obtained. Theis and Tanaka (2006) also proposed an extension of Hoyer's projection algorithm. It is essentially von Neumann's alternating projection method, where closed subspaces have been replaced by "spheres" that are induced by L_p pseudo-norms. Note that these sets are non-convex when $p < 1$, such that convergence is not guaranteed. Further, no closed-form solution for the projection onto an "L_p-sphere" is known for $p \notin \{1, 2, \infty\}$, such that numerical methods have to be employed.

A problem where similar projections are employed is to minimize a convex function subject to group sparseness (see for example Friedman et al., 2010). In this context, mixed norm balls are of particular interest (Sra, 2012). For a matrix $X \in \mathbb{R}^{n \times g}$, the mixed $L_{p,q}$ norm is defined as the L_p norm of the L_q norms of the columns of X, that is $\|X\|_{p,q} := \left\| \left(\|Xe_1\|_q, \ldots, \|Xe_g\|_q \right)^T \right\|_p$. Here, X can be interpreted to be a data point with entries partitioned into g groups. When $p = 1$, then the projection onto a simplex can be generalized directly for $q = 2$ (van den Berg et al., 2008) and for $q = \infty$ (Quattoni et al., 2009). The case when $p = 1$ and $q \geq 1$ is more difficult, but can be solved as well (Liu and Ye, 2010; Sra, 2012).

The last problem discussed here is the elastic net criterion (Zou and Hastie, 2005), which is a constraint on the sum of an L_1 norm and an L_2 norm. The feasible set can be written as the convex set $N := \left\{ s \in \mathbb{R}^n \mid \lambda_1 \|s\|_1 + \lambda_2 \|s\|_2^2 \leq 1 \right\}$, where $\lambda_1, \lambda_2 \geq 0$ control the shape of N. Note that in N only the sum of two norms is considered, whereas the non-convex set $S^{(\lambda_1, \lambda_2)}$ consists of the intersection of two different constraints. Therefore, the elastic net induces a different notion of sparseness than Hoyer's sparseness measure σ does. As is the case for mixed norm balls, the projection onto a simplex can be generalized to achieve projections onto N (Mairal et al., 2010).

4.2 Supervised Online Auto-Encoder

The sparseness-enforcing projection operator π with respect to Hoyer's sparseness measure σ and the projection onto an L_0 pseudo-norm constraint are differentiable almost everywhere. Thus they are suitable for gradient-based optimization algorithms. In Section 3, they were used as transfer functions in a hybrid of an auto-encoder network and a two-layer neural network to infer a sparse internal representation. This representation was subsequently employed to approximate the input sample and to compute a classification decision. In addition, the matrix of bases which was used to compute the internal representation was enforced to be sparsely populated by application of the sparseness projection after each learning epoch. Hence the supervised online auto-encoder proposed in this paper features both sparse activity and sparse connectivity.

These two key properties have also been investigated and exploited in the context of auto-associative memories for binary inputs. If the entries of the training patterns are sparsely populated, the weight matrix of the memory will be sparsely populated as well after training if Hebbian-like learning rules are used (Kohonen, 1972). The assumption of sparsely coded inputs also results in increased completion capacity and noise resistance of the associative memory (Palm, 1980). If the input data is not sparse inherently, feature detectors can perform a sparsification prior to the actual processing through the memory (Baum et al., 1988).

A purely generative model that also possesses these two key properties is non-negative matrix factorization with sparseness constraints (Hoyer, 2004). This is an extension to plain non-negative matrix factorization (Paatero and Tapper, 1994) which was shown to achieve sparse connectivity on certain data sets (Lee and Seung, 1999). However, there are data sets on which this does not work (Li et al., 2001; Hoyer, 2004). Although Hoyer's model makes sparseness easily controllable by explicit constraints, it is not inherently suited to classification tasks. An extension intended to incorporate class membership information to increase discriminative capabilities was proposed by Heiler and Schnörr (2006). In their approach, an additional constraint was added ensuring that every internal representation is close to the mean of all internal representations that belong to the same class. In other words, the method can be interpreted as supervised clustering, with the number of clusters equal to the number of classes. However, there is no guarantee that a distribution of internal representations exists such that both the reproduction error is minimized and the internal representations can be arranged in such a pattern. Unfortunately, Heiler and Schnörr (2006) used only a subset of a small data set for handwritten digit recognition to evaluate their approach.

A precursor to the supervised online auto-encoder was proposed by Thom et al. (2011a). There, inference of sparse internal representations was achieved by fitting a one-layer neural network to approximate a latent variable of optimal sparse representations. The transfer function used for this approximation was a hyperbolic tangent raised to an odd power greater or equal to three. This resulted in a depression of activities with small magnitude, favoring sparseness of the result. Similar techniques to achieve a shrinkage-like effect for increasing sparseness of activity in a neural network were used by Gregor and LeCun (2010) and Glorot et al. (2011). Information processing is here purely local, that is a scalar function is evaluated entrywise on a vector, and thus no information is interchanged among individual entries.

The use of non-local shrinkage to reduce Gaussian noise in sparse coding has already been described by Hyvärinen et al. (1999). Here, a maximum likelihood estimate with only weak assumptions yields a shrinkage operation, which can be conceived as projection onto a scaled canonical simplex. In the use case of object recognition, a hard shrinkage was also employed to de-noise filter responses (Mutch and Lowe, 2006). Whenever a best approximation from a permutation-invariant set is used, a shrinkage-like operation must be employed. Using a projection operator as neural transfer function is hence a natural extension of these ideas. When the projection is sufficiently smooth, the entire model can be tuned end-to-end using gradient methods to achieve an auto-encoder or a classifier.

The second building block from Thom et al. (2011a) that was incorporated into supervised online auto-encoder is the architectural concept for classification. It is well-known that two layers in a neural network are sufficient to approximate any continuous function on a compactum with arbitrary precision (Cybenko, 1989; Funahashi, 1989; Hornik et al., 1989). Similar architectures have also been proposed for classification in combination with sparse coding of the inputs. However, sparse connectivity was not considered in this context. Bradley and Bagnell (2009) used the Kullback-Leibler divergence as implicit sparseness penalty term and combined this with the back-propagation algorithm to yield a classifier that achieved a 1.30% error rate on the MNIST evaluation set. The Kullback-Leibler divergence was chosen to replace the usual L_1 norm penalty term, as it is smoother than the latter and therefore sparsely coded internal representations are more stable subject to subtle changes of the input. A related technique is supervised dictionary learning by Mairal et al. (2009), where the objective function is an additive combination of a classification error term, a term for the reproduction error, and an L_1 norm constraint. Inference of sparse internal represen-

tations is achieved through solving an optimization problem. Such procedures are time-consuming and greatly increase the computational complexity of classification. With this approach, a classification error of 1.05% on the MNIST evaluation set was achieved. These two approaches used the original MNIST learning set without jittering the digits and can thus be considered permutation-invariant. Augmentation of the learning set with virtual samples would have contributed to improve classification performance, as demonstrated by Schölkopf (1997).

Finally consider once more the sparse connectivity property, which is mostly neglected in the literature in favor of sparse activity. It was shown in this paper that sparse connectivity helps to improve generalization capabilities. In practice, this property can also be used to reduce the computational complexity of classification by one order of magnitude (Thom et al., 2011b). This results from exploiting sparseness and using sparse matrix-vector multiplication algorithms to infer the internal representation, which is the major computational burden in class membership prediction. It was shown in this paper and by Thom et al. (2011b) that a small number of nonzero entries in the weight matrix of the hidden layer is sufficient for achieving good classification results. Furthermore, the additional savings in required storage capacity and bandwidth allow using platforms with modest computational power for practical implementations. Sparseness is therefore an elementary concept of efficiency in artificial processing systems.

5. Conclusions

Without sparseness in their brains, higher mammals probably would not have developed to viable life-forms. This important concept of efficiency was discovered by neuroscientists, and practical benefit was obtained by the engineers of artificial information processing systems. This paper studied Hoyer's sparseness measure σ, and in particular the projection of arbitrary vectors onto sets where σ attains a constant value. A simple yet efficient algorithm for computing this sparseness-enforcing projection operator was proposed in this paper, and its correctness was proved. In addition, it was demonstrated that the proposed algorithm is superior in run-time to Hoyer's original algorithm. The analysis of the theoretical properties of this projection was concluded by showing it is differentiable almost everywhere.

As projections onto σ constraints are well-understood, they constitute the ideal tool for building systems that can benefit from sparseness constraints. An original use case was introduced in this paper. Here, the σ projection was implemented as neuronal transfer function, yielding a differentiable closed-form expression for inference of sparse code words. Besides this sparse activity, the connectivity in this system was also forced to be sparse by performing the σ projection after the presentation of learning examples. Because of its smoothness, the entire system can be optimized end-to-end by gradient-based methods, yielding a classification architecture exhibiting true sparse information processing.

This supervised online auto-encoder was applied on a benchmark data set for pattern recognition. Because sparseness constraints reduce the amount of feasible solutions, it is not clear in the first place whether the same performance can be achieved at all. However, when the target degree of sparseness of the activity is in a reasonable range, classification results are not only equivalent but superior to classical non-sparse approaches. This result is supported by statistical evaluation showing that this performance increase is not merely coincidental, but statistically significant. Therefore, sparseness can be seen as regularizer that offers the potential to improve artificial systems in the same way it seems to improve biological systems.

Appendix A. Notation and Prerequisites

This appendix fixes the notation and provides prerequisites for the following appendices. $\mathbb{N}$ denotes the natural numbers including zero, $\mathbb{R}$ the real numbers and $\mathbb{R}_{\geq 0}$ the non-negative real numbers. $\mathbb{R}^n$ is the n-dimensional Euclidean space with canonical basis $e_1, \ldots, e_n \in \mathbb{R}^n$, and $e := \sum_{i=1}^n e_i \in \mathbb{R}^n$ denotes the vector where all entries are identical to unity. For all other vectors, a subscript denotes the corresponding entry of the vector, that is $x_i = e_i^T x$ for $x \in \mathbb{R}^n$. The amount of nonzero entries in a vector is given by the L_0 pseudo-norm, $\|\cdot\|_0$. $\|\cdot\|_1$ and $\|\cdot\|_2$ denote the Manhattan norm and Euclidean norm, respectively. $\langle \cdot, \cdot \rangle$ denotes the canonical dot product in the Euclidean space. Given a vector x, $\mathrm{diag}(x)$ denotes the square matrix with x on its main diagonal and zero entries at all other positions, and $a \circ b = \mathrm{diag}(a)b$ denotes the Hadamard product or entrywise product for vectors a and b. When A and B are square matrices, then $\mathrm{diag}(A, B)$ denotes the block diagonal matrix with the blocks given by A and B. S_n is the symmetric group, and P_τ denotes the permutation matrix for $\tau \in S_n$. For a set $M \subseteq U$, M^C denotes its complement in the universal set U, where $U \in \{\mathbb{R}^n, \{1, \ldots, n\}\}$ is clear from the context. The power set of M is denoted by $\wp(M)$. If $M \subseteq \mathbb{R}^n$, then ∂M denotes its boundary in the topological sense. The sign function is denoted by $\mathrm{sgn}(\cdot)$. A list of symbols that are frequently used throughout the paper is given in Table 2.

The important concept of the projection onto a set was given in Definition 1. The following basic statement will be used extensively in this paper and follows from $\langle x, x \rangle = \|x\|_2^2$ for all $x \in \mathbb{R}^n$ and the fact that the scalar product is a symmetric bilinear form (Laub, 2004):

Proposition 4 *Let $a, b \in \mathbb{R}^n$. Then $\|a \pm b\|_2^2 = \|a\|_2^2 + \|b\|_2^2 \pm 2\langle a, b \rangle$. Further it is $\|a - b\|_2^2 = \|a - p\|_2^2 + \|p - b\|_2^2 + 2\langle a - p, p - b \rangle$ for all $p \in \mathbb{R}^n$.*

As an example, note that the outcome of the sparseness-enforcing projection operator depends only on the target sparseness degree up to scaling:

Remark 5 *Let $\lambda_1, \lambda_2 > 0$ and $\tilde{\lambda}_1, \tilde{\lambda}_2 > 0$ be pairs of target norms such that $\lambda_1/\lambda_2 = \tilde{\lambda}_1/\tilde{\lambda}_2$. Then*

$$\mathrm{proj}_{S^{(\lambda_1, \lambda_2)}}(x) = \tilde{\lambda}_2/\lambda_2 \cdot \mathrm{proj}_{S^{(\tilde{\lambda}_1, \tilde{\lambda}_2)}}(x) \text{ for all } x \in \mathbb{R}^n.$$

Proof It is sufficient to show only one inclusion. Let $x \in \mathbb{R}^n$ be arbitrary, $p \in \mathrm{proj}_{S^{(\lambda_1, \lambda_2)}}(x)$ and $\tilde{r} \in S^{(\tilde{\lambda}_1, \tilde{\lambda}_2)}$. Define $\tilde{p} := \tilde{\lambda}_1/\lambda_1 \cdot p = \tilde{\lambda}_2/\lambda_2 \cdot p \in \mathbb{R}^n$, then $\|\tilde{p}\|_1 = |\tilde{\lambda}_1/\lambda_1| \cdot \|p\|_1 = \tilde{\lambda}_1$ and analogously $\|\tilde{p}\|_2 = \tilde{\lambda}_2$, hence $\tilde{p} \in S^{(\tilde{\lambda}_1, \tilde{\lambda}_2)}$. For the claim to hold it has now to be shown that $\|\tilde{p} - x\|_2 \leq \|\tilde{r} - x\|_2$. Write $r := \lambda_2/\tilde{\lambda}_2 \cdot \tilde{r} \in \mathbb{R}^n$, which in fact lies in $S^{(\lambda_1, \lambda_2)}$. So $\|p - x\|_2 \leq \|r - x\|_2$ by definition of p, and with Proposition 4 follows $\|\tilde{r} - x\|_2^2 - \|\tilde{p} - x\|_2^2 = \|\tilde{r}\|_2^2 + \|x\|_2^2 - 2\langle \tilde{r}, x \rangle - \|\tilde{p}\|_2^2 - \|x\|_2^2 + 2\langle \tilde{p}, x \rangle = 2\langle \tilde{p} - \tilde{r}, x \rangle = \tilde{\lambda}_2/\lambda_2 \cdot 2\langle p - r, x \rangle = \tilde{\lambda}_2/\lambda_2 \cdot \left(\|r - x\|_2^2 - \|p - x\|_2^2 \right) \geq 0$. ∎

Hence only the ratio of the target L_1 norm to the target L_2 norm is important and not their actual scale. This argument can be generalized to projections onto any scale-invariant set and therefore naturally holds also for $S_{\geq 0}^{(\lambda_1, \lambda_2)}$.

Symbol and Definition	Meaning
σ (see Section 1.1)	Sparseness measure by Hoyer (2004)
π and $\pi_{\geq 0}$ (see Section 2.4)	Sparseness projection cast as function
$n \in \mathbb{N}$	Problem dimensionality
$e_1, \ldots, e_n \in \mathbb{R}^n$	Canonical basis of $\mathbb{R}^n$
$e := \sum_{i=1}^n e_i \in \mathbb{R}^n$	Vector where all entries are one
$\lambda_1 \in \mathbb{R}_{>0}$	Target L_1 or Manhattan norm
$\lambda_2 \in \mathbb{R}_{>0}$	Target L_2 or Euclidean norm
$S^{(\lambda_1, \lambda_2)} \subseteq \mathbb{R}^n$ (see Section 1.1)	Target set for sparseness projection
$S_{\geq 0}^{(\lambda_1, \lambda_2)} := S^{(\lambda_1, \lambda_2)} \cap \mathbb{R}_{\geq 0}^n$	Target set for non-negative sparseness projection
$D := S_{\geq 0}^{(\lambda_1, \lambda_2)}$	Short for the non-negative target set
$H := \{ a \in \mathbb{R}^n \mid e^T a = \lambda_1 \}$	Target hyperplane
$K := \{ q \in \mathbb{R}^n \mid \|q\|_2 = \lambda_2 \}$	Target hypersphere
$L := H \cap K$	Target hypercircle
$C := \mathbb{R}_{\geq 0}^n \cap H$	Scaled canonical simplex
$m := \lambda_1/n \cdot e \in \mathbb{R}^n$	Barycenter of L and C
$\rho := \lambda_2^2 - \lambda_1^2/n \in \mathbb{R}$	Squared radius of L
$I \subseteq \{1, \ldots, n\}$	Index set of nonzero entries
$d := |I| \in \mathbb{N}$	Working dimensionality
$L_I := \{ a \in L \mid a_i = 0 \text{ for all } i \notin I \}$	Points in L where certain coordinates vanish
$C_I := \{ c \in C \mid c_i = 0 \text{ for all } i \notin I \}$	Face of simplex C
$m_I := \lambda_1/d \cdot \sum_{i \in I} e_i \in \mathbb{R}^n$	Barycenter of L_I and C_I
$\rho_I := \lambda_2^2 - \lambda_1^2/d \in \mathbb{R}$	Squared radius of L_I

Table 2: A list of symbols used frequently in this paper and their meaning.

Appendix B. Projections onto Symmetric Sets

This appendix investigates certain symmetries of sets and their effect on projections onto such sets. A great variety of sparseness measures fulfills certain symmetries as vector entries are equally weighted, see Hurley and Rickard (2009). This means that no entry is preferred over another, and for negative entries usually the absolute value or the squared value is taken, such that the signs of the entries are ignored. Consider the following definition of symmetries that are to be analyzed:

Definition 6 *Let $\varnothing \neq M \subseteq \mathbb{R}^n$. Then M is called* permutation-invariant *if and only if $P_\tau x \in M$ for all $x \in M$ and all permutations $\tau \in S_n$. Further, M is called* reflection-invariant *if and only if $b \circ x \in M$ for all $x \in M$ and all $b \in \{ \pm 1 \}^n$.*

In other words, a subset M of the Euclidean space is permutation-invariant if set membership is invariant to permutation of individual coordinates. M is reflection-invariant if single entries can be

negated without violating set membership. This is equivalent to $x - 2\sum_{i\in I}x_i e_i \in M$ for all $x \in M$ and all index sets $I \subseteq \{1,\ldots,n\}$, which is a condition that is technically easier to handle. The following observation states that these symmetries are closed under common set operations:

Remark 7 *Let $\emptyset \neq A, B \subseteq \mathbb{R}^n$. When A and B are permutation-invariant or reflection-invariant, then so are $A \cup B$, $A \cap B$ and A^C.*

The proof is obvious by elementary set algebra. Now consider the following general properties of functions mapping to power sets:

Definition 8 *Let $\emptyset \neq M \subseteq \mathbb{R}^n$, let $\wp(M)$ be its power set and let $f \colon \mathbb{R}^n \to \wp(M)$ be a function. f is called* order-preserving *if and only if $x_i > x_j$ implies $p_i \geq p_j$ for all $x \in \mathbb{R}^n$, for all $p \in f(x)$ and for all $i, j \in \{1,\ldots,n\}$. f is called* absolutely order-preserving *if and only if from $|x_i| > |x_j|$ follows $|p_i| \geq |p_j|$ for all $x \in \mathbb{R}^n$, for all $p \in f(x)$ and for all $i, j \in \{1,\ldots,n\}$. f is called* orthant-preserving *if and only if $\operatorname{sgn}(x_i) = \operatorname{sgn}(p_i)$ or $x_i = 0$ or $p_i = 0$ for all $x \in M$ and all $p \in f(x)$.*

Hence, a function f is order-preserving if the relative order of entries of its arguments does not change upon function evaluation. Thus if the entries of x are sorted in ascending or descending order, then so are the entries of every vector in $f(x)$. Orthant-preservation denotes the fact that x and every vector from $f(x)$ are located in the same orthant. The link between set symmetries and projection properties is established by the following result. A weaker form of its statements has been described by Duchi et al. (2008) in the special case of a projection onto a simplex.

Lemma 9 *Let $\emptyset \neq M \subseteq \mathbb{R}^n$ and $p \colon \mathbb{R}^n \to \wp(M)$, $x \mapsto \operatorname{proj}_M(x)$. Then the following holds:*

(a) When M is permutation-invariant, then p is order-preserving.

(b) When M is reflection-invariant, then p is orthant-preserving.

Proof (a) Let $x \in \mathbb{R}^n$ and $p \in \operatorname{proj}_M(x)$. Let $i, j \in \{1,\ldots,n\}$ with $x_i > x_j$. Assume that $p_i < p_j$. Let $\tau := (i, j) \in S_n$ and $q := P_\tau p$, then $q \in M$ because of M being permutation-invariant. Consider $d := \|p - x\|_2^2 - \|q - x\|_2^2$. Because τ is a single transposition, application of Proposition 4 yields $d = 2(p_j - p_i)(x_i - x_j)$. By requirement $d > 0$, which contradicts the minimality of p as being a projection of x onto M. Hence $p_i \geq p_j$ must hold.

(b) Let $x \in \mathbb{R}^n$ and $p \in \operatorname{proj}_M(x)$. Define $I := \{i \in \{1,\ldots,n\} \mid \operatorname{sgn}(x_i) \neq \operatorname{sgn}(p_i)\}$. The claim holds trivially if $I = \emptyset$. Assume $I \neq \emptyset$ and define $q := p - 2\sum_{i\in I}p_i e_i$. It follows $q \in M$ because M is reflection-invariant. Proposition 4 implies $\|q\|_2^2 = \|p\|_2^2$, and clearly $\langle q, x \rangle = \langle p, x \rangle - 2\sum_{i\in I}p_i x_i$. Therefore application of Proposition 4 yields $d := \|p - x\|_2^2 - \|q - x\|_2^2 = -4\sum_{i\in I}p_i x_i$. By the definition of I one obtains $p_i x_i \in \{-1, 0\}$. Hence would there be an index $i \in I$ with $p_i \neq 0$ and $x_i \neq 0$, then $d > 0$, but $\|p - x\|_2^2 > \|q - x\|_2^2$ would contradict the minimality of p. Therefore $I = \{i \in \{1,\ldots,n\} \mid p_i = 0 \text{ or } x_i = 0\}$, and the claim follows. ∎

When the projection onto a permutation-invariant set is unique, then equal entries of the argument cause equal entries in the projection:

Remark 10 *Let $\emptyset \neq M \subseteq \mathbb{R}^n$ be permutation-invariant and $x \in \mathbb{R}^n$. When $p = \operatorname{proj}_M(x)$ is unique, then $p_i = p_j$ follows for all $i, j \in \{1,\ldots,n\}$ with $x_i = x_j$.*

Proof Let $x \in \mathbb{R}^n$, $p = \mathrm{proj}_M(x)$ and $i, j \in \{1, \ldots, n\}$ with $x_i = x_j$. Assume $p_i \neq p_j$ would hold and let $\tau := (i, j) \in S_n$ and $q := P_\tau p \neq p$. With the permutation-invariance of M follows $q \in M$, and $\|q - x\|_2 = \|p - x\|_2$ with $x_i = x_j$. Hence $q \in \mathrm{proj}_M(x)$, so $q = p$ with the uniqueness of the projection, which contradicts $q \neq p$. Therefore, $p_i = p_j$. ∎

The next result shows how solutions to a projection onto reflection-invariant sets can be turned into non-negative solutions and vice-versa. Its second part was already observed by Hoyer (2004), in the special case of the sparseness-enforcing projection operator, and by Duchi et al. (2008), when the connection between projections onto a simplex and onto an L_1 ball was studied. Both did not provide a proof, but in the latter work a hint to a possible proof was given. With Lemma 11 it suffices to consider non-negative solutions for projections onto reflection-invariant sets.

Lemma 11 *Let $\varnothing \neq A \subseteq \mathbb{R}^n$ be reflection-invariant, $B := A \cap \mathbb{R}^n_{\geq 0}$ and $p, x \in \mathbb{R}^n$. Then:*

  (a) *If $p \in \mathrm{proj}_A(x)$, then $|p| \in \mathrm{proj}_B(|x|)$.*

  (b) *If $p \in \mathrm{proj}_B(|x|)$, then $s \circ p \in \mathrm{proj}_A(x)$ where $s \in \{\pm 1\}^n$ is given by $s_i := 1$ if $x_i \geq 0$ and $s_i := -1$ otherwise for all $i \in \{1, \ldots, n\}$.*

Proof First note that if $q \in \mathrm{proj}_A(x)$, then $\mathrm{sgn}(x_i) = \mathrm{sgn}(q_i)$ or $x_i = 0$ or $q_i = 0$ with Lemma 9(b). Hence for all $i \in \{1, \ldots, n\}$ follows $(|q_i| - |x_i|)^2 = (q_i \cdot \mathrm{sgn}(q_i) - x_i \cdot \mathrm{sgn}(x_i))^2 = (q_i - x_i)^2$, and therefore $\|q - x\|_2^2 = \||q| - |x|\|_2^2$. Furthermore, $|q| \in A$ because of A reflection-invariant and $|q| \in \mathbb{R}^n_{\geq 0}$, so $|q| \in B$.

(a) Let $p \in \mathrm{proj}_A(x)$ and $q \in B$, then $|p| \in B$ and it has to be shown that $\||p| - |x|\|_2 \leq \|q - |x|\|_2$. Define $I := \{i \in \{1, \ldots, n\} \mid x_i < 0\}$ and $\tilde{q} := q - 2\sum_{i \in I} q_i e_i$, that is the signs of entries in I are flipped. Clearly $\tilde{q} \in A$, so in conjunction with the remark at the beginning of the proof follows $\||p| - |x|\|_2^2 = \|p - x\|_2^2 \leq \|\tilde{q} - x\|_2^2$. For $i \notin I$ one obtains $x_i \geq 0$ and $\tilde{q}_i = q_i$, hence $\tilde{q}_i - x_i = q_i - |x_i|$. For $i \in I$ follows $x_i < 0$ and $\tilde{q}_i = -q_i$, hence $\tilde{q}_i - x_i = -(q_i - |x_i|)$. This yields $(\tilde{q}_i - x_i)^2 = (q_i - |x_i|)^2$ for all $i \in \{1, \ldots, n\}$, thus $\|\tilde{q} - x\|_2^2 = \|q - |x|\|_2^2$, and the claim follows.

(b) Let $p \in \mathrm{proj}_B(|x|)$. If $i \in \{1, \ldots, n\}$ with $x_i \geq 0$, then clearly $s_i p_i - x_i = p_i - |x_i|$. For $i \in \{1, \ldots, n\}$ with $x_i < 0$ follows $s_i p_i - x_i = -(p_i - |x_i|)$. Therefore, $\|s \circ p - x\|_2^2 = \|p - |x|\|_2^2$. Let $q \in \mathrm{proj}_A(x)$, then the remark at the beginning of the proof yields $\|q - x\|_2^2 = \||q| - |x|\|_2^2$ and $|q| \in B$. $p \in \mathrm{proj}_B(|x|)$ yields $\|p - |x|\|_2^2 \leq \||q| - |x|\|_2^2$, and the claim follows. ∎

Using this result immediately yields a condition for projections to be absolutely order-preserving:

Lemma 12 *Let $\varnothing \neq M \subseteq \mathbb{R}^n$ be both permutation-invariant and reflection-invariant. Then the function $p \colon \mathbb{R}^n \to \wp(M)$, $x \mapsto \mathrm{proj}_M(x)$, is absolutely order-preserving.*

Proof Let $x \in \mathbb{R}^n$, $p \in \mathrm{proj}_M(x)$, and $i, j \in \{1, \ldots, n\}$ with $|x_i| > |x_j|$. Define $L := M \cap \mathbb{R}^n_{\geq 0}$, which is permutation-invariant with Remark 7. Lemma 11 implies that $|p| \in \mathrm{proj}_L(|x|)$, and with Lemma 9 follows $|p_i| \geq |p_j|$. ∎

The application of these elementary results to projections onto sets on which functions achieve constant values is straightforward. Examples were given in Section 2 with the sets Z and $S^{(\lambda_1, \lambda_2)}$.

Appendix C. Proof of Correctness of Algorithm 1 and Algorithm 3

The purpose of this appendix is to rigorously prove correctness of Algorithm 1 and Algorithm 3, that is that they compute projections onto $S^{(\lambda_1, \lambda_2)}_{\geq 0}$. Projections onto $S^{(\lambda_1, \lambda_2)}$ can then be inferred easily as explained in Appendix B.

C.1 Geometric Structures and First Considerations

The aim is to compute projections onto D, which is the intersection of the non-negative orthant $\mathbb{R}^n_{\geq 0}$, the target hyperplane H and the target hypersphere K, see Section 2.1. Further, the intersection of H and K yields a hypercircle L, and the intersection of $\mathbb{R}^n_{\geq 0}$ and H yields a scaled canonical simplex C. The structure of H and L will be analyzed in Section C.1.1 and Section C.1.2, respectively. The properties of C are discussed in Section C.2 and Section C.3. These results will then be used in Section C.4 to prove Theorem 2 and Theorem 3.

For the analysis of subsets where certain coordinates vanish, it is useful to define the following quantities for an index set $I \subseteq \{1, \ldots, n\}$ with cardinality $d := |I|$. The corresponding face of C is denoted by $C_I := \{c \in C \mid c_i = 0 \text{ for all } i \notin I\}$ and has barycenter $m_I := \lambda_1/d \cdot \sum_{i \in I} e_i \in C_I$. Further, $L_I := \{a \in L \mid a_i = 0 \text{ for all } i \notin I\}$ denotes the hypercircle with according vanishing entries, and $\rho_I := \lambda_2^2 - \lambda_1^2/d$ is the squared radius of L_I. Note that m_I is also the barycenter of L_I.

With these definitions the intermediate goal is now to prove that projections onto D can be computed by alternating projections onto the geometric structures defined earlier. The idea is to show that the set of solutions is not tampered by alternating projections onto H, C, L and L_I.

C.1.1 L_1 Norm Constraint—Target Hyperplane

First, the projection onto the target hyperplane H is considered. Lemma 13 is an elaborated version of a result from Theis et al. (2005), which is included here for completeness. Using its statements, it can be assumed that the considered point lies on H without modification of the solution set of the projection onto the target set D.

Lemma 13 *Let $x \in \mathbb{R}^n$. Then the following holds:*

(a) $\text{proj}_H(x) = x + 1/n \cdot (\lambda_1 - e^T x) e$.

(b) *Let $r := \text{proj}_H(x)$. Then $\text{proj}_D(x) = \text{proj}_D(r)$.*

Proof (a) This is essentially a projection onto a hyperplane, yielding a unique result.

(b) With (a) follows $r - x = 1/n \cdot (\lambda_1 - e^T x) e$. Hence $\langle h, r - x \rangle = \lambda_1/n \cdot (\lambda_1 - e^T x)$ holds for arbitrary $h \in H$. This expression is independent of the entries of h, which yields $\langle a - b, r - x \rangle = 0$ for every $a, b \in H$.

Now let $p \in \text{proj}_D(x)$, that is $\|p - x\|_2 \leq \|q - x\|_2$ for all $q \in D$. Let $q \in D$ be arbitrary. With $D \subseteq H$ follows $\langle q - p, r - x \rangle = 0$, and thus Proposition 4 yields $\|q - r\|_2^2 - \|p - r\|_2^2 = \|q - x\|_2^2 - \|p - x\|_2^2 + 2\langle q - p, x - r \rangle = \|q - x\|_2^2 - \|p - x\|_2^2 \geq 0$, hence $\|p - r\|_2^2 \leq \|q - r\|_2^2$, so $p \in \text{proj}_D(r)$.

For the converse let $p \in \text{proj}_D(r)$. Analogously $\|q - x\|_2^2 - \|p - x\|_2^2 = \|q - r\|_2^2 - \|p - r\|_2^2 \geq 0$, hence $p \in \text{proj}_D(x)$. ∎

Therefore, the barycenter m is the projection of the origin onto H. The next remark gathers additional information on the norm of m and dot products with this point.

Remark 14 *It is $\|m\|_2^2 = \lambda_1^2/n$. Further, $\langle m, h \rangle = \lambda_1^2/n$ for all $h \in H$, and thus $\|h - m\|_2^2 = \|h\|_2^2 - \lambda_1^2/n$ with Proposition 4.*

C.1.2 L_1 AND L_2 NORM CONSTRAINT—TARGET HYPERSPHERE

After the projection onto the target hyperplane H has been carried out, consider now the joint constraint of H and the target hypersphere K. First note that $L = H \cap K$ is a hypercircle, that is a hypersphere in the subspace H, with intrinsic dimensionality reduced by one:

Lemma 15 *Consider $L = H \cap K$ and $\rho = \lambda_2^2 - \lambda_1^2/n$. Then the following holds:*

(a) $L = \tilde{L} := \{ q \in H \mid \|q - m\|_2^2 = \rho \}$.

(b) $L \neq \emptyset$ if and only if $\lambda_2 \geq \lambda_1/\sqrt{n}$.

Proof (a) Follows immediately from Remark 14.

(b) L is nonempty if and only if $\rho \geq 0$ using (a), and $\rho = (\lambda_2 + \lambda_1/\sqrt{n})(\lambda_2 - \lambda_1/\sqrt{n})$. Hence, with $\lambda_1, \lambda_2 > 0$ one obtains $\rho \geq 0$ if and only if $\lambda_2 - \lambda_1/\sqrt{n} \geq 0$. ∎

Hence $L \neq \emptyset$ by the requirement that $\lambda_2 \leq \lambda_1 \leq \sqrt{n}\lambda_2$. Further, the following observation follows immediately from Proposition 4 and $\|a\|_2 = \|b\|_2 = \lambda_2$ for all $a, b \in L$:

Remark 16 *For all $a, b \in L$ it is $\|a - b\|_2^2 = 2(\lambda_2^2 - \langle a, b \rangle)$, hence $\langle a, b \rangle = \lambda_2^2 - 1/2 \cdot \|a - b\|_2^2$.*

Therefore, on L the dot product is equal to the Euclidean norm up to an additive constant. Next consider projections onto L and note that the solution set with respect to D is not changed by this operation. The major arguments for this result have been taken over from Theis et al. (2005). Here, the statements from Lemma 15 have been incorporated and the resulting quadratic equation was solved explicitly, simplifying the original version of Theis et al. (2005).

Lemma 17 *Let $r \in H$ with $r \neq m$. Let $s := m + \delta(r - m)$ where $\delta := \sqrt{\rho}/\|r-m\|_2$. Then:*

(a) $\delta > 0$, $s \in L$, and $\|q - r\|_2^2 - \|s - r\|_2^2 = 1/\delta \cdot \|q - s\|_2^2$ for all $q \in L$.

(b) $s = \text{proj}_L(r)$.

(c) $\text{proj}_D(r) = \text{proj}_D(s)$.

Proof (a) First note that $\delta > 0$ because of $r \neq m$. Clearly $s = (1 - \delta)m + \delta r$. Further, $s \in H$ because of $e^T s = \lambda_1$, and $s \in L$ because of $\|s - m\|_2^2 = \rho$ and Lemma 15.

Let $q \in L$ be arbitrary. One obtains $\|q\|_2 = \|s\|_2$ with $q, s \in K$ and therefore application of Proposition 4 yields $\|q - r\|_2^2 - \|s - r\|_2^2 = 2\langle s - q, r \rangle$. With Remark 14 follows $\langle m, r \rangle = \lambda_1^2/n$ and $\|r\|_2^2 = \|r - m\|_2^2 + \lambda_1^2/n$. Hence, $\langle s, r \rangle = (1 - \delta)\langle m, r \rangle + \delta \langle r, r \rangle = \lambda_1^2/n + \delta \|r - m\|_2^2$. On the other hand, from $s - m = \delta(r - m)$ and hence $r = (1 - 1/\delta)m + 1/\delta \cdot s$, and using Remark 16 it follows that $\langle q, r \rangle = (1 - 1/\delta)\lambda_1^2/n + 1/\delta \cdot (\lambda_2^2 - 1/2 \cdot \|q - s\|_2^2)$. Therefore with $\delta\|r - m\|_2^2 = \rho/\delta$ one obtains

$$\langle s - q, r \rangle = \delta\|r - m\|_2^2 + 1/\delta \cdot \left(\lambda_1^2/n - \lambda_2^2 + 1/2 \cdot \|q - s\|_2^2\right) = \frac{1}{2\delta}\|q - s\|_2^2,$$

and the claim follows directly by substitution.

(b) Let $q \in L$, then (a) implies $\|q - r\|_2^2 - \|s - r\|_2^2 = 1/\delta \cdot \|q - s\|_2^2 \geq 0$ with equality if and only if $q = s$ because of $\|\cdot\|_2$ being positive definite. Thus s is the unique projection of r onto L.

(c) With (a) follows $\|q - s\|_2^2 - \|p - s\|_2^2 = \delta(\|q - r\|_2^2 - \|p - r\|_2^2)$ for all $p, q \in D$ because of $D \subseteq L$. For $\text{proj}_D(r) \subseteq \text{proj}_D(s)$, let $p \in \text{proj}_D(r)$ and $q \in D$. By definition $\|p - r\|_2^2 \leq \|q - r\|_2^2$, and thus $\|q - s\|_2^2 - \|p - s\|_2^2 \geq 0$ with $\delta > 0$, hence $p \in \text{proj}_D(s)$. For the converse, let $p \in \text{proj}_D(s)$ and $q \in D$. Similarly, $\|q - r\|_2^2 - \|p - r\|_2^2 = 1/\delta \cdot (\|q - s\|_2^2 - \|p - s\|_2^2) \geq 0$, thus $p \in \text{proj}_D(r)$. ∎

Lemma 17 does not hold when $r = m$, which forms a null set. In practice, however, this can occur when the input vector x for Algorithm 1 is poorly chosen, for example if all entries are equal. In this case, $\text{proj}_L(r) = L$, hence any point from L can be chosen for further processing.

Remark 18 *One possibility in the case $r = m$ would be to choose the point $s := \alpha \sum_{i=1}^{n-1} e_i + \beta e_n$ where $\alpha, \beta \in \mathbb{R}$ for $s \in \text{proj}_L(r)$, that is forcing the last entry to be unequal to the other ones. For satisfying $s \in L$, set $\alpha := \lambda_1/n + \sqrt{\rho}/\sqrt{n(n-1)}$ and $\beta := \lambda_1 - \alpha(n-1) = \lambda_1/n - \sqrt{\rho(n-1)}/\sqrt{n}$. This yields $\alpha - \beta = \sqrt{\rho}\left(1/\sqrt{n(n-1)} + \sqrt{n-1}/\sqrt{n}\right) > 0$, hence $\alpha \neq \beta$. This choice has the convenient side effect of s being sorted in descending order.*

Combining these properties of H and L, it can now be shown that projections onto D are invariant to affine-linear transformations with positive scaling:

Corollary 19 *Let $\alpha > 0$, $\beta \in \mathbb{R}$ and $x \in \mathbb{R}^n$. Then $\text{proj}_D(\alpha x + \beta e) = \text{proj}_D(x)$.*

Proof Let $\alpha > 0$, $\beta \in \mathbb{R}$ and $x \in \mathbb{R}^n$. With Lemma 13 and Lemma 17 it is enough to show that $\text{proj}_L(\text{proj}_H(\alpha x + \beta e)) = \text{proj}_L(\text{proj}_H(x))$. Let $\tilde{x} := \alpha x + \beta e$, $\tilde{r} := \text{proj}_H(\tilde{x})$ and $\tilde{s} := \text{proj}_L(\tilde{r})$. Lemma 13 and $e^T e = n$ yield $\tilde{r} = (\alpha x + \beta e) + 1/n \cdot (\lambda_1 - \alpha e^T x - \beta e^T e) e = \alpha x + 1/n \cdot (\lambda_1 - \alpha e^T x) e$. Hence $\tilde{r}$ is independent of β. Lemma 17 yields $\tilde{s} = m + \tilde{\delta}(\tilde{r} - m)$, where $\tilde{\delta} := \sqrt{\rho}/\|\tilde{r} - m\|_2$. Application of Proposition 4 yields

$$\|\tilde{r}\|_2^2 = \|\alpha x\|_2^2 + \left\| 1/n \cdot (\lambda_1 - \alpha e^T x) e \right\|_2^2 + 2 \left\langle \alpha x, \, 1/n \cdot (\lambda_1 - \alpha e^T x) e \right\rangle$$
$$= \alpha^2 \|x\|_2^2 + 1/n \cdot (\lambda_1 - \alpha e^T x)(\lambda_1 + \alpha e^T x) = \alpha^2 \|x\|_2^2 + 1/n \cdot (\lambda_1^2 - \alpha^2 (e^T x)^2),$$

and with Remark 14 follows $\|\tilde{r} - m\|_2^2 = \|\tilde{r}\|_2^2 - \lambda_1^2/n = \alpha^2 \left(\|x\|_2^2 - 1/n \cdot (e^T x)^2 \right)$. Let $r := \text{proj}_H(x)$ and $s := \text{proj}_L(r)$, then Lemma 13 and Lemma 17 imply $r = x + 1/n \cdot (\lambda_1 - e^T x) e$ and $s = m + \delta(r - m)$, where $\delta := \sqrt{\rho}/\|r - m\|_2$. Likewise $\|r - m\|_2^2 = \|x\|_2^2 - 1/n \cdot (e^T x)^2$, and hence $\delta/\tilde{\delta} = \|\tilde{r} - m\|_2/\|r - m\|_2 = \alpha$, where $\alpha > 0$ must hold. This yields

$$\tilde{s} = m + \tilde{\delta}(\tilde{r} - m) = m + \delta/\alpha \cdot (\alpha x + \lambda_1/n \cdot e - \alpha/n \cdot e^T x e - \lambda_1/n \cdot e) = m + \delta(x - 1/n \cdot e^T x e) = s,$$

which shows that the projection is invariant. ∎

Therefore, shifting and positive scaling of the argument of Algorithm 1 do not change the outcome. An overview of the steps carried out this far is given in Figure 8. Consider a point $x \in \mathbb{R}^n$ and $s := \text{proj}_L(\text{proj}_H(x))$. When $s \in \mathbb{R}_{\geq 0}^n$, then already $s \in D$ and hence $s \in \text{proj}_D(x)$. Therefore only situations in which $s \notin \mathbb{R}_{\geq 0}^n$ holds are relevant in the remainder of this discussion.

C.2 Simplex Geometry

The joint constraint of the target hyperplane H with non-negativity yields simplex C. The following definition is likewise to definitions from Chen and Ye (2011) and Michelot (1986):

Definition 20 *For $n \in \mathbb{N}$, $n \geq 1$, the set $\triangle^n := \{ \alpha \in \mathbb{R}_{\geq 0}^n \mid e^T \alpha = 1 \}$ is called* canonical n-simplex.

It is clear that $C = \mathbb{R}_{\geq 0}^n \cap H = \{ \lambda_1 \alpha \mid \alpha \in \triangle^n \}$ is a scaled canonical simplex. Further, for an index set $I \subseteq \{1, \ldots, n\}$ the set $C_I = \{ c \in C \mid c_i = 0 \text{ for all } i \notin I \}$ is a face of the simplex, which intrinsically possesses the structure of a simplex itself—although of reduced intrinsic dimensionality. Consider the following observation on the topology of C embedded in the subspace H:

Proposition 21 *Let $c = \lambda_1 \alpha \in C$ with $\alpha \in \triangle^n$. Then $c \in \partial C$ in the metric space $(H, \|\cdot\|_2)$ if and only if there is a $j \in \{1, \ldots, n\}$ with $\alpha_j = 0$.*
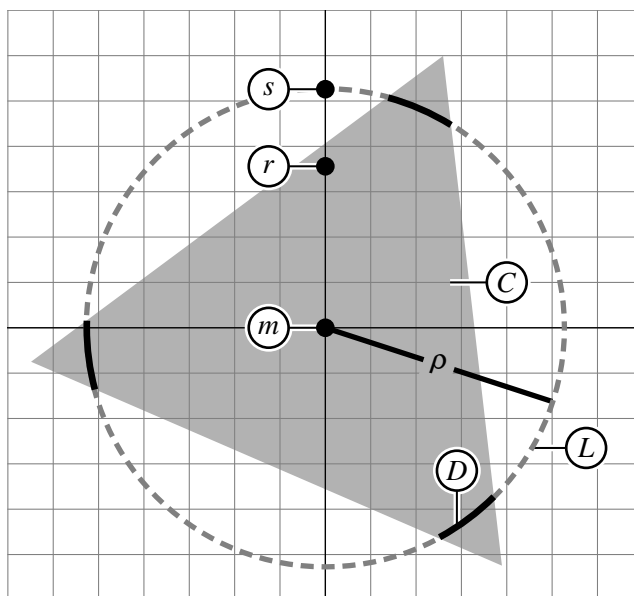
Figure 8: Sketch of the situation in Section C.1, projected onto target hyperplane H. r is the projection of the input point x onto H. s is the projection of r onto the hypercircle L, which has squared radius ρ. The intersection of H with the non-negative orthant is a simplex and denoted by C. The feasible set D is the intersection of C and L, and is marked with solid black lines. With Lemma 13 and Lemma 17 follows that $\mathrm{proj}_D(x) = \mathrm{proj}_D(r) = \mathrm{proj}_D(s)$, hence the next steps consist of projecting s onto C for finding the projection of x onto D.

The proof is simple and omitted as it does not contribute to deeper insight. Hence the faces C_I are subsets of ∂C, which is the topological border of C in $(H, \|\cdot\|_2)$. Using Proposition 21 a statement on the inradius of C can be made, which in turn can be used to show that for $n = 2$ no simplex projection has to be carried out at all:

Proposition 22 *The squared inradius of C is $\rho_{\mathrm{in}} := \frac{\lambda_1^2}{n(n-1)}$. It is $L \subseteq C$ for $n = 2$.*

Proof Because C is closed and convex, it is enough to consider the distance between interior points and boundary points. Hence the insphere radius of a point $p \in C$ can be computed as being the minimum distance to any of the boundary points. With Proposition 21 these points can be characterized as points where at least one entry vanishes. Using Lagrange multipliers it can be shown that $\min_{c \in \partial C} \|m - c\|_2^2 = \rho_{\mathrm{in}}$. Further, it can be shown that no point other than m is center of a larger insphere. This is achieved by constructing projections on certain faces of C, as is discussed in detail in the forthcoming Lemma 26. When $n = 2$ and $\lambda_2 \leq \lambda_1$, which is fulfilled by requirement on λ_1 and λ_2, then $\rho = \lambda_2^2 - \lambda_1^2/n \leq \lambda_1^2/2 = \rho_{\mathrm{in}}$, and $L \subseteq C$ follows with Lemma 15. ∎

The projection within H from outside a simplex is unique and must be located on its boundary:

Remark 23 *Let $s \in H \setminus C$. Then $\mathrm{proj}_C(s) \in \partial C$ in $(H, \|\cdot\|_2)$.*

The proof is obvious because C is closed and convex. By combination of Proposition 21 and Remark 23, it is now evident that the projection within H onto C yields vanishing entries. After the

first projection onto H, this subspace is never left throughout the arguments presented here, such that Remark 23 always applies. It has yet to be shown that the projection onto D possesses zero entries in the same coordinates. This way, a reduction of problem dimensionality can be achieved, and an iterative algorithm can be constructed to compute the projection onto D. The algorithm is guaranteed to terminate at the latest when the problem dimensionality equals two with Proposition 22.

C.2.1 PROJECTION ONTO A SIMPLEX

Quite a few methods have been proposed for carrying out projections onto canonical simplexes. An iterative algorithm was developed by Michelot (1986) which is very similar to Hoyer's original method for computation of the projection onto D. A simpler and more effective algorithm has been developed by Duchi et al. (2008). Building upon this work, Chen and Ye (2011) have proposed and rigorously proved correctness of a very similar algorithm, which is more explicit than that of Duchi et al. (2008). Their algorithm can be adapted to better suit the needs for the sparseness-enforcing projection. This adapted version was given by Algorithm 2 in Section 2. The following note makes the adaptations explicit.

Proposition 24 *Let $x \in \mathbb{R}^n \setminus C$ and $p := \mathrm{proj}_C(x)$. Then the following holds:*

(a) There exists $\hat{t} \in \mathbb{R}$ such that $p = \max(x - \hat{t} \cdot e, \, 0)$, where the maximum is taken element-wise.

(b) Algorithm 2 computes $\hat{t}$ such that (a) holds and the number of nonzero entries in p.

Proof The arguments from Chen and Ye (2011) hold for projections onto $\triangle^n$. The case of the scaled canonical simplex can be recovered using $p = \lambda_1 \cdot \mathrm{proj}_{\triangle^n}(x/\lambda_1)$. Therefore lines 4 and 7 of Algorithm 2 can be adapted from $t := \frac{s-1}{i}$ and $t := \frac{s-1}{n}$ to $t := \frac{s-\lambda_1}{i}$ and $t := \frac{s-\lambda_1}{n}$, respectively. The correct number of nonzero entries in p follows immediately from its expression from (a), the fact that y is sorted in descending order and the termination criterion of Algorithm 2. ∎

As already described in Section 2.2, symmetries can be exploited for projections onto C:

Remark 25 *When x is already sorted in descending order, then no sorting is needed at the beginning of Algorithm 2. The projection p is then sorted also, because C is permutation-invariant. In this case, the nonzero entries of p are located in the first $d := \|p\|_0$ entries, while $p_{d+1} = \cdots = p_n = 0$.*

This fact is useful for optimizing access to the relevant entries of the working vector, which can then be stored contiguously in memory.

C.2.2 PROJECTION ONTO A FACE OF A SIMPLEX

The projection within H onto C yields zero entries in the working vector. It still remains to be shown that the projection onto D possesses zero entries at the same coordinates as the projection onto C. If this holds true, then the dimensionality of the original problem can be reduced, and iterative arguments can be applied. The main building block in the proof is the explicit construction of projections from within the simplex onto a certain face. The next Lemma is fundamental for proving correctness of Algorithm 1. It describes the construction of the result of the projection onto a simplex face and poses a statement on its norm, which in turn is used to prove that the position of vanishing entries does not change upon projection.

Lemma 26 *Let $q \in C$ and let $\varnothing \neq I \subseteq \{1,\ldots,n\}$ be an arbitrary index set. Then there exists an $s \in C_I$ with $\|q - v\|_2^2 = \|q - s\|_2^2 + \|s - v\|_2^2$ for all $v \in C_I$. If additionally $\max_{j \in J} q_j \leq \min_{i \in I} q_i$ holds for $J := I^C$, then $\|s\|_2 \geq \|q\|_2$ with equality if and only if $q_j = 0$ for all $j \in J$.*

More precisely, let $h := |J|$ and let $J = \{j_1,\ldots,j_h\}$ such that $q_{j_1} \leq \cdots \leq q_{j_h}$. Consider the sequence $s^{(0)},\ldots,s^{(h)} \in \mathbb{R}^n$ defined iteratively by $s^{(0)} := q$ and

$$s^{(k)} := s^{(k-1)} - s_{j_k}^{(k-1)} e_{j_k} + \tfrac{1}{n-k} s_{j_k}^{(k-1)} \left(e - \textstyle\sum_{i=1}^{k} e_{j_i} \right)$$

for $k \in \{1,\ldots,h\}$. Write $s := s^{(h)}$. Then the following holds:

  (a) $s^{(k)} \in C_{\{j_1,\ldots,j_k\}^C}$ *for all $k \in \{1,\ldots,h\}$.*

  (b) $\left\langle s^{(0)} - s^{(k)},\ s^{(k)} - s^{(k+1)} \right\rangle = 0$ *for all $k \in \{0,\ldots,h-1\}$.*

  (c) $\left\|s^{(k)} - q\right\|_2^2 = \sum_{i=1}^{k} \left\|s^{(i)} - s^{(i-1)}\right\|_2^2$ *for all $k \in \{0,\ldots,h\}$.*

  (d) $s_{j_k}^{(k-1)} = q_{j_k} + \frac{1}{n-k+1} \sum_{i=1}^{k-1} q_{j_i}$ *for all $k \in \{1,\ldots,h\}$.*

  (e) $\left\langle s^{(0)} - s^{(k)},\ s^{(k)} - v \right\rangle = 0$ *for all $k \in \{0,\ldots,h\}$ and for all $v \in C_I$.*

  (f) $\|q - v\|_2^2 = \left\|q - s^{(k)}\right\|_2^2 + \left\|s^{(k)} - v\right\|_2^2$ *for all $k \in \{0,\ldots,h\}$ and for all $v \in C_I$.*

  (g) $s = \mathrm{proj}_{C_I}(q)$.

If $\max_{j \in J} q_j \leq \min_{i \in I} q_i$, then the following holds as well:

  (h) $s_{j_1}^{(k)} \leq \cdots \leq s_{j_h}^{(k)} \leq \min_{i \in I} s_i^{(k)}$ *for all $k \in \{0,\ldots,h\}$.*

  (i) $s_{j_k}^{(k-1)} \leq \frac{\lambda_1}{n-k+1}$ *for all $k \in \{1,\ldots,h\}$.*

  (j) $\left\|s^{(k-1)}\right\|_2 \leq \left\|s^{(k)}\right\|_2$ *for all $k \in \{1,\ldots,h\}$, and hence $\|s\|_2 \geq \|q\|_2$.*

  (k) $\|s\|_2 = \|q\|_2$ *if and only if $q_j = 0$ for all $j \in J$.*

Proof In other words, $s^{(k)}$ is constructed from $s^{(k-1)}$ by setting entry j_k to zero, and adjusting all remaining entries, but the ones previously set to zero, such that the L_1 norm is preserved. This generates a finite series of points progressively approaching C_I, see (a), where the final point is from C_I. As all relevant dot products vanish, see (b) and (e), this is a process of orthogonal projections. Hence the distance between points can be computed using the Pythagorean theorem, see (c) and (f). In (g) it is then shown that s is the unique projection of q onto C_I.

If the entry j_k in $s^{(k-1)}$ does not vanish, then the L_2 norm of the newly constructed point is greater than that of the original point, see (j) and (k). The entries with indices from J must be sufficiently small for this non-decreasing norm property to hold, see (h) and (i). The magnitude of these entries, however, is strongly connected with the magnitudes of respective entries from the original point q, that is, the rank is preserved from one point to its successor. Figure 9 gives an example for $n = 3$ in which cases the non-decreasing norm property holds.

Let $a_k := \frac{1}{n-k} \left(e - \sum_{i=1}^{k} e_{j_i} \right) - e_{j_k} \in \mathbb{R}^n$ for $k \in \{1,\ldots,h\}$. Then $s^{(k)} = s^{(k-1)} + s_{j_k}^{(k-1)} a_k$, and with induction follows $s^{(k)} = s^{(0)} + \sum_{i=1}^{k} s_{j_i}^{(i-1)} a_i$ for $k \in \{1,\ldots,h\}$.

(a) First note that $e^T a_k = \frac{1}{n-k}(n-k) - 1 = 0$ and that $a_k \in \mathbb{R}^n_{\geq 0}$ for all $k \in \{1,\ldots,h\}$. It is now shown by induction that $s^{(k)}$ lies on the claimed face of C. For $k = 1$, one obtains $s^{(1)}_{j_1} = 0$ and $s^{(1)}_i = q_i + \frac{1}{n-1}q_{j_1}$ for $i \neq j_1$. Thus $s^{(1)}_i \geq 0$ for all $i \in \{1,\ldots,n\}$ because of $q_i \geq 0$ for all $i \in \{1,\ldots,n\}$. Further $e^T s^{(1)} = e^T q + q_{j_1} e^T a_1 = e^T q = \lambda_1$, hence $s^{(1)} \in C_{\{1,\ldots,n\}\setminus\{j_1\}}$.

For $k - 1 \to k$, assume $s^{(k-1)}_i = 0$ for all $i \in \{j_1,\ldots,j_{k-1}\}$, $s^{(k-1)}_i \geq 0$ for all $i \in \{1,\ldots,n\}$ and $e^T s^{(k-1)} = \lambda_1$. Clearly, $s^{(k)}_i = s^{(k-1)}_i = 0$ holds for $i \in \{j_1,\ldots,j_{k-1}\}$. Furthermore, one obtains $s^{(k)}_{j_k} = s^{(k-1)}_{j_k} - s^{(k-1)}_{j_k} = 0$. With $s^{(k-1)} \in \mathbb{R}^n_{\geq 0}$ and $a_k \in \mathbb{R}^n_{\geq 0}$ follows that $s^{(k)} \in \mathbb{R}^n_{\geq 0}$. Finally it is $e^T s^{(k)} = e^T s^{(k-1)} + s^{(k-1)}_{j_k} e^T a_k = e^T s^{(k-1)} = \lambda_1$. Hence $s^{(k)} \in C_{\{1,\ldots,n\}\setminus\{j_1,\ldots,j_k\}}$.

(b) For $i \in \{1,\ldots,k\}$ follows

$$\left\langle e - \sum_{\mu=1}^{i} e_{j_\mu}, \; e - \sum_{\nu=1}^{k+1} e_{j_\nu} \right\rangle = \langle e, \, e \rangle - \left\langle e, \, \sum_{\nu=1}^{k+1} e_{j_\nu} \right\rangle - \left\langle \sum_{\mu=1}^{i} e_{j_\mu}, \, e \right\rangle + \left\langle \sum_{\mu=1}^{i} e_{j_\mu}, \, \sum_{\nu=1}^{k+1} e_{j_\nu} \right\rangle$$

$$= n - (k+1) - i + i = n - k - 1,$$

and therefore

$$\langle a_i, \, a_{k+1} \rangle = \frac{1}{(n-i)(n-k-1)} \left\langle e - \sum_{\mu=1}^{i} e_{j_\mu}, \; e - \sum_{\nu=1}^{k+1} e_{j_\nu} \right\rangle + \left\langle e_{j_i}, \, e_{j_{k+1}} \right\rangle$$

$$- \frac{1}{n-i} \left\langle e - \sum_{\mu=1}^{i} e_{j_\mu}, \, e_{j_{k+1}} \right\rangle - \frac{1}{n-k-1} \left\langle e_{j_i}, \, e - \sum_{\nu=1}^{k+1} e_{j_\nu} \right\rangle$$

$$= \frac{n-k-1}{(n-i)(n-k-1)} + 0 - \frac{1}{n-i} - \frac{0}{n-k-1} = 0.$$

Thus

$$\left\langle s^{(0)} - s^{(k)}, \; s^{(k)} - s^{(k+1)} \right\rangle = \left\langle \sum_{i=1}^{k} s^{(i-1)}_{j_i} a_i, \; s^{(k)}_{j_{k+1}} a_{k+1} \right\rangle = \sum_{i=1}^{k} s^{(i-1)}_{j_i} s^{(k)}_{j_{k+1}} \langle a_i, \, a_{k+1} \rangle = 0.$$

(c) Follows by induction using Proposition 4 and (b).

(d) Clearly, $\langle e_{j_k}, \, a_i \rangle = \frac{1}{n-i}$ for $i \in \{1,\ldots,k-1\}$, hence with induction follows

$$s^{(k-1)}_{j_k} = \left\langle e_{j_k}, \, s^{(k-1)} \right\rangle = \left\langle e_{j_k}, \, s^{(0)} \right\rangle + \sum_{i=1}^{k-1} s^{(i-1)}_{j_i} \left\langle e_{j_k}, \, a_i \right\rangle = q_{j_k} + \sum_{i=1}^{k-1} \frac{1}{n-i} s^{(i-1)}_{j_i}$$

$$\overset{\text{IH}}{=} q_{j_k} + \sum_{i=1}^{k-1} \frac{1}{n-i} q_{j_i} + \sum_{i=1}^{k-1} \sum_{\mu=1}^{i-1} \frac{1}{n-i} \frac{1}{n-i+1} q_{j_\mu} = q_{j_k} + \sum_{i=1}^{k-1} q_{j_i} \left[\frac{1}{n-i} + \sum_{\mu=i+1}^{k-1} \frac{1}{n-\mu} \frac{1}{n-\mu+1} \right].$$

Using $\sum_{i=1}^{k-1} \sum_{\mu=1}^{i-1} = \sum_{1 \leq \mu < i \leq k-1} = \sum_{\mu=1}^{k-1} \sum_{i=\mu+1}^{k-1}$ the order of summation was changed after the induction step, and then the variables i and μ were swapped. For the claim to hold it is enough to show that $\frac{1}{n-i} + \sum_{\mu=i+1}^{k-1} \frac{1}{n-\mu} \frac{1}{n-\mu+1} = \frac{1}{n-k+1}$ for all $i \in \{1,\ldots,k-1\}$, which follows by reverse induction.

(e) Proof by induction. Let $v \in C_I$, that is $e^T v = \lambda_1$ and $v_j = 0$ for all $j \in J$. For $k = 0$, $s^{(0)} - s^{(k)} = 0$ and the claim follows. For $k - 1 \to k$, first note that $\langle a_k, \, q \rangle = \frac{1}{n-k}\left(\lambda_1 - \sum_{i=1}^{k} q_{j_i}\right) - q_{j_k}$, $\langle a_k, \, s^{(k-1)} \rangle = \langle a_k, \, s^{(0)} \rangle + \sum_{i=1}^{k-1} s^{(i-1)}_{j_i} \langle a_k, \, a_i \rangle = \langle a_k, \, q \rangle$ because $\langle a_k, \, a_i \rangle = 0$ for all $i \in \{1,\ldots,k-1\}$ as shown in (b), thus $\langle a_k, \, s^{(0)} - 2s^{(k-1)} \rangle = -\langle a_k, \, q \rangle$. Furthermore $\langle a_k, \, v \rangle = \frac{\lambda_1}{n-k}$ because $\langle e_{j_i}, \, v \rangle = v_{j_i} = 0$ for all $i \in \{1,\ldots,h\}$, and $\langle a_k, \, a_k \rangle = \|a_k\|_2^2 = \frac{1}{(n-k)^2}\left\|e - \sum_{i=1}^{k} e_{j_i}\right\|_2^2 + \|e_{j_k}\|_2^2 = 1 + \frac{1}{n-k}$.
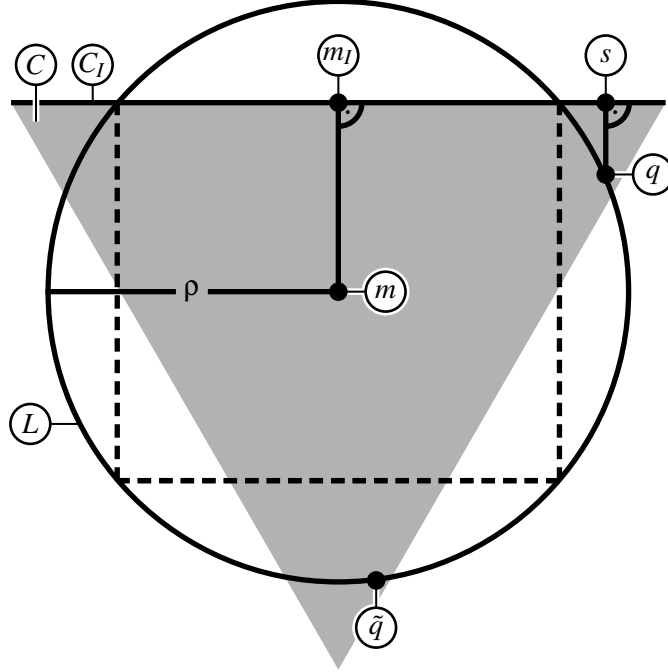
Figure 9: Situation of Lemma 26: The projection of a point q from within the simplex C onto one of its faces C_I yields point s. When q is sufficiently close to C_I, then $\|s\|_2 \geq \|q\|_2$. This does not hold when the point projected onto C_I is too far away. For example, the point $\tilde{q}$ which is located outside the dashed square with edge length $2\|m_I - m\|_2$ would yield a projection with a smaller Euclidean norm.

Therefore,

$$
\begin{aligned}
\left\langle s^{(0)} - s^{(k)},\ s^{(k)} - v \right\rangle &= \left\langle s^{(0)} - s^{(k-1)} - s_{j_k}^{(k-1)} a_k,\ s^{(k-1)} + s_{j_k}^{(k-1)} a_k - v \right\rangle \\
&= \left\langle s^{(0)} - s^{(k-1)},\ s^{(k-1)} - v \right\rangle + s_{j_k}^{(k-1)} \left\langle a_k,\ s^{(0)} - 2s^{(k-1)} - s_{j_k}^{(k-1)} a_k + v \right\rangle \\
&\overset{\text{IH}}{=} s_{j_k}^{(k-1)} \left(-\langle a_k,\ q \rangle - s_{j_k}^{(k-1)} \langle a_k,\ a_k \rangle + \langle a_k,\ v \rangle \right) \\
&= s_{j_k}^{(k-1)} \left(-\tfrac{\lambda_1}{n-k} + \tfrac{1}{n-k}\textstyle\sum_{i=1}^{k} q_{j_i} + q_{j_k} - s_{j_k}^{(k-1)} \left(1 + \tfrac{1}{n-k}\right) + \tfrac{\lambda_1}{n-k} \right) \\
&= s_{j_k}^{(k-1)} \left(q_{j_k}\left(1 + \tfrac{1}{n-k}\right) + \tfrac{1}{n-k}\textstyle\sum_{i=1}^{k-1} q_{j_i} - s_{j_k}^{(k-1)}\left(1 + \tfrac{1}{n-k}\right) \right) = 0,
\end{aligned}
$$

where the final equality was yielded using the statement from (d) and $\left(1 + \tfrac{1}{n-k}\right) \cdot \tfrac{1}{n-k+1} = \tfrac{1}{n-k}$.

(f) Follows using Proposition 4 and (e).

(g) A unique Euclidean projection exists because C_I is closed and convex. $s \in C_I$ with (a), and $\|q - s\|_2^2 \leq \|q - v\|_2^2$ for all $v \in C_I$ with (f). Therefore $s = \text{proj}_{C_I}(q)$.

(h) In the remainder of the proof assume that $\max_{j \in J} q_j \leq \min_{i \in I} q_i$ holds. It is first shown by induction that $s_{j_1}^{(k)} \leq \cdots \leq s_{j_h}^{(k)}$ for all $k \in \{0, \ldots, h\}$. For $k = 0$ this is fulfilled as requirement on q and by definition of J. For $k - 1 \to k$, let $\mu, \nu \in \{j_1, \ldots, j_h\}$ with $\mu < \nu$. Then with χ denoting the

indicator function and with $A := \chi_{\{j_k\}}(\mu) - \chi_{\{j_k\}}(\nu) + \frac{1}{n-k}\left(\chi_{\{j_1,\ldots,j_k\}^C}(\nu) - \chi_{\{j_1,\ldots,j_k\}^C}(\mu)\right)$ follows $s_\nu^{(k)} - s_\mu^{(k)} = s_\nu^{(k-1)} - s_\mu^{(k-1)} + s_{j_k}^{(k-1)}A$. Clearly, when $A \geq 0$ then the claim follows with the induction hypothesis and with $s_{j_k}^{(k-1)} \geq 0$ due to (a).

First consider the case of $\mu \in \{j_1,\ldots,j_{k-1}\}$. If $\nu \in \{j_1,\ldots,j_{k-1}\}$ also, then $A = 0$. If $\nu = j_k$, then $A = -1$, and hence $s_\nu^{(k)} - s_\mu^{(k)} = s_\nu^{(k-1)} - s_\mu^{(k-1)} - s_\nu^{(k-1)} = -s_\mu^{(k-1)}$ which however vanishes with (a). If $\nu \in \{j_{k+1},\ldots,j_h\}$, then $A = \frac{1}{n-k} \geq 0$. If $\mu = j_k$, then $\nu \in \{j_{k+1},\ldots,j_h\}$, and then $A = 1 + \frac{1}{n-k} \geq 0$. If $\mu \in \{j_{k+1},\ldots,j_h\}$, then $\nu \in \{j_{\mu+1},\ldots,j_h\}$, thus $A = 0$. Hence the first claim is always fulfilled.

Next, it is shown that $\max_{j \in J} s_j^{(k)} \leq \min_{i \in I} s_i^{(k)}$ for all $k \in \{0,\ldots,h\}$. For $k = 0$ this is the requirement on q. For $k-1 \to k$, let $i \in I$ and $j \in J$. It is then $\chi_{\{j_k\}}(i) = 0$, $\chi_{\{j_k\}}(j) \in \{0,1\}$, $\chi_{\{j_1,\ldots,j_k\}^C}(i) = 1$ and $\chi_{\{j_1,\ldots,j_k\}^C}(j) = 0$, therefore

$$
\begin{aligned}
s_i^{(k)} - s_j^{(k)} = \quad & s_i^{(k-1)} - s_{j_k}^{(k-1)}\chi_{\{j_k\}}(i) + \frac{1}{n-k}s_{j_k}^{(k-1)}\chi_{\{j_1,\ldots,j_k\}^C}(i) \\
& - s_j^{(k-1)} + s_{j_k}^{(k-1)}\chi_{\{j_k\}}(j) - \frac{1}{n-k}s_{j_k}^{(k-1)}\chi_{\{j_1,\ldots,j_k\}^C}(j) \\
= \quad & s_i^{(k-1)} - s_j^{(k-1)} + s_{j_k}^{(k-1)}\left(\frac{1}{n-k} + \chi_{\{j_k\}}(j)\right) \geq 0,
\end{aligned}
$$

where $s_i^{(k-1)} - s_j^{(k-1)} \geq 0$ holds by induction hypothesis.

(i) With (a) follows $\lambda_1 = e^T s^{(k-1)}$ and $s_{j_i}^{(k-1)} = 0$ for all $i \in \{1,\ldots,k-1\}$. $s_i^{(k-1)} \geq s_{j_k}^{(k-1)}$ holds for all $i \in I$ with (h), and $s_{j_i}^{(k-1)} \geq s_{j_k}^{(k-1)}$ for all $i \in \{k+1,\ldots,h\}$. Therefore,

$$
\begin{aligned}
\lambda_1 = \sum_{i=1}^n s_i^{(k-1)} &= \sum_{i \in I} s_i^{(k-1)} + \sum_{i=1}^{k-1} s_{j_i}^{(k-1)} + s_{j_k}^{(k-1)} + \sum_{i=k+1}^h s_{j_i}^{(k-1)} \\
&\geq \left((n-h) + 1 + (h-k)\right)s_{j_k}^{(k-1)} = (n-k+1)s_{j_k}^{(k-1)},
\end{aligned}
$$

and the claim follows because $n - k + 1 > 0$.

(j) In (e) it was shown that $\|a_k\|_2^2 = 1 + \frac{1}{n-k}$. Furthermore,

$$
\langle s^{(k-1)}, a_k \rangle = \frac{1}{n-k}\left(\lambda_1 - \sum_{i=1}^{k-1} s_{j_i}^{(k-1)} - s_{j_k}^{(k-1)}\right) - s_{j_k}^{(k-1)} = \frac{1}{n-k}\left(\lambda_1 - s_{j_k}^{(k-1)}\right) - s_{j_k}^{(k-1)},
$$

because all $s_{j_i}^{(k-1)}$ vanish for $i \in \{1,\ldots,k-1\}$ using (a). Application of Proposition 4 yields

$$
\begin{aligned}
\left\|s^{(k)}\right\|_2^2 - \left\|s^{(k-1)}\right\|_2^2 &= \left\|s_{j_k}^{(k-1)}a_k\right\|_2^2 + 2s_{j_k}^{(k-1)}\langle s^{(k-1)}, a_k \rangle \\
&= s_{j_k}^{(k-1)}\left[s_{j_k}^{(k-1)}\left(1 + \frac{1}{n-k}\right) + \frac{2}{n-k}\left(\lambda_1 - s_{j_k}^{(k-1)}\right) - 2s_{j_k}^{(k-1)}\right] \\
&= s_{j_k}^{(k-1)}\left[\frac{2\lambda_1}{n-k} - s_{j_k}^{(k-1)}\left(1 + \frac{1}{n-k}\right)\right],
\end{aligned}
$$

which is non-negative when $s_{j_k}^{(k-1)} \leq \left(1 + \frac{1}{n-k}\right)^{-1} \cdot \frac{2\lambda_1}{n-k} = \frac{2\lambda_1}{n-k+1}$. With (i) this is always fulfilled, hence $\left\|s^{(k-1)}\right\|_2 \leq \left\|s^{(k)}\right\|_2$, and $\|s\|_2 \geq \|q\|_2$ follows immediately using a telescoping sum argument.

(k) When $q_j = 0$ for all $j \in J$, then $s = q$ and the claim follows. When there is a $j_k \in \{j_1,\ldots,j_h\}$ with $q_{j_k} \neq 0$, let k be minimal such that either $k = 1$ or $q_{j_{k-1}} = 0$, hence $s_{j_k}^{(k-1)} = q_{j_k}$. With (i) follows $0 < s_{j_k}^{(k-1)} \leq \frac{\lambda_1}{n-k+1} < \frac{2\lambda_1}{n-k+1}$, and hence $\left\|s^{(k)}\right\|_2^2 - \left\|s^{(k-1)}\right\|_2^2 > 0$ with (j), thus $\|s\|_2 > \|q\|_2$. ∎
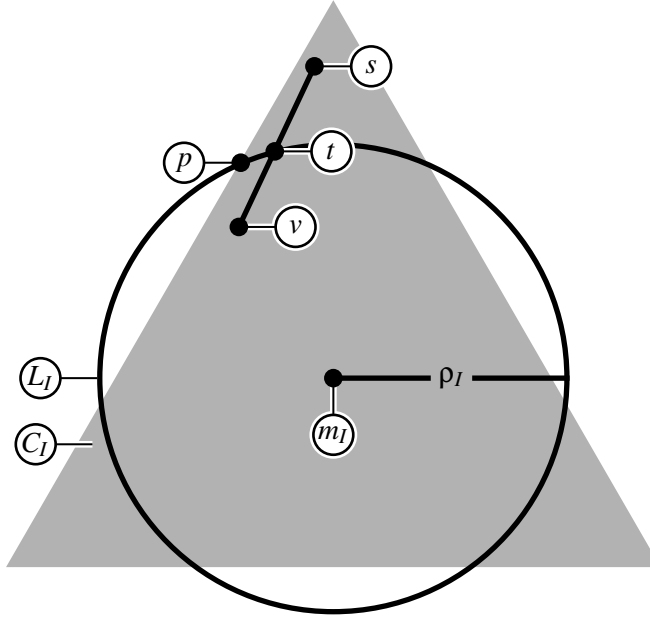
Figure 10: Sketch of the proof of Corollary 27: The projection of v onto D must be located on simplex face C_I. Assume there is a projection $q \notin C_I$, then it must be sufficiently close to C_I for application of Lemma 26. The projection s of q onto C_I is located outside $L_I = \{a \in L \mid a_i = 0 \text{ for all } i \notin I\}$. Hence the intersection t of the line between v and s and L_I is in C_I due to its convexity, and it is farther than the projection p of v onto $\tilde{D} = \{a \in D \mid a_i = 0 \text{ for all } i \notin I\}$. Therefore q cannot be the projection of v onto D.

The application of Lemma 26 then shows that the projection of a point from a face C_I of C onto D must reside on the same face C_I, given the original point is located within a sphere with squared radius ρ_I around m_I. As will be shown in Lemma 28, this is automatically fulfilled for projections from L onto C.

Corollary 27 *Let $I \subseteq \{1, \ldots, n\}$, let $v \in C_I$ with $\|v\|_2 < \lambda_2$, and let $q \in \mathrm{proj}_D(v)$. Then $q \in C_I$.*

Proof Let $J := \{1, \ldots, n\} \setminus I$, and let $q \in \mathrm{proj}_D(v)$. Assume there is at least one $j \in J$ with $q_j \neq 0$. For showing $\max_{j \in J} q_j \leq \min_{i \in I} q_i$, assume there are $i \in I$ and $j \in J$ with $q_j > q_i$. Then $v_i > 0$ and $v_j = 0$ because of $v \in C_I$. D is permutation-invariant using Remark 7 as intersection of permutation-invariant sets. Hence let $\tau := (i, j) \in S_n$ be the transposition swapping i and j, and consider

$$d := \|q - v\|_2^2 - \|P_\tau q - v\|_2^2 = 2(q_j - q_i)(v_i - v_j).$$

It is $d > 0$ because of $q_j - q_i > 0$ and $v_i - v_j = v_i > 0$. Hence $\|P_\tau q - v\|_2 < \|q - v\|_2$ and $P_\tau q \in D$, which violates the minimality of q. Therefore, $\max_{j \in J} q_j \leq \min_{i \in I} q_i$ must hold.

A drawing for the next arguments is given in Figure 10. With Lemma 26 there is an $s \in C_I$ such that $\|q - v\|_2^2 = \|q - s\|_2^2 + \|s - v\|_2^2$ and $\|s\|_2 > \lambda_2$. Consider $f \colon [0, 1] \to \mathbb{R}, \beta \mapsto \|v + \beta(s - v)\|_2$. Clearly $f(0) = \|v\|_2 < \lambda_2$ and $f(1) = \|s\|_2 > \lambda_2$, hence with the intermediate value theorem there
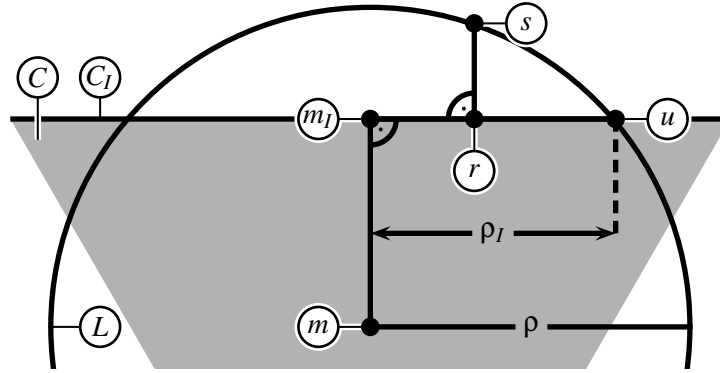
Figure 11: Situation of Lemma 28 and Lemma 30: The point s is projected onto simplex C yielding r, which resides on one of its faces C_I. From there point u can be constructed by projecting within C_I onto the target hypersphere.

exists a $\beta^* \in (0,\ 1)$ with $f(\beta^*) = \lambda_2$. Let $t := v + \beta^* (s-v) \in \mathbb{R}^n$, which lies in C_I because of $v, s \in C_I$ and C_I is convex. By construction $\|t\|_2 = \lambda_2$, hence $t \in \tilde{D} := \{a \in D \mid a_i = 0 \text{ for all } i \notin I\}$. Clearly, $\|v-t\|_2 + \|t-s\|_2 = |\beta^*| \cdot \|s-v\|_2 + |1-\beta^*| \cdot \|v-s\|_2 = \|s-v\|_2$. Let $p \in \text{proj}_{\tilde{D}}(v)$, then $\|v-p\|_2 \le \|v-t\|_2$. Therefore,

$$\|q-v\|_2^2 = \|q-s\|_2^2 + \|s-v\|_2^2 = \|q-s\|_2^2 + (\|v-t\|_2 + \|t-s\|_2)^2 > \|v-t\|_2^2 \ge \|v-p\|_2^2.$$

Because of $\tilde{D} \subseteq D$, $p \in D$ also, hence $\|q-v\|_2 \le \|p-v\|_2$, which contradicts $\|q-v\|_2 > \|v-p\|_2$. Hence, $q_j = 0$ for all $j \in J$ must hold, and thus $q \in C_I$. ∎

Because C_I is isomorphic to a simplex, but with lower dimensionality than C, an algorithm can be constructed to compute the projection onto D, as discussed in the following.

C.3 Self-Similarity of the Feasible Set

The next Lemma summarizes previous results and analyzes projections from L onto C in greater detail. It shows that the solution set with respect to the projection onto D is not tampered, and that all solutions have zeros at the same positions as the projection onto C. Figure 11 provides orientation on the quantities discussed in Lemma 28.

Lemma 28 *Let $s \in L \setminus C$ and $r \in \text{proj}_C(s)$. Let $I := \{i \in \{1,\dots,n\} \mid r_i \ne 0\}$ and $d := |I|$. Then:*

(a) *There exists $\hat{t} \in \mathbb{R}_{\ge 0}$ such that $r = \max(s - \hat{t} \cdot e,\ 0)$, with the maximum taken element-wise.*

(b) *$I \ne \varnothing$ and $I \ne \{1,\dots,n\}$.*

(c) *$s_i > \hat{t}$ and $s_i - r_i = \hat{t}$ for all $i \in I$. $s_i \le \hat{t}$ and $s_i - r_i = s_i$ for all $i \notin I$.*

(d) *$\langle r,\ s-r \rangle = \lambda_1 \hat{t}$.*

(e) *$\langle m_I,\ s-r \rangle = \lambda_1 \hat{t}$, thus $\langle m_I - r,\ s-r \rangle = 0$.*

(f) *$\text{proj}_D(r) \subseteq C_I$, hence from $q \in \text{proj}_D(r)$ follows $q_i = 0$ for all $i \notin I$.*

(g) Let $q \in \text{proj}_D(r)$. Then $\langle q - r, \ s - r \rangle = 0$, and thus $\|q - s\|_2^2 = \|q - r\|_2^2 + \|r - s\|_2^2$.

(h) $\text{proj}_D(r) = \text{proj}_D(s) \subseteq C_I$.

Proof (a) The existence of $\hat{t} \in \mathbb{R}$ such that $r = \max(s - \hat{t} \cdot e, \ 0)$ is guaranteed by Proposition 24. It remains to be shown that $\hat{t} \geq 0$. Consider the index set $J := \{j \in \{1, \ldots, n\} \mid s_j \geq 0\}$ of non-negative entries of s. Because of $s \in L \setminus C$ one obtains $e^T s = \lambda_1$ and there is an index i with $s_i < 0$, hence $J \neq \{1, \ldots, n\}$. Therefore, $\lambda_1 = \sum_{j \in J} s_j + \sum_{j \notin J} s_j < \sum_{j \in J} s_j$. Now assume $\hat{t} < 0$, then $s_j - \hat{t} > 0$ for all $j \in J$, and hence $r_j = s_j - \hat{t}$ for all $j \in J$. Using $r \in C$ yields

$$\lambda_1 = \sum_{j=1}^n r_j \geq \sum_{j \in J} r_j = \sum_{j \in J} (s_j - \hat{t}) = \sum_{j \in J} s_j - |J| \cdot \hat{t} > \sum_{j \in J} s_j,$$

which contradicts $\sum_{j \in J} s_j > \lambda_1$. Hence $\hat{t} \geq 0$ must hold.

(b) Would $I = \varnothing$ hold, then $r = 0$, which is impossible because of $r \in C$. $I = \{1, \ldots, n\}$ would violate the existence of vanishing entries in r, as is guaranteed by Remark 23.

(c) Using (a): When $i \in I$, then $0 < r_i = s_i - \hat{t}$ and the claim follows. When $i \notin I$, then $r_i = 0$, hence $s_i - \hat{t} \leq 0$ and the claim follows.

(d) $e^T r = e^T s = \lambda_1$ because $r, s \in H$, hence using (c) yields

$$0 = \langle e, \ s - r \rangle = \sum_{i \in I} (s_i - r_i) + \sum_{i \notin I} (s_i - r_i) = \sum_{i \in I} \hat{t} + \sum_{i \notin I} s_i = d\hat{t} + \lambda_1 - \sum_{i \in I} s_i,$$

thus

$$\langle r, \ s - r \rangle = \sum_{i \in I} r_i (s_i - r_i) = \sum_{i \in I} (s_i - \hat{t})\hat{t} = \hat{t}\left(\sum_{i \in I} s_i - d\hat{t}\right) = \lambda_1 \hat{t}.$$

(e) $\langle m_I, \ s - r \rangle = \lambda_1 / d \cdot \sum_{i \in I} (s_i - r_i) = \lambda_1 / d \cdot \sum_{i \in I} \hat{t} = \lambda_1 \hat{t}$ with (c), and the claim follows with (d).

(f) $r \in C_I$ by definition of I. Using this, (c) and $\hat{t} \geq 0$ from (a) yields

$$\lambda_2^2 = \|s\|_2^2 = \sum_{i \in I} s_i^2 + \sum_{i \notin I} s_i^2 = \sum_{i \in I} (r_i + \hat{t})^2 + \sum_{i \notin I} s_i^2$$
$$> \sum_{i \in I} r_i^2 + d\hat{t}^2 + 2\hat{t}\sum_{i \in I} r_i = \|r\|_2^2 + d\hat{t}^2 + 2\lambda_1 \hat{t} \geq \|r\|_2^2,$$

thus the claim holds using Corollary 27.

(g) With (f) follows $q \in C_I$. Hence $\langle q, \ s - r \rangle = \sum_{i \in I} q_i (s_i - r_i) = \hat{t}\sum_{i \in I} q_i = \lambda_1 \hat{t}$ with (c), and the claims follow with (d) and Proposition 4.

(h) Let $p \in \text{proj}_D(s)$ and $q \in \text{proj}_D(r)$. Then using (g) and Proposition 4 one obtains that $\|q - s\|_2^2 - \|p - s\|_2^2 = \|q - r\|_2^2 - \|p - r\|_2^2 + 2 \langle p - r, \ s - r \rangle$. With (c) and (d) follows

$$\langle p - r, \ s - r \rangle = \sum_{i \in I} p_i (s_i - r_i) + \sum_{i \notin I} p_i (s_i - r_i) - \langle r, \ s - r \rangle = \sum_{i \in I} p_i \hat{t} + \sum_{i \notin I} p_i s_i - \lambda_1 \hat{t}$$
$$= \hat{t}\left(\lambda_1 - \sum_{i \notin I} p_i\right) + \sum_{i \notin I} p_i s_i - \lambda_1 \hat{t} = \sum_{i \notin I} p_i (s_i - \hat{t}) \leq 0.$$

Now $q \in \text{proj}_D(r)$ yields $\|q - r\|_2^2 \leq \|p - r\|_2^2$, and hence $\|q - s\|_2^2 - \|p - s\|_2^2 \leq 0$, thus $q \in \text{proj}_D(s)$. Similarly, $\|q - s\|_2^2 - \|p - s\|_2 \geq 0$ with $p \in \text{proj}_D(s)$, so $\|q - r\|_2^2 - \|p - r\|_2^2 \geq -2 \langle p - r, \ s - r \rangle \geq 0$, therefore $p \in \text{proj}_D(r)$. ∎

The following corollary states a similar result as in Theis et al. (2005). However, the proof here uses the notion of simplex projections instead of relying on pure analytical statements. The result presented here is stronger, as multiple entries of the vector can be set to zero simultaneously, while in Theis et al. (2005) at most one entry can be zeroed out in a single iteration.

Corollary 29 *Let $s \in L \setminus C$ and $p \in \text{proj}_D(s)$. Then $p_i = 0$ for all $i \in \{1, \ldots, n\}$ with $s_i \leq 0$.*

Proof Let $i \in \{1, \ldots, n\}$ with $s_i \leq 0$. Let $r \in \mathrm{proj}_C(s)$. With Lemma 28(a) follows $r_i = 0$ because $\hat{t} \geq 0$, and the claim follows with Lemma 28(h). ∎

The final step is to meet the hypersphere constraint again. For this, the simplex projection r is projected onto the target hypersphere, simultaneously keeping already vanished entries at zero, yielding a point u. Lemma 30 gives an explicit formulation of this projection and shows that the solution set with respect to the projection onto D stays the same. Refer to Figure 11 for a sketch of the construction of u.

Lemma 30 *Let $s \in L \setminus C$, $r := \mathrm{proj}_C(s) = \max(s - \hat{t} \cdot e, 0)$ with $\hat{t} \in \mathbb{R}_{\geq 0}$ using Lemma 28. Let $I := \{i \in \{1, \ldots, n\} \mid r_i \neq 0\}$ and $d := |I|$. Let $u := m_I + \delta(r - m_I)$ where $\delta := \sqrt{\rho_I}/\|r - m_I\|_2$. Then:*

(a) $u \in L$ and $u_i = 0$ for all $i \notin I$, hence $u \in L_I$.

(b) $\mathrm{proj}_D(u) \subseteq C_I$.

(c) $u = \mathrm{proj}_{L_I}(r)$.

(d) $\mathrm{proj}_D(u) = \mathrm{proj}_D(r) = \mathrm{proj}_D(s) \subseteq C_I$.

Proof (a) Clearly $u \in H$. With $m_I, r \in H$ and Remark 14 follows that $\langle m, m_I \rangle = \langle m, r \rangle = \lambda_1^2/n$. Moreover, $\langle m_I, m_I \rangle = \lambda_1^2/d$ and $\langle r, m_I \rangle = \sum_{i \in I} r_i \cdot \lambda_1/d = \lambda_1^2/d$, therefore $\langle r, m_I - m \rangle = \langle m_I, m_I - m \rangle$. With $u = (1 - \delta)m_I + \delta r$ it is $\langle u, m_I - m \rangle = \langle m_I, m_I - m \rangle = \lambda_1^2(1/d - 1/n)$. Hence $\langle u - m_I, m_I - m \rangle = 0$. Further, $\|m - m_I\|_2^2 = \|m\|_2^2 + \|m_I\|_2^2 - 2\langle m, m_I \rangle = \lambda_1^2(1/d - 1/n)$.

Thus with Proposition 4, $\|u - m\|_2^2 = \|u - m_I\|_2^2 + \|m_I - m\|_2^2 = \rho_I + \lambda_1^2(1/d - 1/n) = \rho$, and with Lemma 15 follows $u \in L$. For $i \notin I$, one obtains $u_i = (1 - \delta)e_i^T m_I + \delta r_i = 0$, hence $u \in L_I$.

(b) If $u \in C$, then $u \in D$ because of $u \in L$ with (a), and hence $u = \mathrm{proj}_D(u)$. The claim then follows with $u_i = 0$ for all $i \notin I$.

If $u \notin C$, then let $q \in \mathrm{proj}_D(u)$. With Corollary 29 applied to u follows that $q_i = 0$ for all i with $u_i \leq 0$, especially for all $i \notin I$. Hence the claim follows.

(c) Write $I = \{i_1, \ldots, i_d\}$ and consider $\varphi \colon \mathbb{R}^n \to \mathbb{R}^d$, $(x_1, \ldots, x_n)^T \mapsto (x_{i_1}, \ldots, x_{i_d})^T$. Further, let $\tilde{H} := \{a \in \mathbb{R}^d \mid e^T a = \lambda_1\}$, $\tilde{K} := \{q \in \mathbb{R}^d \mid \|q\|_2 = \lambda_2\}$, $\tilde{L} := \tilde{H} \cap \tilde{K}$ and $\tilde{D} := \mathbb{R}_{\geq 0}^d \cap \tilde{H} \cap \tilde{K}$. Clearly, when $x_i = 0$ for all $i \notin I$, then $e^T x = e^T \varphi(x)$ and $\|x\|_2 = \|\varphi(x)\|_2$. Thus in this case membership of x in one of H, K, L or D implies membership of $\varphi(x)$ in $\tilde{H}$, $\tilde{K}$, $\tilde{L}$ or $\tilde{D}$, respectively.

Application of Lemma 17 to $\varphi(r)$ and $\varphi(u)$ implies that $\varphi(u) = \mathrm{proj}_{\tilde{L}}(\varphi(r))$. Let $q \in L_I$, then $\varphi(q) \in \tilde{L}$, hence $\|\varphi(u) - \varphi(r)\|_2 \leq \|\varphi(q) - \varphi(r)\|_2$. From $i \notin I$ follows $r_i = u_i = q_i = 0$, hence $\|u - r\|_2 = \|\varphi(u) - \varphi(r)\|_2$ and $\|q - r\|_2 = \|\varphi(q) - \varphi(r)\|_2$, and the claim follows.

(d) For the converse of φ, let $\psi \colon \mathbb{R}^d \to \mathbb{R}^n$, $\tilde{x} \mapsto x$ where $x_i = 0$ for all $i \notin I$ and $x_i = \tilde{x}_j$ when there is a $j \in \{1, \ldots, d\}$ with $i = i_j$. Analogous to the above, membership of $\tilde{y}$ in one of $\tilde{H}$, $\tilde{K}$, $\tilde{L}$ or $\tilde{D}$ implies membership of $\psi(\tilde{y})$ in H, K, L or D, respectively.

With Lemma 28(f) and Lemma 28(h) it is enough to show $\mathrm{proj}_D(u) = \mathrm{proj}_D(r)$. Like in (c), from Lemma 17 follows as well that $\mathrm{proj}_{\tilde{D}}(\varphi(r)) = \mathrm{proj}_{\tilde{D}}(\varphi(u))$. Let $p \in \mathrm{proj}_D(u)$ and $q \in \mathrm{proj}_D(r)$, then $p \in C_I$ with (b) and $q \in C_I$ with Lemma 28(f), and thus $\varphi(p), \varphi(q) \in \tilde{D}$. Assume $\varphi(p) \notin \mathrm{proj}_{\tilde{D}}(\varphi(u))$, then there exists an $a \in \tilde{D}$ with $\|\psi(a) - u\|_2 = \|a - \varphi(u)\|_2 < \|\varphi(p) - \varphi(u)\|_2 = \|p - u\|_2$, violating the minimality of p. Hence $\varphi(p) \in \mathrm{proj}_{\tilde{D}}(\varphi(u))$, and analogously follows $\varphi(q) \in \mathrm{proj}_{\tilde{D}}(\varphi(r))$. Now $\mathrm{proj}_{\tilde{D}}(\varphi(r)) = \mathrm{proj}_{\tilde{D}}(\varphi(u))$ implies that $\varphi(p) \in \mathrm{proj}_{\tilde{D}}(\varphi(r))$ and $\varphi(q) \in \mathrm{proj}_{\tilde{D}}(\varphi(u))$. Thus, $\|p - r\|_2 = \|\varphi(p) - \varphi(r)\|_2 = \|\varphi(q) - \varphi(r)\|_2 = \|q - r\|_2$, so $p \in \mathrm{proj}_D(r)$, and one obtains analogously that $q \in \mathrm{proj}_D(u)$. Therefore $\mathrm{proj}_D(u) = \mathrm{proj}_D(r)$. ∎

With Lemma 30 a point u is constructed. If $u \in C$, then it is already the solution for the projection onto D. Otherwise, Lemma 28 and Lemma 30 can be applied once more, gaining a new point u. Lemma 28(b) states that the amount of nonzero entries of u must decrease, hence this process can be repeated for at most n iterations. If a point with only two non-vanishing entries results, it is guaranteed to be a solution by Proposition 22.

C.4 Proof of Theorem 2 and Theorem 3

Using the previous results it can now be shown that the proposed Algorithm 1 actually computes a correct solution, and that the algorithm always terminates in finite time.

Proof of Theorem 2 For proving partial correctness, let $x \in \mathbb{R}^n$ be arbitrary. Lemma 13 yields $\text{proj}_D(x) = \text{proj}_D(r)$ after line 1, and with Lemma 17 follows $\text{proj}_D(x) = \text{proj}_D(s)$ after line 2. There is a pre-test loop in line 3, and it has to be shown that the loop-invariant is $\text{proj}_D(x) = \text{proj}_D(s)$. At the beginning of the loop, $s \notin \mathbb{R}^n_{\geq 0}$ must hold, thus $s \in L \setminus C$. After line 4, $\text{proj}_D(x) = \text{proj}_D(r)$ holds with Lemma 28. Then with Lemma 30, $\text{proj}_D(x) = \text{proj}_D(s)$ is ensured after line 5, hence the loop-invariant holds. Thus, after the loop it is $\text{proj}_D(x) = \text{proj}_D(s)$ and $s \in D$, so $\text{proj}_D(x) = s$. If $r = m$ in line 2 or $r = m_I$ in line 5, s can be chosen to be any point from L or L_I, respectively, for example the point given in Remark 18. In this case, the projection is not unique, but a valid representative is found.

To prove total correctness, it has to be shown that the loop in line 3 terminates. Remark 23 applied to C_I guarantees that the number of nonzero entries in s is strictly less at the end of the loop than the number of nonzero entries upon entering the loop. Hence, at most n iterations of the loop can be carried out, and when $|I| = 2$ the solution is already in D with Proposition 22. Thus the algorithm terminates in finite time. ∎

It remains to be shown that the optimized variant is also correct.

Proof of Theorem 3 First note that Algorithm 3 consists of a procedure `proj_L` carrying out projections onto L and L_I in-place, and a main body. A function `proj_C` is called to obtain the information on how to perform projections onto C. This is carried out by Algorithm 2. Upon entry of the main body, the input vector x is sorted in descending order, yielding a vector y. The algorithm then operates on the sorted vector y, and undoes the sorting permutation at the end. Because H, L and C are permutation-invariant, the projections onto the respective sets are guaranteed to remain sorted with Lemma 9.

Therefore, y has not to be sorted again for the simplex projection, as Algorithm 2 would require. Also note from Lemma 28 that in the simplex projection the smallest elements are set to zero, and the original Algorithm 1 continues working on the d non-vanishing entries. Because of the order-preservation, entries $d + 1$, ..., n of y are zero, and all relevant information is concentrated in y_1, ..., y_d. Therefore, Algorithm 3 can continue working on these first d entries only, and the index set of non-vanishing entries is always $I = \{1, \ldots, d\}$. As the nonzero elements are stored contiguously in memory, access to y can be realized as a small unit-stride array. This is more efficient than working on a large and sparsely populated vector. Therefore, the loop starting at line 13 corresponds to the loop starting at line 3 in Algorithm 1. At the end of the main body, the sorting permutation τ is inverted and the entries from the sorted result vector y are stored in a new vector s. Because y_{d+1}, ..., $y_n = 0$, these entries can be ignored by setting the entire vector s to zero before-hand. ∎

The proposed optimizations hence lead to the same solution which the original algorithm computes.

Appendix D. Analytical Properties of the Sparseness-Enforcing Projection Operator

In this appendix, it is studied in which situations $\pi_{\geq 0}$ and π as defined in Section 2.4 are differentiable, and hence continuous. Further, an explicit expression for their gradient is sought. It is clear by Theorem 2 that the projection of any point onto D can be written as finite composition of projections onto H, L, C and L_I, respectively. In other words, for all points $x \in \mathbb{R}^n \setminus R$ there exists a finite sequence of index sets $I_1, \ldots, I_h \subseteq \{1, \ldots, n\}$ with $I_j \supsetneq I_{j+1}$ for $j \in \{1, \ldots, h-1\}$ such that

$$\pi_{\geq 0}(x) = \left[\bigcirc_1^{j=h} \left(\text{proj}_{L_{I_j}} \circ \text{proj}_C \right) \circ \text{proj}_L \circ \text{proj}_H \right] (x),$$

where $\bigcirc_1^{j=h}$ denotes iterated composition of functions, starting with $j = h$ and decreasing until $j = 1$, that is $\pi_{\geq 0}(x) = \text{proj}_{L_{I_h}} (\text{proj}_C(\cdots \text{proj}_{L_{I_1}} (\text{proj}_C(\text{proj}_L (\text{proj}_H(x)))) \cdots))$. The sequence $I_1, \ldots, I_h$ here depends on x. The intermediate goal is to show that this sequence remains fixed in a neighborhood of x, and that each projection in the chain is differentiable almost everywhere. This then implies differentiability of $\pi_{\geq 0}$ except for a null set. Because of the close relationship of π with $\pi_{\geq 0}$, π is then also differentiable almost everywhere as shown in the end of this appendix.

The projection onto H is differentiable everywhere, as is clear from its explicit formula given in Lemma 13. Considering L and L_I for $I \subseteq \{1, \ldots, n\}$, the projection is unique and can be cast as function $\mathbb{R}^n \to \mathbb{R}^n$ unless the point to be projected is equal to the barycenters m and m_I, respectively. By considering the explicit formulas given in Lemma 17 and Lemma 30, it is clear that these functions are differentiable as composition of differentiable functions. Thus only the projection onto the simplex C demands attention. Note that the number $\hat{t}$ from Proposition 24 is equal to the mean value of the entries of the argument, that survive the projection, modulo an additive constant:

Proposition 31 *Let $x \in \mathbb{R}^n \setminus C$ and $p := \text{proj}_C(x)$. Then there is a set $I \subseteq \{1, \ldots, n\}$ such that $p = \max(x - \hat{t} \cdot e, \, 0)$ where $\hat{t} = 1/|I| \cdot \left(\sum_{i \in I} x_i - \lambda_1 \right)$.*

Proof Follows directly from Proposition 24 and Algorithm 2 by undoing the permutation τ. ∎

Note that when $I = \{1, \ldots, n\}$, this is very similar to the projection onto H, see Lemma 13. The next result states a condition under which I is locally constant, and hence identifies points where the projection onto C is differentiable with a closed form expression:

Lemma 32 *Let $x \in \mathbb{R}^n \setminus C$, and let $p := \text{proj}_C(x)$, $I \subseteq \{1, \ldots, n\}$ and $\hat{t} \in \mathbb{R}$ be given as in Proposition 31. When $x_i \neq \hat{t}$ for all $i \in \{1, \ldots, n\}$, then the following holds where $u := \sum_{i \in I} e_i \in \mathbb{R}^n$ and $v := e - u \in \mathbb{R}^n$ are the indicator vectors of I and I^C, respectively:*

(a) $p_i > 0$ if and only if $i \in I$.

(b) $p = x + 1/d \cdot \left(\lambda_1 - u^T x \right) u - v \circ x$, where $\circ$ denotes the Hadamard product and $d := |I|$.

(c) There exists a constant $\varepsilon > 0$ and a neighborhood $U := \{ s \in \mathbb{R}^n \mid \|x - s\|_2 < \varepsilon \}$ of x, such that $\text{sgn}(\text{proj}_C(s)) = \text{sgn}(p)$ for all $s \in U$.

(d) $\text{proj}_C(s) = s + 1/d \cdot \left(\lambda_1 - u^T s \right) u - v \circ s$ for all $s \in U$.

(e) $s \mapsto \text{proj}_C(s)$ is differentiable in x.

Proof Only a sketch of a proof is presented here.

(a) Follows from the characterization of $\hat{t}$ given in Proposition 31.

(b) The identity can be validated directly using (a) and Proposition 31.

(c) Follows by choosing $\varepsilon := 1/2 \cdot \min_{i \in \{1,\dots,n\}} |x_i - \hat{t}|$, which is positive by requirement on x.

(d) Validation follows like in (b) using (c).

(e) The projection onto C can be written locally in closed form using (d). In the same neighborhood, the index set of vanishing entries of the projected points does not change. Hence, the projection is differentiable as a composition of differentiable functions. ∎

It is clear that there are points in which $s \mapsto \mathrm{proj}_C(s)$ is continuous but not differentiable, for example points that are projected onto one of the vertices of C. The structure in this situation is locally equivalent to that of the absolute value function. However, for every point where the projection onto C is not differentiable, a subtle change is sufficient to find a point where the projection is differentiable:

Lemma 33 *Consider the function $p \colon \mathbb{R}^n \setminus C \to C$, $s \mapsto \mathrm{proj}_C(s)$ and let $x \in \mathbb{R}^n \setminus C$ be a point such that p is not differentiable in x. Then for all $\varepsilon > 0$ there exists a point $y \in \mathbb{R}^n$ with $\|x - y\|_2 < \varepsilon$ such that p is differentiable in y.*

Proof Let $\hat{t} \in \mathbb{R}$ be the separator from Proposition 31 for the projection onto C. Let the index set of all collisions with $\hat{t}$ be denoted by $J := \{ j \in \{1,\dots,n\} \mid x_j = \hat{t} \}$, which is nonempty with Lemma 32 because p is not differentiable in x. Define $\delta := \varepsilon / \sqrt{4|J|} > 0$ and consider $y := x - \delta \sum_{j \in J} e_j \in \mathbb{R}^n$. Clearly, $\|x - y\|_2 = \varepsilon/2$. From Proposition 31 follows that the separating $\hat{t}$ for the projection onto C is independent of the entries of x with indices in J, as long as they are less than or equal to $\hat{t}$. Because $\delta > 0$, these entries in y are strictly smaller than $\hat{t}$, hence p is differentiable in y with Lemma 32. ∎

Therefore the set on which $s \mapsto \mathrm{proj}_C(s)$ is not differentiable forms a null set. The next result gathers the gradients of the individual projections involved in the computation of the sparseness-enforcing projection operator with respect to σ. Using the chain rule, the gradient of $\pi_{\geq 0}$ can be derived afterwards as multiplication of the individual gradients.

Lemma 34 *The individual projections for $\pi_{\geq 0}$ are differentiable almost everywhere. Their gradients are given as follows:*

(a) $\frac{\partial \mathrm{proj}_H(x)}{\partial x} = E_n - 1/n \cdot ee^T$, *where $E_n \in \mathbb{R}^{n \times n}$ is the identity matrix.*

(b) $\frac{\partial \mathrm{proj}_L(x)}{\partial x} = \frac{\sqrt{\rho}}{\|x-m\|_2} \left(E_n - 1/\|x-m\|_2^2 \cdot (x-m)(x-m)^T \right)$.

(c) $\frac{\partial \mathrm{proj}_C(x)}{\partial x} = E_n - 1/d \cdot uu^T - \mathrm{diag}(v)$. *Here, $I := \{ i \in \{1,\dots,n\} \mid e_i^T \mathrm{proj}_C(x) \neq 0 \}$ is the index set of nonzero entries of the projection onto C, $d := |I|$, $u := \sum_{i \in I} e_i \in \mathbb{R}^n$ and $v := e - u \in \mathbb{R}^n$.*

(d) $\frac{\partial \mathrm{proj}_{L_I}(x)}{\partial x} = \frac{\sqrt{\rho_I}}{\|x-m_I\|_2} \left(E_n - 1/\|x-m_I\|_2^2 \cdot (x-m_I)(x-m_I)^T \right)$.

Proof (a) Follows from the closed form expression in Lemma 13.

(b) Lemma 17 yields $\mathrm{proj}_L(x) = m + \delta(x) \cdot (x-m)$ with $\delta(x) = \sqrt{\rho}/\|x-m\|_2$. With the quotient rule follows $\partial \delta(x)/\partial x = -\sqrt{\rho}/\|x-m\|_2^3 \cdot (x-m)^T$, as ρ does not depend on x. The claim then follows by application of the product rule.

(c) Follows from Lemma 32, similar to (a), using $v \circ x = \mathrm{diag}(v)x$.

(d) Follows exactly as in (b). ∎

Clearly, the gradients for proj_H and for proj_L are special cases of the gradients of proj_C and proj_{L_I}, respectively. Therefore, they need no separate handling in the computation of the overall gradient. Exploiting the special structure of the matrices involved and the readily sorted input as in Algorithm 3, the gradient computation can be further optimized. For the remainder of this appendix, let $O_{a\times b} \in \{0\}^{a\times b}$ and $J_{a\times b} \in \{1\}^{a\times b}$ denote the matrices with a rows and b columns where all entries equal zero and unity, respectively.

Theorem 35 *Let $x \in \mathbb{R}^n$ be sorted in descending order and $\pi_{\geq 0}$ be differentiable in x. Let $h \in \mathbb{N}$ denote the number of iterations Algorithm 3 needs to terminate. In every iteration of the algorithm, store the following values for $i \in \{1,\ldots,h\}$, where line numbers reference Algorithm 3:*

- *$d_i \in \mathbb{N}$ denoting the current dimensionality as determined by lines 12 and 14.*

- *$\delta_i := \sqrt{\rho/\varphi} \in \mathbb{R}$ where ρ and φ are determined in lines 2 and 3, respectively.*

- *$r(i) := y - m_I \in \mathbb{R}^{d_i}$ as computed in line 7.*

Let $N := d_h = \|\pi_{\geq 0}(x)\|_0$ denote the number of nonzero entries in the projection onto D, and define $s(i) := \left(e_1^T r(i), \ldots, e_N^T r(i)\right)^T \in \mathbb{R}^N$ as the first N entries of each $r(i)$. For $i \in \{1,\ldots,h\}$ let

$$A_i := \delta_i E_N - \delta_i/d_i \cdot J_{N\times N} - \alpha_i s(i)s(i)^T + \alpha_i/d_i \cdot s(i)s(i)^T J_{N\times N} \in \mathbb{R}^{N\times N} \text{ where } \alpha_i := \delta_i/\|r(i)\|_2^2,$$

and let $A := \prod_1^{i=h} A_i = A_h \cdots A_1 \in \mathbb{R}^{N\times N}$. Then the gradient of $\pi_{\geq 0}$ in x is $\text{diag}\left(A, \, O_{(n-N)\times(n-N)}\right)$, that is a block diagonal matrix where the quadratic submatrix with row and column indices from 1 to N is given by A, and where all other entries vanish.

Proof The gradient of projections onto H is merely a special case of projections onto C, which also applies to the respective projections onto L and L_I, see Lemma 34. Hence, the very first iteration is a special case of iterations with $i > 1$. Consider one single iteration $i \in \{1,\ldots,h\}$ of Algorithm 3, that is the computation of $\text{proj}_{L_I} \circ \text{proj}_C$ for some $I \subseteq \{1,\ldots,n\}$. Write $d := d_i$, $\delta := \delta_i$, $\alpha := \alpha_i$ and $r := r(i)$ for short. Because the input vector x is sorted by requirement, all intermediate vectors that are projected are sorted as well using Lemma 9. Thus $I = \{1,\ldots,d\}$ holds.

With Lemma 34, the gradient $G_C \in \mathbb{R}^{n\times n}$ of the projection onto both H and C is of the form $G_C := E_n - 1/d \cdot uu^T - \text{diag}(v)$, where $u := \sum_{i=1}^d e_i$ and $v := e - u$. Let $q := (r_1,\ldots,r_d,0,\ldots,0)^T \in \mathbb{R}^n$ be a copy of r padded with zeros to achieve full dimensionality n. The gradient of the projection onto L, and in general L_I, is given by $G_L := \delta E_n - \alpha qq \in \mathbb{R}^{n\times n}$ using Lemma 34. The gradient of the whole iteration is then given by the chain rule, yielding

$$G := G_L G_C = \delta E_n - \delta/d \cdot uu^T - \delta \text{diag}(v) - \alpha qq^T + \alpha/d \cdot qq^T uu^T + \alpha qq^T \text{diag}(v) \in \mathbb{R}^{n\times n}.$$

Write $O := O_{(n-d)\times(n-d)}$, then G is a block diagonal matrix of a matrix from $\mathbb{R}^{d\times d}$ and O: Note that $E_n - \text{diag}(v) = \text{diag}(E_d, O)$, $uu^T = \text{diag}(J_{d\times d}, O)$, and $qq^T = \text{diag}(rr^T, O)$. Therefore, $qq^T \text{diag}(v) = \text{diag}(rr^T, O) \cdot \left(\begin{smallmatrix} 0 & 0 \\ 0 & E_{n-d} \end{smallmatrix}\right) = 0$, and $qq^T uu^T = \text{diag}(rr^T J_{d\times d}, O)$. Thus

$$G = \delta \text{diag}(E_d, O) - \delta/d \cdot \text{diag}(J_{d\times d}, O) - \alpha \text{diag}(rr^T, O) + \alpha/d \cdot \text{diag}(rr^T J_{d\times d}, O)$$
$$= \text{diag}\left(\delta E_d - \delta/d \cdot J_{d\times d} - \alpha rr^T + \alpha/d \cdot rr^T J_{d\times d}, \, O\right).$$

By denoting the gradient of iteration i by matrix $G_i \in \mathbb{R}^{n\times n}$ for $i \in \{1,\ldots,h\}$ and by application of the chain rule follows that the gradient of all iterations is given by $\prod_1^{i=h} G_i$. In this matrix, all

entries but the top left submatrix of dimensionality $N \times N$ are vanishing, where $N = d_h$. This is because the according statement applies to G_i for the top left submatrix of dimensionality $d_i \times d_i$, and $d_1 > \cdots > d_h$ holds, and only the according entries survive the matrix multiplication. Therefore it is sufficient to compute only the top left $N \times N$ entries of the gradients of the individual iterations, as the remaining entries are not relevant for the final gradient. This is reflected by the definition of the matrices A_i for $i \in \{1, \ldots, h\}$ from the claim. ∎

The gradient can thus be computed using matrix-matrix multiplications, where the matrices are square and the edge length is the number of nonzero entries in the result of the projection. This computation is more efficient than using the $n \times n$ matrices of the individual projections. However, when the target degree of sparseness is low, and thus the amount of nonzero entries N in the result of the projection is large, gradient computation can become very inefficient. In practice, often only the product of the gradient with an arbitrary vector is required. In this case, the procedure can be sped up by exploiting the special structure of the gradient of $\pi_{\geq 0}$:

Corollary 36 *Let $x \in \mathbb{R}^n$ be sorted in descending order and $\pi_{\geq 0}$ be differentiable in x. The product of the gradient of $\pi_{\geq 0}$ in x with an arbitrary vector can be computed using vector operations only.*

Proof Note that because of the associativity of the matrix product it is enough to consider the product of the gradient $G \in \mathbb{R}^{n \times n}$ of one iteration of Algorithm 3 with one vector $y \in \mathbb{R}^n$. Because of the statements of Theorem 35, it suffices to consider the top left $N \times N$ entries of G and the first N entries of y, as all other entries vanish. Therefore let $A := \delta E_N - \delta/d \cdot J_{N \times N} - \alpha ss^T + \alpha/d \cdot ss^T J_{N \times N} \in \mathbb{R}^{N \times N}$ be the non-vanishing block of G as given by Theorem 35, let $u := J_{N \times 1} \in \mathbb{R}^N$ be the vector of ones such that $uu^T = J_{N \times N}$, and let $z := (y_1, \ldots, y_N)^T \in \mathbb{R}^N$ denote the vector with the first entries of y. Using matrix product associativity and distributivity over multiplication with a scalar yields

$$Az = \delta \left(z - 1/d \cdot \langle z, u \rangle \cdot u \right) + \alpha \left(1/d \cdot \langle s, u \rangle \langle z, u \rangle - \langle s, z \rangle \right) s,$$

where $\langle z, u \rangle = \sum_{i=1}^{N} z_i$ and $\langle s, u \rangle = \sum_{i=1}^{N} s_i$. Hence Az can be computed in-place from z by subtraction of a scalar value from all entries, rescaling by δ, and adding a scaled version of vector s. ∎

Although in Theorem 35 and Corollary 36 it was necessary that the input vector is sorted, the general case can easily be recovered:

Proposition 37 *Let $x \in \mathbb{R}^n$ be a point, $\tau \in S_n$ such that $y := P_\tau x \in \mathbb{R}^n$ is sorted in descending order and $\pi_{\geq 0}$ be differentiable in y with gradient $G \in \mathbb{R}^{n \times n}$. Then $\pi_{\geq 0}$ is also differentiable in x, and the gradient is $P_\tau^T G P_\tau$.*

Proof Follows with $P_\tau^T = P_{\tau^{-1}} = P_\tau^{-1}$, $\pi_{\geq 0}(x) = P_\tau^T \pi_{\geq 0}(P_\tau x) = P_\tau \pi_{\geq 0}(y)$ and the chain rule. ∎

Likewise, the gradient for the unrestricted projection π can be computed from the gradient for $\pi_{\geq 0}$:

Proposition 38 *Let $x \in \mathbb{R}^n$ be a point such that $\pi_{\geq 0}$ is differentiable in $|x|$ with gradient $G \in \mathbb{R}^{n \times n}$. Let $s \in \{\pm 1\}^n$ be given such that $\pi(x) = s \circ \pi_{\geq 0}(|x|)$. Then π is differentiable in x, and the gradient is $\mathrm{diag}(s) G \mathrm{diag}(s)$.*

Proof Follows analogously to Proposition 37, using $|x| = s \circ x = \mathrm{diag}(s)x$. ∎

Summing up, the gradient of the projection onto $S_{\geq 0}^{(\lambda_1, \lambda_2)}$ and $S^{(\lambda_1, \lambda_2)}$ can be computed efficiently by bookkeeping a few values as discussed in Theorem 35, and applying simple operations to recover the general case. When only the product of the gradient with a vector is required, the computation can be made more efficient as stated in Corollary 36. Direct application of Theorem 35 should be avoided in this situation because of the high computational complexity.

Appendix E. Gradients for SOAE Learning

The objective function E_{SOAE} is a convex combination of two similarity measures s_R and s_C. The degrees of freedom W, W_{out} and θ_{out} of the SOAE architecture should be tuned by gradient-based methods to minimize these functions. This appendix reports the gradient information needed for reproduction of the experiments. The first statement addresses the reconstruction module.

Proposition 39 *It is $\left(\partial s_R(\tilde{x}, x)/\partial W\right)^T = xgWf'(u) + g^T h^T \in \mathbb{R}^{d \times n}$ where $g := \partial s_R(\tilde{x},x)/\partial \tilde{x} \in \mathbb{R}^{1 \times d}$ is the gradient of the similarity measure with respect to its first argument. Additionally, $\left(\partial s_R(\tilde{x}, x)/\partial W_{\text{out}}\right)^T = 0 \in \mathbb{R}^{n \times c}$ and $\partial s_R(\tilde{x}, x)/\partial \theta_{\text{out}} = 0 \in \mathbb{R}^{1 \times c}$.*

Proof As s_R does not depend on W_{out} or θ_{out}, the respective gradients vanish. The symmetry between encoding and decoding yields $\tilde{x} := W \cdot f\left(W^T x\right)$. The gradient for W follows using the chain rule and the product rule for matrix calculus, see Neudecker (1969) and Vetter (1970). ∎

The correlation coefficient is the recommended choice for the similarity measure of the reconstruction module because it is normed and invariant to affine-linear transformations. It is also differentiable almost everywhere:

Proposition 40 *If s_R is the correlation coefficient and $x, \tilde{x} \in \mathbb{R}^d \setminus \{0\}$, then*

$$\left(\frac{\partial s_R(\tilde{x},x)}{\partial \tilde{x}}\right)^T = \frac{1}{\sqrt{\lambda \mu}}\left(x - \frac{\langle e, x \rangle}{d}e\right) - \frac{s_R(\tilde{x},x)}{\lambda}\left(\tilde{x} - \frac{\langle e, \tilde{x} \rangle}{d}e\right) \in \mathbb{R}^d,$$

where all entries of $e \in \mathbb{R}^d$ are unity, $\lambda := \|\tilde{x}\|_2^2 - 1/d \cdot \langle e, \tilde{x} \rangle^2 \in \mathbb{R}$ and $\mu := \|x\|_2^2 - 1/d \cdot \langle e, x \rangle^2 \in \mathbb{R}$.

Proof One obtains $\sqrt{\lambda \mu} \cdot s_R(\tilde{x},x) = \langle x, \tilde{x} \rangle - 1/d \cdot \langle e, \tilde{x} \rangle \langle e, x \rangle$ because s_R is the correlation coefficient. The claim then follows with the quotient rule. ∎

The gradients of the similarity measure for classification capabilities are essentially equal to those of an ordinary two-layer neural network, and can be computed using the back-propagation algorithm (Rumelhart et al., 1986). However, the pairing of the softmax transfer function with the cross-entropy error function provides a particularly simple structure of the gradient (Dunne and Campbell, 1997). For completeness, the gradients of the classification module of SOAE are summarized:

Proposition 41 *If s_C is the cross-entropy error function, g is the softmax transfer function and the target vector for classification t is a one-of-c code, then $\left(\partial s_C(y, t)/\partial W_{\text{out}}\right)^T = h \cdot (y - t)^T \in \mathbb{R}^{n \times c}$, $\partial s_C(y, t)/\partial \theta_{\text{out}} = (y - t)^T \in \mathbb{R}^{1 \times c}$ and $\left(\partial s_C(y, t)/\partial W\right)^T = x \cdot \left((y - t)^T W_{\text{out}}^T f'(u)\right) \in \mathbb{R}^{d \times n}$.*

Proof Basic matrix calculus (Neudecker, 1969; Vetter, 1970) yields $\partial s_C(y, t)/\partial \theta_{\text{out}} = \left(\partial s_C(y, t)/\partial y\right) \cdot g'(y)$, $\left(\partial s_C(y, t)/\partial W_{\text{out}}\right)^T = h \cdot \left(\partial s_C(y, t)/\partial \theta_{\text{out}}\right)$ and $\left(\partial s_C(y, t)/\partial W\right)^T = x \cdot \left(\left(\partial s_C(y, t)/\partial \theta_{\text{out}}\right) \cdot W_{\text{out}}^T f'(u)\right)$. By requirement $\partial s_C(y, t)/\partial y = -(t \oslash y)^T$, where $\oslash$ denotes the element-wise quotient, $g'(y) = \text{diag}(y) - yy^T$ and $\sum_{i=1}^c t_i = 1$. Therefore $\left(\partial s_C(y, t)/\partial \theta_{\text{out}}\right)^T = \left(yy^T - \text{diag}(y)\right) \cdot (t \oslash y) = y \cdot \langle y, t \oslash y \rangle - y \circ t \oslash y = y - t$ using $\langle y, t \oslash y \rangle = \sum_{i=1}^c y_i \cdot t_i/y_i = 1$, and the claim follows. ∎

As E_{SOAE} is a convex combination of the reconstruction error and the classification error, its overall gradient follows immediately from Proposition 39 and Proposition 41. Proposition 40, the results from Appendix D, and the gradient of the L_0 projection as described in Section 2.4 can then be used to compute the explicit gradients for the procedure proposed in this paper.

References

L. Acion, J. J. Peterson, S. Temple, and S. Arndt. Probabilistic index: An intuitive non-parametric approach to measuring the size of treatment effects. *Statistics in Medicine*, 25(4):591–602, 2006.

E. B. Baum, J. Moody, and F. Wilczek. Internal representations for associative memory. *Biological Cybernetics*, 59(4–5):217–228, 1988.

D. P. Bertsekas. *Nonlinear Programming*. Athena Scientific, 2nd edition, 1999.

C. M. Bishop. *Neural Networks for Pattern Recognition*. Clarendon Press, 1995.

T. Blumensath and M. E. Davies. A simple, efficient and near optimal algorithm for compressed sensing. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 3357–3360, 2009.

L. Bottou and Y. LeCun. Large scale online learning. In *Advances in Neural Information Processing Systems 16*, pages 217–224, 2004.

D. M. Bradley and J. A. Bagnell. Differentiable sparse coding. In *Advances in Neural Information Processing Systems 21*, pages 113–120, 2009.

Y. Chen and X. Ye. Projection onto a simplex. Technical Report arXiv:1101.6081v2, University of Florida, 2011.

D. C. Cireşan, U. Meier, L. M. Gambardella, and J. Schmidhuber. Deep, big, simple neural nets for handwritten digit recognition. *Neural Computation*, 22(12):3207–3220, 2010.

D. C. Cireşan, U. Meier, and J. Schmidhuber. Multi-column deep neural networks for image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3642–3649, 2012.

G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals, and Systems*, 2(4):303–314, 1989.

D. DeCoste and B. Schölkopf. Training invariant support vector machines. *Machine Learning*, 46 (1–3):161–190, 2002.

J. Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006.

F. Deutsch. *Best Approximation in Inner Product Spaces*. Springer, 2001.

D. L. Donoho. For most large underdetermined systems of linear equations the minimal ℓ_1-norm solution is also the sparsest solution. *Communications on Pure and Applied Mathematics*, 59(6): 797–829, 2006.

F. Downton. The estimation of $\Pr(Y < X)$ in the normal case. *Technometrics*, 15(3):551–558, 1973.

J. Duchi, S. Shalev-Shwartz, Y. Singer, and T. Chandra. Efficient projections onto the ℓ_1-ball for learning in high dimensions. In *Proceedings of the International Conference on Machine Learning*, pages 272–279, 2008.

R. A. Dunne and N. A. Campbell. On the pairing of the softmax activation and cross-entropy penalty functions and the derivation of the softmax activation function. In *Proceedings of the Australasian Conference on Neural Networks*, pages 181–185, 1997.

R. L. Dykstra. An algorithm for restricted least squares regression. *Journal of the American Statistical Association*, 78(384):837–842, 1983.

R. L. Dykstra and J. P. Boyle. An algorithm for least squares projections onto the intersection of translated, convex cones. *Journal of Statistical Planning and Inference*, 15(3):391–399, 1987.

J. Friedman, T. Hastie, and R. Tibshirani. A note on the group lasso and a sparse group lasso. Technical Report arXiv:1001.0736v1, Stanford University, 2010.

K. Funahashi. On the approximate realization of continuous mappings by neural networks. *Neural Networks*, 2(3):183–192, 1989.

S. Geman, E. Bienenstock, and R. Doursat. Neural networks and the bias/variance dilemma. *Neural Computation*, 4(1):1–58, 1992.

X. Glorot, A. Bordes, and Y. Bengio. Deep sparse rectifier neural networks. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, pages 315–323, 2011.

K. Gregor and Y. LeCun. Learning fast approximations of sparse coding. In *Proceedings of the International Conference on Machine Learning*, pages 399–406, 2010.

R. J. Grissom. Probability of the superior outcome of one treatment over another. *Journal of Applied Psychology*, 79(2):314–316, 1994.

M. Heiler and C. Schnörr. Learning sparse representations by non-negative matrix factorization and sequential cone programming. *Journal of Machine Learning Research*, 7:1385–1407, 2006.

G. E. Hinton, S. Osindero, and Y. W. Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7):1527–1554, 2006.

J.-B. Hiriart-Urruty. At what points is the projection mapping differentiable? *The American Mathematical Monthly*, 89(7):456–458, 1982.

Y. Hochberg and A. C. Tamhane. *Multiple Comparison Procedures*. John Wiley & Sons, Inc., 1987.

K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989.

H. Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24(6):417–441, 1933.

P. O. Hoyer. Non-negative matrix factorization with sparseness constraints. *Journal of Machine Learning Research*, 5:1457–1469, 2004.

D. H. Hubel and T. N. Wiesel. Receptive fields of single neurones in the cat's striate cortex. *Journal of Physiology*, 148(3):574–591, 1959.

N. Hurley and S. Rickard. Comparing measures of sparsity. *IEEE Transactions on Information Theory*, 55(10):4723–4741, 2009.

A. Hyvärinen, P. O. Hoyer, and E. Oja. Sparse code shrinkage: Denoising by nonlinear maximum likelihood estimation. In *Advances in Neural Information Processing Systems 11*, pages 473–478, 1999.

J. Karbowski. How does connectivity between cortical areas depend on brain size? Implications for efficient computation. *Journal of Computational Neuroscience*, 15(3):347–356, 2003.

T. Kohonen. Correlation matrix memories. *IEEE Transactions on Computers*, C-21(4):353–359, 1972.

W. H. Kruskal and W. A. Wallis. Use of ranks in one-criterion variance analysis. *Journal of the American Statistical Association*, 47(260):583–621, 1952.

A. J. Laub. *Matrix Analysis for Scientists and Engineers*. Society for Industrial and Applied Mathematics, 2004.

S. B. Laughlin and T. J. Sejnowski. Communication in neuronal networks. *Science*, 301(5641): 1870–1874, 2003.

Y. LeCun and C. Cortes. The MNIST database of handwritten digits, 1998. Available electronically at `http://yann.lecun.com/exdb/mnist`.

Y. LeCun, J. S. Denker, and S. A. Solla. Optimal brain damage. In *Advances in Neural Information Processing Systems 2*, pages 598–605, 1990.

D. D. Lee and H. S. Seung. Learning the parts of objects by nonnegative matrix factorization. *Nature*, 401(6755):788–791, 1999.

H. Levene. Robust tests for equality of variances. In *Contributions to Probability and Statistics: Essays in Honor of Harold Hotelling*, pages 278–292. Stanford University Press, 1960.

S. Z. Li, X. Hou, H. Zhang, and Q. Cheng. Learning spatially localized, parts-based representation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 207–212, 2001.

J. Liu and J. Ye. Efficient euclidean projections in linear time. In *Proceedings of the International Conference on Machine Learning*, pages 657–664, 2009.

J. Liu and J. Ye. Efficient ℓ_1/ℓ_q norm regularization. Technical Report arXiv:1009.4766v1, Arizona State University, 2010.

J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman. Supervised dictionary learning. In *Advances in Neural Information Processing Systems 21*, pages 1033–1040, 2009.

J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online learning for matrix factorization and sparse coding. *Journal of Machine Learning Research*, 11:19–60, 2010.

H. Markram, J. Lübke, M. Frotscher, A. Roth, and B. Sakmann. Physiology and anatomy of synaptic connections between thick tufted pyramidal neurones in the developing rat neocortex. *Journal of Physiology*, 500(2):409–440, 1997.

A. Mason, A. Nicoll, and K. Stratford. Synaptic transmission between individual pyramidal neurons of the rat visual cortex in vitro. *Journal of Neuroscience*, 11(1):72–84, 1991.

C. Michelot. A finite algorithm for finding the projection of a point onto the canonical simplex of $\mathbb{R}^n$. *Journal of Optimization Theory and Applications*, 50(1):195–200, 1986.

J. Mutch and D. G. Lowe. Multiclass object recognition with sparse, localized features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11–18, 2006.

B. K. Natarajan. Sparse approximate solutions to linear systems. *SIAM Journal on Computing*, 24 (2):227–234, 1995.

H. Neudecker. Some theorems on matrix differentiation with special reference to Kronecker matrix products. *Journal of the American Statistical Association*, 64(327):953–963, 1969.

B. A. Olshausen and D. J. Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381(6583):607–609, 1996.

B. A. Olshausen and D. J. Field. Sparse coding of sensory inputs. *Current Opinion in Neurobiology*, 14(4):481–487, 2004.

P. Paatero and U. Tapper. Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values. *Environmetrics*, 5(2):111–126, 1994.

G. Palm. On associative memory. *Biological Cybernetics*, 36(1):19–31, 1980.

J. Pizarro, E. Guerrero, and P. L. Galindo. Multiple comparison procedures applied to model selection. *Neurocomputing*, 48(1–4):155–173, 2002.

A. Quattoni, X. Carreras, M. Collins, and T. Darrell. An efficient projection for $l_{1,\infty}$ regularization. In *Proceedings of the International Conference on Machine Learning*, pages 857–864, 2009.

M. Ranzato, Y. Boureau, and Y. LeCun. Sparse feature learning for deep belief networks. In *Advances in Neural Information Processing Systems 20*, pages 1185–1192, 2008.

M. Rehn and F. T. Sommer. A network that uses few active neurones to code visual input predicts the diverse shapes of cortical receptive fields. *Journal of Computational Neuroscience*, 22(2): 135–146, 2007.

J. L. Rodgers and W. A. Nicewander. Thirteen ways to look at the correlation coefficient. *The American Statistician*, 42(1):59–66, 1988.

D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986.

B. Schölkopf. *Support Vector Learning*. PhD thesis, Technische Universität Berlin, 1997.

S. S. Shapiro and M. B. Wilk. An analysis of variance test for normality (complete samples). *Biometrika*, 52(3–4):591–611, 1965.

P. Y. Simard, D. Steinkraus, and J. C. Platt. Best practices for convolutional neural networks applied to visual document analysis. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 958–962, 2003.

S. Sra. Fast projections onto mixed-norm balls with applications. *Data Mining and Knowledge Discovery*, 25(2):358–377, 2012.

F. J. Theis and T. Tanaka. Sparseness by iterative projections onto spheres. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 709–712, 2006.

F. J. Theis, K. Stadlthanner, and T. Tanaka. First results on uniqueness of sparse non-negative matrix factorization. In *Proceedings of the European Signal Processing Conference*, pages 1672–1675, 2005.

M. Thom, R. Schweiger, and G. Palm. Supervised matrix factorization with sparseness constraints and fast inference. In *Proceedings of the International Joint Conference on Neural Networks*, pages 973–979, 2011a.

M. Thom, R. Schweiger, and G. Palm. Training of sparsely connected MLPs. In *Lecture Notes in Computer Science*, volume 6835, pages 356–365, 2011b.

E. van den Berg, M. Schmidt, M. P. Friedlander, and K. Murphy. Group sparsity via linear-time projection. Technical Report TR-2008-09, Department of Computer Science, University of British Columbia, 2008.

W. J. Vetter. Derivative operations on matrices. *IEEE Transactions on Automatic Control*, 15(2):241–244, 1970.

W. E. Vinje and J. L. Gallant. Sparse coding and decorrelation in primary visual cortex during natural vision. *Science*, 287(5456):1273–1276, 2000.

J. von Neumann. *Functional Operators, Volume II: The Geometry of Orthogonal Spaces*. Princeton University Press, 1950.

J. Weston, A. Elisseeff, B. Schölkopf, and M. Tipping. Use of the zero-norm with linear models and kernel methods. *Journal of Machine Learning Research*, 3:1439–1461, 2003.

D. R. Wilson and T. R. Martinez. The general inefficiency of batch training for gradient descent learning. *Neural Networks*, 16(11):1429–1451, 2003.

H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):301–320, 2005.