A Cascading Chi-shapes based Decoder for Constraint-handling in Distributed Energy Management

Joerg Bremer[1] and Sebastian Lehnhoff[2]
[1]*Department of Computing Science, University of Oldenburg, Uhlhornsweg, Oldenburg, Germany*
[2]*R&D Division Energy, OFFIS – Institute for Information Technology, Escherweg, Oldenburg, Germany*

Keywords: χ-shapes, Decoder, Flexibility Modeling, CMA-ES, Constraint-handling.

Abstract: A steadily rising share of small, distributed, and volatile energy units like wind energy converters solar panels, co-generation plants, or similar assigns new tasks and challenges to the smart grid regarding operation and control. The growing complexity of the grid also imposes a growing complexity of constraints that restrict the validity of solutions for operation schedules, resource capacity utilization or grid compliance. Using surrogate models as an abstraction layer has recently become a promising approach for constructing algorithms independently from any knowledge about the actual device or operation restricting constraints. So called decoders as a special constraint handling technique allow for systematically generating feasible solutions directly from a learned surrogate model. Some decoder approaches based on support vector machines have already been implemented, but suffer from performance issues and a sensible parametrization. We propose a new type of decoder based on a cascade of χ-shapes to overcome these problems. The applicability is demonstrated with a simulation study using different types of flexible energy units.

1 INTRODUCTION

Electrical power supply is currently undergoing a transition towards a decentralized provision by renewable resources. Political decisions target a change from traditional control schemes with rather few, large power plants to a power grid operated by numerous small (volatile and hardly predictable) renewable energy resources (DER). Grid control by decentralized and individually configured units leads to a need for new algorithms in order to cope with growing (constraint) complexity. If a huge number of small devices is responsible for control operations, they will have to situationally group together to jointly gain potential and flexibility like in virtual power plants (Sonnenschein et al., 2014).

Individual search spaces of different units – representing the individual capabilities within a group – have to be integrated to a model for the operation planning at runtime. Each individually operated unit has its own set of distinct schedules to offer for a scheduling algorithm. Such flexibility depends on the current, individual configuration, several (technical) constraints for operation, current operational state, and if applicable, on state and requirements of coupled units – e. g. on the thermal demand of a house in case of a co-generation plant (Bremer et al., 2010). Thus, a static model is not targeted as it had to be continuously adapted and corrected.

If a simulation model of an energy unit is regarded as a characteristic function able to indicate whether an arbitrary, given schedule is operable by the DER or not, it can be converted to another model that allows for a standardized access to feasibility information. The characteristic function that indicates operability of schedules may be captured e. g. by machine learning approaches. Built on top, a so called decoder provides a mapping function to turn any given schedule into a similar feasible one (with respect to the energy unit model that has been used for training). As constraint-handling technique such decoder may be used for solution repair or for guiding any algorithm where to look for feasible solutions (Koziel and Michalewicz, 1999).

In (Bremer and Sonnenschein, 2013) a decoder has been proposed based on a 1-class support vector machine (SVM). This decoder works fine on a wide range of energy unit types but has drawbacks regarding training time and parametrization for precision in high-dimensional cases. The works of (Fröhlinger, 2017; Neugebauer et al., 2016) improved precision by extending the high-dimensional SVM approach to

Bremer, J. and Lehnhoff, S.
A Cascading Chi-shapes based Decoder for Constraint-handling in Distributed Energy Management.
DOI: 10.5220/0006926101840191
In *Proceedings of the 10th International Joint Conference on Computational Intelligence (IJCCI 2018)*, pages 184-191
ISBN: 978-989-758-327-8

a cascade of classifiers for the flexibility model. But, for this cascading approach no cascading decoder is known so far.

We present a new approach for flexibility modeling and decoder based on χ-shapes that constitute a concave hull as model for the feasible region of an energy unit. With this approach performance as well as precision can be improved by integrating the cascade architecture. In addition, as the χ-shape model works directly in data space as oppose to SVM approaches that model in reproducing kernel Hilbert spaces, new applications regarding intersection or union of flexibilities from different energy units become possible.

The rest of the paper is organized as follows. We start with an introduction to flexibility modeling and recap concave hull techniques. A description of the new approach based on χ-shapes is complemented by a hybridization with evolution strategies for demonstrating integration into optimization. Finally, a simulation study with evaluation results concludes.

2 RELATED WORK

2.1 Constraints and Flexibility Modeling

In order to understand, the concept of flexibility surrogate models and derived decoders for constraint-handling, we first have to look at the constraints and the representation of schedules.

Usually, a DER might achieve the main task it has been built for in different alternative ways. For example, a CHP (combined heat and power plant) is supposed to supply the heat for varying demand in a household at every moment in time. But, heat usage is often decoupled from heat production by using a thermal buffer store. Thus, different production profiles may be used for generating the heat. This leads, in turn, to different respective electric load profiles that may be offered as choice to a scheduling controller.

A schedule is a data vector $\boldsymbol{x} \in \mathbb{R}^d$, with number of periods d. For each period the i-th element of $\boldsymbol{x}$ describes the respective amount of electric energy produced or consumed in this period or respectively the mean active power output or input during this period. The term operable denotes that a schedule does not violate any constraint. The term constraint comprises hard (usually technically rooted) and soft (often economically or ecologically rooted or subject to personal preferences) as well as often non-linear constraints resulting from system embedding.

Real world problems often face nonlinear and/ or

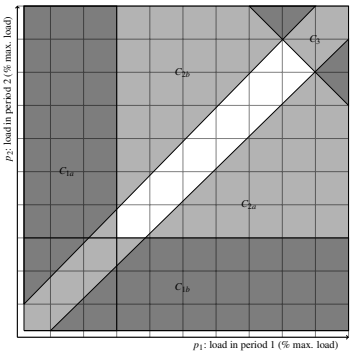

Figure 1: Simplified 2-dimensional example for the scope of action of a modulating co-generation plant with the superposition of three constraints (cf. (Bremer et al., 2010)).

combined constraints. The set of constraints defines the shape of a region within the search space (the hypercube defined by operation parameter limits) that contains only feasible solutions. This feasible region might be arbitrary shaped and discontinuous. It is this region that defines feasibility and flexibility and that has to be modeled to allow distinguishing operable and not operable schedules.

As the feasible region is a sub-vector space, constraints can be interpreted geometrically. Without any constraint, the whole hypercube $[0,1]^d$ (active power between 0 and 100%) would be a model for the region of feasible schedules. When applying constraints, different parts (sub-regions) of the hypercube drop off the feasible region, because the respective schedules are not operable. Only the remaining region (hypercube minus superposition of all regions prohibited by constraints) is the feasible region of the DER. Figure 1 shows a simplified, linear, 2-dimensional example for a modulating CHP (power may vary between min. and max. power: constraints C_1) with a connected thermal buffer store (state of charge prohibits region C_3); instantaneous changes in production level are prohibited by inertia (constraints C_{2a} and C_{2b}). The remaining white region is the feasible region $\mathcal{F}$. We denote with $\mathcal{F}_u$ the feasible region that is specific to an energy unit u. Examples from a productive system would usually comprise 96 dimensions and many more constraints, heavy non linearity and discontinuity.

A flexibility model is a surrogate model that substitutes a simulation model and checks whether a given schedule is operable under current circumstances or not. These models are classifiers to tell feasible and infeasible schedules apart (with regard to a specific unit and a specific initial operation state) without needing to simulate.

A decoder is a constraint-handling technique that imposes a relationship between feasibility and

decoder solutions. For example, (Koziel and Michalewicz, 1999) proposed a homomorphous mapping between an n-dimensional hypercube and the feasible region in order to transform the problem into an topological equivalent one that is easier to handle, although with a need for extra parameters that have to be fit empirically and with a need for explicitly given constraint formulations. Earlier approaches e.g. used Riemannien mapping (Kim, 1998).

An example for the smart grid domain is given in (Bremer et al., 2010). There, a flexibility model based on a one-class support vector data description (SVDD) (Tax and Duin, 2004) was proposed. The goal of building such a model is to learn the feasible region of the schedules of a DER by learning the enclosing boundary around the set of operable schedules. This task is achieved by determining a mapping $\Phi : \mathcal{X} \subset \mathbb{R}^d \rightarrow \mathcal{H}, \boldsymbol{x} \mapsto \Phi(\boldsymbol{x})$ such that all data from a given region $\mathcal{X}$ is mapped to a minimal hypersphere in some high- or indefinite-dimensional Hilbert space $\mathcal{H}$. The images enclosing ball is used as an abstract model.

A support vector decoder can harness this relationship by mapping an arbitrary (infeasible) schedule into this high-dimensional space and moving the mapped image towards the center of the ball until it touches the surface. This translated image can then be mapped back to a now feasible schedule at the border of the feasible region. In the same way also a proportional move into the ball is possible to harness the whole feasible region.

2.2 χ-Shapes

Another general approach for modeling the enclosing envelope around a point cloud is using a concave hull. A concave hull constitutes a polygon that represents the characteristic shape of a set of input points (Duckham et al., 2008). One approach to representing characteristic shapes of point clouds was introduced in (Edelsbrunner et al., 2006), where α-shapes were used. In (Melkemi and Djebali, 2000) $\mathcal{A}$-shapes where introduced. Both approaches are based on Delaunay triangulation and Voronoi diagrams and quickly become intractable with growing number and dimension of points. Different approaches are based on putting a lattice on the plane and defining the union of cells containing points as shape. An improvement to these so called s-shapes are r-shapes that are constituted by a union of discs around points from the given point cloud. In the 2-dimensional plane these shapes can be calculated in linear time but require a preceding step of finding suitable parameters which cannot be determined in a closed form. More geographical information systems related approaches can e. g. be found in (Park and Oh, 2012; Moreira and Santos, 2007). In (Braune et al., 2016) a multi shape approach has been developed for clustering. In (Duckham et al., 2008) so called χ-shapes are proposed that are easy to calculate and calculation can be done in $O(n \cdot \log n)$.

χ-shapes are defined for finite sets of at least 3 points in $\mathbb{R}^2$. χ-shapes model the spatial distribution of point sets and constitute simple polygons containing all points and bounding an area inside (or equal to) the convex hull (Duckham et al., 2008). The polygon is generated based on a convex Delaunay triangulation that is iteratively shrunk to a concave hull by removing outer edges (Rosen et al., 2014). Figure 2 gives some examples of concave hulls around point clouds.

So far, none of these methods had been used for flexibility modeling due to their disability to scale well with dimensionality. In (Hörding, 2017), a first attempt was made to combine ideas from (Fröhlinger, 2017; Neugebauer et al., 2016) – where high dimensional training sets have been disintegrated into a set of lower dimensional ones by using a cascade of classifiers for flexibility modeling – and concave hulls and χ-shapes as base model from (Duckham et al., 2008).

3 ALGORITHM

3.1 Model and Decoder

Here, we propose a novel method of generating a decoder for energy management algorithms that are based on χ-shape models of the feasible region of possible operation of an energy unit. The approach is based on the cascading χ-shape model from (Hörding, 2017) that also predicted the possible extension to a fully-fledged decoder method.

In general, the approach works as follows. In a first step, a training set is generated from an energy units' simulation model that serves as a stencil for the feasible region as in the approach from (Bremer et al., 2010; Bremer and Sonnenschein, 2013). This training set contains instances of feasible schedules with a dimension that covers the whole planning horizon. In day-ahead scheduling a schedule usually has a dimension of 96 of 15-minute intervals for 24 hours. This training set is subdivided into a set of training sets with schedules of dimension two. With this, we follow the cascading approach for better classification as proposed in (Fröhlinger, 2017). All sub training sets are overlapping in one neighboring dimension. In a second step we calculate a χ-shape for each of these

sub training sets after (Duckham et al., 2008; Hörding, 2017). Finally, a decoder is derived. The task of a decoder is to map points from the exterior of the concave hull onto a point inside the hull.

Let $\boldsymbol{x} = (p_1, \ldots, p_d) \in [0,1]^d$ be a schedule for an energy unit for d time intervals. The elements of $\boldsymbol{x}$ denote generated or consumed real power in p.u. (percentage of rated power) during the respective time interval. Let $\mathcal{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ be a training set with instances $\boldsymbol{x}_i$ of random, feasible schedules.

Let $\boldsymbol{x}$ be an infeasible schedule (point) from the exterior. We start by mapping the first dimension of $\boldsymbol{x}$. This has to be done by line search. The first point along the x-axis x_0 that guarantees an intersection of the line segment $\overline{(x_0,0)(x_0,1)}$ with the first χ-shape polygon is taken as an anchor for the mapping.

Let $\mathcal{X}^{(j)} = \{(p_j, p_{j+1})\}_n$ with $j \in \{1, \ldots, d\} \subseteq \mathbb{N}$ be a 2-dimensional training set comprising only an intersection plane of the original training set along axis j and $j+1$. We now break down the original high-dimensional training set $\mathcal{X}$ into a set of 2-dimensional training sets $\{\mathcal{X}^{(1)}, \mathcal{X}^{(2)}, \ldots, \mathcal{X}^{(d-1)}\}$.

For each of these 2-dimensional planes a χ-shape $\mathcal{S}_j(\mathcal{X}^{(j)})$ is constructed enclosing the values of feasible power for 2 succeeding time intervals j and $j+1$ in the schedule. The χ-shape consists of a set of line segments $\{\ell_i\}_{i \in \mathbb{N}^+}$ defining a closed polygon enclosing points of feasible power projected onto the respective plane.

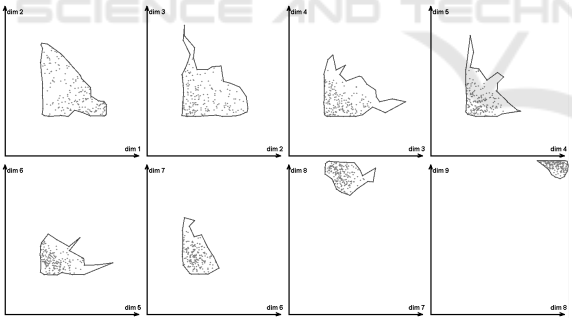

Figure 2: χ-shape model of the 9-dimensional flexibility of a co-generation plant with hourly resolution.

For a simple polygon like $\mathcal{S}_i$ it can be easily checked whether a given point lies inside (or on the boundary) or outside, e. g. with the ray-casting algorithm (Shimrat, 1962). We denote with $\boldsymbol{x}^{(j)} = (p_j, p_{j+1}) \sqsubseteq \mathcal{S}_j$ that dimensions j and $j+1$ of $\boldsymbol{x}$ are contained in (lie inside) $\mathcal{S}_j$ or lie on the boundary.

Now we can define a model for the feasible region: Let $\mathcal{M}(\boldsymbol{x})$ be a general indicator function

$$\mathcal{M}(\boldsymbol{x}) = \begin{cases} \text{true} & \text{if } \boldsymbol{x} \in \mathcal{F} \\ \text{false} & \text{else} \end{cases}. \tag{1}$$

$\mathcal{M}$ can be defined with the help of the χ-shape set $\{\mathcal{S}_j\}_{j \in \{1,\ldots,d-1\}}$:

$$\mathcal{M}_\chi(\boldsymbol{x}) = \begin{cases} \text{true} & \text{if } \boldsymbol{x}^{(j)} \sqsubseteq \mathcal{S}_j \forall\, j \in \{1, \ldots, d-1\} \\ \text{false} & \text{else} \end{cases}. \tag{2}$$

A schedule is classified feasible iff in every intersection plane the respective 2-dimensional intersection of the schedule lies inside or on the polygon defined by the respective χ-shape. Figure 2 gives an example for 9-dimensional schedules of a co-generation plant on an hourly basis. The feasible region is modeled by 8 intersection planes each modeling two dimensions with an overlap of one.

Based on this flexibility model we can now define a decoder. Let $\boldsymbol{z} = (p_1, \ldots, p_d)$ be an infeasible schedule. For simple solution repair one could harness the shortest path to the feasible region defined by the set of χ-shapes. For each intersection plane j one just had to determine the point on the polygon $\mathcal{S}_j$ with the shortes distance to $\boldsymbol{z}^{(j)}$.

For a more sophisticated space mapping decoder we propose the following approach: For a single, 2-dimensional intersection plane the following holds. Let $\boldsymbol{z}^{(j)} = (z_j, z_{j+1})$ be a point in $\mathbb{R}^2$ with the first coordinate fixed and inside the χ-shape $\mathcal{S}_j$. Then $\boldsymbol{z}^{(j)}$ can be moved along the line segment $\overline{(z_j,0)(z_j,1)}$ (remember all values are scaled to $[0,1]$) until $\boldsymbol{z}^{(j)}$ lies inside $\mathcal{S}_j$. Let $\tilde{z}_j$ denote the repaired coordinate of $\boldsymbol{z}_j$. As neighboring intersections overlap by 1, the second coordinate z_{j+1} is identical with the first coordinate of the succeeding intersection. Therefore, the first coordinate there is now fixed (i. e. $z_{i+1} = \tilde{z}_i$) and the second can be moved inside. This process may continue until the last coordinate of the schedule has been pushed inside the feasible region. Only for the first coordinate in the first plane a coordinate inside has to be chosen by line search as an anchor as there is no predecessor. All other coordinates can be determined systematically.

Coordinate mapping is done by moving along $\overline{(z_j,0)(z_j,1)}$. To do this we determine the intersections of $\overline{(z_j,0)(z_j,1)}$ and $\mathcal{S}_j$. In case there is just a single intersection point $\boldsymbol{\iota} \in [0,1]^2$ between the χ-shape and the line trajectory, the second coordinate of the intersection is directly taken: $z_{j+1} = \boldsymbol{\iota}_2$.

In case of multiple intersections $\{\boldsymbol{\iota}_1, \boldsymbol{\iota}_2, \ldots, \boldsymbol{\iota}_m\}$, $m > 2$ we rank them according to distance $\delta[\boldsymbol{\iota}_i, (\tilde{z}_{j-1}, z_{j+1})]$ and direction. Let $\boldsymbol{\iota}_A$ denote the nearest intersection, $\boldsymbol{\iota}_B$ the second nearest in the same direction as $\boldsymbol{\iota}_A$ and $\boldsymbol{\iota}_C$ the nearest in the opposite direction. Now we determine $\tilde{\boldsymbol{z}}_{j+1}$ by translating $P = (\tilde{z}_{j-1}, z_{j+1})$ proportionately in

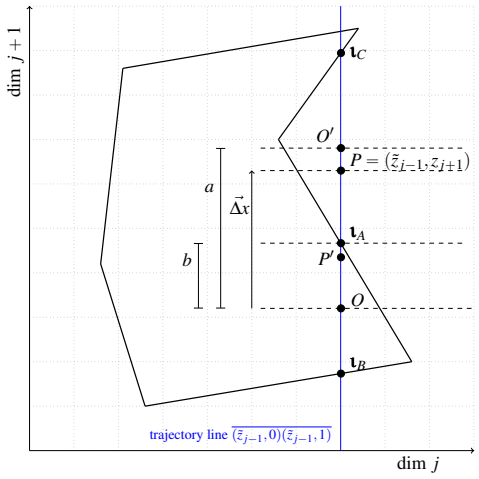

Figure 3: Basic scheme for mapping an infeasible coordinate proportionally into the feasible region.

between $\mathfrak{t}_A$ and $\mathfrak{t}_B$: $P^* = O + \frac{b}{a} \cdot \vec{\Delta x}$. Figure 3 illustrates the idea. The center O between $\mathfrak{t}_A$ and $\mathfrak{t}_B$ and O' between $\mathfrak{t}_A$ and $\mathfrak{t}_C$ is determined (outer dashed lines). The aim is to map the region in between P and midway to $\mathfrak{t}_C$ onto the first half of the nearest inner part. The length ratio between $a = O'_2 - O_2$ and $b = \mathfrak{t}_{A,2} - O_2$ is used to scale Δx accordingly for determining $P^* = (\tilde{z}_{j-1}, \tilde{z}_{(j+1)}) \overset{\text{overlap}}{=} (\tilde{z}_j, \tilde{z}_{j+1})$.

In case of only 2 intersections, $(\tilde{z}_{j-1}, 0)$ or $(\tilde{z}_{j-1}, 1)$ is taken as $\mathfrak{t}_C$ depending on the direction of $\mathfrak{t}_A$. To wrap it up, a decoder function

$$\chi_{dec} : [0,1]^d \to \mathcal{M}_\chi \approx \mathcal{F}_\mathcal{U} \tag{3}$$

is defined by (1) line searching an anchor for the first coordinate and (2) mapping the second and all succeeding coordinate in every intersection plane proportionally into the nearest part of the respective χ-shape along a trajectory with fixed first coordinate.

In this way, the complete exterior of the feasible region $\mathcal{F}$ inside $[0,1]^d$ is mapped onto $\mathcal{F}$. Prior to mapping, we first check with the help of the χ-shape model $\mathcal{M}_\chi$. whether a questionable schedule $\boldsymbol{z}$ is feasible and thus is already inside $\mathcal{F}$. If this is the case, $\boldsymbol{z}$ is used directly and the mapping step can be omitted. On the other hand, at least in case of modeling energy units, studies have shown for the example of cogeneration plants a feasible portion of less than 10^{-23}; so mapping is the more frequent case (Bremer et al., 2010).

3.2 Hybridizing with Evolution Strategies

For evaluation, we need to hybridize the decoder with an optimization algorithm. The covariance matrix adaption evolution strategy (Ostermeier et al., 1994; Hansen, 2006) (CMA-ES) is an evolution strategy well-known for it's good performance on multi modal black box problems (Hansen, 2011).

A good introduction can for example be found in (Hansen, 2011). CMA-ES is initially not designed for integrated constraint handling in constrained optimization. Nevertheless, some approaches for integrating constraint handling have been proposed. In (Kramer et al., 2009) a CMA-ES is introduced that learns constraint function models and rotates mutation distributions accordingly. In (Arnold and Hansen, 2012) an approximation of the directions of the local normal vectors of the constraint boundaries is built by accumulating steps that violate the respective constraints. Then, the variances of these directions are reduced for mutation.

CMA-ES can be hybridized with decoders as follows (Bremer and Lehnhoff, 2017; Bremer and Lehnhoff, 2018). In every iteration g of CMA-ES a multivariate distribution is sampled to generate a new offspring solution population:

$$\boldsymbol{x}_k^{(g+1)} \sim \boldsymbol{m}^{(g)} + \sigma^{(g)} \mathcal{N}(0, \boldsymbol{C}^{(g)}),\ k = 1, \ldots, \lambda. \tag{4}$$

$\boldsymbol{C}^{(g)} \in \mathbb{R}^{n \times n}$ defines the covariance matrix of the search distribution at generation (iteration) g with overall standard deviation $\sigma^{(g)}$ which can also be interpreted in terms of an adaptive step size. The mean of the multivariate distribution is denoted by $\boldsymbol{m}^{(g)}$, $\lambda \geqslant 2$ denotes the population size.

The new mean $\boldsymbol{m}^{(g+1)}$ for generating the sample of the next generation in CMA-ES is calculated as weighted average

$$\boldsymbol{m}^{(g+1)} = \sum_{i=1}^{\mu} w_i \boldsymbol{x}_{i:\lambda}^{(g+1)},\ \sum w_i = 1,\ w_i > 0, \tag{5}$$

of the best (in terms of objective function evaluation) individuals form the current sample $\boldsymbol{x}_i^{(g)}, \ldots, \boldsymbol{x}_\lambda^{(g)}$. In order to introduce the decoder into CMA-ES, ranking is done with the help of the decoder mapping χ_{dec} and objective function f:

$$\boldsymbol{x}_{1:\lambda}^{(g+1)}, \ldots, \boldsymbol{x}_{\lambda:\lambda}^{(g+1)} = f(\chi_{dec}(\boldsymbol{x}_1^{(g)}, \ldots, \chi_{dec}(\boldsymbol{x}_\lambda^{(g)})) \tag{6}$$

to define $\boldsymbol{x}_{i:\lambda}^{(g+1)}$ as the new ith ranked best individual. For scheduling of energy units, $\boldsymbol{x}$ as solution candidate is the concatenation of schedules

$$\boldsymbol{x} = \boldsymbol{x}_1 \boldsymbol{x}_2 \ldots \boldsymbol{x}_m = (p_{11}, \ldots, p_{1d}, p_{21}, \ldots, p_{2d}, \ldots, p_{md}) \tag{7}$$

with $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m$ denoting schedules for the energy units. Finally, the covariance matrix is updated as usual, but also based on the decoder based ranking

Eq. 6:

$$\boldsymbol{C}_{\mu}^{(g+1)} = \sum_{i=1}^{\mu} w_i \left(\boldsymbol{x}_{i:\lambda}^{(g+1)} - m^{(g)} \right) \left(\boldsymbol{x}_{i:\lambda}^{(g+1)} - m^{(g)} \right)^{\top}. \tag{8}$$

CMA-ES has a set of parameters that can be tweaked to some degree for a problem specific adaption. Nevertheless, default values that are applicable for a wide range of problems are usually available. This also constituted its suitability for use cases in automation. For our experiments, we used default settings after (Hansen, 2011).

4 EVALUATION

For evaluation, we used simulations with models of several different energy units. Among them where co-generations plants with attached thermal buffer stores (of different size) for heating a detached house, PV solar panels, ventilation systems, and heat pumps.

Table 1: Comparison of flexibility models using support vector data description and χ-shapes; npv: negative prediction value; ccr: correct classification rate.

indicator	SV	χ-shapes
fallout	0.0300 ± 0.0018	0.0061 ± 0.0003
precision	0.4106 ± 0.0095	0.7256 ± 0.0097
npv	0.9881 ± 0.0005	0.9835 ± 0.0006
recall	0.6415 ± 0.0143	0.4902 ± 0.0163
miss rate	0.3585 ± 0.0143	0.5098 ± 0.0163
specifity	0.9700 ± 0.0018	0.9939 ± 0.0003
ccr	0.9597 ± 0.0013	0.9780 ± 0.0005

For comparison with the χ-shape decoder (XSD), we used the support vector decoder (SVD) and the related flexibility model as proposed in (Bremer et al., 2011; Bremer and Sonnenschein, 2013). In a first test, we compared the flexibility models.

Table 1 shows the results. As both flexibility models are basically classifiers, standard indicators for classifier evaluation can be used. The comparison is done using a co-generation plant. Due to the tiny share of feasible schedules for co-generation plants (Bremer et al., 2010), it is only possible to calculate the confusion matrix for rather small dimensional cases. Thus, the experiment has been conducted using 8-dimensional schedules in order to generate a sufficient number of true positive instances. The confusion matrix was calculated for both flexibility models with the help of the simulation models for comparing classification results respectively. The results show an almost equally good performance for both flexibility models types. The χ-shapes model shows some improvement regarding the specifity which denotes the share of correctly classified infeasible schedules and a better precision but classifies a larger share of actually feasible schedules falsely as infeasible.

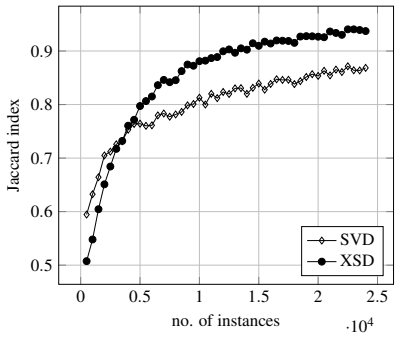

Figure 4: Quality of reproduces flexibilities with different training set sizes based on Jaccard index; a value of 1 denotes perfect accordance.

As a first test for evaluating the decoders we generated training sets from different energy units and respectively trained a support vector and a χ-shape decoder. We then generated 10000 random schedules (power for each time interval $\sim \mathcal{U}(0,1)$) and mapped each random schedule with both decoders. The mapped schedules where tested regarding feasibility using the original simulation model. Table 2 shows the result for 96-dimensional schedules. We used two differently parameterized (regarding shady phases) PV models and a ventilation system. All three are perfectly reconstructed by both decoders with the meaning that all schedules generated by the decoders were feasible. Nevertheless, the χ-shape decoder performs a little better here with regard to generating also schedules from the outer regions of the feasible region. This can better be seen with the specifity in Table 1. Nevertheless, for the co-generation plant, the support vector decoder shows a better performance. This is rooted in the fact that for the co-generation plant with thermal buffer store each feasible amount of energy in a period relays on schedule decisions from preceding time periods (e. g. early charging of the buffer reduces options for later operation). Because the χ-shape decoder has only an overlap of one (two succeeding time periods), it has trouble encoding relations over larger chains of time periods. Thus, the result for the high-dimensional schedules is degrades. For shorter time frames the performance is still sufficient. Reducing the time dependability by using a larger thermal store (CHP 2) reduces this effect.

Next, we evaluated the similarity between the reproduced set of feasible schedules with the actual feasible region as it is determined by the simulation model. A comparison is given in Figure 4. We used the Jaccard index for evaluating similarity between

Table 2: Comparison of the share of correctly reproduced feasible schedules for SVD and XSD for 96-dimensional schedules.

DER	SVD	XSD
PV 1	1.0000 ± 0.0000	1.0000 ± 0.0000
PV 2	0.9998 ± 0.0001	1.0000 ± 0.0000
ventilation	1.0000 ± 0.0000	1.0000 ± 0.0000
CHP 1	0.9434 ± 0.1265	0.3422 ± 0.0173
CHP 2	0.7521 ± 0.2267	0.7453 ± 0.0087

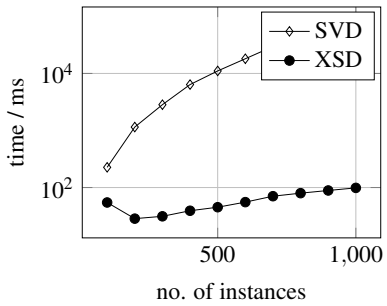

Figure 5: Comparison of training time of SVD and XSD for different training set size.

the region that is covered by two point clouds. Thus, we can compare the similarity between a set of schedules generated by the simulation model and a set of random schedules that has been mapped by a decoder. As Figure 4 shows, the χ-shape approach needs a larger training set than the support vector approach but then outperforms the SVD in terms of similarity of the reproduced region to the original one.

Needing larger training sets is always an issue regarding performance of the approach. Hence, we compared the computation times of SVD and XSD for both training and usage. From analytics it is already known that training can be done in average case for the SVD in $O(n^2)$ with training set size n, if approximated and for the XSD in $O(d \cdot n \cdot \log n)$ with n schedules of dimension d (Duckham et al., 2008; Bremer et al., 2011). The impact in practical applications

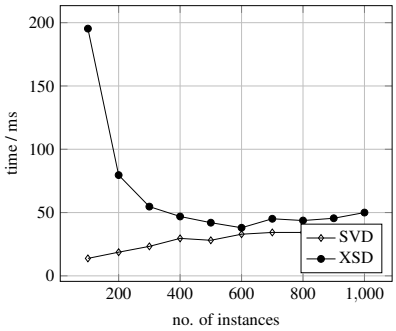

Figure 6: Comparison of mapping performance of SVD and XSD for different training set size (and resulting number of support vectors and polygon lines).

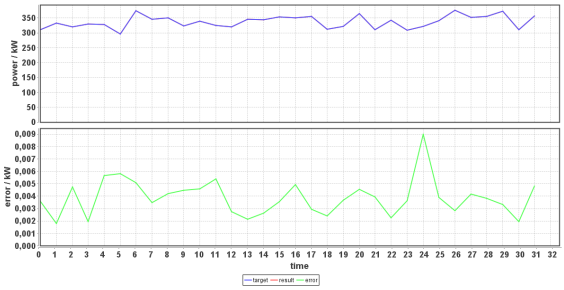

Figure 7: Example result for planning 750 co-generation plants with residual error.

can be seen in Figure 5. For usage, the performance of the SVD is determined by some matrix operations with a matrix size of $n_s << n$; and thus by the number of support vectors n_s. The XSD approach needs a calculation of the intersection $d-1$ times and thus depends on the number of edges of the set of polygons. In practice, XSD is only slightly slower than SVD. Especially for larger training set sizes when the approximated boundary becomes smoother and thus fewer line segments have to be intersected; cf. Figure 6. Nevertheless, there is still room for improvement with concepts e. g. from ray-tracing.

Finally, we tested the XSD in some practical optimization problems. Figure 7 shows an exemplary result for the predictive scheduling problem with 750 co-generation plants. In predictive scheduling for each unit in a set of energy units, a schedule has to be found such that a desired target load profile is resembled as close as possible for a given time frame (Sonnenschein et al., 2014). Because all individual schedules have to be taken from the individual feasible region of the respective energy units, the optimization algorithm has to use decoders for generating candidate solutions. More formal, we want to minimize the distance δ (e. g. Euclidean) between target load profile ζ and the sum of individual schedules for units u: $\delta(\sum_{u \in U}, \zeta) \rightarrow \min$. For solving this problem we use the CMA-ES approach as described before.

The upper chart shows the desired target load profile (artificially chosen such that a residual error of zero is theoretically possible) and as result the aggregated load profile of the co-generation plants; the lower shows the residual error. The mean absolute percentage error for the experiment was 0.00451 ± 0.00479.

5 CONCLUSION

The task of flexibility modelling is an important prerequisite to many planning and control tasks in the smart grid. We presented a novel model and decoder

approach that well suits a niche of energy resources where it outperformed the established SVD.

All in all, the XSD has several advantages regarding performance in scenarios where a frequent training is necessary and it produces a set of schedules for optimization that better resembles the original feasible region especially at the boundary. Thus the flexibility of the energy unit is better harnessed. Nevertheless, it has some problems capturing the operational interdependencies over time with some devices like an electric vehicle charging station, where a set of functionals has to be captured by the decoder that all produce a given value for the integral over the planning horizon. So, the χ-shape decoder may not completely replace the support vector decoder but is a good extension beside it with advantages for many types of energy units.

REFERENCES

Arnold, D. V. and Hansen, N. (2012). A (1+1)-cma-es for constrained optimisation. In *Proceedings of the 14th Annual Conference on Genetic and Evolutionary Computation*, GECCO '12, pages 297–304, New York, NY, USA. ACM.

Braune, C., Dankel, M., and Kruse, R. (2016). Obtaining shape descriptors from a concave hull-based clustering algorithm. In Boström, H., Knobbe, A., Soares, C., and Papapetrou, P., editors, *Advances in Intelligent Data Analysis XV*, pages 61–72, Cham. Springer International Publishing.

Bremer, J. and Lehnhoff, S. (2017). *Hybrid Multi-ensemble Scheduling*, pages 342–358. Springer International Publishing, Cham.

Bremer, J. and Lehnhoff, S. (2018). Phase-space sampling of energy ensembles with CMA-ES. In *EvoApplications*, volume 10784 of *Lecture Notes in Computer Science*, pages 222–230. Springer.

Bremer, J., Rapp, B., and Sonnenschein, M. (2010). Support vector based encoding of distributed energy resources' feasible load spaces. In *IEEE PES Conference on Innovative Smart Grid Technologies Europe*, Chalmers Lindholmen, Gothenburg, Sweden.

Bremer, J., Rapp, B., and Sonnenschein, M. (2011). Encoding distributed search spaces for virtual power plants. In *IEEE Symposium Series on Computational Intelligence 2011 (SSCI 2011)*, Paris, France.

Bremer, J. and Sonnenschein, M. (2013). Constraint-handling for optimization with support vector surrogate models – a novel decoder approach. In Filipe, J. and Fred, A., editors, *Proceedings of the 5th International Conference on Agents and Artificial Intelligence (2)*, pages 91–105, Barcelona, Spain. SciTePress.

Duckham, M., Kulik, L., Worboys, M., and Galton, A. (2008). Efficient generation of simple polygons for characterizing the shape of a set of points in the plane. *Pattern Recognition*, 41(10):3224 – 3236.

Edelsbrunner, H., Kirkpatrick, D., and Seidel, R. (2006). On the shape of a set of points in the plane. *IEEE Trans. Inf. Theor.*, 29(4):551–559.

Fröhlinger, J. (2017). *Abstract Flexibility Description for Virtual Power Plant Scheduling*. phd thesis, Carl von Ossietzly Universität, Oldenburg.

Hansen, N. (2006). The CMA evolution strategy: a comparing review. In Lozano, J., Larranaga, P., Inza, I., and Bengoetxea, E., editors, *Towards a new evolutionary computation. Advances on estimation of distribution algorithms*, pages 75–102. Springer.

Hansen, N. (2011). The CMA Evolution Strategy: A Tutorial. Technical report.

Hörding, J. P. (2017). Umsetzung eines Dekoders für kaskadierende χ-Shape Modelle verteilter Energieanlagen. Bachelorthesis, Department of Energyinformatics, University of Oldenburg, Oldenburg, Germany.

Kim, D. G. (1998). Riemann mapping based constraint handling for evolutionary search. In *SAC*, pages 379–385.

Koziel, S. and Michalewicz, Z. (1999). Evolutionary algorithms, homomorphous mappings, and constrained parameter optimization. *Evol. Comput.*, 7:19–44.

Kramer, O., Barthelmes, A., and Rudolph, G. (2009). Surrogate constraint functions for cma evolution strategies. In *Proceedings of the 32Nd Annual German Conference on Advances in Artificial Intelligence*, KI'09, pages 169–176, Berlin, Heidelberg. Springer-Verlag.

Melkemi, M. and Djebali, M. (2000). Computing the shape of a planar points set. *Pattern Recognition*, 33(9):1423 – 1436.

Moreira, A. and Santos, M. Y. (2007). Concave hull: A k-nearest neighbours approach for the computation of the region occupied by a set of points. *Proceedings of the Second International Conference on Computer Graphics Theory and Applications*.

Neugebauer, J., Bremer, J., Hinrichs, C., Kramer, O., and Sonnenschein, M. (2016). Generalized cascade classification model with customized transformation based ensembles. In *IJCNN*.

Ostermeier, A., Gawelczyk, A., and Hansen, N. (1994). A derandomized approach to self-adaptation of evolution strategies. *Evolutionary Computation*, 2(4):369–380.

Park, J.-S. and Oh, S.-J. (2012). A new concave hull algorithm and concaveness measure for n-dimensional datasets. *Journal of Information science and engineering*, 28(3):587–600.

Rosen, E., Jansson, E., and Brundin, M. (2014). Implementation of a fast and efficient concave hull algorithm. Technical report, Uppsala Univ, Finland.

Shimrat, M. (1962). Algorithm 112: Position of point relative to polygon. *Commun. ACM*, 5(8):434–.

Sonnenschein, M., Lünsdorf, O., Bremer, J., and Tröschel, M. (2014). Decentralized control of units in smart grids for the support of renewable energy supply. *Environmental Impact Assessment Review*, (0):–.

Tax, D. M. J. and Duin, R. P. W. (2004). Support vector data description. *Mach. Learn.*, 54(1):45–66.