Mach Learn (2013) 92:5–39
DOI 10.1007/s10994-013-5340-0

A reinforcement learning approach to autonomous decision-making in smart electricity markets

Markus Peters · Wolfgang Ketter · Maytal Saar-Tsechansky · John Collins

Received: 20 November 2012 / Accepted: 4 March 2013 / Published online: 9 April 2013

Abstract The vision of a Smart Electric Grid relies critically on substantial advances in intelligent decentralized control mechanisms. We propose a novel class of autonomous broker agents for retail electricity trading that can operate in a wide range of Smart Electricity Markets, and that are capable of deriving long-term, profit-maximizing policies. Our brokers use Reinforcement Learning with function approximation, they can accommodate arbitrary economic signals from their environments, and they learn efficiently over the large state spaces resulting from these signals. We show how feature selection and regularization can be leveraged to automatically optimize brokers for particular market conditions, and demonstrate the performance of our design in extensive experiments using real-world energy market data.

Keywords Energy brokers · Feature selection · Reinforcement learning · Smart electricity grid · Trading agents

1 Introduction

Liberalization efforts in electricity markets and the advent of decentralized power generation technologies are challenging the traditional ways of producing, distributing, and consuming

Editors: Tijl De Bie and Peter Flach.

M. Peters (✉) · W. Ketter
Rotterdam School of Management, Erasmus University, Rotterdam, The Netherlands
e-mail: peters@rsm.nl

W. Ketter
e-mail: wketter@rsm.nl

M. Saar-Tsechansky
McCombs School of Business, University of Texas at Austin, Austin, TX, USA
e-mail: maytal@mail.utexas.edu

J. Collins
Dept. of Computer Science and Engineering, University of Minnesota, Minneapolis, MN, USA
e-mail: jcollins@cs.umn.edu

electricity. The Smart Grid "aims to address these challenges by intelligently integrating the actions of all users connected to it ... to efficiently deliver sustainable, economic and secure electricity supplies" (ETPSG 2010). This ambitious vision requires substantial advances in intelligent decentralized control mechanisms that increase economic efficiency, while keeping the physical properties of the network within tight permissible bounds (Werbos 2009).

A fundamental objective of the Smart Grid is to maintain a tight balance of supply and demand in real-time. Presently, the task of balancing the output of large-scale power plants with customer demand is handled via centralized control mechanisms. The increasing penetration of small-scale production from renewable sources like solar and wind, however, introduces inherently intermittent, variable, and geographically dispersed supply, and renders real-time balancing significantly more challenging. In addition, proposals for Demand-side Management (DSM) and for tariffs with time-of-use or dynamic pricing complicate the prediction of consumption patterns. Existing centralized control mechanisms are unable to accommodate this combination of intermittent and variable supply, a grid of staggering scale including vast numbers of small-scale producers, and dynamic changes in demand in response to price variations.

A promising approach to effective balancing in the Smart Grid is the introduction of **electricity brokers** (Ketter et al. 2012b), intermediaries between retail customers and large-scale producers of electricity. Brokers offer a distributed alternative to the centralized system of today's grid, facilitate localized markets that reduce inefficiencies from wide-area transmission, and attain socially desirable market outcomes in response to appropriate economic incentives. Because brokers serve as intermediaries, they must also trade in multiple interrelated (e.g., retail and wholesale) markets simultaneously—a structure that Bichler et al. (2010) refer to as *Smart Markets*. Smart Markets constitute a novel class of complex, fast-paced, data-intensive markets, in which participants ought to employ (semi-)autonomous trading agents in order to attain good trading results.

It is imperative that the design of an electricity broker agent can adapt to a wide variety of market structures and conditions. This is because there is considerable variability in the structure that a future Smart Electricity Market might have, and also because such flexibility is generally beneficial for high performance in dynamic environments. We present several important innovations beyond the class of autonomous electricity brokers for retail electricity trading that we presented in Peters et al. (2012). Our brokers can accommodate arbitrary economic signals from their environments, and they learn efficiently over the large state spaces resulting from these signals. Existing approaches (Reddy and Veloso 2011a, 2011b) are limited in the state space size they can accommodate, and are thus constrained in terms of the economic environments they can be deployed into. These works have also not considered customers' variable daily load profiles (instead, assuming fixed consumption), or the broker's wholesale trading—both core challenges for real-world electricity brokers. Our design alleviates these limitations.

The research we report here extends our previous work (Peters et al. 2012) by exploring alternatives for the data-driven identification of particularly informative signals from the broker's data-rich Smart Electricity Market environment. We explore the role that feature selection and regularization techniques play in the broker's adaptation process. Specifically, we explore the benefits of two different feature selection procedures based on Genetic Algorithms and greedy forward selection, and compare them to L1-regularized online learning techniques over the full state space. We find that the inexpensive regularization approach yields satisfactory results under some market conditions; however, the more extensive feature selection techniques can be highly effective across different market regimes (Ketter et al. 2012a) if adequate precautions are taken against environmental overfitting. Based on

our empirical results, in this paper we also provide guidance on how such overfitting can be alleviated, and discuss approaches which are specialized to the Smart Grid challenge we study here. Our empirical evaluations are based on real-world electricity market data from the Ontario Wholesale Market and a revised model of individual customers' consumption decisions. The customer model we employ captures intra-day variability in demand and has been shown to yield realistic aggregate load curves, rendering our empirical results significantly more meaningful as compared to earlier studies, including our own work (Peters et al. 2012). Our empirical results demonstrate that our broker design is highly effective and that it consistently outperforms prior approaches despite the additional challenges we consider.

More generally, research on autonomous electricity brokers for the Smart Grid constitutes a nascent, emerging field, in which most of the challenges are largely unexplored. Improving our understanding of methods that address these challenges has far-reaching implications to society at large. For example, our broker design contributes to current research on economic mechanism design for the Smart Grid by providing effective strategies against which such mechanisms can be validated, e.g. (de Weerdt et al. 2011). Our extensive evaluation of feature selection techniques raises new and interesting questions about connections between overfitting and market stability in the presence of autonomous trading strategies. We also offer an example of how Machine Learning research can inform important developments in the future Smart Grid in Sect. 5.5. In addition to the development of a novel broker agent design, important objectives of this paper are to contribute to our understanding of key design decisions that enable broker agents to operate effectively in the Smart Grid, and to inform future work of challenges and promising research directions.

The paper is organized as follows. In Sect. 2 we give an overview of our Smart Electricity Market Simulation (SEMS). Section 3 describes foundations in Reinforcement Learning, feature selection, and regularization that our approach builds on. Section 4 introduces SELF, our class of Smart Electricity Market Learners with Function Approximation. A thorough empirical evaluation of our learners in comparison to strategies proposed in the literature follows in Sect. 5. In Sect. 6 we review relevant literature. Finally, we conclude with directions for future research.

2 Smart electricity market simulation

Smart Electricity Markets aim to intelligently integrate the actions of customers, generating companies, and the Distribution Utility, cf. Fig. 1. We developed SEMS, a data-driven **S**mart **E**lectricity **M**arket **S**imulation, based on wholesale prices from a real-world electricity market[1] and a complete, micro-level model of appliance usage in private households. An important property of our simulation, with implications for the broker we design to operate in this environment, is to relax the assumption in previous work that consumers exhibit fixed demand (Reddy and Veloso 2011a, 2011b). Fixed demand simplifies the broker's task, however the resultant brokers may not offer an adequate response to the realities of electricity markets. In particular, a key challenge for real-world brokers is to effectively deal with *patterns* in consumer demand. This is important for effective grid balancing, as some patterns, such as high consumption during midday peak-hours, are significantly more costly for the broker to offset in the wholesale market (cf. Fig. 2).

[1]For this study we used data from Ontario's Independent System Operator, http://www.ieso.ca, which has also been used in a related study (Reddy and Veloso 2011b).

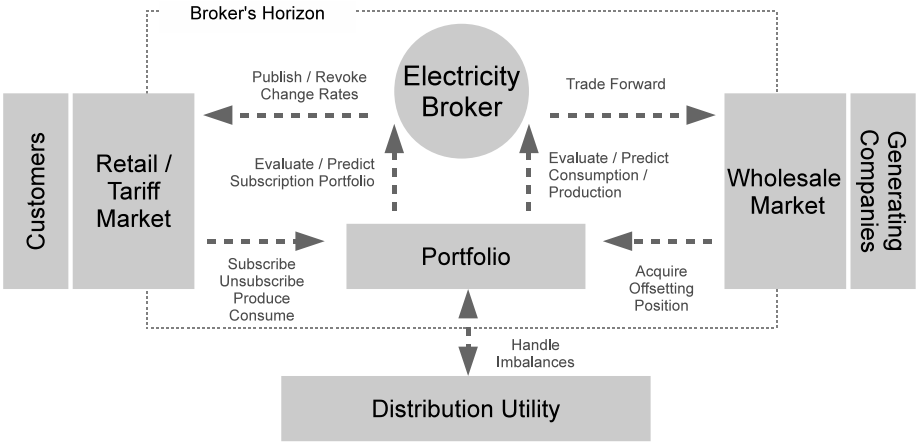


Fig. 1 Smart electricity market structure

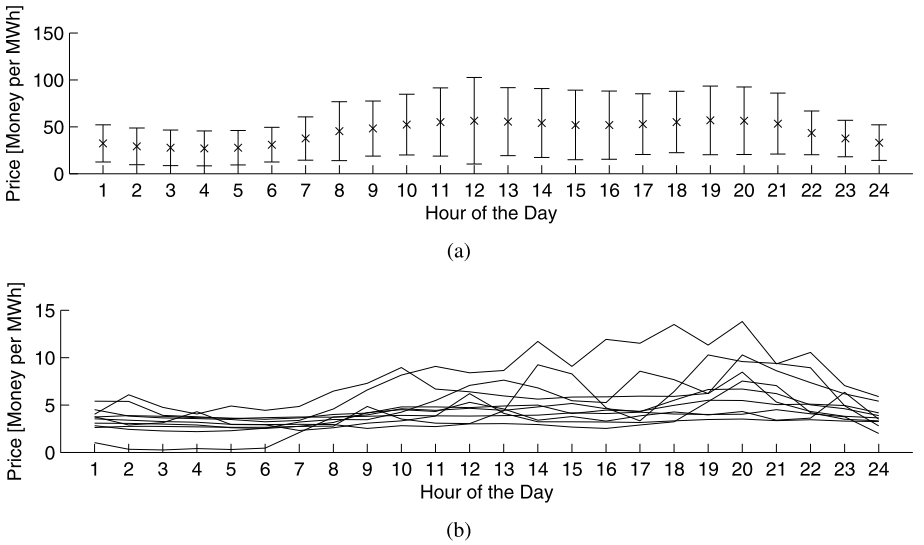


Fig. 2 (**a**) Price distribution (mean ± one standard deviation) for 10 years of price data from the Ontario wholesale market; (**b**) Price curves for 10 randomly selected sample days

Below we outline the key elements of a Smart Grid, along with the models that represent them in our simulation.

– **Customers** $\mathbf{C} = \{C_j\}$ are small-to-medium-size consumers or producers of electricity, such as private households and small firms. Each C_j denotes a group of one or more customers with similar characteristics and a joint, aggregate consumption profile. Customers buy and sell electricity through the *tariff market*, where they subscribe to standardized tariff offerings, including fixed-rate, time-of-use (ToU), and variable-rate tariffs. We describe our customer model in more detail below. Presently, only a small proportion of electric-

ity is produced decentrally, and central production will continue to play a significant role in the near future. As a liberal upper bound consider that, of the 592 TWh of electricity produced in Germany in 2009, merely 75 TWh were produced decentrally under the country's Renewable Energy Act (12.6 %) (European Commission 2011). Accordingly, the customers in our model act exclusively as consumers of electricity.

- **Generating Companies** (GenCos) are large-scale producers of energy, such as operators of fossil-fueled power plants and wind parks. GenCos are wholesalers of electricity production commitments. Because changes in power plant production levels have significant lead times, wholesale electricity is traded *forward* from several hours to several months in advance.
- The **Distribution Utility** (DU) is responsible for operating the electric grid in real-time. In particular, the DU manages imbalances between the total energy consumption and the total outstanding production commitments at any given time. To this end, the DU provides or absorbs energy on short notice and charges the responsible broker imbalance penalties. In SEMS, the DU charges balancing fees that are roughly twice as high as the long-term average cost of electricity in the wholesale market, and thus provides brokers a strong incentive to build easily predictable portfolios of subscribers.
- **Electricity Brokers** $\mathbf{B} = \{B_i\}$ are profit-seeking intermediaries, trading for their own account.[2] Brokers are retailers of electricity in the tariff market, and they offset the net consumption of their tariff subscribers by acquiring production commitments in either the tariff (small-scale producers) or wholesale market (GenCos). The **portfolio** of contractual arrangements that brokers obtain in this way is executed in real-time by the DU. Brokers aim to compile a portfolio of high-volume, high-margin tariff subscriptions with predictable consumption patterns, that can be offset with production commitments at a low cost. In SEMS, brokers publish one fixed-rate tariff at any given time. This design reflects the fact that fixed rates are currently still the dominant tariff model, mainly due to the absence of advanced metering capabilities among electricity customers. We are interested in the performance of methods for autonomous *retail electricity trading*. To this end, we endow both, our own strategies and our benchmark strategies, with a fixed wholesale trading algorithm based on Neural Network load forecasting, and brokers learn to develop a profitable retail trading strategy against this backdrop. Our choice of Neural Networks is mainly due to their good out-of-the-box performance in timeseries forecasting tasks. Alternatives, e.g. based on ARIMA models (Conejo et al. 2005), exist but we do not consider them further as they would impact the performance of all examined strategies in the same way.

The **SEMS Simulation Environment** is responsible for coordinating brokers, customers, and the DU. It manages the tariff market, and provides a wholesale market based on actual market data from Ontario's Independent System Operator. The wholesale market in SEMS determines prices by randomly selecting a window of sufficient size for the simulation run from almost ten years of real-world wholesale market pricing data. Figure 2 shows the long-term daily price distribution as well as daily price curves for 10 randomly selected days from that dataset. Once these prices have been determined, broker orders have no impact on them. Modeling brokers as price-takers is reflective of liberalized retail electricity markets, where an increasing number of small brokers compete against each other. For 2008, for example, the European Commission reported close to 940 non-main electricity retailers in Germany that shared 50 % of the German market (European Commission 2011).

[2]Electricity Brokers are sometimes also referred to as *Load Serving Entities* (LSEs) or *Aggregators* in the electricity market literature.

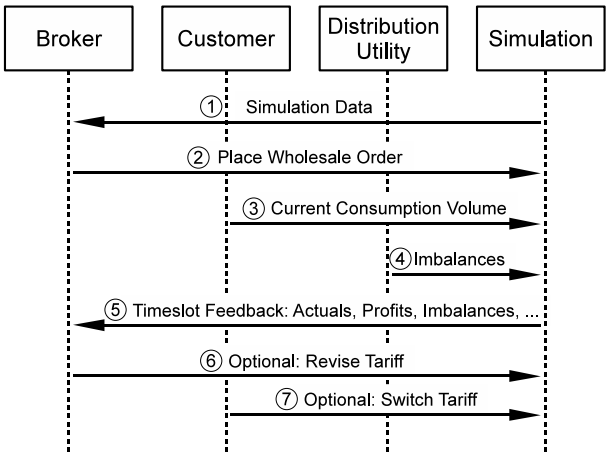


Fig. 3 Sequence diagram for one simulation timeslot

A SEMS simulation runs over N timeslots $1, \ldots, n, \ldots, N$ which are structured as described in Fig. 3.

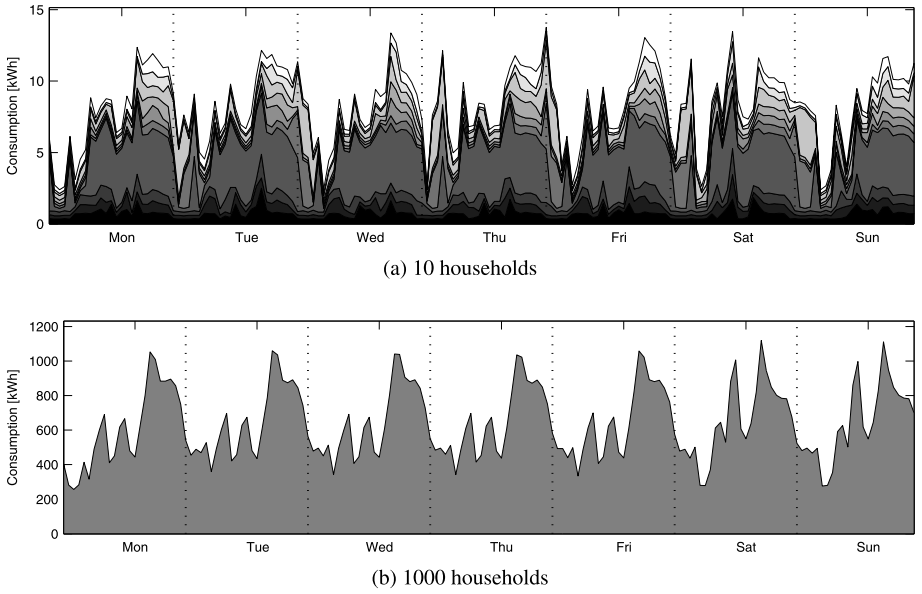
1. Each broker B_i receives information about its current customers $\mathbf{C}_n(B_i)$, the history of wholesale prices $W_1, \ldots, W_{n-1}$, the tariffs offered by all brokers at the end of the last timeslot $\mathbf{T}_{n-1} = \{\tau_{B_1}, \ldots, \tau_{B_{|\mathbf{B}|}}\}$, and its current cash account balance.[3]
2. Each broker indicates the volume of energy $\hat{V}_n^c$ that it wishes to procure in the current timeslot. Note, that the broker has no previous knowledge of its customers' actual consumption nor of the wholesale prices for the current timeslot. There is no acquisition uncertainty; the indicated volume $\hat{V}_n^c$ is always filled by the simulation.
3. Each customer C_j decides the volume of electricity $V_n^c(C_j)$ to consume given its current tariff, and announces this volume to the simulation. The volume consumed, $V_n^c(C_j)$, is derived from the corresponding customer's consumption model, which we describe below.
4. Based on the consumption decisions of its customers, its current tariff, and its acquisition in the wholesale market, each broker's cash account is credited (debited) with a trading profit (loss) $\tau^c(V_n^c) - \hat{V}_n^c \cdot W_n$, where $\tau^c(V_n^c)$ denotes the cost of consuming V_n^c under the current tariff τ^c to the customers (i.e., the revenue of the broker), and $\hat{V}_n^c \cdot W_n$ denotes the cost of procuring $\hat{V}_n^c$ units of energy at the prevailing wholesale price W_n. Any imbalance between the broker's forecast, and the actual amount of energy consumed by its customers is made up for by the Distribution Utility. An imbalance penalty of I per unit of mismatch, or $|V_n^c - \hat{V}_n^c| \cdot I$ in total, is debited from the cash account of the broker for this service.
5. Each broker receives ex-post information on the actual aggregate consumption volume of its customers in the current timeslot V_n^c, its trading profit, its imbalance penalty, and its cash account balance at the end of the timeslot.
6. Each broker is queried if it wishes to change its offered tariff. A fixed amount reflecting administrative costs on the side of the broker is charged for each tariff update.
7. Each customer is queried if it wishes to subscribe to a different tariff.

[3] We summarize the mathematical notation used here and below in Table 1.

Table 1 Summary of mathematical notation

Symbol	Definition
$\mathbf{A} = \{a_i\}$	Constant set of actions available to the SELF reinforcement learner
α_{max}, α'	Initial learning rate and decay of learning rate (1.0 = linear, 0.5 = square root, etc.)
$\mathbf{B} = \{B_i\}$	Set of all competing brokers
$\mathbf{C} = \{C_j\}$, $\mathbf{C}_n(B_i)$	Set of all retail customers, customers of broker B_i at time n
δ	Temporal difference in $Q(s, a)$ between subsequent observations
$\mathbf{e}(\lambda)$, λ	Eligibility trace vector, $\mathbf{e}(\lambda)$ has the same dimensionality as $\mathbf{F}(s, a)$, $\boldsymbol{\theta}$ and measures the eligibility of θ's elements for temporal difference updates based on recent observations; $0 \leq \lambda \leq 1$ determines the degree of recency, with greater values leading to a longer memory of observations
$\varepsilon_{max}, \varepsilon'$	Initial exploration rate and decay of exploration rate (1.0 = linear, 0.5 = square root, etc.)
$\mathbf{F}(s, a), \boldsymbol{\theta}$	Vector of features of the state-action pair (s, a) and their weights in a linear action-value function, respectively
γ	MDP discount parameter
μ	Markup parameter of the benchmark strategies TableRL and Fixed
π, π^*	(Optimal) policy of a given MDP
$\Phi \in \{0, 1\}^n$	Vector indicating the features actually employed out of the set of all available features
$\Psi \in R \subseteq \mathbb{R}^m$	Vector of learning parameters such as α_{max}, λ, etc.
q	Customer switching probability; probability that a customer model considers a new tariff in any given timeslot
$Q(s, a)$	Action-value of state s given action a
$r_n = R_a(s, s')$	Immediate reward earned in timeslot n when the current state is s, the learner takes action a, and is sent to state s' by the environment
$\mathbf{S} = \{s_i\}$	Discrete set of all possible states of the environment
$\mathbf{T}$	Set of all tariffs offered in the tariff market
τ	Customer irrationality $\tau \in [0; \infty)$, where greater values represent less rational or less informed tariff selection behavior
$P = P_a(s, s')$	Transition probability of the environment moving the learner to state s' when in s and choosing action a
V_n^c, $V_n^c(C_j)$	Actual net electricity consumption in time n and net consumption of customer C_j, respectively; from the perspective of one broker
$\hat{V}_n^c$	Estimate of one broker for its customers' electricity consumption at time n
W_n	Actual wholesale market price of electricity at time n

Customers in SEMS are represented by a customer model, each instance of which represents the aggregate behavior of a group of customers. The customer model consists of a **consumption model**, which computes the amount of energy consumed in a given timeslot, and a **tariff evaluator**, which defines how customers select a tariff from a set of offered tariffs. Separating the consumption decision from the tariff selection decision in this way is economically well-motivated. In the short run, the electricity demand of private households is unresponsive to changes in price level. There is some empirical evidence for customers' willingness to *shift* electricity consumption over the day in response to changing electricity prices, e.g., (Herter et al. 2007). However, this phenomenon does not apply to our scenario of a fixed-rate tariff.



(a) 10 households

(b) 1000 households

Fig. 4 Simulated load curves from the SEMS customer model. Panel (**a**) shows the individual consumption profiles of 10 households, panel (**b**) illustrates how the averaging effect leads to a smoother aggregate consumption pattern in a population of 1000 households

The **consumption model** in SEMS employs a micro-level simulation of electric appliance usage in private households based on the work of Gottwalt et al. (2011). This model incorporates statistical data on household types, household sizes, appliance saturation, seasonality on multiple timescales, vacations, etc. to replicate consumption decisions of a real-world customer population, and has been shown to yield realistic aggregate load curves. Our model makes use of all of these features, except for long-term seasonal effects which have no significant impact on the broker's primary challenges of short-term demand forecasting and balancing. Figure 4 shows weekly consumption profiles generated by our consumption model for populations of 10 and 1000 households, respectively. The profiles reflect the characteristic consumption peaks exhibited by private households around noon and during the early evening hours, seasonality effects between weekdays and weekends, as well as the typical consumption averaging behavior in large customer populations (Fig. 4(b)) with significant reductions in the aggregate consumption's noise level.

Our **tariff evaluator** is based on current insights about customers' tariff selection behavior and works as follows:[4] If the tariff that a customer is currently subscribed to is still available, the customer considers selecting a new tariff with a fixed probability q. With probability $1 - q$ it remains in its current tariff without considering any other offers. This

[4]Standard models of electricity tariff selection behavior are currently still an open question in Behavioral Economics and Smart Grid research. Our model captures two central characteristics of customer choice that are thought to lead to sub-optimal tariff choices by human decision-makers: *Inertia* or *switching probability* refers to customers' tendency to remain in their current tariff even if better alternatives surface, e.g. (Nicolaisen et al. 2001), and *customer irrationality* refers to sub-optimality resulting from a range of behavioral factors, e.g. (Wilson and Price 2010).

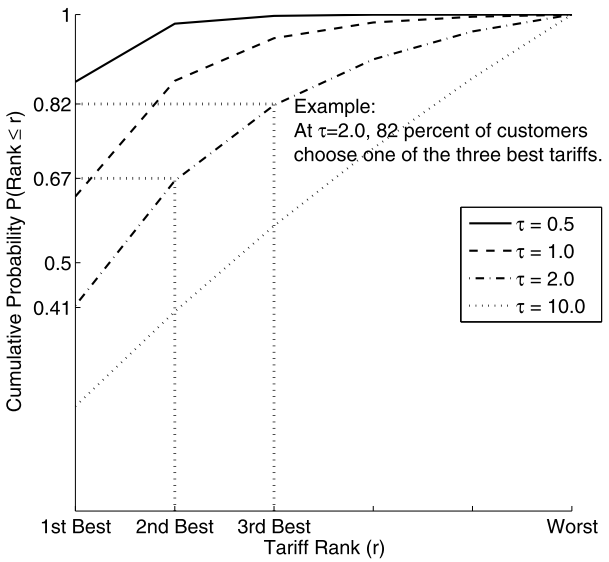


Fig. 5 CDF for the Boltzmann distribution. The parametrized discrete distribution is used to model departures from rationality in tariff selection

behavior captures customers' *inertia* in selecting and switching to new tariffs. If the tariff that the customer is currently subscribed to is not available any longer, the customer selects a new tariff with probability 1. To select a new tariff, the customer ranks all tariffs according to their fixed rates; ties are broken randomly. A perfectly informed and rational customer would simply select the lowest-rate tariff from this ranking, because the lowest-rate tariff minimizes the expected future cost of electricity. In reality, however, customer decisions will tend to deviate from this theoretical optimum for reasons that include (1) customers do not possess perfect information about all tariffs, either because it is unavailable to them, or because they eschew the effort of comparing large numbers of tariffs; and (2) they make decisions based on non-price criteria such as trust and network effects that are absent from our model. We capture these deviations from a simple price rank-order using a Boltzmann distribution.

Assume a customer has to decide among a total of $|\mathbf{T}|$ tariffs. Then the probability of selecting the r-th best tariffs is:

$$\Pr(\text{Rank} = r) = \frac{e^{-r/\tau}}{\sum_{i=1}^{|\mathbf{T}|} e^{-i/\tau}}$$

Here, τ is the so-called *temperature* parameter with $\tau \in (0, \infty)$. The temperature can be interpreted as the customers' *degree of irrationality* relative to the theoretically optimal tariff decision. Consider the Cumulative Distribution Functions (CDF) depicted in Fig. 5 for different values of τ. For $\tau \to 0$, only the best-ranked tariff has considerable mass, i.e., the tariff decision is perfectly rational. For $\tau \to \infty$, the distribution approaches a discrete uniform distribution, i.e., customers select their tariff at random.

3 Reinforcement learning and strategies for high-dimensional state spaces

To operate effectively in the Smart Electricity Market outlined in Sect. 2, an electricity broker agent ought to learn from its environment in multiple ways. In particular, it must

learn about potential customers and their behavior in terms of tariff selection and electricity consumption. A broker should also learn the behavior of its competitors, and derive tariff pricing policies that strike a balance between competitiveness and profitability. Furthermore, because a broker also acts in the wholesale market, it must learn ways to match its tariff market actions with wholesale trading strategies in order to maximize its profit. Note, that the broker's only means of learning is its ability to act in the markets it trades in, and to observe the (long-term) consequences that its actions entail.

3.1 Reinforcement learning

Reinforcement Learning (RL) offers a suitable framework to address the challenges faced by a broker acting in environments with unknown dynamics, and with the objective to collect the highest net present value over all current and future rewards. This can entail foregoing some immediate rewards for higher rewards in the future (Sutton and Barto 1998). More formally, the Reinforcement Learning task we consider here is defined as a finite **Markov Decision Process** (MDP) with observable states and a known, fixed set of actions: $MDP = \langle \mathbf{S}, \mathbf{A}, P, R \rangle$ where $\mathbf{S}$ denotes a finite set of states, $\mathbf{A}$ denotes a finite set of actions, and P and R define the transition probability function and immediate reward function as follows:

$$P_a(s, s') = \Pr(s_{n+1} = s' | s_n = s, a_n = a)$$

that is, $P_a(s, s')$ gives the probability of the environment choosing s' as the following state when s is the current state and the learner chooses action a. And

$$R_a(s, s') = E(r_{n+1} | s_n = s, a_n = a, s_{n+1} = s')$$

that is, $R_a(s, s')$ denotes the expected immediate reward received from the environment when choosing action a in state s and being sent to state s' by the environment thereafter. The solution to such an MDP is the **optimal policy** π^* that maximizes the net present value of all current and future expected immediate rewards, i.e.,

$$\pi^* = \arg\max_\pi \sum_{n=0}^{\infty} \gamma^n R_{a_n = \pi(s_n)}(s_n, s_{n+1})$$

where the learner follows the policy π that gives, for each state s, a corresponding action $a = \pi(s)$ to pursue. $0 \leq \gamma < 1$ denotes the discount parameter where smaller values of γ lead to greater emphasis on current rewards.

Many algorithms have been proposed for finding good policies (Szepesvári 2010). For our agent, we use SARSA: a Temporal Difference (TD) algorithm, that is designed for on-line control problems, such as our retail electricity trading task. The algorithm starts out with some initial model of an **action-value function** $Q(s, a)$, which captures the learner's estimate of the net present value of being in state s, choosing action a next, and following the policy implied by Q thereafter. The learner acts (approximately, except for occasional exploration) greedily with respect to the policy implied by Q, and updates Q with the true feedback it receives from the environment in each timeslot according to

$$Q(s, a) \leftarrow Q(s, a) + \alpha \underbrace{\left[r_{n+1} + \gamma Q(s_{n+1}, a_{n+1}) - Q(s_n, a_n) \right]}_{\text{temporal difference}} \tag{1}$$

where α denotes the learning rate. With probability ε, the learner chooses explorative actions instead of the greedy action implied by $Q(s, a)$ to investigate the value of other state-action

pairs. In our experiments below we let α and ε decay over time to obtain stronger learning and more aggressive exploration towards the beginning of the simulation.[5] In general, SARSA only converges to an exact estimate of Q when each state-action pair is visited an infinite number of times, and when the policy followed by the learner converges to a fixed policy. In our empirical evaluation we show that our learner performs well in spite of not fully meeting these theoretical requirements.

A key challenge of using RL for the problem we address here is the definition of an effective state space. Because it is not well understood which environmental features are useful for capturing changes in the action-value, it is beneficial to employ a wide array of features so as to avoid the exclusion of particularly relevant ones. However, even with a limited number of features, the state space quickly becomes too large to hold in memory. Furthermore, when the state space is large, the extent of exploration required for the learner to arrive at a reliable estimate of the action values $Q(s,a)$ for each $a \in \mathbf{A}$ becomes prohibitive. Previous work has dealt with this challenge by introducing *derived features* that combine multiple environmental features into a single feature for the learner (Reddy and Veloso 2011a, 2011b). However, these derived features are inherently less informative for learning, and there is no principled approach to constructing them. We address these challenges through a two-pronged strategy: (1) We employ **function approximation** to enable the learner to deal with potentially large state spaces; and (2) we explore the performance of **feature selection** and **regularization** techniques that reduce the (effective) state space size.

3.2 Function approximation

Function approximation refers to a parametrized, functional representation of $Q(s,a)$ that allows the broker to explore the effectiveness of strategies over a wider array of potentially relevant states (see, e.g., Rummery and Niranjan 1994 for one of the earliest uses of function approximation in online RL). The most common type of function approximation uses the representation

$$Q(s,a) = \boldsymbol{\theta}\mathbf{F}(s,a)^T$$

where $Q(s,a)$ is linear in $\mathbf{F}(s,a)$, a vector of selected *features* of the current state s given an action a. The reinforcement learner continually updates the weights in $\boldsymbol{\theta}$ to make Q more representative of the experiences gathered from the environment. With linear function approximation this gradient descent update of $\boldsymbol{\theta}$ takes the particularly simple form

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha\delta\mathbf{e}(\lambda) \tag{2}$$

where α again denotes the learning rate, δ denotes the temporal difference (equivalent to the update term in (1)), and $\mathbf{e}$ is the so-called *eligibility trace* which captures the weights eligible for a learning update based on the recently visited state-action pairs. The degree of recency is determined through the parameter $0 \leq \lambda \leq 1$ with $\lambda = 0$ representing updates only based on current observations, whereas greater values of λ introduce increasing degrees of memorization into the update process. Note, that the features in $\mathbf{F}(s,a)$ can themselves be nonlinear functions of features from the environment. Other types of function approximation have also been proposed instead of this linear scheme, e.g. (Busoniu et al. 2010; Pyeatt et al. 2001).

[5] We use α_{max} and ε_{max} to denote maximal rates, and α' and ε' to denote the degree of the parameter decay monomial, where α', $\varepsilon' > 0$ and a value of 1.0 stands for linear decay, 0.5 for square root decay, 2.0 for quadratic decay, and so forth.

3.3 Feature selection and regularization

To improve our broker's learning performance in its information-rich Smart Market environment, we complement function approximation with a principled reduction in (effective) state space size. To this end, we explore different feature selection techniques as well as regularization, and examine their performance and resulting trade-offs in our setting. It is important to note that an MDP's state space size must be fixed and that the reduction referred to above is achieved in two fundamentally different ways: The *feature selection* techniques we study select a subset of relevant features offline, *before* the MDP is constructed. Once the relevant features are selected, the learning process proceeds on a fixed MDP with a state space that is reduced as compared to the space over all original candidate features. Regularization, on the other hand, aims to reduce the dimensions of the state space that are *effectively used*. That is, while the MDP is constructed using a large fixed state space, a regularized learner will likely assign zero weights to many of these features.

While both approaches have been studied extensively in supervised learning, e.g. (Guyon and Elisseeff 2003), they have only recently been considered in RL (Loth et al. 2007; Painter-Wakefield and Parr 2012; Parr et al. 2008; Petrik et al. 2010), and it is important to understand their contributions to the effectiveness of the broker's actions and any implications for future work.

Feature selection here refers to methods that select informative projections $\mathbf{F}'(s,a)$ of the complete feature vector $\mathbf{F}(s,a)$ as basis for learning. Formally, let $\mathbf{F}(s,a)$ be a vector of n candidate features of the current state-action pair, and Ψ a vector of m learning parameters. Then

$$\mathbf{B}_{LinFA} = \left\{ B_{LinFA}(\phi_1, \ldots, \phi_n, \psi_1, \ldots, \psi_m) \| \Phi \in \{0,1\}^n, \Psi \in R \subseteq \mathbb{R}^m \right\}$$

is a class of linear function approximation based RL brokers that use the feature $(\mathbf{F}(s,a))_i$ as part of their state space iff the indicator $\phi_i = 1$. Note, that we combine the feature selection task (i.e., finding good values of Φ) and the related parameter learning task (i.e., finding good values for Ψ), because they can be conveniently tackled simultaneously during the heuristic optimization described below.

We evaluate how well a particular broker $B \in \mathbf{B}_{LinFA}$ competes in a given environment by the **fitness function** $F : B \mapsto [0,1]$ which measures the empirical average profit share that B captures in a given number of sample simulations. This procedure is also known as the **wrapper approach** to feature selection (Blum and Langley 1997). The best broker B^* for the given environment is then $B(\arg\max_{\Phi,\Psi} F(B(\Phi,\Psi)))$.[6]

Optimizing F with respect to B is, in general, intractable due to the lack of structure in F and the size of its domain $\{0,1\}^n \times R$. To alleviate this challenge, we employ one of two types of heuristic optimization:

– **Greedy Feature Selection / Hill Climbing**: Starting with a cohort $C^1 = \{B_1^1, \ldots, B_n^1\}$ of all possible single-feature brokers, we determine their fitness values $F(B_1^1), \ldots, F(B_n^1)$ and select the broker B^{1*} with the maximal F value. We then construct the next cohort C^2

[6]Note, that a profit-maximizing fitness function does not preclude other social desiderata such as fairness, efficiency, or sustainability considerations from our Smart Market. By properly setting the market's *economic mechanisms*, market designers can create incentive structures that lead self-interested, profit-maximizing brokers to jointly aim towards socially desirable outcomes. Because we are primarily interested in the performance of autonomous retail electricity trading strategies themselves, we consider these economic mechanisms to be given and refer the reader to, e.g., (Dash et al. 2003; Parkes 2007) for further details.

by augmenting B^{1*} with each possible second feature and evaluate $F(B_1^2), \ldots, F(B_{n-1}^2)$ for the $n-1$ resulting brokers. We repeat the process until no further improvement in F values is achieved between cohorts, or until a predefined time limit is reached. We select the feature set Φ^* of the overall F-maximizing broker B^*. This process is commonly known as *forward selection* (Guyon and Elisseeff 2003).

To select the broker's parameters we use a hill-climbing procedure where we first draw p parameter vectors $\Psi_1, \ldots, \Psi_p$ and corresponding gradients $\nabla\Psi_1, \ldots, \nabla\Psi_p$ at random. We then construct the first cohort as $C^1 = \{B(\Phi^*, \Psi_1), \ldots, B(\Phi^*, \Psi_p)\}$ and subsequently develop each broker along the predetermined gradient until no further improvement is possible. For example $B(\Phi^*, \Psi_1)$ from C^1 is developed into $B(\Phi^*, \Psi_1 + \nabla\Psi_1)$ in C^2, $B(\Phi^*, \Psi_1 + 2\nabla\Psi_1)$ in C^3, and so forth. We finally select the learning parameters Ψ^* of the overall F-maximizing broker.

– **Genetic Algorithms**: Genetic Algorithms are a well-suited heuristic optimization procedure for our problem given their good performance over large binary domains, such as that of our feature indicators (De Jong 1988). Starting with a randomly initialized cohort $C^1 = \{B_1^1, \ldots, B_q^1\}$ we apply a Genetic Algorithm with Mutation and Crossover operators defined as usual, and a small number of elite individuals that is carried over from one cohort to the next (Liepins et al. 1989). As with the greedy procedure outlined above, the Genetic Algorithm runs until no further improvements in fitness value take place or a predefined time limit is reached. Note that the Genetic Algorithm modifies features and parameter values of the individuals concurrently, whereas our greedy approach selects features and parameters in two separate tasks.

Regularization, in contrast to feature selection, shrinks or penalizes the weights in $\boldsymbol{\theta}$ so as to obtain sparse inner products $\boldsymbol{\theta}\mathbf{F}(s,a)^T$. The resulting approximations are less prone to overfitting the peculiarities in action-values because strong, repeated evidence is required for a particular weight to become and remain non-zero. Regularized function approximations are also quicker to evaluate due to their inherent sparsity. One of the key advantages of regularization over feature selection is its natural integration into online learning processes which obviates the need for a separate offline learning phase.

Despite its seeming appeal, little theoretical groundwork has so far been done on the use of regularization in online RL. One of few exceptions is the work by Painter-Wakefield et al. (Painter-Wakefield and Parr 2012) who extend the regularized batch RL algorithm LARS-TD (Kolter and Ng 2009) into L1TD, an L1-regularized online RL algorithm. L1TD adds the shrinkage operation

$$\boldsymbol{\theta} \leftarrow \operatorname{sgn}(\boldsymbol{\theta}) \odot \max\left\{|\boldsymbol{\theta}| - \nu, 0\right\}$$

(with all operators defined component-wise) to the gradient descent update from Eq. (2). The shrinkage operation effectively moves each component of $\boldsymbol{\theta}$ towards zero by ν on each update, and the combined procedure can be shown to yield equivalent results to the L1-regularized regression formulation in the batch case (Painter-Wakefield and Parr 2012).

4 Learning strategies

In this section, we introduce SELF, our class of **S**mart **E**lectricity Market **L**earners with **F**unction Approximation. A thorough empirical evaluation of our learners in comparison to strategies proposed in the literature follows in Sect. 5.

Table 2 Candidate features for SELF state-action spaces

Feature / Encoding	Plain	RBF	RBF(T)	Bin	Description
Bias	■				Constant 1.0
ActionIndex	■				Index of the selected action
ActionOneInK	■				One-In-K representation of the selected action
BetterConsumptionRates	■	■	■		Number of better (lower) rates in the tariff market
CashGradient	■	■	■		Change in cash account balance over the last 48 hours
CustomerGradient	■			■	Change in number of customers over the last 48 hours
MarketBreadth	■	■	■		Range from lowest to highest rate in the tariff market
MarketShare	■	■	■		Percentage of all customers subscribed to SELF
MarkupLeader	■	■	■		Relative margin, as percentage of smoothed wholesale price, between SELF and the cheapest tariff in the market
NumberCustomers	■			■	Number of subscribed customers
RateChangeIndicator	■				1 if selected action would result in a rate change, 0 otherwise
TargetMargin	■			■	Margin over smoothed wholesale price after performing a given action
WholesalePrice	■	■	■		Smoothed electricity wholesale price
WorseConsumptionRates	■	■	■		Number of worse (higher) rates in the tariff market

4.1 SELF

Our candidate strategy SELF is a class of SARSA reinforcement learners with linear function approximation. The state set of each SELF instance reflects selected aspects of its observable economic environment (e.g., its own tariff rate, competitors' tariff rates, market competitiveness indicators, etc.), and its action set contains possible actions in the tariff market. The learning objective is to find a policy π that approximately maximizes the learner's long-term reward in a Smart Electricity Market environment, while competing against other, both learning and non-learning, strategies.

As outlined in Sect. 3.1, one of the key challenges in our Smart Electricity Market setting is the definition of an effective state space for the learner to learn over. We address this challenging problem by defining a large set of candidate features that captures as much environmental detail as possible, and then applying feature selection and regularization techniques to identify a suitable subset of features that benefit learning. Table 2 shows a grid of features (vertical) and related encodings (horizontal), and shaded cells mark the feature/encoding pairs that are available to the SELF learner for learning.[7]

[7]In the table, *Plain* denotes the unencoded feature, *RBF* and *RBF(T)* denote Radial Basis Function encoding (optionally with thresholding) (Sutton and Barto 1998), and *Bin* denotes a sign binary encoding which, given a real value x, transforms $x \mapsto (\mathbf{I}(\text{sgn}(x) = -1), \mathbf{I}(\text{sgn}(x) = 0), \mathbf{I}(\text{sgn}(x) = +1)) \in \{0, 1\}^3$.

Table 3 Available actions for SELF instances

Action	Margin over Wholesale Price
MarginLeader	Slightly lower than cheapest competitor
MarginAvg	Average of all competitors
MarginTrailer	Slightly higher than most expensive competitor
LowMargin	Constant 10 % margin
HighMargin	Constant 20 % margin
NoOp	Keep the current *tariff rate*. Could lead to changes in margin if wholesale prices change.

This example list of candidate features provides a good coverage of all economic information available from the environment. It is important to note that because a primary goal of our design is to substitute laborious, manual state space construction with principled optimization techniques, our methods can accommodate arbitrary additions to this feature set.

Another important element of an electricity broker design are the actions it can take in the retail market. Generally, a broker can either (a) set a new rate on its tariff, or (b) maintain its existing rate. The canonical model for this action set is a continuous or a discretized set of plausible target rates. However, our evaluations revealed that simultaneously learning the variability in the wholesale price-level *and* the variability among its competitors in this way overburdens the learner. To facilitate learning, we propose a set of economically meaningful actions for the broker to choose from. In particular, SELF brokers can choose among the discrete action set shown in Table 3, which is normalized relative to the prevailing wholesale market price level: A SELF broker can set its tariffs *relative* to other tariffs in the market. In doing so, the broker can choose among attacking its competitors (MarginLeader), positioning itself in the middle of the market (MarginAvg), or avoiding competition altogether by posting the most expensive tariff (MarginTrailer). Alternatively, rather than setting its tariffs relative to the market, the broker can set its tariffs in an *absolute* fashion, choosing between LowMargin and HighMargin, irrespective of the competing tariffs in the market. We chose the margins in Table 3 for their good observed performance in our experiments. The broker may also leave its current tariff unchanged (NoOp).

4.2 Reference strategies

We evaluated SELF against the learning and non-learning strategies proposed in Reddy and Veloso (2011b). To address the need for a limited state space, the reference learning strategy uses derived features, referred to as *PriceRangeStatus* and *PortfolioStatus*. Importantly, the simulation model for which this strategy was evaluated did not include an explicit representation of a wholesale market, represented consumers demand as fixed throughout, and the brokers' only sources of electricity production commitments were small-scale producers. Brokers offer one *producer tariff* in addition to the consumer tariff used by the brokers in our study. These differences make some of the published results for this strategy difficult to interpret in the context of the market settings we consider here.[8]

[8]To incorporate these strategies in our simulation setting we used wholesale prices for producer prices, and suppressed actions pertaining to small-scale producer tariffs. We also excluded the PortfolioStatus feature, which is not meaningful for learning the TableRL strategy in our simulation model.

The relevant benchmark strategies for evaluating our SELF Electricity Broker Agent are

- **Learning** a table-based reinforcement learner operating over the reduced, manually constructed state space outlined above. For clarity, we henceforth refer to the Learning strategy as **TableRL**.
- **Fixed** a strategy which charges a constant markup μ over the smoothed wholesale price
- **Greedy** an adaptive strategy which charges either the highest rate in the market or an average rate, depending on the current PriceRangeStatus PRS_n. PRS_n is defined to be *Rational* if the difference between consumption and production rates in the market is at least μ (i.e., if the market charges a reasonable markup). In this case, the strategy opportunistically chooses the currently highest rate in the market. Otherwise, PRS_n is *Irrational* and the strategy chooses an average rate next.
- **Random** a strategy which chooses the next action at random

We refer the reader to Reddy and Veloso (2011b) for complete details on these strategies.

5 Experimental evaluation

We evaluated our SELF broker against the benchmark strategies from Sect. 4.2 in a series of experiments. Each experiment ran over 10 simulated days (240 timeslots) since longer durations had very little impact on performance differences. The performance of each individual broker was computed as the share of the overall profits they captured. We repeated each experiment 70 times to obtain confidence intervals; all confidence intervals and significance claims reported below are at the 95 % confidence level. The customer population was fixed to five customer groups based on our customer model, each representing the aggregate behavior of a *group* of ten households.[9] Each customer model instance was parametrized with the same switching probability q and degree of irrationality τ as indicated below. Note, that the parameter settings only imply equal *levels* of switching probability and irrationality among customer groups, whereas the actual *decisions* vary among groups. The markup parameter μ of the reference strategies Fixed and TableRL was set to 0.10, at which we found that these strategies performed best.

5.1 Manual broker construction

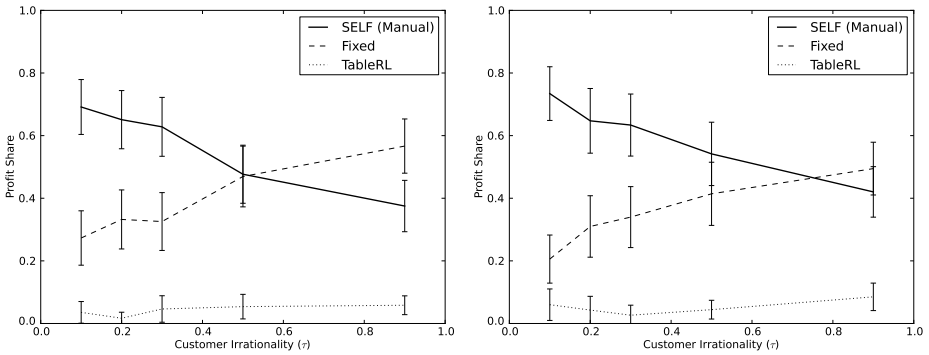
We first constructed several instances of SELF manually by selecting learning parameters and features based on our best knowledge of the problem domain. One rather typical example configuration is summarized in Table 4 where gray cells mark candidate feature/encoding combinations, and black cells mark pairs that were actually used as part of the state space. This particular instance uses features that reflect the broker's own profitability (CashGradient) and customer base (NumberCustomers), the competitiveness of the market (MarketBreadth), as well as the broker's own aggressiveness in the market (MarkupLeader)—arguably some of the fundamental variables in tariff pricing decisions.

The empirical performance of the manually constructed instance in competitions against a Fixed and a TableRL benchmark broker is shown in Fig. 6. The tariff switching probability q was set to a low value ($q = 0.1$, left panel) and a moderate value ($q = 0.5$, right

[9] We found that a larger numbers of customer groups had no significant impact on the results as they did not change the diversity of the population, while fewer customer groups produced an unrealistic "winner takes it all" competition (see also Sect. 5.5).

Table 4 Configuration of a SELF instance constructed manually. Gray shading indicates all candidate features, black shading represents features that were manually selected using domain knowledge

Feature	Plain	RBF	RBF(T)	Bin	Parameter	Value
Bias	■				α_{max}	0.40
ActionIndex	■				α'	1.00
ActionOneInK	■				ε_{max}	0.20
BetterConsumptionRates	■	■	■		ε'	0.70
CashGradient	■	■				
CustomerGradient	■			■	γ	0.90
MarketBreadth	■	■	■			
MarketShare	■					
MarkupLeader	■	■	■			
NumberCustomers	■			■		
RateChangeIndicator	■					
TargetMargin	■			■		
WholesalePrice	■	■	■			
WorseConsumptionRates	■	■	■			



(a) $q = 0.1$ (b) $q = 0.5$

Fig. 6 Performance of the manually constructed SELF instance from Table 4. While it is possible to manually select features that perform well over a limited parameter range, constructing a broker that performs universally well proves challenging

panel), and we recorded the broker's performance while varying the customer irrationality parameter τ. SELF beats the reference strategies Fixed and TableRL by a statistically significant margin in many of these environments. It is interesting to note that TableRL's performance lags not only behind SELF, but also behind the Fixed strategy. This does not contradict the good performance reported in Reddy and Veloso (2011b), as the settings we explore here differ from those for which TableRL was constructed (see Sect. 4.2). However, this result underscores the importance of a well-chosen state space, and the need for a broker design that is able to identify and accommodate any effective state space for a given environment.

Importantly, our results also demonstrate a common outcome for manually constructed SELF brokers: While it is possible to construct broker instances that perform very well under *some* market conditions, achieving robustness over a wide range of market conditions is exceedingly difficult. For high levels of customer irrationality, the performance of the manually-constructed broker approaches that of the Fixed strategy. This result may seem

counter-intuitive, because even for the challenging case of customers choosing their tariffs at random, there is a winning strategy: by raising tariff rates, a broker can increase its profit margin without affecting its customer base. The diminishing performance of SELF for large values of τ stems from an implicit assumption behind its manually constructed state space. Recall that this broker's state space is constructed from the number of subscribed customers, a profitability measure (CashGradient), a competitiveness measure (MarketBreadth), as well as a measure of the broker's own aggressiveness in the market (MarkupLeader). This is a well-chosen feature set for capturing rational, consistent market conditions; however, these features are far less informative or even distracting in settings with significant randomness in customers' choices.

In further experiments we analyzed the performance of manually constructed SELF instances for a wide array of settings by varying the simulation length, the number of customers, values of the markup parameter μ of the reference strategies, and the settings of the learning parameters. We omit details here for the sake of brevity, but we note that this SELF instance performs competitively in all cases except for pathological choices of learning parameters.

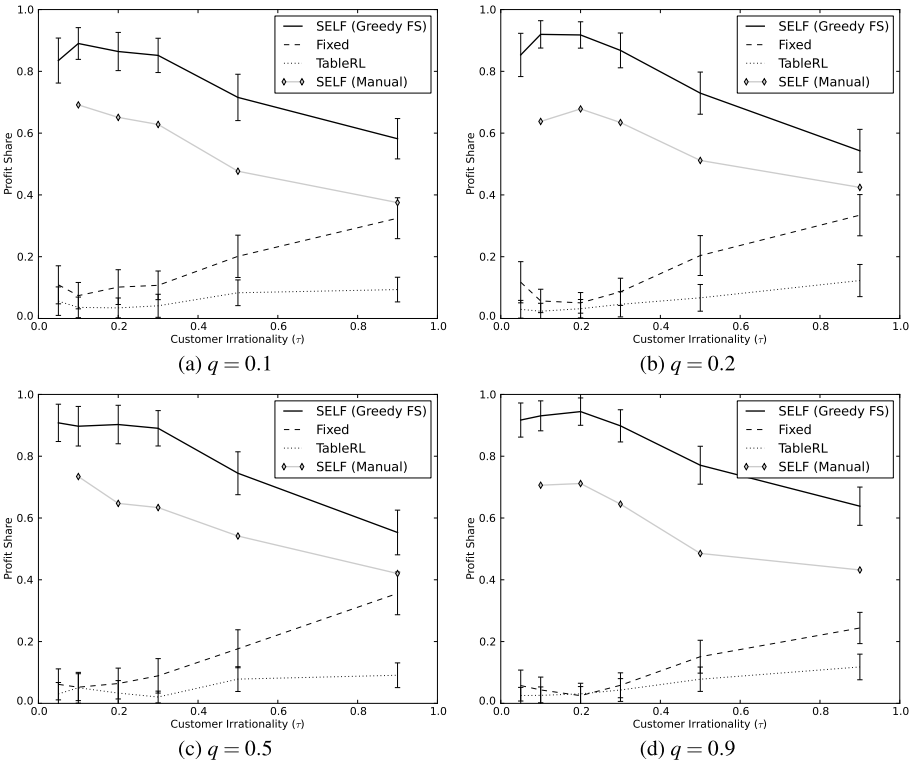
5.2 Feature selection

To further improve the SELF broker's learning and to overcome the challenges that arise from manual feature and parameter selection, we explored the use of the feature selection approaches described in Sect. 3.3 to automatically adapt SELF instances to different market conditions. We fixed a customer population with relatively low switching probability ($q = 0.1$) and irrationality ($\tau = 0.1$) and employed greedy feature selection or a Genetic Algorithm to identify high performing SELF instances. Both methods were allotted a maximum of 24 hours for practical reasons; during the search, the fitness of each candidate instance was evaluated over 25 simulation runs.

Figure 7 shows the performance of the broker configuration obtained with the **greedy feature selection** process. In contrast to the manually constructed instance, the data-driven feature-selection instance consistently outperforms the Fixed strategy over the full range of environmental settings, even while declining noticeably in high-noise environments (higher values of τ). The latter behavior may suggest an adaptation to the low-τ environment used for feature selection, and may also reflect inherently greater difficulty of deriving a profitable policy when customers exhibit random behaviors.

Evidence for the former hypothesis can be found in the configuration summary in Table 5. The extremely high initial learning rate $\alpha_{max} = 0.73$, along with slow decay ($\alpha' = 0.22$), and a high exploration rate hint at strong overfitting. This result is a direct consequence of the relatively stable environment for which the SELF instance's features are optimized. In fact, in most simulation runs we observed under this configuration, SELF learned to price its tariff slightly below the Fixed strategy's rate very early on, and remained in that position for the rest of the simulation. While this policy does well in many settings, it will not likely perform well in environments with high levels of wholesale market volatility or customer irrationality (see the remarks on winning strategies in these environments above). We give a complete example of such a simulation run in Appendix A.

Support for the overfitting hypothesis also comes from the performance shown in Fig. 7. Somewhat surprisingly, SELF's performance in high-τ environments first decreases with increasing values of q (closing gap on right side in panels (a) though (c)) but then improves significantly for $q = 0.9$ (widening gap in panel (d)). We interpret this as a low-bias / high-variance phenomenon with high variance results away from the original feature selection



(a) $q = 0.1$ (b) $q = 0.2$

(c) $q = 0.5$ (d) $q = 0.9$

Fig. 7 Performance of the SELF instance obtained through greedy feature selection, see also Table 5. Its performance compares favorably to the manually constructed instance over a wide range of environmental parameters

environment. This insight gives rise to opportunities for increasing the robustness of the feature selection outcome, which we explore below.

The performance of the SELF instance constructed using **Genetic Algorithm** feature selection is shown in Fig. 8. This broker significantly outperforms its benchmarks, but it performs slightly worse than the SELF instance derived with greedy feature selection. Here, as in the greedy case, we find evidence of overfitting in the empirical results and in the learning parameters, with very high learning and exploration rates (Table 6) and high-variance behavior in environments that are different from those for which the broker was optimized. Moreover, the GA produces solutions to the feature selection problem that are significantly denser than the greedy solutions and it takes the algorithm longer to find them. In all our experiments, the GA feature selection exhausted the maximum allotted 24 hours, while greedy feature selection typically terminated within 5–10 hours, well below the limit. We therefore recommend the use of greedy feature selection over Genetic Algorithms and henceforth employ the greedy procedure exclusively.

5.3 Market stability/guarding against overfitting

Our findings above have significant practical implications for the Smart Grid domain and beyond: Overfitting, either from automatically adapting autonomous trading strategies to

Table 5 SELF instance configuration obtained through greedy feature selection. The resulting state space is sparse, the parameter values hint at a strong overfitting effect, however

Feature	Plain	RBF	RBF(T)	Bin	Parameter	Value
Bias	■				α_{max}	0.73
ActionIndex	■				α'	0.22
ActionOneInK					ε_{max}	0.45
BetterConsumptionRates					ε'	0.04
CashGradient						
CustomerGradient					γ	0.71
MarketBreadth		■				
MarketShare	■					
MarkupLeader						
NumberCustomers						
RateChangeIndicator						
TargetMargin						
WholesalePrice						
WorseConsumptionRates						

Table 6 SELF instance configuration obtained through GA feature selection, overfitting effects are less pronounced than with greedy feature selection but still noticeably present

Feature	Plain	RBF	RBF(T)	Bin	Parameter	Value
Bias	■				α_{max}	0.66
ActionIndex	■				α'	0.37
ActionOneInK	■				ε_{max}	0.16
BetterConsumptionRates					ε'	0.62
CashGradient						
CustomerGradient					γ	0.83
MarketBreadth			■			
MarketShare						
MarkupLeader						
NumberCustomers	■					
RateChangeIndicator						
TargetMargin	■					
WholesalePrice						
WorseConsumptionRates		■				

certain environments or from comparable manual optimization, threatens the stability of Smart Markets. As such, measures against overtfitting are important to both designers of autonomous trading strategies and policy makers. In this section, we first consider two measures that can be built into the optimization process, **bootstrapping** and **noise injection**, before we turn our attention to regularization as an alternative to offline feature selection in Sect. 5.4.

In the experiments above, all SELF instances were initialized with zero weights for their value functions, corresponding to an initially random policy that evolves into a meaningful decision-making strategy over time. Initializing brokers with a random policy has two important implications in our setting. In the case of offline optimization, this initialization encourages the selection of features and parameters that allow for very fast learning early on, at the cost of not generalizing well. In addition, this sets SELF instances at a significant disadvantage relative to non-learning strategies, such as Fixed, which can take reasonable actions from the very beginning.

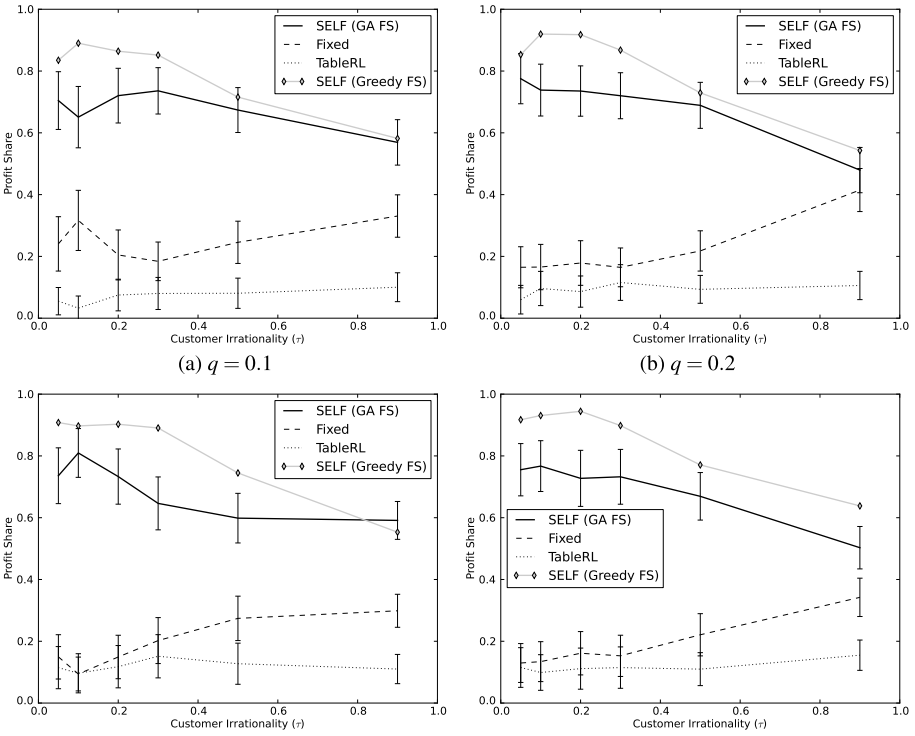


Fig. 8 Performance of a SELF instance obtained through GA feature selection, see also Table 6. The more thorough search for the best state space does not result in better performance as compared to a simple greedy approach

To counteract the overfitting tendency of greedy optimization, we explored the use of **bootstrapping** within the optimization process. Specifically, we first trained the candidate on one run of the simulation, as before. The fitness measure, however, was then evaluated on a second run of the simulation in which we bootstrapped the candidate using the policy learned during the first run. This procedure can be interpreted as a form of cross-validation of the broker's configuration.

An examination of the configurations obtained in this manner reveals that several automatically selected parameters now have more moderate values compared to when bootstrapping is not used. Importantly, with bootstrapping the decay rates $\alpha' = 1.41$ and $\varepsilon' = 1.09$ are both super-linear, yielding a quick decline in learning and exploration, and correspond to a stable learned policy towards the end of each simulation. While the initial learning rate $\alpha_{max} = 0.37$ and the initial exploration rate $\varepsilon_{max} = 0.34$ are relatively high, they are both significantly lower than those produced without bootstrapping.[10]

In comparison with the non-bootstrapping case (Fig. 7), the broker's performance shown in Fig. 9 is promising in two important ways. First, as shown, the broker's performance towards the left of the graphs (low-τ, the environment for which the broker is optimized) is

[10]The full configuration is given in Appendix C, Table 7.

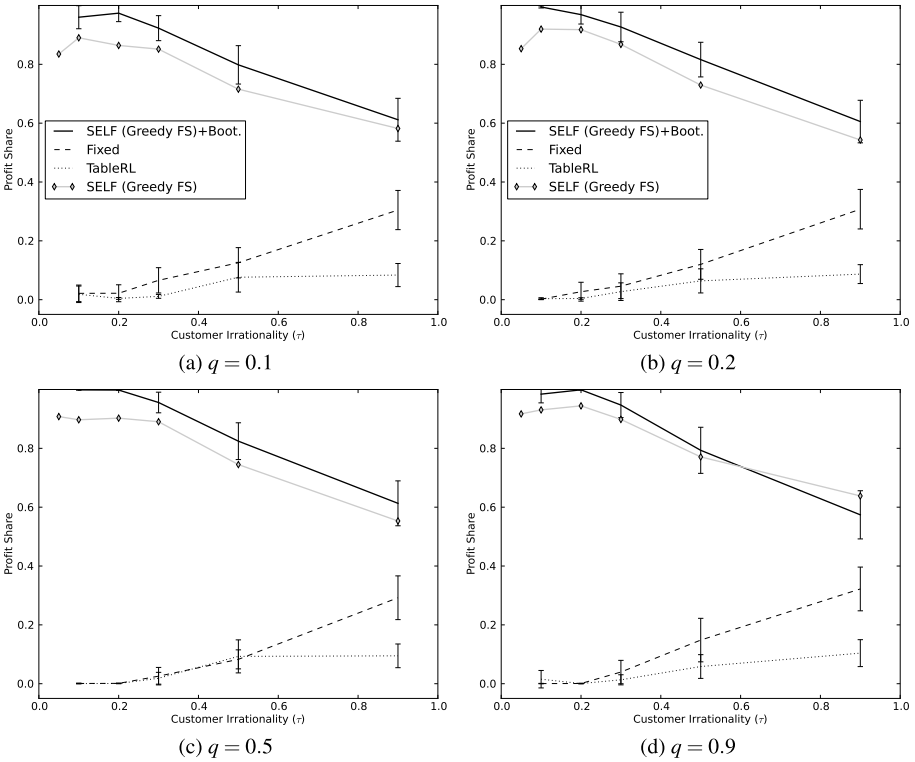


Fig. 9 Performance of a SELF instance obtained through greedy feature selection with bootstrapping. Bootstrapping leads to a performance increase due to the lowered impact of costly initial explorations

better than with the non-bootstrapping instance. This results from the broker taking informed actions from the beginning. In addition, as shown in Fig. 9, while performance towards high-τ values declines, it does so in a consistent, predictable fashion across different values of q. Taken together, our findings suggest that bootstrapping is an effective countermeasure against the overfitting effects associated with plain greedy feature selection.

We now consider **noise injection**, which has long been recognized for its potential benefits in alleviating overfitting and improving generalization in supervised settings, e.g. (Bishop 1995). A challenging choice with this approach is setting the level of noise to be injected into the learning process, such that generalizable patterns remain while spurious patterns are masked by noise. We propose that in competitive Smart Market settings this problem can be circumvented by introducing additional brokers into the market. In particular, in the experiments that we present here, we included in the environment additional brokers which follow a Random strategy. While purely random brokers cannot be expected to be present in actual Smart Electricity Markets, they are interesting to consider because they allow the separation of noise injection effects from the effects of additional competition, i.e., random traders inject noise while not capturing any significant market share.[11]

[11] We also considered the case of additional smart trading strategies. Specifically, we let two instances of SELF compete against each other, as well as against the benchmarks used earlier to explore whether our

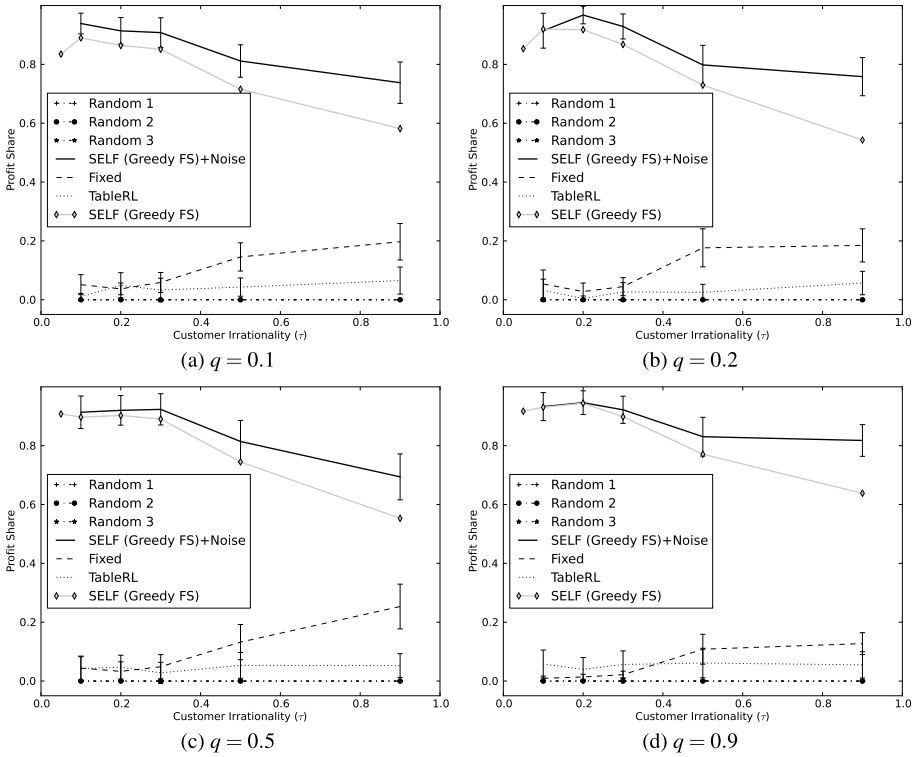


Fig. 10 Performance of a SELF instance obtained through greedy feature selection with noise injection. Note, that all three random strategies are superimposed at the zero-line

The performances with noise injection are presented in Fig. 10, and reveal several insightful differences to the performance of the broker learned with bootstrapping. Perhaps the most significant of those is that the broker's performance for high-τ regimes is not only drastically improved over brokers learned with bootstrapping and plain greedy feature selection, but it is also better over a wider range of environmental settings. At the same time, without the use of bootstrapping, the broker must initially learn a policy at the cost of uninformed exploration; hence, the profit share of the SELF instance is lower in the low-τ regimes for which it is optimized, as compared to when bootstrapping is used. Interestingly, the benefits from bootstrapping and noise injection are complementary. In subsequent experiments, we show that the benefits of noise injection and bootstrapping can indeed be combined to obtain even more robust strategies that achieve both improved performance in their target environment, as well as lower-variance performance for environments that differ from the anticipated ones. We refer the reader to Appendix C for further details on these studies.

Finally, our findings above raise several useful questions for Reinforcement Learning scholars as well as for designers of Smart Market mechanisms. We show that better and more robust learning performance can be achieved via the injection of additional noise from

broker's strategy remains stable under self-play. Our results indicate that both SELF strategies deliver stable performance over a wide range of environments. We refer the reader to Appendix B for details.

randomized brokers, who, importantly, do not capture market share at all. The mere presence of these brokers prompted the greedy optimization procedure to choose a more robust configuration and the resulting broker to act more profitably. While the introduction of purely randomized traders is infeasible in Smart Markets, there may well be market designs that introduce *equivalent noise* to be perceived by brokers (e.g., trading delays, artificial noise on price and volume data, etc.). From a market design perspective, it is interesting to consider whether such distortions can in fact lead to *better* allocation outcomes in the presence of automated trading strategies. From a theoretical perspective, reinforcement learners typically face stochastic reward and transition functions, and they introduce additional randomness through their own action selection (exploration). However, to our knowledge, the potential benefits of noise injection into the learner's *observations* have so far only been explored for supervised learning tasks.

5.4 Regularization

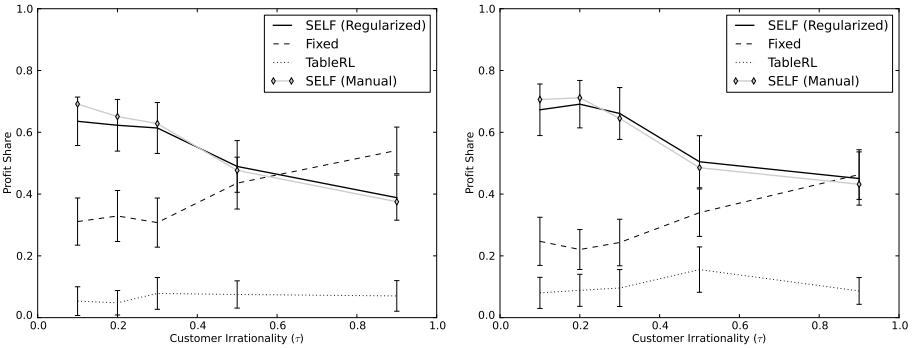
As an alternative to the feature selection techniques discussed above we explored the use of weight regularization for our function approximator. Regularization automatically selects an effective subset from the set of all available features during online learning, and obviates the need for an explicit, offline feature selection phase. It is conceptually simple, and it can be easily integrated into the online learning process.

Figure 11 shows the performance of a SELF instance using regularization as roughly comparable to that of the manually constructed instance shown in Fig. 6. While regularization is not uniformly effective, it is important to note that this level of performance has been achieved with far less domain knowledge and without a model of the environment against which the broker instance could be optimized. Our regularized broker performs well for environments with high customer switching probabilities ($q = 0.9$, right panel) and low levels of customer irrationality (τ small, left end of both panels). Both findings are intuitive: a regularized function approximation requires frequent, strong, and consistent feedback to form an effective policy; in our example such environments arise when customers exhibit high switching probabilities (high q, more frequent retail market feedback) and low levels of irrationality (low τ, more consistent retail market feedback). Thus, while regularization is not consistently effective, it appears to be a beneficial low-cost strategy for learning without explicit feature selection in stable market environments.[12]

5.5 Impact of customer characteristics

In a final experiment, we aim to highlight an interesting case of how Machine Learning research can further inform policy decisions in the Smart Grid domain. This does not aim to offer a full-fledged policy analysis, but a demonstration of how Machine Learning research can contribute to shaping this important, challenging domain. Consider the concept of *microgrids*, self-organizing communities of small-scale electricity consumers and producers, who act as one customer in the retail market, and who only trade net imbalances (i.e., consumption they are unable to meet with own production, or production they cannot absorb with own consumption) in the upstream grid (cf. Fig. 1).

[12] An anonymous reviewer offered the following potential explanation for the comparatively weak performance of regularization: Shrinkage tends to drive weights to zero in areas of the state space that the learner is currently not exploring (based on the sequential nature of exploration in online RL). In other words, the learner "forgets" about regions of the state space not recently visited. We find this explanation quite plausible.

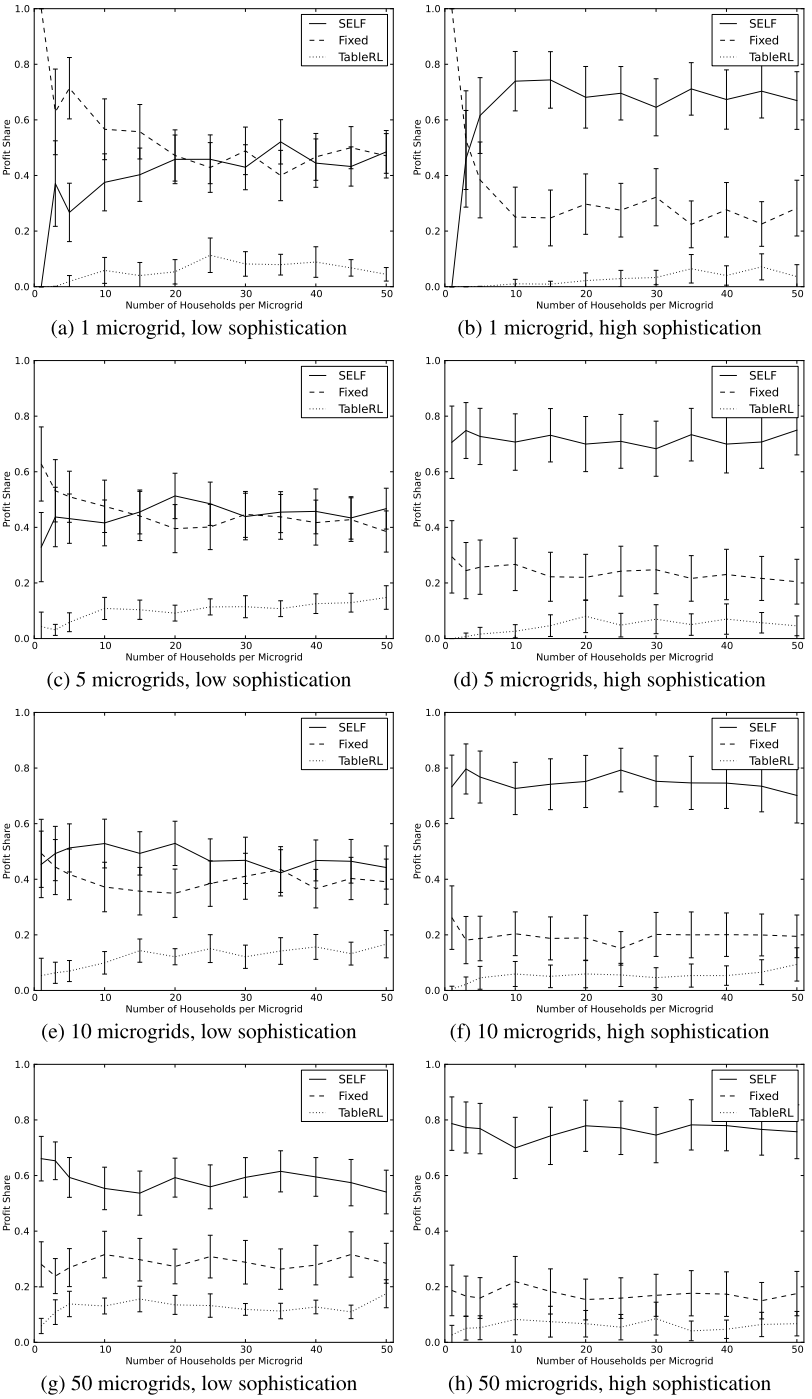


(a) $q = 0.1$ (b) $q = 0.9$

Fig. 11 Performance of a SELF instance using regularization. The instance performs at par with a manually constructed broker instance but does not require domain knowledge

Important open questions pertaining to microgrids are whether their introduction is beneficial to market stability; and, if so, what are the expected implications of the *number*, *size*, and *sophistication* of microgrids in the market. One can also interpret these factors as follows: the *number* of microgrids captures the *lumpiness* of decision making (where fewer microgrids corresponds to fewer independent decisions in the retail market); the *size* of microgrids reflects idiosyncratic risk, because the aggregated behavior of a larger number of households tends to appear smoother, cf. Fig. 4; and, finally, the *sophistication* of microgrids pertains to the degree of rationality in their decision-making, and is inversely related to τ in our notation. Higher levels of sophistication can be a consequence of implementing advanced decision-support within each microgrid, but can also be related to the amount of information flowing from the upstream market into the microgrid. We expect the autonomous retail trading task to become measurably harder as the lumpiness of decision-making increases (up to the limiting case of "winner takes all" competition), as the size of each microgrid decreases (up to the limiting case of single households, as in traditional electricity retail markets), and as sophistication decreases.

We studied the implications of these factors on the performance of SELF, as it represents a state-of-the-art autonomous retail trading agent. The left column in Fig. 12 shows increasing numbers of microgrids with low levels of sophistication (high τ values), whereas the right column presents microgrids with higher sophistication; each panel shows the performances of the broker across different sizes of microgrids. One insight from these results is that the number and sophistication of microgrids are key factors in the performance of our strategy, whereas the implications of each microgrid's size is less significant. Specifically, the performance curves for SELF and the alternative strategies are roughly flat, with no clear trend as microgrid sizes change. The only exceptions are the single-grid cases in panels (a) and (b), where SELF starts to perform consistently from about 10–20 households. This result is partly due to a degenerate "winner takes it all" competition as reflected in the wider confidence bounds, and partly due to the Fixed strategy generally ignoring consumption patterns and tariff selection decisions. A second useful insight is the decisive role that the sophistication of individual microgrid decisions plays in autonomous retail trading strategies' performance. For low sophistication (left column), SELF performs comparably to a simple Fixed strategy in most cases; only for relatively large numbers of microgrids does SELF perform significantly better than the alternative strategies. In



(a) 1 microgrid, low sophistication

(b) 1 microgrid, high sophistication

(c) 5 microgrids, low sophistication

(d) 5 microgrids, high sophistication

(e) 10 microgrids, low sophistication

(f) 10 microgrids, high sophistication

(g) 50 microgrids, low sophistication

(h) 50 microgrids, high sophistication

Fig. 12 Impact of microgrid characteristics on broker performance, *left column*: low level of microgrid sophistication ($\tau = 1.0$) for increasing numbers of microgrids, *right column*: ditto for higher level of microgrid sophistication ($\tau = 0.1$)

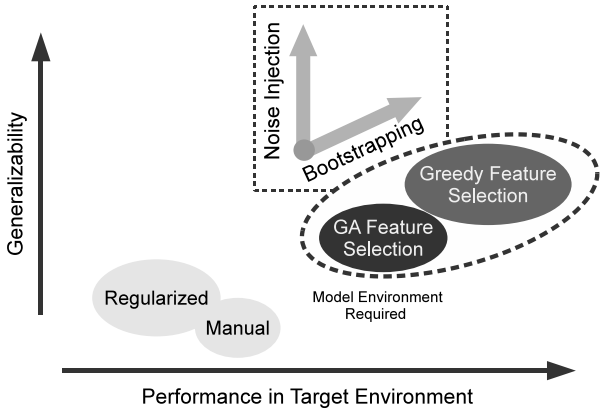


Fig. 13 Summary of results. Darker colored ellipses indicate higher computational demand

contrast, the right column shows SELF outperforming other strategies almost across the board.

It is important to note, that the superior performance of SELF is not merely beneficial to the self-interested broker. Rather, it also reflects higher overall economic welfare: consumers' own preference of SELF over competing strategies is indicative of its desirable, competitive tariff offerings, especially as compared to the customer-agnostic Fixed strategy. Furthermore, higher profit shares are a result of better balanced portfolios with lower imbalance charges, and ultimately contribute to a more efficient and stable grid. From a policy perspective, our results indicate that further research into decision-support systems for microgrids will likely benefit both consumers and the grid overall. Specifically, further inquiry is needed into the benefits that accrue to microgrids from using data-driven, decision-support technology; into whether microgrids would invest in such technology by their own choice; and, alternatively, if public-sector investments into novel decision-support systems could lead to increased economic welfare and a more stable Smart Grid in the future.

5.6 Summary of results

Figure 13 summarizes the characteristics of the methods we proposed by juxtaposing their performance in a Smart Market environment, their capacity to generalize, and their computational requirements.

The simple manual and regularized approaches performed reasonably well for the environments for which they were constructed, but their performance deteriorated quickly as environmental conditions changed. The regularized approach generalized slightly better and, importantly, required significantly less domain knowledge in the design process. Both approaches are computationally efficient and can be applied without a model of the target environment.

When an environmental model is available, feature selection techniques can be leveraged to obtain significantly better performance in the target environment and beyond. Both, greedy and GA feature selection, led to strategies that generalized significantly better than, e.g., the simple regularized approach. Generalization is desirable because of potential shifts in a Smart Market environment, but also to effectively accommodate potential mismatches between the environmental model used for feature selection and the actual target environment. Both feature selection techniques require significant computation before executing

the strategy, but have little impact on the derived strategies' runtime requirements. In our experiments we found greedy feature selection to deliver generally better, sparser results at lower computational costs and we therefore recommend it over GA feature selection for our application.

Finally, we demonstrated how bootstrapping and noise injection can be integrated into the feature selection process to improve performance (bootstrapping) in a given environment, and generalizability (noise injection). Importantly, we show that both techniques can be combined to benefit both objectives.

6 Related work

To date, research on retail electricity trading has received relatively little attention. To our knowledge, Reddy and Veloso (2011b) were the first to suggest RL as an appropriate framework for constructing brokers for retail electricity markets. A key distinguishing feature of our approach is the automated, data-driven construction of the state space. In contrast, the strategies developed in (Reddy and Veloso 2011b) are derived from manually constructed features and are limited in the number of economic signals they can accommodate as well as in their ability to incorporate new signals when the market environment changes. Another key distinction is that the brokers presented in (Reddy and Veloso 2011b) are derived for an environment with fixed rates of electricity consumption and production for all market participants where brokers source electricity exclusively from small-scale producers. Consequently, the broker learns to steer towards an optimal *consumer/producer ratio* among its subscribers by changing tariff rates. These settings yield a broker which is unable to develop appropriate responses to any variability of consumption and production over time or between different customers.

Reinforcement Learning has been used on a wide range of problems in electronic commerce in which agents aim to learn optimal policies through interaction with the environment. For example, Pardoe et al. (2010) develop a data-driven approach for designing electronic auctions based on notions from RL. In the electricity domain, RL has primarily been used to derive wholesale trading strategies, or to build physical control systems. Examples of electricity wholesale applications include (Rahimiyan and Mashhadi 2010), who derive bidding strategies for electricity wholesale auctions, and (Ramavajjala and Elkan 2012) who study Next State Policy Iteration (NSPI) as an extension to Least Squares Policy Iteration (LSPI) (Lagoudakis and Parr 2003) and demonstrate the benefits of their extension on the day-ahead commitment problem of a wind farm. Physical control applications of RL include load and frequency control within the electric grid and autonomous monitoring applications, e.g., (Venayagamoorthy 2009).

Feature selection and regularization have been studied widely in supervised settings, e.g., (Guyon and Elisseeff 2003), but have only recently gained momentum in the Reinforcement Learning community. In our experiments we implemented the L1 regularized version of LARS-TD by Painter-Wakefield and Parr (2012) due to its conceptual simplicity. An alternative approach is Sparse TD with Equi-Gradient Descent (EGD) by Loth et al. (2007) and we are planning on exploring its relative merits in future work. Wrapper approaches to feature selection are commonly used in RL as they are easily integrated as a pre-processing step to the actual RL task. For example, Whiteson et al. (2005) present FS-NEAT, an extension of the well-known NEAT algorithm, to incorporate feature selection capabilities. They demonstrate the benefit of this approach on two standard RL benchmarks. Another feature selection technique specifically targeted at RL applications is the LSTD-RP

method by Ghavamzadeh et al. (2010). They extend the classic Least Squares TD (LSTD) algorithm (Bradtke and Barto 1996) to work with random projections of high-dimensional state spaces, and show how their work translates to online RL settings by replacing LSTD with LSPI.

Whiteson et al. (2011) provide interesting insights into the role of *environment overfitting* in empirical evaluations of RL applications. They argue that *fitting*, i.e., the adaptation of a learner to environmental conditions known to be present in the target environment, is an appropriate strategy. *Overfitting*, i.e., the adaptation of the learner to conditions only present during evaluation, on the other hand, is inappropriate. Our experiments suggest techniques that strike a good balance between fit and performance levels for autonomous trading strategies.

7 Conclusions and future work

The Smart Grid vision relies critically on decentralized control methods that can help balance electric grids in real-time. Developing an understanding of such methods is one of the cornerstones of an efficient, safe, and reliable Smart Grid, with far-reaching benefits for society at large.

We presented SELF, a novel design for autonomous Electricity Broker Agents built on insights from Reinforcement Learning, and from Machine Learning more generally. The key design objectives behind SELF are flexibility and robustness. We framed the broker challenge as optimal control problem and used RL with function approximation to derive robust long-term policies for our SELF brokers. Towards the flexibility objective, we explored the use of feature selection and regularization techniques to automatically adapt brokers to a broad range of market conditions. Using these techniques, SELF brokers can identify and accommodate arbitrary sets of informative signals from their environment, resulting in significantly better performances compared to previous designs. We also evaluated complementary bootstrapping and noise-injection methods to reduce overfitting, and we showed how their use leads to more robust, generalizable feature selection outcomes.

Our work formalizes a class of Smart Electricity Markets by means of our simulation model SEMS, which is a contribution in its own right. SEMS employs real-world wholesale market data and a complete, micro-level model of electric appliance usage in private households, making it a more realistic model of future Smart Electricity Markets than those used in previous studies. We demonstrated the efficacy of our broker design for a range of Smart Electricity Markets which varied substantially in terms of tariff choice behaviors among their customer populations. Our experimental results demonstrate that both, the broker's capacity to accommodate arbitrary state spaces, and its selection of informative features, are important for learning robust policies. Our SELF brokers are significantly more flexible in this regard than previously suggested strategies.

Research on autonomous electricity brokers for the Smart Grid is an emerging field. Hence, beyond the development of a novel broker agent design, we aimed to generate useful insights on key design decisions that enable broker agents to operate effectively in the Smart Grid. For instance, we studied the use of L1 regularization and found that it offers a viable alternative to manual broker construction under stable market conditions. We contrasted regularization with greedy and GA feature selection and found that a simple, greedy feature selection approach can yield significant performance improvements when a model of the environment is available, and when overfitting can be avoided. We presented effective strategies for counteracting overfitting, including an innovative approach for injecting effective noise via *random* broker strategies.

In future work it would be beneficial to further explore SELF's performance in increasingly sophisticated Smart Electricity Markets. Key features to explore include advanced tariff structures, renewable energy sources, and storage devices such as electric vehicles. A question of great practical import is whether the performance of the learner we present here, possibly extended with more advanced RL techniques, will translate to these more complex environments as well. Customers, for example, may eventually adopt electronic agents of their own. In fact, this is commonly thought to be a prerequisite for the success of more complicated tariff models. Our preliminary analysis in Sect. 5.5 gives reason to believe that the use of such agents might actually *benefit* broker agents if they act closer to perfect rationality. How broker agents cope with strategic departures from perfect rationality is, however, unclear.

Our noise injection experiments entail the question whether the extensive work on overfitting that the Machine Learning community has done can be connected to questions about market stability that are under study in the Finance field. Smart Market designs could, for example, be drastically improved if artificially introduced noise (e.g., trading delays) could indeed be proven to be generally beneficial in the presence of autonomous trading strategies. To our knowledge, this connection has not been studied previously.

Another related, and possibly complementary, objective is to derive policies that are *comprehensible* for human decision-makers. Comprehensible policies serve as a further safeguard against overfitting. But they also increase the trust of human decision-makers in autonomous trading strategies, an important precondition to the adoption of Machine Learning techniques in the safety-conscious Smart Grid domain.

We believe that our proposed strategies offer important benchmarks for future work and that this work offers a meaningful contribution to our understanding of key design decisions for broker agents to operate effectively in the Smart Grid.

Acknowledgements We would like to thank three anonymous Machine Learning reviewers and three anonymous ECML-PKDD 2012 reviewers for their insightful comments on this work. The extensive exploration of alternative feature selection and regularization techniques we present here and the subsequent enhanced performance of the agent, was, among other things, inspired by their remarks.

Appendix A: SEMS example run

We referred to the tendency of feature selection to overfit SELF instances to their target environment in Sect. 5.2. In Fig. 14 we give a concrete example of this behavior. The figure depicts the development of tariff rates (Fig. 14b) and cash account balances (Fig. 14c) for various strategies in one example run of SEMS against a stylized step-function wholesale market (Fig. 14a). After the jump in wholesale prices at timeslot 25, the TableRL strategy first fails to adjust its rate upwards; and while SELF first increases its rate based on its target margin over the prevailing wholesale price, it quickly reverts to its initially successful strategy of offering the lowest margin in the market (recall from Sect. 4.1 that SELF learns target *margins*, not target *rates*). In the process, it undersells the unprofitable TableRL strategy and falls behind Fixed in terms of cash balance.

Appendix B: Robustness and self-play

In the experiments above we have considered SELF instances in competition against the benchmark strategies available in the literature to date. As another test of robustness we let

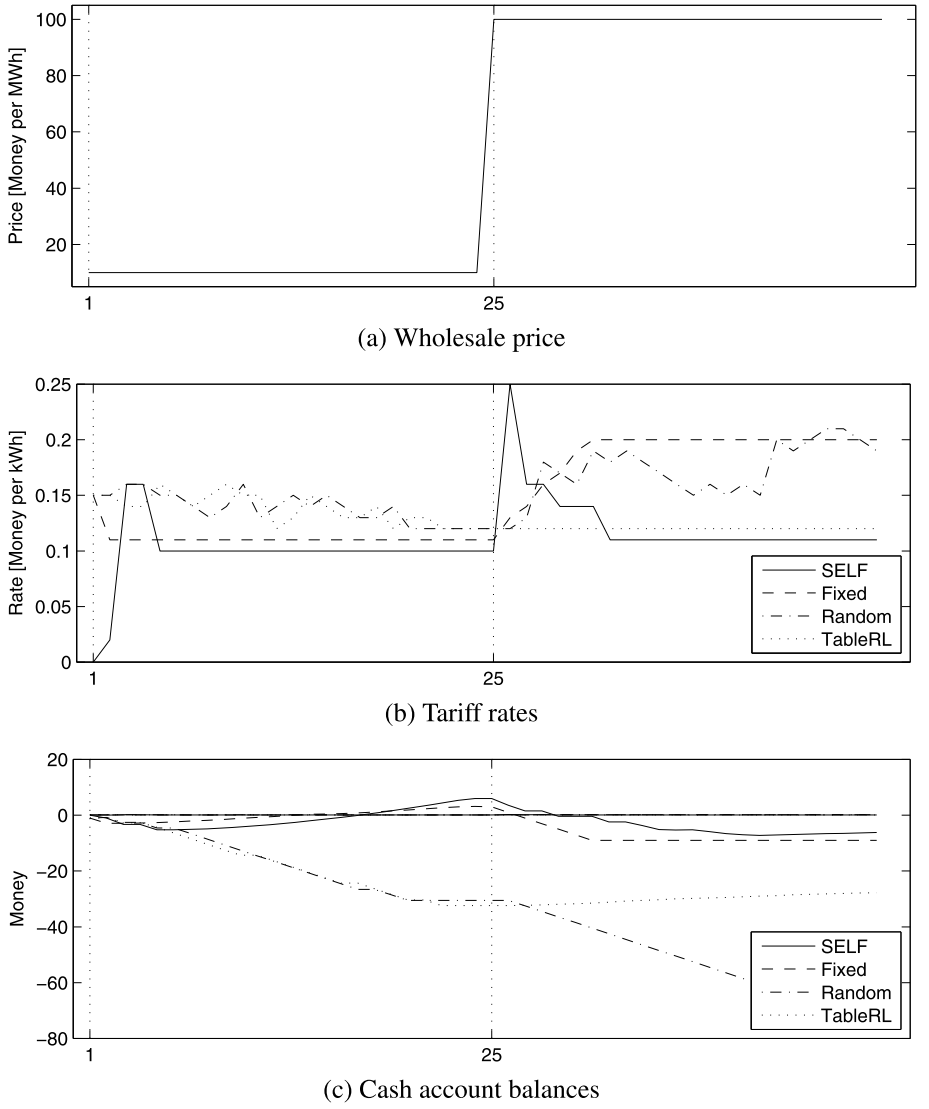
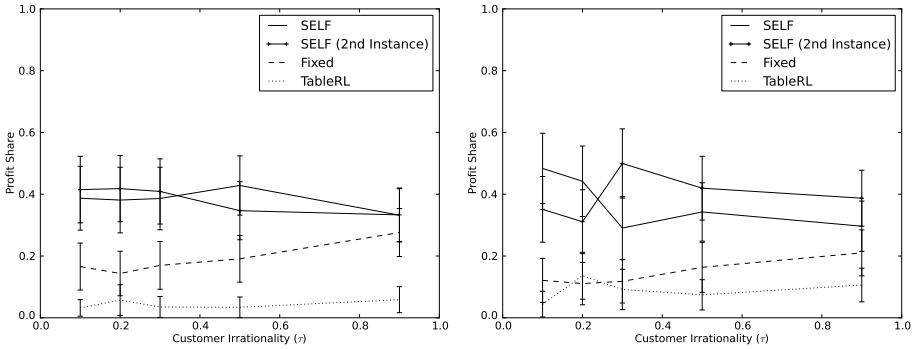


Fig. 14 Simulation using an overfitted strategy against a stylized step-function wholesale market

two instances of SELF compete against each other, as well as against the benchmarks used earlier to explore whether our broker's strategy remains stable under self-play. The results of this evaluation are presented in Fig. 15.

As shown, both SELF brokers perform mostly better than their benchmark strategies under self-play, and overall exhibit consistent performance in comparison to the situation when only a single SELF broker is present. The two brokers' performances are similar as well: When customers exhibit low switching rate (q), the SELF brokers performances are statistically indistinguishable; when q is high their performances are barely distinguishable, but with no clear advantage of one or the other instance. Rather, the SELF brokers' respective



(a) $q = 0.1$ (b) $q = 0.9$

Fig. 15 Performance of two SELF instances in self-play

Table 7 SELF instance obtained through greedy feature selection with bootstrapping

Feature	Plain	RBF	RBF(T)	Bin	Parameter	Value
Bias					α_{max}	0.37
ActionIndex	■				α'	1.41
ActionOneInK					ε_{max}	0.34
BetterConsumptionRates					ε'	1.09
CashGradient						
CustomerGradient					γ	0.76
MarketBreadth						
MarketShare						
MarkupLeader	■					
NumberCustomers				■		
RateChangeIndicator						
TargetMargin						
WholesalePrice						
WorseConsumptionRates						

profit shares are lower than in previous settings, as they simply share the profit available in the market among themselves. Overall, we interpret the consistent, good performance shown by SELF in this more challenging setting as further evidence for the robustness of our broker.

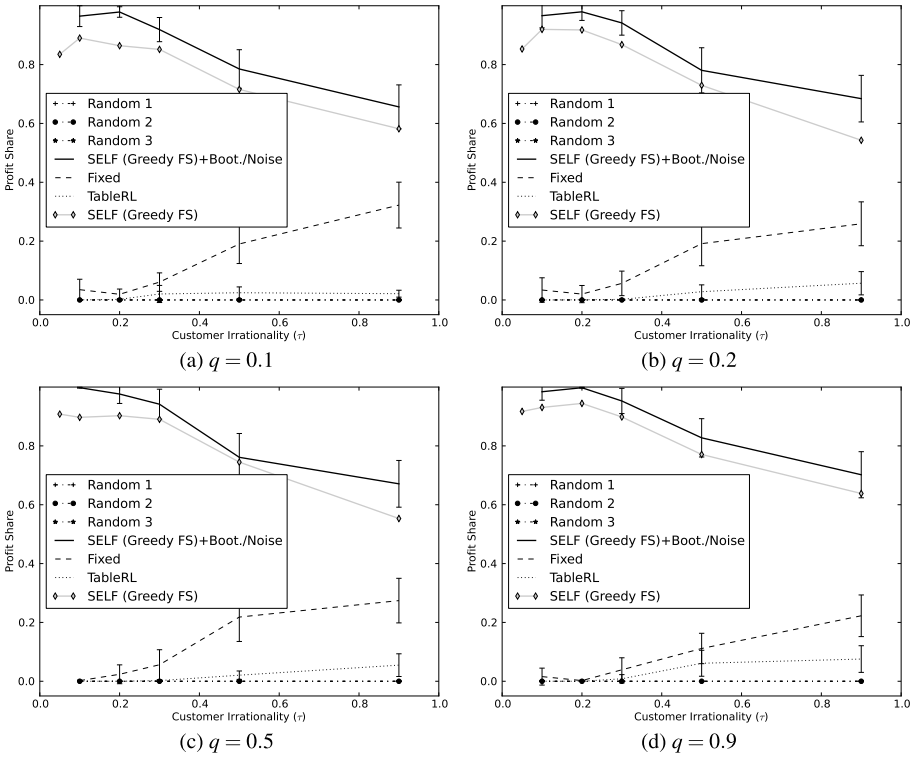
Appendix C: Miscellaneous experimental results

In Sect. 5.3 we explored the role that bootstrapping and noise injection can play in counteracting overfitting tendencies in the feature and parameter selection process. The detailed configurations of the SELF configuration obtained under bootstrapping and noise injection are given in Tables 7 and 8, respectively.

An interesting property of bootstrapping and noise injection is that they work in a complementary fashion. Figure 16 shows the performance of a SELF instance obtained through greedy feature selection using both add-on techniques. The results indicate that the strategy benefited in terms of both, performance in the target environment and generalizability.

Table 8 SELF instance obtained through greedy feature selection with added noise

Feature	Plain	RBF	RBF(T)	Bin	Parameter	Value
Bias	■				α_{max}	0.37
ActionIndex	■				α'	1.41
ActionOneInK	■				ε_{max}	0.34
BetterConsumptionRates	■	■	■		ε'	1.09
CashGradient	■	■	■			
CustomerGradient	■			■	γ	0.76
MarketBreadth	■	■	■			
MarketShare	■	■	■			
MarkupLeader	■	■	■			
NumberCustomers	■			■		
RateChangeIndicator	■					
TargetMargin	■			■		
WholesalePrice	■	■	■			
WorseConsumptionRates	■	■	■			



(a) $q = 0.1$ (b) $q = 0.2$

(c) $q = 0.5$ (d) $q = 0.9$

Fig. 16 Performance of a SELF instance obtained through greedy feature selection with noise injection and bootstrapping. Note, that all three random strategies are superimposed at the zero-line

References

Bichler, M., Gupta, A., & Ketter, W. (2010). Designing smart markets. *Information Systems Research*, *21*(4), 688–699.

Bishop, C. (1995). Training with noise is equivalent to Tikhonov regularization. *Neural Computation*, *7*(1), 108–116.

Blum, A., & Langley, P. (1997). Selection of relevant features and examples in machine learning. *Artificial Intelligence*, *97*(1), 245–271.

Bradtke, S., & Barto, A. (1996). Linear least-squares algorithms for temporal difference learning. *Machine Learning*, *22*(1), 33–57.

Busoniu, L., Babuska, R., De Schutter, B., & Ernst, D. (2010). *Reinforcement learning and dynamic programming using function approximators*. Boca Raton: CRC.

Conejo, A. J., Contreras, J., & Plazas, M. A. (2005). Forecasting electricity prices for a day-ahead pool-based electric energy market. *International Journal of Forecasting*, *21*(3), 435–462.

Dash, R., Jennings, N., & Parkes, D. (2003). Computational-mechanism design: a call to arms. *IEEE Intelligent Systems*, *18*(6), 40–47.

De Jong, K. (1988). Learning with genetic algorithms: an overview. *Machine Learning*, *3*(2), 121–138.

de Weerdt, M., Ketter, W., & Collins, J. (2011). A theoretical analysis of pricing mechanisms and broker's decisions for real-time balancing in sustainable regional electricity markets. In *Conference on information systems and technology*, Charlotte (pp. 1–17).

ETPSG (2010). European technology platform smart grids: strategic deployment document for Europe's electricity networks of the future.

European Commission (2011). EU energy country factsheet.

Ghavamzadeh, M., Lazaric, A., Maillard, O., & Munos, R. (2010). LSTD with random projections. In *Proceedings of the twenty-fourth annual conference on advances in neural information processing systems* (pp. 721–729).

Gottwalt, S., Ketter, W., Block, C., Collins, J., & Weinhardt, C. (2011). Demand side management—a simulation of household behavior under variable prices. *Energy Policy*, *39*, 8163–8174.

Guyon, I., & Elisseeff, A. (2003). An introduction to variable and feature selection. *Journal of Machine Learning Research*, *3*, 1157–1182.

Herter, K., McAuliffe, P., & Rosenfeld, A. (2007). An exploratory analysis of California residential customer response to critical peak pricing of electricity. *Energy*, *32*(1), 25–34.

Ketter, W., Collins, J., Gini, M., Gupta, A., & Schrater, P. (2012a). Real-time tactical and strategic sales management for intelligent agents guided by economic regimes. *Information Systems Research*, *23*, 1263–1283.

Ketter, W., Collins, J., Reddy, P., & de Weerdt, M. (2012b). *The 2012 power trading agent competition* (Tech. Rep. ERS-2012-010-LIS). RSM Erasmus University, Rotterdam, The Netherlands. http://ssrn.com/paper=2144644.

Kolter, J., & Ng, A. (2009). Regularization and feature selection in least-squares temporal difference learning. In *Proceedings of the 26th annual international conference on machine learning* (pp. 521–528). New York: ACM.

Lagoudakis, M., & Parr, R. (2003). Least-squares policy iteration. *Journal of Machine Learning Research*, *4*, 1107–1149.

Liepins, G.,& Hilliard, M. (1989). Genetic algorithms: foundations and applications. *Annals of Operations Research*, *21*(1), 31–57.

Loth, M., Davy, M., & Preux, P. (2007). Sparse temporal difference learning using LASSO. In *IEEE international symposium on approximate dynamic programming and reinforcement learning* (pp. 352–359). New York: IEEE.

Nicolaisen, J., Petrov, V., & Tesfatsion, L. (2001). Market power and efficiency in a computational electricity market with discriminatory double-auction pricing. *IEEE Transactions on Evolutionary Computation*, *5*(5), 504–523.

Painter-Wakefield, C., & Parr, R. (2012). *L1 regularized linear temporal difference learning* (Tech. Rep. TR-2012-01) Duke University, Computer Science.

Pardoe, D., Stone, P., Saar-Tsechansky, M., Keskin, T., & Tomak, K. (2010). Adaptive auction mechanism design and the incorporation of prior knowledge. *INFORMS Journal on Computing*, *22*(3), 353–370.

Parkes, D. C. (2007). Online mechanisms. In *Algorithmic game theory* (pp. 411–439). Cambridge: Cambridge University Press.

Parr, R., Li, L., Taylor, G., Painter-Wakefield, C., & Littman, M. L. (2008). An analysis of linear models, linear value-function approximation, and feature selection for reinforcement learning. In *Proceedings of the 25th international conference on machine learning* (pp. 752–759). New York: ACM.

Peters, M., Ketter, W., Saar-Tsechansky, M., & Collins, J. (2012). Autonomous data-driven decision-making in smart electricity markets. In P. Flach, T. Bie, & N. Cristianini (Eds.), *Lecture notes in computer science: Vol. 7524. Machine learning and knowledge discovery in databases* (pp. 132–147). Berlin: Springer.

Petrik, M., Taylor, G., Parr, R., & Zilberstein, S. (2010). Feature selection using regularization in approximate linear programs for Markov decision processes. In *International conference on machine learning (ICML)*.

Pyeatt, L., Howe, A., et al. (2001). Decision tree function approximation in reinforcement learning. In *Proceedings of the third international symposium on adaptive systems: evolutionary computation and probabilistic graphical models* (Vol. 2, pp. 70–77).

Rahimiyan, M., & Mashhadi, H. (2010). An adaptive Q-learning algorithm developed for agent-based computational modeling of electricity market. *IEEE Transactions on Systems, Man and Cybernetics*, *40*(5), 547–556.

Ramavajjala, V., & Elkan, C. (2012). Policy iteration based on a learned transition model. In *Machine learning and knowledge discovery in databases* (pp. 211–226).

Reddy, P., & Veloso, M. (2011a). Learned behaviors of multiple autonomous agents in smart grid markets. In *Proceedings of the twenty-fifth AAAI conference on artificial intelligence (AAAI-11)*.

Reddy, P., & Veloso, M. (2011b). Strategy learning for autonomous agents in smart grid markets. In *Proceedings of the twenty-second international joint conference on artificial intelligence (IJCAI)* (pp. 1446–1451).

Rummery, G., & Niranjan, M. (1994). *On-line Q-learning using connectionist systems*. Tech. Rep. CUED/F-INFENG/TR 166, University of Cambridge.

Sutton, R., & Barto, A. (1998). *Reinforcement learning: an introduction* (Vol. 116). Cambridge: Cambridge University Press.

Szepesvári, C. (2010). Algorithms for reinforcement learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, *4*(1), 1–103.

Venayagamoorthy, G. (2009). Potentials and promises of computational intelligence for smart grids. In *Power & energy society general meeting* (pp. 1–6). New York: IEEE.

Werbos, P. (2009). Putting more brain-like intelligence into the electric power grid: what we need and how to do it. In *International joint conference on neural networks* (pp. 3356–3359). New York: IEEE.

Whiteson, S., Stone, P., Stanley, K., Miikkulainen, R., & Kohl, N. (2005). Automatic feature selection in neuroevolution. In *Proceedings of the 2005 conference on genetic and evolutionary computation* (pp. 1225–1232). New York: ACM.

Whiteson, S., Tanner, B., Taylor, M. E., & Stone, P. (2011). Protecting against evaluation overfitting in empirical reinforcement learning. In *IEEE symposium on adaptive dynamic programming and reinforcement learning (ADPRL)*.

Wilson, C., & Price, C. (2010). Do consumers switch to the best supplier? *Oxford Economic Papers*, *62*(4), 647–668.